FUTURE PROOF YOUR CAREER

PLC PROGRAMMING ESSENTIALS

Everything you need to learn to get a job in PLC programming today

SolisPLC

PLC Programming Essentials

Everything you need to learn to get a job in PLC programming today

By SolisPLC

Table of contents

INTRODUCTION

Why this book?

Over the past few years, we helped thousands of engineers, technicians, and electricians learn PLC programming, and get jobs in the field of automation at some of the world's top companies including Procter & Gamble, Amazon, Kraft Heinz, and hundreds of others.

One thing that we've noticed is that many students assume that you need to know how to create PID loops, how to scale analog signals, how to manage complex FOR loops, and other complex concepts to land a PLC programming job. As a result, many can't figure out where to start, get quickly overwhelmed trying to learn too much before starting to apply for jobs, and eventually get discouraged and give up.

We are writing this book to help you get started with PLC programming with the goal of learning what you need to know to get a job. If you can confidently talk about the topics covered within this book you will come out above 90% of candidates. Many PLC programmers struggle with basic concepts, and yet that's exactly what employers will test for.

Who is this book for?

Naturally, we would like to write a book for everyone to find interesting. However, we're deliberately concentrating on students or working professionals who are new

to PLC programming and are looking to learn the fundamentals they need to get a job as a controls engineer or progress in their career. With this in mind, we will leave the excessive theory aside, and focus on what you need to get the job done.

In addition to walking you through the essential concepts, this book will provide you with practice interview problems, small challenge projects, and links to video tutorials for more in-depth visual explanations.

Our hope is that this book will help you get started on the right track and give you the confidence to keep learning with the goal of becoming a skilled PLC programmer.

How to get the most out of this book?

As you go through this book, some concepts will be easy to grasp, but some others might be a bit challenging, and that's totally ok. To make sure you have all the tools & support needed to learn what you need to get a PLC programming job, we want to point you to a few resources you might find useful as you go through this book:

1. **The SolisPLC Forum –** If you need help, make sure to **post your questions on our forum** so you can get help from our community & team of experts. Remember, there is no such thing as a stupid question. We're all here to learn.
2. **Free PLC tutorials –** As you learn new concepts, don't be afraid to explore further. We've **written 100+ technical tutorials** on a variety of topics, and they're absolutely free to access.

CHAPTER 1 | LEARNING PLC PROGRAMMING

1.1 | Why Programmable Logic Controllers (PLCs)?

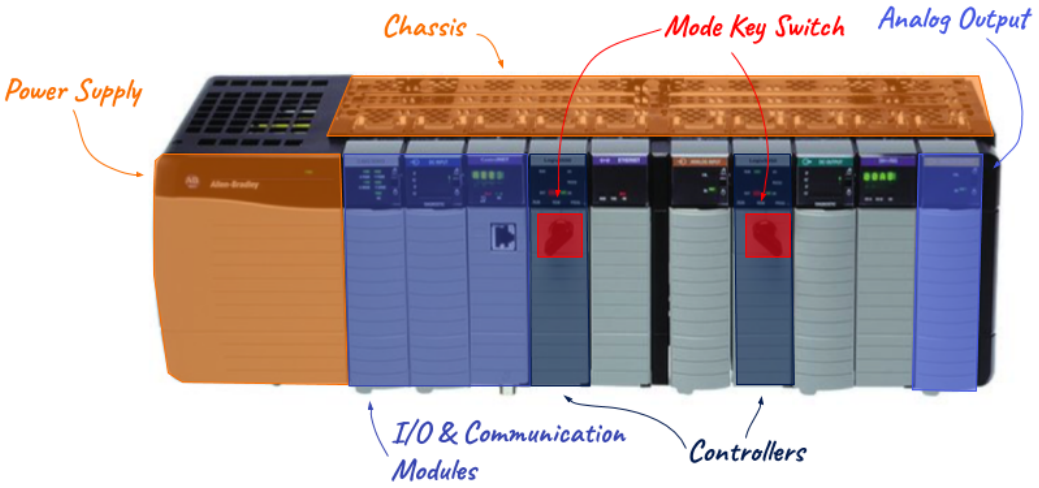
Most consumer goods, from food to cell phones, are manufactured at a production facility, shipped through a distribution channel, and delivered to a retailer or directly to your door using automation. The process of eliminating the human element from this process is not a new concept. Machines are capable of producing more, have a higher tolerance level, and have a higher quality level of craftsmanship when it comes to repeated production. Thus, **there is a constant strive to automate every step of the manufacturing process**.

Different systems are used in each segment of the automation. However, they have many common goals: high reliability, high repeatability, and ease of deployment and maintenance. Based on these principles and the demand of the manufacturing segment, Programmable Logic Controllers (PLCs) were created in the last century.

Their goal was to replace the relay-based circuits, simplify the process of changing operations, and improve the reliability of the system.

Programmable Logic Controllers were created as a cost-effective alternative to relay-based systems a few years after transistors became more commonplace. **A Programmable Logic Controller at the core is a computer with a high degree of reliability** capable of running a program without interruption in a 24/7/365 environment.

At first, PLCs were simple and easy to set up. However as the needs of the business changed and manufacturing floor complexities became apparent, PLCs evolved. They now **required a dedicated person that knew PLC programming** in order to create the program, optimize it for a specific project and change it depending on business needs.

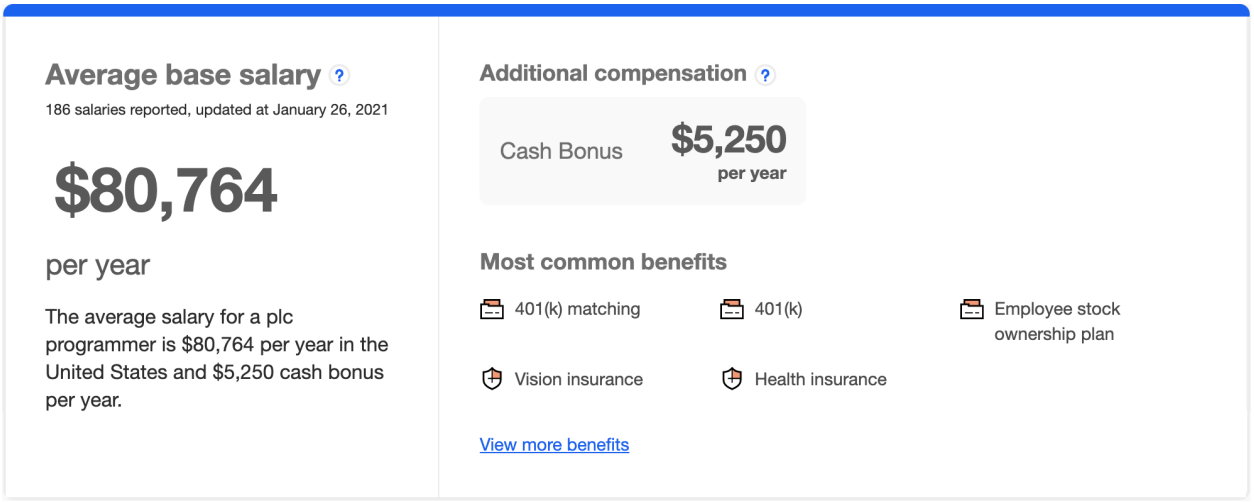


A Programmable Logic Controller is typically the hardware that will control the process. In case you're wondering, a PLC looks like a box filled with electrical circuits similar to your desktop computer. A PLC is typically housed in what's called an electrical enclosure in order to protect it from the harsh environments it may control. For the context of the discussion, it's not uncommon to reference PLCs as the entire plant floor control system as they're always tied to sensors, motors, switches, valves, etc.

Since their inception, PLCs have come a long way. They're still robust, process-driven machines. However, they're starting to incorporate some of the features we'd typically see in an Information Technology (IT) environment. Examples include Ethernet connectivity for data collection, sensor monitoring through technologies such as IOLink, MQTT protocols allowing server-based connections to be made, and much more. In other words, PLCs aren't what they used to be a few decades ago.

As manufacturing facilities around the globe rely heavily on PLCs, and the technology continues to evolve, there is an increasing need for experts capable of developing, supporting, and managing these systems. The opportunity to learn PLC

programming has pushed many toward higher-paying jobs, secure work environments, and excellent career growth.



PLC Programming average salaries in the USA (Source: Indeed.com)

1.2 | The Different Paths to Becoming a PLC Programmer

There are many paths to becoming a competent PLC programmer. You may choose to **pursue a traditional college degree, learn through online tutorials, or take a course that will issue a plc programming certification upon completion**. However, what is the best option, and what are the best PLC programming certifications?

Engineering and Engineering Technology Degrees

A bachelor's degree in electrical engineering, a closely related field, or engineering technology is a common path of getting into industrial automation. Multiple universities offer a program that will incorporate PLC programming, HMI development, Robotics, and control systems. Furthermore, at the end of the program, the recipient will be issued a degree that will carry more weight than many PLC programming certifications, especially from a known university.

Although it may be a suitable path for some, a university degree will typically take three to four years, will come with a heavy financial burden, and will not always result in a desired position in the end. Yet another drawback of this path is that the program may not necessarily require you to take all classes in automation. You'll most likely be needed to learn calculus, linear algebra, basic chemistry, and more.

Official OEM PLC Programming Certifications and Classes

Every OEM, Allen Bradley, Siemens, Automation Direct, etc., provides classes and certifications based on proficiency. These classes will often take place at a physical location, be taught by education professionals, and span from one day to several weeks.

Although it is possible to get a PLC programming certification by going through these classes, they don't always carry the weight students expect. During these sessions, students will become familiar with how to program PLCs. Still, they typically won't come out with any projects, assignments, or tests that would require them to think beyond the material presented.

We've spoken to countless engineers who took such classes and were disappointed by the experience and the impact their PLC programming certification played when applying for a job. They all mentioned one or more of the following reasons:

- Lack of completed projects
- Information overload
- Employers do not value certifications
- Expensive

A typical class from Rockwell Automation on PLC programming can cost anywhere from USD 2,000 to 6,000. Although it is an investment a manufacturing company can

make in their employees, this price tag is often outside of reach for most individuals on their terms.

Online PLC Programming Certifications

Online learning is not new. However, it is not a fully mature industry in PLC programming, industrial automation, and control systems. There is a wide range of quality when it comes to PLC programming certifications issued by various authorities.

Here is a list of checkpoints that we believe you should consider when selecting the authority to train and issue a PLC programming certification:

- **Instructor Experience** - Has the individual been in the field, or are they just a teacher? Although it is possible to learn control systems from a book, an engineer's industry experience is priceless. An instructor who has been in the field will know what to emphasize, provide you with industry-specific knowledge, prepare you for the real world, and the interview.
- **Issuing Company** - What will the employer be presented with if they decide to research the company that has issued the certification? Is this company a practitioner of their craft or an aggregator of certification programs?
- **Project Work** - What kind of projects will you complete during your certification? Will you be challenged as you learn new material? Will you apply your knowledge to solve a problem, design a system, or create something from scratch?
- **Reputation** - What are others saying about the certification program? Has it brought them value?

Significant Advantages of an Online PLC Programming Certification

Low Cost – An online certification will come at a much lower cost than the information supplied in an in-person class. We typically recommend our students

take a portion of the money saved to purchase a small training kit they'll be able to use for the years to come, showcase during an interview and grow their skills beyond the class.

Self-Paced Learning – For years, we've underestimated the quality of learning materials that we can easily pause and resume with a click of a button. Unlike an in-person class, you can spend as much or as little time on specific topics during your PLC programming certification. Master the skills at your own pace and go back to lectures you need to review before your interview.

Projects, Assignments, and Quizzes – An online-based PLC programming certification allows the instructor to assign challenges that the students complete independently. These projects are critical for the mastery of specific concepts and reinforce what was learned during the lectures.

Although we recommend that you explore all of the options available when it comes to PLC programming certifications, we strongly believe that a thorough online program will provide you with the knowledge you need as well as the certification you need to land a job without spending thousands of dollars.

CHAPTER 2 | CHOOSING A PLC PLATFORM

The first step on your learning journey is to pick which platform you will learn first. Although there are many different platform options available, We highly recommend that you choose one of the two most significant platforms on the current market if you're looking to land a job in the industry. These platforms are:

- Allen Bradley (Rockwell Automation)
- Siemens

Furthermore, we recommend that you choose Allen Bradley if you're in the North American region and Siemens if you're elsewhere in the world.

> 💡 **[NOTE]**
> In this book, we will focus on Allen Bradley. If you're interested in learning Siemens instead, you can get started here.

Your first task is to become familiar with the tools required to learn either platform. For Allen Bradley, we recommend that you become familiar with RSLogix 500 or RSLogix 5000 as your first software package. For Siemens, you'll have to get used to the TIA Portal software.

Ideally, you would have access to PLC hardware & software from your employer or school. But if you don't, the second-best way to get started is to download the free version of RSLogix500, RSLogix500 Emulate, and RSLinx. Siemens provides an equivalent trial software package for 30 days on their website.

2.1 | Getting Started with Allen Bradley PLC Programming

Allen Bradley PLC Programming is a highly desired skill in industrial automation. It's what allows one to create a set of instructions in order to control different devices, pieces of machinery, and entire manufacturing plants. However, there are multiple challenges when it comes to learning PLC programming: high complexity of the platforms, costly hardware & software costs as well as low availability of online materials.

Most PLC programming packages require a paid license. Clients, which include large manufacturing companies, OEMs & Machine Builders, have absolutely no trouble paying for these. However, they are quite inaccessible for those looking to get into Allen Bradley PLC programming.

An excellent alternative to acquiring these licenses is to find a package that doesn't require one. Every manufacturer has a package that will be free, feature a limited trial, or otherwise. It's highly recommended to learn through one of these softwares if you don't have access to the expensive packages through your employer.

In the case of RSLogix or Studio 5000, the alternative would be RSLogix 500. This package comes in different license levels. However, it can be downloaded for free.

Students become discouraged by the fact that there seems to be a major difference between RSLogix 500 and 5000. However, that isn't the case. The tag names, as well as the UI presented, will differ, but the fundamentals of PLC programming remain the same. If you can confidently say that you've mastered PLC programming concepts in RSLogix 500, you won't have any trouble applying these skills to RSLogix or Studio 5000.

With this being said, let's see how you can download & configure RSLogix 500 for free.

2.2 | Downloading RSLogix 500 for Free

The simplest method of getting into PLC Programming is to use an emulator. However, locating the proper files may be troublesome as Rockwell Automation has created a very intricate process to access them. Furthermore, downloading from third-party sites may be unsafe due to viruses and corrupted versions of the required software.

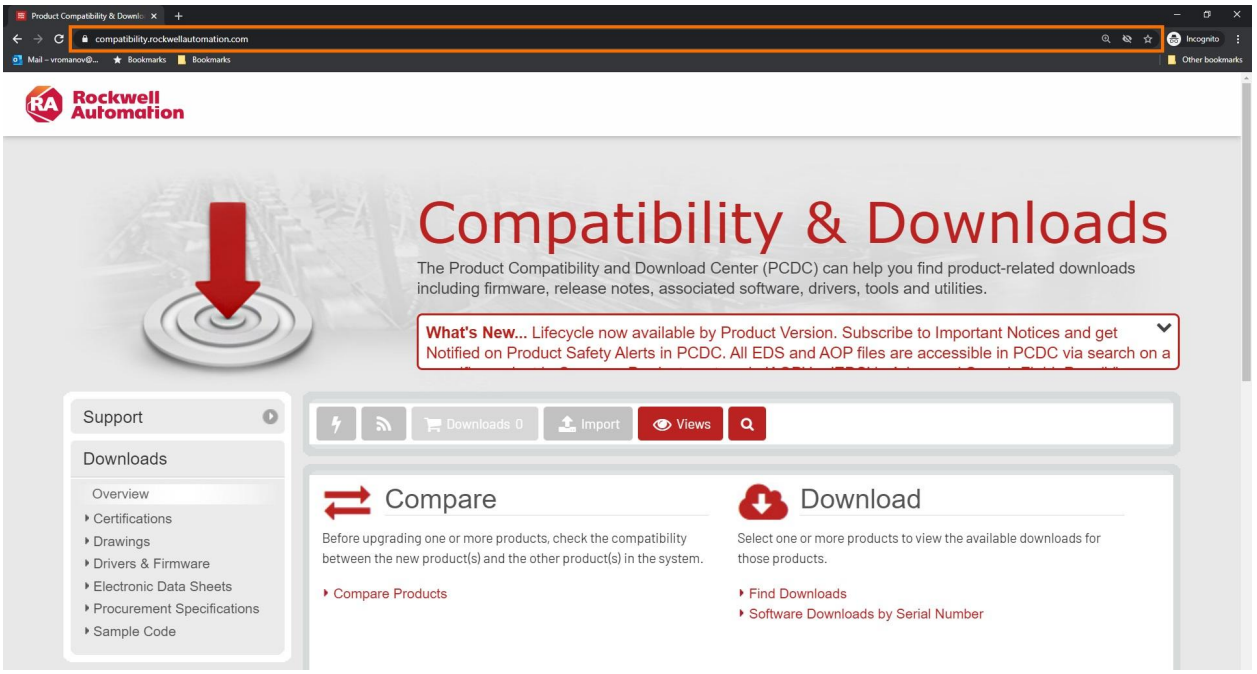
We will now show the latest method of downloading the following tools absolutely free from the original source:

- RSLogix500
- RSLogix500 Emulate
- RSLinx

With these tools, you can start practicing PLC Development without the need of purchasing hardware.

