

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 requirement for the degree of

Master

In Electrical and electronic engineering

Option: Control engineering

Title:

**STEAM BOILER CONTROLLER DESIGN
AND SUPERVISION using SIEMENS S7-300**

Presented by:

-ZEMOULI Abd Raouf

-Ahmed SERIER Ishak

Supervisor:

Dr. OUADIA

Registration Number: /2021

Abstract

Today in the field of electronic the automation is very important for dehumanizing the human effort from the field operation. To perform the operation of the devices which are supposed to do at their own with a minimum help from the human. In Our project entitled by project steam Boiler Automation we used the concept of automation to automate our system to do secure operations, maintain the safety of peripherals, and run under a proper operation of heating. We propose the parameters like the steam temperature, water level and the pressure inside the vessel. The pressure of the steam should be measured and observed for safe reliable operation of the system. Same for the water that has to be maintained in a certain level and the steam temperature should not exceed certain value. This kind of operation can be done more efficiently and implemented by employing Programmable Logic Controller (PLC). Our design monitors boiler temperature (steam and water), pressure and volume through the variety of sensors which provide input to PLC. The PLC controls the different parameters and provides the user required volume of steams. All parameter variables are controlled and shown on the display panel which is the HMI in our case. For safety purpose mechanical safety valves are added to the boiler to release extra pressure.

ACKNOWLEDGEMENT

We would like to express our sincerest gratitude and warm appreciation to our project guide, Dr OUADIA, for believing in our ability to work and his contribution on helping us to shape this piece of work and enriching us with knowledge throughout our project that crowned our efforts with success, and the invaluable guidance and support that he has offered.

Our appreciations extend also the principle engineer of SOPI Company MR MOUNGLA Mohamed who helped and encouraged us during the internship; it was a great pleasure to work and study under his guidance. We are extremely grateful for what he has offered us. We would also want to thank him for his friendship, empathy and great sense of humor.

Another special appreciation goes to all the teachers and workers of INELEC and everyone who contributes in our good formation at this institute.

Finely we are extremely grateful for our parents for their love, caring and sacrifices for educating and preparing us to the future

ZEMOULI Abd Raouf

Ahmed SERIER Ishak

Boumerdes University

Institute of Electrical and Electronics Engineering

Control Engineering

Dedications

To the most courageous, sensitive, generous, most beautiful woman in my eyes, to the one who knew how to give me love and joy of living, to the one who had always showed affection and understanding towards me, my mother that I love.

To the man of courage and strength, to the one who had always been present, who taught me the true values of life, to the one who supported me in all circumstances, my father that I love.

To those who gave me joy and happiness, my sister and my brothers from the youngest to the oldest and to all of my family members.

To everyone I love, to everyone who loves me, I dedicate this modest work to you.

ZEMOULI A. Raouf.

Dedications

I dedicate this project to my wonderful parents Benouda and Rashida; you have raised me to be the person who I am to today, Thank you for your unconditional love and support. You provided me with Thank you for everything. To the source of my success and achievements my brothers Zoubir Abdel Ghafour and Mohamed and without forgetting my beloved sisters Kawther and Sarah, you have stood by my side in each step of my life.

To my teacher whom I express my deep sincerity for their assistance and support through my studies, to the reason of my happiness my friends I dedicate this work to you and I wish all the happiness.

Finally, I dedicate this work to everyone who contributed from far or close to the realization of this project and all the good and beneficial things that I did in my life.

AHMED SERIER Ishak.

List of Figures

Figure	Title	N°
I.1	Examples basic elements of automation	3
I.2	Manual control	4
I.3	Automatic control	4
I.4	Basic schematic of a PLC	5
I.5	Siemens modular PLCs	7
I.6	Presentation of PLC S7-300	7
I.7	Power supply module	8
I.8	An interface module	9
I.9	Digital module	9
I.10	Analog module	10
I.11	FM35 counter module	10
I.12	CM35 counter module	11
I.13	SM374 simulator	11
I.14	DM 370 dummy module	12
I.15	Front connector method	12
I.16	Top connector method	13
II.1	Flow control valve	16
II.2	Pressure control valve	17
II.3	Gate flow	18
II.4	Butterfly valve	18
II.5	Pump	19
II.6	Motor	19
II.7	Circuit breakers and their symbols	20
II.8	Isolator switch and its symbol	20
II.9	Thermal relay and its power and control symbol	20
II.10	Contactor and its power and control symbol	21
II.11	Signal lamp	21
II.12	Flow sensor	22
II.13	RTD PT100 temperature sensor	22

II.14	Types of pressure transmitters	23
II.15	Shell type boiler	25
II.16	A typical two-pass boiler configuration	25
II.17	Economic boiler (two-pass, dry back)	26
II.18	Economic boiler (three-pass, wet back)	27
II.19	Structure of the PID controller	32
II.20	Water level response.	35
II.21	Water level response of our process.	36
II.22	Time and water level entered in the workspace.	36
II.23	Importing input and output.	37
II.24	Transfer function identification.	37
II.25	Process model window.	38
II.26	Driving the transfer function $G(s)$ with a unit step.	39
II.27	The response of $G(s)$ to a unit step input.	39
II.28	Process response to the unit step with the tangential lines drawn on it	40
II.29	Closed loop response with PI controller of the tuned parameters.	41
III.1	Basic Tasks to perform a project	42
III.2	A project structure	43
III.3	Networks Configuration and communication connections	44
III.4	linear program processing	47
III.5	A scheme of a STEP7 program blocks	48
III.6	Creation of a new function	49
III.7	Controlling water level function program in FC20	50
III.8	Water level main program in OB1	51
III.9	Simulation results of the FC20	52
III.10	Pomp1 and pomp2 program in FC40 and FC50	53
III.11	Burner function program in FC30	54
III.12	Tank PID program in FC10	54
III.13	Controlling pumps in manual mode program in FC80	55
IV.1	Menu and toolbox of WinCC flexible	58
IV.2	Work area of WinCC flexible	58

IV.3	Work area of WinCC flexible	59
IV.4	Property view of WinCC flexible	59
IV.5	Library or database of WinCC flexible	60
IV.6	Simulation of the output view of WinCC flexible	60
IV.7	Object view of WinCC flexible	61
IV.8	Home screen of HMI	62
IV.9	Gas boiler view in the HMI	62
IV.10	Water tank view in the HMI	63
IV.11	PID boiler view in the HMI	63
IV.12	Discrete alarms	64
IV.13	Analog alarms	64
IV.14	Alarm view	65
IV.15	LM317 Pin out	67
IV.16	detailed circuit of IC LM317	67
IV.17	Final form of sub circuit 1	68
IV.18	connection of the ultrasonic sensor with Arduino	69
IV.19	sub-circuit 2 connection of L293D with the output of Arduino	69
IV.20	connection of the different input and output to the PLC	70
IV.21	PLC program	71

List of Tables

Table	Title	N°
II.1	Classification of boilers	24
II.2	Size range of two-pass, dry back economic boilers	27
II.3	Heat transfer details of a modern three pass, wet back, economic boiler	27
II.4	Measurement and detection components	28
II.5	Actuators	29
II.6	Tuned parameters based on Zeigler-Nichol method.	33
II.7	Water level versus time.	35
II.8	PID parameter after zeigler-nichol's tuning method.	40
III.1	Hardware configuration table	44

III.2	symbol table	45
III.3	types of variables used in STEP7	46

Nomenclature

CP	Communication process.
PLC	Programmable logic controller.
CPU	Central process unit.
C	Degree centigrade.
Kp	Proportional gain
E (t)	Error signal.
SP	Set point.
PV	Process variable.
Ti	Integral time.
Ki	Integral gain.
Td	Derivative time.
Kd	Derivative gain.
Gc (s)	The PID transfer function.
τ	Pure delay time.
K	The flying-up speed of the step response.
G(s)	Transfer function of filling the tank.
T(s)	The timing of the filling.
L(s)	The output level inside the water tank.
T1, T2	Time constant of the water level change(s).
u	Step input.
AI	Analog input.
DI	Digital input.
PS	Power supply.
IM	Interface module.

LAD	Ladder Logic.
STL	Statement List.
FBD	Function Block Diagram.
OB	Organization block.
FB	Function block.
DB	Data block.
F	Function.
HMI	Human Machine Interface.

Abstract.....	I
Acknowledgement.....	II
Dedications.....	III
List of Figures.....	V
List of Tables.....	VII
Nomenclature.....	VIII

Content

General introduction	1
----------------------------	---

Chapter I: Programmable Logic Controller

I.1. Introduction	3
I.2. Automation systems	3
I.3. Control system	3
I.3.1. Basic Elements	3
I.3.2. Manual control	4
I.3.3. Automatic control	4
I.3.4. Importance of process control / control systems.....	4
I.4. PLC (Programmable logic controller)	5
I.4.1. Basic PLC operation	5
I.4.2. Advantages of PLCs.....	6
I.4.3. Siemens modular PLCs	6
I.4.4. Presentation of PLC S7-300.....	7
I.4.4.1. Power supplies	8
I.4.4.2. Central processing units	8

I.4.4.3. Interface modules.....	9
I.4.4.4 Digital modules.....	9
I.4.4.5 Analog modules.....	9
I.4.4.6. Function modules.....	10
I.4.4.7. Special modules.....	11
I.4.4.8. Connection methods.....	12
I.5. STEP 7 program general overview.....	13
I.5.1. Objects.....	14
I.6. Presentation of Win CC Runtime Advanced.....	14
I.6.1. Introduction.....	14
I.6.2. Tasks of an HMI system.....	14
I.7. Conclusion.....	15

Chapter II Boiler description and PID control

II.1 Introduction.....	16
II.2. Elements of a Measurement System	16
II.3. Control valves	16
II.3.1. Pneumatic Valves	17
II.3.2. Flow control valve.....	17
II.4. Hydraulic valves	18
II.4.1. Gate Valves	18
II.4.2. Butterfly Valves	19
II.5. Pumps	19
II.6. Motors	21
II.7. Electrical circuit components	21
II.8. Sensors.....	22

II.8.1. Flow sensors	22
II.8.2. Temperature Sensors	23
II.8.3. Pressure transmitter	23
II.9. Boilers	24
II.9.1. Classification of boilers.....	25
II.9.1.1. Packaged Boilers	26
II.9.1.2. Shell Type	27
II.9.1.2.1. Economic boiler (two-pass, dry back)....	28
II.9.1.2.2. Economic boiler (three-pass, wet back).....	28
II.10. Burner.....	29
II.11. System Description.....	29
II.11.1. Boiler composition.....	30
II.11.2. Measurement and detection components.....	30
II.11.3. Actuators.....	30
II.11.4. Operation conditions.....	30
II.12. Tuning of PID parameters	31
II.12.1PID.....	31
II.12.1.1. Introduction.....	31
II.12.1.2. Proportional control.....	31.
II.12.1.3. Integral Control.....	33
II.12.1.4. Derivative Control.....	33
II.12.2. Ziegler and Nichols tuning method.....	34
II.12.2.1. Steps to determine PID controller parameters.....	34
II.12.3. Case Study.....	35

II.12.4. System identification using ‘ident’ tool.....	36
II.13. Conclusion.....	41

Chapter III SIMATIC MANAGER software description and programming

III.1 Introduction	43
III.2. Overview of STEP 7.....	43
III.2.1. Basic Tasks	43
III.2.1.1 Alternative Procedures	44
III.2.1.2. Brief Description of the Individual Steps	44
III.2.2. Types of variables used in step7.....	47
III.3. Programming Languages	47
III.4. Program Processing	48
III.4.1. Linear program processing	48
III.4.2. Structured programming.....	49
III.5. Types of a step7 program blocks.....	49
III.5.1. Organization block (OB).....	49
III.5.2. Function block (FB).....	50
III.5.3. Function (FC).....	50
III.5.4. Data block (DB).....	50
III.6. Blocks programming.....	50
II.7. Conclusion.....	56

Chapter IV Simulation and supervision using WINCC IV.

IV.1. Introduction	57
IV.2. Elements of WINCC flexible	57
IV.3. Case study.....	61
IV.4. Implementation.....	65

IV.4.1. Introduction.....	65
IV.4.2. List of equipment required.....	65
IV.4.3. Required software.....	66
IV.4.4. Building the circuit.....	66
IV.4.4.1. The 1st sub-circuit.....	67
IV.4.4.2. The 2 nd sub-circuit.....	69
IV.4.4.3. The 3rd sub circuit.....	70
IV.4.5. Programming the controllers	
IV.4.5.1. Programming the PLC s7-1200 1212c AC/DC/RLY.....	70
IV.5. Conclusion	73
General Conclusion	74
Appendix	
Bibliography	

GENERAL INTRODUCTION

A control system is a dynamic system that contains a controller as a main part. The purpose of this controller is to generate control signals which drives the process to be controlled in the desired manner. Sensors and actuators are necessary to measure output signals and perform control actions. In conventional control engineering, a control problem is solved by wiring up contactors and relays individually to perform the required task. Contactor and relay controllers and electronic controllers assembled from separate components are referred to as hard-wired programmed controllers. The program is in the wiring where programmable logic controllers, on the other hand, are made up of standard components and the desired control function is implemented by a user program in the CPU.

The steam boiler of SOPI Company has various automation systems based on wired logic technology, the electrical control and monitoring of the steam generator are controlled by interconnected inflexible integrated circuits. Currently, through time there are disadvantages such as the difficulty of detecting repetitive breakdowns and maintenance depending on the complexity of the existing installation. Faced with the current conditions, the efficiency which is minimum, responsible have taken the decision to renew the classic control with a more recent modern control and more reliable based on the industrial programmable logic controller in order to maintain the user required volume of steam.

Our objective in this project is to study the operating system of the steam generator of the company, and to replace the existing control system based on hard-wired logic with a programmed solution based on an industrial programmable logic controller SIEMENS S7-300. This solution will ensure the proper functioning of the steam generator in an expandable, interconnected to field networks and capable of communicating with industrial and supervisory PCs.

Our work is divided into these following chapters:

- In the first chapter we will introduce the basic information and a description of the programmable logic controllers and the programming tool used in our project.
- The second chapter is devoted to the various components of the steam generator and its function, also we have identified the transfer function of the regulating water pump and we designed our PID controller and tuned its parameters.
- The third chapter we have shown the features of the SIMATIC MANAGER software and how to create a new projects and program using ladder language.

- In the last chapter, we introduced another software which is the SIMATIC WINCC Flexible that is used to create Human Machine Interfaces or supervision screens in order to test our program functionality and provide the user a sight to the process variations in the real time and implementing water level control using PLC s7-1200.

Chapter I

Programmable Logic Controller

I.1. Introduction:

This chapter is meant to give some information on automation and the basic elements used on it. We will deal with functions and configurations of PLCs, with emphasis on the S7-300 PLC family and state its advantages, after that we will states programming and interfacing software used in this project which are SIMATIC STEP 7 and WinCC flexible.

I.2. Automation systems:

Automation is a process to accomplish a specific task by a machine in the particular time period without any human intervention. It is a self-governing execution of a function by a device or the system which was previously carried out by a human. The Automation (automatic control) is useful in numerous process control systems for operating equipment such as a machine, a variable frequency drive (VFD), AC, DC and the servo motors in the manufacturing plants, to control the heater, boiler, compressor, and actuators in chemical process plant, switching in electrical transmission lines, telephone networks, regulation and stabilization of ships, flying machine and radar other applications with minimal human intervention. Automation involves the following [17]:

- Capture the measurement and control command for particular operation.
- Transmute specific command into controlling action for particular operation.

I.3. Control system:

A Control System is a device or a collection of devices that manage the behavior of other devices. And it is the interconnection of components connected or related in such a manner as to command, direct or regulate itself or another system [18].

I.3.1. Basic Elements:

- Sensor - Final Element - Transmitter
- Controller (Distributed Control System / DCS).



Fig I.1: examples basic elements of automation [9].

I.3.2. Manual control:

Operator is required to run the plant and has to calculate the manipulated variable manually.

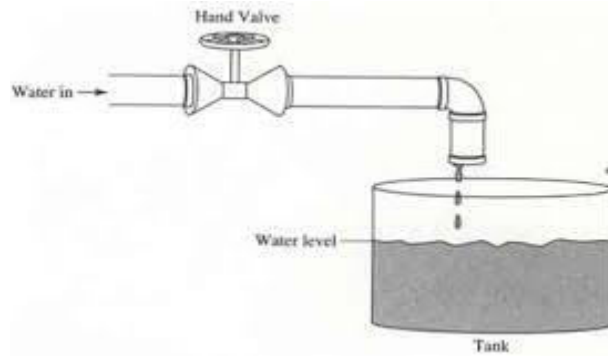


Fig I.2: Manual control.

I.3.3. Automatic control:

In an automatic control loop, a controller compares a measured value of a process with a desired set value, and processes the resulting error signal to change some input to the process, in such a way that the process stays at its set point despite disturbances.

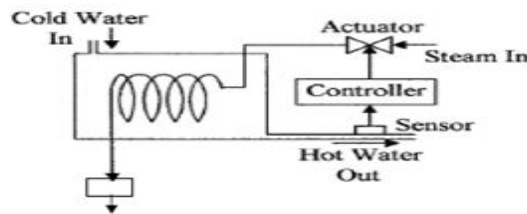


Fig I.3: Automatic control.

I.3.4. Importance of process control / control systems:

Safety: The primary purpose of a Process Control system is safety such as: personnel safety, environmental safety and equipment safety. The safety of plant personnel and the community is the highest priority in any operation.

Quality In addition to safety, process control systems are central to maintaining product quality. In blending and batching operations control systems maintain the proper ratio of ingredients to deliver a consistent product.

Profit: when safety and quality concerns are met, process control objectives can be focused on profit. The challenge in process control is to do so safely without compromising product quality, process optimization and improve system productivity and reliability.

I.4. PLC (Programmable logic controller):

Programmable logic controller (PLC) is the name given to a type of Computer commonly used in commercial and industrial control applications. PLCs differ from office computers in the types of tasks that they perform. While the specific applications vary widely, all PLCs monitor inputs and other variable values make decisions based on a stored program, and control outputs to automate a process or a machine [8].

I.4.1. Basic PLC operation:

The basic elements of a PLC include input modules or points, a central processing unit (CPU), output modules or points and a programming device. The type of input modules or points used by a PLC depends upon the types of input devices used. Some input modules or points respond to digital inputs also called discrete inputs, which are either on or off. Other modules or inputs respond to analog signals. These analog signals represent machine or process conditions as a range of voltage or current values, the primary function of a PLC's input circuitry is to convert the signals provided by these various switches and sensors into logic signals that can be used by the CPU.

CPU evaluates the status of inputs, outputs and other variables as it executes a stored program. The CPU then sends signals to update the status of outputs.

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 PLC's program or to monitor or change stored values. Once entered, the program and associated variables are stored in the CPU. In addition to these basic elements, a PLC system may also incorporate an operator interface device to simplify monitoring of the machine or process [8].

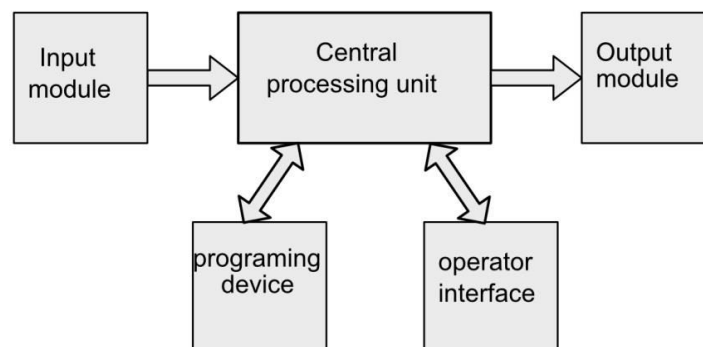


Fig I.4: basic schematic of a PLC.

I.4.2. Advantages of PLCs:

PLC not only are capable of performing the same tasks as hard-wired control, but are also capable of many more complex applications. In addition, the PLC program and electronic communication lines replace much of the interconnecting wires required by hard-wired control. Therefore, hard-wiring, though still required to connect field devices, which are less intensive. This also makes correcting errors and modifying the application easier. Some of the additional advantages of PLCs are as follows:

- Smaller physical size than hard-wire solutions.
- Easier and faster to make changes.
- PLCs have integrated diagnostics and override functions.
- Diagnostics are centrally available.
- Applications can be immediately documented.
- Applications can be duplicated faster and less expensively [8].

I.4.3. Siemens modular PLCs:

Siemens SIMATIC PLCs are the foundation upon which our Totally Integrated Automation (TIA) concept is based. Because the needs of end users and machine builders vary widely, SIMATIC PLCs are available as conventional modular controllers, embedded automation products, or as PC-based controllers.

Modular SIMATIC controllers are optimized for control tasks that can be adapted to meet application requirements using plug-in modules for input/output (I/O), special functions, and communications. Examples of products in this category include: LOGO, S7-200, S7-1200 micro automation products, S7-300 and S7-400 modular system PLCs, C7 combination controller and panel and ET 200 distributed I/O system with local intelligence [8].

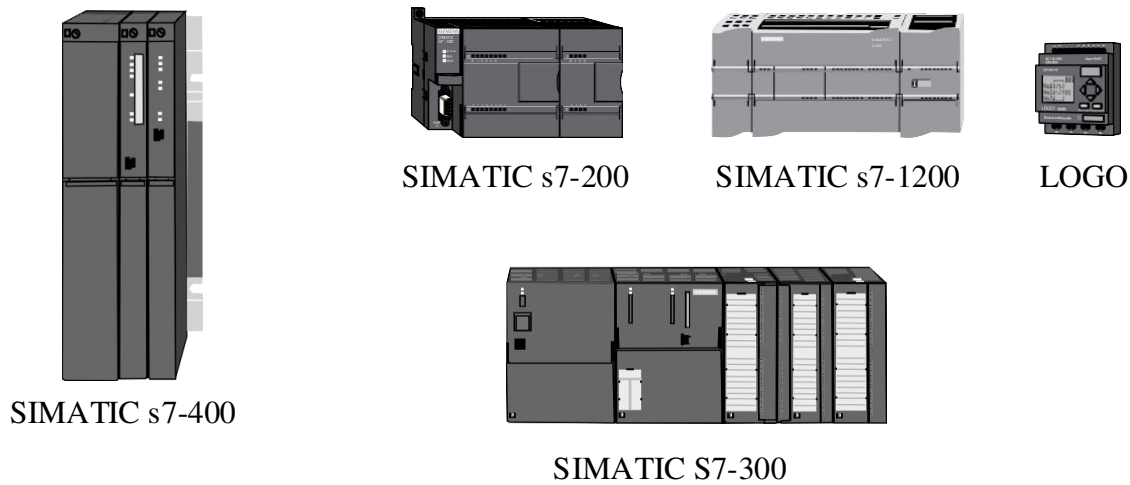


Fig I.5: Siemens modular PLCs.

I.4.4. Presentation of PLC S7-300:

The S7-300 controller is a modular mini controller for entry-level and mid-range applications, manufactured by the firm SIMENS. We can compose it according to our needs from a wide range of modules. The range of modules includes:

- CPUs of different performance levels.
- Signal modules for digital and analog inputs / outputs.
- Function modules for various technological functions.
- Communication processes (CP) for communication tasks.
- Power supply modules for S7-300.

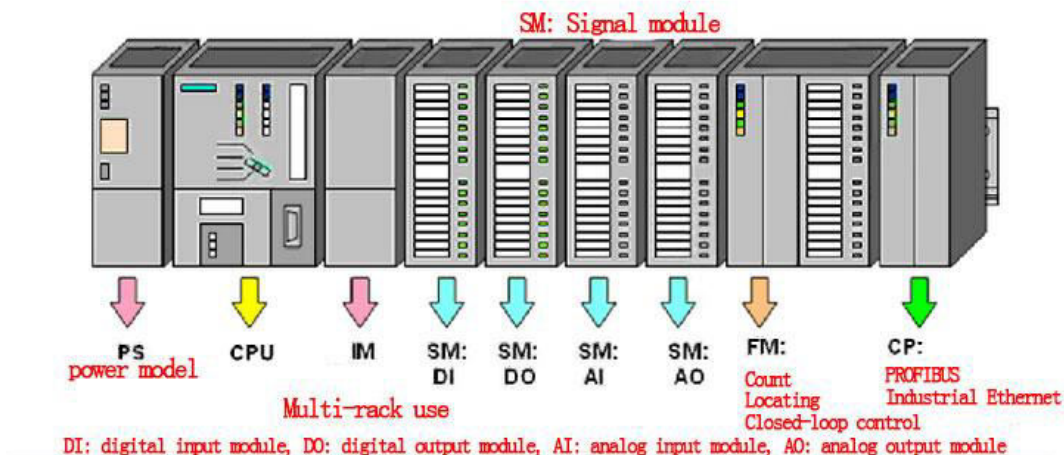


Fig I.6: presentation of PLC S7-300.

