CHAPTER 1

INTRODUCTION

This chapter includes introduction to water treatment system, system overview, scopes of thesis, aim and objectives of the system and outlines of thesis.

1.1 Introduction of Water Treatment System

Water treatment systems play a critical role in ensuring the safety and quality of water supplied to communities. The integration of Programmable Logic Controllers (PLCs) into these systems have revolutionized to monitor and control water treatment processes.

A well-designed water treatment system is crucial for providing clean and safe water. Water treatment system encompasses the various stages, such as coagulation, flocculation, sedimentation, and filtration, each requiring precise control and monitoring to ensure optical performance. This thesis is aimed to design and implement a working model of water treatment system with Delta PLC for process control, water level sensors and Human Machine Interface (HMI) to monitor water levels. Water Treatment System is shown in Figure 1.1 [1].

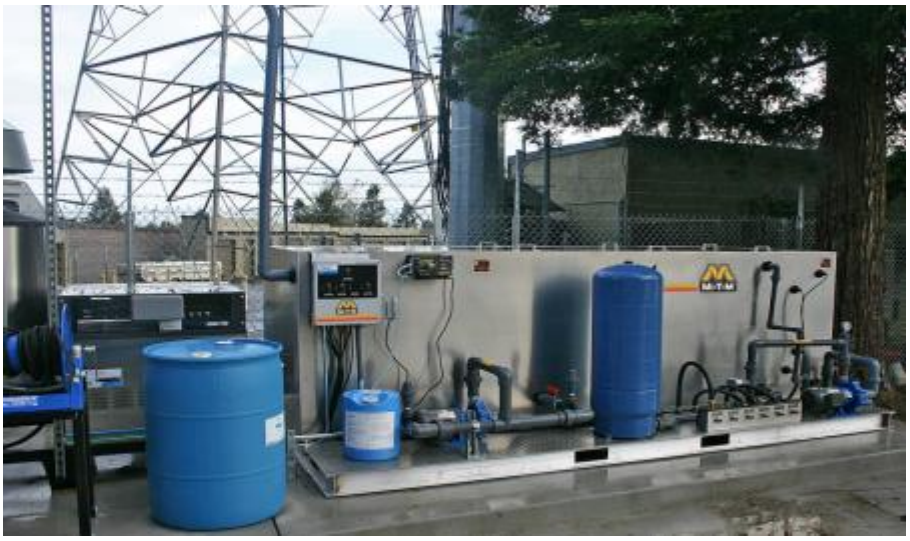
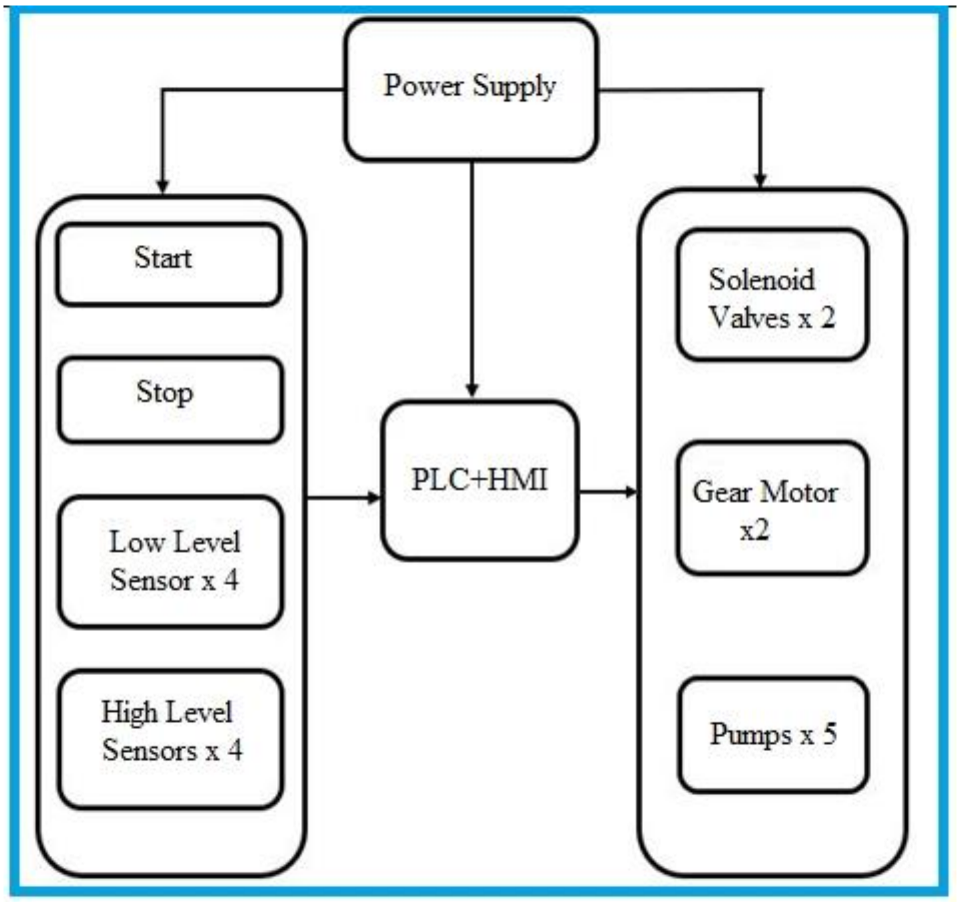


Figure 1.1 Water Treatment System [1]

1.2 System Overview of Water Treatment System

The working principle of water treatment system is that the PLC adjusts the mixing intensity and duration based on input from the level sensor and pre-programmed settings. The mixing pump is regulated to ensure effective coagulation and flocculation without breaking the formed flocs. The PLC acts as the central control unit, receiving inputs from level sensors and controlling pumps across all stages. The HMI provides a user-friendly interface for operators to monitor and control the entire water treatment process. It offers real-time data visualization, system status updates, and alerts for any anomalies. The block diagram of the water treatment system is shown in Figure 1.2.


Power Supply
Start
Stop
Low Level Sensor x 4
High Level Sensors x 4
PLC+HMI
Solenoid Valves x 2
Gear Motor x2
Pumps x 5

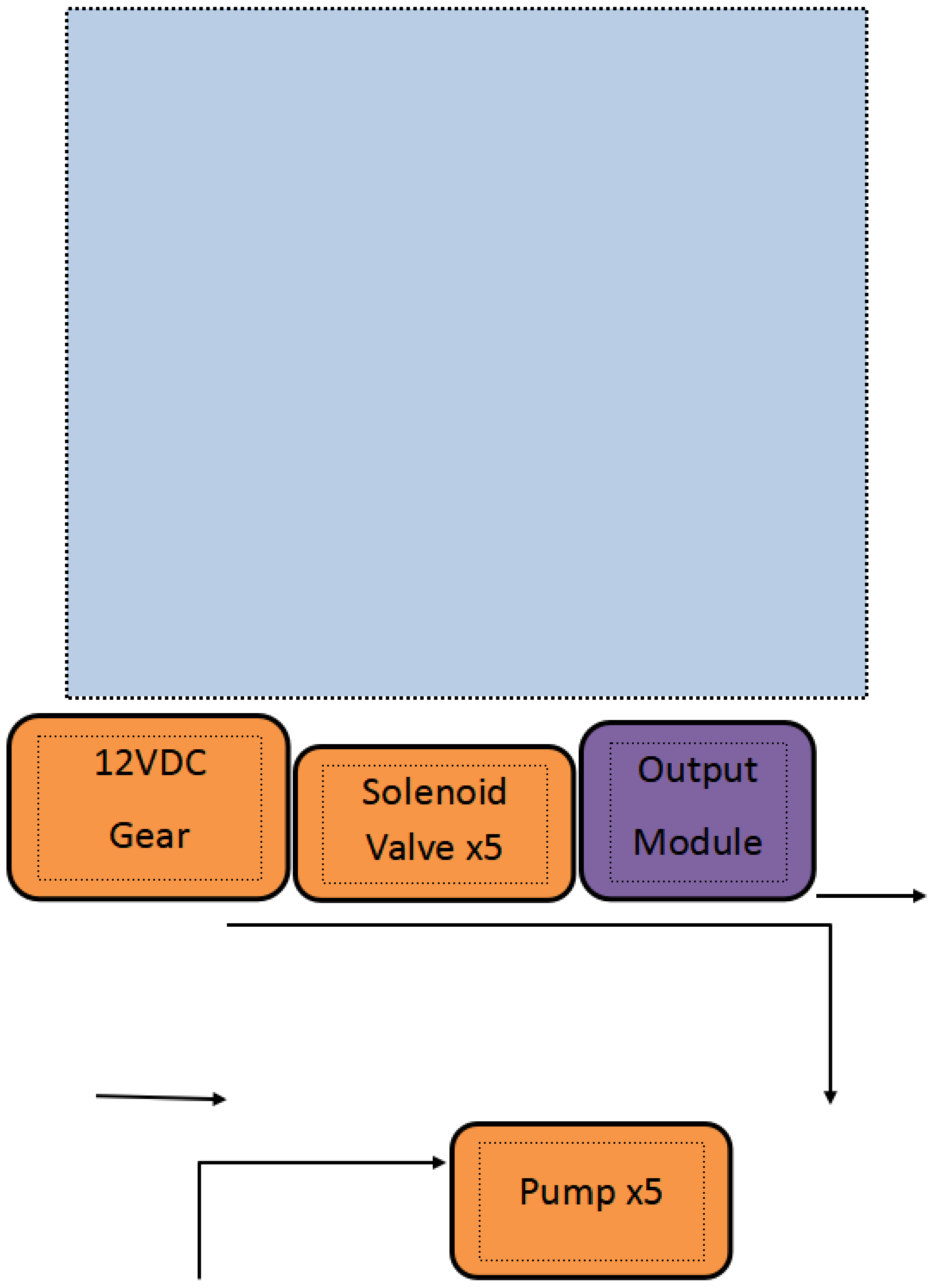

12VDC Gear
Solenoid Valve x5
Output Module
Pump x5

Figure 1.2 Block Diagram of Water Treatment System

1.3 Aim and Objectives

The aim of the thesis is to design and simulate systems that effectively and efficiently produce clean water for real-world applications.

