

PLC HANDBOOK



Practical Guide to Programmable Logic Controllers

PLC Handbook

TABLE OF CONTENTS

Chapter 1

What is a PLC	6
A Brief History of the PLC.....	7
Modern PLCs	8

Chapter 2

PLC Basics	9
------------------	---

Chapter 3

How to Choose a PLC Platform.....	11
-----------------------------------	----

Chapter 4

PLC and I/O Hardware Components	12
PLC Form Factors.....	12
I/O Form Factors.....	12
I/O Wiring Options.....	13
I/O Signals.....	14

Chapter 5

Industrial Certifications	16
Ensuring Compliance	16
Why Testing Matters	17
Understanding Quality	17

Chapter 6

Digital Communications.....	19
Connectivity Hierarchies.....	19
Communications Openness and Determinism.....	20
Open-Source and Proprietary	20
Determinism.....	21
Communications Media and Protocols.....	21
Industrial Physical Media	21
Industrial Communications Protocols.....	23

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Chapter 7

PLC Programming Software 26

 Programming IDE 26

 Basic Functionality 26

 Updates 26

 Licensing and Support..... 26

Key Features 27

 Programming Connectivity..... 27

 Hardware Configuration and Auto Discovery 28

 Run Time Transfers and Online Edits..... 28

 Customizable Layouts 28

 Data View and Histograms 29

 Search, Replace, and Cross Reference 29

 Project Compare 30

 Task Manager 30

 Debugging Tools..... 30

 Instruction Sets..... 30

 Powerful Math Functions..... 31

 Integrated Function Blocks 31

 User-Defined Structure (UDS) Data, and User-Defined Instructions (UDIs)..... 31

 PID Loops..... 32

 Security and Cybersecurity 32

 Simulator..... 32

 Help Documentation and Videos..... 33

 Remote Connectivity: MQTT, Web Server, Mobile Apps, StrideLinx Cloud..... 33

 Email and Texting 33

Jump to Chapter 

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

Chapter 8

PLC Programming Languages	34
Memory Addressing	34
Fixed Addressing.....	34
Tagname Addressing.....	34
Find/Replace and Import/Export.....	34
Object-Oriented Programming	35
User-Defined Structure (UDS) Data.....	35
User-Defined Instructions (UDIs)	35
OOP, UDS, and UDI Advantages and Drawbacks.....	36
OT Programming Languages, IEC 61131-3.....	36
IEC Programming Capable PLCs.....	36
CODESYS	37
ProductivityCODESYS.....	37
IT Programming Languages.....	37
ProductivityOpen	38
CLICK PLUS Node-RED	38
CLICK PLUS OPC UA	39

Chapter 9

Ladder logic Programming	40
Structure	40
Logic Instructions, Physical Devices, and Fail-Safe Design	41
Ladder Logic Execution	41
Functions	42
Basic Boolean Logic	42
Basic PLC Instructions	43
Free Downloads and Learning Resources.....	46

Chapter 10

Practical PLC Topics.....	47
Discrete Sensing.....	47
Analog Sensing.....	47
Feedback: Open-Loop Versus Closed-Loop Control	48
PID Control	48
Motion Control	50
Security and Cybersecurity	52

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Chapter 11

AutomationDirect PLC Platform Solutions..... 54

Chapter 12

Applications 56

 Applying Industrial-Grade Automation to Small-Scale Processes 56

 Automating an Animatronic Robot..... 56

 Automated Robotic Welding Cuts Production Time by Over 90% 56

 Automating Escape Room Experiences..... 57

 Installing a Rock-Solid Automation Retrofit..... 57

 Mining Sunbeams - Solar Thermal Leverages the Sun for Producing Heat Energy..... 58

 A Recipe for Automation Success 58

 Automating a Big Rig Light Show 58

 Picking a Cotton Press Automation Upgrade 59

 Mission Control Built on an Industrial Foundation 59

Jump to Chapter 

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

What is a PLC

A programmable logic controller (PLC) can be described as an industrial-grade computer—along with associated hardware and software—employed to accurately monitor real-world sensors and perform careful control of physical equipment like motors, solenoid valves, and much more. PLCs are specifically designed to work non-stop while withstanding demanding environmental circumstances, including extremes of temperature, moisture, vibration, electromagnetic noise, and more.

For these reasons, PLCs are deployed to automate a wide range of applications, including:

- Machinery and test equipment
- Discrete manufacturing and factory production lines
- Batch and continuous processes
- Packaging and material handling operations
- Amusement rides and entertainment venues
- Any place where reliable, always-on monitoring and control is needed in challenging situations

While many people are relatively familiar with the information technology (IT) associated with their computers and mobile devices, the industrial applications listed above are often referred to as operational technology (OT). Early PLCs operated strictly in the realm of OT, but today many OT deployments depend on or are connected to IT systems via various digital communications and networking technologies.

Modern industrial control system (ICS) applications range in size from an individual PLC for a single small machine, or many PLCs interconnected to a variety of field devices and higher-level systems. Expanded connectivity provides incredible visualization capabilities for end users to monitor and control their systems, especially when PLCs are used in collaboration with various forms of local and remote user interfaces such as:

- Operator interface terminals (OITs): Usually smaller hardware-based graphical terminals.
- Human-machine interfaces (HMIs): Can be hardware-based, or also PC-, mobile-, and/or cloud-based interfaces.
- Supervisory control and data acquisition (SCADA): A system of networked HMIs.

In fact, PLCs have become a leading way for connecting OT-sourced data up to cloud resources, for internet of things (IoT), industrial IoT (IIoT), and Industry 4.0 applications. However, the extensive connectivity of PLCs both on-premises at a site or to the cloud via internet connectivity also introduces significant concerns surrounding cybersecurity and the possibility of bad actors causing trouble for critical operations.

This eBook touches on why PLCs were developed, how today's PLCs have stayed the same in some ways, and more importantly how they have changed and gained more expansive capabilities for motion control, data processing, communications, visualization, and more. It will also explore the details that designers need to know when selecting monitoring and control platforms for their applications, and it will explain how AutomationDirect offers a range of PLC platforms to address the varied needs of end users.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

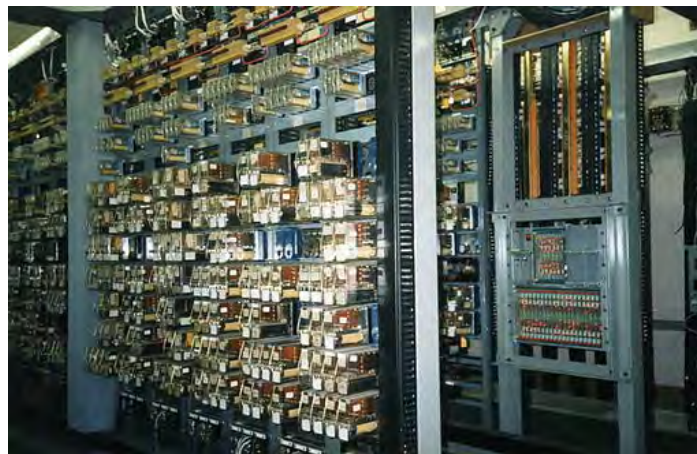
Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

A Brief History of the PLC

First developed in the late 1960's, PLCs were rapidly implemented in subsequent years and became the worldwide standard for industrial automation by the 1980s and onward. But what makes the PLC relevant today, over 50 years after widespread adoption of the technology, and considering the massive improvements in computing power and connectivity over that time?

Before the PLC as we know it existed, automation was largely performed using hardwired relays, electrical control circuits, pushbuttons, lights, solenoids, and more. Even basic automation of this type was hard to design, costly to install, challenging to maintain, and difficult to modify. Achieving any degree of sophisticated control required exponentially increasing numbers of relays, associated devices, and extensive interwiring.



In the late 1960's, recognizing the pressing need to overcome these and other challenges while gaining flexible benefits for their manufacturing lines, automotive giant General Motors (GM) created a list of requirements for a new "standard machine controller." this platform would be:

- A solid-state system that was flexible like a computer
- Priced competitively with the hardwired devices it was displacing
- Easily programmed by personnel already familiar with ladder logic wiring schematics
- Suitable for deployment in an industrial environment
- Modular to support simplified maintenance component changes and expandability

GM then invited companies to submit proposals for a controller meeting these requirements. Dick Morley, who helped found Bedford and Associates, recognized the opportunity and drafted a set of specifications on January 1, 1968, for a controller designed to satisfy GM's requirements. These specifications led to the design of a controller called the "084", so-called because it was their eighty-fourth project. Later that year, another company called Modicon (which stood for Modular Digital Controller) was created to produce the controller, and GM became a large customer. Soon, the "184" controller was developed, and many other makes and models have followed.



Many histories are available online detailing the inception of the PLC and its rapid development. Suffice it to say that today's PLCs are clearly descendants of the original 084, but they have gained significant capabilities, making them a foundational technology for applications today and moving into the future.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Modern PLCs

The earliest PLCs only handled basic on/off signals and logic. Later iterations have gained the ability to handle variable analog signals and perform complex math operations, and they are much more flexible to program. Today's PLCs can do far more, such as executing advanced automation strategies and sequences, data collection, and communication with other intelligent systems. In general terms, PLC evolution can be summarized by decades:

- 1970s: Basic standalone control capabilities
- 1980s: Integrating multiple PLCs to automate systems
- 1990s: HMI/SCADA provides plant floor and enterprise visibility
- 2000s: Advanced functionality; data handling, motion control, vision
- 2010s/2020s: Smart factories thoroughly integrated with the enterprise, IIoT, advanced data analytics and optimization, increased need for cybersecurity
- Future: Preserving local real-time control, while delivering seamless and secure interoperability with all types of higher-level systems, simplified programming

Some people have tried to apply consumer- or commercial-grade IT hardware and software products for use in the industrial OT space, with varying results. Of course, many IT technologies have been adopted into OT products, but the reality is that for most any critical automation application, products like PLCs which are specifically designed for the OT environment are the best choice.

There are other variations of digital controllers overlapping to some degree with PLC capabilities, going by names such as:

- Process automation controller (PAC)
- Edge controller
- Single board controller (SBC) or microcontroller
- Software-based PLC (soft PLC)
- Embedded PLCs (any device which includes PLC-type functionality)

However, this eBook will focus on the details and benefits of today's most typical PLC platforms.

Perhaps the greatest benefit of automating with a PLC is the ability to easily make changes, and then to replicate and expand an application once the desired operation is proven. To select the right PLC platform for a given project, designers need to understand the hardware basics, what kind of hardwired and digital communication signals will be needed, and the level of computing performance required so that an appropriate and cost-effective product can be chosen.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

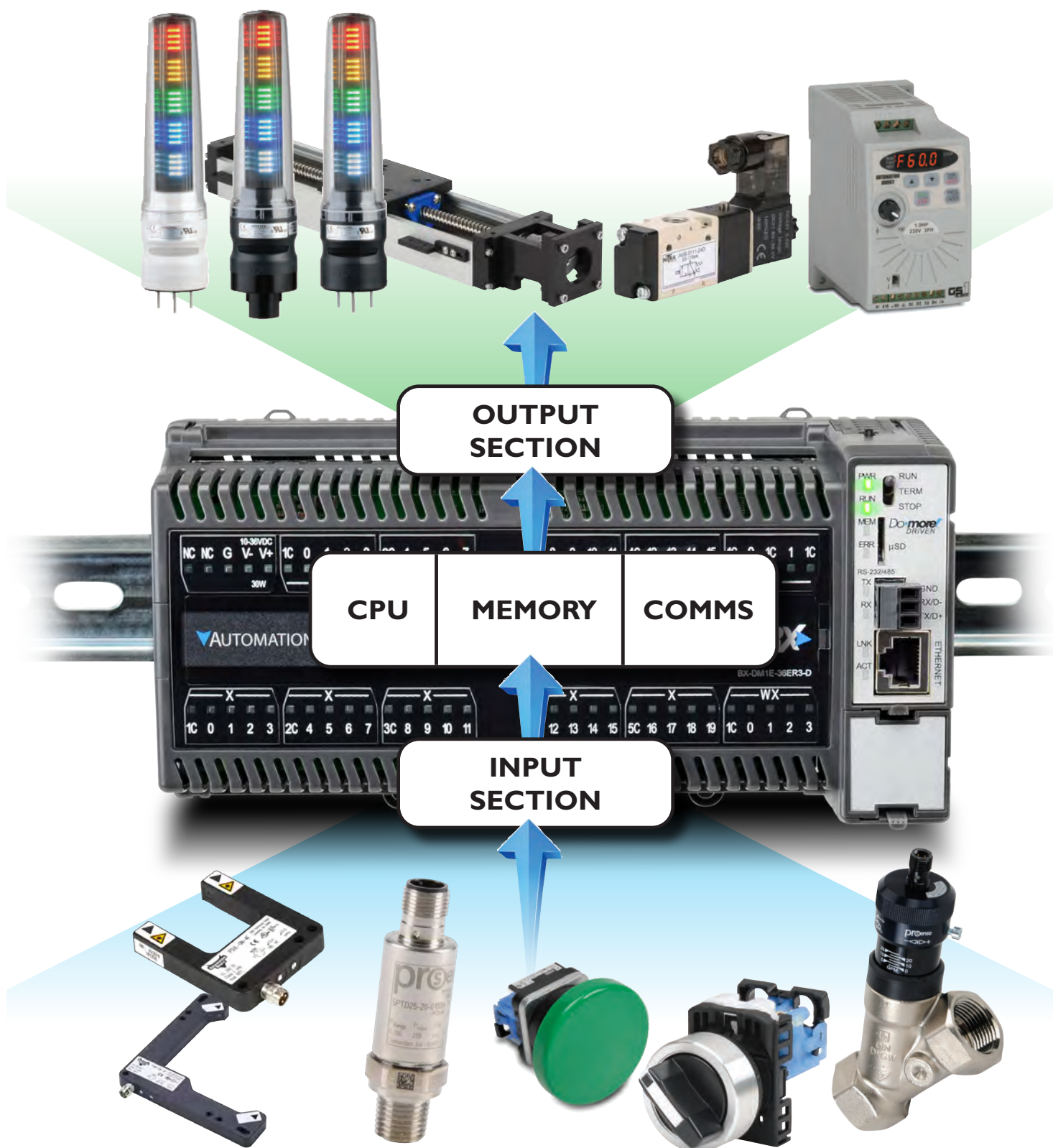
Chapter 12
Applications

PLC Basics

This section identifies the classic characteristics of a PLC as a dedicated automation platform. A bit later we will look at other more recent developments and options.

PLCs are available in varying styles, but they share several typical elements as follows:

- Memory
- Central processing unit (CPU), or processor; also called the controller
- Input/Output (I/O)
- Communications



Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

The memory is where the operating control logic code is stored (program memory), and where the CPU reads and writes data (data memory) as it executes the code. The CPU also handles reading from inputs, writing to outputs, and communications with other devices as needed. Sometimes people compare PLC functionality with that of a living creature: the memory and CPU are like the brain; connectivity internal and external to the PLC is like the nervous system; and I/O signals are like senses.

The CPU has different operating modes. In program mode, the CPU is effectively paused from operating but will accept changes to the downloaded logic from a PC. When the CPU is placed in run mode it will execute the program and operate the process. Some PLCs have other possible modes such as stop or remote, with slightly different functionality.

PLCs are programmed using a PC-based integrated development environment (IDE), which is software used to create, configure, and maintain the control program. The IDE is typically provided by the PLC manufacturer and is specific to that platform. It combines tools for hardware configuration, program development, and system monitoring in a single application.

Within the IDE, control logic is written using standardized PLC programming languages. Different languages are available to support different types of applications, ranging from logic-based control to structured, text-based programming.

Once the hardware configuration is complete, many modern PLCs allow the program to be monitored and modified while the controller remains in run mode. In simpler systems, configuration and programming may still require the CPU to be placed in program or stop mode before changes are applied.

While in run mode, there are four basic steps that continually take place in a repeating loop:

- Input scan – Detects the discrete on/off state or analog signal at each input point from devices that are connected to the PLC (such as switches, sensors, transmitters, and others)
- Program scan – Executes the user created program logic, based on I/O and other memory locations
- Output scan – Energizes or de-energizes each discrete output point, or sends an analog signal, to devices that are connected to the PLC (to operate lights, solenoids, motor controls, and other devices)
- Housekeeping – Includes communicating with programming devices and performing internal diagnostics

The time it takes for one cycle through the program is called scan time, and each scan happens very quickly (in the range of a few milliseconds, depending on the program). Unlike other programming languages and platforms which can freeze or lock up, a PLC is architected to always perform scans unless there is a drastic failure, which will trigger a specific watchdog timer to notify the user that such a fault has occurred.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

How to Choose a PLC Platform

Before diving too far into the details of PLC and I/O functionality, hardware, software, connectivity, and more, it is a good idea to first understand the considerations involved at a high level when choosing a PLC platform.

As with any design or engineering process, each application will have certain requirements, and other nice-to-have aspects, which must be weighed together with the upfront and sustaining costs. Many years ago, processing power and memory were key considerations, along with I/O count. This is still somewhat true, but today the focus has shifted more toward functionality and connectivity.

AutomationDirect offers a helpful online PLC selection tool at <https://www.automationdirect.com/systembuilder>

Here are the top issues to be addressed when selecting a PLC platform:

- Controller type
 - Traditional
 - Integrated motion control required, and if so how many axes?
 - Open source
- Basic functionality
 - Processing power
 - Memory (both volatile and non-volatile/persistent/retentive)
 - How many PID loops?
- I/O counts and types, specialty modules, local/remote/network I/O needs
- Communication ports and protocols, both wired and wireless
- Form factor (brick, stackable, modular rack-based)
- Ensure a configuration is valid and within power budget limitations
- Programming languages
 - Relay ladder logic (RLL), also known as ladder diagram (LD)
 - Function block diagram (FBD)
 - Structured text (ST)
 - Sequential function chart (SFC)
 - Other IT-type languages
- Other features
 - On-board data logging
 - Embedded web server
 - REST API
 - Simulator
 - Miscellaneous functions
- Certifications (environmental and quality control)

These are a lot of details! Fortunately, today's PLCs are very feature rich and usually there are multiple product families that can fulfill any given application requirements. The following chapters will explore these topics, helping users to determine the best options for any application.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

PLC and I/O Hardware Components

The earliest PLCs and I/O components were quite large and heavy due to manufacturing techniques, electronics technologies, and engineering goals of the time. They required significant design and installation effort to locate them in a protected environment and provide wiring accessibility.

Today's PLCs and I/O are differentiated by their much more compact physical size, and by a wide range of packaging form factor options so designers can select a platform that fits their application, while providing the right price/performance ratio.

Of course, regardless of how small the electronics become, the overall form factor size can only be reduced by a certain amount while maintaining the ability to make wiring and cable connections.

