People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Power and Control

Final Year Project Report Presented in Partial Fulfilment of
the Requirements for the Degree of

MASTER

In **Electrical and Electronic Engineering**

Option: **Control**

Title:

Design and Implementation of Automatic Bottle Filling System Using PLC and SCADA

Presented by:

- **MOHAMMED Abdelmalek**
- **OULD AMARA Mouloud**

Supervisor:

Dr.A.OUADI

Registration Number:……/2016

To our families, and friends

ACKNOWLEDGEMENTS

First of all, we would like to praise almighty ALLAH for facilitating this work and for granting us the opportunity to be surrounded by great and helpful people at IGEE Institute.

Also we would like to express our everlasting gratitude to our supervisor, Dr. Abderrehmane OUADI for his valuable encouragement, guidance and monitoring, without which this work would not have reached the point of fruition, so we ask Allah to reward him on our behalf.

Many thanks are given to all professors and workers at IGEE especially the maintenance laboratory workers for their support and help.

No acknowledgement would be complete without expressing our appreciation and thankfulness to our beloved friends especially Oualid, Oussama , Youcef and Abdessalam for their help and support without forgetting Lathe shop.

At last and not least, our gratitude goes to our fathers, mothers, all our families' members for their unconditional and invaluable support in the entire life, we would never forget their continuous prayer for the sake of our success.

Abstract

This Project objective is to design and Implement a Programmable Logic Controller based automatic bottle filling using a Siemens SIMATIC S7-200 (CPU 226) PLC which acts as a field controller that runs the designed prototype by sensing the presence of bottle in a conveyor belt and then filling it accordingly up to a fixed level. In order to keep track of the overall process in real time, a four layers distributed Supervisory Control And Data Acquisition system was built to monitor and remotely control the bottle filling station over a local area network through an Human Machin Interface designed specifically for the proposed system that indicates the status of different process variables, displays alarms and log events. The SCADA system was deployed using LabVIEW software which makes the system cost efficient, flexible with an open environment and reliable solution for automation of small scale industries.

Table of contents

Chapter 3: SCADA software for automatic bottle filling system

Chapter 4: system implementation and test

List of tables

List of figures

Nomenclature

List of abbreviations

AC: Alternative Current

CAD: Computer Aided Design

CPU: Central Processing unit

DC: Direct Current

DSC: data logging and supervisory control

FBD: Function Block Diagram

GND: Ground

HMI: Human Machin Interface

ICS: Industrial Control System

IEC: International Electrotechnical Commission

IED: Intelligent Electronic Device

IL: Instruction List

LABVIEW: Laboratory Virtual Instrumentation Engineering Workbench

LAN: Local Area Network

LCD: Liquid Crystal Display

LD: Ladder Diagram

NC: Normally Closed

NI: National Instrument

NO: Normally Open

OPC: Object Linking and Embedding for Process Control

PB: Push Button

PLC: Programmable Logic Controller

PPI: point to point interface

PROFIBUS: Process Field bus

RTU: Remote Terminal Unit

SCADA: Supervisory Control and Data Acquisition

SFC: Sequential Flow Chart

ST: Structured Text

TCP/IP: Transmission Control Protocol /Internet Protocol

VI: Virtual instrument

WAN: Wide Area Network

General Introduction

Nowadays automation and control process play a major role in the development of the different industries around the globe for many reasons such as:

Requirement of less human operators, decrease human errors, increase human safety in the field of work, cost efficient, increase production rate, good quality of the products, flexibility and convertibility in manufacturing processes, etc. All these advantages made more and more companies switch to automation.

The development of Supervisory Control and Data Acquisition (SCADA) systems made the automation process more reliable and efficient because it does not only control the process but also monitor with a user friendly HMI screen and make history of the process which is used to analyze its performance.

Unfortunately, in our country many industries are still not automated and even for those who are automated they do not have SCADA system implemented in their manufactures due to their high cost and lack of local providers for this technology.

One of the important applications of automation which is used in the Food and beverage as well as pharmaceutical industries is the automatic bottle filling system, where a particular liquid has to be filled continuously.

Therefore our objective is to design and implement a PLC based automatic bottle filling system and to implement a low cost SCADA technology for this system in order to cover and investigate the procedures used to make a fully automated process and the main components of SCADA technology through the understanding of the bottle filling application.

This project report is organized as follows:

Chapter one: in this first chapter we cover the major elements that compromise a SCADA system and its different architectures used in automated industries.

Chapter two: gives an overview of the bottle filing system and discusses the working principle and the main components needed to make this system work.

Chapter three: throughout this chapter we explain the phases of building the SCADA software and merging it to hardware

Chapter four: In this chapter a description of all the hardware construction including the mechanical and electrical design and the software used to implement and test the system is presented.

Chapter One

Automation Using SCADA

1.1 Introduction

Industrial Automation is the use of control systems to control industrial machinery and processes, reducing the need for human intervention and increasing efficiency. Control systems are used by many infrastructures and industries to monitor and control sensitive processes and physical functions.

A primary type of modern Industrial Control Systems (ICS) is supervisory control and data acquisition (SCADA) systems, this chapter considers the main components of SCADA, its architecture, communication infrastructure and different modules, in order to best select a SCADA technologies for the proposed automatic bottle filling system.

1.2 Definition and Working Concept

A SCADA at it most basic is an automation control system that is used for controlling, monitoring and analyzing a large variety of industrial processes, some examples include energy, oil and gas, water , transportation, manufacturing , food and beverage and many more. SCADA system starts by communicating in real time via channels with field controllers that are either PLCs or RTUs which are running the actual process to collect information, once the real time data acquisition is complete, this data is presented using HMI to the operators who are running the process, this allows the operators to monitor in real time what the process is doing, react to alarms, control the process remotely and change settings.

The SCADA system will often include a historian product, so these real time process information can be stored and tracked over the long term, which allows predictions of future events based on past events and efficient diagnosis of the process faults.

1.3 SCADA Major Components

While these systems simplify a given infrastructure, their components are quite complex. Not all SCADA systems are the same, but they can be broken down as shown in **Figure1.1** (next page), into the following 4 layers that are present in every system in one form or another [1], which are data acquisition, conversion, communication and data presentation and control .

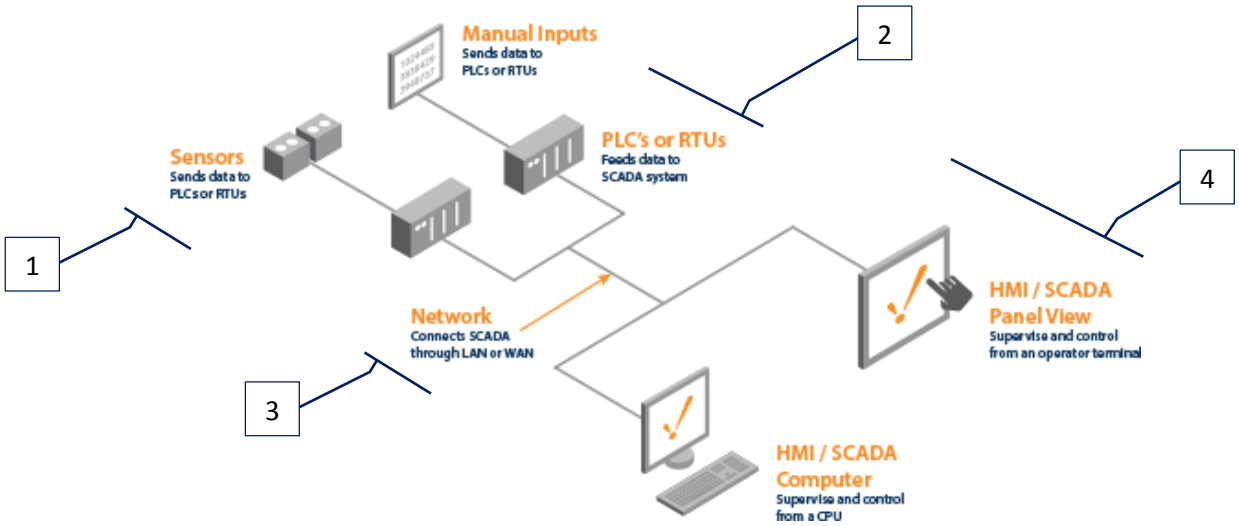


Figure 1.1. Basic SCADA System components [2]

Each component has a well-defined function or purpose. Furthermore, each component has a specific relationship with the components that it communicates with. SCADA systems can be broken down into following major layers, which form a chain. Each component communicates with the component before and after itself.

1.3.1 Data Acquisition

The first component in the chain is data acquisition. It is not preceded by another component, but it connects to the data conversion component. Data acquisition consists of:

- **Sensors** detect and transmit readings of important physical parameters to the PLCs, such as photo sensors, pressure sensors, level sensors and flow sensors.
- **End devices** or Actuators which include equipment from valves, motors to large machinery like pumps and turbines that are controlled by the PLCs to start, stop and function as required.

Depending on the type of SCADA system these devices could be physically located hundreds of miles away from each other or could be inside the same plant. Data acquisition is also known as input output or I/O.

1.3.2 Data Conversion

Data conversion receives and collect data generated by the acquisition component from the field process. Devices that fall under this category are:

1.3.2.1 Intelligent Electronic Devices

An IED, as it relates to the protection and power system automation industry, is a device that performs electrical protection functions, advanced local control intelligence, has the ability to monitor processes and can communicate directly to a SCADA system [3]. IEDs receive data from sensors and power equipment, and can issue control commands, such as tripping circuit breakers if they sense voltage, current, or frequency anomalies, or raise/lower voltage levels in order to maintain the desired level. A typical IED can contain around 5-12 protection functions, 5-8 control functions controlling separate devices, an auto reclose function, self-monitoring function, communication functions etc. Hence, they are aptly named as Intelligent Electronic Devices, an example of an IED is shown on **Figure 1.2**

Figure 1.2. Protective relay is an example of an Intelligent Electronic Device [8]

1.3.2.2 Remote Terminal Unit

An RTU (sometimes referred to as a remote telemetry unit) as the title implies, is a standalone data acquisition and control unit, generally microprocessor based, which monitors and controls equipment at some remote location from the central station. Its primary task is to control and acquire data from process equipment at the remote location and to transfer this data back to a central station [4]. It generally also has the facility for having its configuration and control programs dynamically downloaded from some central station. There is also a facility to be configured locally by some RTU programming unit.

Although traditionally the RTU communicates back to some central station, it is also possible to communicate on a peer-to-peer basis with other RTUs. The RTU can also act as a relay station (sometimes referred to as a store and forward station) to another RTU, which may not be accessible from the central station.

Small sized RTUs generally have less than 10 to 20 analog and digital signals, medium sized RTUs have 100 digital and 30 to 40 analog inputs. RTUs, having a capacity greater than this can be classified as large. A typical RTU configuration is shown in **Figure 1.3** bellow. RTU are often equipped with wireless radio interfaces to support remote situations where wire-based communications are unavailable.

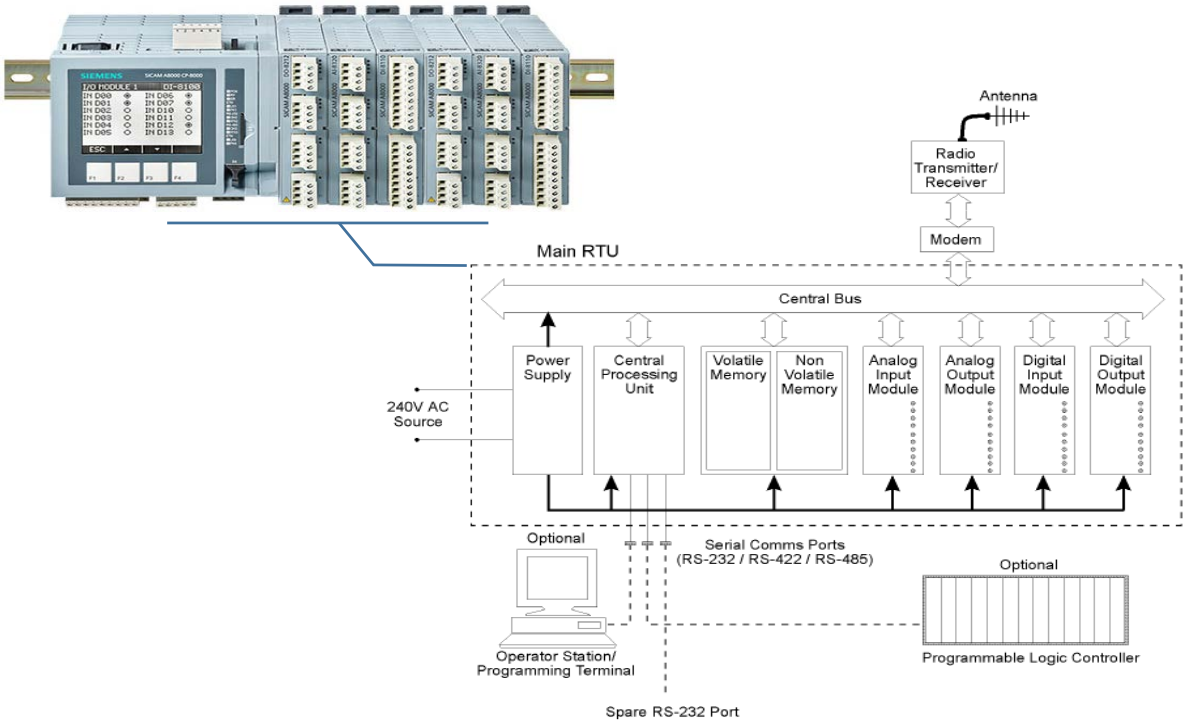


Figure 1.3. Typical RTU hardware structure [4]

1.3.2.3 Programmable Logic Controller

An introduction to programmable logic controller, its general function, hardware form and the programming languages used to program it are presented in details in the subsequent bulleted list.

- **Definition**

(PLC) is a special form of microprocessor-based controller that uses programmable memory to store instructions and to implement functions such as on/off control, sequencing, timing, counting, and arithmetic in order to control machines and processes [5].

- **Hardware of PLC system**

Typically a PLC system has the basic functional components, as shown in **Figure 1.4** below. They compromise of:

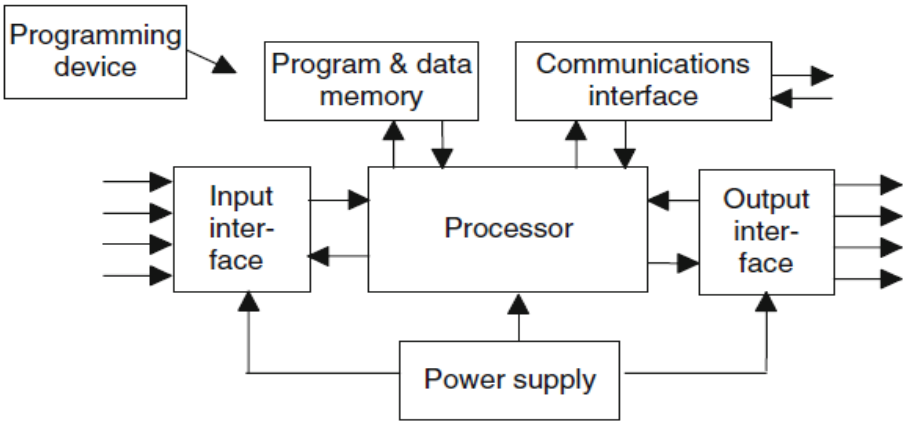


Figure 1.4. The functional components of a PLC [5]

i. The processor unit or central processing unit (CPU) is the unit containing the microprocessor and this interprets the input signals and carries out the control actions, according to the program stored in its memory, communicating the decisions as action signals to the outputs [5].

ii. The power supply unit is needed to convert the mains A.C voltage to the low D.C voltage (5 V) necessary for the processor and the circuits in the input and output interface modules [5].

iii. The programming device is used to enter the required program into the memory of the processor. The program is developed in the device and then transferred to the memory unit of the PLC [5].

iv. The memory unit is where the program is stored that is to be used for the control actions to be exercised by the microprocessor and data stored from the input for processing and for the output for outputting [5].

v. The input and output (I/O) sections are where the processor receives information from external devices and communicates information to external devices. The inputs might thus be from switches, sensors such as photo-electric cells, temperature sensors, or level sensors, etc. The outputs might be to motor starter coils, solenoid valves, etc. [5]

vi. The communications interface is used to receive and transmit data on communication networks from or to other remote PLCs. It is concerned with such actions as device verification, data acquisition, synchronization between user applications and connection management [5].

- **Programming PLCs**

Programs for use with PLCs can be written in a number of formats. Most PLC manufacturers adopted ladder-logic for writing programs. However, each tended to develop its own versions and so an international standard has been adopted for ladder programming and indeed all the methods used for programming PLCs. The standard, published in 1993, is International Electrotechnical Commission IEC 61131-3 include 5 programming languages as shown in **Figure 1.5** below:

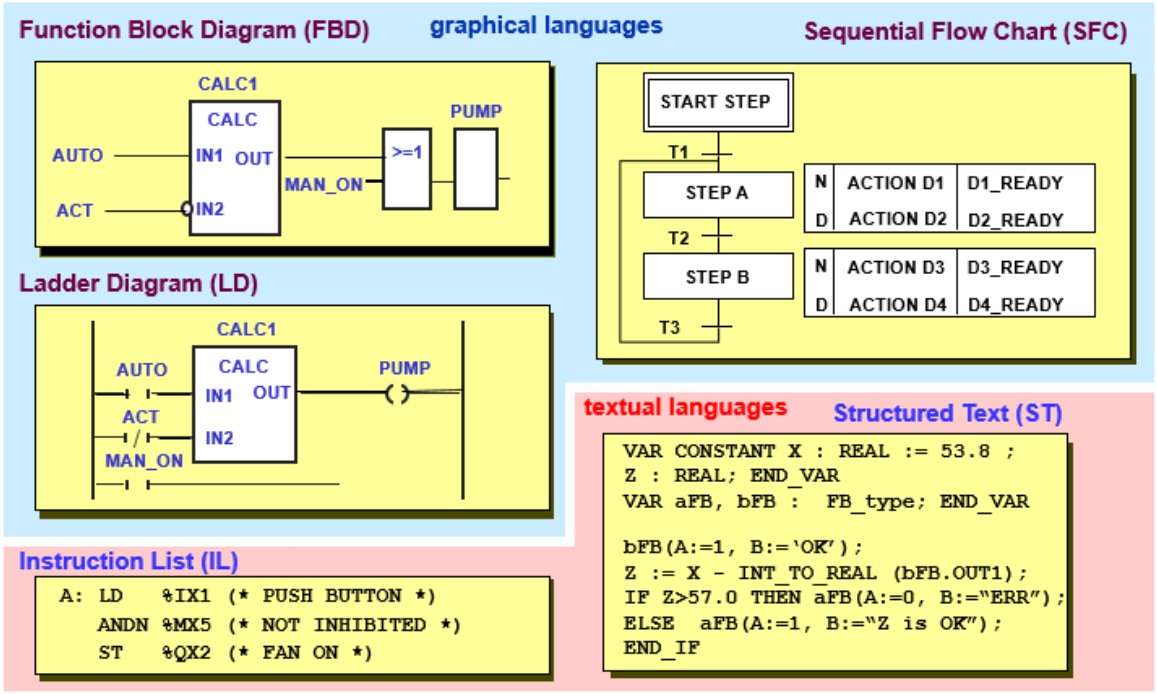


Figure 1.5. The five IEC 61131-3 Programming languages [10]

i. Instruction List IL is a convenient assembler-like programming language. IL is universally usable and is often employed as a common intermediate language to which the other textual and graphical languages are translated [6].

ii. Structured Text ST is a textual language of IEC 61131-3. ST is called a High-Level Language, because it does not use low level, machine-oriented operators but offers a large range of abstract statements describing complex functionality in a very compressed way [6].

iii. Function Block Diagram (FBD) is a graphical language similar to Structured Analysis. Controllers are modeled as signal and data flows through processing elements

(*function blocks*). FBD transforms textual programming (ST) into connecting (already defined) building blocks, thus improving modularity and software reuse [6].

iv. Ladder Diagrams (LD) is an evolution of electrical wiring diagrams. LDs supply a programming style borrowed from electronic and electrical circuits. This programming language is primarily designed for processing Boolean signals (1 ≡ TRUE or 0 ≡ FALSE).

v. Sequential Function Chart (SFC) is primarily a graphical language, although a textual description is also defined. SFC was defined to break down a complex program into smaller manageable units and to describe the control flow between these units. Using SFC, it is possible to design sequential and parallel processes.

- **The programming software**

PLC manufacturers have programming software for their PLCs. For example, Mitsubishi has MELSOFT. The company's GX Developer supports all MELSEC controllers, from the compact PLCs of the MELSEC FX series to the modular PLCs, including the MELSEC System Q, and uses a Windows-based environment [5].

As another illustration, Siemens have SIMATIC STEP 7. This fully complies with the international standard IEC 61131-3 for PLC programming languages. Likewise, Rockwell Automation has RSLogix for the Allen-Bradley PLC-5 family of PLCs, OMRON has CX-One and Telemecanique have ProWorx 32 for its Modicon range of PLCs [5].

Modern PLCs can even be compared to desktop PCs in regards to their power and functionality. Data conversion has a two way communication with data presentation and control via the data communication component.

1.3.3 Data Communication

Data communication consists of some communication medium that transfers data back and forth between data conversion and data control. The communication medium could be wired, wireless, radio, satellite or others. The communication takes place using one of the many SCADA protocols [1] , a protocol controls the message format common to all devices on a network. Common Protocols are designed to be very compact. Many are designed to send information only when the master station polls the PLC or RTU. Typical legacy SCADA protocols include Modbus, Foundation Fieldbus and Profibus.These communication protocols are all SCADA-vendor specific but are widely adopted and used. Standard protocols are IEC 60870-5-101 or 104, IEC 61850 and DNP3 [7].