- Rack 1: power model 24V/10A.
- Rack 2: CPU (Central Processing Unit) 314C-2 DP.
- Rack 3: interface module.
- Rack 4: digital input module.
- Rack 5: digital output module.
- Rack 6: analog input module.
- Rack 7: analog output module.
- Rack 8: function module.
- Rack 9: communication process module [7].

I.4.4.1. Power supplies:

PS 307 power supply modules:

- Load power supplies for S7-300/ET 200M.
- For conversion of the line voltage to the required operating voltage of 24 V DC.
- Output current of 2 A, 5 A or 10 A.



Fig I.7: power supply module.

I.4.4.2. Central processing units:

20 different CPUs:

- 6 compact CPUs (with integrated technology functions and I/O).
- 3 redesigned standard CPUs (CPU 312, CPU 314 and CPU 315-2 DP).
- 5 standard CPUs (CPU 313, CPU 314C-2DP, CPU 315, CPU 315-2 DP, CPU 316-2 DP); superseded in the medium-term by redesigned standard CPUs.
- CPU 315F-2 DP.
- 4 SIMATIC S7-300 Outdoor CPUs (CPU 312 IFM, CPU 314 IFM, CPU 315-2 DP).
- CPU 318-2 DP [6].

I.4.4.3. Interface modules:

IM 360/-361/-365 interface modules:

- For connecting the racks in multi-tier configurations of the SIMATIC S7-300.
- IM 365:** For configuring a central controller and no more than one expansion rack.
- IM 360/IM 361:** For configuring a central controller and up to four expansion racks.



Fig I.8: an interface modules.

I.4.4.4 Digital modules:

- Digital I/O for the SIMATIC S7-300
- For connecting switches and 2-wire proximity switches [6].



Fig I.9: a digital module.

I.4.4.5 Analog modules:

SM 331 analog input modules

- Analog inputs for the SIMATIC S7-300
- For the connection of voltage and current sensors, thermocouples, resistors and resistance thermometers [6].



Fig I.10: Analog module.

I.4.4.6. Function modules:

- **FM 350-1 counter module:**

- One-channel intelligent counter module for simple counting tasks
- For direct connection of incremental encoders
- Comparison function with 2specifiable comparison values
- Integrated digital outputs to output the response upon reaching the comparison value.
- Operating modes:
 - Continuous counting.
 - One-shot counting.
 - Periodic counting.
- Special functions:
 - Set counter.
 - Latch counter.
- Start/stop counter with gate function [6].



Fig I.11: a FM35 counter module.

- **CM 35 counter module:**

- 8-channel intelligent counter module for universal metering and measurement tasks, as well as for simple positioning tasks (max. 4 axes).
- 8 counter inputs (optionally for 5 V or 24 V signal level).
- 8 integrated digital outputs for fast output of module reactions. Provided that the outputs are not occupied by the set operating mode, they can be used freely as a process I/O by the user program [6].



Fig I.12: CM35 counter module.

I.4.4.7. Special modules:

- **SM 374 simulator:**

- Simulator module for testing programs during startup and operation.
- For simulation of sensor signals using switches.
- For indicating signal statuses at the outputs using LEDs [6].



Fig I.13: SM374 simulator.

- **DM 370 dummy module:**

- Dummy module for reserving slots for non-parameterized signal modules.
- Structure and address allocation is retained when replaced with a signal module [6].



Fig I.14: DM 370 dummy module.

I.4.4.8. Connection methods:

- **Front connector:**

- For simple and user-friendly connection of sensors and actuators.
- For retaining the wiring when replacing modules.
- With coding to avoid mistakes when replacing modules [6].



Fig I.15: Front connector method.

- **SIMATIC TOP connect; fully modular connection:**

- The standard connection for SIMATIC S7-300.
- For fast and error-free connection of sensors and actuators for distances of up to 3 m.
- For clear and understandable wiring in the switching cabinet.
- Comprising front connector module, connecting cable and terminal block.
- All components are easy to plug in and can be replaced individual [6].



Fig I.16: Top connector method.

- **SIMATIC TOP connect; flexible connection:**

- For fast, direct connections to individual elements in the control cabinet.
- Comprises front connector with individual cores attached.
- Core type H05V-K or UL/CSA.
- 0.5 mm² wire cross section also allows higher currents [6].

I.5. STEP 7 program general overview:

Following the replacement of SIMATIC S5 by SIMATIC S7 in the autumn of 1995, a new programming language (STEP 7) was developed based on the IEC 1131 standard. This programming language provides the entire functionality for parameterizing, configuring and programming the S7-300 automation unit. STEP 7 runs under Windows XP or higher. The symbols for the objects are mapped on the graphical user interface. STEP 7 has an integrated online help system that provides valuable hints and tips [2].

I.5.1. Objects:

On the graphical user interface, the system represents objects with symbols which are assigned to a specific object.

STEP 7 objects offer access to the following processing functions:

- Create and open objects.
- Edit and save objects.
- Rename objects.
- Delete objects.
- Copy and paste objects.

An object can for example contain as symbol, a project, a SIMATIC station, an S7 program and so on tips [2].

I.6. Presentation of Win CC Runtime Advanced:

I.6.1. Introduction:

When the complexity of the processes increases, machines and plants have to meet ever more stringent functional specifications, the operator needs maximum transparency. This transparency is obtained by means of the Human-Machine Interface (HMI). An HMI system constitutes the interface between man (operator) and process (machine / installation). The actual process control is ensured by the automation system.

Therefore there is an interface between the operator and WinCC Runtime Advanced (on the operator panel) and an interface between WinCC Runtime Advanced and the automation system [5].

I.6.2. Tasks of an HMI system:

- Process representation.
- Process control.
- Alarm view.
- Archiving of process values and messages.
- Management of process and machine parameters [5].

I.7. Conclusion:

In this chapter, we have presented the different components of the industrial programmable logic controller of the S7-300 range and their characteristics that allowed us to assess their importance and see their essential role in the management of industrial processes, which is the controller chosen to regulate the various tasks of controlling the boiler. As well as the role of the STEP7 software and the WinCC Runtime Advanced supervision software (which is presented in the 3rd and the 4th chapter consecutively).

Chapter II

Boiler description and PID control

II.1.Introduction:

Before we start automating the system we should know all components of our system and how they behave. This chapter is divided into two main parts which are the boiler and the PID control of the water level inside the tank. In the first part we will states different kind of boilers and define more specifically the one we used, and all the actuators and sensors that exist in the system and we will define the functionality of each one of it, as well as the electrical circuit components. In addition to that we will describe the operating conditions of the system and how it works with all its sensors and actuators without forgetting the security part of the whole system. In the second part we will talk specifically about PID and the method used in tuning its parameters in order to stabilize the system of filling up the tank. Finely we will discuss the method used in system identification for the regulating water pump.

II.2. Elements of a Measurement System:

Measurement system exists to provide information about the physical value of some variable being measured. In simple cases, the system can consist of only a single unit that gives an output reading or signal according to the magnitude of the unknown variable applied to it. However, in more complex measurement situations, a measuring system consists of several separate elements. These components might be contained within one or more boxes, and the boxes holding individual measurement elements might be either close together or physically separate. The term measuring instrument is used commonly to describe a measurement system, whether it contains only one or many elements, and this term is widely used throughout this text.

II.3. Control valves:

The term control valve refers more to the function than the style of the valve. It can refer to any valve that serves to regulate the steady-state flow or pressure in a system. This includes isolation, block, or sectionalizing valves that are used to prevent flow in certain sections of the pipe. Of more interest are control valves used to control the flow without creating undesirable transients, excess cavitation, or head loss and able to function under all expected flow conditions. There are numerous valves that can be classified as control valves. A brief description of the common types follows [11].

II.3.1. Pneumatic Valves:

The function of pneumatic valves is to control the pressure or flow rate of pressure media. Depending on design.

These can be divided into the following categories:

- Directional control valves.
 - Input / signaling elements.
 - Processing elements.
 - Control elements.
- Non-return valves.
- Flow control valves.
- Pressure control valves.
- Shut-off valves [10].

II.3.2. Flow control valve:

The flow control valve restricts or throttles the air in a particular direction to reduce the flow rate of the air and hence control the signal flow. Ideally it should be possible to infinitely vary the restrictor from fully open to completely closed. The flow control valve should be fitted as close to the working element as is possible and must be adjusted to match

The requirements of the application. If the flow control valve is fitted with a check valve then the function of flow-control is unidirectional with full free flow in one direction.

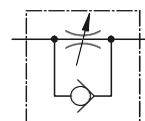


Fig II.1 Flow control valve.

Flow control valve, adjustable



One-way flow control valve



Pressure control valves are used in pneumatic systems. There are three main groups:

- Pressure limitant valves.
- Pressure regulating valves.
- Pressure sequence valves.

The pressure limiting valves are used on the up-stream side of the compressor to ensure the receiver pressure is limited, for safety, and that the supply pressure to the system is set to the correct pressure.

The pressure regulating valve keeps the pressure constant irrespective of any pressure fluctuations in the system. The valve regulates the pressure via a built-in diaphragm.

The pressure sequence valve is used if a pressure-dependent signal is required for the advancing of a control system [10].



Fig II.2 Pressure control valve.

II.4. Hydraulic valves:

II.4.1. Gate Valves:

This type of valve has a totally enclosed body with a circular or rectangular disk or gate that moves perpendicular to the flow direction. Some gate valves have a tapered circular disk and tapered guide slots. The matching taper of the valve seat causes a watertight metal to metal contact as the disk is wedged into the seating surface. The disk is normally raised by rotation of a hand wheel. The gate valve also comes in a double-disk gate design such that when the valve is closed both sides of the disk are wedged against the seats.

When a full-ported gate valve is wide open, the flow passage is only slightly less than the area of the pipe due to projecting seats and guides, so it has a large discharge capacity and small pressure drop. Some gate valves have a reduced port where the valve body is formed into a venturi shape, causing the fully open area to be reduced. Gate valves are normally operated only

wide open or closed and not for regulating flow [11].

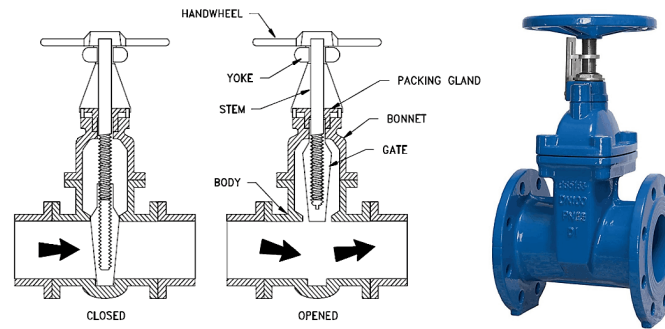


Fig II.3 Gate flow.

II.4.2. Butterfly Valves:

A typical butterfly valve consists basically of a disk which rotates 90° from fully open to fully close. There are numerous alternative disk designs. This includes symmetrical, unsymmetrical, eccentric and flow-through. The shape of the disk influences the capacity and flow torque.

The butterfly is a popular valve at the present time due to its light weight, compact size, satisfactory performance, and low cost. The valves are suitable for throttling as well as on-off service. With certain disk designs, the flow capacity of a butterfly valve can approach that of a gate valve in the full open position. Flow torque and cavitation can be controlled by alterations of the shape of the disk and the seat. A variety of materials can be used for the body, disk, and seat to make it suitable for use with almost any liquid [11].



Fig II.4 Butterfly valve.

II.5. Pumps:

A pump is a device that moves fluids (liquids or gases) by mechanical action, typically converted from electrical energy into hydraulic energy. Pumps can be classified into three major groups according to the method they use to move the fluid: direct lift, displacement and gravity

Chapter II: Boiler Description and PID Control

pumps. Pumps operate by some mechanism (typically reciprocating or rotary) and to perform energy mechanical work moving the fluid. Pumps operate via many energy sources, including manual operation, electricity, engines or wind power and come in many sizes, from microscopic for use in medical applications to large industrial pumps.



Fig II.5 Pump.

II.6.Motors:

An electric motor is an electrical machine that converts electrical power into mechanical energy.

Most electric motors operate through the interaction between the motor's magnetic field and electric current in a wire winding to generate force in the form of torque applied on the motor's shaft.



Fig II.6 Motor.

II.7. Electrical circuit components:

- **Circuit breaker**

It is a protective device that has two relays, a magnetic relay that protects against short circuits and a thermal relay that protects against overloads.

Chapter II: Boiler Description and PID Control



Fig II.7 circuit breakers and their symbols.

- **Isolator switch**

Isolator switch are devices or systems that isolate a particular circuit for maintenance and



prevent currents from passing through.

Fig II.8 Isolator switch and its symbol.

- **Thermal relay:**

The thermal relay switches a magnetic contactor to protect a motor from overload.

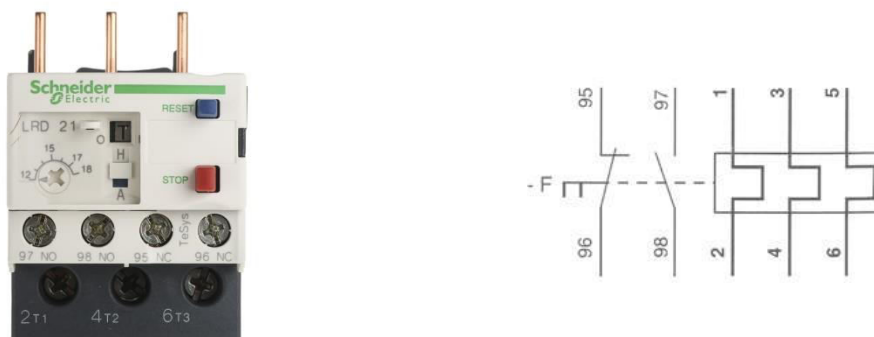


Fig II.9 Thermal relay and its power and control symbol.

- **Contactors**

The contactor is a control device capable of establishing or interrupting the passage of electrical energy. It performs the SWITCHING function.

Chapter II: Boiler Description and PID Control

In Automated Systems Technology this component is called pre-actuator since it is located before the actuator in the energy chain.

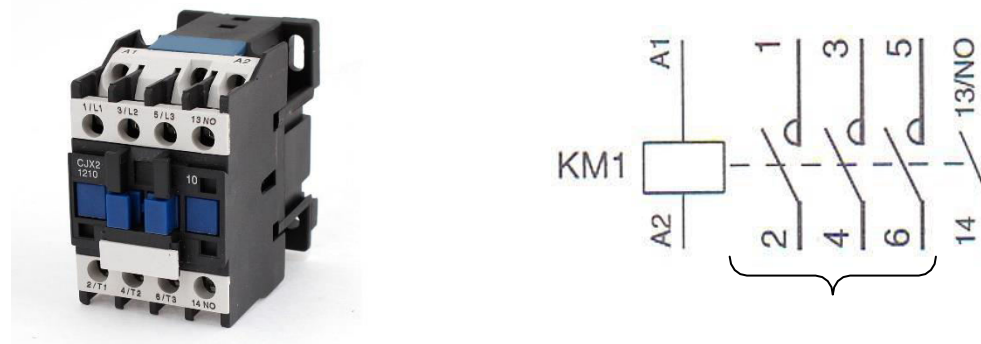


Fig II.10 contactor and its power and control symbol.

Signal lamps

Visual indication of normal system operation, or faults.



Fig II.11 Signal lamp.

II.8. Sensors:

II.8.1. Flow sensors:

In almost all fields of process and plant engineering liquids or gases are used for coolant and lubricant supply of machines and units, ventilation of installations and buildings and the processing of products. In case of no flow of these media considerable damage and downtime may result. Thus, it is very important to monitor these media. In modern installations electronic flow monitors are used for this purpose. They work without wear and tear and without mechanical components. This guarantees reliable monitoring even in case of difficult media over a long period.



Fig II.12 Flow sensor.

II.8.2. Temperature Sensors:

In most temperature measuring devices, the temperature is sensed through heat transfer from the source to the measuring device. The physical (or chemical) change in the device caused by this heat transfer is the transducer stage. Several temperature sensors are outlined below.

- Thermocouple.
- Resistance Temperature Detector (RTD).
- Thermistor.



Fig II.13 RTD PT100 temperature sensor.

II.8.3. Pressure transmitter:

Pressure transmitter is a mechanical device that measures the expansive force of a liquid or gaseous sample also known as a pressure transducer, this type of sensor is typically composed of a pressure sensitive surface area made of steel, silicon or other materials depending upon the analytic composition. Behind these surfaces are electronic components capable of converting the applied force of the sample upon the pressure sensor into an electrical signal.

Chapter II: Boiler Description and PID Control

Pressure is generally measured as a quantity of force per unit of surface area, and is expressed as the value required to stop a liquid, gas, or vapor from expanding. It can be quantified using various derived units including:

- As a proportion of / relation to a Pascal (Pa), or a single newton per square meter (1 N/m^2).
- A value of pounds per square inch (psi).



Fig II.14 types of pressure transmitters.

II.9. Boilers:

A boiler is a device for heating water and generating steam above atmospheric pressure. The boiler consists of a compartment where the gas is burned and a compartment where water can be evaporated into steam. The hot water or steam is used to transfer heat to a process.

II.9.1. Classification of boilers:

The boiler, the most commonly used piece of energy conversion equipment, varies widely in design depending on the firing method used, fuel fired, field of application, type of water circulation employed, and pressure of the steam. Table 1-2 gives a list of different types of boilers [12].

Chapter II: Boiler Description and PID Control

Firing method	Energy source	Use of steam	water circulation	steam pressure	construction
Stoker	coal	Utility	Natural	Atmosphere	Packaged (Shell)
Front firing	Liquid fuel	Industrial	Forced	Subcritical	
Tangential firing burner	Gas	Domestic	Once through	Supercritical	Packaged (water tube)
Opposed firing burner	Solid wastes	Marine	Combined	Sliding pressure	Field erected
Downjet firing burner	Biomass	Naval	/	/	/
Cyclone firing Bubbling	Recovery	Hot gas boiler	/	/	/
Fluidized circulating	waste heat	Cogeneration	/	/	/
Fluidized No firing method	Nuclear fuel	/	/	/	/

Table II.1 Classification of boilers [12].

II.9.1.1. Packaged Boilers:

Packaged boilers are usually smaller in capacity. They are preassembled units with all components mounted on a shippable mount. Thus, these boilers do not need assembly at site. Smaller units are shell type, larger ones are water-tube type [13].

II.9.1.2. Shell Type:

Shell boilers may be defined as those boilers in which the heat transfer surfaces are all contained within a steel shell. Shell boilers may also be referred to as ‘fire tube’ or ‘smoke tube’ boilers because the products of combustion pass through the boiler tubes, which in turn transfer heat to the surrounding boiler water.

Several different combinations of tube layout are used in shell boilers, involving the number of passes the heat from the boiler furnace will usefully make before being discharged [13].

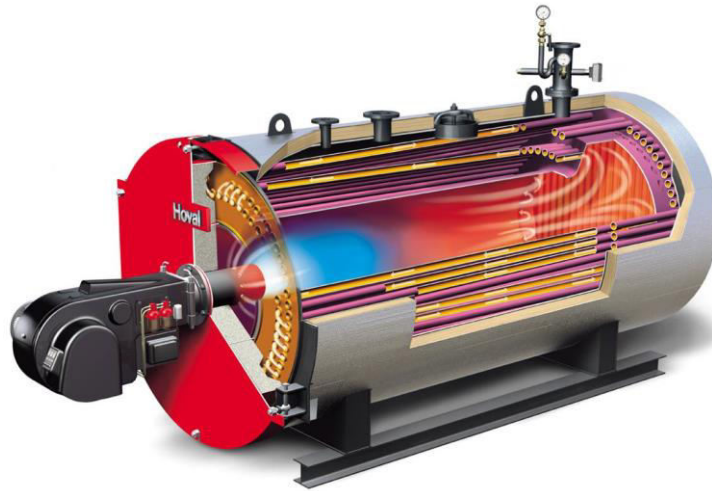


Fig II.15 Shell type boiler.

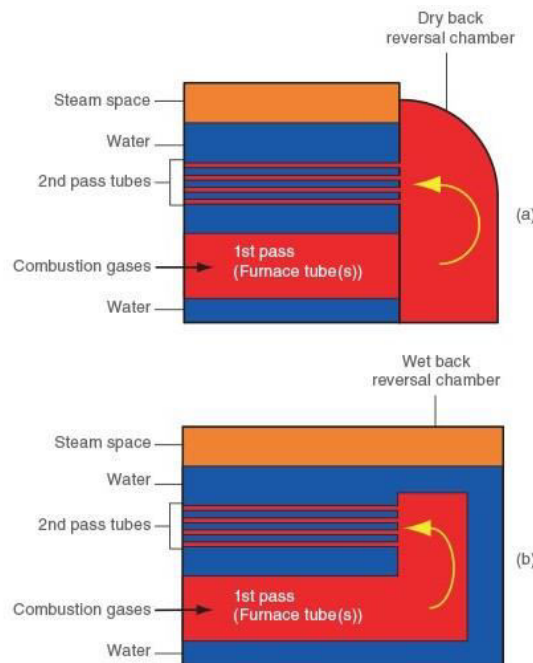


Fig II.16 a typical two-pass boiler configuration.

Figure II.16.a shows a dry back boiler where the hot gases are reversed by a refractory lined chamber on the outer plating of the boiler.

Figure II.16.b shows a more efficient method of reversing the hot gases through a wet back boiler configuration. The reversal chamber is contained entirely within the boiler. This allows for a greater heat transfer area, as well as allowing the boiler water to be heated at the point where the heat from the furnace will be greatest - on the end of the chamber wall.

It is important to note that the combustion gases should be cooled to at least 420°C for plain steel boilers and 470°C for alloy steel boilers before entering the reversal chamber. Temperatures in excess of this will cause overheating and cracking of the tube end plates. The boiler designer will have taken this into consideration, and it is an important point if different fuels are being considered.

Several different types of shell boilers have been developed, which will now be looked at in more detail [13].

II.9.1.2.1. Economic boiler (two-pass, dry back):

The two-pass economic boiler was only about half the size of an equivalent Lancashire boiler and it had a higher thermal efficiency. It had a cylindrical outer shell containing two large-bore corrugated furnace flues acting as the main combustion chambers. The hot flue gases passed out of the two furnace flues at the back of the boiler into a brickwork setting (dry back) and were deflected through a number of small-bore tubes arranged above the large-bore furnace flues. These small-bore tubes presented a large heating surface to the water. The flue gases passed out of the boiler at the front and into an induced draught fan, which passed them into the chimney.

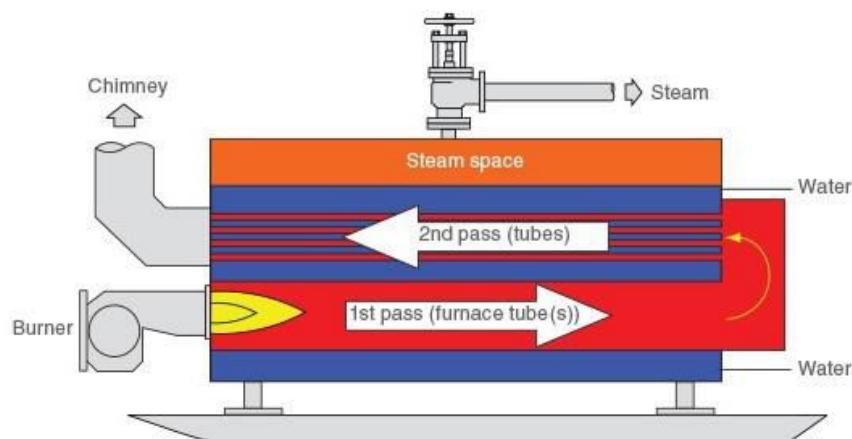


Fig II.17 Economic boiler (two-pass, dry back).

Capacity	Small	Large
Dimensions	3 m long * 1.7 m diameter	7m long * 4m diameter
Output	1000kg/h	15000kg/h
Pressure	Up to 17 bar g	Up to 17 bar g

Table II.2 Size range of two-pass, dry back economic boilers.

