

EVERYTHING ABOUT

# PLC PROGRAMMING

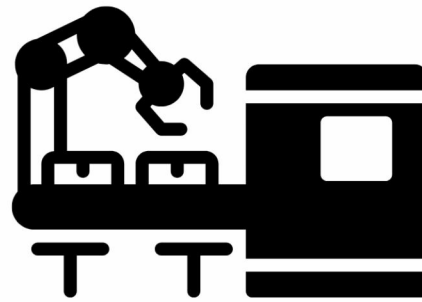
*Practical lessons on PLC programming using AB, Siemens, and Mitsubishi PLCs with examples*



**AVINASH MALEKAR**

# **EVERYTHING ABOUT PLC PROGRAMMING**

Practical lessons on PLC programming using Allen Bradley, Siemens, and Mitsubishi PLCs with examples



**Avinash Malekar**

**Copyright © 2021 Avinash Prakash Malekar**

**All rights reserved.**

**ISBN: 9798766321217**

Dedicated to.....

# **Learners**

# Contents

[Preface](#)

## [INTRODUCTION](#)

[1.1 Basics of PLC](#)

[1.2 Processor Memory Organization](#)

[1.3 Types of PLC](#)

[1.4 PLC manufacturers and their market share](#)

[1.5 PLC programming languages](#)

[1.6 PLC program execution](#)

## [ALLEN BRADLEY PLC](#)

[2.1 AB PLC series](#)

[2.2 Allen bradley PLC wiring configuration](#)

[2.3 PLC and their programming software](#)

[2.4 Allen bradley communication protocols](#)

[2.5 Addressing for ladder logic programming](#)

[2.6 Digital signal programming](#)

[2.7 Latch / Unlatch instructions](#)

[2.8 Use of timer and its types](#)

[2.9 Use of counter](#)

[2.10 Math instructions](#)

[2.11 Compare instructions](#)

[2.12 Move and copy instructions](#)

[2.13 Analog signal programming](#)

[2.14 Other instructions](#)

[2.15 RS logix 500 PLC software](#)

[2.16 PLC programming examples using rslogix 500 software](#)

[1. Logic gates](#)

[2. Water level control](#)

[3. Milk bottle filling and capping application](#)

[4. Packaging application](#)

[5. Weighing application](#)

[6. Boiler application](#)

[Case study 1-Hydraulic press machine](#)

[SIEMENS PLC](#)

[3.1 Siemens PLC series](#)

[3.2 Siemens S7- 1200 PLC wiring configuration](#)

[3.3 PLC and their programming software](#)

[3.4 Siemens communication protocols](#)

[3.5 Addressing for ladder logic programming](#)

[3.6 Digital signal programming](#)

[3.7 Bit logic](#)

[3.8 Timer](#)

[3.9 Counter](#)

[3.10 Math instructions](#)

[3.11 Comparator](#)

[3.12 Move, jump, and call instructions](#)

[3.13 Analog signal programming](#)

[3.14 Converter](#)

[3.15 Simatic manager step7 software](#)

[3.16 Plc programming examples using Simatic manager step 7 software](#)

[1. Tower lamps control](#)

[2. Traffic light control](#)

[3. Box sorting application](#)

[4. Material conveying application](#)

[5. Box lifter](#)

[6. Reaction tank](#)

[Case study 2- Pneumatic hammering machine](#)

[Mitsubishi PLC](#)

[4.1 Mitsubishi PLC series](#)

[4.2 Mitsubishi PLC wiring configuration](#)

[4.3 PLC and their programming software](#)

[4.4 Mitsubishi communication protocols](#)

[4.5 Addressing for ladder logic programming](#)

[4.6 Digital signal programming](#)

[4.7 Rising / Falling pulse instructions](#)

[4.8 Use of a timer](#)

[4.9 Use of counter](#)

[4.10 Math instructions](#)

[4.11 Compare instructions](#)

[ZCP Instruction](#)

[4.12 Move, jump, and logical instructions](#)

[4.13 Analog signals](#)

[4.14 Other instructions](#)

[4.15 Mitsubishi GX works2 PLC programming software](#)

[4.16 PLC programming examples using works2 software](#)

[1. Conveyor application](#)

[2. Heating application](#)

[3. Flash light](#)

[4. Home automation](#)

[5. Shrink tunnel](#)

[6. Production report](#)

[Case study 3- Road roller assembly line](#)

[Assignments](#)





# Preface

In the previous 'Everything about factory automation' book, we had an introduction to the basics of the factory automation. We came to know that a PLC is an inevitable part of an industrial automation. An industry cannot be automated without the aid of a PLC.

There are a number of PLC manufacturers available in the market each PLC has its different aspects. Even though they are dissimilar, they work on the same principle.

In this book, we will dig deeper into the basics and advanced PLC programming. We are going to learn about Allen Bradley, Siemens, and Mitsubishi PLC, their programming software with real-world examples.

## ***What makes this book different?***

- ***Well organized information***
- ***Simple diagrams***
- ***Digestible lessons***
- ***Programming software elaboration***

## **Important vocabulary**

NO - Normally Open

NC - Normally Closed

PLC - Programmable Logic Controller

HMI - Human Machine Interface

Bool - Single bit

Integer - 8 bits

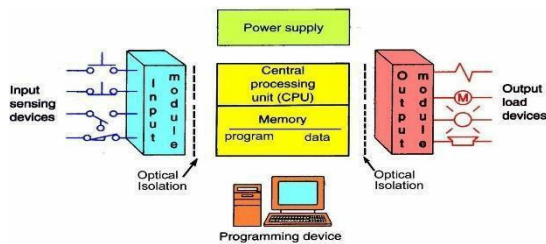
Double integer - 16 bits

# 1. INTRODUCTION

## 1.1 Basics of PLC

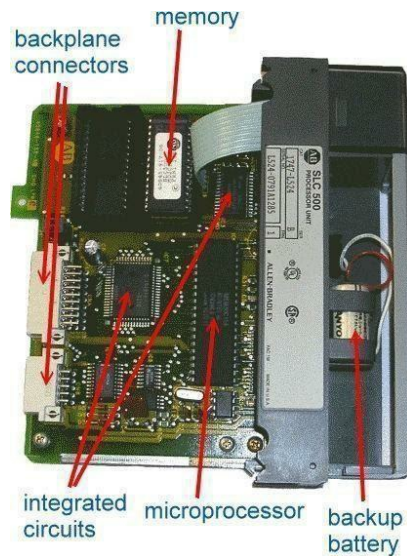
**National Electrical Manufacturers Association (NEMA)** defines PLC as a digital electronic device that uses a programmable memory to store instructions and to implement specific functions such as logic, sequence, timing, counting, and arithmetic operations to control machines and processes.

- The PLC is an assembly of solid-state digital logic elements designed to make logical decisions and provide outputs.
- PLC is a programmed interface between input sensors & output devices.



**The entire PLC system can be divided as follows.**

- a) Central Processing Unit
  - b) Input Modules
  - c) Output Modules
  - d) Power Supply
  - e) Bus system
- a) Central Processing Unit**



- The brain of the whole PLC is the CPU module. This module typically lives in the slot beside the power supply.
- Manufacturers offer different types of CPUs based on the complexity needed for the system.
- The CPU consists of a microprocessor, memory chip, and other integrated circuits to control logic, monitoring, and communications.
- The CPU has different operating modes.
- In programming mode, it accepts the downloaded logic from a PC.
- The CPU is then placed in the run mode so that it can execute the program and operate the process.

## b) Input Module

These modules act as an interface between the real-time status of the process variable and the CPU.

**Analog input module:** Typical input to these modules is 4-20 mA, 0-10 V

Ex: Pressure, Flow, Level Tx, RTD (Ohm), Thermocouple (mV).

**Digital input module:** Typical input to these modules is 24 V DC, 115 V AC, 230 V AC.

|   |                 |   |                     |
|---|-----------------|---|---------------------|
|  | Selector switch |  | Through beam sensor |
|  | Limit switch    |  | Inductive sensor    |
|  | Push button     |  | Limit Switch        |

### c) Output module

These modules act as a link between the CPU and the output devices in the field.

**Analog output module:** Typical output from these modules is 4-20 mA, 0-10 V

Ex: Control Valve, Speed, Vibration

**Digital output module:** Typical output from these modules is 24 V DC.

|   |               |   |               |
|---|---------------|---|---------------|
|  | Valve         |  | Buzzer        |
|  | Lamp          |  | Relay         |
|  | Solenoid coil |  | Motor starter |
|  | Alarm         |  | Fan           |

### d) Power Supply

- The power supply gives the voltage required for the electronics module (I/O module, CPU module memory unit) of the PLC from the line supply.
- The power supply provides the isolation necessary to protect the solid-state devices from the highest voltage line spikes.
- As I/O is expanded, some PLC may require additional power supplies in order to maintain proper power levels.

**e) Bus System**

- It is the path for the transmission of the signal between the power supply module, CPU, and I/O modules.
- The bus consists of several single lines i.e. wires or tracks.
- Data bus
- Address bus
- Control bus

## 1.2 Processor Memory Organization

The memory of a PLC is organized by type.

The memory space can be divided into two broad categories:

1. Program Memory
2. Data memory.

Advanced ladder logic functions allow controllers to perform calculations, make decisions, and do other complex tasks. Timers and counters are examples of ladder logic functions. They are more complex than basic input contacts and output coils and rely heavily upon data stored in the memory of the PLC.

A memory map can be used to show how memory is organized in a PLC.

### 1. Data table

- Input/output locations
- Internal relay and timer/counter locations

### 2. User program

The user program causes the controller to operate in a particular manner

### 3. Housekeeping memory

Used to carry out functions needed to make the processor operate (no access by user).

## 1.3 Types of PLC

**1. Compact PLC:-** Inbuilt power supply, I/O modules, CPU, Communication Ports.

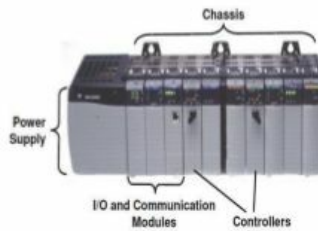


**2. Modular PLC:-** Chassis System, Separate power supply module, CPU Module, I/O modules.



**3. Sidewise:-** Nano, micro, medium.

Rockwell  
Alpha-Ready PLC  
SLC 500 PLC





## 1.4 PLC manufacturers and their market share

According to research by grandviewresearch, the global industrial automation, and control system market was worth almost 126.3 billion USD and expected to rise 8.6% up to 2025. Companies are significantly reducing labor and operational expenses, and additionally minimizing human errors by the use of industrial robots through automation.

The manufacturers are using various control systems for their industrial processes. The primary control systems are Programmable Logic Controller(PLC), Supervisory Control And Data Acquisition(SCADA), Distributed Control System(DCS), and Human Machine Interface(HMI).

Based on the region, the market for industrial automation has been categorized into North America, Europe, Asia Pacific, Latin America, and the Middle East and Africa.

A large portion of the growth is expected to come from the Asia Pacific. Kawasaki Robotics, Mitsubishi Electric Factory Automation, Yokogawa Electric Corporation are some of the key players from the Asia Pacific.

### PLC brands list by country-wise

You can find the countrywide PLC brands list in the below-mentioned table.

| Country | Leading Automation Brands    |
|---------|------------------------------|
| Japan   | Omron Industrial Automation  |
|         | Yaskawa Electric Corporation |
|         | Mitsubishi Electric          |
|         | Fuji Electric                |
|         | Yokogawa Electric            |

|                      |                                   |
|----------------------|-----------------------------------|
|                      | Toshiba                           |
|                      | Panasonic                         |
|                      | Keyence                           |
| <b>United States</b> | Emerson Electric                  |
|                      | Rockwell Automation               |
| <b>Germany</b>       | Siemens                           |
|                      | Bosch                             |
|                      | Phoenix Contact                   |
| <b>Switzerland</b>   | ABB                               |
| <b>France</b>        | Schneider Electric                |
| <b>Taiwan</b>        | Delta Electronics                 |
|                      | Fatek                             |
| <b>Israel</b>        | Unitronics                        |
| <b>China</b>         | Wecon Technology                  |
|                      | Kinco                             |
| <b>India</b>         | RS Enterprises                    |
|                      | General Industrial Controls (GIC) |

The complete list of the 17 most popular PLCs and their manufacturing company, according to market share, is shown below:

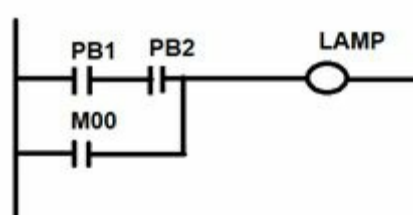
| <b>Market Share Ranking</b> | <b>PLC Manufacturers</b>     | <b>PLC Brand Name/s</b>    |
|-----------------------------|------------------------------|----------------------------|
| 1                           | Siemens                      | Simatic                    |
| 2                           | Rockwell Automation          | Allen Bradley              |
| 3                           | Mitsubishi Electric          | Melsec                     |
| 4                           | Schneider Electric           | Modicon                    |
| 5                           | Omron                        | Sysmac                     |
| 6                           | Emerson Electric (GE)        | RX3i & VersaMax (GE Fanuc) |
| 7                           | Keyence                      | KV & V-8000                |
| 8                           | ABB (B&R Automation)         | AC500 X20 & X90            |
| 9                           | Bosch                        | Rexroth ICL                |
| 10                          | Hitachi                      | EH & H                     |
| 11                          | B&R Automation (part of ABB) | X20 & X90                  |
| 12                          | Phoenix Contact              | AXC                        |
|                             |                              |                            |

|    |                    |                                |
|----|--------------------|--------------------------------|
| 13 | Panasonic          | FP                             |
| 14 | LS Electric (LSIS) | XG, Master-K & GM              |
| 15 | Eaton              | XC & EasyE4<br>(Cutler-Hammer) |
| 16 | Delta Electronic   | DVP, AS & AH                   |
| 17 | Fuji Electric      | Micrex                         |

## 1.5 PLC programming languages

PLC programming languages are defined by the international electrotechnical commission (IEC) under sections 61131-3 standards are explained below.

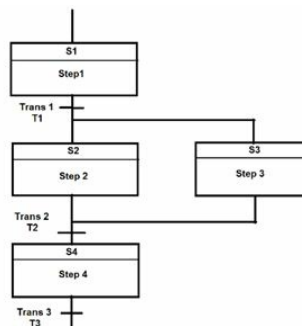
### 1. Ladder logic (LD)



- Modeled from relay logic panel.
- Easy to understand and troubleshoot.
- Difficult for motion and batching programming

### 2. Sequential function chart (SFC)

- Language is similar to flow charts.
- The process can be broken into steps.
- To have direct access to the logic for a piece of equipment fault.
- Minimal programming effort and greater clarity through graphic programming.
- Excellent legibility for maintenance personnel.

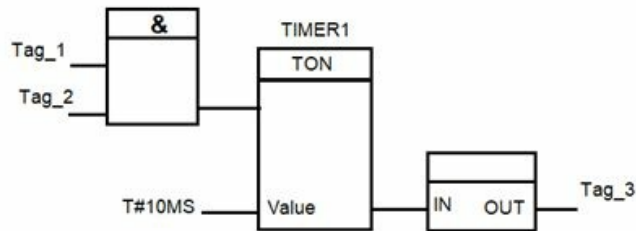


- Less time in the commissioning phase thanks to the graphical programming interface.
- Minimal implementation effort because there are few possibilities for errors when generating code.
- High availability of the machine through process diagnostics

functions (interlock and supervision).

- Fast error detection through PLC code display and criteria analysis on the HMI.

### 3. Functional block diagram (FBD)



- FBD describes functions between inputs and outputs that are connected by connection lines.
- It is in the graphical form language with repeated blocks.
- It is good for motion control programming.
- It can combine many lines of the program into a single block.
- But it is difficult to troubleshoot using this language.

### 4. Structured text (ST)

```
IF #IN1 & #IN2,  
THEN  
#OUT1 :=TRUE;  
END IF;  
"IEC_Timer_0_DB".TON (IN:=10,  
                        PT:=15,  
                        Q=>10,  
                        ET=>10):
```

- Similar to C language, this language uses instructions like- for, while, if, else, case, if-else, etc.
- Very organized and good for computing large mathematical calculations.
- It covers some instructions that are not available in the ladder diagram.
- It is difficult to edit online.

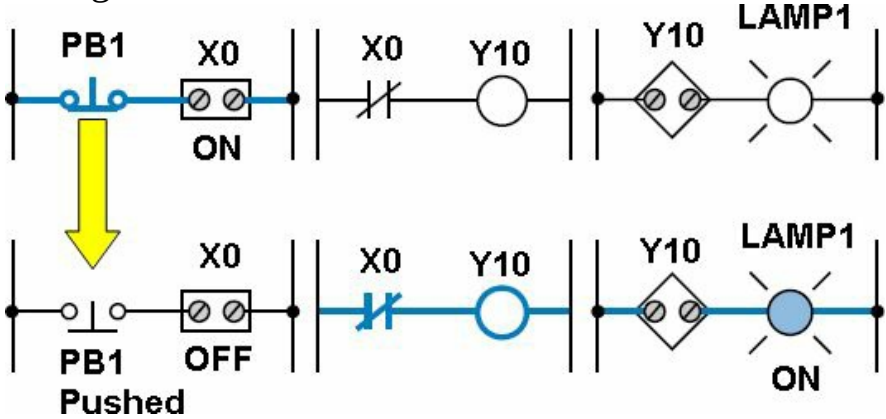
### 5. Instruction List (IL)

| Network 1: .... |           |                          |     |       |
|-----------------|-----------|--------------------------|-----|-------|
| Comment         |           |                          |     |       |
|                 |           |                          | RLO | Value |
| 1               | A         | "Motor_1_Enabled"        | 1   | 1     |
| 2               | AN        | "Motor_1_EmergencyStop"  | 0   | 1     |
| 3               | JC        | n_OK                     | 1   |       |
| 4               | =         | "Motor_1_Start"          | 1   | 1     |
| 5               | AN        | "Motor_1_SpeedOK"        | 0   | 1     |
| 6               | AN        | "Motor_1_BreakesEnabled" | 0   | 0     |
| 7               | =         | "Motor_1_Stop"           | 0   | 0     |
| 8               | JU        | End                      |     |       |
| 9               | n_OK: SET |                          |     |       |
| 10              | AN        | "Motor_1_BreakesEnabled" |     |       |
| 11              | =         | "Motor_1_Stop"           |     |       |
| 12              | End: NOP  | 0                        | 0   |       |

- Instruction list language consists of predefined instructions, which are used to program a PLC.
- The use of instructions makes this program very compact.
- It is difficult to edit online.

# 1.6 PLC program execution

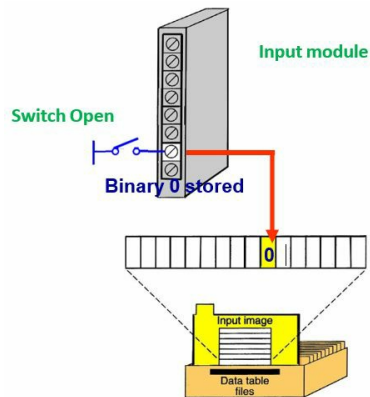
The below circuit shows the physical wiring along with the programming rung of the wiring.



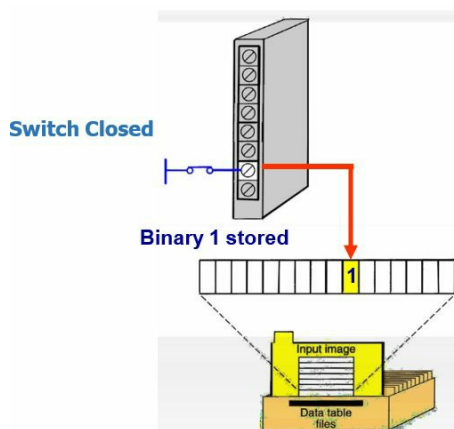
## Program execution process

- a) Input Table File Operation



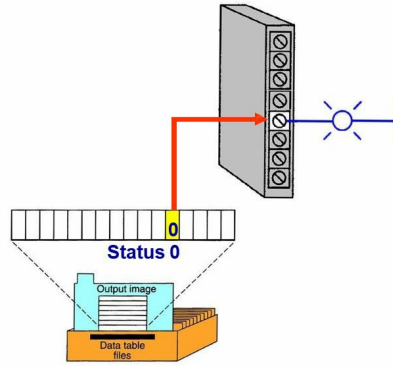


When the switch is open, 0 is stored in the file.

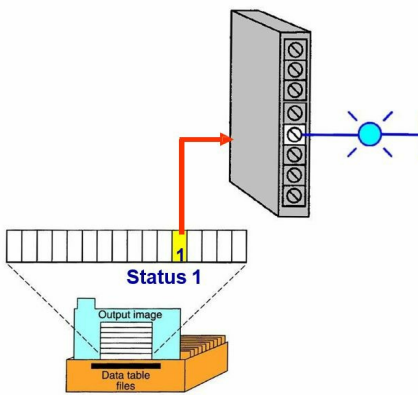


Once the switch is closed, 1 is stored in the file.

## b) Output Table File Operation



Since 0 is stored in the file, a lamp will not glow.



Once 0 becomes 1, the lamp will glow.

## 2. ALLEN BRADLEY PLC



Allen-Bradley became involved with programmable logic controllers by the inventor, Odo Josef Struger. He is often referred to as the father of the programmable logic controller.

Odo contributed the ideas leading up to the invention of Allen-Bradley PLCs over a period from 1958 to 1960 and is accredited with inventing the acronym PLC.

Allen-Bradley became a major PLC manufacturer in the United States during his employment with AB.

He also contributed as a leader in the development of IEC 61131-3 PLC programming language standards.

## 2.1 AB PLC series

### A. Pico PLC

Bulletin 1760 Pico Programmable Logic Controllers and PicoGFX Controllers are discontinued and no longer available for sale. AB recommends these replacement products by Micro Series PLC.

### B. Micro800 Control Systems

#### 1. Micro810 Controllers



These controllers support 12 I/O points with 4 high current relay outputs (8A) for smart relay applications.

#### 2. Micro820 Controllers



These controllers support up to 36 I/O points with many embedded features such as Ethernet, micro SD slot for recipe and data log, and analog I/O.

#### 3. Micro830 Controllers



These controllers support up to 88 I/O points with high-performance I/O, interrupts, and PTO motion.

#### 4. Micro850 Controllers



These controllers support up to 132 I/O points with high-performance I/O, interrupts, and PTO motion plus embedded Ethernet and 2085 expansion I/O.

### C. MicroLogix Control Systems

**1. MicroLogix 1000 Controllers:** Discontinued and no longer available for sale

**2. MicroLogix 1100 Controllers:** The built-in LCD panel shows controller status, I/O status, and simple operator messages. With 2 analog inputs, 10 digital inputs and 6 digital versions. You can expand the I/O count using rackless I/O modules.

**3. MicroLogix 1400 Controllers:**



Controllers without embedded analog I/O points provide 32 digital I/O points, while analog versions offer 32 digital I/O points and 6 analog I/O points. You can expand all versions with up to seven 1762 expansion I/O modules.

**4. MicroLogix 1500 Controllers:** Controller Systems will be discontinued and no longer available for sale outputs, the MicroLogix 1100 controller can handle a wide variety of tasks.

**5. MicroLogix 1200 Controllers:** The controller is available in 24-point and 40-point

## **D. ControlLogix Control Systems**

The ControlLogix family was introduced in 1997. This platform was rack-based having much faster scan times (speed) and memory than the PLC-5 or SLC products. Communication modules supported Ethernet, DeviceNet, DH485, and ControlNet.

Servo motor control cards using SERCOS fiber-optics provided competitiveness in the coordinated system servo market.

## **E. CompactLogix Control Systems**

The CompactLogix family was released in 2008 as a lower-cost solution to ControlLogix for competitive reasons.

And like the MicroLogix, the products do not use a rack-based solution but instead, use add-on modules to the ends of the power supply or CPU modules.

### **1. 1769 CompactLogix 5370 Controllers**

Ideal for small, to mid-size applications that require low axis motion and I/O point counts

## 2. 5069 CompactLogix 5380 Controllers

Ideal for small to midsize applications that require low axis motion and I/O point counts

## 3. 1768 CompactLogix Controllers

Let you control distributed I/O via EtherNet/IP, ControlNet, or DeviceNet

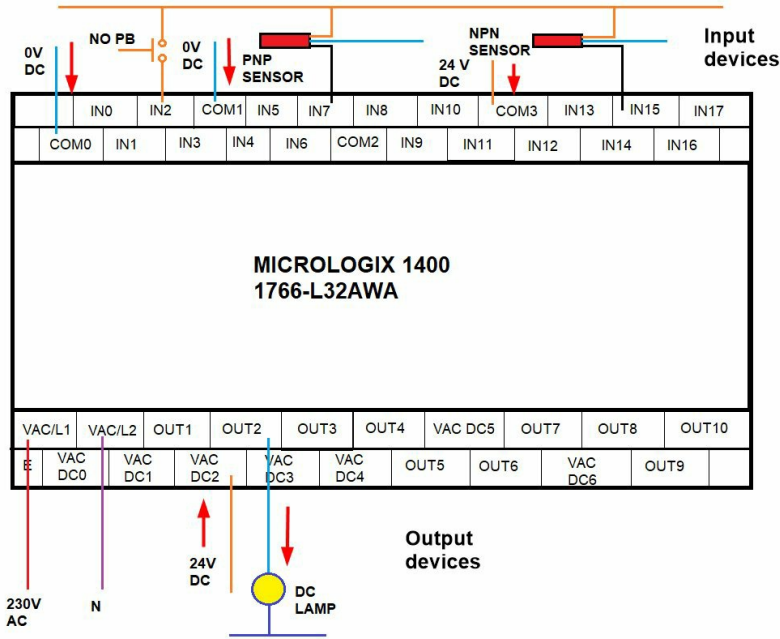
## 4. 1769 CompactLogix L23x and L3x Controllers

Let you control distributed I/O via EtherNet/IP, ControlNet, or DeviceNet

### Data types

| Data Type      | Abbreviation | Memory bits | Range                               |
|----------------|--------------|-------------|-------------------------------------|
| Boolean        | BOOL         | 1           | 0-1                                 |
| Short Integer  | SINT         | 8           | -128 to 127                         |
| Integer        | INT          | 16          | -32,768 to 32,767                   |
| Double Integer | DINT         | 32          | -2,147,483,648 to 2,147,483,647     |
| Real Number    | REAL         | 32          | +/-3.402823E38 to +/- 1.1754944E-38 |

# 2.2 Allen bradley PLC wiring configuration





## 2.3 PLC and their programming software

| AB PLC       | Programming Software |
|--------------|----------------------|
| Pico PLC     | Picosoft             |
| Micro        | CCW                  |
| Micrologix   | RSLogix 500          |
| SLC 500      | RSLogix 500          |
| CompactLogix | RSLogix 5000         |
| ControlLogix | Studio 5000          |

## 2.4 Allen bradley communication protocols



### 1. DF1 protocol

- It goes about the communication protocol designated for the classic serial link (RS232/RS422). The protocol allows the communication of either Full-duplex (point-to-point type of connection, RS232) or Half-duplex with addressing the PLC stations. This protocol is supported by devices that work with data areas (Data File).
- **Example: SLC 500, Micrologix, etc.**



### 2. Ethernet/IP protocol

- It is built on the NetLinx architecture that is implemented even in DeviceNet and ControlNet networks. Ethernet/IP is administered by the ODVA independent organization that even takes charge of DeviceNet. This protocol is supported by devices that work with data objects (contrary to the data areas in DF1 protocol).
- **Example: Micro 820, controlLogix, compactlogix, softlogix, drive logix, guardlogix, etc.**

### 3. Modbus protocol

- [Modbus](#) is a standard, very commonly used protocol for the serial

link and Ethernet (the PLC is slave). The Modbus RTU protocol is supported for example by: Micro820, MicroLogix 1200/1100/1500. The Modbus TCP/IP protocol is supported.

- **Example: Micro 820, Micrologix 1400.**

#### 4. ASCII protocol

- It goes about the configurable char communication protocol designated for the classic serial link (RS232/RS422). This protocol is supported by some PLC's.
- **For example Micro 820, compactlogix (on the second serial link port), micrologix 1100/1200/1500, etc.**

#### 5. DH+ protocol

- It supports 64 nodes at the most, 230 kB/s (SLC 501 to 3 can't it). It is the proprietary Allen- Bradley protocol. Two methods can be used for communication.



#### 6. DH485 protocol

- It goes about the communication protocol designated for the serial link. It supports 32 nodes at the most, 19 kB/s (multi-master = each-to-each). It is the proprietary Allen-Bradley protocol. Two methods can be used for communication.

## 2.5 Addressing for ladder logic programming

| File | Type    | Description  |
|------|---------|--|
| O0   | Output  | This file stores the state of output terminals for the controller.   |
| I1   | Input   | This file stores the state of input terminals for the controller.  |
| S2   | Status  | This file stores controller operation information useful for troubleshooting controller and program operation                  |
| B3   | Bit     | This file stores internal relay logic  |
| T4   | Timer   | This file stores the timer accumulator and preset values and status bits   |
| C5   | Counter | This file stores the counter accumulator and preset values and status bits.  |
| R6   | Control | This file stores the length, pointer position, and status bits for control instructions such as shift registers and sequencers |
|      |         |  |

|    |                |  |
|----|----------------|--|
| N7 | Integer        | This file is used to store bit information or numeric values with a range of -32767 to 32768 |
| F8 | Floating-point | This file stores a # with a range of 1.1754944e-38 to 3.40282347e+38.                        |

## 2.6 Digital signal programming

### 1. XIO



**Examines a bit for an off condition.**

Use an XIO instruction in your ladder logic to determine if a bit is off.

1 = True

0 = False

#### **Devices**

- Start/Stop push buttons
- Selectors
- Limit switch
- Proximity switch
- Light
- Internal bit

### 2. XIC



**Examines a bit for an On condition**

Use the XIC instruction in your ladder logic to determine if a bit is ON.

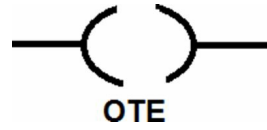
0 = False

1 = True

#### **Devices**

- Start/Stop push buttons
- Selectors
- Limit switch
- Proximity switch
- Light
- Internal bit

### 3. OTE



**Turns a bit on or off.**

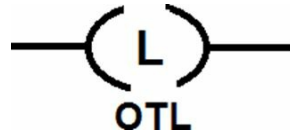
Use OTE instruction in your ladder logic to turn on a bit when rung condition is evaluated as true.

#### **Devices**

- Light
- Motor run signal
- Internal bits

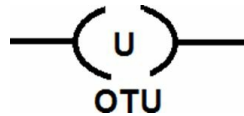
## 2.7 Latch / Unlatch instructions

### 1. OTL



This instruction functions much the same as the OTE with the exception that once a bit is set with an OTL, it is "Latched" on, once an OTL bit has been set "ON" (1 in Memory) it will remain "ON" even if the rung condition goes false the bit be reset with an OTU instruction. Latch and unlatch instructions must be assigned the same address in your logic program.

### 2. OTU

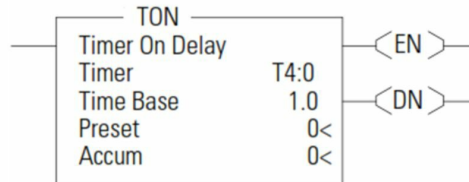


Use this output instruction to unlatch (Reset) a latched (Set) bit which was set by an OTL instruction. The OTU address is identical to the OTL address which originally set the bit.