Step 1 - Go to Rockwell Compatibility & Downloads

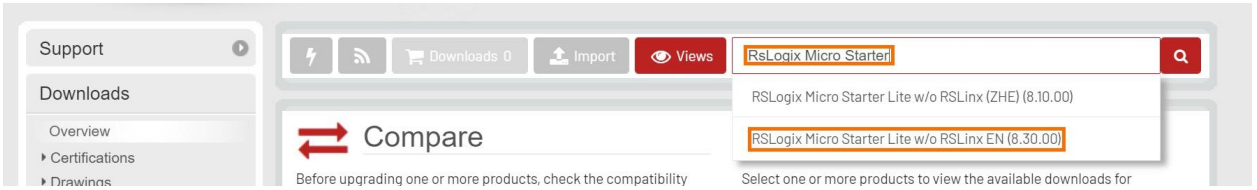
Link: https://compatibility.rockwellautomation.com/



RSLogix 500 Free Download - Rockwell Compatibility & Downloads

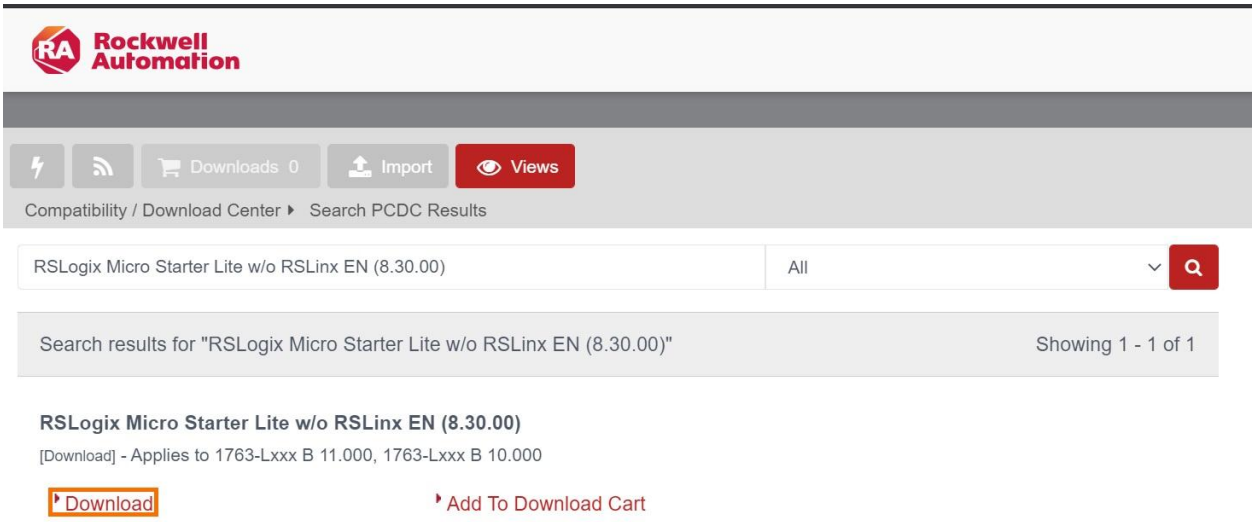
Step 2 - Search for "RsLogix Micro Starter"

Step 2.1 Select the English (EN) version as shown below.



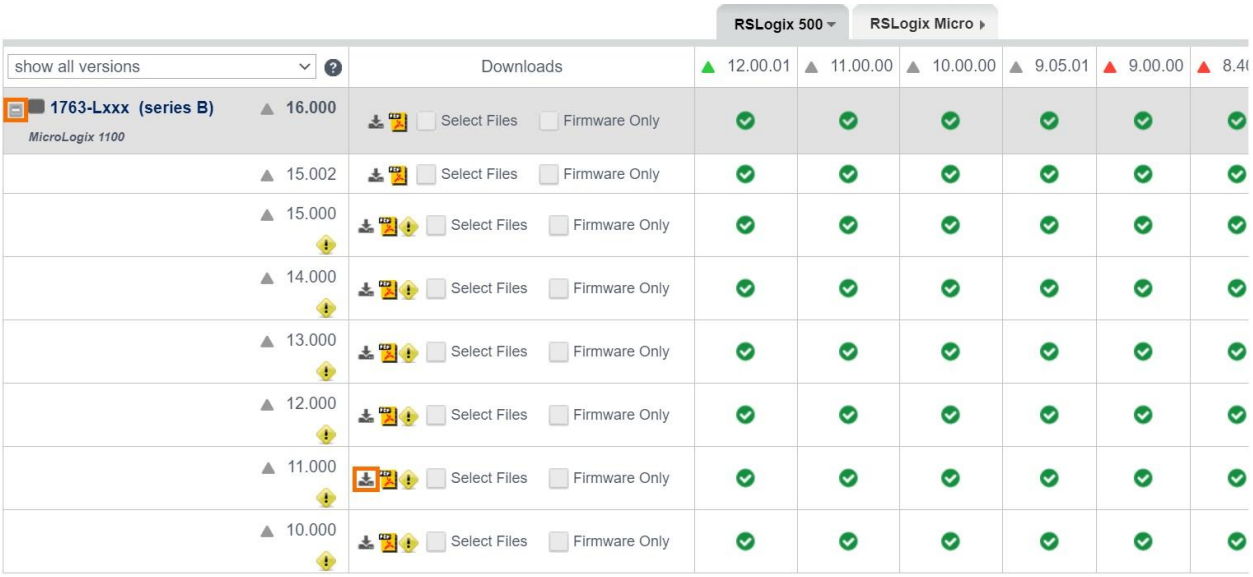
RSLogix 500 Free Download - RSLogix Micro Starter

Step 3 - Select "Download"



RSLogix 500 Free Download - Download RSLogix

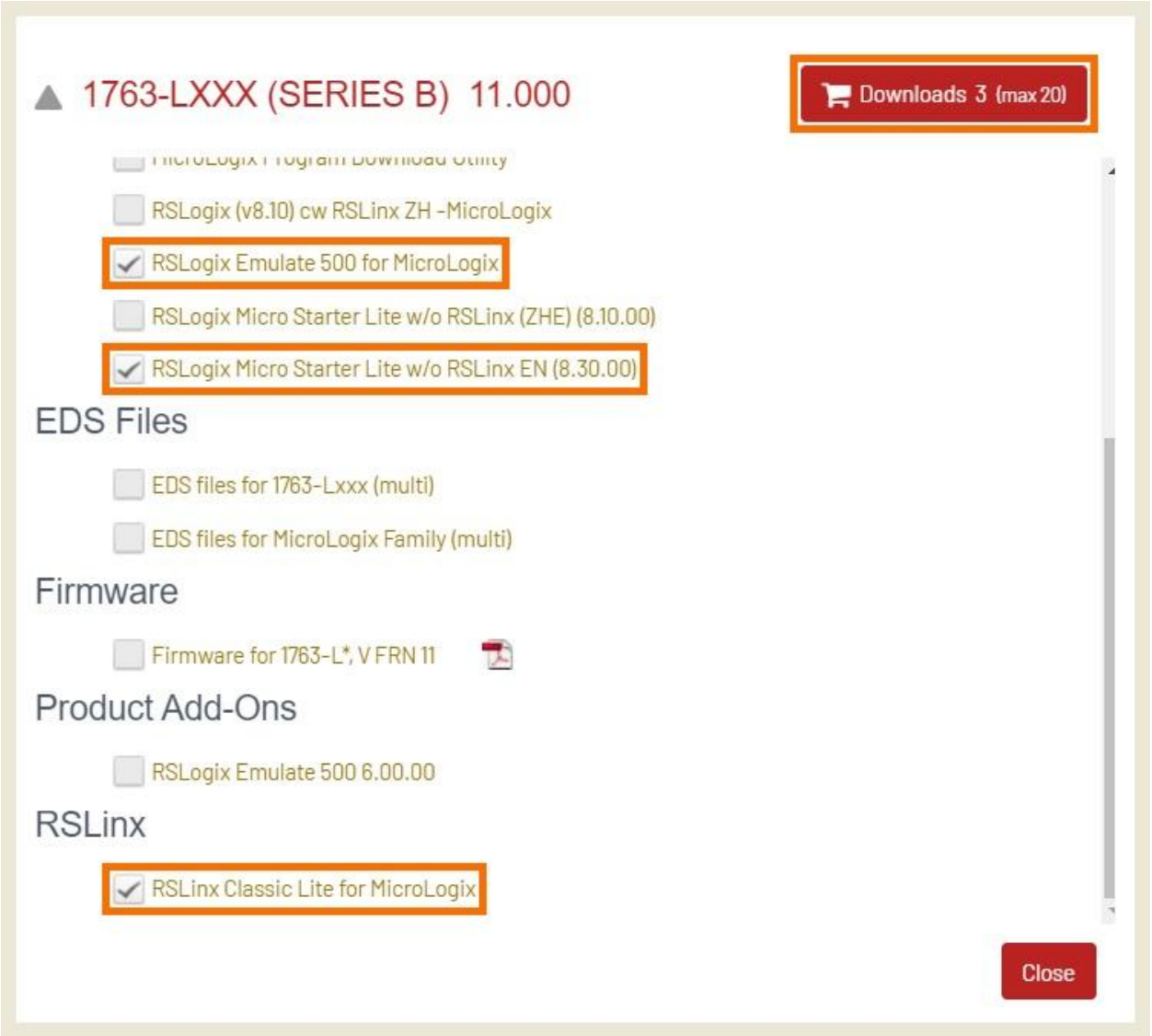
Step 4 - Expand the Menu and Press the Download Button on the v11.000 RSLogix Software



RSLogix 500 Free Download - Choosing the Version

Step 5 - Choose the RSLogix Tools | Emulate, RSLinx, and RSLogix 500

Step 5.1 - Press on "Download"

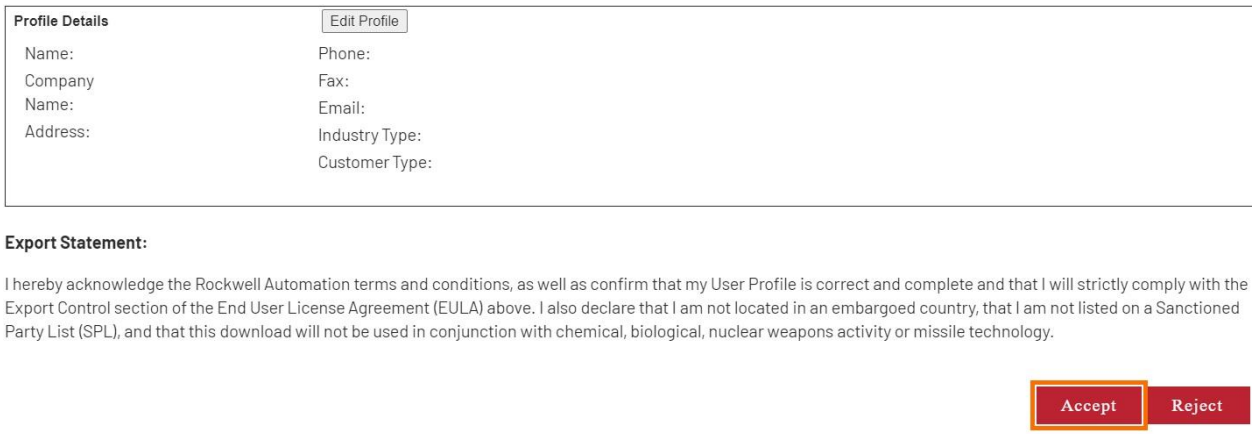


RSLogix 500, RSLogix Emulate and RSLinx Lite Free Download

Step 6 - Sign In to a Rockwell Account

Step 7 [Optional] - Create a New Account - Free

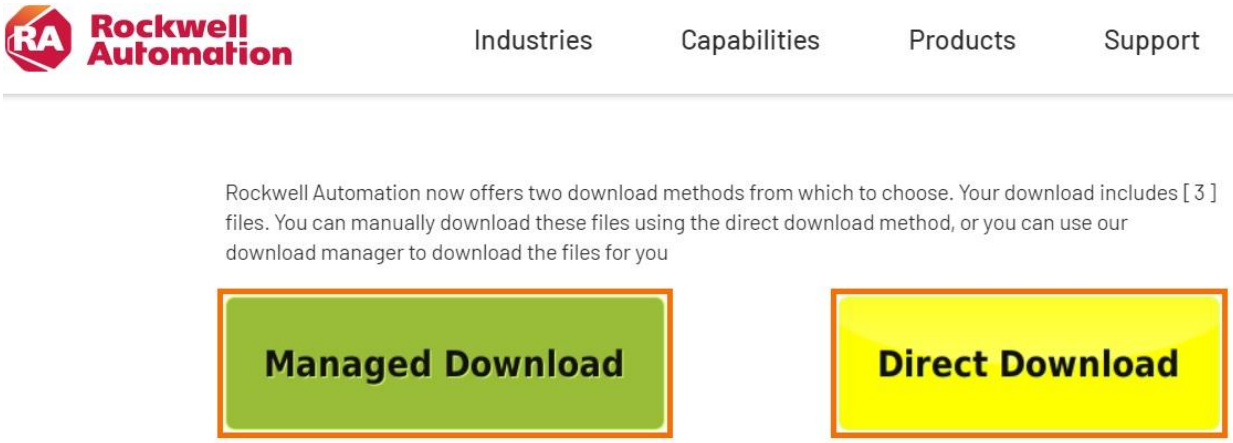
Step 8 - Accept the Terms of Service



RSLogix 500 Free Download Terms of Service

Step 9 - Use "Managed Download" or "Direct Download"

The difference between the two methods is minimal. The managed download utilizes Rockwell Automation's download tool while the direct download method uses direct files.



RSLogix 500 - Managed Download and Direct Download

Step 10 - Install the Files

At this point, you'll have access to the three tools you need to get started with RSLogix 500 PLC programming. These are the most commonly used tools for engineers to get into control systems programming.

> 💎 **[BONUS]**
> If you're having trouble, make sure to check out this **free mini-course**, that will walk you through all the steps to install everything on your computer.

CHAPTER 3 | PLC PROGRAMMING LANGUAGES

Now that you choose your platform, it’s time to pick a programming language. A PLC can be programmed in several different languages including:

- Ladder logic
- Function block diagrams
- Structured text
- Sequential flow charts

Although it may be tempting to pick one approach over others based on their simplicity or familiarity with other languages, ladder logic should be the first language you master. The reason is that it's the most widely spread type of PLC programming due to the roots of relay logic, simplicity to implement, and ease of debugging.

Most of the logic built-in ladder logic uses the following three instructions:

- Examine if closed (XIC),
- Examine if opened (XIO)
- Output energizes (OTE).

Furthermore, a high emphasis should be placed on learning rung structures as well as branching at this stage of the process.

Start by implementing logic routines that use the three instructions above. Understand how they impact the booleans they're tied to and experiment with different rung configurations. As you build different structures, think about how they

may reflect systems in our daily life. Can you build a two-way switch or a backup system that turns on a secondary source if the main one fails?

We will cover each of these topics in-depth in the upcoming chapters.

Before we delve into Ladder Logic, it's important to be aware of the PLC programming languages that exist and their differences.

The 5 most popular types of PLC programming languages of 2020 are:

- Structured Text (ST)
- Sequential Function Charts (SFC)
- Ladder Logic Diagram (LD)
- Function Block Diagram (FBD)
- Instruction List (IL)

The International Electrotechnical Commission 61131-3 outlines 5 different PLC Programming languages: ladder logic, structured text, function block diagrams, sequential function charts, and instruction lists. Each one of these languages has advantages, weaknesses, and best use cases. As a competent PLC programmer, it's essential to be aware of these options to troubleshoot existing code, utilize the right tool for the job, and have a different perspective on how problems may be solved.

Furthermore, depending on the PLC platform of choice, specific languages may come at a premium or not be available at all. For example, only the full license of RSLogix 5000 will have access to languages beyond Ladder Logic.

