**Mechatronics**

# PLC and HMI Programming
## Advanced PLC Training System

## Course Sample
8088960

Order no.: 8088960 (Printed version) 8088961 (CD-ROM)
First Edition
Revision level: 07/2018

By the staff of Festo Didactic


Internet: www.festo-didactic.com
e-mail: did@de.festo.com

Printed in Canada

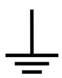
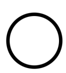
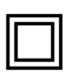
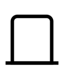
ISBN 978-2-89789-319-4 (Printed version)
ISBN 978-2-89789-320-0 (CD-ROM)
Legal Deposit – Bibliothèque et Archives nationales du Québec, 2018
Legal Deposit – Library and Archives Canada, 2018

# Safety and Common Symbols

The following safety and common symbols may be used in this course and on the equipment:

| Symbol | Description |
|---|---|
| ⚠ DANGER | **DANGER** indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury. |
| ⚠ WARNING | **WARNING** indicates a hazard with a medium level of risk which, if not avoided, could result in death or serious injury. |
| ⚠ CAUTION | **CAUTION** indicates a hazard with a low level of risk which, if not avoided, could result in minor or moderate injury. |
| CAUTION | **CAUTION** used without the *Caution, risk of danger* sign , indicates a hazard with a potentially hazardous situation which, if not avoided, may result in property damage. |
|  | Caution, risk of electric shock |
|  | Caution, hot surface |
|  | Caution, risk of danger. Consult the relevant user documentation. |
|  | Caution, lifting hazard |
|  | Caution, belt drive entanglement hazard |
|  | Caution, chain drive entanglement hazard |
|  | Caution, gear entanglement hazard |
|  | Caution, hand crushing hazard |
|  | Notice, non-ionizing radiation |
|  | Consult the relevant user documentation. |
| ⎓ | Direct current |

# Safety and Common Symbols

| Symbol | Description |
|---|---|
|  | Alternating current |
|  | Both direct and alternating current |
|  | Three-phase alternating current |
|  | Earth (ground) terminal |
|  | Protective conductor terminal |
|  | Frame or chassis terminal |
|  | Equipotentiality |
|  | On (supply) |
|  | Off (supply) |
|  | Equipment protected throughout by double insulation or reinforced insulation |
|  | In position of a bi-stable push control |
|  | Out position of a bi-stable push control |

# Table of Contents

# Table of Contents

# Table of Contents

# Table of Contents

Table of Contents

# Preface

Programmable logic controllers (PLCs) are at the center of a multitude of automated systems. Some industries that benefit from these systems include manufacturing plants, breweries, wastewater treatment plants, mining facilities, and automotive assembly plants.

This course introduces students to four PLC programming languages: ladder, sequential function chart, function block, and structured text. The most common PLC instructions are used within the Siemens' *TIA Portal* software environment. In addition to PLC programming, students learn how to conceive interface screens, simulate them with *WinCC Runtime Advanced* and make them work along with the PLC project.

In terms of hardware, the Advanced PLC Training System uses modern, industrial-grade equipment that is instrumental in teaching theoretical and hands-on knowledge required to work in the automation industry. Optional applications can be purchased and connected to the PLC suitcase to add real-world sensors and actuators to the setups.

In order to perform the exercises in this course, the *TIA Portal (version 15)* software and the *WinCC Runtime Advanced (version 15)* software must be installed on the student's computer.

We invite readers to send us their tips, feedback, and suggestions for improving the course.

Please send these to did@de.festo.com.

The authors and Festo Didactic look forward to your comments.

# About This Course

## Course objectives

When you have completed this course, you will be able to navigate through the *TIA Portal* software from Siemens and use the help documentation. You will know how to create PLC routines in ladder diagram (LAD), function block diagram (FBD), and structured control language (SCL), and how to develop operator interfaces.

## Safety considerations

Safety symbols that may be used in this course and on the equipment are listed in the Safety and Common Symbols table at the beginning of this document.

Safety procedures related to the tasks that you will be asked to perform are indicated in each exercise.

Make sure that you are wearing appropriate protective equipment when performing the tasks. You should never perform a task if you have any reason to think that a manipulation could be dangerous for you or your teammates.

## Reference material

The resource kit DVD-ROM contains the files that will be needed throughout the exercises.

## Prerequisite

As a prerequisite to this course, you should have a basic understanding of electricity. It is assumed that you have a general understanding of these concepts:

- DC and ac voltage, current, and power
- Basic electrical components (resistor/potentiometer, inductor, capacitor, diode, transistor)
- Ohm's law
- Series and parallel circuits
- Electrical measurement

## Systems of units

Units are expressed using the International System of Units (SI) followed by units expressed in the U.S. customary system of units (between parentheses).

# To the Instructor

You will find in this Instructor Guide all the elements included in the Student Manual together with the answers to all questions, results of measurements, graphs, explanations, suggestions, and, in some cases, instructions to help you guide the students through their learning process. All the information that applies to you is placed between markers and appears in red.

### Accuracy of measurements

The numerical results of the hands-on exercises may differ from one student to another. For this reason, the results and answers given in this course should be considered as a guide. Students who correctly perform the exercises should expect to demonstrate the principles involved and make observations and measurements similar to those given as answers.

Sample

Extracted from

Instructor Guide

# Clamp and Stamp Application (LAD)

**EXERCISE OBJECTIVE**

In this exercise, you will:

- Open an existing PLC project
- Create tags
- Create a ladder routine
- Insert rungs and bit instructions
- Test the PLC program with an application (real or simulated)

**DISCUSSION OUTLINE**

The Discussion of this section covers the following points:

- Clamp and work
- The clamp and work application
  *Two types of sensors. Pneumatic cylinders. Directional control valves.*
- Set and reset instructions
  *Set instruction. Reset instruction.*

**DISCUSSION**

## Clamp and work

Clamp and work systems usually consist of two hydraulic or pneumatic cylinders. The clamp cylinder extends until it stalls against the workpiece. The work cylinder advances and retracts a machining tool to perform a task on the clamped workpiece, such as bending, pressing, drilling, cutting, stamping, or grinding. An example of a clamp and work sequence is shown in Figure 59.

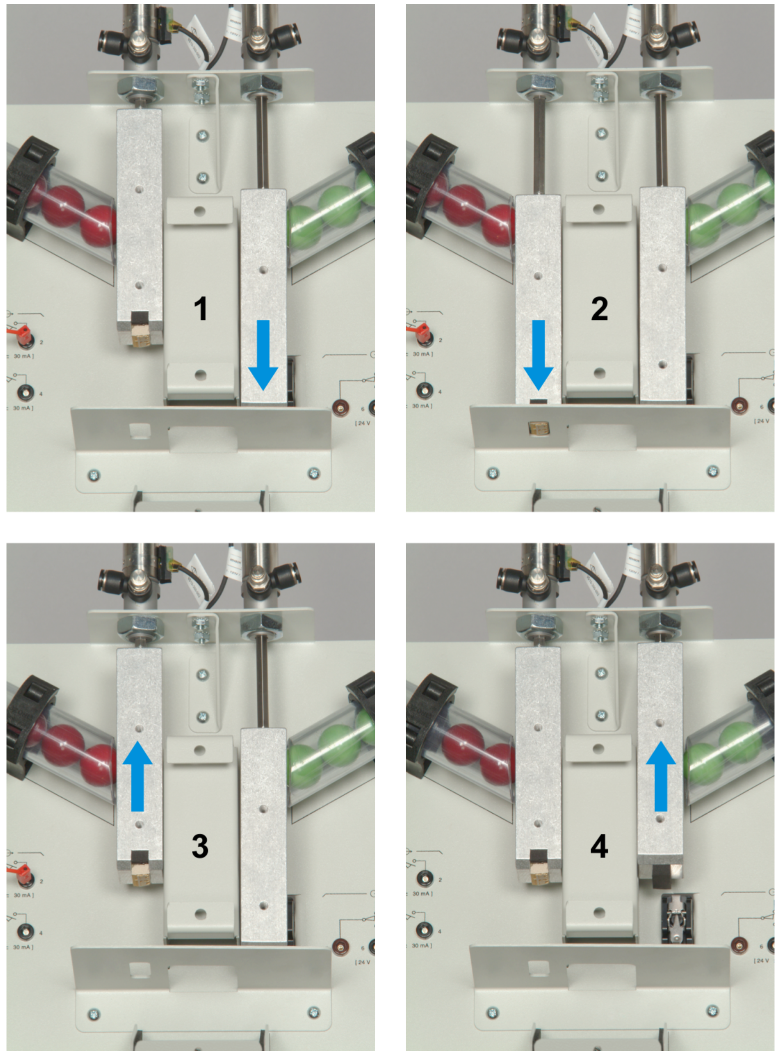



**Figure 59. Clamp and work sequence.**

1. The clamp cylinder extends.
2. The work cylinder extends.
3. The work cylinder retracts.
4. The clamp cylinder retracts.

In this particular example, the clamp cylinder cannot release its grip on the workpiece until the work cylinder has completely retracted. This ensures that the workpiece remains clamped while being worked on. Proximity detectors may be used to verify the position of the cylinders during the operation.

***Animation displayed on the HMI when the clamp and stamp application is working.***
*ip.festo-didactic.com/FDCAQRCodes/qrcode0005.html*

## The clamp and work application

*Two types of sensors*

The clamp and work application has two different types of sensors: a **mechanical limit switch** and two reed switches.

The proximity detector at the bottom of the right cylinder is a mechanical limit switch, shown in Figure 60. It is a single-pole, double-throw switch, which means that it has both a normally open and a normally closed contact. The normally open contact closes when the mechanical arm is pushed. A spring returns the arm to its original position.

**Figure 60. Mechanical limit switch.**

Two **reed switches** are located on the left cylinder, as shown in Figure 61. These single-pole, single-throw switches are only closed when the permanent magnet attached to the piston of the cylinder comes close to the switch.

*Reed switches are actuated when an applied magnetic field joins the pair of ferrous metal reed contacts located inside a hermetically sealed envelope.*

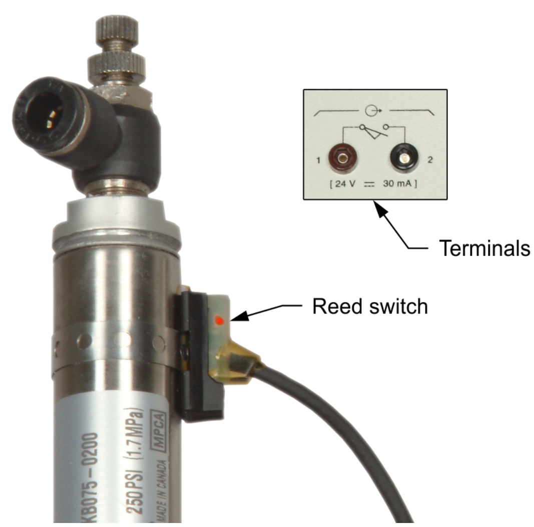



**Figure 61. Reed switch on a pneumatic cylinder.**

*Pneumatic cylinders*

A **pneumatic cylinder** is an actuator that converts fluid energy into straight-line or linear mechanical energy. Single-acting cylinders generate forces in a single direction whereas double-acting cylinders generate forces during both extension and retraction of the rod.