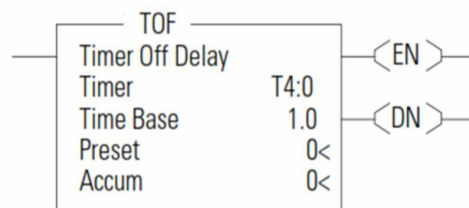
## 2.8 Use of timer and its types

### 1. TON- Timer, On Delay



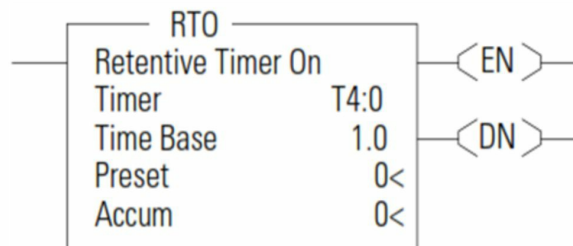
| Bit              |                          | Is Set When:  |
|------------------|--------------------------|---|
| bit 13 - T4:0/DN | <b>DN - timer done</b>   | accumulated value $\geq$ preset value                     |
| bit 14 - T4:0/TT | <b>TT - timer timing</b> | rung state is true and accumulated value $<$ preset value |
| bit15 - T4:0/EN  | <b>EN - timer enable</b> | rung state is true  |

### 2. TOF- Timer, Off Delay



| Bit              |                          | Is Set When:  |
|------------------|--------------------------|---|
| bit 13 - T4:0/DN | <b>DN - timer done</b>   | rung conditions are true  |
| bit 14 - T4:0/TT | <b>TT - timer timing</b> | rung conditions are false and accumulated value is less than the preset value |
| bit15 - T4:0/EN  | <b>EN - timer enable</b> | rung conditions are true  |

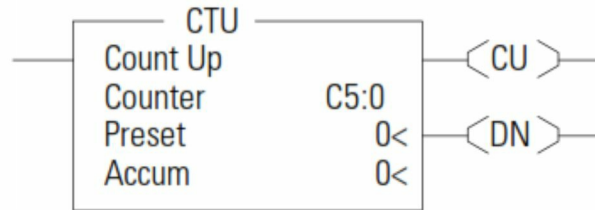
### 3. RTO- Retentive Timer, On Delay



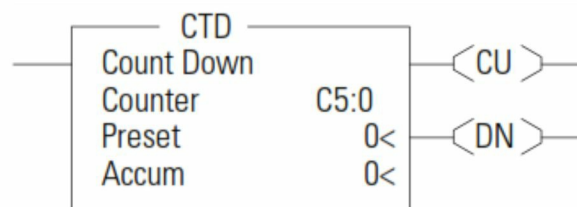
| <b>Bit</b>              |                          | <b>Is Set When:</b>                                     |
|-------------------------|--------------------------|---|
| <b>bit 13 - T4:0/DN</b> | <b>DN - timer done</b>   | accumulated value $\geq$ preset value                   |
| <b>bit 14 - T4:0/TT</b> | <b>TT - timer timing</b> | rung state is true and accumulated value < preset value |
| <b>bit15 - T4:0/EN</b>  | <b>EN - timer enable</b> | rung state is true                                      |

## 2.9 Use of counter

### 1. CTU- Count Up



### 2. CTD- Count down



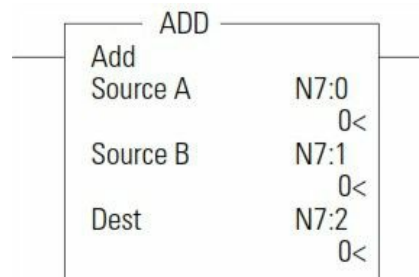
- The CTU and CTD instructions are used to increment or decrement a counter at each false-to-true rung transition.
- When the CTU rung makes a false-to-true transition, the accumulated value is incremented by one count.
- The CTD instruction operates the same, except the count is decremented.

| Bit              |                                | Is Set When:  |
|------------------|--------------------------------|---|
| bit 12 - C5:0/OV | <b>OV - overflow indicator</b> | the accumulated value wraps from +32,767 to -32,768 and continues to count up |
| bit 13 - C5:0/DN | <b>DN - done indicator</b>     | accumulated value $\geq$ preset value   |
| bit 15 - C5:0/CU | <b>CU - count up enable</b>    | rung state is true  |

## 2.10 Math instructions

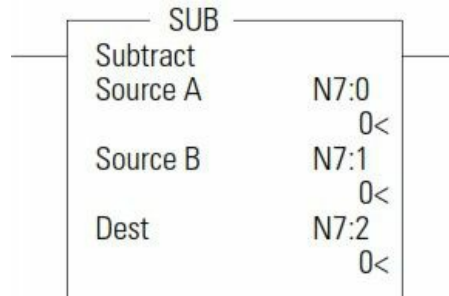
| Instruction                | Used to   |
|----------------------------|---|
| ADD- Add                   | Add two values  |
| SUB- Subtract              | Subtract two values   |
| MUL-Multiply               | Multiply two values   |
| DIV- Divide                | Divide one value by another   |
| NEG-Negate                 | Change the sign of the source value and place it in the destination   |
| CLR-Clear                  | Set all bits of words to zero   |
| ABS- Absolute value        | Find the absolute value of the source value                           |
| SQR- Square root           | Find the square root of value   |
| SCL- Scale                 | Scale a value   |
| SCP- Scale with Parameters | Scale a value to a range determined by creating a linear relationship |

### 1. Add instruction



Use the ADD instruction to add one value to another value (Source A + Source B) and place the sum in the destination.

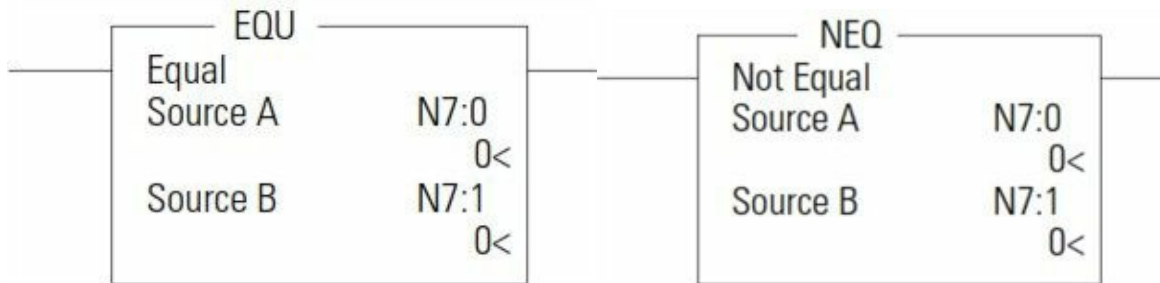
### 2. Subtraction instruction



Use the SUB instruction to subtract one value from another value (Source A - Source B) and place the sum in the destination.

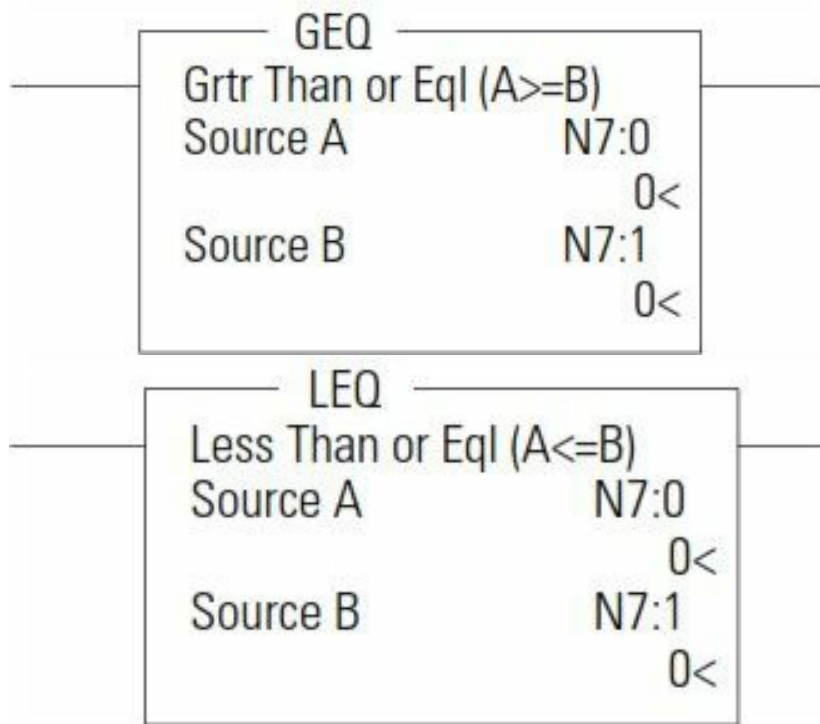
## 2.11 Compare instructions

### 1. Equal and not equal to instruction



| Instruction | Relationship of Source Values | Resulting Rung State |
|-------------|-------------------------------|----------------------|
| EQU         | A = B                         | true                 |
|             | A ≠ B                         | false                |
| NEQ         | A = B                         | false                |
|             | A ≠ B                         | true                 |

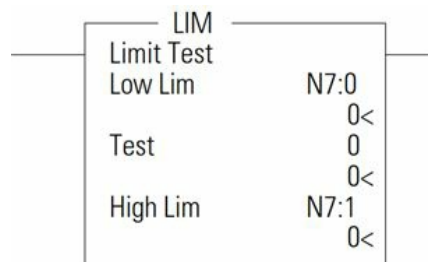
### 2. Greater than or equal to and Less than or equal instruction



**GEQ and LEQ Instruction Operation**

| Instruction | Relationship of Source Values | Resulting Rung State |
|-------------|-------------------------------|----------------------|
| GEQ         | $A \geq B$                    | true                 |
|             | $A < B$                       | false                |
| LEQ         | $A > B$                       | false                |
|             | $A \leq B$                    | true                 |

**3. Limit instruction**

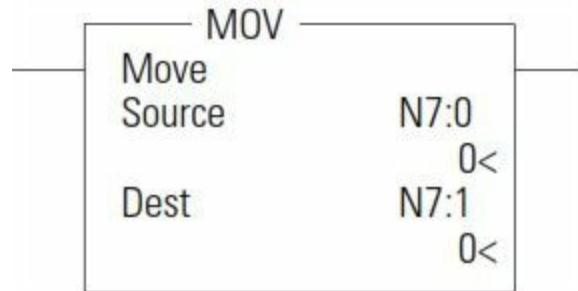


**LIM Instruction Operation Based on Low Limit, Test, and High Limit Values**

| When:                       | And:  | Rung State |
|-----------------------------|---|------------|
| Low Limit $\leq$ High Limit | Low Limit $\leq$ Test $\leq$ High Limit         | true       |
| Low Limit $\leq$ High Limit | Test $<$ Low Limit or Test $>$ High Limit       | false      |
| High Limit $<$ Low Limit    | High Limit $<$ Test $<$ Low Limit               | false      |
| High Limit $<$ Low Limit    | Test $\geq$ High Limit or Test $\leq$ Low Limit | true       |

## 2.12 Move and copy instructions

### 1. Move instruction



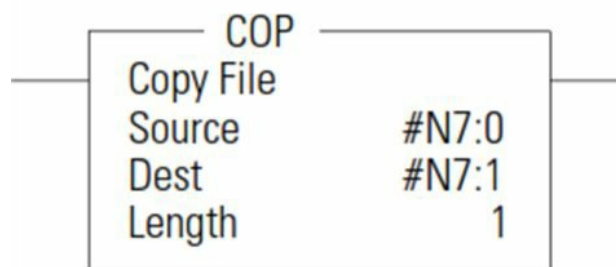
#### When using the MOV instruction, observe the following:

1. Source and Destination can be different data sizes.

The source is converted to the destination size when the instruction executes.

2. If the signed value of the Source does not fit in the Destination, the overflow is handled as follows:
  - If the Math Overflow Selection Bit is clear, a saturated result is stored in the Destination. If the Source is positive, the Destination is 32767 (word). If the result is negative, the Destination is -32768.
  - If the Math Overflow Selection Bit is set, the unsigned truncated value of the Source is stored in the Destination.
3. The source can be a constant or an address.

### 2. Copy instruction



The COP instruction copies blocks of data from one location into another. The source and destination file types must be the same except bit (B) and integer (N); they can be interchanged. It is the address that determines the maximum length of the block to be copied.



## 2.13 Analog signal programming

### Converting analog data

$$N = V_{in} \times 1023/10$$

Where,  $V_{in}$  (analog signal) is in volts (V)

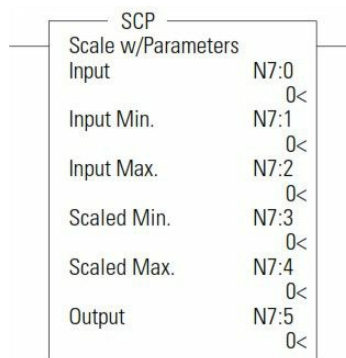
| Analog signal | Data word |
|---------------|-----------|
| 0V            | 0         |
| 5V            | 512       |
| 10V           | 1023      |

### Scale with parameter

The SCP instruction produces a scaled output value that has a linear relationship between the input and scaled values.

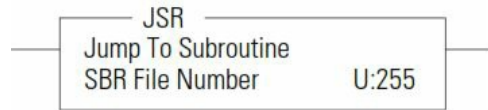
This instruction solves the following equation listed below to determine scaled output.

$$y = [(y1-y0) / (x1-x0)] (x-x0) + y0$$



## 2.14 Other instructions

### 1. JSR instruction



- The JSR instruction causes the controller to start executing a separate subroutine file within a ladder program. JSR moves program execution to the designated subroutine (SBR file number). After executing the SBR, control proceeds to the instruction following the JSR instruction.

### 2. Return instruction



- The RET instruction marks the end of subroutine execution or the end of the subroutine file.
- It causes the controller to resume execution at the instruction following the JSR instruction, user interrupt, or user fault routine that caused this subroutine to execute.

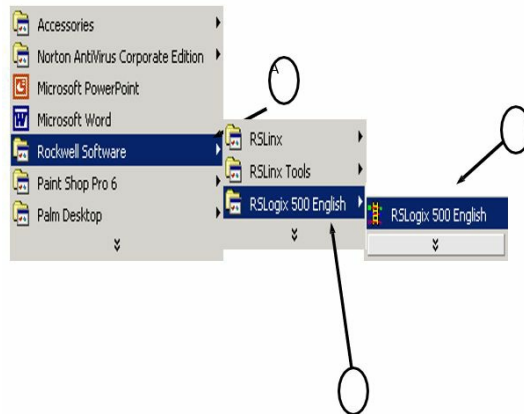
## 2.15 RS logix 500 PLC software

### 1. Getting started

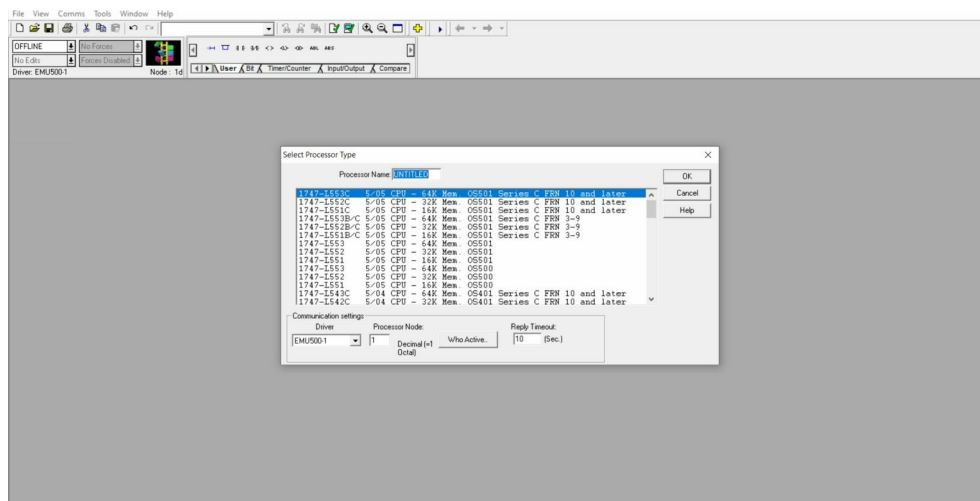
1. On your computer desktop locate the menu.

Start>Programs>Rockwell\_Software>RSLogix500 English>RSLogix 500 English.

1.Click to open the program.

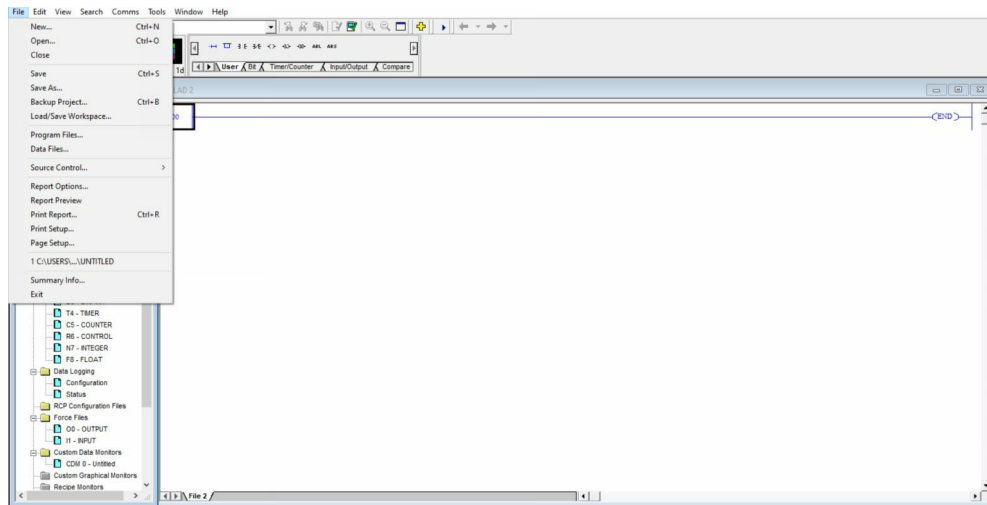


### 2. Creating a project



- Select the series of PLC

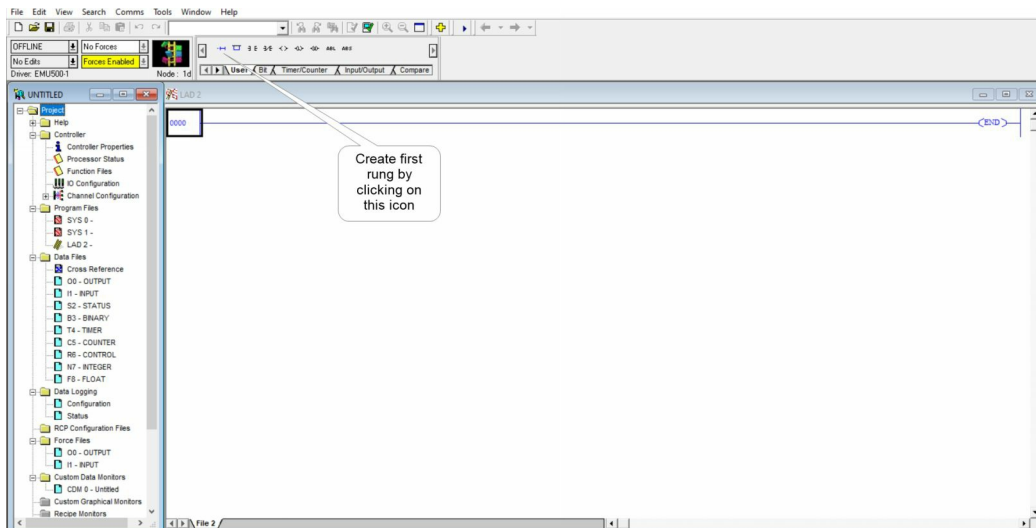
### 3. Save Project



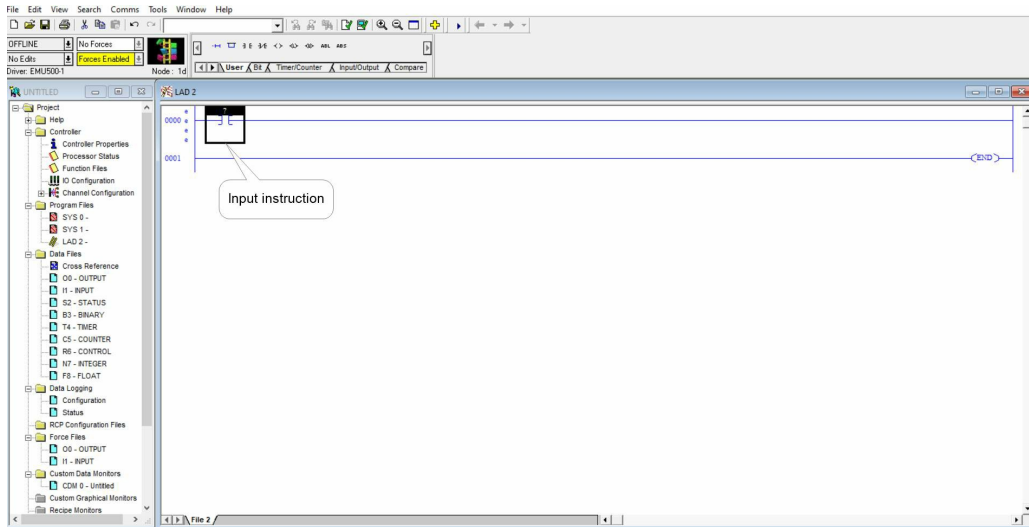
- Click “File” Menu
- Select “Save As”
- Type “MicroEconomix LAB1” in the “File name” box
- Click “Save”

#### 4. Creating the 1st Rung of Ladder Logic

Program this 1st rung:



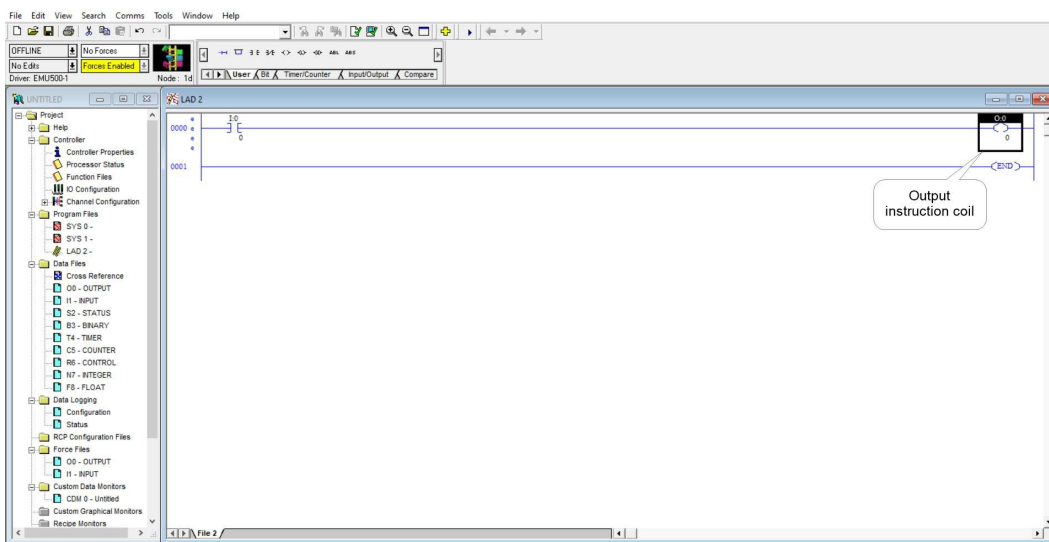
a) Add an Input Instruction



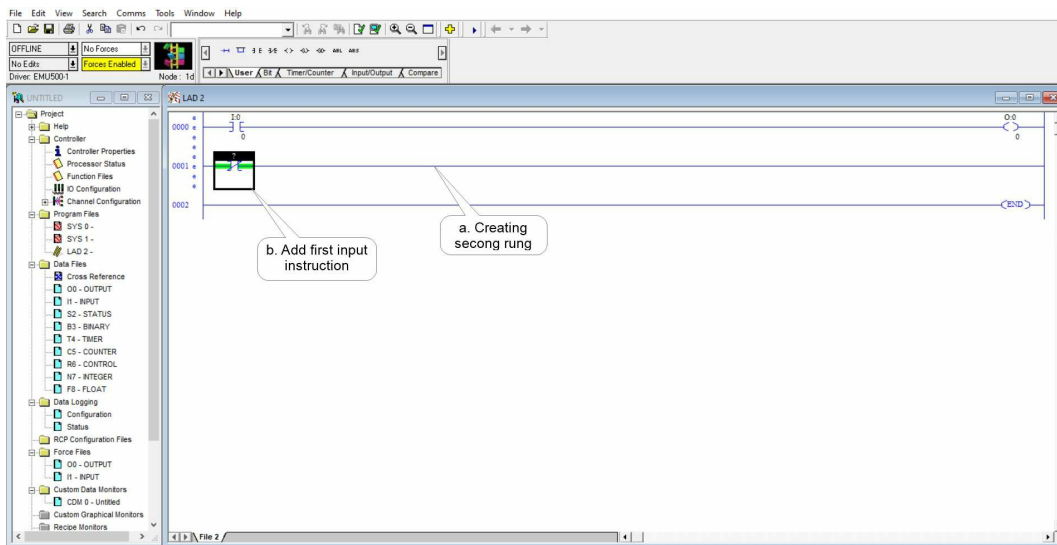
- Click, hold the left mouse button and drag the XIC(examine if closed) button onto the left side of the rung you just created. When you see a green box, release the mouse button.
- With the instruction highlighted Type I:0/0 [Enter]. This is the address of the XIC (examine if closed) instruction.

#### b) Add Output Instruction

- Click, hold the left mouse button and drag the OTE (output energized) button onto the right side of the rung you just created.
- When you see a green box, release the mouse button.
- With the instruction highlighted Type O:0/0 [Enter]. This is the address of the OTE instruction.



## 5. Creating the 2nd Rung of Logic



### a) Create a New Rung

Click, hold the left mouse button and drag the “New Rung” button over rung “0001”.

When you see a green box, release the mouse button.

### b) Add the 1st Input Instruction

Click, hold the left mouse button and drag the XIO (examine if open) button onto the left side of the rung you just created. When you see a green box, release the mouse button.

## 6. Verify your work

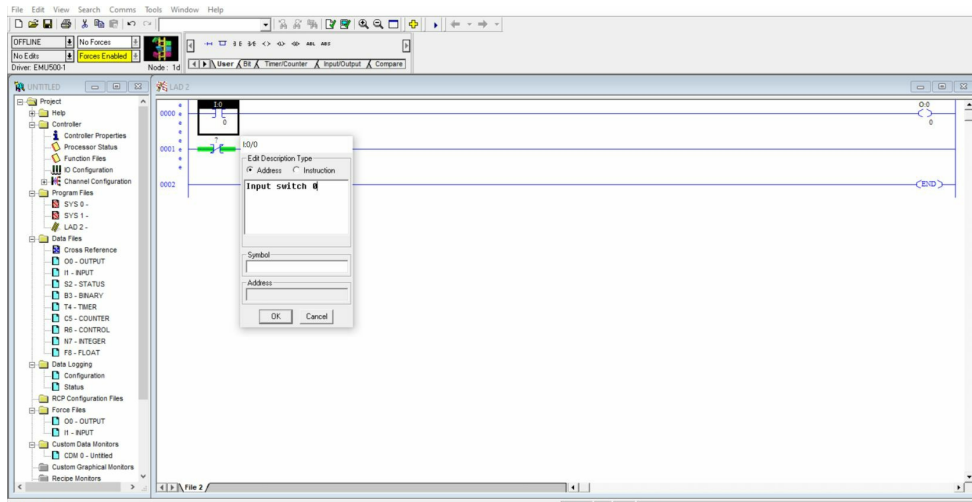
If Program has errors

- To find the errors in the program click on the error message in the “Verify results window” the error is then highlighted in the ladder window.
- Fix the error and run “Verify file” again.
- When all the errors are fixed you can then save and download the program.

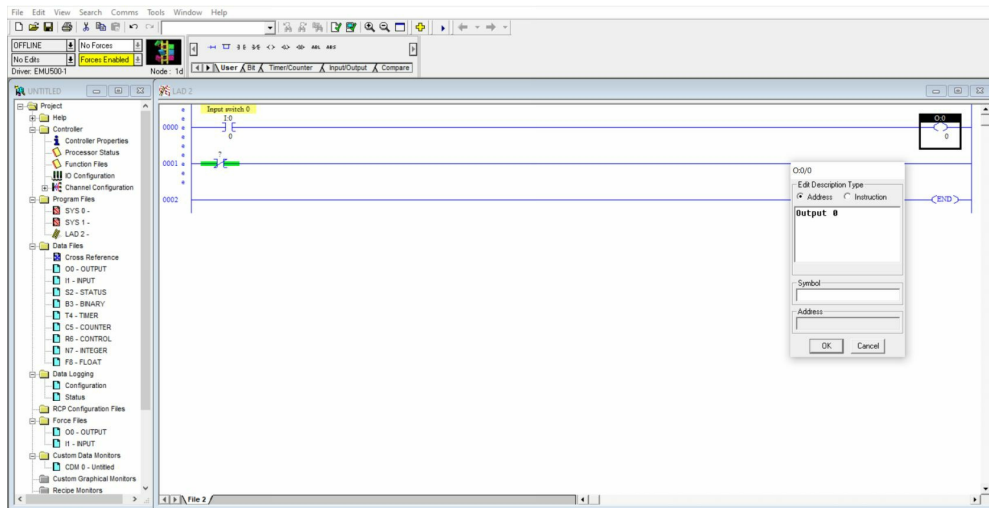
## 7. Documenting your Program

- Click on the Input I:0/0 to highlight
- Right mouse on I:0/0 and select “Edit Description- I:0/0”

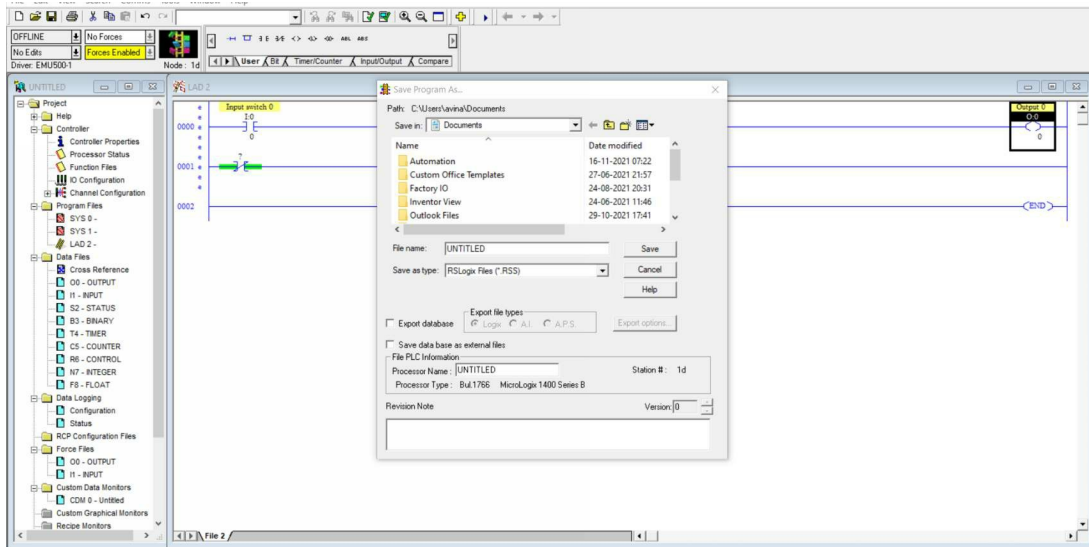
- Select “Address”
- Type “Input Switch 0” in the Edit window
- Select “OK”



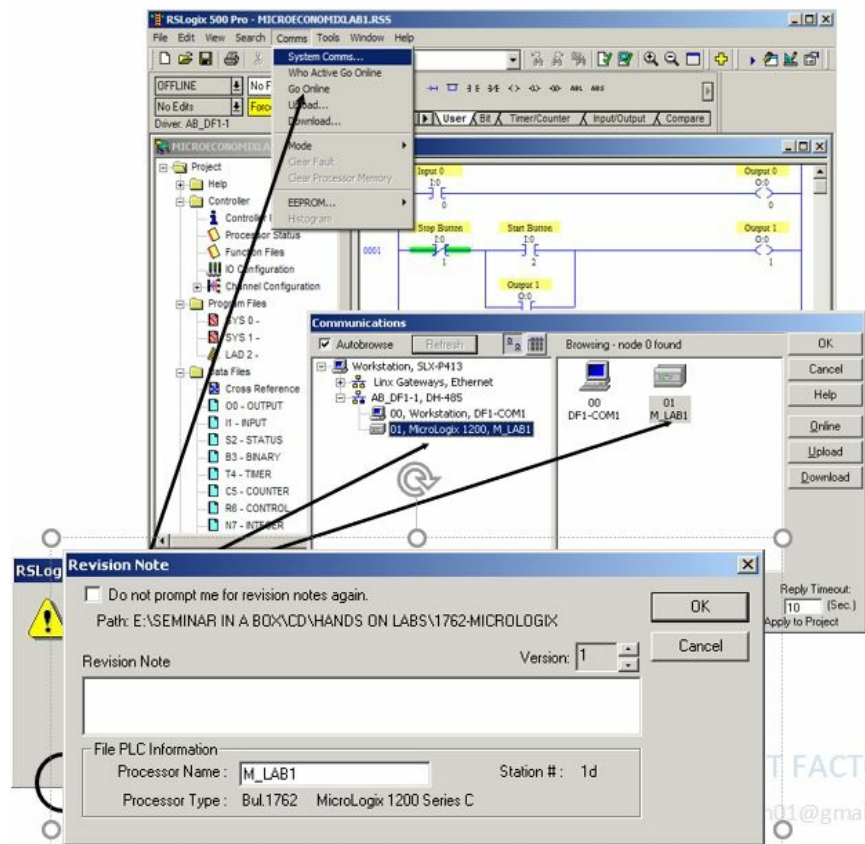
- Click on the Output O:0/0 to highlight
- Right mouse on O:0/0 and select “Edit Description- O:0/0”
- Select “Address”
- Type “Output 0” in the Edit window
- Select “OK”



## 8. Save your work



## 9. Download the Program

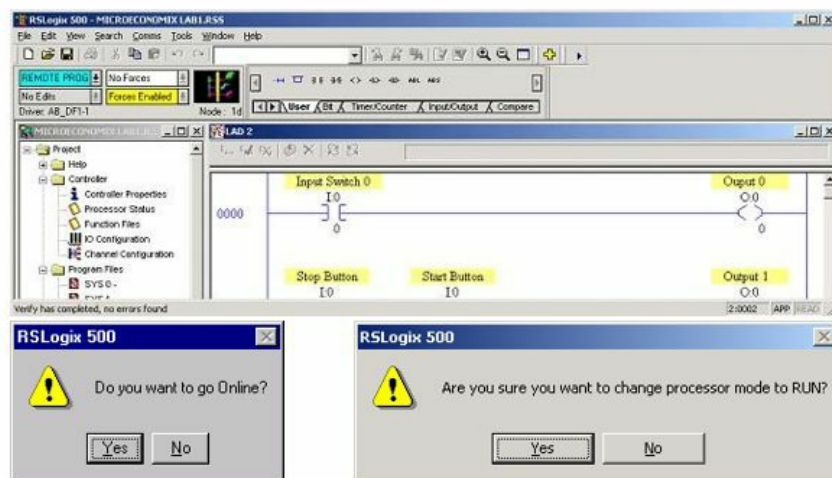


- Select the menu item “Comms > System Comms”
- Three primary selections
- “Online” Establish the “path”
- “Upload” Receive from the controller



- “Download” Send to the controller
- Highlight the device at Node 01.
- Select “Download
- Select “OK” in the Revision note window. The revision note box is used to keep track of changes made to the existing program. You can create many revisions of the same program. This feature can be disabled if desired.
- Select “Yes” to download your program over the existing program that resides in the processor. This window will appear whenever a program is being downloaded to the processor.
- Download verification window will appear when the download takes place.