These communication protocols are standardized and recognized by all major SCADA vendors. Many of these protocols now contain extensions to operate over TCP/IP. Recently, OLE for process control (OPC) has become a widely accepted solution for intercommunicating different hardware and software, allowing communication even between devices originally not intended to be part of an industrial network.

1.3.4 Data Presentation and Control

As the name suggests data presentation and control consists of devices used to monitor and control data received from various data communication channels. It may include:

1.3.4.1 Human Machine Interface

HMI also referred to as User Interface, Operator Panel or Terminal is the input-output device through which the human operator controls, monitors the process, which is presented graphically, in the form of a mimic diagram. An example is an operator panel which allows an industrial machine operator to interact with a machine in a graphical, visual way. With controls and read-outs graphically displayed on the screen, the operator can use either external buttons or the touch screen to control the machinery. Ranging from simple segmented displays to high-resolution LCD panels, HMIs can be located on the machine, in battery-operated, portable handheld devices, and also in centralized control rooms.

HMI is usually linked to the SCADA system's databases and software programs, to provide trending, diagnostic data and visual alarms.

1.3.4.2 An I/O Server

An input output data acquisition server is a software service which uses industrial protocols to connect software services, via telemetry, with field devices such as RTUs and PLCs. It allows clients to access data from these field devices using standard protocols.

1.3.4.3 A Historian

A historian is a database service which accumulates time-stamped data, boolean events, and boolean alarms in a database which can be queried or used to populate graphic trends in the HMI. The historian is a client that requests data from a data acquisition server [4].

1.3.4.4 Supervisory station

The term supervisory station refers to the servers and software responsible for communicating with the field equipment (RTUs, PLCs, SENSORS etc.), and then to the HMI software running on workstations in the control room, or elsewhere. In smaller SCADA systems, the master station may be composed of a single PC. In larger SCADA systems, the master station may include multiple servers, distributed software applications, and disaster recovery sites [8].

1.4 SCADA architectures

SCADA systems have grown in parallel with the development and sophistication of modern computing technology. In following section a description of the three generations of SCADA systems would be covered.

1.4.1 Monolithic SCADA Systems

When SCADA systems were first developed, the concept of computing in general centered on “mainframe” systems. Networks were generally non-existent, and each centralized system stood alone. As a result, SCADA systems were standalone systems with virtually no connectivity to other systems [9]. Connectivity to the SCADA master station itself was very limited by the system vendor. Connections to the master typically were done at the bus level via a proprietary adapter or controller plugged into the CPU backplane. **Figure 1.6** (next page) shows a typical first generation SCADA architecture.

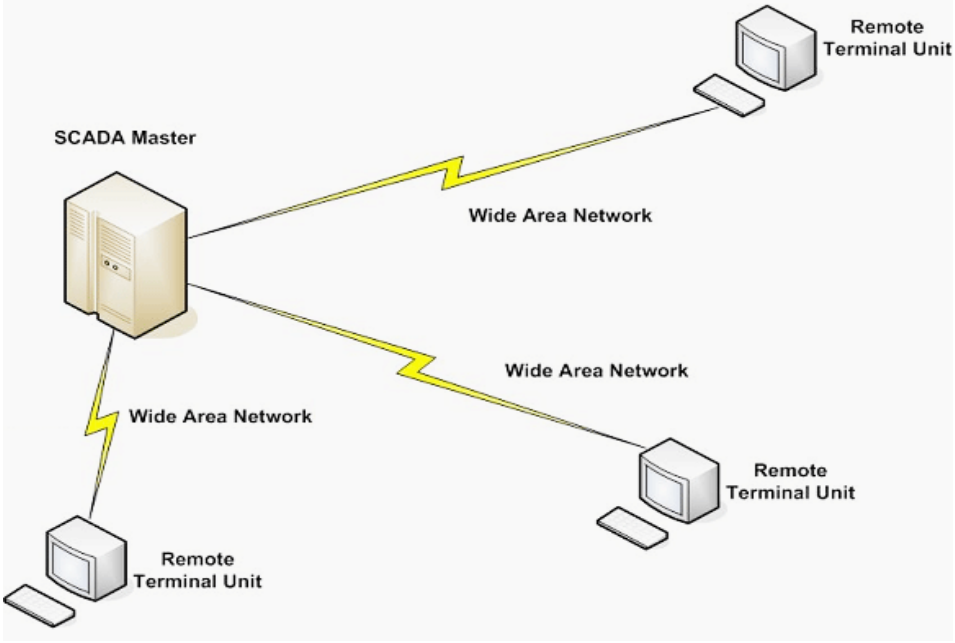


Figure 1.6. First Generation SCADA Architecture [9]

The standby system's primary function was to monitor the primary and take over in the event of a detected failure. This type of standby operation meant that little or no processing was done on the standby system [9].

1.4.2 Distributed SCADA Systems

The next generation of SCADA systems took advantage of developments and improvement in system miniaturization and Local Area Networking (LAN) technology to distribute the processing across multiple systems. Multiple stations, each with a specific function, were connected to a LAN and shared information with each other in real-time.

These stations were typically of the mini-computer class, smaller and less expensive than their first generation processors [10].

The distribution of individual SCADA system functions across multiple systems provided more processing power for the system as a whole than would have been available in a single processor. The networks that connected these individual systems were generally based on LAN protocols as shown in **Figure 1.7** (next page) and were not capable of reaching beyond the local environment and these networks were still provided or at least selected by the vendor.

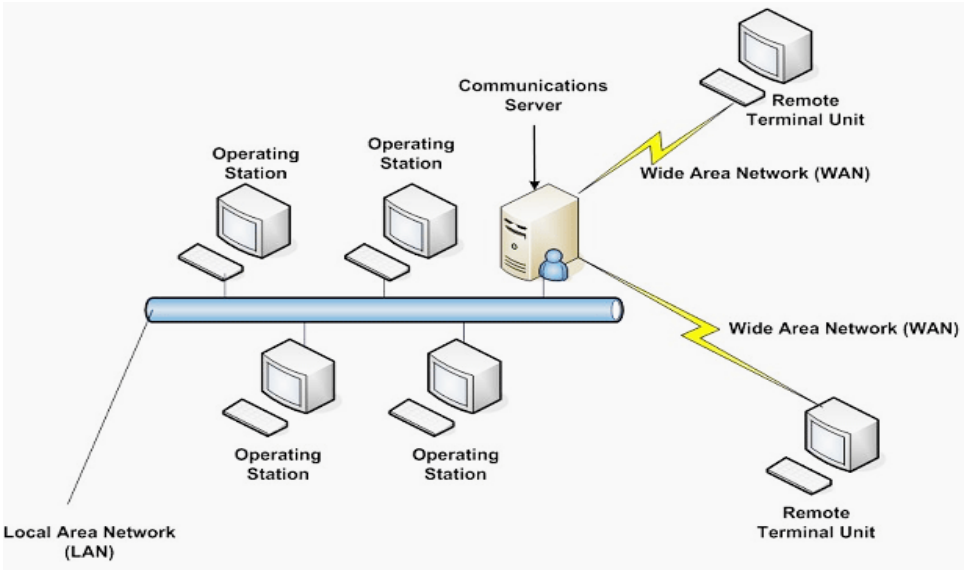


Figure 1.7. Second Generation SCADA Architecture [9]

1.4.3 Networked SCADA Systems

The current generation of SCADA master station architecture is closely related to that of the second generation, with the primary difference being that of an open system architecture rather than a vendor controlled, proprietary environment.

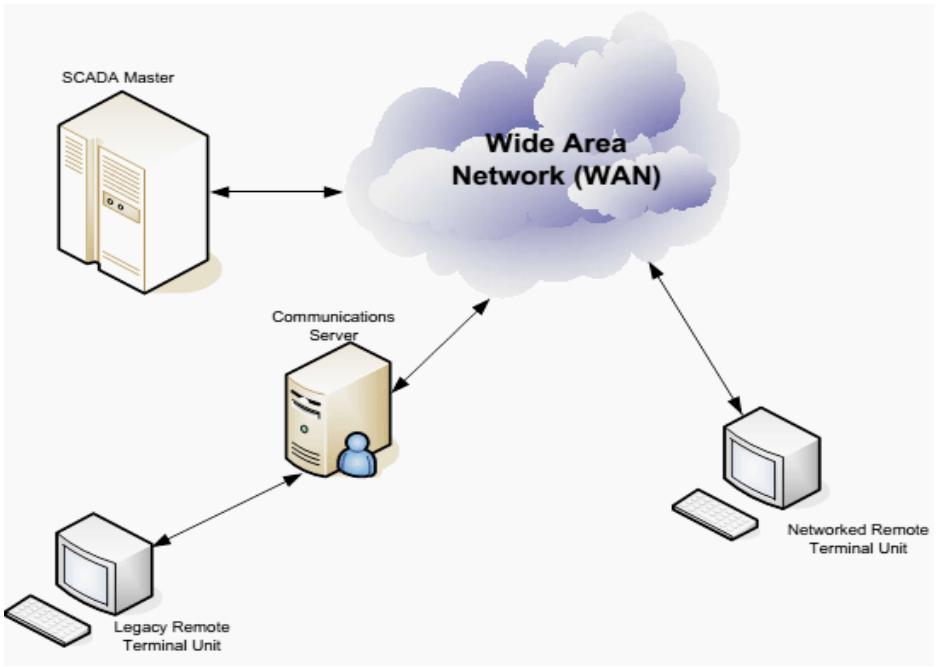


Figure 1.8. Third Generation SCADA System [9]

Another advantage brought about by the distribution of SCADA functionality over a WAN as show in **Figure 1.8** (previous page) is that of disaster survivability and improvement of the overall reliability of the system.

In this project the design and implementation of an automatic filling system using Siemens S7-200 PLC as a field controller, SIMATIC STEP 7 Micro/WIN as a software development tool and National Instrument LabVIEW software to simulate and deploy a four layer distributed SCADA system over local area network.

1.5 Conclusion

This chapter introduced an overview of SCADA as part of modern ICS and described the core elements of SCADA systems and its different architectures with considerations of their features and limitations.

Chapter Two

Automatic Bottle Filling System

2.1 Introduction

Bottle filling system is an automatic process used in many food and beverage industries where empty bottles are first placed on the conveyor belt. The conveyor moves these bottles to the filling station then it stops to let the bottle fill-up with a fixed amount of liquid. The liquid is kept at an overhead tank near the filling station. After the bottle is filled-up, the belt starts again to move away the filled-up bottle and to bring another empty bottle to the filling station.

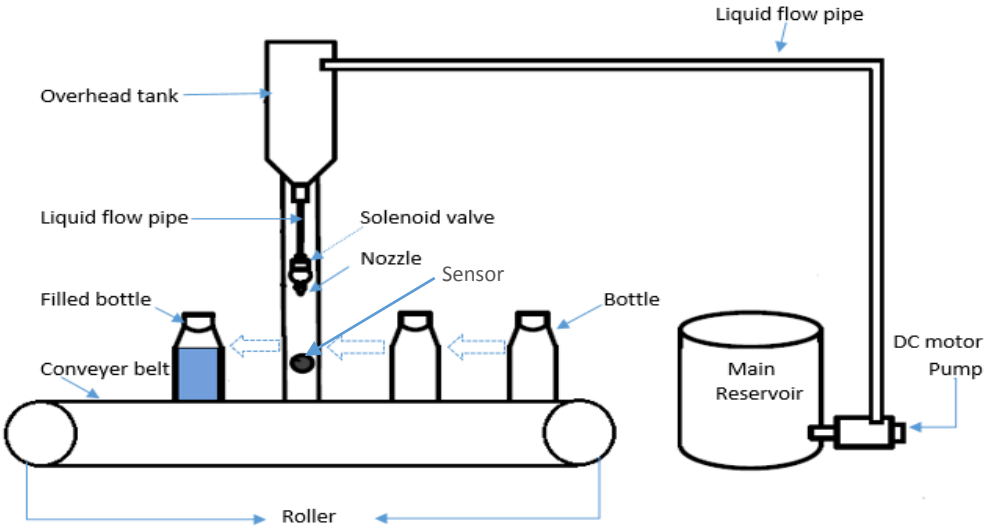


Figure 2.1. Block diagram of automatic liquid filling system

2.2 Working principle

The design of the system can be divided into several parts:

2.2.1 Conveyor Belt Mechanism

It presents a continually moving surface that is designed to move objects from one location to another. Different types of conveying machine are available regarding principle of operation, means and direction of conveyance, like vibrating screw conveyors, pneumatic conveyors and the moving floor system. The one we used is roller conveyor system. It consists of a belt and rollers with DC gear motor. It is coupled with rollers and runs continuously with the support of a DC gear motor when the system is started the conveyor belt carries the bottle along the prouduction line to be filled up with liquid up to a particular level and then the conveyor belt runs again to take the filled up bottle to the other end to be collected.

2.2.2 Bottle Detection Mechanism

The task of bottle detection is performed using a photoelectric sensor. A photoelectric sensor is placed on the side of the conveyor belt at the filling station to detect the presence of a bottle. The sensor has an infrared light transmitter which transmit light. This light is reflected back to the receiver by a reflector which is placed on the other side of the conveyor, when a bottle is brought in front of the sensor by the conveyor belt, the infrared light emitted from the emitter does not get reflected back hence the receiver does not receive any signal. Based on these two conditions the PLC will give command to the conveyor motor to run or to stop.

2.2.3 Liquid Flow Control Mechanism from Overhead Tank to the Bottle

When the bottle is detected by the photoelectric sensor, the task of filling the bottle with liquid starts. An electrovalve is used to control the flow of liquid from overhead tank to the bottle. The electrovalve is positioned at the filling station. When a bottle stops underneath the valve, it gets a command from the PLC to open the valve and liquid flows from the overhead tank to the bottle up to a particular level.

2.2.4 Liquid Level Maintaining Mechanism at the Overhead Tank

It is crucial to maintain a minimum level of liquid at the overhead tank to continue the smooth operation of the bottle filling task. On the other hand, while refilling the overhead tank with liquid, care should be taken so that it does not overflow. Therefore, the option for sensing the minimum and maximum liquid levels should be provided.

2.3 Hardware components of the system

The hardware components used in this project are listed below:

2.3.1 PLC Module (Siemens S7-200)

The PLC S7-200(**Figure 2.2)** is used to implement the control logic required to monitor and control the input and output devices of the system, The S7-200 CPU combines a microprocessor, an integrated power supply, input circuits, and output circuits in a compact housing to create a powerful Micro PLC.

Siemens provides different S7-200 CPU models with a diversity of features and capabilities that help us create effective solutions for our varied applications. The CPU 226 is the one used in this project. The features of the CPU used are introduced in Appendix A.[5]

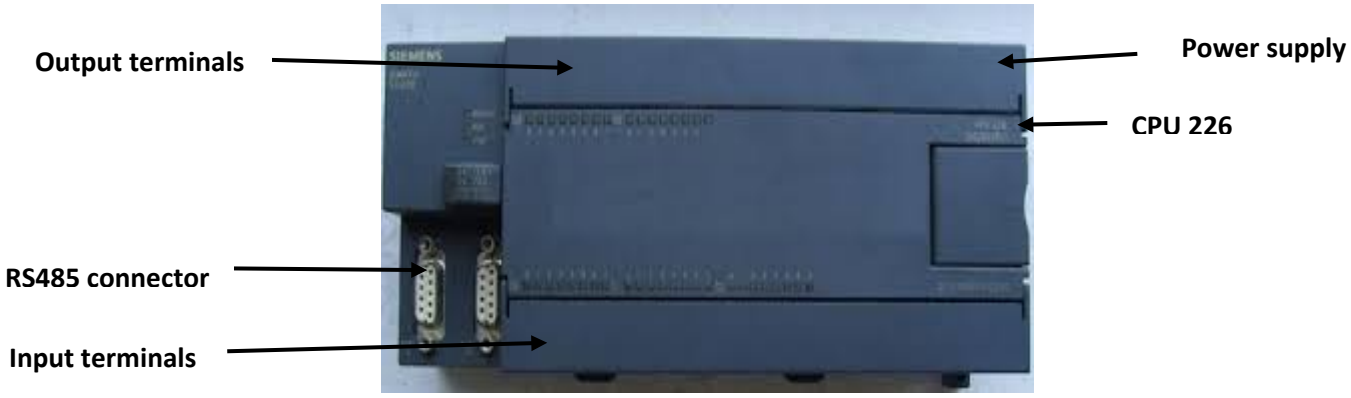
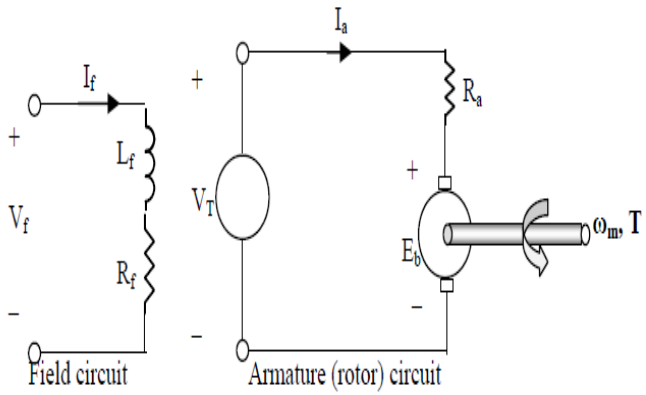


Figure 2.2. Siemens S7-200 (CPU 226) [5].

2.3.2 Motors

A motor is an electrical machine that can transform an electrical energy into a mechanical one (**motor mode**) or transform a mechanical energy into an electrical one [20] (**generator mode**).Two types of motors are used in this project.

The equivalent circuit (Field and armature circuit) and component of DC motor are shown in **Figures 2.3** and **2.4** respectively



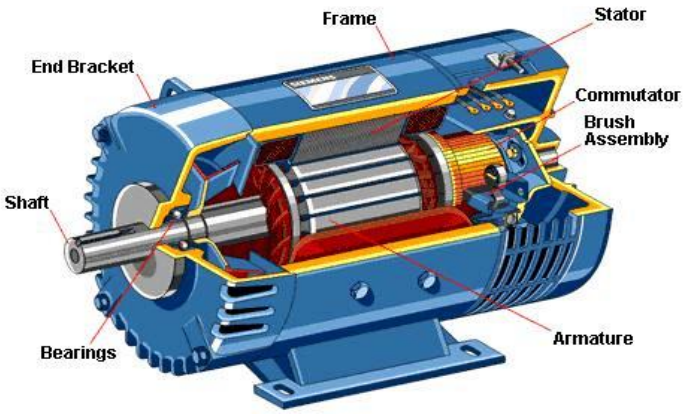


Fig 2.3. Equivalent circuit of DC motor [20]

Fig 2.4. DC motor components [18]

2.3.2.1 DC gear motor

A DC gear motor is a type of DC motor that has a gear assembly attached to the motor. As a result, the motor can run at high torque and carry heavy loads.

The DC gear motor used in this project is produced by motion king company holding the reference number 37ZYJ-36ZY (**Figure 2.5)** which has the following specification:

- Reduction ratio is 1/30
- 200 rpm at 12 V
- 2.5 kg/cm
- No load current <=350 m
- Loaded current <=1300mA

Since this motor cannot operate at 24 V supplied by the PLC an external power circuit is required.

Figure 2.5. DC gear motor [14]

2.3.2.2 DC motor pump

It is DC motor which has pump assembly attached to it to pump liquids as shown in **Figure 2.6** (next page)

It is used to pump the liquid from the main reservoir to the overhead tank. The motor used in this pump is rated 12V DC and draws up to 3 A current this amount of current can not be supplied by the PLC therefore an external power circuit is required.

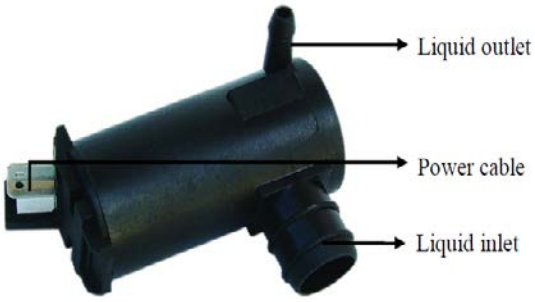


Figure 2.6. DC motor pump

2.3.3 Electrovalve

A valve is a device that controls the flow of liquid. It can be opened and closed to control the flow of fluid. Valves can be hydraulic or electrical. In a hydraulic valve, the valve opens or closes depending on the liquid or air pressure. In electrical valves, an electric signal is used to open or close the valve. The electrovalve shown in **Figure 2.7** is used in this project.

When it receives an electrical signal, the valve is opened, otherwise it is left closed.

It operates at 220V AC as for the motor and the pump an external power circuit is required.

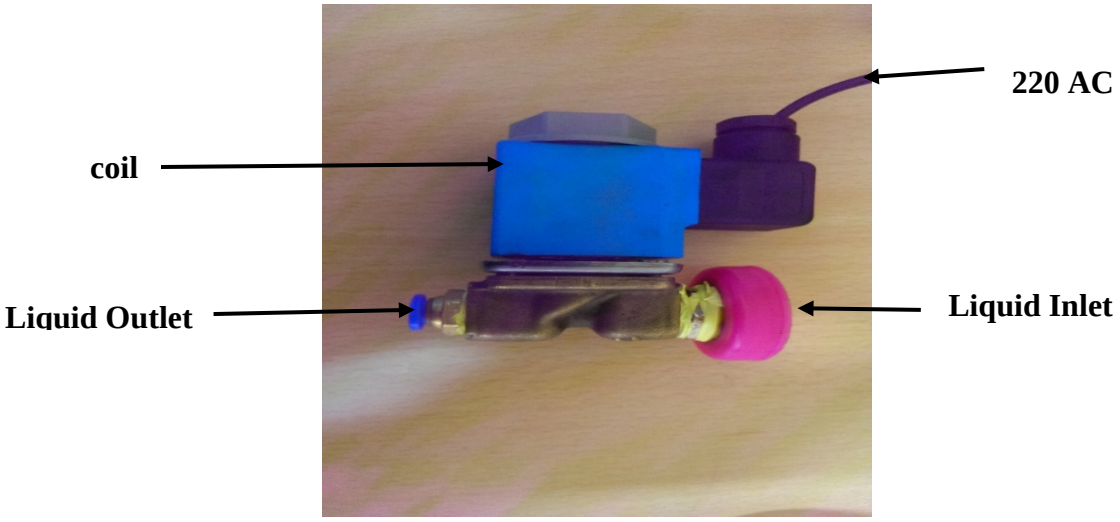


Figure 2.7. Electrovalve

2.3.4 Reservoirs

2.3.4.1 Overhead tank

A bottle is used to simulate overhead tank. It can hold up to 1.25 Liters of water.it supplies the liquid to the electrovalve which is connected to it. The electrical liquid level sensing system is incorporated in the overhead tank. The system allows maintaining the level of liquid in the overhead tank which is very important to continue bottle filling operation.