PLC Form Factors

Modern PLCs still need to be installed in a protected location, usually a control panel. Today's PLC platforms are often described as micro or small and are available in different form factor arrangements suited for a variety of installation conditions:

- **Brick:** An all-in-one configuration with CPU, power supply, and I/O. Usually quite small and available in a few different models with varying I/O counts and types, and often the most compact option for small applications.
- **Stackable:** Much like the brick configuration and may even include some I/O onboard the main CPU, but with the option to stack on some additional modules for I/O or communications. May require a power supply module. Still a small size, but able to operate applications with moderate I/O counts.
- **Modular:** Modular PLCs are configured with a base or rack which accepts power supply, CPU, I/O, communications/networking, and other modules. Slightly larger than a brick or stackable style, but usually able to handle higher I/O counts and more complex applications.
- **Specialty:** PLC functionality is so ubiquitous that it has been incorporated into other related devices. For example, an AC variable frequency drive (VFD) or a dedicated motion controller may include on-board PLC functionality. For certain applications, this provides designers with an effective all-in-one option for expanded automation functionality.

Keep in mind that PLC form factors may be a hybrid of the general types listed above. As the flexibility, expansion, and capability of a particular product series increases, so does the price.

I/O Form Factors

Local I/O, which means I/O points and modules directly associated at the CPU location, is available with most PLC types. This arrangement is appropriate for smaller, localized applications. Local I/O may be:

- Available right onboard a brick PLC
- Clicked onto a stackable PLC
- Inserted into slots of a modular PLC.

Larger applications will require many more I/O modules, which may need to be co-located in the same control panel or could be distributed to other locations.

Local Expansion I/O provides a way to install several I/O module groupings or racks near the main CPU, using a form of serial or USB connectivity.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Remote Expansion I/O is similar to Local Expansion I/O but typically takes advantage of a connectivity media/protocol so the I/O module groupings or racks can be located remotely at a distance from the main CPU. Like Local Expansion I/O, this I/O is seen as native to the CPU and functions as if were in the local rack.

Field I/O, also known as Distributed I/O, Network I/O, or simply Remote I/O, is another arrangement using a connectivity media/protocol allowing Field I/O to be located remotely at greater distances from the main CPU. Unlike Remote Expansion I/O, this I/O is typically not native to the main CPU product family and will require extra coding (read/write instructions) to transfer the data.

While Remote Expansion I/O is usually in the same product family as the associated PLC, Field I/O systems are more inter-operable and often third-party products, intended for use with almost any PLC as the host. Field I/O is available in modular, stackable, and slice form factors, often allowing designers to mix and match just a few I/O points economically. Also, many types of field I/O are available with connectorized fittings, integration with pneumatics, and exposed surface-mount ratings, allowing direct cable and tubing connections to field devices to simplify installation.



I/O Wiring Options

The most traditional method of connecting (also called landing or terminating) a wire on a PLC or an I/O module was to use a screw-type attachment, often with a pressure-plate acting as a type of serrated washer to firmly grip the wire. As I/O module point densities have increased, screw terminals became cumbersome and limited miniaturization.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

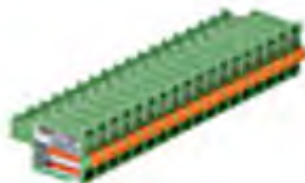
Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Most PLC and I/O wiring is fairly small in diameter (wire gauge) and can be subjected to a lot of machinery vibration. For these reasons, many newer I/O connections are available as spring-clamp connectors, which are a fast, compact, reliable, and vibration resistant connection technology.



With some PLCs and I/O modules, wire terminations are made directly on the device. This means that if the device must be replaced, then the technician needs to unwire and rewire it. Fortunately, many PLCs and I/O modules now use pull-apart terminal block strips, making it fast and easy to replace the underlying device if needed without disconnecting reconnecting field wiring.

For best serviceability, many designers require that field wiring first be landed on more robust panel-mounted terminal blocks, which in turn are wired up to the PLC and I/O modules by a control panel assembly shop.

As a better option, I/O modules using pull-apart terminal block strips can also support the use of pre-wired connection cables and modules, such as the ZIPLink. ZIPLink pre-wired cables and modules save valuable time, keep installations clean and efficient, and use half the space at a fraction of the cost of standard terminal blocks.



I/O Signals

Wired I/O signals are a fundamental way for a PLC to interface with real-world sensors and controlled devices. These I/O signals are available in certain functional and electrical types, which impacts the availability and form factor of the I/O points and modules themselves.

A discrete signal is one that is only off or on. These states may also be described as off=false=0=low, and on=true=1=high. Some examples of discrete inputs are signals from a limit switch or pushbutton, while discrete outputs may energize a solenoid or a control relay. Discrete I/O signaling is commonly performed using 24V DC for safety and other reasons, but other options such as 24V AC and 120V AC are common as well. Discrete outputs can be of the solid-state type (lower power, but fast-acting and compact) or relay type (higher power, but lower density). Other types of signals, such as high-speed discrete pulses, are also possible. It is also important to understand the difference between sinking and sourcing signals (see also: <https://library.automationdirect.com/sinking-sourcing-concepts/>)

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

An analog signal is one that can change over a range between pre-defined low and high limits to indicate a variable signal. There are many raw and scaled ways of representing these values within the PLC, such as 12-bit (0 to 4,095), 16-bit (-32,768 to 32,767, or 0 to 65,535), or as a real floating-point number. Some examples of analog inputs are signals from a pressure transmitter or a potentiometer, while analog outputs may command the position of a control valve or the speed of a motor from 0 to 100%. Three of the most common signaling classes are:

- Current: 4-20mA, 0-20mA, ± 20 mA
(see also: <https://library.automationdirect.com/4-20ma-analog-transmitter-troubleshooting/>)
- Voltage: 0-5V, 0-10V, ± 5 V, ± 10 V
- Millivolt thermocouple (TC) or resistance temperature detector (RTD)

A sensor I/O signal, such as IO-Link or HART, works by superimposing digital communications over a traditional wired discrete or analog I/O connection, so that a host I/O module or gateway can read and/or write other extended signals, configuration, and diagnostics in addition to the primary signal channel.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Industrial Certification

What exactly does “industrial-grade” mean, and what qualifies a PLC platform or component as such? How can specifiers, designers, engineers and end users be assured that the products they are selecting will perform properly and safely?

Ensuring Compliance

To understand how industrial-grade products are conceived, and how they are deemed proper and compliant for an intended purpose, it is important to recognize the details of specifications, standards, codes, regulations, testing, and certification, as well as the types of organizations associated with these topics.

- **Specifications:** These are developed by end-user companies or private design firms to identify criteria that must be met by a product or services supplier, often as a contractual obligation.
- **Standards:** These are commonly developed and curated by standards development organizations (SDOs) or other non-governmental organizations (NGOs). Standards are a guideline describing the best practices and technical details for a product or activity, but they are recommendations and not legal mandates.
- **Codes:** These are developed by governments or approved agencies. They may be based on other specifications or standards, and they define the minimum safety, performance, and/or quality requirements. Codes may be enforced by a regulatory body or law.
- **Regulations:** These are developed by governments and are legally binding rules for protecting public health, safety, and welfare. A regulation may incorporate standards or codes, or it may be developed separately. Penalties may apply to those who violate regulations.
- **Testing:** Testing is performed to determine if products or services meet the applicable specifications or standards.
- **Certification:** Certain independent bodies and nationally recognized testing laboratories (NRTLs) will provide written assurance, usually based on testing, that a product or service meets applicable requirements.

Products that comply with various standards, codes, and regulations provide end user assurance that they will perform their intended function, operate in specified environments, be safe for workers, avoid creating hazards like excessive heat or fire, and resist generating or being affected by electromagnetic interference (EMI) or radio frequency interference (RFI), among other things.

A few of the standards and testing organizations in the industrial automation space are listed here.

UL	Underwriter’s Laboratories
FM	Factory Mutual
ANSI	American National Standards Institute
ISO	International Organization for Standards
CE	Conformité Européene
FCC	U.S. Federal Communications Commission
VDE	Verband der Elektrotechnik
TÜV	Technischer Überwachungsverein
IEC	International Electrotechnical Commission
NFPA	National Fire Protection Association
CSA	Canadian Standards Association
DIN	Deutsches Institut für Normung
ABS	American Bureau of Shipping
CCC	China Compulsory Certificate
CGMP	Current Good Manufacturing Practices, U.S. FDA
ATEX	Atmosphères Explosibles EU Directive
Baseefa	British Approval Service for Electrical Equipment in Flammable Atmospheres

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Why Testing Matters

Product testing and certifications do not create quality, but they ensure a product is designed and manufactured with quality as a key objective. Testing always requires using some standard as the yardstick. For example, making sure a PLC can operate without getting excessively hot can be examined under IEC 61131-2. The components must be appropriate for the application and sourced from reliable suppliers, the circuit board and internal layout must be sufficient for heat dissipation, and so on. The more complex the standard, the more complex the evaluation and testing.

Standards may be imposed externally or internally. Manufacturing a product so it can carry the UL mark means it has to be tested to the relevant UL standard, verified by UL. In other cases, another lab certified by the standard-issuing organization can perform the test, but it can't be done by the manufacturer. On the other hand, a manufacturer may perform its own tests in addition. It can follow an existing standard, such as a MIL-STD, and say that the testing has been carried out in conformance with the standard.

If an existing standard is not available or doesn't pertain to the product as desired, a company can impose its own standard on a product. For example, let's say it wants to advertise a new device designed for use in cold climates, able to perform in an ambient temperature as low as -40 degrees F. The manufacturer may hire an external testing lab to carry out the test, or it may do it internally since it is an inhouse standard. It will write its own procedure saying the unit must be placed in a freezer at -42 degrees F for 24 hours before being powered up, etc. The objective is to determine a method that can be used to prove the point being claimed in a repeatable and documentable manner. Prospective purchasers can decide if they accept its validity or not.

Understanding Quality

It is all too easy to overlook hazards when it comes to electronic and electrical equipment, and then take it for granted that such products are well built and tested to ensure safe performance. However, counterfeit and unlicensed clone products of inferior quality, and lacking any meaningful testing or certification, are an extreme concern.



Caption: The internals of an unbranded, uncertified temperature display. It carries no marks at all, so there is no assurance that it was designed well, uses quality components, or was assembled properly. For example, the MA+ connection (second from left) does not appear to be soldered, and the adjacent connections are hand soldered and inconsistent.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

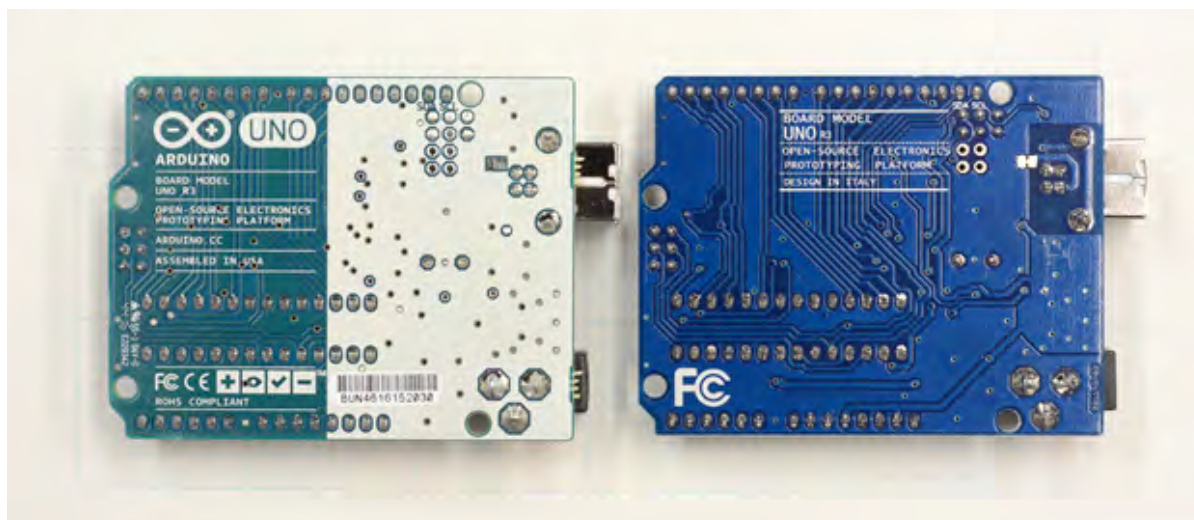
Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Alternately, there are increasingly a wide range of consumer-grade and hobbyist “maker” products, often inexpensive and from no-name suppliers, which might serve for experimental purposes, but are certainly not certified or suitable for industrial-type applications.



Caption: Two functionally equivalent “Uno” boards, one from Arduino (left) and the other a clone, do not carry the same certifications.

Quality automation product manufacturers, such as AutomationDirect, address this problem head-on by offering properly certified devices. Often this means launching a testing program that will be recognized by every major country where a product will be offered. This creates a lot of duplicated effort, but at least most countries have relatively similar testing criteria, and a truly well-designed and manufactured product can pass most examinations.



Caption: Quality-built electronic devices, such as the industrial automation products offered by AutomationDirect, carry certification markings indicating they have passed a variety of tests. The UL mark shown here indicates extensive evaluation and testing has been performed.

Nonetheless, paying for all that testing increases costs and delays product introductions. To further complicate matters, NRTLs usually insist on testing actual production models of a product rather than prototypes. This means a manufacturer must gear up and produce devices months before they can have the necessary marks applied.

The number of certifications required by a given customer will depend on a variety of factors. Some are regionally based and required by law. Specific types of plants, particularly those with hazardous environments such as oil refining or chemical manufacturing, may have requirements outlined by a regulatory agency or industry group. Similarly, an insurance provider can impose minimum requirements needed to keep policies in force.

Product designers who know their efforts will be dissected under a microscope by a dispassionate evaluator realize they can’t cut corners. Everything must be executed correctly at every phase of the process. Test and certification are simply a verification of the correct process.

Passing the test is important, but not as important as a desire to create and build quality products.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Digital Communications

Early PLCs were configured by means of dedicated programming terminals, often in the form of large suitcase-type devices. However, the proliferation of PCs running programming software applications soon became the norm. Users would connect their programming station to the PLC using a specialized parallel or serial cable. Because many PLCs originally operated individual machines or processes in a largely standalone manner, this programming communication link was often their only form of digital communication.

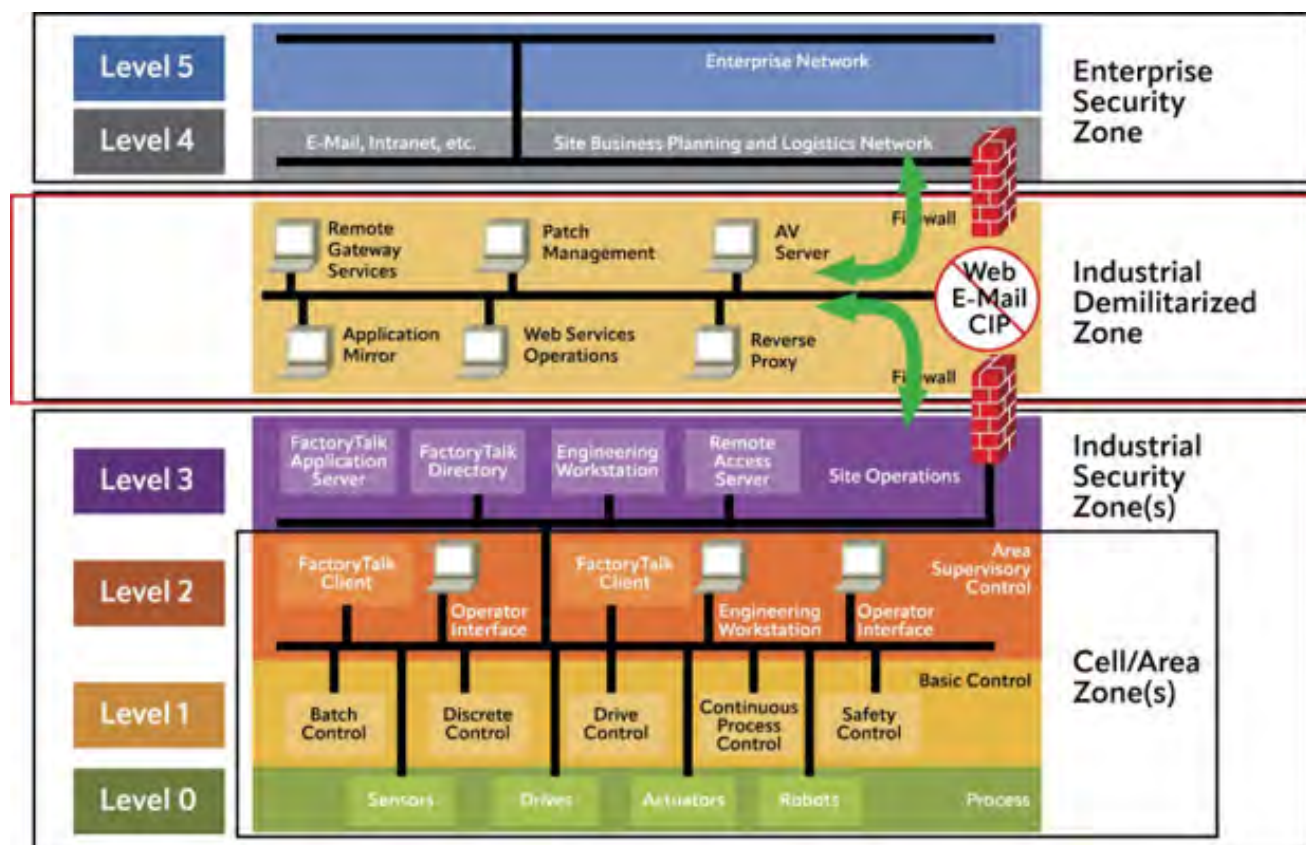
But in the 1980s, it became more common for multiple PLCs to interact with each other as part of larger, integrated systems. The 1990s witnessed new options for hardware OITs and PC-based HMIs, a first step to bridge the gap between the plant floor, and the control room or business front office. By the 2000s, it was commonplace for PLCs to interact with very advanced motion, vision, and robotics systems, as well as with enterprise-level business applications. Today, the need exists to gather even more information from the field for integration with analytics, machine learning (ML), and artificial intelligence (AI) applications.

In support of this shifting role, PLC suppliers have progressively incorporated increasingly capable digital communications and capabilities into their products. Original fieldbus systems were proprietary and focused on delivering the reliability and determinism needed for specific applications. Eventually, modern Ethernet has been extended to provide these capabilities in an open framework, improving performance, interoperability, and security.

Industrial automation systems often remain in service for years or decades, and even new designs may require connection to legacy devices. Therefore, it is important to understand how PLC digital communications have been implemented over time.

Connectivity Hierarchies

To address the growing needs for digital communications among OT and IT areas throughout an organization, it became clear that some form of definition and best practices would be necessary. Fittingly, Purdue University researchers in the 1990s created the Purdue Enterprise Reference Architecture (PERA), often referred to as the Purdue model.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

The Purdue model provides a framework for understanding the hierarchical structure of ICS OT and IT networks, and it consists of five levels:

- Level 0, physical process: This level includes the sensors, actuators, and other physical devices that interact directly with the industrial process. It is also known as the process control level.
- Level 1, basic control: This level includes the PLCs and other devices that control the physical process. It is also known as the control level.
- Level 2, supervisory control: This level includes the OITs, HMIs, and SCADA systems, which provide monitoring over the industrial processes.
- Level 3: The production control: This level includes the manufacturing execution system (MES) and other systems that manage production scheduling, inventory management, and quality control.
- Levels 4 and 5: The enterprise control: This level includes the enterprise resource planning (ERP) system and other business systems that manage the overall business operations.