The objectives of the thesis are as follows:

- To optimize the water flow and power distribution within the system
- To develop PLC programs for controlling pumps, valves, and other devices
- To conduct experiments to validate the system functionality
- To develop an HMI to monitor real-time data from sensors
- To know how to write the program by using WPLSoft software
- To understand the advantages of PLC

1.4 Scopes of the Thesis

This thesis focuses on the development and implementation of a water treatment process using automation technologies. The main functions of this system are checking the water levels in each stage with level sensors, and controlling the pump outputs according the controlled program. The level of turbidity is tested with turbidity sensor and pH value is checked with pH sensor and the results are shown on Human Machine Interface (HMI).

1.5 Outlines of the Thesis

Chapter one is overview introduction of thesis. Chapter two represents the background theory of PLCs and the characteristics such as operation principle, memory unit, input and output module and introduction of Delta PLC and introduction of WPL Soft software. Chapter three includes system operation and implementation and system software tool of WPL and the contacts and function block characteristics that use in the water treatment system. Chapter four is test and results of the program. Chapter five includes discussion, conclusion and further extension of the thesis.

CHAPTER 2

BACKGROUND THEORY

This chapter includes introduction to programmable logic controller (PLC) and components, introduction of Delta PLC and WPL Soft software.

2.1 Introduction to Programmable Logic Controller

A programmable logic controller (PLC) is a specialized computer used to control machines and processes. It is an industrial computer designed to automate control processes. It uses a combination of hardware and software to monitor inputs, process logic, and control outputs in real-time. PLCs are widely used in industries such as manufacturing, water treatment, and energy management to enhance efficiency and reliability in automated systems. It therefore shares common terms with typical PCs like central processing unit (CPU) and memory. Modern PLCs offer advanced features like networking capabilities, data logging, and remote access, enabling seamless integration with other control systems and supervisory software. For the input, PLC had two types of input that is digital inputs and analog inputs. Digital inputs include push-buttons, limit switches, relay contacts, proximity switches, photo-sensors (On/Off) and pressure switches. For analog inputs, electric values act as analog inputs such as level sensors, thermocouples and flow sensors. The examples of hardware PLC control are shown in Figure 2.1 [2].

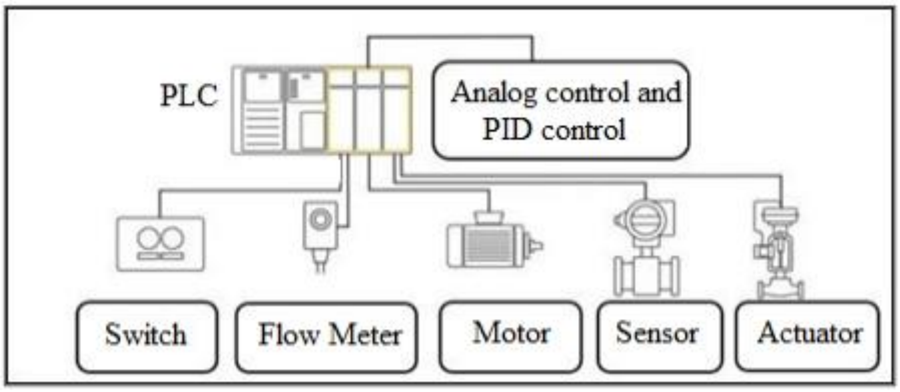


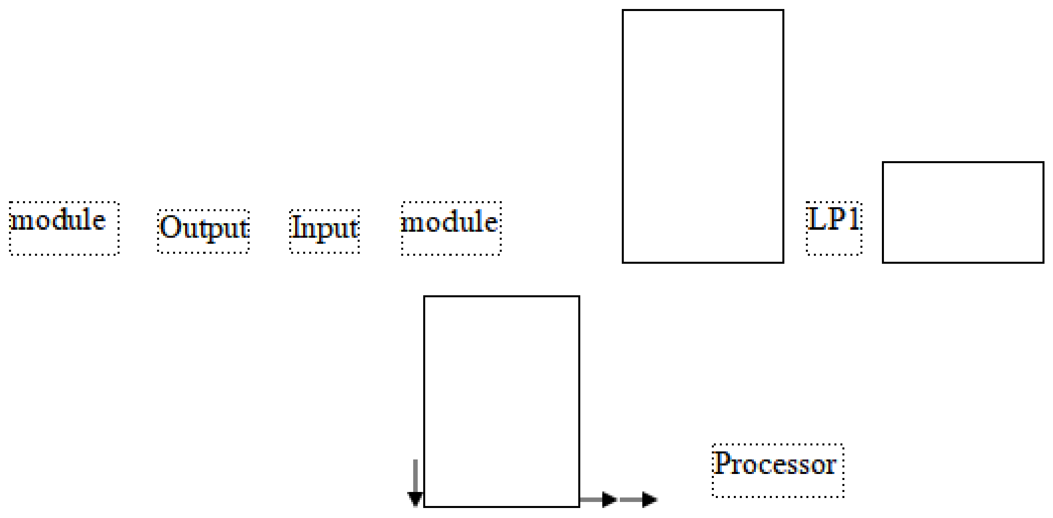
Figure 2.1 Examples of Hardware PLC Control [2]

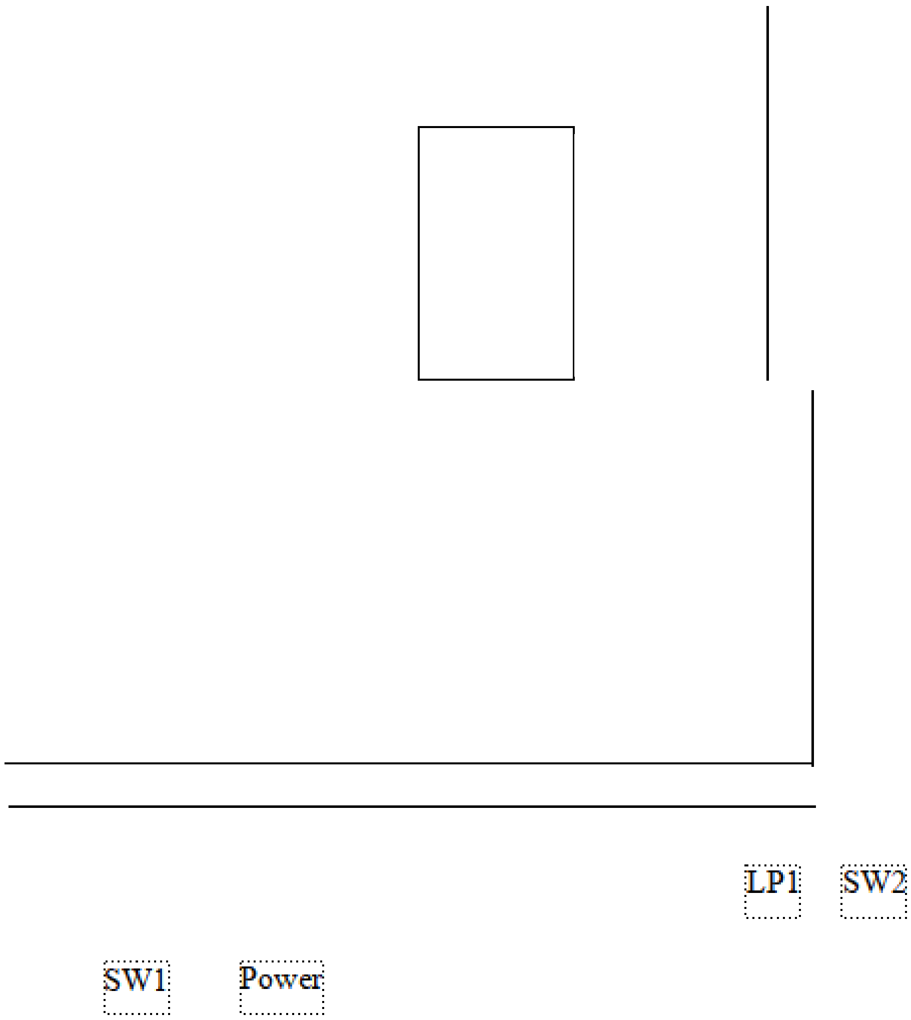
2.2 Components of Programmable Logic Controller

Programmable controllers have grown throughout industrial control applications because of the ease of bringing to create a controller, ease of programming, ease of wiring, ease of installation and ease of changing. PLCs span a wide range of sizes, but all contain five basic components.

