Dissertation on

"Simulating an Industrial Internet of Things using SCADA and analysing potential vulnerabilities"

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Anirudh Murthy	**01FB16ECS059**
N Kantharvan	**01FB16ECS216**
Rohan B Sahu	**01FB16ECS305**

Under the guidance of

Internal Guide
S. Nagasundari
Associate Professor,
PES University

January – May 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

'Simulating an Industrial Internet of Things using SCADA and analysing potential vulnerabilities'

is a bonafide work carried out by

Anirudh Murthy	**01FB16ECS059**
N Kantharvan	**01FB16ECS216**
Rohan B Sahu	**01FB16ECS305**

In partial fulfilment for the completion of eighth semester project work in the Program of Study Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2020 – May. 2020. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature S. Nagasundari Associate Professor	Signature Dr. Shylaja S S Chairperson	Signature Dr. B K Keshavan Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. _____	**1.** _____
2. _____	**2.** _____

DECLARATION

We hereby declare that the project entitled **"Simulating an Industrial Internet of Things using SCADA and analysing potential vulnerabilities"** has been carried out by us under the guidance of S. Nagasundari, Associate Professor and submitted in partial fulfillment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2020. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

01FB16ECS059	**Anirudh Murthy**	
01FB16ECS216	**N Kantharvan**	
01FB16ECS305	**Rohan B Sahu**	

ACKNOWLEDGEMENT

We take this opportunity to thank our Guide **S Nagasundari, Associate Professor, PES University** and Co-Guide **Prof. Prasad Honnavalli, Professor, PES University**, for guidance and encouragement throughout this project.

We also thank our Project coordinators **Prof. Preet Kanwal, Professor, PES University** and **Prof. Sangeeta V, Professor, PES University** for their guidance.

We take this opportunity to thank **Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University**, for all the support we have received from the department.

We would like to thank **Dr. B.K. Keshavan, Dean of Faculty, PES University** for his help.

We are very thankful to **Dr. Suryaprasad J, Vice-Chancellor – PES University**, **Prof. Jawahar Doreswamy, Pro Chancellor – PES University**, **Dr. M. R. Doreswamy, Chancellor – PES University** for providing us various opportunities every way possible.

Concluding, We also like to thank our family and friends for continuous support and motivation.

ABSTRACT

Internet of Things (IoT) is being used all over the world to collect data and perform actions where it requires huge man power or nearly impossible for a human to spend his time, also in difficult situations relating to temperature and pressure for a human to function. And such data transfers need an Intelligent System to handle and process, In case of Industries SCADA systems do the role of tracking, collecting and managing data from these IoTs. Here is where security comes into play valuing the data being stored and processed. Which affects a significant part of a company's revenue. Our approach to the problem is to build a working prototype of an IOT system connected to a SCADA system and understand the various challenges of building such a model to get a holistic approach on the topic, followed by fixing any security vulnerabilities if possible.

TABLE OF CONTENTS

LIST OF FIGURES

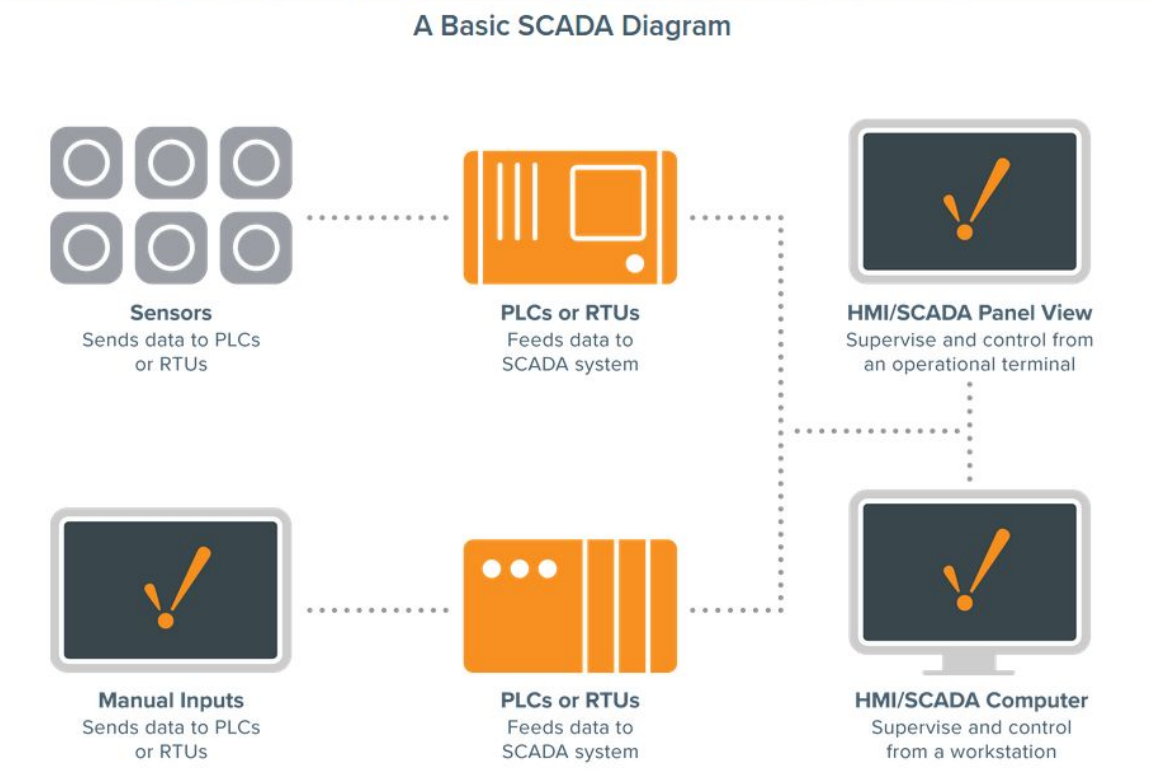
CHAPTER-1

INTRODUCTION

In modern times, the Internet of Things (IoT) has been dominating the industrial world in domains like manufacturing, maintenance etc. Comparing the scale of industries being deployed and managed, a huge number of IoT devices are being used which sends and receives tons of data sequentially. This has led to growing importance of Industrial Internet of Things also known as IIOT, which is basically the smart use of the existing sensors coupled with actuators to improve industrial processes and manufacturing pipeline in general. The whole idea and concept of IIOT is also one of the kind components of what is being described as the Fourth Revolution or in simpler words Industry 4.0. It provides the key opportunity to tap the power of smart machinery and real-time analysis of the data that have been produced in industry setup for years.

The concept of the Industrial Internet of Things (IIoT) is to take the advantage of Internet of Things (IoT) technology in the industrial control systems (ICSs). ICSs have been an important part of highly critical infrastructures and have been utilized for a very long time to supervise industrial machinery and various of its processes. They perform real-time monitoring, interacting with devices, collection and analysis of the data. Logging of all events is also very vital in any key industrial systems.

The Supervisory control and data acquisition (SCADA) is a control system architecture to monitor systems, gather and analyze real time data in an industrial setting. It is a subset of ICS. It provides a graphical user interface (GUI) through its human-machine interface (HMI). With HMI it makes its easy to observe the status of the system, interact with the IIoT devices, and receive alarms indicating abnormal behaviors.



CHAPTER-2

PROBLEM DEFINITION

- Many Industrial control systems use SCADA(Supervisory Control and Data Acquisition) for monitoring and control of automated systems.
- Though SCADA systems have undergone 4 generations of evolution and the last being Web-Based, even now the major concern is the lack of security and authentication in the design of SCADA networks with IIoT.
- The lack of security is a major concern for industries that employ IIoT for automation, it affects operations, causes financial loss, theft of intellectual property.

There are three stages of implementation involved in this problem -

1. To build and demonstrate a physical working prototype of a SCADA monitored IIOT system, where the SCADA system can connect to the IIOT setup through Modbus/TCP protocol and can monitor data.
2. To be able to retrieve data from Programmable Logic Controller (PLC) which in our case is Raspberry Pi 3 Model B to the SCADA System and generate our own dataset.
3. Do security vulnerability analysis on the MODBUS protocol traffic and try to fix any potential security vulnerability if possible.

CHAPTER-3

LITERATURE SURVEY

3.1 Machine Learning - Based Network Vulnerability Analysis of Industrial Internet of Things. (2019) [1]

- Maede Zolanvari, Graduate Student Member, IEEE, Marcio A. Teixeira, Senior Member, IEEE, Lav Gupta, Senior Member, IEEE, Khaled M. Khan, Member, IEEE, and Raj Jain, Life Fellow, IEEE, (IEEE Internet Of Things Journal, Vol. 6, No. 4, August 2019)

3.1.1 Observations

It is the involvement of Internet that has led to the development of Industry 4.0 and it is critical to secure the IIOT devices as these devices are connected to the Internet and are exposed to all the attacks that are prevalent with Internet like Distributed Denial of Service Attacks (DDoS), Man in the Middle Attacks, Spoofing, and many more. The impact is even more devastating because if the IIOT devices are compromised then the damage can collapse an industry, putting millions of people's lives in danger. Hence, it is very critical to secure the IIOT system consisting of thousands of sensors, actuators, Programmable Logic Controllers (PLI), etc. In the paper they have first discussed some of the common IIOT protocols and along with their associated vulnerabilities. They also have discussed various Attack methods. They have proposed that Machine Learning (ML) and big data analytics as two powerful technologies that can be leveraged for analysing and securing the Internet of Things (IoT) technology. Following the literature survey and comparative study of the common IIOT protocols and attack methods, they have worked on a case study on a real-world IIOT setup that was built to conduct cyber-attacks and designed an intrusion detection system (IDS). They demonstrate how a ML-based anomaly detection system can perform well in detecting these attacks.

This paper was very helpful in providing us an overview on all the IIOT protocols, its working, vulnerabilities, demerits and its working with the SCADA system. It also gave us the idea of building and developing our own IIOT environment and trying to understand its working.

3.2 Trends and Detection Avoidance of Internet - Connected Industrial Control Systems. (2019) [2]

- David Hasselquist, Abhimanyu Rawat, Andrei Gurtov, ADIT-IDA, Linköping University, Linköping, Sweden, Citation information: DOI 10.1109/ACCESS.2019.2948793

3.2.1 Observations

This paper talks about a search engine called Shodan which has the ability to search for devices that are connected to the internet. On one hand a very standard search engine like for example Google search engine works by indexing the content in the web, Shodan on the other hand indexes all the kinds of devices such as Industrial Control Systems (ICSs), webcams (web cameras), smart refrigerators, and many such devices that are connected to the internet. The best part is that Shodan is available freely and is widely used as a tool to detect vulnerable and compromised devices. But unfortunately this same tool, if held by wrong hands with malicious intents can also be used to gain access to such devices and exploit the vulnerability.

The study was conducted with the help of Shodan over a period of time in Sweden and the trend tries to understand the overall percentage distribution and the adoption of ICS protocols.
But, here something unnatural was observed. The study showcased that all studied countries except the United States of America (USA) have seen a decrease in the percentage of world total ICS devices.

However the research conducted suggested that in reality the number of ICS devices are actually increasing. Their findings showcase that the usage of age-old ICS protocols have been increasing and that industry devices running years old communication protocols have shown an increase in connection to the internet.

The paper also discusses the method in which it is possible to by-pass the Shodan radar and not get detected. This process as mentioned in the paper is called port knocking. They have a detailed case-study.

This paper gave us insights on how industrial devices that are connected to the internet can get compromised. It also showcased a process with which we could easily by-pass the Shodan search engine.

3.3 A Review of Intrusion Detection Systems for Industrial Control Systems [3]

- Mohamad Kaouk, Jean-Marie Flaus, Marie-Laure Potet, Roland Groz. Univ. Grenoble Alpes, France, 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT'19) | Paris, France / April 23-26, 2019

3.2.1 Observations

This paper talks about previous work done in and around IDS systems in Industrial control systems, its challenges and implementations. Gives more information about the ICS categorizing into 5 layers, level 0 to 4 through Physical process, Basic control, Supervisory control, Process management and enterprise system.

Talking about threats and security vulnerabilities in ICS, few of the considerations include Timeliness of data processing, Availability, Risk management such as confidentiality, integrity of data, human safety and fault tolerance, Physical Effects and Communications through protocols that are different from an IT industry

Shows various IDSs for ICS such as Signature based, Anomaly based which in turn are sub classified into Statistical, Machine learning and Specification basedes. And a tabular form displayed different IDS type, Data associated with it, and Method in which it's implemented.

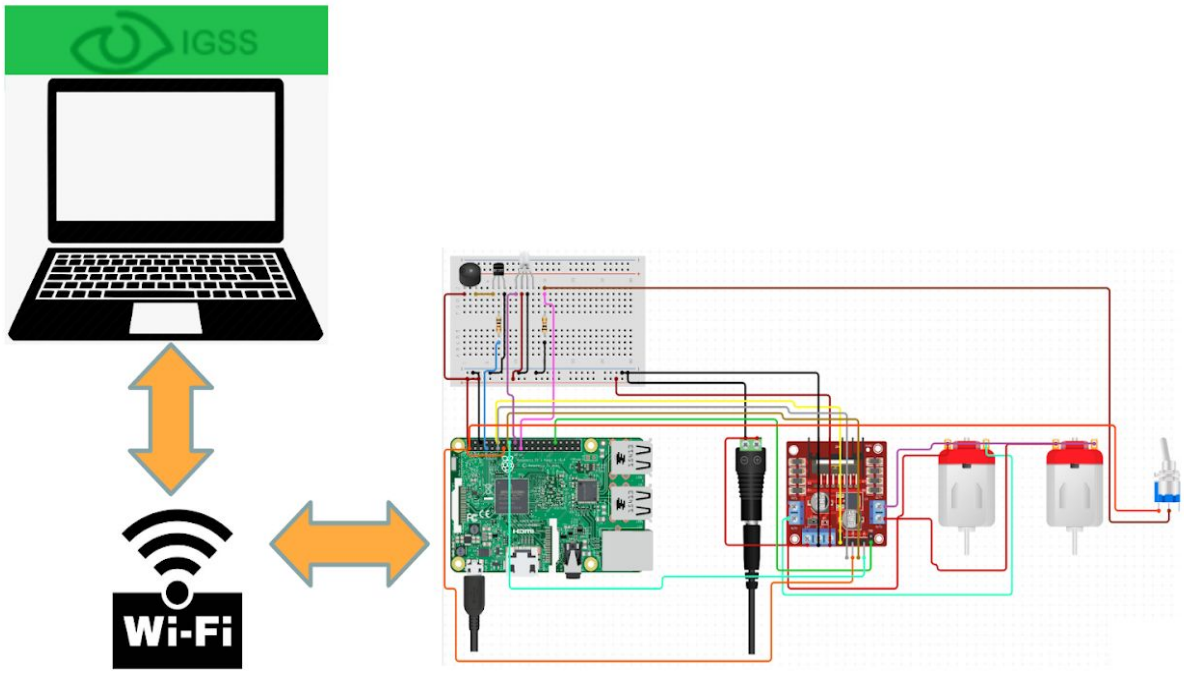
Which could be used for further enhancements of the IDS systems and increasing the security aspects of an Industrial control systems.