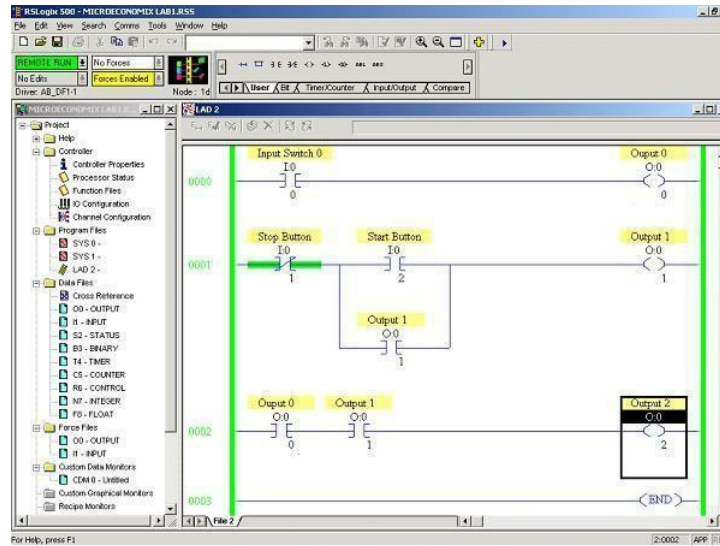
## 10. Put the Micro into the RUN mode



Select “Yes” to go online. This will allow you to monitor the program that now resides in the processor.

- Click on the down arrow next to the word REMOTE PROG
- Three Run selections
- One scan cycle with outputs disabled
- Select “Run”
- Select “Yes” for “Are you sure window”

## 11. Monitor the Controller



- With the controller in “Remote Run”, you can monitor or edit data within the controller.
- This allows:
- Program debugging
- Change data variables while in run When “Green” bars are shown on either side of logic elements, this indicates “Logical Continuity”, this helps to determine how the application is operating.
- This design is to help in debugging an application's logic.

## 2.16 PLC programming examples using rslogix 500 software

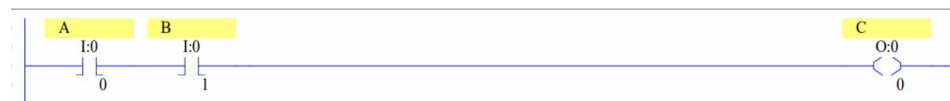
### 1. Logic gates

#### 1. AND Gate



**Truth table**

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |



#### 2. OR Gate

**Truth table**

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |



### 3. NOT Gate

#### Truth table

| A | C |
|---|---|
| 0 | 1 |
| 1 | 0 |



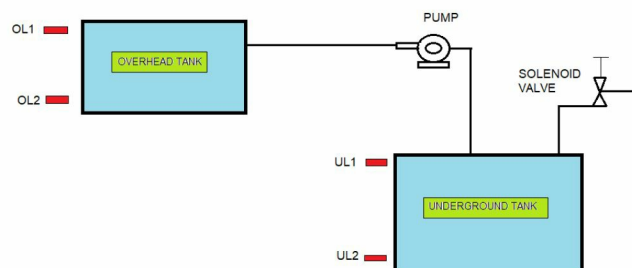
### 4. NOR GATE

#### Truth table

| A | B | C |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |



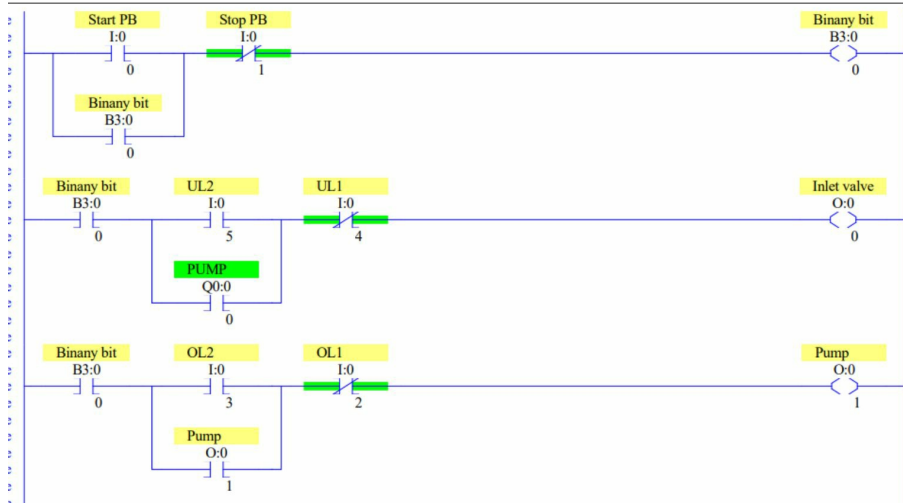
## 2. Water level control



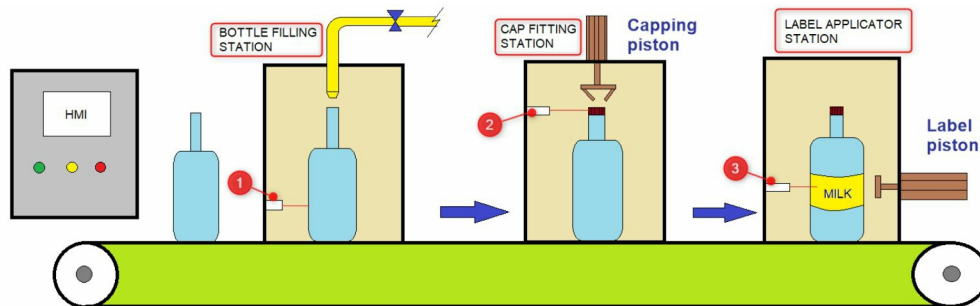
- The Inlet valve opens when the water level goes to the lower level of the underground tank and gets closed when the underground tank gets full.
- The pump starts only when the water level in the overhead tank goes to the lower level of the overhead tank.
- The pump stops when the overhead tank gets full or the water level in the underground tank goes to the lower level. I.e. The pump should not dry run.

### Input and Output devices list

| Input Devices                 | Output devices |
|-------------------------------|----------------|
| Start PB                      | Pump           |
| Stop PB                       | Solenoid valve |
| OL1- Overhead sensor1         |                |
| OL2- Overhead sensor2         |                |
| UL1- Underground tank sensor1 |                |
| UL2- Underground tank sensor2 |                |



### 3. Milk bottle filling and capping application

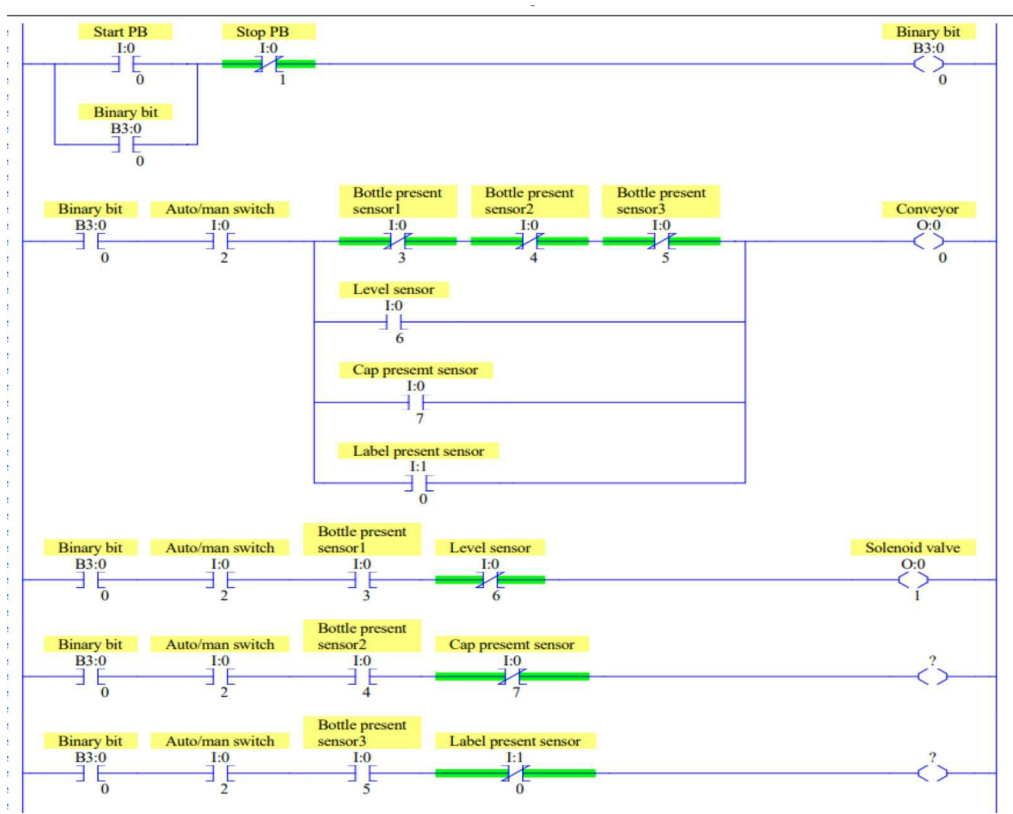


- Start the conveyor in auto mode. When a bottle present sensor 1 detects the bottle, the conveyor gets stopped and the controller turns on the valve. The milk starts pouring into the bottle up to a certain level, then the valve gets off and the conveyor starts again.
- When a bottle present sensor 2 detects the bottle, the conveyor gets stopped, and the capping piston operates, when a cap is detected by a sensor, the capping piston stops and the conveyor starts again.
- When a bottle present sensor 3 detects the bottle, the conveyor gets stopped, and the labeling piston operates, when a label is detected by a sensor, the labeling piston stops and the conveyor starts again.
- This cycle continues until a stop PB is pressed.

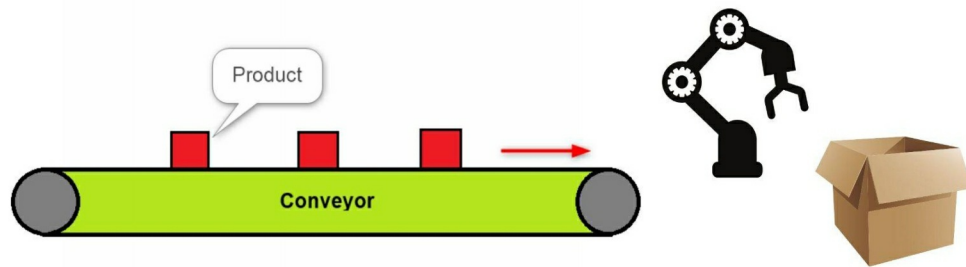
#### Input and Output devices list

| Input devices | Output devices |
|---------------|----------------|
|               |                |

|                         |                |
|-------------------------|----------------|
| Start PB                | Conveyor motor |
| Stop PB                 | Solenoid valve |
| Auto/Man switch         | Capping piston |
| Bottle present sensor 1 | Label piston   |
| Level sensor            | SPARE          |
| Bottle present sensor 2 | SPARE          |
| Cap present sensor      | SPARE          |
| Bottle present sensor 3 | SPARE          |
| Label present sensor    | SPARE          |



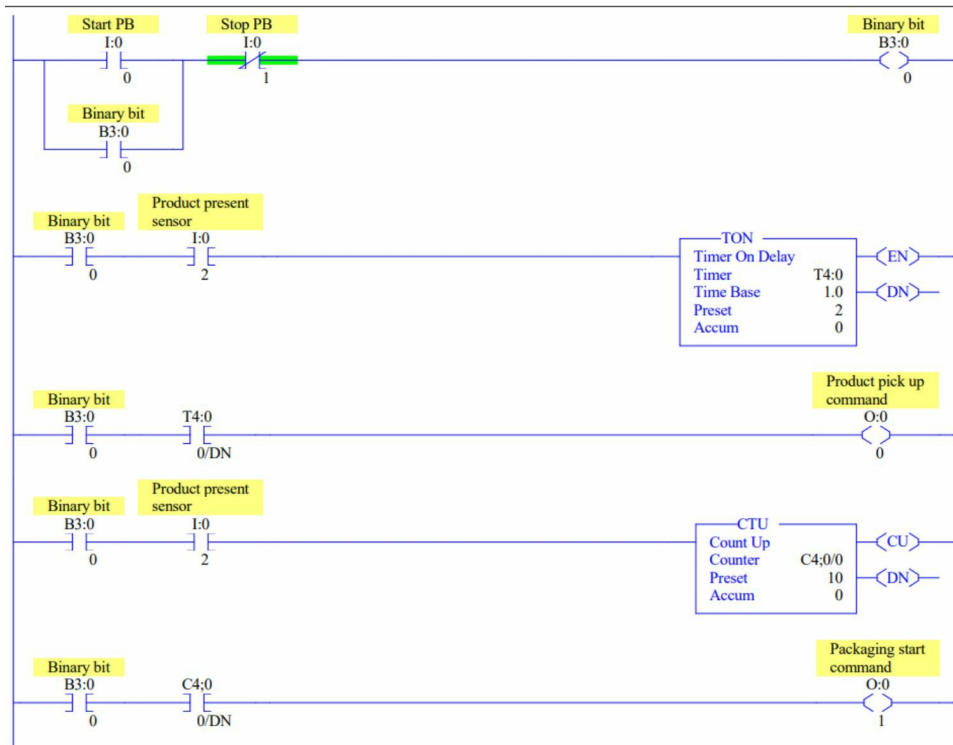
## 4. Packaging application



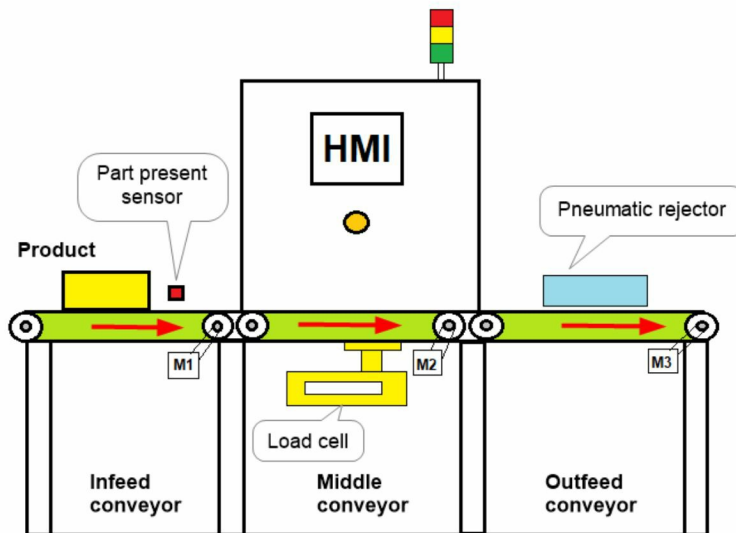
- Start PB is used to start the whole process.
- A photo Sensor is used to detect products. When a product approaches a sensor, the controller sends a signal to the robotic arm after 2 sec to pick up the product.
- Once the product count reaches 10, the controller sends the signal to the packaging unit.

| Input devices          | Output devices          |
|------------------------|-------------------------|
| Start PB               | Product pick up command |
| Stop PB                | Start packaging command |
| Product present sensor |                         |



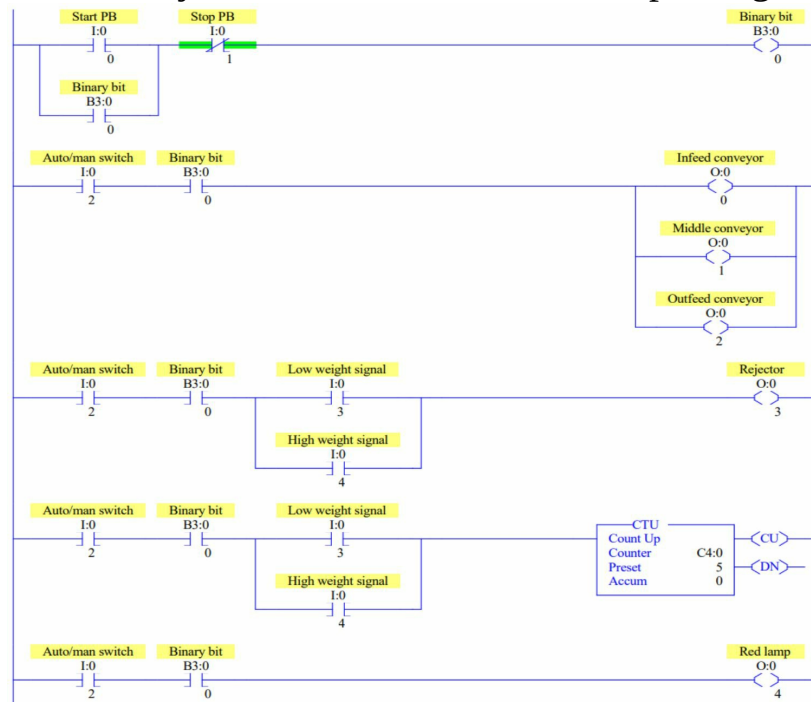


## 5. Weighing application

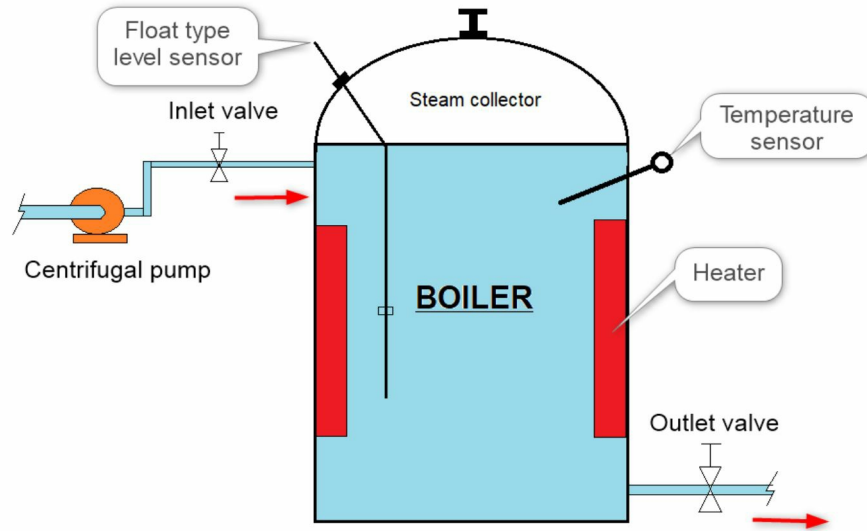


- A checkweigher consists of an infeed conveyor, middle conveyor with a load cell, and an output conveyor with a pneumatic rejector. A product present sensor is mounted on the infeed conveyor that detects the product.

- When a product passes over the middle conveyor, the load cell records the weight of the product during its travelling and sends signals to the PLC. The PLC receives three types of signals from the load cell (High weight, low weight, and accurate weight).
- If the product's weight is OK, the product will pass from the checkweigher.
- But if the product's weight is low or high, a pneumatic rejector will reject the product from the outfeed conveyor.
- If products are rejected 5 times, then a red lamp will glow.



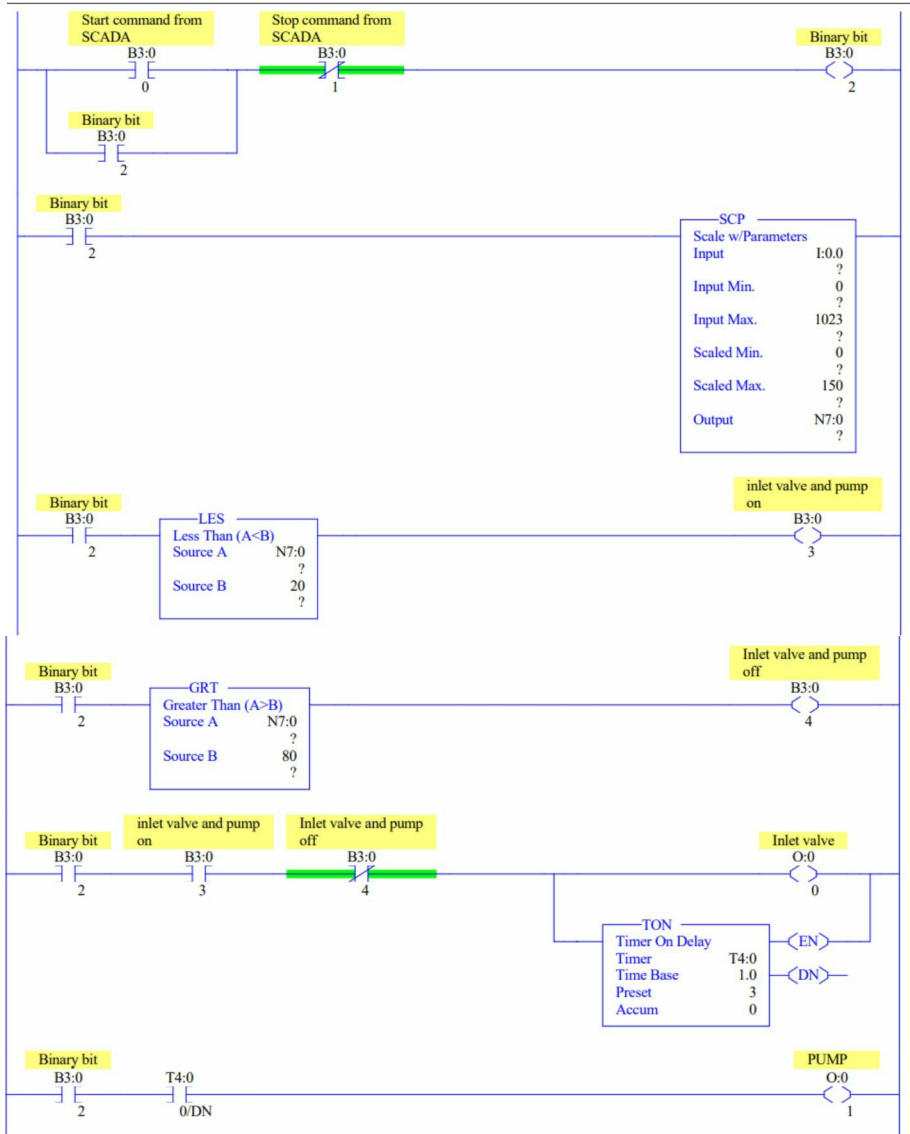
## 6. Boiler application

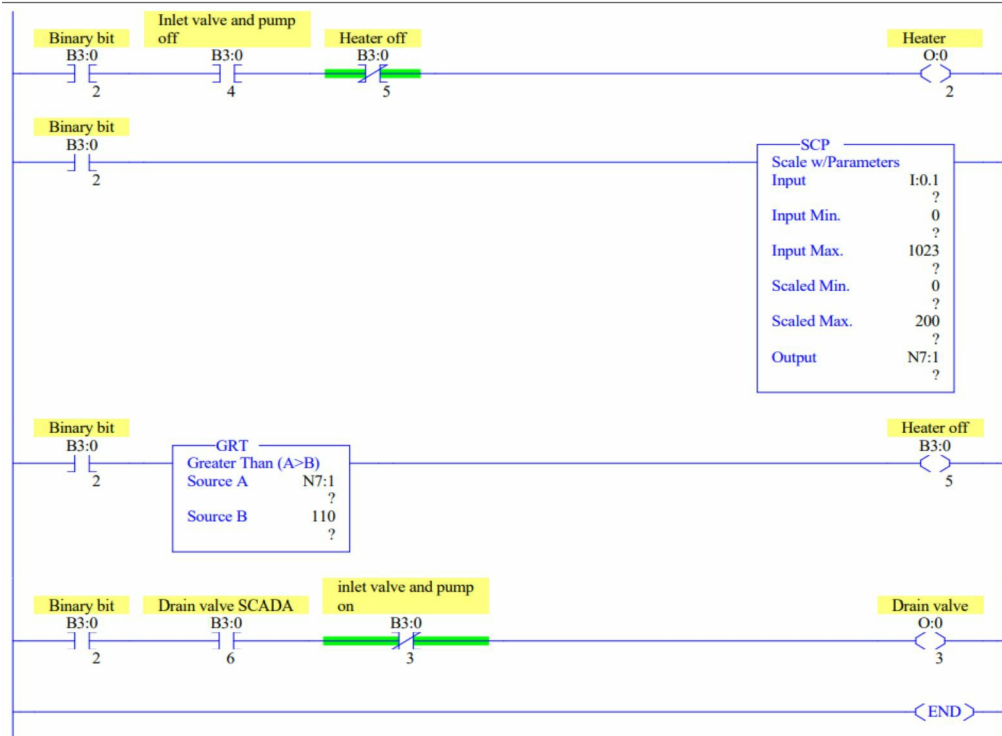


- The boiler process starts when a start button is pressed from SCADA. If the level of water in the boiler is less than 20 ltr, the inlet valve will get on first and the centrifugal pump will get on after 3 sec. And they remain on until the water level reaches 80 ltr.
- Once the water level reaches 80 ltr, the pump and inlet valve will stop. The heater will start heating the water inside the boiler.
- As soon as the temperature reaches 110 degrees celsius, the heater will turn off. The outlet valve will be opened from SCADA to drain the boiled water.
- The outlet valve remains open until the water level reaches 20 ltr.
- This cycle continues until a stop button is pressed.

### Input and Output devices list

| Input devices      | Output devices   |
|--------------------|------------------|
| Temperature sensor | Inlet valve      |
| Level sensor       | Outlet valve     |
|                    | Centrifugal pump |





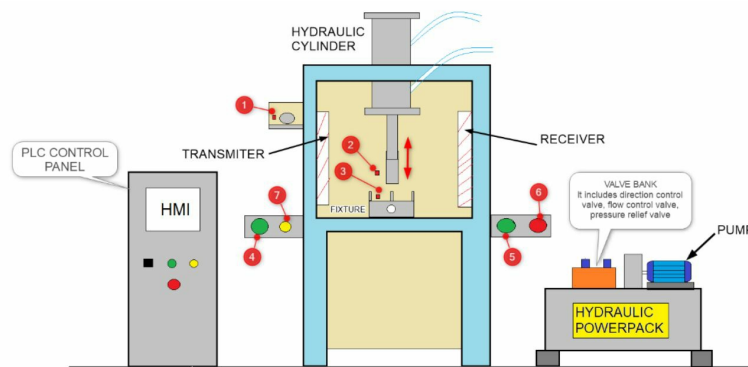
# Case study 1-Hydraulic press machine

## Scope of work



As shown in the above image, we have to press two LM bearings into the fork. Of Course we can go for a hammer or any mechanism, but customer needs these forks in huge numbers. So we provide him with a hydraulic press machine which will work effectively and save time as well as effort.

## Hydraulic press machine



## Input and Output devices list

| Input devices               | Output devices              |
|-----------------------------|-----------------------------|
| 1. POKA YOKE sensor         | Pump on                     |
| 2. Bearing present sensor   | Hydraulic cyl UP sol coil   |
| 3. Part present sensor      | Hydraulic cyl DOWN sol coil |
| 4. Palm PB1                 | Operation complete lamp     |
| 5. Palm PB2                 | SPARE                       |
| 6. Emergency PB             | SPARE                       |
| 7. Auto/man selector switch | SPARE                       |
| Safety light curtain        | SPARE                       |

### Hardware and software details

PLC - AB micrologix 1400

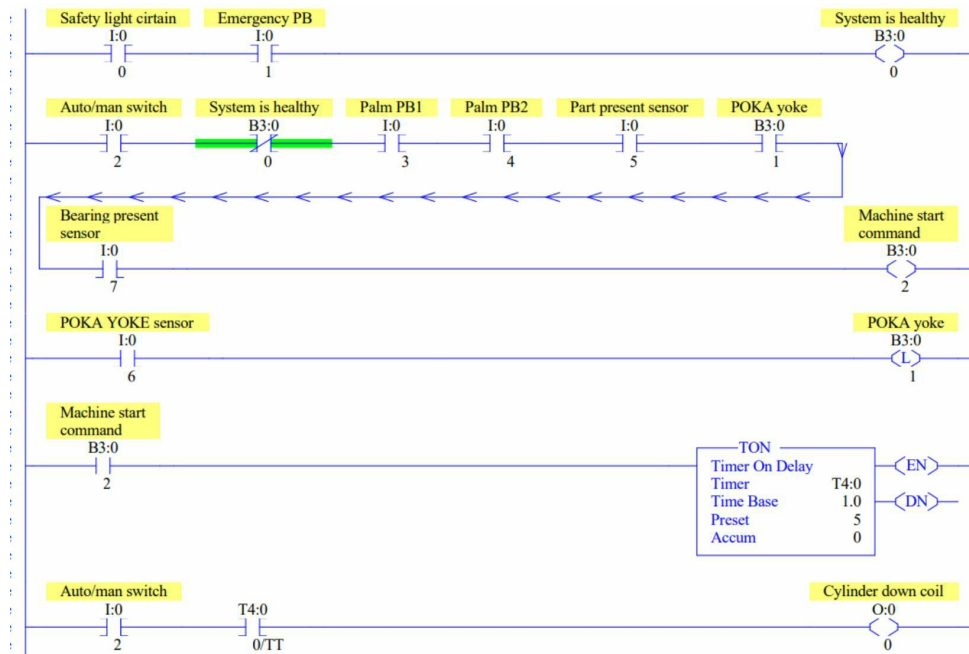
HMI- Panelview

PLC & HMI communication- Ethernet

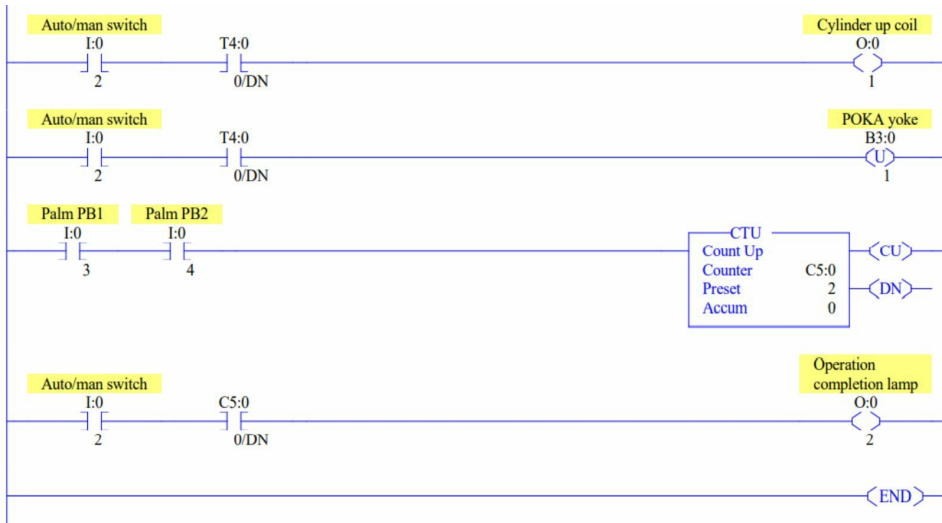
Pump Motor control- Hardwired (via contactor)

## Operation

- Keep the fork on the fixture properly, (part present sensor detects the fork).
- Take a LM bearing, sense it to the POKA YOKE sensor and then attach it below the hydraulic ram, (bearing present sensor detects the bearing).
- When the machine is in auto mode and there is no fault, a machine should start after pressing both palm PBs simultaneously. Switch on the cylinder DOWN solenoid coil for 5 sec, after completion of 5 sec, switch on the cylinder UP solenoid coil.
- Take another LM bearing, sense it to the POKA YOKE sensor, attach it to the ram, and press both palm PBs simultaneously. A lamp must glow after the completion of the cycle.
- If someone tries to interrupt machine operation (light curtain gives signal) a machine must stop immediately.
- A POKA YOKE sensor operation cannot be bypassed.







## b. SIEMENS PLC

The Siemens logo, consisting of the word "SIEMENS" in a bold, blue, sans-serif font, is centered within a light blue rectangular box.

## 3.1 Siemens PLC series

### 1. Logo

- Perform small-scale automation tasks more quickly and free up space in your control cabinet: The LOGO! controllers from Siemens offer you the support you need at a price you can afford. With 8 basic logic functions and 30/35 special functions, LOGO! can replace a large number of conventional switching and control devices.

### 2. Logo 8

- With LOGO 8, the successful logic module from Siemens starts the next generation. The new module meets almost all customer requests with easier handling, a new display and full communication options over Ethernet. It also makes the Web server application extremely easy to use. Plus remote communication through wireless networks or a communication module rounds off the range of new opportunities associated with LOGO.