- processor or central processing unit (CPU)
- input and output module
- memory unit
- power supply
- programming unit, devices or PCs/software

The major component of PLC is shown in Figure 2.2 [3].





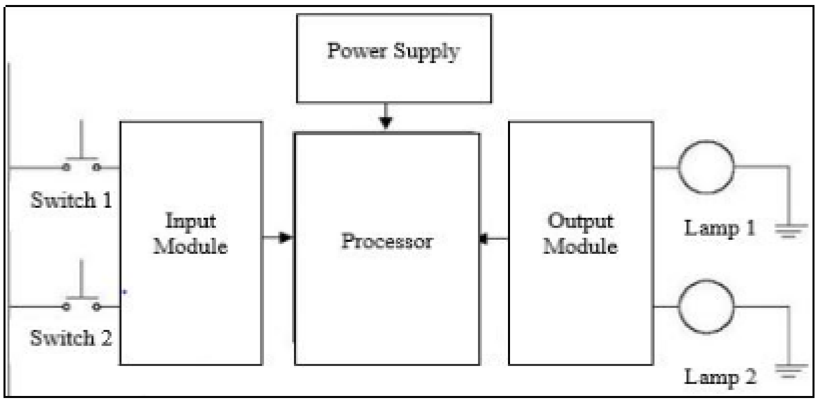


Figure 2.2 Major Components of PLC [3]

2.2.1 Processor or Central Processing Unit (CPU)

The CPU operates in a cyclic manner, known as the scan cycle, which includes reading inputs, executing the program, and updating outputs. Its speed and processing power determine how fast the PLC can react to changes in the system. Additionally, they are designed to handle various communication protocols, making it easy to integrate with other devices and systems for complex automation tasks.

The CPU operation is explained with the following steps. In the first step, Fetch Instruction, the PLC retrieves the next command from its memory, such as ladder logic or structured text. Then, in the Decode Instruction phase, it interprets the command to determine the specific operation, whether it's reading an input, controlling an output, or performing an internal function like a calculation. The PLC performs the required operation in the ALU Operation stage. It then accesses necessary memory locations in the Access Memory phase to store or retrieve data. The results are updated in the memory during the Update Register File stage and finally the Update PC phase adjusts the program counter to the next instruction. CPU instruction execution steps are shown in Figure 2.3 [4].

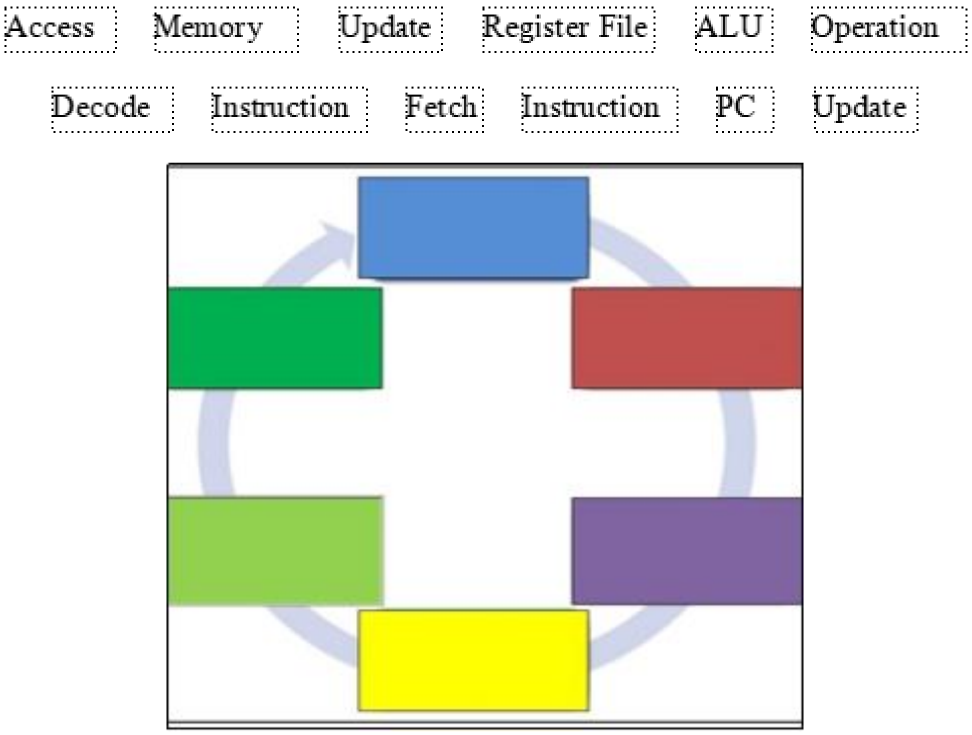


Figure 2.3 CPU Instruction Execution Steps [4]

2.2.2 Working Principle of Input and Output Modules

The type of input modules used by a PLC depends on the type of input device. For example, some respond to digital inputs, which are either on or off while others respond to analog signals. In this case, analog signals represent machine or process conditions as a range of voltage or current values. The PLC input circuitry converts signals into logic signals that the CPU can use.

Output modules convert control signals from the CPU into digital or analog values that can be used to control various output devices. The programming device is used to enter or change the PLCs program or to monitor or change stored values. In addition to these basic elements, a PLC system is also incorporate an operator interface device to simplify monitoring of the machine or process [5].

2.2.3 Memory Unit in PLC

PLC memory consists of the operating system and firmware of the processor and modules, and the program and data that is used by the programmer. There are volatile and non-volatile areas of memory, and that the volatile part of memory needs a battery, super-capacitor or other rechargeable energy storage module to hold its program or data.

Though the program is saved on Flash or SD RAM cards without a battery, the data exchange rate is too slow to use this for the actual interfacing of the program with its data. When the PLC is powered on, the program is loaded from non-volatile RAM cards into the user memory of the controller. Not all PLC platforms back up the user memory with a battery or other energy storage device, data memory may be lost when a processor loses power. Some platforms, however, ensure that the data is kept intact even when power is lost by use of battery backed RAM. This means that the values in data registers will be retained and the program will start in its last state.

Other PLC platforms assign some parts of RAM to be retentive and other parts non-retentive. Omron separates its retentive bits into holding relays and non-retentive CIO, and its data into the retentive DM Area and non-retentive Work Area, Siemens allows its general marker memory to be assigned as retentive or non-retentive and defaults to only 16 bytes of retentive marker memory. Siemens data blocks however are retentive unless defined not to be. Allen-Bradley’s memory is all retentive.

The operating system itself on a processor is held in non-volatile memory, called firmware. To change the firmware on a PLC a Flash program or tool needs to be used to download it. This is usually included with the programming software.

I/O, communications and other modules also often have firmware built in. The firmware update tools can also update these modules and the firmware is usually available from the manufacturer’s website. It is necessary to have software that is at least as up-to-date as the firmware being installed.

The RAM part of memory in a PLC is separated into two general areas; Program memory and Data memory. Program memory consists all of the lists of instructions and program code. This is what is sent to the processor. The act of sending the program instructions to the PLC is called downloading on most brands of PLC, however this may differ on some platforms.

Data memory includes the input and output image tables as well as Numerical and Boolean Data. It would be find that most of the data used in the PLC program is internal memory and not directly related to I/O.

As the program executes and keeps track of whether bits (BOOLS) are on or off and the values of numbers in Data Memory. Different platforms have different ways of organizing data [6].

2.2.4 Power Supply of PLC

A PLC without a power supply is much like a car without gas, or a laptop without a battery. It is the fuel for the PLC. How powerful a PLC is will greatly depend on how powerful the power supply. First and foremost, the PLC’s power supply converts a line voltage, commonly 120 or 240 volts AC, or alternating current, into a useable DC, or direct current, voltage, commonly 24 volts, to power on the PLC and its components. Line voltage is stepped down with a transformer, rectified to convert it to DC, filtered capacitors, and protected during this process. Not all PLC platforms back up the user memory with a battery or other energy storage device, data memory may be lost when a processor loses power. Some platforms however, ensure that the data is kept intact even when power is lost by use of battery backed RAM. This means that the values in data registers is retained and the program will start in its last state.

A PLC’s power supply is the workhorse of the PLC system. It converts electrical energy from a source into the correct voltage, current, and frequency required to power a device or system. In some systems the power supply provides the power for all of these components through a bus system in the rack. In other systems a technician must have to wire these components individually throughout the rack. PLC’s power supplies are come in different sizes and power ratings depending on the PLC.