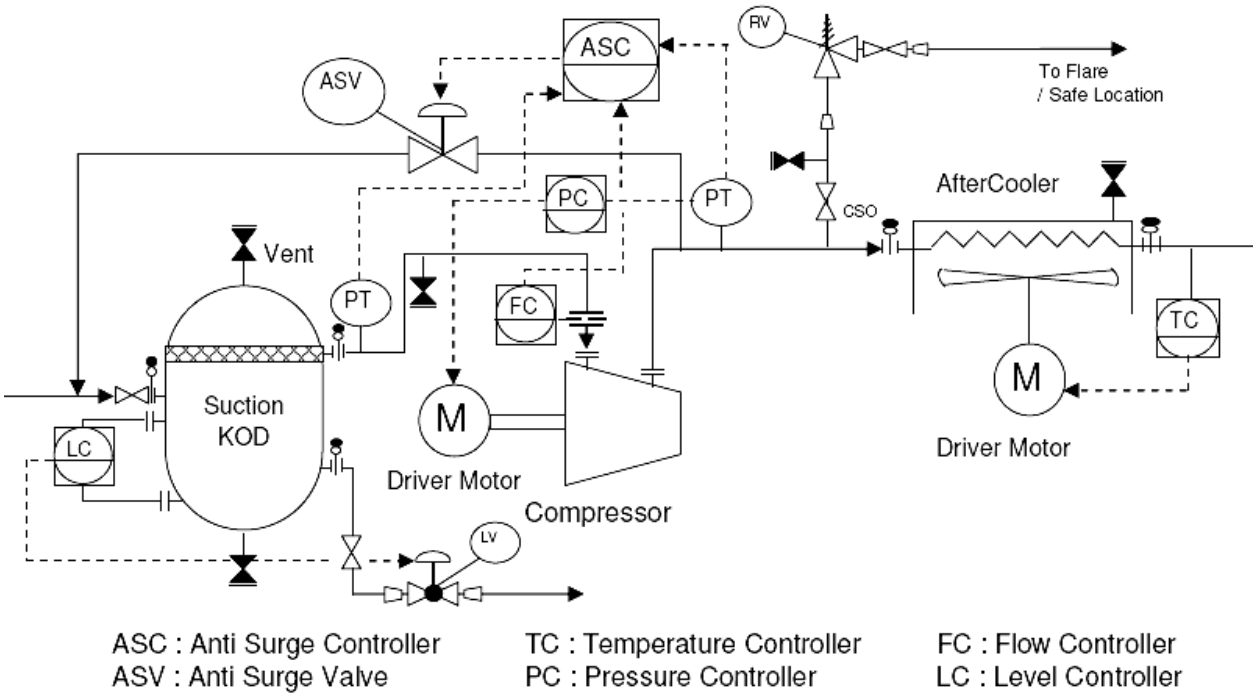
Let's take a more in-depth look at each one of these languages, go over their applications, general structure, and use cases.

What is the most popular programming language for a PLC?

This question is debated among PLC programmers across the world. The consensus is that the most utilized language for PLC programming is ladder logic. This is due to the fact that it's highly flexible, easy to learn, and very well understood by electricians who have worked with schematics that model the same architecture.

However, over the last decade, a younger workforce has entered manufacturing. These engineers and technicians have been primarily taught modern languages such as Java, Python, and Javascript. These languages have a closer resemblance to Structured Text [ST] and thus promote its usage.

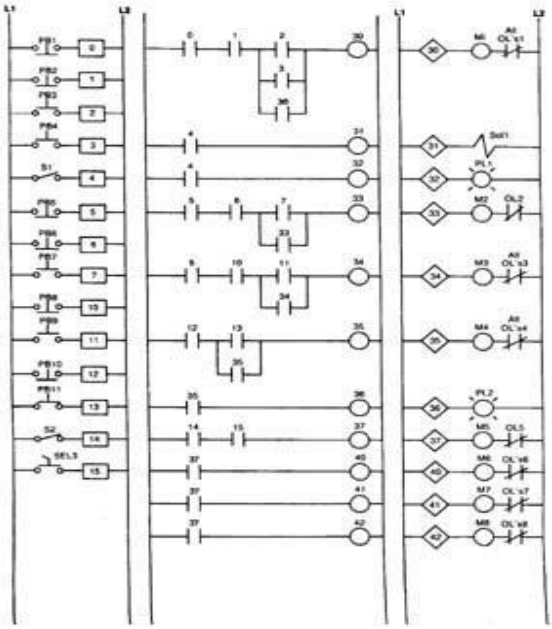
It's also important to consider the industry in question. A chemical process is typically designed using Piping & Instrumentation Diagrams (P&ID). These specific schematics are easily replicated through the use of Function Block Diagrams [FBD].



PLC Programming Languages - An example of a chemical process that would be easier to implement in Function Block Diagrams (FBD) PLC programming

3.1 | Ladder Logic

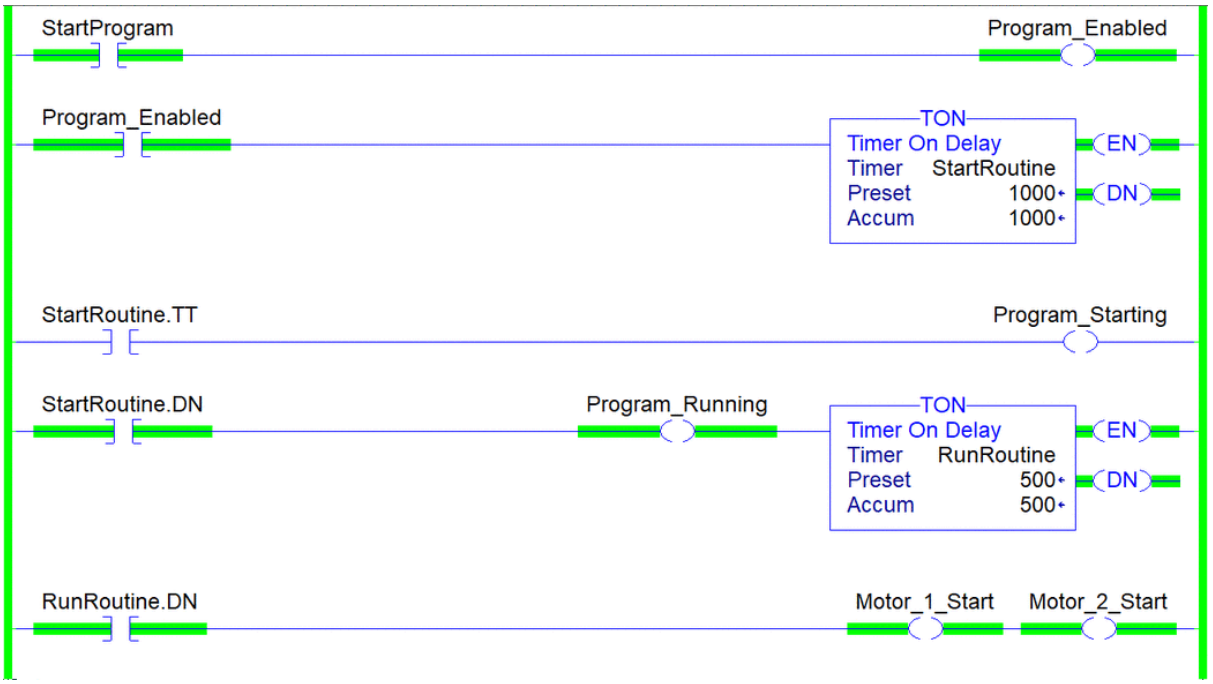
Before Programmable Logic Controllers became popular, relay-based controls were the norm at most manufacturing sites. Relays drove loads based on the simple logic that was implemented through the physical wiring of the devices. The wiring of these devices was specified in electrical drawings that assumed the layout resembling a ladder. As the most basic PLCs were introduced into the field, ladder logic PLC programming was designed to mimic the layout of relay-based circuits. In other words, ladder logic was one of the first PLC programming languages that's still used today due to simplicity.



PLC Programming Languages - PLC Schematics Leading to Ladder Logic PLC Programming

Since its inception, ladder logic has evolved significantly. However, the basic principles of operation remain the same. Ladder logic PLC programming evaluates

each rung of a ladder in sequential order, assesses conditional instructions, and if the result evaluates to “TRUE,” the output instructions are executed.



PLC Programming Languages | Ladder Logic PLC Programming Example in RSLogix 5000

Advantages of Ladder Logic PLC Programming

- **Simple to Implement and Troubleshoot** | Ladder Logic is a visual language that provides confirmation of status for most instructions. In other words, it’s easy for someone with little knowledge of a specific process to walk through the program and understand the logic.
- **Modular Design** | Ladder Logic can be easily modified through the addition or subtraction of logic. Each rung is a separate condition and can be removed or added as needed.
- **Resilience and Consistency** | Ladder Logic allows the user to implement many functions. However, the language is heavily standardized and doesn’t give full flexibility, thus keeping the code consistent between different implementations.

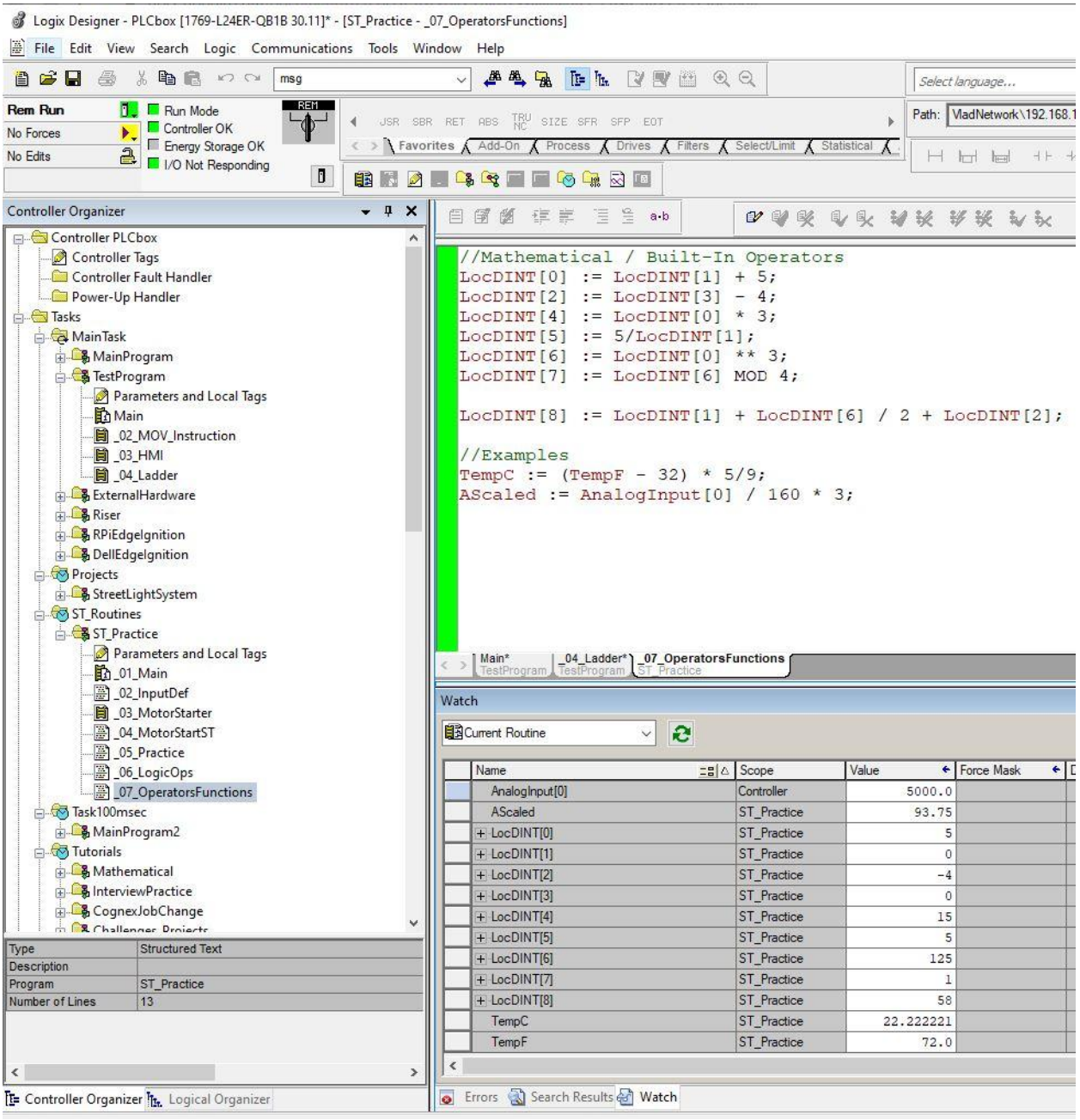
Drawbacks of Ladder Logic

- Steep Learning Curve | Ladder Logic is a simple language, yet not very intuitive to those who come with backgrounds in C, C++, Java, or Python. That being said, it may be easier to grasp for electrical engineers and those with basic knowledge of assembly programming.
- Slow Deployment | Because of the visual nature of ladder logic, it takes a programmer longer to create the logic they've envisioned. There's a need to drag and drop elements that slow down the development process versus other modern programming languages.
- Unintuitive for Complex Applications | Ladder Logic shines when it comes to sequential boolean tasks. However, when it comes to modern control theory that involves PIDs, flow control, analog sensors, and feedback loops, it's not always easy to implement and decipher.

Ladder Logic is the most used PLC programming language around the world. It's easy to work with and maintain for those who don't have constant exposure to PLC programming.

3.2 | Structured Text

Structured Text is a PLC programming language that closely resembles C or assembly. The user enters lines of code that execute sequentially, evaluate specific functions, boolean checks, and energize appropriate outputs of the PLC. Structured Text provides a simple transition into PLCs for those who have a background in a traditional programming language such as C, C++, Java, or Python. Furthermore, it can be easily manipulated in text processors, thus making it fast to implement without the need for hardware.



Structured Text PLC Programming | Example in Studio 5000 CompactLogix PLC

Advantages of Structured Text PLC Programming

- **Intuitive to Other Programming Languages** | As mentioned above, Structured Text is easy to learn for those who are looking to transition from a software engineering background. It features the same structures, programming paradigms, and functions that one would expect to see in C or Java.

- **High Complexity** | Structured Text allows for greater flexibility than other languages and thus makes it easier to implement advanced functionality for those who master the language.
- **Transferability** | Structured Text is standardized among most PLC systems, thus making it easy to migrate between platforms. You'll find significant differences in other languages between platforms, yet structured Text can be implemented in hardware and software platforms.

Disadvantages of Structured Text

- **Difficult to Troubleshoot** | When compared to ladder logic programming, structured Text is much more complex from a troubleshooting standpoint. There are no visual queues, fewer visual aids, and typically more code on a single line. Those who aren't comfortable with this language will have a hard time figuring out the process flow.
- **Error-Prone** | Structured Text provides greater flexibility to the user. However, this flexibility comes at the cost of standardization. Users must use software engineering best practices to create safe fallbacks and trap any potential failures of the software.

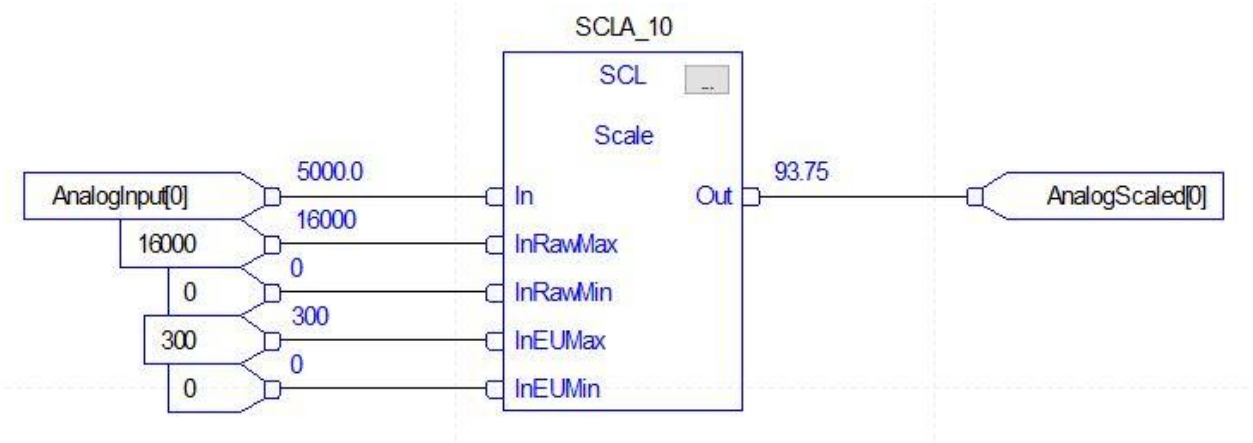
We typically recommend that you learn Structured Text only after you've mastered Ladder Logic unless you have a background in another programming language. It's not often seen in production environments due to the drawbacks mentioned above. However, it's an excellent way to manipulate data, implement FOR loops and other structures that require extra steps in Ladder Logic.

3.3 | Function Block Diagrams

Function Block Diagram, or FBD, is a programming language developed with chemical processes in mind. It allows the user to create a visual representation and flow of the process with appropriate transitions between the instructions. The visual

editor is user-friendly, intuitive, and creates a natural way to implement specific flows.

The most common application we've used for Function Block Diagrams in our PLC programs is to establish PID controllers. The visual aspect of FBD makes the PID easy to implement, visualize, tune, and troubleshoot in the field.



