EXPERIMENT 3
PROGRAMMABLE LOGIC CONTROLLER (PLC) - SIEMENS

OBJECTIVE

Understanding automation process in the industrial field is essential for an engineer. Engineering student have to be used to work and practice in industrial environment.therefore, they are ready with the real system. In this experiment, we will use PLC Siemens that shows basic automation command clearly and easy to understand. We would have introduced about basic instruction, counter, and timer.

REFERENCE

David W. Pessen, Industrial Automation: Circuit Design and Components, 2008.

EQUIPMENT REQUIRED

PLC SIEMENS S7-300	1 piece
kit PC with STEP 7	1 piece

PRE-EXPERIMENT TASK

1. What kind of industries that use or apply PLC to their system? Why industries use them?
2. What is company that produces the PLC?

INTRODUCTION

Automation System is a system that makes human life easier. It has so many advantages, because it can work effectively and efficiently in case of time, energy, and cost. because we can achieve it all, many industries use PLC. A PLC is an industrial computer control system that continuously monitors the state of input devices and makes decisions based on a custom program to control the state of output devices. Almost all production line, machine function, or process can be greatly enhanced using this type of control system. However, the biggest benefit in using a PLC is the ability to change and replicate the operation or process while collecting and communicating in vital information.

There is two type of PLC, that is Compact and Modular PLC. Compact PLCs are used in automation as replacements of relays. PLC is cheaper than some of relays because we use program to design the system and the programming is flexible and user friendly. The disadvantages of this PLC type are low memory for the program and data, low processor performance, low number of PLC timers and counters, and missing data types (like floating point and string). On the other hand, compact PLC more expensive than modular type but have the same functionalities and it is the fixed number of I/Os so we can not installing other module. The examples of compact PLC are Festo FEC PLC, Siemens Logo, and S7-200.

A modular PLC contains several different modules that can be combined together to build a customized controller. Typically, a base module contains core functions such as electrical power regulation, the computer processor, and input connections. Additional modules, including analog to digital signal converters or additional outputs, can be added to this core unit as needed.

This modular design allows a PLC to be customized and changed easily. The actual number of connections can be expanded easily by adding modules. This provides a wide range of flexibility and is typical of a modular PLC. While Ladder Logic is the most commonly used PLC programming language but it is not the only one. In fact, There are five programming language:

- **Ladder Diagram (LD)**

  Traditional ladder logic is graphical programming language. Initially programmed with simple contacts that simulated the open and closed relays, Ladder Logic programming has been expanded to include such functions as counters, timers, shift registers, and math operations.

- **Function Block Diagram (FBD)**

  The Function Block Diagram (FBD) is a graphical language for programmable logic controller design, that can describe the function between input variables and output variables. A function is described as a set of elementary blocks. Input and output variables are connected to blocks by connection lines.fbd is very useful for expressing the interconnection of control system algorithms and logic.

- **Structured Text (ST)**

  A high-level text language that encourages structured programming. It has a language structure (syntax) that strongly resembles PASCAL and supports a wide range of standard functions and operators.

- **Instruction List (IL)**

  A low level “assembler like” language that is based on similar instructions list languages found in a wide range of today’s PLCs.

- **Sequential Function Chart (SFC)**

  A method of programming complex control systems at higher structured level. A SFC program is an overview of the control system, which is the basic building blocks for entire program files. Each program file is created using one of the other types of programming languages. The SFC approach coordinates large, complicated programming tasks into smaller, more manageable tasks.

In this experiment, we will use PLC Siemens and STEP 7 for its software. PLC Siemens has many types that were S7-300 series, S7-400 series, S7-1200 series, and S7-1500 series. For this experiment, we use a S7-300 series that is shown in Figure 1.