CHAPTER-4

PROJECT REQUIREMENT SPECIFICATION

4.1 Methodology

4.1.1 High Level System Architecture



(Figure 1 : High Level System Architecture)

4.2 Environment Requirements

4.2.1 Software Requirements

- **Python 2 or above** - Python is used to run the scripted code where data transaction between Raspberry pi and the IoT devices takes place.
- **Rasbian OS -** It is the Operating System that is installed in the Raspberry Pi Model B which provides GUI to see and run the python scripts over Raspberry terminal.
- **VNC Viewer -** It is a Remote desktop client that is installed in both the Rasbian OS, installed in the Raspberry Pi and a secondary desktop/laptop from which we can view the Rasbian OS over the internet.
- **IGSS -** Interactive Graphical SCADA System - is a state-of-the art SCADA system used for monitoring and controlling industrial processes. This system is provided by Schneider-Electric. The one that we are using is an open-source version of the software which has a limit of connecting upto 50 objects/sensors concurrently.

4.3 Use Case

4.3.1 Industries using SCADA systems

- Petroleum Refining
- Conventional Electric Power Generation
- Water Purification System
- Crane Control
- Chemical Plant
- Embedded Systems

4.3.2 Cases where SCADA Security is Important

- **HACKERS -** Individuals or groups with malicious intent can gain access to key SCADA components that can cause disruption in services to cyber warfare

- **MALWARE -** Viruses, spyware and ransomware can pose a risk to SCADA systems, it can still pose a threat to the key infrastructure that helps to manage the SCADA network.

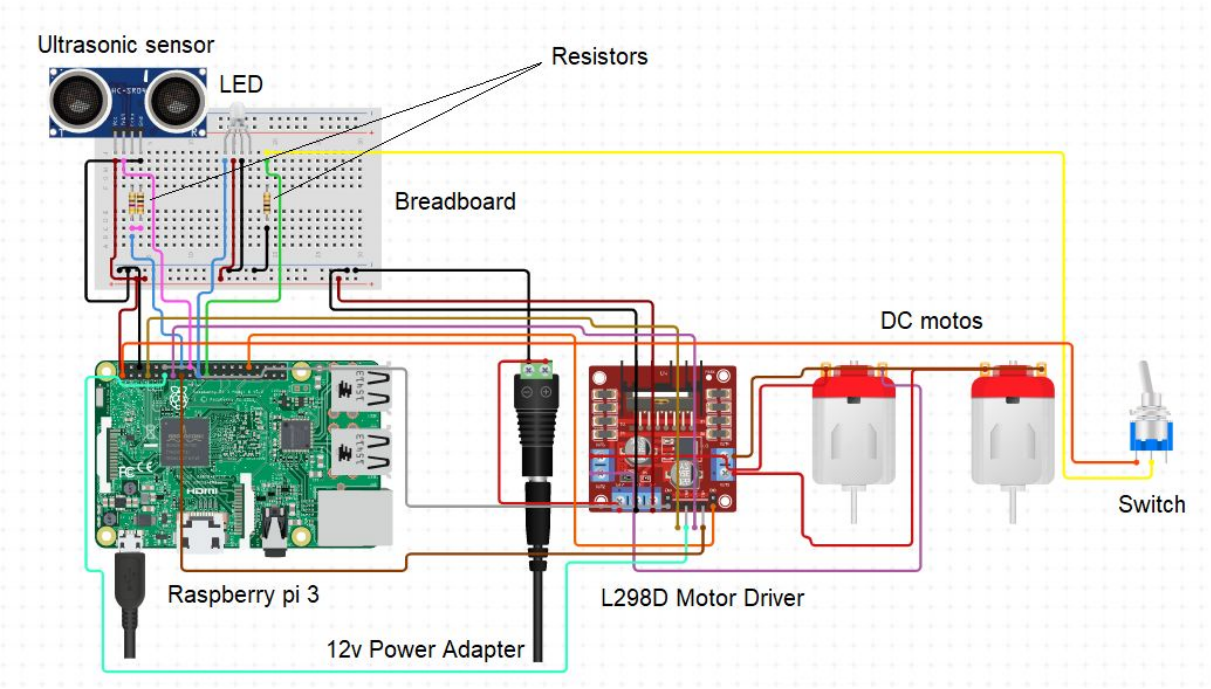
- **TERRORISTS -** Hackers are usually motivated by sordid gain desire to cause as much mayhem and damage as possible

- **EMPLOYEES** - Insider threats can be as damaging as external threats

CHAPTER-5

DETAILED DESIGN

5.1 Circuit Diagram



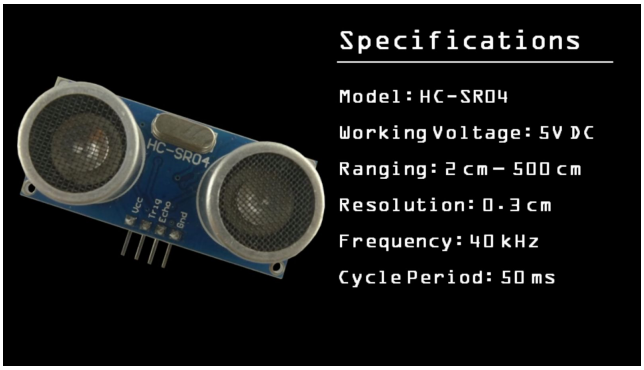
(Figure 2 : Circuit Diagram)

Three components, HC-SR04 Ultrasonic sensor, One Led and a motor are connected to a PLC here Raspberry Pi 3 model B using a breadboard.

The Led has been coded to change its state as and when a signal is received, Similarly the ultrasonic sensor has been coded to give out information for every specified time interval. Which is being controlled by the PLC that is being used. By powering up the sensor and led with appropriate voltage inputs and outputs.

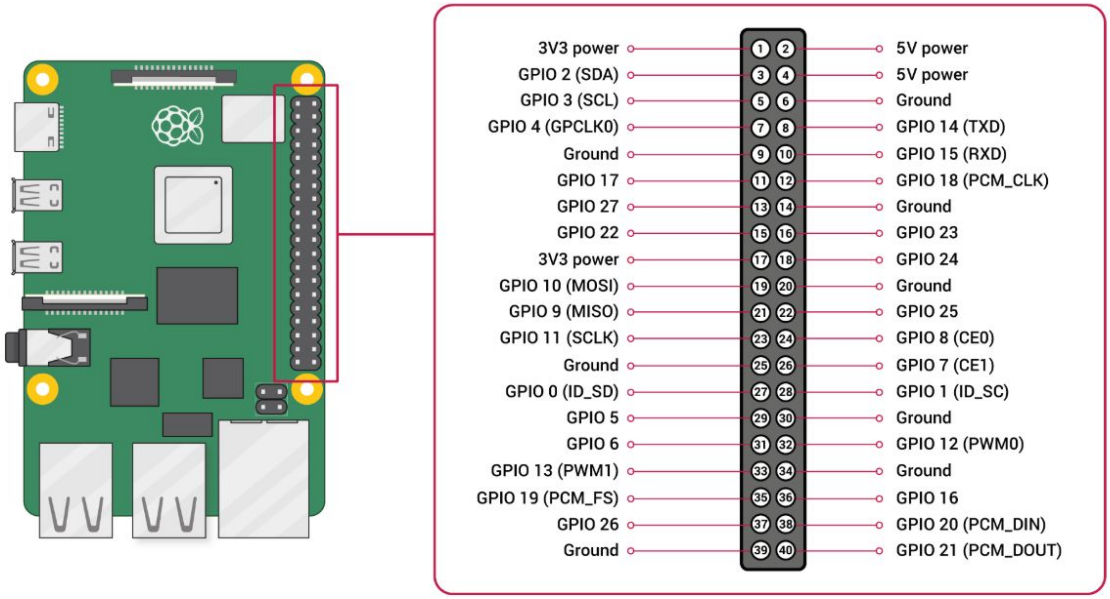
5.2 Components

5.2.1 HC-SR04 Ultrasonic Sensor



(Figure 3 : HC-SR04 Ultrasonic Sensor)

5.2.2 Raspberry Pi 3 model B



(Figure 4 : Raspberry Pi 3 model B)

5.3 Circuit setup

5.3.1 LED

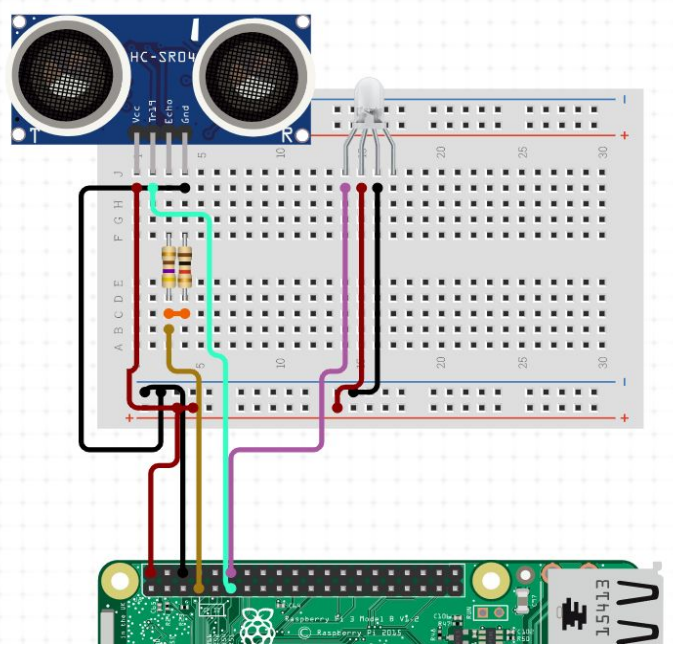
Led is connected in series with a resistor to 18-GPIO out and to ground.

5.3.2 Ultrasonic sensor

HC-SR04 Ultrasonic sensor operates at a constant 5 V. But Raspberry Pi 3 works at 3.3 V. In order to balance this we use the voltage divider rule, having 1 : 2 ratio of resistors.
Here 10k : 20k resistance.
PINS

- Vcc pin getting constant 5v supply
- Trig pin to 24-GPIO out
- Echo pin to 23-GPIO in
- Gnd splitting the voltage from echo to ground

Bread board was used to give a reusable circuit connection.

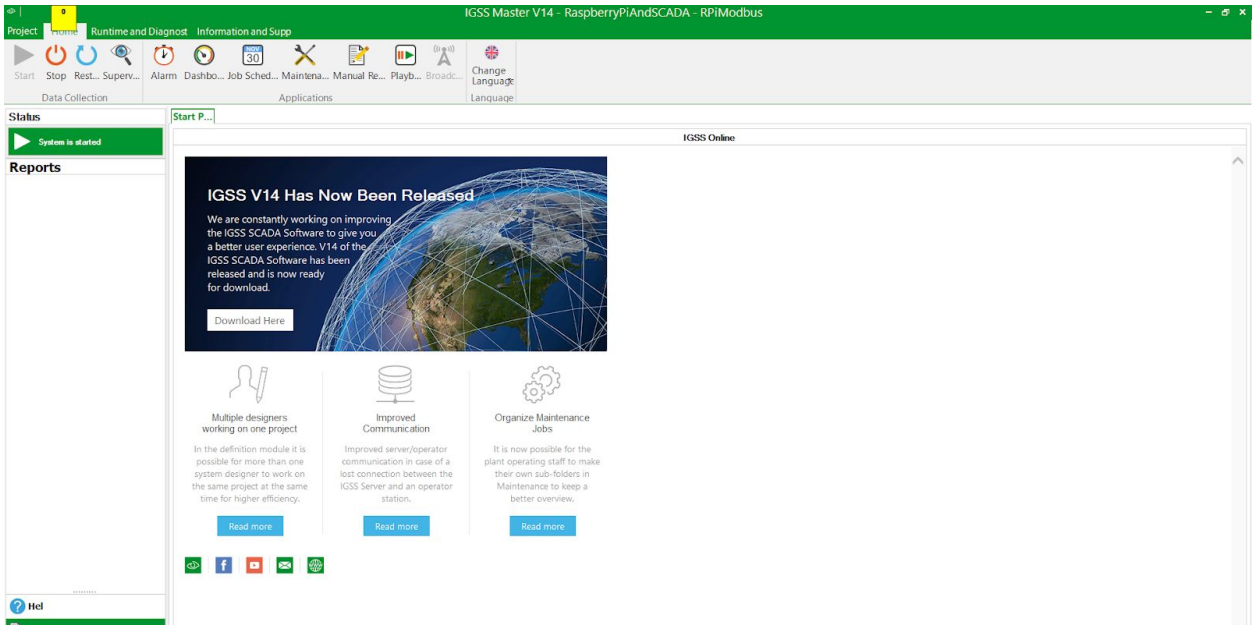


(Figure 5 : Device and GPIO Pin Diagram)