The Purdue model is a useful tool for understanding the hierarchical structure of ICS and OT networks, and it can be used to design and organize these networks in a more efficient and secure manner. By understanding the different levels and the communication protocols between them, network designers and security professionals can implement efficient and secure data flows.

PLCs reside squarely in Level 1, performing machine and process control. To do this, they need to interact or link with other Levels.

- PLC-to-Level 0: PLCs interacting with remote/distributed I/O and intelligent field devices.
- PLC-to-Level 1: PLCs interact with each other in a peer-to-peer manner.
- PLC-to-Level 2 and above: PLCs interacting with OITs, HMIs, SCADA, and even directly with higher-level enterprise applications.

Modern communications media and protocols have made it possible to collapse the Purdue model structure in certain cases, such as having field instruments connect directly up to the enterprise network but these connections must be cybersecure.

Communications Openness and Determinism

Digital communications options are sometimes associated with geographic regions, industries, and suppliers, which can impact the choices designers make. Two other considerations are how open the technology is, and what degree of determinism is provided.

Open-Source and Proprietary

Because of the relatively specialized and low-volume nature of industrial automation applications, at least compared with other consumer and commercial technologies, there have been many cases where one or more suppliers developed a communications media and/or protocol inhouse and then sold and licensed the products and technologies for others to use. These are called “proprietary” technologies. Usually, a proprietary technology became highly aligned with products offered by the company or companies that developed that technology.

On the other hand, some communications technologies are effectively free for anyone to use, and these are called “open-source” technologies. Interestingly, in some cases a technology that was originally proprietary become open-source when the developer released it as such.

Many end users prefer open-source tech because they can save money and choose what they feel to be the best products to form a solution. However, open-source technologies may not be completely standardized or compatible with each other. Proprietary technologies typically cost more but are managed by the developer, providing better assurance that they will perform as intended.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Determinism

By their nature, digital communications systems send data in groups or packets. This means that there will be some kind of delay from one packet to the next. The ability of a digital network technology to guarantee that data will be transmitted within a specific, predictable time frame is referred to as determinism.

Low determinism might be acceptable for general monitoring, or for signals where speed isn't important like monitoring a tank level. However, a high degree of determinism is needed for PLC I/O signals to operate many types of common equipment. And, for specialized applications like motion control, PLCs need digital networks with exceptionally high determinism.

Communications Media and Protocols

For digital communications, every link between a PLC and anything else relies on some kind of physical media connection, along with a communications protocol which acts as the language carried over the media. The media and protocol must be suitable for both ends of the link, and for the application at hand. It would be exceedingly convenient if one type of physical media connection, along with a single industrial communication protocol, was well-suited for all use cases. However, due to technological evolution and varying performance requirements, there are many connectivity options to consider.

In many cases, commercial IT technologies have been adapted and ruggedized to meet OT needs, but there are also OT-specific implementations. To choose the best solutions, designers will need to understand the following conditions for their applications:

- Communication distance
- Environmental conditions
- How much data must be transferred
- The acceptable bandwidth, or data transfer rate
- Degree of determinism

Further complicating the topic is that OT installations face demanding requirements to withstand mechanical stresses, moisture, extreme temperatures, and EMI/RFI, all over potentially long distances. Also, each PLC and field device may only be available with certain communications technologies.

Following are some key concepts regarding openness, protocols, and media, addressing some of the most well-known, but certainly not all, versions.

Industrial Physical Media

- RS-232, RS-485, RS-422 serial copper 2-wire or 4-wire:
This is a classic form of digital connectivity but it is still relevant for certain devices today. Depending on the version, it can be a point-to-point connection between two devices, or a multidrop bus connecting up to 32 devices along a single cable. Also depending on the version, the connection distance can be limited to only 15m, or up to 1,200m.
 - Pros: Inexpensive, well understood, can work well in OT environments over long distances.
 - Cons: Slow speeds, usually not over 115.2 kbps.
- Fieldbus, media varies:
There are several OT-specific fieldbuses, often developed in proprietary manner, which use twisted pair or coaxial cables. They are usually a form of serial communications which has been modified to improve communication speed, distance, robustness, or other characteristics.
 - Pros: Fieldbus networks are purpose-built to work reliably throughout industrial environments, with adequate performance and good compatibility with certain PLCs.
 - Cons: As a specialized technology, fieldbuses often lock-in designers to only certain PLCs and other compatible products, at a greater expense and requiring some specialized knowledge.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

- Ethernet, copper with twisted pairs:

Commercial Ethernet media has been adapted for OT use in many ways. Properly installed cables with shielding or in conduit can work well, and ruggedized M12 connectors may be used instead of common RJ45 connectors for unprotected locations. It is usually preferred to choose industrial-specific Ethernet switches for OT use, as they incorporate features to ensure best performance. Two twisted pairs (used in 4-pin D-coded M12 connectors) are generally limited to speeds up to 100 Mbps (10/100BASE-T), while four twisted pairs (used in 8-pin X-coded or 8-pin A-coded M12 connectors) are required to achieve Gigabit (1000BASE-T) and 10 Gigabit Ethernet.



- Pros: High speeds (commonly 100 Mbps, 1 Gbps, or even more) are possible and are usually more than sufficient for OT applications. In some cases Power over Ethernet (PoE) can be useful for energizing a remote device.
- Cons: Standard segment length is 100m, although switches, repeaters and other devices can extend this. Requires careful design, IT management, and good installation practices.
- Single Pair Ethernet (SPE) and Ethernet Advanced Physical Layer (Ethernet-APL), copper single twisted pair: SPE and Ethernet-APL are relatively recent evolutions (from around 2015 to 2021) of Ethernet media, supporting native Ethernet functionality over a single twisted pair cable. These media may be able to re-use existing installed cables, or more likely can be installed as new cables.
 - Pros: 10 Mbps speed is sufficient for most OT applications, reaches up to 1,000m, and can supply data to the remote device. Ethernet-APL can be suitable for installations in hazardous explosive environments.
 - Cons: This is a new infrastructure, and devices such as switches and instruments are only recently becoming available.
- Fiber optics (FO), both multimode and singlemode:

Fiber optics are completely immune to EMI/RFI and can reach very long distances.



- Pros: Ethernet over FO, using transceivers, is an effective technology for long OT communication runs.
- Cons: High design, installation, maintenance, and device costs.
- FWi-Fi, wireless Ethernet:

Commercial Wi-Fi can carefully applied in OT settings, especially if it is used in non-critical roles.

 - Pros: Familiar to many users, and a convenient way to implement HMI/SCADA using portable devices.
 - Cons: Very susceptible to EMI/RFI, which is abundant in many OT environments.
- Other wireless:

Bluetooth is a low-power, low-range wireless technology which can be used with some PLCs and field instruments, usually for local configuration and diagnostics, but not for run-time communications due to its limitations. Cellular 4G/5G, WirelessHART, long range wide area network (LoRaWAN), and industrial cellular (both LTE-M and NB-IoT) are other digital wireless communication technologies used in industrial environments. Each has strengths and weaknesses with regards to bandwidth, update rates and other considerations.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Industrial Communication Protocols

- Serial and legacy protocols (1960s, 1970s):

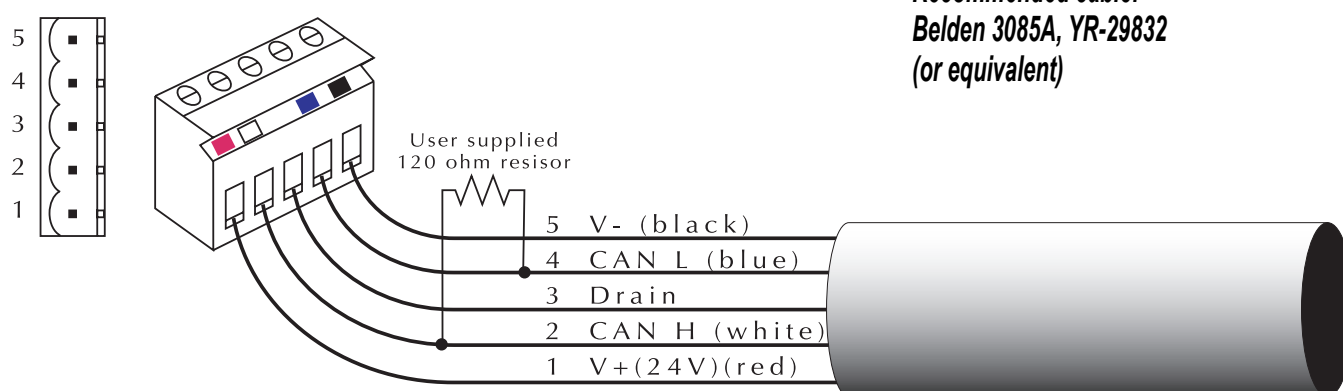
These typically operate over serial networks, with a single active node (master) polling one or more passive nodes.

- American Standard Code for Information Exchange (ASCII) was developed in the 1960s and is a basic way of encoding characters digitally. Character strings can be used to send and receive data, but they must be formed and parsed carefully.
- Modbus remote terminal unit (RTU) was developed around 1980 by Modicon, and later supported by Schneider Electric and then the Modbus Organization. Modbus RTU is one of the original protocols developed for PLCs, useful for mapping raw data using 16-bit registers. Although this protocol is quite old, it remains commonly used.
- DF1 and Data Highway + (DH+) are legacy protocols developed by Allen-Bradley (Rockwell Automation).

- Classic fieldbuses (1980s, 1990s):

These are usually specialized serial networks, mostly used by PLCs to connect with field devices.

- DeviceNet (DN) was developed by Rockwell Automation and is based on the Controller Area Network (CAN) architecture. DN is a proprietary tech, using specialized cabling, with various restrictions on node counts and cable lengths, to provide power and data connectivity for up to 64 nodes. Rockwell later made this technology open, and it is managed by independent North American standards development organization ODVA. DeviceNet is part of the Common Industrial Protocol (CIP) family of protocols.



- PROFIBUS Decentralized Peripherals (DP) and Process Automation (PA) was developed in Europe and is commonly associated with Siemens, but today is managed by PROFIBUS and PROFINET International (PI). DP is used for high-speed machine/factory applications, while PA is for process instrumentation and intrinsic safety.
- Foundation Fieldbus (FF) was developed as a digital two-way communications protocol operating over two-conductor cables to connect field instruments with a host. Today, FF is managed by the FieldComm Group. FF is relatively slow, but useful for process industries, especially when intrinsic safety is needed.
- HART protocol is not exactly a fieldbus, but it does superimpose digital communications over a traditional 4-20mA analog signal loop. HART is easy to use, although relatively slow, and it offers a way to monitor several signals over a single channel.

- Modern fieldbuses (2010s):

Networking technologies add cost and complexity to a device, so it has not been practical to push full networkability out to every single field device, especially those that are small, distributed throughout a machine, and with little or no processing power. However, it is possible to apply right-sized digital technologies even here.



Chapter 6

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

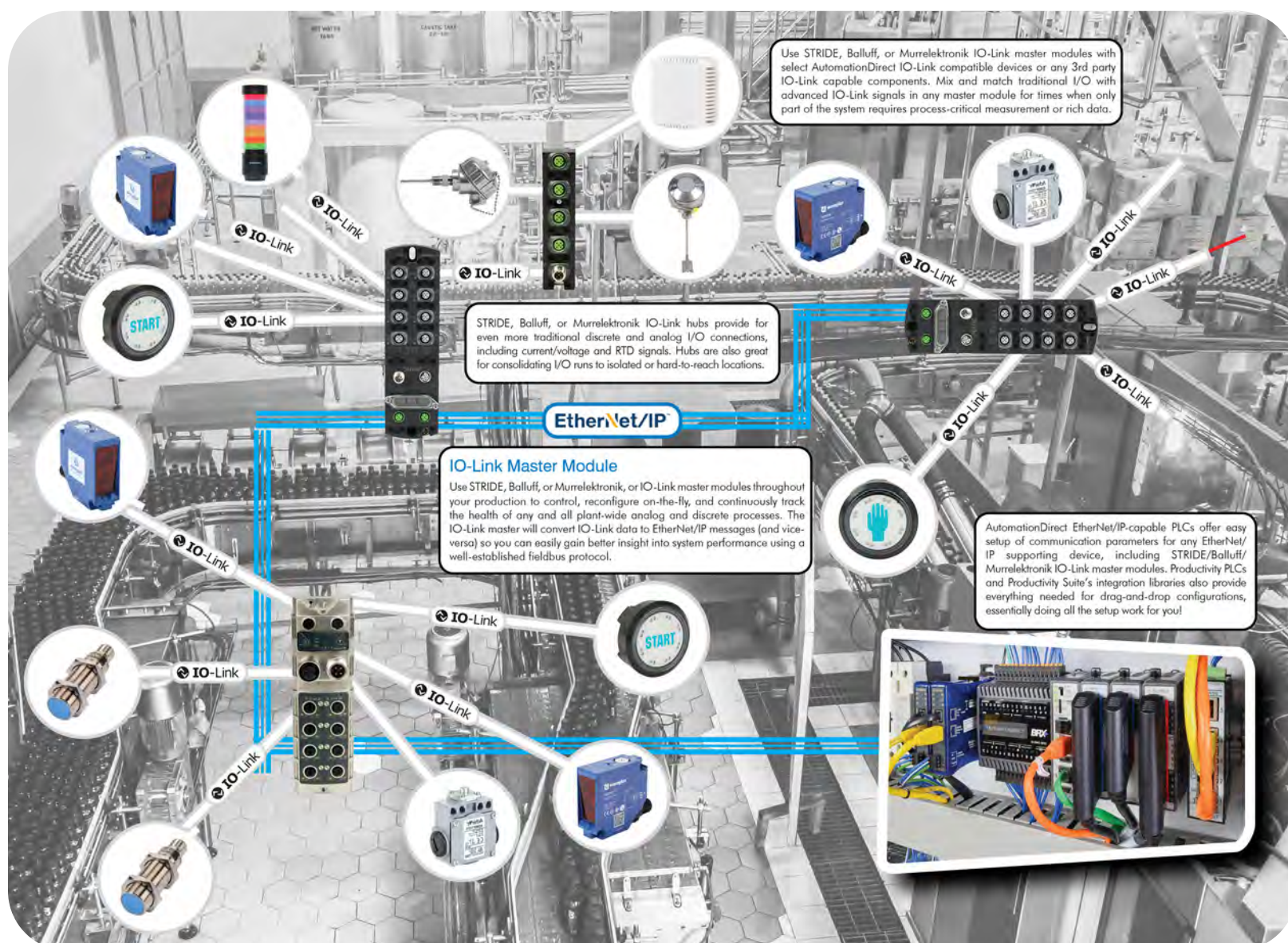
Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

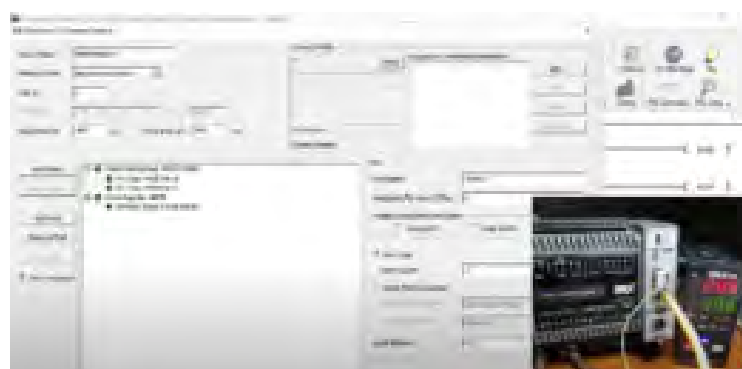
Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

- IO-Link is a relatively new standard protocol that enables bi-directional data at low cost between field devices (such as sensors and actuators) and a host device (such as a PLC, or an IO-Link gateway which translates to an industrial Ethernet protocol). With an I/O-Link connection, the field device is powered and a primary device signal of On/Off for discrete or 0-100% for analog are still read or commanded. However, the digital link allows more extensive process data and diagnostics to be read, and configuration to be performed, by the host.



- Modern Ethernet-based industrial protocols (2000s and beyond):
Although Ethernet was becoming common for commercial applications by the 1990s, it was not until the 2000s that it was deemed suitable for many industrial applications, especially related to PLCs and I/O. In some cases, commercial off-the-shelf (COTS) Ethernet can be installed to handle certain industrial locations, and there are also many media options specifically designed for industrial installations. Furthermore, in many cases multiple protocols can operate concurrently over a single Ethernet network.
- Modbus TCP wraps the basic Modbus RTU serial packet into a standard Ethernet TCP/IP network frame. This is a simple protocol, with no special hardware or licensing needed, but it lacks determinism and integrated diagnostics. This protocol is open and had about a 4% market share in 2025.



Jump to Chapter

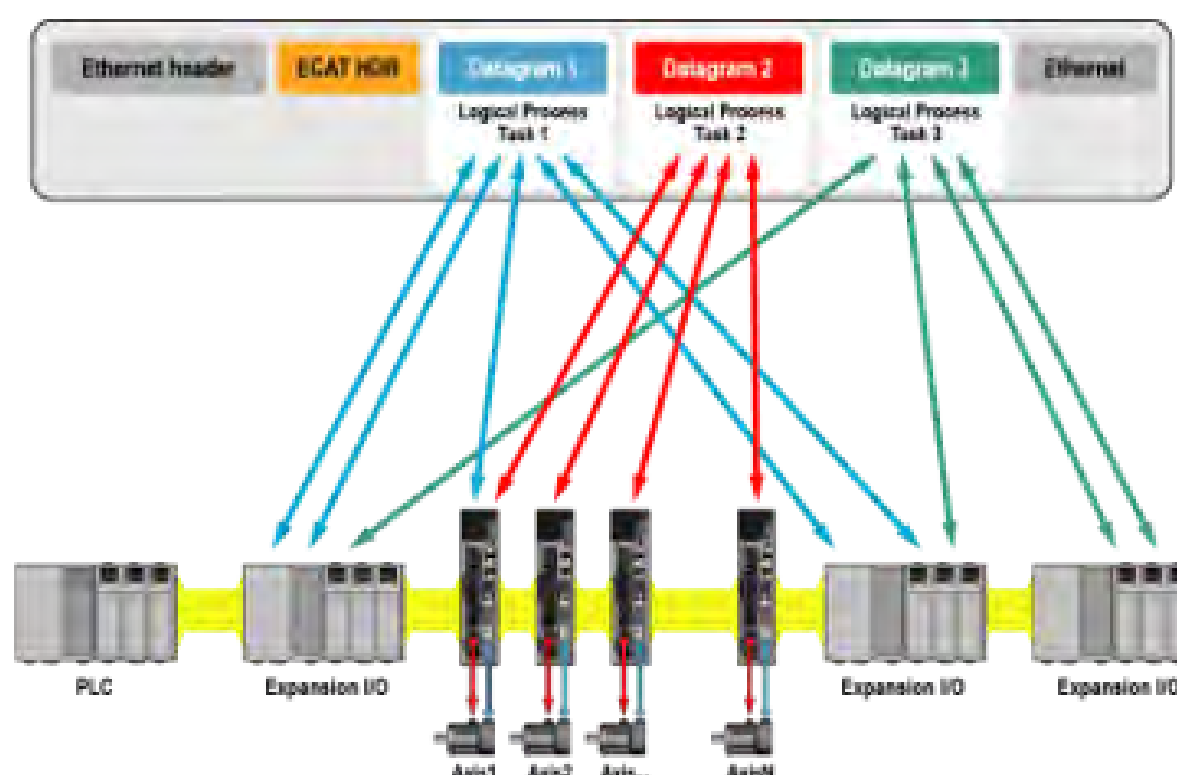
Chapter 1
What is a PLC?**Chapter 2**
PLC Basics**Chapter 3**
How to Choose a PLC Platform**Chapter 4**
PLC and I/O Hardware Components**Chapter 5**
Industrial Certifications**Chapter 6**
Digital Communications**Chapter 7**
PLC Programming Software**Chapter 8**
PLC Programming Languages**Chapter 9**
Ladder Logic Programming**Chapter 10**
Practical PLC Topics**Chapter 11**
AutomationDirect PLC Platform Solutions**Chapter 12**
Applications

- EtherNet/IP extends CIP (related to DeviceNet) into standards TCP/UDP networking. It provides good performance and some motion control capability. EtherNet/IP is popular in North America, and it remains associated with Rockwell Automation although it is now managed and supported by ODVA, with about 23% market share in 2025.