II.9.1.2.2. Economic boiler (three-pass, wet back):

A further development of the economic boiler was the creation of a three-pass wet back boiler which is a standard configuration in use today, (see Figure 3.2.4).

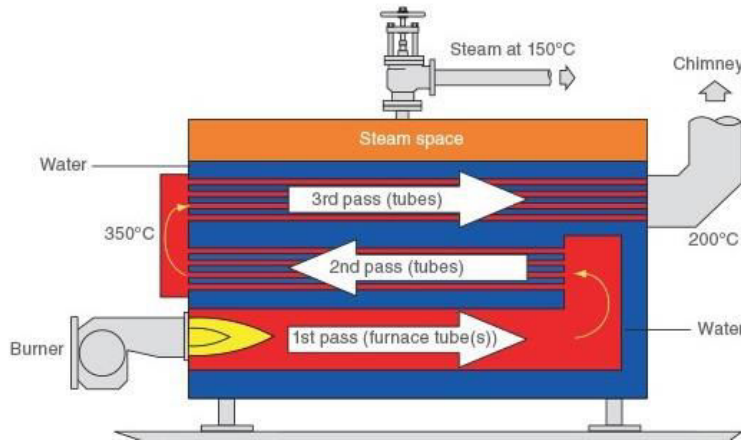


Fig II.18 Economic boiler (three-pass, wet back).

The design has evolved as materials and manufacturing technology has advanced: thinner metal tubes were introduced allowing more tubes to be accommodated, the heat transfer rates to be improved, and the boilers themselves to become more compact [13].

Typical heat transfer data for a three-pass, wet back, economic boiler is shown in Table II.3.

	Area of tubes	Temperature	Proportion of total heat transfer
1 st pass	11m ²	1600° C	65%
2 nd pass	43m ²	400° C	25%
3 rd pass	46 m ²	350° C	10%

Table II.3 Heat transfer details of a modern three pass, wet back, economic boiler.

II.10. Burner:

The burner is used to heat the water in the boiler before it is injected into the heat emitters. Operating both with an oil boiler and with a gas boiler, the burner is fitted with a fan responsible for supplying the mechanical element with air. This operating principle has earned it the name of pulsed burner. This fan has a dual role:

- It allows the air necessary for combustion to enter the appliance
- It ensures the evacuation of fumes associated with combustion

The fuel oil or gas required for combustion enters the burner from its storage location: an oil tank or a town gas connection.

II.11. System Description:

We will therefore describe this system then define the actuators, the measuring devices and detection as well as start and stop conditions.

II.11.1. Boiler composition:

In this part the system is composed of a shell type boiler and 2 hydraulic valves. Automatic and manual hydraulic regulator valve. Burner RS 70 type, 2 gas valves: a regulating butterfly valve and security valve and pneumatic security valves.

II.11.2. Measurement and detection components:

After describing the elements, here is all the instruments used in this part

Assignements	Scale	Lower scale	Upper scale	Alarm lower scale	Alarm upper scale	Unit
Level sensor	0 to 100	75	81	70	85	%
Pressure sensor	0 to 15	10	11	9	12	Bar
Temperature sensor	0 to 200				160	C
Flower sensor	0 to 10000					kg/h

Table II.4 Measurement and detection components.

II.11.3. Actuators:

After the description of the instruments, here is the set of actuators used in this part:

Assignments	State
Water pump	Start /stop, Auto/Manual, principal/secondary
Burner	Auto
Butterfly gas valve	Auto
Regulator valve	Start/stop, Auto
Pneumatic valve	Auto/Manuel

Table II.5 actuators.

II.11.4. Operation conditions:

Start:

- Water pumps in auto mode (principal and secondary).
- water level sensor $\geq 75\%$.
- pressure level ≥ 10 bar.
- All valves in auto mode.

Once these conditions have been verified, the system will do the following tasks:

Opening of hydraulic and pneumatic valves.

Opening one of the water pumps.

Burner start functioning.

Stop:

System operation stops if at least one of the following conditions is true:

-Water level sensor $\geq 81\%$.

-pressure level ≥ 11 bar.

-temperature $\geq 160^{\circ}$ C.

II.12. Tuning of PID parameters:

II.12.1PID:

II.12.1.1.Introduction:

Process control plays an essential role in the safe manufacture of quality products at market demand, while protecting the environment. Flow rates, pressures and temperatures within pipes and vessels, inventories of liquids and solids, and product quality are all examples of measured variables that must be controlled to meet certain objectives. While there are several means available for controlling these variables, the PID family of controllers has historically carried the major share of this responsibility and, because of their simplicity and reliability, will continue to do so in the future. The term PID stands for proportional integral derivative and it is one kind of device used to control different process variables [14].

II.12.1.2. Proportional control:

The proportional element of PID examines the magnitude of the error and the PID control reacts proportionally. A large error receives a large response, for example if there is a large temperature error, the fuel valve would be opened a lot, while a small error would receive a small response. In mathematical terms, the proportional term (Pout) is expressed as:

$$Pout = Kp * e \quad (eq II.1)$$

Where

Pout : proportional portion of controller output.

Kp: Proportional gain.

e: Error signal .

$e = \text{set point} - \text{process variable}.$

$e = SP - PV$ here represents a reverse acting loop.

$e = PV - SP$, it refers to a direct acting loop.

Chapter II: Boiler Description and PID Control

In a direct acting loop, the process variable is greater than the set point; therefore, the appropriate controller action is to increase the output [15].

II.12.1.3. Integral Control:

To address the first issue with the proportional control, integral control attempts to correct a small error (offset). The integral examines the error over time and increases the importance of even a small error over time. The integral is equal to error multiplied by the time the error has persisted. A small error at time zero has zero importance. A small error at time 10 has an importance of 10 times error. As such, the integral increases the response of the system to a given error over time until it is corrected. The integral can also be adjusted, and the adjustment is called the reset rate. The reset rate is a time factor. The shorter the reset rate, the quicker the correction of an error. The mathematical expression of an integral-only controller is:

$$I_{out} = \frac{1}{T_i} \int e \, dt = K_i \int e \, dt \quad (\text{eq II.2})$$

Where:

I_{out} : Integral portion of controller output.

T_i : Integral time, or reset time.

K_i : Integral gain.

e : Error signal, $e = \text{set point} - \text{process variable}$. [15]

II.12.1.4. Derivative Control:

The derivative part of the control output attempts to look at the rate of change in the error signal. The derivative will cause a greater system response to a rapid rate of change than to a small rate of change. In other words, if a system's error continues to rise, the controller must not be responding with sufficient correction.

The derivative senses this rate of change in the error and provides a greater response. The derivative is adjusted as a time factor and therefore is also called rate time. It is essential that too much derivative should not be applied or it can cause overshoot or erratic control. In mathematical terms, the derivative term D_{out} is expressed as:

$$D_{out} = T_d \frac{d}{dt} e = K_d \frac{d}{dt} e \quad (\text{eq II.3})$$

Where

D_{out} : Derivative portion of controller output.

Chapter II: Boiler Description and PID Control

Td: Derivative time.

Kd: Derivative gain.

e : Error signal, $e = \text{set point} - \text{process variable}$.

To summarize the tasks of all three controls, proportional control causes an input signal to change as a direct ratio of the error signal variation. It responds immediately to the current tracking error, but it cannot achieve the desired set point accuracy without an unacceptably large gain. Thus, the proportional term usually needs the other terms. Integral control causes the output signal to change as a function of the integral of the error signal over time duration. The integral term yields zero steady state error in tracking a constant set point, it also rejects constant disturbances, derivative action reduces transient errors and causes an output signal to change as a function of the rate of change of the error signal, the contributions of the three terms will yield the control output or the control variable:

$$\text{Control Variable} = P_{out} + I_{out} + D_{out} .$$

- Use K_p to decrease the rise time.
- Use K_D to reduce the overshoot and settling time.
- Use K_I to eliminate the steady-state error.

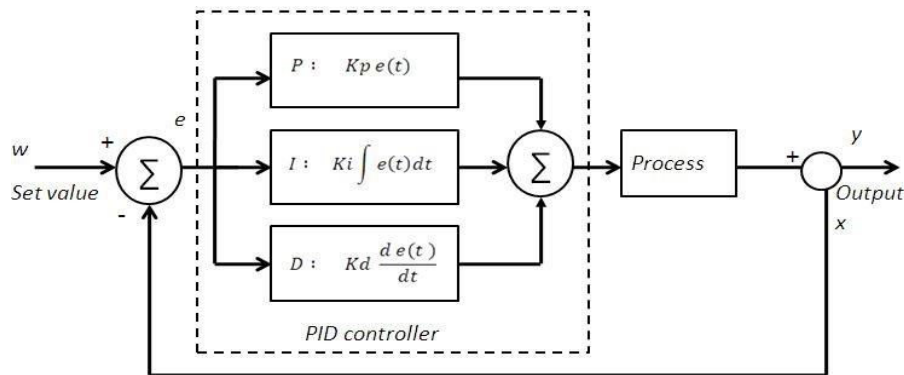


Fig II.19 Structure of the PID controller.

In this project we are going to use **Ziegler-Nichols** tuning method because of its simplicity and efficiency of getting the best values of the PID controller [15].

II.12.2. Ziegler and Nichols tuning method:

In 1942 Ziegler and Nichols, both employees of Taylor Instruments described simple mathematical procedures for tuning PID controllers. These procedures are now accepted as standard in control systems practice. Ziegler-Nichols formula for specifying the controllers are based on plant step responses [16]. There exist 2 methods of tuning the parameters which are the open loop and the closed loop methods. In our case we will use the open loop method or what called the reaction curve process.

II.12.2.1. Steps to determine PID controller parameters:

Using the reaction curve process:

1. Feed the unit step signal to the process in an open loop and plot the response.
2. Draw a tangent line at the inflection point and let it cross the steady state level and the 0 axes line.
3. Now τ is equal to the difference between where the tangent line crosses the 0 axe, and T is equal to (the intersection between the tangent line and the steady state level – τ).
4. The controller gain is equal to the steady state level.

By following these steps we can fill the table and define the parameters of the PID transfer function which is as follow:

$$G_c(S) = K(1 + \frac{1}{T_i.S} + T_d.S). \quad (\text{eq II.4})$$

PID type	K_p	T_i	K_d
P	$\frac{T}{k\tau}$	/	/
PI	$0.9\frac{T}{k\tau}$	0.33τ	/
PID	$1.2\frac{T}{k\tau}$	2τ	0.5τ

Table II.6 Tuned parameters based on Zeigler-Nichol method.

In order to proceed in the Ziegler-Nichols method we need to find the transfer function of our process.

The response of the level can be described as:

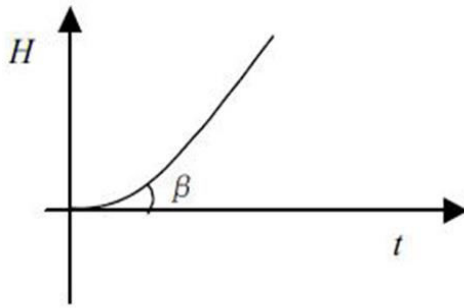


Fig II.20 Water level response.

The transfer function of the response can be given by:

$$G(s) = \frac{L(s)}{T(s)} = \frac{k}{(1+Tp1.s)(1+Tp2.s)} \quad (\text{eq II.5})$$

Where k is the flying-up speed of the step response, $-1/Tp1$ and $-1/Tp2$ are the poles of the transfer function, $L(s)$ is the output level inside the water tank and $T(s)$ is the timing of the filling.

II.12.3. Case Study:

through the data collected from the implementation where we used a 12 volts pump factory.

We have got the following table

Time	Water level
0	0
1	0.005
2	0.012
3	0.022
4	0.034
5	0.05
6	0.07
7	0.095
8	0.125
9	0.16

10	0.195
11	0.23
12	0.265
13	0.295
14	0.33
15	0.36
16	0.395
17	0.425
18	0.46
19	0.495
20	0.53

Table II.7 Water level versus time.

Due to the instrument used to capture the water level we could take measurement each 1 sec, we have plotted the graph representing these data and we obtained it to be as follow:

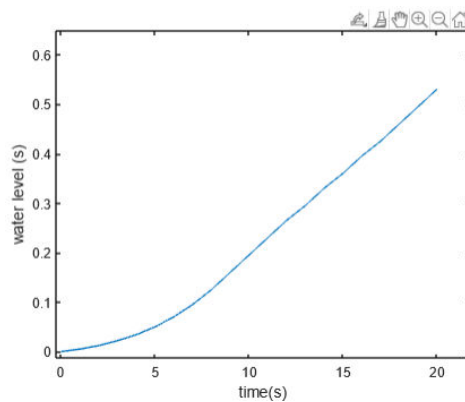


Fig II.21 Water level response of our process.

In order to identify this process we have 2 methods. The least square method and the ident tool that exist on MATLAB.

II.12.4. System identification using ‘ident’ tool:

In this part we are going to identify our system using the” ident” tool in MATLAB

Chapter II: Boiler Description and PID Control

First of all we have to enter the data to the workspace as matrices as shown in the following figure

```
>> t=[0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
1]

L =
0
0.0050
0.0120
0.0220
0.0340
0.0500
0.0700
0.0950
0.1250
0.1600
0.1950
0.2300
0.2650
0.2950
0.3300
0.3600
0.3950
0.4250
0.4600
0.4950
0.5300
```

Fig II.22 Time and water level entered in the workspace.

After typing” ident “in the workspace we have got the following figure:

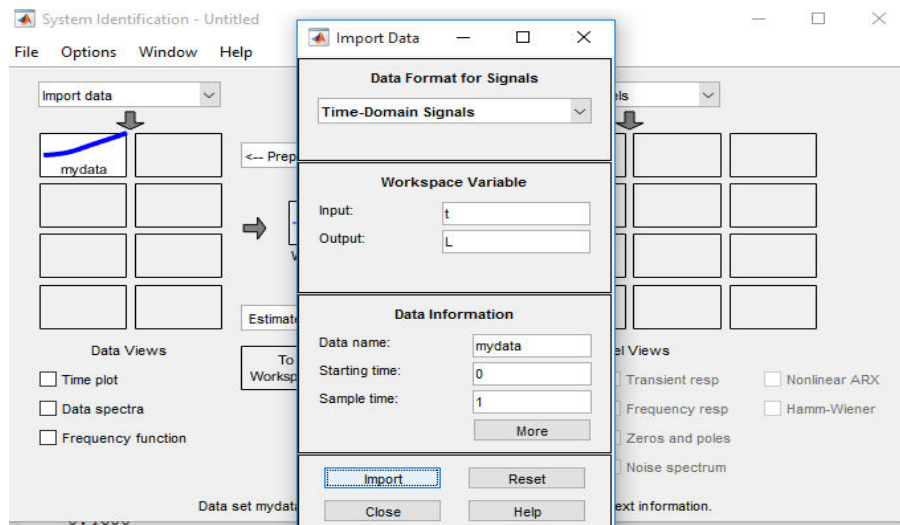


Fig II.23 Importing input and output.

We just have to type the variable names at input and output and setting the starting time to 0 and we press import. After that we need to know the best approximation of our system, in order to do that we enter the option “transfer function identification”. We can estimate the number of poles and zeros.

Chapter II: Boiler Description and PID Control

In this case after the assumption of 2 poles and 0 zeros we have got the following window.

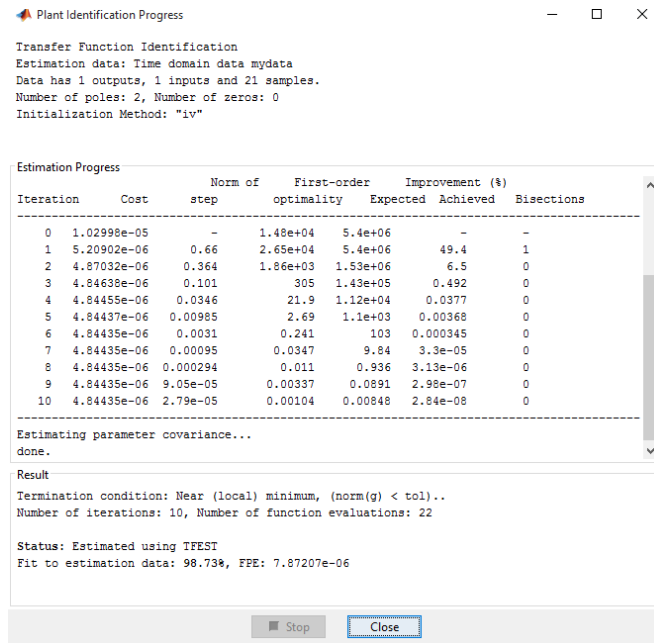


Fig II.24 transfer function identification.

It has been seen that the percentage of the estimation is equal to 98.738 %, which is a very good value that indicate we have a good approximation of the transfer function .After that we need to find the value of these 2 poles .To do that we have to enter the option “process model”. After we select the number of poles and zeroes and by pressing estimate, we have got the following window:

Process Models

Transfer Function

$$\frac{K}{(1 + Tp1 s)(1 + Tp2 s)}$$

Poles
2 All real

☐ Zero
☐ Delay
☐ Integrator

Disturbance Model: None
Focus: Simulation

Initial condition: Auto
Covariance: Estimate

☐ Display progress

Name: P2

Estimate Close Help

Par	Known	Value	Initial Guess	Bounds
K	<input type="checkbox"/>	0.034202	Auto	[-Inf Inf]
Tp1	<input type="checkbox"/>	1.7444	Auto	[0 10000]
Tp2	<input type="checkbox"/>	2.4221	Auto	[0 10000]
Tp3	<input type="checkbox"/>	0	0	[0 Inf]
Tz	<input type="checkbox"/>	0	0	[-Inf Inf]
Td	<input type="checkbox"/>	0	0	[0 Inf]

Initial Guess
☒ Auto-selected
☐ From existing model:
☐ User-defined: Value-->Initial Guess

Regularization... Options...

Continue

Fig II.25 process model window.

Chapter II: Boiler Description and PID Control

As it has been seen in the figure 4 we have got the values of the poles and the gain of the transfer function, which are as follow:

$$k=0.034202.$$

$$Tp1=1.7444.$$

$$Tp2=2.4221.$$

After replacing these values in the shown transfer function we got $G(s)$

$$G(s) = \frac{k}{(1+Tp1.s)(1+Tp2.s)} \quad (\text{eq II.6})$$

After putting the numerical values of k , $Tp1$ and $Tp2$ we get:

$$G(s) = \frac{0.034202}{(1+1.7444.s)(1+2.4221.s)} \quad (\text{eq II.7})$$

After numerical simplification of equation II.7 we get:

$$G(s) = \frac{0.034202}{4.22s^2 + 4.16s + 1} \quad (\text{eq II.8})$$

The least square method is done in the appendix where we verified this result.

After getting the transfer function now we can tune PID parameters using Zeigler-Nichol method.

We have to follow the stated rules on part II.1 to get the values of Kp and Ki .

Driving the transfer function $G(s)$ with a unit step function as shown in the following figure

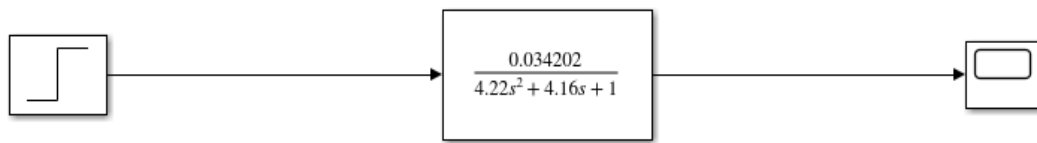


Fig II.26 Driving the transfer function $G(s)$ with a unit step.

Chapter II: Boiler Description and PID Control

By clicking on the scope, we get the response in the following figure

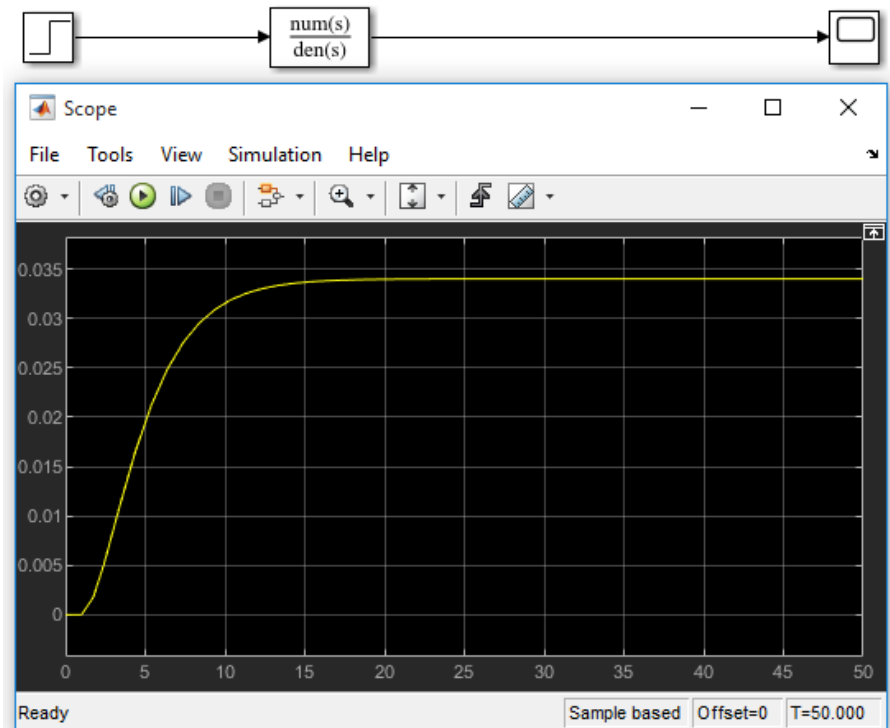


Fig II.27 the response of $G(s)$ to a unit step input.

In order to find the parameters, we need to draw a tangent line at the inflection point which will be specified in the following figure



Fig II.28 Process response to the unit step with the tangential lines drawn on it

Chapter II: Boiler Description and PID Control

By looking at the figure 7 we can find τ , T and k which will be as follow

$$k = 0.034$$

$$\tau = 1.2 \text{ s}$$

$$T = 7.8 - 1.2 = 6.6 \text{ s}$$

Now we can calculate PID parameters for each type of controller by replacing numerical values on the equation of **table II.6**.

For P controller:

$$K_p = \frac{6.6}{0.034 \times 1.2} = 161.76.$$

T_i and T_d are undefined.

For PI controller:

$$K_p = 0.9 \times \frac{6.6}{0.034 \times 1.2} = 145.6.$$

$$T_i = 0.33 \times 1.2 = 0.39 \text{ s.}$$

and T_d is undefined.

For PID controller:

$$K_p = 1.2 \times \frac{6.6}{0.034 \times 1.2} = 194.11$$

$$T_i = 2 \times 1.2 = 2.40 \text{ s}$$

$$T_d = 0.5 \times 1.2 = 0.60$$

And then all the calculation can be summarized in **table II.8**:

PID type	K_p	T_i	T_d
P	161.76	/	/
PI	145.60	3.9	/
PID	194.11	2.40	0.60

Table II.8 PID parameter after zeigler-nichol's tuning method.

When dealing with level control, PI are the best for this kind of process because of the inaccuracies due to the offsets in P-only control. Derivative control is not considered due to the rapid fluctuations in flow dynamics with lots of noise.

