PLC... The code that moves the world we live in

Programming machines: the underrated techniques of industrial automation

Fabio Matricardi
Published in Predict · 9 min read · Apr 25

PLC (Programmable Logic Controller) is a digital computer that is commonly used in industrial automation processes. It is essentially the code that moves the world we live in, allowing for the automation of processes and the control of machinery. While often underrated, the techniques involved in programming machines with PLCs are critical to industrial automation and are utilized in various fields, including manufacturing, automotive, and energy production.

Photo by Simon Kadula on Unsplash

In this article we will go through the following:

1. What is a PLC
2. Where and Why it is used
3. How a PLC is different from a PC

4. What are the PLC programming languages
5. Conclusions

What is a PLC?

A programmable logic controller (PLC) is an industrial computer used to manage and control manufacturing processes, such as assembly lines, machines, and robots. It is designed to be reliable, easy to program, and able to diagnose faults in the process.

You have to think to PLC like they are action machines: I mean that there is a brain, there are sensors to help the brain to understand the situation and to assess the next move, and there are actuators (yes a PLC alway does actions) that move the physical world with valves, motors, heaters, conveyors, robotic arms...

The sensors and actuators are commonly called I/Os (Input/Outputs): based on the number of I/Os programmable Logic Controllers (PLCs) vary in size and complexity. They can be as small as a single unit with 10–20 input/output, or as large as a rack-mounted device with thousands of I/O. Many PLCs are also connected to other PLCs and Supervisory Control and Data Acquisition (SCADA) systems.

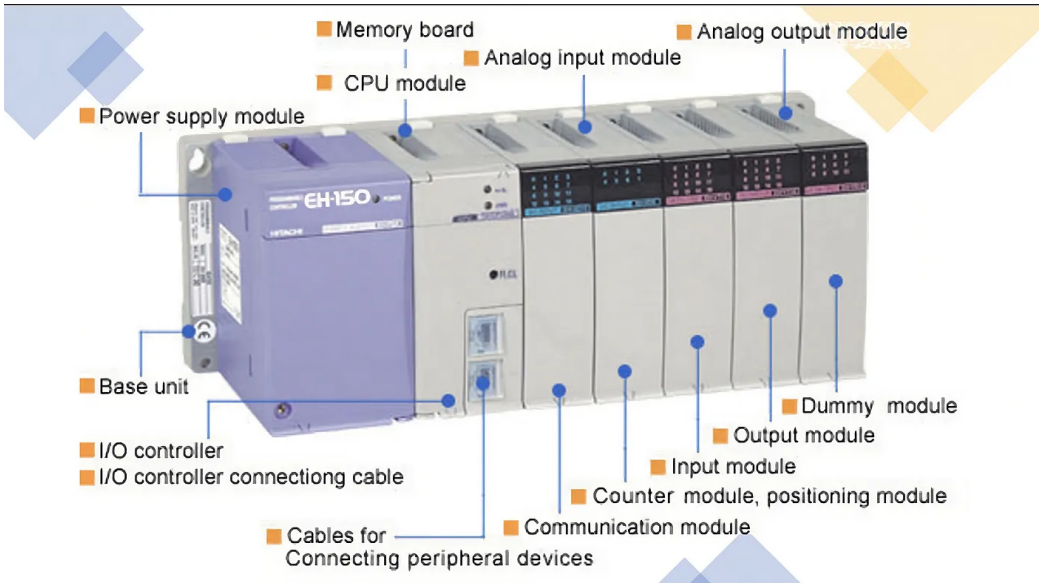


Image from https://www.microcontrollertips.com/ modified by the author

Because a PLC is a specialized computer used in industrial and manufacturing applications to control machinery and processes.,the hardware components of a typical PLC must be able to interact with industrial device. So a typical PLC hardware include:

1. Central Processing Unit (CPU) — The CPU is the “brain” of the PLC and is responsible for executing the PLC program, performing data processing, and communicating with other devices.

