



FANSHAWE COLLEGE

Electronics and Embedded Systems Development ELNC-6008 Practical Project Report

Solar Power Monitoring System

by

Student Names	Student Numbers
Abdul Baasit Abdul Baasit	0945195
Helly Mohod	0925351
Jaswanth Mandava	0925562
Leona Michelle Pinto	0943621
Manoj Reddy Bolledla	0925124
Meera Unni	0925104
Nikhil James	0943607
Sanoop Thattampakil	0925732

Faculty Advisor:	Prof. Brent Matthews
Date Submitted:	December 6, 2019

Executive Summary

The team evaluated two methods for publishing data from PLCs, installed for solar panels in Fanshawe College by German Solar Corporation, on to an HMI display. The first method is, configuring a webserver on Red Lion HMI device connected to PLCs which monitors the solar panels. This is an efficient way of monitoring because it can be monitored remotely using the internet. The alternative method is to retrieve data from PLCs using the Modbus protocol over ethernet. Although using webserver is an efficient way, it requires additional input to PLCs. Whereas using the Modbus protocol is pre-installed and easy to use. The data retrieved by the Raspberry Pi will be published on a webpage with user understandable metrics.

Table of Contents

Executive Summary	i
Acknowledgments	iv
Introduction	1
Project Scope	2
Background	3
Purpose	4
Proposed Solution	5
Operational Overview	6
Technical Problem	7
Results	9
Analysis of Results	11
Assessment of Results	12
Conclusion	17
Recommendations	18
References	19
Declaration of Originality	20
Appendix I	21
Appendix II	30

List of Figures

Figure 1. Solar Panels Installed on D-Building at Fanshawe College [1]	1
Figure 2. Current HMI display.	4
Figure 3. Block diagram.....	5
Figure 4. Project Overview.....	6
Figure 5. Firmware update issue.....	7
Figure 6. Modbus Connection Failure.....	8
Figure 7. Calculated metrics.....	12
Figure 8. Mapped Registers.	12
Figure 9. Modified HMI.	13
Figure 10. Overview of System.	13
Figure 11. Fetched data from registers.....	14
Figure 12. Successful connection.....	15
Figure 13. Final Layout.....	16
Figure 14. Choosing the operating system Raspbian.....	30
Figure 15 Installing Raspbian OS.....	31
Figure 16 Installing nodes on Node-RED.....	32
Figure 17 Nodes in the dashboard.....	33
Figure 18. Nodes arranged to create dynamic display	34
Figure 19. Data fetched from the registers.....	35
Figure 20. Initializing the address	36
Figure 21. Register addresses and label.....	36

List of Tables

Table 1. Team-PLC Scope Deliverables.	2
Table 2. Team-HMI Scoped Deliverables.	2

Acknowledgments

We take pleasure in acknowledging the contributions of many individuals who have helped us throughout the project.

We are extremely thankful to Prof. Brent Matthews, our Project Manager, for the guidance and valuable suggestions provided during the project work. We express our sincere gratitude to Prof. Martin Volkening for his technical support and constant supervision which contributed immensely to the development of the project. We are also thankful to Prof. Chris Talbot for providing us the best facilities for the completion of our project.

Our acknowledgment would be incomplete without mentioning the valuable guidance given to us by Prof. Abdul Majeed Mahmood (Lab instructor) and Prof. Steve Roch. We take this opportunity to express our sincere gratitude to both professors for the careful and precious guidance provided to us.

Once again, we wish to express our profound gratitude to all those whose dedicated efforts helped us accomplish the project into a reality.

Introduction

Due to the extensive use of non-renewable energy sources like fossil fuels, the problem of energy crisis is increasing day by day. Since the population is expected to increase by six-fold within 30 years, people should make a transition from the usage of non-renewable sources to renewable sources so as to make the world sustainable for upcoming generations.

This project emphasizes on spreading awareness among people about the advantages of switching to solar energy through a solar power monitoring system. The solar power monitoring system will display the amount of power being generated on a daily basis and the amount of carbon dioxide emissions being reduced. Using this Solar Power Monitoring System, real-time data can be obtained such as Voltage, Current, Temperature, Flow of Water, and Power in BTU.

The team has been sub-divided into PLC and HMI. The PLC team was tasked to make the data available to the HMI team via ethernet. The HMI team is tasked to fetch the available data and publish it onto a web page. The retrieved information will be displayed in a meaningful and simplified metrics on a web page.



Figure 1. Solar Panels Installed on D-Building at Fanshawe College [1]

Project Scope

Team-PLC	
Calculation of metrics	<input checked="" type="checkbox"/>
Configuring Registers for HMI team to fetch data	<input checked="" type="checkbox"/>
HMI display	<input checked="" type="checkbox"/>
Establish connection between PLC and inverter	<input type="checkbox"/>

Table 1. Team-PLC Scope Deliverables.