### 3. S7-200



- For small applications s7-200 can be used.
- This Controller are no more available for sale

### 4. S7-1200



- The SIMATIC S7-1200 is the controller for control tasks in machine building and plant construction.
- The small controllers offer perfect interaction with the Basic Panels and are programmed with STEP 7 Basic in the TIA Portal.

## 5. S7-300



- The S7-300 is the individual solution for fast process and automation tasks that contain additional data processing tasks.
- It is high-performance, fast, versatile, and future-proof. For engineering either STEP 7 V5.5/STEP, 7 Professional 2010 or STEP 7 Professional in the TIA Portal can be used.

## 6. S7 400



- The SIMATIC S7-400, which is fast, robust, and strong in

communication, has been proven as a controller in the upper-performance range of factory automation and can be found in plants for process automation as well.

- Similar to the S7-300, for engineering, the established STEP 7, STEP 7 Professional in the TIA Portal or in the mighty, process-oriented PCS7 can be used.

## 7. S7-1500



- The new SIMATIC S7-1500 controller sets new standards in productivity with its many innovations. SIMATIC S7-1500 is perfectly integrated with the TIA Portal for maximum efficiency in engineering.
- The ultimate plus in automation is furthermore convincing through its system performance, its integrated technology, its security concept, easiest handling, and maximum usability and last but not least through its integrated system diagnostics with consistent visualization concept in the CPU display and in the engineering, on the HMI-Panel and in the Web server.

## 8. S7-1500 Software Controller



- The S7-1500 Software Controller CPU 1507S implements the function of an S7-1500 controller as software on a SIMATIC IPC with Windows. This allows a SIMATIC IPC to be used for control of machinery or plants. The integrated Ethernet and PROFIBUS interfaces of the SIMATIC IPC can be used to connect distributed I/O via PROFINET or PROFIBUS.
- In addition, the CPU offers extensive control functions through easily

configurable modules, as well as the connection of drives via standardized PLC-open blocks. Its special strengths as a software controller come into play when specific automation functions are integrated via the programming language C or C++, or when a close connection between Windows software and the software controller is required.

## 9. Simatic ET 200SP



- The SIMATIC ET 200SP distributed I/O system is a scalable and highly flexible, distributed I/O system in the degree of protection IP20 for linking of process signals to a central controller via PROFINET.

## 10. Simatic ET200S



- The ET 200S is a modular, distributed I/O device in the degree of protection IP 20 with integrated SIGUARD safety technology. In addition to input/output modules, the ET 200S integrates technology modules and load feeders.

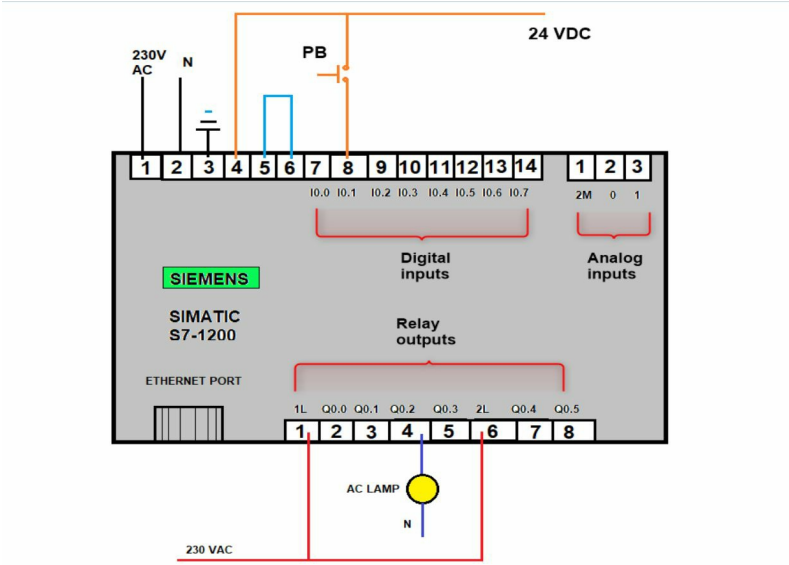
## 11. Simatic ET200Pro



- SIMATIC ET 200pro is a modular, rugged and versatile I/O system with IP65/66/67 degree of protection. It comprises Interface Modules to connect with PROFINET or PROFIBUS environments with standard as well as failsafe functionality. Interface modules are also available with

CPU functionality. Due to its rugged design ET 200pro can be used when mechanical load is high. Hot swapping of electronics and terminal modules during operation increases plant availability.

# 3.2 Siemens S7- 1200 PLC wiring configuration





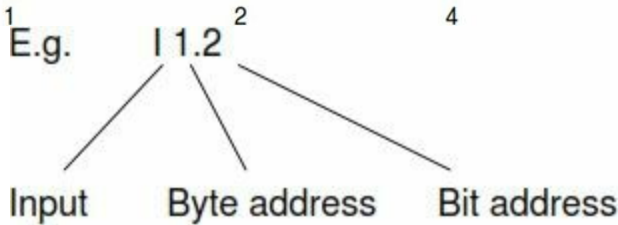
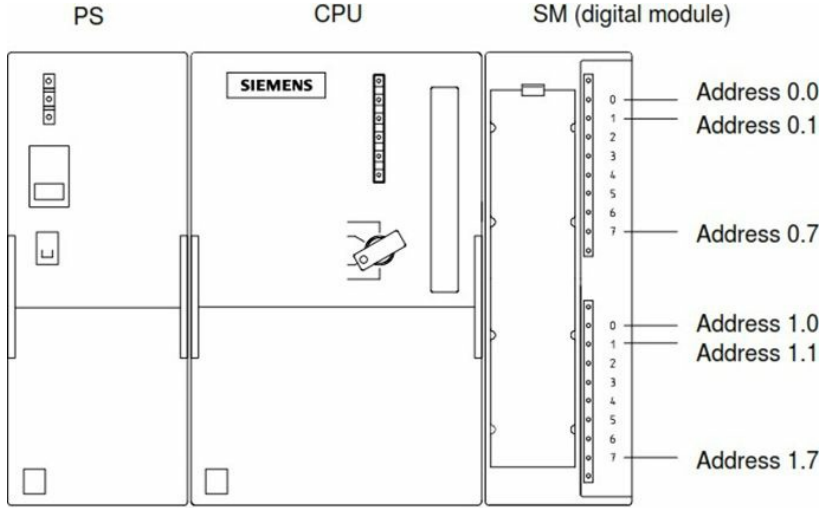
### 3.3 PLC and their programming software

| <b>PLC</b> | <b>PROGRAMMING SOFTWARE</b>                         |
|------------|---|
| Logo       | Logosoft  |
| S7-200     | MicroWin  |
| S7-300     | Simatic Manager Step7 V 5.4/5.5 / TIA V12 and Above |
| S7-400     | Simatic Manager Step7 V 5.4/5.5 / TIA V12 and Above |
| S7-1500    | TIA V12 and Above                                   |
| S7-1200    | TIA V 10.5 and Above                                |

## 3.4 Siemens communication protocols

1. **MPI communication**
2. **PROFIBUS communication**
3. **PPI communication**
4. **Communication by the ASCII protocol**
5. **Ethernet communication (protocol S7, PROFINET, IE-Industrial Ethernet)**

# 3.5 Addressing for ladder logic programming



## 3.6 Digital signal programming

A Digital signal is that signal which is in a discrete manner. A digital signal could be either 0Vdc or 24V dc supply.

### **Byte addressing**

Compact PLC- I124.0 TO I124.7, I125.0 to I125.7, and so on

Modular- I0.0 to I0.7, I1.0 to I1.7, and so on

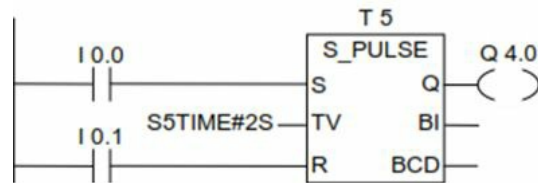
## 3.7 Bit logic

Instead of using the latching circuit, again and again, it is better to go for the set and set bits, these bits help to reduce the size of rungs.

## 3.8 Timer

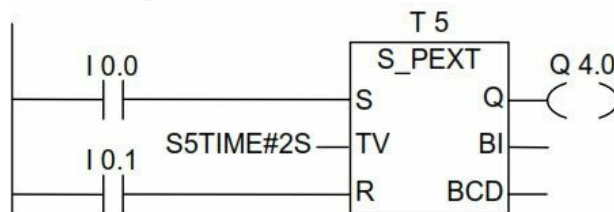
### Types of timer

#### 1. S-PULSE-Pulse S5 timer

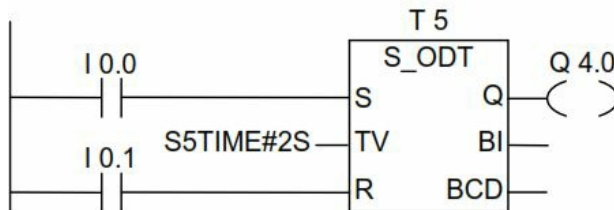


- If the signal state in input changes from “0” to “1”. The timer will start. The timer will continue to run for 2 sec as long as I0.0 is “1”. The timer is reset by I0.1 input.
- Q0.0 is “1” as long as the timer is running.

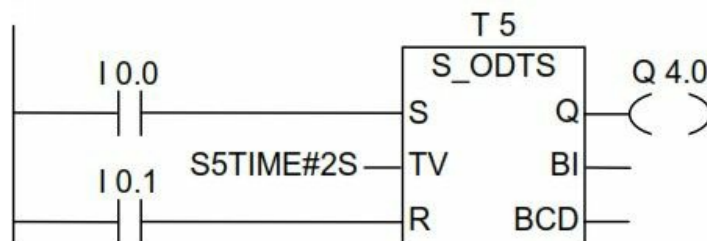
#### 2. S-PEXT-Extended pulse S5 Timer



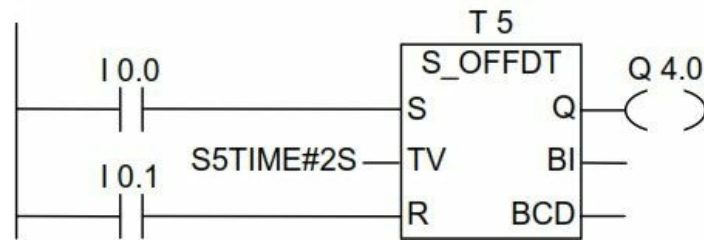
#### 3. ON Delay Timer- On Delay S5 Timer



#### 4. S\_ODTS-Retentive On Delay S5 Timer



## 5. S-OFFDT- Off Delay S5 Timer

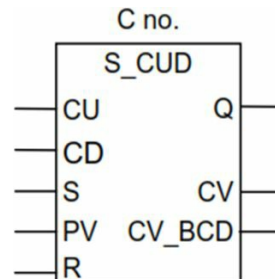


- If the signal state of I0.0 changes from “1” to “0”, the timer is started.
- Q4.0 is “1” when I0.0 is “1” or the timer is running (If the signal state at I0.0 changes from “0” to “1” while the time is running, the timer is reset).

## 3.9 Counter

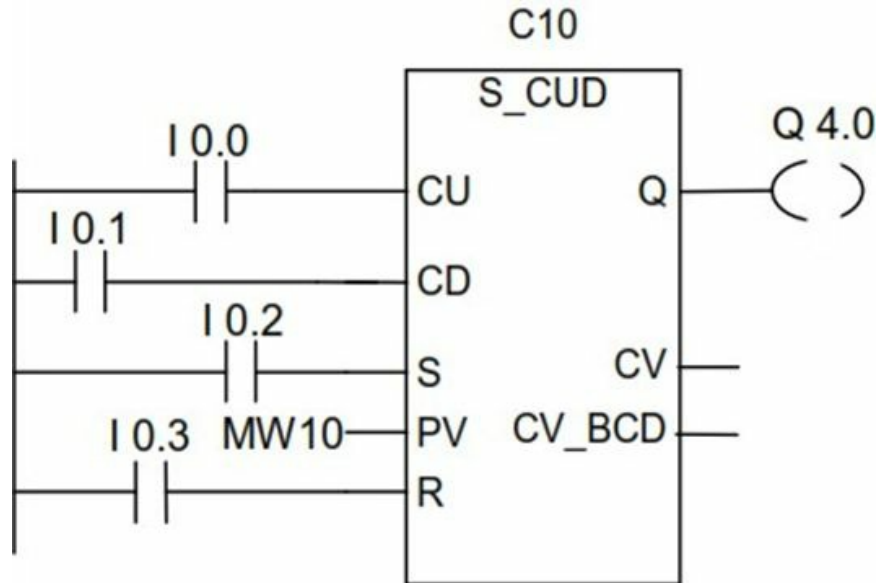
### Types of counter

#### 1. S\_CUD- Up Down Counter

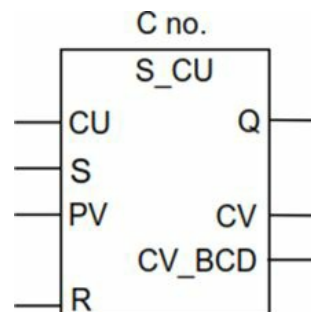


- S\_CUD is preset with the value at input PV if there is a positive edge at input S. If there is a 1 at input R, the counter is reset and the count is set to zero. The counter is incremented by one if the signal state at input CU changes from “0” to “1” and the value of the counter is less than “999”. The counter is decremented by one if there is a positive edge at input CD and the value of the counter is greater than “0”.
- If there is a positive edge at both count inputs, both instructions are executed and the count value remains unchanged.
- If the counter is set and if RLO=1 at inputs CU/CD, the counter is counted once in the next scan cycle, even if there was no change from a positive to a negative edge or vice versa.
- The signal state at output Q is “1” if the counter is greater than zero and ‘0” if the count is equal to zero.

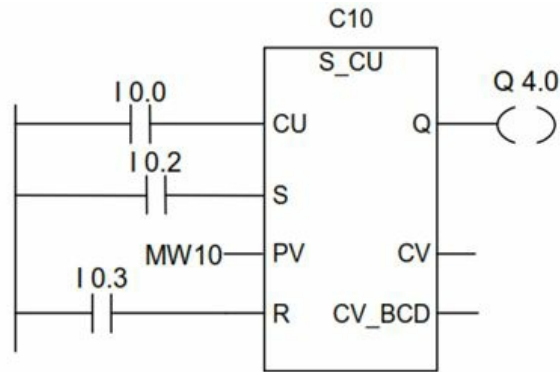




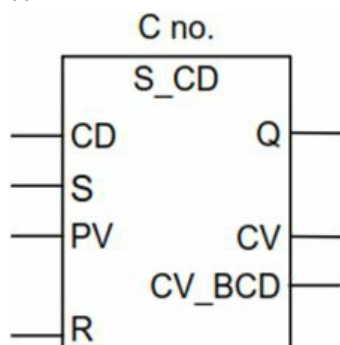
## 2. S\_CU- Counter Up



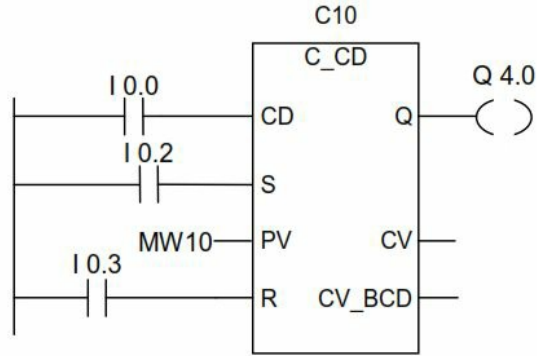
- S\_CU (Up counter) is preset with the value at input PV if there is a positive edge at input S.
- The counter is reset if there is “1” at input R and the count value is then set to zero.
- The counter is incremented by one if the signal state at input CU changes from “0” to “1” and the value of the counter is less than “999”.
- If the counter is set and if RLO=1 at the inputs CU, the counter will count once in the next scan cycle, even if there was no change from the positive to a negative edge or vice versa.
- The signal state at output Q is “1” if the counter is greater than zero and ‘0” if the count is equal to zero.



### 3. S\_CD- Counter Down



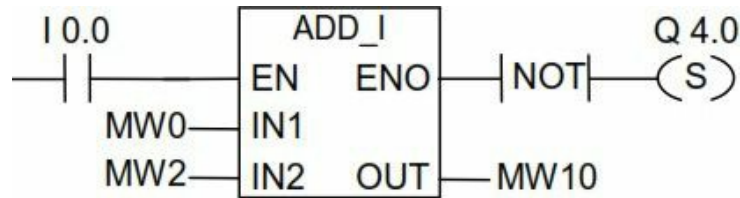
- S\_CD (Down counter) is preset with the value at input PV if there is a positive edge at input S.
- The counter is reset if there is “1” at input R and the count value is then set to zero.
- The counter is decremented by one if the signal state at input CU changes from “0” to “1” and the value of the counter is less than “999”.
- If the counter is set and if RLO=1 at the inputs CU, the counter will count once in the next scan cycle, even if there was no change from the positive to a negative edge or vice versa.
- The signal state at output Q is “1” if the counter is greater than zero and ‘0” if the count is equal to zero.



| Parameter English | Parameter German | Data Type | Memory Area               | Description   |
|-------------------|------------------|-----------|---------------------------|---|
| C no.             | Z no.            | COUNTER   | C                         | Counter identification number; range depends on CPU         |
| CU                | ZV               | BOOL      | I, Q, M, L, D             | Count up input  |
| CD                | ZR               | BOOL      | I, Q, M, L, D             | Count down input  |
| S                 | S                | BOOL      | I, Q, M, L, D             | Set input for presetting counter                            |
| PV                | ZW               | WORD      | I, Q, M, L, D or constant | Enter counter value as C#<value> in the range from 0 to 999 |
| PV                | ZW               | WORD      | I, Q, M, L, D             | Value for presetting counter                                |
| R                 | R                | BOOL      | I, Q, M, L, D             | Reset input   |
| CV                | DUAL             | WORD      | I, Q, M, L, D             | Current counter value, hexadecimal number                   |
| CV_BCD            | DEZ              | WORD      | I, Q, M, L, D             | Current counter value, BCD coded                            |
| Q                 | Q                | BOOL      | I, Q, M, L, D             | Status of the counter                                       |

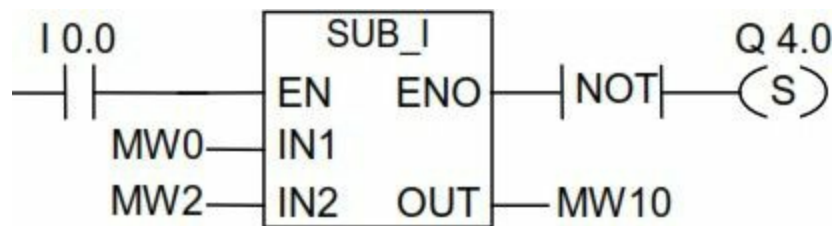
## 3.10 Math instructions

### 1. Add instruction



ADD\_I (Add integer) is activated by logic “1” at the Enabled (EN) input. IN1 and IN2 are added and the result can be scanned at OUT. If the result is outside the permissible range for an integer (16 bits), the OV bit and OS bit will be “1” and ENO is logic “0”, so that other functions after this math box which are connected by the ENO are not executed.

### 2. Subtract instruction



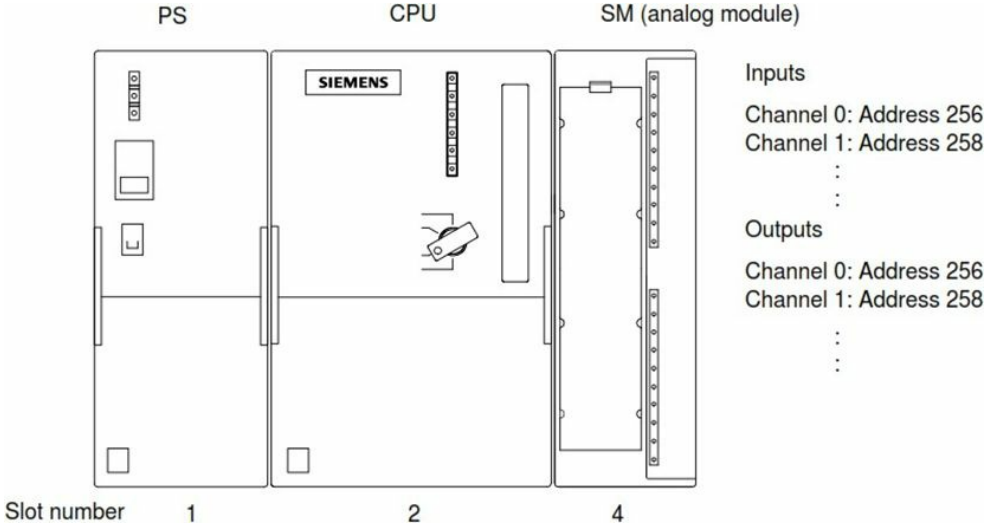
SUB\_I (Subtract integer) is activated by logic “1” at the Enabled (EN) input. IN2 is subtracted from IN1 and the result can be scanned at OUT. If the result is outside the permissible range for an integer (16 bits), the OV bit and OS bit will be “1” and ENO is logic “0”, so that other functions after this math box which are connected by the ENO are not executed.

## 3.11 Comparator

## 3.12 Move, jump, and call instructions

Conversion instruction

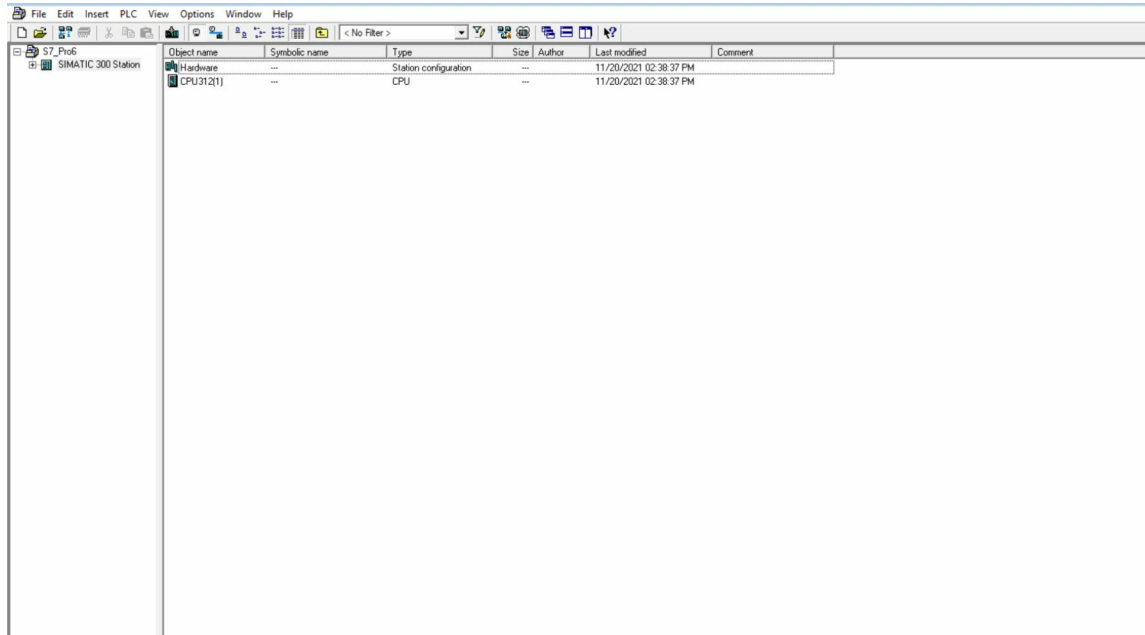
# 3.13 Analog signal programming



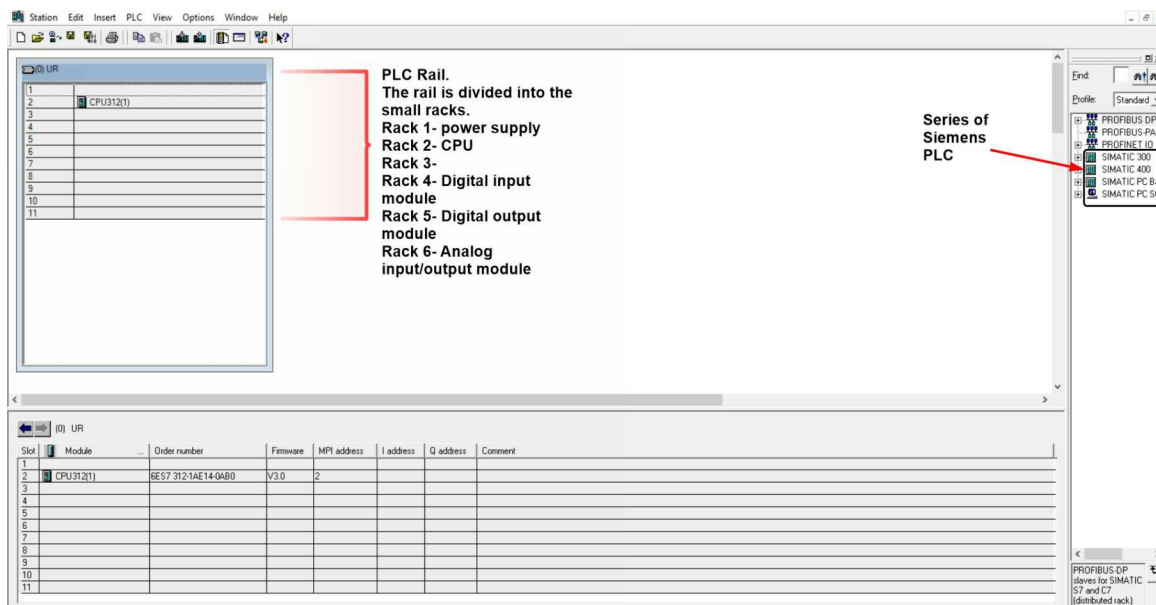
## 3.14 Converter



## 3.15 Simatic manager step7 software

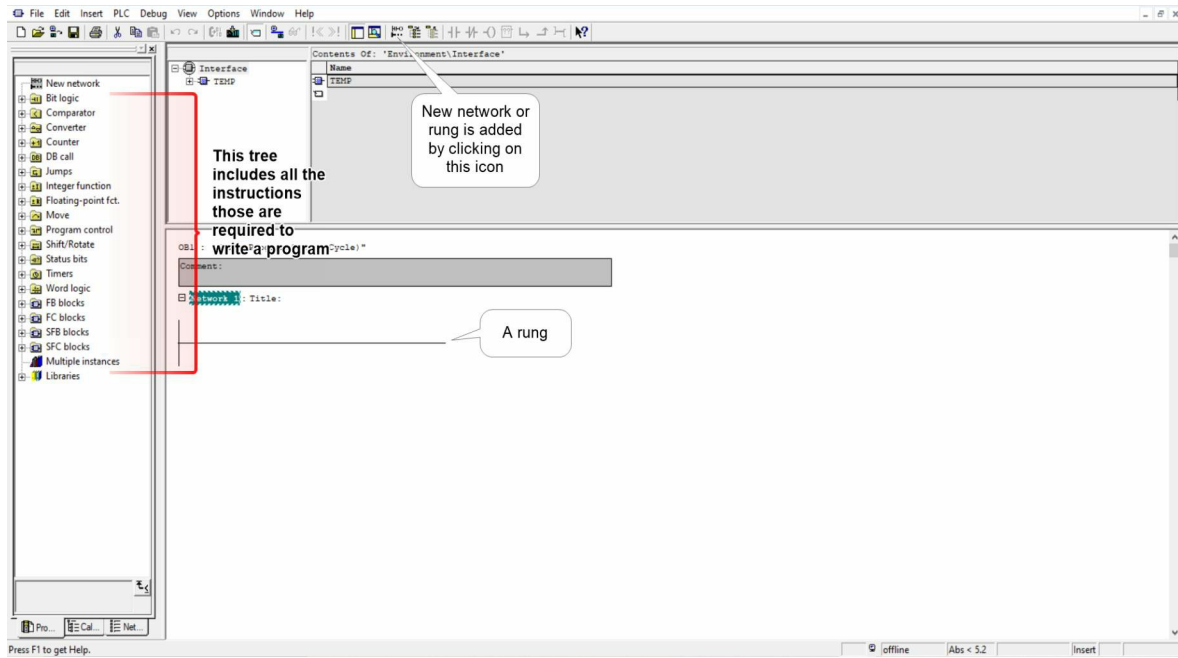


### Hardware configuration

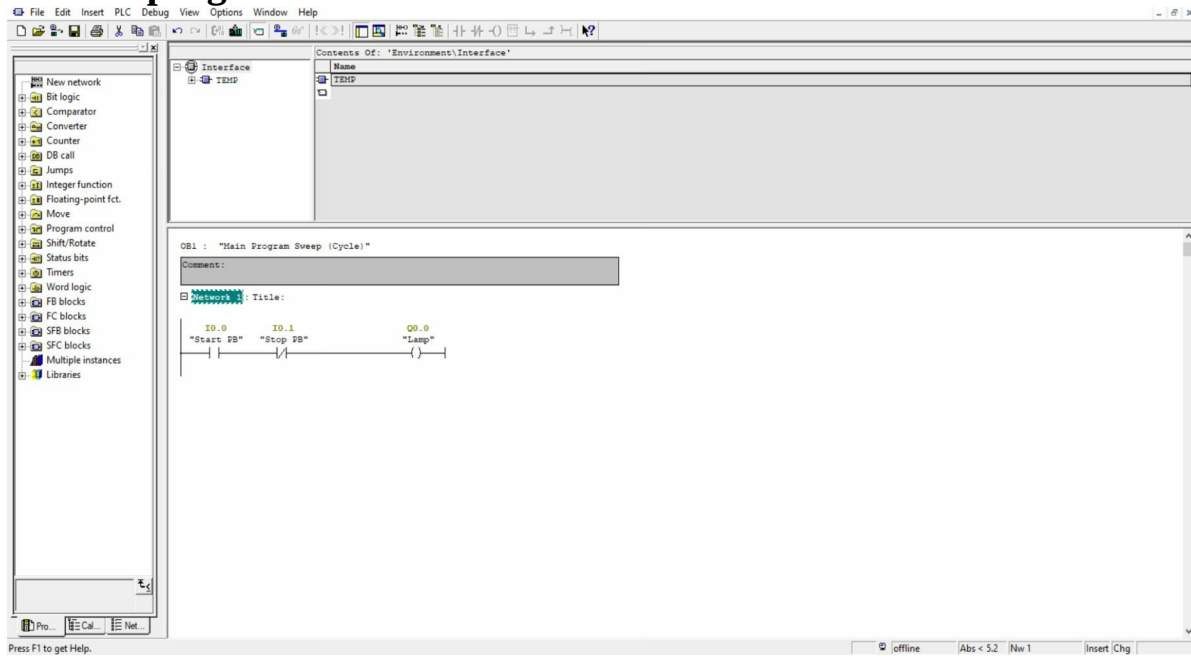


After selecting PLC and other modules click on save and compile.

### Overview screen



## Write a program



## 3.16 Plc programming examples using Simatic manager step 7 software

### 1. Tower lamps control

When switch 1 is pressed, lamps 1,2, and 3 get on,

When switch 2 is pressed, lamps 4 and 5 get on,

When switch 3 is pressed, lamps 1, and 5 will get off.