A double-acting cylinder is shown in Figure 62. It contains two fluid ports and does not include a spring. When fluid enters one port of the cylinder, the piston moves toward the other port. The corresponding symbol is shown in Figure 63.

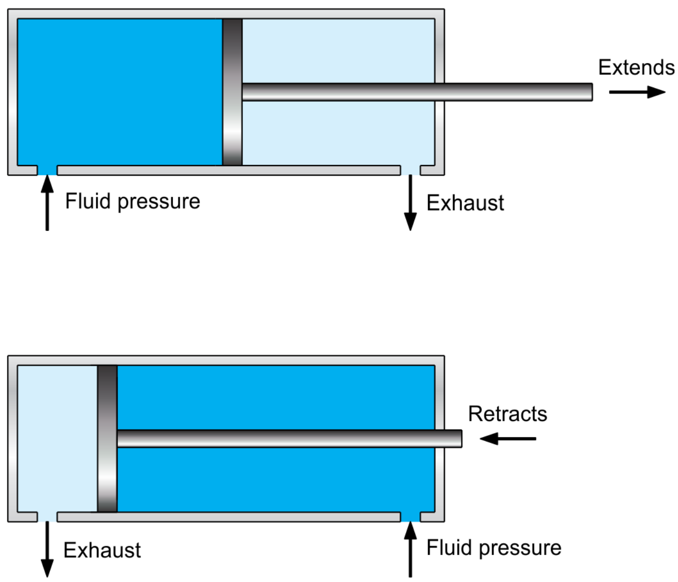



**Figure 62. Double-acting cylinder extending and retracting.**

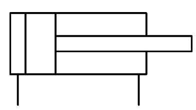


**Figure 63. Double-acting cylinder symbol.**

### *Directional control valves*

**Directional control valves** (DCVs) such as the one shown in Figure 64 are used to control the flow in a fluid power branch circuit. They can perform three functions:

- Block fluid flow
- Allow fluid flow
- Change the direction of fluid flow

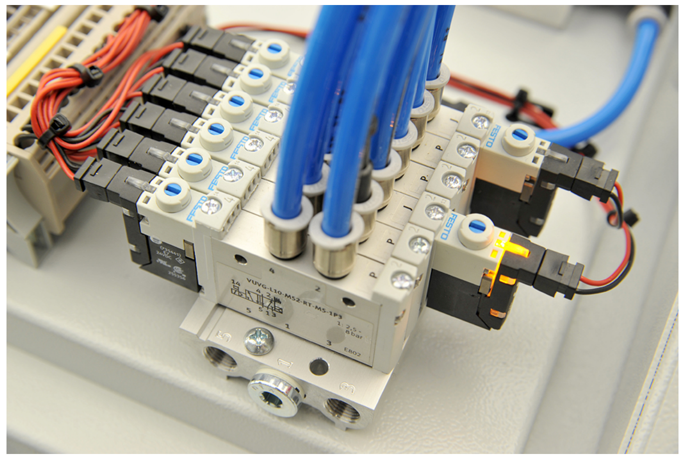


**Figure 64. Directional control valves.**

Directional control valves are described according to:

- the number of controlled openings, called ports or ways.
- the number of positions or states that the valve can assume.
- the type of valve actuators (also called operators) used to switch the valve from one position to another.

The symbols used for directional control valves indicate the above three characteristics and also show a simplified flow path for each position. The symbols do not provide any information about the physical construction of the valve; they simply indicate the function of the valve.

Let us ignore the valve operators for the moment and consider only the number of ports and positions of a valve. Most directional control valves have two or three positions. Each position of the valve provides a different flow path configuration. Valve types are identified using two numbers of the form "*m/n*" where the first number *m* is the number of ports and the second number *n* is the number of positions.

The symbol for a directional control valve consists of two or more blocks (sometimes referred to as envelopes). There is a separate block for each position and each block illustrates a different flow path. Ports are shown as lines protruding from one of the blocks. The block with lines protruding, or with ports identified by a letter or a number, shows the flow path through the valve in its normal position (or return position). The other positions are called operated positions (or actuated positions or working positions).

As an example, the symbol for a two-port, two-position (2/2) directional control valve is shown in Figure 65. On the block showing the normal position of the valve, the ports are numbered. The open (passing) flow path is shown as an arrow. Ports that are closed (non-passing) are shown using short lines drawn at right angles.

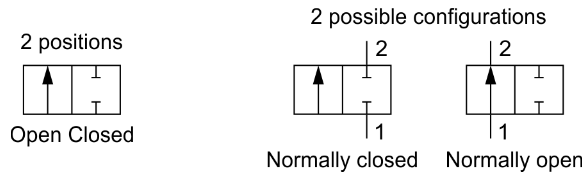



**Figure 65. 2-port, 2-position directional control valve.**

The devices used to switch the valve from one position (state) to another are called valve actuators or operators. Different types of valve actuators may be used. Figure 66 shows a valve with an electrical actuator (a solenoid) and a spring return. Since this valve is normally open, the normal position shows the flow path from port 1 to 2.

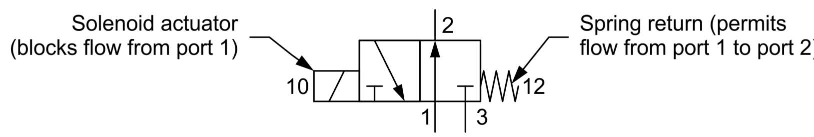



**Figure 66. 3-port, 2-position, normally open, solenoid operating, spring return valve symbol.**

## Set and reset instructions

The coil output instruction is a non-retentive instruction. In other words, this instruction does not retain its value when the rung is false. The output status changes with the rung status:

- If the rung is true, the coil output instruction becomes true and the output turns on.
- If the rung is false, the coil output instruction becomes false and the output turns off.

There are, however, some conditions where an output needs to remain on even after the conditions in the rung are no longer true. For example, if separate start and stop buttons are used, you would not want to hold the start button for the duration of the activation. In this event, instructions with a retentive operation are desired.

These instructions are the set and reset instructions described in Figure 67. The set and reset instructions are almost always used in pairs with the same bit address since both instructions modify the same bit.

| Instruction | Symbol |
|---|---|
| Set | -( S )- |
| Reset | -( R )- |

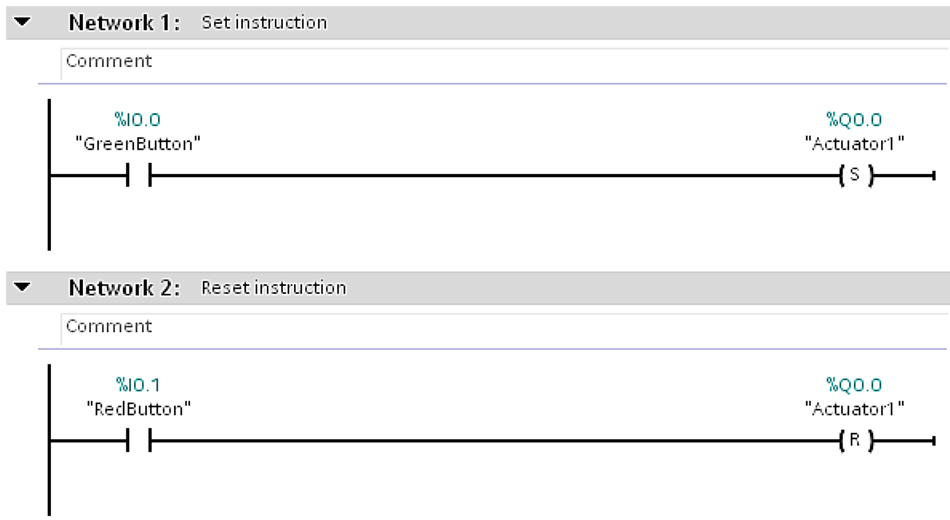



**Figure 67. Associated set and reset instructions triggered by two inputs.**

***This video shows how to create associated set and reset instructions in TIA Portal.***
*ip.festo-didactic.com/FDCAQRCodes/qrcode0002.html*

### *Set instruction*

The set instruction sets the addressed bit to logic state 1 when the rung is true. After, the bit remains at logic state 1, regardless of the current rung conditions, until it is set to logic state 0 by the associated reset instruction.

### *Reset instruction*

The reset instruction sets its addressed bit to 0 when the rung is true. Thereafter, the bit remains at logic state 0 regardless of the current rung conditions, until it is set to logic state 1 by the associated set instruction.

Table 7 summarizes the logic state operation of the set and reset instructions.

**Table 7. Set and reset instructions states.**

| Instruction | Previous state | Rung condition | New state of bit Q0.0 |
|---|---|---|---|
| Set | 0 | True | 1 |
| | | False | 0 |
| | 1 | True | 1 |
| | | False | 1 |
| Reset | 1 | True | 0 |
| | | False | 1 |
| | 0 | True | 0 |
| | | False | 0 |

***This video shows associated set and reset instructions running in a controller.***
*ip.festo-didactic.com/FDCAQRCodes/qrcode0001.html*

## PROCEDURE OUTLINE

The Procedure is divided into the following sections:

- Setup and connections
  *Real application. Simulated application.*
- Checking the device configuration
- Inserting the logic
  *Extending the stamp cylinder. Retracting the stamp cylinder. Extending the clamp cylinder. HMI animations.*
- Running the PLC and HMI projects
- Running the project
- Testing the application
- End of the procedure

## PROCEDURE

### Setup and connections

*In this exercise, you will complete a program that controls a clamp and stamp application. The program will use a push-button to clamp and another one to stamp a fictive sheet of metal. The logic will only permit stamping if the workpiece is clamped first. You can either connect the real application to the trainer or use the version that is simulated on the HMI and shown in Figure 68.*