Earlier, the common output voltage is 24 volts DC on a PLC’s power supply. The different sizes are going reference current and be rated in amps or amperes. The common current ratings for PLC’s is anywhere from 2 to 10 amps for smaller systems and up to 50 amps for larger, more powerful controllers.

This is an important rating for engineers and maintenance personnel to consider when designing a system or even modifying one. The current rating will directly affect how much work your PLC system is actually done. While the main power supply is the primary source of power for the PLC, there is usually a battery backup as well.

This will provide energy to the memory of the PLC in case of a power supply failure or power outage in general. Replacing the batteries in a PLC is a common preventative maintenance procedure. A PLC’s power supply is the workhorse of the PLC system. It converts line voltage, 120 or 240 volts AC, to a lower DC voltage, commonly 24 volts DC. This DC voltage is then sent into the rack to power the rest of the PLC components. There are many sizes of PLC power supply available. Those sizes are a current rating in amps or amperes [7].

2.2.5 Programming Device

The PLC is programmed using a specialty programmer or software on a computer that load and change the logic inside. Most modern PLCs are programmed using software on a PC or laptop computer. Older systems used a custom programming device. PLC programs are typically written in a special application on a personal computer and downloaded by a direct-connection cable or over a network to the PLC. PLCs can be programmed using standards-based programming languages. IEC 1131-3 is the international standard for programmable controller programming languages.

Most recently, PLCs are programmed using application software on personal computers, which now represent the logic in graphic form instead of character symbols.

The computer is connected to the PLC through USB, Ethernet, RS-485 or RS-422 cabling. The programming software allows entry and editing of the ladder-style logic. The following is a list of programming languages specified by this standard:

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

One of the primary benefits of the standard is that it allows multiple languages to be used within the same programmable controller. This allows the program developer to select the language best suited to each particular task.

Ladder logic is the main programming method used for PLC's. As mentioned before, ladder logic has been developed to mimic relay logic. The decision to use the relay logic diagrams was a strategic one. By selecting ladder logic as the main programming method, the amount of retraining needed for engineers and trades people was greatly reduced. The first PLC was programmed with a technique that was based on relay logic wiring schematics. This eliminated the need to teach the electricians, technicians and engineers how to program. This programming method has stuck and it is the most common technique for programming in today PLC.

There are other methods to program PLCs. One of the earliest techniques involved mnemonic instructions. These instructions is derived directly from the ladder logic diagrams and entered into the PLC through a simple programming terminal. A ladder diagram is a method of drawing control circuit which pre-dates PLCs. The ladder diagram resembles the schematic diagram of a system built with electromechanical relays.

SFC have been developed to accommodate the programming of more advanced systems. These are similar to flowcharts, but much more powerful. This method is much different from flow charts because it does not have to follow a single path through the flowchart.

Programming has been developed as a more modern programming language. It is quite similar to languages such as BASIC and Pascal. Structured Text (ST) is a high level textual language that is block structured and syntactically resembles Pascal, on which it is based. All of the languages share IEC61131 Comment Elements. It is very flexible and intuitive for writing control algorithms. FBD is another graphical programming language.

There is no such as software which is widely used. It is depended upon the requirements of the users. There are many programming logic controller software which are used in industry according to the need and mechanism of the product. The different types of PLC are as follow [8].

- Siemens S7 PLC- TIA Portal, Simatic Manager for S7 controller
- Delta PLC - WPL Software
- Allen Bradely - RS Logix

- ABB - Automation Builder
- Mitsubishi - MELSEC
- Schneider - Twido Suit
- B & R -Automation Studio
- Omron - CX Programmer
- Honeywell - Control Edge
- Kinco - KincoBuilder

2.2.6 Communication of PLC to PLC and PLC to Other Devices

Many models of PLCs have built-in communications ports, using RS-232, RS-422, RS-485, or Ethernet. Many of these protocols are specific. Most modern PLCs communicate over a network, such as a computer running a SCADA system. PLC-to-PLC communication is typically achieved through industrial protocols such as Modbus, Profibus, or Ethernet/IP, allowing multiple PLCs to share data and coordinate processes. This communication ensures that complex systems with distributed control functions work in sync, improving efficiency and reducing response times. PLC-to-PLC communication is typically achieved through industrial protocols such as Modbus, Profibus, or Ethernet/IP, allowing multiple PLCs to share data and coordinate processes.

PLCs used in larger I/O systems have peer-to-peer (P2P) communication between processors. This allows separate parts of a complex process to have individual control while allowing the subsystems to co-ordinate over the communication link.

These communication links are also often used for HMI devices such as keypads or PC-type workstations [9].

2.3 Delta DVP-16EC

Delta DVP-16EC is belonged to the DVP-EC3 series and it’s known for its reliability and flexibility in control and monitoring tasks. DVP-16EC operates on a voltage of 24V DC. It comes with 8 digital inputs and 8 digital outputs. This model is equipped with built-in RS-232 and RS-485 communication port which support the modbus RTU protocol. DVP-16EC can be expanded using extension modules for complex project requirements. Delta DVP-16EC is shown in Figure 2.4 [5].

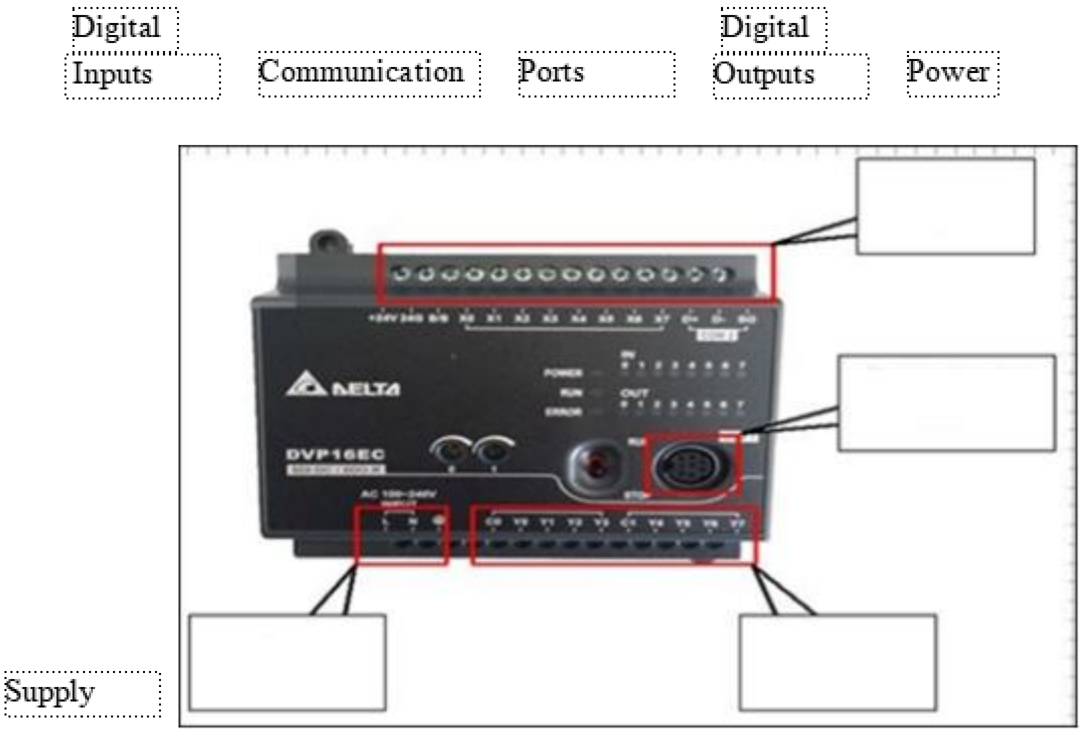


Figure 2.4 Delta DVP-16 EC [5]

2.4 WPL Soft Software

Delta PLC is used WPL Soft programming software and the same instructions, same as DVP series. WPL Soft is the programming software for Delta-DVP series PLC, and it is a user-friendly and high-efficient development system with the powerful functions.

2.4.1 Ladder Diagram Programming

Ladder Diagram (LD) is one of the most used graphical languages. LD language is based on the relay ladder logic. In addition, the IEC LD language allows the use of user defined function blocks and functions and is used in a hierarchical design. Moreover, the modular structure of ladder diagrams facilitates easy troubleshooting and system modifications, making it an efficient tool for designing complex automation systems. A sample ladder diagram is shown in Figure 2.5.

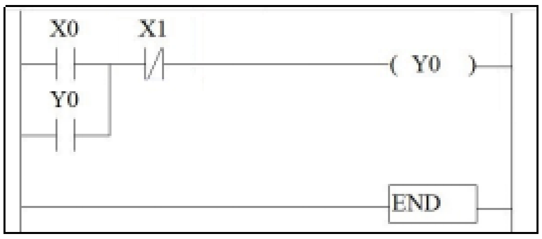


Figure 2.5 Sample Ladder Diagram

2.4.2 Offline Simulation

To simulate the network, the following description is needed.

- Select the simulator command
- Click the icon write to PLC on the toolbar
- Click the icon RUN command on the toolbar

Offline simulation of ladder program is shown in Figure 2.6.