So the transfer function of the PI controller can be written as follow

$$G_c(s) = K_p \left(1 + \frac{1}{T_i s} \right). \quad (\text{eq II.9})$$

Chapter II: Boiler Description and PID Control

After we replace Kp and Ti we get

$$G_c(s) = 145.60 \left(1 + \frac{1}{3.9s} \right) \quad (\text{eq II.10})$$

Now we are going to record the response of PI controller

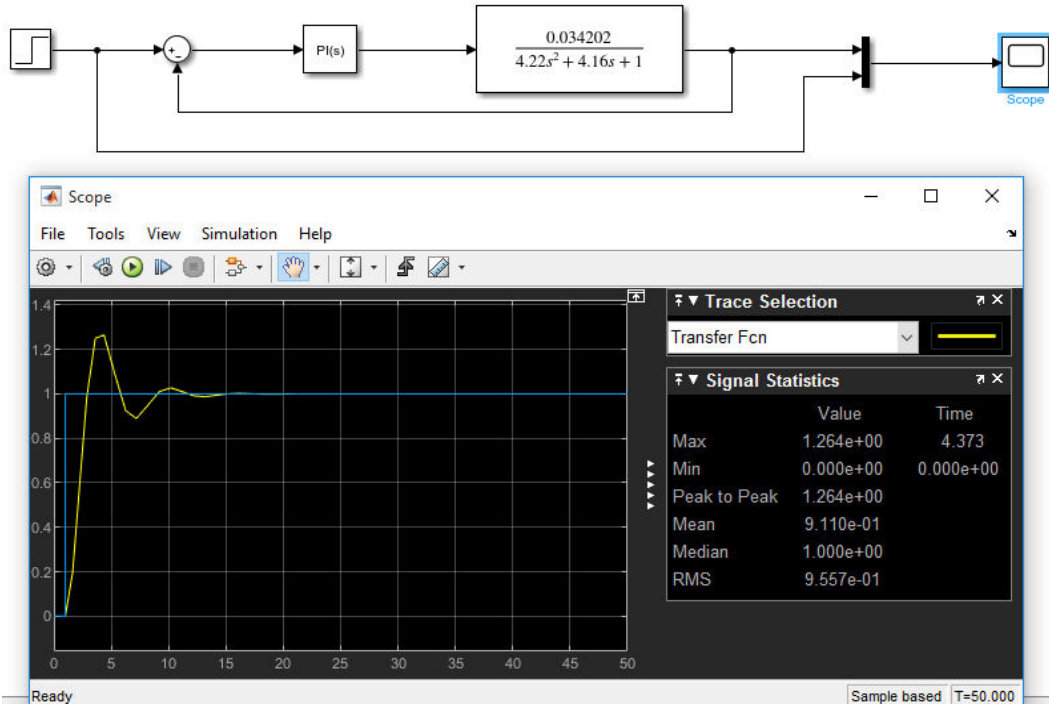


Fig II.29 Closed loop response with PI controller of the tuned parameters.

From the response above, we can see that the system is stable with a considerable peak overshoot.

II.13. Conclusion:

At the end of this chapter we can say that we could successfully describe the system with all its components like sensors and actuators in order to facilities things to the upcoming programming part. We could identify the transfer function of the pump using two methods which are the ident tool using MATLAB and the least square method.

After that we tuned the parameters using Zeigler Nichole method which helped us to stabilize the system.

Chapter III

SIMATIC MANAGER Software Description and Programming

III.1.Introduction:

After introducing the boiler system and how it functions in the previous chapter where pressure and water level have to be measured and monitored for safe and reliable operation of the system. In this chapter we will program the PLC S7-300 stated in the first chapter using SIMATIC manager software STEP 7.

III.2. Overview of STEP 7:

STEP 7 is the standard software package used for configuring and programming SIMATIC programmable logic controllers. It is part of the SIMATIC industry software. There are the following versions of the STEP 7 Standard package:

- STEP 7 Micro/DOS and STEP 7 Micro/Win for simpler stand-alone applications on the SIMATIC S7-200.
- STEP 7 for applications on SIMATIC S7-300/S7-400, SIMATIC M7-300/M7-400, and SIMATIC C7 with a wider range of functions [4].

III.2.1. Basic Tasks:

When we want to create an automation solution with STEP 7, there are a series of basic tasks. The following figure shows the tasks that need to be performed for most projects and assigns them to a basic procedure.

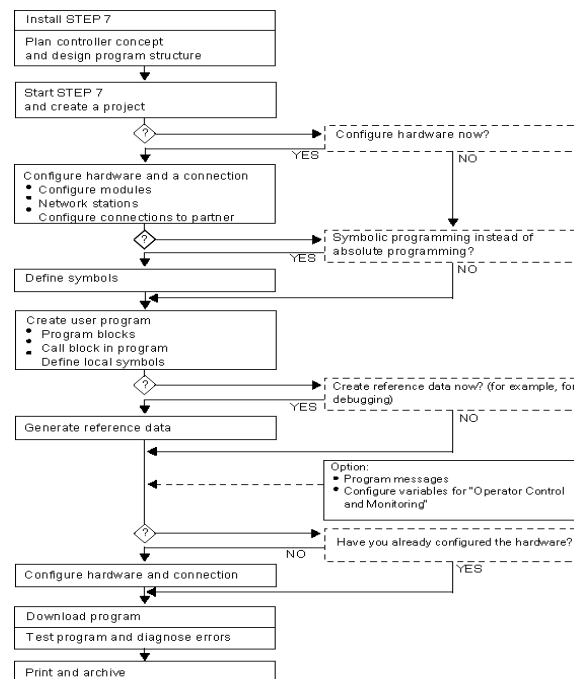


Fig III.1 Basic Tasks to perform a project [4].

III.2.1.1 Alternative Procedures:

As shown in figure III.1, we have two alternative procedures:

- We can configure the hardware first and then program the blocks.
- We can program the blocks first without configuring the hardware. This is recommended for service and maintenance work, for integrating programmed blocks into in an existing project.

III.2.1.2. Brief Description of the Individual Steps:

- **Install STEP 7 and license keys**

The first time we use STEP 7, install it and transfer the license keys from diskette to the hard disk

- **Plan our controller**

Before we work with STEP 7, we need to plan our automation solution from dividing the process into individual tasks to create a configuration diagram.

- **Design the program structure**

We need to turn the tasks described in the draft of our controller design into a program structure using the blocks available in STEP 7.

- **Start STEP 7**

We start STEP 7 from the Windows user interface.

- **Create a project structure**

A project is like a folder in which all data are stored in a hierarchical structure and are available to us at any time. After we have created a project, all other tasks are executed in this project.

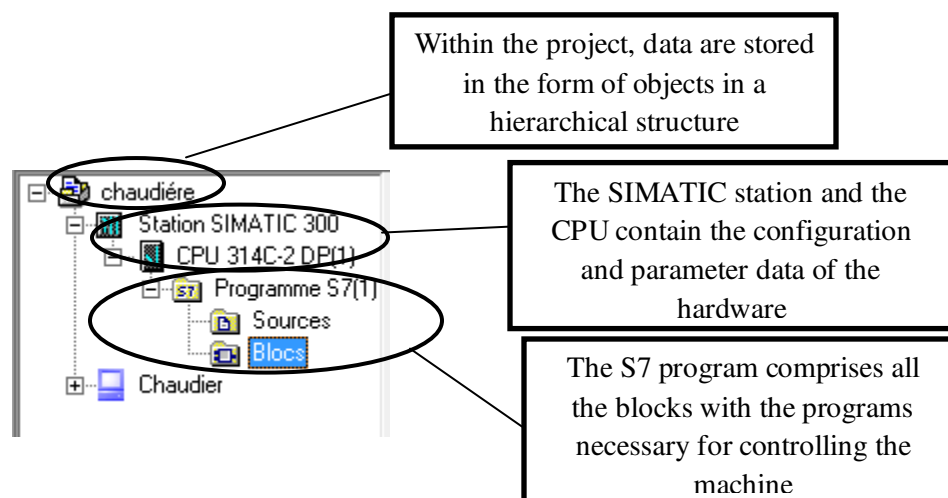


Fig III.2 A project structure.

- **Configure a station**

When we configure the station, we specify the programmable controller we want to use; for example, SIMATIC 300, SIMATIC 400.

- **Configure hardware**

When we configure the hardware, we specify in a configuration table which modules we want to use for our automation solution and which addresses are to be used to access the modules from the user program. The properties of the modules can also be assigned using parameters.

Slot	Module	Order number	Firmware	MPI address	I address	Q address
1	PS 307 10A	6ES7 307-1KA00-0AA0				
2	CPU 314C-2 DP(1)	6ES7 314-6CG03-0AB0	V2.6	2		
2.2	DP				1023*	
2.2	DI24/DO16				124...126	124...125
2.3	AI5/AO2				752...761	752...755
2.4	Comptage				768...769	768...769
2.5	Positionnement				784...799	784...799
3						
4	AI2x12Bit	6ES7 331-7KB00-0AB0			256...259	
5	AI2x12Bit	6ES7 331-7KB01-0AB0			272...275	
6	AI2x12Bit	6ES7 331-7KB02-0AB0			288...291	
7	AI2x12Bit	6ES7 331-7KB00-0AB0			304...307	

Table III.1 Hardware configuration table.

-Rack 1: power supply module PS 307 10A

-Rack 2: CPU 314

-Rack 3: Interface module (empty)

-Rack 4, 5, 6 and 7: Analog inputs

- **Configure networks and communication connections**

The basis for communication is a pre-configured network. For this we will need to create the subnets required for our automation networks, we set the subnet properties, and set the network connection properties and any communication connections required for the networked stations.

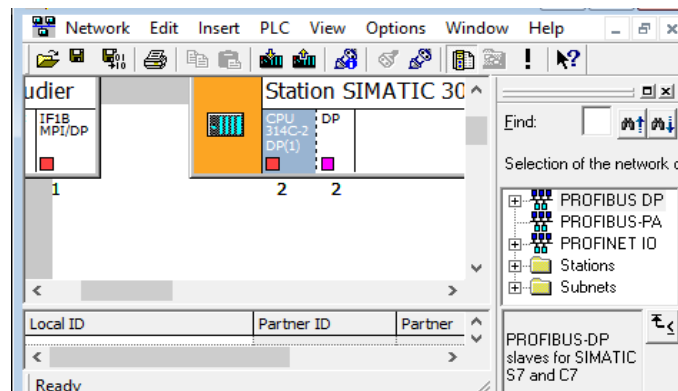


Fig III.3 networks Configuration and communication connections.

- **Define symbols**

We can define local or shared symbols, which have more descriptive names in a symbol table to use instead of absolute addresses in your user program.

Statu	Symbol /	Address	Data type	Comment
	add1	MD 1300	REAL	
	AL_BURNER	MW 204	WORD	
	Al_br	MW 203	WORD	alarm burner
	AL_temp	MW 305	WORD	alarm temperature
	Alarm	MW 102	WORD	alarmr
	AU	M 201.0	BOOL	emergency pushbutton
	AU_S	MW 200	WORD	
	AUTO_MODE	I 1.7	BOOL	auto-mode switch
	BI_POLAR2	M 120.6	BOOL	
	BT_stop	I 0.1	BOOL	stop pushbotton
	BURNER	Q 0.2	BOOL	burner
	but_valve	I 0.6	BOOL	butterfly valve
	c_hmi_water	MW 900	INT	
	C_L_Water_ch	MD 364	REAL	set point for HMI
	C_LEVEL	MD 290	REAL	
	clevel	IW 4	WORD	water level analog input
	cnt	M 100.3	BOOL	
	consigne	MD 201	REAL	setpoint
	CONT_C	FB 41	FB 41	Continuous Control
	Cycle Execution	OB 1	OB 1	
	D_SEL	M 200.4	BOOL	derivative select
	DJ_p1_S	MW 201	WORD	
	DJ_p2_S	MW 202	WORD	
	Em.stop	I 0.4	BOOL	emergency stop
	ERROR	MD 300	REAL	error
	ff	MD 1000	DINT	
	fire_detector	I 0.5	BOOL	fire detector
	FLOW_RATE	MD 1600	REAL	flow rate
	gain	MD 220	REAL	gain
	hight	Q 0.7	BOOL	hight alarm
	I_SEL	M 200.3	BOOL	integral select
	L_Water_ch	MD 360	REAL	level water real values
	L_water_hmi	MW 700	INT	water level HMI
	LAMP_PUMP1	Q 1.3	BOOL	lamp pump1
	LAMP_PUMP2	Q 1.4	BOOL	lamp pump2

Table III.2 symbol table

- **Create the program**

Using one of the available programming languages we create a program linked to a module or independent from it and store it as blocks, source files or charts.

- **Generate and evaluate reference data**

We can make use of these reference data to make debugging and modifying our user program easier

- **Configure messages**

We create block-related messages for example with their texts and attributes. Using the transfer program, we transfer the message configuration data created to the operator interface system database (for example, SIMATIC WINCC, SIMATIC PROTOOL).

- **Configure operator control and monitoring variables**

We create operator control and monitoring variables once in STEP 7 and assign them to the required attributes. Using the transfer program, we transfer the operator control and monitoring variables created to the database of the operator interface system WINCC.

- **Download programs to the programmable controller**

After all configuration, parameter assignment and programming tasks are completed, we can download our entire user program or individual blocks from it to the programmable controller (programmable module for our hardware solution). The CPU already contains the operating system.

- **Test programs**

For testing we can either display the values of variables from our user program or a CPU, we assign values to the variables or create a variable table for the variables that we want to display or modify [4].

III.2.2. Types of variables used in STEP7:

There are several types of distinct variables in STEP7, these variables are declared before each start of the program. The following table brings together these different variables:

TYPE	Size
Bool	1 bit
Real	32 bits
Int	16 bits
Byte	8 bits
Word	16 bits
DWord	32 bits
DInt	32 bits
Char	8 bits
Time	32 bits
S5Time	16 bits
Date	16 bits
Time of Day	32 bits

Table III.3 types of variables used in STEP7 [1].

III.3. Programming Languages:

The programming languages Ladder Logic, Statement List and Function Block Diagram for S7-300 and S7-400 are an integral part of the standard package.

- Ladder Logic (or LAD) is a graphic representation of the STEP 7 programming language. Its syntax for the instructions is similar to a relay ladder logic diagram. Ladder allows us to track the power flow between power rails as it passes through various contacts, complex elements and output coils.
- Statement List (or STL) is a textual representation of the STEP 7 programming language similar to machine code. If a program is written in Statement List, the individual instructions correspond to the steps with which the CPU executes the program. To make programming easier, Statement List has been extended to include some high-level language constructions (such as structured data access and block parameters).
- Function Block Diagram (FBD) is a graphic representation of the STEP 7 programming language and uses the logic boxes familiar from Boolean algebra to represent the logic. Complex functions (for example, math functions) can be represented directly in conjunction with the logic boxes [4].

III.4. Program Processing:

III.4.1. Linear program processing:

Linear program processing as shown in figure III.4 is mostly used for simple, not too comprehensive controllers and can be implemented in a single organization block (OB). In this connection, the control unit processes the statements in the order in which they are stored in the program memory when the end of program (BE) is reached, program processing starts from the beginning again. The time that a control unit needs to carry out processing of all the statements once is called the cycle time. This is why this type of processing is also referred to as cyclical processing [2].

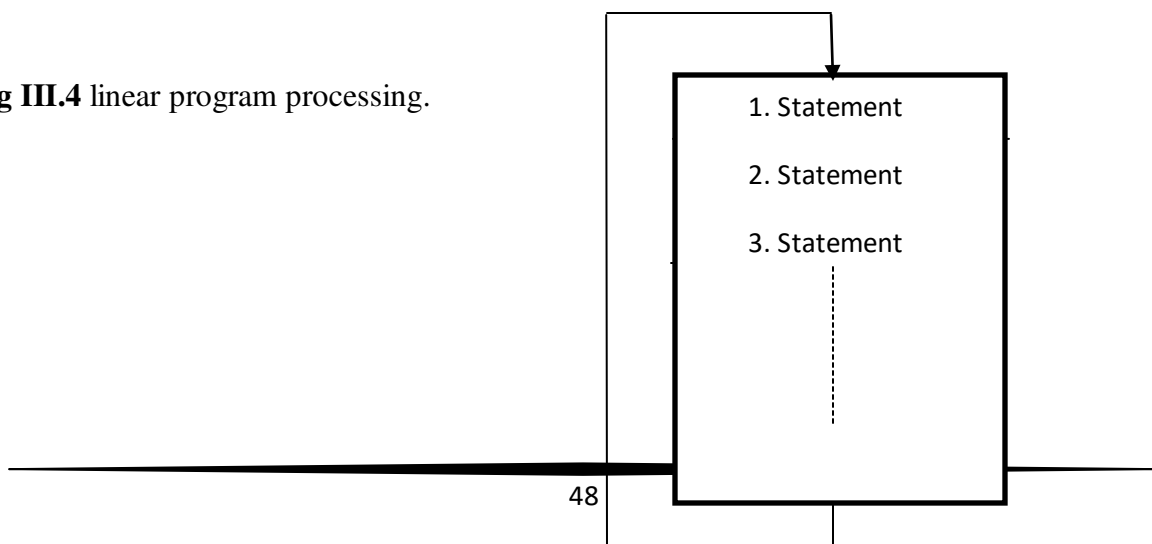


Fig III.4 linear program processing.

III.4.2. Structured programming:

With comprehensive control jobs, we divide the program into small manageable program blocks that are arranged by function. This has the advantage of allowing us to test program sections individually and if they function, to combine them into an overall function. STEP 7 provides the following user blocks for this: [2]

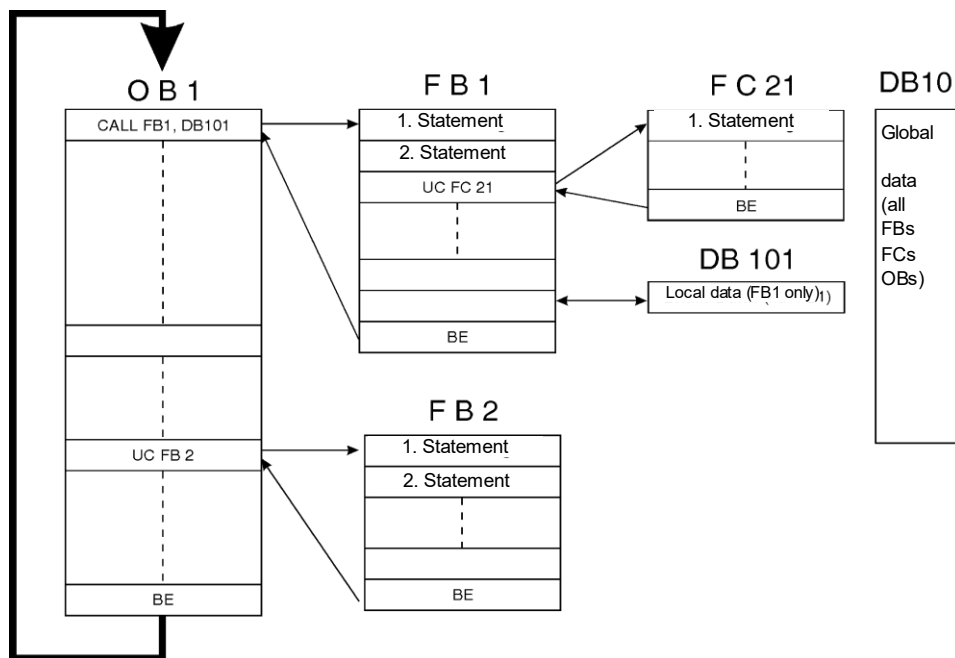


Fig III.5: A scheme of a STEP7 program blocks [2].

III.5. Types of a step7 program blocks:

III.5.1. Organization block (OB):

The operating system cyclically calls an OB which functions as the interface between the user program and the operating system. In the OB, the system informs the processor of the programmable controller (PLC) by means of block call commands about the program blocks that it has to process.

III.5.2. Function block (FB):

FBs have an assigned memory area. When an FB is called, it is possible to assign a data block (DB) to it and access the data in this instance DB by means of calls from the FB. An FB can be assigned to different DBs. It is possible to call further FBs and FCs via block call commands in a function block.

III.5.3. Function (FC):

A FC has no assigned memory area. The local data of a function is lost after processing of the function.

III.5.4. Data block (DB):

DBs are used to make available memory for data variables. There are two types of data blocks: global DBs where all the OBs, FBs and FCs read stored data or can write data themselves to the DB and instance DBs that are assigned to a specific FB [2].

III.6. Blocks programming:

The basic structure of programming is to create Functions (FC) or Function Blocks (FB) with the associated Data blocks (DB) and then gather all the functions and function blocks in the first Organization Block (OB1).

As an example, we are going to create and simulate the water level functional block and the rest of the program will be found in [APPENDIX].

We create a function FC1 and choose the ladder language as follows:

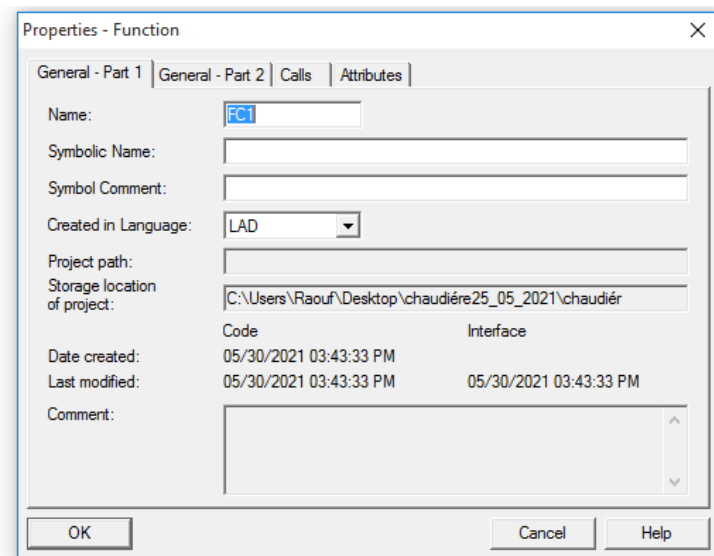


Fig III.6 creation of a new function.

We open the FC1 and enter the following instructions where the main function is to scale the analog input which is the water level and do the comparison between the water level and thresholds:

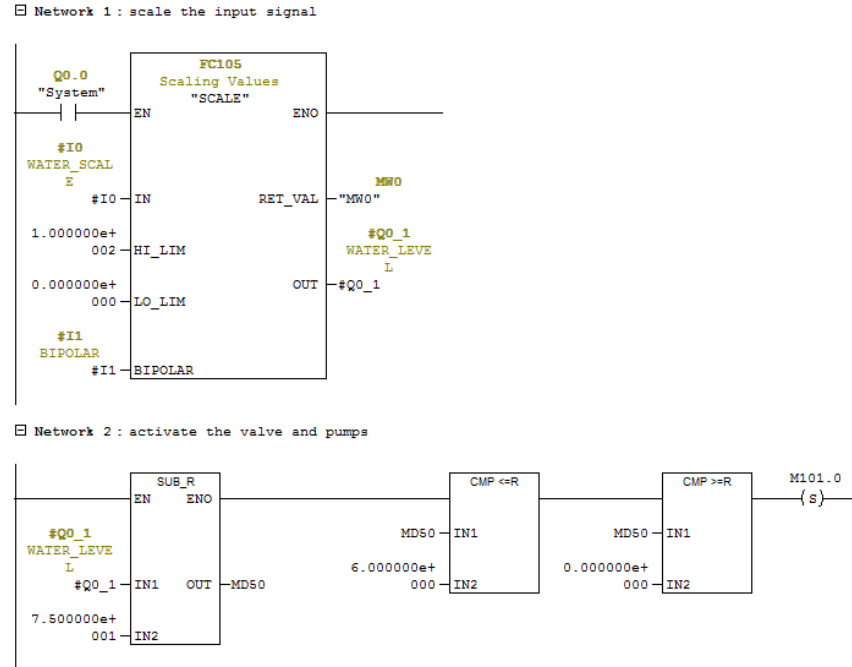


Fig III.7.a controlling water level function program in FC20.

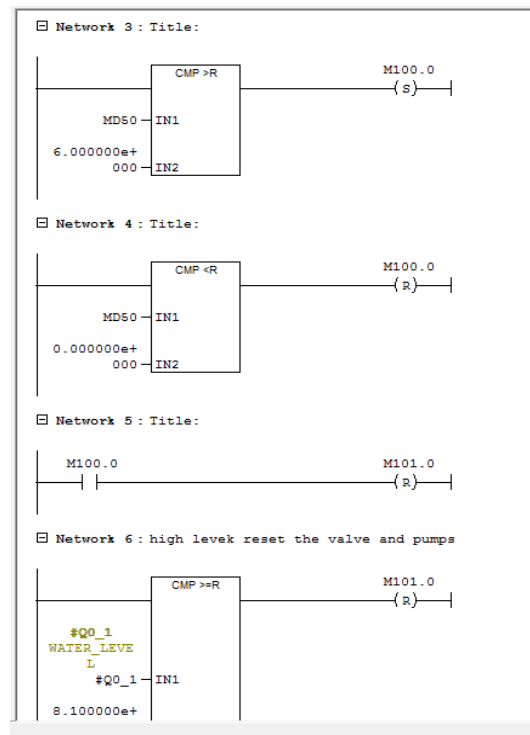


Fig III.7.b controlling water level function program in FC20.