2.3.4.2 Main reservoir

A ten liter plastic pail is used as a main reservoir in our project. It is situated at the ground. It has a DC motor pump attached to it. The DC motor pump extracts liquid from main reservoir to overhead tank through pipelining system.

2.3.5 Sensors

2.3.5.1 Photoelectric sensor

Figure 2.8 shows the photoelectric sensor and its reflector used in our project.

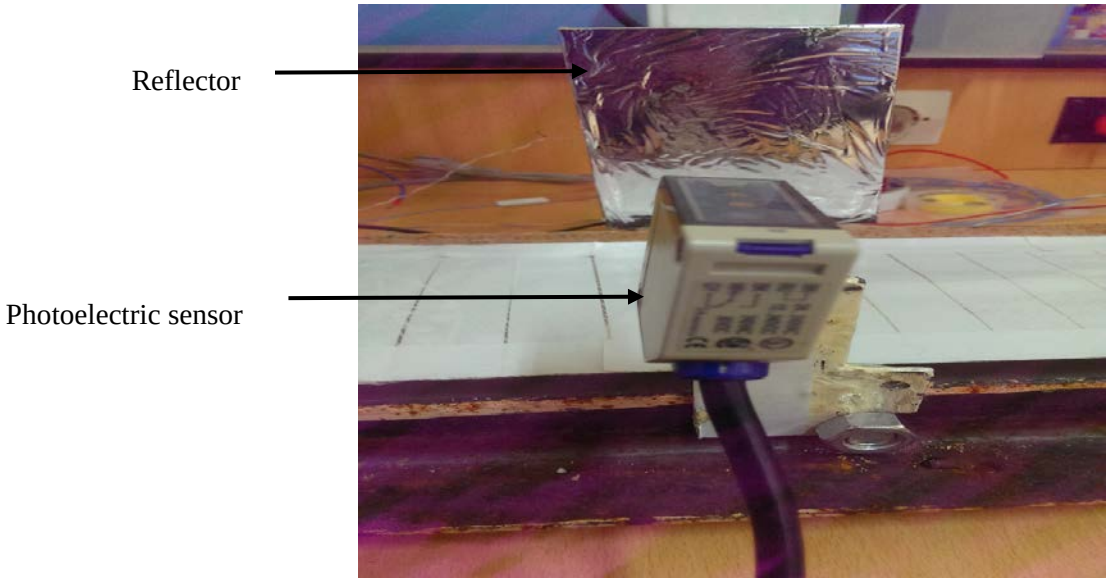


Figure 2.8. Photoelectric sensor and its reflector

A photoelectric sensor is a device used to detect the absence or presence of an object using infrared transmitter and photoelectric receiver. They are largely used in industrial manufacturing.

There are three different types: opposed (through beam), retro-reflective, and proximity sensor (diffused).Our sensor is of type retro-reflective.

A retro-reflective arrangement places the transmitter and receiver at the same location and uses a reflector to bounce the light beam back from the transmitter to the receiver. An object is sensed when the beam is interrupted and fails to reach the receiver.

2.3.5.2 Liquid level sensor

For sensing the level of liquid in the overhead tank, we made a sensor ourselves. It uses the electricity conducting property of liquids. We used a nap to make four nodes to sense the level of liquid, (a) the upper level node, (b) the middle level node, (c) the lower level node (d) the common node. The common node is connected to one end of a 24 V DC supply. These sensor are wired in such a way that, when the liquid in the tank is in contact with the common node and any other (upper/middle/lower level) sensors, an electrical circuit is completed. This condition is detected by the PLC to understand whether the overhead tank is low in liquid, or has sufficient liquid.

2.3.6 S7-200 RS-485/PPI Multi-Master Cable

The S7-200 RS-232/PPI Multi-Master Cable shown in **Figure 2.9** is programming device cable which is used to transfer the program from a computer to the S7-200 PLC. It comes factory set for optimal performance with the STEP 7--Micro/WIN 3.2 programming package. The factory setting for this cable is different than for the PC/PPI cables.

We can configure the S7-200 RS-485/PPI Multi-Master Cable to operate the same as the PC/PPI cable and to be compatible with any version of a STEP 7--Micro/WIN programming package by setting Switch 5 to the PPI/Freeport setting and then selecting our required baud rate. [16]

Figure 2.9. S7-200 RS485/PPI Multi Cable

2.3.7 Relay

The operating voltages and currents of our actuators (electrovalve, pump, and DC gear motor) are different than the outputs of the PLC so Relays are used to make the output of the PLC drive the actuators through external power circuit.

A relay is an electromagnetic switch operated by a relatively small electric current that can turn ON or OFF a much larger electric current. The heart of a relay is an electromagnet (a coil of wire that becomes a temporary magnet when electricity flows through it). Relays are used where it is necessary to control a circuit by a different signal with complete electrical isolation between control and controlled circuits.

G2R-1-E Omron relays (**Figure 2.10.)** are used in this project which has the following specifications:

- **Description:** General purpose power relay.
- **Rated coil voltage:** 24VDC / 110VAC.
- **Rated switching current:** 16A at 250VAC / 16A at 30VDC.[19]

Figure 2.10. OmronG2R-1-E Relay [19]

2.4 Conclusion

In this chapter the working principle of an automatic bottle filling system is described and all the hardware components are discussed.

Chapter Three

SCADA Software for Automatic Bottle Filling System

3.1 Introduction

As mentioned in the previous chapters the design and implementation of the proposed control process requires both hardware and software. This chapter discusses the role of SCADA software's used in this project and how they interact on different levels.

3.2 SIMATIC STEP 7 Micro/WIN

Step 7 Micro/WIN is an automation software provided by Siemens Industry for automation solutions (www.industry.siemens.com). This software package can be handled like a standard Windows application and includes all necessary tools for programming of the SIMATIC S7-200, from the high-performance SIMATIC instruction set to IEC 611131-3 compliant programming and all the way to trend charts and wizards.

3.2.1 Merging Hardware and Software

Using the STEP 7 software, the program can be created within a project, then download it to the PLC and setting it in run mode to control the process with the S7 program. The I/O modules are addressed in the S7 program via the addresses.

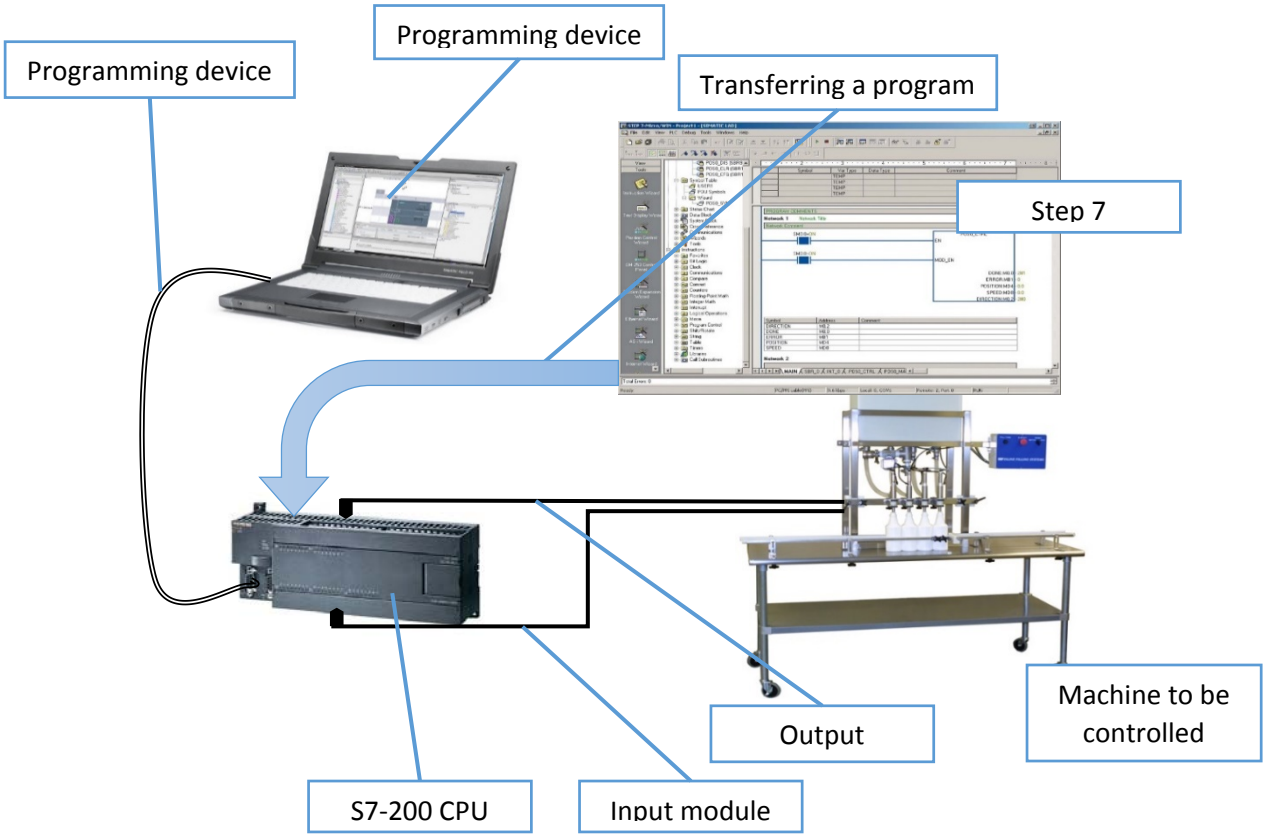


Figure 3.1. PLC connected with programing device and controlled machine

3.3 NI LabVIEW

NI LabVIEW stands for Laboratory Virtual Instrumentation Engineering Workbench provided by National Instruments (www.ni.com) is a development environment designed specifically to accelerate the productivity of engineers and scientists in a wide area of industries with a graphical programming syntax that makes it simple to visualize, create, and code engineering systems.

One great benefit of graphical programming (called G) in LABVIEW is that application development for prototyping takes much less time and because the easy and intuitive design that executes by 'Data Flow', rather than the more traditional procedural approach, which gives an advantage to programmers with different background in engineering than IT to debug their programs using a full suite of debugging tools. The built in compiler of LabVIEW as well as a linker and the fact that the flow of data between nodes regulates program execution rather than a prioritized list, that makes this style of programming different, and extremely powerful for the purposes that it is used for.

Basic LabVIEW program or as they are called virtual instruments (VIs).consists of two windows, the front panel and the block diagram.

3.3.1 LabVIEW Front Panel

In LabVIEW, you build a user interface by using a set of tools and objects. The user interface is known as the front panel. As shown in the **Figure.3.2** bellow.

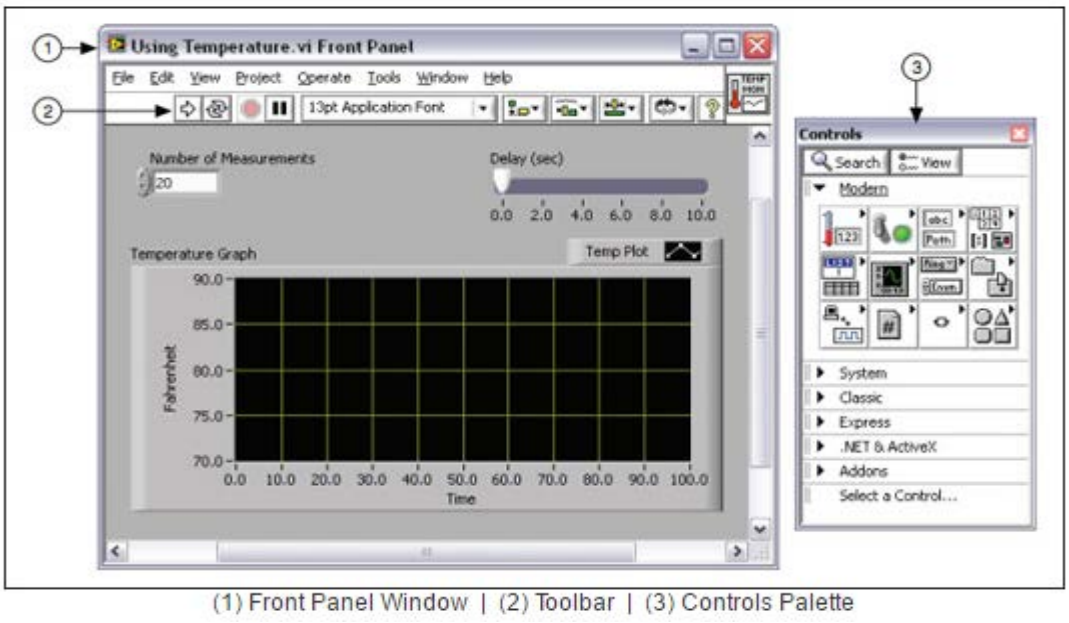


Figure 3.2. Example of a Front Panel [21]

Users interact with the Front Panel when the program is running. Users can control the program, change inputs, and see data updated in real time. Controls are used for inputs such as, adjusting a slide control to set an alarm value, turning a switch on or off, or to stop a program.

3.3.2 LabVIEW Block Diagram

The block diagram contains the graphical source code of a LabVIEW program. The concept of the block diagram is to separate the graphical source code from the user interface in a logical and simple manner. Front panel objects appear as terminals on the block diagram. Terminals on the block diagram reflect the changes made to their corresponding front panel objects and vice versa.An example is shown in **Figure3.3** below.

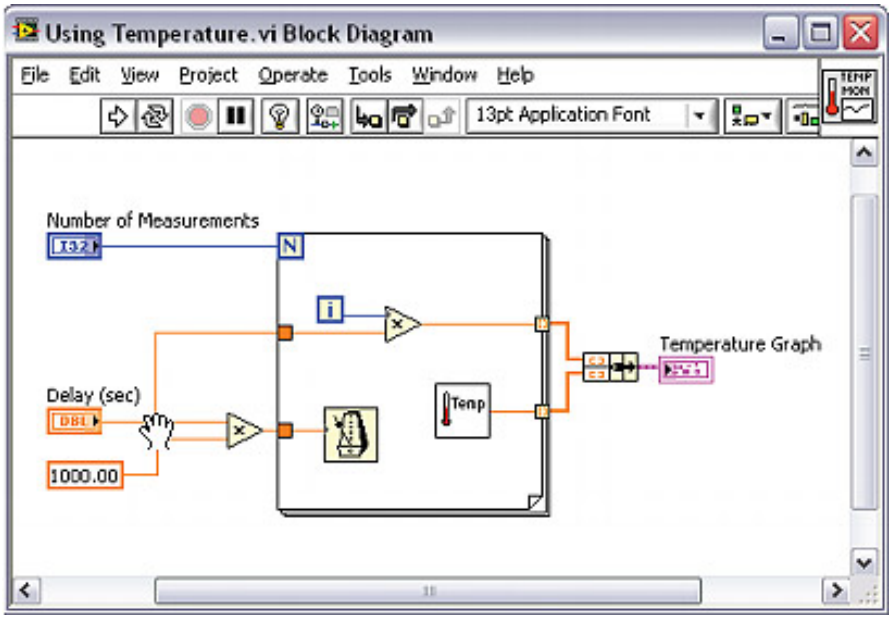


Figure 3.3. Example of a Block Diagram [21]

LABVIEW program also contains a three pallets which give the options that are need to create and edit the front panel and block diagram, these pallets include:

- **Tools Palette**

The Tools palette is available on the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. When a tool is selected the cursor icon changes to the tool icon. The tools are used to operate and modify front panel and block diagram objects.

- **Controls Palette**

The Controls palette is available only on the front panel. The Controls palette contains the controls and indicators that are used to create the front panel.

- **Functions Palette**

The Functions palette is available only on the block diagram. The Functions palette contains the VIs and functions that are used to build the block diagram. The **Figure 3.4** below shows the three types of pallets:

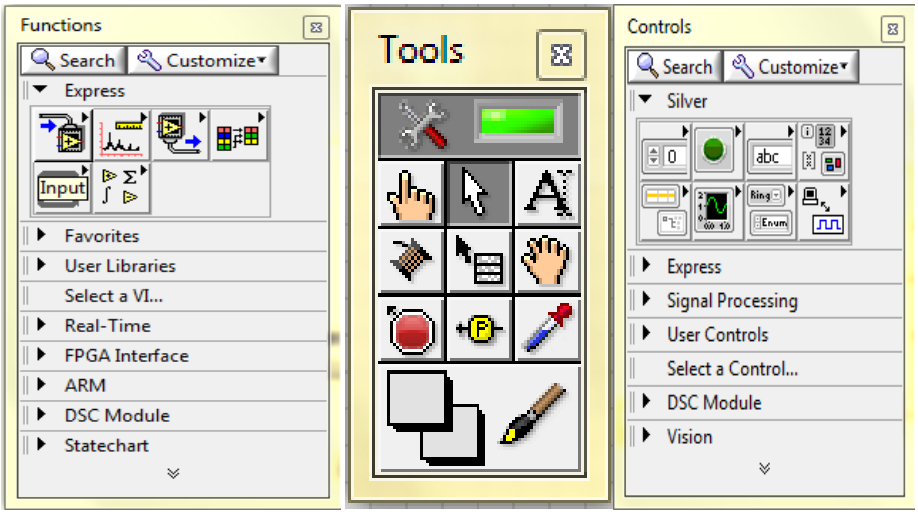


Figure 3.4. Function, Tools and Controls pallets. [21]

3.4 Developing HMI/SCADA System in LabVIEW

In the past, engineers developing industrial systems had to learn different software tools to program embedded controllers and HMI/SCADA applications even when buying everything from the same vendor.

With National Instruments LabVIEW graphical system design software, it is now possible to program human machine interface (HMI) and logic in the same environment, minimizing development cost, training time, and maximum skill re-use. Specifically, the use of the LabVIEW data logging and supervisory control (DSC) module to develop a full-fledged SCADA application.

3.4.1 Overview of LabVIEW DSC Module

Applications today require open SCADA environments with OLE for process control (OPC), Modbus connectivity, and the flexibility of a programming language, so that high-speed measurements and analysis are added to existing systems. The LabVIEW DSC Module provides configuration-based data logging, alarming, scaling, and security for developing the Windows-based HMI/SCADA system. A read or write operations are easily made on OPC connections, PLCs, or custom I/O servers that are created by the use of DSC Module that provides solutions for supervisory control of a wide variety of distributed systems.

3.4.2 Steps of Building a SCADA system

In order to increase efficiency and make troubleshooting easy if any problems occur a break down structure approach to the deployment of the SCADA system into different phases as show in **Figure 3.5** below was taken.

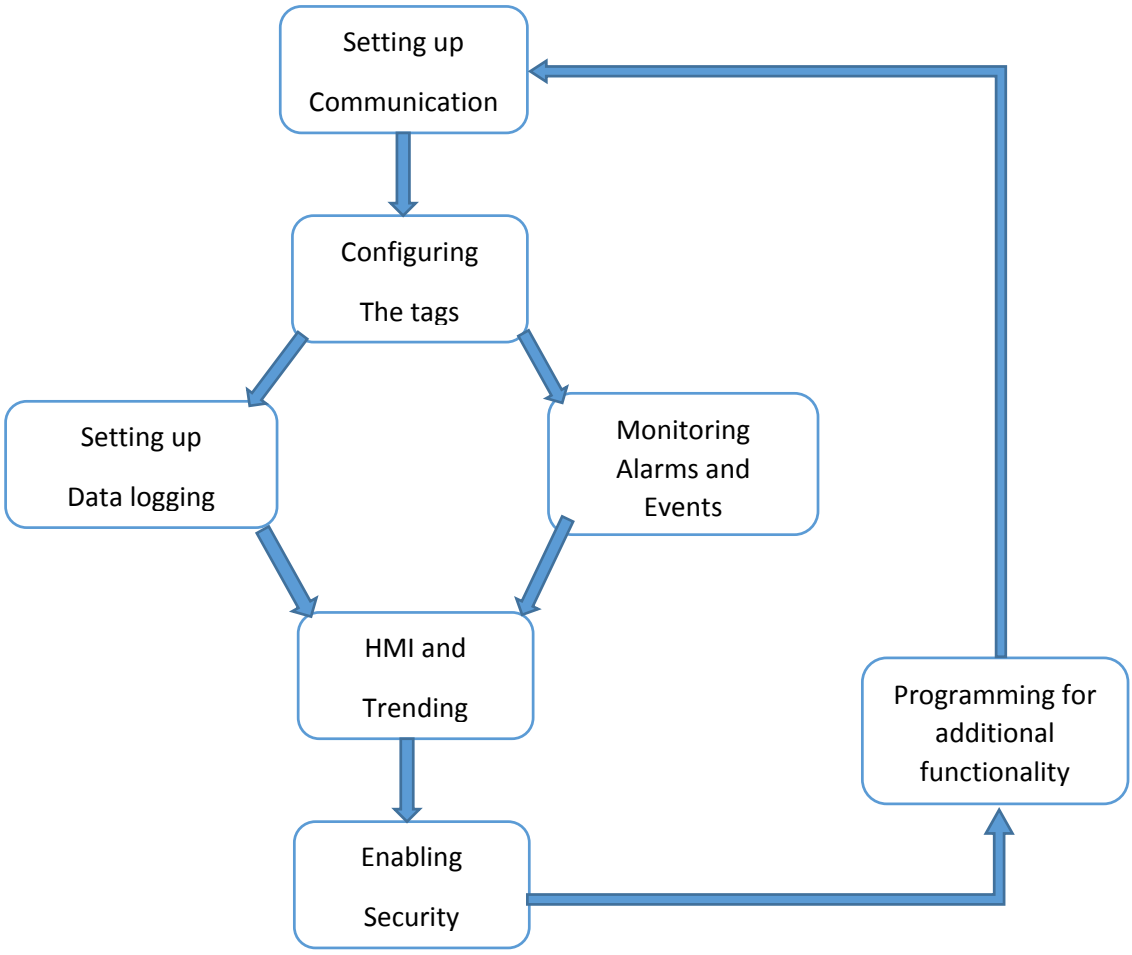


Figure 3.5. Pahses of building a SCADA

3.4.2.1 Setting up communication

The first step in the process of building a SCADA involve establishing the communications and data framework for the system which is considered as the backbone of the system as it plays a vital role in the operation of the system.