Function Block Diagram PLC Programming | Analog Signal Scaling Example in RSLogix 5000

Advantages of Function Block Diagrams PLC Programming

- **Flexible Visual Editor** | The editor for Function Block Diagram programming is very user-friendly and provides a simple way to create any layout.
- **Ideal for Complex Programming Structures** | In ladder logic, the user will have to use multiple rungs for what's possible to accomplish on a single page of FBD. The instructions can be brought directly into complicated PLC Instructions that implement PID loops, Motion Control, and Add-On-Instructions (AOIs).
- **User Friendly** | The visual editor of FBD comes naturally for most users. The layout of the process can be re-created through a drag-and-drop methodology that leaves little to guesswork.

Disadvantages of Structured Text

- **Hard to Standardize** | Due to the flexibility in the layout, it's challenging to standardize programs written in FBD. Each PLC programmer will have an approach that's different from others. Those who come behind will have a hard time understanding the flow of information.
- **Troublesome at Scale** | FBD shines when it comes to small implementations of specific areas of a process. However, as the program becomes complex, it's easy to get lost in all the sheets.

Function Block Diagrams are critical in analog scaling, PID loops, and Motion Control sequences. As you learn about these topics, you should start exposing yourself to this type of PLC programming. Before then, we would recommend mastering ladder logic.

3.4 | Sequential Function Charts

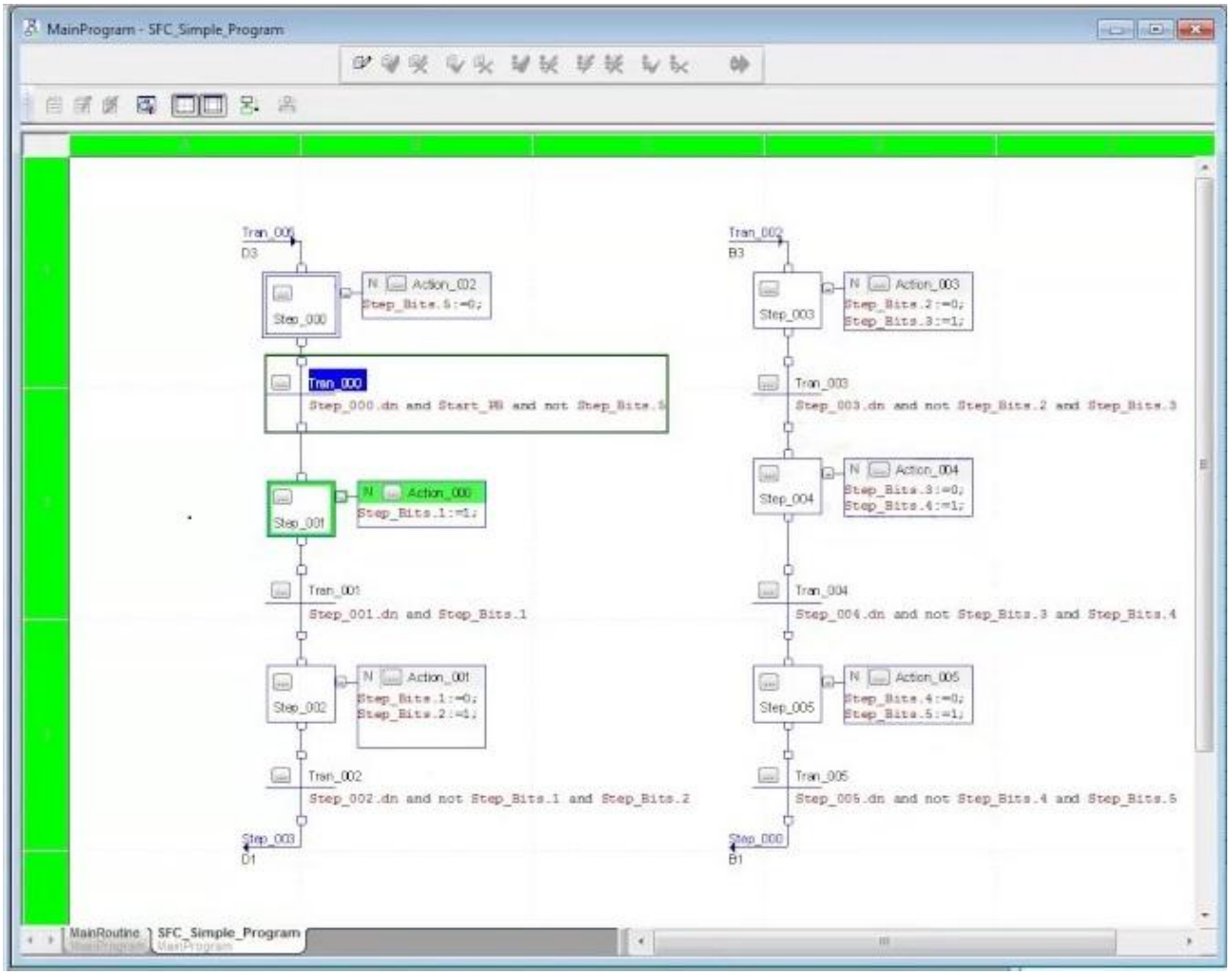
As the name implies, sequential function charts, or SFC, shine when it comes to a subsequent process. For those who aren't familiar with this notion, an example would be a chemical transformation from raw materials into the finished product. Let's take a simple brewing process as an example.

Picture a large beer brewing facility with numerous tanks, valves, pressure sensors, heating elements, and a packaging section. When an operator initiates the production of a new batch, the process goes through the following sequence of steps. Note that these steps are simplified.

- **Step 1** - The system is verified for readiness. Are all the appropriate ingredients available? Are the tanks empty? Are the valves in the right state? If the answer is valid to all checks, proceed. If not, abort.

- **Step 2** - Initiate a tank filling sequence that may call on multiple ingredients (water, sugar, salt, yeast, etc.). Validate the state and proceed once the tank is full.
- **Step 3** - Initiate the brewing process. Raise and maintain the temperature for a specified period. Monitor tank pressure and react accordingly. Add ingredients if necessary. Proceed to the next step after brewing is complete.
- **Step 4** - Initiate transfer to holding tank. Our batch is ready; verify that all the appropriate valves are set to the right position, the holding tank is empty, and begin the transfer process.
- **Step 5** - Transfer the batch to the bottling facility.

As you can see from the example outline above, the process steps are executed in a sequence, have defined start conditions and flow as the process would run in the production facility. In ladder logic, this process can be implemented through an SQI/SQO Instruction. However, a better approach would be to utilize SFC.



Sequential Function Charts PLC Programming | Sequential Process Example in RSLogix 5000

Advantages of Sequential Function Charts PLC Programming

- **Mimic Process Flows of Most Chemical Processes** | Batching is a common chemical process approach that takes a set number of raw ingredients and transforms them into the final product. SFCs shine in these applications.
- **Combined with ST** | Most SFC editors allow the use of Structured Text in specific cases to create advanced logic flows.

Disadvantages of Sequential Function Charts

- **Inapplicable in Most Applications** | It's challenging to apply sequential function charts to a process that isn't sequential. In other words, it has a limited number of use cases.
- **Parallel Flows are Difficult to Implement and Troubleshoot** | You may implement an unlimited amount of process flows through SFCs. However, as the process paths split into multiple flows, it becomes difficult to implement separate flow paths that would result in a robust sequence.

Sequential Function Charts are extremely useful in specific cases. However, trying to fit this type of programming language into a case that isn't sequential quickly leads to frustration. As you work in a manufacturing environment, we recommend that you become familiar with the process, understand the flow of the product, and seek to build a model on paper before diving into SFC programming.

3.5 | Instruction Lists

Instruction Lists are often confused with Structured Text due to their similar editors. These two PLC programming languages are typically seen on different platforms as their flow is similar. For example, Codesys-based controllers would allow users to implement logic in Instruction Lists, while RSLogix 5000 based controllers only have access to Structured Text.

In terms of program flow, each line specifies instruction as well as the conditions and outcomes of the execution. In many aspects, Instruction Lists are closer to how you'd implement ladder logic programs than structured text. However, either language is capable of creating the same process flow.

Advantages of Instruction Lists PLC Programming

- **Highly Standardized** | Instruction Lists follow a tight structure that requires the user to create variables explicitly, specify conditions, and list every instruction. There's little variation between program implementation which leads to easy-to-understand code.
- **Instruction Focused** | As the name would suggest, there's a high level of importance placed on instructions rather than data flow. This style of programming creates a level of clarity of how data is processed in the program.

Disadvantages of Instruction Lists

- **Unavailable on Most PLC Platforms** | As mentioned above, Instruction Lists aren't a popular method of programming as they come unnaturally to most programmers. They're closer to what one would see in assembly rather than any other programming language on the market

As we mentioned earlier, **we recommend that every PLC programmer starts with ladder logic** as this method is most common in the industry. That being said, as you get exposed to advanced programming methodologies, it's important to learn the other languages that may provide an easier way to implement specific processes.

CHAPTER 4 | INTRODUCTION TO LADDER LOGIC

4.1 | How to read Ladder Logic

Ladder Logic is one of the top 5 most popular types of PLC programming languages used in manufacturing environments. Before Programmable Logic Controllers, manufacturing plants employed relay-based circuitry to energize different loads based on how the relays were wired together. Relays were costly, required constant maintenance, and could not be easily reconfigured. As PLCs took over this process, it was essential to keep a similarity to the old system; thus, ladder logic was created as the first PLC programming language.

Ladder Logic is labeled as such because the software is laid out in the shape of a ladder. On the left side, ladder logic instructions are set as conditions, while the ones on the right side are instructions that are triggered if the conditions are met. Each rung of the ladder spans from left to right and is executed from top to bottom by the PLC.

As mentioned above, ladder logic is extremely popular among PLC programmers. It's easy to learn, mimics electrical circuits, and is easy to troubleshoot once deployed.

Learning ladder logic is typically the entry point into a career in control systems as a PLC programmer. In this post, we will go over ladder logic components, cover basic principles, and outline what it takes to master this programming language.

Ladder Logic Basics

Just like computers, PLCs operate with binary signals; each one can be set to zero or one. In the programming world, this data type is called a boolean. A boolean takes a

single bit in the memory, can be set to 0 or 1, and is used in most basic PLC instructions.

The PLC executes the program loaded into it one rung at a time. As the PLC begins to process the rung, it reads the instructions on the left and determines if the Logic on that side of the rung is set to TRUE. The Logic evaluates to TRUE when a hypothetical current is able to pass through the instructions. Each instruction has a set of conditions that make it TRUE or FALSE.

> 💡 **[IMPORTANT NOTE]**
>
> In Ladder Logic, instructions are also referred to as symbols or bit logic instructions.

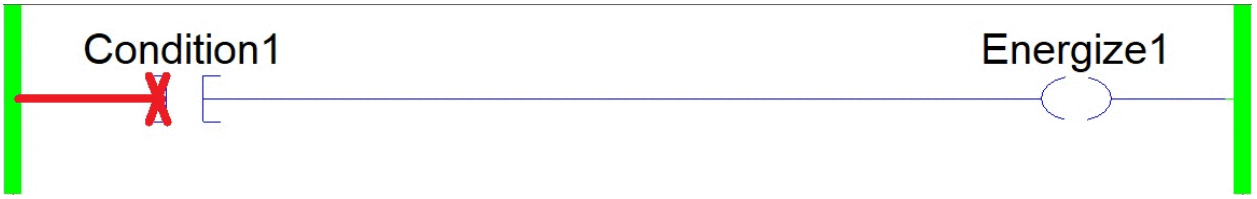
We'll start with two of the most basic instructions in ladder logic plc programming: **Examine if Closed** and **Output Energize**.

Examine If Closed [XIC] - This input instruction will look at the specified boolean bit and evaluate the condition to TRUE when the bit is set to 1 (or HIGH). While the bit is set to 0 (or LOW), the instruction evaluates to FALSE.

Output Energize [OTE] - This output instruction will set the specified bit to 1 (or HIGH) if the input instruction conditions are TRUE. If they're FALSE, the Output Energize instruction will set the bit to 0 (or LOW).

Basic Ladder Logic Rung Analysis

- **Step 1** - The hypothetical current starts moving from left to right.
- **Step 2** - When the hypothetical current encounters an XIC Instruction, it checks if the condition is TRUE or FALSE. If the XIC is False, the PLC aborts this rung.
- **Step 3** - The hypothetical current goes to the next instruction. Repeats Step 2 until the rung is completed.
- **Step 4** - The PLC moves to the rung below.



Ladder Logic PLC Programming XIC = OFF Example

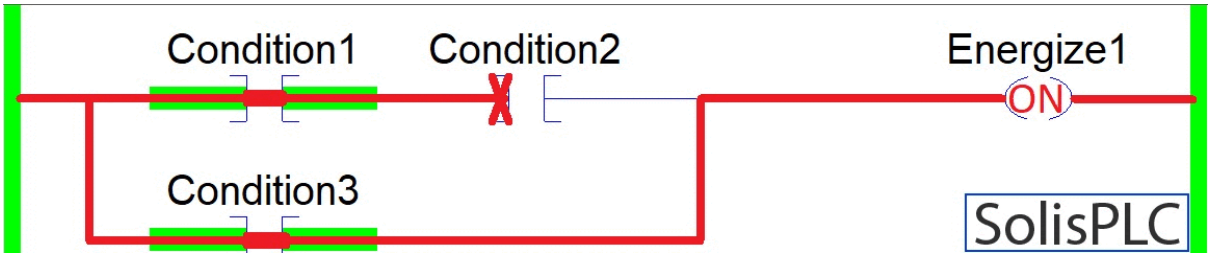
In the example above, the XIC Instruction is tied to the bit “Condition1”. Since the bit is OFF (or 0), the hypothetical current stops at the instruction.

Ladder Logic PLC Programming XIC = ON Example

In the example above, the XIC Instruction is tied to the bit “Condition1”. Since the bit is ON (or 1), the hypothetical current is allowed to pass through and go to the OTE Instruction. The OTE Instruction sets the “Energize1” bit to HIGH (or 1).

Ladder Logic Structure | Circuit Branches

Now that we've seen a basic example that illustrates how the execution of a single ladder logic rung is completed, it's time to discuss circuit branches. Circuit branches create a way for the current to pass through a different path as the rung executes. The instructions are executed in the same way, but we now need to analyze different paths the current may take.

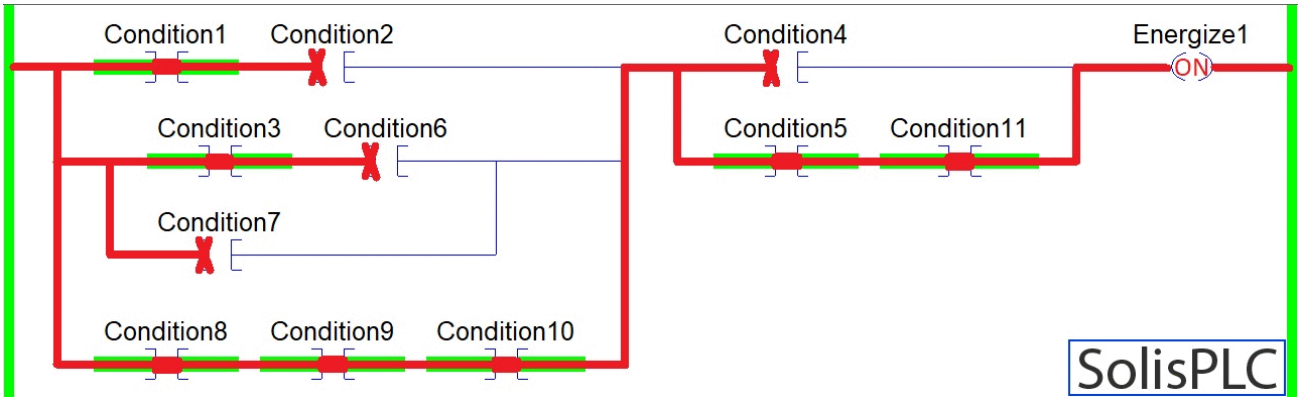


Ladder Logic PLC Programming Circuit Branch Example

The rung above has the primary rung and a branch that jumps the first two conditions with a 3rd one. Let's analyze what's happening with the execution of the Logic.

- **Step 1** - The hypothetical current starts on the main branch of the rung. As it reaches "Condition1", it evaluates the XIC Instruction. The XIC Instruction is TRUE and allows the current to proceed.
- **Step 2** - The hypothetical current flows to the next XIC Instruction and attempts to evaluate it. Since "Condition2" is set to 0, the XIC Instruction evaluates to FALSE. The current is stopped.
- **Step 3** - The hypothetical current goes back to the first branch. The XIC Instruction tied to bit "Condition3" is executed. Since the "Condition3" bit is HIGH, the XIC evaluates to TRUE. The current proceeds.
- **Step 4** - The current reaches the OTE Instruction and sets the "Energize1" bit to ON (or HIGH).

Here's a much more complex example for you to consider. It's not abnormal to find multiple branched circuits in ladder logic.Ladder Logic PLC Programming Circuit Branch Advanced Example



Ladder Logic PLC Programming Circuit Branch Advanced Example

Advanced Circuit Branching Ladder Logic Practice

Now that you're familiar with how circuit branches work in ladder logic, it's important to practice tracing the logic as you would in the field. Most of your work as a PLC programmer is going to be looking at rungs of logic and figuring out why the output is energized or what's preventing it from turning on.