[SEE ALSO: <https://library.automationdirect.com/ethernetip-implicit-vs-explicit-messaging/>]

- PROFINET uses standard Ethernet, although there are some specialized versions. It provides good performance. PROFINET is popular in Europe, and remains associated with Siemens, although it is now managed by PI, with about a 27% market share in 2025.

- EtherCAT is a preferred and popular option for motion control over Ethernet. Developed by Beckhoff Automation in the early 2000s, and then standardized under IEC 61158, the EtherCAT protocol works over standard Ethernet media and architectures much more efficiently than other general industrial protocols. It is designed to deliver deterministic cycle times, which is essential for maintaining accurate synchronization among all communicating devices in high-performance applications, such as multi-axis coordinated motion control.



- There are other industrial Ethernet protocols not covered here.

• Interoperability protocols:

The previously described protocols are very PLC-centric, enabling PLCs to interact with a wide range of devices. However, there are some other protocols more focused on transporting rich data, although they can be used with PLCs.

- OPC Unified Architecture (OPC UA) is a platform independent, service-oriented, open-source, IEC 62541 standard architecture designed for secure and reliable communications. OPC UA is managed by the OPC Foundation.
- Message Queuing Telemetry Transport (MQTT) is a lightweight, publish-subscribe messaging protocol designed for efficient communications, especially between devices with limited resources or where network connectivity is unreliable. Devices publish data to a central broker, which then distributes the information to subscribed clients. Many IoT implementations adopt Sparkplug B to extend the MQTT protocol with a standardized data framework. MQTT is an OASIS open standard and an ISO 20922 standard.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

PLC Programming Software

While most of this eBook so far has focused on hardware, for most users the software they interact with plays an important part in the PLC selection process. Generally, each family of PLC has a specific programming software integrated development environment (IDE) associated with it. The IDE is needed to configure the PLC hardware and develop logic and data-handling programs—and to test, commission, and troubleshoot the resulting application.

The right IDE software makes all these steps much easier. Here are just a few of the top aspects to think about regarding PLC software:

- Overall usability of the IDE and support for learning its capabilities
- Upfront costs, and ongoing licensing or support fees
- Incorporation of modular and reusable coding, import/export, and similar features to improve efficiency
- Availability of testing and troubleshooting tools

Some features of the IDE are based specifically on what the associated PLC hardware platform supports, such as:

- Available OT and IT programming languages and protocols
- Memory organization (tagname versus fixed memory addressing)
- How smoothly can other common automation devices, such as HMIs and VFDs, be interfaced via the software

Programming IDE

Most modern PLCs are configured, monitored, and programmed using an IDE application running under Windows on a desktop or laptop PC. The IDE will have provisions to save, restore, and transfer programming projects. There are some cases where browser-based interfaces, or mobile apps, can be used for some or all these tasks.

Basic Functionality

The IDE can be used offline to develop programs, or it can be connected to a PLC in an online mode for uploading/downloading, monitoring, and testing. Classic PLCs often used serial ports for this purpose, with generic or even proprietary cables. However, today's PLCs generally use readily available USB cables or wired/wireless Ethernet to connect a programming PC with the PLC. In some cases, this connectivity, although perhaps with limited functionality, may be accomplished with Bluetooth.

Updates

While PLC hardware can often run non-stop for years or decades without being touched, many installations will require attention over their lifetime for troubleshooting, to revise the program, or for other reasons. Occasionally, newer firmware will become available for in-service PLCs, and updating this firmware may be optional or required, depending on the issue and the manufacturer. Therefore, ongoing availability of the proper IDE for these purposes is important.

Unlike the hardware, the IDE software generally receives frequent updates to add both usability and core programming features, incorporate new hardware types, address compatibility and cybersecurity issues, and more. While it is sometimes possible to use a given IDE version on a maintenance laptop for many years without performing updates, this is usually not recommended.

Licensing and Support

PLC hardware is a one-time, upfront purchase, and the user owns the product from then on. IDE software can be another story entirely. Some suppliers charge an upfront fee for the software while other suppliers charge a recurring fee (usually annually) for ongoing software updates, often associated with their phone/web support for that software. In the worst case, the software from a supplier may cease to work if it falls out of support. If software support lapses, an end user may need to purchase it again to resume support.

A paid software model might be acceptable, although not ideal, for large enterprises or systems integrators, but it can be problematic for smaller end users and those who only need to access their PLC hardware infrequently.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

To address these issues, AutomationDirect has long provided free software and support for their modern PLC families. Current and potential users can download the latest versions of CLICK, Do-more Designer (for Do-more and BRX PLCs), and Productivity Suite programming software right from the AutomationDirect website. Or, for a nominal fee, they can order a USB copy. In this way, current users can keep their systems up-to-date, and future users or curious support personnel can try the software out and learn how to use it.

Similarly, while most suppliers charge for support or limit access to resources, AutomationDirect provides a wealth of product support for free, including:

- Comprehensive documentation and manuals
- CAD drawings where appropriate
- A user forum
- Many videos showing “how to” perform configuration and programming tasks
- Phone support

The availability and variety of software and support options make AutomationDirect PLC platforms exceptionally open and accessible for all types of users.

Key Features

Every user has their own needs and preferences when programming a PLC and integrating it with other automation elements. There is no way to encompass every aspect of what makes up good PLC programming software. However, following is a series of thoughts on features and tools that AutomationDirect believes are helpful for developers as they consider what might be the most important. Remember, each platform will offer some but possibly not all these features, so it is crucial to consider what is most important for a given application, and then determine which platform is the best fit.

Programming Connectivity

Once the programming IDE is launched, the next natural step will be to connect with a PLC to configure the hardware, download a program, or monitor an existing system. A lot of PLCs today offer plug-and-play USB connections for programming, which are fast and easy to use but require direct connections between the PC and PLC’s CPU module. Of course, this direct connection is sometimes essential upon first startup of the hardware, or if the network is not accessible.

With networked PLCs, Ethernet is typically the connection method, so the programming PC will need proper credentials to access that network. One last option users may find helpful when it comes to downloading is USB project transfers, but only if the PLC allows projects to be downloaded via an onboard USB port. This can be important for remote PLCs that aren’t easily accessible with a PC.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

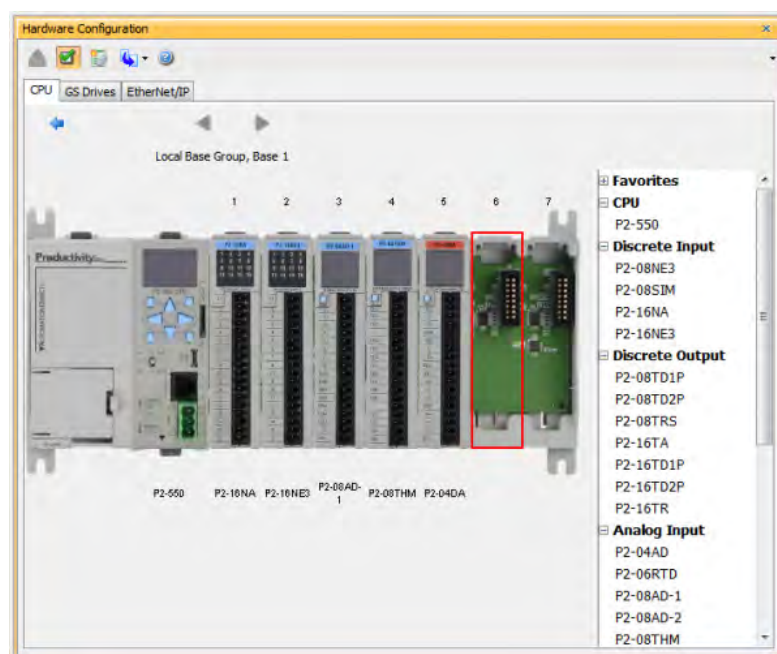
Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Hardware Configuration and Auto Discovery

The PLC IDE will have provisions for the developer to define what controller, I/O modules, and other hardware will make up the configuration. This typically can be performed manually. However, when auto discovery is available, it is a huge timesaver because the developer can simply install the I/O modules with the PLC and power up the equipment. The software will then automatically discover the installed modules and assign physical I/O tags based on each module's position. From there, the PLC is ready to program, or it can be manually reconfigured with a new arrangement and tags if needed. Some software can also auto detect other connected devices, such as VFDs, to make configuration even easier.



Run Time Transfers and Online Edits

Once a project is up and running, system modifications can be more challenging, especially if other portions of the system need to keep running. Older and smaller PLCs may not allow run time transfers of an entire program—or even online edits of just a rung or two—to be performed while the PLC is in run mode. Instead, the PLC may need to be placed into stop mode, then edited, then placed into run mode, which is disruptive if the PLC is actively controlling equipment.

Fortunately, more modern PLC platforms and IDEs generally feature a way to transfer project edits into the CPU without stopping the CPU scan, which helps to avoid costly shutdowns and production losses whenever changes are needed.

Customizable Layouts

No matter what the PLC IDE offers, user friendliness is a big advantage. Users should look for a clear and consistent basic interface, with the ability to personalize the programming experience to fit individual needs by using capabilities such as window docking, display/view options, accessibility options, and text options.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

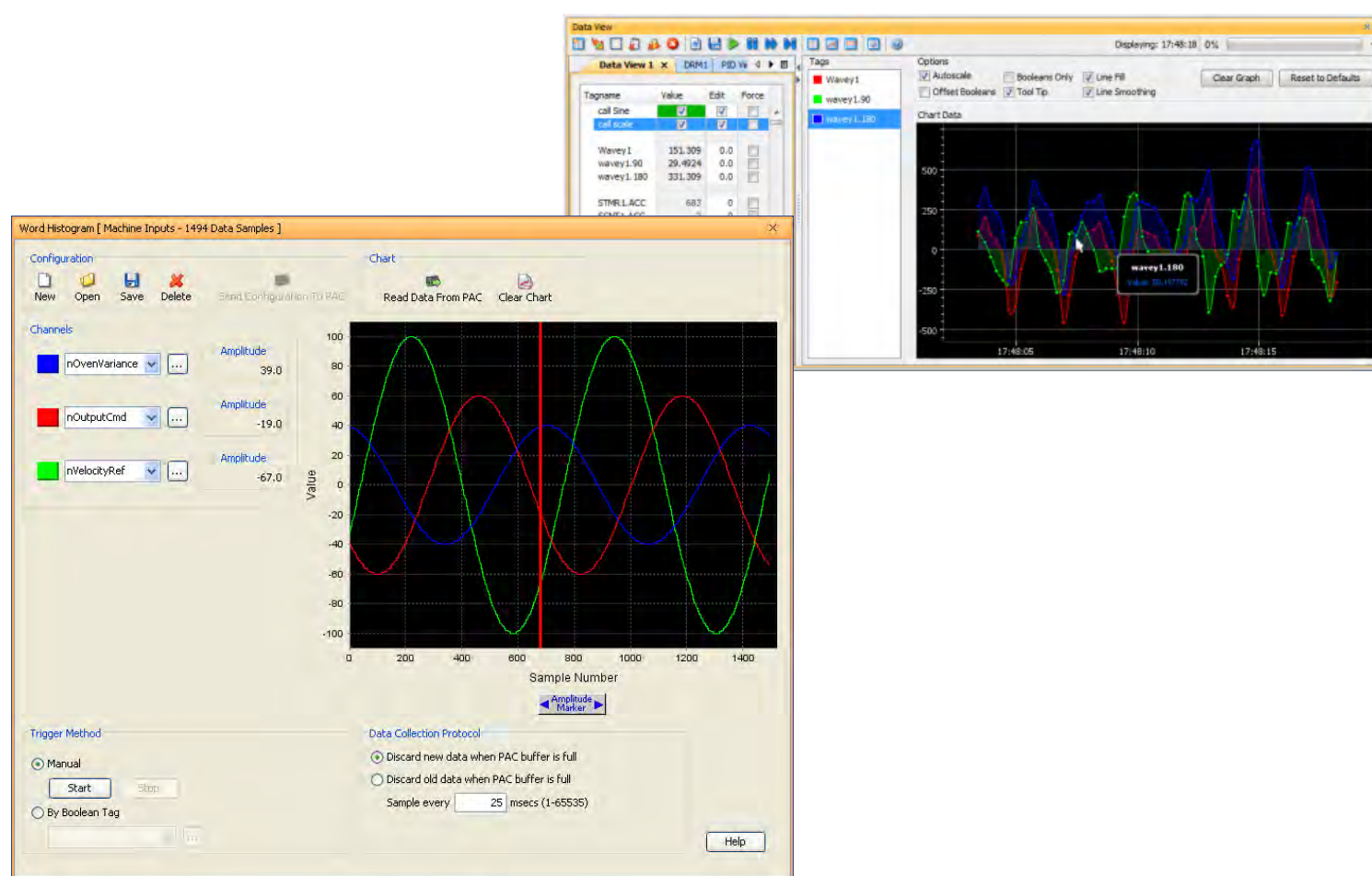
Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Data View and Histograms

Data view windows allow users to monitor and manipulate PLC logic and data values in real time. Although most PLC software will have a way to explore the memory addresses or tagnames to do this, there are some advanced capabilities, such as creating custom data view groupings, that can be extremely helpful.

Histograms (graphical trend charts) are especially useful during program operation to view both discrete and analog data changes over time. Custom data views and histograms are helpful during program development and testing/commissioning, and then for troubleshooting once the program is in operation.



Search, Replace, and Cross Reference

Three of the most often used IDE functions when creating a PLC program are search, replace, and cross reference. These allow the programmer to find certain tags or addresses (with a variety of filtering options), replace tags (useful for duplicating logic), and identify all instances where a tag is used. These are essential capabilities for supporting efficient programming and troubleshooting.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

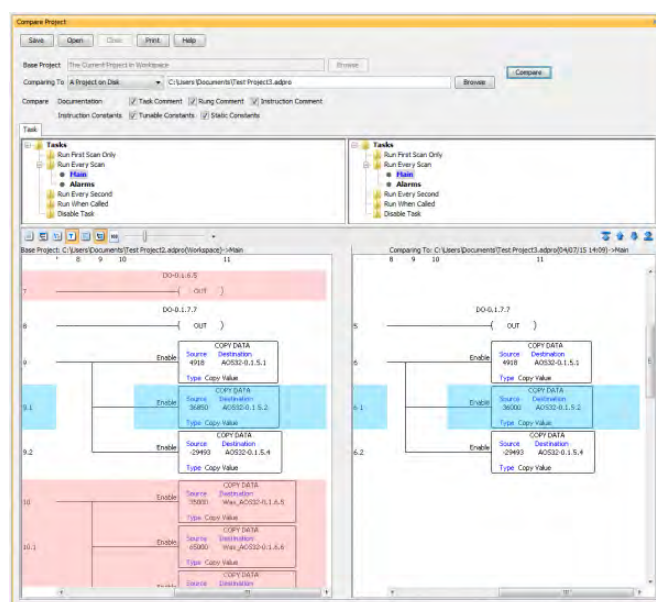
Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Project Compare

A typical PLC user works with many PLC systems and programs, especially if they are a systems integrator. Sometimes two machines may have their own programs which are very similar but not exactly the same. A project compare function allows the user to perform project-to-project program comparisons to obtain detailed information about the differences, a powerful tool for code reuse.

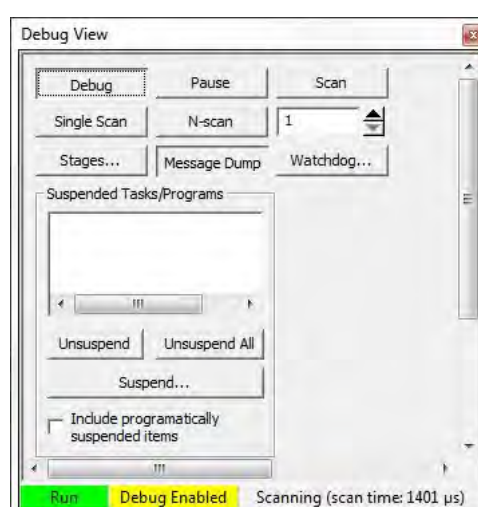


Task Manager

The simplest PLC programs may just be a few rungs long. However, most users will appreciate the ability to organize their programs functionally, often as subroutines which may execute when needed (every scan, on first scan only, on a fixed time basis, or only when called). Older PLCs may require extra lines of code to do this, while newer platforms and IDEs usually support some form of visual-based task manager functionality so users can structure and optimize code execution.

Debugging Tools

A PLC in normal operation scans rungs continuously in accordance with how the tasks are configured. If there is a problem with the code, sometimes called a bug, it may be hard to spot because the code executes so quickly. In these cases, users will appreciate debugging tools that make it possible to step from one rung to the next, execute only a few rungs, run one complete scan, and/or pause the execution altogether of selected tasks or routines.