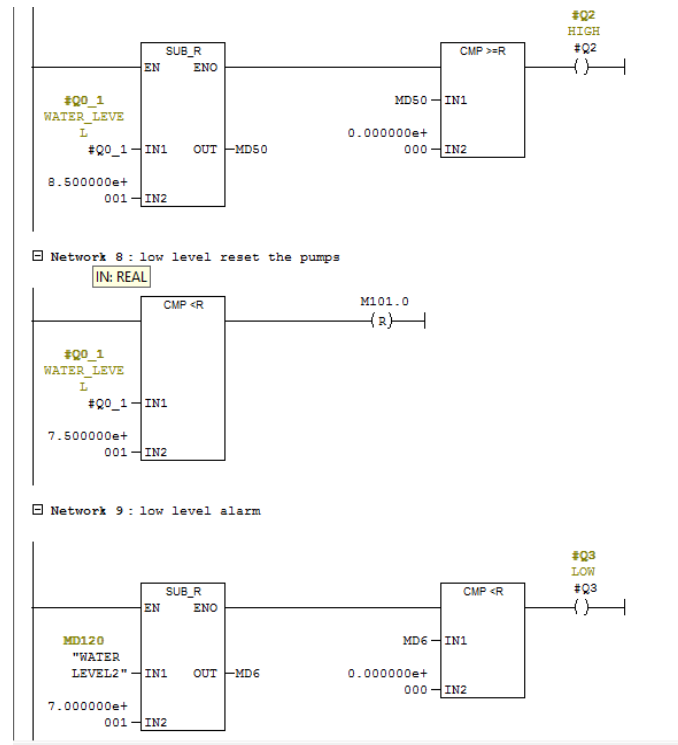


Fig III.7.c controlling water level function program in FC20.

After that, This FB1 must be brought to The OB1 which the Main program window, and then be inserted as a function as shown in the following figure

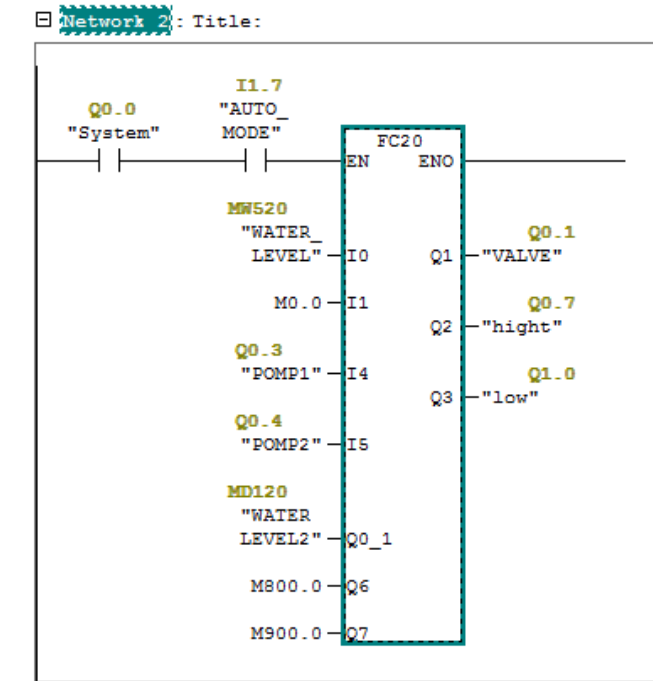


Fig III.8 Water level main program in OB1.

Finally, we need to turn on the online simulator and the program is transferred to the PLC in order to simulate it and test it and the results were satisfactory as shown below:

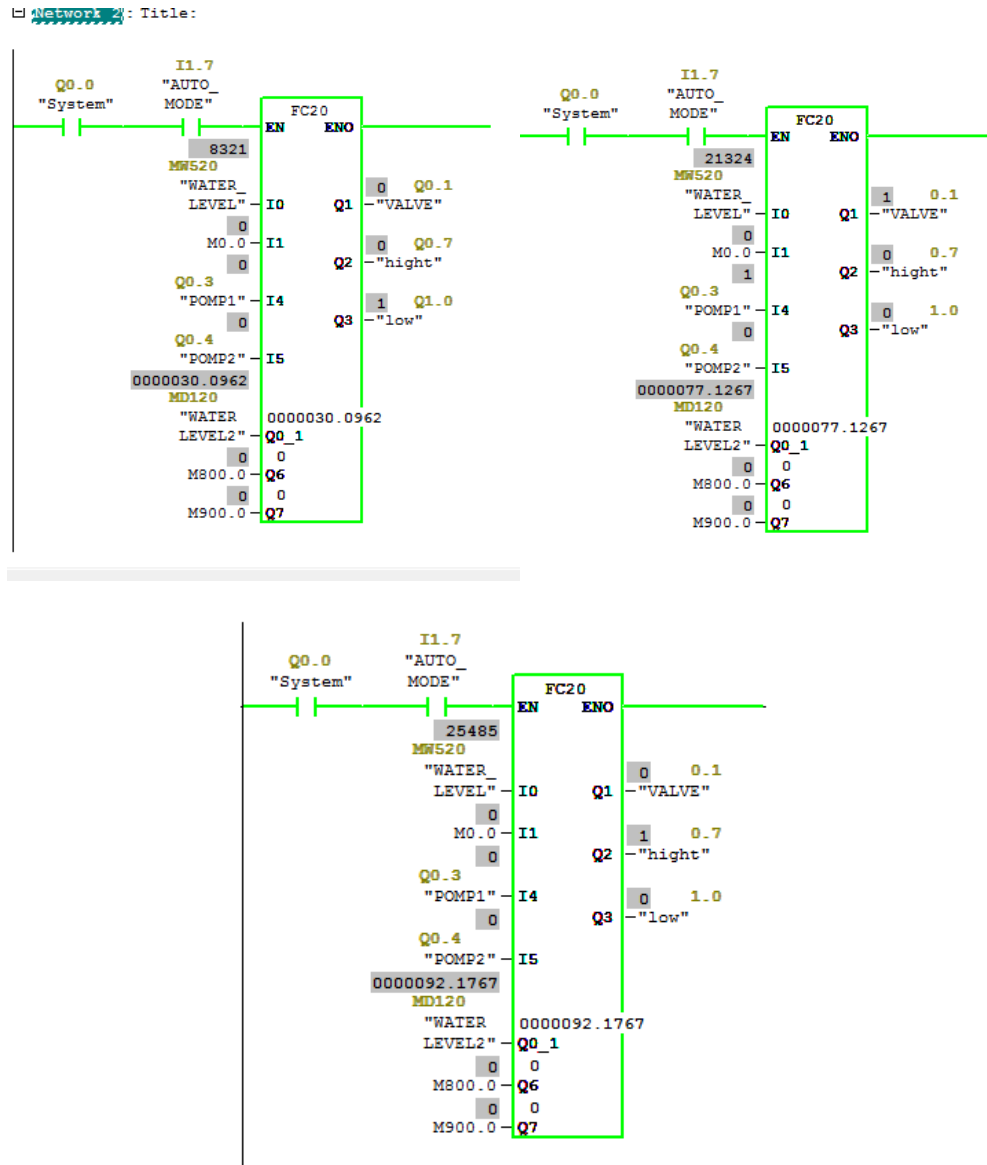


Fig III.9 simulation results of the FC20.

Results:

We can see that the function FC20 is work properly where:

- When the level is below 70% the low-level alarm is indicated.
- When the level is between 75% and 81% the valve and pumps are activated.
- When the level is higher than 85% the high-level alarm is indicated.

FC40 and FC50: the function FC40 and FC50 is programmed for activating and running the pump1 and pump2 sequentially in automatic mode where we can activate it by activating the valve in FC 20 and can be deactivated when an emergency or an overload cases occurred so the thermal relay will cut the power to the pump and an alarm will be on, also when the pump1 is operating the pump 2 must be off.

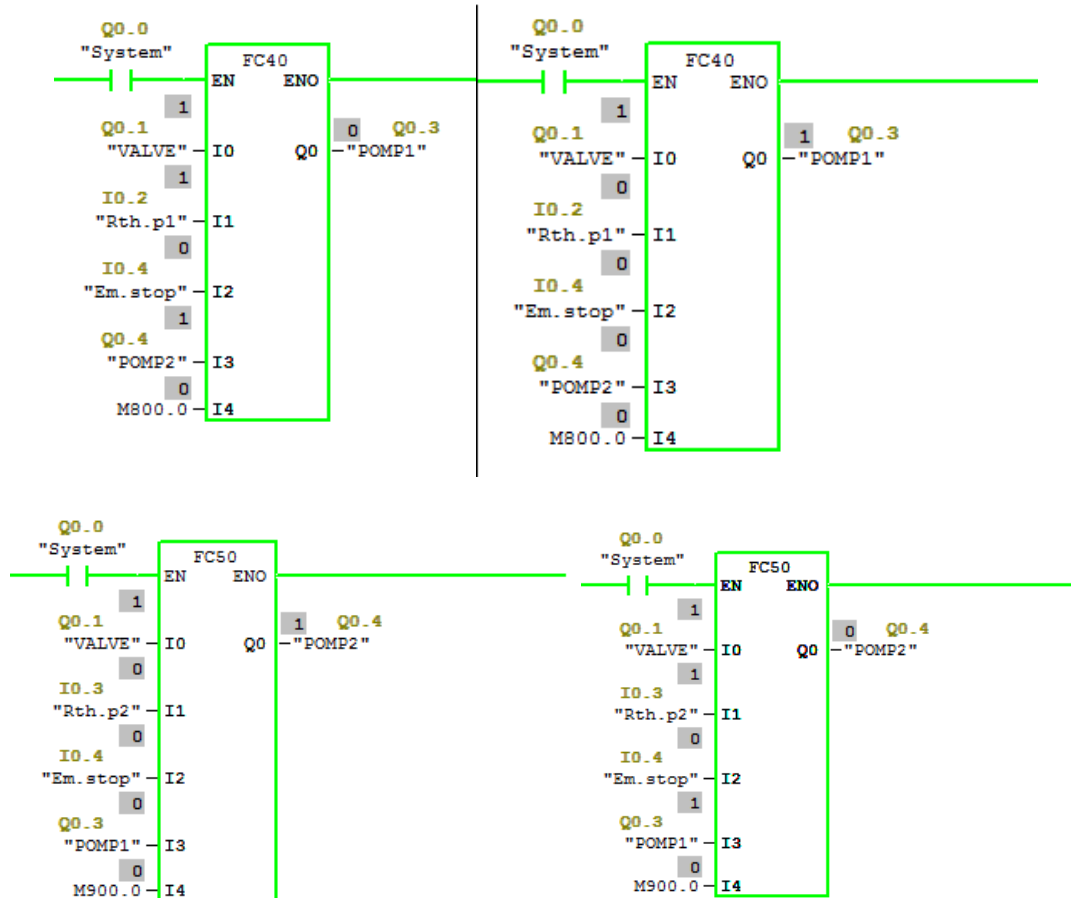


Fig III.10 pump1 and pump2 program in FC40 and FC50.

FC30: the function FC30 is programmed for activating and deactivating the burner where it needs electricity for the command circuit also gas so butterfly valve must be open, and for security it needs a sensor to indicate the fire, the burner will be deactivated if one of those conditions are not satisfied.

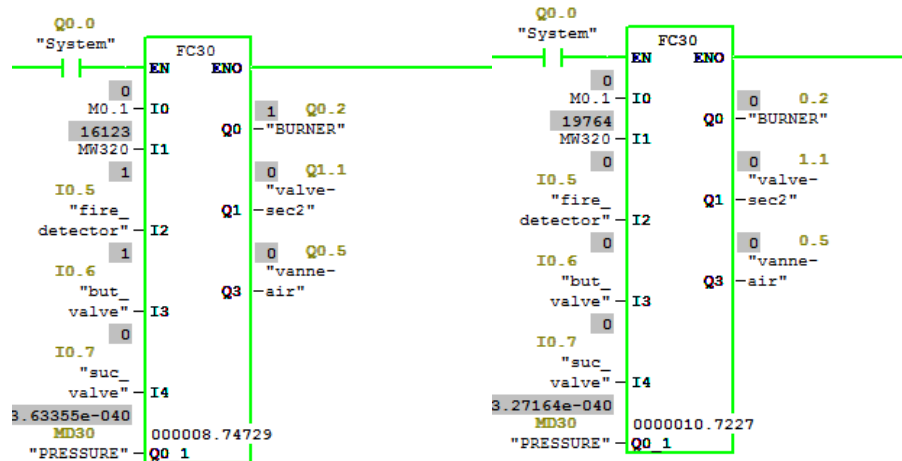


Fig III.11 Burner function program in FC30.

FC10: the function FC10 is programmed for PID controlling and it contains a PID control block where we can use the controller as a fixed set-point controller or in multi-loop controls as a cascade, blending or ratio controller. The functions of the controller are based on the PID control algorithm of the sampling controller with an analog signal

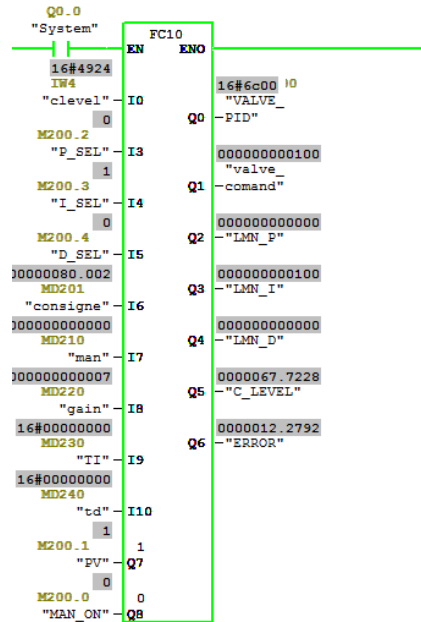


Fig III.12 Tank PID program in FC10.

FC80: the function 80 was created for controlling the pumps in manual mode where each pump consist of a power and control circuit where we can start the pump in one direction of travel with a push button S1 and stop it with an S0 push button and protect those pumps with a thermal relays, a flash lamps were used to indicate the operation of pumps and the occurrence of alarms.

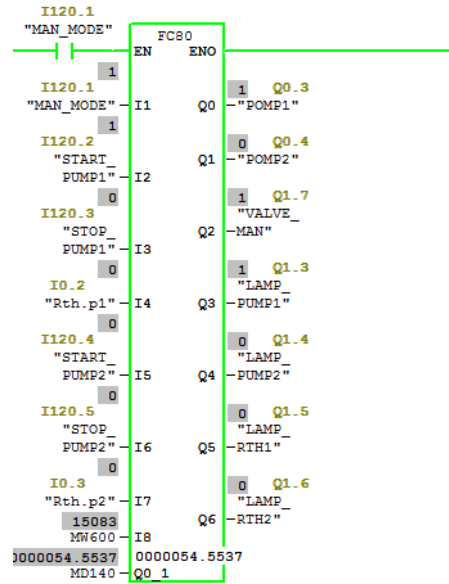


Fig III.13 controlling pumps in manual mode program in FC80.

II.7. Conclusion:

In this chapter, we have provided an overview to the procedures used to create our program for the PLC, we have also shown the different functions used in order to command the PLC that controls the boiler and drive it under a secured functioning.

Chapter IV

Simulation and supervision using WINCC & Implementing water level control

IV.1. Introduction:

Maximum transparency is essential for the operator who works in an environment where processes are becoming more complex, and requirements for machine and plant functionality are increasing. The Human Machine Interface (HMI) provides this transparency and represents the interface between man (operator) and process (machine/plant). The PLC is the actual unit which controls the process. Hence, there is an interface between the operator and WINCC flexible (at the HMI device) and an interface between WINCC flexible and the PLC.

An HMI system assumes the following tasks:

- **Process visualization:**

The process is visualized on the HMI device. The screen on the HMI device is dynamically updated. This is based on process transitions.

- **Operator control of the process:**

The operator can control the process by means of the GUI. For example, the operator can preset reference values for the controls or start a motor.

- **Displaying alarms:**

Critical process states automatically trigger an alarm, for example, when the set-point value is exceeded.

- **Archiving process values and alarms:**

The HMI system can log alarms and process values. This feature allows you to log process sequences and to retrieve previous production data.

- **Process values and alarms logging:**

The HMI system can output alarms and process value reports. This allows you to print-out production data at the end of a shift.

- **Process and machine parameter management:**

The HMI system can store the parameters of processes and machines in recipes. For example, you can download these parameters in one pass from the HMI device to the PLC to change over the product version for production [17].

IV.2. Elements of WINCC flexible:

WINCC flexible consists of the following elements:

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

- **Menus and Toolbars:**

You can access all the functions provided by WinCC flexible by means of its menus and toolbars.

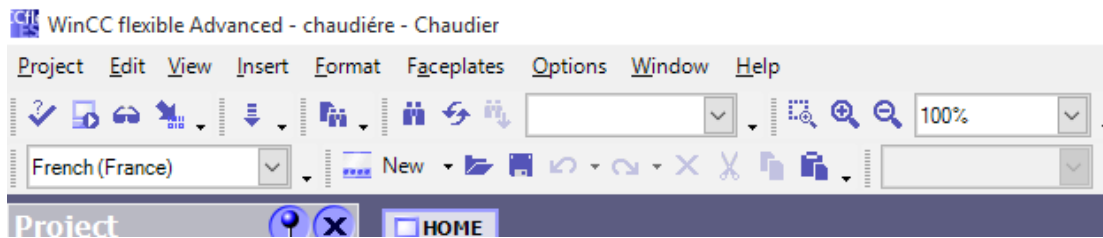


Fig IV.1 Menu and toolbox of WinCC flexible.

- **Work area:**

Project objects are edited in the work area. All WinCC flexible elements are arranged on the borders of the work area. With the exception of the work area, you can organize and configure.

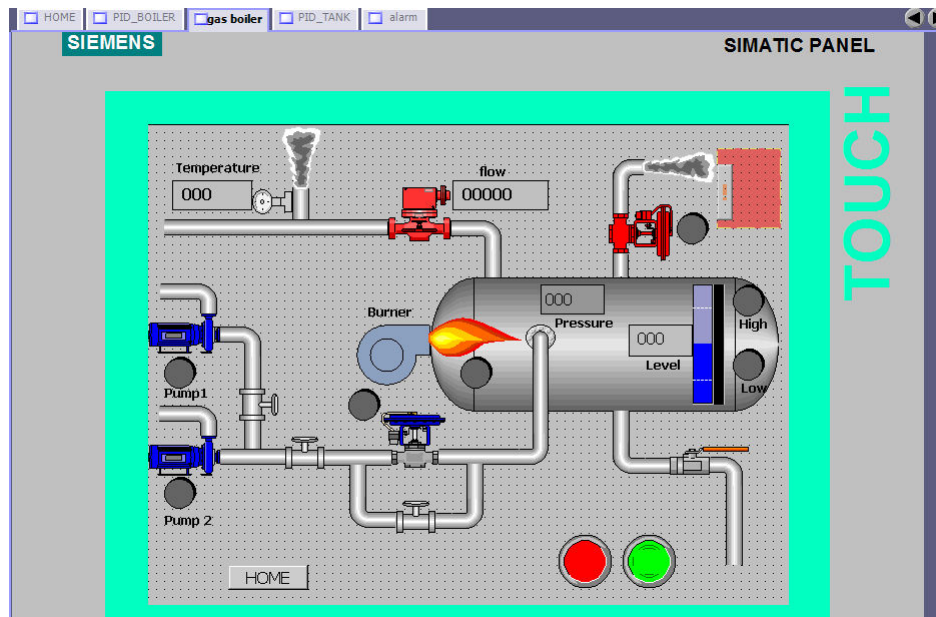


Fig IV.2 Work area of WinCC flexible.

- **Project View:**

All component parts and editors available in a project appear in a tree structure in the Project view. Folders are provided as sub-elements of each editor in which you can save objects in a structured way. In addition, direct access to the configured objects is available for screens, recipes, scripts, protocols and user dictionaries. In the project windows you have the access to the device settings of the HMI device, language settings and version management.

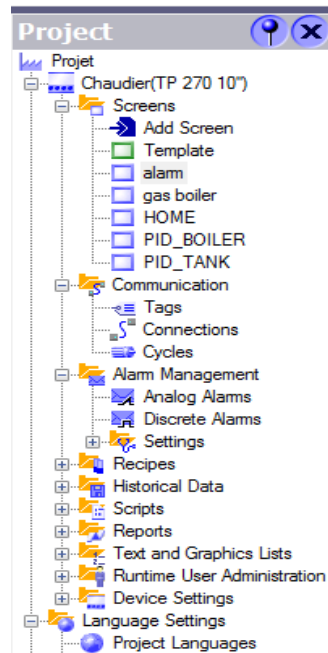


Fig IV.3 Project view of WinCC flexible.

- **Property view:**

The Property View is used to edit object properties, e.g. the color of screen objects. Property view is only available in specific editors.

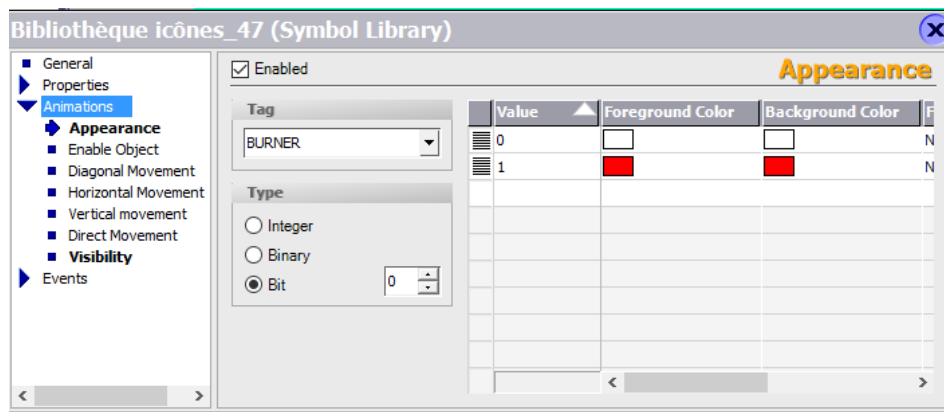


Fig IV.4 Property view of WinCC flexible.

- **Toolbox:**

The toolbox contains a selection of objects which you can add to your screens, e.g. image objects or operator control elements. In addition, the toolbox also provides libraries containing object templates and collections of faceplates.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

- **Library:**

The "Library" is an element of the Toolbox view which provides access to screen object templates. You can always add screen objects and thus increase programming efficiency either by multiple use or reuse of object templates. The library is the central database for storing frequently used objects, such as screen objects and tags.

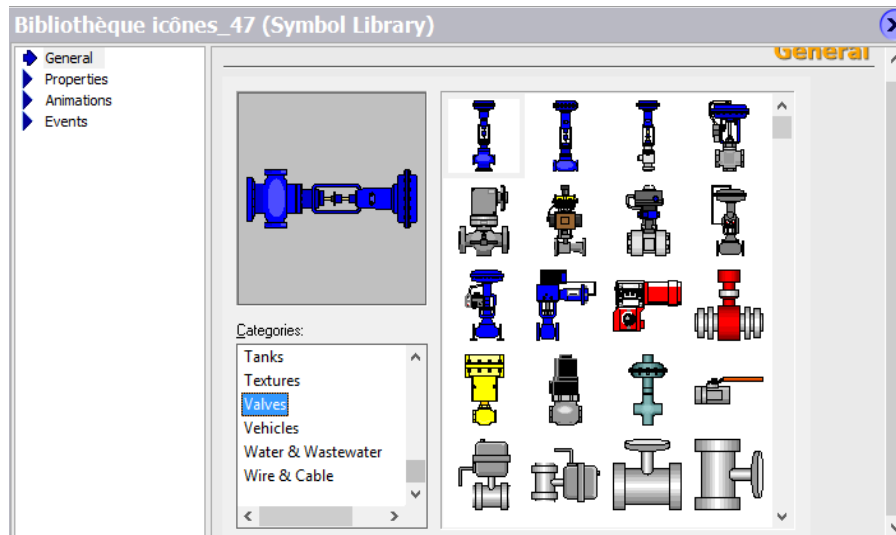


Fig IV.5 library or database of WinCC flexible.

- **Output View:**

The output window displays system alarms generated, for example, in a project test run.