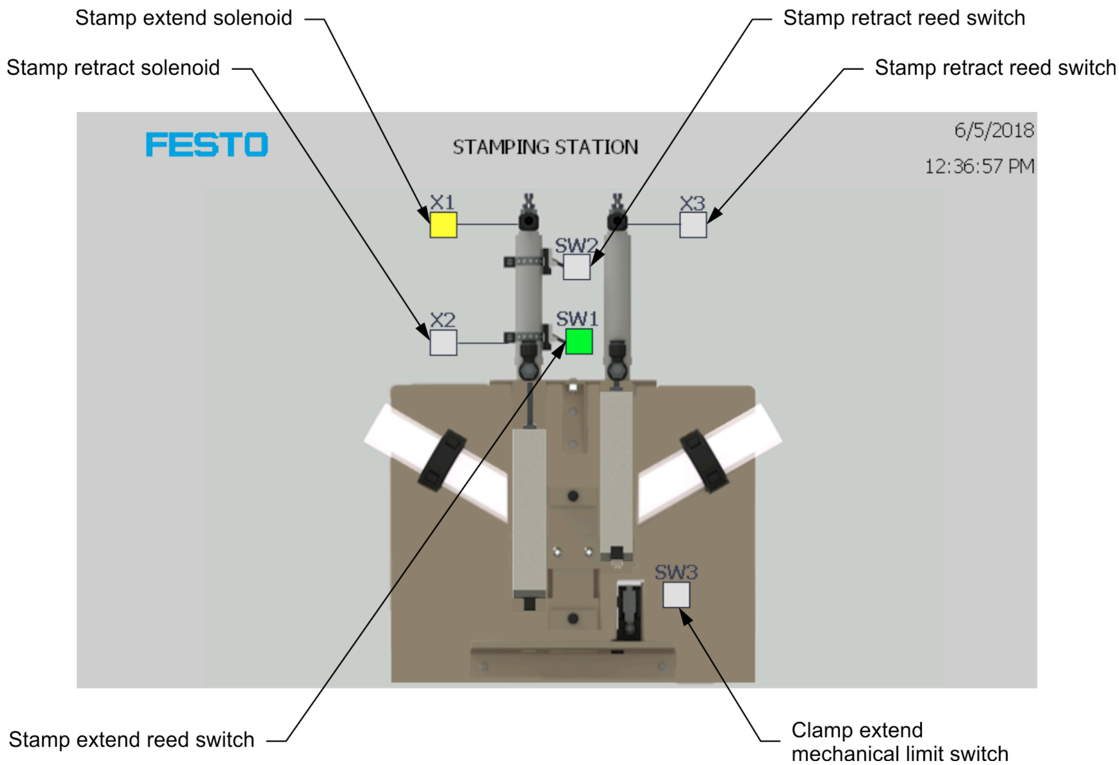



**Figure 68. HMI application.**

*Green squares indicate actuated sensors. Yellow squares show energized actuators. Gray squares indicate that the state of the sensor or actuator is off.*

*The clamp cylinder is a spring-return cylinder, hence it retracts when X3 is not actuated.*

## *Real application*

*If you are using the simulated application, jump to the Simulated application section.*

1. Connect the equipment as shown in Figure 69 and Figure 70. The tags used in this exercise are shown in Figure 71. Adjust the air pressure between 200 kPa and 275 kPa (30 psi and 40 psi).

*To avoid air leaking from the connectors, make sure that plastic tubes are cut clean and inserted deep enough into the connectors. Plastic tubes are removed by pressing the release button towards the body of the connector before pulling out the tubing.*

Together with the students, make sure that no fault is inserted in the training system.

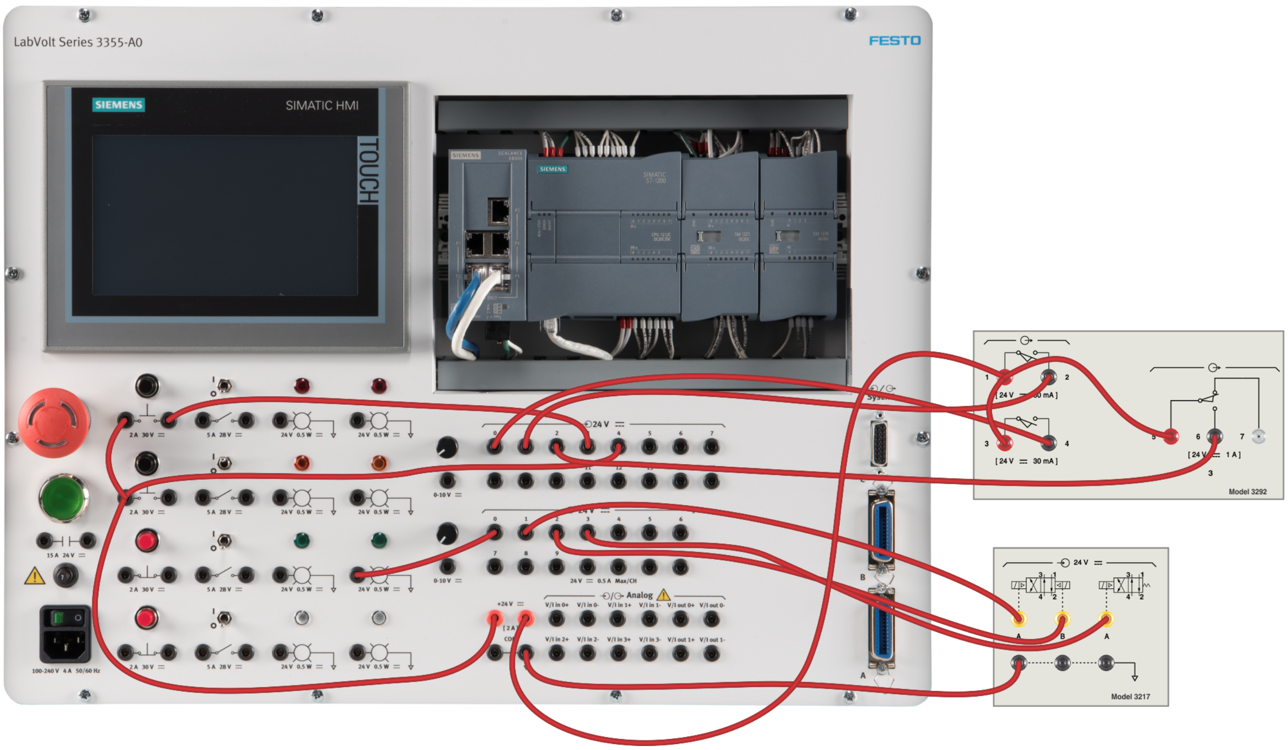



**Figure 69. Clamp and stamp setup (real application).**

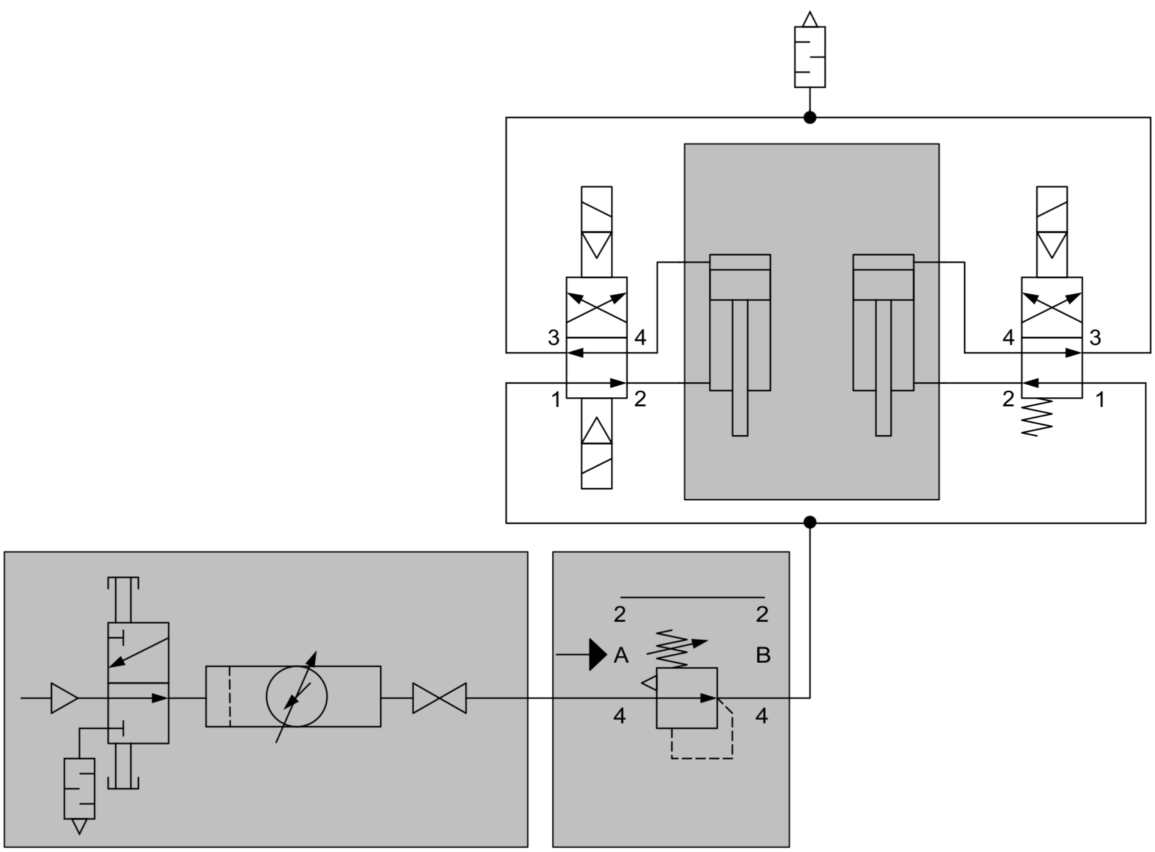



**Figure 70. Pneumatic connections.**

| | | Name | Data type | Address | Retain | Acces... | Writa... | Visibl... | Comment |
|---|---|---|---|---|---|---|---|---|---|
| 1 | | SW1 | Bool | %I0.0 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Extended Limit Switch |
| 2 | | SW2 | Bool | %I0.1 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Retracted Limit Switch |
| 3 | | SW3 | Bool | %I0.2 | ☐ | ☑ | ☑ | ☑ | Clamping Cylinder Extended Cylinder |
| 4 | | PB1 | Bool | %I0.3 | ☐ | ☑ | ☑ | ☑ | NO Clamping Push Button |
| 5 | | PB2 | Bool | %I0.4 | ☐ | ☑ | ☑ | ☑ | NO Stamping Push Button |
| 6 | | L1 | Bool | %Q0.0 | ☐ | ☑ | ☑ | ☑ | Clamp Indicator (Light) |
| 7 | | X1 | Bool | %Q0.1 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Pneumatic Valve Solenoid (Extend) |
| 8 | | X2 | Bool | %Q0.2 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Pneumatic Valve Solenoid (Retract) |
| 9 | | X3 | Bool | %Q0.3 | ☐ | ☑ | ☑ | ☑ | Clamping Cylinder Pneumatic Valve Solenoid (Extend) |

**Figure 71. Tags (real application).**

2. Turn on the computer and the S7-1200 PLC.

3. Push the green start button under the red emergency stop button on the trainer to activate the outputs of the PLC.

4. Start the *TIA Portal* software.

5. Retrieve the *Ex2_Real_YYYYMMDD_Festo_V15.zap15* project in *TIA Portal*. The project should automatically open in *TIA Portal*.