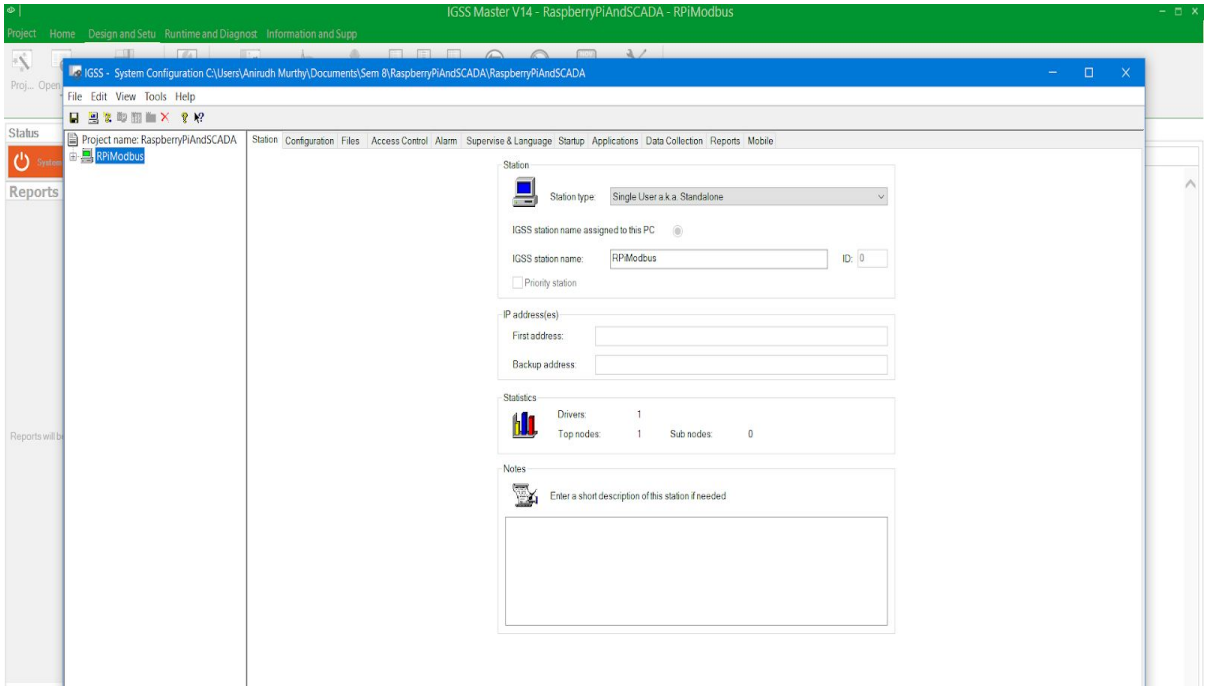
CHAPTER 6

SETUP AND SNAPSHOTS

6.1 Setting up IGSS project



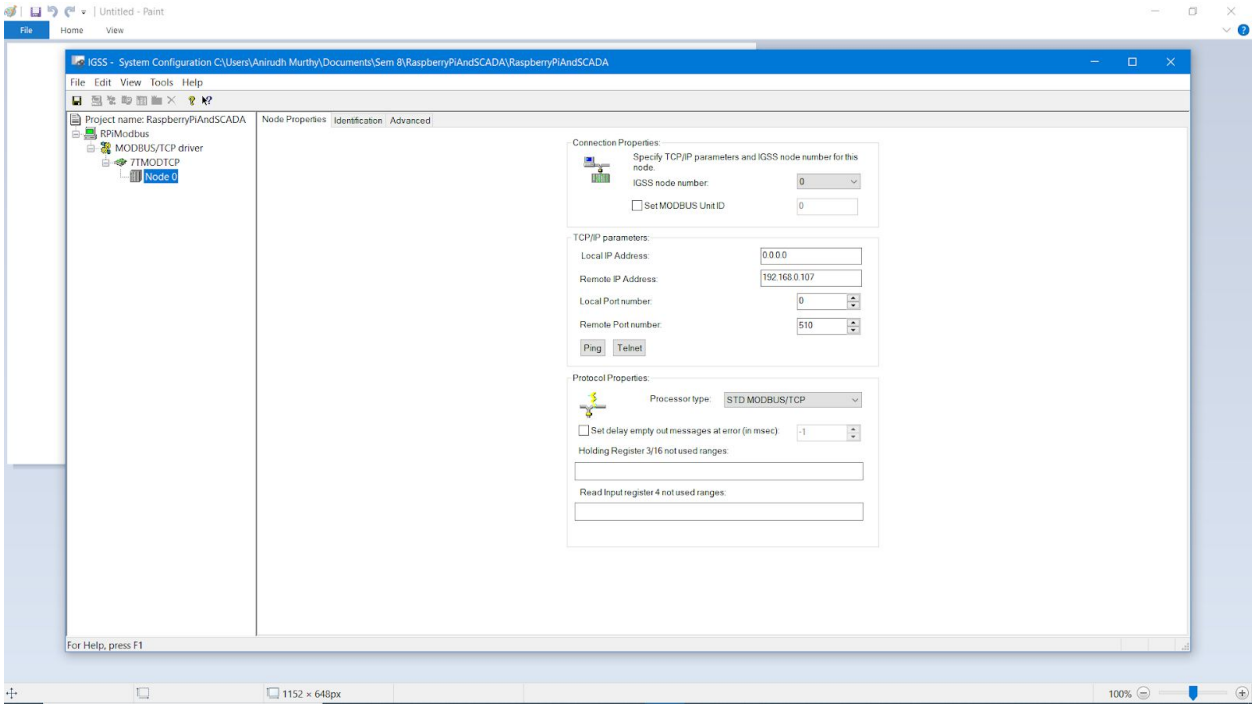
To set up the SCADA system, we are using IGSS by Schneider. This software allows us to set up a supervisor with objects that are a graphical representation of the sensors and actuators on the PLC. These objects control or reflect the state of the PLC which is used to monitor the system. This software also has an alarm feature that can be configured to give alerts if certain conditions are not met or when something fails.



To set up a connection to the PLC which in this case is a Raspberry pi 3, and modbus protocol is being used, we need to configure the station as RPIModbus.
We are using a single server station type since the SCADA system will be running as one instance on just one laptop.

The communication happens via TCP, so under the RPIMOdbus we are using the 7TMODTCP driver which enables communication through TCP/IP using a packet version of the Modbus protocol.

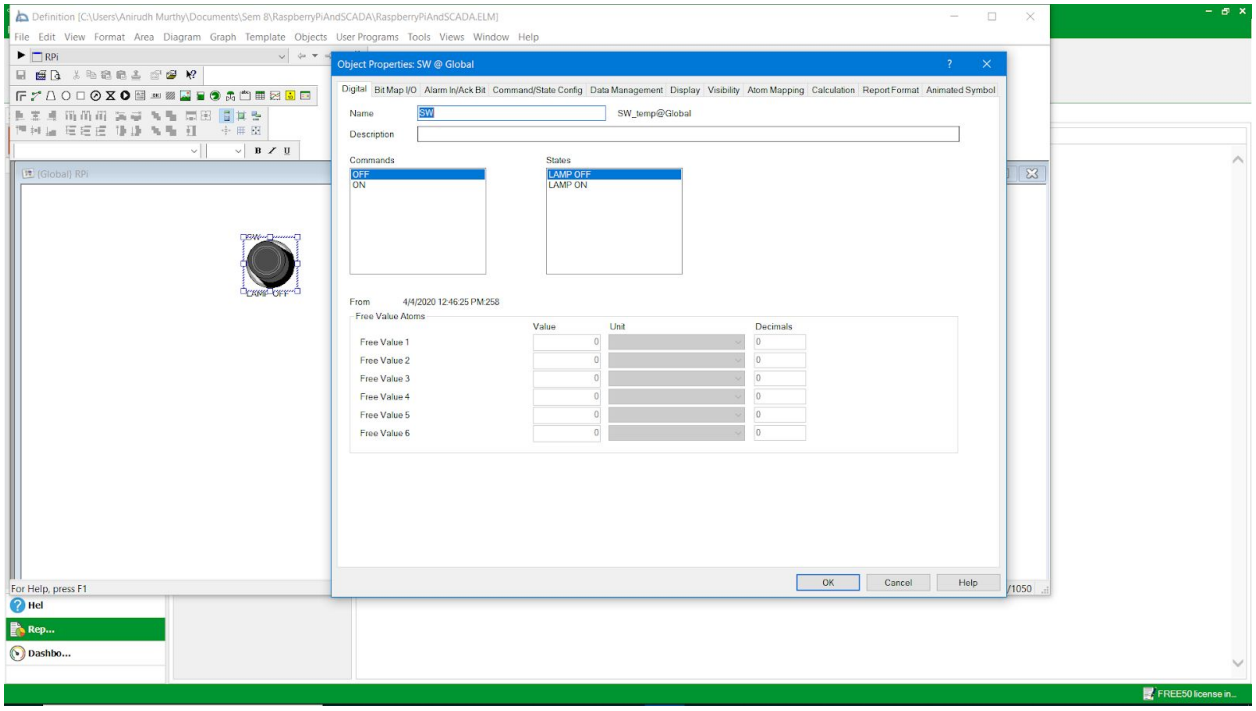
6.2 Setting up Remote IP address and port no.



The node here is numbered 0 and this refers to our only PLC which we are using the raspberry pi for, if other PLC units were to be used, there would be more nodes.
The IP of the PLC on the network is specified and the port number on which the program in the PLC is running is specified too. The IP of the SCADA system is set to 0.0.0.0 which is used to accept all incoming requests, so the program running in the PLC does not have to specify the IP and all data from it sent through modbus is forwarded to the SCADA system.

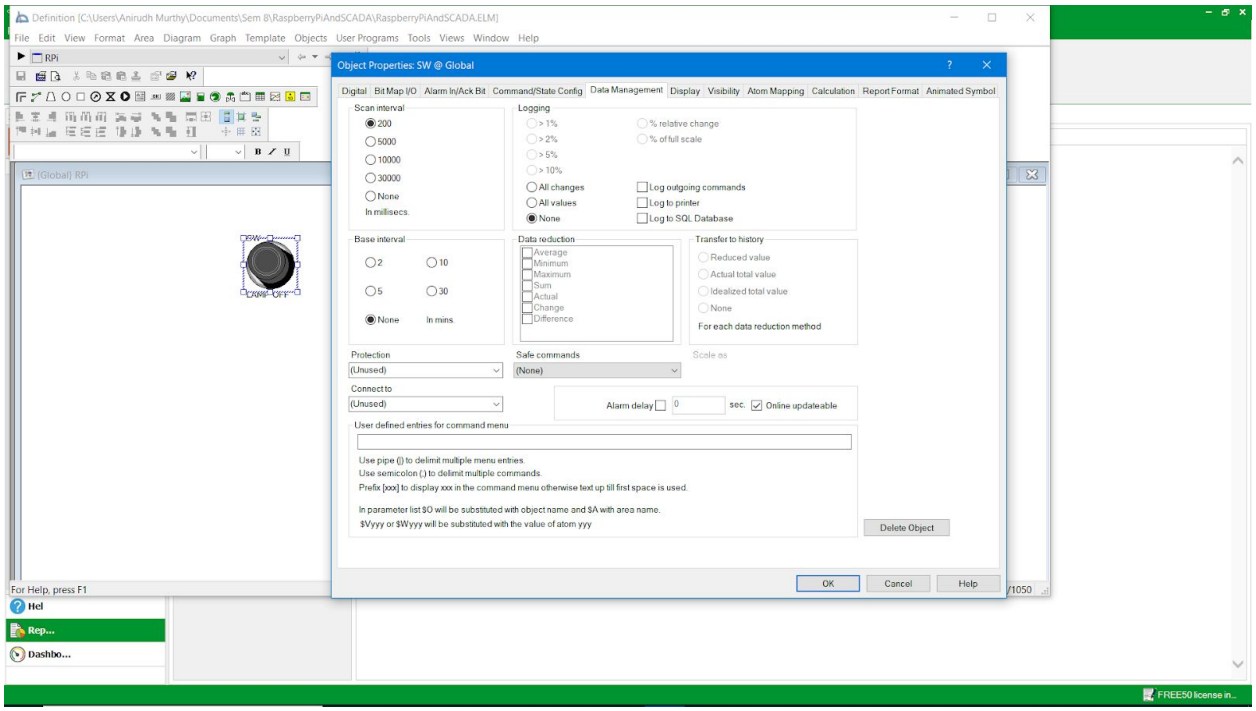
6.3 Setting up a switch for LED

6.3.1 Selecting a preset figure



An object in the IGSS software depicts the state of the PLC elements. These are monitored in the supervisor when a project is running, theses can be analog or digital, in the case of LED which has 2 states ON and OFF, it is configured as a digital object and the two states are represented in different colours with lamp off being black representing the LED to be switched off and lamp on being blue representing the LED to be on.

6.3.2 Data Management

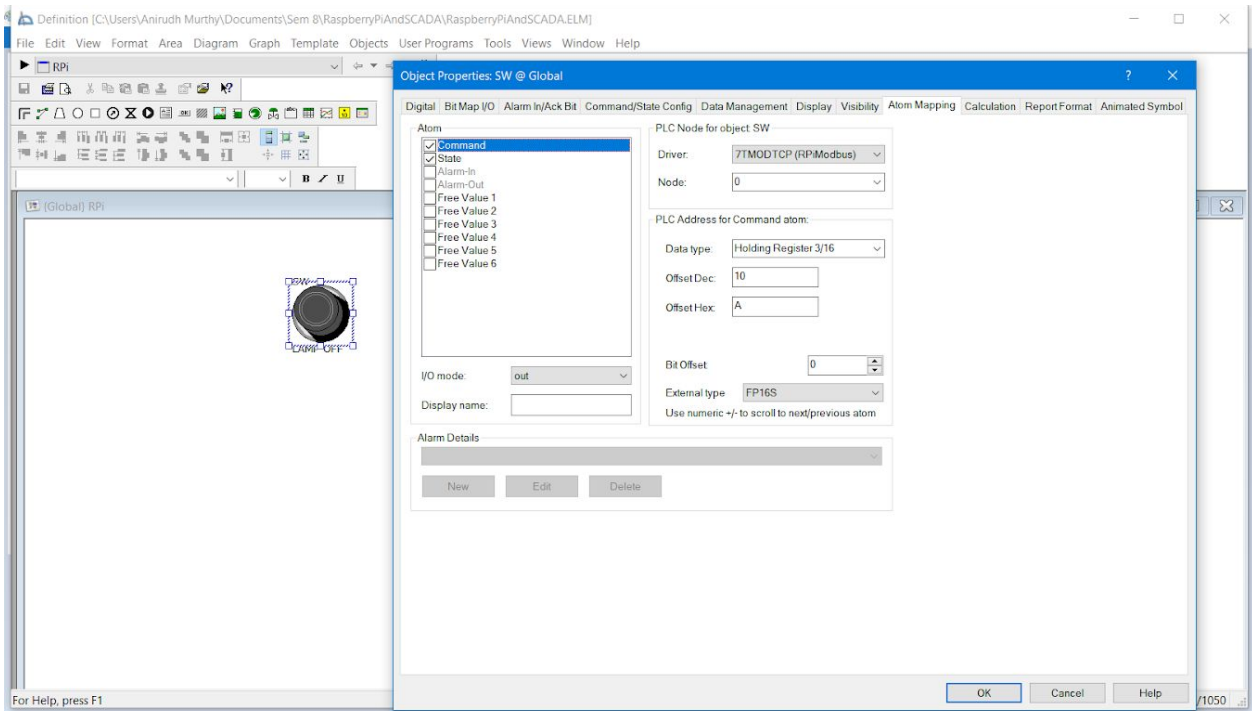


The LED switch sends data to the PLC, we need to configure the object as to how it sends the data.