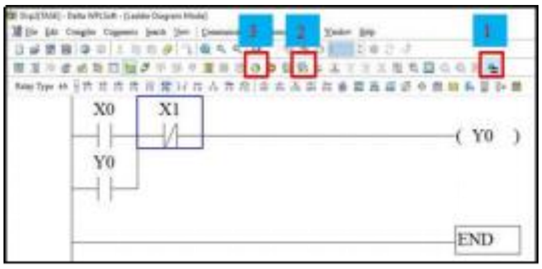


Figure 2.6 Steps of Ladder Program Simulation in WPL Soft

2.4.3 Ladder Diagram Network

When a program in ladder diagram is written, a network of logic is constructed. Ladder Diagram network is delimited on the left by a vertical line known as the left power rail, and on the right by a vertical line known as right power rail. No state is defined for the right rail. Ladder diagram instruction is shown in Figure 2.7.

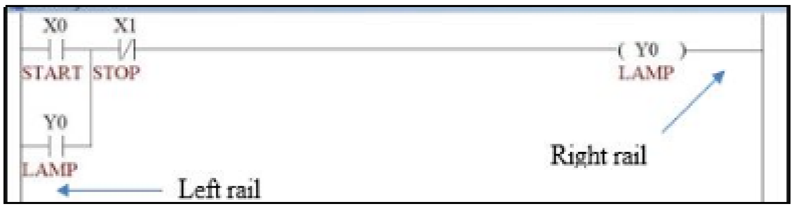
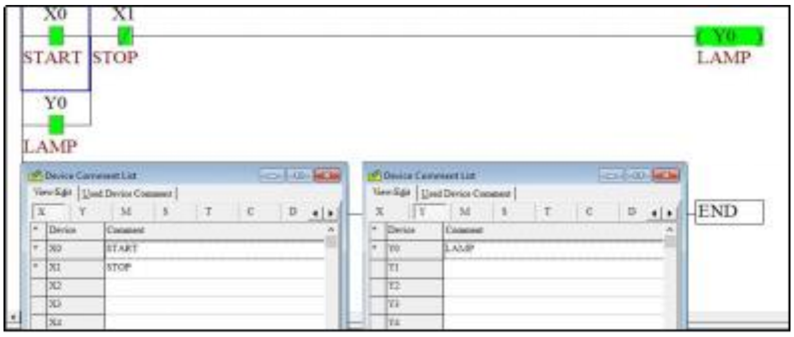


Figure 2.7 Network of LD Instruction

2.4.4 Comments Dialog Box

To open the comment dialog box, click on the device comment list icon on the tool bar. After clicking the icon, some comments are written in view/edit column in comment dialog box. Comments dialog box is shown in Figure 2.8.



Device	Comment
X0	START
X1	STOP
X2	
X3	
X4	

Device	Comment
Y0	LAMP
Y1	
Y2	
Y3	
Y4	

Figure 2.8 Comments Dialog Box in the Network Instruction

2.4.5 Functions and Function Blocks

A function or function blocks are represented with a rectangular block, and its actual variable connections are shown by writing the appropriate variable outside the block adjacent to the formal variable name on the inside. Function blocks also enhance code readability by encapsulating specific logic into manageable units, reducing duplication and errors. They can store internal states, making them ideal for tasks like controlling timers, counters, and motor control loops. Additionally, their standardized format ensures compatibility across different platforms and systems, promoting ease of integration in various automation environments. The example function block is shown in Figure 2.9.

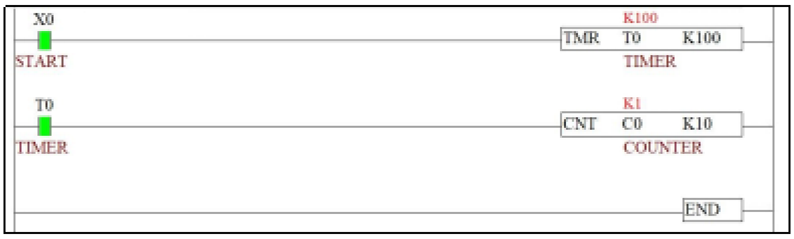
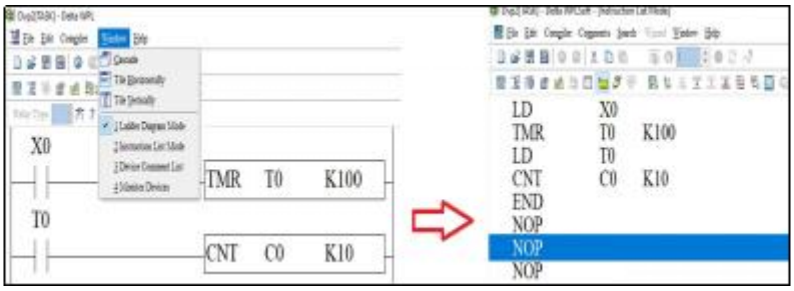


Figure 2.9 Function and Function Block

2.4.6 Switching from Ladder Diagram (LD) to Instruction List (IL) in WPL Soft

In WPL Soft, ladder diagram mode and instruction list mode can be switched from each other in the window toolbar. This switching mode is shown in Figure 2.10.



```
LD    X0
TMR   T0   K100
LD    T0
CNT   C0   K10
END
NOP
NOP
NOP
```

Figure 2.10 Switching from Ladder Diagram Mode to Instruction List Mode

CHAPTER 3

SOFTWARE IMPLEMENTATION OF THE SYSTEM

This chapter includes system operation, flow chart, introduction to ladder diagram programming of WPL Soft software, user interface and function blocks of TMR (Timer), MODRD (Modbus Read), MOV (Move to), CMP (Comparator) and inputs/outputs assignment of the program.

3.1 System Operation of Water Treatment Process

The water treatment process is designed to be fully automated, eliminating the need for manual intervention. This automation ensures that the entire process, including pump control and various treatment stages. When the system starts, the HMI immediately displays all relevant information and provides a comprehensive interface for monitoring of the entire water treatment process.

The water treatment process is designed to purify water through a series of stages, each based on distinct principles to effectively remove contaminants. The process is divided into five key stages: coagulation, flocculation, sedimentation, filtration and distribution. Each stage plays a crucial role in achieving the desired water quality. Water treatment process is shown in Figure 3.1 and the flow chart of the water treatment process is shown in Figure 3.2.