   If the project does not open automatically, open the PLC Project file (*Ex2_Real_YYYYMMDD_Festo_V15.ap15*) in the target directory.

6. Open the PLC tags window from the *Project tree*. Some tags are missing: create the missing tags using the tags list shown in Figure 71. Double-check the addresses. The tags are the link between the physical inputs/outputs on the PLC and the logic of the PLC program.

### *Simulated application*

1. Connect the equipment as shown in Figure 72. The tags used in this exercise are shown in Figure 73.

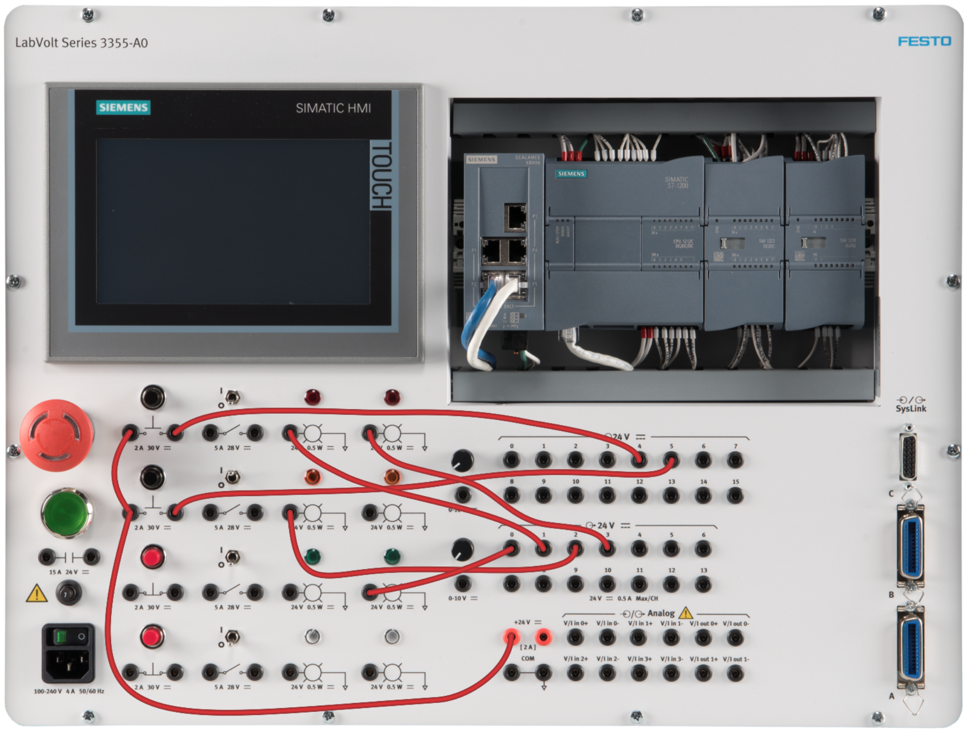



**Figure 72. Clamp and stamp setup (virtual application).**

| | | Name | Tag table | Data type | Address ▲ | Retain | Acces... | Writa... | Visibl... | Comment |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | PB1 | Default tag table | Bool | %I0.3 | ☐ | ☑ | ☑ | ☑ | NO Clamping Push Button |
| 2 | | PB2 | Default tag table | Bool | %I0.4 | ☐ | ☑ | ☑ | ☑ | NO Stamping Push Button |
| 3 | | L1 | Default tag table | Bool | %Q0.0 | ☐ | ☑ | ☑ | ☑ | Clamp Indicator (Light) |
| 4 | | X1 | Default tag table | Bool | %Q0.1 | ☐ | ☑ | ☑ | ☑ | Clamping Cylinder Pneumatic Valve Solenoid (Extend) |
| 5 | | X2 | Default tag table | Bool | %Q0.2 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Pneumatic Valve Solenoid (Extend) |
| 6 | | X3 | Default tag table | Bool | %Q0.3 | ☐ | ☑ | ☑ | ☑ | Stamping Cylinder Pneumatic Valve Solenoid (Retract) |
| 7 | | Clock_Byte | Default tag table | Byte | %MB100 | ☐ | ☑ | ☑ | ☑ | |
| 8 | | Clock_10Hz | Default tag table | Bool | %M100.0 | ☐ | ☑ | ☑ | ☑ | |
| 9 | | Clock_5Hz | Default tag table | Bool | %M100.1 | ☐ | ☑ | ☑ | ☑ | |
| 10 | | Clock_2.5Hz | Default tag table | Bool | %M100.2 | ☐ | ☑ | ☑ | ☑ | |
| 11 | | Clock_2Hz | Default tag table | Bool | %M100.3 | ☐ | ☑ | ☑ | ☑ | |
| 12 | | Clock_1.25Hz | Default tag table | Bool | %M100.4 | ☐ | ☑ | ☑ | ☑ | |
| 13 | | Clock_1Hz | Default tag table | Bool | %M100.5 | ☐ | ☑ | ☑ | ☑ | |
| 14 | | Clock_0.625Hz | Default tag table | Bool | %M100.6 | ☐ | ☑ | ☑ | ☑ | |
| 15 | | Clock_0.5Hz | Default tag table | Bool | %M100.7 | ☐ | ☑ | ☑ | ☑ | |
| 16 | | System_Byte | Default tag table | Byte | %MB101 | ☐ | ☑ | ☑ | ☑ | |
| 17 | | FirstScan | Default tag table | Bool | %M101.0 | ☐ | ☑ | ☑ | ☑ | |
| 18 | | DiagStatusUpdate | Default tag table | Bool | %M101.1 | ☐ | ☑ | ☑ | ☑ | |
| 19 | | AlwaysTRUE | Default tag table | Bool | %M101.2 | ☐ | ☑ | ☑ | ☑ | |
| 20 | | AlwaysFALSE | Default tag table | Bool | %M101.3 | ☐ | ☑ | ☑ | ☑ | |

**Figure 73. Tags (virtual application).**

2. Turn on the computer and the S7-1200 PLC.

3. Push the green start button under the red emergency stop button on the trainer to activate the outputs of the PLC.

4. Start the *TIA Portal* software.

5. Retrieve the *Ex2_Virtual_YYYYMMDD_Festo_V15.zap15* project in *TIA Portal*. The project should automatically open in *TIA Portal*.

   If the project does not open automatically, open the PLC Project file (*Ex2_Virtual_YYYYMMDD_Festo_V15.ap15*) in the target directory.

6. Open the PLC tags window from the *Project tree*. Some tags are missing: create the missing tags using the tags list shown in Figure 73. Double-check the addresses. The tags are the link between the physical inputs/outputs on the PLC and the logic of the PLC program.

   *Several tags for the virtual application are used to simulate the behavior of the real application.*

## Checking the device configuration

7. From the *Project tree*, select the *PLC_1* item. Right-click and select *Properties* from the contextual menu to open the Properties window. This window lists the properties and configuration of the programmable logic controller (Figure 74).

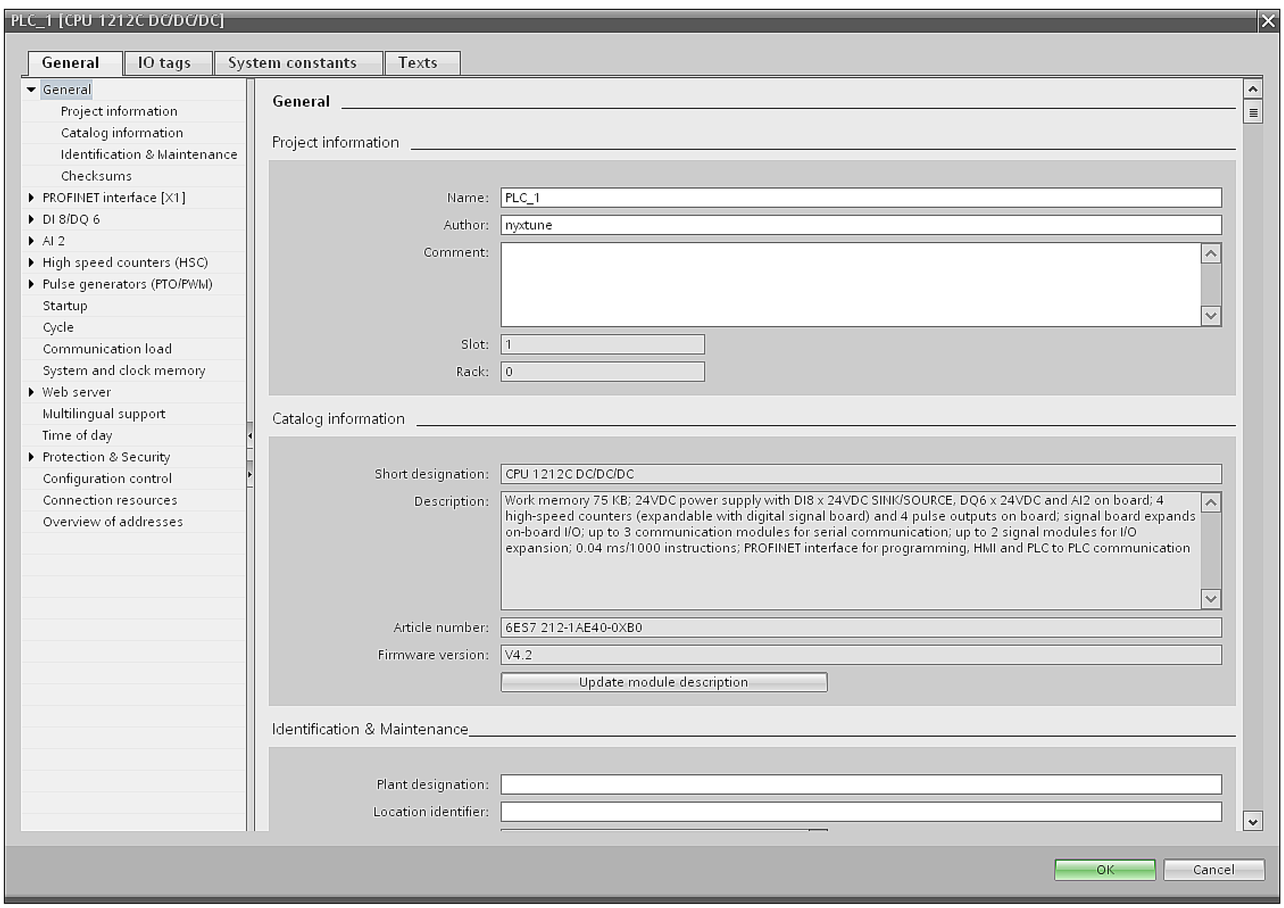



**Figure 74. Programmable logic controller properties.**

**8.** Using the information under the General tab of this window, determine the short designation for the controller configured in this project?