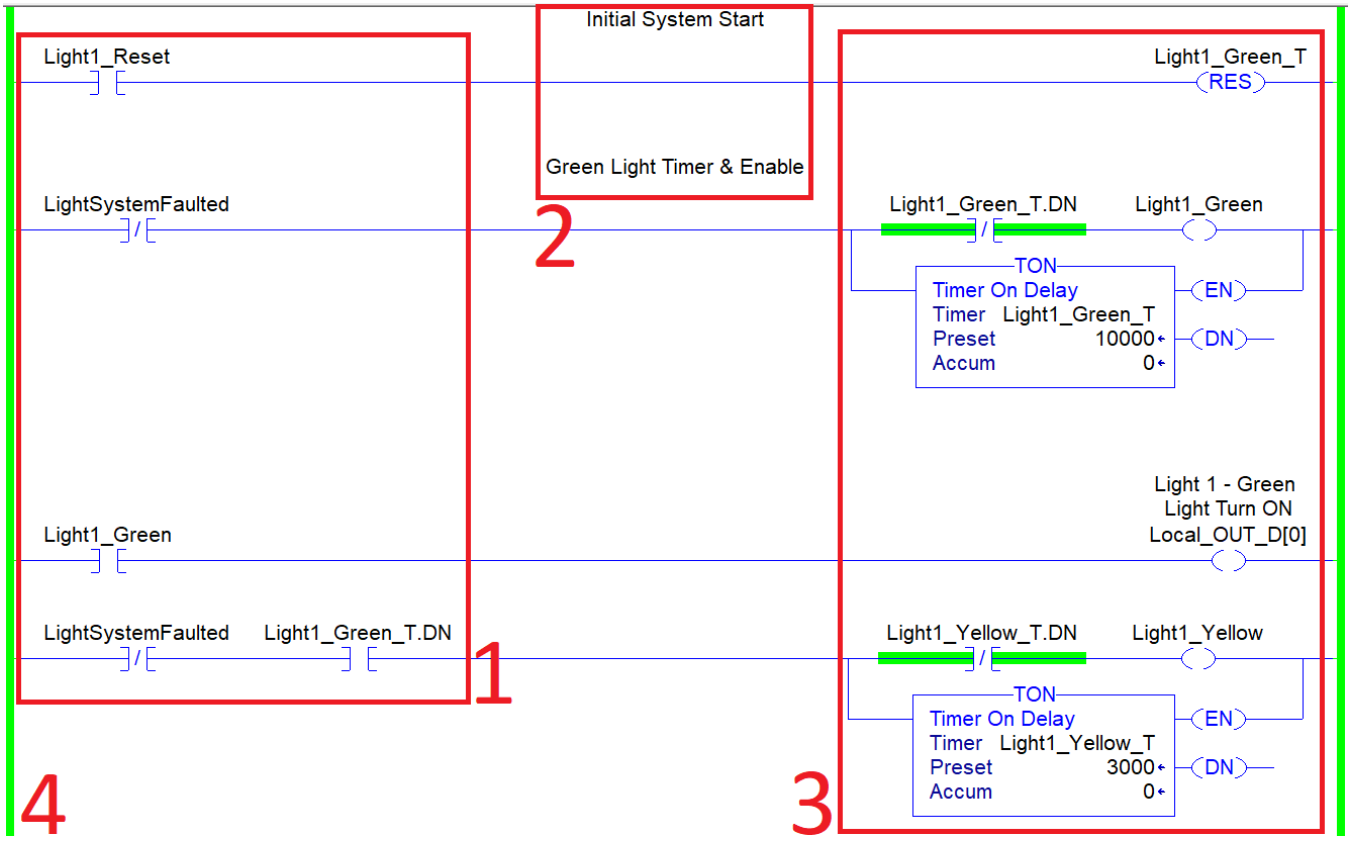
Consider the following situation: your supervisor calls you due to a problem on a production line. For some reason, the pump that's going to deliver raw materials to a specific tank isn't turning ON. As you show up at the operator station, he shows you that when he pushes the button, the pump won't do anything.

Resolution: you look at the panel, press the button yourself, and confirm that it doesn't start. This pump worked in the past, so you decide to see what's happening in the PLC logic. As you trace the output tied to the pump, you notice a complex rung with multiple circuit branches. The reason is that there are numerous conditions for that pump to start. Since you're familiar with the approach above, you can quickly figure out that the pump wasn't able to start because one of the start conditions was

that the tank must be empty. As you realized that the tank was, in fact, empty, the conclusion was that the level sensor was broken. You replaced the sensor, and the pump resumed regular operation.

Ladder Logic RSLogix 5000 Components

Now that we have some familiarity with how a basic rung structure is laid out, let's discuss other components of ladder logic.



Ladder Logic Components

1 - Ladder Logic Inputs

As we discussed above, the ladder logic instructions on the left side are called inputs. Their condition is evaluated on a true or false basis. If the evaluation is concluded with TRUE, the output of the ladder logic rung is executed. If it evaluates to FALSE, the PLC goes to the following rung.

2 - Ladder Logic Rung Comments

Every programming language allows the user to add documentation to their software. In ladder logic, this opportunity comes with every rung, instruction, and data structure. By adding a comment above the rung, you're making it easier for the person after you to understand your train of thought and troubleshoot the logic as needed. Furthermore, the comments may be used to indicate a change or temporary fix of a certain problem that was encountered by a PLC programmer.

3 - Ladder Logic Outputs

There are many instructions that will execute on the output side. In the example we covered above, our focus was on the OTE Instruction. However, the screenshot above also includes TON or Timer On Delay Instructions. As you gain experience as a PLC programmer, you'll encounter and master additional instructions.

4 - Ladder Logic Rails

Each rung of ladder logic lies between the two side rails (just like a regular ladder). These rails energize each rung as they are executed. In the screenshot above, you can see two rails within the RSLogix / Studio 5000 environment. The rails remain grayed out until the main routine calls the program. In the screenshot, the rails are green, which means that this specific logic is being executed.

5 - Tag Names

Each instruction will be tied to one or more tags. Each tag requires a data structure element as well as a name or label. In the examples we looked at above, tags were labeled as "Condition1", "Condition2", "Condition3", etc. In production circumstances, tags would typically reflect the physical element they control or a set of PLC-based tags. For example, tags that control motors may have the label of MTR1_Start,

MTR2_Stop, MTR2_Status, etc. Furthermore, tags may also have a description that allows the user to give the tag a text-based description.

4.2 Ladder Logic Symbols

As you invest yourself in PLC Programming, you'll quickly realize that the list of different instructions available to you is vast.

However, assuming that you're here for the basics, let's discuss the most useful instructions you should start working with as you tackle industrial automation.

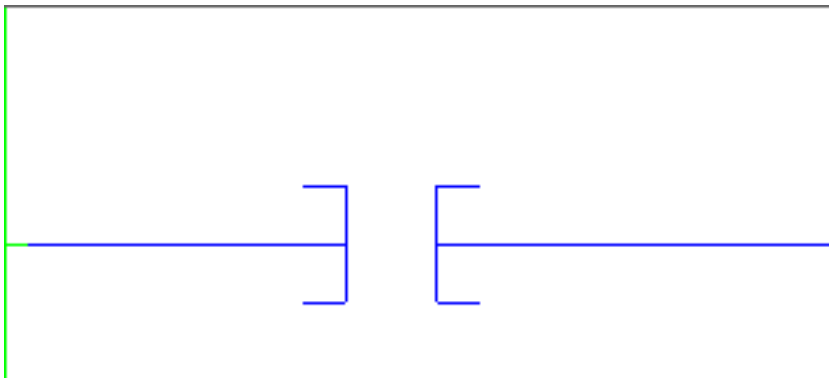
The standards of the language are well documented by the International Electromechanical Commission (IEC) in exhibit 61131-3. However, official documentation of the ladder logic symbols isn't easy to digest and doesn't provide concrete examples of each one.

Ladder Logic symbols are foundational elements that are memorized by every plc programmer. They're essential to know if you plan to do any work with this PLC programming language.

In the next few pages, we will discuss each symbol, the functionality it brings to the ladder logic plc programming language as well as illustrate two examples where they may be used.

Normally Open (NO) Contact / Examine if Closed (XIC)

The most fundamental symbol of ladder logic programming is the Normally Open Contact or the Examine If Closed XIC Instruction. This symbol was created as a direct reapplication of the relay-based contact used in early electrical drawings.



Ladder Logic Symbols - Examine if Closed Instruction (XIC) in Studio 5000

How does the Normally Open Contact Work?

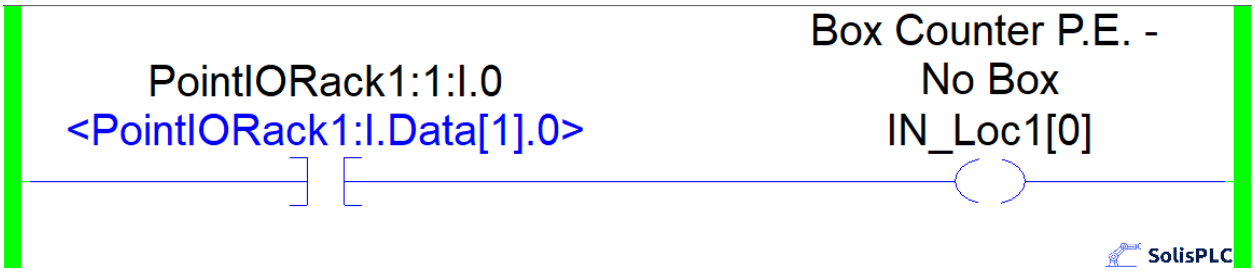
Initially, the contact was tied to a coil of an electrical relay. When the coil of the relay was energized, the contact would close. The ladder logic symbol operates in the same way. It will specify a logical bit that can be set to 0 (LOW) or 1 (HIGH). Based on the state, the instruction will either be TRUE or FALSE. If the instruction is TRUE, it will let the current through and allow the PLC to evaluate the next instruction. If it's FALSE, the ladder logic symbol will stop the execution there.

Practical Application of the Ladder Logic Symbol - NO Contact / XIC

The Normally Open Contact symbol is prevalent in ladder logic. It's the most basic logical check for most conditions in PLC programming.

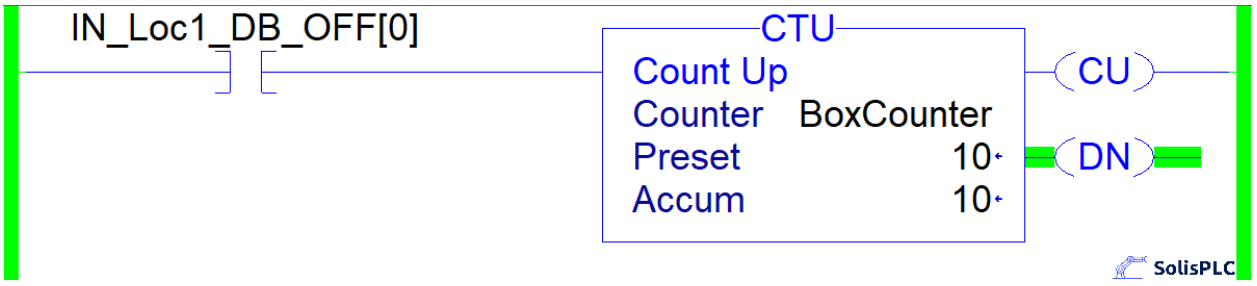
1. Verification of an Input

The rung above is using the Normally Open Contact to verify the “PointIORack1:1:I.0” input. If the input is energized (HIGH), the condition indicates that the “Box Counter Photo Eye - No Box” is turned ON. In other words, there’s no box in front of the Photo-Eye present on the line.



Ladder Logic Symbols - Examine if Closed (XIC) and Output Energize (OTE) instructions in Studio 5000

2. Count Up Condition



Ladder Logic Symbols - Examine if Closed (XIC) and Count Up (CTU) instructions in Studio 5000

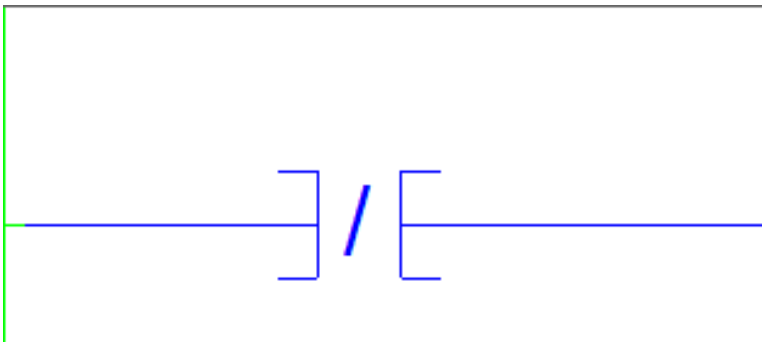
The rung above is using the Normally Open Contact to enable the “BoxCounter” CTU instruction. Each time that the NO Contact transitions from LOW to HIGH, the counter will increment by 1. As shown in the rung, the counter has counted ten boxes and is now set to the DN (Done) condition.

💎 **[BONUS]**

Here is a video explaining the XIC instruction in-depth. **Watch the video**

Normally Closed (NC) Contact / Examine if Open (XIO)

The opposite of the Normally Open Contact is the Normally Closed. This validation will look at the specified bit and evaluate to TRUE when the bit is de-energized and FALSE when it's energized. The application would allow the user to check if the coil of the specified bit is de-energized and take appropriate action in ladder logic PLC programming.



Ladder Logic Symbols - Examine if Open (XIO) instruction in Studio 5000

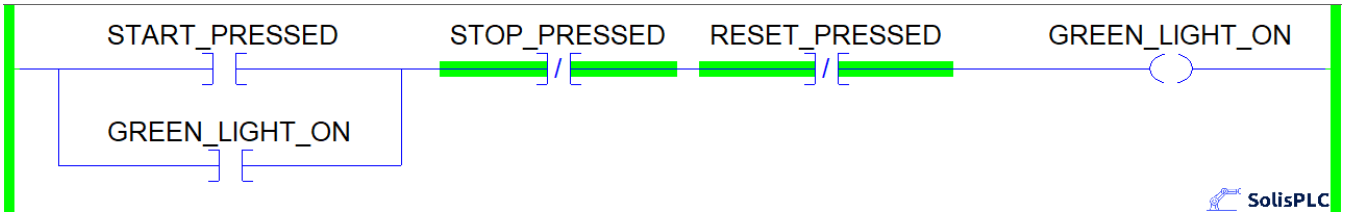
How does the Normally Closed Contact Work?

The normally closed contact would also be tied to the coil of a solid-state relay. When the coil has no current running through it, the contact would let current flow through. However, when the coil would be energized, no current would flow through the contact. The NC Contact or the XIO instruction in PLC programming ladder logic would work the same way. In other words, the bit would allow the current to flow through when it's LOW and no current would flow through when the bit is HIGH.

Practical Application of the Ladder Logic Symbol - NC Contact / XIO

The XIO is very common in the ladder logic plc programming language. It's an instruction that allows us to examine the OFF state of a bit as described above. Here are two common examples of where this instruction is used.

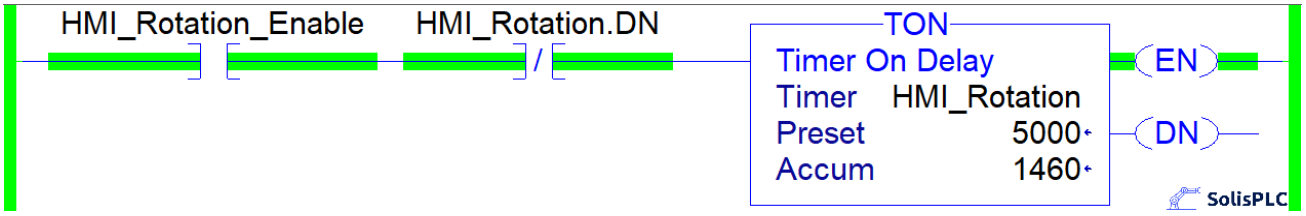
1. Stop Button Condition



Ladder Logic Symbols - Motor Starter Seal In Logic in Studio 5000

The rung above incorporates the normally open and normally closed ladder logic symbols. It creates a condition that will energize the GREEN_LIGHT_ON bit when the "START_PRESSED" is energized. However, the XIO is tied to two bits: STOP_PRESSED and RESET_PRESSED. When either one of these conditions is set to HIGH, the "GREEN_LIGHT_ON" bit will be set to LOW during the rung evaluation cycle.