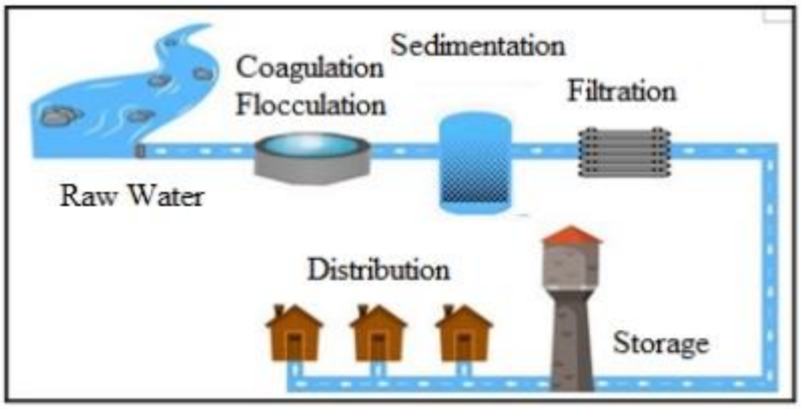


Figure 3.1 Water Treatment Process

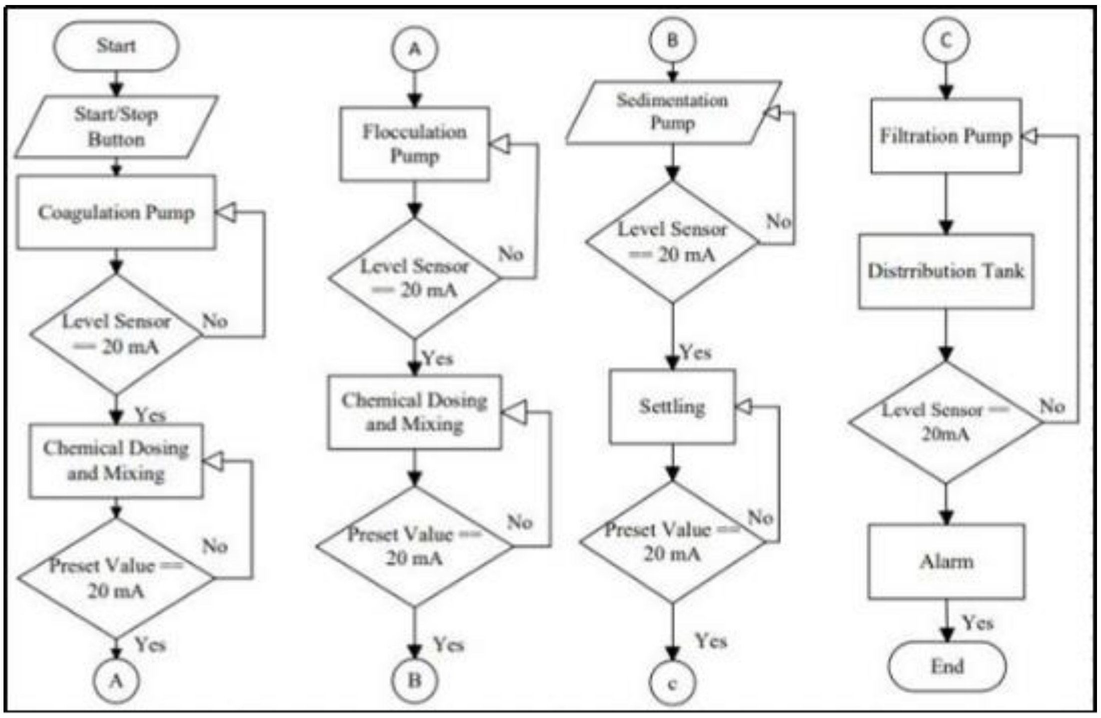


Figure 3.2 Flow Chart of the Water Treatment Process

3.1.1 Coagulation Stage

In the coagulation stage of the water treatment process, aluminum sulfate is used as the coagulant to destabilize and aggregate suspended particles in the water. Alum is chosen for its efficiency and safety, as it effectively neutralizes particle charges and facilitates floc formation while posing minimal health risks. To achieve optimal coagulation, the water is subjected to high-speed mixing with a gear motor. This vigorous agitation ensures thorough dispersion of the alum and promotes the collision and aggregation of particles into larger flocs. The high-speed mixing is crucial for the effectiveness of the coagulation process, setting the stage for successful sedimentation and subsequent treatment stages.

3.1.2 Flocculation Stage

The flocculation process is a crucial stage in water treatment, where fine suspended particles that do not settle during coagulation are agglomerated into larger flocs. This is achieved by gently stirring the water to promote collisions between the particles. In this prototype, the flocculation stage is controlled by the Delta PLCs, which regulate the mixing speed and duration based on predefined parameters. The process is carefully monitored through sensors, ensuring optimal floc formation, which is essential for the subsequent sedimentation stage. The efficiency of flocculation directly impacts the clarity and quality of the treated water. Coagulation and flocculation stages are used primarily in the treatment stages. Coagulation stage and flocculation stage is shown in Figure 3.3.

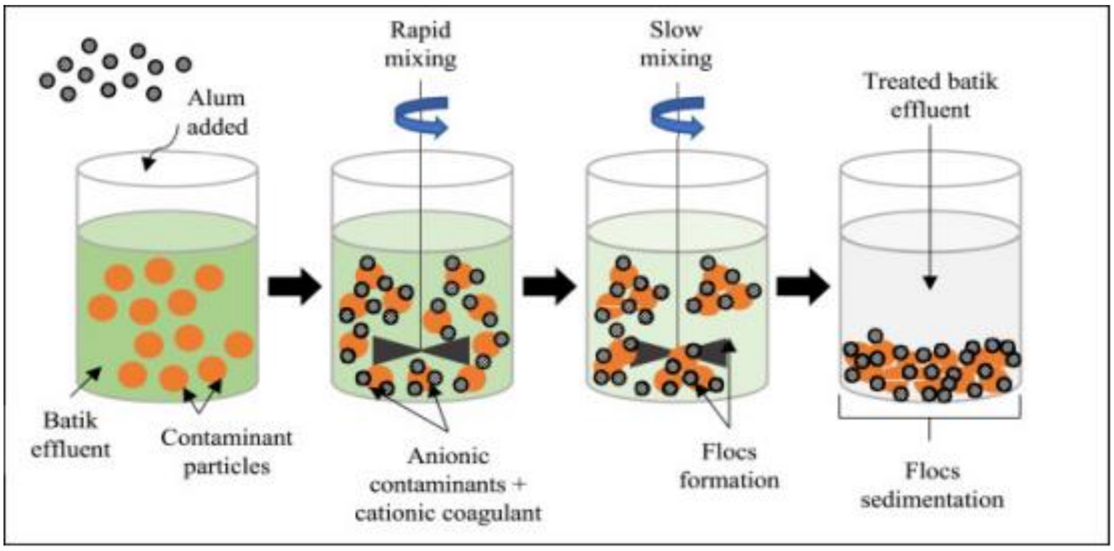


Figure 3.3 Coagulation and Flocculation Stages

3.1.3 Sedimentation Stage

The sedimentation process follows flocculation, where larger flocs settle at the bottom of a tank due to gravity, separating solids from water. By controlling Delta PLC, this stage ensures efficient settling by regulating flow rates and monitoring water levels. The clarified water is then moved to the filtration stage, while the settled sludge is periodically removed. Regular maintenance of the sedimentation tank is essential to prevent blockages and ensure optimal performance. Proper monitoring and control of the sedimentation process are critical to achieving consistent water quality. This step significantly reduces water turbidity, preparing for further purification. The settling time is adjusted based on the desired result.

3.1.4 Filtration Stage

In the filtration step, the water passes through a cotton filter followed by a cork filter to remove any remaining fine particles and impurities. This dual filtration system enhances the water's clarity and quality by trapping smaller contaminants that were not removed during sedimentation.

3.1.5 Distribution Stage

In the distribution step, the purified water is pumped from storage tanks to households for use. The Delta PLCs monitor the amount of purified water in the distribution tank, ensuring consistent water supply across the network. This step is essential for delivering safe, clean water directly to consumers.

3.2 Ladder Diagram Programming of WPL Soft

Ladder Diagram programming in WPL Soft software is a key component in designing and implementing control logic for the Delta PLC used in this system. WPL Soft provides an intuitive interface for creating Ladder Diagrams, which represent the electrical control circuits with a series of rungs and logic operations. Each rung in the diagram corresponds to a specific operation, such as starting a pump or opening a valve, allowing for precise control over the water treatment process. The software also supports simulation and debugging tools, enabling the user to test and refine the program before deploying it to the PLCs. This ensures that the system operates efficiently and reliability, with all processes synchronized and automated according to the desired parameters.

In WPL Soft, control logic for processes can be programmed using both Ladder Diagrams and Instruction Lists. Ladder Diagrams provide a visual representation of control logic, making it straightforward to design and understand process flows, such as controlling valves and sensors. Instruction Lists, on the other hand, offer a text-based approach, allowing for detailed and flexible control with commands like MOV, and CMP. This dual support enables the selection of the programming method that best suits specific project requirements and preferences.

3.3 User Interface of WPL Soft Software

The user interface uses standard windows interface functionality along with a few additional features to make development environment easy to use**.** The user interface window also includes a simulation mode, allowing users to test and validate their control logic before deploying it to the PLC. Real-time feedback and error reporting features help in identifying and resolving issues quickly. Comprehensive help documentation and tutorials are also available within the interface, supporting users in mastering both programming methods and optimizing their projects. User interface of WPL Soft software is shown in Figure 3.4.

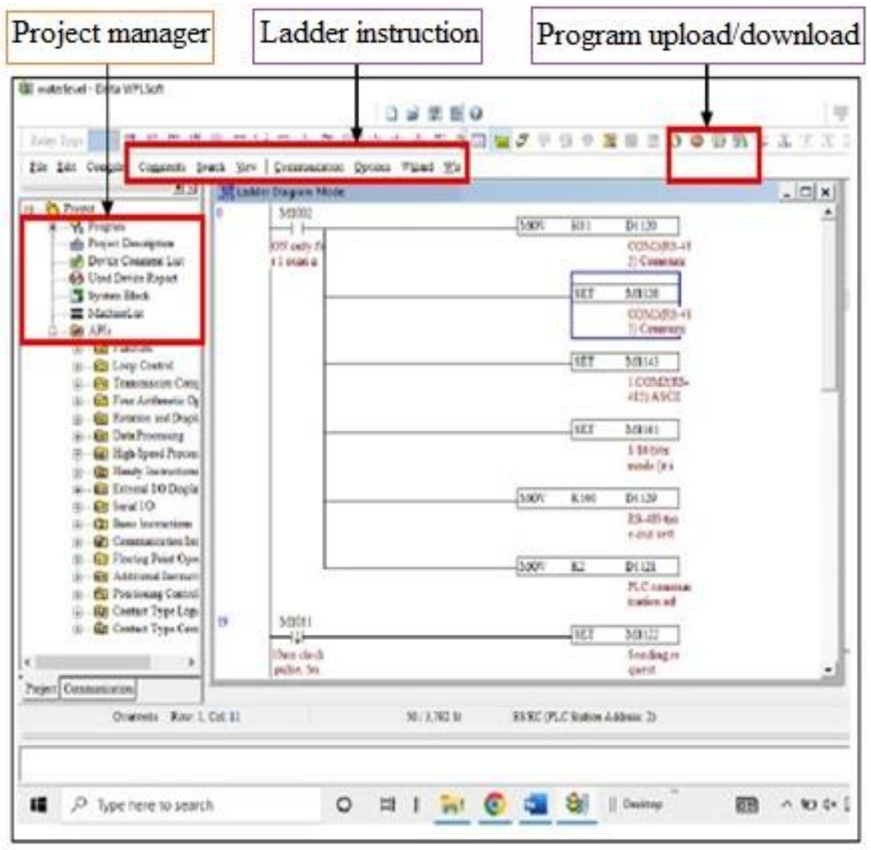


Figure 3.4 User Interface of Software

3.4 Function Block Instance

In WPL Soft, a Function Block (FB) instance must be declared before it is used in a program. The software automatically handles the declaration of FB instances in the Global Variable Table. The memory space allocated to each FB instance is determined by the hardware resources of the specific Delta PLC being used. As a result, different Delta PLC models may allocate varying memory ranges for FB instances, depending on the available resources and capabilities. The function blocks are used to go to the desired state. The function blocks used in this process include modbus read and compare functions. Function blocks make programming PLCs in **WPL Soft** more efficient. The detailed descriptions are given in the following table. The counter, timer and high-speed counter instructions are listed in the following Table 3.1. Modbus-related instructions are shown in Table 3.2.