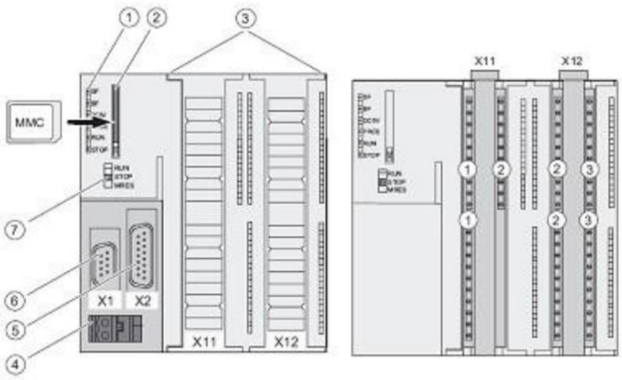


Figure 1. Siemens S7-300

Addressing PLC Siemens S7-300

Every input and output has an absolute address predefined by the hardware configuration. This address is specified directly. The absolute address can be replaced by any symbolic name you choose

1. Input Address

   Input for PLC Siemens start for address I0.0 until I65535.7. For this module, address can be use is form I124.0 until I124.7, I125.0 , and I125.1.
2. Output Address

   Output for PLC Siemens S7-300 start for address Q0.0 until Q65535.7. For this module, address can be use is form Q124.0 until Q124.5
3. Memory Address

   PLC Siemens provide a memory location with a very unique addressing. We can choose which memory will be used by selecting the address specification, which includes memory area, its address, bit number and byte number. Memory area on the PLC consist 5 kinds: I, QV, and M. all of which can be accessed as a byte, word, or double word.

Configuration PLC SIEMENS S7-300 with STEP 7 1.

1. First, run SIMATIC Manager.

Figure 2. Symbol STEP 7 SIMATIC Manager

2. Create a new project by choosing **File → New,** and give a name project you create with **“Project 1”,** and then click **OK**

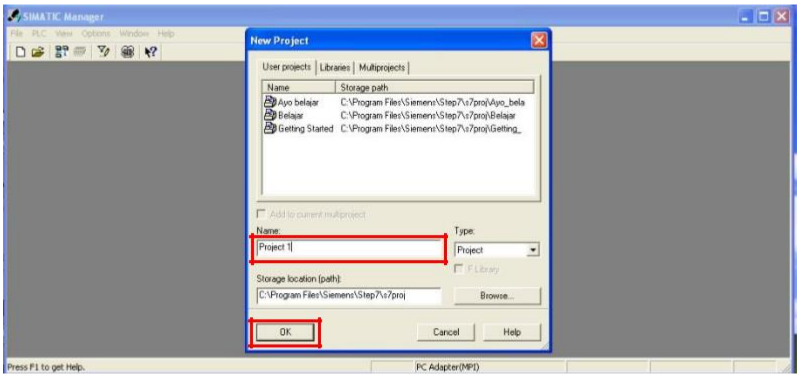


Figure 3. Menu will we create a new project

3. Choose a PLC we used, in SIMATIC Manager, **right click → insert new object → Simatic 300 Station**

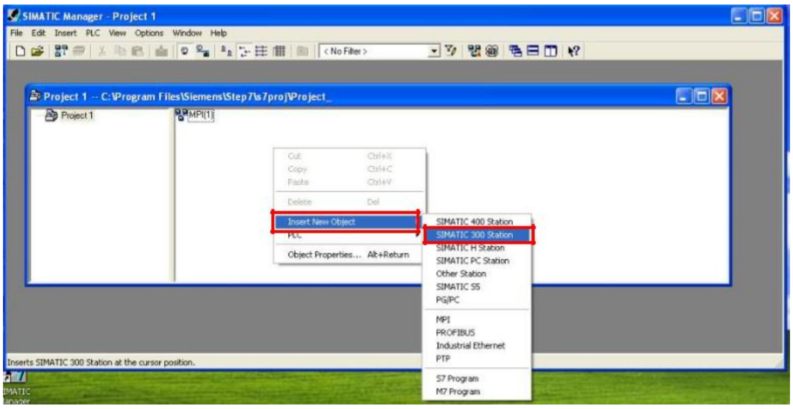


Figure 4. Screen we will choose PLC type

After we insert SIMATIC 300 station, it will appear as shown in Figure 5.