In this case the data is sent at intervals of 200 milliseconds or 5 times in one second, so while the Modbus connection is open The PLC queries the SCADA 5 times a second.

Data logging is disabled and no protection, either encryption or authentication, is used.

6.3.3 Mapping the data

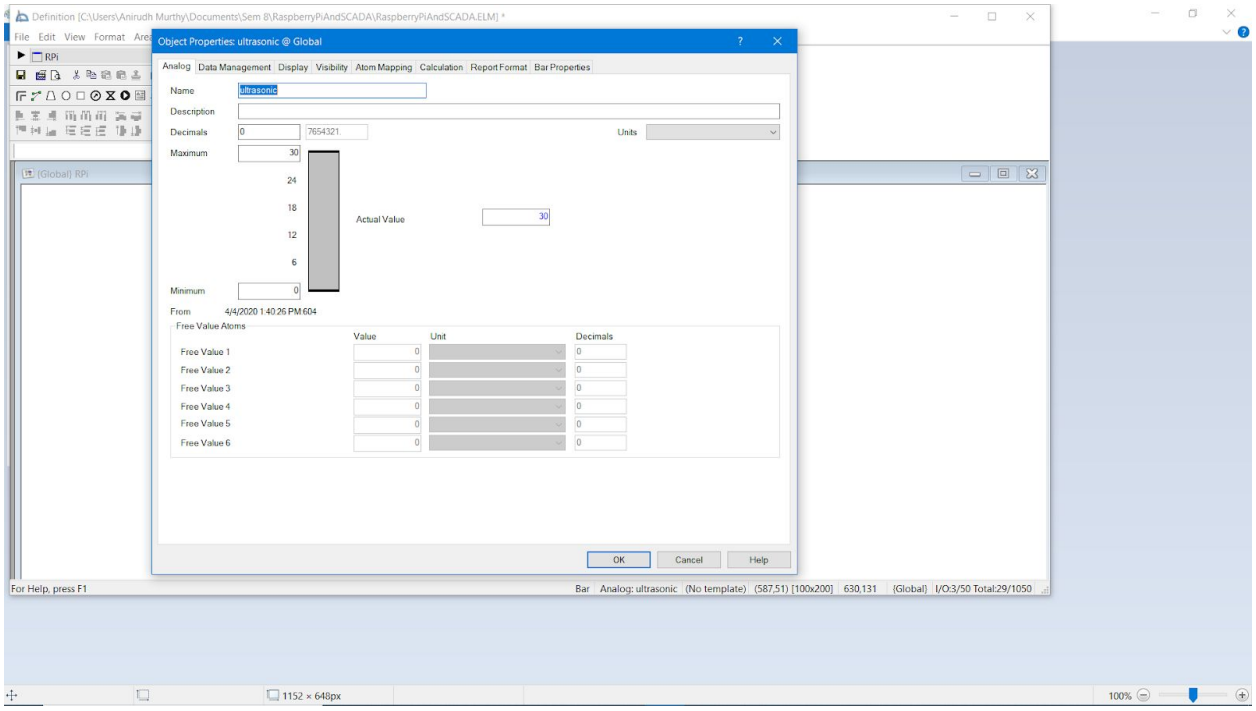


The data is sent and received over modbus but once sent, the data is just a contiguous block of memory, to make sense of it, the mapping needs to be configured.

Since we are sending the commands from SCADA to LED as this simulates an actuator, the I/O mode is out. The driver which is the protocol being used, here TCP/IP with Modbus, and the PLC, here node 0, is also specified. The register used here is the third one and the bit with the ON or OFF state (1 or 0 respectively) is stored in the 10th bit.

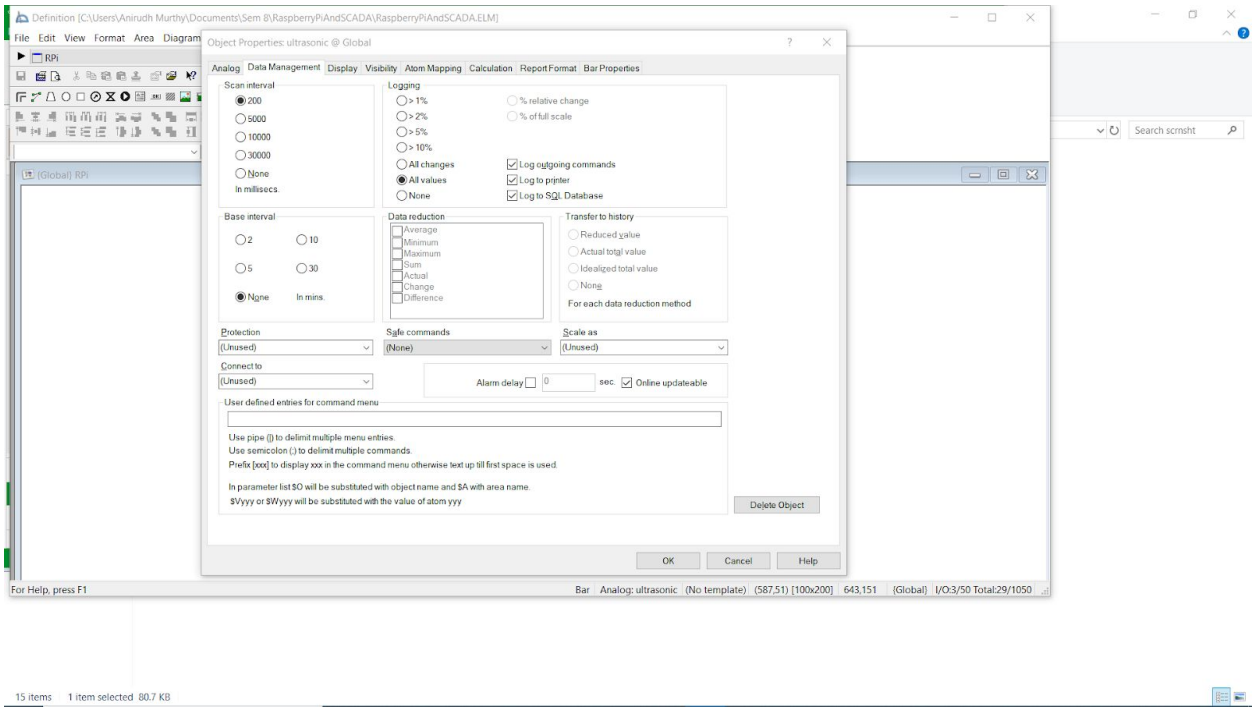
6.4 Setting up a display for the distance values

6.4.1 Selecting a preset figure



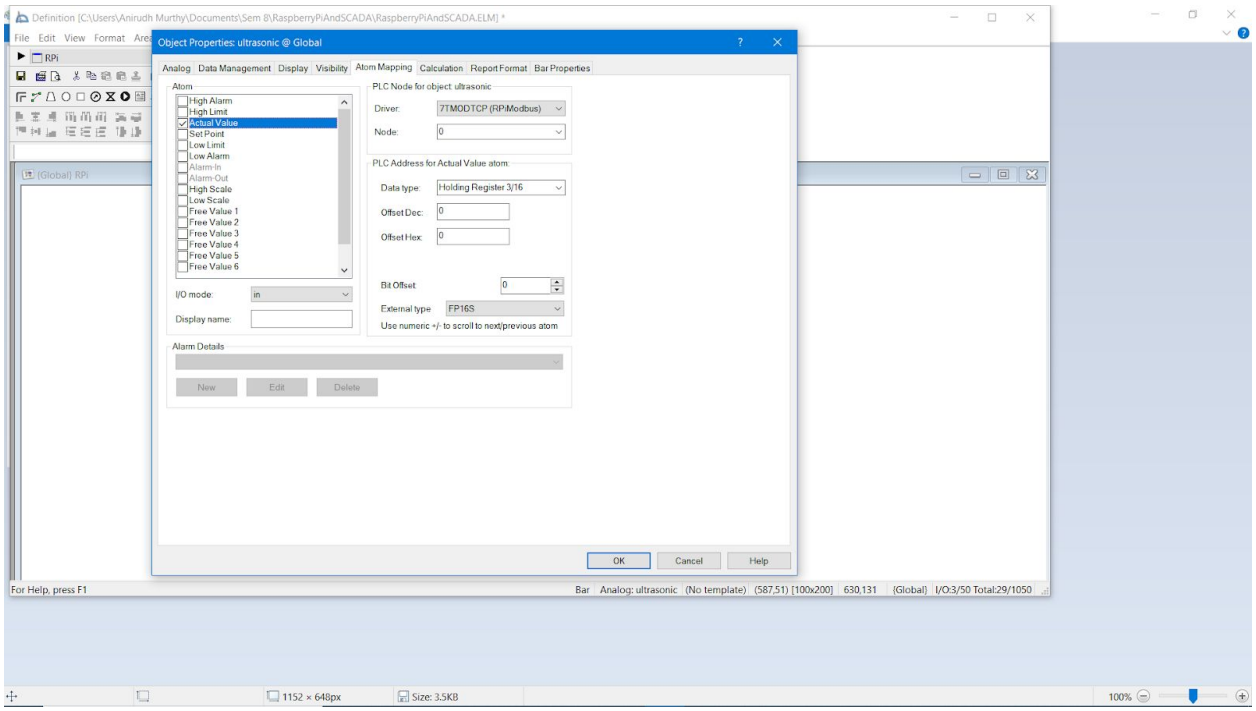
To display the distance values read by the ultrasonic sensor, a bar is used values of which range from 0 to 30. The bar represents the distance by filling the distance till the sensor in centimetres with blue and the rest black, so if the sensor when calibrated with no item in front of it within a 30 cm, is completely blue and considers no item to be in front of it and when an item passes by it, only the distance between them is filled with blue.

6.4.2 Data management



Here The data received happens in intervals of 200 milliseconds or 5 times a second.
All the values received are logged by the SCADA system and stored in .BCL files which can be converted to .csv files or can be extracted from object historian. No protection, either authentication with a password or encryption, is used.

6.4.3 Mapping the data

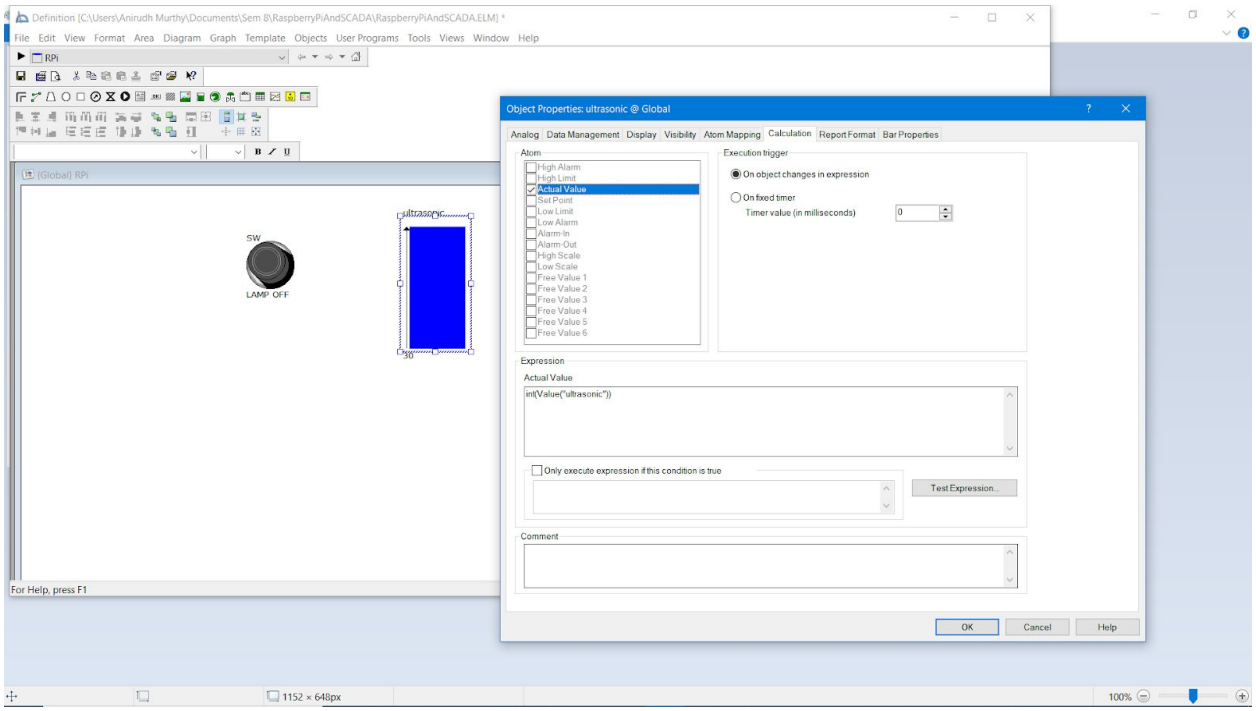


Since here the data is being read from the sensor in the PLC and SCADA receives the data, the I/O mode is in.

The driver is 7TMODTCP and the node is 0 referring to the protocol used for communication and the PLC which this object belongs to.

The register used here is the 3rd one out of 16 available registers. The data being received is read completely hence the address offset is 0 and bit offset is 0 too since the first integer value sent by the PLC is to be read.