2. Timer Continuous Latch



Ladder Logic Symbols - Continuous Timer Logic in Studio 5000

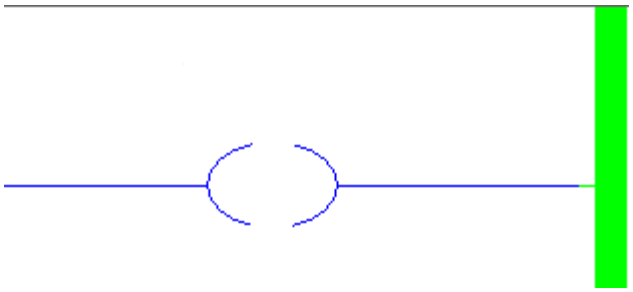
The rung above will allow the Timer to operate based on the HMI_Rotation_Enable condition. However, a typical timer would count until it reaches the "Preset" value. In the rung above, the timer will reset once the timer is set to DN (Done) due to the XIO being tied to the same bit of the timer.

💎 **[BONUS]**

Here is a video explaining the XIO instruction in-depth. [Watch the video]

Output Energize (OTE)

When certain conditions are met, the system should take a certain action. Unlike the two symbols above, the output energize will be used to execute an action. Within the scope of an electrical diagram, this symbol would indicate that a coil of a relay needs to be energized when conditions are met.



Ladder Logic Symbols - Output Energize (OTE) instruction in Studio 5000

How does the Output Energize Symbol Work?

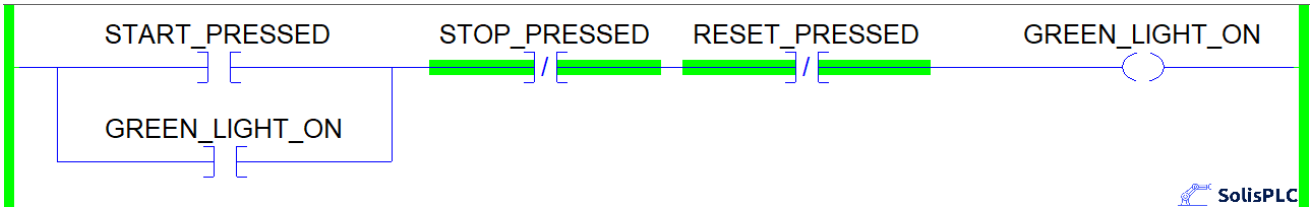
The output energize ladder logic symbol will change the state of a bit based on the conditions specified on the left side of the rung. When the conditions are TRUE leading to the OTE instruction, the value of the specified bit will be set to HIGH or 1. When the conditions are FALSE, the OTE instruction will set the value of the same bit to LOW or 0.

Practical Application of the Ladder Logic Symbol - Output Energize

The OTE instruction is very common in ladder logic applications. As mentioned above, it's used to drive outputs based on certain conditions. This translates to

operating external PLC hardware such as relays, motor contactors, valves, cylinders, etc. By energizing the bit that is tied to the output, a PLC programmer can change the state of the output to the desired position.

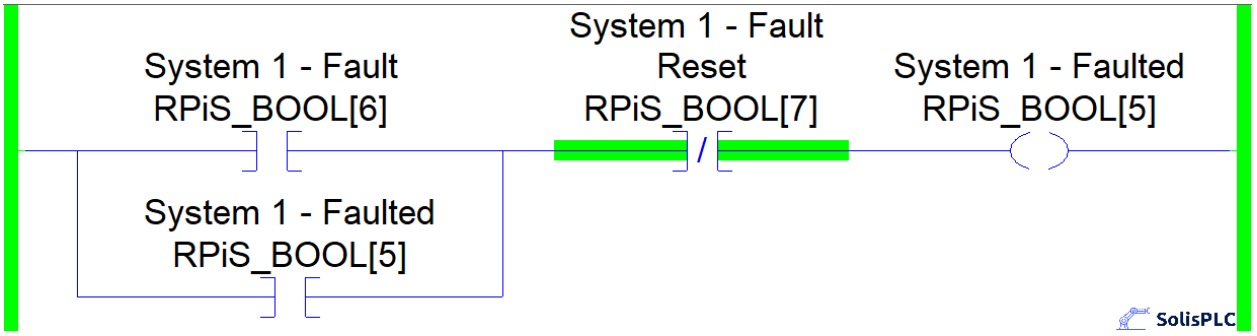
1. Turn ON Light / Output



Ladder Logic Symbols - Light Seal In Logic in Studio 5000

In the rung above that we've already seen, the output is energized when the conditions are met. The "GREEN_LIGHT_ON" bit is tied to an output of the PLC that will turn on an LED in the field. By using the Output Energize (OTE) Instruction, the PLC programmer will turn ON the light on the plant floor.

2. Set the System to Faulted State



Ladder Logic Symbols - Fault Logic in Studio 5000

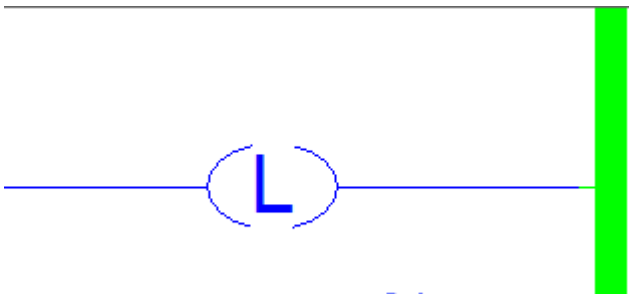
The following rung verifies one faulted condition: System 1 - Fault. When the system is faulted for that specific reason, the "RPiS_BOOL[5]" bit will be set to HIGH through the Output Energize (OTE) Instruction. Once the system is no longer faulted, the faulted status will remain ON until the Reset button is energized and validated

through the XIO condition. The Reset will allow the OTE instruction to clear the bit and set the faulted state back to LOW.

> 💎 **[BONUS]**
> Here is a video explaining the OTE instruction in-depth. **Watch the video**

Output Latch (OTL)

The Output Latch ladder logic symbol is not something that can be created with relay-based logic. This instruction will permanently keep a bit set to 1 when the condition holds.



Ladder Logic Symbols - Output Latch (OTL) instruction in Studio 5000

How does the Output Latch Symbol Work?

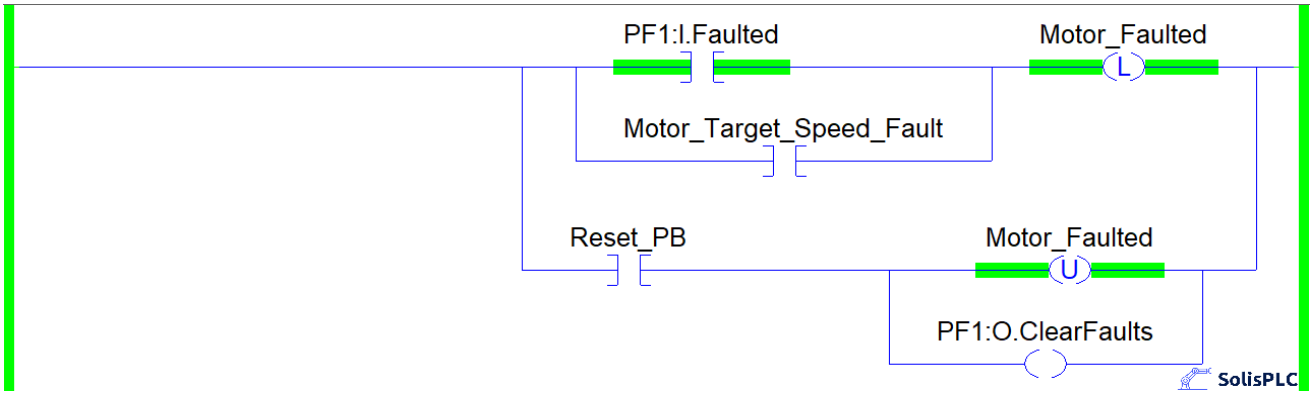
The output latch instruction will execute only when the preceding conditions are TRUE. If they are, the instruction will set the bit associated with the OTL to HIGH (1). If the bit is set to 1 or the conditions are no longer true, the bit will remain HIGH (1). This difference is important as the Output Energize (OTE) will set the bit back to 0.

Practical Application of the Ladder Logic Symbol - Output Latch

The OTL instruction is not commonly used in ladder logic programming. The reason is mentioned above: the instruction will not automatically reset the bit back to 0. This

minor difference leads to code confusion and potential issues when it comes to executing, changing, or evaluating conditions after implementation.

1. Fault Latching

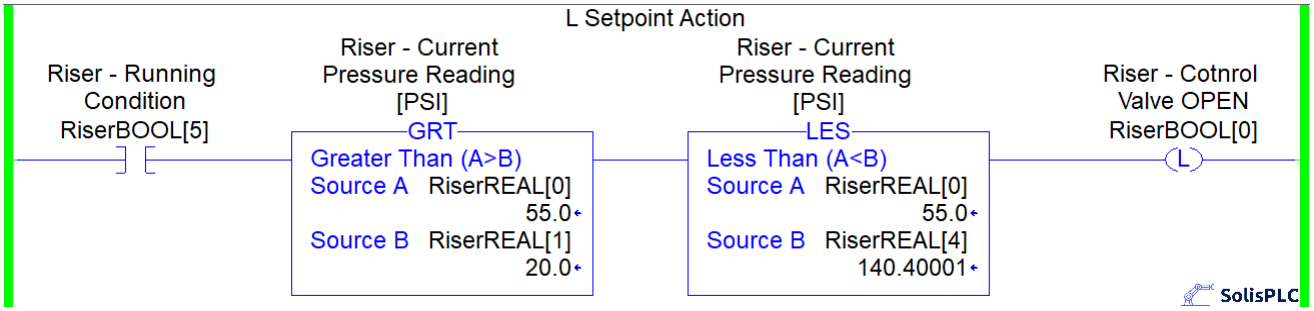


Ladder Logic Symbols - Fault Latching Logic in Studio 5000

As we discussed before, faults play a critical role in PLC programming. It's important to properly detect, act upon and identify the faults occurring within the system. Once they occur, the user will throw the faults to the operator in order to troubleshoot. For that reason, it's important to keep the faults in place until the system is audited and reset when deemed operational.

The rung above displays a condition in which we are required to clear a fault on a variable frequency drive PowerFlex 525. Once the fault is latched, the motor is kept in a faulted state while a separate routine takes care of safely stopping the drive. The OTL will set the bit to HIGH and wait until the fault is reset.

2. Condition Setting



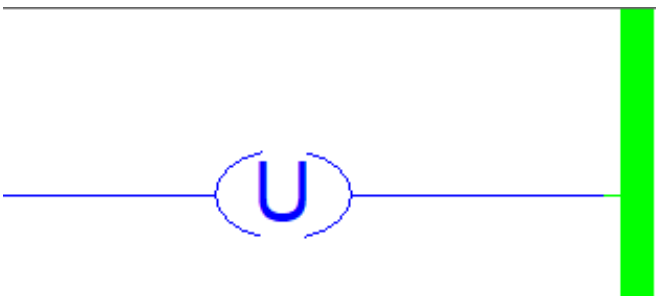
Ladder Logic Symbols - Conditional Instructions Greater Than (GRT) and Less Than (LES) in Studio 5000

In the rung above, the OTL instruction is used to open the valve of the riser. Although this could have been achieved through an output energize (OTE) instruction, we've decided to use the OTL due to a number of conditions that may set the bit RiserBOOL[0] to HIGH. Note that this routine also contains the OTU that will reset the bit back to LOW as needed by the PLC programmer.

> 💎 **[BONUS]**
>
> Here is a video explaining the OTL instruction in-depth. **Watch the video**

Output Unlatch (OTU)

The Output Unlatch ladder logic symbol is often used in conjunction with the OTL. It's a way to create a disable of the bit specified within the logic of the controller.



Ladder Logic Symbols - Output Unlatch (OTU) instruction in Studio 5000

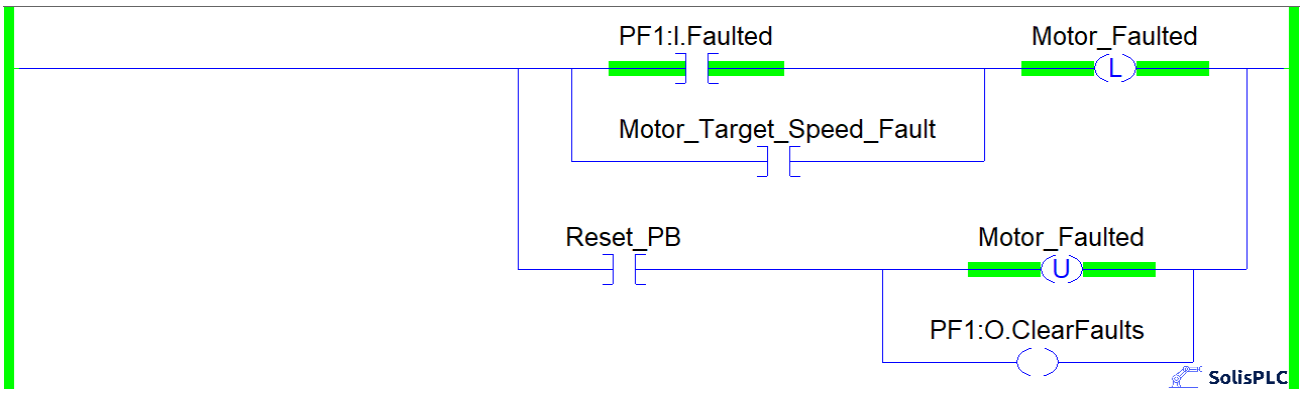
How does the Output Unlatch Symbol Work?

The output unlatch instruction will execute only when the preceding conditions are TRUE. If they are, the instruction will set the bit associated with the OTU to LOW (0). If the bit is set to 0 or the conditions are no longer true, the bit will remain LOW (0).

Practical Application of the Ladder Logic Symbol - Output Unlatch

The OTU instruction will have to be used with the OTL in order to reset the bit back to LOW as discussed above. Therefore, this instruction will always be found whenever the OTL is used. Let's examine the same two examples, as we saw above.

1. Fault Latching



Ladder Logic Symbols - Fault Latching Logic in Studio 5000

In the rung above, once the fault is cleared through the Reset_PB XIC instruction, the fault is unlatched using the OTU instruction. Note that the unlatch is within the same branch as the PF1:O.ClearFaults instruction will be energized once the reset is set.

> 💎 **[BONUS]**
> Here is a video explaining the OTU instruction in-depth. **Watch the video**

There you have it. The five most used ladder logic symbols are:

1. Normally Open Contact
2. Normally Closed Contact
3. Output Energize
4. Output Latch
5. Output Unlatch.

These five instructions are commonly used in ladder logic for bit manipulation. The first two are conditional instructions that will allow the current to flow depending on the status of the bit. The last three are output instructions that will execute if the

logic leading to them is TRUE. They will set the bit to either 0 or 1 depending on the instruction used.

Ladder Logic is the most common PLC Programming language. It's easy to learn, easy to use, and has been adopted since the early days of Programmable Logic Controllers. The iconic resemblance to a ladder was what gave this type of logic its name. Such diagrams were used for specifying electrical drawings that were used in many industrial environments. Since those days, ladder logic has been involved significantly, yet retains some of the basic elements: rails, rungs, input conditions, output instructions, comments, etc.

To learn ladder logic, you'll need to start with understanding the current flow from the left rail to the right one. In short, the current will attempt to flow through one rung at a time. As it encounters an input condition, it evaluates the result to TRUE or FALSE. If the condition is FALSE, the current will attempt to use a secondary path which may be through a circuit branch. If the result is TRUE, the current will proceed through the instruction. When it reaches an output instruction, it will execute the specified logic.

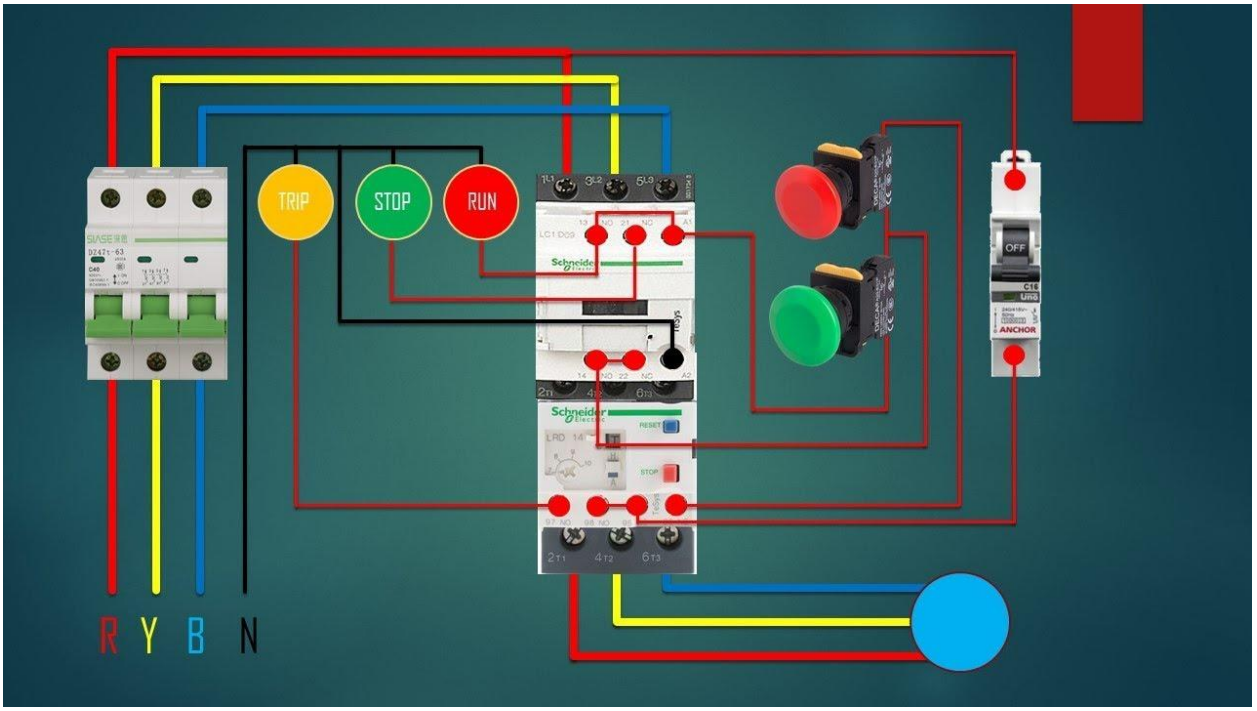
CHAPTER 5 | INTERVIEW PRACTICE PROJECTS

PLC Interviews aren't very complex. Within a short period of time, you may be asked to implement a simple logical structure such as a Motor Starter, a stack light system or a basic FIFO set of rungs. By practicing implementing such a structure, you will build your knowledge of PLCs, the software package of your choice and you'll become a more proficient programmer.