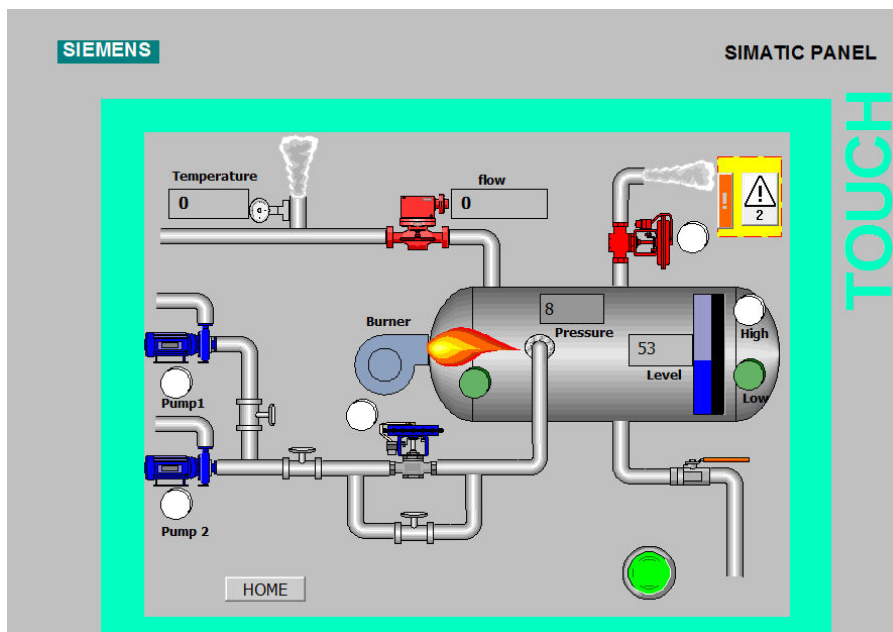


Fig IV.6 simulation of the output view of WinCC flexible.

- **Object view:**

The "Object View" shows all elements of the area selected from the "Project View".

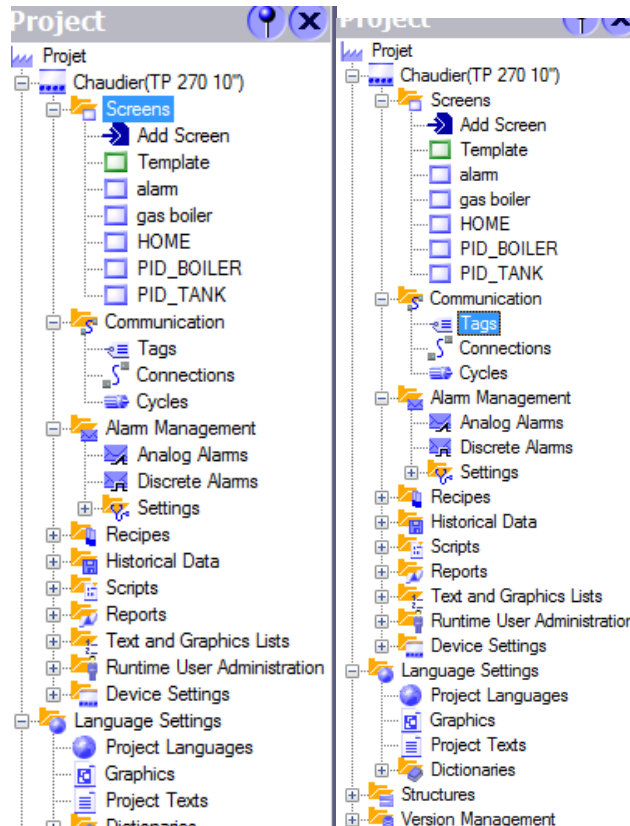


Fig IV.7 object view of WinCC flexible.

IV.3. Case study:

Using the SIMATIC WINCC flexible 2008 we created our supervision platform to master the process and always keep our machinery and units running which consists of five screens

The different screens of the Supervision:

- **Home screen:**

The main screen is the front page which is called HOME where we can navigate to the different screens of the supervision platform.

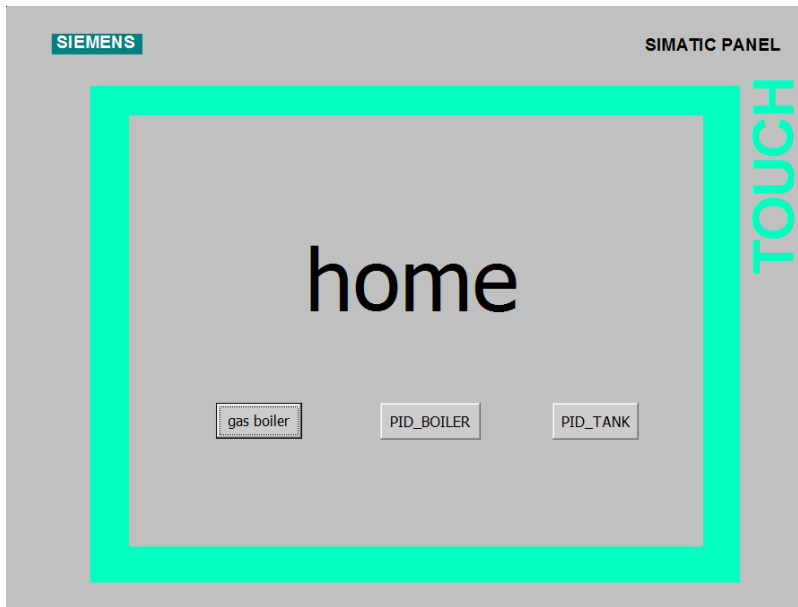


Fig IV.8 Home screen of HMI.

- **Gas boiler view:**

This view shows the whole process of our boiler where we can observe level, pressure and temperature indicators also different hydraulic and pneumatic actuators, where the operator can supervise the system and alarms indicated, further-more a button to return to the HOME page, the gas boiler view is shown below in **Fig IV.9**:

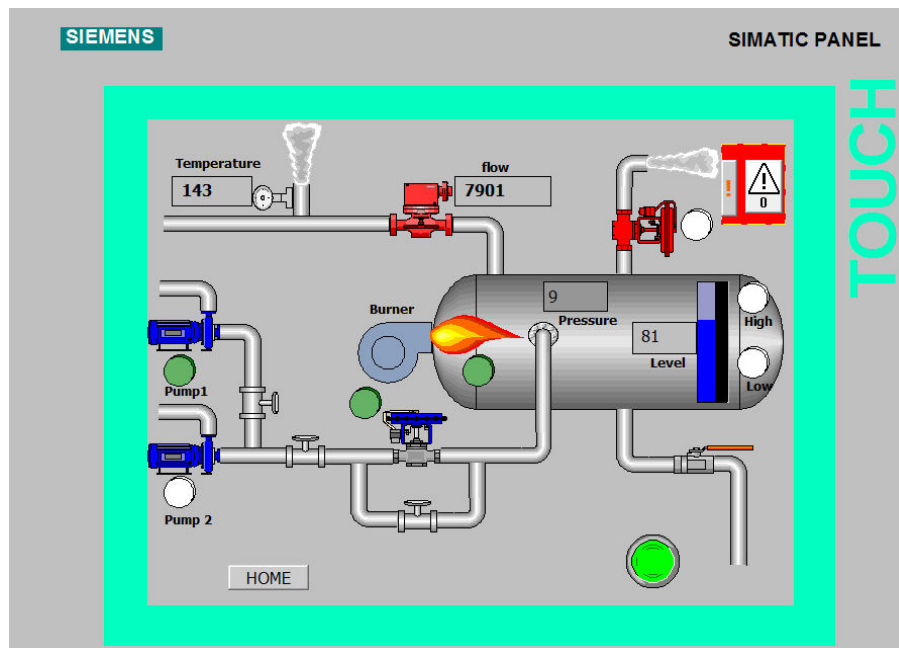


Fig IV.9 Gas boiler view in the HMI.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

- **PID tank view:**

This view shows the PID function of filling a tank with water which will make the system much more stable in a particular set-point, by using a regulator hydraulic valve we can control and regulate the water entering to the tank as showed in the following figure:

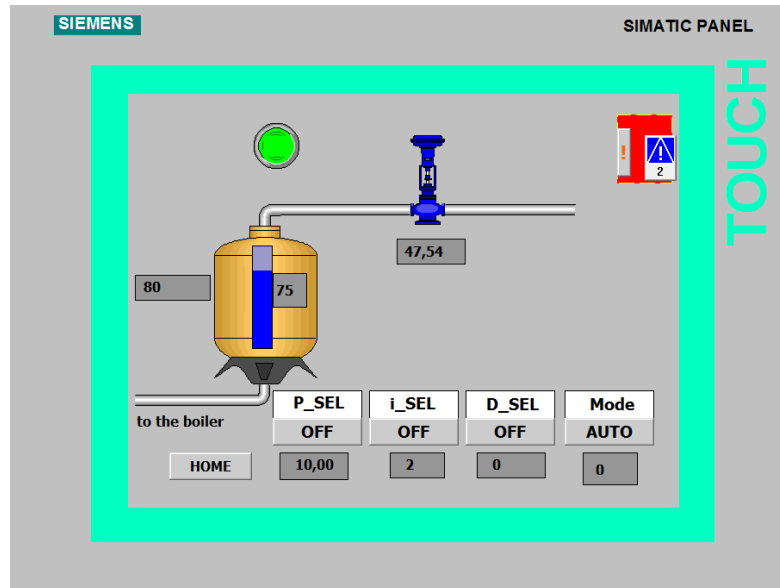


Fig IV.10 water tank view in the HMI.

- **PID boiler view:**

This view shows the visualization of PID function for control the water in our boiler as follows:

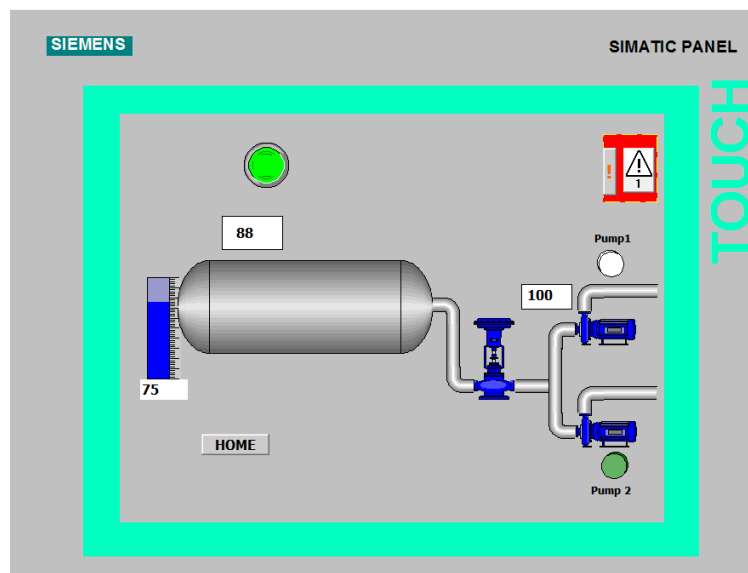


Fig IV.11 PID boiler view in the HMI.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

• Alarm view:

The HMI device triggers an alarm when a certain bit is set in the PLC. For this, we have configured digital and analog alarms in SIMATIC WINCC. It is possible to make it compulsory to acknowledge discrete and analog alarms signaling critical or dangerous states to ensure that: The person operating the installation is familiar with it and has to know the following means to acknowledge alarms:

- Acknowledgment in the alarm window.
- Acknowledgment in the alarms view.

The chosen alarm class is the "Error" class, the alarms of this class must be acquitted. Figure IV.12 and Figure IV.13 below shows the configuration of the alarm class and their animations which are as follows:

- When the triggering condition of an alarm is true, a triangle of signaling appears in the main view and the alarm panel is displayed.
- When the operator has acknowledged the alarm, the triangle disappears.

DISCRETE ALARMS						
Text	Number	Class	Trigger Tag	Trigger bit	Trigger address	
disjonction pomp1	1	Errors	Dj_p1_S	0	M 202.0	
disjonction pomp2	2	Errors	Dj_p2_S	0	M 203.0	
Arrêt d'urgence	3	Errors	AU_S	0	M 201.0	
alarme	4	Errors	Alarm	0	M 103.0	
alarm bruleur	9	Errors	Al_br	0	M 204.0	

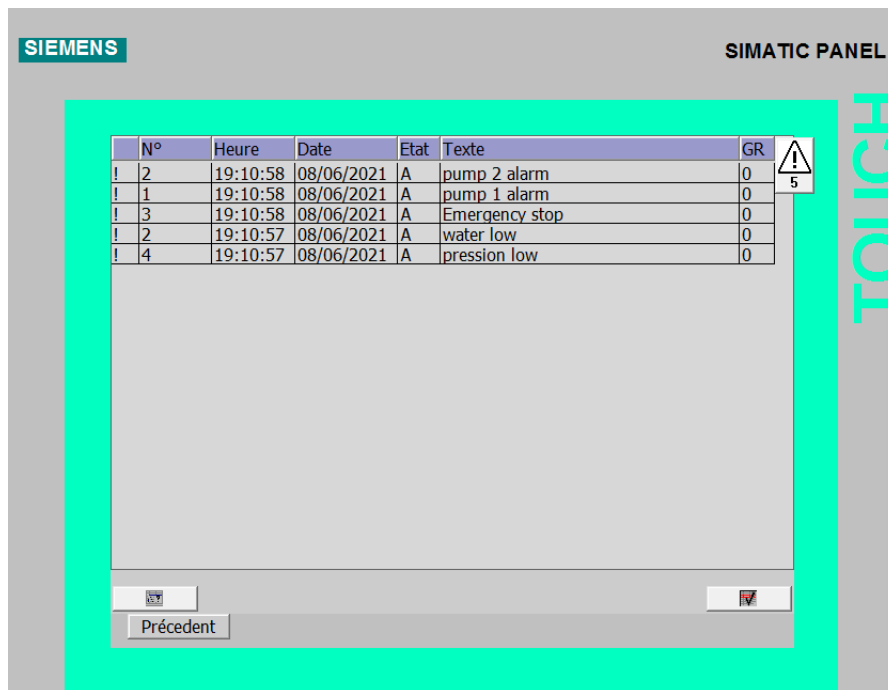
Fig IV.12 Discrete alarms.

ANALOG ALARMS						
Text	Number	Class	Trigger tag	Limit	Trigger mode	
water high	1	Errors	WATER LEVEL2	85	On rising edge	
water low	2	Errors	WATER LEVEL2	60	On falling edge	
pression high	3	Errors	PRESSION	12	On rising edge	
pression low	4	Errors	PRESSION	9	On falling edge	
alarm temperature	5	Errors	temp	160	On rising edge	

Fig IV.13 Analog alarms.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

Finally Figure IV.13 represents the alarm view.



The screenshot shows the SIMATIC PANEL interface with a table of active alarms. The table has columns for N°, Heure, Date, Etat, Texte, and GR. There are five rows of alarm data. A 'Précédent' button is visible at the bottom left of the table area. A vertical 'TOUCH' label is on the right side of the panel.

N°	Heure	Date	Etat	Texte	GR
! 2	19:10:58	08/06/2021	A	pump 2 alarm	0
! 1	19:10:58	08/06/2021	A	pump 1 alarm	0
! 3	19:10:58	08/06/2021	A	Emergency stop	0
! 2	19:10:57	08/06/2021	A	water low	0
! 4	19:10:57	08/06/2021	A	pression low	0

Fig IV.14 Alarm view.

IV.4. Implementation:

IV.4.1.Introduction:

In this part we will try to implement a process or a part from the steam boiler. After looking for what we can do with the instruments, controllers, sensors and actuators that we have, we come up on implementing the control of the water level inside the tank.

IV.4.2. List of equipment required:

- 12v DC power supply.
- PLC (SIMATIC S7-1200 CPU 1212C AC/DC/RLY).
- Digital multi-meter.
- Arduino Uno.
- Ultrasonic sensor HC-SR04.
- Circuit breaker.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

- Water pumps 12v.
- Switch.
- Jumpers.
- Ethernet cable.
- Variable resistors 1k Ω .
- Voltage regulator 10v (LM317).
- Capacitors: 1 μ F, 0.1 μ F.
- Motor drive IC L293.
- Water tube 0.6cm \varnothing .
- Graduated water tanks 3.
- Breadboards 2.

IV.4.3.Required software:

- TIA Portal v13 or higher.
- ARDUINO IDE.
- XRelais.

IV.4.4.Building the circuit:

Before building the circuit we are going to explain our choices for the required equipment: the SIMATIC S7-1200 CPU contains 2 analog inputs and 7 digital outputs and 5 relay outputs, but in our process in order to drive the water pump we need an analog output (which can be replaced with analog output module mentioned in chapter I) but we proposed a reliable solution which is using the relay outputs and drive it the circuit 1 to produce different voltages). The second part is how to measure the water level, to do that we proposed the ultrasonic sensor HC-SR04 connected to the Arduino. In our case we need an analog input to our PLC but the output of the Arduino is a 5voltes PWM signal so we need a circuit that interface between the output of the Arduino and the input of the PLC that is why we used Motor drive IC L293 in order to map the PWM signal to a 0 to10 volte analog signal. To summarize our circuit is divided into 3 sub-circuits:

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

- Voltage regulating circuit that contains the connection between the relays terminal of the PLC, the voltage regulator and the water pump.
- The interfacing circuit that contains Arduino, ultrasonic sensor and the motor drive IC L293.
- The PLC circuit that contains the different connection between the PLC and the circuit 1 and 2.

IV.4.4.1. The 1st sub-circuit:

The main component is the voltage regulator LM317 described in the figure 1:

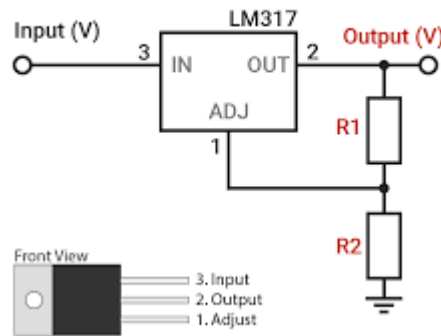


Fig IV.15 LM317 Pin out.

We will choose the equation 1 in order to select the values of the resistors:

$$V_{out} = 1.25 * \left(1 + \frac{R_1}{R_2}\right) + I_{adj} * R_2 \quad (\text{eq IV.1})$$

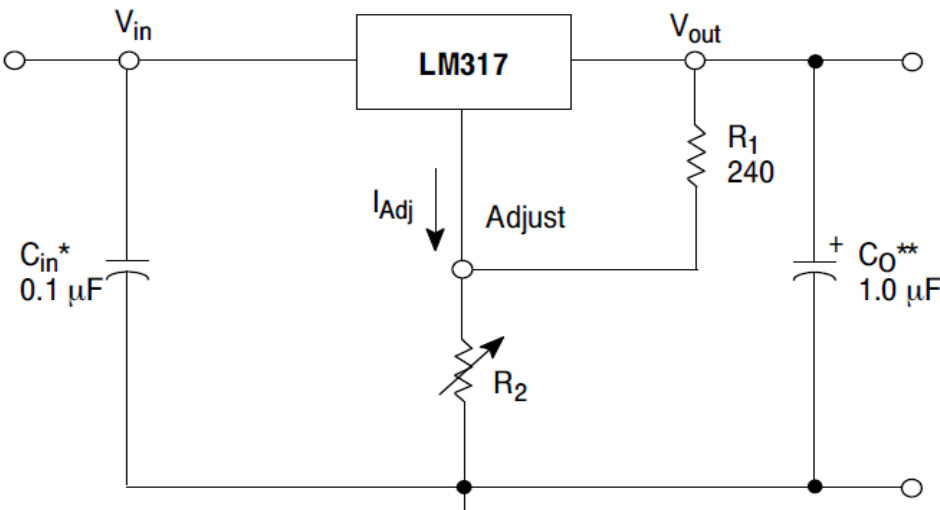


Fig IV.16 detailed circuit of IC LM317.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

Since I_{adj} is a neglected value we can approximate equation IV.1 to become as follow:

$$V_{out} = 1.25 * \left(1 + \frac{R1}{R2}\right) \quad (\text{eq IV.2})$$

In our case we will take $R1=240\Omega$.

We changed the voltages across the water pump in order to know different voltages that should be taken. We come up with different voltages 1.25v, 3.5v, 5v and 10v that should be relied with each relay of the PLC.

After numerical application we got the values of the resistors which are as follow:

$R3=432\Omega$.

$R4=288\Omega$.

$R5=960\Omega$.

We got the final form for our sub-circuit 1 presented in figure 3:

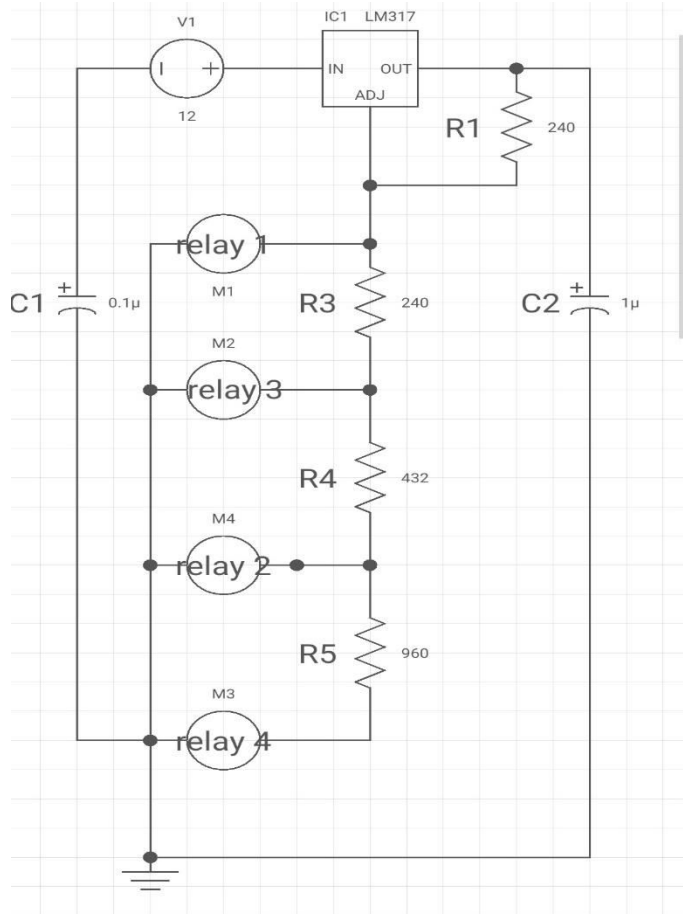


Fig IV.17 Final form of sub circuit 1.

IV.4.4.2. The 2nd sub-circuit:

In this circuit we will show how we connect the ultrasonic sensor and the motor drive IC L293D (that converts the PWM signal to analogue signal) to the Arduino.

The connection of the ultrasonic is shown in figure Fig IV.18.

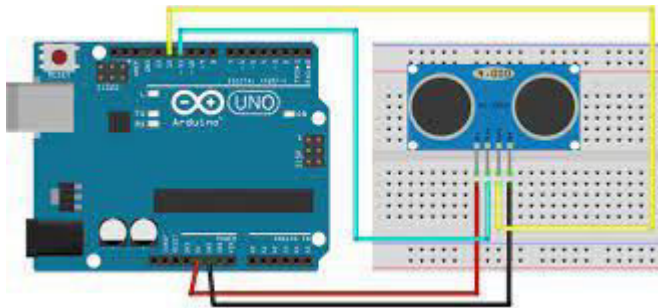


Fig IV.18 connection of the ultrasonic sensor with Arduino.