In order to assure communication with hardware an I/O server was created, LabVIEW DSC offers either OPC, Modbus or Custom I/O servers. The choice was to use OPC because of its flexible design and open environment that supports many PLC and other industrial devices. An OPC communication consists of a server and a client as shown in **Figure 3.6** below:

Figure 3.6. Typical OPC communication [21]

- **NI LabVIEW OPC Server**

OPC stands for Object Linking and Embedding for Process Control. It is a 'standard defined' interface to link devices in industries, it is used for connecting with different databases, laboratory equipment and test systems. Most of the industrial data acquisition devices and control devices such as PLCs are compatible with the OPC standard [21].

The LabVIEW (DSC) Module provides OPC Client I/O servers for communicating with any server implementing the OPC Foundation OPC-DA protocol, which is a Microsoft COM-based standard. An OPC Client I/O server lists all OPC servers installed on the computer and makes accessible groups and items on the server. When an OPC Client I/O server is created, data items can be accessed on a local or remote OPC server. In **Figure 3.7** (on next page) shows the relationship of the components involved in communication between LABVIEW and a PLC.

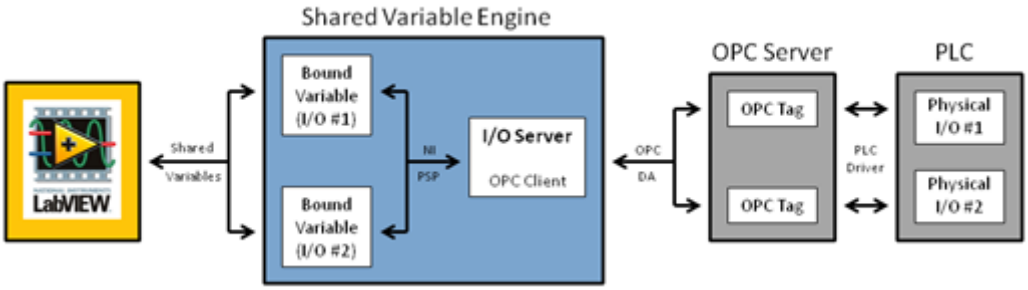
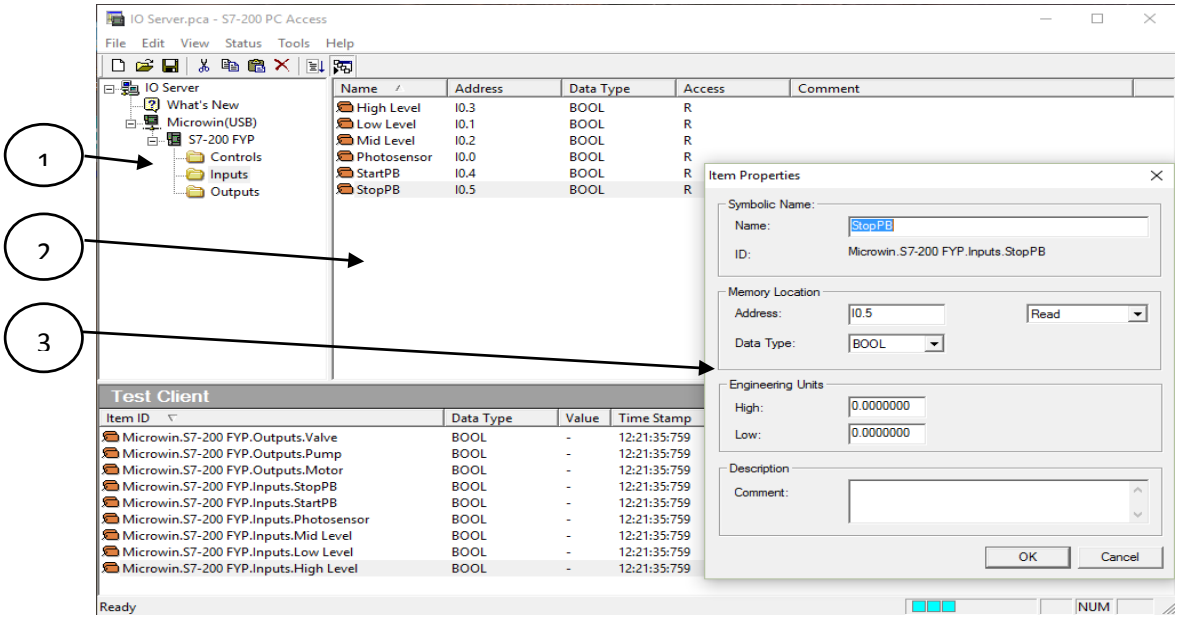


Figure 3.7. LABVIEW and the SVE Can Communicate with PLCs through OPC Network [22].

For the legacy PLC S7-200 in use, it was a tedious task to connect it to LabVIEW because NI OPC server did not recognize it due to the proprietary PLC protocol drivers which is the PC/USB PPI by Siemens. After trying many approaches to solve the problem, communication through PC Access provided by SIMATIC Siemens was established, that acts as a separate server software outside of LabVIEW whereas NI OPC as a client.

3.4.2.2 Tag Configuration

OPC Client I/O Servers allow for the Shared Variable Engine to bind the OPC tags from an OPC Server to Shared Variables. These bound Shared Variables provide an easy to use way for LabVIEW to read and write data to the OPC Tags. These tags was first created in PC Access as shown in **Figure 3.8** (on next page) by assigning each tag to a specific address of process variable, these tags consists of memory tags and I/O tags.



(1) Project Process Variables | **(2)** Tag description | **(3)** Tag configuration

Figure 3.8. Tag configuration in PC Access.

After the creation and configuration of the project tags, they can be bind in LabVIEW through OPC client and Shared Variable Engine.

3.4.2.3 Data Logging and Alarm Monitoring

A fundamental feature of the LabVIEW DSC Module is the ability to log data from SCADA and high-channel-count systems. In the proposed project the Citadel database was used, which is a historical database that enables the programmer to create, manage, and visualize I/O data collected through the DSC Module and allows for integration with any existing relational database infrastructures.

Using Distributed System Manager tool in labview it is possible to enable data logging, set resolution and cofiguring alarms and events, as shown in Figure 3.9 below.

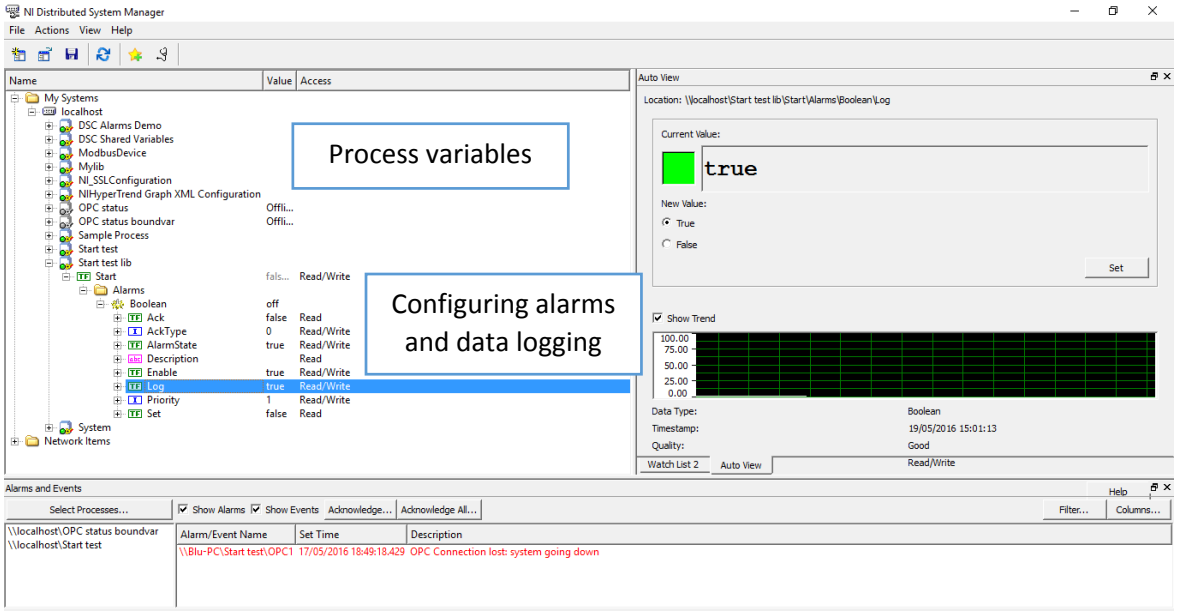


Figure 3.9 . Creating and configuring alarms and data logging for different Variables

3.4.2.4 Human Machine interface

HMI plays a significant role in creating a friendly visual interface between the user and the technology. In designing the HMI a fair consideration must be met, the use of simple design principals. For example, keeping the layout of the interface clean, and making sure to call attention to the important information.

By using the large library of the DSC module and respecting the above steps, an HMI / SCADA was developed for the automatic bottle filing process as shown on **Figure 3.10**

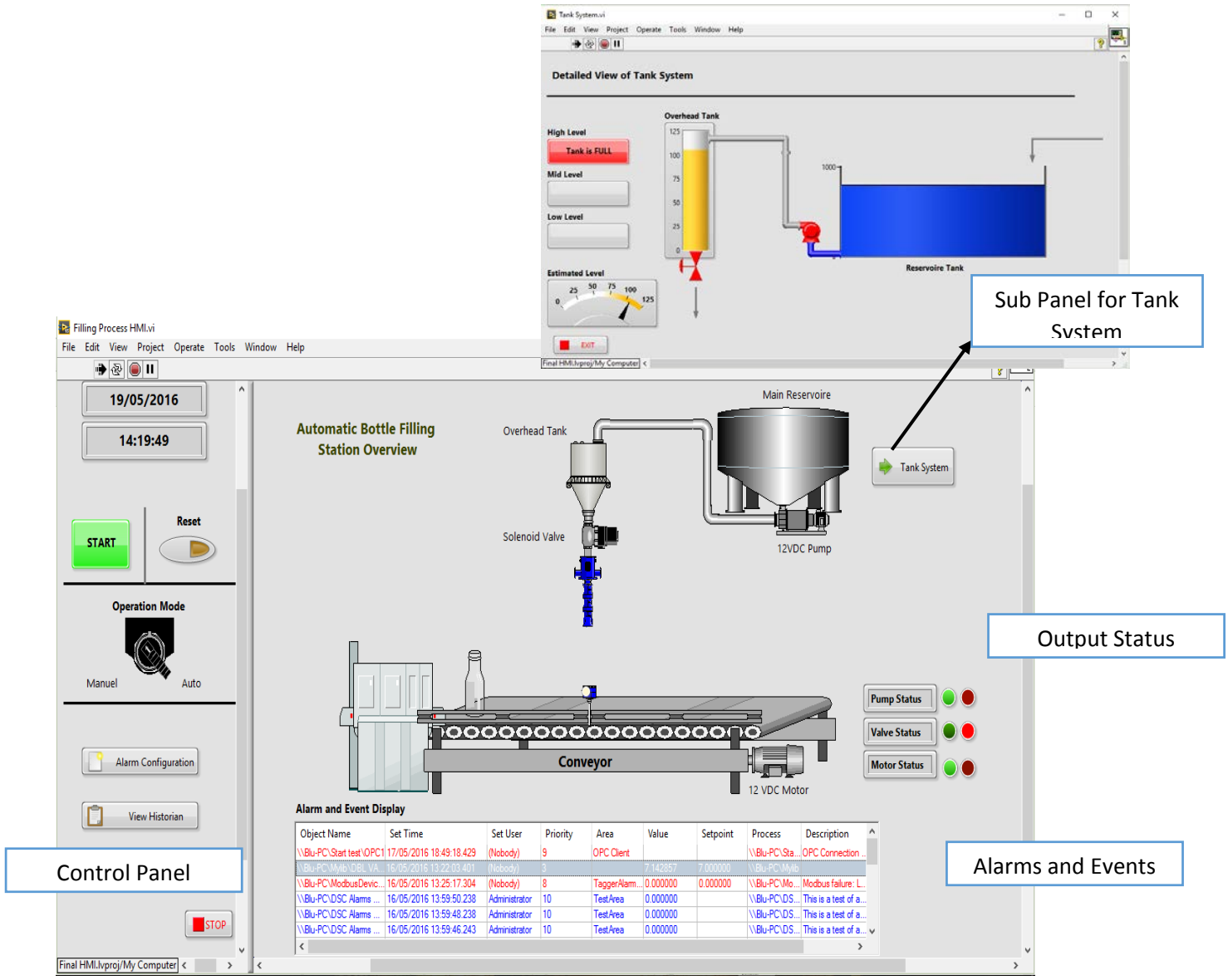


Figure 3.10. HMI for automatic bottle filling process

The designed HMI can be accessed from anywhere from a client station within the local area network after the deployment of the I/O server and publishing the created shared variable that were bind by data socket in LabVIEW. This offers the operator to monitor and control the filling process remotely and in real time.

3.4.2.5 Enabling Security

In LabVIEW locking an HMI is easily done by setting protection password in LabVIEW modifying the properties of front panels, as shown in **Figure 3.11** below.

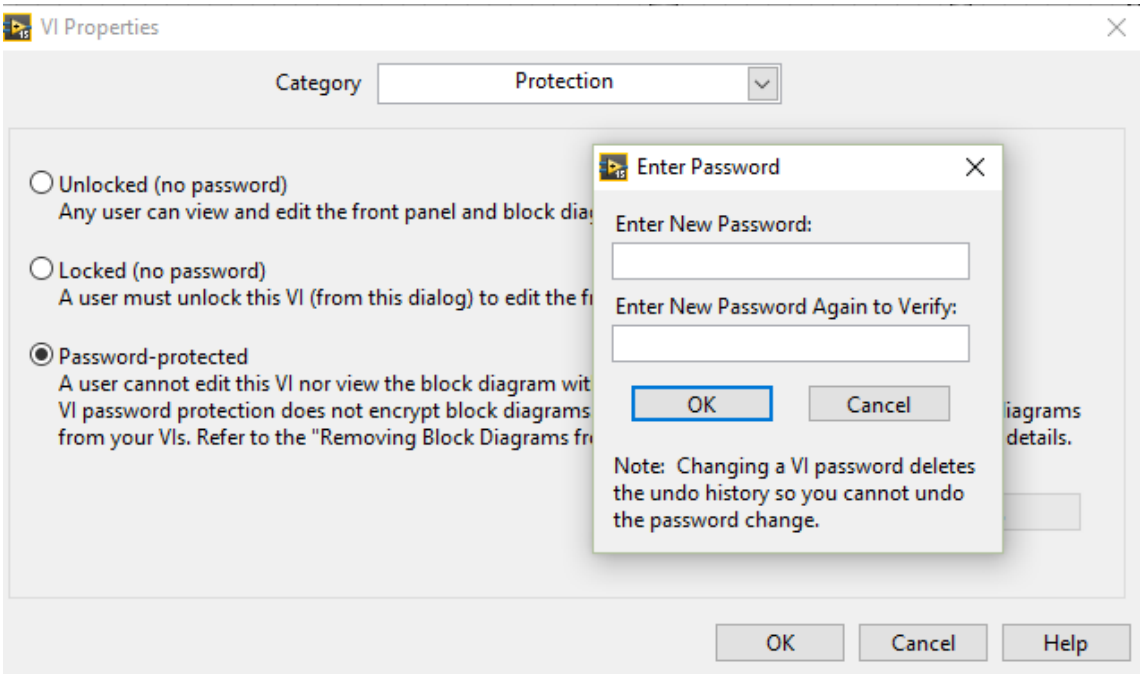


Figure 3.11. Setting Security options.

3.4.2.6 Programing for additional functionalities

Most SCADA systems extend by time, so it is important to keep in mind any future enhancement of the process or extensions. When developing the SCADA system a fair consideration to this aspect was taken by using an open environment for any further extension of hardware or applications.

3.5 Conclusion

This chapter introduced the design and development phases of the SCADA system for automatic bottle filling process.

3.1 Introduction

As mentioned in the previous chapters the design and implementation of the proposed control process requires both hardware and software. This chapter discusses the role of SCADA software's used in this project and how they interact on different levels.

3.2 SIMATIC STEP 7 Micro/WIN

Step 7 Micro/WIN is an automation software provided by Siemens Industry for automation solutions (www.industry.siemens.com). This software package can be handled like a standard Windows application and includes all necessary tools for programming of the SIMATIC S7-200 ,from the high-performance SIMATIC instruction set to IEC 611131-3 compliant programming and all the way to trend charts and wizards.

3.2.1 Merging Hardware and Software

Using the STEP 7 software, the program can be created within a project, then download it to the PLC and setting it in run mode to run and control the process with the S7 program. The I/O modules are addressed in the S7 program via the addresses.

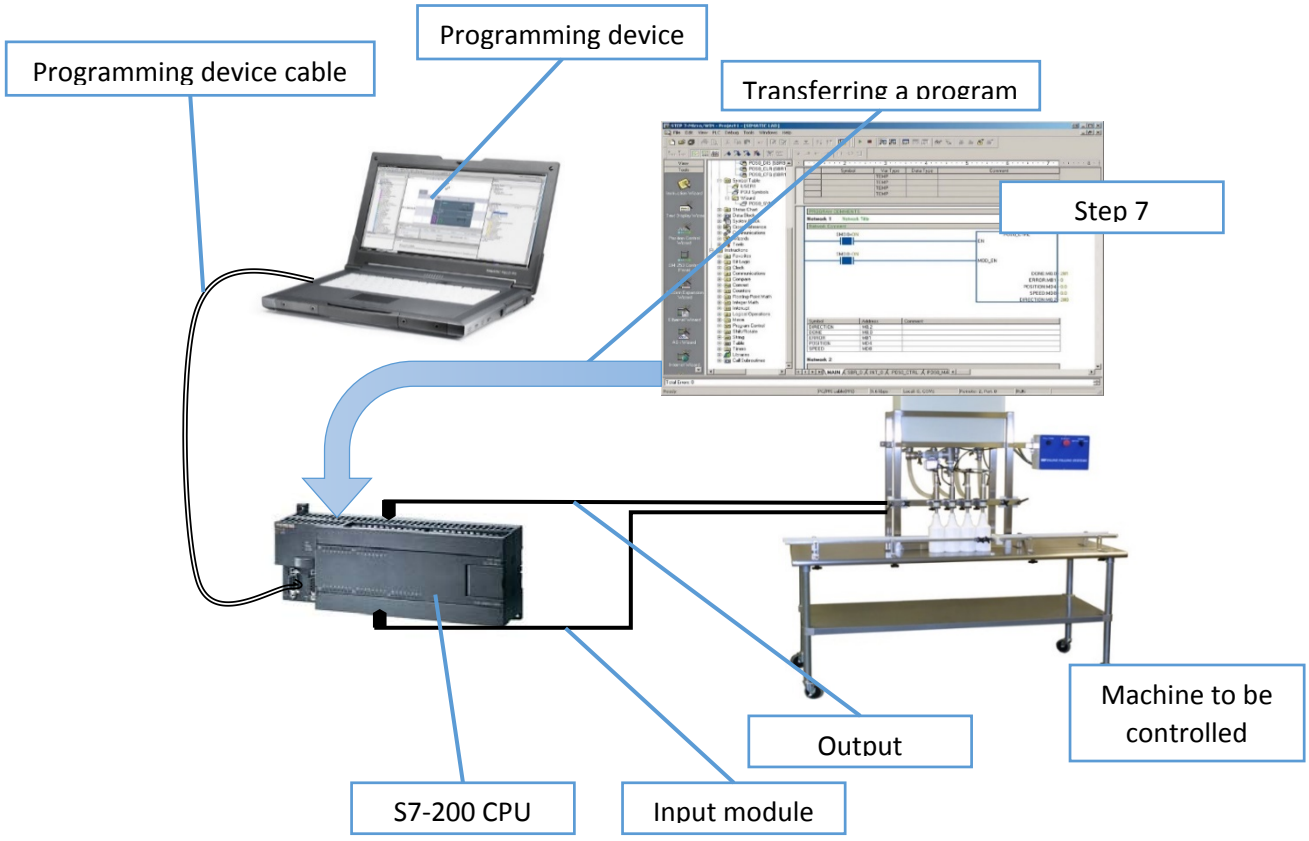


Figure 3.1. PLC connected with programing device and controlled machine

3.3 NI LabVIEW

NI LabVIEW stands for Laboratory Virtual Instrumentation Engineering Workbench provided by National Instruments (www.ni.com) is a development environment designed specifically to accelerate the productivity of engineers and scientists in a wide area of industries with a graphical programming syntax that makes it simple to visualize, create, and code engineering systems.