Instruction Sets

A simple reality is that smaller and more basic PLC platforms offer a limited set of fundamental instructions, while larger and more sophisticated platforms will have far larger instruction and function sets. Before starting a project, it can be hard to know exactly what functions will be needed. However, designers should determine what more advanced capabilities—such as elaborate math, PID loop control, motion control, and communications—may be needed, and then make sure to research what each PLC platform offers in this regard. This investigation can be conducted by reading the documentation, or by downloading and installing the free software to get a hands-on look at what it offers.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

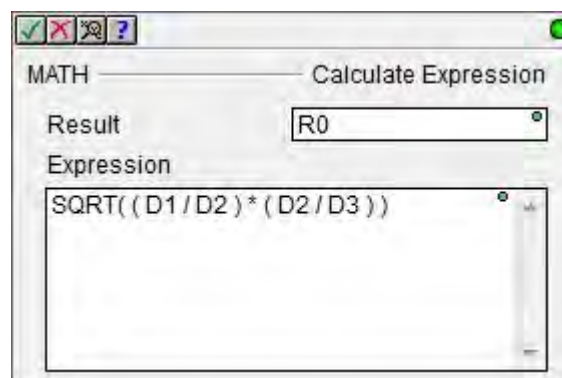
Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

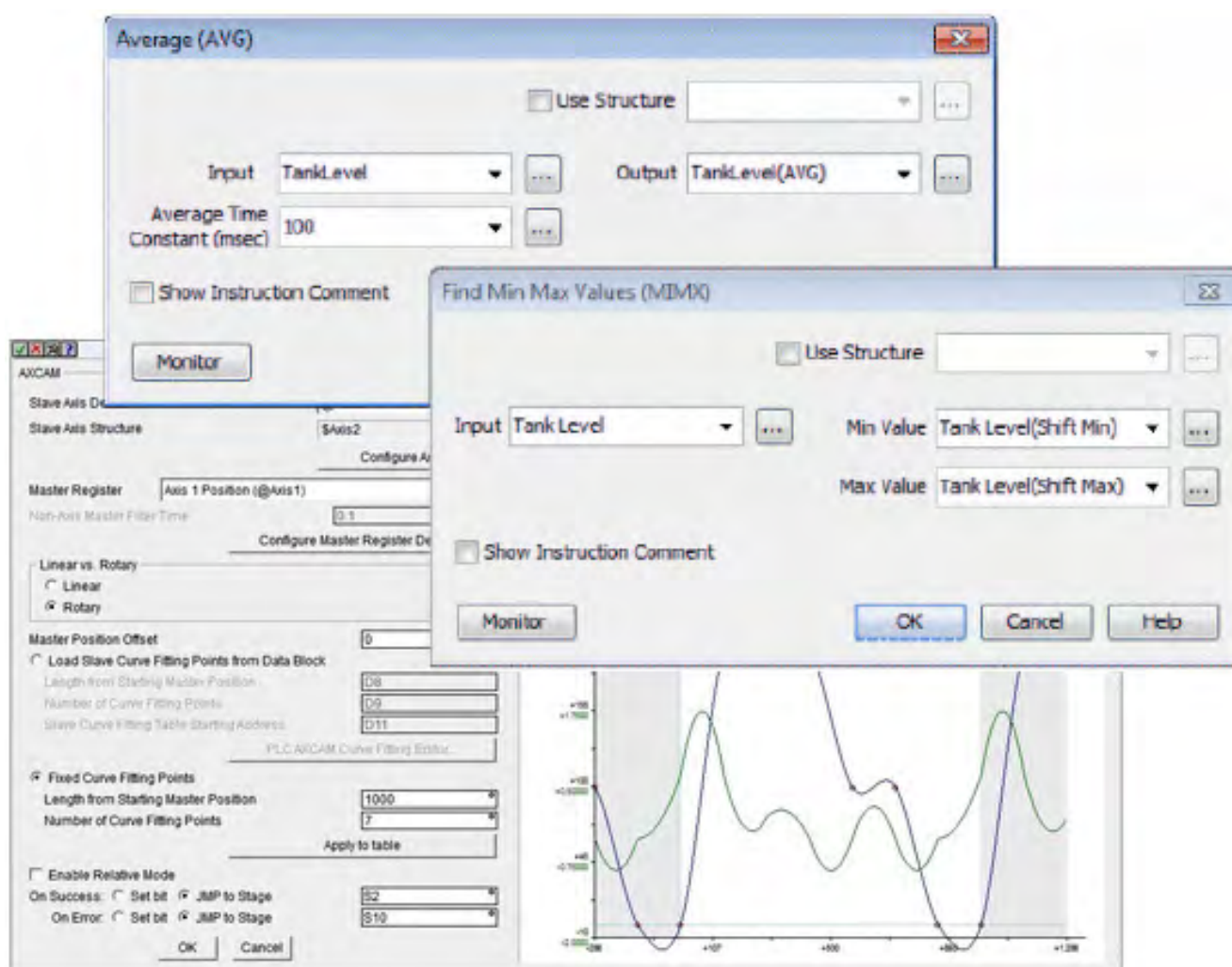
Powerful Math Functions

While the earliest PLCs were designed for on/off discrete logic, they quickly gained math functions to support analog signals and a host of other capabilities. Ladder logic originally provided some rudimentary math functions, but many applications need the ability for programmers to easily code complex mathematical expressions using integer and floating-point numbers directly into a text-based editor, and some PLC platforms provide this capability.



Integrated Function Blocks

Similar to math functions, integrated function blocks can make programming a lot faster and cleaner. Instead of creating a large amount of ladder logic rungs to handle a complex task, a solid set of function blocks can do most of the work associated with PID loops, motion commands, communications, and more. These blocks can then be reused as needed, a huge time saver for programming and testing, because each type of block only needs to be tested once.



User-Defined Structure (UDS) Data, and User-Defined Instructions (UDIs)

Developers can use UDS to create specialized data types, and UDIs to create custom blocks of logic, supporting efficiency, reusability, and consistency throughout the programming. [Refer to Chapter 8 for a more in-depth explanation of UDS and UDI.](#)

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

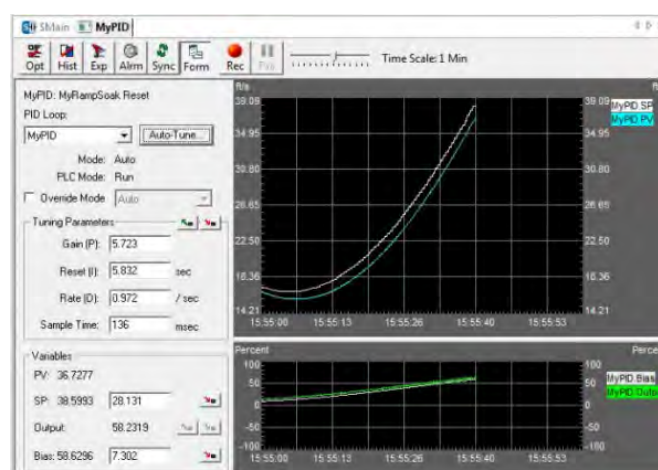
Chapter 12
Applications

PID Loops

PID loops are a very specific form of regulatory control for analog process signals. [Refer to Chapter 10 for more in-depth detail about PID loop control; a summary follows here.](#)

A PID loop commands a controlled output (CO)—such as a VFD-driven motor speed or a valve position—to make adjustments so as to maintain a process variable (PV)—such as flow, level, pressure, temperature, or other measured signals—at a desired setpoint (SP). The PID algorithm is based on the difference between the SP and the PV, and it makes adjustments based on the current error (proportional), the history of past error (integral), and how fast the error is changing (derivative) to provide smooth control without over- or under-correcting, each of which would result in overshooting, oscillation, and other problems.

While a PID algorithm can be coded and then tuned manually, it is such a common need that using a dedicated function block—which may also provide other benefits such as auto-tuning, trending, and simulation—is the best choice. The most basic PLCs may not offer PID functionality, but if it is needed then users should investigate how many PID loops a PLC can support, whether auto-tuning is available, and what additional functions are provided.

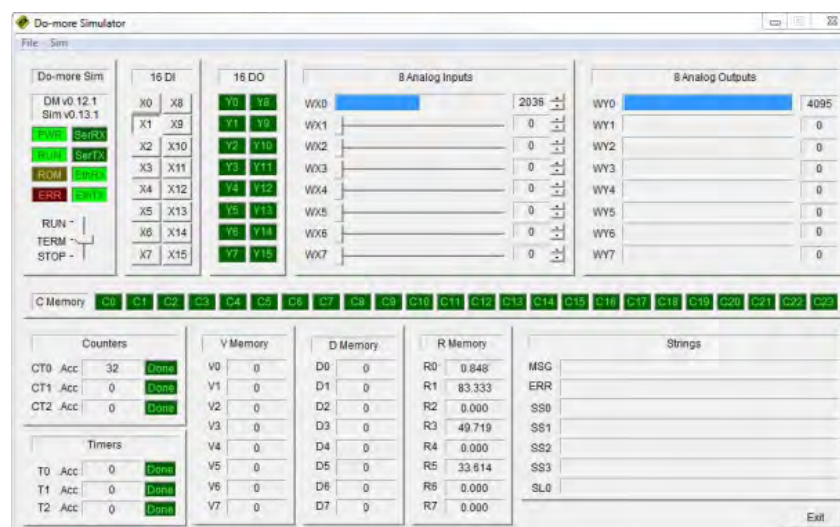


Security and Cybersecurity

Cybersecurity is a much bigger topic than can be covered here, but each PLC platform, especially those with network connectivity, should support some basic security functions. One example is role-based passwords to control who has what type of access to the system. Another type of security empowers developers to protect portions of code that they deem as intellectual property (IP) so they can't be changed or even viewed. Refer to Chapter 10 for more in-depth detail about security and cybersecurity.

Simulator

A built-in simulator is extremely useful in PLC project development because it provides a virtual PLC running right on board the programming PC, acting like a real-world physical PLC, with some limitations. The developer can download code, execute the program, and monitor operation. A good simulator will emulate analog and discrete I/O, and it will provide access to logic elements such as timers, counters, control bits, and more.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Help Documentation and Videos

Good PLC programming software should have good help files. Embedded help files are a great asset when learning about an IDE, investigating a new instruction/function, or when troubleshooting. Help files should provide information in a way that is easy to comprehend, with visuals, screenshots, and application examples. They should also include the ability to search for items, with results printable.

Taking advantage of newer technology, AutomationDirect has also developed extensive help videos, with some embedded right in the IDE and a huge library available online via the IDE, providing step-by-step instructions on how to configure hardware, use the instructions/functions, and more.

Remote Connectivity: MQTT, Web Server, Mobile Apps, StrideLinx Cloud

Modern PLCs offer many forms of remote connectivity over the local site networks and the internet/cloud, empowering users to view and access diagnostics and program information. Users should investigate what options may enhance their applications, such as:

- **MQTT:** Mentioned in a previous section, a PLC supporting the MQTT protocol can publish data to a remote broker, and users can then access that data via a web browser or an app.
- **Web Server:** Some PLCs can actually host an onboard web page. This allows properly credentialed remote users on the site network, or even via the internet, to access the web page via a browser.
- **Mobile Apps:** A mobile app provides another way for users to view and edit PLC registers and view PLC status from an iOS or Android device.



Email and Texting

Integrated email and texting capabilities allow a PLC to notify personnel when events occur, providing a flexible way to stay informed about operations and possible failures so staff can respond rapidly. Some PLCs may even offer secure emailing or data attachment capabilities.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

PLC Programming Languages

The previous chapter focused on the IDE used by developers to configure and program PLCs. This chapter shifts to examine the various PLC programming languages themselves, and some related software development concepts. Just as when considering the IDE, each user needs to evaluate their needs and development preferences to select the best-fit languages and features for their application.

[PLCs have traditionally been programmed using ladder logic, refer to Chapter 9 for an in-depth discussion of this language.](#) Ladder logic is a graphical programming language designed to resemble electrical ladder diagrams, allowing electricians and engineers to read and modify control programs without extensive software training.

However, over the years many other options have become available to best meet the needs of various automation and data handling tasks.

The following sections will look at memory addressing schemes, and the benefits of following an object-oriented approach when creating programs. Then we will cover the main OT-based programming languages, the role of CODESYS for PLC programming, and some IT-based programming languages that are useful for PLC applications.

Memory Addressing

Fixed Addressing

Older PLCs, and even some newer models, use a memory structure scheme called fixed addressing. The I/O module addressing may be based on a rigorous combination of rack, slot, and point numbers, while general program addressing depends on assigned groupings of memory types (binary, integer, floating point, and others). For some PLCs, the memory structure is predefined with a finite number of elements allowed for each memory type.

When programming a BRX PLC, for instance, the element consists of the rigorous address, and the user can also associate a brief nickname/symbol and an optional and more verbose description with each element.

While fixed addressing is often considered an old-school concept, it can offer better searchability and provides a way for the developer to rigorously arrange the memory structure to meet their needs.

Tagname Addressing

Many contemporary PLCs, such as the Productivity series, use tagname addressing. Because there is no predefined memory structure, it is possible to create as many elements as needed for each memory type (within the constraints of the PLC overall memory capacity).

In the IDE for these PLCs, the developer creates the data elements they need, each with a unique name/tagname, and an associated and optional verbose comment/description. Behind the scenes, each tagname may have some sort of rigorous system ID, but the developer does not need to manage this in any way.

A major benefit of tagname addressing is that it offers plain-language readability, preferred by many developers, and more closely resembles variables in IT-type languages. Tagname databases can also be easily integrated with HMIs and other database software.

Find/Replace and Import/Export

While all memory addressing can be performed within the IDE environment, there are additional functions that can help accelerate the efforts of developers. Find/replace, just like in a word processing program, helps the developer to search out tags when needed, and to make bulk editing changes or replications.

Import/export is used by many programmers to improve their efficiency at duplicating memory addresses. For example, the programmer might create a memory addressing structure for one machine using the IDE. They could then export this structure to a comma separated variable (CSV) or other spreadsheet file, and then use a spreadsheet program to replicate and bulk-edit elements, tags, comments/descriptions, and other attributes to accommodate several other identical machines, but of course with unique tags and descriptions. Finally, they could import this file back into the IDE, with all addressing in place to support the additional production lines.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Object Oriented Programming

Object-oriented programming (OOP) helps manage complexity and promotes the reuse of logic without duplication. OOP speeds development, improves maintainability, and reduces the risk of programming errors. OOP is just one of many IT-centric paradigms that can also be applied to OT-based technology like PLCs, and even to ladder logic.

Classic PLC code is usually copy/pasted many times to support multiple instances of a certain type of program function. For example, if an application had 10 control valves, there would be 10 almost identical sections of ladder logic code.

A better approach is to use OOP, where logically related data and functionality are structured as cohesive and easily reusable “objects.” Terminology varies, but every main “parent” object is typically instantiated many times as “child” objects that derive their data structure and functionality from a definition that is shared with every other object of the same type.

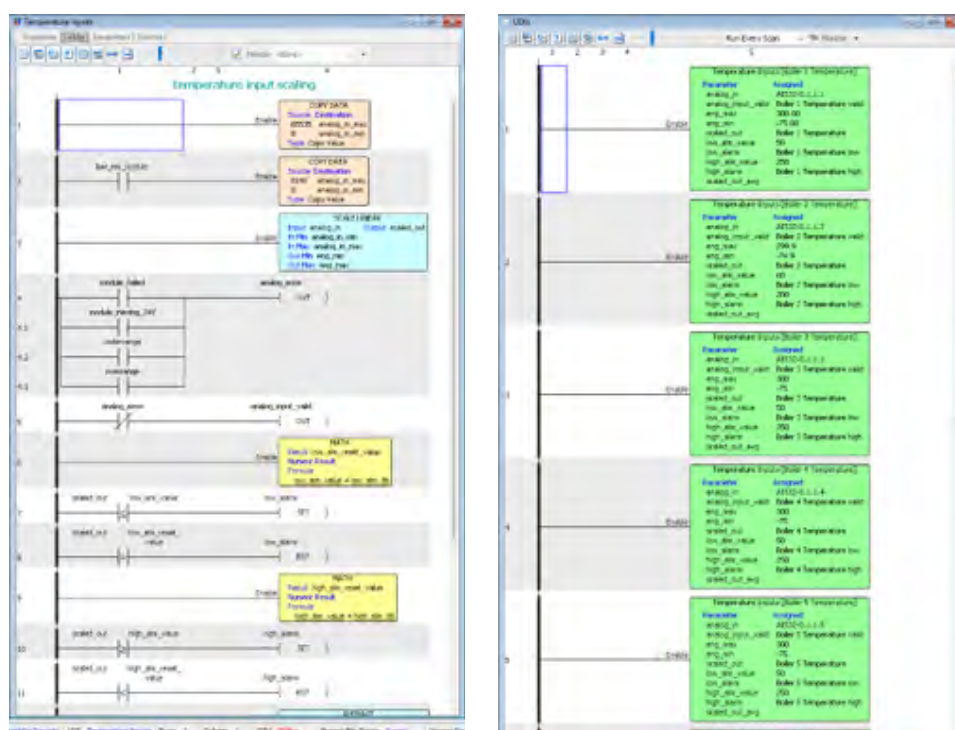
In PLC programs, logical objects typically correspond to the different types of industrial components, such as motors, control valves, sensors, and so on. Usually, there are management provisions so that if a parent object definition is modified it can be deployed throughout the child objects as needed, eliminating the need to manually update every individual instance. AutomationDirect implements OOP for the Productivity Series PLCs, for both data and instructions.

User-Defined Structure (UDS) Data

With UDS, developers can create grouped sets of similar or dissimilar data types (user defined tag) for organization and programming efficiency. This allows grouping PLC tags based on a functional portion of the program, or really on any organizational basis. For example, the logic for a VFD-controlled motor may include binary tags (run command, running status, alarms) and analog tags (speed command, speed status), all as sub-elements of one main tag. This can be a great time saver during the programming phase, and a convenient way to monitor and troubleshoot the program once it is in operation.

User-Defined Instructions (UDIs)

UDIs are custom blocks of logic that can be easily reused throughout a project, with each instance having its own internal memory and input/output tag assignments. Engineers can use these modular, self-contained blocks to build complex programs with minimal logic duplication and improved maintainability. UDI instances appear as compact instruction blocks and can be used on any rung throughout the program, just like built-in instructions supplied with the IDE software. Aside from reducing duplicated logic, UDIs can make programs easier to read by encapsulating lower-level details inside unified instruction blocks, each with clear inputs and outputs. UDIs can even be nested inside of other UDIs as an effective strategy for managing complex, multi-level systems.



Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

OOP, UDS, and UDI Advantages and Drawbacks

When choosing a PLC platform, designers need to consider what languages they need to use and whether OOP will be a benefit for their application. Some IT-based languages will naturally include OOP functionality, but only some OT-based languages running on PLC platforms will support capabilities like UDS and UDI.

Advantages:

- Code is portable and easy to use.
- Code can be more compact and readable.
- Works well with math- and loop-intensive applications, and where there are numerous instances of similar logic and data structures.
- Is familiar to the newest generations of programmers and developers.

Drawbacks:

- Can demand higher CPU overhead, but this is rarely a limiting factor.
- Requires a steeper learning curve, especially for programmers trained in more traditional approaches.
- May not always be as easy to troubleshoot for maintenance personnel because the logic within blocks is not always easily visible or modified.
- Usually has additional constraints which must be followed when programming, especially when managing updates.

OT Programming Language, IEC61131-3

For the first generations of PLCs, each vendor had their own programming language implementations, making it more difficult for developers to work on various brands and to transport code.

To address this issue, IEC 61131-3 (first published in 1993, and most recently updated in 2025) specifies the structures, syntax, and semantics of PLC programming languages as a first step to improving interoperability, code re-use and portability, and overall user efficiency. This standard defines four different languages, three of them graphical and one text-based:

- Ladder Diagram (LD): Also known as ladder logic, this is a graphical programming language modeled after hardwired electrical relay logic, but with additional extensions to perform many other tasks. This eBook includes more detailed information on LD programming in a following section.
- Function Block Diagram (FBD): FBD is another form of graphical programming language, and its function blocks that can be “wired” together in a logic diagram representing signal flows. It can be useful for discrete logic, but FBD is especially well-suited for defining analog control loops, such as the data flow from an input block, through a filter, to a PID block, and to an output block.
- Sequential Function Chart (SFC): SFC is also a graphical programming logic, but it follows a paradigm of steps and transitions, making it very useful for batch processing where actions are performed in order.
- Structured Text (ST): ST is a high-level text-based language, with concise code that looks a lot like BASIC. It is especially good for math operations, if-then-else logic, looping logic, and other data manipulation.