### Input and Output devices list

| Input devices   | Output devices |
|-----------------|----------------|
| Switch 1 (I0.0) | Lamp 1 (Q0.0)  |
| Switch 2 (I0.1) | Lamp 2 (Q0.1)  |
| Switch 3 (I0.2) | Lamp 3 (Q0.2)  |
|                 | Lamp 4 (Q0.3)  |
|                 | Lamp 5 (Q0.4)  |

Network: 1



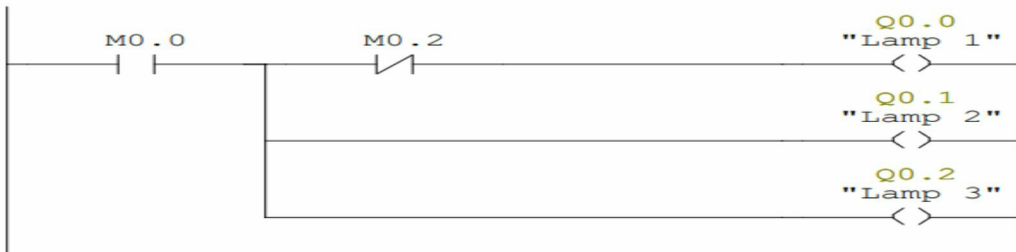
Network: 2



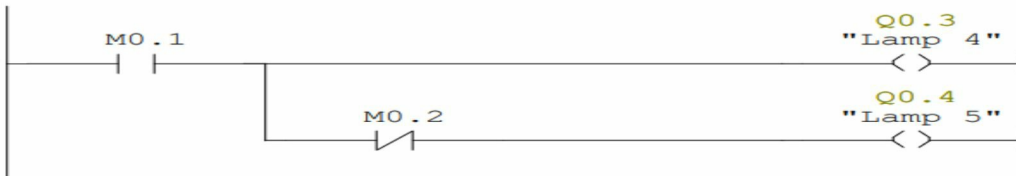
Network: 3



Network: 4



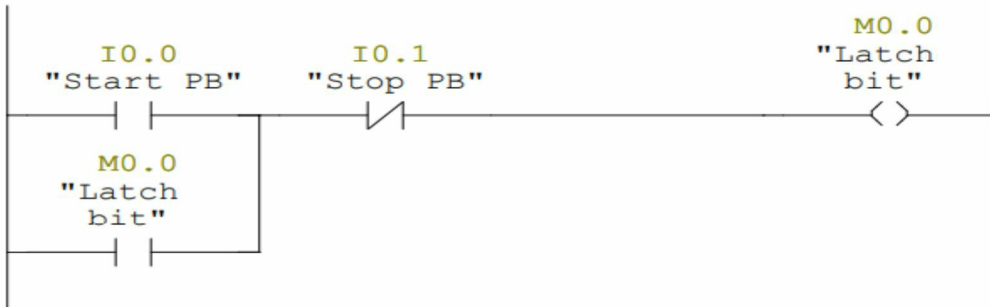
Network: 5



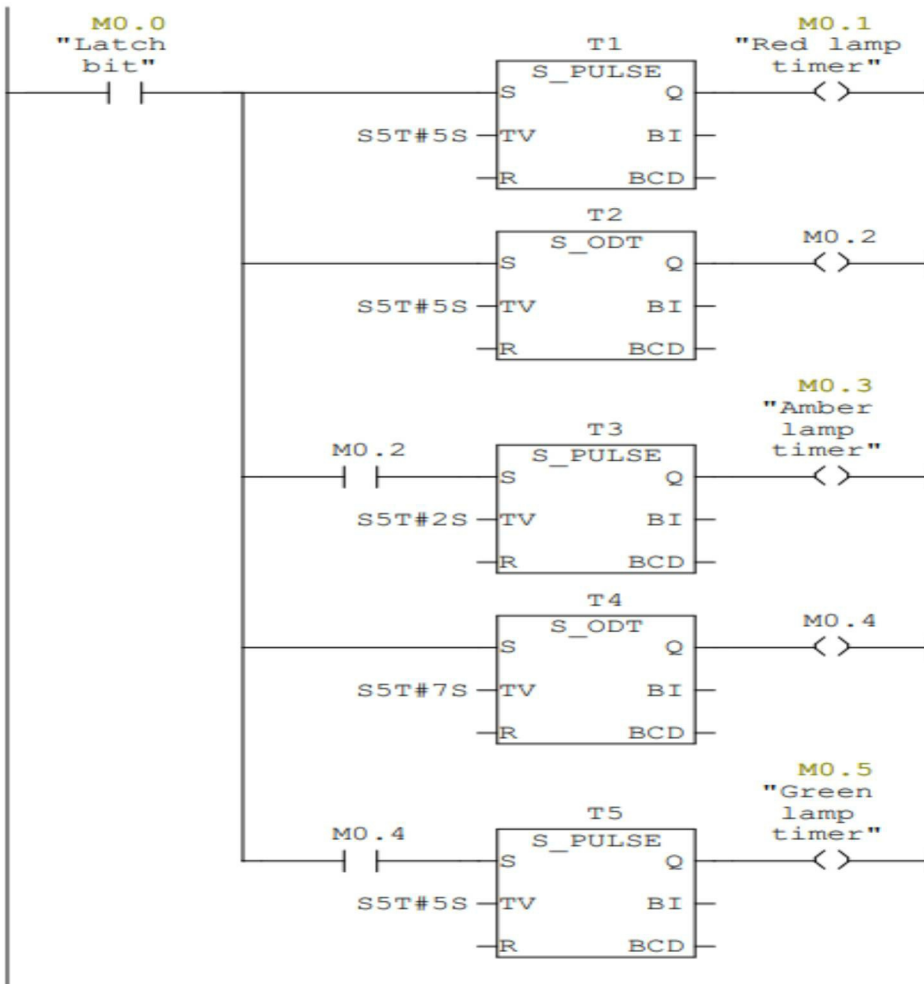
## 2. Traffic light control

The traffic light has red, amber, and green lamps. A red lamp gets on for 5 sec, then an amber lamp gets on for 2 sec and finally, the green lamp will get on for 5 sec. This cycle will continue for 1 minute.

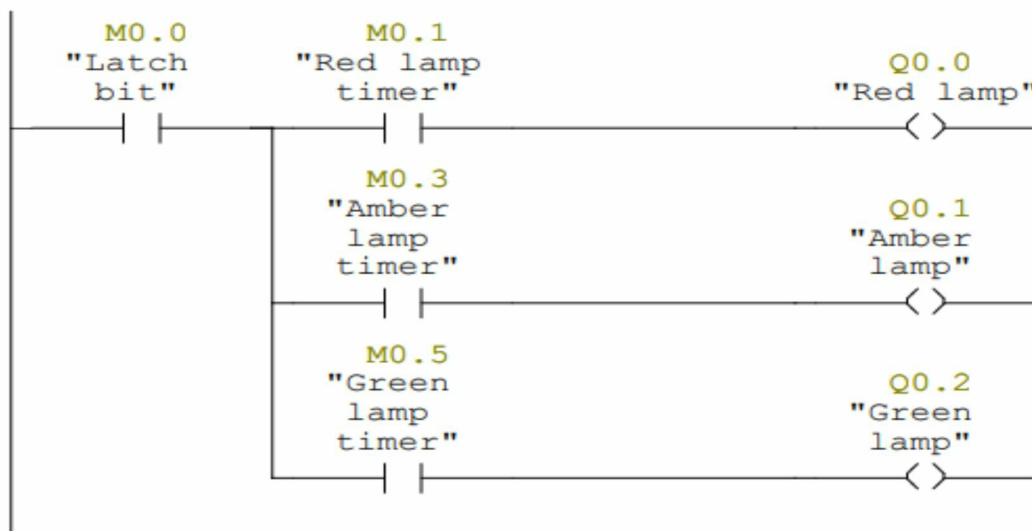
Network: 1 Holding circuit



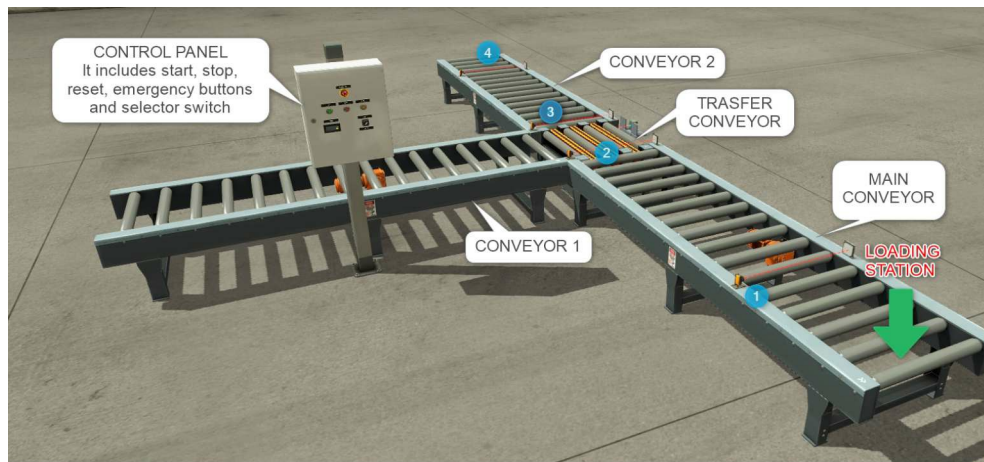
Network: 2 Timers



Network: 3      Lamps



### 3. Box sorting application



A box sorting application is shown in the above image. It consists of the main conveyor from where boxes are loaded and two unloading converters. Unloading conveyors carry the boxes in two different directions.

Four reflex sensors have been used along with the reflectors as shown in the image.

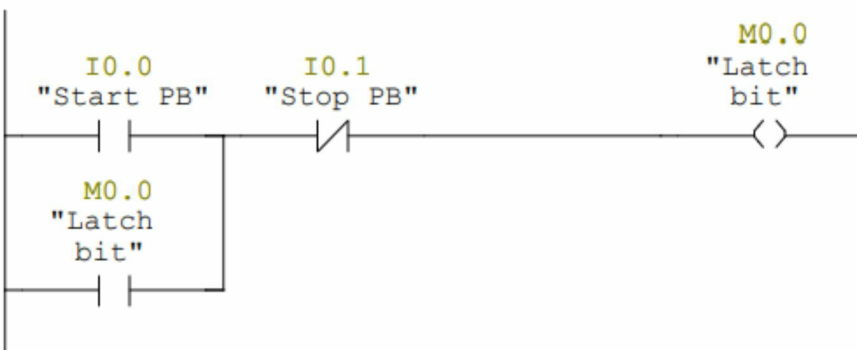
- When a reflex sensor detects the box on the main conveyor, the main conveyor will start after 1 second.

- Once a box approaches sensor 2, the main conveyor will stop.
- Transfer conveyor will transfer 5 boxes to conveyor 1 first, and then it will transfer the remaining boxes to the conveyor 2.
- The counter will reset when the stop button is pressed.

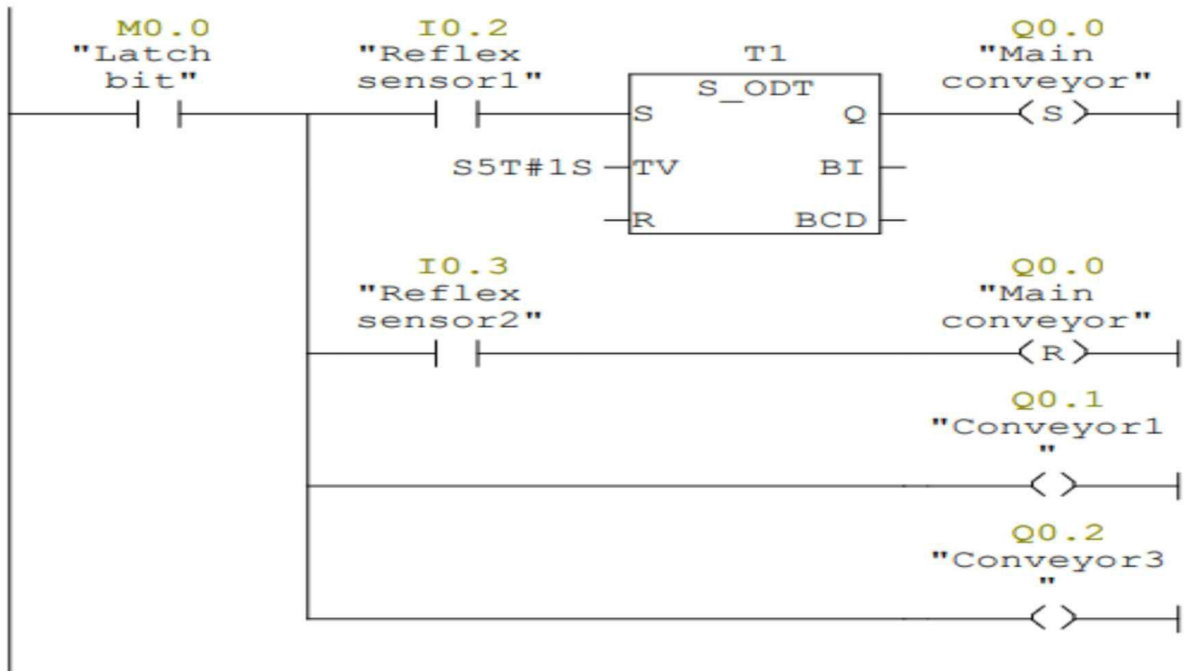
### Input and Output devices list

| Input devices     | Output devices           |
|-------------------|--------------------------|
| Start push button | Main conveyor motor      |
| Stop push button  | Conveyor 1 motor         |
| Reflex sensor 1   | Conveyor 2 motor         |
| Reflex sensor 2   | Transfer conveyor motor1 |
| Reflex sensor 3   | Transfer conveyor motor2 |
| Reflex sensor 4   |                          |

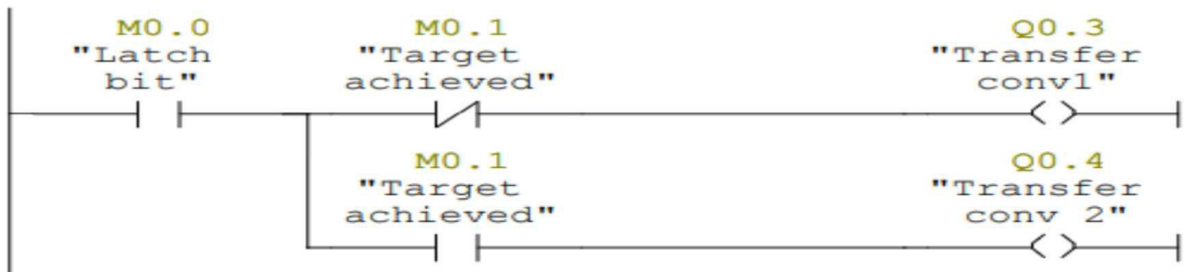
Network: 1      Holding circuit



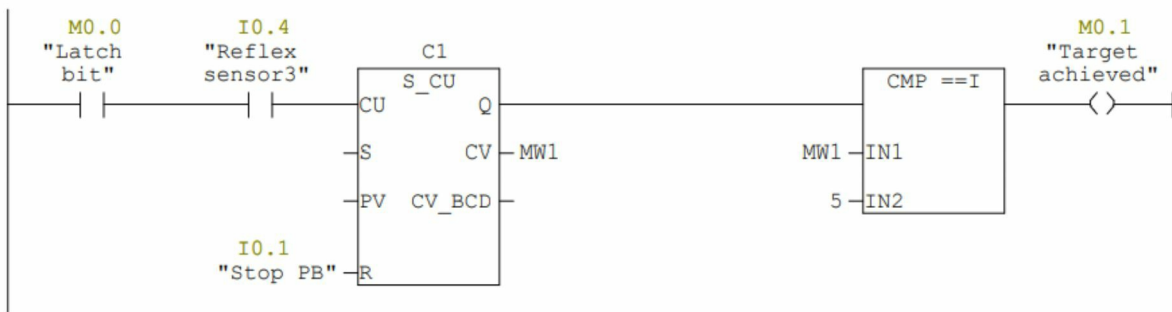
Network: 2 Conveyors



Network: 3 Transfer conveyor

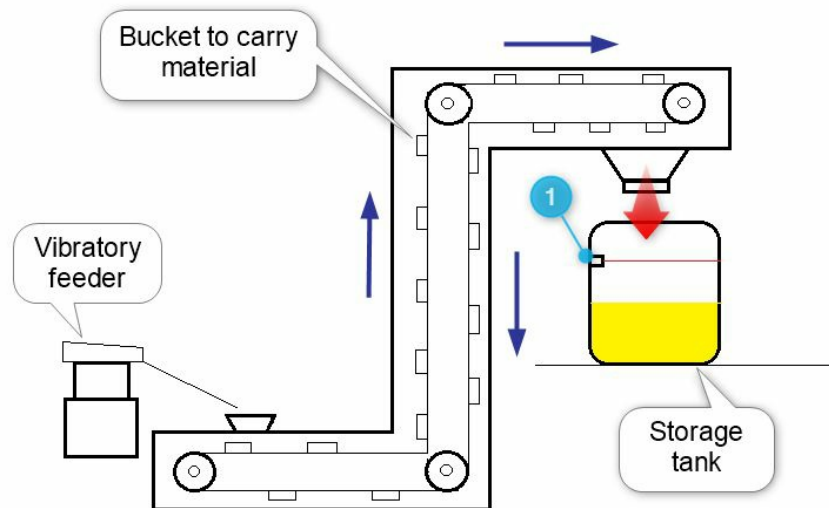


Network: 4 Counter



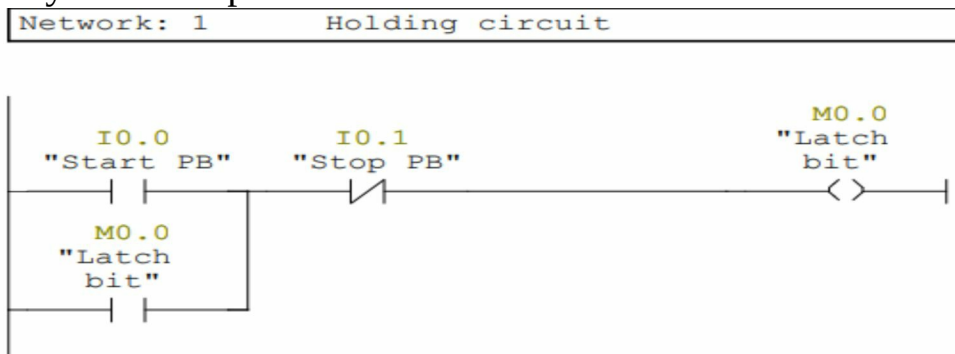


## 4. Material conveying application

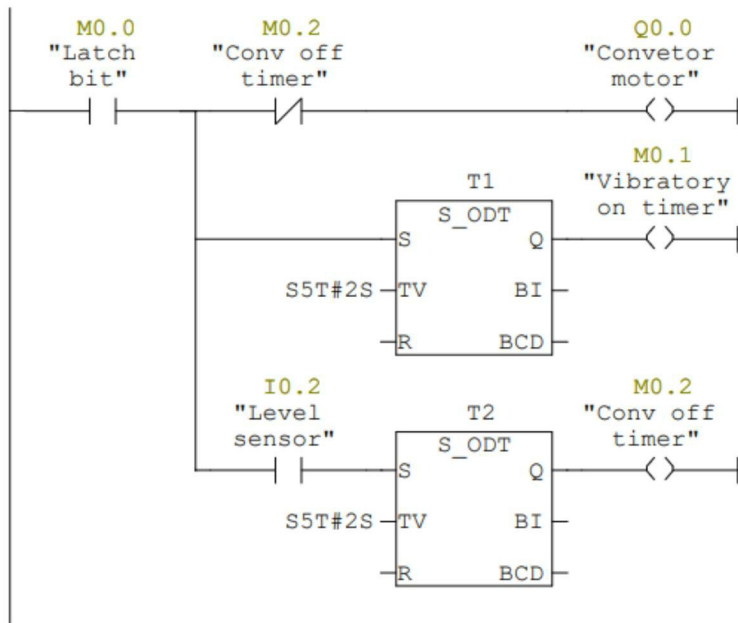


This type of material conveying system is used to carry material such as grains, powder, and small granules from ground level to a certain height. An electric motor is used to move the chain.

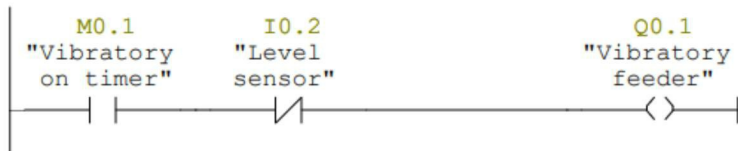
- When a start PB is pressed, the chain starts rotating and the vibratory feeder starts after 2 sec. A vibratory feeder continuously feeds the material into the bucket.
- Then the material is unloaded into the storage tank. If the material reaches a level sensor, it will stop the vibratory feeder first and then the conveyor will stop after 2 sec.



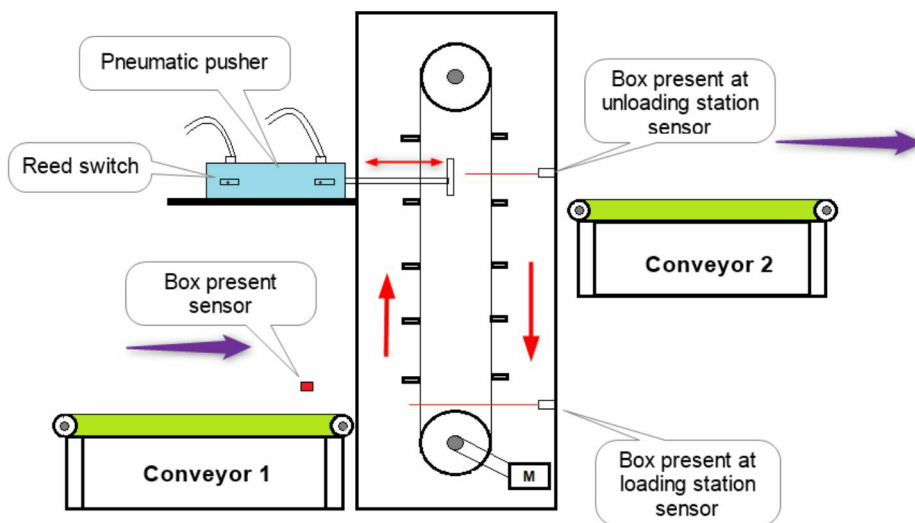
Network: 2 On delay Timers and controlling the conveyor



Network: 3 Controlling the vibratory feeder



## 5. Box lifter



This type of material conveying system is used to lift the boxes and transfer them to a certain height.

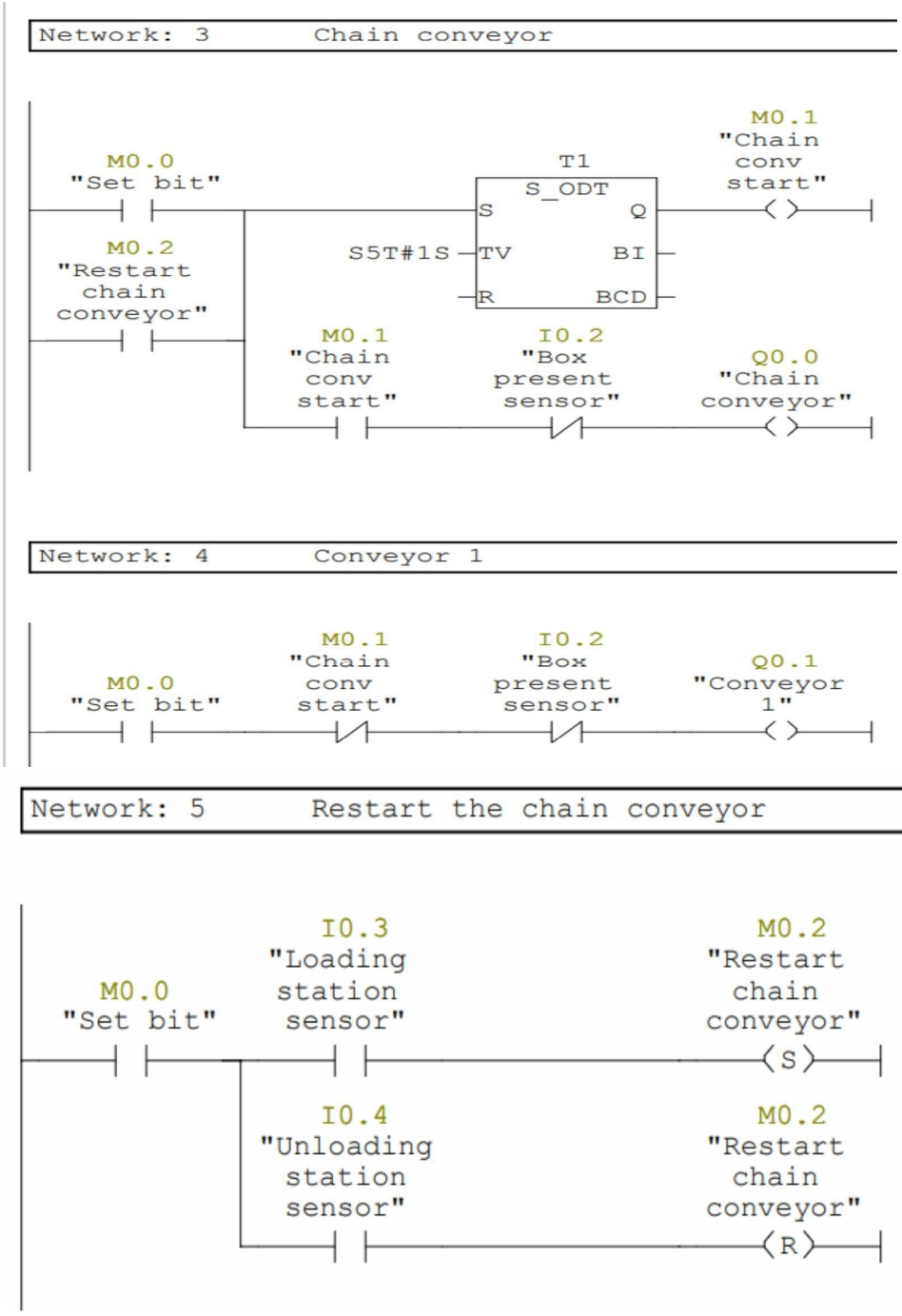
- When a start PB is pressed, the conveyor will start rotating. if a box is present at the conveyor 1 chain conveyor will stop. Then the box will enter into the conveyor, a box present sensor will sense the presence of the box at the loading station and the chain conveyor will start again after 1 sec.
- As the box reaches the unloading sensor, the chain conveyor will stop. A pusher will actuate after 2 sec and it will push the box on the conveyor 2. After pushing the box, the pusher will retract and the reed switch will sense the pusher (homing sensor). Once the reed switch senses the pusher, the conveyor will start again after 1 sec.
- This cycle will continue until a stop PB is pressed.

Network: 1      Set bit

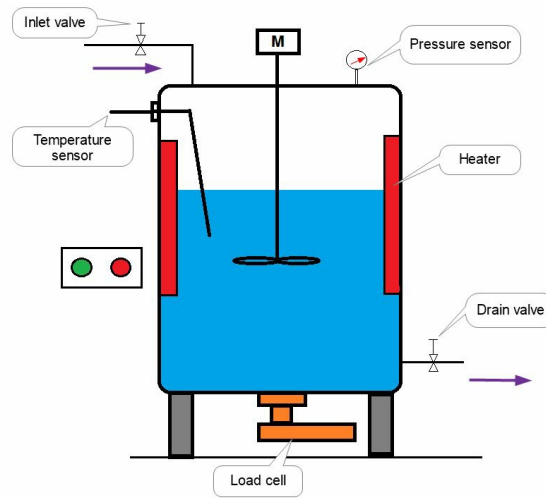


Network: 2      Reset bit





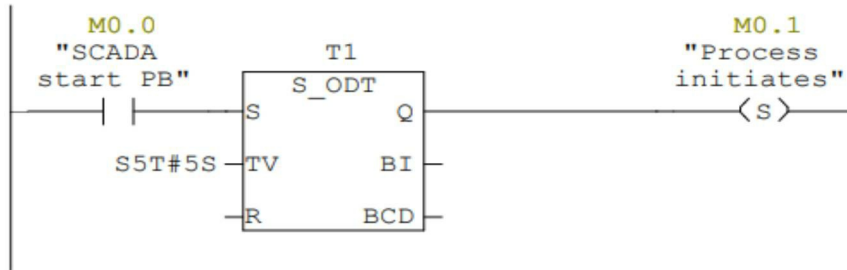
## 6. Reaction tank



**A reaction tank is used to heat the chemical solution liquid to a certain temperature, then maintain it to that temperature to achieve the required results, and then drain it.**

- The process initiates when the start command is given from the SCADA screen. The PLC will switch on the inlet valve and allow the chemical liquid to enter into the tank. A load cell under the tank will measure the amount of liquid. Once it measures the 10kg weight of resin, PLC will switch off the inlet valve.
- After 10 sec, an agitator will start stirring the chemical liquid for 60 sec, then after 30-sec heaters will start heating the liquid until 120 deg C to 125 deg C temperature is achieved.
- Once the temperature inside the tank is achieved, PLC will switch on the supply of nitrogen gas and it will maintain 2 bar pressure inside the tank for 5 minutes.
- After the completion of 5 minutes, PLC will open the drain valve to drain out the chemical solution liquid. The agitator will stop after draining out (load cell will send a signal)

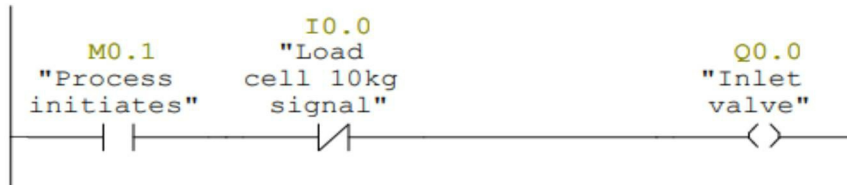
Network: 1      Process initiates



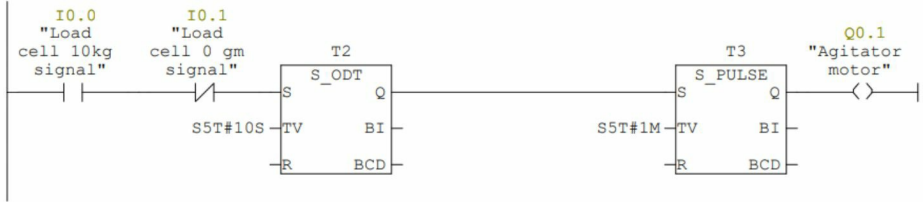
Network: 2      Process stops



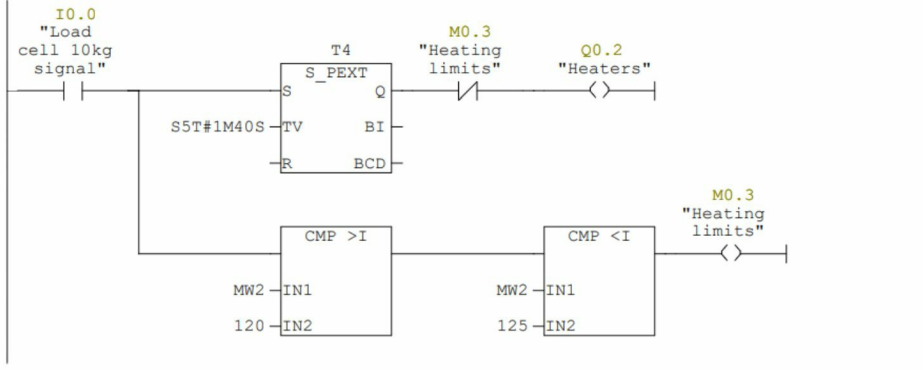
Network: 3      Inlet valve



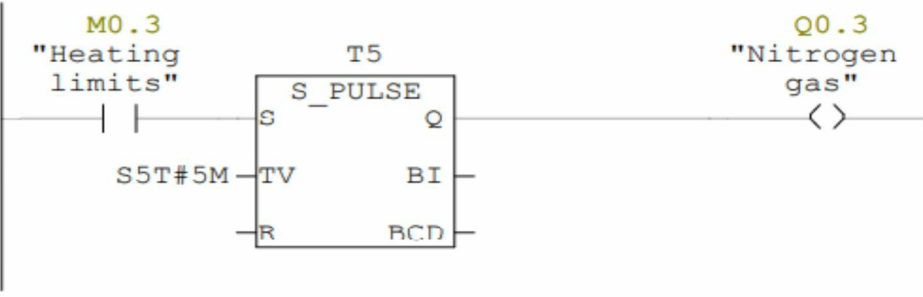
Network: 4      Agitator



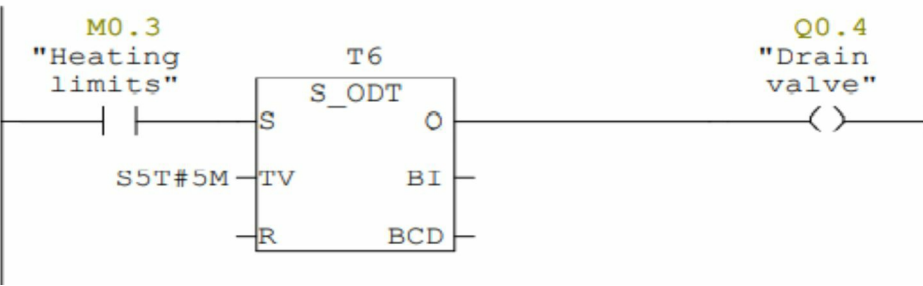
Network: 5      Heaters



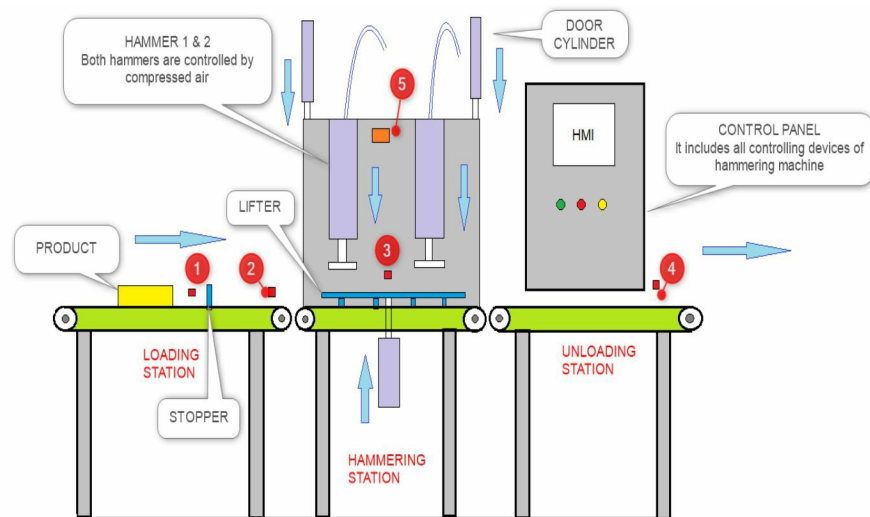
Network: 6      Nitrogen gas pressure



Network: 7      Drain valve



## Case study 2- Pneumatic hammering machine



### Operating procedure

- A product is loaded manually at the loading station. If the machine is in auto mode and there is no fault (emergency pressed) then the machine should start when start PB is pressed.
- If a product is present inside the cabin (sensor 3), then the door1 will remain closed and the stopper will pop up and it will stop the product.
- Door 1 will open only when a product is not present at the hammering station.
- Once the product reaches sensor 3, the hammering station conveyor will stop after 1 second and both doors will get closed. A lifter will lift the product and start the hammer after 2 seconds. The hammering operation will continue as per set the time from HMI.
- After completion of the hammering operation, a lifter will come down, both doors will go up and restart the hammering conveyor.
- Keep unloading station conveyor in running mode. Sensor 4 will keep counting the production.