6.4.4 Value expression

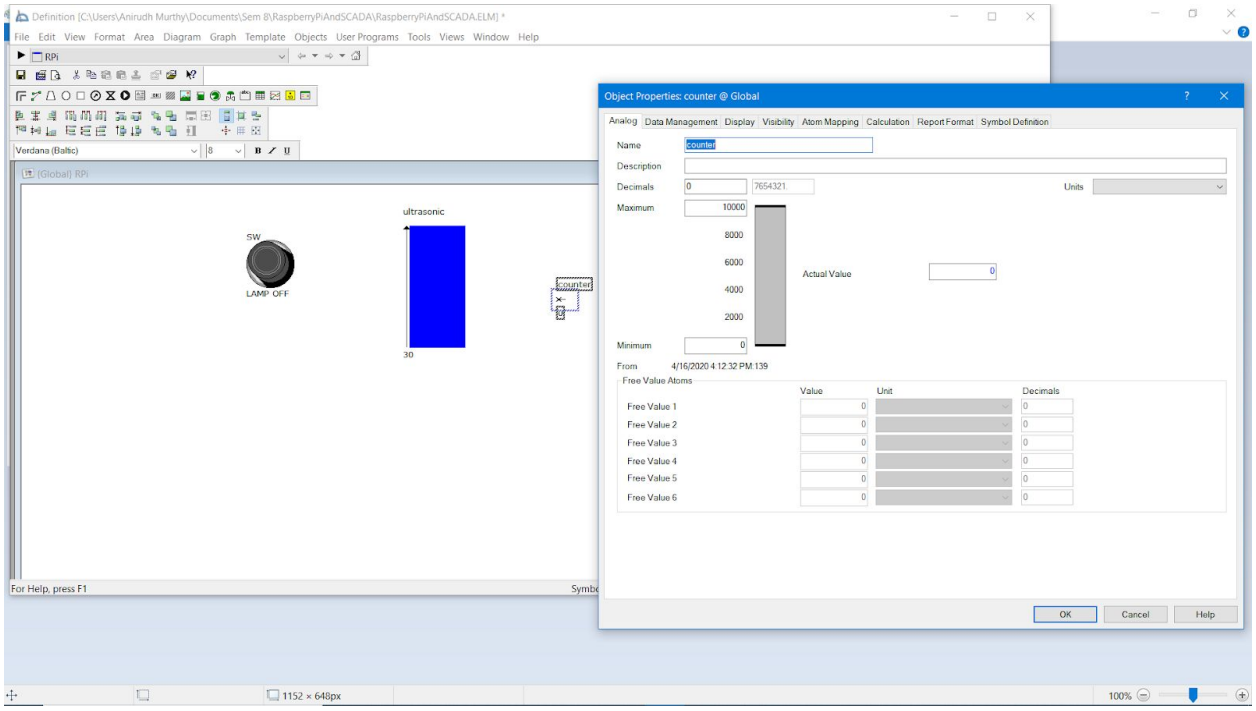


The value used for display is the data read from the sensor in integer form which removes any fractional part.

The trigger here is change in expression as opposed to a change every interval of time and this expression is not used conditionally.

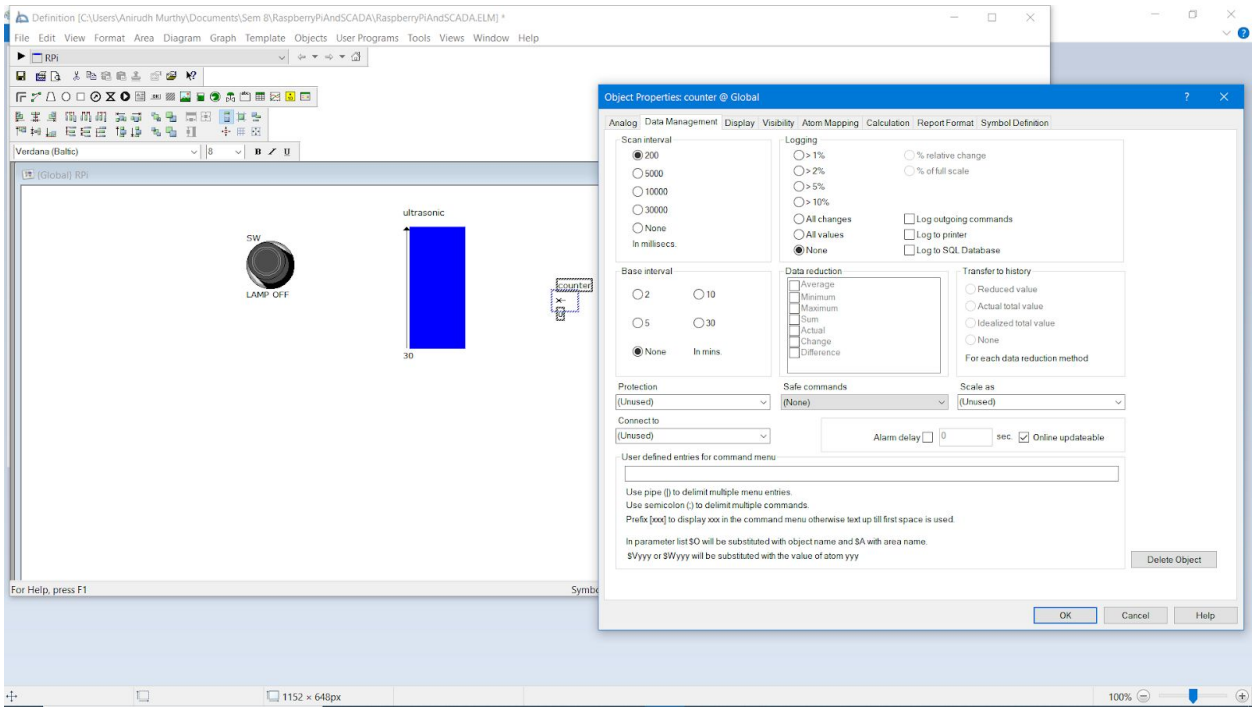
6.5 Setting up counter

6.5.1 Selecting display of the counter



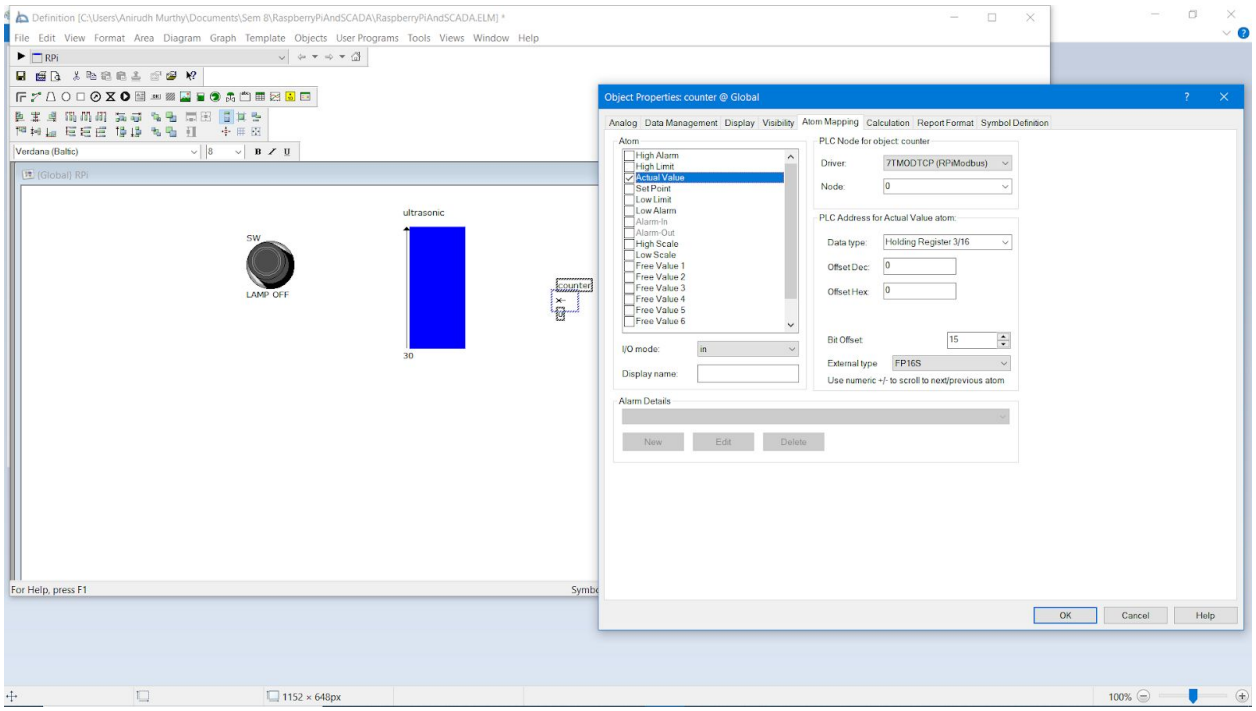
The counter displays the number of items that have passed by which it calculates from the data read by the ultrasonic sensor. This is implemented in the PLC itself but does not make use of a separate sensor, since this data is generated by the PLC the count value has to be received along with the ultrasonic sensor's distance values already being received.

6.5.2 Data Management



Here The data received happens in intervals of 200 milliseconds or 5 times a second.
All the values received are logged by the SCADA system and stored in .BCL files which can be converted to .csv files or can be extracted from the object historian. No protection, either authentication with a password or encryption, is used.

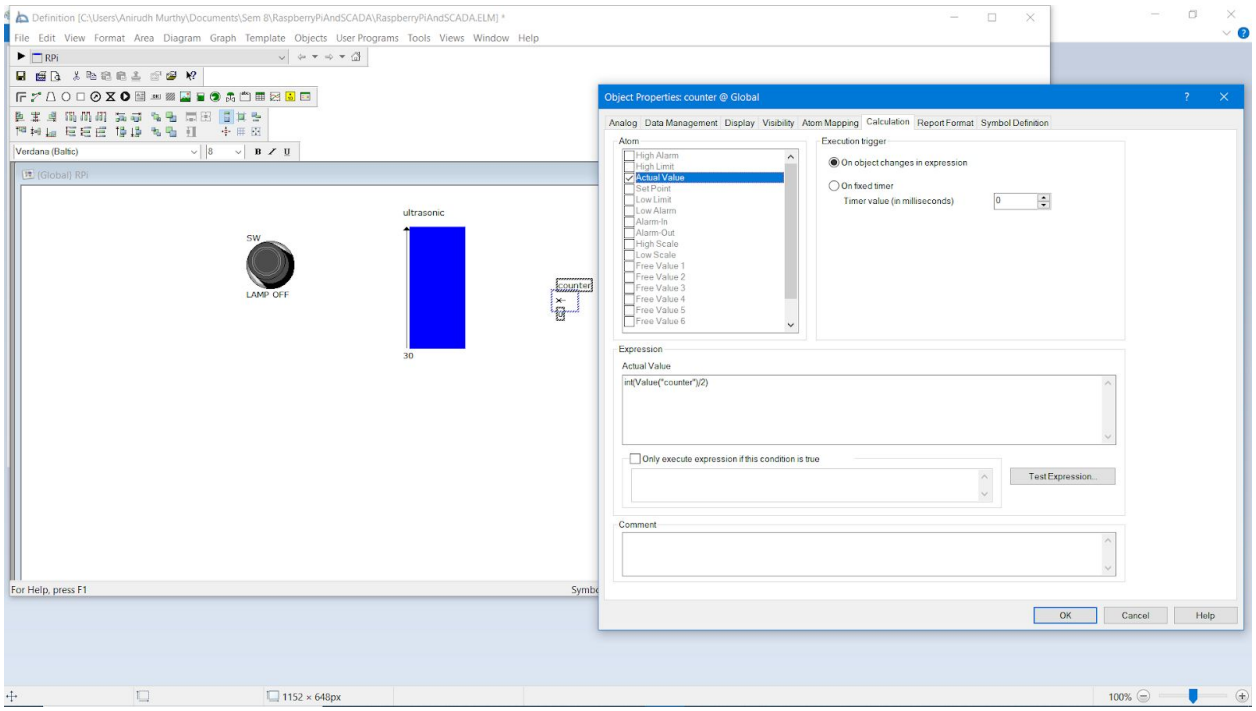
6.5.3 Mapping the data



Since data is being received from the PLC, the I/O mode is in, but the ultrasonic sensor's distance values are also being received from the PLC for the object displaying distance. Data is sent in one contiguous block of memory so there is only one chunk of data being received by the node, node 0 here referring to the PLC. The data is on the 3rd register, the address is taken as 0 so as to read the whole block of data including all the bits.

The Bit offset here should be so that the second integer value is taken from the data chunk, the first being data of the ultrasonic sensor. To do this a bit offset should be 16 since int in python has 16 bits(the PLC is running python code), but the software allows 15 to be the max permissible offset, so the value obtained will be left shifted once from the originally intended value sent by the PLC.

6.5.4 Correction and Value expression



The value to be displayed is the data received from the PLC divided by 2 since it is left shifted, so the expression is the integer value of the left shifted data read by the PLC divided by 2.
This corrects the left shit and displays the correct value of the items counted by the sensor, this expression is not used conditionally and the trigger here is change in expression and not over a time interval.

CHAPTER-7

IMPLEMENTATION AND PSEUDO CODE

7.1 PLC python code

For this IIOT simulation, we are connecting the SCADA (system running on a laptop) to the PLC (here raspberry pi3) via a TCP over Modbus protocol. The PLC(here raspberry pi3) is running a python program which receives and sends data with the help of the py modbus library. When running on a certain port number, the SCADA system establishes a connection with the PLC(here the raspberry pi 3) over the local network with the IP of the PLC(here the raspberry pi 3) and port number where the python program is running. The actual code itself provides the low level instructions for the actuator to be operated(here simulated by the LED) and to read and format the data read by the sensor(here the ultrasonic sensor).

```
from pymodbus.server.asynchronous import StartTcpServer
from pymodbus.datastore import ModbusSequentialDataBlock
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext
from twisted.internet.task import LoopingCall
import RPi.GPIO as GPIO

LED=18
BZ=17
SW=27
TRIG=24
ECHO=23
cur=0
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)
GPIO.setup(LED,GPIO.OUT)
GPIO.setup(BZ,GPIO.OUT)
GPIO.setup(SW, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.output(LED, GPIO.LOW)
GPIO.output(BZ, GPIO.HIGH)

```

The code also specifies the GPIO pins corresponding to the LED and the sensor, it also sets up the GPIO pins for input and output of the signal.