CPU 1212C DC/DC/DC

**9.** What is the cycle monitoring time for this controller?

150 ms (under the *Cycle* item in the *General* tab)

**10.** Using the information from this window and the properties of the other modules, fill out Table 8.

**Table 8. Module characteristics.**

| Module | Slot | Digital inputs | Digital outputs | Analog inputs | Analog outputs |
|---|---|---|---|---|---|
| | 1 | | | | 0 |
| DI 8/DQ 8x24VDC | | | | 0 | |
| | | | | | |

**Module characteristics.**

| Module | Slot | Digital inputs | Digital outputs | Analog inputs | Analog outputs |
|---|---|---|---|---|---|
| CPU 1212C DC/DC/DC | 1 | 8 | 6 | 0 | 0 |
| DI 8/DQ 8x24VDC | 2 | 8 | 8 | 0 | 0 |
| AI 4x13BIT/AQ 2x14BIT | 3 | 0 | 0 | 4 | 2 |

**11.** Inputs and outputs on different modules use different addresses. Complete Table 9 with the information in the various tabs of the Properties window.

**Table 9. Module input and output addresses.**

| Module | Slot | Input addresses | Output addresses |
|---|---|---|---|
| | | | |
| | | | |
| | | | |

**Module input and output addresses.**

| Module | Slot | Input addresses | Output addresses |
|---|---|---|---|
| CPU 1212C DC/DC/DC | 1 | I0.0 to I0.7 | Q0.0 to Q0.5 |
| DI 8/DQ 8x24VDC | 2 | I8.0 to I8.7 | Q8.0 to Q8.7 |
| AI 4x13BIT/AQ 2x14BIT | 3 | IW112, IW114, IW116, IW118 | QW112, QW114 |

### Inserting the logic

*In this section, you will create a ladder routine to operate a clamp and stamp application.*

**12.** There is a Program blocks folder in the *Project tree*. In this folder, there is one program named *Main [OB1]*. The sole purpose of this program block is to call two program blocks called:

- **fbLD01_StampingStation [FB2].** You will create this routine. It will contain the logic for the stamping application.
- **fbLD02_Reserved [FB2].** This routine induces time delays between cylinder displacements when the (virtual) valve solenoids are actuated. This provides a more realistic result on the touch screen.

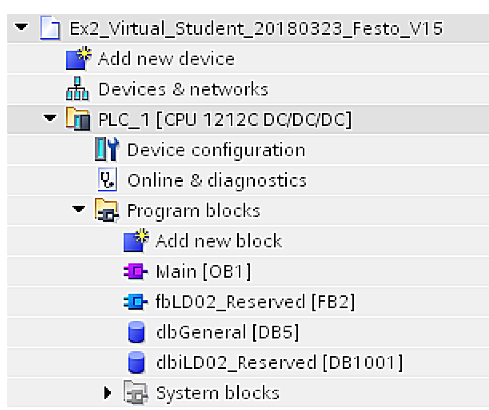



**Figure 75. Program blocks (virtual version).**

13. In the Program blocks folder, click on the *Add new block* item. Select *Function block* and enter the information shown in Figure 76 to create a new block called *fbLD01_StampingStation*.

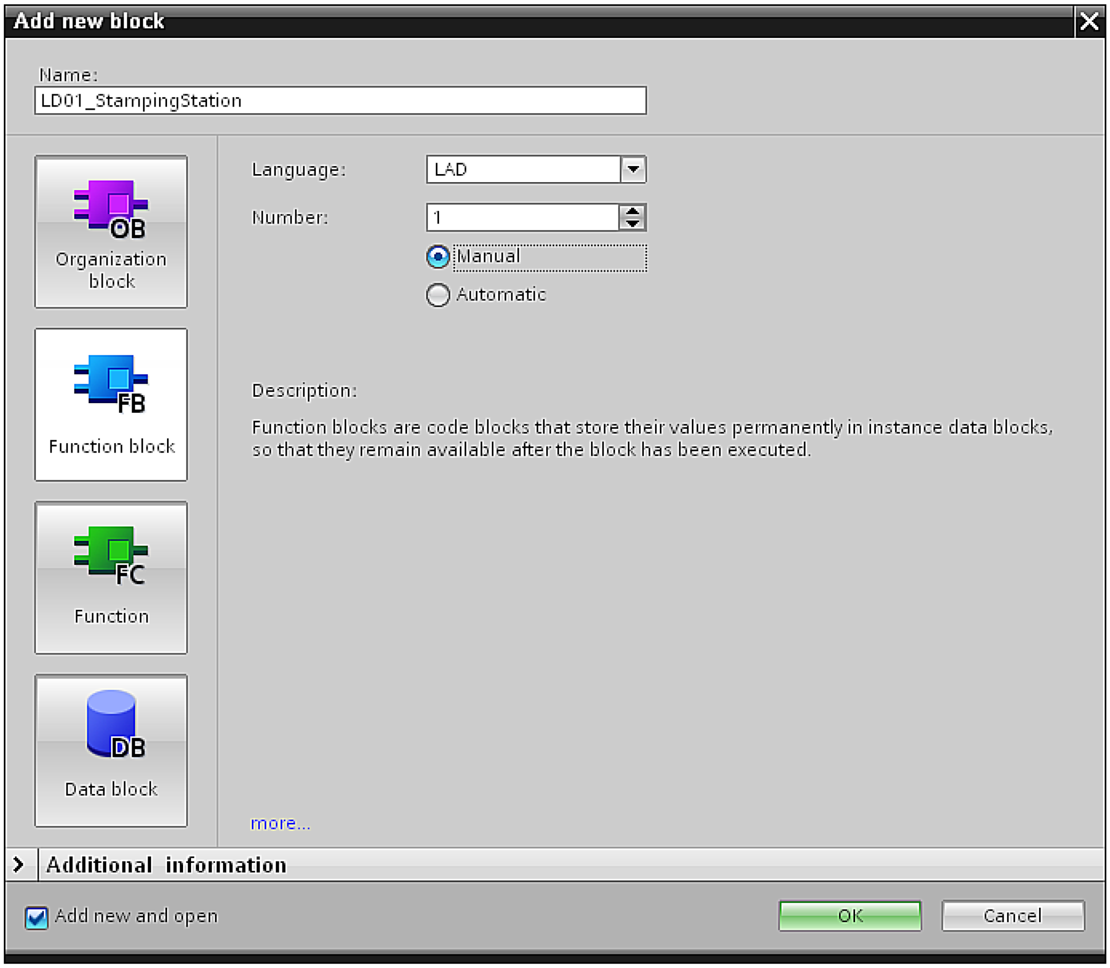



**Figure 76. Create a new block.**

### *Extending the stamp cylinder*

**14.** Open the newly created block (Figure 77). There is already a network in this block. A network is a way of organizing your ladder program and keeping it tidy. It helps if a network performs only one function.

The network in the new block contains a rung without any logic.

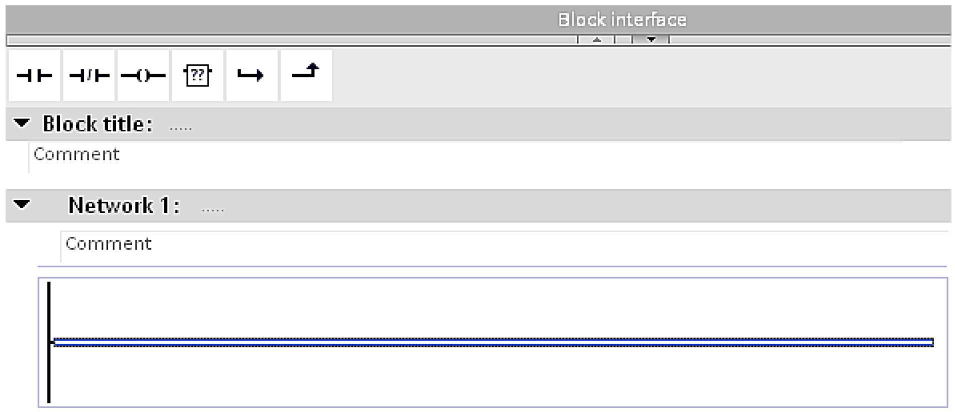



**Figure 77. Empty block using the ladder programming language.**

**15.** To implement the logic for the clamp and stamp application, we need to create several networks, each containing condition and logic that actuates the appropriate output. The first network will be responsible for extending the stamp cylinder. To extend the cylinder, we need to actuate X1. This could be implemented in several ways, but the logic we want is the following:

- The user must press the second button to trigger the extension of the cylinder.
- The cylinder is not already fully extended.
- The cylinder is fully retracted.
- The clamp cylinder is fully extended.
- If all the aforementioned conditions are fulfilled, X1 is actuated and remains actuated even if one of the conditions is no longer true.
- When all conditions are true, we also want to deactivate X2, which maintains the cylinder retracted.

These conditions translate as follows in terms of buttons and switches:

- PB2 is pressed.
- SW1 (stamp extend switch) is not actuated.
- SW2 (stamp retract switch) is actuated.
- SW3 (clamp extend switch) is actuated.

First, add contacts to the rung to implement the conditions above. To add a contact to the rung, drag and drop the appropriate symbol from the toolbar, as shown in Figure 78.

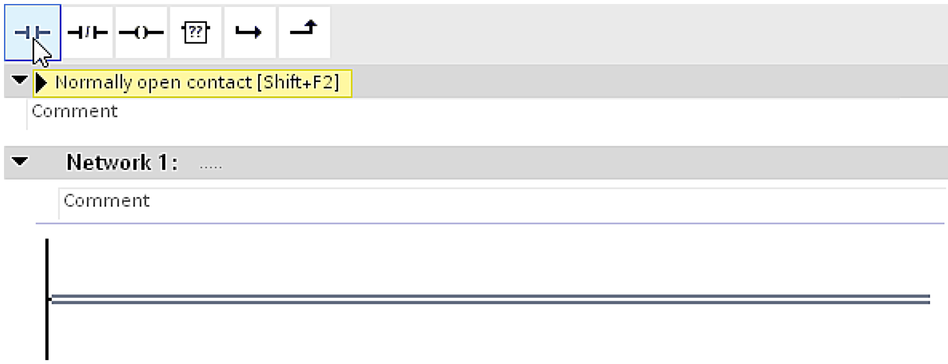



**Figure 78. Adding a contact to the rung.**

16. A newly added contact is not linked to any tag. This is represented by the question marks above the symbol. Click on the question marks and start typing the appropriate tag name in the text box. A list of tags matching what you are typing will be automatically populated, as shown in Figure 79. Select the appropriate tag from the list. In our case, we need to use the PB2 tag.