2. Input/Output (I/O) Modules — I/O modules are used to interface the PLC with the external world. They receive signals from input devices and send signals to output devices. I/O modules can be digital or analog, and can support a wide range of input and output types.
3. Power Supply — The power supply provides power to the PLC and its components. It may include features such as overload protection and voltage regulation.
4. Memory — The memory stores the PLC program, data, and configuration information. It includes both volatile (temporary) and non-volatile (permanent) memory.
5. Communication Ports — Communication ports allow the PLC to communicate with other devices, such as other PLCs, operator interfaces, and supervisory control and data acquisition (SCADA) systems. Communication ports may support a wide range of protocols, such as Ethernet, serial, and fieldbus protocols.
6. Programming and Monitoring Interface — The programming and monitoring interface allows the user to create, modify, and monitor the PLC program. It may include a text editor, a graphical programming tool, and a display for monitoring the PLC's status and performance.
7. Chassis — The chassis is the physical enclosure that holds the PLC and its components. It may include features such as mounting holes, cable management, and ventilation.

These components work together to provide the functionality and reliability needed for industrial control applications. The selection and configuration of these components depend on the specific requirements of the application.

Where and why it is used?

A PLC is a hard real-time system, meaning output must be generated in response to input conditions within a finite amount of time or else incorrect operation will occur.

Why the real-time system aspect is so relevant? Real-time computing (RTC) is when a system's hardware and software must respond to an event in a specified time frame, known as a "deadline". RTC ensures that programs meet these time constraints.

This condition is the minimum requirement in the industrial automation. The industrial sector requires that plant and personnel safety, as well as process control, be monitored and controlled in real-time. If a fault occurs, the PLC must take immediate action. If a hazard presents a safety risk, valves

must be shut and the process must go into shutdown to prevent further escalations.

Photo by American Public Power Association on Unsplash

One of the several advantages PLC offers is related to maintenance and replacement. Firstly, PLCs are highly reliable, meaning that they require minimal maintenance and have a long operational life. Secondly, since they are programmable and modular, they can be easily reconfigured and replaced as needed, without the need for specialized hardware or expertise. Additionally, PLCs can be remotely monitored and diagnosed, allowing for proactive maintenance and troubleshooting. These features make PLCs an ideal choice for businesses looking to minimize downtime and increase efficiency.

Another great advantage is the redundancy: PLC can be configure to have a hot backup systems to ensure that critical processes continue to operate even in the event of a failure. Redundancy can be achieved in various ways, such as through redundant power supplies or redundant PLC controllers. One common method of redundancy is to use a primary and secondary PLC, where the primary controller is responsible for the process control, and the secondary controller takes over in case of a failure. This ensures that the system remains operational and prevents costly downtime. Redundancy in PLC is especially important in industries where safety and reliability are critical, such as manufacturing plants, chemical plants, and power plants.

How a PLC is different from a PC

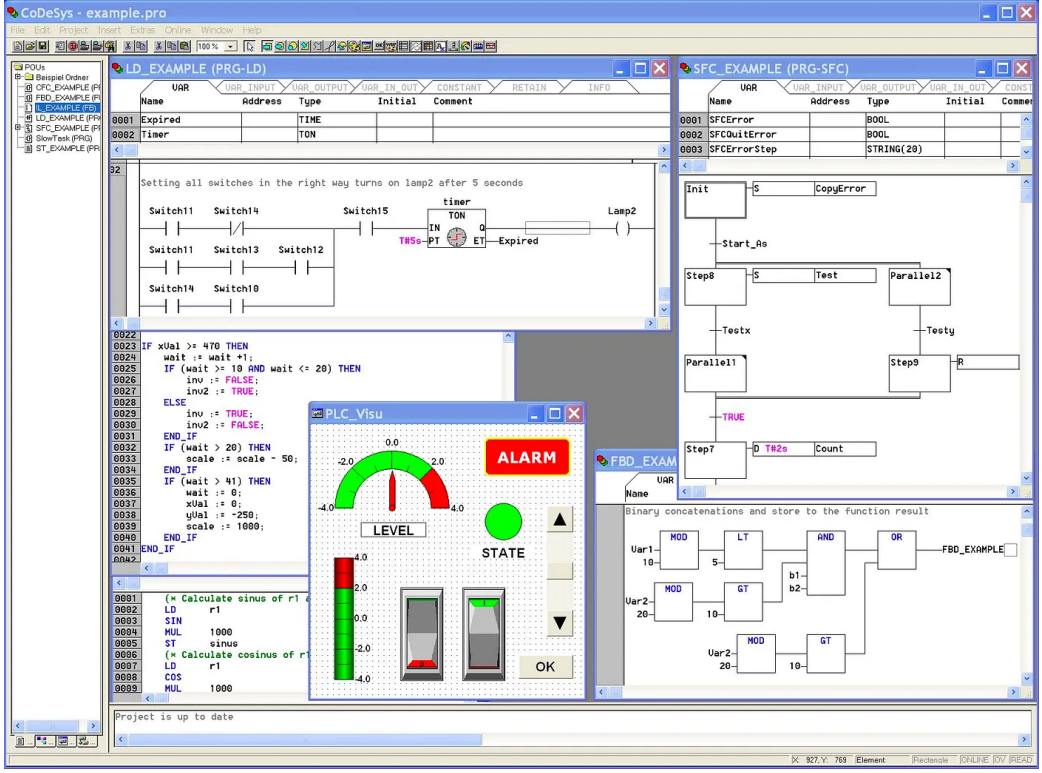
PCs are not designed for real time monitoring and control like PLCs, which are built for harsh conditions (dust, moisture, heat, cold) and have various I/O functions to connect with sensors and actuators.