7.1.1 Context reading

The code has two main functions which query the SCADA system for values regarding the actuator in this case the LED which lights up via a GUI switch for the LED in the supervisor window running on the laptop designated as the SCADA system. When an operator switches it on, data is sent (either 1 or 0, cause the switch is a configured as a digital object in the SCADA) and the code communicates with the PLC(here the raspberry pi3) and sets the gpio pin to which the LED and resistor circuit is connected to to high. This is done by a Pymodbus function getvalues which has a parameter which has the data value, another for a certain address(10 here) of the bit where the data is and one for mentioning the register address(3 here). This is done on the context object which has a slave id for the node of the PLC, here node being 0.

```
22
23  store = ModbusSlaveContext(
24      di = ModbusSequentialDataBlock(0, [0]*100),
25      co = ModbusSequentialDataBlock(0, [0]*100),
26      hr = ModbusSequentialDataBlock(0, [0]*100),
27      ir = ModbusSequentialDataBlock(0, [0]*100))
28  context = ModbusServerContext(slaves=store, single=True)
29
30
31
32
```

```
72
73
74  def read_context(a):
75      context  = a[0]
76      register = 3
77      slave_id = 0
78      address  = 10
79      value = context[slave_id].getValues(register,address,10)
80      if value[0]==0:GPIO.output(LED, GPIO.LOW)
81      if value[0]==1:GPIO.output(LED, GPIO.HIGH)
82      dateTimeObj = datetime.now()
83
84      if(value[0]==0):
85          print(" : LED OFF ")
86      else:
87          print(" : LED ON ")
88
89
90
```

The read context function calls get value and depending on the state of the switch, it sends HIGH or LOW signal to the GPIO pin number 18 represented by the variable LED.

7.1.2 Context writing

The other main function is for writing values read by the sensor, here the values being distance in centimetres from an ultrasonic sensor and sends it to the SCADA system, in an array since this is not a digital sensor and has a range of values, so it sends it to an analog object as a continuous data block. With the help of the sensor apart from displaying the distance there is a mechanism to count the number of objects that pass by the sensor to implement an object counter of items that may be moving in a conveyor system in a production line for example. To implement that the distance from the sensor where no item is placed in front is calibrated, in our case 100 cm, if an object is passed in front of it, the distance will be less than 100 and for that time span where the sensor values are read(here 2 times in a second) for every read, the value is less than 100cm, the counter increases value by one. To send both the distance and count values, they are put in the contiguous block of data as two integer values of 16 bits.

```
GameFragment.kt   cool.py   README.txt   from pymodbus.server.asynchronous import StartTcpS   import time

38
39
40
41  values = [0,0]
42  def updating_writer(a):
43      context  = a[0]
44      register = 3
45      slave_id = 0
46      address  = 0
47      GPIO.output(TRIG, False)
48      time.sleep(0.5)
49      GPIO.output(TRIG, True)
50      time.sleep(0.00001)
51      GPIO.output(TRIG, False)
52      while GPIO.input(ECHO)==0:
53          pulse_start = time.time()
54      while GPIO.input(ECHO)==1:
55          pulse_end = time.time()
56      pulse_duration = pulse_end - pulse_start
57      distance = pulse_duration * 17150
58      distance = int(distance)
59      print "Distance:",distance,"cm"
60      values[0] = distance
61      if(distance  < 100):
62          values[1]= values[1]+1
63      context[slave_id].setValues(register,address,values)
64      print("Ultrasonic value",values)
65
66
67
68
69
```

Here the output of the trigger pin in the ultrasonic sensor of the is activated and the ultrasonic wave is sent for 0.00001 seconds and the wave itself is sent every 0.5 seconds. Once the wave is sent and the sensor stops sending more, the echo pin's value is read and time is kept till it receives back the sound. With this time value which is the time taken for sound to go to the nearest object the sensor is facing and come back, with which using the speed of sound in air the distance value is calculated using distance = speed * time/2 since we have calculated the time for the wave to cover twice the distance, which results in the expression time * 17150 which gives the distance in centimetres.The distance value is stored in the first array element of the value integer array and the counter value which counts the number of times the distance value goes below 100 is stored in the second array element.

7.1.3 Looping and server start

These functions need to be called repeatedly for the data to be queried on both the PLC and SCADA ends continuously. This is done through a looping call and the values are printed in the terminal here for convenience but in reality the PLC is not operated through a GUI.

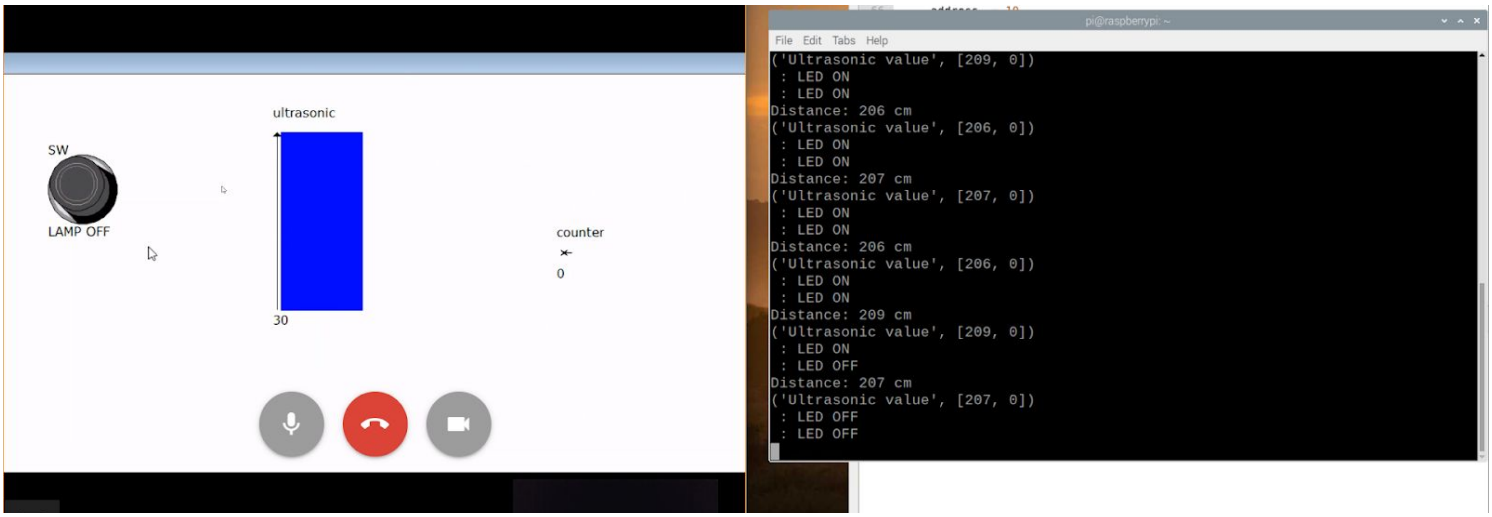
The TCP server here is running on port number 510 which can be modified and the functions are called 5 times a second.

CHAPTER-8

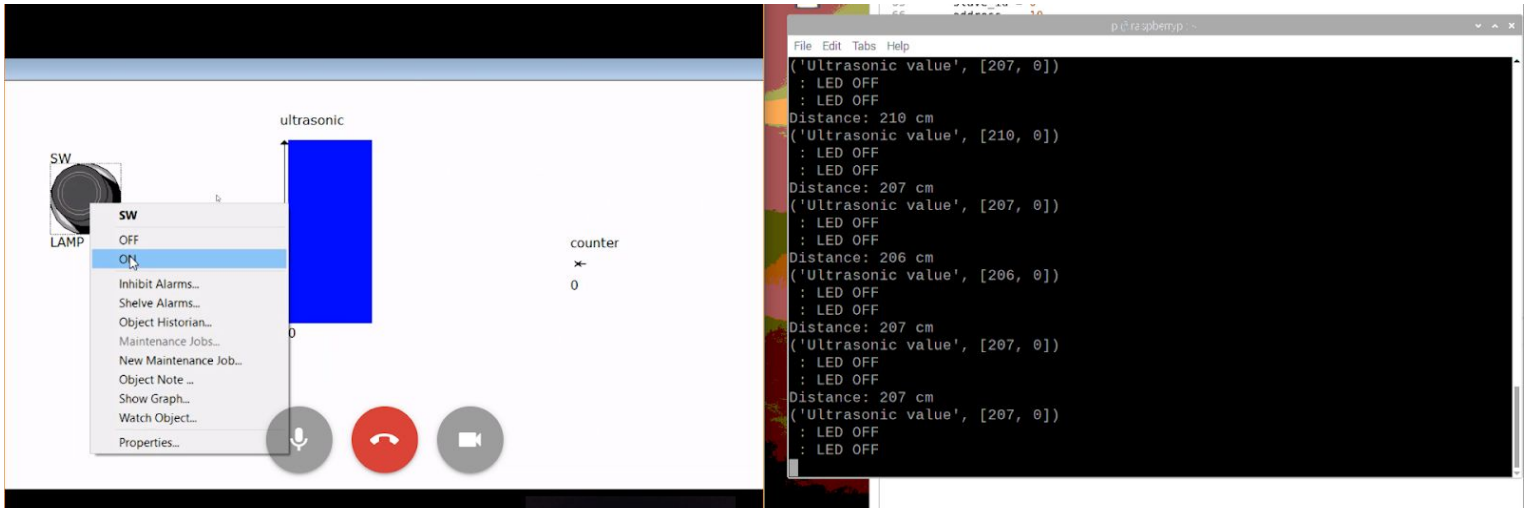
RESULTS AND DISCUSSION

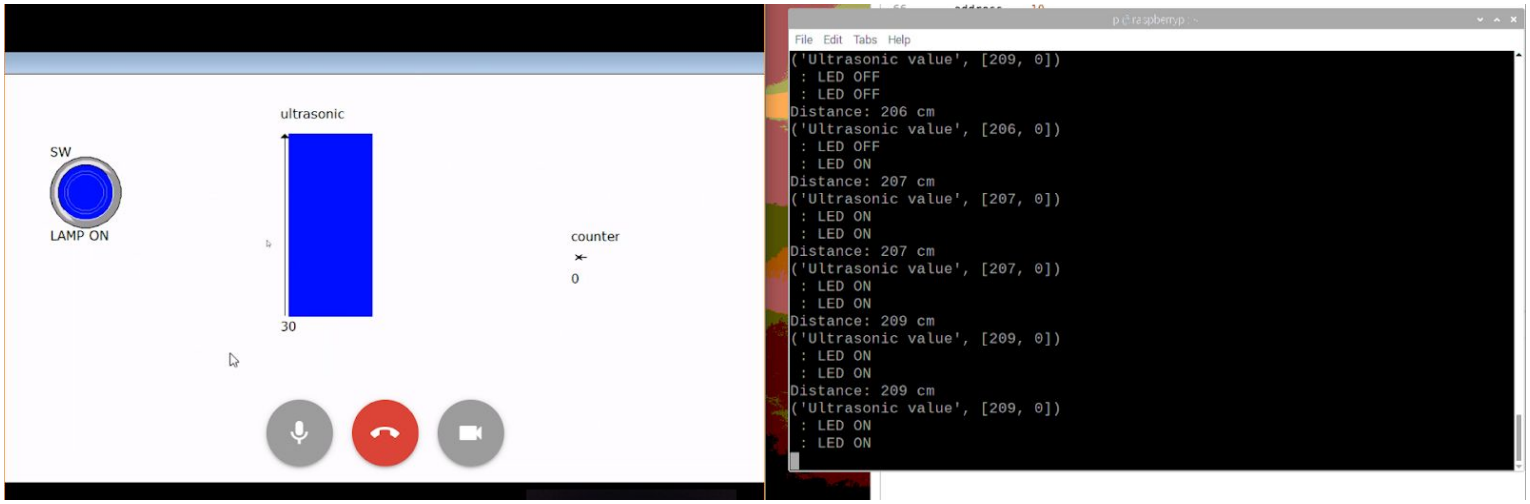
8.1 LED OFF, Counter = 0

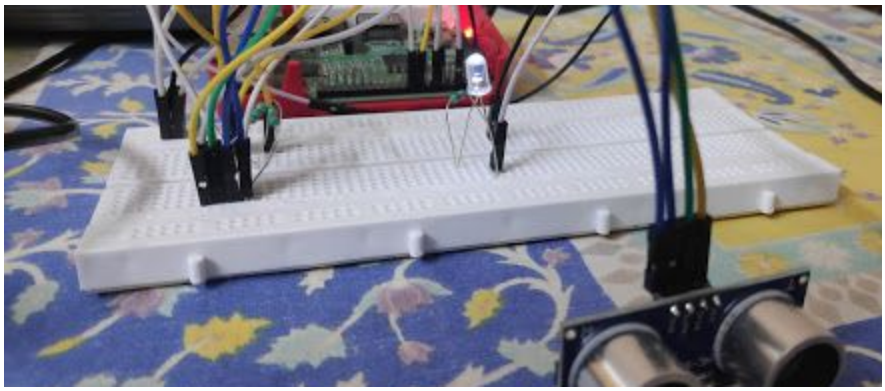
Left : IGSS interface
Right : Raspberry Pi terminal
And a picture of the circuit