5.1 | Building a 2 Button Motor Starter System

One of the most simple, yet challenging questions for a beginner, is the famous motor starter. A simple system that utilizes two buttons and an output stumbles those who've little to no experience with PLCs. Furthermore, this question will reveal if the candidate asks the right questions, makes the correct assumptions, and is able to reason through a simple assignment presented to him.

Although it is possible to wire a motor starter to work without the use of a PLC. There are many benefits to wiring the inputs and outputs into the controller. Once the inputs and outputs are established, a PLC programmer will create a ladder logic-based routine that would accomplish what the following circuit was intended to create in hardware.



Three Phase Motor Starter Circuit with Push Buttons and Contactor

Before you dive into programming, it is important to understand the operation of the circuit above. Here are the key components and stages of operation:

- **3 Phase Circuit Overload** - Each phase is protected by an overload that will trip as high current flows through due to a motor or circuit fault.
- **Motor Contactor** - The contactor acts as a bridge between the high voltage (motor) and low voltage (control (24VDC)) circuits. When the control circuit is energized, the power circuit is allowed to conduct the necessary current.
- **Start Push Button** - When pressed, the contactor is allowed to energize. The current flows through and the motor starts to operate.
- **Stop Push Button** - When pressed, the contactor is unlatched and stops conducting current which stops the motor.

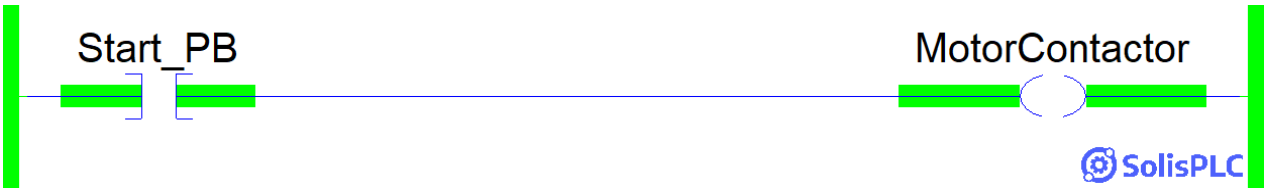
Although the circuit is straightforward, there is one key feature: the momentary pushbuttons used for starting and stopping the motor latch in the contactor. In other words, once the momentary Start push button is pressed, the motor will continue to

run until a Stop command is issued. In the industry, this is referred to as a latch circuit. This scheme is used in many applications including machinery starters, conveyors, process initiation, and more.

The momentary push buttons are wired into the digital inputs of a PLC. When either button is pressed, the appropriate input bits are set to HIGH (1). When the buttons are released, the same input bits are set to LOW (0).

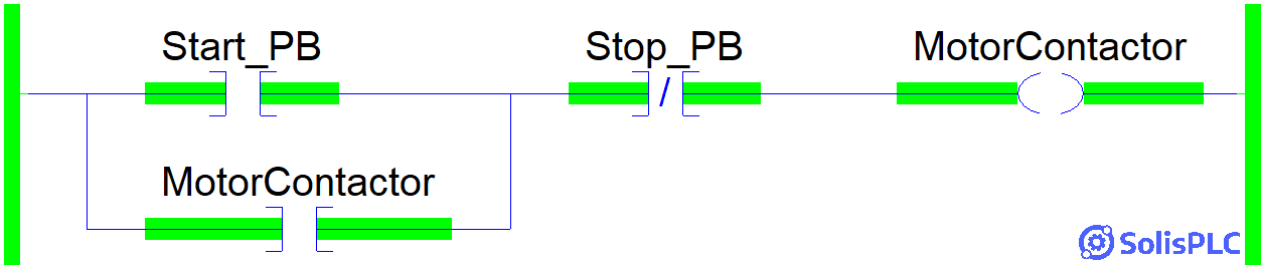
The motor is tied to an output when set to HIGH (1), energizes the coil in the contactor, and allows the current to flow.

Based on the above, we start by building a circuit that will run the motor when the Start push button is pressed.



Ladder Logic Example - Motor Starter Part 1

The ladder logic above will take input through an XIC instruction and energize an output over an OTE instruction. However, a major problem is a fact that the user must keep the button pressed for the motor to run. Typically, we'd want the motor to keep running after the button has been released. Let's look at the second iteration of our ladder logic circuit.



Ladder Logic Example - Motor Starter Part 2

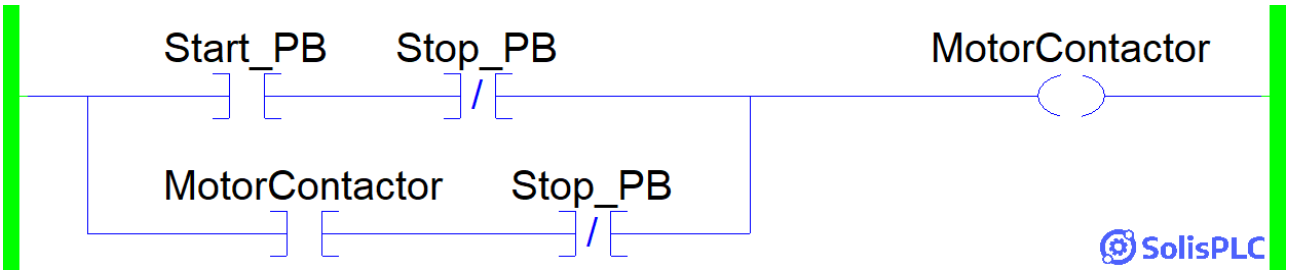
The second iteration of the ladder logic motor starter is able to start the motor and keep it running. However, we're now faced with another problem: there is no way to stop the motor.

The stop pushbutton needs to be integrated into the logic. However, let's take a moment to understand the operation of this button. It has to have two functions:

- The stop push button should prevent the motor from getting started.
- The stop push button should stop the motor when it's running.

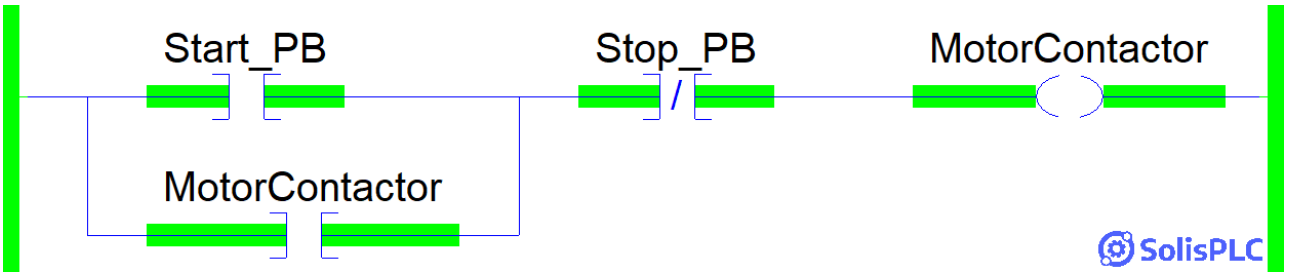
Based on the two requirements above, it's possible to add an XIO instruction into each branch of the circuit. However, ladder logic is such that the user can utilize a single instruction to cover both of those scenarios after the branch.

Example of a functionally sound rung based on the above requirements.

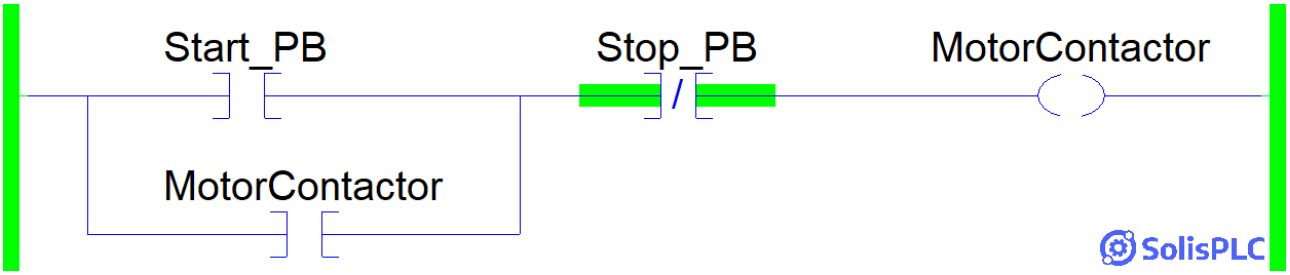


Ladder Logic Example - Motor Starter Part 3 (Not Optimized)

The rung above will operate as expected. However, it's important to create efficient ladder logic and utilize instructions in conjunction to the branch structures we've covered above.



Ladder Logic Example - Motor Starter Part 3 [Energized]



Ladder Logic Example - Motor Starter Part 3 [De-Energized]

The rung above operates as follows:

- **Stage 1 -** The Start_PB is pressed and the MotorContactor is Energized while Stop_PB is not pressed.
- **Stage 2 -** The MotorContactor bit is used to keep the motor energized while the Start_PB is released.
- **Stage 3** - The MotorContactor is de-energized when the Stop_PB is pressed.

The motor starter ladder logic example is easy to follow. However, as you expand your knowledge of ladder logic principles, you will create complex branches around similar circuits. It's not uncommon to have multiple stop conditions that are set in series with the "Stop_PB" bit. Similarly, it's common to see other sources of starting the motor. For example, a sequence may be used to start a specific pump through software.

Even the simplest interview questions require some thought & careful consideration. The 2 button motor starter question is one of the most fundamental interview questions you will encounter as a PLC programmer & control engineer or technician. The main purpose is to see that you're capable of asking the right questions, reasoning & writing basic ladder logic programs.

> 💎 **[BONUS]**
> Explore how the same program can be implemented in Structured Text.
> **Watch the video**

5.2 | FIFO Implementation Through FFL and FFU Instructions

As mentioned earlier, you could be asked during an interview to implement a basic FIFO set of rungs.

In this project, we're looking at an example that involves a FIFO Implementation and results in a way to store data based on short periods in order to compute a rolling average of such data. In other words, the logic can be used to report a rolling average of the number of products made in the last hour.

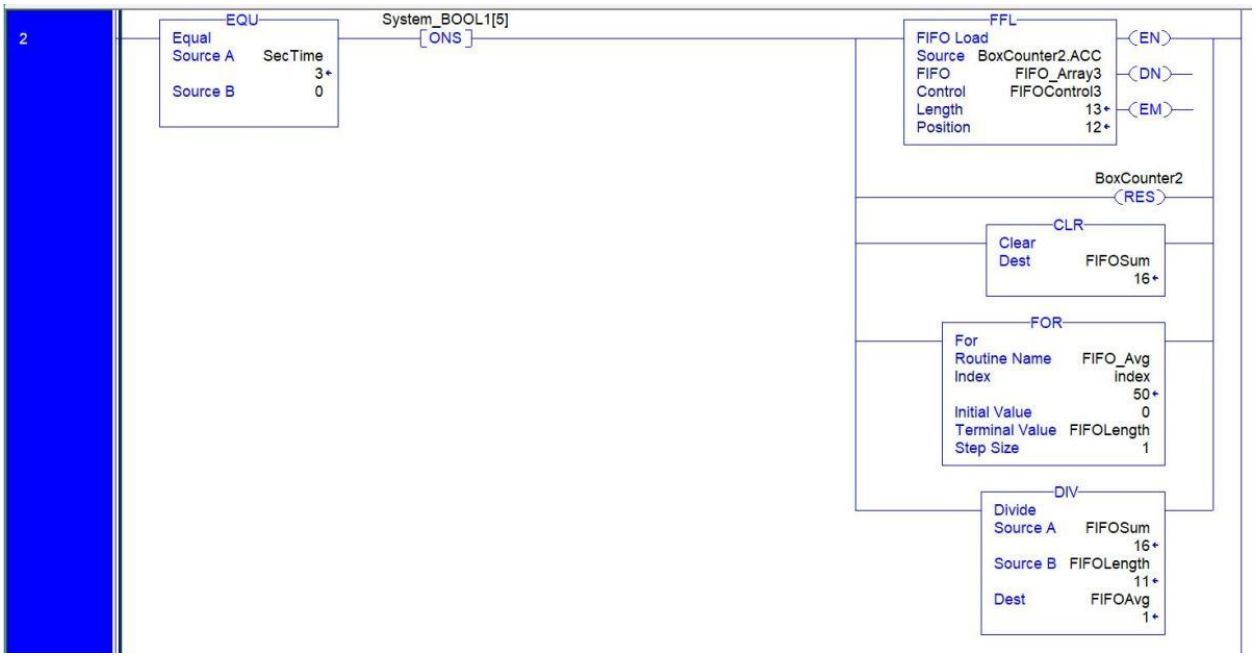
At first glance, it may seem that such systems already exist, but from our experience, most will report the quantity produced at every hour. The nuance here is that as you walk up to the production line at 7:43 AM, you should be able to see how many cases were made from 6:43 AM to 7:43 AM. A traditional system will give you the number of cases from 6 AM to 7 AM and sometimes a rolling number starting from 7 AM.

FIFO Logic in RSLogix 5000 PLC Programming

Since our logic will require a time-stamp, we need to leverage the internal clock of the PLC system. This is achieved through the first rung in our routine which can be seen below.

In “Rung 0”, we’re using two basic instructions which we’ve seen before (SUB Instruction, MOD Instruction). The SUB instruction is applied on the FIFO and allows us to scan the first 11 values. This will become relevant in a future rung. The MOD instruction is critical to breaking down a minute into 6 chunks of 10 seconds. This is achieved by dividing the seconds clock by 10 and taking the remainder. In other words, we’re truncating the seconds timer to give us the least significant digit at all times. This allows us to have a counter for each chunk of 10 seconds.

In “Rung 1”, we have a CTU Instruction that will count the number of boxes flagged by a certain sensor. This counter will increment as boxes go by and will be reset by the logic below every cycle. Note that this application can be repurposed for many different applications and the counter can be converted to a timer.



The FIFO relies on two critical instructions: FFL and FFU. The FFL will transfer an element into the array while shifting everything else by a specified step size. The FFU will remove an element at the end of the array into a specified tag. The rung above has many instructions; let's examine them in detail. The EQU instruction is used to execute the logic when the time is equal to 0. In other words, we want to record the number of cases every 10 seconds since this "SecTime" register will cycle from 0 to 9 and repeat.

The ONS instruction will perform a single cycle of the instruction. This means that we're only going to move the array once as the time is at zero. The FFL will load the element which is computed by the counter into the array. In this case, we're loading into the 12th element as the length specified is 13. We're also required to specify a control structure that is common between the FFL and the FFU instructions.

We have a few interesting instructions following the FFL: RES, CLR, FOR, and DIV. The RES will reset our counter. The FOR Instruction is used to compute a rolling sum. Lastly, the DIV instruction will give us an average by dividing the sum we compute by the number of time slots which results in a rolling average.

The last rung contains the FIFO Unload or the FFU Instruction. As briefly mentioned above, this instruction will remove a single element from the specified array and transfer it into a specified register. The FFU is required for a “rolling” effect of the FIFO implementation.

Understanding the Logic of a FIFO

The logic above contains few instructions, but may not seem obvious to new programmers. The goal is to count the number of boxes with a sensor, transfer this value into an array every ten seconds and compute the sum of the last 10 values. By using a FIFO, the system will automatically write the new counter into the array and remove the oldest value thus achieving the desired result.

Although we’re using the FIFO implementation for a specific problem, it’s extremely versatile and can be the solution to many challenges. It’s a common application for systems that have indexed movements. In other words, you’re producing a single product at a time and the system needs to track the location of said production. An inexperienced programmer may use a timer in this situation which will often fail as the system is started and stopped, is changed, etc.