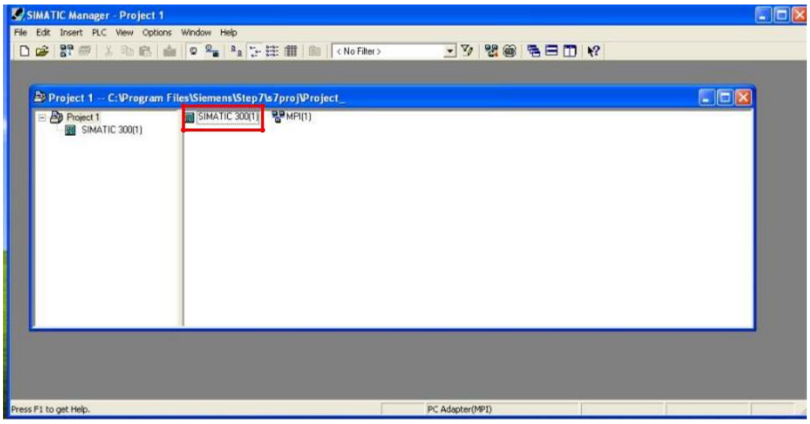
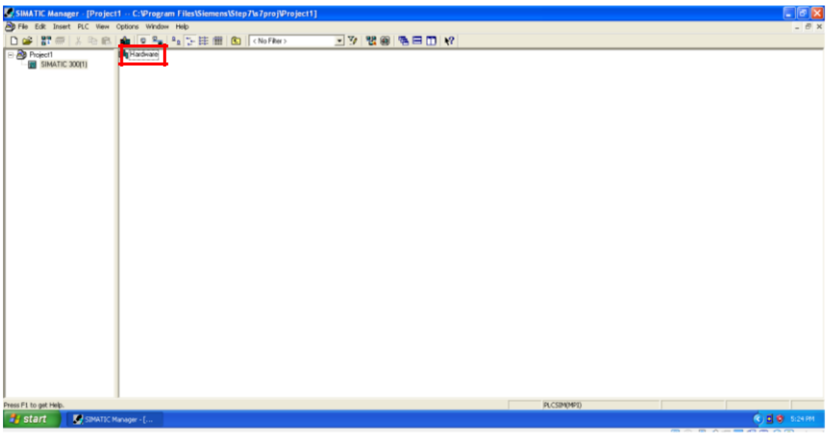


Figure 5. Screen after we insert PLC type

4. Configure hardware a PLC we used, double click **SIMATIC 300(1)** → **Hardware**



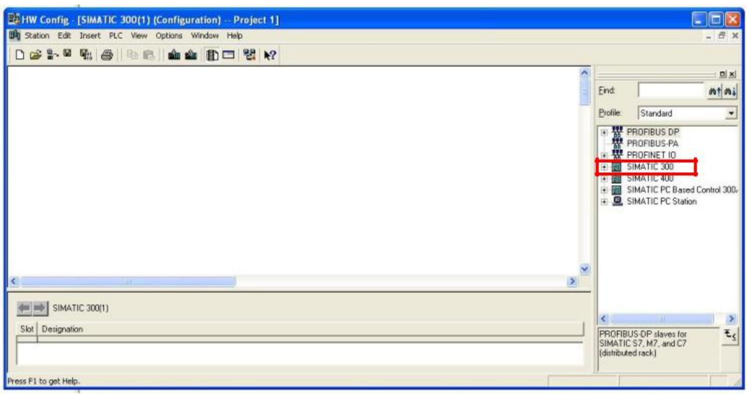


Figure 6. Screen for configuration hardware

5. Choose a CPU and I/O module we use with choose **RACK-300 → RAIL (Drag to HW Config Window)**, **CPU 300 → CPU 312C** [Part Number (PN) See in CPU Hardware] **(Drag to Rack)**, and **SM-300 → AI/AO-300** (PN See in Analog Module)