One great benefit of graphical programming (called G) in LABVIEW is that application development for prototyping takes much less time and because the easy and intuitive design that executes by 'Data Flow', rather than the more traditional procedural approach, which gives an advantage to programmers with different background in engineering than IT to debug their programs using a full suite of debugging tools. The built in compiler of LabVIEW as well as a linker and the fact that the flow of data between nodes regulates program execution rather than a prioritized list, that makes this style of programming different, and extremely powerful for the purposes that it is used for.

Basic LabVIEW program or as they are called 'virtual instruments' (VIs).consists of two windows, the Front Panel and the Block Diagram.

3.3.1 LabVIEW Front Panel

In LabVIEW, you build a user interface by using a set of tools and objects. The user interface is known as the front panel. As shown in the Figure.3.2 bellow.

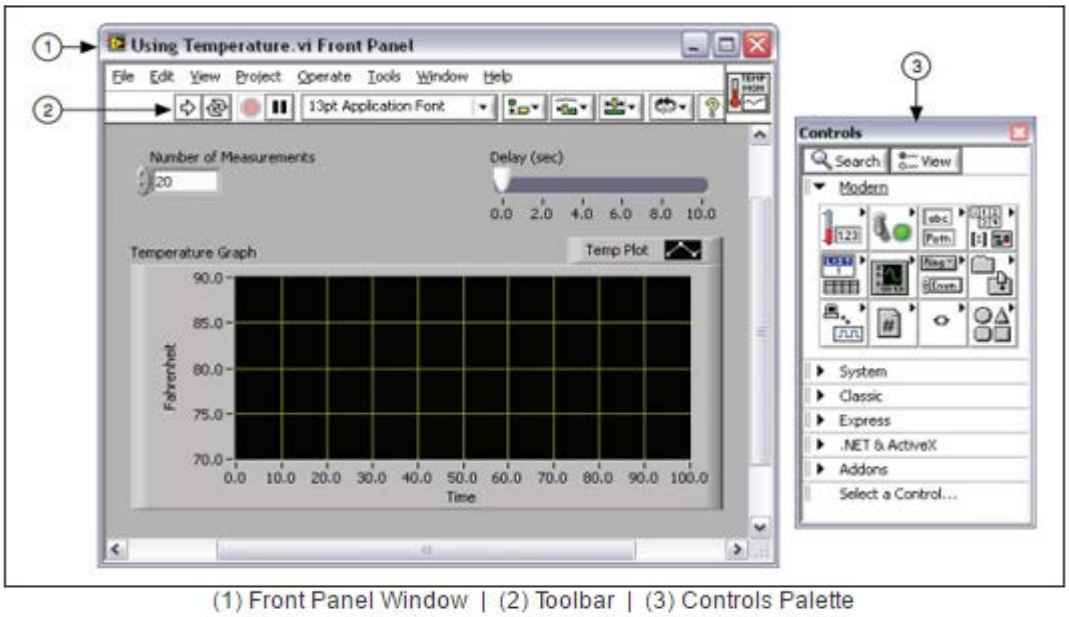


Figure 3.2. Example of a Front Panel [1]

Users interact with the Front Panel when the program is running. Users can control the program, change inputs, and see data updated in real time. Controls are used for inputs such as, adjusting a slide control to set an alarm value, turning a switch on or off, or to stop a program.

3.3.2 LabVIEW Block Diagram

The block diagram contains the graphical source code of a LabVIEW program. The concept of the block diagram is to separate the graphical source code from the user interface in a logical and simple manner. Front panel objects appear as terminals on the block diagram. Terminals on the block diagram reflect the changes made to their corresponding front panel objects and vice versa.

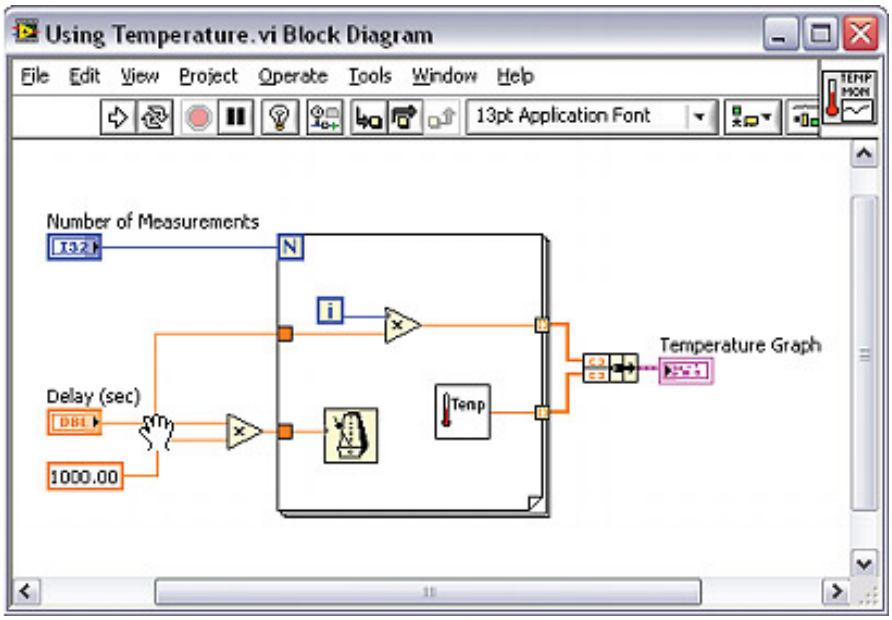


Figure 3.3. Example of a Block Diagram [1]

LABVIEW program also contains a three pallets which give the options that are need to create and edit the front panel and block diagram, these pallets include:

- **Tools Palette**

The Tools palette is available on the front panel and the block diagram. A tool is a special operating mode of the mouse cursor. When a tool is selected the cursor icon changes to the tool icon. The tools are used to operate and modify front panel and block diagram objects.

- **Controls Palette**

The Controls palette is available only on the front panel. The Controls palette contains the controls and indicators that are used to create the front panel.

- **Functions Palette**

The Functions palette is available only on the block diagram. The Functions palette contains the VIs and functions that are used to build the block diagram. The Figure 3.4 below shows the three types of pallets:

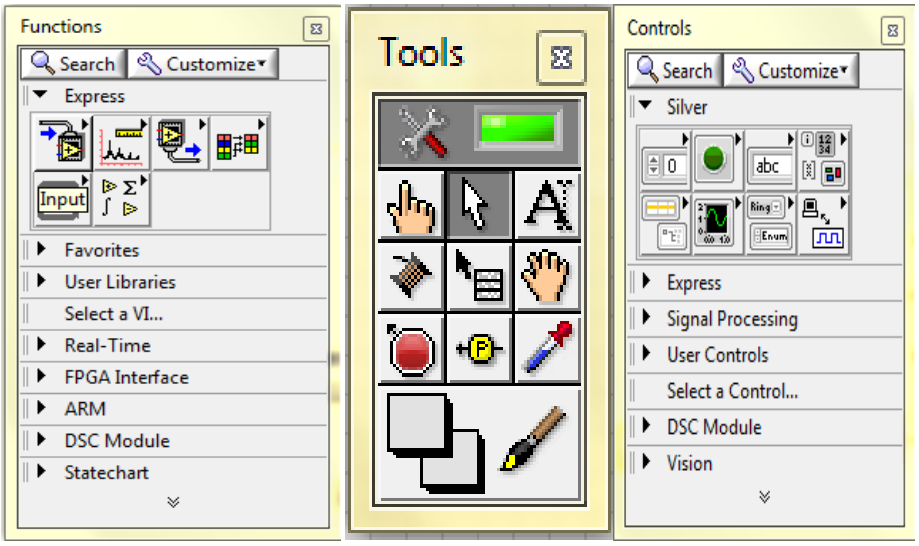


Figure 3.4. Function, Tools and Controls pallets. [1]

3.4 Developing HMI/SCADA System in LabVIEW

In the past, engineers developing industrial systems had to learn different software tools to program embedded controllers and HMI/SCADA applications even when buying everything from the same vendor.

With National Instruments LabVIEW graphical system design software, it is now possible to program human machine interface (HMI) and logic in the same environment, minimizing development cost, training time, and maximum skill re-use. Specifically, the use of the LabVIEW data logging and supervisory control (DSC) module to develop a full-fledged HMI/SCADA application.

3.4.1 Overview of LabVIEW DSC Module

Applications today require open HMI/SCADA environments with OLE for process control (OPC), Modbus connectivity, and the flexibility of a programming language, so that high-speed measurements and analysis are added to existing systems. The LabVIEW DSC Module provides configuration-based data logging, alarming, scaling, and security for developing the Windows-based HMI/SCADA system. A read or write operations are easily made on OPC connections, PLCs, or custom I/O servers that are created by the use of DSC Module that provides solutions for supervisory control of a wide variety of distributed systems.

3.4.2 Steps of Building a SCADA system

In order to increase efficiency and make troubleshooting easy if any problems occur a break down structure approach to the deployment of the SCADA system into different phases as show in Figure 3.5 below was taken.

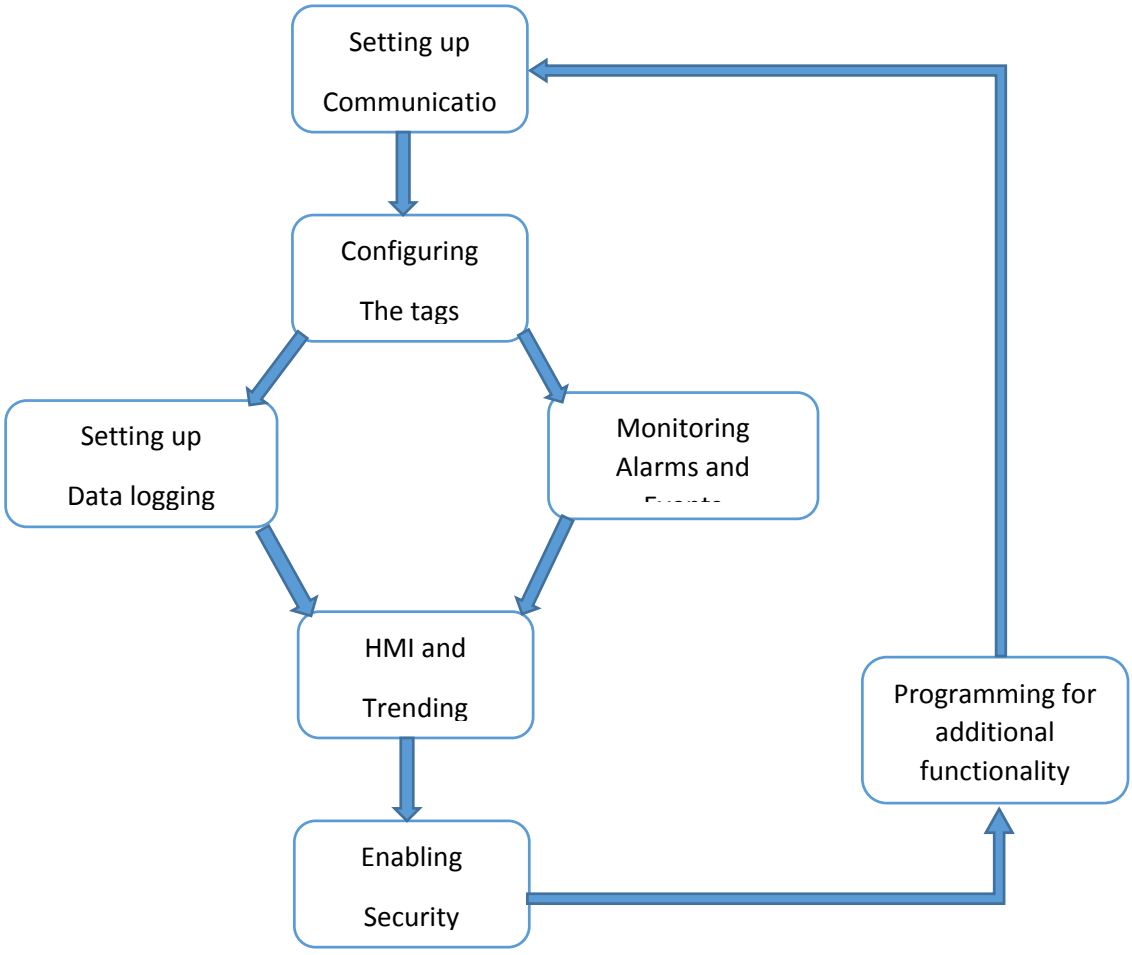


Figure 3.5. Pahses of building a SCADA

3.4.2.1 Setting up communication

The first step in the process of building a SCADA involve establishing the communications and data framework for the system which is considered as the backbone of the system as it plays a vital role in the operation of the system.

In order to assure communication with hardware an I/O server was created, LabVIEW DSC offers either OPC, Modbus or Custom I/O servers. The choice was to use OPC because of its flexible design and open environment that supports many PLC and other industrial devices. An OPC communication consists of a server and a client as shown in Figure 3.6 below:

Figure 3.6. Typical OPC communication [1]

- **NI LabVIEW OPC Server**

OPC stands for Object Linking and Embedding for Process Control. It is a 'standard defined' interface to link devices in industries, it is used for connecting with different databases, laboratory equipment and test systems. Most of the industrial data acquisition devices and control devices such as PLCs are compatible with the OPC standard [1].

The LabVIEW (DSC) Module provides OPC Client I/O servers for communicating with any server implementing the OPC Foundation OPC-DA protocol, which is a Microsoft COM-based standard. An OPC Client I/O server lists all OPC servers installed on the computer and makes accessible groups and items on the server. When an OPC Client I/O server is created, data items can be accessed on a local or remote OPC server. In Figure 3.7 (on next page) shows the relationship of the components involved in communication between LABVIEW and a PLC.

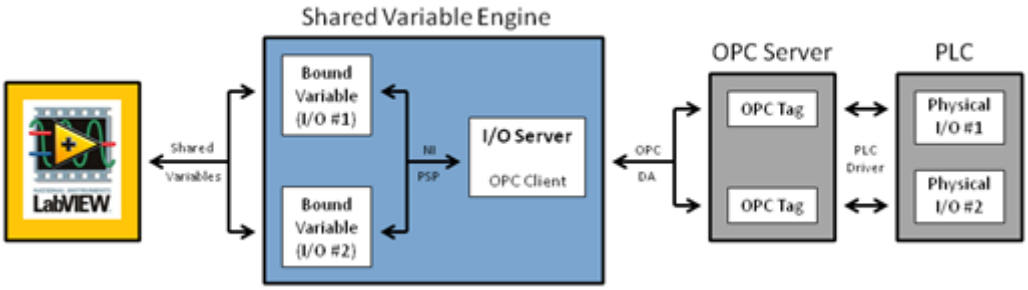
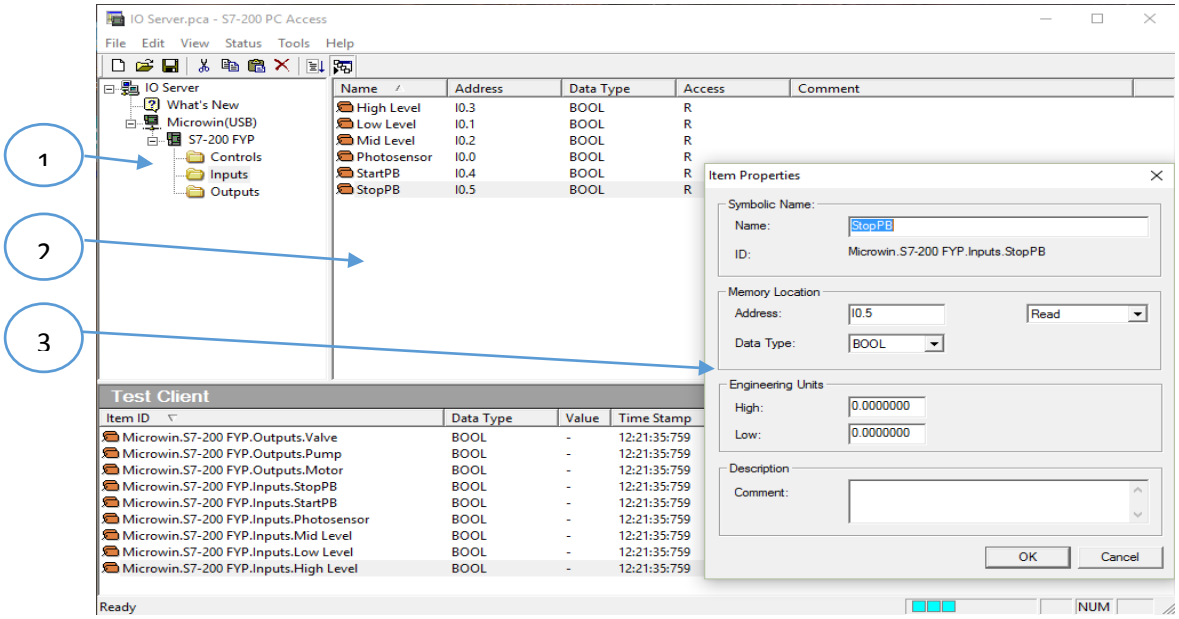


Figure 3.7. LABVIEW and the SVE Can Communicate with PLCs through OPC Network [2].

For the legacy PLC S7-200 in use, it was a tedious job to connect it to LabVIEW because NI OPC server did not recognize it due to the proprietary PLC protocol drivers which is the PC/USB PPI by Siemens. After trying many approaches to solve the problem, communication through PC Access provided by SIMATIC Siemens was established, that acts as a separate server software outside of LabVIEW whereas NI OPC as a client.

3.4.2.2 Tag Configuration

OPC Client I/O Servers allow for the Shared Variable Engine to bind the OPC tags from an OPC Server to Shared Variables. These bound Shared Variables provide an easy to use way for LabVIEW to read and write data to the OPC Tags. These tags was first created in PC Access as shown in Figure 3.8 (on next page) by assigning each tag to a specific address of process variable, these tags consists of memory tags and I/O tags.



(1) Project Process Variables | **(2)** Tag description | **(3)** Tag configuration

Figure 3.8. Tag configuration in PC Access.

After the creation and configuration of the project tags, they can be bind in LabVIEW through OPC client and Shared Variable Engine.

3.4.2.3 Data Logging and Alarm Monitoring

A fundamental feature of the LabVIEW DSC Module is the ability to log data from SCADA and high-channel-count systems. In the proposed project the Citadel database was used, which is a historical database that enables the programmer to create, manage, and visualize I/O data collected through the DSC Module and allows for integration with any existing relational database infrastructures.

Using Distributed System Manager tool in labview it is possible to enable data logging, set resolution and cofiguring alarms and events, as shown in Figure 3.9 below.

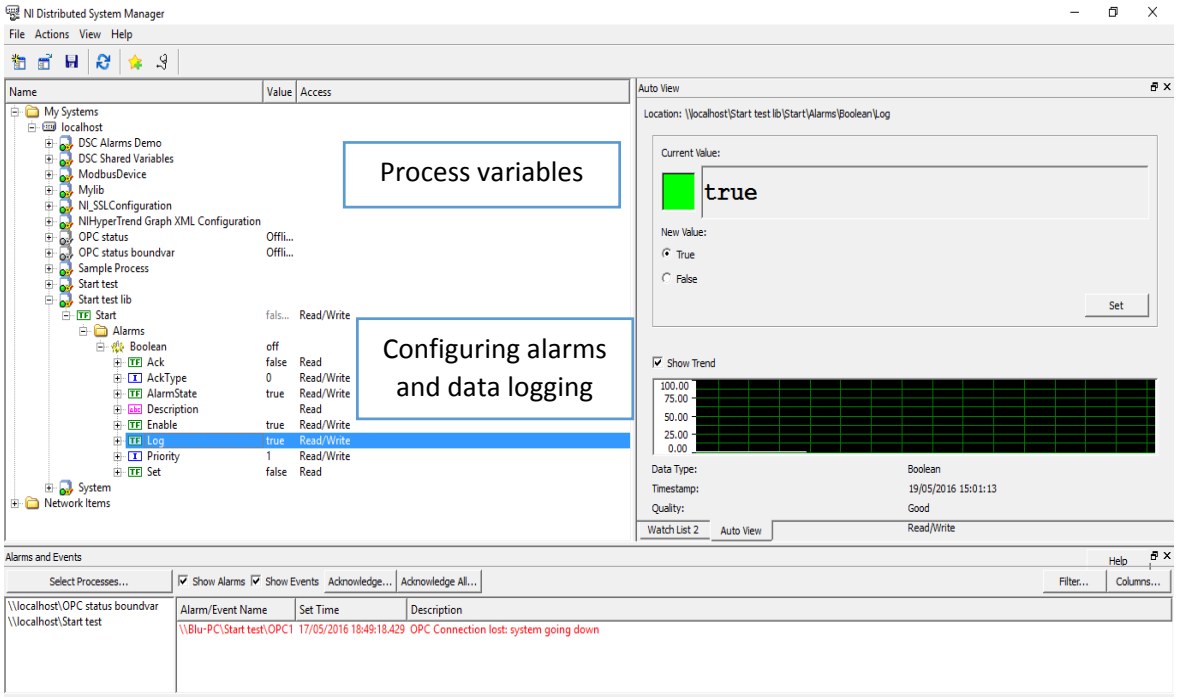


Figure 3.9 . Creating and configuring alarms and data logging for different Variables

3.4.2.4 Human Machine interface

HMI plays a significant role in creating a friendly visual environment between the user and the technology. In designing the HMI a fair consideration must be met to the use of simple design principals. For example, keeping the layout of the interface clean, and making sure to call attention to the important information.