A real-time system is one that quickly processes and responds to data to control an environment in the present moment, without significant latency. For these reasons software developers must utilize synchronous programming languages, real-time operating systems (RTOSes), and real-time networks to create real-time software applications.

PLCs can accept digital signals like limit switches, analog variables (temp, pressure, etc.) and complex data from positioning or vision systems. PLCs can then send outputs through indicator lamps, sirens, motors, cylinders, relays, solenoids, and analog signals. These input/output systems may either be part of the PLC, or have external I/O modules connected via a fieldbus or network.

Here a summary of the main differences between PLC and PC:

- PLCs are built to operate in industrial settings with varying temperatures, vibrations, and humidity levels, and are highly resistant to electrical noise.
- Computers receive input from floppy drives and CD Roms and output to a printer, whereas Programmable Logic Controllers (PLCs) receive input from control elements (e.g. push-buttons, limit switches, temperature switches, pressure switches, transducers) installed on machines to be controlled. Output from PLCs includes final control elements (e.g. contractors, solenoids, valves for positioning, indication lights).
- Programmable Logic Controllers (PLCs) have ROM memory containing Operating Systems (OS) and Application Programs. As the OS is stored in ROM, it does not require loading. It can be challenging to differentiate between PLCs 'BIOS, OS and Application Programs.
- PLC programming is not conducted using common computer languages such as Python, Ruby, or C#. It is specifically designed for plant engineers and maintenance personnel to use a specialized language. Find out more in the next section.



CodeSyS — programming languages and HMI

What Programming Language is used in PLC?

The International Electrotechnical Commission (IEC) has developed a set of standards for programmable controllers, including the programming languages used for PLCs. The relevant standard that defines the five programming languages for PLCs is IEC 61131–3.

IEC 61131–3 defines a common framework for programming languages used in industrial automation. The standard includes five programming languages, which are:

- Ladder logic (which reads similarly to electrical drawings)
- Structured Text — similar to traditional command-based computer programming; used in the previous example
- SFC (Sequential Flow Chart) — very similar to a traditional flowchart
- Instruction List
- Function Block Diagram

IEC 61131–3 also defines a common syntax and set of rules for each of these languages, allowing for interoperability between different PLCs and programming tools. The standard helps to ensure consistency and reliability in PLC programming across different manufacturers and applications. Let's have a quick look at them, one by one

1. Ladder diagram (LD)

Ladder diagram (LD or Ladder Logic) is a universal PLC programming language that is one of the oldest in use. It is used with Programmable Logic

Controllers (PLCs).

The Ladder Diagram language is widely popular due to its easy learning process utilizing basic logic gates and programming rules.

2. Instruction List (IL)

Instruction List (IL) is a low-level programming language that uses a mnemonic code to represent individual instructions that are executed by the PLC. The language is similar to assembly language used in computer programming. IL is often used for simple control applications where performance is critical, or when the other programming languages may not be suitable.

IL programs consist of a series of instructions, each of which performs a specific operation on the data stored in the PLC's memory. The instructions are executed sequentially, one after the other, until the program reaches the end or encounters an error.

3. Structured Text (ST)

Structured Text (ST) is a high-level programming language used in industrial automation. ST is a text-based language that uses structured programming concepts, such as loops and conditional statements, to write PLC programs.