Table 3.1 Counter, Timer and High-Speed Counter Instruction

Instruction	Description	Data Type	Example
Counter (C0_C199)			
CNT	Set the counter preset value (K) when counter is used.	16 bits integer	CNT C0 K10
Timer (T0-T199)			
TMR	Set the timer's preset value (K) based on its time base.	16-bit integer	TMR T0 K100
High Speed Counter (C235-C255)			
HSC	Represents the specific high-speed counter used.	32-bit integer	HSC C325
HSC.CV	Displays the current value of high-speed counter.	32-bit integer	HSC C325.CV
HSC.RST	Reset the high-speed counter to 0.	32-bit integer	HSC RST C235

HSC Mode	Defines the counting mode (up, down or quadrate).	configuration	HSC MODE

Table 3.2 Modbus-Related Instructions

Instruction	Description	Data Type	Example
MODRD	Reads data from modbus device into a specific register	16-bit integer	MODRD S1 S2
MOV	Move data from one register to another or to/from a variable	16-bit integer	MOV S D
CMP	Compares two values based on results	16-bit integer	CMP S1 S2 D

3.5 Introduction to the Program Components

When Delta PLC is used, about of instructions are needed to know. There are many instructions. The following instructions are used in the thesis.

- TMR (Timer)
- CMP (Compare to)
- MOV (Move to)
- MODRD (Modbus Read)

3.5.1 Ladder Diagram Using TMR (Timer) Function Block

When M0 is ON, it activates Timer T0, setting it for a 10-second delay (assuming a 1-second time base). After 10 seconds, T0's output will trigger. T0 is used as a function block for controlling M0 and the whole process is completed with END. Timer function block is shown in Figure 3.5.

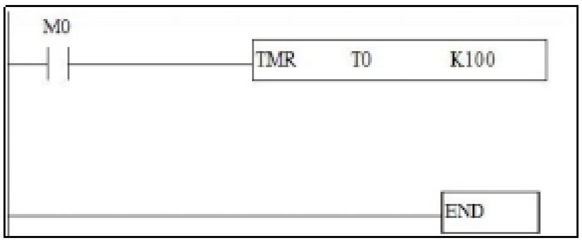


Figure 3.5 Instruction of TMR (Timer) Function Block

3.5.2 Ladder Diagram Using CMP (Compare to)

This program compares the contents of D100 and D200. The outcome of this comparison determines which internal bit M10 is set. The program logic tied to **M10** will then proceed based on the result of the comparison. CMP function block is used to control M0 and result is stored in M10 data register. CMP (compare to) function block is shown in Figure 3.6.

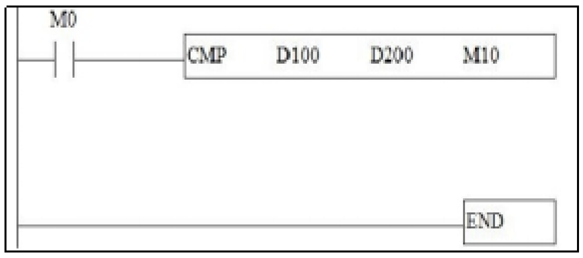


Figure 3.6 Instruction of CMP (compare to) Function Block

3.5.3 Ladder Diagram Using MOV (move to)

When the program begins execution when M0 is ON, the value in the **source of** register D100 is moved (copied) to the **destination** register D200. MOV (move to) function block is shown in Figure 3.7.

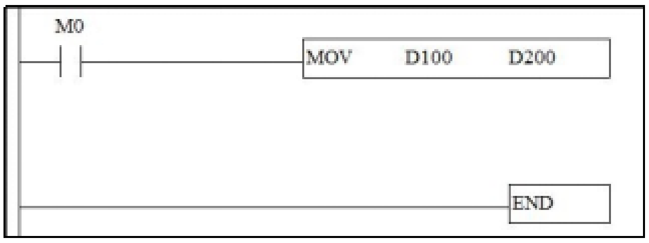


Figure 3.7 Instruction of MOV (move to) Function Block

3.6 Implementation of the System

To implement the water treatment system, Delta PLC is programmed using the WPL Soft. The system reads water levels as the analog values by linking the PLC to an Arduino Uno via RS485 Modbus communication. Ultrasonic sensors are employed to measure the water levels in four different tanks. The MODRD (Modbus read) function is utilized to retrieve sensor values through RS485. The MOV (Move to) function is then used to store these read values into data registers D0, D1, D2, and D3. Input sensors and the data register for each sensor of the system is shown in Table 3.3.

Table 3.3 Input Sensors and Data Register Assignment

Input sensors	Data Registers
Ultrasonic sensor_1	D1050
Ultrasonic sensor_2	D1051
Ultrasonic sensor_3	D1052
Ultrasonic sensor_4	D1053

The PLC reads the values from the sensors and uses the CMP (Compare) function to evaluate them. Before the desired level is reached, the PLC is controlled to repeat the process of turning on the operation according to the program instructions. When the read values are reached the desired level, the PLC automatically controls the operation of pumps, valves, and mixers, turning them on or off as needed. Input and output address assignments of the system is shown in Table 3.4.

Table 3.4 Input and Output Addresses Assignment

Devices	Comment	Address
Input Devices	Start Button	%M0.0
	Emergency Stop Button	%M0.1
Output Devices	Mixing Motor 1	%Y2
	Mixing Motor 2	%Y4

	Solenoid Valve 1	%Y2
	Solenoid Valve 2	%Y4
	Alarm Buzzer	%Y7
	Run Light	%Y0
	Pump 1	%Y1
	Pump 2	%Y3
	Pump 3	%Y5
	Pump 4	%Y6

CHAPTER 4

TEST AND RESULTS

This chapter includes test and results of the water treatment process with five stages including coagulation, flocculation, sedimentation, filtration and distribution stages.

4.1 Modbus Parameter Configuration

The PLC program starts by loading the status of memory bit M1002 into the accumulator. If M1002 is true, it moves the hexadecimal value H81 into a designated register. Then, it sets three memory bits M1120, M1143, and M1161, which could be used as flags or indicators for further processing. Next, it moves the constant value K100 into a specified register, likely for use in subsequent operations. The program then checks the status of memory bit M1011. If M1011 is false, it sets the memory bit M1122. This sequence is typically part of a control logic where specific conditions trigger actions, and the use of memory bits helps in managing state and flow within the PLC program. Modbus communication parameter setting is tested by the following Figure 4.1.

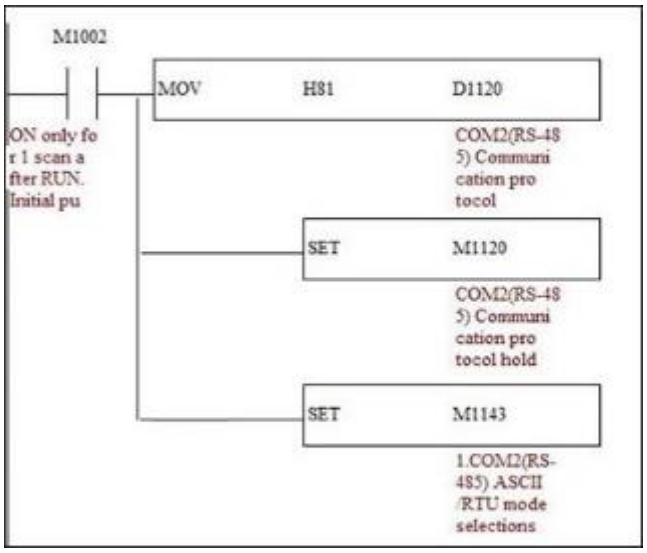

M1002
MOV
H81
D1120
ON only fo r 1 scan a fter RUN. Initial pu
COM2(RS-485) Communi cation pro tocol
SET
M1120
COM2(RS-485) Communi cation pro tocol hold
SET
M1143
1.COM2(RS-485) ASCII /RTU mode selections

Figure 4.1 Modbus Communication Parameter Setting

4.2 First State of the Program

The program is designed to initiate the process when the push button is pressed. At this point, raw water is filled into the coagulation tank by pump 1 (Y0). The water level being detected by an ultrasonic sensor. The data from the sensor is read from the PLC using modbus read instruction (MODRD). The result of program is shown in Figure 4.2.