Team-HMI	
Fetch data from the registers	<input checked="" type="checkbox"/>
Establish communication between PLC and Raspberry Pi	<input checked="" type="checkbox"/>
Conversions on received data	<input checked="" type="checkbox"/>
Develop an effective display	<input checked="" type="checkbox"/>
Store data into database	<input type="checkbox"/>

Table 2. Team-HMI Scoped Deliverables.

Background

The following information provides details of the current product that has been taken from [2]

- Protocol Conversion of over 300+ Drivers
- Built-In Web Server for Remote Access
- Real-Time Data Logging to SD Card or via FTP
- Rugged Construction for Extreme Protection
- Wide Operating Temperature Range
- Outdoor Sunlight-Readable Models
- Scalable Modules and Expansion Options
- IEC 61131 control capabilities with Crimson Control

FEATURES & BENEFITS

- Sleek Full-Color HMI Touchscreens
 - ❖ 7" to 15" models with narrow tablet-style bezels
 - ❖ Outdoor sunlight-readable and widescreen models
- Versatile Module and Expansion Options
 - ❖ Choose from a mix of I/O, PID control or communication to populate up to 8 local module slots.
 - ❖ Use Crimson Control module for enhanced IEC 61131 logic control programming
 - ❖ Connect Graphite Expansion Racks to easily scale
 - ❖ Extend even further with E3 I/OTM high-density modules
- Rugged Environmental Specifications
 - ❖ Wide -20° to 60°C HMI operating temperature
 - ❖ High shock and vibration tolerance
 - ❖ CE, UL/cUL, UL/cUL Hazardous, ATEX, IECEx, and ABS approvals
- Industry-Leading Protocol Conversion
 - ❖ Communicate with over 300 industrial protocols
 - ❖ Support up to 20 simultaneous protocols
 - ❖ Convert between serial, USB and Ethernet devices
 - ❖ Manage multi-vendor environments with ease
- Powerful Integration Functionality
 - ❖ Intuitive Crimson 3.0 software for easy drag-and-drop configuration
 - ❖ Ethernet, USB and serial ports make communication simple
 - ❖ Built-in data logging enhances troubleshooting and helps meet regulatory requirements
 - ❖ Robust web server provides remote access and control to reduce costly site visits

Purpose

The crimson software offers very few options for displaying the information in a non-technical method. As shown in figure 2, the current HMI displays the characteristics read from plc connected sensors. Mathematical calculations can be performed using crimson to display metrics such as power in Kilo-Watt hours (KWh) and British Thermal Units(BTU). Hence, this information is too complex for the audience who does not have a technical background.



Figure 2.Current HMI display.

Proposed Solution

The following block diagram illustrates an overview of the proposed solution. The plc is configured using red Lion's crimson software as a slave. Post configuration of the plc as a slave using Modbus protocol, HMI team will configure the raspberry module as master and will fetch data over ethernet connection. The fetched data is then displayed as a webpage in the form of gauges and graphs.