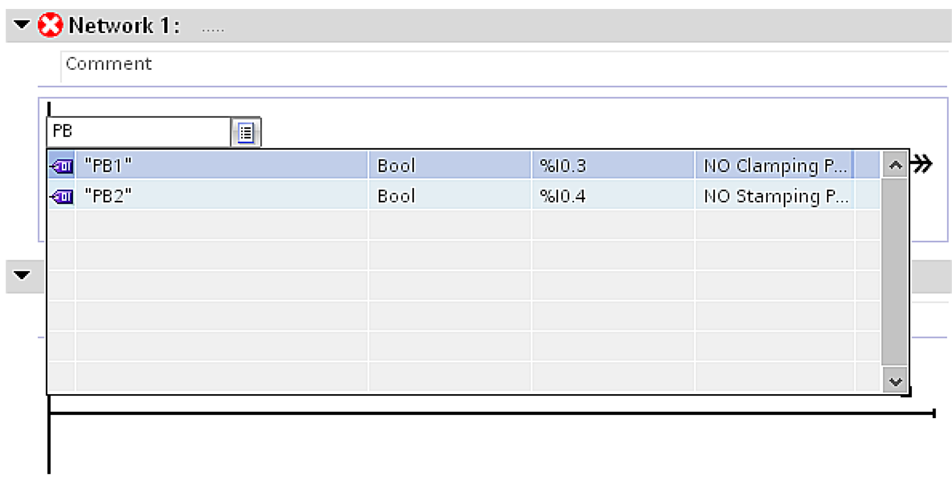



**Figure 79. Assigning a tag to an instruction.**

17. Add the four conditions to actuate X1 to the rung.

*There is a small difference in the tags names between the real and the virtual application. Since the virtual application do not use physical inputs, it uses internal tags located in the dbGeneral data block. Hence, the tags for the switches of the real application are simply "SWX" whereas the tags for the virtual application are "dbGeneral".SWX. The screenshots in this exercise show the tags for the virtual application.*

**18.** Once the four conditions to extend the cylinder are added, add a coil as shown in Figure 80. Since we want this coil to remain active, even if some of the conditions are no longer true, we must use a set instruction. To transform a normal coil output into a set instruction, double-click on the coil and select S from the drop-down list.

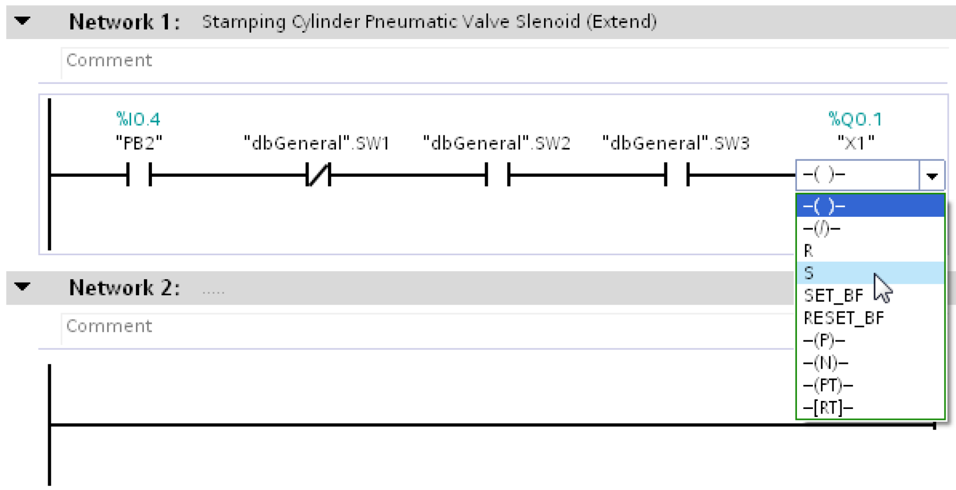



**Figure 80. Transforming a coil output instruction into a set instruction.**

**19.** To be able to extend the cylinder via X1, we must disable X2. In the next network, we will actuate X2 using a set instruction, like the one used to actuate X1 in Network 1. Hence, we need to use a reset instruction to disable X2.

To execute a reset instruction using the same set of conditions, we need to add a branch to the rung after the contacts. To do so, drag the *Open branch* symbol from the toolbar and drop it after the contacts, as shown in Figure 81.

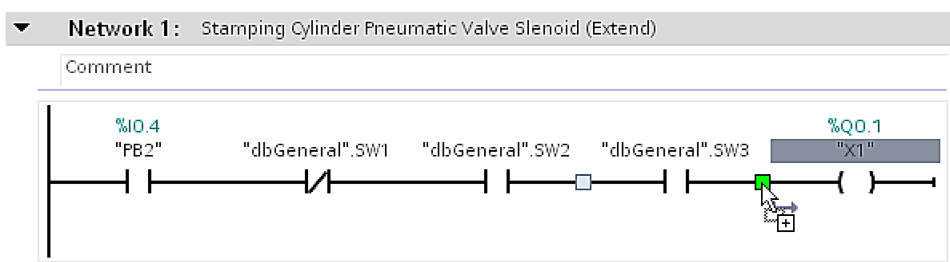



**Figure 81. Open a new branch after the contacts.**

**20.** On the new branch, add a reset instruction to disable X2 (Figure 82).

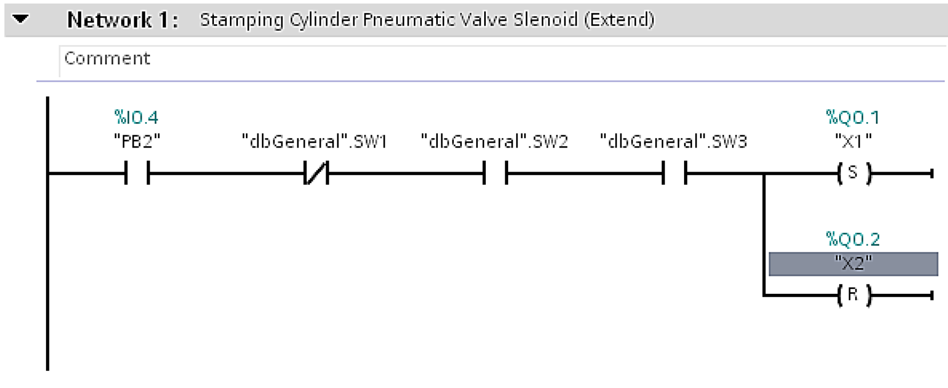



**Figure 82. Add a reset instruction to disable X2.**

*Retracting the stamp cylinder*

**21.** The conditions to retract the stamp cylinder are the reverse of the conditions to extend it, with the exception of SW3 which must be actuated.

These conditions translate as follows in terms of buttons and switches:

- PB2 is not pressed.
- SW1 (stamp extend switch) is actuated.
- SW2 (stamp retract switch) is not actuated.
- SW3 (clamp extend switch) is actuated.

**22.** In a new network, create the logic to test these conditions. Add a set instruction for X2 and a reset instruction for X1.

This network is slightly different depending if you are using the real or the virtual application. Figure 83 shows the network you should use for the real application while step 23 and Figure 84 show how you should proceed if you are using the virtual application.

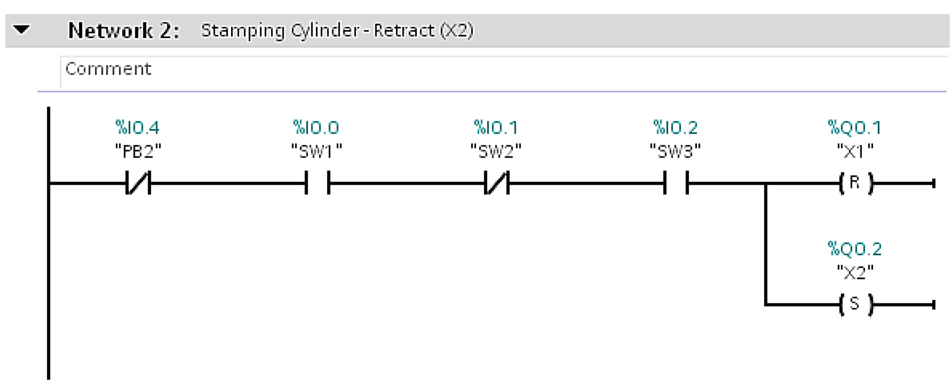



**Figure 83. Create a network to set X1 and reset X1 (real application only).**

23. If you are using the virtual application (which is simulated in the fbLD02_Reserved block), you must add an additional condition to this network so that X2 is set and X1 is reset when the application starts.

    To do this, we will use a special bit called the first scan bit. This bit is set to true only during the first scan of the ladder program. After the first scan, the bit is set to false and remains false unless the program is restarted.

    If you are using the virtual application, modify the second network to add a contact that checks if the first scan bit is enabled, as shown in Figure 84.

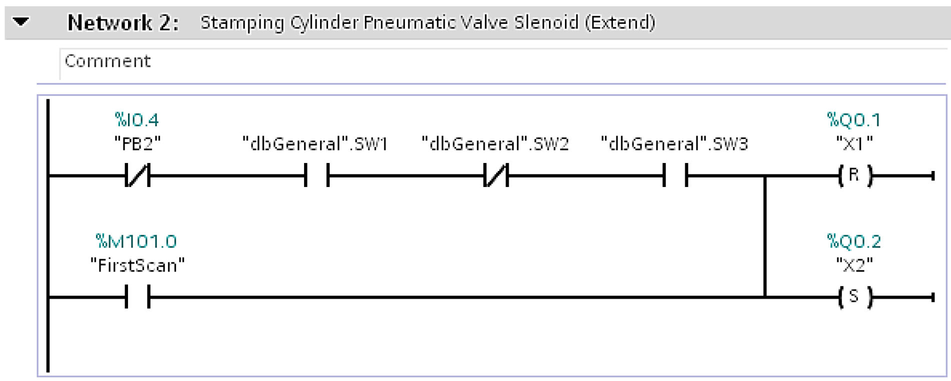



**Figure 84. Add the first scan bit to the logic (virtual application only).**

### *Extending the clamp cylinder*

24. In a new network, we need to create the logic to extend the clamp cylinder. We want to extend the clamp cylinder if one of the two following conditions is true:

    - PB1 is pressed.
    - SW2 is not actuated and X3 is actuated.

    Create this logic in a network as shown in Figure 85.