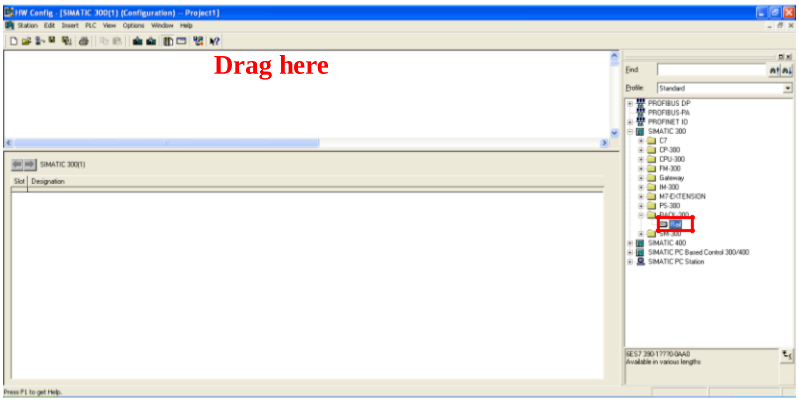


Figure 7. Select Rail in hardware configuration

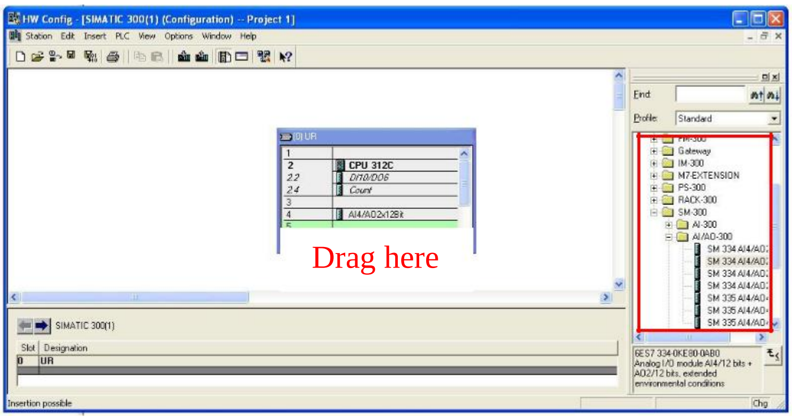


Figure 8. Select a component in hardware configuration

After choose CPU and I/O module, click **save and compile**

6. To start programming to PLC, start on click **CPU 312C** → **S7 Program (1)** → **Blocks** →**OB1**, then choose a programming language. Click on LAD (ladder) for programming language, and then screen to start programming will be open like Figure 9.

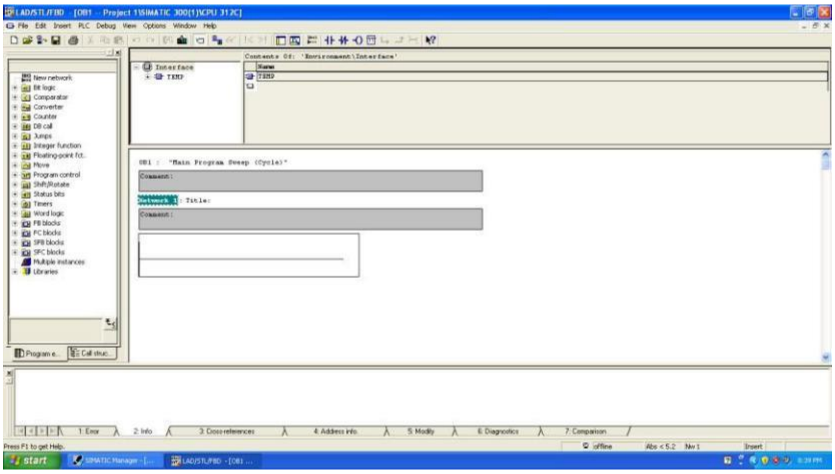


Figure 9. Block OB1

Communication PLC SIEMENS S7-300 with STEP 7

1. In **Window SIMATIC Manager**, choose **option**→ **Set PG/PC Interface**

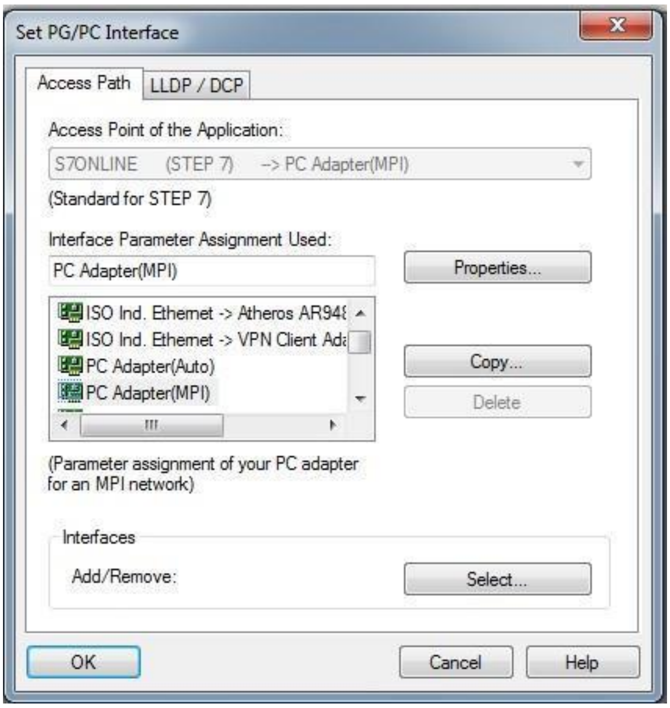


Figure 10. Menu Set PG/PC Interface

2. In the **Interface Parameter Assignment Used**, choose **PC Adapter MPI**, and then click **OK**

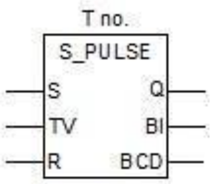
Programming Instruction STEP 7 A. Bit Logic