By using the large library of the DSC module and respecting the above steps, an HMI / SCADA was developed for the automatic bottle filing process as shown on **Figure 3.10**

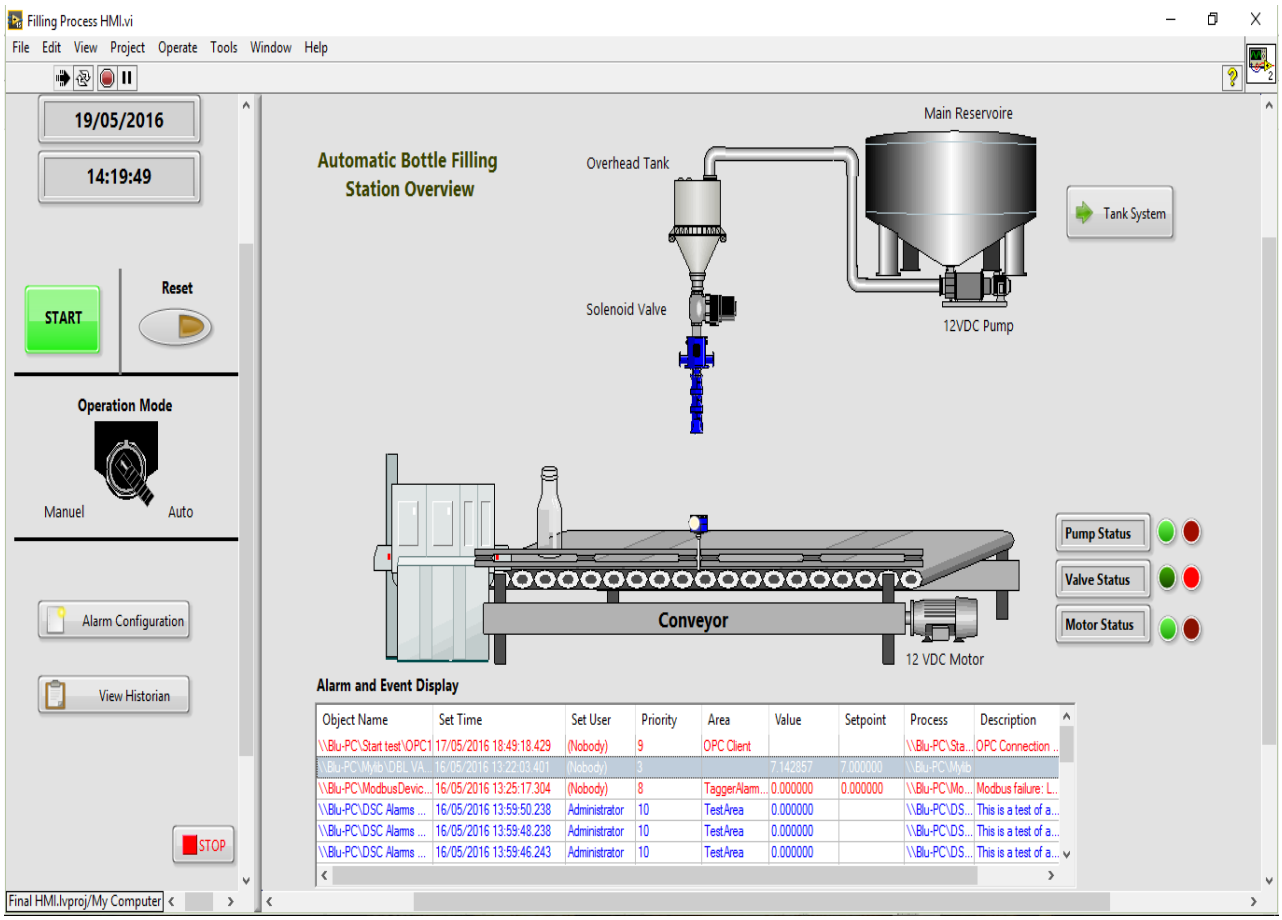


Figure 3.10. HMI for automatic bottle filling process

3.4.2.5 Enabling Security

In LabVIEW locking an HMI is easily done by setting protection password in LabVIEW modifying the properties of front panels, as shown in Figure 3.11 below.

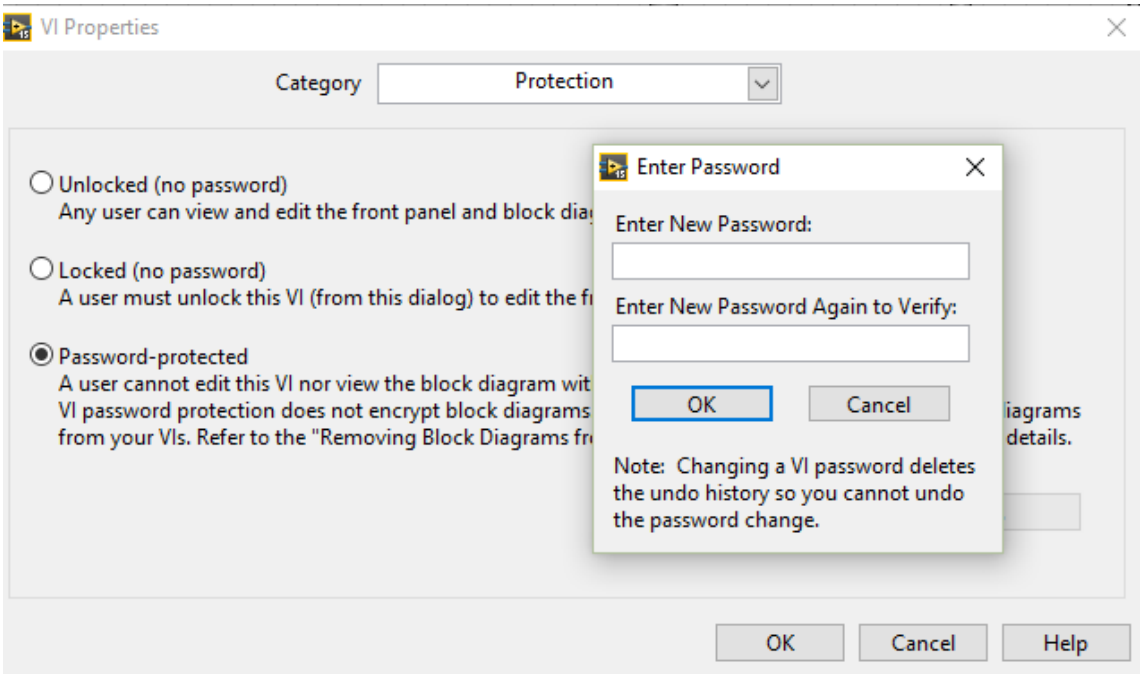


Figure 3.10. Setting Security options.

3.4.2.6 Programing for additional functionalities

Most SCADA systems extend by time, so it is important to keep in mind any future enhancement of the process or extensions. When developing the SCADA system a fair Consideration to this aspect was taken by using an open environment for any further extension of hardware or applications.

3.5 Conclusion

This chapter introduced the design and development phases of the SCADA system for automatic bottle filling process.

Chapter Four

System Implementation and Test

4.1 Introduction

In this chapter we will talk about the hardware construction of each individual part and the software used to implement and test the system.

4.2 Hardware construction

4.2.1 Mechanical design and implementation

4.2.2 Construction of the conveyor belt

Figure 4.1 shows the frame of the conveyer which is 70 cm long and 20 cm wide. It has one driving rollers which is connected to DC gear motor and 3 free rollers with bearings, the distance between the rollers is 20 cm.

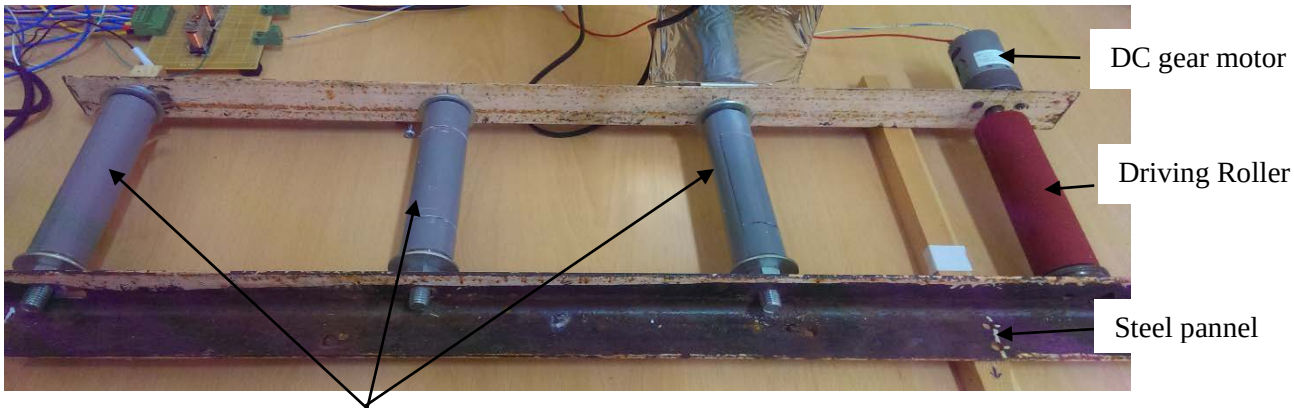


Figure 4.1. Frame of the conveyor belt

Figure 4.2 shows the belt used for this project. It is made of plastic, its inner surface is non slippery to provide enough friction with the rollers in order to make the driving roller drive the free rollers, this movement of the rollers makes the conveyor run smoothly.

Figure 4.2. Belt

4.2.2.1 Construction of the Filling Station

- We used a bottle to simulate the overhead tank as mentioned in chapter 2.
- The electrovalve is connected to the bottle and the level sensors are placed inside this bottle.
- Two holes are made on top of this bottle: one for the pipe that comes from the main tank and the other is for the nap used in making the level sensors.
- The photoelectric sensor and its reflector is placed in the sides of the conveyor and it was calibrated in such way to detect the bottle when it is exactly beneath the overhead tank.

Figure 4.3 shows the filling station

1: Electrovalve

2: Pipe

3: Reflector

4: Photoelectric sensor

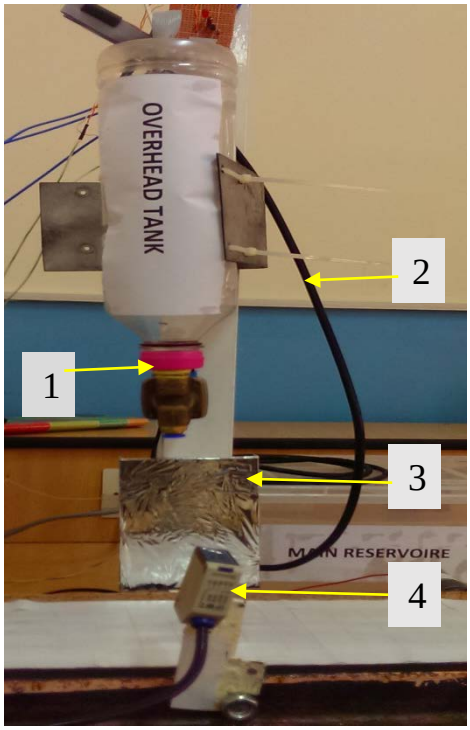


Figure 4.3. The filling station

4.2.2.2 Construction of the main reservoir

Figure 4.4 shows the main reservoir with the DC motor pump used for this project. It is placed at a distance from the conveyer belt.

The DC motor pump is screwed tightly into the reservoir to prevent the water loss.

A pipelining system is used to connect the main reservoir to the overhead tank.

Figure 4.4. The main reservoir

4.2.3 Electrical design and implementation of the system

4.2.3.1 Input circuits

We have basically 6 inputs going to the PLC:

- Start push button
- Stop push button
- Low level sensor
- Middle level sensor
- High level sensor
- Photoelectric sensor

A. Start and stop push button circuit

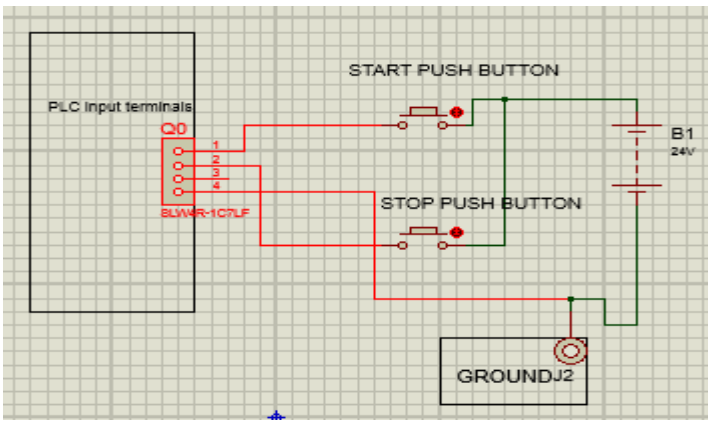


Figure 4.5. Start and stop PB circuit

When one of the push buttons is pressed the corresponding PLC input terminal receive a high signal.

B. Level sensor circuit

Figure 4.6 shows the level sensor circuit

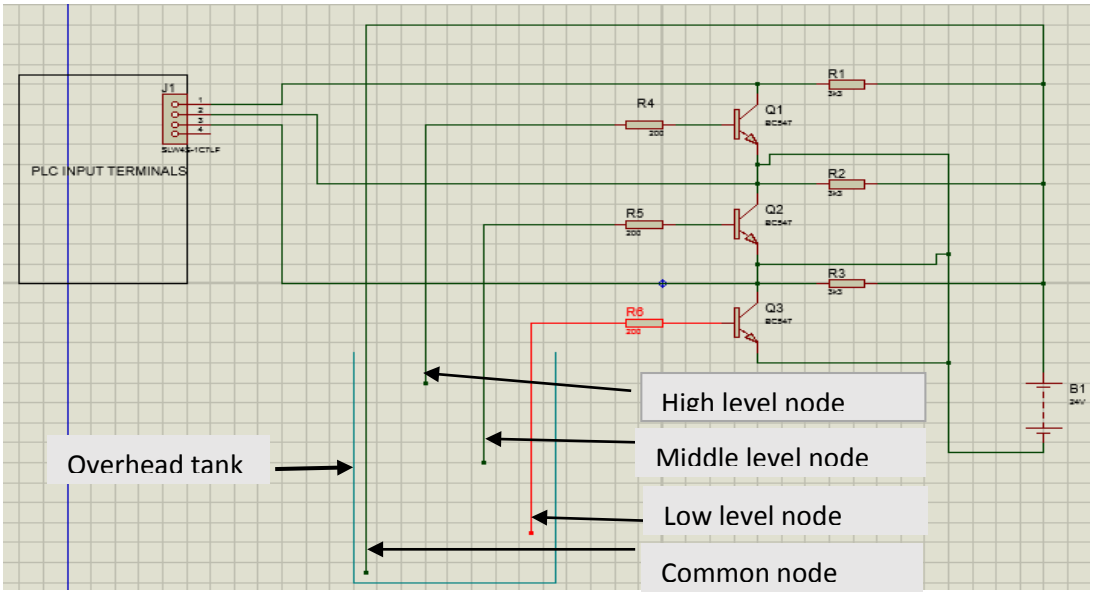


Figure 4.6. Level sensor circuit

When the level of the liquid reaches one of the nominated levels the positive side of the battery gets connected to the base of its corresponding transistor and a current will flow from the collector to the emitter which makes the transistor in the ON state.

Remarks:

- The PLC and the level sensor circuit must have the same ground.
- The outputs is taken from the collectors to the input terminals of the PLC's.
- If the water goes below the lower level the corresponding PLC input terminals receive a high signal and when it reaches the high level the corresponding PLC input terminal receive a low signal.
- The resistors are used to limit the maximum base and collector current.

C. Photoelectric sensor connection

Figure 4.7 (next page) shows the terminals connection of the photoelectric sensor

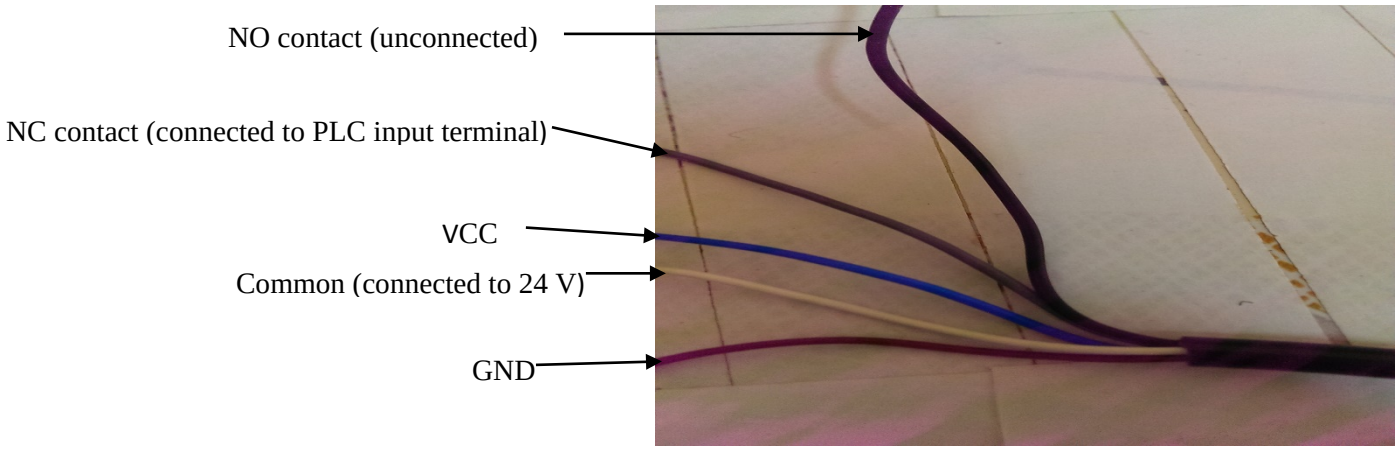


Figure 4.7. Photoelectric sensor terminals connection

When a bottle is detected the PLC input terminal receive high signal.

4.2.3.2 Output circuit

Three actuators are used in this project so we have three outputs:

- Conveyor motor
- Motor pump
- Electrovalve

Figure 4.8 shows the output circuit

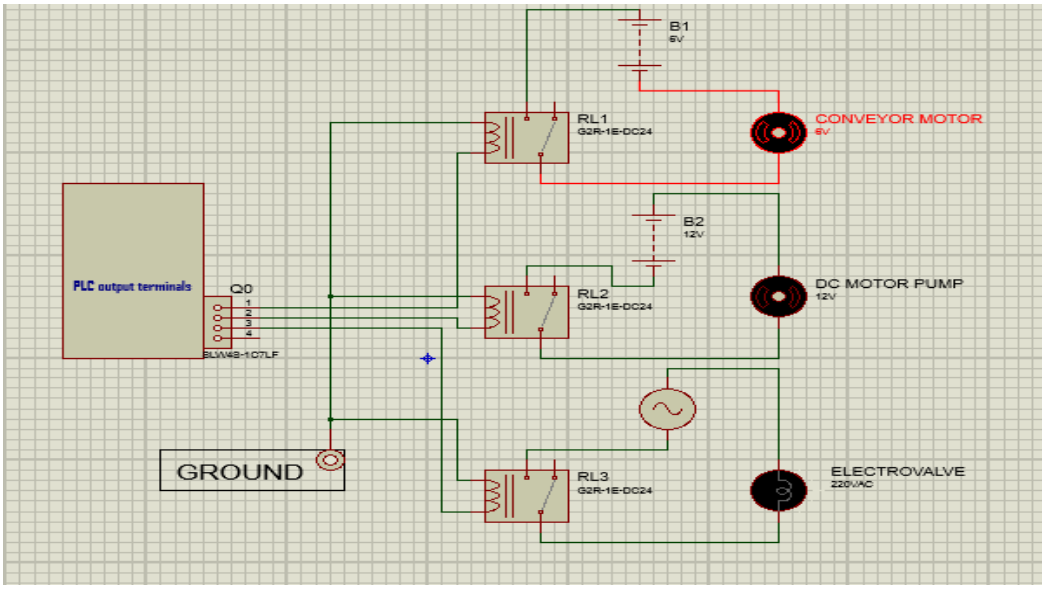


Figure 4.8. Output circuit

When the PLC send a high signal to one of the relays the removable contact switch from the NC to NO contact making the external power circuit drive its corresponding actuator.

4.2.3.3 Overall circuit

Figure 4.9 shows the overall circuit

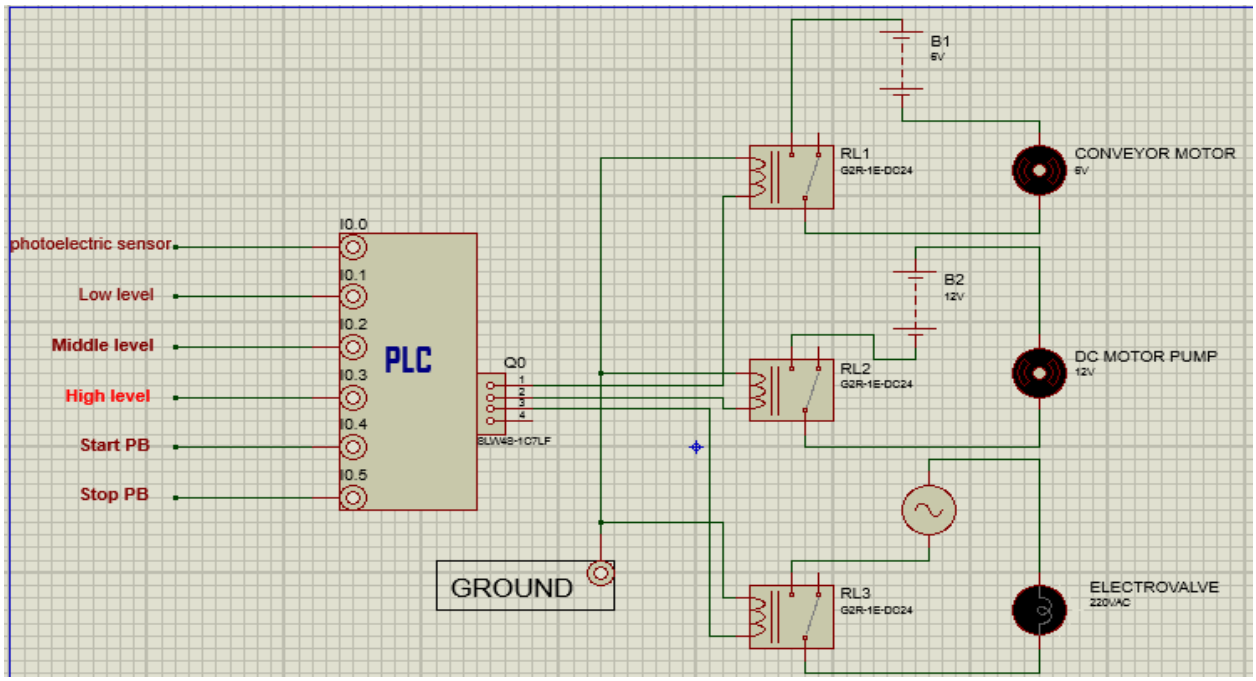


Figure 4.9. Overall circuit

4.3 Software

4.3.1 SIMATIC STEP 7 Professional

STEP 7 PROFESSIONAL (STEP 7 in short) is an automation software from Siemens Software Industry. It is used for configuring and programming SIMATIC PLC stations. There are different versions of STEP7 standard package the one used in this project is STEP 7 MicroWin.