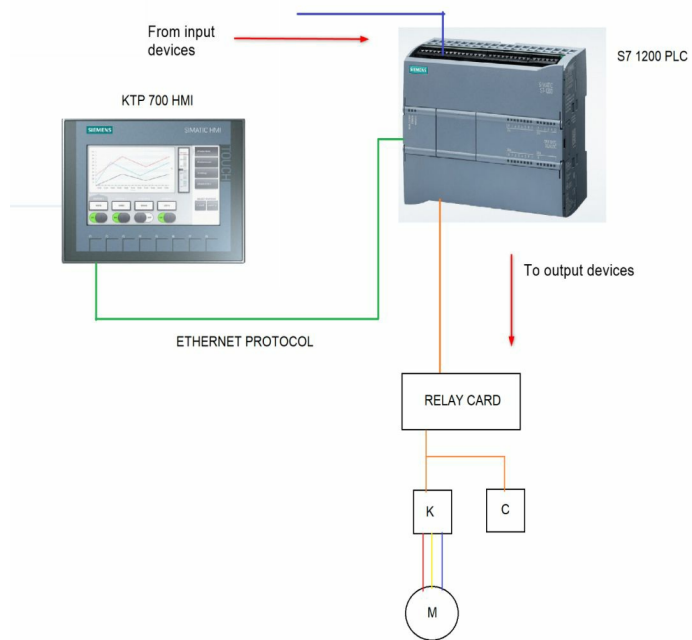
### Input and Output devices list

|  |  |
|--|--|
|  |  |
|--|--|



| Input devices                         | Output devices                            |
|---------------------------------------|---|
| Start PB (I0.0)                       | Loading station conveyor (Q0.0)           |
| Stop PB (I0.1)                        | Hammering station conveyor (Q0.1)         |
| Emergency PB (I0.2)                   | Unloading station conveyor (Q0.2)         |
| 1. Product present sensor1 (I0.3)     | Pneumatic hammer solenoid coil (Q0.3)     |
| 2. Product present sensor2 (I0.4)     | Door cylinder1 solenoid coil (Q0.4)       |
| 3. Product present sensor3 (I0.5)     | Door cylinder2 solenoid coil (Q0.5)       |
| 4. Product present sensor4 (I0.6)     | Lifter cylinder up solenoid coil (Q0.6)   |
| 5. Door limit switch (I0.7)           | Stopper cylinder coil (Q0.7)              |
| Door cylinder down reed switch (I1.0) | Red lamp (Q1.0)                           |
| Door cylinder up reed switch (I1.1)   | Green lamp (Q1.1)                         |
| Lifter up reed switch (I1.2)          | Amber lamp (Q1.2)                         |
| Auto/man selector switch (I1.3)       | Lifter cylinder down solenoid coil (Q1.3) |
|                                       | Hooter (Q1.4)                             |

## Architecture



**PLC Controller- S7 1200**  
**HMI- KTP 700**  
**PLC & HMI communication- Ethernet**  
**Motor control- Hardwire (via contactor)**

**Program using simatic manager step7 software**

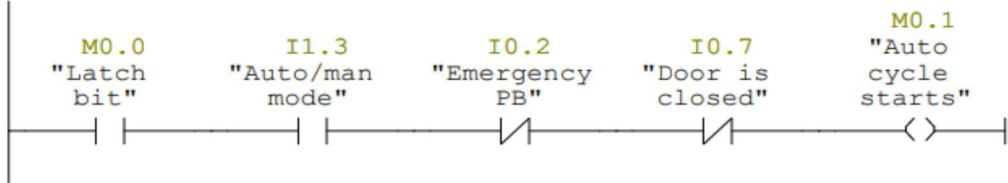
Network: 1      Set bit



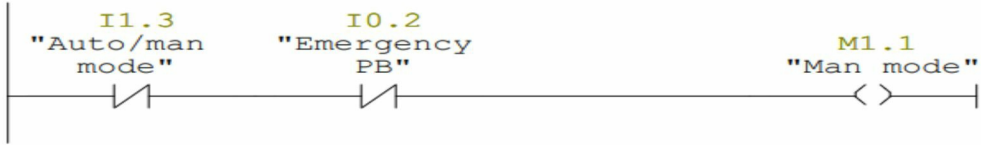
Network: 2      Reset bit



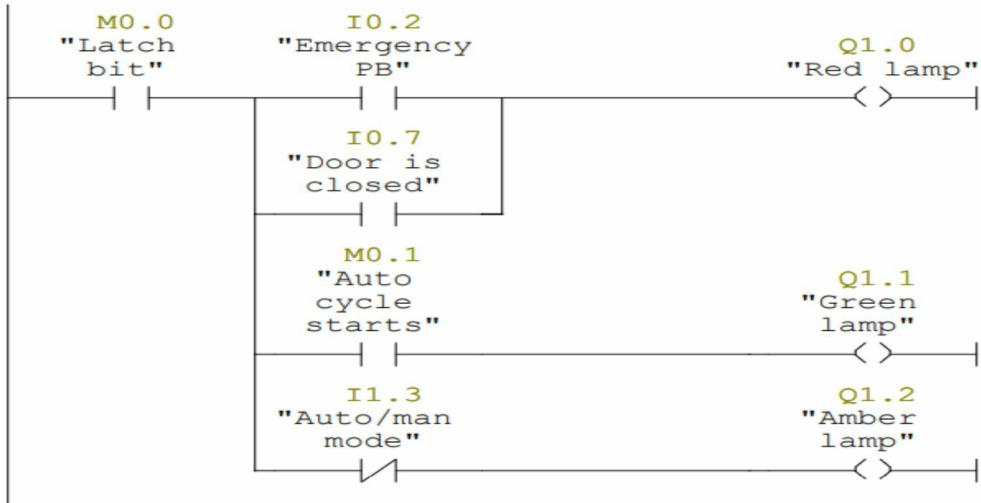
Network: 3      Auto cycle



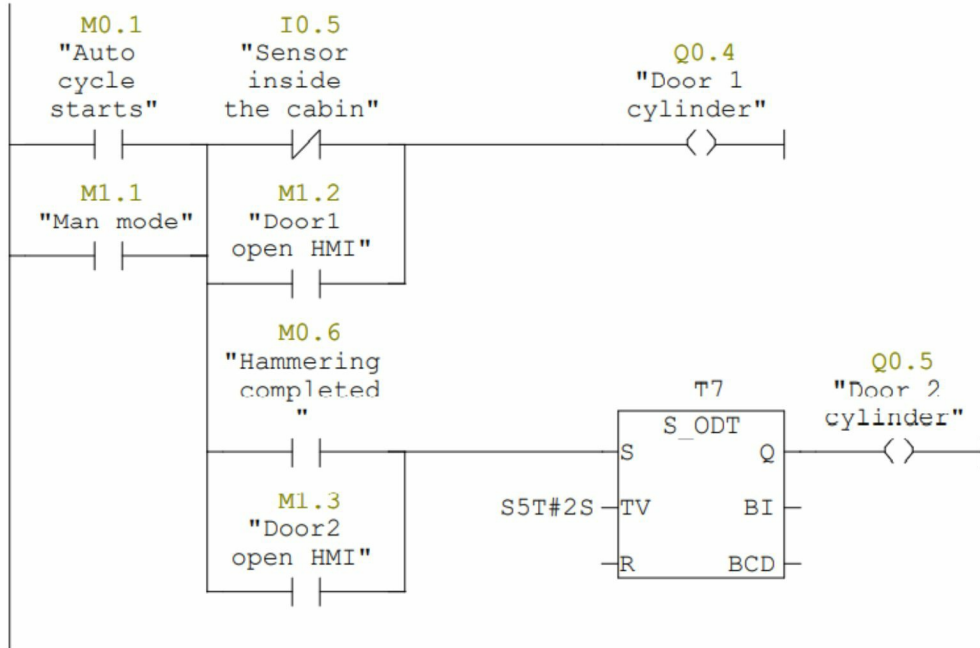
Network: 4



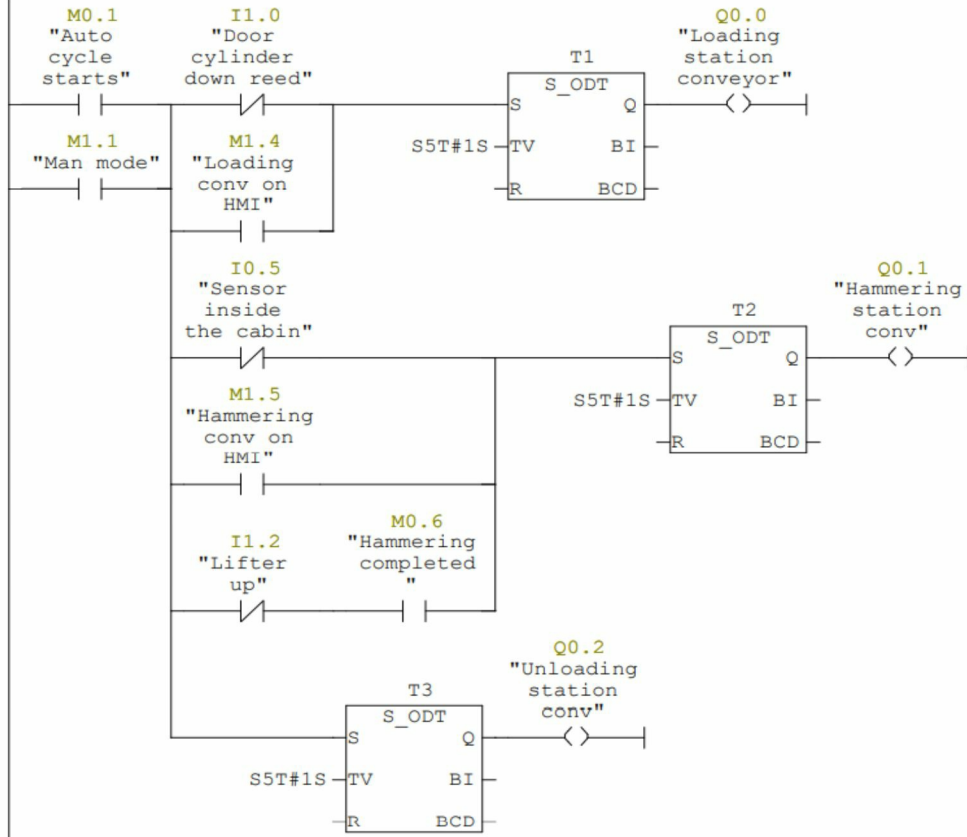
Network: 5 Tower lamp



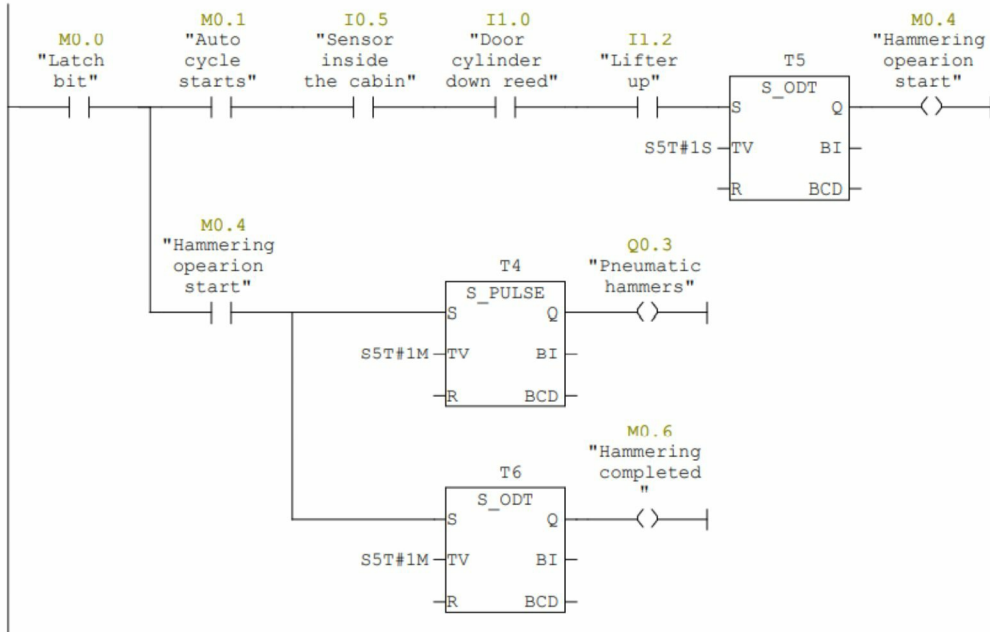
Network: 6      Hammering cabin door



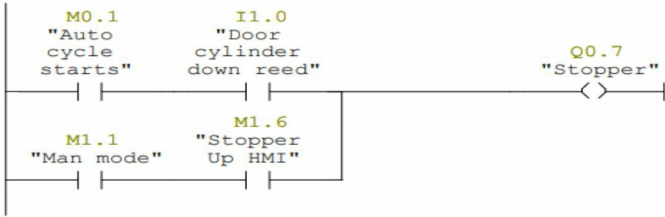
Network: 7 Conveyors



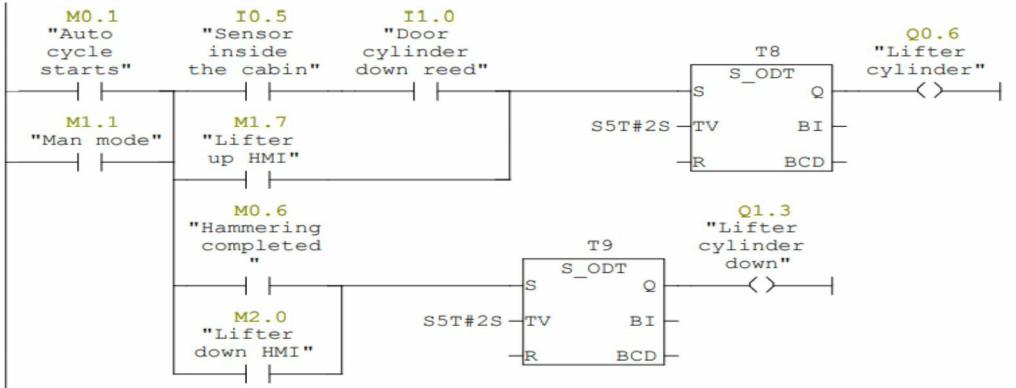
Network: 8 Pneumatic hammer operation



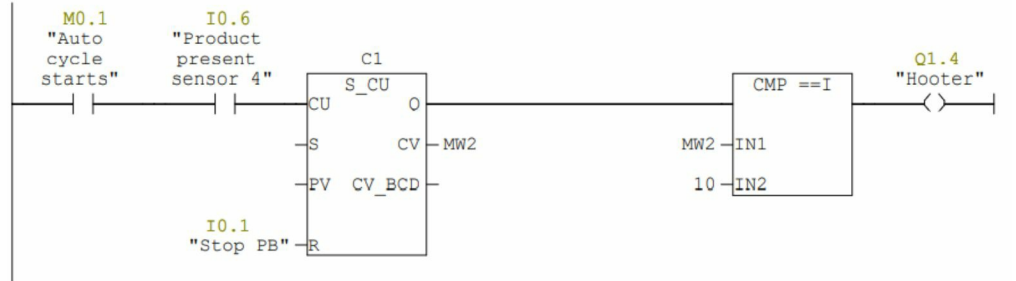
Network: 9 Stopper



Network: 10 Lifter



Network: 11 Production report



## 4. Mitsubishi PLC



### 4.1 Mitsubishi PLC series

#### FX PLC SERIES

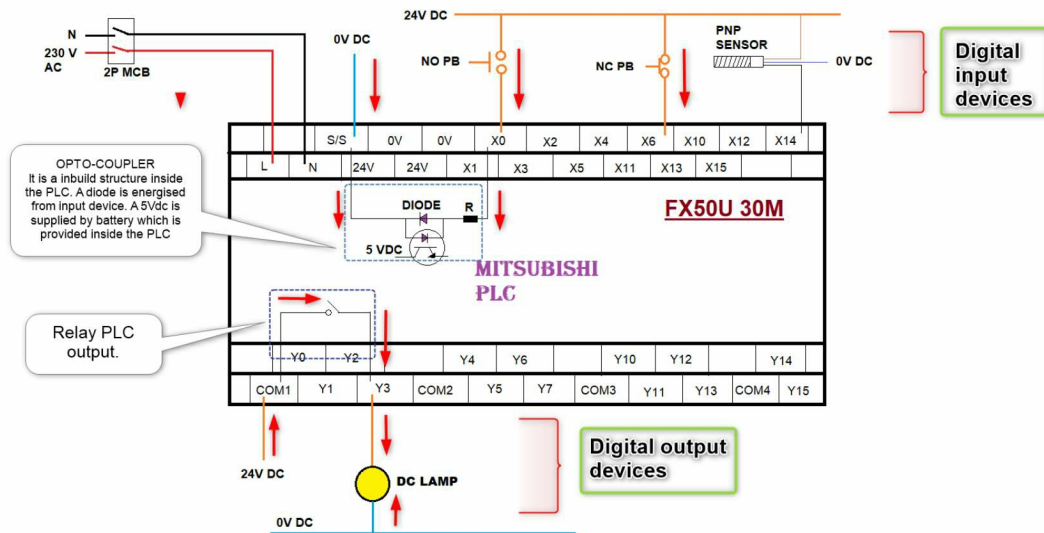
1. FX-1S PLC
2. FX-1N PLC
3. FX-2N PLC
4. FX-3U PLC



5. FX-3G PLC
6. FX5U PLC



## 4.2 Mitsubishi PLC wiring configuration



## 4.3 PLC and their programming software

| <b>PLC</b> | <b>Software</b> |
|------------|-----------------|
| Q series   | GT developer    |
| FX series  | GX works2       |
| FX5U       | GX Works3       |

## 4.4 Mitsubishi communication protocols

1. USB cable
2. Ethernet

## 4.5 Addressing for ladder logic programming

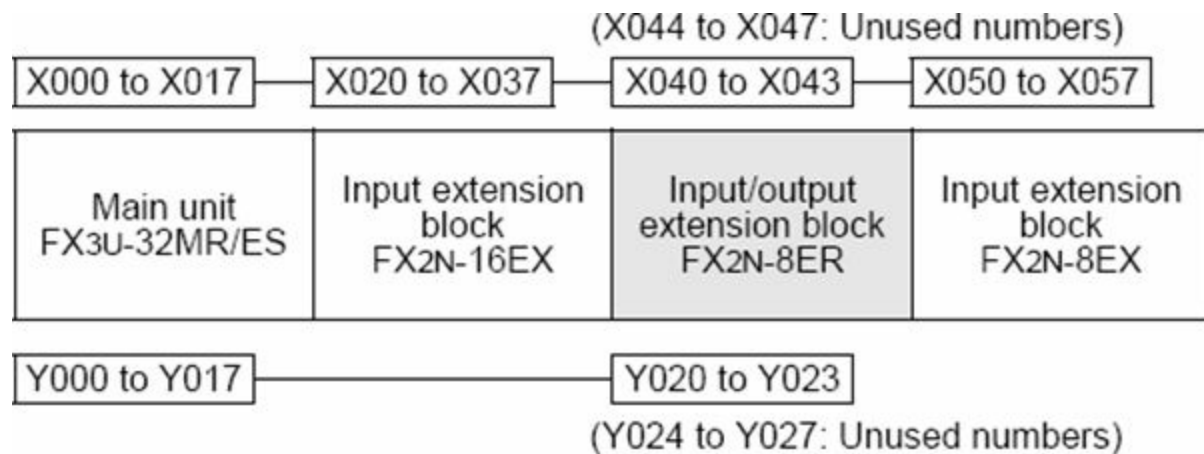
**Input (X) :** Inputs are designed to give commands and data from external devices,

e.g. pushbuttons, select switches, etc., to the PLC

**Output (Y):** Outputs are used to provide the control results of a program from the PLC to external devices, e.g. solenoids, signal lamps

**Internal relays (M):** Internal relays are auxiliary relays used in a CPU module and not latched

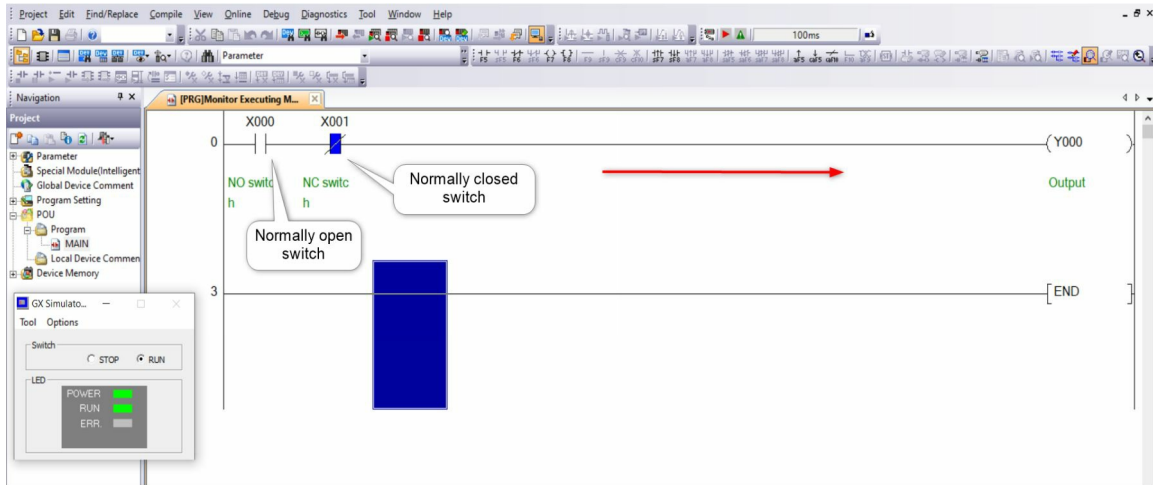
**Latch relays (L) :** Latch relays are auxiliary relays used in a CPU module and latched (backed up at power failure)



- Addressing of inputs and outputs is in Octal (X0-X7, X10-X17)
- Addressing for both inputs and outputs start at 0 (X0 and Y0)
- Addressing is consecutive
- If input/output powered extension units/blocks have been connected when the power is turned on, the main unit automatically assigns the input/output numbers (X/Y) (octal) to the units/blocks.
- Therefore, it is unnecessary to specify the input/output numbers with parameters.



## 4.6 Digital signal programming



We have chosen FX3U PLC in which digital inputs start from X00, X01, X02, X03, X04, X05, X07, X10, X11....

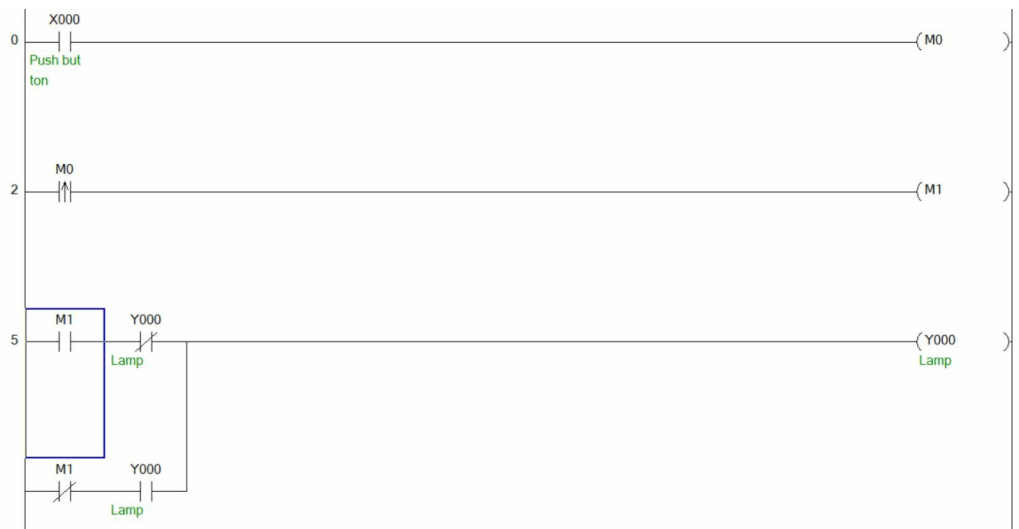
And outputs start from Y00, Y01, Y02, Y03, Y04, Y05, Y06, Y07, Y10, Y11...

## 4.7 Rising / Falling pulse instructions

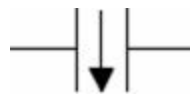
### 1. Rising pulse



- True for one scan on up transition at address
- **Operating a lamp from a push button**
- When a push button is pressed the first time a lamp will glow. When a push button is pressed a second time a lamp will turn off.

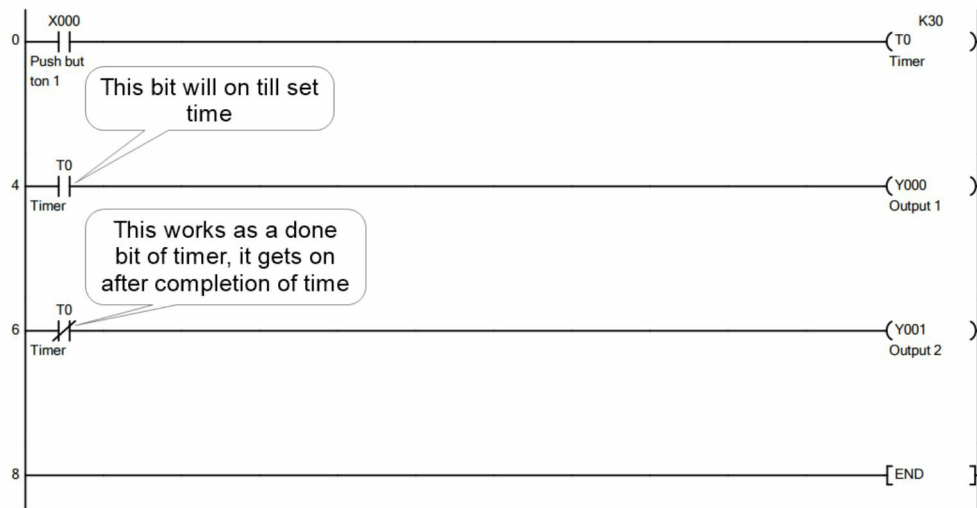


### 2. Falling pulse



- True for one scan on the down transition at address

## 4.8 Use of a timer

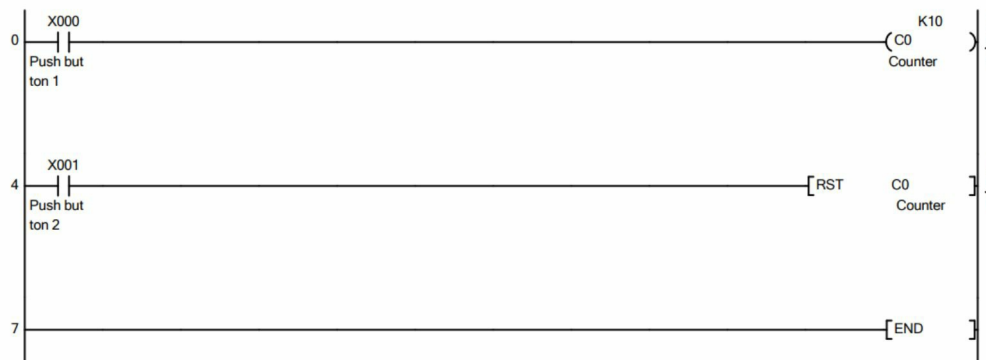


| Time Base         | PLC Timer Address |                 |         |
|-------------------|-------------------|-----------------|---------|
|                   | FX1S              | FX1N/FX2N/FX2NC | FX3U    |
| 100ms             | 0-62              | 0-199           | 0-199   |
| 10ms              | 32-62             | 200-245         | 200-245 |
| 1ms (Retentive)   | -                 | 246-249         | 246-249 |
| 100ms (Retentive) | -                 | 250-255         | 250-255 |
| 1ms               | 63                | -               | 256-511 |

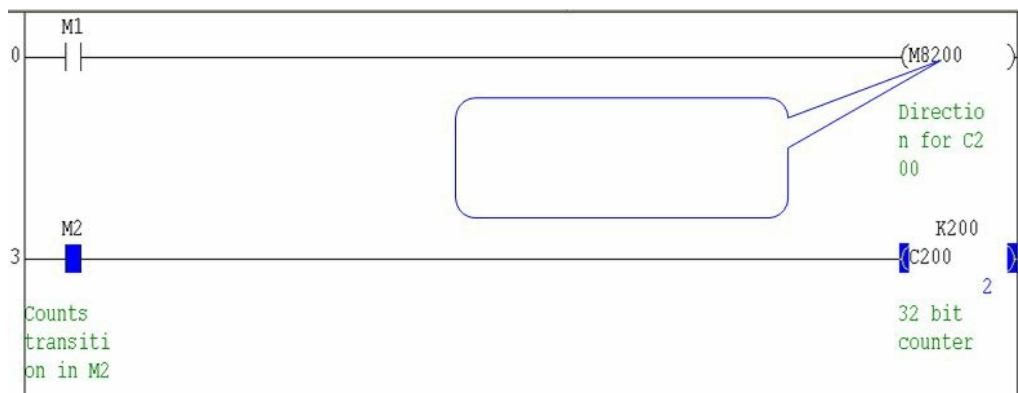
| Device name                                  | Description  |            |                     |
|--|--------------|------------|---------------------|
| <b>Timer (on-delay timer)</b>                |              |            |                     |
| 100 ms                                       | T0 to T191   | 192 points | 0.1 to 3,276.7 sec  |
| 100 ms [for subroutine or interrupt routine] | T192 to T199 | 8 points   | 0.1 to 3,276.7 sec  |
| 10 ms  | T200 to T245 | 46 points  | 0.01 to 327.67 sec  |
| Retentive type for 1 ms                      | T246 to T249 | 4 points   | 0.001 to 32.767 sec |
| Retentive type for 100 ms                    | T250 to T255 | 6 points   | 0.1 to 3,276.7 sec  |
| 1 ms   | T256 to T511 | 256 points | 0.001 to 32.767 sec |



## 4.9 Use of counter



- It is also an output instruction
- Counter is retentive
- Reset instruction is used to reset it
- 16/32 bit counters available
- Understand the following:
  - –Preset: Constants or Register
  - –Accumulator: Current status of the counter, use the address of Counter Device
  - –Contact or Done bit: Can be accessed using the address of Counter Device



Direction control M8### for C###

For C200 , M8200 is used to control the direction

| Counter  |              |            |   |
|--|--------------|------------|---|
| General type up counter<br>(16 bits) [variable]                              | C0 to C99    | 100 points | Counts 0 to 32,767<br>The setting can be changed between the latched (battery backed) type and the non-latched type using parameters. |
| Latched (battery backed) type up counter<br>(16 bits) [variable]             | C100 to C199 | 100 points |   |
| General type bi-directional counter<br>(32 bits) [variable]                  | C200 to C219 | 20 points  | -2,147,483,648 to +2,147,483,647 counts   |
| Latched (battery backed) type bi-directional counter<br>(32 bits) [variable] | C220 to C234 | 15 points  | The setting can be changed between the latched (battery backed) type and the non-latched type using parameters.                       |

## 4.10 Math instructions

| Math Function  | 16-Bit Data | 32-Bit Data | Floating Point |
|----------------|-------------|-------------|----------------|
| Addition       | ADD         | DADD        | DEADD          |
| Subtraction    | SUB         | DSUB        | DESUB          |
| Multiplication | MUL         | DMUL        | DEMUL          |
| Division       | DIV         | DDIV        | DEDIV          |
| Square Root    | SQR         | DSQR        | DESQR          |

## 4.11 Compare instructions

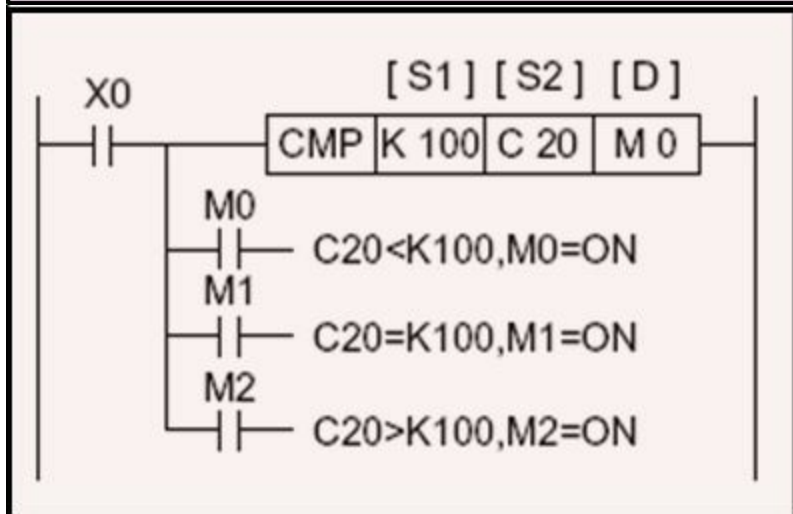
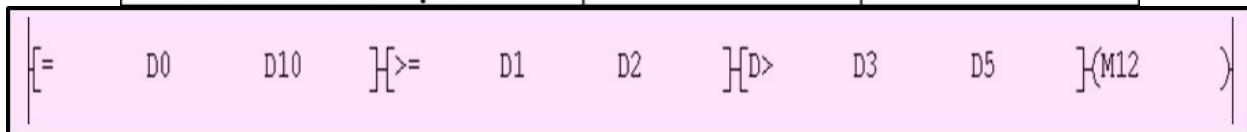
- Instruction compares two data and results in a true or false condition
- It can be used with other sequence instructions
- All instructions are input instructions except CMP which is an output instruction
- Compare instructions are available to compare

-16-bit data

32-bit data

### INLINE comparisons

| Comparison Function | 16-Bit Data | 32-Bit Data |
|---------------------|-------------|-------------|
| Greater Than        | >           | D>          |
| Equal To            | =           | D=          |
| Less Than           | <           | D<          |
| Greater Than Equal  | >=          | D>=         |
| Not Equal           | <>          | D<>         |
| Less Than Equal     | <=          | D<=         |



The data of S1 is compared to the data of S2. The result is indicated by 3 bit devices specified from the head address entered as D.