1. Normally Open Contact ---| |---
   This bit is used to enter input which is open normal state.
2. Normally Closed Contact ---| / |---
   This bit is used to enter input which normally closed state.
3. Invert Power Flow --|NOT|--
   This bit is used to reverse the value of the logic operation before.
4. Output Coil ---( )
   Output coil work like a coil in a relay logic diagram. This bit is used to indicate the value of the result of logic operation.

B. Timer

Timer is an instruction which serves to provide a time delay (delay). The timer can be used to set the length of time on or off as output. In STEP 7 Programming, there are various kinds of timers, such as pulse timer (S_Pulse), on delay timer (S_ODT), and off delay timer (S_OFFDT).

1 Pulse Timer (S_PULSE)



The timer is used to specify the delay time a logic operation

2 On Delay Timer (S_ODT)

This timer is used to delay on a logic operation

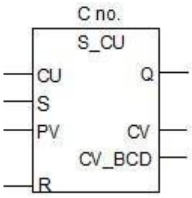
3 Off Delay Timer (S_OFFDT)

This timer is used to delay off a logic operation

C. Counter

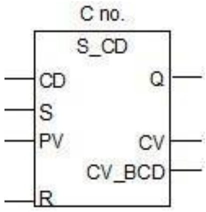
Counter is an instruction that is used for the calculation operation. There are several types of programming counter at STEP 7, such as counter up (S_CU), counter down (S_CD),and counter up / down (S_CUD)

1 Counter Up (S_CU)



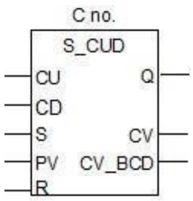
This counter is a function that could counting from zero to a value that have assigned

2 Counter Down (S_CUD)



Counter down is a function that could counting from value that have assigned to zero

3 Counter Up / Down (S_OFFDT)



Counter up / down is used to calculate a sequence of up or down

EXPERIMENT

Experiment 1. Basic Instructions and Logic Circuit

A. Operational Procedure

1. Configure Ladder Diagram for contacts and coils as shown in Figure 12.
2. Place each contact (Normally Open and Normally Close) for NO coil.
3. Write condition of lamp when it is pressed and released.

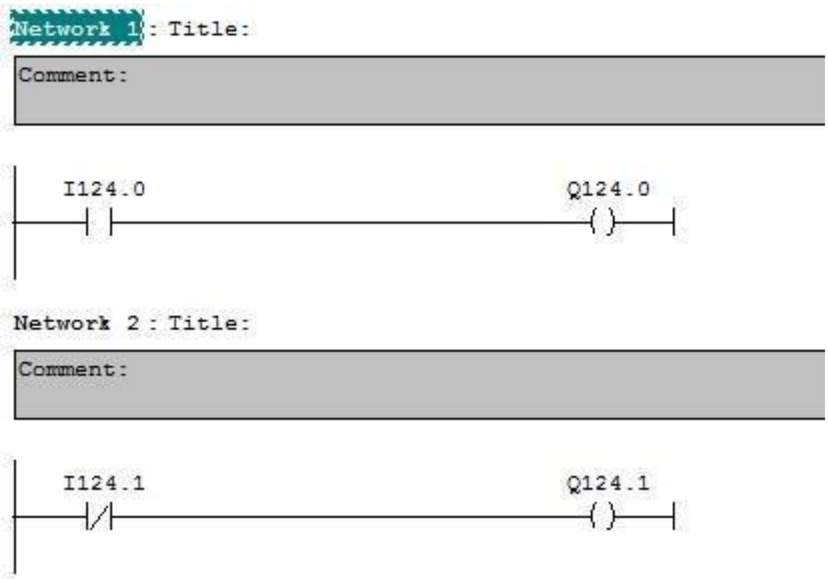


Figure 12. Ladder diagram of contacts and coil circuit experiment

4. Configure ladder diagram for AND circuit as shown in Figure 13
5. Place two NO contact consecutively with a NO coil. Place both of them in same row with NO coil
6. Write condition of lamp when it is pressed and released