4.3.2 Basic procedures in using step 7

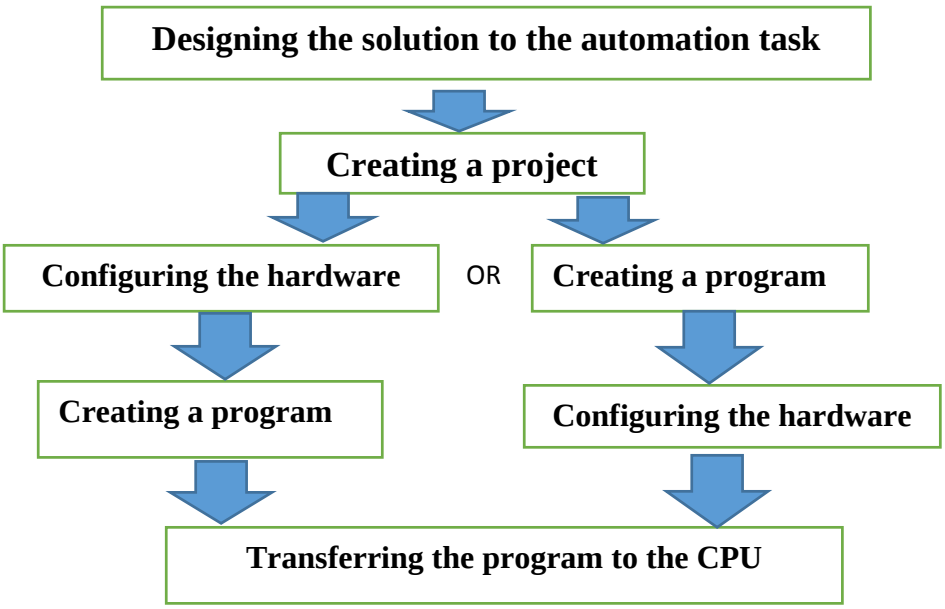


Figure 4.10. Basic procedures to create a program in step 7

4.3.3 SIMATIC Manager

The SIMATIC Manager manages all the data that belong to an automation project. It is used to allow the user to build his control system and simulate it and even download it to the PLC. The tools needed to edit the selected data are started automatically by the SIMATIC Manager.

4.3.4 Creating a program

To control a process using PLCs, it is required to program the PLC according to the desired control logic flow so before attempting to program the PLC the flow of the control logic must be well defined.

4.3.4.1 Flow Chart for Bottle Sensing and Filling System

Figure 4.11 shows the flow chart of the bottle sensing and filling system. When the system is powered on, the conveyor belt starts running. The conveyor belt keeps running if the photoelectric sensor does not detect the presence of any bottle in front of it. If the sensor detects any bottle then the conveyor belt stops, Electrovalve opens and bottle filling starts, the amount of liquid filling is controlled using a weighting sensor. Depending on the weight of the bottle

the opening of the valve is decided. The Electrovalve then closes and after some time delay conveyor belt starts again with the filled bottle and carries the bottle to the other end where the bottle is collected.

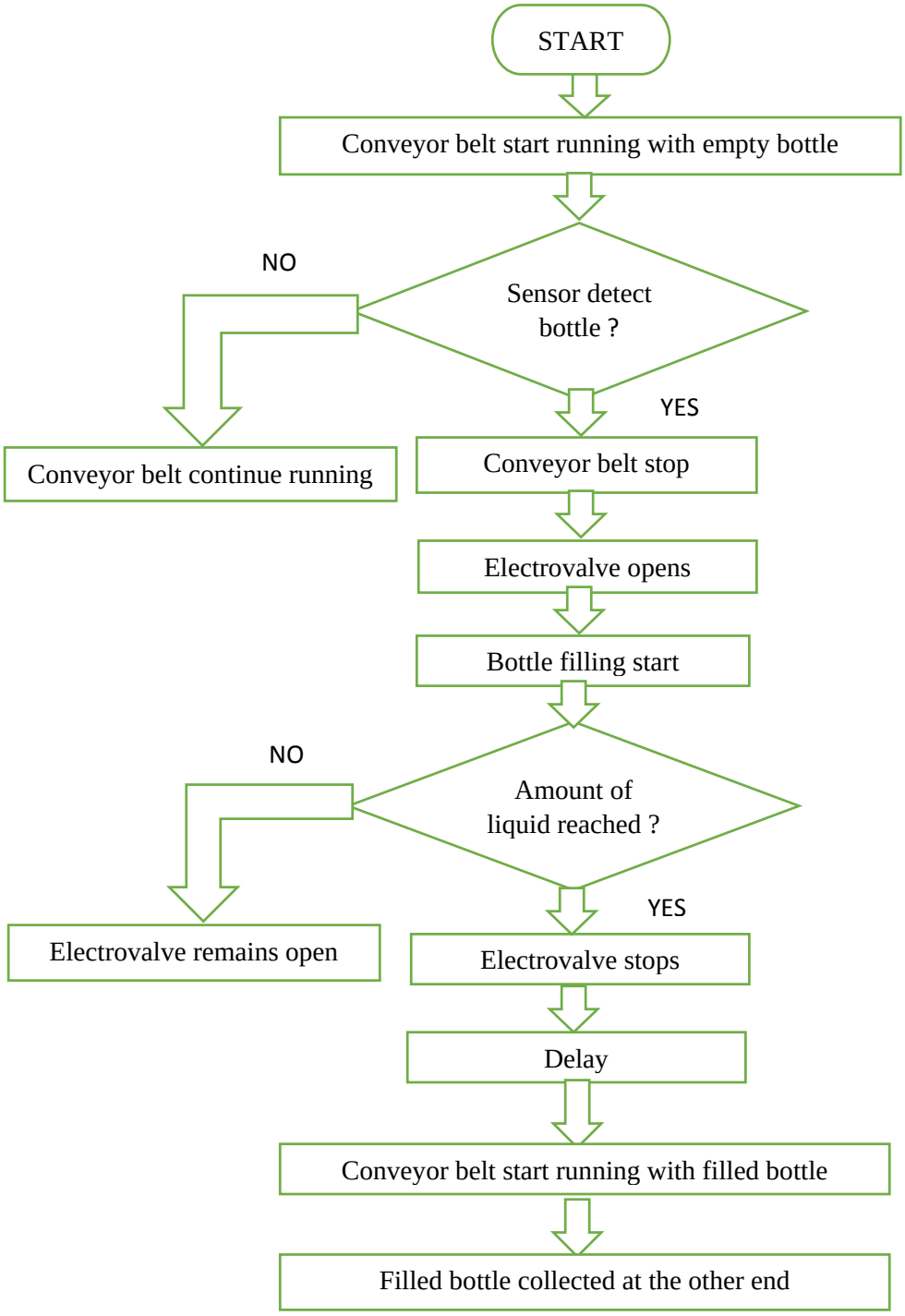


Figure 4.11. Flow Chart for Bottle Sensing and Filling System

4.3.4.2 Flow Chart for Maintaining Liquid Level at the Overhead Tank

Figure 4.12 shows the flow chart of liquid level control at the overhead tank. As we need to control the liquid level at the overhead tank, leveling sensors are used. When the liquid goes below low level, the DC motor starts pumping liquid from the main reservoir to the overhead tank. When the liquid touches upper level, DC motor pump stops to prevent overflow of liquid at the overhead tank. If the liquid does not touch the upper level, the DC motor pump remains on.

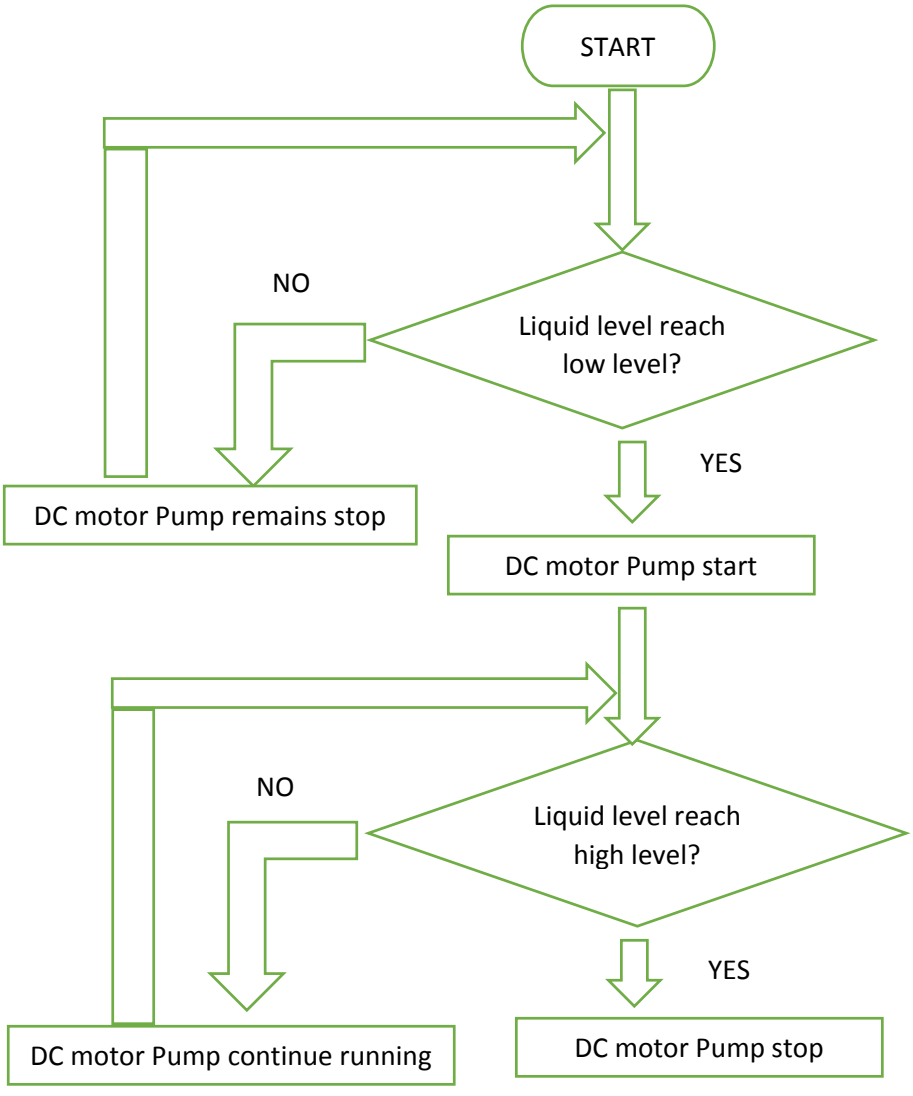


Figure 4.12. Flow chart for maintaining liquid level at the overhead tank

4.3.4.3 Assigning inputs and outputs

Before creating the ladder diagram we must assign each input and output of the system to a specific input and output addresses of the PLC .The input and output assignment are shown in the flowing tables.

Inputs	Addresses
Photoelectric sensor	**I0.0**
Low level sensor	**I0.1**
Middle level sensor	**I0.2**
High level sensor	**I0.3**
Star push button	**I0.4**
Stop push button	**I0.5**

Table 4.1. Input assignment

Outputs	Addresses
Conveyor motor	**Q0.0**
DC motor pump	**Q0.1**
Electrovalve	**Q0.2**

Table 4.2. Output assignment

4.3.4.4 The Ladder Diagram

Different languages are used to program the PLC some of them are FBD (Function Block Diagram), LD (Ladder Diagram), and SCL (Statement List). In this project the Ladder Diagram is used to create the program.

4.4 Overview of the implemented prototype

Figure.4.13 (next page) show the General view of the realization of the prototype

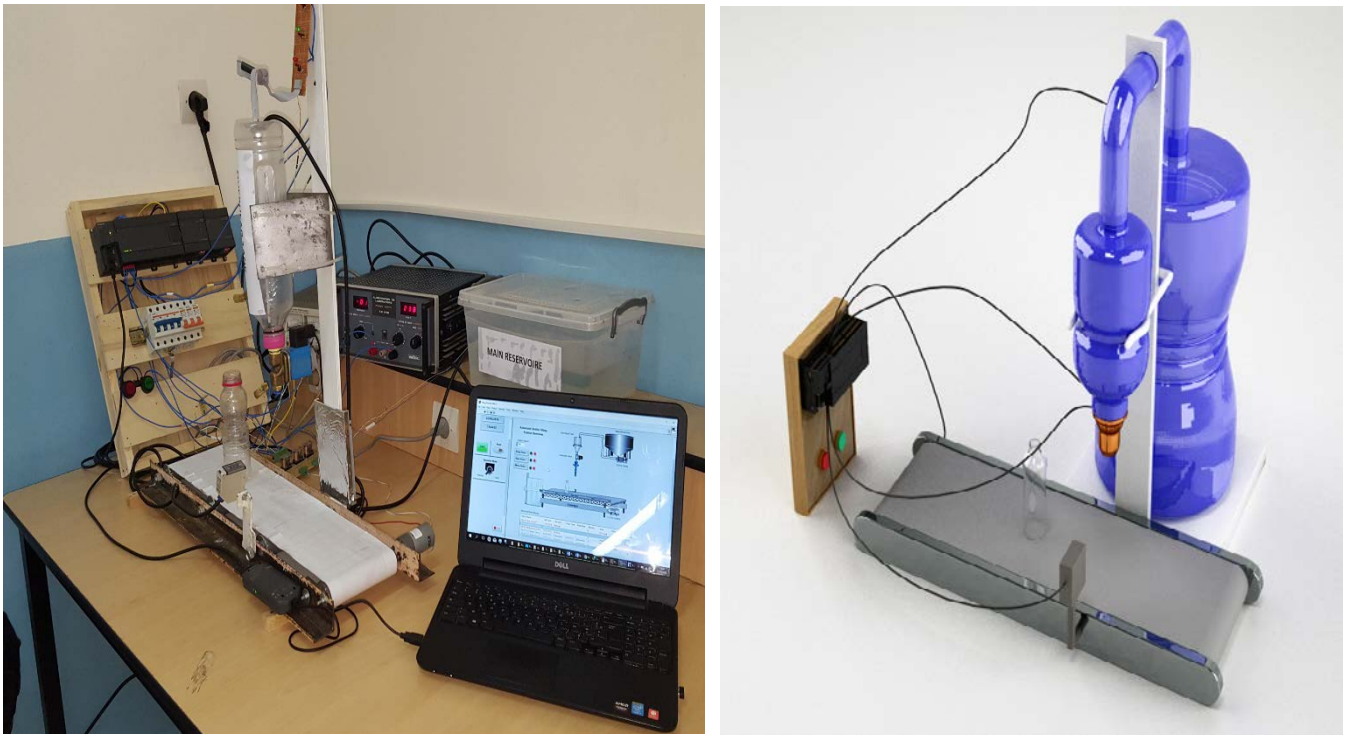


Figure 4.13. Overview of the implemented prototype

4.5 Conclusion

After implementing the system many tests have been conducted all of which showed that the system worked properly and according to the desired control logic.

Also we have tested the HMI and it answered to the user request correctly.

Comment:

Since weighting sensor was not available to control the amount of liquid to be filled while implementing the system. The amount is controlled using a timer instead, which results in different amount of liquid for different bottles.

General Conclusion

In this project the design and implementation of PLC-based automatic bottle filling system with SCADA capabilities was carried. The aim of this project was to build a prototype for an automatic bottle filling station, control the process by use of a PLC and keep track of the overall system by monitoring and controlling the bottle filling process remotely in real-time.

First, the mechanical design of the prototype was build using metallic frame conveyor and an overhead filling tank system according to the CAD concept. Secondly, the program that controls the process was written in Step 7 and thirdly for remote monitoring and control a distributed SCADA system was deployed using LabVIEW over a LAN network that connected a master station and a client station with near real time responses (150 ms). The objectives were met, however the project is still at its early stages and can transfer data on a local area network only.

A four layers architecture was adopted to SCADA system, which are the field instrument, the process control, communication infrastructure, and supervisory control layer, make the system modular and flexible and the use of LabVIEW make it an open system, this approach revealed one major advantage which is the possibility to treat any process automation system in same manner as the bottle filling process.

All the advantages we mentioned make this project a reference that helps electrical engineers to apply in an easier and more flexible way a SCADA system to many applications such as food packing, fertilizer bags and different production industries.

There are many features that could be added to the implemented system, due to limited resources and time we couldn.t implement them. These features can be used to improve the performance or to complete the cycle of liquid filling system such as:

- **Inline filling system**

In our project the conveyor has to stop when the bottle reaches the filling station and wait for the bottle till it is filled with the liquid to start moving again. In inline liquid filling system the bottles are filled while moving on the conveyor which will increase the productivity.

- **Filling more than one bottle at the same time**

It could be done by creating a system of filling that has many nozzles at the filling station. This also will increase the production rate.

- **Bottle feeding system**

A system that feeds the bottles to the conveyor could well be added.

- **Bottle capping system**
- **Packaging system**

References

[1] C. Gould, "Inductive Automation," Mars 2016. [Online]. Available: https://inductiveautomation.com/.

[2] A. Sarwate, "Qualys," March 2012. [Online]. Available: https://qualys.com/securitylabs/2012/03/27/scada-system-fundamentals.

[3] "SUBNET Soulutions INC," May 2014. [Online]. Available: http://www.subnet.com/resources/dictionary/intelligent-electronic-device.aspx.

[4] W. Bolton, "Programmable Logic Controllers", Fifth Edition, Newnes, 2009.

[5] "Siemens," 2016. [Online]. Available:

http://w3.siemens.com/smartgrid/global/en/products-systems-solutions.

[6] E. Wright, "Practical SCADA for Industry", Elsevier, 2003.

[7] "Siemens Industry,"2016. [Online]. Available:

http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/Protection/.

[8] M. Tiegelkamp, "IEC 61131-3:Programming Industrial Automation Systems", Springer, 2010.

[9] I. C. S. a. Tutorials, " Introduction to Industrial Control Networks," IEEE, 2012.

[10] "Industrial Automation," *IEEE Communications Surveys and Tutorials*, 2012.

[11] A.Boyer, "SCADA :Supervisory control and Data Acquisition", ISA-The Instrumentation ,Systems and Automation Society, 2004.

[12] E. Csanyi, "Electrical Engineering Portal," 2013. [Online]. Available:

http://electrical-engineering-portal.com/.

[13] "Supervisory Control and Data,"a Technical Information Bulletin 04-1 by NATIONAL COMMUNICATIONS SYSTEM, Virginia,USA, 2004.

[14] motionking, "37ZYJ-36ZY DC Gear Motor,"2016. [Online]. Available: http://www.motionking.com/Products/DC_Motors/37ZYJ-36ZY_gear_motor.htm.

[15] Siemens, S7-200 Programmable Controller system manual, 2005.

[16] Siemens, SIMATIC S7-200 RS-232/PPI Multi-Master Cable and S7-200 USB/PPI Multi-Master Cable data sheet, 2005 .

[17] Siemens, SIMATIC, “working with step 7 SIMATIC Getting Started, 2005.

[18] electrical-knowhow, "electrical-knowhow,"2012 [Online]. Available:
http://www.electrical-knowhow.com/2012/05/classification-of-electric-motors.html.

[19] Omron, "www.omron.com," 2016 [Online]. Available:
http://www.omron.com/ecb/products/pry/121/g2r_1.html.

[20] Ben M.Chen, "Electrical engineering Part 2 Application examples".

[21] W. p. LabVIEW Environment Basics, "National Instruments,"2016. [Online].
Available: http://www.ni.com/getting-started/labview-basics/environment.

[22] "How LabVIEW Uses I/O Servers White paper," 2012. [Online]. Available:
http://www.ni.com/white-paper/13865/en/.