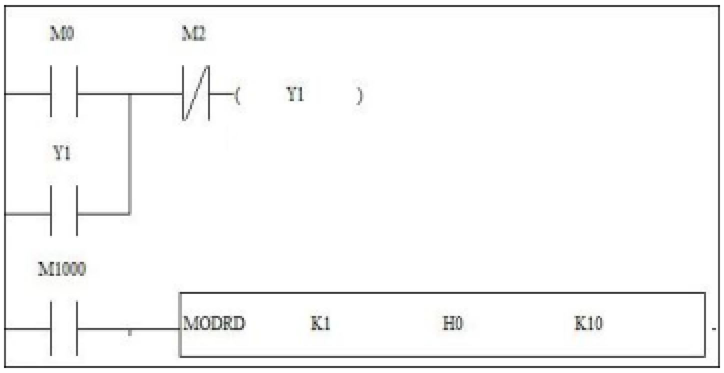


Figure 4.2 First State of the Program Run

4.2.1 Water Level Control of Coagulation Tank

The level of water in coagulation tank is read by “modrd” modbus read instruction and compare it with the preset value. Modbus read instruction and compare instruction is shown in the following Figure 4.3.

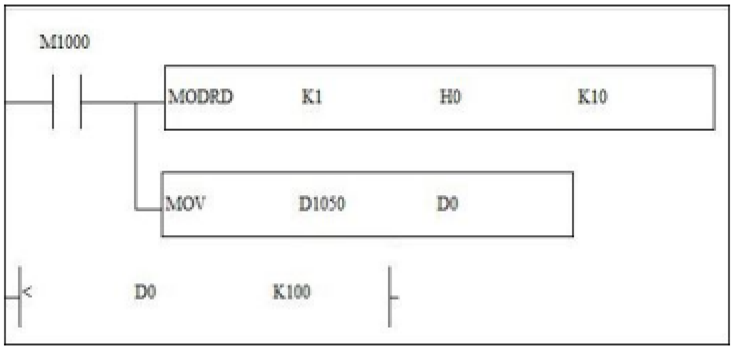


Figure 4.3 Test Result of Ultrasonic Sensor 1 Reading

4.3 Test and Result of Flocculation Stage

The water is transferred from the coagulation tank to the flocculation tank by Pump 2 (Y3). The water level in the flocculation tank is read by a second ultrasonic sensor. When the level reaches the preset value, Pump 2 is deactivated. At this stage, chemical dosing and mixing motor (Y2) begin and are regulated by a timer. Upon the timer expired, chemical dosing and mixing motor are stopped. The level of water in flocculation tank is shown in Figure 4.4 and Figure 4.5.

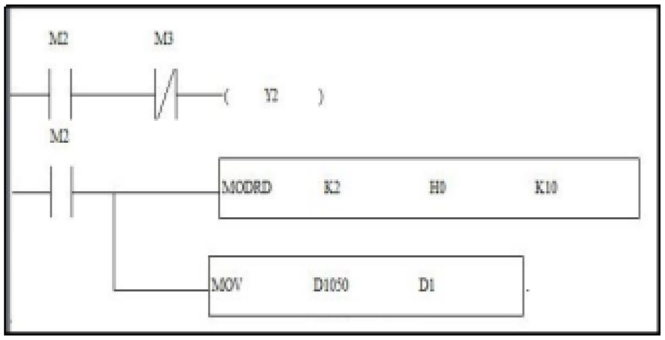


Figure 4.4 Chemical Dosing and Mixing Stage of Flocculation Tank

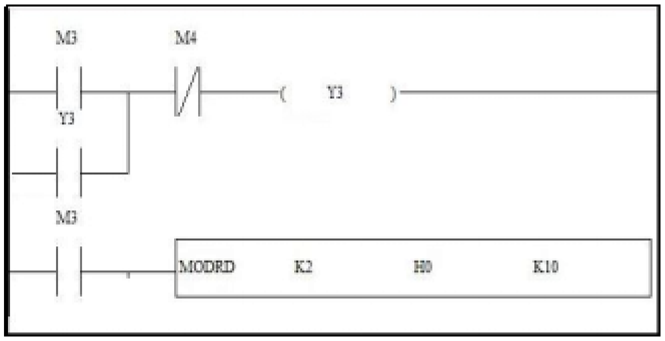


Figure 4.5 Filling Condition of Flocculation Tank

4.4 Test and Result of Sedimentation Stage

The water from the flocculation tank is pumped into the sedimentation tank using Pump 3 (Y5). The level in the sedimentation tank is monitored by third ultrasonic sensor. Internal relay M5 is used to trip in sedimentation tank. The level sensor value is stored in MOV function of D1052. Once the preset water level is achieved, Pump 3 ceases operation. Figure 4.6 shows program simulation result of sedimentation stage.

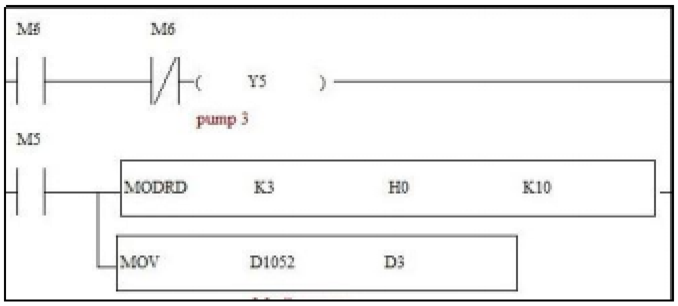


Figure 4.6 Test Result of Sedimentation Stage

4.5 Test and Result of Filtration and Distribution Stage

In this stage, the settled water is filtered through cotton and cork filter tubes. After filtration, the filtered water is transferred to the distribution tank using Pump 4 (Y6). The water level in the distribution tank is monitored by a fourth ultrasonic sensor. Once the preset level is detected, Pump 4 is stopped, signalling the completion of one full process cycle with the indicator light to notify that the cycle has been successfully completed. Testing result is shown is Figure 4.7.

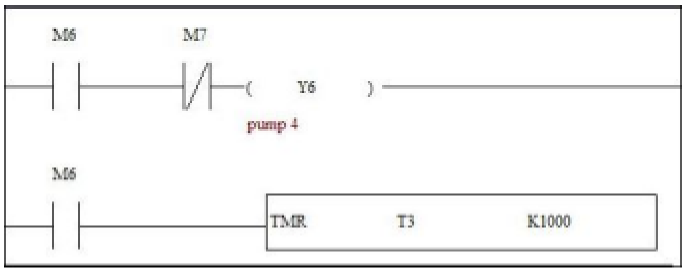


Figure 4.7 Water Filling Condition to Distribution Tank Through Filter

CHAPTER 5

DISCUSSION, CONCLUSION AND FURTHER EXTENSION

This chapter includes discussion and conclusion of water treatment system. Finally, further extension of the thesis is described.

5.1 Discussion

The water treatment process encompasses several stages, including coagulation, flocculation, sedimentation, filtration, and distribution. The integration of multiple microcontrollers (PLCs and Arduino) ensures automated monitoring and control, optimizing each stage. Coagulation initiates the process by binding suspended particles, while flocculation strengthens those bonds. Sedimentation separates solids, leaving clarified water, which then passes through filtration systems. The challenge of maintaining real-time data flow between controllers, particularly during filtration and distribution, required thorough calibration of communication protocols such as Modbus RTU. The inclusion of ultrasonic sensors to regulate tank levels effectively reduced human intervention, creating a more efficient system. However, slight delays in sensor readings during high-demand periods revealed the need for optimizing sensor data synchronization. The role of HMI in overseeing the process also proved instrumental, allowing for intuitive control and feedback. Overall, the system displayed strong performance, though minor adjustments are needed to enhance real-time response accuracy.

5.2 Conclusion

The Water Treatment Process automation was successful in integrating various hardware and software components to manage a multi-stage purification system. By combining the coagulation, flocculation, sedimentation, filtration, and distribution processes into an automated workflow, the system was able to efficiently treat water while reducing manual intervention. The Delta PLCs and Arduino performed well in managing their respective roles, and the integration of HMI provided user-friendly control over the process.

The turbidity value before the water is treated is 178.35 showing that some particles are floating and the raw water is a little bit cloudy. The pH level is 6.08 showing that the raw water contains 0.1 mg/L (milligrams per liter) of iron. The turbidity value after the water is treated is 160.15 showing that water clarity is significantly improved. The pH level is 7.05 showing that the iron content is 0.03 mg/L (milligrams per liter).

5.3 Further Extension

Future improvements in the Water Treatment Process system could focus on several areas. First, enhancing the communication protocol between the PLCs and sensors to minimize latency in real-time readings would significantly improve system responsiveness. Expanding the system to include solar-powered components for energy efficiency, especially in remote or off-grid areas, could also enhance sustainability. Furthermore, scaling the project to handle larger volumes of water or different water sources (e.g., seawater desalination) could open new avenues for research and application.