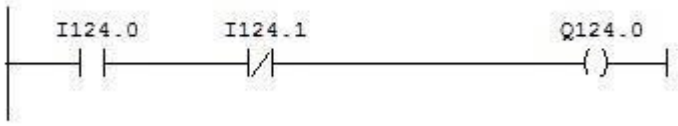


Figure 13. Ladder diagram of AND circuit

7. Configure Ladder Diagram for OR circuit as shown in Figure 14.
8. Place two NO contact consecutively with a NO coil. Place one of them in same row with NO coil and the other one in second row, connected parallelly with first row
9. Write condition of lamp when it is pressed and released

Figure 14. Ladder diagram of OR circuit

B. Experimental Data

All of condition, experience and image that you observe after contacts and coils experiment can be written in Table A, for AND & OR circuit experiment can be written in Table B

Table A

contact	Condition	Lamp condition
Normally Open	1	
	0	
Normally Closed	1	
	0	

Table B

Logic	Contact State		Lamp Condition
	Normally Open	Normally Open 2	
AND	0	0	
	0	1	
	1	0	
	1	1	
OR	0	0	
	0	1	
	1	0	
	1	1	

Information:
0 = Released **1 = Pressed**

C. Analysis and Experiment Task

1. What is difference between contacts and coils? How about Normally Open and Normally Close?

Experiment 2. Self-Holding Circuit

A. Operational Procedure

1. Configure ladder diagram for self-holding circuit as shown in Figure 15.

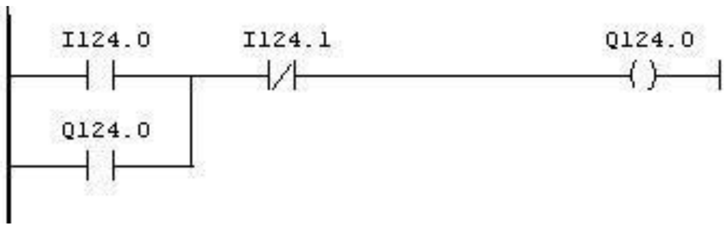


Figure 15. Self-holding circuit

2. Place two NO contact, and one NC contact with a NO coil. Place NO contact in same row with NC contact, and connect with NO coil. The other NO contact in second row, connected parralelly with first NO contact. Address for this contact is same with NO coil.
3. Write condition of lamp when it is pressed and released

B. Experimental Data

All of condition, experience and image that you observe after contacts and coils experiment can be written in Table C

Table C		
Contact condition		Lamp condition
Normally open	Normally closed	
0	0	
0	1	
1	0	
1	1	

Information: **1 = pressed** **0 = released**

C. Analysis and Experiment Task

1. Why should we put an NC contact in the self-holding circuit?
2. Make a ladder design of blinking lamp with self-holding circuit. Explain your design!

Experiment 3. Timer

A. Operational Procedure

1. Configure ladder diagram for timer on delay as shown in Figure 16.

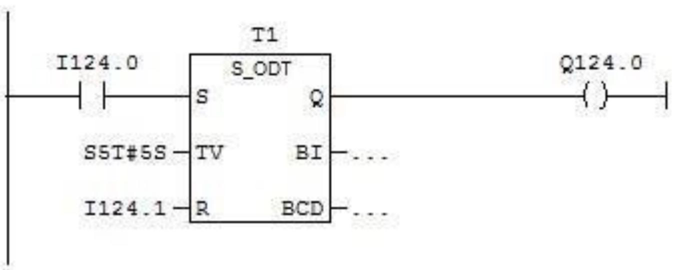


Figure 16. Ladder diagram of Timer On Delay

2. Write the condition of lamp when the contact is pressed and released
3. Configure ladder diagram for timer off delay as shown in Figure 17.