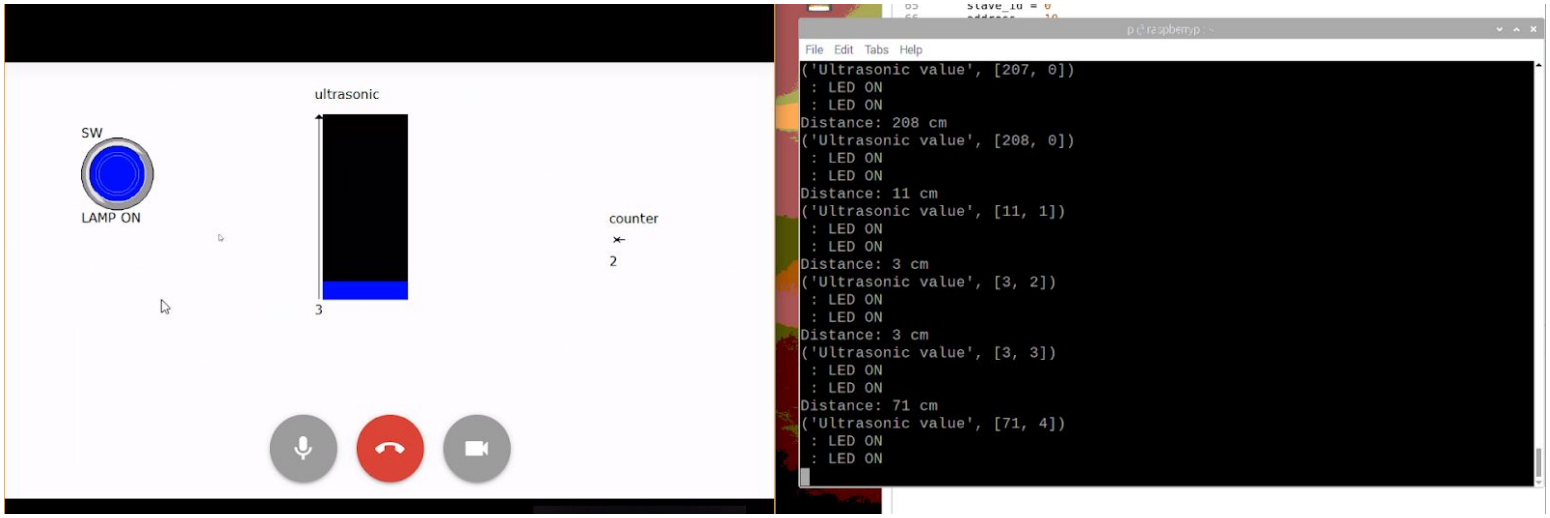
8.2 LED ON

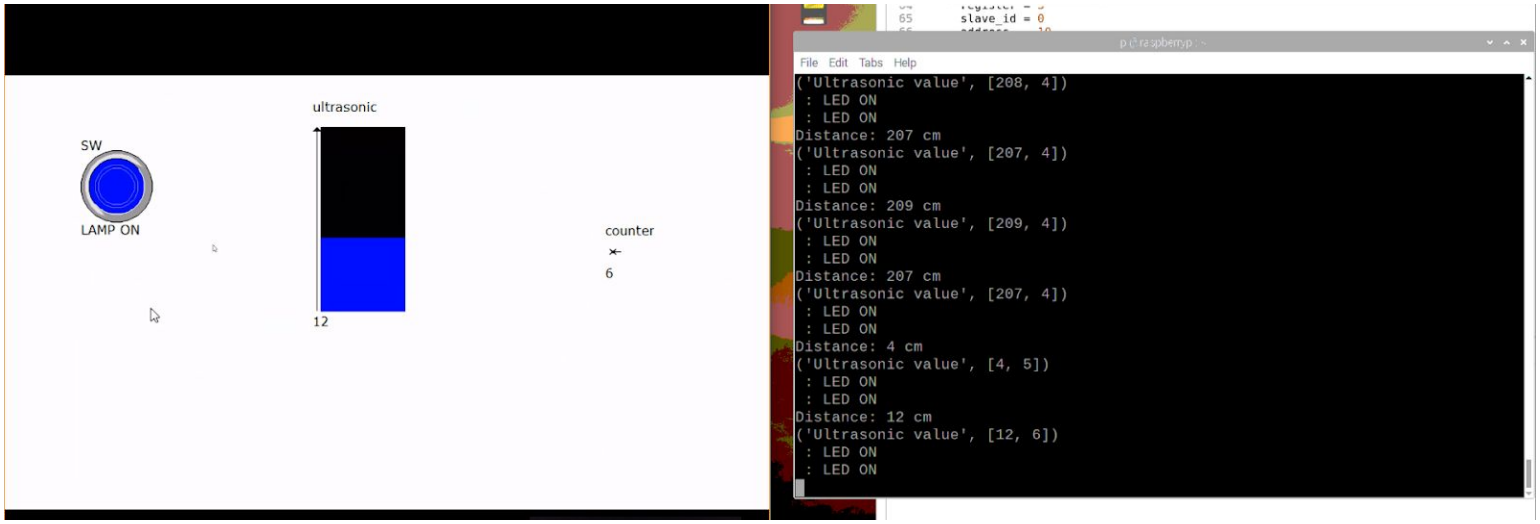

ultrasonic
SW
LAMP
SW
OFF
ON
Inhibit Alarms...
Shelve Alarms...
Object Historian...
Maintenance Jobs...
New Maintenance Job...
Object Note ...
Show Graph...
Watch Object...
Properties...
counter
0
0
File Edit Tabs Help
('Ultrasonic value', [207, 0])
: LED OFF
: LED OFF
Distance: 210 cm
('Ultrasonic value', [210, 0])
: LED OFF
: LED OFF
Distance: 207 cm
('Ultrasonic value', [207, 0])
: LED OFF
: LED OFF
Distance: 206 cm
('Ultrasonic value', [206, 0])
: LED OFF
: LED OFF
Distance: 207 cm
('Ultrasonic value', [207, 0])
: LED OFF
: LED OFF
Distance: 207 cm
('Ultrasonic value', [207, 0])
: LED OFF
: LED OFF


ultrasonic
SW
LAMP ON
counter
0
30
File Edit Tabs Help
('Ultrasonic value', [209, 0])
: LED OFF
: LED OFF
Distance: 206 cm
('Ultrasonic value', [206, 0])
: LED OFF
: LED ON
Distance: 207 cm
('Ultrasonic value', [207, 0])
: LED ON
: LED ON
Distance: 207 cm
('Ultrasonic value', [207, 0])
: LED ON
: LED ON
Distance: 209 cm
('Ultrasonic value', [209, 0])
: LED ON
: LED ON
Distance: 209 cm
('Ultrasonic value', [209, 0])
: LED ON
: LED ON



8.3 Counter Increased by passing objects





8.4 Exploiting the vulnerabilities

When the HMI requires a value from the PLC it sends a request message to initiate data transfer.

The PLC then sends back a response with information written to the response.

Each message has function code that is set by the master which shows what actions need to be performed by the slave. Which generally tells which table of information to look up to read or write.

8.4.1 Return illegal response

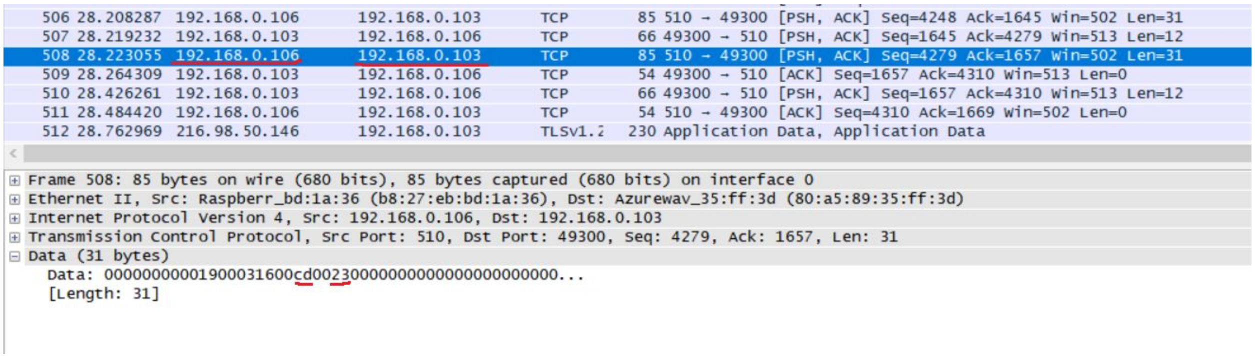
- Slave may return an illegal function exception response for an unsupported function code.
- An attacker can exploit this by sending generated function codes to perform actions on the target system.

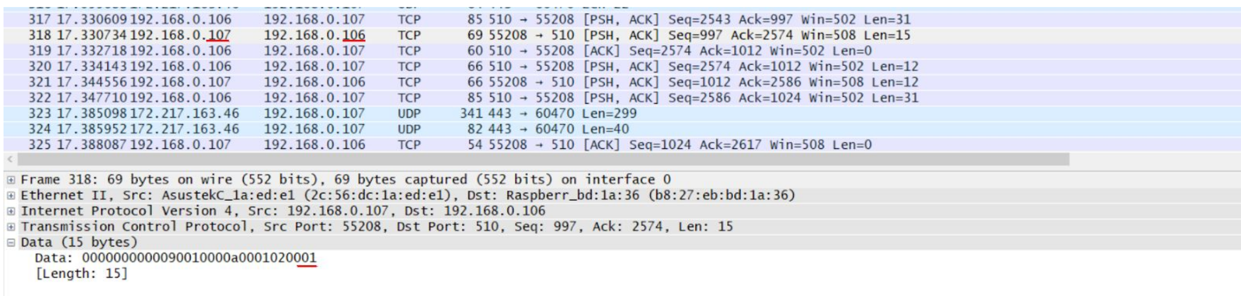
8.4.2 Illegal address Exception response

- For queries that have illegal slave address and illegal address exception response is generated.
- An attacker could exploit this by sending invalid slave addresses and collecting information of targeted hosts.

8.4.3 Lack of security checks in SCADA Modbus/TCP

- Protocol does not involve an authentication for validating the requests or response.
- An attacker could issue any commands to any slave device through a master.
- This could also lead to spoofed packets being sent for a masquerade attack which can be achieved by observing packets on wireshark ,which is possible due to lack of confidentiality of data being sent over the network, and a python script using the scapy library to create packets with spoofed addresses.
  Capturing IP, MAC and data pattern being sent and received between SCADA and raspberry pi. As marked, we can see the two values, the distance and count values in the first image(205cm and 35) and in the second image we can see that the led is switched on.





This demonstrates lack of confidentiality and serves as the base for spoofing as it gives us the required information. Using the following code, which employs the scapy library we set up a tcp connection between the SCADA and the raspberry pi by faking a three way handshake and sending data to switch on the LED pretending to be the SCADA sending requests, this can be done the other way around too, from raspberry to SCADA faking the sensor data.

send.py

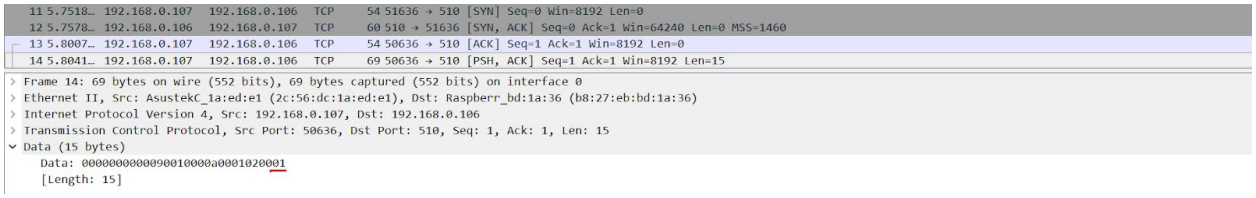
```
from scapy.all import *

#sport = random.randint(1024,65535)

# SYN
ip=IP(src='192.168.0.107',dst='192.168.0.106')
SYN=TCP(sport=53917,dport=510,flags='S',seq=100)
SYNACK=sr1(ip/SYN)

# SYN-ACK
data='\x00\x00\x00\x00\x00\x09\x00\x10\x00\x0a\x00\x01\x02\x00\x01'
#data='\x00\x00\x00\x00\x00\x19\x00\x03\x16\x00\xcd\x00\x0a\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
my_ack=SYNACK.seq+1
ACK=TCP(sport=53917, dport=510, flags='A', seq=101, ack=my_ack)
send(ip/ACK)
PUSH=TCP(sport=53917, dport=510, flags='PA', seq=101, ack=my_ack+1)
arp =Ether(src="ab:cd:ef:ab:cd:ef", dst="b8:27:eb:bd:1a:36")
reply=send(ip/PUSH/data)
#tcp handshake with 3rd packet sent along with payload
#this is to spoof packets and gain access to the plc maliciously
```

The result of running this code will result in the LED being turned on, and these are the packets captured by wireshark.



The three way handshake being set up and the false data to trigger the LED being sent once the connection is established.

8.4.4 Chances of Dos attacks

- There is presence of implementation error in the protocol when reading many input request and response messages, which could lead to Dos attacks.

8.4.5 Exceeding maximum PDU(Protocol data unit)

- Modbus limits PDU to 253 bytes plus 7 bytes for MBAP(Modbus Application Protocol) header, which adds to 260 bytes encapsulated in a TCP packet.
- An attacker could create a packet of size more than 260 bytes and if master or slave were implemented incorrectly, this can lead to Buffer Overflow or Dos attacks.

8.5 Overcoming some vulnerabilities of Modbus :

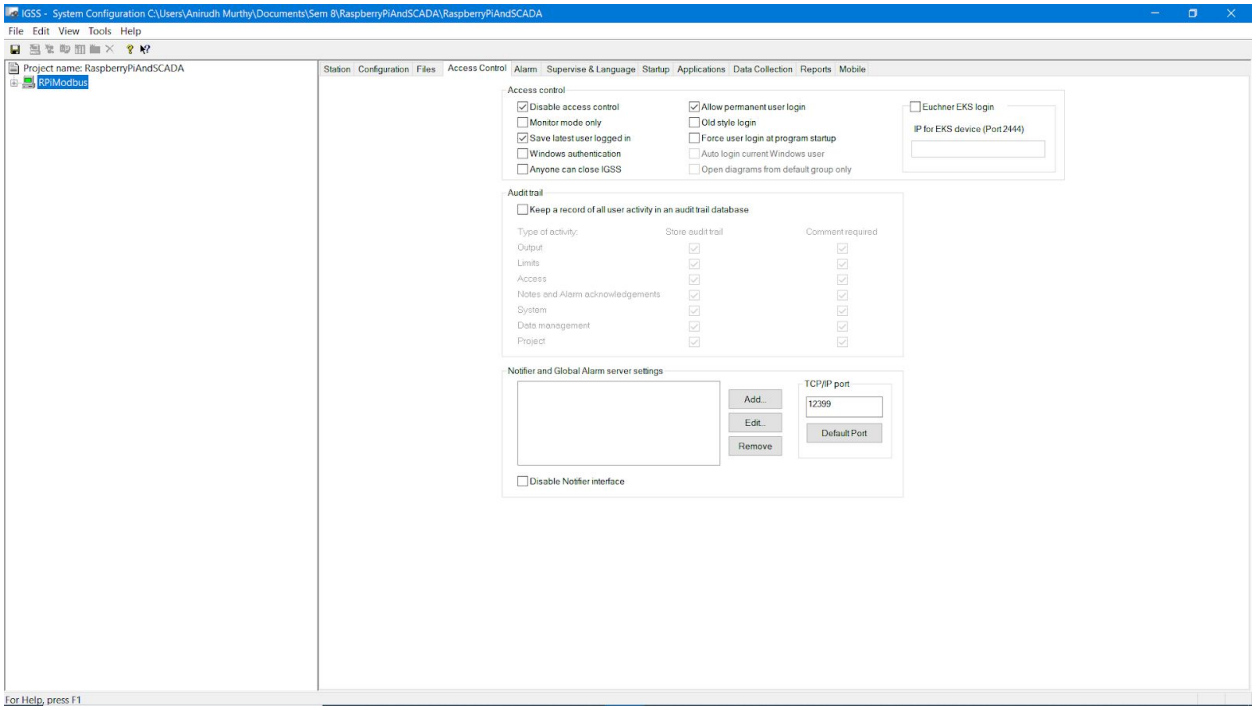
The biggest vulnerability of modbus is that it sends data in clear text, if provided access to the network, these packets can be sniffed and the contents along with the destination address can be obtained. Apart from confidentiality, the integrity can also be compromised if the slave response has address spoofing since modbus uses an non-secure master slave architecture. There is also no authentication employed in Modbus at any level.