As shown, we connect Vcc and Gnd to the board of arduino and Trig pin to pin9 and echo pin to pin 10 in the PWM pins of the arduino

Now we have to convert the output called 'enA' (which is placed on pin 11) to an analog signal, all we have to do is to connect it to the IC L293D as shown in figure 6

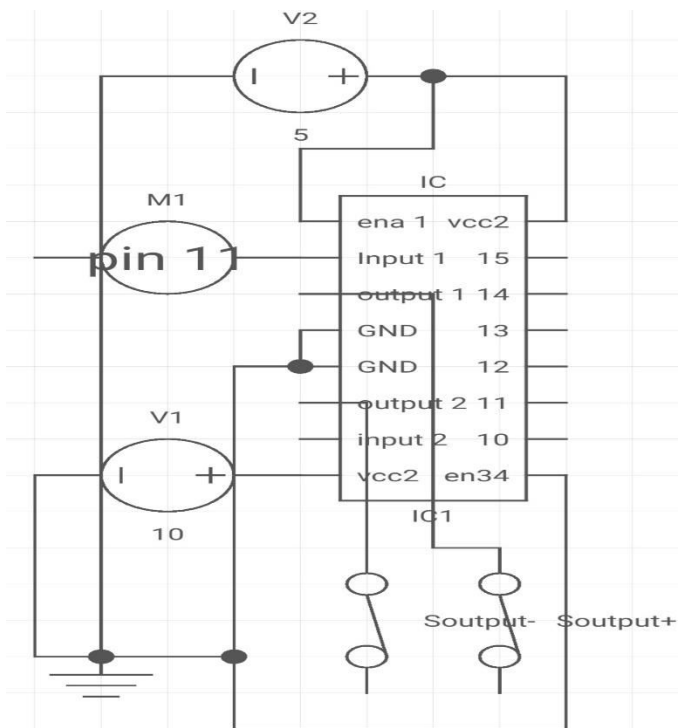


Fig IV.19 sub-circuit 2 connection of L293D with the output of Arduino.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

As shown in figure 6 to activate output 1 and 2 we have to set 5v to ena 1 and ground en34 and then we set the output of the Arduino which is a varying PWM signal and supply it as shown with 5 and 10 Voltes.

Finely we can take our converted signal to the PLC through the terminals output+ and output-.

IV.4.4.3. The 3rd sub circuit:

in this circuit we will show the connections to the PLC and the 2 previous sub circuit 1 and 2.

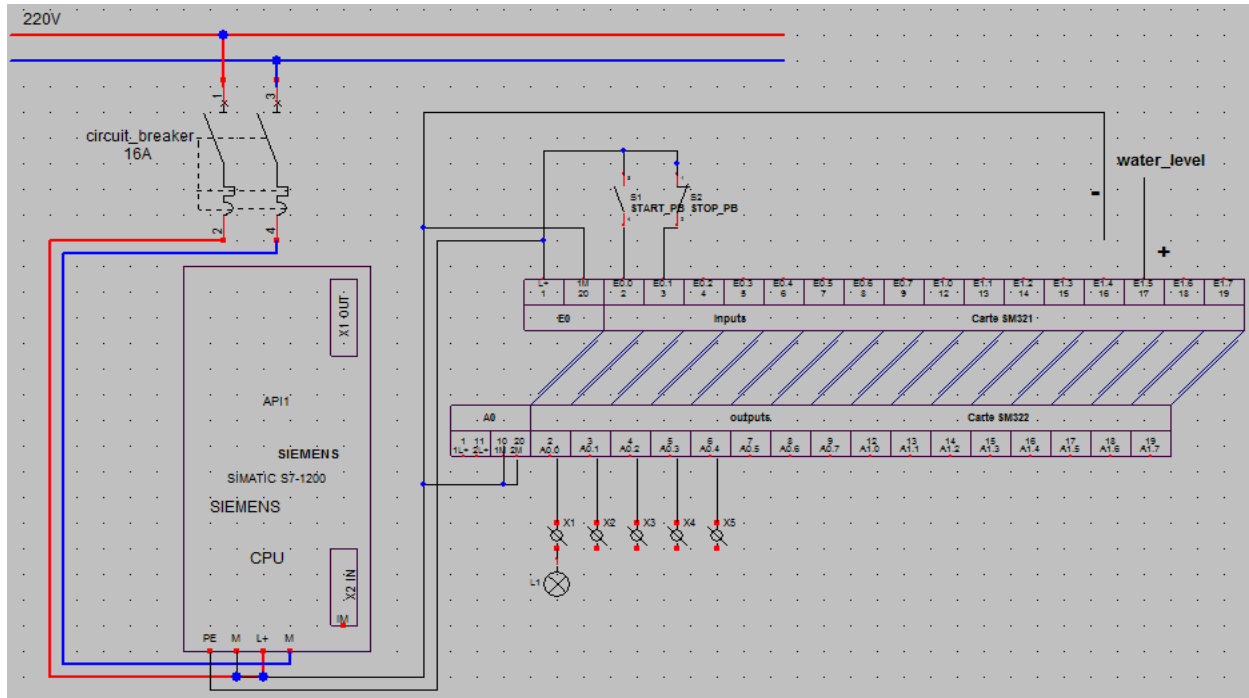


Fig IV.20 connection of the different input and output to the PLC.

In the figure 7 we showed how we connected the 220v AC supply to the PLC through the circuit breaker, push buttons ON and OFF, analog input (that comes from sub-circuit 2), output pins or relays that goes to sub-circuit 1 (to select different voltages).

IV.4.5. Programming the controllers:

In this implementation we are going to use 2 programs, one for our controller which is the PLC using TIA portal v13, and the second one is for the Arduino using Arduino IDE that captures the water level.

IV.4.5.1. Programming the PLC s7-1200 1212c AC/DC/RLY:

in this part we will show how the program uploaded to the PLC function and interconnect the

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

inputs and outputs .

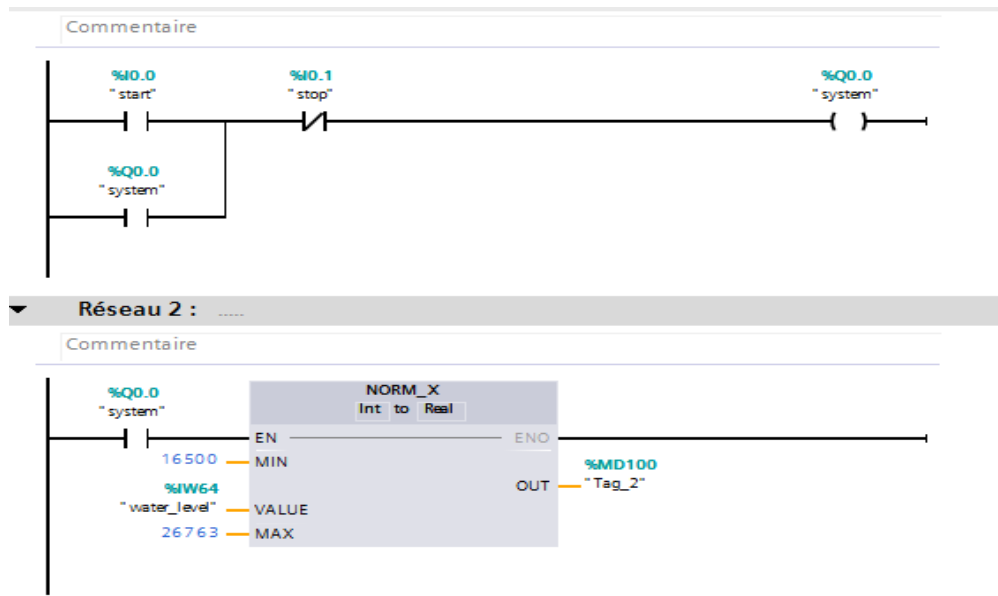


Fig IV.21a setting push buttons and converting analog signal to real.

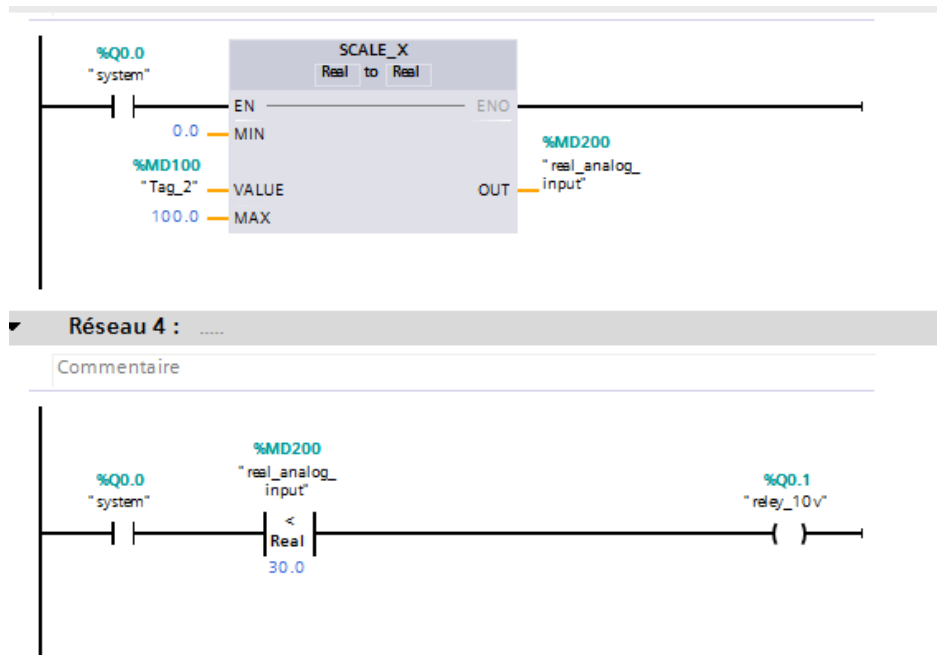


Fig IV.21b scaling analog signal and comparing it with predefined values.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

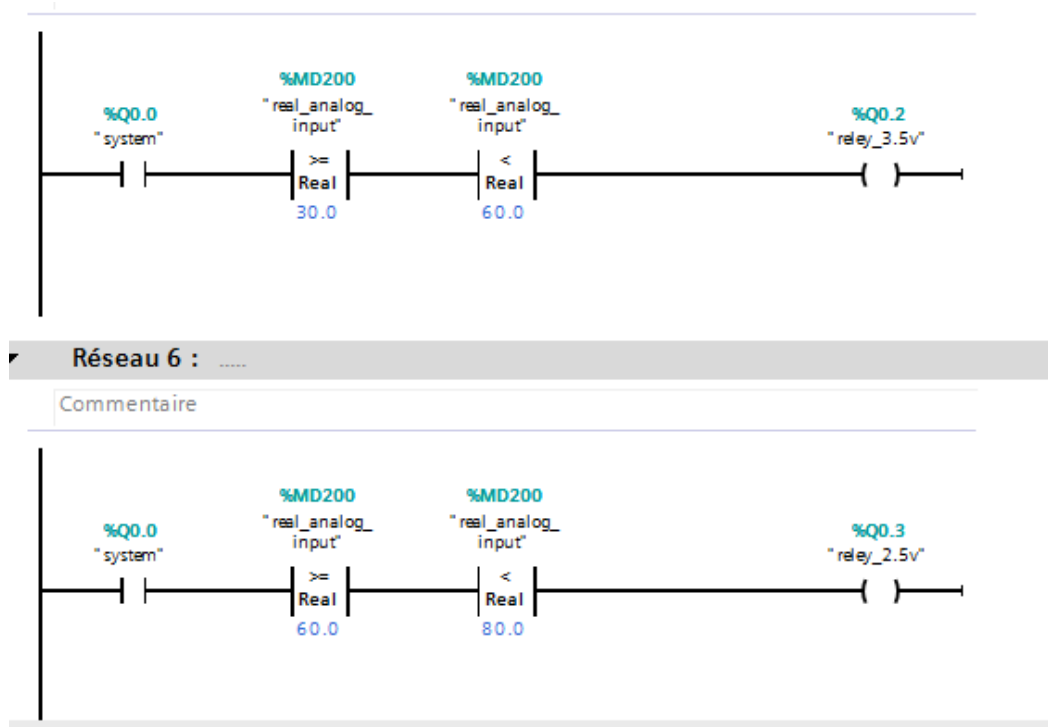


Fig IV.21c comparing the process variable with levels 30, 60, 80%.

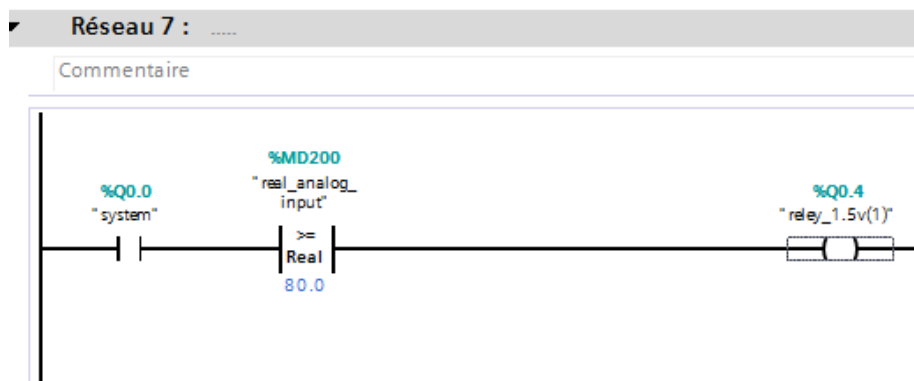


Fig IV.21d comparing the process variable with level 80% and setting the output.

As it is shown in the figures 7a, b, c and d, the programming of our PLC s7-1200, on the figure 7a we showed how to set push buttons and tag them with digital input variable %I0.0 and %I0.1 for start and stop push buttons consecutively, and in the second network we showed how did we convert our process variable from integer to Real that having the rang of 16500to 26763 which maps the range of 5.6vto 9.8v of the output of sub-circuit 2 shown in figure 5.

After converting the input signal, now we have to scale it to perform operation on it as it is shown in figure 7b we converted the rang of process variable to a range from 0 to 100%.

Chapter IV: Simulation and Supervision using WINCC & implementing water level control

After that, it is shown in figure 7c and 7d how we compared the process percentage and set values to our relays which select different stages of voltages as shown on their names.

IV.5. Conclusion:

In this chapter we have presented the procedures to be followed in order to create a Human Machine Interface to command and control gas boiler station .The creation of a HMI requires not only a good knowledge to the procedure and stages to create the supervision, but also of the language which is used in programming PLC in order to communicate correctly to the addresses of the variables.

Four screens have been created in order to organize the work:

- The main screen which is the front page.
- Boiler Main View to see the whole process.
- PID tank view.
- PID boiler view.
- Alarm view.

In the implementation part we can say that we could successfully implement our process using PLC S7-1200 C1212 CPU and Arduino UNO and the different component and software TIA portal and Arduino IDE.

GENERAL CONCLUSION

GENERAL CONCLUSION

In this modest work, we studied "The Automation of a steam boiler "the one that belongs to SOPI company, which allowed us to identify a solution for different problems, especially in a complex project such as the creation of an industrial unit, thus enriching our knowledge of industrial automation and to get an experience acquired in this field.

The aim of this work was to install the SIMATIC manager software and upload a program in programmable industrial controllers PLCs to control our steam boiler, as well as the HMI screen on the computer controlled by the S7-300, which would help the operator to monitor operating conditions and respond to failures. Perform simulations to see the terminal control response and potential alarms before starting, taking into account the faults that may damage our system.

The programming software Step7 allowed us to export inputs / outputs variables directly to the table of symbols to use it as external tags in the SIMATIC WinCC Runtime advanced program.

Interfacing with a machine is not only mastering the physical interface between the human user and the system but it is also mastering the mental model implemented in the machine architecture and logic. If the machine design has been well thought out and user-centered this should mirror the user's mental model. Interfacing or supervising a machine, or any automated system is a matter of human performance and as such, it should always include a reasonable measure of caution in order to avoid complacency and overreliance on the boiler

In this project we have achieved how to identify a process and simulate it in MATLAB and control it using a PID that we could have tuned its parameters using zeigler-nichol method.

The professional background has allowed us to better understand the project and to familiarize ourselves with the responsibilities of field engineers in this area.

We hope that this humble work will be the starting point of our professional life and that it will be useful for future promotions, God willing.

APPENDIX

I. Least square approach:

The least square approach is a popular tool for estimating parameters from experimental data. While there are other ways for estimating parameters, such as the likelihood technique, the least square technique remains the most popular since it is simple to apply and impartial (the error is very small when the noise is present).

Identification of the transfer function G_s :

The transfer function $G(s)$ has a form of

$$G(s) = \frac{\varepsilon}{(1+sT_1)(1+sT_2)} \quad (\text{eq V.1})$$

Where ε , T_1 and T_2 are unknowns to be determined.

So to determine those unknown values, we will follow the procedure given below:

$G(s)$ Can be written as:

$$G(s) = \frac{\varepsilon}{(1+sT_1)(1+sT_2)} = \frac{A_1}{1+sT_1} + \frac{A_2}{1+sT_2}. \quad (\text{eq V.2})$$

We can find A_1 and A_2 using partial fraction expansion to be:

$$\bullet \quad A_1 = \frac{\varepsilon T_1}{T_1 - T_2} \quad (\text{eq V.3})$$

$$\bullet \quad A_2 = -\frac{\varepsilon T_2}{T_1 - T_2}. \quad (\text{eq V.4})$$

Then $G(s)$ can be written as

$$G(s) = \frac{\varepsilon}{T_1 - T_2} \left(\frac{1}{s + \frac{1}{T_1}} - \frac{1}{s + \frac{1}{T_2}} \right). \quad (\text{eq V.5})$$

In order to find the unknowns we have to go from s domain to z domain we get

$$G(z) = \frac{\varepsilon}{T_1 - T_2} \left(\frac{1}{1 - e^{-\frac{1}{T_1}} z^{-1}} - \frac{1}{1 - e^{-\frac{1}{T_2}} z^{-1}} \right) \quad (\text{eq V.6})$$

After rearranging equation V.6 we get

$$G(z) = \frac{\varepsilon}{\frac{1}{a} - \frac{1}{b}} \left(\frac{z^{-1}(e^{-a} - e^{-b})}{(1 - e^{-a} z^{-1})(1 - e^{-b} z^{-1})} \right) \quad (\text{eq V.7})$$

Where $a = \frac{1}{T_1}$ and $b = \frac{1}{T_2}$.

After rearranging equation V.6 we get:

APPENDIX

$$G(z) = \frac{\varepsilon(A-B)}{\frac{1}{\ln B} - \frac{1}{\ln A}} \left(\frac{z^{-1}}{(1-Az^{-1})(1-Bz^{-1})} \right) \quad (\text{eq V.8})$$

Where $A=e^{-\frac{1}{T_1}}$ and $B=e^{-\frac{1}{T_2}}$

Finally, we get the following equation:

$$G(z) = \frac{\varepsilon(A-B)}{\frac{1}{\ln B} - \frac{1}{\ln A}} \left(\frac{z^{-1}}{ABz^{-2} - (A+B)z^{-1} + 1} \right). \quad (\text{eq V.9})$$

By letting $Y0 = \frac{\varepsilon(A-B)}{\frac{1}{\ln B} - \frac{1}{\ln A}}$, $Y1 = (A + B)$ and $Y2 = AB$.

Now we can write $G(z)$ as

$$G(z) = \frac{Y0.z^{-1}}{1-z^{-1}.Y1+z^{-2}.Y2} = \frac{L(z)}{T(z)} \quad (\text{eq v.10})$$

Where T represent the time and L represent the water level.

From the equation V.10 we can get

$$Y0.T(K-1) = L(K) + Y1.L(K-1) + Y2.L(K-2) \quad (\text{eq v.11})$$

by rearranging equation V.11 we get

$$L(K) = Y0.T(K-1) - Y1.L(K-1) - Y2.L(K-2) \quad (\text{eq v.12})$$

so we can write:

$$L(K) = \begin{bmatrix} T(K-1) & L(k-1) & L(k-2) \end{bmatrix} * \begin{matrix} Y0 \\ Y1 \\ Y2 \end{matrix} \quad (\text{eq v.13})$$

Or $L(K)=D(K)*\text{delta}(k)$ where:

$$- D(K) = \begin{bmatrix} T(K-1) & L(k-1) & L(k-2) \end{bmatrix} \quad (\text{eq v.14})$$

$$- \text{delta}(k) = \begin{pmatrix} Y0 \\ Y1 \\ Y2 \end{pmatrix} \quad (\text{eq v.15})$$

using the least square method we get

$$\text{delta}(k) = (D^T * D)^{-1} * D^T * L \quad (\text{eq v.16})$$

And from the data in the table 2.1 we get the following matrix

APPENDIX

d =

	0	0	0
	0	0	0
1.0000	-0.0050	0	
2.0000	-0.0120	-0.0050	
3.0000	-0.0220	-0.0120	
4.0000	-0.0340	-0.0220	
5.0000	-0.0500	-0.0340	
6.0000	-0.0700	-0.0500	
7.0000	-0.0950	-0.0700	
8.0000	-0.1250	-0.0950	
9.0000	-0.1600	-0.1250	
10.0000	-0.1950	-0.1600	
11.0000	-0.2300	-0.1950	
12.0000	-0.2650	-0.2300	
13.0000	-0.2950	-0.2650	
14.0000	-0.3300	-0.2950	
15.0000	-0.3600	-0.3300	
16.0000	-0.3950	-0.3600	
17.0000	-0.4250	-0.3950	
18.0000	-0.4600	-0.4250	
19.0000	-0.4950	-0.4600	

Table 2.1: Data of filling a tank with water.

From equation (9) we can get the unknown vector

$$\Delta(k) = \begin{pmatrix} Y0 \\ Y1 \\ Y2 \end{pmatrix} = \begin{pmatrix} 0.0049 \\ 1.22 \\ 0.3696 \end{pmatrix}.$$

So, the values of A and B can be found as

$$Y1 = (A + B)$$

$$Y2 = AB$$

After solving equations (10) and (11) we get the values of A and B is found to be:

A=0.56, B=0.66.

$$\text{We have } Y0 = \frac{\varepsilon(A-B)}{\frac{1}{\ln B} - \frac{1}{\ln A}} \text{ so } \varepsilon = 0.036.$$

From equation (2) we get T1=1.74 and T2=2.422

So the unknown values T1, T2 and ε are approximately

$\varepsilon=0.036$.