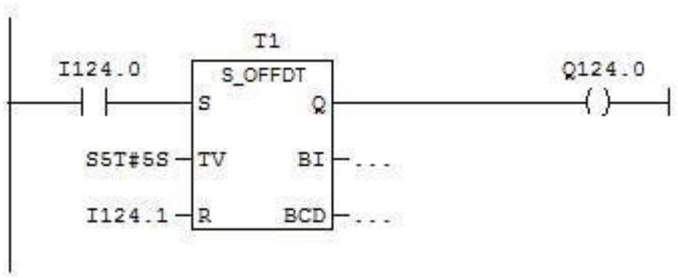


Figure 17. Ladder diagram of Timer Off Delay

4. Write condition of lamp when the contact is pressed and released

B. Experimental Data

All of condition, experience and image that you observe after timer on delay experiment can be written in Table D and Timer Off-Delay experiment in Table E

Table D	**Time (seconds)**	**Lamp Condition**
Contact Condition	0	
	5	
Timing Diagram		
Input		
Output		

Table E	**Time (seconds)**	**Lamp Condition**
Contact Condition	0	
	5	
Timing Diagram		
Input		
Output		

C. Analysis and Experiment Task

1. How many variables in the timer function? Explain!
2. Could we make two Timer function in a rung? Explain!

Experiment 4. Counter

A. Operational Procedure

1. Configure ladder diagram for counter up as shown in Figure 18

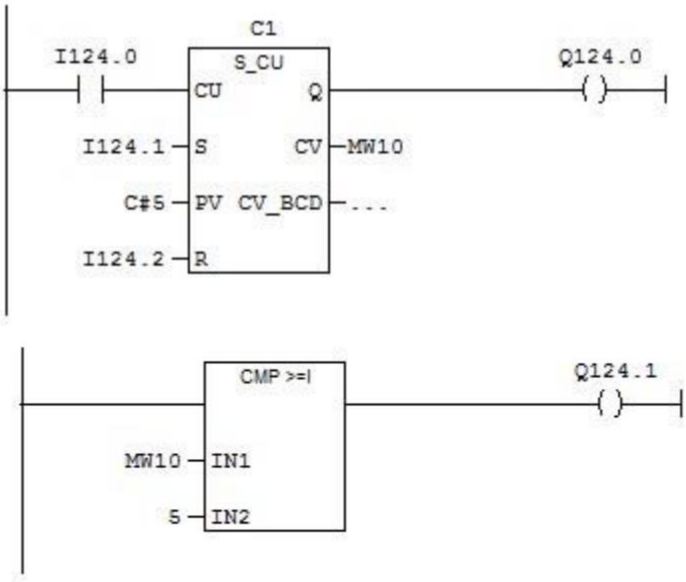


Figure 18. Ladder diagram of Counter Up

2. For counter down, configure ladder diagram as shown in Figure 19

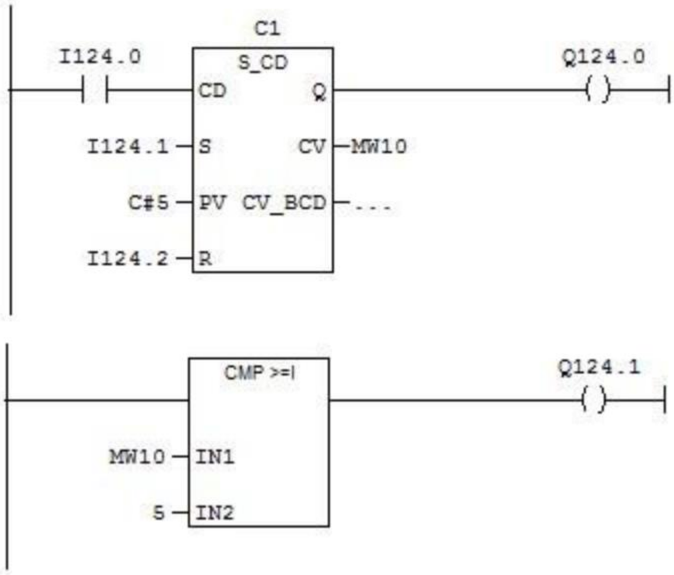


Figure 19. Ladder diagram of Counter Down

3. Write condition of lamp

B. Experimental Data

All of condition, experience and image that you observe after counter up experiment can be written in Table F and counter down experiment in Table G

Table F	Count	Lamp Condition
Contact	0	
	5	

Table G	Count	Lamp Condition
Contact	0	
	5	

C. Analysis and Experiment Task

1. How many variables in the counter function? Explain?
2. Draw a state diagram input and output for counter experiment!