IEC Programming Capable PLCs

As a point of reference, most AutomationDirect PLCs have simply supported LD type programming, which has been the greatest demand from the customer base. However, the LS Electric PLCs and the Productivity CODESYS PLC (see next section) support most if not all the IEC programming languages. Recognizing an increased call for data handling applications, in 2026 AutomationDirect introduced ST programming to the Productivity Series.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

CODESYS

While IEC 61131-3 provided important definition around programming, the fact remained that each PLC vendor's instruction set and data addressing remained unique enough that code portability (developing code for one manufacturer's PLCs and then transferring it to another) simply was not possible.

To address this challenge, a German company called CODESYS Group has developed the CONtrols DEvelopment SYStem (CODESYS) IDE and runtime engine as a manufacturer-independent way to implement industrial automation functionality. Each PLC vendor, and indeed individual users with a computing platform, can incorporate CODESYS. The key benefits to using CODESYS are:

- Compliance with IEC 61131-3 programming languages.
- Hardware independence; code created using CODESYS for one hardware brand/platform can be deployed to another brand/platform that also supports CODESYS.
- Support for OOP concepts.
- Integrated support for major industrial communications protocols and fieldbuses.

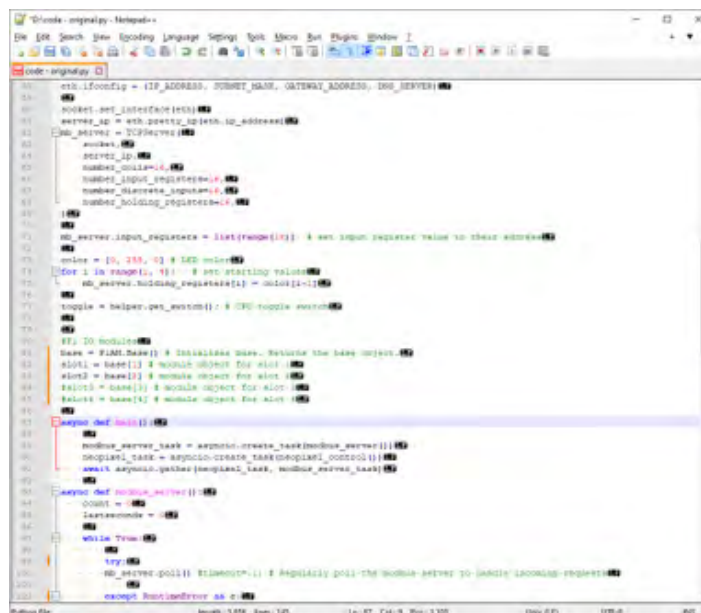
ProductivityCODESYS

To support users who prefer to develop in CODESYS, AutomationDirect offers the ProductivityCODESYS controller, which works with the proven Productivity2000 platform of I/O and associated components.

IT Programming Languages

As we have seen, the modern PLC has gained significant capabilities over the years. However, there are some applications where end users have a need to use modern and open IT-based programming languages, such as C++ and CircuitPython, or to use tools such as Node-RED, or data exchange standards such as OPC UA.

Modern text-based programming languages are better adapted for some types of computing—especially data handling and analysis—and they are also more familiar to many young developers entering the work force. Open languages often mean a reduction of software and licensing fees for end-users, and other efficiencies as designers scale and adapt their work. When developers have access to more communications protocols and standardized computing options, they have more freedom to create valuable solutions.



ProductivityOpen

AutomationDirect offers the ProductivityOpen controller, designed to integrate with the proven Productivity1000 platform of I/O and other components. The brains of the product are a MKRZero Arduino-compatible microcontroller, built into a rugged housing and certified to meet UL and other standards. It can accept some open-source and do-it-yourself shields, but it really shines by offering industrialized power supply, wired input/output modules, Ethernet, and serial communications shields. With ProductivityOpen, end users can develop and deploy C++ and CircuitPython automation and data handling programs for industrial-grade applications.



CLICK PLUS Node-RED

Node-RED is an open-source visual programming tool for connecting physical hardware devices, APIs, and online services together as part of the IoT. Node-RED's low-code, drag-and-drop interface makes it easy to create complex data flows without writing code. A huge assortment of information and templates are available online at <https://flows.nodered.org/>, where there are many nodes available for various tasks, including input/output, data processing, logic, and social media functions; see also: <https://library.automationdirect.com/get-ahead-with-node-red/>.

AutomationDirect's CLICK PLUS C2-NRED module is their answer to the growing demand for edge intelligence in industrial automation. This Node-RED intelligent module plugs directly into the CLICK PLUS PLC backplane, giving machine builders and OEMs a purpose-built edge computing environment running alongside—not instead of, or adding additional burden to—their deterministic PLC logic. It runs a full Node-RED runtime on embedded Linux, providing real-time data flows, cloud connectivity, protocol bridging, and lightweight analytics, all within the familiar CLICK PLUS form factor.

What makes the C2-NRED compelling is that it doesn't ask users to abandon their existing PLC investment. The module communicates directly over the CLICK PLUS backplane to exchange data with the CPU, while simultaneously connecting to enterprise systems,

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

MQTT brokers, or cloud platforms via its onboard Ethernet port. This integrated approach lets small and mid-sized machine builders add genuine edge capability—dashboards, alerting, condition monitoring, OEE calculations—without the cost and complexity of a separate edge gateway or IPC. Because the Node-RED module is physically separate from the runtime module, cybersecurity is simpler to implement.



CLICK PLUS OPC UA

OPC UA was already covered in the digital communications chapter of this eBook. As an open-source standard supporting multiple platforms, protocols, and architectures, OPC UA is a seamless and scalable way, with built-in cybersecurity, to implement contextual communications between devices.

The C2-OPCUA module for the CLICK PLUS PLC is AutomationDirect's answer to edge-level IT/OT convergence — purpose-built, industrial-grade, and priced to make OPC UA connectivity accessible to virtually any application. Simply inserted into the CLICK PLUS option slot, the module transforms the CLICK PLUS into a full OPC UA v1.04 server, securely exposing all I/O and internal registers to SCADA, MES, cloud platforms, and analytics tools—all without any external gateway hardware. For machine builders and system integrators connecting plant-floor equipment to platforms like Microsoft Azure IoT, AWS IoT, Ignition SCADA, or Siemens WinCC, the CLICK PLUS with OPC UA module is a compelling and cost-effective edge solution.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Ladder Logic Programming

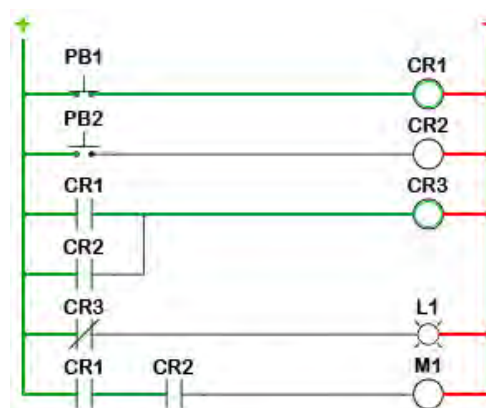
Despite many hardware and software advances, ladder logic (also known as ladder diagram) continues to be a very popular PLC programming language, even as many modern PLCs incorporate additional programming languages. Of course, this assertion varies depending on what brand one is working with, or what world region is being considered. But the fact remains that ladder logic endures for many reasons:

- Electricians, often tasked with working on PLCs, appreciate the similarity of ladder logic to electrical diagrams.
- Engineers and students can follow the logical nature of the language.
- A massive installed base of ladder logic programming exists throughout all industry sectors.
- Its graphical nature is easier to troubleshoot than text-based alternatives.

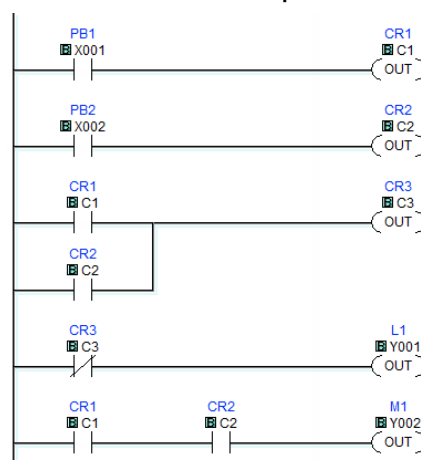
This chapter is aimed at helping the reader understand ladder logic programming basics.

Structure

Standard electrical ladder diagrams document how connections are made between devices in relay panels. They are called “ladder” diagrams because they look like a ladder, with a pair of vertical side rails and many horizontal rungs. The positive power rail (on the left) flows to the negative power rail (on the right) through the physical devices connected on the rung. Inputs are toward the left side and outputs are on the right side. In this example, “PB” stands for “pushbutton”, “CR” stands for “control relay”, “L1” is a light and “M1” would be a motor.



Ladder logic was designed to resemble electrical ladder diagrams, but its contacts, coils, and other instructions operate on PLC addresses or tags rather than directly representing hard-wired components. The equivalent ladder logic program shown below provides the same basic control as the electrical ladder diagram above. Some addresses or tags correspond to physical inputs and outputs, while others store internal program states or data. To monitor and control real-world devices, the program’s addresses or tags correspond to physical points on the PLC’s I/O modules, and the field devices are wired to those I/O points.



For the PLC in this example, physical inputs and outputs use fixed X and Y addresses. Input address X001, given the nickname PB1, corresponds to the input point connected to the pushbutton. Output address Y001 corresponds to the output point connected to light L1. The program evaluates X001 to determine the input state and controls Y001 to turn the light ON or OFF.

Other addresses, such as CR1 and CR3, represent internal bits. These bits store and communicate program conditions without being directly connected to physical input or output points.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Logic Instructions, Physical Devices, and Fail-Safe Design

The normally open contact instruction assigned to X001 evaluates the state of that address; it does not identify the construction or wiring of the physical pushbutton.

A normally open contact instruction is true when its assigned address is ON and false when the address is OFF. A normally closed contact instruction performs the opposite evaluation: it is true when its assigned address is OFF and false when the address is ON.

Therefore, the ladder logic contact symbol describes how the instruction evaluates an address—not whether the associated field device has normally open or normally closed physical contacts. Physical wiring and program logic must be considered both individually and together when designing or troubleshooting a control system.

A subtle, but important design consideration related to this topic concerns fail-safe design for important applications. The type of physical field device contacts and signals, in conjunction with the associated PLC logic, should be configured so that if there is a failure then the overall system operates in the safest way possible. A failure could be a wire coming loose, power loss, or any number of other conditions. For example, a level switch—whether high or low level—should be configured with a normally open contact that is closed when the level switch is energized and the level is normal. If there is any problem, the PLC will see a lack of input signal and will be programmed to act accordingly.

Ladder Logic Execution

A PLC repeatedly executes its control program in a sequence known as the scan cycle. Ladder logic is generally evaluated from left to right within each rung and from top to bottom through the program.

Beginning with the first rung, the CPU evaluates each instruction according to the current value of its assigned address. It then determines the result of the rung and executes the appropriate output instruction before continuing to the next rung.

Assume the pushbutton in the preceding example is wired so that pressing it turns X001 ON. When the pushbutton is pressed, the normally open PB1 instruction becomes true. The CPU then executes the output instruction for CR1, causing the CR1 internal bit to turn ON.

Because rung 3 is evaluated later in the program, the CR1 instruction on that rung sees the newly updated state. This causes CR3 to turn ON. Rung 4 contains a normally closed contact instruction assigned to CR3. When CR3 is ON, that instruction evaluates as false, interrupting logical continuity to the L1 output instruction and turning L1 OFF.

Although ladder instructions are commonly called contacts and coils, they do not act as physical relay components. Contact instructions evaluate PLC addresses, while output instructions write values to PLC addresses.

After the CPU reaches the end of the program, it begins another scan and evaluates the logic again. This process continues as long as the PLC is powered and operating in RUN mode.

The time required to complete one scan cycle is called scan time. The exact scan sequence varies among PLCs, but a typical cycle includes:

1. Reading the states of physical inputs
2. Executing the control program
3. Updating physical outputs
4. Performing communications, diagnostics, and other background tasks

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Scan time is important in applications that require fast or precisely timed responses. A change at an input may not affect an output until the PLC detects the change, executes the applicable logic, and updates the output.

Many PLCs provide specialized features for time-critical operations, including interrupt routines, high-speed inputs and outputs, immediate I/O instructions, and dedicated motion or counter modules. Organizing a program so that unnecessary logic is not executed during every scan can also help manage program-execution time, although the available methods and their effects depend on the PLC platform.

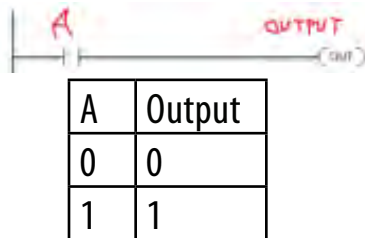
Functions

Ladder logic was first conceived to perform basic Boolean operations, but through a variety of function blocks it can do much more. Here is a quick summary of basic functionality.

- **Boolean Logic:** Evaluating ON/OFF input values with basic operators such as AND, OR, and NOT to drive ON/OFF output values.
- **Timing:** Timer instructions measure elapsed time and support functions such as delaying an output after a condition becomes true or maintaining an output for a specified period. Exact timer behavior depends on the instruction type.
- **Counting:** Count-up and count-down functions increase or decrease the counter value on every transition of the input.
- **Comparisons:** Compare instructions are available to determine whether one numeric value is equal to, not equal to, greater than, greater than or equal to, less than, or less than or equal to another value.
- **Math:** These instructions not only allow for simple addition and subtraction, but also for more complex operations, like tangents, square roots, and more.
- **Special functions:** Depending on the PLC, available instructions may include PID control, communications, shift registers, drum sequencers, data handling, ramp generation, and motion-control functions.

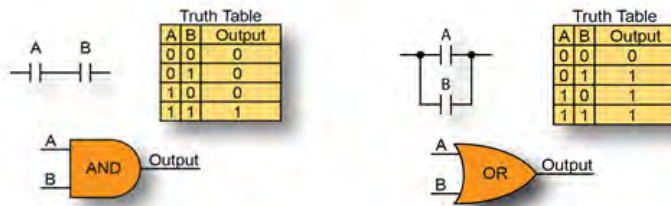
Basic Boolean Logic

A very simple ladder logic rung and truth table would have one input and one output.



This could also be interpreted as IF A THEN OUTPUT.

A slightly more complicated example uses the two logic gates most commonly represented in ladder logic, the AND and OR gates. The diagrams below show ladder logic, Boolean logic, and truth tables that are equivalent for (A AND B), and for (A OR B).



As the truth tables below show, the AND gate needs all inputs to be true before the output becomes true, whereas the OR gate needs at least one input to be true to make the output true. These truth tables can be duplicated with relay contact logic by connecting normally open contacts in series (AND gate) or parallel (OR gate).

To equate this to a control circuit, consider that Input A and Input B (often referred to as permissive bits) could be the Door Open and Motion Detected inputs of a home alarm system. If the contacts are arranged in series to form AND logic, then both conditions must be met before the alarm is activated, so the door will have to be opened and the motion detector triggered before the alarm is triggered. If the instructions are arranged in parallel to form OR logic, then either condition can activate the alarm. In this case, the door opening or the motion detection will trigger the alarm. The option chosen is dependent on the application and how the system is expected to perform.

Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

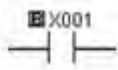
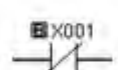
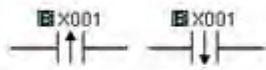
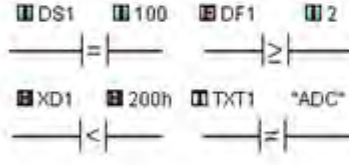
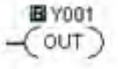
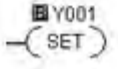
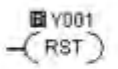
Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

Basic PLC Instructions

As the smallest and simplest (yet still extremely capable!) PLC in AutomationDirect's portfolio, the CLICK PLC instruction set shown below provides a comprehensive overview of its fundamental ladder logic functionality.

CLICK PLC Instruction Set			
Ladder Symbol	Title	Type	Description
	Normally Open Contact	Bit Instruction	The Normally Open Contact mimics the behavior of a physical contact and changes in response to the status of a Bit Memory Address. The Normally Open Contact is ON when the related bit is ON.
	Normally Closed Contact	Bit Instruction	The Normally Closed Contact mimics the behavior of a physical contact and changes in response to the status of a Bit Memory Address. The Normally Closed Contact is ON when the related bit is OFF.
	Edge Contact	Bit Instruction	The Edge Contact turns ON when the related bit transitions from OFF to ON (Rising Edge) or ON to OFF (Falling Edge).
	Compare Contact	Word Instruction	The Compare instruction uses a Mathematical Operator as a basis for comparison of two data values. When the data values satisfy the selected mathematical relationship (>, <, =, etc.) the Compare Contact turns ON.
	Out Coil	Bit Instruction	An Out instruction turns ON its associated Bit Memory when the status of the rung is true. The Out instruction turns OFF its associated Bit Memory when the status of the rung is false.
	Set Coil	Bit Instruction	The Set instruction turns ON the associated Bit Memory when the status of the rung is true. The Bit Memory stays on after the rung becomes false.
	Reset Coil	Bit Instruction	The Reset instruction turns OFF the associated bit memory when the status of the rung is true. The Bit Memory stays OFF after the rung becomes false.

Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

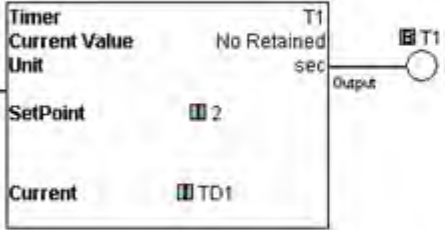
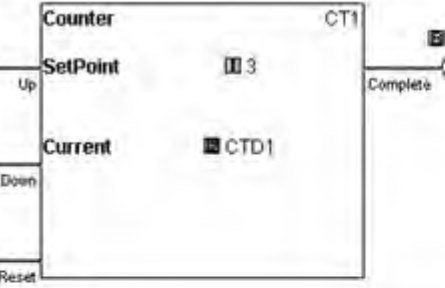
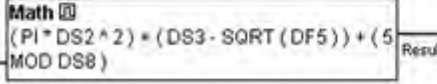
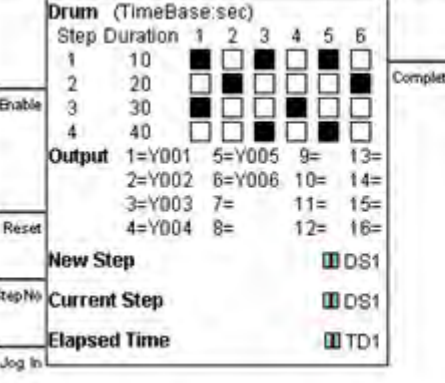
Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

	Timer	Word Instruction	An ON Delay Timer measures the time duration that begins with a transition of the enable rung from OFF to ON. Beyond this transition point, the Timer increases the Current Value; when it reaches the SetPoint, the Timer Bit is turned ON. An OFF Delay Timer measures the time duration that begins with a transition of the enable rung from ON to OFF. Beyond this transition point, the Timer increases the Current Value; when it reaches the SetPoint, the Timer Bit is turned OFF.
	Counter	Word Instruction	When enabled, a Counter instruction counts up or down (depending on user settings) until it reaches the SetPoint. The Counter counts in response to the transition from OFF to ON of the enabling rung (up or down rung).
	Math	Word Instruction	The Math instruction solves a user-defined formula during the execution of the Ladder Program. Once the enable rung transitions from OFF to ON the formula will be solved and the result will be stored in the data format and location selected for the Result.
	Drum Instruction	Special Instruction	The Drum instruction simulates an electromechanical drum sequencer, using either a Time Based or an Event Base sequencing strategy. Each Drum instruction is capable of sequencing through 1 to 16 steps and turning ON as many as 16 outputs in a user defined pattern. Outputs can be either physical outputs or internal control relays.

Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

	Shift Register	Bit Instruction	The Shift Register instruction shifts a range of control bits one memory location with each OFF-to-ON transition of the Clock pulse. If the Starting Address is lower than the Ending Address, the Shift register will Shift from the Starting Address to the Ending Address. If the Ending Address is lower than Starting Address then Shift Register will Shift from Ending Address to the Starting Address.
	Copy Instruction	File Instruction	The Copy instruction is used to copy a data or text value from its Source location to a specified Destination register.
	Search Instruction	File Instruction	The Search instruction is used to search for a data value that meets the specified condition and that is located within a specified range of data registers.
	Call Instruction	Program Control	The Call instruction is required to call or execute a Subroutine Program from the Main Program.
	For Instruction	Program Control	The For instruction indicates the starting point of a For-Next loop, and based on its user setting, determines how many times the For-Next loop will be executed in one program scan. Between the For instruction rung and the Next instruction rung, place the rungs of logic that should be repeated multiple times.
	Next Instruction	Program Control	The Next instruction indicates the end of a For-Next loop.
	End Instruction	Program Control	The End instruction marks the termination point of the normal program scan. The End instruction tells the CPU that there are no more rungs to be processed.
	Receive Instruction	Communication	The Receive instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and read data from external devices.
	Send Instruction	Communication	The Send instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and write data to external devices.

Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

Some instructions, like Timer, mimic simple real-world electrical components, like a timing relay. The Drum Instruction is a more complicated example, as it simulates an electromechanical drum sequencer which can move through multiple steps and turn numerous outputs ON/OFF according to a user-defined pattern.

Other instructions—particularly the File, Program Control, and Communication instructions—have no real equivalency with physical wired devices, but are essential for carrying out advanced PLC functionality.

The Compare contact is a great example of a frequently used instruction for evaluating whether two values are:

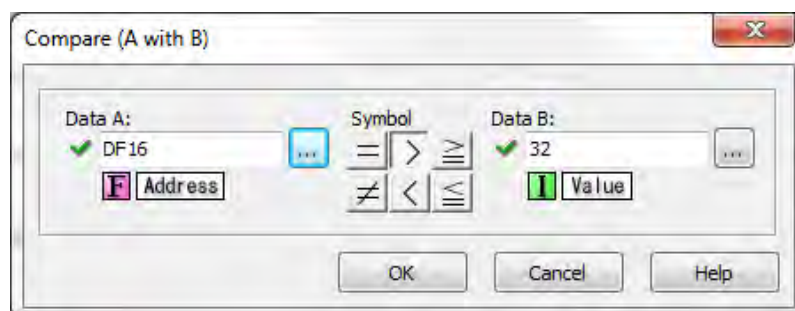
- Equal
- Not Equal
- One Greater Than the other
- One Greater Than or Equal To the other
- One Less Than the other
- One Less Than or Equal To the other



If the evaluation condition is met, then the result will be true, passing the logic path along to the next instruction or turning the rung output ON.

How can the Compare contact be used in an application? Imagine having a freezer with a sensor installed inside to monitor the temperature. The temperature sensor's output produces an analog signal that is wired into a PLC. The PLC is programmed to take the analog signal and convert it into degrees Fahrenheit. If the temperature indicates greater than 32 degrees, an alarm horn must sound to alert that the temperature is too high.

Using the Compare contact instruction dialog box as seen below, the actual Freezer Temperature (stored in memory location DF16) is evaluated against the constant 32, which represents 32 degrees Fahrenheit. The Compare contact dialog box allows selecting any of the six different comparison methods. For this example, Greater Than is required so that if the Freezer Temperature ever becomes Greater Than the number 32, the Compare contact will become true.



An Out coil is then programmed from the Compare contact, and will provide the signal that is wired to the external Alarm Horn, which alerts users to a problem with the freezer, and hopefully provides enough warning to avoid disaster! Of course, there are many other ways to improve this scheme such as with an application-appropriate threshold, time delay, hysteresis, sensor-fault detection, and alarm acknowledgment. All of these features are easily added in a PLC, but would be far more difficult to incorporate using hard-wiring.

Free Downloads and Learning Resources

If this section has been unclear in any way, not to worry! Because AutomationDirect programming software is free to download and use, it provides a great way to learn about PLC programming and ladder logic. [Beginners should download the CLICK and CLICK PLUS programming software to start.](#) [Then, check out the free online PLC training from AutomationDirect to get some great hands-on experience!](#)

Jump to Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

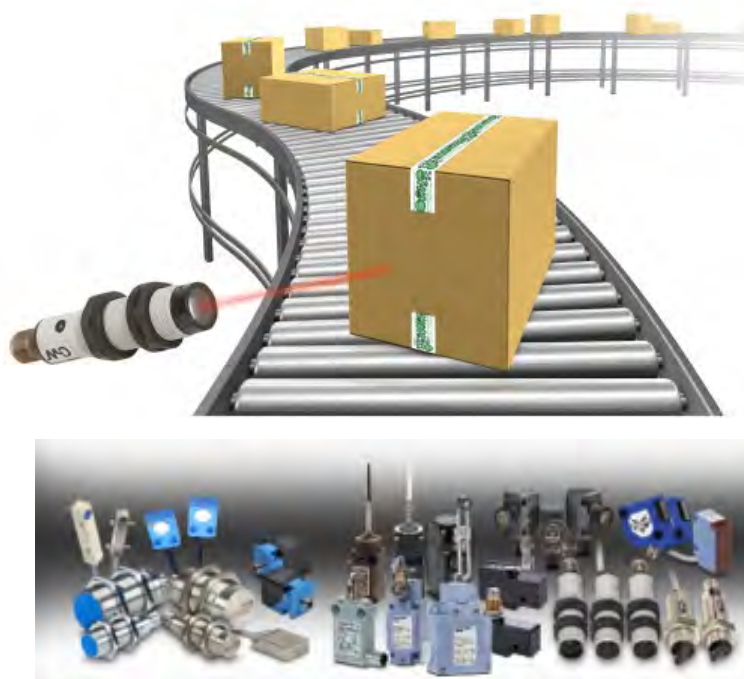
Practical PLC Topics

Preceding chapters have focused on the many details involved with selecting a PLC platform. This chapter pivots to look at the implementation of practical PLC applications.

Discrete Sensing

While a PLC could be implemented using only outputs, it is very common for almost every application to have at least some discrete sensing capabilities. Discrete sensing is used to monitor the presence (or absence) of something—perhaps an action, equipment position, or an object—so the logic can take further action or confirm an action was completed. A few examples of discrete sensing include:

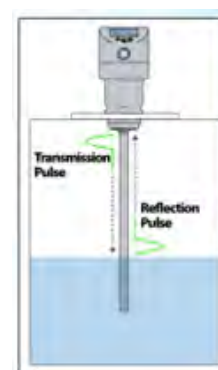
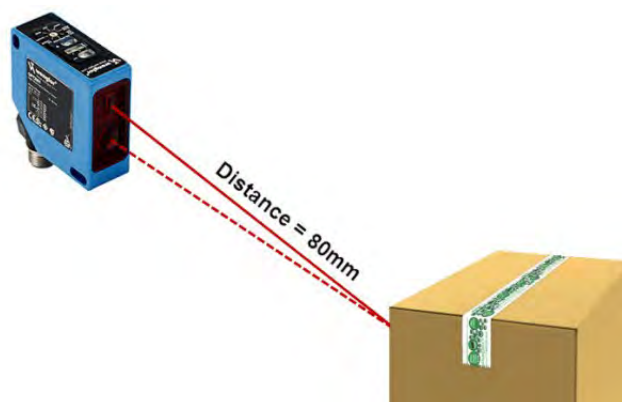
- Detecting actuation of a pushbutton or selector switch (this can be from a physical device I/O-connected field device, or from an HMI).
- Monitoring a motor starter or equipment “running” contact.
- Sensing if an object is present using a mechanical limit switch (physical contact), a proximity switch (inductive for metal, capacitive for other materials, ultrasonic using reflected sound), a photoelectric sensor (diffuse, reflective, or through-beam), or other sensor type.
- Determining if the flow, level, pressure, temperature or some other characteristic has risen or fallen to a certain value.



Analog Sensing

In many cases, it is necessary for the PLC logic to access an analog value—something within the range of an expected minimum and maximum—to perform the desired task. This could be for a complex processing application, or it could be as simple as filling a tank to a desired level. Just a few examples of analog sensing include:

- Detecting a potentiometer signal or using an analog data entry entered by a user at an HMI.
- Monitoring the speed at which a VFD is operating a motor, whether from the VFD itself or from a tachometer.
- Sensing the distance from an object to the sensing device using an ultrasonic or laser sensor.
- Determining the actual value of flow, level, pressure, temperature or some other characteristic. There are many types of instrumentation technologies for these types of measurements, depending on the application details.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Feedback: Open-loop Versus Closed-loop Control

PLCs can control machinery and equipment by commanding outputs—primarily via discrete and analog signals, but also by using digital communications—to operate valves, motors, and other devices.

When a PLC program commands outputs based strictly on time or a sequence, without regard to sensing information, it is called open-loop control. For simple systems in reliable environments, open-loop control can run well, even though the logic is effectively operating blindly. However, if there are any failures along the way—such as if the equipment breaks, a wire falls off, a circuit breaker trips, air pressure goes low, or any number of other issues—the program will not detect the issue and handle it correctly, so undesirable behavior is likely to result. Open-loop systems save money initially but will almost always exhibit some inefficiency and non-repeatability, resulting in a higher total cost of ownership.

On the other hand, improved performance can be achieved by adding discrete and analog sensing as feedback to the PLC, which can then be programmed using closed-loop control. With closed-loop control, the PLC logic and algorithms monitor important input signals and can adjust the output commands as needed for correct overall operation. For example, if a valve is commanded to open and extend a pneumatic cylinder, the cylinder operation can be confirmed by a limit switch, at which point the PLC can perform the next step.

Feedback and closed-loop control are preferred whenever possible because they:

- Allow the PLC program to perform and verify the action, reducing human involvement and error.
- Improve/reduce cycle times of sequential operations.
- Prevent mechanical interference (crashes) between moving parts.
- Enable automatic corrective responses to dynamically changing conditions.
- Improve process stability, repeatability, and reliability.
- Standardize machine operation and process control.
- Can reduce the total cost of ownership.

Closed-loop control is not only for discrete on-off machine control logic; it is also applicable to analog process control, and hybrid combinations of discrete/analog.

When developing a closed-loop automation system, designers will need to ensure that the right kind of feedback signals are included. Sensors will need to monitor the important physical points, and analog sensors in particular will need to have the appropriate measurement range and accuracy. A degree of redundancy and fail-safe operation is important also. For instance, if an individual level transmitter stops working properly then fail-safe operation might shut down the system; but if a secondary sensor is installed redundantly it would be possible to keep the system running safely.

PID Control

Proportional-integral-derivative (PID) control is a specific form of closed-loop control with feedback used for analog processes, such as maintaining the level of fluid in a vessel. Although there have been many industrial automation computing advances over the years, traditional PID control remains the leading workhorse for closed-loop analog process automation.

The measured input could be one of the “big four”—flow, level, pressure, or temperature—or another characteristic, such as an analytical value, or even the position of a mechanism. The commanded output is commonly an analog device such as a modulating valve or a variable speed pump. However, sometimes the output is implemented using a discrete device, like an open/close valve or an on/off heating circuit. In this case, a pulse width modulation function, time-proportioned control logic, or some other similar capability will be needed so that the discrete output operation behaves much like an analog function.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Despite the elegant mathematics behind the scenes, PID loops are easily implemented with modern PLC technology. A PID control loop involves the following analog variables:

- Process variable (PV): The characteristic measured by the PID equation.
- Controlled output (CO): The output signal as commanded by the PID equation.
- Setpoint (SP): The desired value for the PV to reach and remain at.

The PV and SP are usually expressed in engineering units, such as PSI or DegF, but it is also possible for these to be represented in %, such as for a tank level. The CO is often expressed in %, such as 0 to 100% for a valve opening or pump speed, but it can also be in other engineering units. It is important for the user to carefully assign and scale the associated real-world signal values, which may interact with the controller via wired 4-20mA signals or communicate over a serial or network connection, and then apply them with reasonable limits.

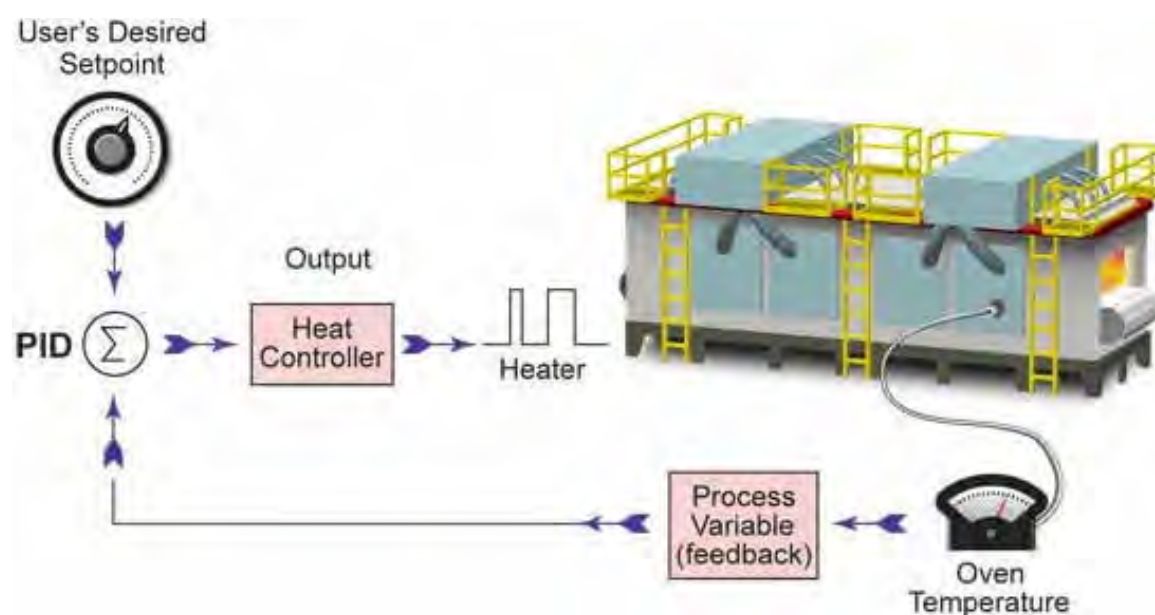
There are several forms of the PID equation itself, and it is beyond the scope of this document to describe the equation in detail, but users typically need to apply the following tuning parameters, or terms:

- Proportional (P): Also known as gain, associated with the scale of corrective action based on the error (or the difference between PV and SP).
- Integral (I): Also known as reset, associated with the scale of corrective action based on the integral sum of errors over time.
- Derivative (D): Also known as rate, associated with the scale of corrective action based on the rate of change of error.

Simplistically, the P term acts on the immediate error, the I term acts increasingly as error persists over time, and the D term acts to anticipate error as it changes. Many applications run fine with only PI terms, and some work adequately with just a P term. More difficult applications, notably temperature control, often require all three PID terms.

It is possible to manually tune a PID equation, but it is difficult to tune all three terms at once, and users need to ensure the process continues to operate safely during the tuning. A better approach is to use automatic tuning—also called autotune—if available in the PLC or separate software, to reduce or eliminate the trial and error of manual tuning.

Integrated autotuning is included in several AutomationDirect PLC platforms. A typical autotune cycle controls the CO, sometimes in a step wave format, and measures the rate of change, overshoot, and response time of the PV to calculate the ideal PID terms. Sometimes a little further manual adjustment is needed, but in most cases a built-in autotune capability quickly provides workable parameters, saving time and effort.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Motion Control

In an industrial automation context, the term motion control usually refers to using one or more electric motors—either servos or steppers—to positively drive the position, velocity, and acceleration of a physical system. There are other motion control methods as well—pneumatics are economical, and hydraulics can provide the greatest force—but electric motors often provide the right balance of accuracy, speed, power, durability, and operating cost.

A typical motion control system includes:

- One or more motors
- A drive for each motor to power it
- A motion controller to operate one or more drives in coordination

Motion control once required specialized equipment, a great deal of hardwiring, and deep expertise, along with significant cost and complexity, to achieve a successful deployment. In recent years, the field of industrial motion control has undergone a dramatic transformation on several fronts. Motors, drives, and controllers have decreased in cost, gained performance, and become easier to use.

Discrete signals or pulse and direction signals were the original and most basic way of connecting motion controllers with drives, and this solution is still a workable and cost effective approach for basic applications. The development of a standardized serial real-time communication system (SERCOS) interface, released in the early 1990s and improved thereafter, provided a digital connectivity option promising better overall performance. Today, there are Ethernet-based options in the form of EtherNet/IP which provides acceptable responsiveness for straightforward control applications, and the Ethernet for Control Automation Technology (EtherCAT) protocol for the most demanding applications where deterministic cycle control and the best synchronization among devices is required. These digital motion control fieldbuses simplify wiring, and they provide a wealth of configuration and diagnostic data transfer benefits.

Traditionally, motion control applications have often been designed as a subsystem with a dedicated motion controller, which would then be integrated with a PLC or other host to provide a complete industrial automation solution. However, many modern PLCs incorporate native motion control capabilities and can take on the role of motion controller, while also performing related automation. PLCs can control motion using hardwired signals or digital communications.