The bit devices indicate:

- S2 is less than S1 - bit device D is ON
- S2 is equal to S1 - bit device D+1 is ON
- S2 is greater than S1 - bit device D+2 is ON

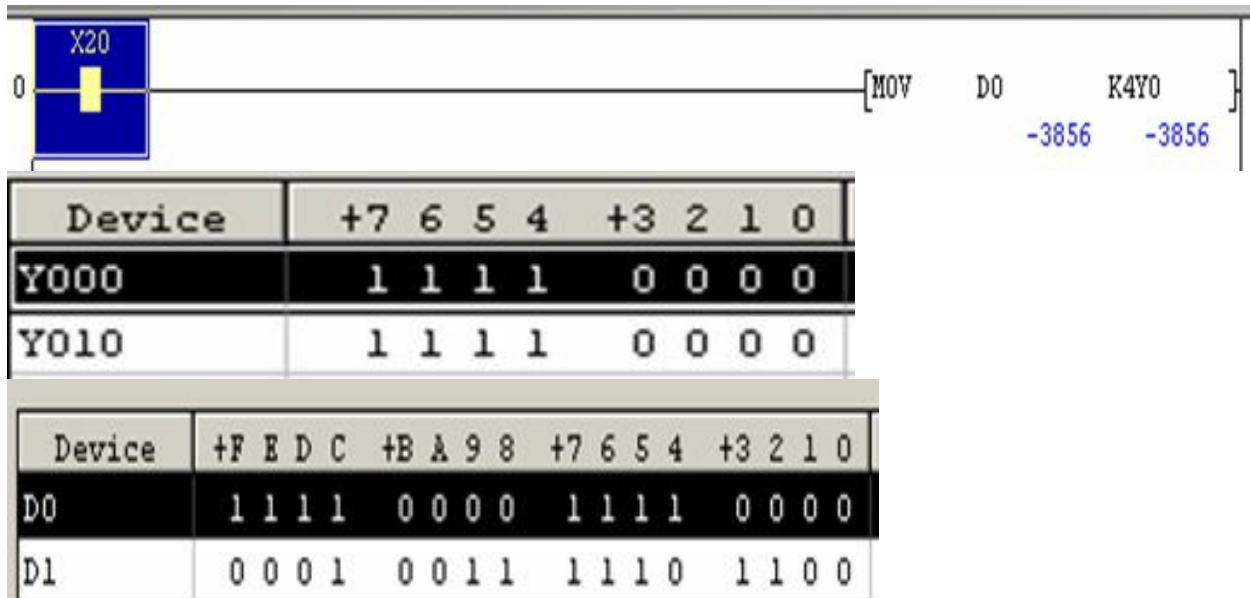
Try other variations like DCMF

## ZCP Instruction

- A single data value (S) is compared against a data range (S1-S2)
- S is less than S1, bit device D is ON
- S is equal to or between S1 and S2 - bit device D+1 is ON
- S is greater than S2 - bit device D+2 is ON

## 4.12 Move, jump, and logical instructions

### 1. Move instruction



## 4.13 Analog signals



Two types:

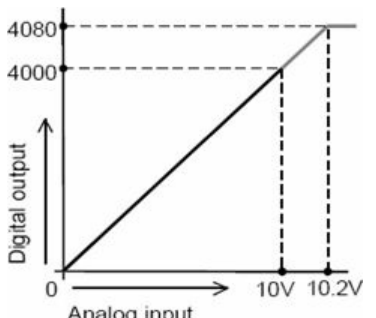
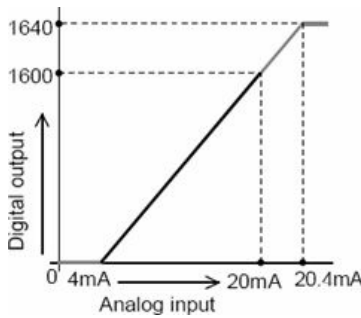
- Special Adapter (Left hand side )
- Special Function modules (Right Hand Side)

### 1. Special Adapter

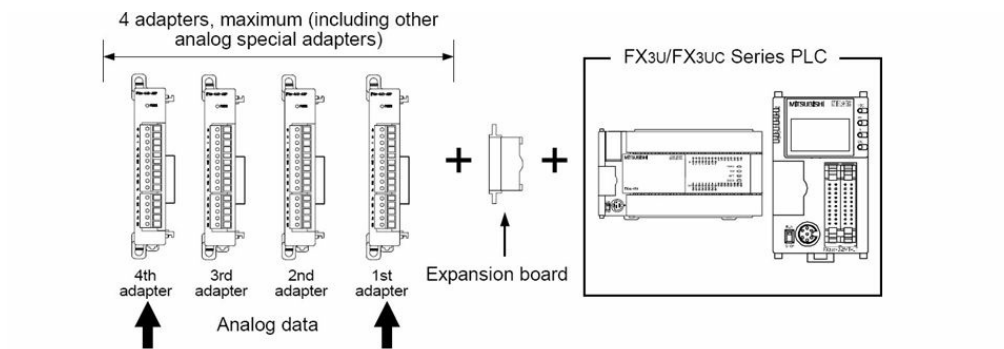
- Always connected on left side of PLC
- Up to four no. of adapter can be connected
- The expansion board is needed to connect the special adapter
- To use the high-speed input/output special adapter, be sure to connect the high-speed input/output special adapter first, and then connect the analog special adapter
- No need to use the traditional To/From instructions to configure and operate.
- The input analog data will be converted into digital data and then stored in the special devices of the FX Series PLC.
- The number of averaging time and the input mode to be specified
- Special auxiliary relays (10 points) and special data registers (10 points) are assigned starting from the adapter nearest the main unit
- Analog special adapter nearest the main unit is counted as the 1st adapter, and the next adapter as the 2nd adapter, and so on
- Do not include the high speed input/output special adapter and the communication special adapter



## Input characteristics

|                       | Voltage Input   | Current Input  |
|-----------------------|---|--|
| Resolution            | 2.5mV (10V/4000)  | 10 $\mu$ A (16mA/1600)   |
| Conversion time       | 200 micro sec per channel   | The data will be updated every scan time   |
| Input Characteristics |  |  |

## FX3U-4AD-ADP

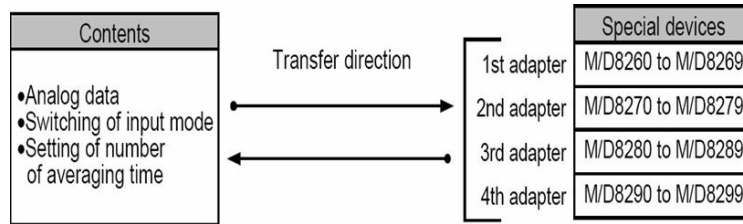


- It is a four channels analog input adapter
- Current or voltage inputs or combination of both.
- Up to four Adapter (including analog outputs also) can be used with FX

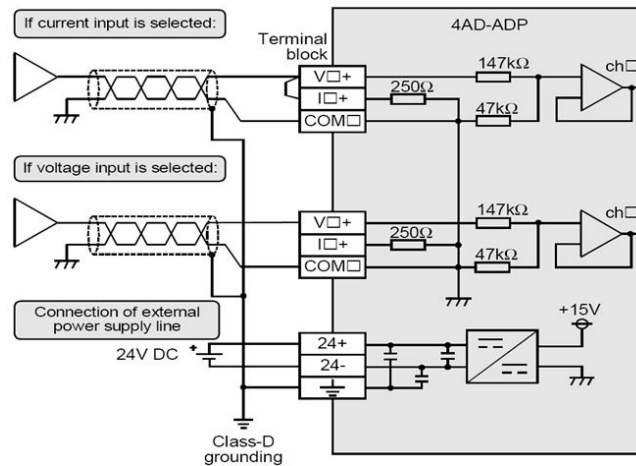
3U PLC as shown.

- Converted data of each channel will be automatically written in the special data register of the FX3U/FX3UC Series PLC.

Assignment of data registers and bit devices are as shown.



### Analog sensors' wiring with FX3U-4AD-ADP adapter



| Device | Function   | Status       |
|--------|--|--------------|
| M8260  | Input mode of channel 1; ON=Current Input, OFF=Voltage Input | Read / Write |
| M8261  | Input mode of channel 2; ON=Current Input,                   | Read / Write |

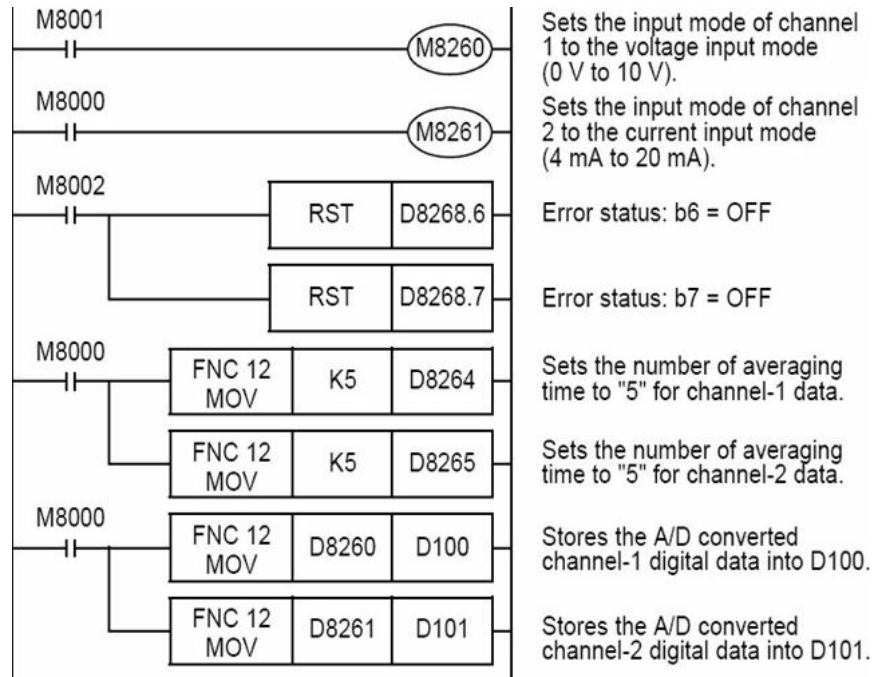
|       |   |              |
|-------|---|--------------|
|       | OFF=Voltage Input   |              |
| M8262 | Input mode of channel 3; ON=Current Input, OFF=Voltage Input      | Read / Write |
| M8263 | Input mode of channel 4; ON=Current Input, OFF=Voltage Input      | Read / Write |
| D8260 | Channel-1 input data  | Read         |
| D8261 | Channel-2 input data  | Read         |
| D8262 | Channel-3 input data  | Read         |
| D8263 | Channel-4 input data  | Read         |
| D8264 | Number of averaging time for channel-1 (Setting range: 1 to 4095) | Read / Write |
| D8265 | Number of averaging time for channel-2 (Setting range: 1 to 4095) | Read / Write |
| D8266 | Number of averaging time for channel-3 (Setting range: 1 to 4095) | Read / Write |
| D8267 | Number of averaging time for channel-4 (Setting range: 1 to 4095) | Read / Write |
| D8268 | Error status  | Read / Write |

D8269

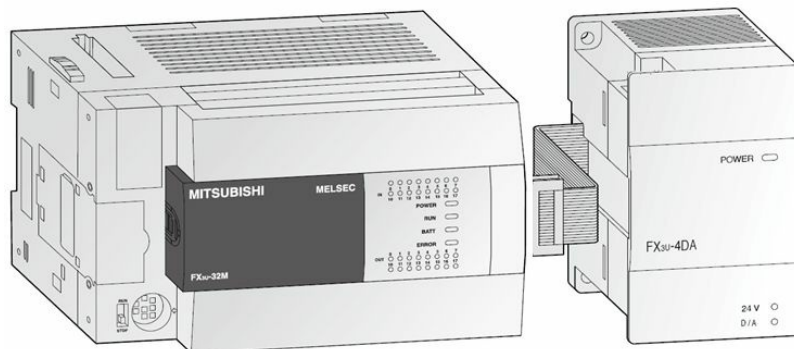
Model code

Read

### Example



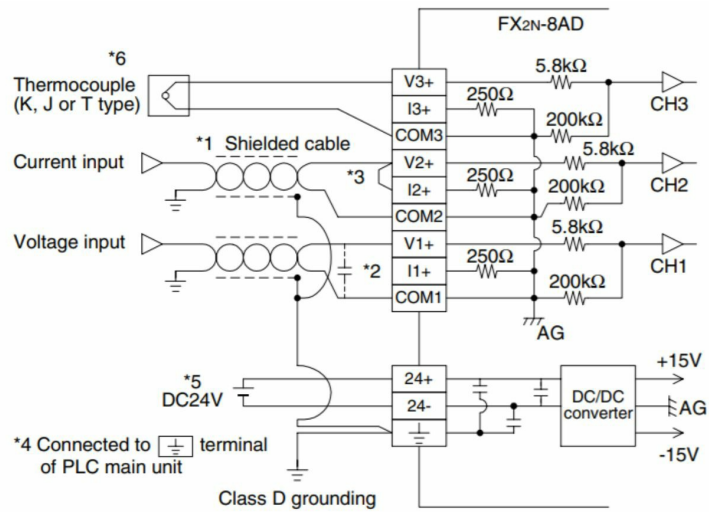
## 2. Special function module



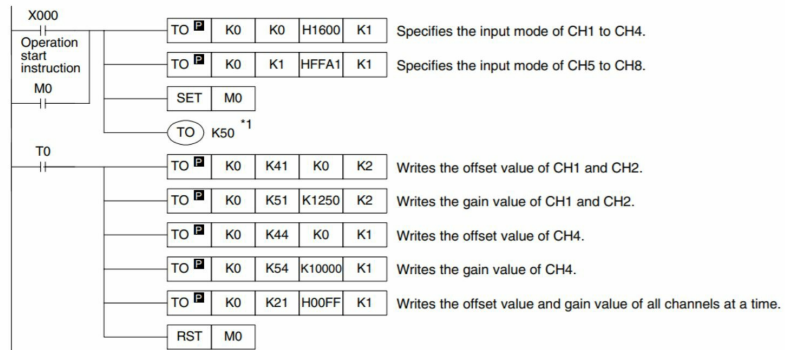
- The FX2N-8AD analog input block converts 8 points of analog input values into digital values and transfers them to the PLC.
- 
- Connect the special function block to the right side of the FX3U Series PLC
- Up to 8 special function blocks can be connected

- To/From instructions are required to interface

## Wiring



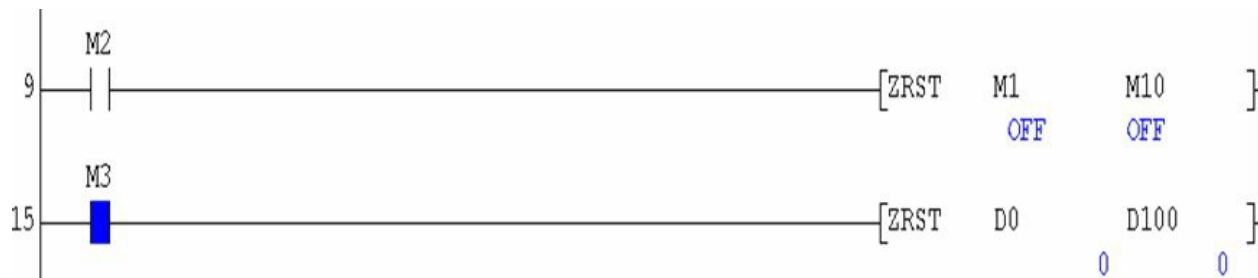
## Program



## 4.14 Other instructions

### 1. Zone Reset Instruction :ZRST

- It is used to reset a range of addresses.
- The arguments can be bit or word devices

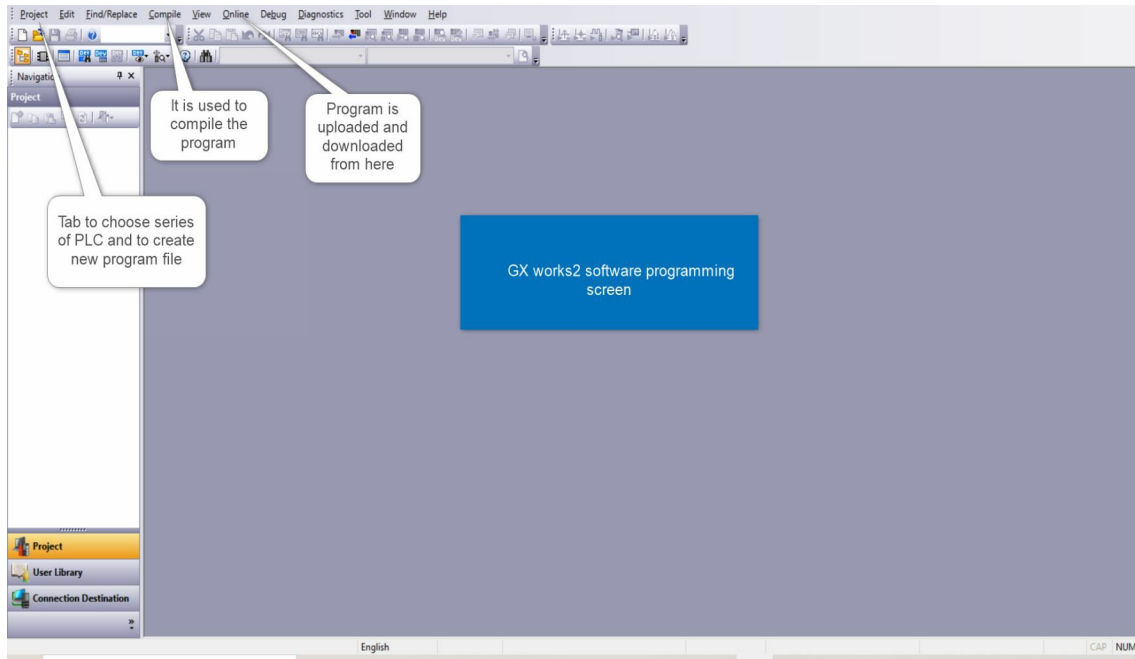


### 2. Master Control Instruction

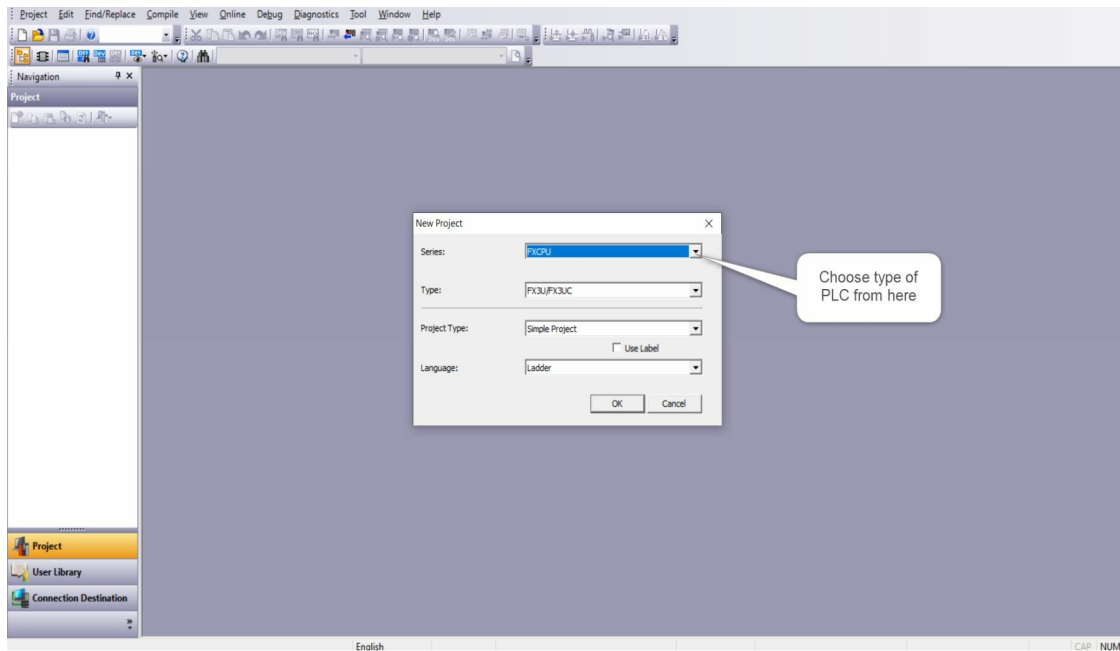
- It is an output instruction
- It activates or deactivates the execution of a group or zone of ladder rungs
- Nesting pointer is used with MC
- Nesting Range: N0 to N7
- Third argument is bit device which is turned on if MC is enabled
- It is always followed with MCR instruction
- MCR ends the zone or group of rungs

# 4.15 Mitsubishi GX works2 PLC programming software

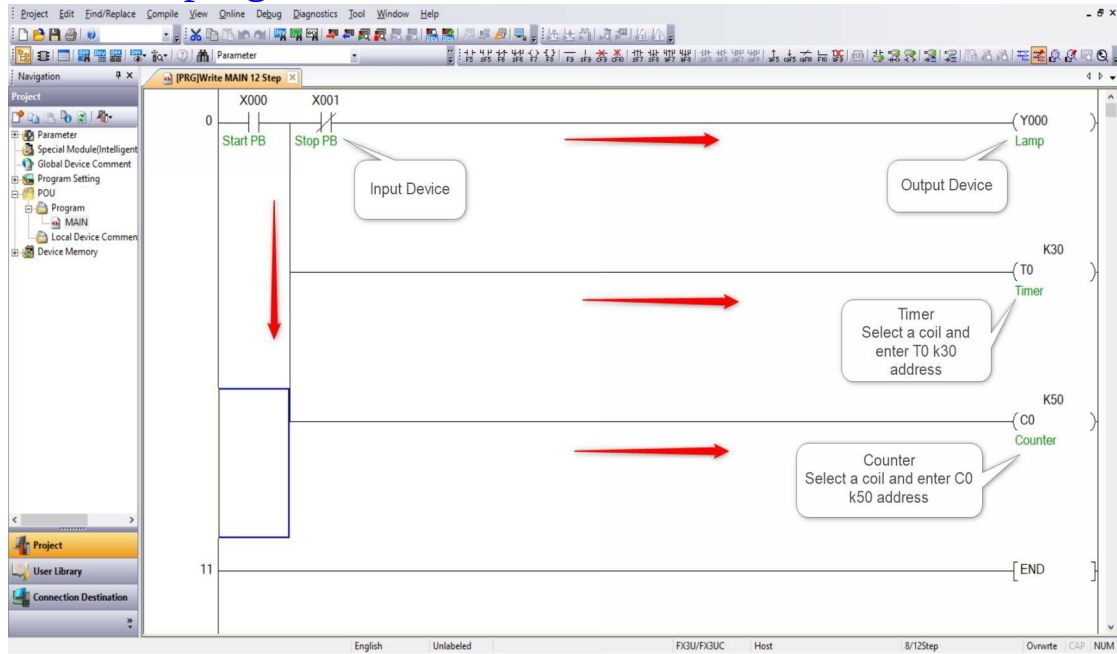
## 1. Overview screen



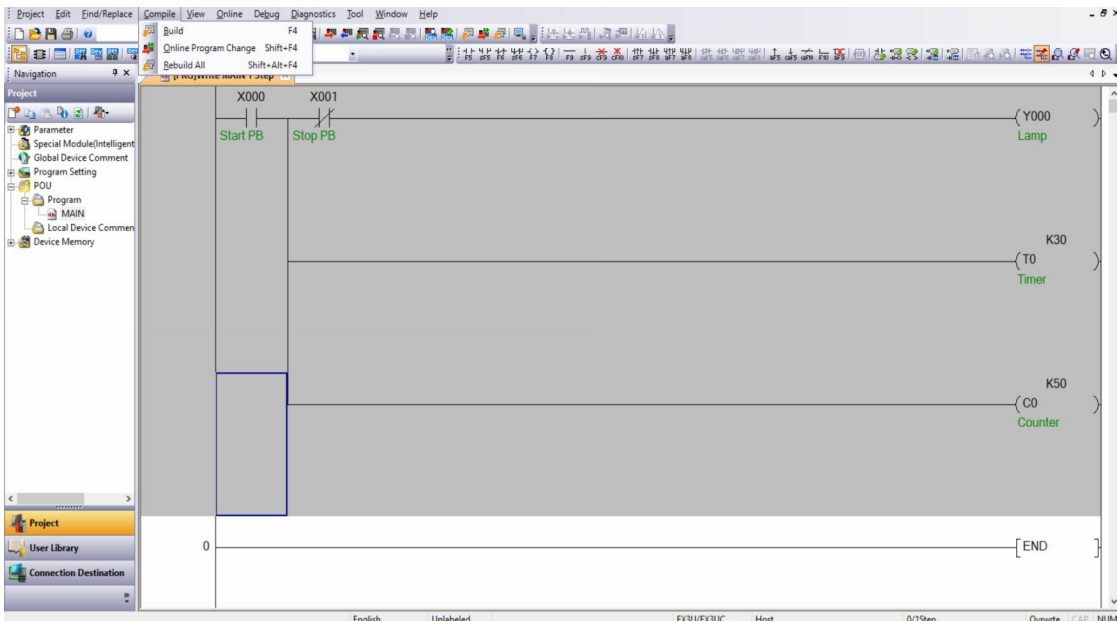
## Selecting series of PLCs



## Write a PLC program

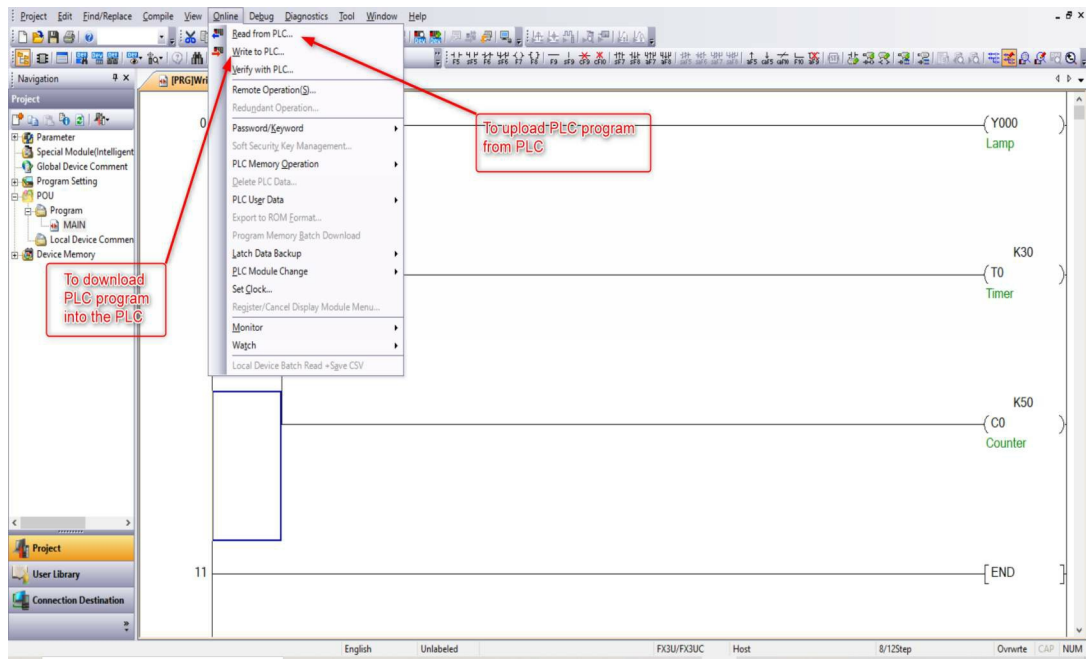


## Screen before rebuild



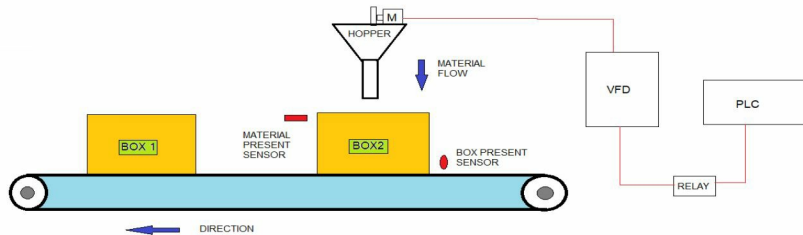
## Upload and download PLC program





## 4.16 PLC programming examples using works2 software

### 1. Conveyor application



A box filling line is as shown above. It consists of a flat belt conveyor which carries boxes and a hopper which fills powder into the box. Both conveyor and hopper are controlled by two different VFDs. The quantity of powder can be controlled by rotation of the motor through VFD.

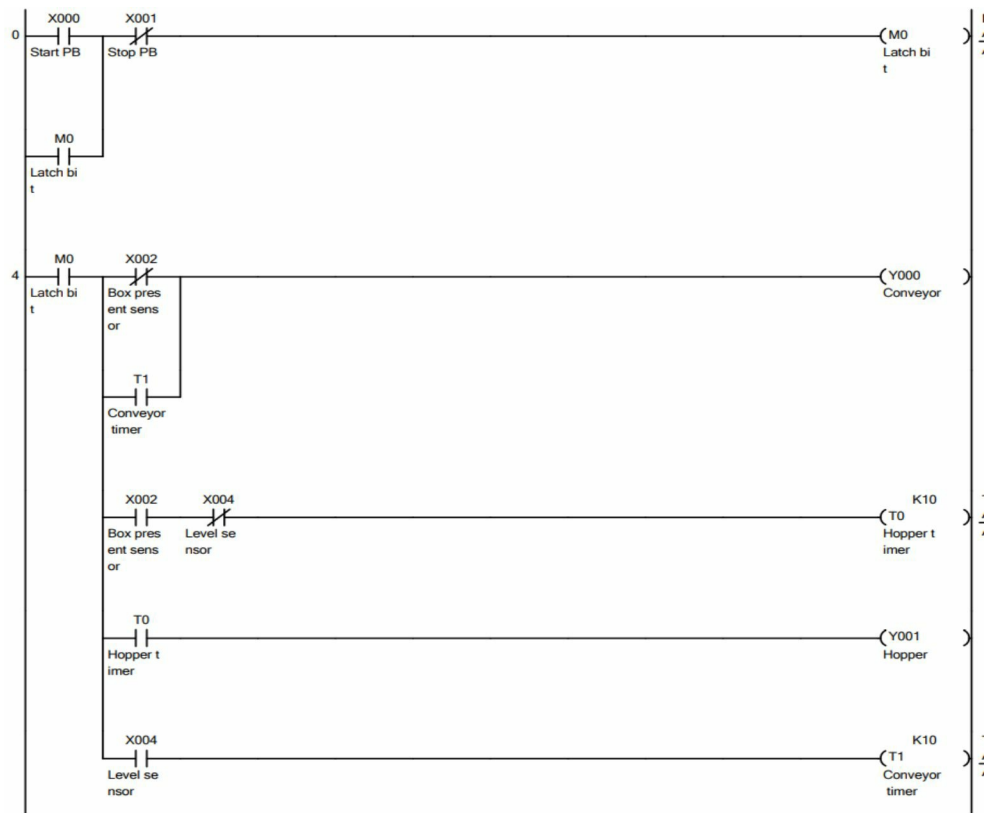
A detects the presence of the box at the filling station and another detects the level of material filled into the box.

- When a start PB is pressed, the system starts.
- When a sensor detects the presence of a box, the controller stops the conveyor.
- After detecting the presence of a box, the hopper's motor will start (1 sec delay) until a level sensor detects the presence of material.
- Once the box gets filled by powder, the controller will start the conveyor after 2 sec.
- This cycle will continue till stop PB is pressed.