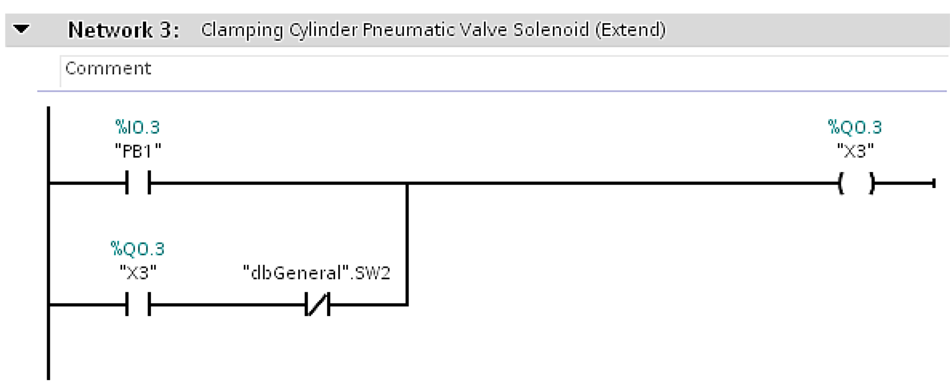
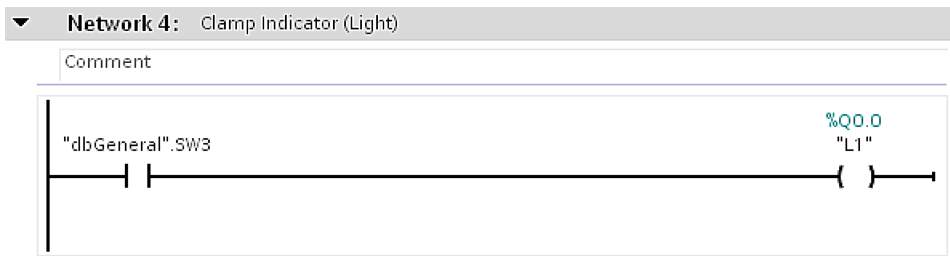



**Figure 85. Logic to extend the clamp cylinder.**

**25.** We would also like to have a visual indicator that confirms that the clamp cylinder is fully extended. Add a new network that turns on the clamp indicator light (L1) when the clamp cylinder is fully extended. In the box below, draw the logic of your program that turns on L1.




**Turning on L1 when the clamp cylinder is extended.**

### *HMI animations*

**26.** The HMI contains several visual indicators that show the state of different components when the application is running. In addition to these indicators, the HMI has animations that show when the cylinder is retracted, moving, or extended. Table 10 lists the tags that trigger the visibility of these animations.

**Table 10. Visual indicators on the HMI.**

| Visibility tag | Conditions for visibility | Description |
|---|---|---|
| V1 | SW2 true and X2 enabled | The stamp cylinder is retracted. |
| V2 | SW1 and SW2 false | The stamp cylinder is moving. |
| V3 | SW1 true and X1 enabled | The stamp cylinder is extended. |
| V4 | X3 disabled and TmrOffX3 false | The clamp cylinder is retracted. |
| V5 | X3 enabled and SW3 true<br>or<br>TmrOffX3 false and SW3 true | The clamp cylinder is moving. |
| V6 | X3 enabled and SW3 true | The clamp cylinder is extended. |

**27.** Figure 86 shows how to enable the visibility of an element on the HMI. This network displays the image of the retracted cylinder if SW2 is true and X2 is enabled.

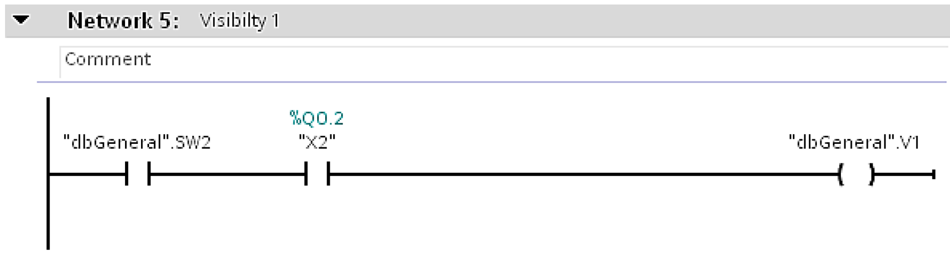
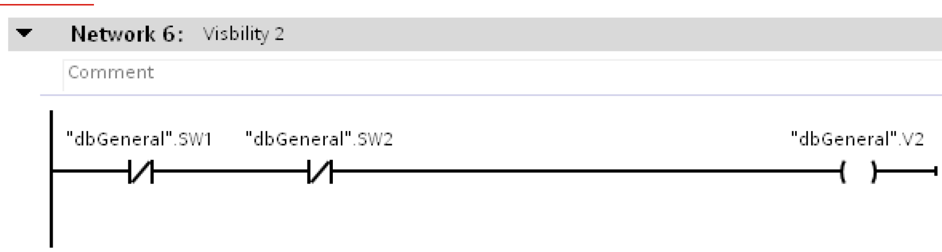
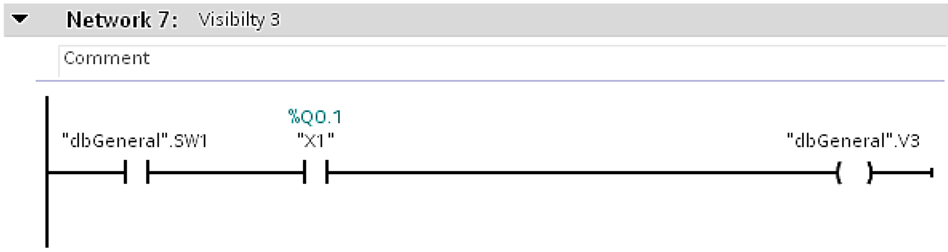



**Figure 86. Stamp cylinder retracted.**

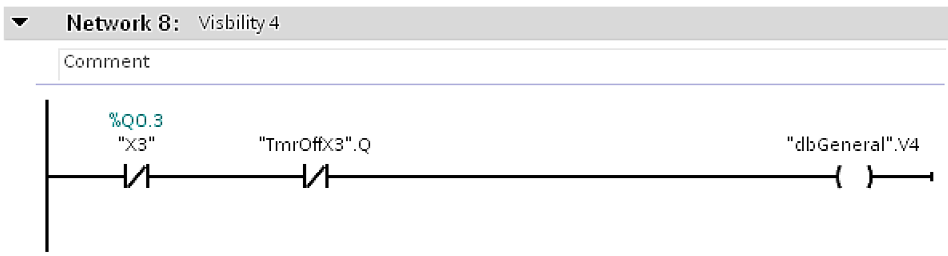
**28.** Using information from Table 10, add five more networks that will trigger the visibility of the appropriate image when the conditions for visibility are met. In the box below, draw the logic added to your program to trigger the visibility of the different elements.



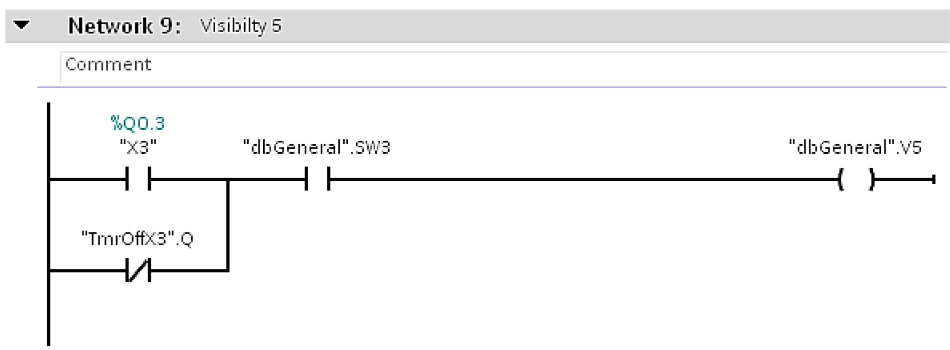

**Stamp cylinder moving.**



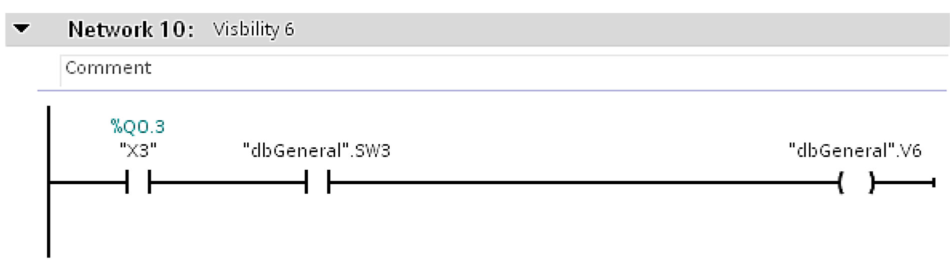

**Stamp cylinder extended.**




**Clamp cylinder retracted.**




**Clamp cylinder moving.**




**Clamp cylinder extended.**

29. At this point, the logic of our program is completed. Make sure to save your work under a new name.

## Running the PLC and HMI projects

**30.** Before trying to download your program to the controller and the HMI project to the touch screen, you should compile them to test if they contain any errors.

Select the PLC_1 folder from the *Project tree* and press the *Compile* button in the toolbar. If there are errors in your program, the compile tab will display something somewhat similar to Figure 87. Use the information from the error messages to debug your code. If no errors are found, the compile tab will look as in Figure 88.

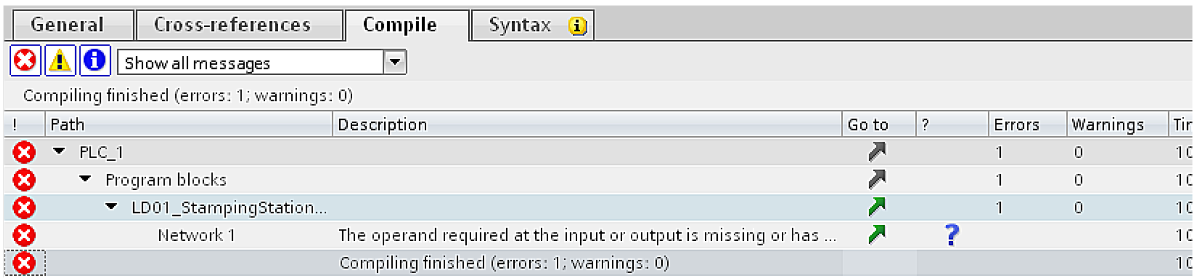



**Figure 87. Errors while compiling the program blocks.**

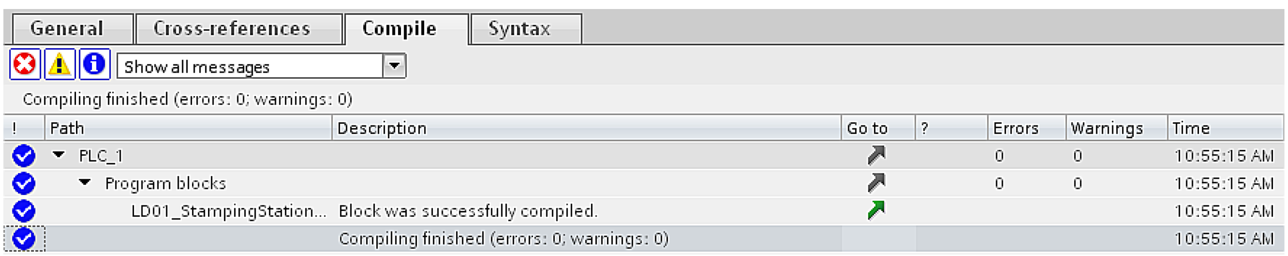



**Figure 88. No errors while compiling the program blocks.**

**31.** Once the program compiles without errors, repeat the process with the HMI application. Select the HMI_1 folder from the *Project tree* and press the compile button. If you did not modify the interface screen, there should be no errors.