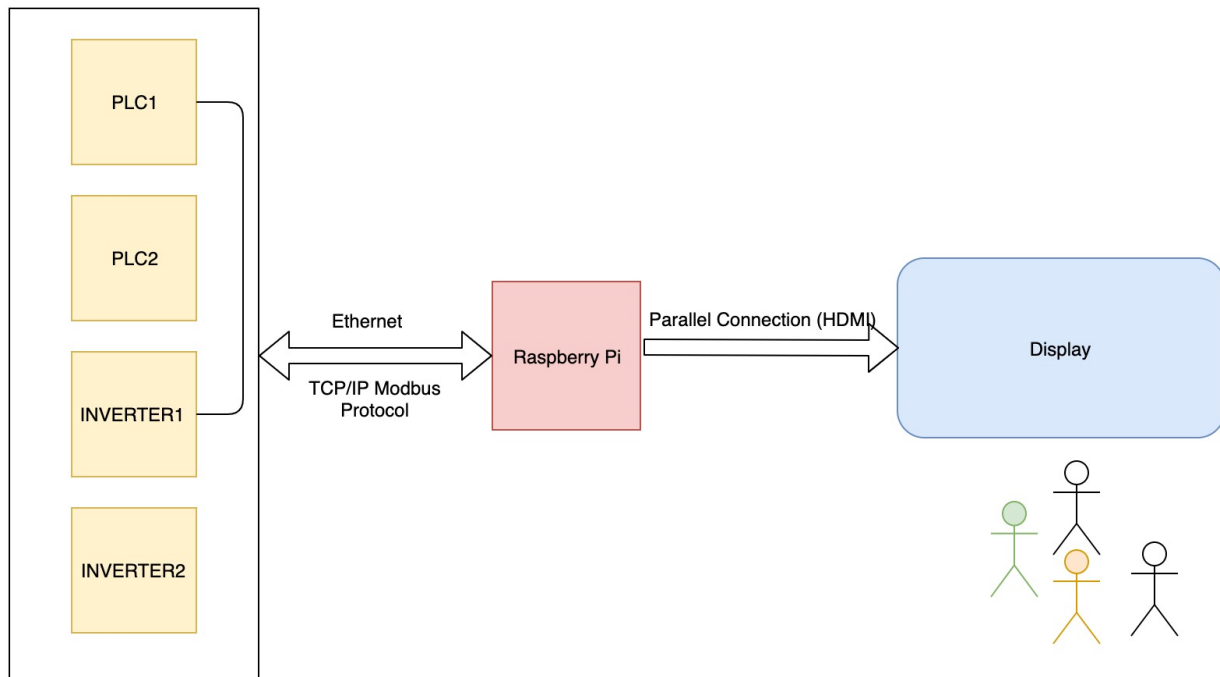


Figure 3. Block diagram

Operational Overview

To establish reliable communication between client(PLC) and user(HMI-team), we recommend using the Modbus protocol. Modbus is a master-slave communication model. Any device on a network can be configured as either a master or a slave. Any PLC connected to a Red Lion HMI device can be set as a slave through Red Lion Crimson 3.1 software. Once the above-mentioned configuration has been set up, the Modbus protocol will allow the master(HMI-Team) to fetch data from the slave (PLCs) over ethernet. The data linked to the registers on PLC can be fetched and updated in real-time through this proposed solution. The following flowchart provides in-depth information about how exactly the proposed solution works

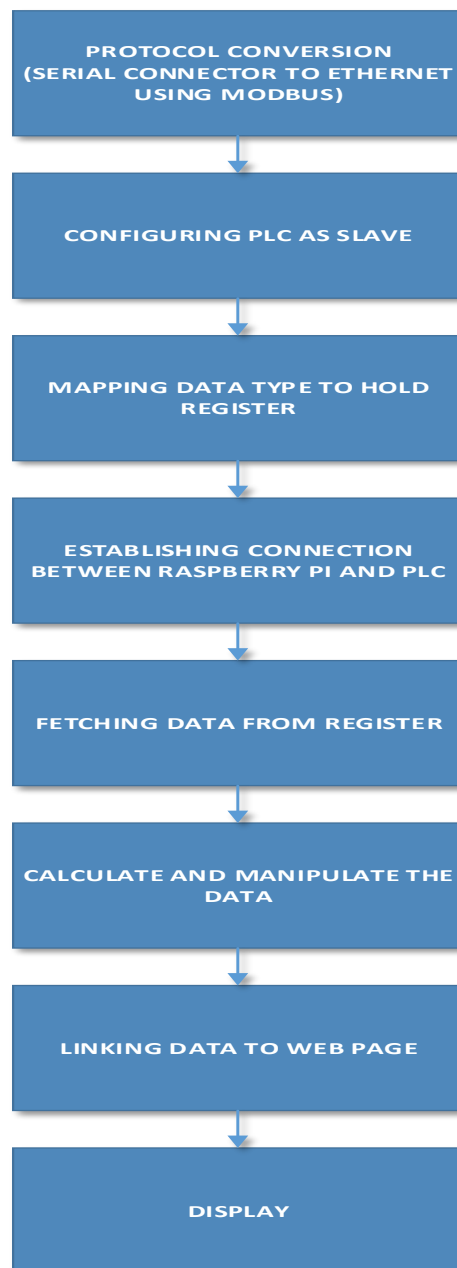


Figure 4. Project Overview

Technical Problem

Technical Problems encountered by Team-PLC

Problem: Firmware Update

Crimson software throws an exception whenever the updated file is pushed onto PLC through webserver.