Appendix A: S7-200 CPU specifications

A-1 SIMATIC S7-200 CPU 226 DC/DC/DC and CPU 226 AC/DC/Relay:

The features of CPU 226 DC/DC/DC and CPU 226 AC/DC/Relay are summarized below:

Description Order Number	CPU 226 DC/DC/DC 6ES7 216–2AD21–0XB0	CPU 226 AC/DC/Relay 6ES7 216–2BD21–0XB0
Physical Size		
Dimensions (W x H x D)	196 mm x 80 mm x 62 mm	196 mm x 80 mm x 62 mm
Weight	550 g	660 g
Power loss (dissipation)	11 W	17 W
CPU Features		
On-board digital inputs	24 inputs	24 inputs
On-board digital outputs	16 outputs	16 outputs
High speed counters (32 bit value) Total Single phase counters Two phase counters	 6 High-speed counters 6, each at 20 kHz clock rate 4, each at 20 kHz clock rate	 6 High-speed counters 6, each at 20 kHz clock rate 4, each at 20 kHz clock rate
Pulse outputs	2 at 20 kHz pulse rate	2 at 20 kHz pulse rate
Analog adjustments	2 with 8 bit resolution	2 with 8 bit resolution
Timed interrupts	2 with 1 ms resolution	2 with 1 ms resolution
Edge interrupts	4 edge up and/or 4 edge down	4 edge up and/or 4 edge down
Selectable input filter times	7 ranges from 0.2 ms to 12.8 ms	7 ranges from 0.2 ms to 12.8 ms
Pulse Catch	14 pulse catch inputs	14 pulse catch inputs
Time of Day Clock (clock accuracy)	2 minutes per month at 25° C 7 minutes per month 0° C to 55° C	2 minutes per month at 25° C 7 minutes per month at 0° C to 55° C
Program size (stored permanently)	4096 words	4096 words
Data block size (stored permanently): Stored permanently Backed by super capacitor or battery	2560 words 2560 words 2560 words	2560 words 2560 words 2560 words
Number of expansion I/O modules	7 modules	7 modules
Maximum digital I/O	256 points	256 points
Maximum analog I/O	32 inputs and 32 outputs	32 inputs and 32 outputs
Internal memory bits Stored permanently on power down Backed by super capacitor or battery	256 bits 112 bits 256 bits	256 bits 112 bits 256 bits
Timers total Backed by super capacitor or battery 1 ms 10 ms 100 ms	256 timers 64 timers 4 timers 16 timers 236 timers	256 timers 64 timers 4 timers 16 timers 236 timers
Counters total Backed by super capacitor or battery	256 counters 256 counters	256 counters 256 counters
Boolean execution speed	0.37 µs per instruction	0.37 µs per instruction
Move Word execution speed	34 µs per instruction	34 µs per instruction
Timer/Counter execution speed	50 µs to 64 µs per instruction	50 µs to 64 per µs instruction
Single precision math execution speed	46 µs per instruction	46 µs per instruction
Real math execution speed	100 µs to 400 µs per instruction	100 µs to 400 µs per instruction
Super capacitor data retention time	190 hours, typical, 120 hours minimum at 40° C	190 hours, typical, 120 hours minimum at 40° C

Description Order Number	CPU 226 DC/DC/DC 6ES7 216–2AD21–0XB0	CPU 226 AC/DC/Relay 6ES7 216–2BD21–0XB0
On-board Communication		
Number of ports	2 ports	2 ports
Electrical interface	RS-485	RS-485
Isolation (external signal to logic circuit)	Not isolated	Not isolated
PPI/MPI baud rates	9.6, 19.2, and 187.5 kbaud	9.6, 19.2, and 187.5 kbaud
Freeport baud rates	0.3, 0.6, 1.2, 2.4, 4.8, 9.6, 19.2, and 38.4 kbaud	0.3, 0.6, 1.2, 2.4, 4.8, 9.6, 19.2, and 38.4 kbaud
Maximum cable length per segment up to 38.4 kbaud 187.5 kbaud	 1200 m 1000 m	 1200 m 1000 m
Maximum number of stations Per segment Per network	 32 stations 126 stations	 32 stations 126 stations
Maximum number of masters	32 masters	32 masters
PPI master mode (NETR/NETW)	Yes	Yes
MPI connections	4 total, 2 reserved: 1 for PG and 1 OP	4 total, 2 reserved: 1 for PG and 1 OP
Cartridge Options		
Memory cartridge (permanent storage)	Program, Data, and Configuration	Program, Data, and Configuration
Battery cartridge (data retention time)	200 days, typical	200 days, typical
Power Supply		
Line voltage–permissible range	20.4 to 28.8 VDC	85 to 264 VAC 47 to 63 Hz
Input current CPU only/max load	150/1050 mA	40/160 mA at 240 VAC 80/320 mA at 120 VAC
In rush current (maximum)	10 A at 28.8 VDC	20 A at 264 VAC
Isolation (input power to logic)	Not isolated	1500 VAC
Hold up time (from loss of input power)	10 ms at 24 VDC	80 ms at 240 VAC, 20 ms at 120 VAC
Internal fuse, not user-replaceable	3 A, 250 V, Slow Blow	2 A, 250 V, Slow Blow
+5 Power for Expansion I/O (max)	1000 mA	1000 mA
24 VDC Sensor Power Output		
Voltage range	15.4 to 28.8 VDC	20.4 to 28.8 VDC
Maximum current	400 mA	400 mA
Ripple noise	Same as input line	Less than 1 V peak-to-peak (maximum)
Current limit	1.5 A Approx.	1.5 A Approx.
Isolation (sensor power to logic circuit)	Not isolated	Not isolated

Description Order Number	CPU 226 DC/DC/DC 6ES7 216–2AD21–0XB0	CPU 226 AC/DC/Relay 6ES7 216–2BD21–0XB0
Input Features		
Number of integrated inputs	24 inputs	24 inputs
Input type	Sink/Source (IEC Type 1)	Sink/Source (IEC Type 1)
Input Voltage		
Maximum continuous permissible	30 VDC	30 VDC
Surge	35 VDC for 0.5 s	35 VDC for 0.5 s
Rated value	24 VDC at 4 mA, nominal	24 VDC at 4 mA, nominal
Logic 1 signal (minimum)	15 VDC at 2.5 mA, minimum	15 VDC at 2.5 mA, minimum
Logic 0 signal (maximum)	5 VDC at 1 mA, maximum	5 VDC at 1 mA, maximum
Isolation (Field Side to Logic Circuit)		
Optical isolation (galvanic)	500 VAC for 1 minute	500 VAC for 1 minute
Isolation groups of	13 points and 11 points	13 points and 11 points
Input Delay Times		
Filtered inputs and interrupt inputs	0.2 to 12.8 ms, user-selectable	0.2 to 12.8 ms, user-selectable
HSC clock input rate		
Single Phase		
Logic 1 level = 15 to 30 VDC	20 kHz	20 kHz
Logic 1 level = 15 to 26 VDC	30 kHz	30 kHz
Quadrature		
Logic 1 level = 15 to 30 VDC	10 kHz	10 kHz
Logic 1 level = 15 to 26 VDC	20 kHz	20 kHz
Connection of 2 Wire Proximity Sensor (Bero)		
Permissible leakage current	1 mA, maximum	1 mA, maximum
Cable Length		
Unshielded (not HSC)	300 m	300 m
Shielded	500 m	50 m
HSC inputs, shielded	50 m	50 m
Number of Inputs ON Simultaneously		
40 ° C	24	24
55 ° C	24	24

Description Order Number	CPU 226 DC/DC/DC 6ES7 216–2AD21–0XB0	CPU 226 AC/DC/Relay 6ES7 216–2BD21–0XB0
Output Features		
Number of integrated outputs	16 outputs	16 outputs
Output type	Solid state–MOSFET	Relay, dry contact
Output Voltage		
Permissible range	20.4 to 28.8 VDC	5 to 30 VDC or 5 to 250 VAC
Rated value	24 VDC	–
Logic 1 signal at maximum current	20 VDC, minimum	–
Logic 0 signal with 10 K Ω load	0.1 VDC, maximum	–
Output Current		
Logic 1 signal	0.75 A	2.00 A
Number of output groups	2	3
Number of outputs ON (maximum)	16	16
Per group – horizontal mounting (maximum)	8	4/5/7
Per group – vertical mounting (maximum)	8	4/5/7
Maximum current per common/group	6 A	10 A
Lamp load	5 W	30 W DC/200 W AC
ON state resistance (contact resistance)	0.3 Ω	0.2Ω, maximum when new
Leakage current per point	10 µA, maximum	–
Surge current	8 A for 100 ms, maximum	7 A with contacts closed
Overload protection	No	No
Isolation (Field Side to Logic)		
Optical isolation (galvanic)	500 VAC for 1 minute	–
Isolation resistance	–	100 M Ω, minimum when new
Isolation coil to contact	–	1500 VAC for 1 minute
Isolation between open contacts	–	750 VAC for 1 minute
In groups of	8 points	4 points/5 points/7 points
Inductive Load Clamping		
Repetitive Energy dissipation ≤ 0.5 LI^2 x switching rate	1 W, all channels	–
Clamp voltage limits	L+ minus 48V	–
Output Delay		
Off to On (Q0.0 and Q0.1)	2 µs, maximum	–
On to Off (Q0.0 and Q0.1)	10 µs, maximum	–
Off to On (Q0.2 through Q1.7)	15 µs, maximum	–
On to Off (Q0.2 through Q1.7)	100 µs, maximum	–
Switching Frequency (Pulse Train Outputs)		
Q0.0 and Q0.1	20 kHz, maximum	1 Hz, maximum
Relay		
Switching delay	–	10 ms, maximum
Lifetime mechanical (no load)	–	10,000,000 open/close cycles
Lifetime contacts at rated load	–	100,000 open/close cycles
Cable Length		
Unshielded	150 m	150 m
Shielded	500 m	500 m

Table A-1: Specification for CPU 226 DC/DC/DC and CPU 226 AC/DC/Relay.

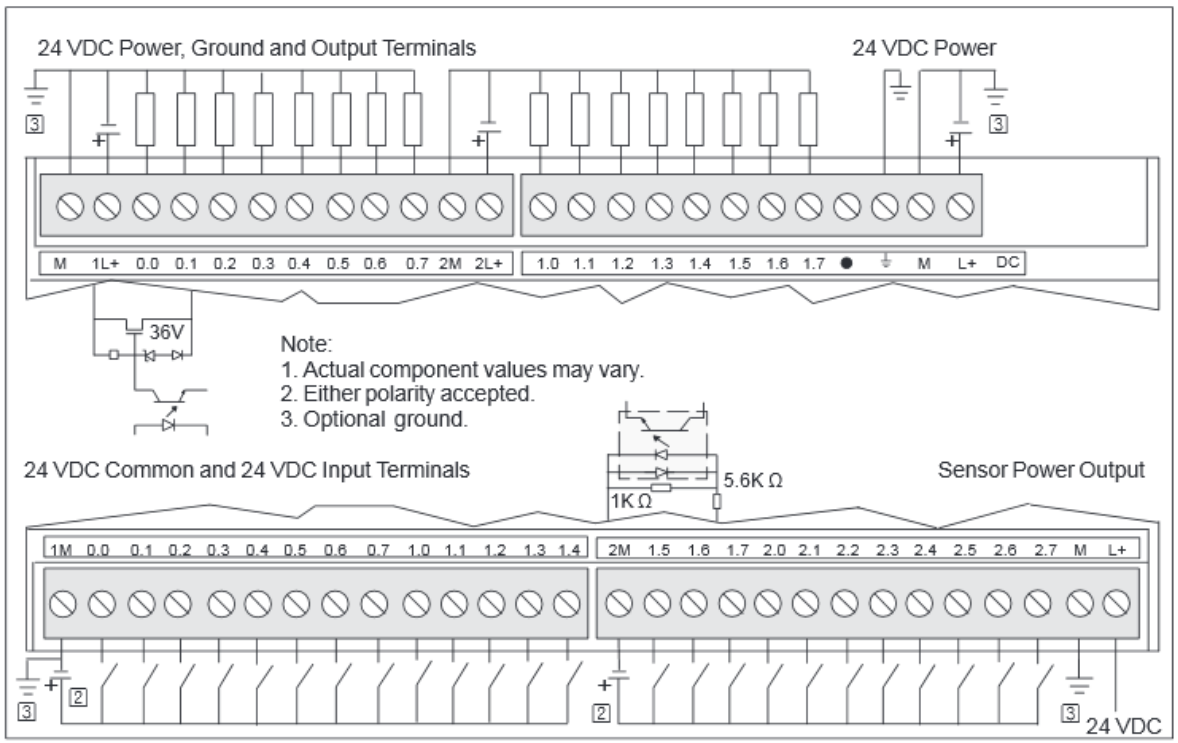


Figure A-1: Connector Terminal Identification for CPU 226 DC/DC/DC

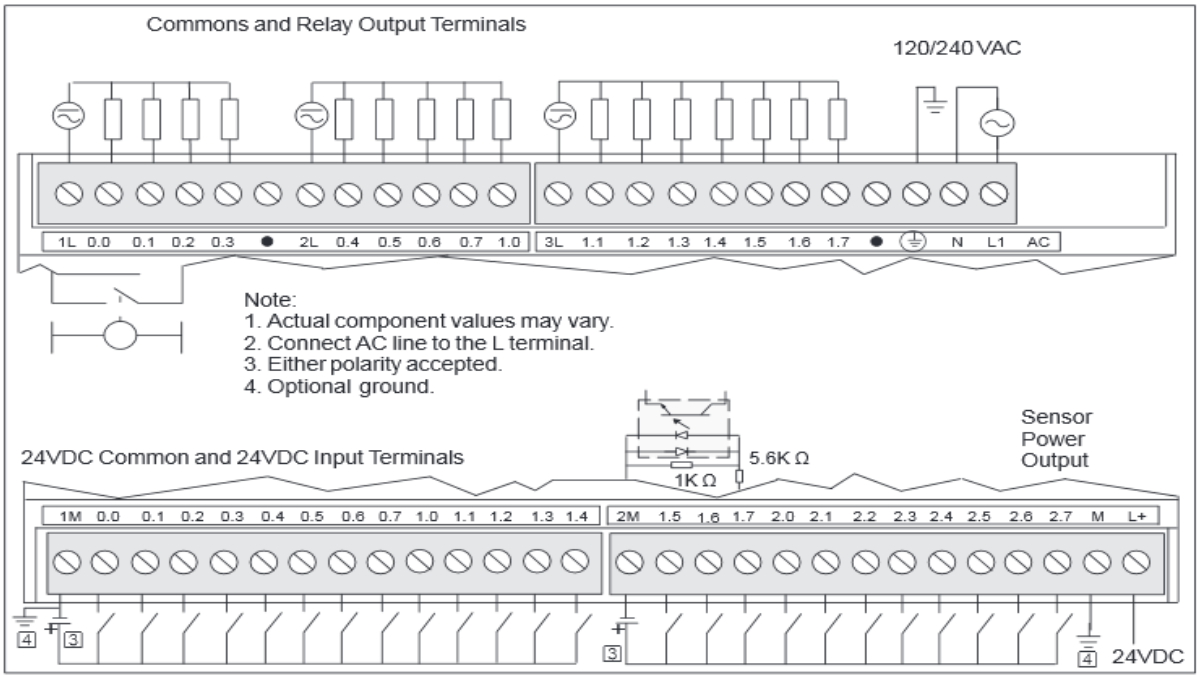


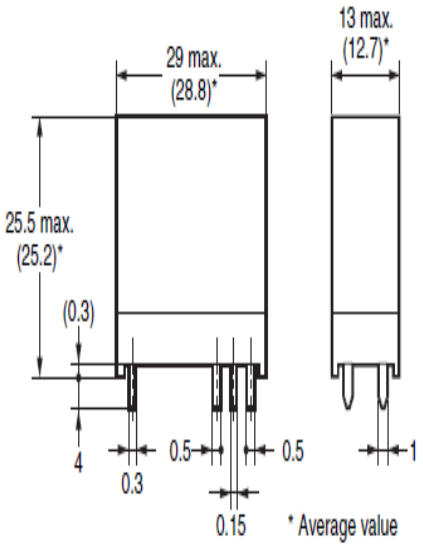
Figure A-2: Connector Terminal Identification for CPU 226 AC/DC/Relay

Appendix B: OMRON Relays data sheet

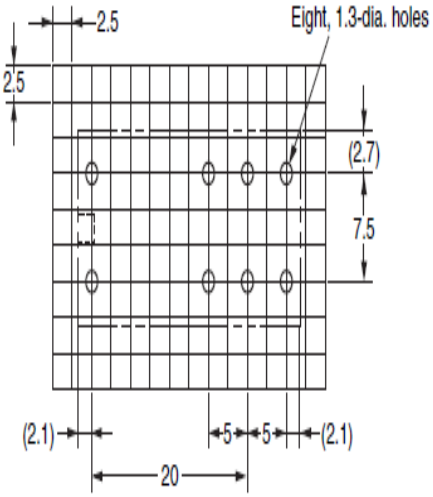
Model configuration:

Terminal Shape	Classification	Enclosure rating	Number of poles / Contact form	1-pole		2-pole		Minimum packing unit
				SPST-NO (1a)	SPDT (1c)	DPST-NO (2a)	DPDT (2c)	
PCB terminals	High-capacity	Flux protection	AC	G2R-1A-E	G2R-1-E	–	–	50 pcs/tray
			DC					

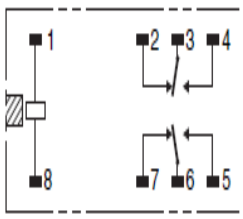
Shape, dimension and connection diagram:



PCB Mounting Holes
(BOTTOM VIEW)
Tolerance: ±0.1 mm



Terminal Arrangement/
Internal Connections
(BOTTOM VIEW)



(No coil polarity)

Appendix C: SIMATIC S7-200 RS-485&232/PPI Multi-Master Cable and S7-200 USB/PPI Multi-Master Cable specification

Table 1 Specifications

Description Order Number	S7-200 RS-232/PPI Multi-Master Cable 6ES7 901-3CB30-0XA0	S7-200 USB/PPI Multi-Master Cable 6ES7-901-3DB30-0XA0
General Characteristics		
Supply voltage	14.4 to 28.8 VDC	14.4 to 28.8 VDC
Supply current at 24 V nominal supply	60 mA RMS max.	50 mA RMS max.
Direction change delay: RS-232 stop bit edge received to RS-485 transmission disabled	-	-
Isolation	RS-485 to RS-232: 500 VDC	RS-485 to USB: 500 VDC
RS-485 Side Electrical Characteristics		
Common mode voltage range	-7 V to +12 V, 1 second, 3 V RMS continuous	-7 V to +12 V, 1 second, 3 V RMS continuous
Receiver input impedance	5.4 K Ω min. including termination	5.4 K Ω min. including termination
Termination/bias	10K Ω to +5 V on B, PROFIBUS pin 3 10K Ω to GND on A, PROFIBUS pin 8	10K Ω to +5 V on B, PROFIBUS pin 3 10K Ω to GND on A, PROFIBUS pin 8
Receiver threshold/sensitivity	+/-0.2 V, 60 mV typical hysteresis	+/-0.2 V, 60 mV typical hysteresis
Transmitter differential output voltage	2 V min. at R_L=100 Ω, 1.5 V min. at R_L=54 Ω	2 V min. at R_L=100 Ω, 1.5 V min. at R_L=54 Ω
RS-232 Side Electrical Characteristics		
Receiver input impedance	3K Ω min.	-
Receiver threshold/sensitivty	0.8 V min. low, 2.4 V max. high 0.5 V typical hysteresis	-
Transmitter output voltage	+/- 5 V min. at R_L=3K Ω	-
USB Side Electrical Characteristics		
Full speed (12 MB/s), Human Interface Device (HID)		
Supply current at 5V	-	50 mA max.
Power down current	-	400 uA max.

Table.C.1. Specification

Table 2 S7-200 RS-232/PPI Multi-Master Cable – Pin-outs for RS-485 to RS-232 Local Mode Connector

RS-485 Connector Pin-out		RS-232 Local Connector Pin-out	
Pin Number	Signal Description	Pin Number	Signal Description
1	No connect	1	Data Carrier Detect (DCD) (not used)
2	24 V Return (RS-485 logic ground)	2	Receive Data (RD) (output from PC/PPI cable)
3	Signal B (RxD/TxD+)	3	Transmit Data (TD) (input to PC/PPI cable)
4	RTS (TTL level)	4	Data Terminal Ready (DTR)[1]
5	No connect	5	Ground (RS-232 logic ground)
6	No connect	6	Data Set Ready (DSR)[1]
7	24 V Supply	7	Request To Send (RTS) (not used)
8	Signal A (RxD/TxD–)	8	Clear To Send (CTS) (not used)
9	Protocol select	9	Ring Indicator (RI) (not used)

Table C.2. Pin out for RS-485 to RS 232 Local Mode Connector

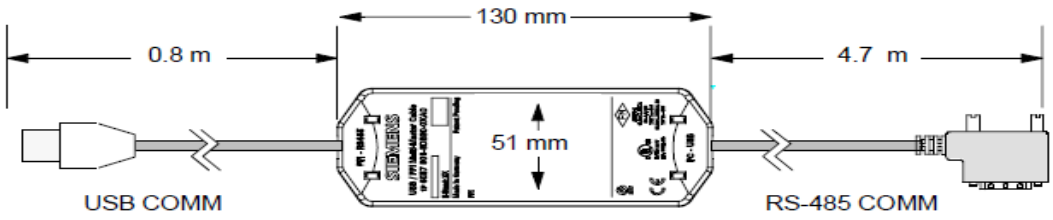
Table 3 S7-200 RS-232/PPI Multi-Master Cable – Pin-outs for RS-485 to RS-232 Remote Mode Connector

RS-485 Connector Pin-out		RS-232 Remote Connector Pin-out[1]	
Pin Number	Signal Description	Pin Number	Signal Description
1	No connect	1	Data Carrier Detect (DCD) (not used)
2	24 V Return (RS-485 logic ground)	2	Receive Data (RD) (input to PC/PPI cable)
3	Signal B (RxD/TxD+)	3	Transmit Data (TD) (output from PC/PPI cable)
4	RTS (TTL level)	4	Data Terminal Ready (DTR)[2]
5	No connect	5	Ground (RS-232 logic ground)
6	No connect	6	Data Set Ready (DSR)[2]
7	24 V Supply	7	Request To Send (RTS) (output from PC/PPI cable)
8	Signal A (RxD/TxD–)	8	Clear To Send (CTS) (not used)
9	Protocol select	9	Ring Indicator (RI) (not used)

[1] A conversion from female to male, and a conversion from 9-pin to 25-pin is required for modems.
[2] Pins 4 and 6 are connected internally.

Table C.3. Pin out for RS-485 to RS 232 Remote Mode Connector

Figure .C.1shows the S7-200 USB/PPI Multi-Master Cable dimensions and LEDs.



LED	Color	Description
Tx	Green	USB transmit indicator
Rx	Green	USB receive indicator
PPI	Green	RS-485 transmit indicator

Figure .C.1. S7-200 USB/PPI Multi-Master Cable dimensions and LEDs.