## Running the project

**32.** Download the program to the controller and the interface to the touch screen.

**33.** Open the *fbLD01_StampingStation* program block and turn on program monitoring by clicking the *Monitoring on/off* button in the toolbar. When the monitoring mode is on, actuated contacts, coils, or rungs turn green. Figure 89 shows the *fbLD01_StampingStation* program block with the monitoring mode turned on.

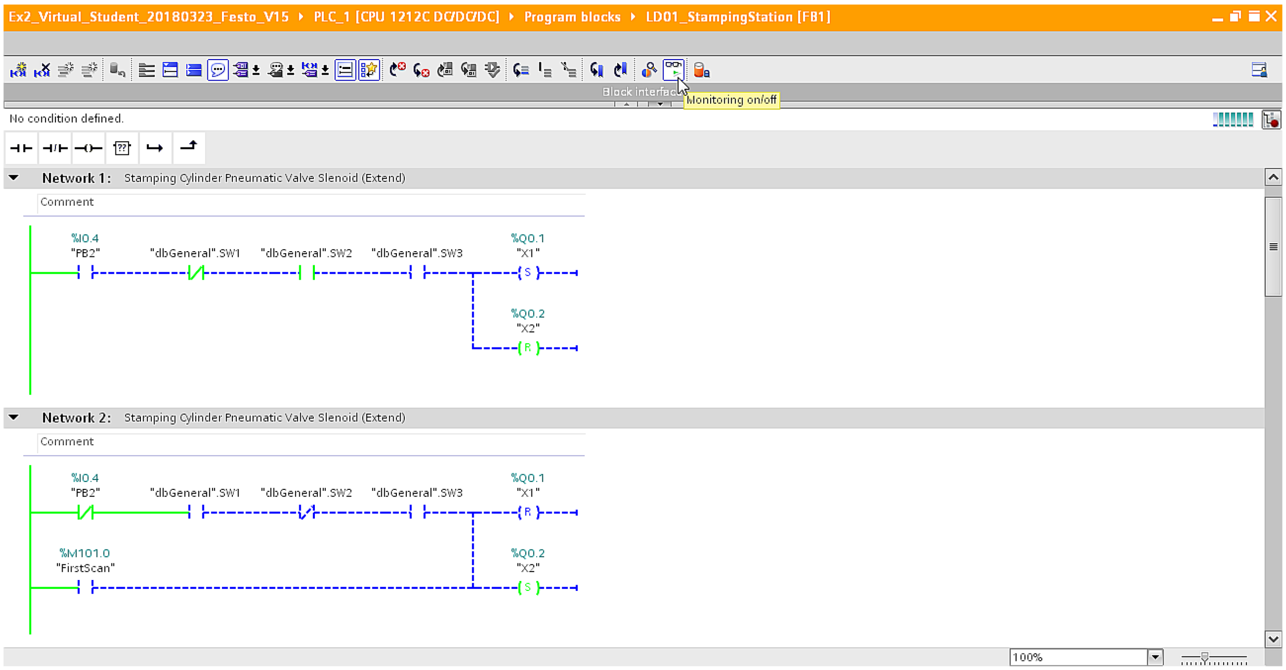



**Figure 89. Turn monitoring on.**

34. Once both the program and the interface are running on the devices, the HMI program should start automatically on the touch screen. The interface should look as in Figure 90. If errors show up on the screen, check connectivity with the PLC and verify the tags in the PLC program.

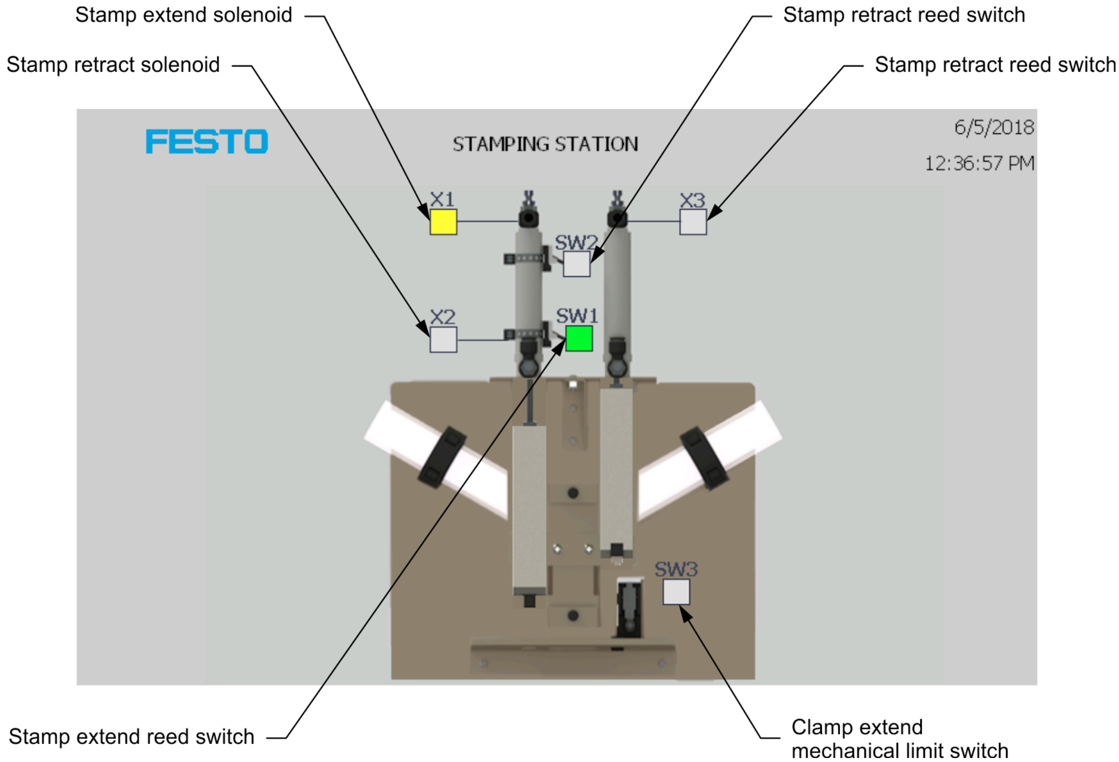



**Figure 90. Running the HMI program.**

## Testing the application

35. Push the green button under the red emergency button on the trainer to activate the PLC outputs. Using the push-buttons, verify that the program operates as expected. Use Table 11 as a checklist.

**Table 11. Clamp and stamp application behavior checklist.**

| Test | Yes | No |
|---|---|---|
| Pressing PB1 extends the clamp cylinder. | | |
| Output 0 (light) energizes when the clamp cylinder is extended. | | |
| Pressing PB2 extends the stamp cylinder ONLY if the clamp cylinder is fully extended. | | |
| The clamp cylinder cannot retract if the stamp cylinder is extended. | | |
| The stamp cylinder retracts if PB2 is released. | | |

36. If you answered “No” to any of the preceding questions, please revise your project, make some modifications, and download it again to the PLC.

### End of the procedure

**37.** Turn off the trainer, disconnect the leads, and clean the work surface.

## CONCLUSION

In this exercise, you opened an incomplete PLC project and you created new tags. Then, you created a ladder routine to control the logic of the application using bit instructions. Finally, you transferred, ran, and tested your program with a physical or simulated clamp and stamp application.

## REVIEW QUESTIONS

1. What is a first scan bit?

A special bit that is set to true only during the first scan of a ladder program.

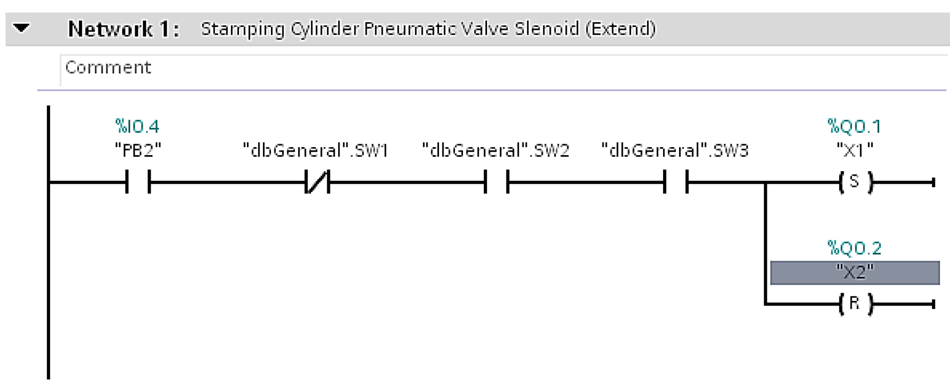
2. Which ladder instruction changes status along with the rung status that contains it?

Coil output instruction

3. Which instruction sets an addressed bit to 0 when the rung containing it is true?

Reset

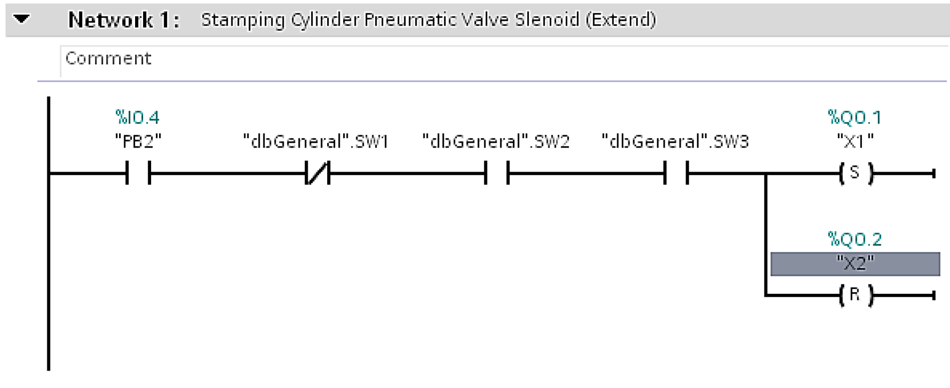
4. What are the conditions necessary to reset X2 in the rung below?




**Resetting X2.**

PB2=1
SW1=0
SW2=1
SW3=1

5. What happens simultaneously when X2 is reset in the rung below?




**Resetting X2.**

X1 is set.