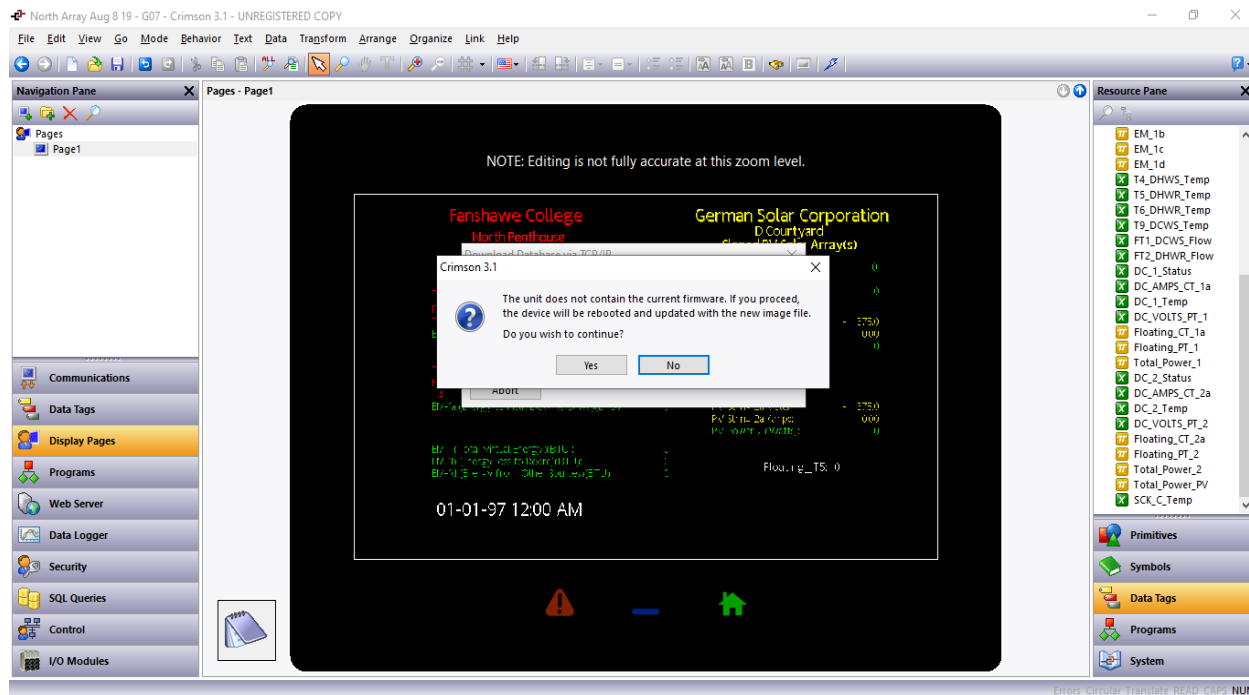


Figure 5. Firmware update issue.

Solution: Every time a device is used to push updates onto PLC through webserver the firmware should be updated.

Problem: Set-up a successful communication between PLCs and Inverters.

To establish a communication the PLC was to be configured as slave and inverter as the master. Information from inverters was already being retrieved by another host configured as master causing team-plc to fail in establishing a reliable connection.

Solution: A slave can respond to only one master. So, the information from the inverter should not be retrieved in parallel.

Technical problems encountered by Team- HMI

Problem: Memory allocation of Raspberry Pi.

The team used a 32GB SD card at first and thus the cache was full. As a result, the Raspberry Pi displayed a blank screen while powered up.

Solution: Due to the low memory of Raspberry Pi, the team started the node with an additional argument to free up the cache memory section. To do this, node-red-pi command is used to clear the max-old-space-size argument.

Problem: Establishing communication between PLC and Raspberry Pi

To fetch the data from the registers of the PLC to Raspberry Pi, the pi was configured as the master and PLC as a slave using a protocol called Modbus. The application used was Modbus PY. Due to improper configuration, Modbus PY was not able to connect to the registers.