T1=1.74, T2=2.422.

$$\text{Finally the identified transfer function } G(s) = \frac{0.036}{(1+1.74s)(1+2.422s)}$$

APPENDIX

II. Mnemonic Table:

Statu	Symbol /	Address	Data type	Comment
	add1	MD 1300	REAL	
	AL_BURNER	MW 204	WORD	
	Al_br	MW 203	WORD	alarm burner
	Al_temp	MW 305	WORD	alarm temperature
	Alarm	MW 102	WORD	alarmr
	AU	M 201.0	BOOL	emergency pushbuton
	AU_S	MW 200	WORD	
	AUTO_MODE	I 1.7	BOOL	auto-mode switch
	BI_POLAR2	M 120.6	BOOL	
	BT_stop	I 0.1	BOOL	stop pushbotton
	BURNER	Q 0.2	BOOL	burner
	but_valve	I 0.6	BOOL	butterfly valve
	c_hmi_water	MW 900	INT	
	C_L_Water_ch	MD 364	REAL	set point for HMI
	C_LEVEL	MD 290	REAL	
	clevel	IW 4	WORD	water level analog input
	cnt	M 100.3	BOOL	
	consigne	MD 201	REAL	setpoint
	CONT_C	FB 41	FB 41	Continuous Control
	Cycle Execution	OB 1	OB 1	
	D_SEL	M 200.4	BOOL	derivative select
	Dj_p1_S	MW 201	WORD	
	Dj_p2_S	MW 202	WORD	
	Em.stop	I 0.4	BOOL	emergency stop
	ERROR	MD 300	REAL	error
	ff	MD 1000	DINT	
	fire_detector	I 0.5	BOOL	fire detector
	FLOW_RATE	MD 1600	REAL	flow rate
	gain	MD 220	REAL	gain
	hight	Q 0.7	BOOL	hight alarm
	I_SEL	M 200.3	BOOL	integral select
	L_Water_ch	MD 360	REAL	level water real values
	L_water_hmi	MW 700	INT	water level HMI
	LAMP_PUMP1	Q 1.3	BOOL	lamp pump1
	LAMP_PUMP2	Q 1.4	BOOL	lamp pump2

6		sub1	MD 1100	REAL	
6		sub2	MD 1200	REAL	
7		suc_valve	I 0.7	BOOL	security valve
7		System	Q 0.0	BOOL	system lamp
7		td	MD 240	TIME	derivative time
7		TI	MD 230	TIME	integral time
7		V_L_water	MD 368	REAL	
7		VALVE	Q 0.1	BOOL	valve
7		valve-sec2	Q 1.1	BOOL	security valve 2
7		valve_comand	MD 250	REAL	
7		VALVE_MAN	Q 1.7	BOOL	manual valve
7		VALVE_PID	MW 100	WORD	PID valve
8		vann-sec	Q 0.6	BOOL	security valve
8		vanne-air	Q 0.5	BOOL	air valve
8		WATER LEVEL	MW 257	WORD	water level for manual mode
8		WATER LEVEL2	MD 120	REAL	
8		water_high	MW 304	WORD	high level indicator
8		WATER_LEVEL	MW 520	INT	water level
8		water_low	MW 303	WORD	low indicator

APPENDIX

	Statu	Symbol /	Address	Data type	Comment
3		LAMP_PUMP1	Q 1.3	BOOL	lamp pump1
3		LAMP_PUMP2	Q 1.4	BOOL	lamp pump 2
3		LAMP_RTH1	Q 1.5	BOOL	thermal relay lamp1
3		LAMP_RTH2	Q 1.6	BOOL	thermal relay lamp2
4		LMN_D	MD 280	REAL	LMN D
4		LMN_I	MD 270	REAL	LMN I
4		LMN_P	MD 260	REAL	LMN P
4		low	Q 1.0	BOOL	low alarm
4		man	MD 210	REAL	manual PID
4		MAN_MODE	I 120.1	BOOL	manual mode
4		MAN_ON	M 200.0	BOOL	man_on PID
4		MED	MW 500	INT	
4		MW0	MW 0	WORD	
4		P_SEL	M 200.2	BOOL	proportional select
5		p1	M 360.0	BOOL	
5		p2	M 360.1	BOOL	
5		p3	M 360.2	BOOL	
5		p4	M 360.3	BOOL	
5		POMP1	Q 0.3	BOOL	
5		POMP2	Q 0.4	BOOL	
5		pres_low	MW 301	WORD	low pressure
5		press_high	MW 302	WORD	high pressure
5		PRESSURE	MD 30	REAL	pressure scaling
5		PV	M 200.1	BOOL	
6		Rth.p1	I 0.2	BOOL	thermal relay 1
6		Rth.p2	I 0.3	BOOL	thermal relay 2
6		SCALE	FC 105	FC 105	Scaling Values
6		START_PUMP1	I 120.2	BOOL	pump1 start push button
6		START_PUMP2	I 120.4	BOOL	pump2 start push button
6		START_PUSH	I 0.0	BOOL	system start push button
6		STOP_PUMP1	I 120.3	BOOL	pump 1 stop push button
6		STOP_PUMP2	I 120.5	BOOL	pump 2 stop push button
6		sub1	MD 1100	REAL	
6		sub2	MD 1200	REAL	

III. Data blocks:

The following data block DB41 is used for PID control block

	Address	Declaration	Name	Type	Initial value	Actual value	Comment
1	0.0	in	COM...	BOOL	FALSE	FALSE	complete restart
2	0.1	in	MAN...	BOOL	TRUE	TRUE	manual value on
3	0.2	in	PVP...	BOOL	FALSE	FALSE	process variable periphery on
4	0.3	in	P_SEL	BOOL	TRUE	TRUE	proportional action on
5	0.4	in	I_SEL	BOOL	TRUE	TRUE	integral action on
6	0.5	in	INT...	BOOL	FALSE	FALSE	integral action hold
7	0.6	in	I_IT...	BOOL	FALSE	FALSE	initialization of the integral action
8	0.7	in	D_SEL	BOOL	FALSE	FALSE	derivative action on
9	2.0	in	CYCLE	TIME	T#1S	T#1S	sample time
10	6.0	in	SP_I...	REAL	0.00000...	0.00000...	internal setpoint
11	10.0	in	PV_IN	REAL	0.00000...	0.00000...	process variable in
12	14.0	in	PV...	WORD	W#16#0	W#16#0	process variable periphery
13	16.0	in	MAN	REAL	0.00000...	0.00000...	manual value
14	20.0	in	GAIN	REAL	2.00000...	2.00000...	proportional gain
15	24.0	in	TI	TIME	T#20S	T#20S	reset time
16	28.0	in	TD	TIME	T#10S	T#10S	derivative time
17	32.0	in	TM...	TIME	T#2S	T#2S	time lag of the derivative action
18	36.0	in	DEA...	REAL	0.00000...	0.00000...	dead band width
19	40.0	in	LMN...	REAL	1.00000...	1.00000...	manipulated value high limit
20	44.0	in	LMN...	REAL	0.00000...	0.00000...	manipulated value low limit
21	48.0	in	PV...	REAL	1.00000...	1.00000...	process variable factor
22	52.0	in	PV...	REAL	0.00000...	0.00000...	process variable offset
23	56.0	in	LMN...	REAL	1.00000...	1.00000...	manipulated value factor
24	60.0	in	LMN...	REAL	0.00000...	0.00000...	manipulated value offset
25	64.0	in	I_IT...	REAL	0.00000...	0.00000...	initialization value of the integral action
26	68.0	in	DISV	REAL	0.00000...	0.00000...	disturbance variable
27	72.0	out	LMN	REAL	0.00000...	0.00000...	manipulated value
28	76.0	out	LMN...	WORD	W#16#0	W#16#0	manipulated value periphery
29	78.0	out	QLM...	BOOL	FALSE	FALSE	high limit of manipulated value reached
30	78.1	out	QLM...	BOOL	FALSE	FALSE	low limit of manipulated value reached
31	80.0	out	LMN_P	REAL	0.00000...	0.00000...	proportionality component
32	84.0	out	LMN_I	REAL	0.00000...	0.00000...	integral component
33	88.0	out	LMN_D	REAL	0.00000...	0.00000...	derivative component
34	92.0	out	PV	REAL	0.00000...	0.00000...	process variable
35	96.0	out	ER	REAL	0.00000...	0.00000...	error signal
36	100.0	stat	sInvAlt	REAL	0.00000...	0.00000...	
37	104.0	stat	sInv	REAL	0.00000...	0.00000...	

APPENDIX

35	96.0	out	ER	REAL	0.00000...	0.00000...	error signal
36	100.0	stat	sInvAlt	REAL	0.00000...	0.00000...	
37	104.0	stat	sIant...	REAL	0.00000...	0.00000...	
38	108.0	stat	sRestInt	REAL	0.00000...	0.00000...	
39	112.0	stat	sRes...	REAL	0.00000...	0.00000...	
40	116.0	stat	sRueck	REAL	0.00000...	0.00000...	
41	120.0	stat	sLmn	REAL	0.00000...	0.00000...	
42	124.0	stat	sbAr...	BOOL	FALSE	FALSE	
43	124.1	stat	sbAr...	BOOL	FALSE	FALSE	
44	124.2	stat	sbLi...	BOOL	TRUE	TRUE	

IV. Organizational Blocks:

OB1: which is the main block where we can have our main program including functions

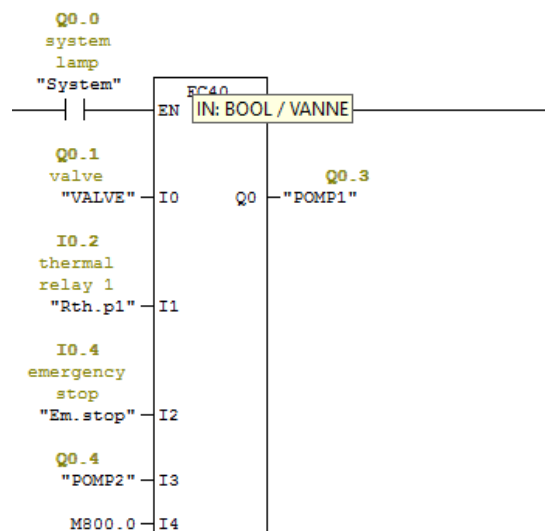
OB1 : "Main Program Sweep (Cycle)"

Comment:

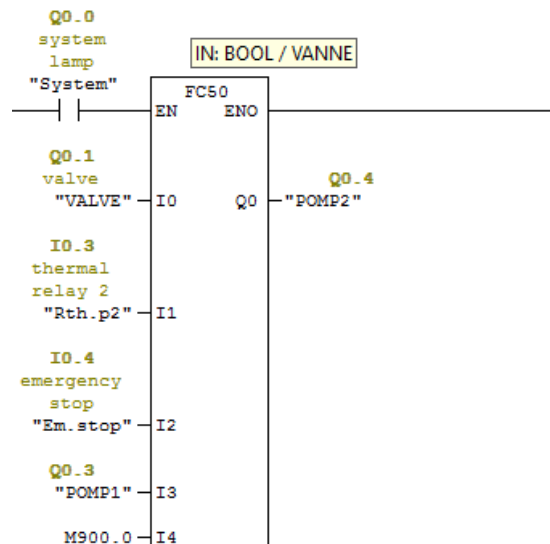
Network 1 : system



Network 3 : pump1

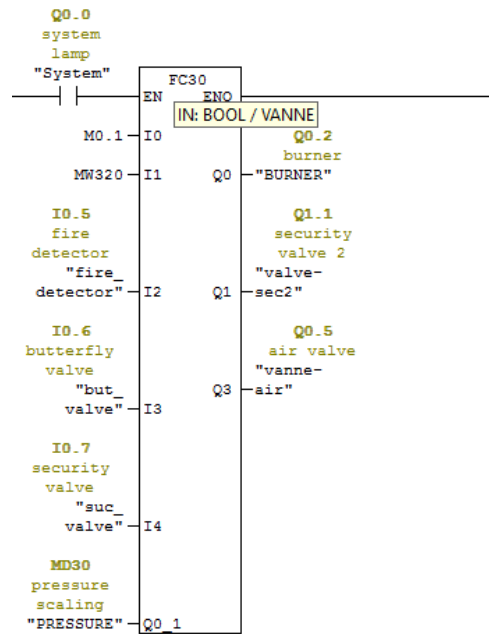


Network 4 : pump2

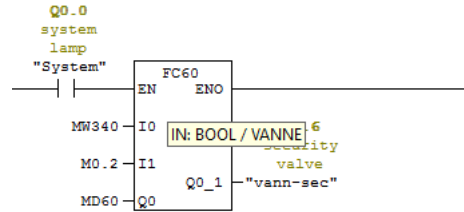


APPENDIX

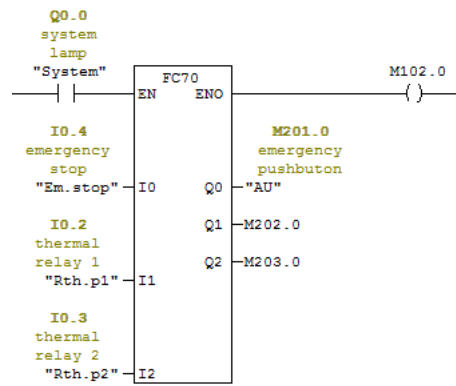
Network 5 : burner system



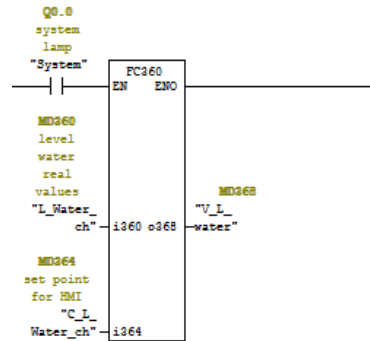
Network 6 : temperature protection



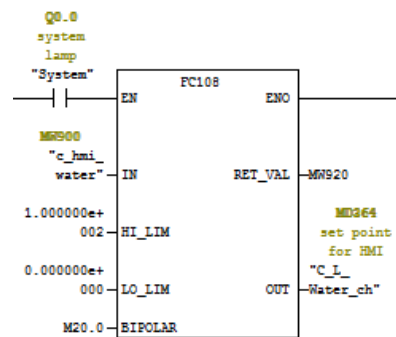
Network 7 : Security



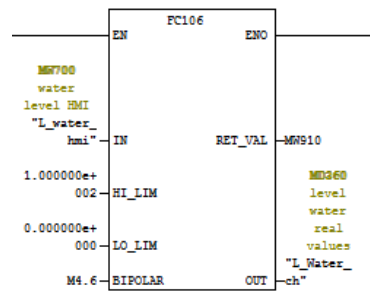
Network 9 : boiler PID



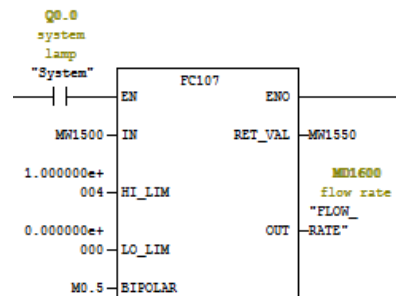
Network 11 : Title:



Network 10 : Title:

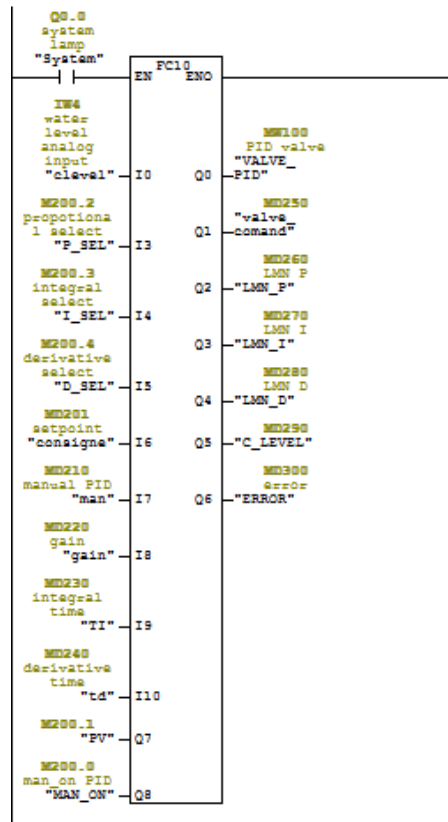


Network 12 : Title:

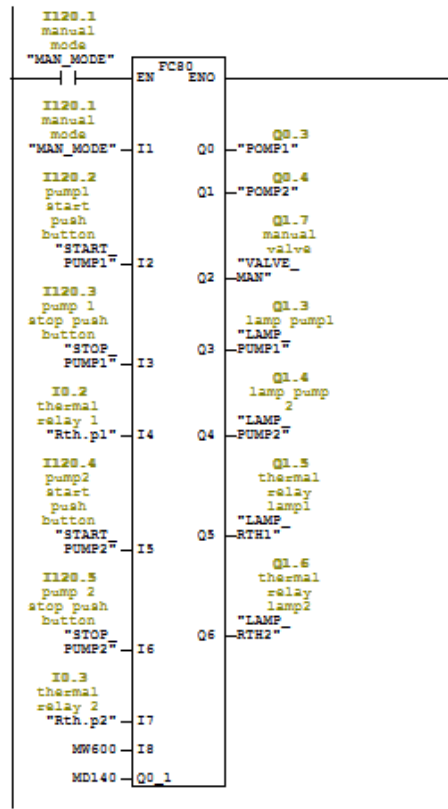


APPENDIX

Network 8 : PID LAMP system



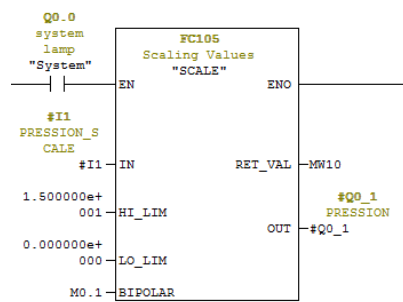
Network 13 : manual mode pumping system



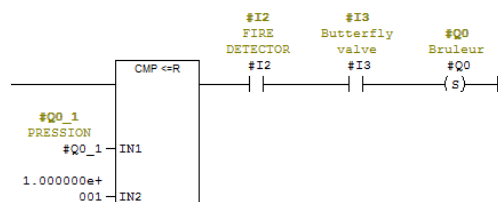
V.Functions:

FC30: burner system

Network 1 : Title:



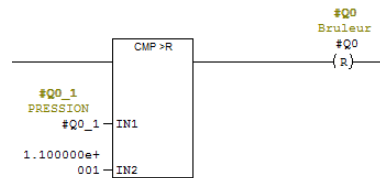
Network 2 : Title:



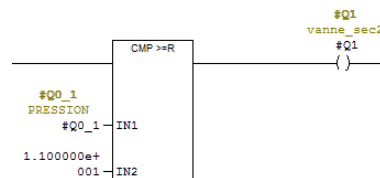
Network 3 : Title:



Network 4 : Title:



Network 5 : Title:



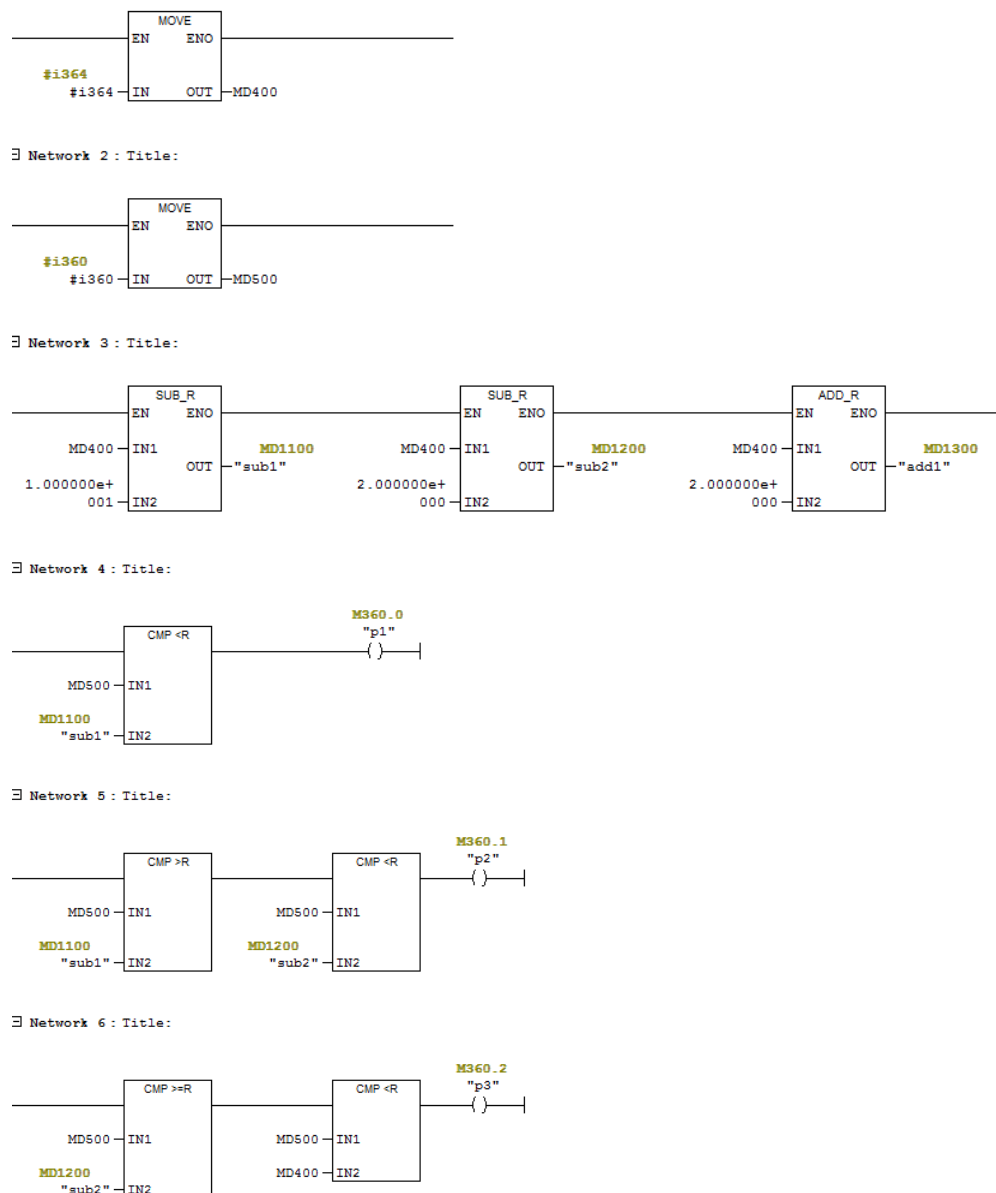
APPENDIX

FC 360:

We created this function for controlling the hydraulic regulator valve that will feed the boiler with water where we have chosen four intervals, each interval of water level represent an opening percentage of the valve as follows:

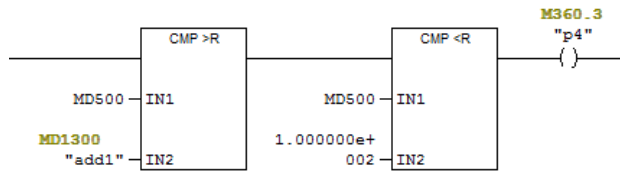
- For water level between 0 and 70% the valve will be fully opened 100%=p1.
- For water level between 70% and 78% the valve will be opened 20% =p2.
- For water level between 78% and 80% the valve will be opened 5%=p3.
- For water level between 80% and 100% the valve will be fully closed 0%=p4.

The program is as follows

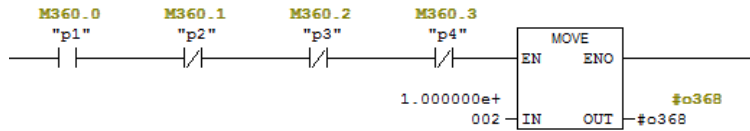


APPENDIX

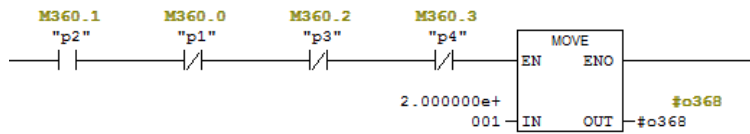
Network 7 : Title:



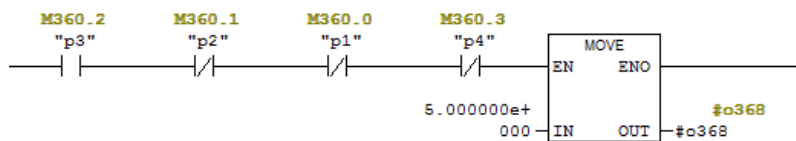
Network 8 : Title:



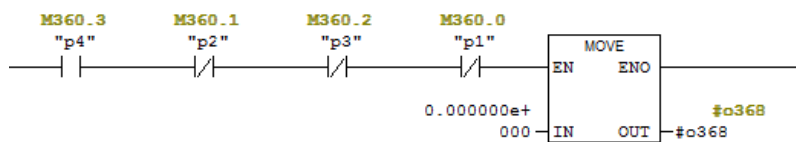
Network 9 : Title:



Network 10 : Title:



Network 11 : Title:



BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] Siemens, A. G. (2019). S7-300 Programable controller System Manual.
- [2] Kaftan, J. (2011). PLC Basic Course with SIMATIC S7. Vogel Business Media GmbH & Company KG.
- [3] MICHEL. G API Architecture Et Application Des Automates Programmables Industriels. DUNOD, Paris (1987)
- [4] Ren, J., Zuo, Q., & Li, H. (2010, March). Structure Control Language and its Application in Control Algorithm Programming. In 2010 International Conference on Measuring Technology and Mechatronics Automation IEEE.
- [5] «SIMATIC WinCC flexible», Brochure, Mars 2010.
- [6] SIMATIC S7-300 manual, SIEMENS AG,2007.
- [7] MICHEL. G, Les API, Architecture Et Application Des Automates Programmables. Industriels. DUNOD, Paris [1987].
- [8] SIEMENS PLC training document, SIEMENS AG ,2010.
- [9] Measurement system and control basics 4th edition by Thomas A. Hughes 17 august 2006.
- [10] Croser, P., & Ebel, F. (2002). Pneumatics: basic level. Festo Didactic.
- [11] Tullis, J. P. (1989). Hydraulics of pipelines: Pumps, valves, cavitation, transients. John Wiley & Sons.
- [12] Basu, P., Kefa, C., & Jestin, L. (2012). Boilers and burners: design and theory. Springer Science & Business Media.
- [13] SPIRAX SARCO, learn about steam, the boiler house, shell-boilers ,2021.
- [14] Webster, J. G., & Eren, H. (Eds.). (2018). Measurement, Instrumentation, and Sensors Handbook: Two-Volume Set. CRC press.
- [15] Fundamental of PID Control written by Anthony K. Ho, PE (2014).
- [16] Copeland, B. R. (2008). The design of PID controllers using Ziegler Nichols tuning
- [17] Industrial Automation: Learn the current and leading-edge research on SCADA. By VIKALP JOSHI, MANOJ SINGH ADHIKARI, RAJU PATEL, RAJESH SINGH, ANITA GEHLOT
- [18] electrical engineering, control systems modelling of systems, research paper may 2019 by William Bolton