Going through an example that utilizes a FIFO implementation through FFL and FFU instructions is an excellent step toward becoming a better programmer. The concepts outlined in this short piece of logic are straightforward yet often misunderstood by many programmers. As you implement your own FIFO, make sure to experiment with different ranges, values, and ways to store the data.

CHAPTER 6 | GETTING A PLC PROGRAMMING JOB

Getting a PLC programming job is ultimately why we're all here. We study all there is to know about the hardware and software in order to land a position that is challenging, satisfying, and financially rewarding. In this article, we're going to cover all you need to know about PLC programming jobs, what you can expect, what you should prepare for, and ultimately give you an edge over the other applicants.

6.1 | Entry Level PLC Programming Jobs

An entry-level PLC programming job is hard to come by. The reality is that it's difficult to master the hardware and software platforms without having some experience in the manufacturing setting. You will most likely notice that almost every "entry-level" job in this field will list at least a few years of experience under the requirements section. If you've completed a class, have a PLC that you've practiced on, and feel comfortable answering basic PLC programming questions, you should apply.

In the current market, employers are in dire need of programmers that may not have all the know-how but are willing to learn and are capable of demonstrating this capability during the interview. Furthermore, they understand that the required knowledge of hardware and software to fill these jobs is extremely difficult to come by.

What to Expect for Entry Level PLC Programming Jobs?

An employer that is looking to hire an entry-level PLC programmer will either ask for an electrical engineering or engineering technology degree. Both of these programs demonstrate a candidate's ability to learn as well as a certain degree of proficiency in electrical systems. However, it is possible to demonstrate the same knowledge through vocational training, an online course, or projects that you've built-in control

systems. In the last decade, employers have become much more open to non-traditional paths of learning; some may even prefer demonstrated projects over a college degree.

In addition to a degree, almost every listing you will encounter will have a platform listing. Depending on the region, this platform may include Allen Bradley, Siemens, Omron, Automation Direct, or other PLC brands. On some occasions, more than one brand will be listed. Although every employer would love to hire one person that could “do it all”, you should prepare to master a single platform. When asked about others, you may indicate that it’s something you’ve heard about on a regular basis and would be open to learning if given the opportunity.

What we Recommend

1. Apply to jobs for which you don’t always meet the “education” and “experience” requirements. Be ready to back up your lack in any of the above with projects, demonstrated learning materials, and knowledge.

2. Prepare to encounter questions about various platforms, but don’t expect or spend the time to master all of them. Most employers will close the gaps through on-the-job training once you’re hired.

6.2 | Transitioning into PLC Programming

A number of those who work in manufacturing are looking to switch direction in their career for PLC programming. Although it’s possible to find a job with a new employer, it’s generally easier to find positions within. However, we often receive questions about how this can be achieved.

If you work in manufacturing and are looking to land a PLC programming job within your existing company, you need to speak up. Take the time to speak to your

manager about the direction of your career and inquire about the opportunities you have. Most companies would rather promote from within than find outside talent, provided you can demonstrate competency and interest in the domain. The last point is extremely important as you should do your homework before you ask for the job. Be prepared to answer questions about why you want to shift, what you've done to learn the desired profession and how this shift will benefit the company in the long run.

Showcase Knowledge with Education

As you express an interest in PLC programming to your current or potential employer, keep in mind that they may need some tangible proof of your proficiency. PLC programming jobs do not require formal education, but your desire to enter the field can be backed by an accredited university degree, a certification, or personal projects. Although we don't believe that one is better than the other in every situation, you may want to research further each one of these paths and see what suits you.

What to Expect for PLC Programming Jobs within Your Company?

A position is always listed internally before it is open to the public. Having access to this list is critical if you're looking to move into PLC programming. Although it is possible to make a request while no position is open, it's much easier if there is one. The listing should be no different than what you'd see outside of the company. The advantage you have is that you may speak to the hiring manager directly and inquire about what's important for that department and clarify any confusion you may have.

What we Recommend

1. Prepare for the job before you make your intentions known. Be prepared to answer questions about the job, your desire to move departments, and what steps you have taken to learn the skills required.

2. Speak with your manager as well as the hiring manager who's looking to fill the position. Ask questions that express your interest. Take notes and make sure that you position yourself within the frames of what they're looking for. In other words, if you can find out that they're looking for an individual who's capable of managing small projects, prepare a few examples where you've demonstrated this ability.

6.3 | PLC Programming Job Interviews

What are some of the most commonly asked questions during a PLC programming interview? Although every company is different, we want to give you an idea of what to expect in an interview in a North American manufacturing company.

Step 1 - Phone HR Screening

After you've submitted your resume and have been successfully vetted by the hiring manager, you will receive a call from Human Resources. This individual will have access to your application as well as the job position. Their questions will typically be of general nature. You should expect to be able to answer the following:

- **Q:** Tell me about your experience with PLC programming.
- **Q:** Tell me about your experience in manufacturing.
- **Q:** Please describe your experience with Allen Bradley (or other platforms) PLCs and hardware.
- **Q:** We'd like to test your basic PLC programming understanding:
- What is an XIC instruction?
- Describe what a timer does within a PLC.
- **Q:** What are your salary expectations for this position?

How to Prepare

At this stage, you should prepare to answer general PLC programming questions, the position you're applying for, as well as your resume. You won't encounter any difficult questions at this stage; prepare to talk about yourself and your experience.

An important process of an HR interview is to demonstrate that you'll make a great addition to the company. Your goal is to engage in conversation with the HR interviewer by asking questions about the company, the role as well as the opportunity in general. Keep in mind that

Step 2 - Hiring Manager Phone Interview

A PLC Programming Job will screen every candidate for technical knowledge. If you've answered the HR questions correctly, you will be contacted by the hiring manager. The hiring manager will generally be your future boss and will need to know that you're capable of performing the tasks required. He will ask questions that will test your technical capabilities. You should expect the following:

- **Q:** Where have you studied PLC programming?
- **Q:** What other systems have you programmed or designed?
- **Q:** What other hardware are you familiar with?

How to Prepare

At this stage of the interview process, you've passed the initial interview. The company has selected you as well as 2-3 others for the next round with the hiring manager. Unlike an HR representative, the hiring manager is likely to be experienced in PLC systems, know about the engineering operations of the company, and be knowledgeable in the technical aspects of the job to be performed. He is likely to ask questions of technical nature in addition to what was already covered by the HR rep.

You should be ready to answer questions that will go in-depth on what you know and have done when it comes to PLC programming. A typical approach is to ask a general question about a project listed on your resume and to dig deeper into each component of the project. Be prepared to answer detailed questions about each project you've done.

Step 3 - On-Site Technical & Behavioural Interview

Once you check all the boxes above, you'll receive an invitation for an on-site interview. At this stage, you are likely to meet your future team, your boss, and see the manufacturing process. You should take this opportunity to learn about the company as much as impress them with your knowledge. It's hard to predict all the interview questions you may encounter. However, here are some questions you may encounter at this stage:

- **Q:** Tell us about your experience and knowledge of control systems.
- **Q:** Look at the wiring diagram and walk us through the circuit elements.
- **Q:** Tell us about the different brands of PLCs you're familiar with and what you like / dislike about them.
- **Q:** Write a simple PLC / HMI program to accomplish a task (start a motor, display an integer, toggle a function).

How to Prepare

An in-person interview is a formality. The team is convinced that you're able to perform the job at hand. What they're trying to see is if you'd fit within the ranks of the company. Are you approachable? Easy to work with? Do you admit that you don't know certain things? The goal is to evaluate the person they will be spending a lot of time with. For managers, this is key as they rely on your performance for their success. They want someone who's trustworthy, competent, able to learn, and can communicate clearly.

Take the time to review the materials you've covered with the interviewer. Be ready to answer questions about your resume and background. Prepare a few key projects you can discuss in detail. Be yourself, be respectful, and be curious. Prepare questions to ask the employer and your potential team.

Step 4 - Offer

If the team is satisfied with your performance, the hiring manager / HR will extend an offer. Typically, you'll receive a phone call and a written offer by email. It's important to review the offer and take a bit of time to get back to the employer promptly with questions (if any). A PLC programming job will typically command an excellent premium, a bonus structure, and some time off. Be clear on what you're getting.

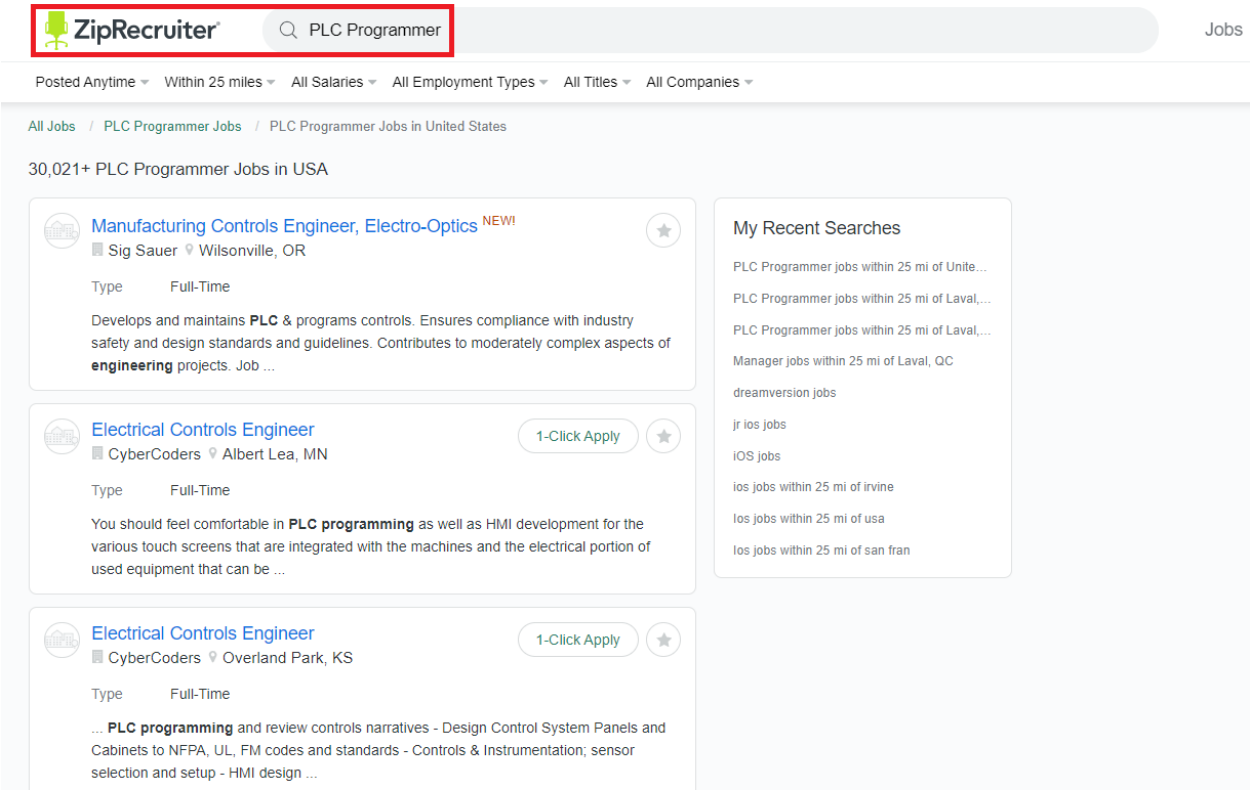
6.4 | Where to Find PLC Programming Jobs

PLC programming jobs aren't always easy to come by. Just like the rest of the industry, those who manage these job postings aren't always in touch with the latest trends and behaviors of recruits. Therefore, we recommend the following places for your job search:

Start by searching the manufacturing companies in your area and neighboring cities. In general, almost every production plant will be listed under a Google search. Look at the career section of these plants as well as the parent company.

Look on large search sites:

- Monster
- Indeed
- ZipRecruiter



PLC Programming Jobs on ZipRecruiter

Create a strong social profile on LinkedIn. The site that was built for employers and job seekers has seen tremendous growth in the last few years and has helped thousands of people find new positions. In addition to job postings, LinkedIn provides an excellent networking opportunity.

Networking with Recruiters

A significant amount of all jobs are now sourced through LinkedIn. Along with plc programming jobs, you'll find a number of recruiters that are scouting for potential candidates. One of the best ways to secure a job is to reach out to a reliable recruiter that specializes in automation and control systems. In today's market, large companies are utilizing 3rd party companies to find talent; these recruiters have access to jobs you may not find any other way. We highly recommend that every engineer, technician, or manager create a professional profile on LinkedIn and network with professionals in their respective fields.

6.5 | Applying to PLC Programming Jobs

Building a resume that outlines your experience and knowledge in an easy-to-understand manner is key to landing a job. If you have less than five (5) years of professional experience, list all of your experience and projects on a single page. Make sure to include industry-specific keywords and relevant software you're most familiar with.

WHERE TO NEXT

Congratulations on making it to the end of the book! If you can confidently talk about the topics covered within this book you will come out above 90% of candidates. Many PLC programmers struggle with basic concepts, and yet that's exactly what employers will test for.

From Beginner to Advanced PLC Programmer

Once you have mastered the basics, it's now time to move to more intermediate topics. First, you should continue learning additional PLC instructions. Starting from the most important, we should focus on timers and mathematical computations. Instructions such as TON, TOF, ADD, MUL, SUB, and DIV should be familiar to you.

As you start working with these new instructions, you'll also notice new data types that may not be familiar to you. RSLogix and other PLC platforms provide particular constructs depending on the instructions. For example, the TON and TOF instructions require a structure of type COUNTER. Take note of these as they become essential in your PLC programming journey.

Advanced PLC Programmers would start to learn concepts such as Add-On Instructions, User-Defined Data Tags, Fault-Tolerant Programming, External Hardware, and more. It's the time to start investing your focus into advanced techniques, understand how to build robust code, how to create layouts for entire machines and/or plants, and more. At this stage, you should be comfortable working with most instructions in the PLC programming world; advanced applications such as recipe control, plant automation, and full machine development should be of great practice at this point.

Another area of focus at this point should be other methods of programming your PLC: Structured Text, Function Block Diagrams, and Sequential Function Charts.

These methods of programming are useful in their own unique way and should be leveraged depending on the situation. Furthermore, it's recommended that you start becoming familiar with external pieces of hardware such as Variable Frequency Drives (VFDs), Servo Motion (Kinetix platform), safety circuits, and others.

The last piece of a competent PLC programmer is platform diversification. As you reach a certain proficiency in your domain, it's important to start looking outside and become familiar with other technologies in the field. This may mean learning SCADA applications from different suppliers, understanding how to interface your PLC data into a SQL database or how to send your data to a cloud-based application.

Additional Resources

Although this book is a great place to start learning PLC programming, we'd like to point you to a few other useful resources to accelerate your learning and career.

- **Free technical PLC programming & automation tutorials –** We have created over 100 technical tutorials covering various topics in PLC programming & automation. From the basics to advanced topics, you'll find everything you need to learn every day. You can **explore the tutorials here**.
- **Free YouTube videos –** This channel has over 100 videos and is all about teaching real-world practical automation skills through in-depth explanations and examples. We cover the theory, implementation & common pitfalls to make sure you always get the whole picture. New videos are added every week. You can **explore and subscribe to the channel here**.
- **PLC programming & automation courses –** If you're looking to learn PLC programming in a fast, efficient, and structured way without breaking the bank, we got you covered! We put together courses that combine a concise curriculum with step-by-step implementation instructions, and hands-on expert support to teach you real-world practical skills. Over the past 3 years,

10 000+ students have taken our courses to get jobs at companies like P&G, Amazon, Tesla, and many more. Check out **SolisPLC** to learn more.

- **Get your questions answered –** If you need help with a project you're working on, a topic you're trying to learn, or any else related to the world of automation & controls, make sure to **post your questions on our forum** so you can get help from our community & team of experts. Remember, there is no such thing as a stupid question. We're all here to learn.

There you have it! We hope this book has been helpful, and that you are now confident to go out there and get started with your career in PLC programming! If you know anyone who would benefit from this book, please make sure to share it with them. This would mean the world to us.

If you would like to get the best new PLC programming & automation tutorials sent to your inbox each week, make sure to **subscribe to our newsletter**.

We wish you the best of luck on your learning journey and hope to stay in touch.

With ❤️ – The SolisPLC Team

About SolisPLC

SolisPLC is the leading online platform designed to teach industry professionals the automation skills they need at their own pace. From PLC programming to Machine Vision, all our courses are taught by practicing industry experts. Over +300 000 engineers, technicians and operators come every year to learn PLC programming & automation.

If you're a business, with SolisPLC Enterprise, you can continuously upskill everyone on your team so they can make better technical and business decisions, and help your organization stay ahead of the curve.

You can learn more at SolisPLC.com

EVERYTHING YOU NEED TO GET A JOB IN PLC PROGRAMMING

"ABSOLUTELY BRILLIANT, SOLISPLC MADE MY TRANSITION INTO A CONTROL SYSTEMS ENGINEER ROLE SO MUCH EASIER, SUCH A LEGEND, THANK YOU!"

JAMIE CHAPLIN

SolisPLC