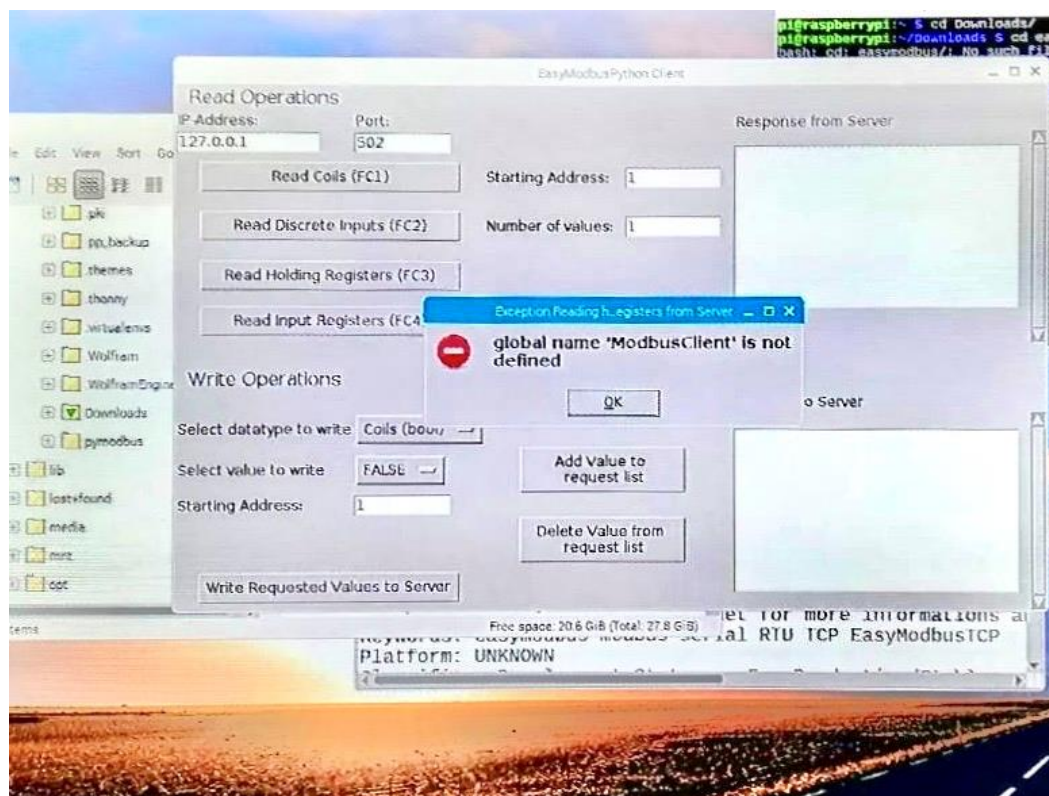


Figure 6. Modbus Connection Failure.

Solution: The protocol was changed into Modbus TCP and as a result, a reliable connection was established between Raspberry Pi and PLC.

Results

Abdul Baasit:

- Creating data tags
- Customizing Crimson HMI
- Testing Configuration of Modbus register
- Configuring PLC as a slave.

Manoj Reddy:

- Configuring Modbus registers
- Tagging data tags to Modbus registers
- Formatting the data tags

Helly Mohod:

- Installing Crimson software
- Reading and writing data logs
- Research tools for testing
- Documentation

Jaswanth Mandava:

- Protocol Identification
- Calculations of metrics
- Documentation

Leona Michelle Pinto :

- Identifying Protocol
- Modbus TCP/Ip Package
- Fetching Data from PLC registers
- Dashboard Layout

Meera Unni :

- Calculation of Metrics
- Installation of Raspbian-Os
- Documentation

Nikhil James :

- Development of Node-red Flow
- Flow Testing
- Dashboard Layout

Sanoop Thattampakil :

- Node-red Installation
- Functionality of Nodes
- Documentation

Analysis of Results

The final result of this project will help in knowing Fanshawe students, facility and other staff members how Fanshawe is contributing to the environment. Even though the solar panels are installed no one is aware of the benefits being utilized from it daily. The power being generated is used to heat up the water required by different facilities of Fanshawe. The result of the HMI display will bring awareness among Fanshawe students about the environment. It motivates being a part of Fanshawe college how they are indirectly contributing to the environment in the form number of oil barrels saved or reduction of carbon dioxide emissions.

Assessment of Results.

Team-PLC

Scoped:

1) Calculations of Metrics

Achieved:

```
Power_Array1_Watts:      2.013
Power_Array2_Watts:      1.997
Power_Array1_BTU:        6864.33
Power_Array2_BTU:        6809.77
Total_Power_Watts:       40.10
Total_Power_BTU:         13674.10
```

Figure 7.Calculated metrics.

2) Configuring Registers for HMI team to fetch data

Achieved:

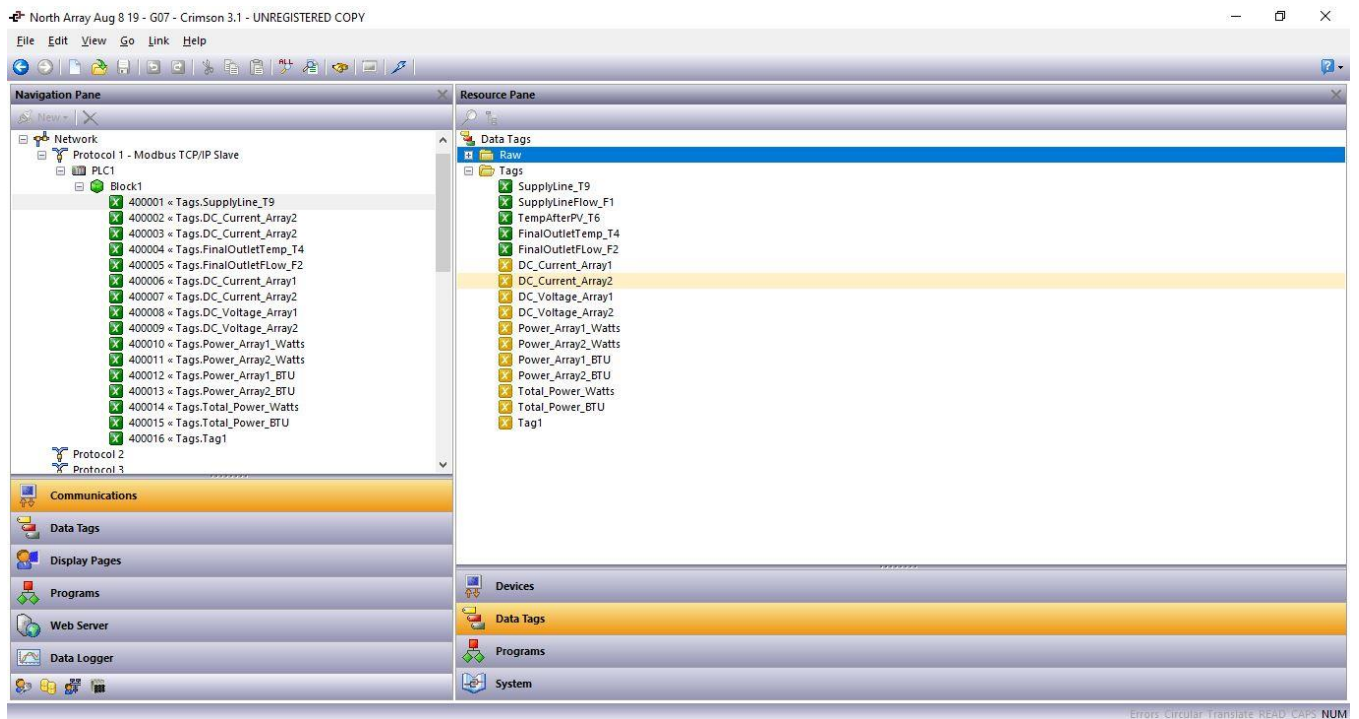


Figure 8.Mapped Registers.

3) HMI display

Achieved:



Figure 9.Modified HMI.

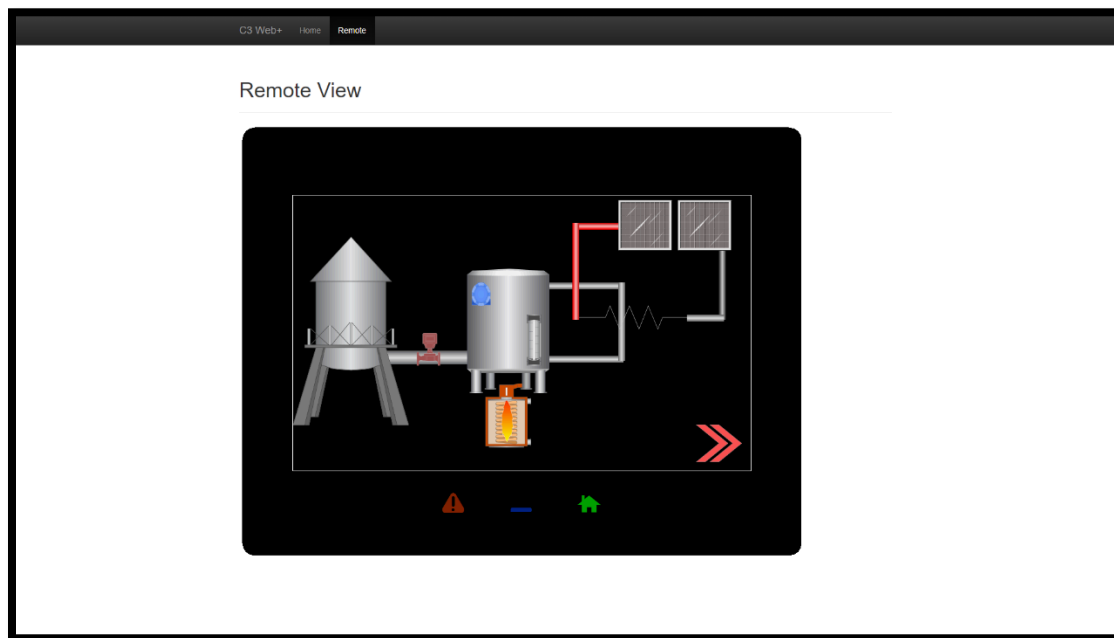


Figure 10.Overview of System.

TEAM-HMI

Scope:

1)Fetch data from the registers

Achieved:

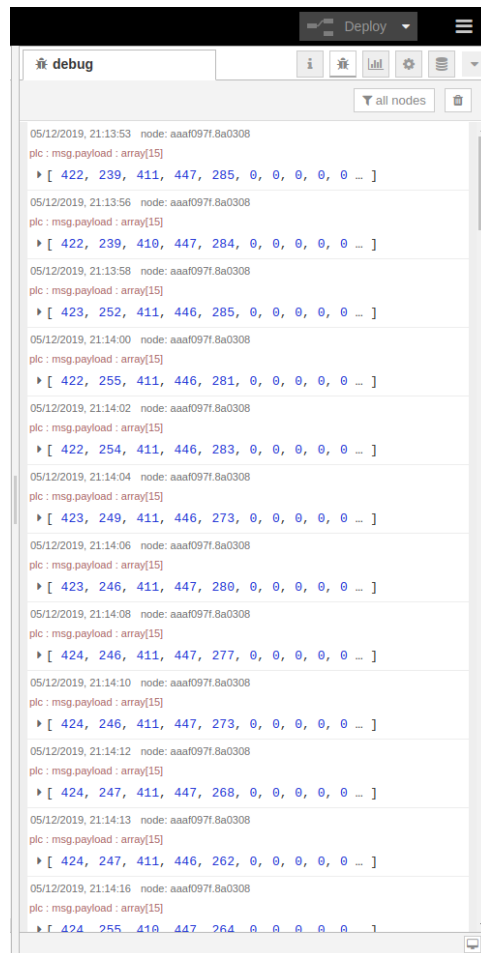


Figure 11.Fetch data from registers.

2) Establish communication between PLC and Raspberry Pi

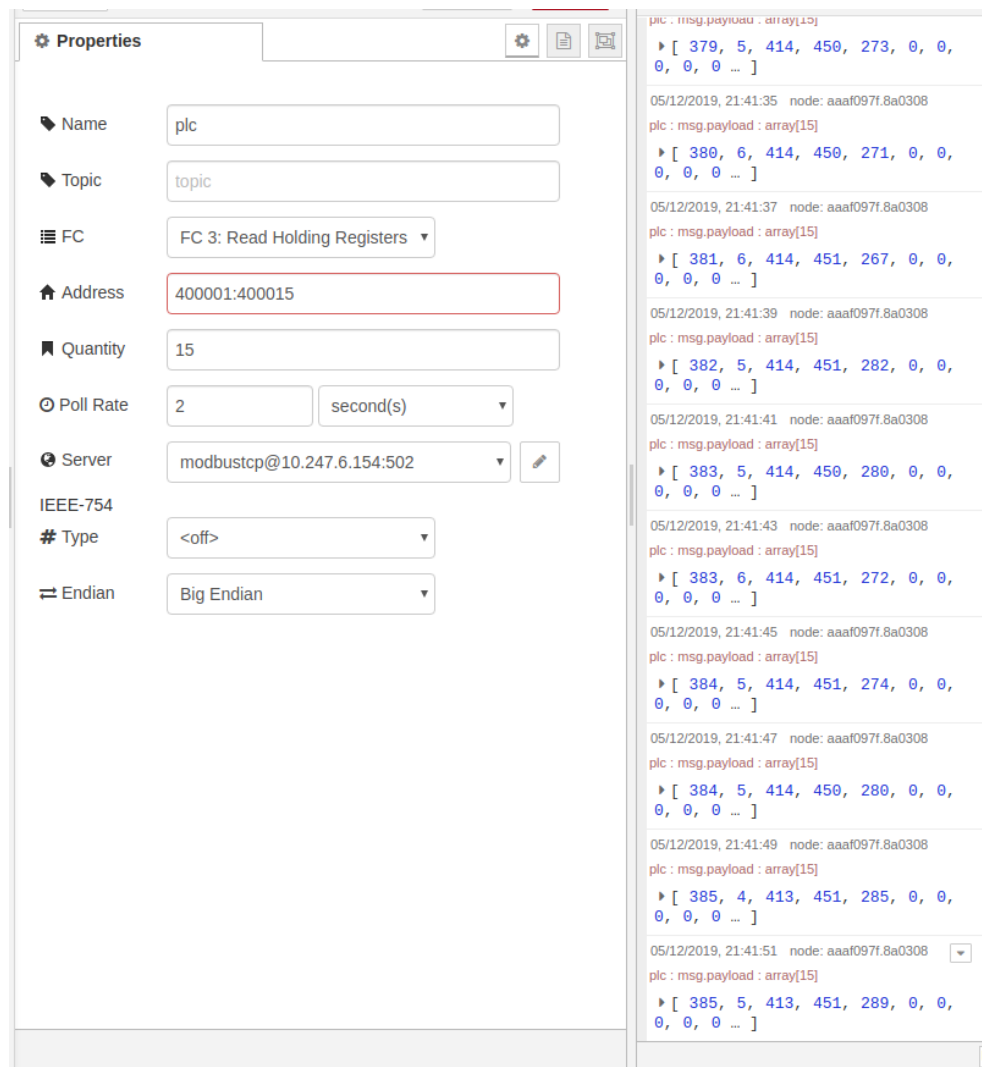


Figure 12.Successful connection.