When an application requires motion control, designers are likely to find that a PLC platform, empowered with wizard-like options, pre-built libraries, and graphical interfaces for easily applying advanced strategies, can provide an integrated motion and industrial automation solution. Here are a few options:

- Basic applications with just a few axes of motion—usually three axes or fewer—and fundamental moves at moderate speeds. The CLICK PLUS stackable micro modular PLC provides this essential motion control functionality at an unprecedented low cost.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

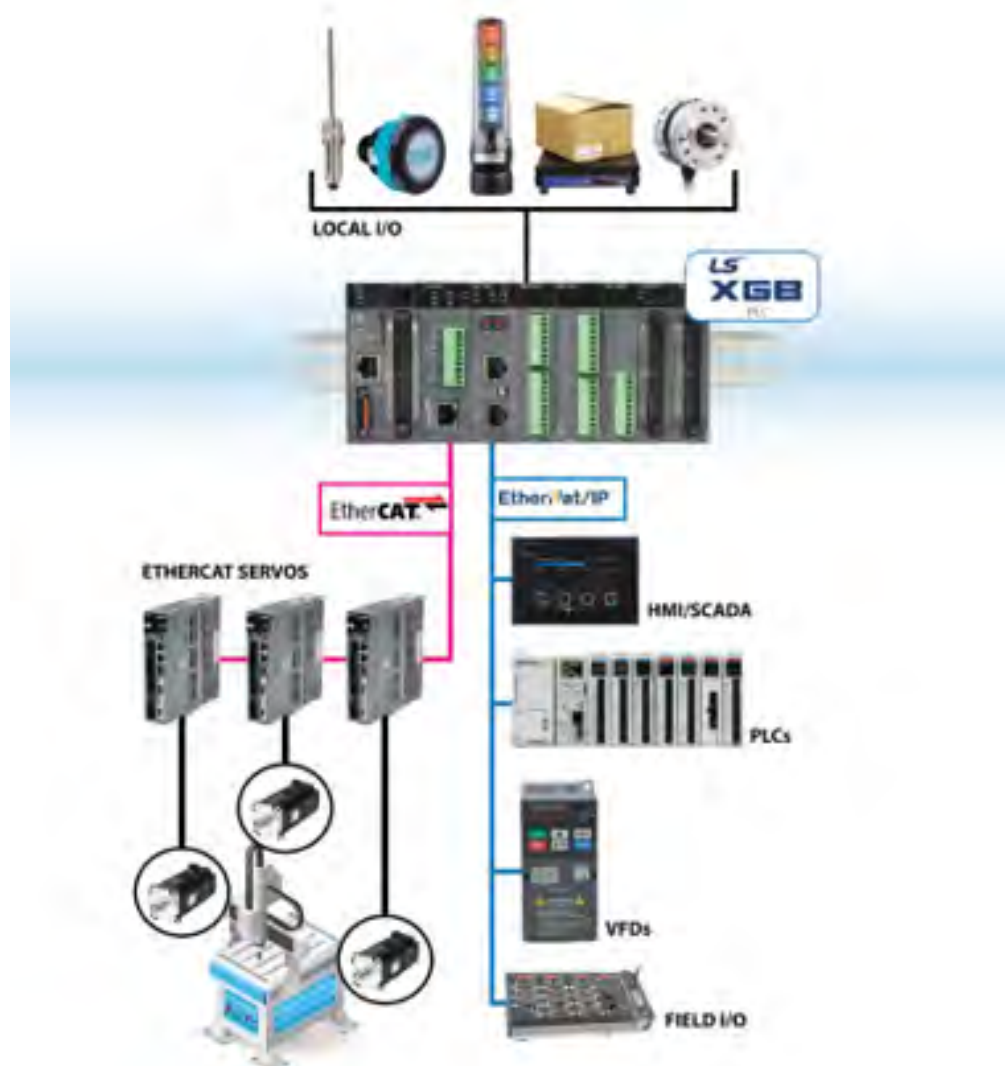
Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

- Mid-tier applications require more extensive motion command sets and larger axis counts. The AutomationDirect Productivity line and BRX stackable micro brick series PLCs can operate up to 12 or more axes.



- High-performance applications operate at rapid speeds and use sophisticated motion instructions beyond what a typical PLC can provide. The LS Electric XGB PLC with EtherCAT connectivity provides industry-leading performance, flexibility, and cost advantages for even the most demanding applications up to 16 axes.



Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

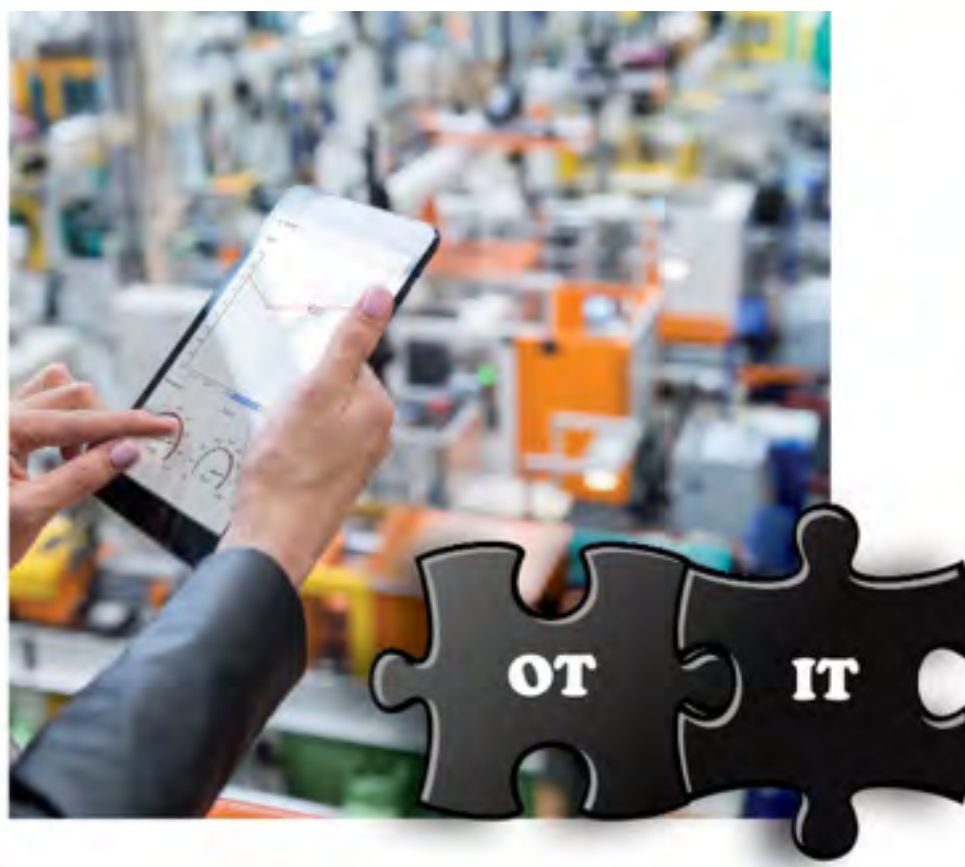
Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Security and Cybersecurity

For many years, the industrial sector was “air-gapped” using localized automation systems that were isolated from the outside world. When digital communications were used, they often were proprietary types. This approach provided a level of inherent security from prying eyes and those wishing to do harm. But with the acceptance and proliferation of Ethernet in the industrial world, combined with internet connectivity and a drive to increase IoT data access, industry is more vulnerable than ever. It’s imperative in today’s connected world for industrial facilities to secure their automated systems and guard them from possible cyber-attacks.



For industrial automation users, the cybersecurity struggle is compounded by the variety and vintages of devices and technologies present in their operations, and often in service for many years before cybersecurity was a serious consideration. The volume of cyberattacks continue to increase and the threats become more sophisticated, and both offensive and defensive technologies evolve. Furthermore, standards and regulations continue to be updated.

There are several cybersecurity standards, but two of the most relevant for industrial OT users are ISA/IEC 62443 “Security for industrial automation and control systems,” and ISO/IEC 27001 “Information security, cybersecurity and privacy protection — Information security management systems — Requirements.” The United States Cybersecurity & Infrastructure Security Agency (CISA) develops services, tools, and programs aimed at improving the cyber resilience of technology. More recently, at the end of 2024, the European Union Cyber Resilience Act (CRA) regulation has come into force and will impact industrial automation and control products and systems going to or from Europe, but it is too early to say exactly how.

Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

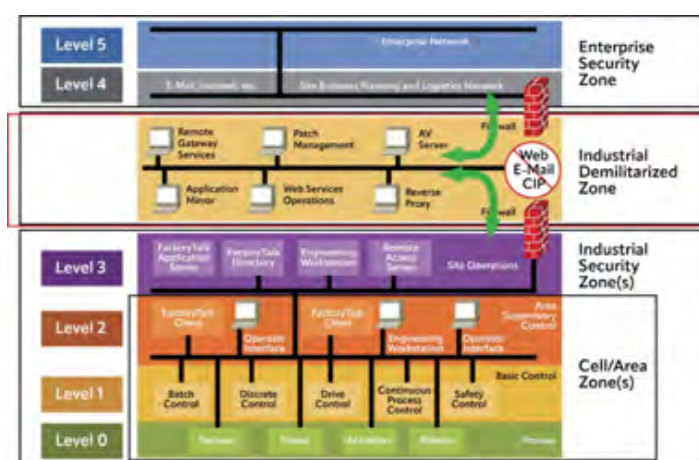
Many end users and designers may not be staffed to understand the intricacies of cybersecurity standards, and the full impact of merging OT and IT systems, but they can look for suppliers and products certified as compliant with these standards. Various AutomationDirect PLC platforms include provisions to help end users achieve their cybersecurity goals, including.

- Passwords and role-based user accounts: These efficiently limit who has access to the controller, and what each user can and cannot do. A tracking log may be available.
- Guest memory: The BRX PLC platform reserves a group of internal memory blocks specifically for external communication. These blocks are isolated, preventing external devices such as OITs and HMIs from being able to directly control the PLC's I/O and operational memory.
- Session-based communication: The BRX programming IDE establishes communication sessions with the PLC using a unique ID, ensuring that only the authorized user gains access. A timeout system will terminate a session after a period of inactivity.
- Operating system write protect: The BRX includes on-board hardware DIP switches so users can enable/disable firmware downloads to the controller, protecting the PLC from unauthorized changes.



Establishing robust physical security of PLC installations is always important because cyberattacks can be perpetrated via in-person access to USB ports and other network connections. However, some of the most insidious assaults happen remotely from internet or wireless network connections. End users supporting existing automation or designing new systems can follow some ever-useful guidelines.

For example, for OT systems it remains useful to preserve a degree of network hierarchy and segmentation, much like the classic Purdue Model. Although modern systems can be architected into much flatter and converged arrangements, a more segmented design restricts unexpected access by enabling just associated devices to interact. For example, field devices connect with their associated PLC, the PLCs connect with related HMIs, and PLC/HMIs connect with higher-level enterprise computing assets.



AutomationDirect offers a complete line of StrideLinx networking products and services to help end users improve their industrial automation cybersecurity posture.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

AutomationDirect PLC Platform Solutions

AutomationDirect offers several PLC families, because we strive to provide affordable solutions for anyone who needs a practical control system, regardless of the application. We didn't invent the PLC ... we just made them affordable! While our PLC platforms share many common programming concepts, they differ in performance, I/O capacity, communications, programming environment, and intended use. The following overview highlights the primary characteristics for each AutomationDirect PLC product series:

- **CLICK Stackable Micro Brick**
Very low cost, with a basic memory structure, but suitable for relay replacement and other small applications.
- **CLICK PLUS Stackable Micro Modular**
Offers a streamlined approach for simple control applications, with an easy learning curve. Combines the simplicity of the original CLICK, with advanced communications features (MQTT, OPC UA, and Node-RED), motion control, and more.
- **Do-More BRX Stackable Micro Brick**
A high-performance modern PLC controller, with the biggest instruction set available from AutomationDirect, remote I/O, a REST API, OPC UA, MQTT with SparkPlug B, and motion control. Uses a memory structure with nicknames.
- **Productivity Series**
The entire Productivity PLC family provides advanced all-purpose control, with exceptional versatility and fully tagname-based addressing. The three families (1000, 2000, and 3000) offer varying form factors and I/O count capacities.
 - **Productivity1000 Stackable Micro and Brick**
Available in a standard (stackable micro) form factor supporting up to 240 local and 512 remote expansion I/O, or a mini (brick) configuration with built-in I/O and supporting up to 80 local and 512 remote expansion I/O.
 - **Productivity2000 Micro Modular**
Uses bases and hot-swappable I/O modules to support up to 4,000 I/O points, depending on module mix, and has more communication ports.
 - **Productivity3000 Modular**
Uses slightly larger bases and hot-swappable I/O modules with built-in LCD displays to support up to 7,300 I/O points, depending on module mix, with no restrictions on module location or power budget. The Productivity3000 PLC provides the expansive I/O, robust communication, integrated data storage, and processing power needed for complex automation systems.
- **ProductivityOpen (Arduino Compatible) Stackable Micro**
This hybrid product family provides an Arduino-compatible programming experience, combining the benefits of an open-source platform with the reliability of an industrial controller and the capability of proven Productivity1000 I/O. Popular with specialty developers and the maker community as a robust platform enabling the use of IT-based programming languages such as C++ and Circuit Python.
- **ProductivityCODESYS Micro Modular**
This controller uses the CODESYS development system, in compliance with the manufacturer-independent IEC 61131-3 standard, and works with proven Productivity2000 I/O. One of the most economical ways for developers to access the CODESYS ecosystem. This product is mostly intended for logic implementations, but is available in a dual-quad CPU version to support additional functions and graphics implementations.
- **LS Electric Stackable Micro and Brick XGB PLC, and XMC Motion Controller**
These controllers combine PLC functionality with extensive motion control capabilities and support for IEC programming languages.
- **DURApulse GS30 and GS4 AC VFDs**
These drives contain a built-in PLC, perfect as an all-in-one solution for smaller motor-centric applications.
- **DirectLogic and Do-More H2**
These are legacy product families; while some part numbers remain available to support existing installations, new designs should use any of the newer product families. For an upgrade path, in some cases legacy discrete logic can be imported into the newer Do-More BRX series.

Jump to
Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications



AutomationDirect offers a helpful online PLC selection tool at <https://www.automationdirect.com/systembuilder>

Here is a table comparing PLC families/series with the most essential features and capabilities.

Controller Family	Controller Type			Hardware					Programming Language				Motion Control		I/O Points	
	Traditional PLC	Open-Source Controller	EtherCAT Motion Controller	Stackable Brick	Rack-based	Wired Ethernet	Wi-Fi	Serial	Relay Ladder Logic	Structured Text (ST), Sequential Function Chart (SFC), Function Block (FB), Instruction List (IL)	CircuitPython, C++, JavaScript	Node-RED	Motion Control 1-4 axes	Motion Control 5-16 axes	100-1,000	1000+
CLICK Stackable Micro Brick	X			X		X			X						X (local I/O)	
CLICK PLUS Stackable Micro Modular	X	X (w/ Node-RED module)		X		X	X	X	X			X (w/ Node-RED module)	X		X (local I/O)	
Do-More BRX Stackable Micro Brick	X			X		X	X	X	X				X	X		X (local/remote I/O)
Productivity1000 Stackable Micro (standard)	X			X		X		X	X	X (ST)			X	X (w/PS-AMC)	X (local/remote I/O)	
Productivity1000 Brick (mini)	X			X		X		X	X	X (ST)			X	X (w/PS-AMC)	X (local/remote I/O)	
Productivity2000 Micro Modular	X				X	X		X	X	X (ST)			X	X (w/PS-AMC)		X (local/remote I/O)
Productivity3000 Modular	X				X	X		X	X	X (ST)			X	X (w/PS-AMC)		X (local/remote I/O)
ProductivityOpen (Arduino Compatible) Stackable Micro		X		X		X	X	X			X				X (local I/O)	
ProductivityCODESYS Micro Modular					X	X		X	X	X (All)						X (local/remote I/O)
LS Electric Stackable Micro and Brick XGB PLC, and XMC Motion Controller			X	X		X			X	X (ST, SFC, IL)			X	X		X (local/remote I/O)

Jump to Chapter

Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose a PLC Platform

Chapter 4
PLC and I/O Hardware Components

Chapter 5
Industrial Certifications

Chapter 6
Digital Communications

Chapter 7
PLC Programming Software

Chapter 8
PLC Programming Languages

Chapter 9
Ladder Logic Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect PLC Platform Solutions

Chapter 12
Applications

Applications

AutomationDirect has supported countless end users, systems integrators (SIs), and original equipment manufacturers (OEMs) as they develop PLC-based automation for an incredibly wide range of personal, educational, consumer, commercial, facility, and industrial applications. The following links represent just a few of the many examples.

[Applying Industrial-Grade Automation to Small-Scale Processes](#)

Even desktop- and garage-sized process and manufacturing applications can benefit greatly from automation, but only when the right products are available to help designers easily create a solution.



[Automating an Animatronic Robot](#)

This home-built robot was created with industrial-grade automation technologies, providing a unique platform for learning career-enhancing skills.



[Automated Robotic Welding Cuts Production Time by Over 90%](#)

Automated Robotic Welding Cuts Production Time by Over 90%

A manufacturing company teamed up with a systems integration firm to significantly accelerate an infrastructure project by integrating a robotic welder with precision motion control, intelligent safety systems, and other advanced automation components.



Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Automating Escape Room Experiences

Industrial automation products can also be applied to unique applications, like the advanced control needed for a sophisticated escape room.



Installing a Rock-Solid Automation Retrofit

This electrical contractor relied on AutomationDirect PLC and HMI hardware and software to rapidly rip-and-replace the controls on a failing rock crusher.



Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

Mining Sunbeams - Solar Thermal Leverages the Sun for Producing Heat Energy

Solar thermal installations need advanced control systems to optimize energy harvesting, storage, and distribution at remote mining operations.



A Recipe for Automation Success

To improve performance and reliability, this grain blending company knew it needed to upgrade its homebrew controls to a modern control system platform.



Automating a Big Rig Light Show

A truck customization shop expanded their capabilities by offering AutomationDirect PLCs and HMIs to control expansive on-vehicle lighting displays.



Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications

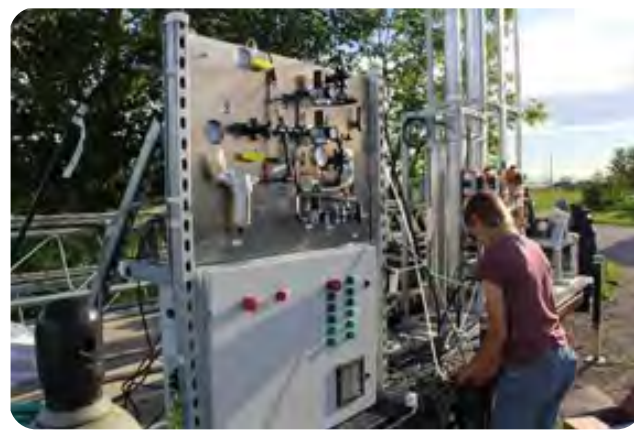
Picking a Cotton Press Automation Upgrade

A modern PLC automation platform will keep a classic industrial machine running for years to come.



Mission Control Built on an Industrial Foundation

These engineering students relied on an industry-proven automation platform and other products, to rapidly create a reliable rocketry ground test system.



Jump to
Chapter



Chapter 1
What is a PLC?

Chapter 2
PLC Basics

Chapter 3
How to Choose
a PLC Platform

Chapter 4
PLC and I/O
Hardware
Components

Chapter 5
Industrial
Certifications

Chapter 6
Digital
Communications

Chapter 7
PLC Programming
Software

Chapter 8
PLC Programming
Languages

Chapter 9
Ladder Logic
Programming

Chapter 10
Practical PLC Topics

Chapter 11
AutomationDirect
PLC Platform
Solutions

Chapter 12
Applications