ST is similar to other high-level programming languages like C, Pascal, and BASIC, and supports a wide range of data types, including integers, floating-point numbers, and Boolean values. It also supports arrays, structures, and user-defined data types.

ST programs consist of a series of statements that are executed sequentially. The language supports a wide range of operators and functions for performing arithmetic, logical, and bitwise operations on data.

4. Function Block Diagram (FBD)

Function Block Diagram (FBD) is a graphical programming language. FBD represents logic functions as blocks with inputs and outputs. The blocks can be connected together to create a network that represents the logic of a control system. The inputs to the blocks are connected to the outputs of other blocks or to input devices, and the outputs of the blocks are connected to the inputs of other blocks or to output devices.

Each block in FBD performs a specific function, such as a logical operation, arithmetic operation, or data conversion. The blocks can be customized and combined to create complex control algorithms: they are like custom circuits, equivalent to the functions or classes used in python.

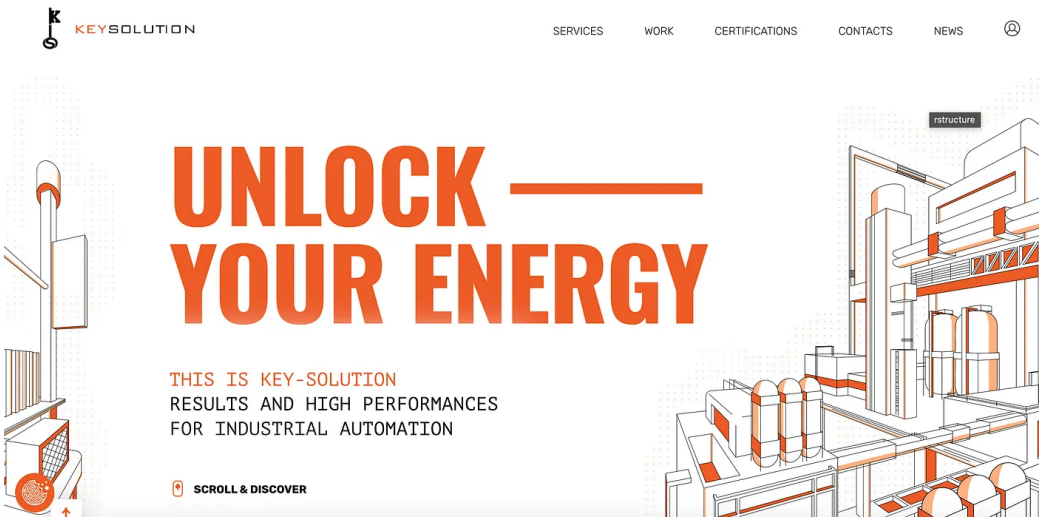
FBD is a visual language that makes it easy to understand and modify complex control algorithms. The visual representation of the logic can help to identify errors and make changes quickly.It is a powerful language that allows for the creation of complex control algorithms with minimal coding.

5. Sequential Function Charts (SFC)

Sequential function charts (SFC) is also a **graphical programming language.** It is not a textbase. It has become a popular method of accurately specifying sequential control requirements.

The benefit of SFC is easy to understand. Because you can visualize what is happening and when it is happening in the procedure of the code.

The main function of SFC, and that is different from all the other program and task execution of the PLC, is only the active parts of the code are executed. Due to this, it makes easier to troubleshoot and to change the code if problems occur.



key-solution.eu home page

Conclusions

Programming machines that run plants and entire industries is really cool, and the greatest part is that you can see the result of your code. An additional reward to the joy of programming is related to the HMI (Human Machine Interface).

When you program a PLC you cannot focus only on the instructions and logic (backend) but you have to take care also of the interaction with the operator, the plant, the process and the safety (frontend).

If you want to see a showcase of projects that are running with PLC have a look at Key Solution Srl website (it is the company I work for, by the way... Click on one item in the timeline to see the details).

. . .

If this story provided value and you wish to show a little support, you could:

1. Clap 50 times for this story (this really, really helps me out)
2. Follow me on Medium
3. read my latest articles (https://medium.com/@fabio.matricardi)

Timer on Delay

When counting to 10 is really a good idea...

medium.com

The twilight of maintenance?

What to do when no one knows anymore how to fix broken things

web3usecase.co

Plc | Industrial Automation | Programming | Human Interface | Future Of Work