### Input and Output devices list

| Input devices      | Output devices |
|--------------------|----------------|
| Start PB           | Conveyor motor |
| Stop PB            | Hopper motor   |
| Box present sensor |                |
|                    |                |

# Level sensor



## 2. Heating application

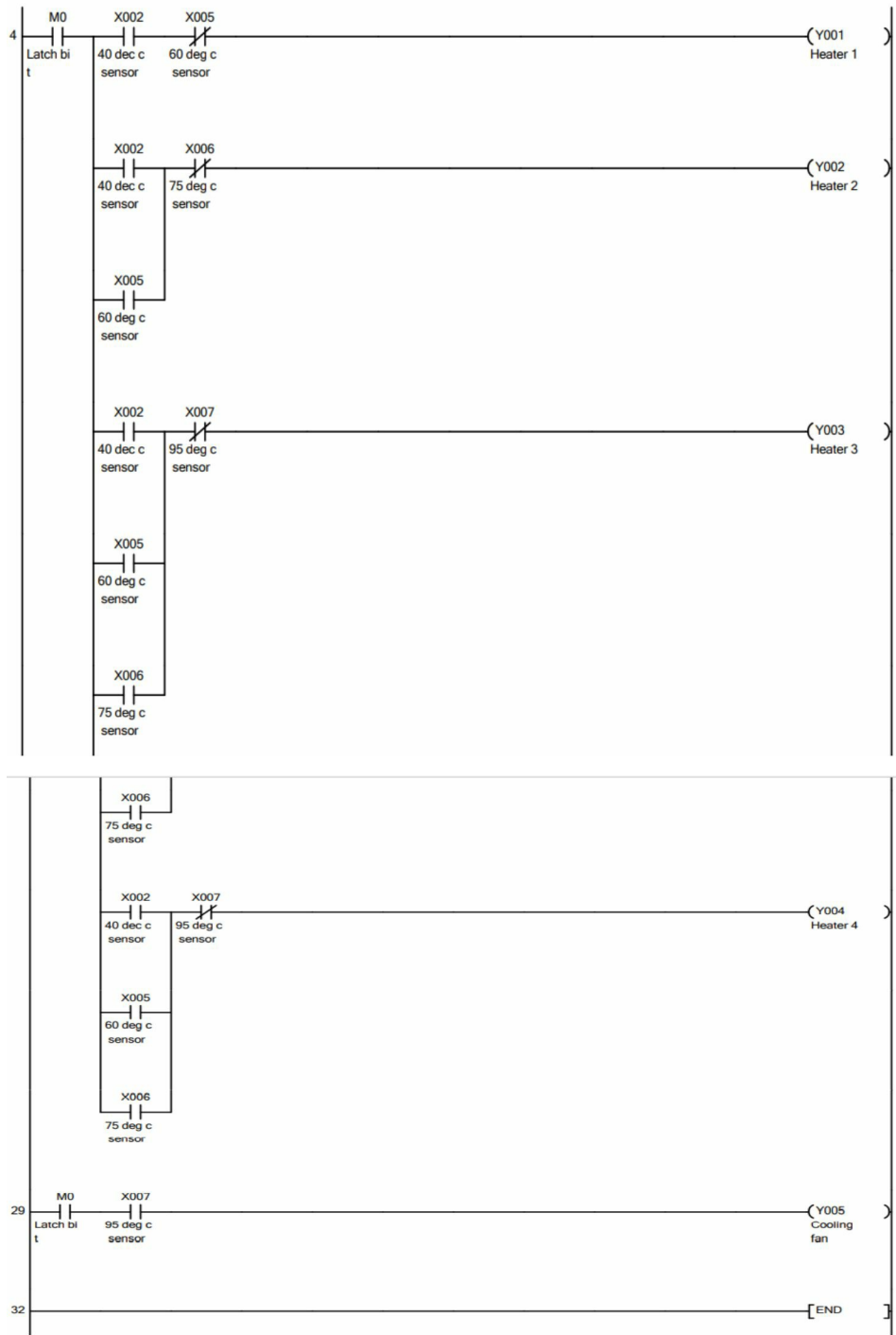
When a start PB is pressed and if the boiler's temperature is less than 40 deg cel, then all heaters (H1,H2,H3, and H4) will turn on.

If temperature is greater than 60 deg cel, then heater 1 gets off.

If temperature is greater than 75 deg cel, then heaters 1 and 2 get off.

If temperature is greater than 95 deg cel, then all heaters get off and switch on the fan.

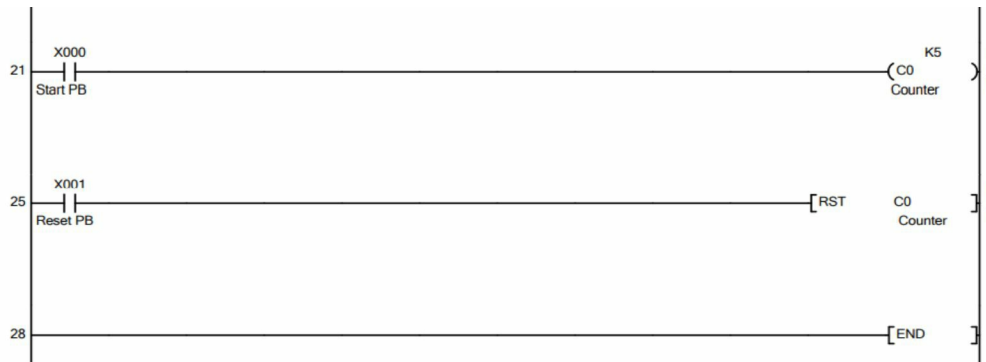
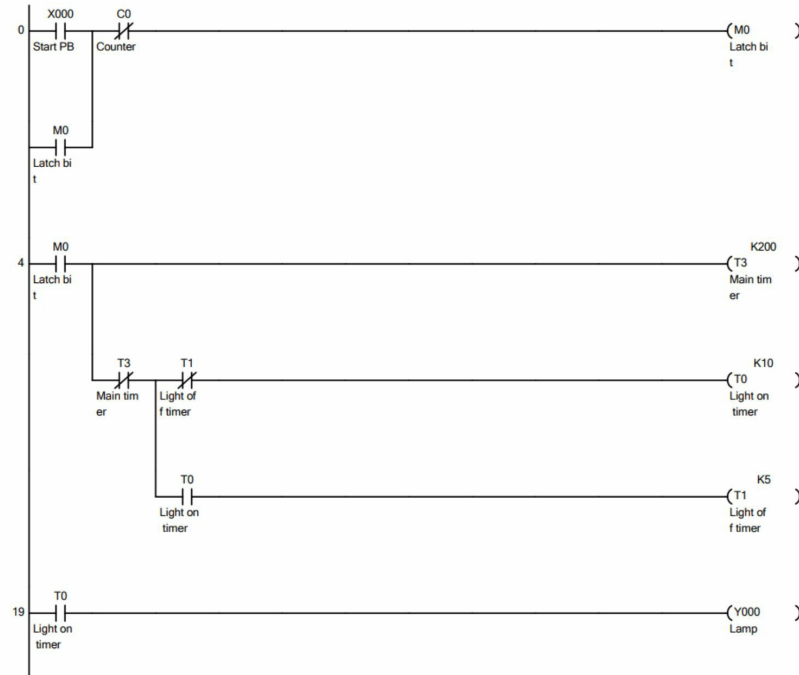




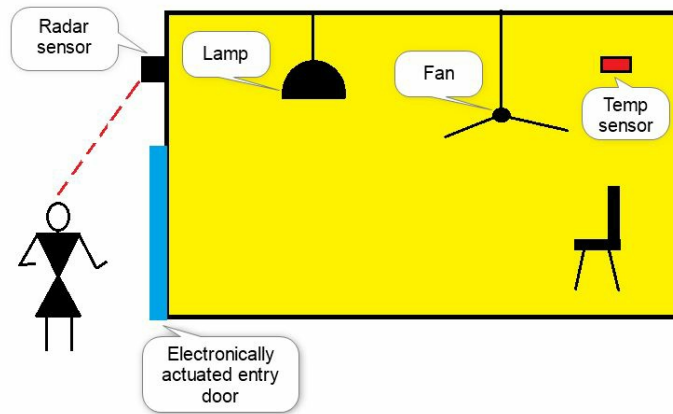
### 3. Flash light

- When a button is pressed, a lamp gets on for 1 sec and gets off for 0.5 sec. The lamp will continue flashing for 20 sec.

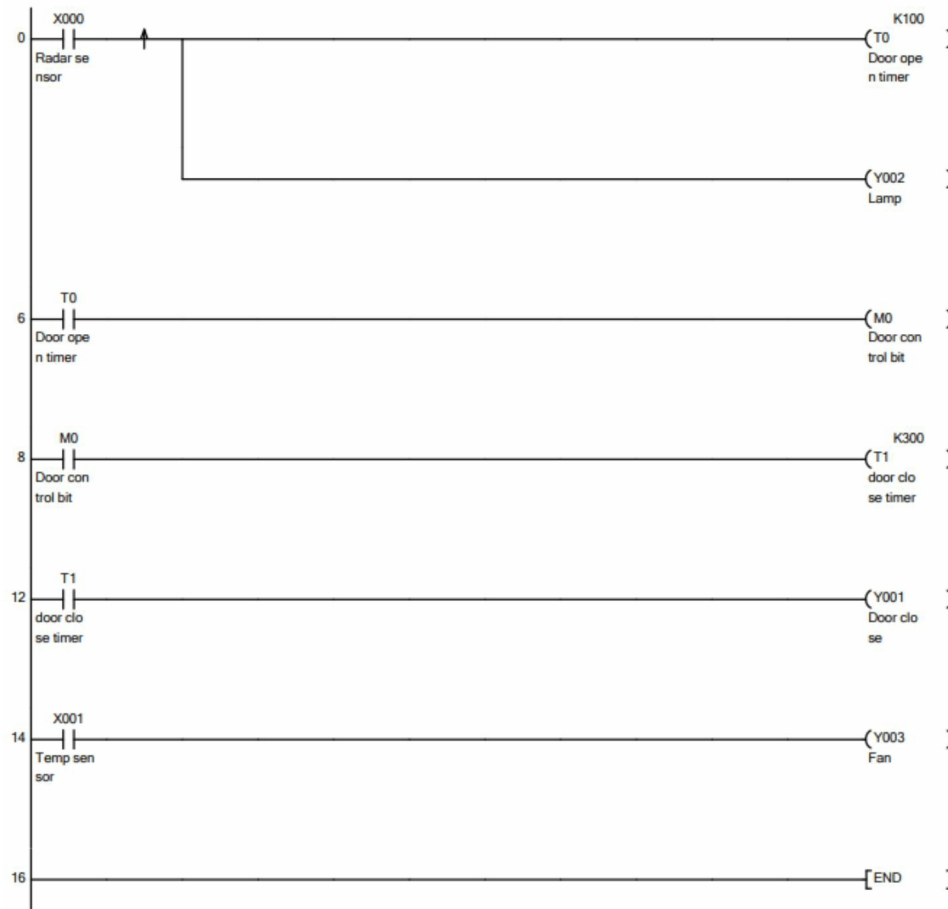
- A lamp will not flash after pressing PB 5 times, it will reset by reset button only.



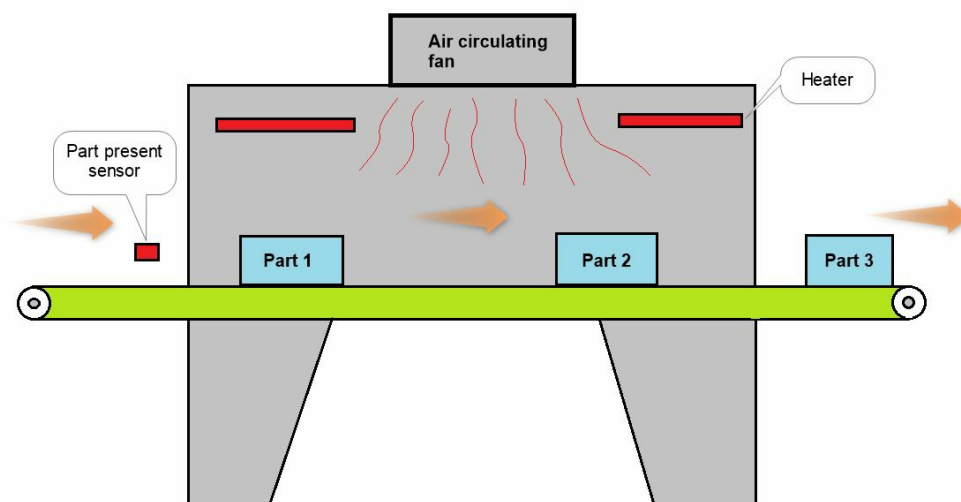
## 4. Home automation



- A controller is used to control the door, lamp, and a fan inside the room. A radar sensor is provided at entry of the room which detects the entry on an object or a human. When it detects anything, it sends a signal to the controller and the door starts sliding to the left side after 1 sec and gets closed after 3 sec.
- As soon as an object enters the room, the lamp will glow.
- If the temperature inside the room is higher than 38 deg cel, a fan will start. A digital temperature sensor is used to detect temperature inside the room, it sends a signal to the controller if the temperature is greater than 65 deg cel.



## 5. Shrink tunnel

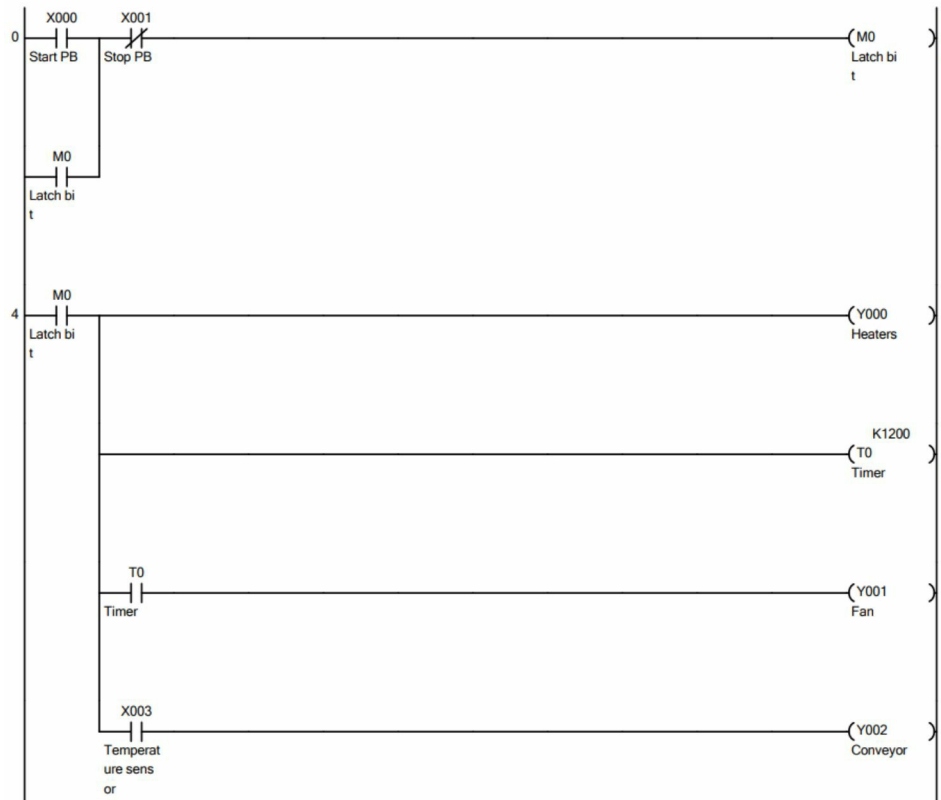


A shrink tunnel consists of a conveyor that carries the parts through the



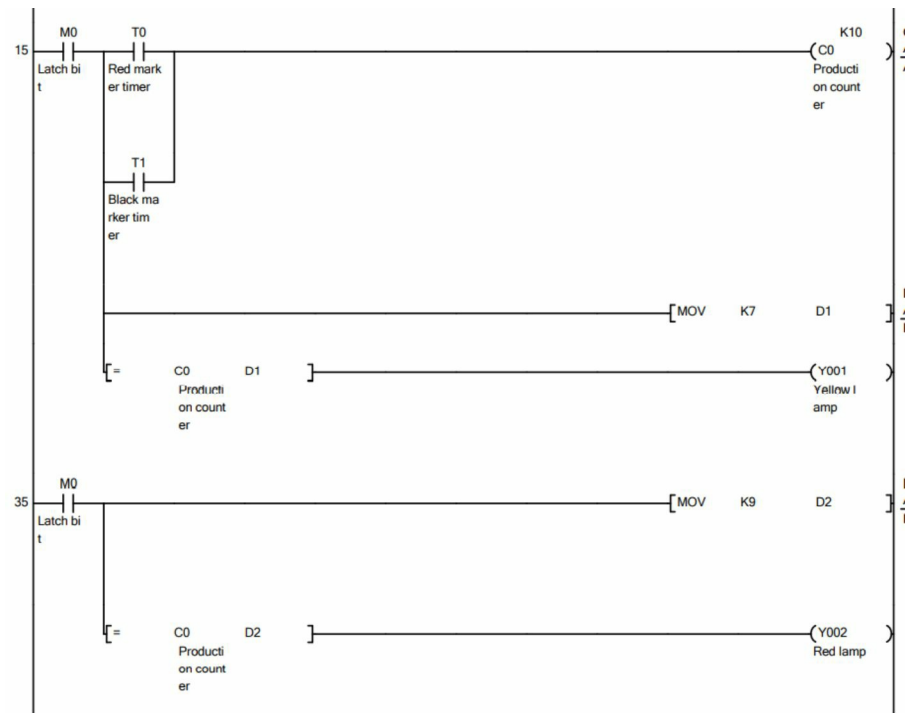
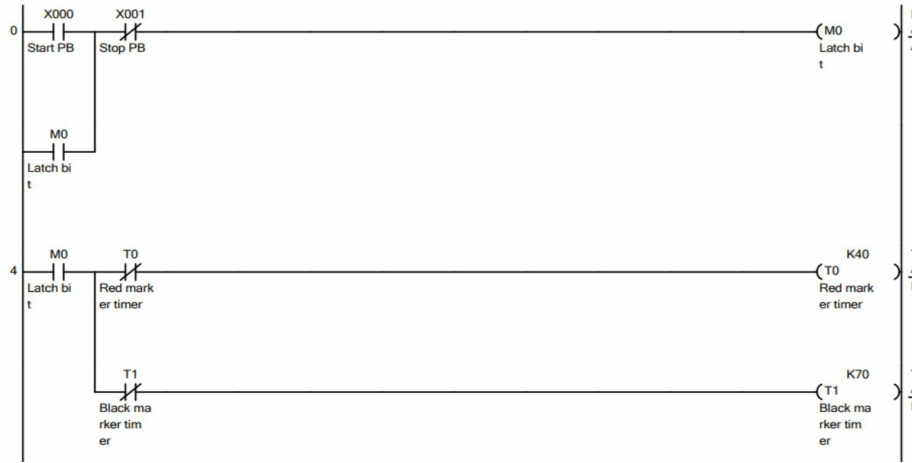
tunnel. A part present sensor is provided at entry of the conveyor which detects the presence of part.

When a start PB is pressed, heaters will start heating and after 2 minutes a fan inside the tunnel will start circulating hot air. When heaters achieve 120<sup>0</sup> C inside the tunnel, a conveyor will start. This cycle will stop after pressing stop PB.



## 6. Production report

When the start PB is pressed after every 4 seconds a red marker is manufactured, and after every 7 seconds a black marker is manufactured. When total markers are manufactured in 7 and 9 quantities then the yellow lamp and red lamp will glow respectively.



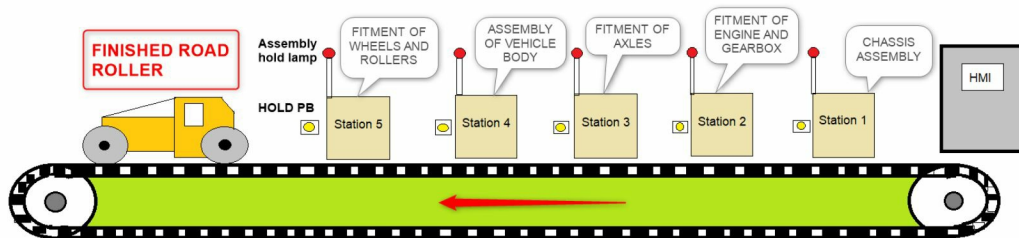
## Case study 3- Road roller assembly line



**Scope of work**- Assembly of road roller.

An assembly line includes fitting of engine on chassis, fitment of axles, mounting of body, and fitment of wheels and rollers. Conveyor is driven by an electric motor coupled with a gearbox. An electric brake is also connected to the motor that ensures braking of the electric motor.

### Assembly line of road roller



### Input and Output devices list

| Input devices            | Output devices                 |
|--------------------------|--------------------------------|
| Start PB (X00)           | Conveyor forward command (Y00) |
| Stop PB (X01)            | Station 1 hold lamp (Y01)      |
| Emergency PB (X02)       | Station 2 hold lamp (Y02)      |
| Station-1 Pause PB (X03) | Station 3 hold lamp (Y03)      |
| Station-2 Pause PB (X04) | Station 4 hold lamp (Y04)      |
| Station-3 Pause PB (X05) | Station 5 hold lamp (Y05)      |

|                          |                             |
|--------------------------|-----------------------------|
| Station-4 Pause PB (X06) | Conveyor motor brake (Y06)  |
| Station-5 Pause PB (X07) | Conveyor stop command (Y07) |

### **Hardware and software details**

PLC - FX 3U 30 M

HMI- MELSEC

PLC & HMI communication- Ethernet

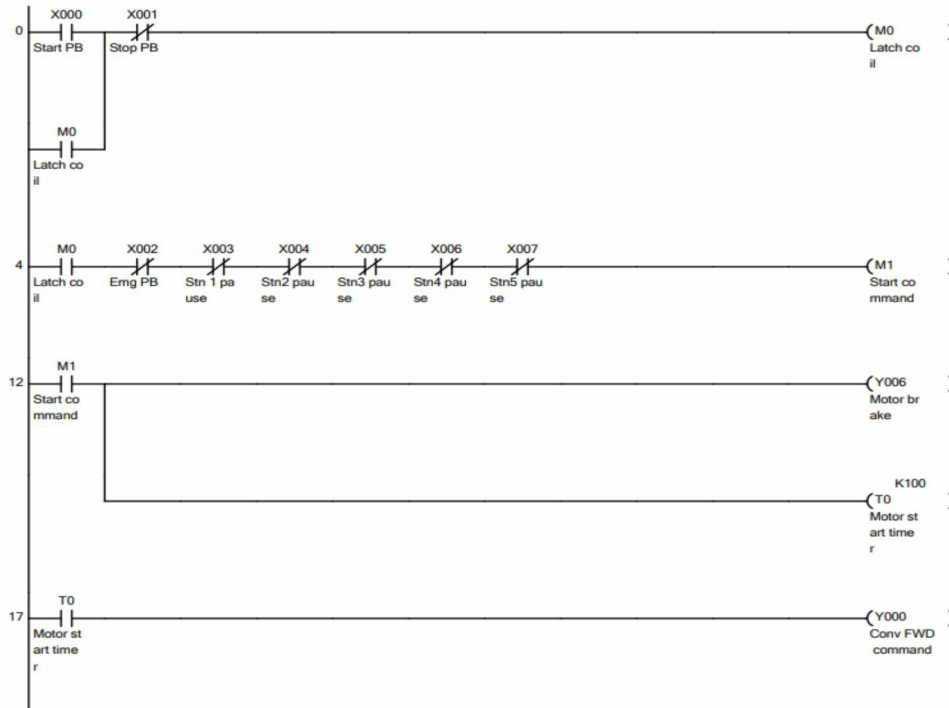
PLC Programming software- Works2

### **Operation**

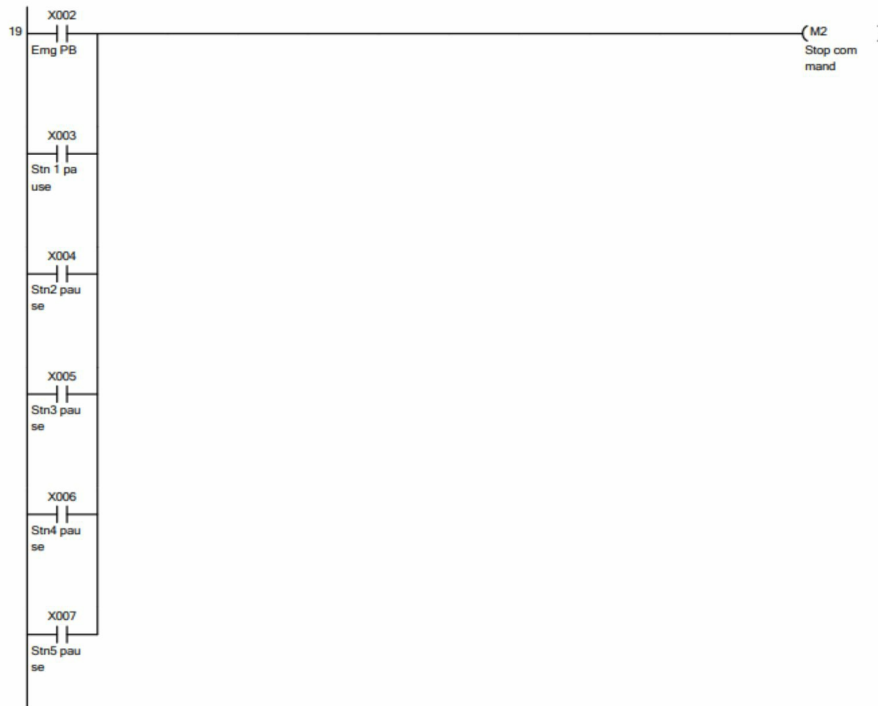
- When a start PB is pressed, a motor brake will be released first and after 1 sec the conveyor forward command will be given to the VFD.
- A conveyor will move in forward direction with 2 mm/1 sec speed.
- Assembly of components is done during forward movement of the conveyor. In case an operator requires more time, he can pause the conveyor and finish his work, and release the pause button so that the conveyor will start again. All 5 stations have been provided with a pause button.
- When a stop PB is pressed, the conveyor will be stopped and after 1 sec a brake will be applied.

### **Programming using works 2 software**

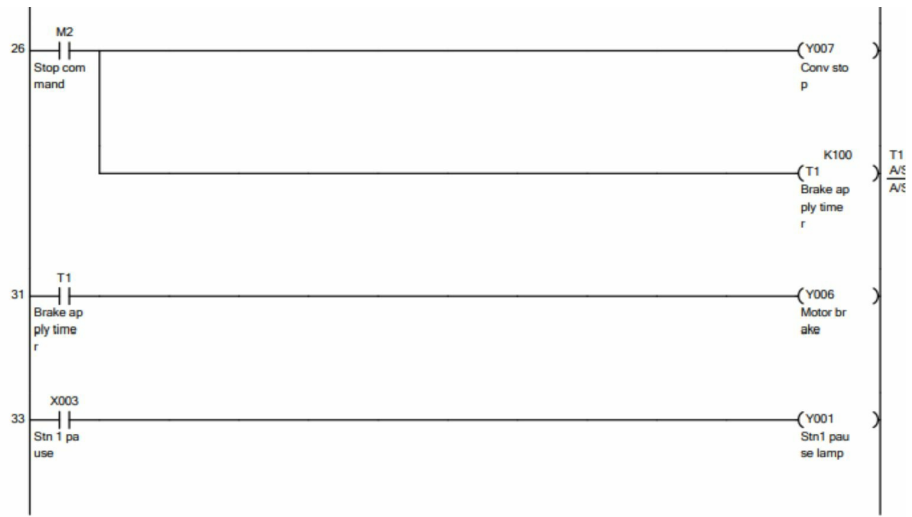
- Program is written as follows. When a start press button is pressed, the memory bit gets latched.
- If one of the pause buttons is pressed, it will unlatch the start command.
- Once the start command is received, it will release the brake first and switch on the motor after 1 sec.



- If one of the push buttons is pressed, it latch the stop command bit first.



- Once the stop command bit gets latched, it will stop the motor first and then apply brake after 1 sec.



- Pause button glows the lamp as shown.

# Assignments

## Assignment 01

When the start PB is pressed light gets on for 10 seconds then gets off for 5 seconds. This cycle will continue until a stop PB is pressed.

## Assignment 02

A pick and place mechanism consists of a pneumatic vacuum gripper, that picks up the pouches when a sensor detects the presence of a pouch and keeps that pouch in a box.

- A conveyor starts when the start PB is pressed. A pouch starts traveling over the conveyor.
- When a sensor detects the pouch, the conveyor stops after 2 sec. Then a slider moves towards the pouch, picks up the pouch, and keeps the pouch into the box.
- As soon as the gripper picks up the pouch, the conveyor starts after 3 sec.

## Assignment 03

- A Saw, Fan, and oil pump all go ON when a start button is pressed.
- If the saw has operated less than 20s, the oil pump should go off when the saw is turned off and the fan is to run for an additional 5s after the shutdown of the saw.
- If the saw has operated for more than 20s, the fan should remain on until reset by a separate fan reset button and the oil pump should remain on for an additional 10 s after the saw is turned off.
- Write a program that will implement this process.

## Assignment 04

- The oven is heated by an electrical heater, and inside there are



ventilation motors to cool the oven after use.

- The electrical heater and the cooling fans should turn on simultaneously. The cooling fans have to turn on too, to circulate the hot air and spread the heat.
- Since both the fans and the heater have to start at the same time, the two outputs should work simultaneously. But keep in mind that the cooling fans have to run for some time after the heater is turned off.

### **Assignment 05**

A classroom has a capacity of a maximum of 120 students. There are two doors, one for Entry and the other for Exit. When the number of students in the classroom is less than 120, the Entry door has a Greenlight on it which remains ON. When the number of students in the classroom is 120 or more than that, the Red light goes ON, turning OFF the Greenlight which indicates that the classroom has reached its maximum capacity and is full.

### **Problem Description**

- Considering the availability of two separate doors for Entry and Exit, two separate Proximity Switches can be used to detect the entry and exit of students.
- One proximity switch is mounted at the Entry door and the other at the Exit door.
- Both the switches will generate two different outputs which can be then fed to PLC to operate the lights according to the Ladder Logic Program written in its memory.
- The counter must be used to count the number of students entering and exiting.
- A comparator must also be used to compare the count value with the given maximum capacity of 120.

### **Assignment 06**

Consider the design of a Burglar Alarm for a house. This alarm will be activated if an unauthorized person is detected by a Window Sensor or a Motion Detector. Implement this alarm System in PLC using Ladder Diagram programming language.

### **Assignment 07**

In a water bottle manufacturing factory, different capacity bottles are manufactured from 1 litre to 20 litre. All bottles with range from 1 litre to 5 litre are collectively checked and moved to section 1 for separation. Similarly bottles from 6 litre to 10 litre moved to section 2 and bottles from 11 litre to 20 litre moved to section 3.

- Check 1 litre to 5 litre bottles and switch on the green lamp.
- Check 10 litre to 20 litre bottles and switch on the yellow lamp.
- If a bottle is found above 20 liters, switch on the red lamp.

### **Assignment 08**

- When a start push button is pressed, the process will start. If the level of water in a reaction tank is low then start the pump till water level reaches high level. When it reaches a high level, stop the pump.
- Now sense the PH value of liquid in the reaction tank, if the PH value is greater than 7, start the acid valve and let the acid enter into the tank. Simultaneously switch on the stirrer till the PH value becomes less than or equal to 4, then stop the stirrer and stop the acid valve. Then start the drain valve.
- Stop the drain valve as soon as the liquid touches the low level of reaction tank and the inlet valve again.

### **Assignment 09**

When the start PB is pressed both conveyors get on simultaneously. First conveyor gets off after 5 sec and the second conveyor gets off after 10 sec. Both conveyors remain off for 10 sec and this cycle continues until a stop button is pressed.

### **Assignment 10**

A ladder logic for the forward and reverse direction of the motor and display pilot lights for the direction of the motor.

### **Assignment 11**

A railway station has 3 platforms A, B and C. A train is coming into the station, it has to be given entry to platform A, if platform A is empty, otherwise to platform B. In case both platforms A and B are occupied, then it has to be given entry to platform C. If all platforms are occupied, then the

train has to wait.

### **Assignment 12**

When a switch is pressed, motor 1 gets started for 10 sec, then motor 2 starts for 8 sec, and motor 3 starts for 7 sec. This process repeats 3 times, then switches off all motors.

### **Assignment 13**

A parking plot has a total capacity of Cars. A number of empty spots are displayed on the display outside the Parking Plot and which spots are available is to be indicated by LEDs. Implement this in PLC using Ladder Diagram programming language.

- The counter is used to count the number of empty spots.
- Proximity Sensors or IR Sensors are used to detect the presence of the car.
- Here in this system, IR Sensors can be well installed to make this system cost-efficient since Proximity Sensors are more costly than IR Sensors.
- Value of the counter is displayed on the display which is mounted outside the parking plot.
- This counter value is converted into decimal.

### **Assignment 14**

- A. When the entered password is OK (111111) and an enter key is pressed, the lock should get opened.
- B. If the entered password is wrong and an enter key is pressed, the lock should not get opened and an alarm will turn on.

### **Assignment 15**

A system consists of a single push button and two lamps.

- When a push button is pressed the first time, lamp1 will glow.
- When a push button is pressed a second time, lamp2 will glow.
- When a push button is pressed a third time, lamp1 will glow.