A general way to prevent breaches is to deploy firewalls to cut off unauthorised access to the SCADA network. The confidentiality and integrity can be maintained if SSL/TLS is used for data transfer.

The general security measures will be effective, but considering the master slave architecture of modbus, it is important to secure the HMI s which often send requests to the PLC (master to slave). If the HMI system gets compromised, the whole SCADA system would be as a result, compromised.

There are security measures that can be implemented in the IGSS software to reduce vulnerability of the supervisor which here acts like the HMI.

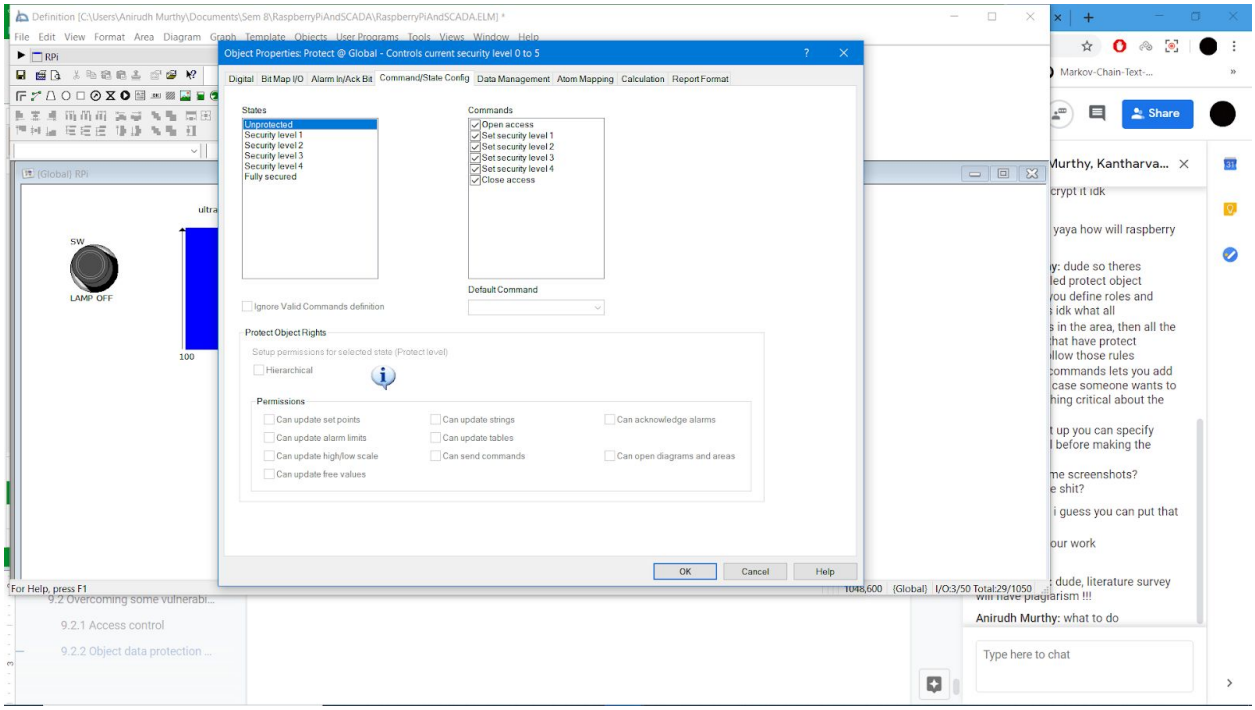
8.5.1 Access control



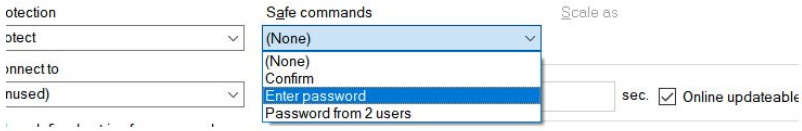
Before the SCADA supervisor is set up, Access control can be specified so that the HMI is more secure.

8.5.2 Object data protection and safe commands

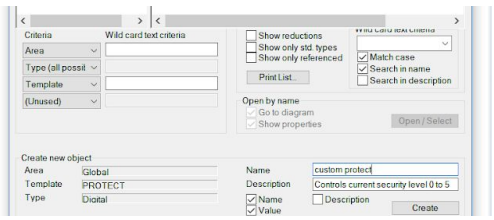
The objects in the supervisor can be protected with predefined levels of security which go up to level 5



Each object can be configured to use the protection and a level of authentication provided by the safe commands, one or 2 unique passwords



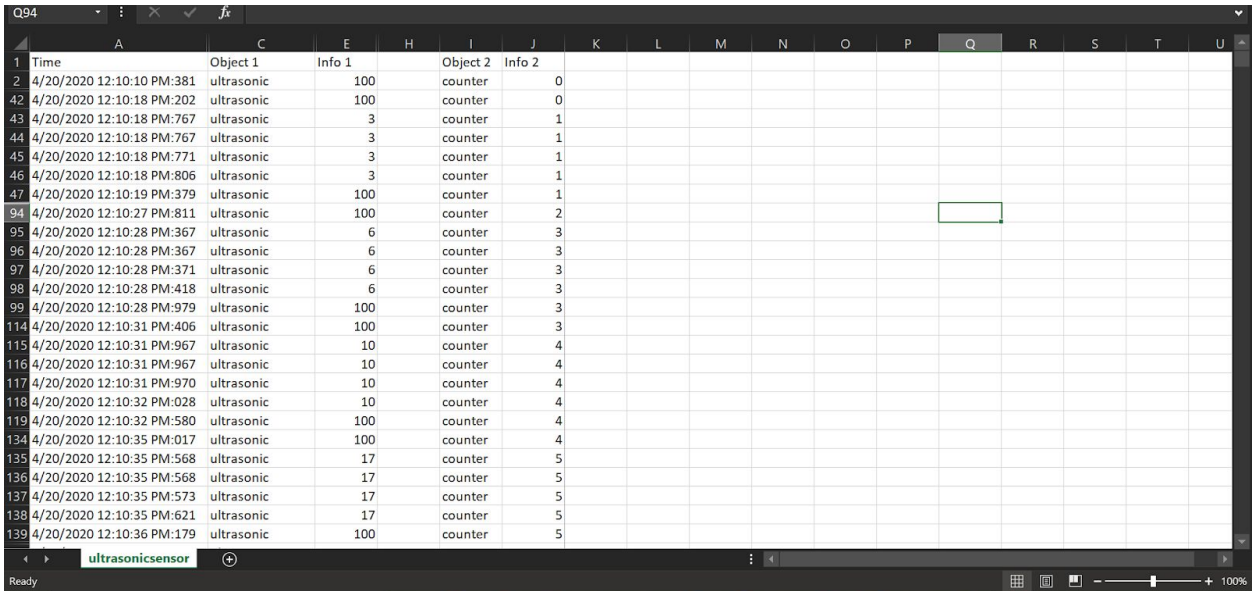
Apart from this a custom protect object can be created where different security levels can be configured and any objects representing the end devices in the supervisors can be protected by either the custom protect object or the default one.



These measures can secure the HMI and also which is very crucial to the safety of the IIOT system.

8.6 Log file and Graphical representation

All these Data are constantly being monitored in the IGSS setup. Which can be converted to a CSV Log file



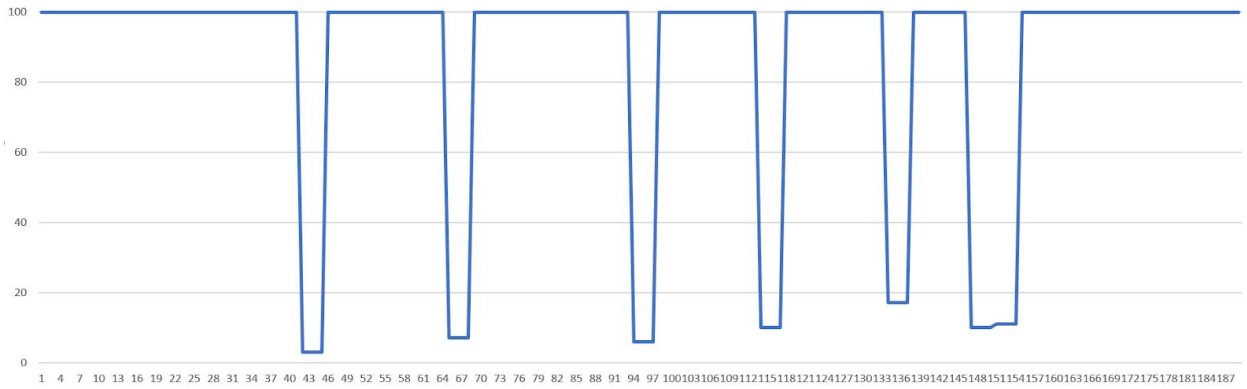
	Time	Object 1	Info 1	Object 2	Info 2
2	4/20/2020 12:10:10 PM:381	ultrasonic	100	counter	0
42	4/20/2020 12:10:18 PM:202	ultrasonic	100	counter	0
43	4/20/2020 12:10:18 PM:767	ultrasonic	3	counter	1
44	4/20/2020 12:10:18 PM:767	ultrasonic	3	counter	1
45	4/20/2020 12:10:18 PM:771	ultrasonic	3	counter	1
46	4/20/2020 12:10:18 PM:806	ultrasonic	3	counter	1
47	4/20/2020 12:10:19 PM:379	ultrasonic	100	counter	1
94	4/20/2020 12:10:27 PM:811	ultrasonic	100	counter	2
95	4/20/2020 12:10:28 PM:367	ultrasonic	6	counter	3
96	4/20/2020 12:10:28 PM:367	ultrasonic	6	counter	3
97	4/20/2020 12:10:28 PM:371	ultrasonic	6	counter	3
98	4/20/2020 12:10:28 PM:418	ultrasonic	6	counter	3
99	4/20/2020 12:10:28 PM:979	ultrasonic	100	counter	3
114	4/20/2020 12:10:31 PM:406	ultrasonic	100	counter	3
115	4/20/2020 12:10:31 PM:967	ultrasonic	10	counter	4
116	4/20/2020 12:10:31 PM:967	ultrasonic	10	counter	4
117	4/20/2020 12:10:31 PM:970	ultrasonic	10	counter	4
118	4/20/2020 12:10:32 PM:028	ultrasonic	10	counter	4
119	4/20/2020 12:10:32 PM:580	ultrasonic	100	counter	4
134	4/20/2020 12:10:35 PM:017	ultrasonic	100	counter	4
135	4/20/2020 12:10:35 PM:568	ultrasonic	17	counter	5
136	4/20/2020 12:10:35 PM:568	ultrasonic	17	counter	5
137	4/20/2020 12:10:35 PM:573	ultrasonic	17	counter	5
138	4/20/2020 12:10:35 PM:621	ultrasonic	17	counter	5
139	4/20/2020 12:10:36 PM:179	ultrasonic	100	counter	5

(Figure 6 : CSV Log file)

With timestamp, first object ultrasonic sensor and its values, second object counter and its values.

Timestamp can be specified to get a particular period of data to be noted in a csv file. With access to log checked attributes whenever required.

Graph : Time-series graph depicting Ultrasonic sensor detecting objects as a series of drops in the graph as an when an object passes through the sensor.



- X-axis : Each entry in the log file
- Y-axis : Ultrasonic value ranging from 0 to 100 centimeters (cm).

Analysis of the graph :

When there is no obstruction sensor value shows the maximum value which is 100 cm.
Each dip in the value(less than 100 cm) represents the obstruction of an object across the ultrasonic sensor

When Data would be generated not manually as in this case, rather be generated by constant movement of any conveyor belt, Analysis over the graph could be done.

- If a nonuniform graph of values shows up, we could stop the conveyor belt to check any missing product.
- By using two or more sensors, a physical defective product could be easily traced.
- Analysis on the graph can result in any anomalies which further can be resolved by if its a physical conveyor belt problem or any breach in security of the data.

CHAPTER-9

CONCLUSION

Digital Data from IGSS is sent over the Modbus TCP over the local area network to the Raspberry Pi which toggles the LED on and off. Similarly Analog data from Ultrasonic sensor is being sent to IGSS with the help of pymodbus python module over Modbus TCP protocol. And a counter to count the objects being passed across the sensor. From this Data, consistency and continuity can be analysed to check for any missing or corrupted data. Modbus TCP protocol does not provide confidentiality, authentication, or integrity. Because it's visible that all Modbus traffic is communicated in clear text there is no sufficient security check mechanism during the traffic, but a constant analysis on the data received can prevent any hazard, using basic authentication and security checks.

REFERENCES

1. Maede Zolanvari, Graduate Student Member, IEEE, Marcio A. Teixeira, Senior Member, IEEE, Lav Gupta, Senior Member, IEEE, Khaled M. Khan, Member, IEEE, and Raj Jain, Life Fellow, IEEE. "Machine Learning - Based Network Vulnerability Analysis of Industrial Internet of Things". *IEEE Internet Of Things Journal, Vol. 6, No. 4, August 2019*

2. David Hasselquist, Abhimanyu Rawat, Andrei Gurtov, ADIT-IDA, "Trends and Detection Avoidance of Internet - Connected Industrial Control Systems", *Linköping University, Linköping, Sweden Citation information: DOI 10.1109/ACCESS.2019.2948793*

3. Mohamad Kaouk, Jean-Marie Flaus, Marie-Laure Potet, Roland Groz, "A Review of Intrusion Detection Systems for Industrial Control Systems", *Univ. Grenoble Alpes, France 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT'19) | Paris, France / April 23-26, 2019*

4. Patrick Strauß, Markus Schmitz Innovation, Digitalization, Data Analytics BMW Group Munich, Germany, René Wöstmann, Jochen Deuse Institute of Production Systems (IPS) Technical University of Dortmund Dortmund, Germany, "Enabling of Predictive Maintenance in the Brownfield through Low-Cost Sensors, an IIoT-Architecture and Machine Learning", *2018 IEEE International Conference on Big Data (Big Data)*

5. Roman Zhohov, Dimitar Minovski, Per Johansson, Karl Andersson Lule˚a University of Technology InfoVista Sweden AB Skellefte˚a, Sweden, "Real-time Performance Evaluation of LTE for IIoT", *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*