3) Conversions on received data and Develop an effective display

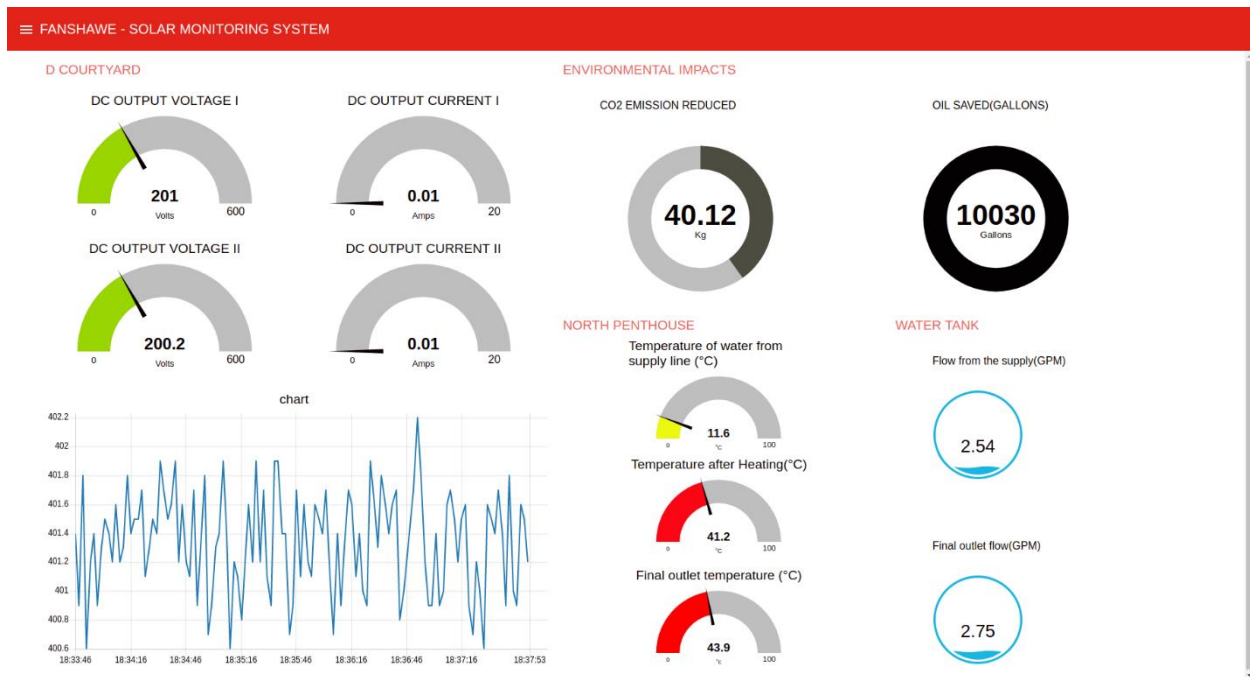


Figure 13.Final Layout.

Conclusion

This report mainly focuses on analyzing the amount of solar power generated based on real-time. The Modbus protocol provides several advantages for the proposed Fanshawe college. This makes the user more customizable. The data can be easily accessed that is stored on the devices. Using Modbus protocol reduces maintenance. The Modbus protocol is simple to configure and retrieve data as it is pre-installed in devices.

The main technical problem faced by the team is to find a method to display the power generated. The graphical representation using charts and dynamic images were chosen, as it is understandable for the non-technical audience. The display shows the amount of power generated. This helps in portraying the importance of renewable energy sources and also provides the metric of non-renewable resources being saved. In addition, the system also calculates the amount of CO₂ emission is reduced. Displaying this data on a simplistic webpage makes the monitoring system divergent from the existing monitoring system.

Recommendations








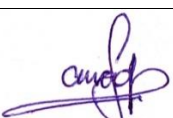
We recommend establishing a connection between PLC and inverter which will allow the raspberry pi to fetch all the information from PLC alone. By doing so the complexity in the communication being established can be reduced to a great extent.

References

- [1] Fanshawe College, "Solar Panels Installed at Fanshawe College," 25 October 2019. [Online]. Available: <https://www.fanshawec.ca/about-fanshawe/news/harnessing-solar-energy-reduce-emissions-campus>. [Accessed 12 November 2019].
- [2] FANSHAWE, "Google Drive," [Online]. Available: <https://drive.google.com/drive/folders/18BdVFwTsc924iuLgbfkUhkY0A5U4pH9t>.
- [3] "raspberrypi.org," [Online]. Available: <https://www.raspberrypi.org/documentation/installation/noobs.md>.
- [4] "nodered.org," [Online]. Available: <https://nodered.org/docs/getting-started/raspberrypi>.

Declaration of Originality

We, the undersigned, do hereby declare that this report is entirely original and was authored by the below. All included images, objects, tables, and/or other figures are entirely original and/or where/when appropriate, all citations and/or references have been suitably detailed.

Student Name	Signature	Date (DD MMM YYYY)
Abdul Baasit Abdul Baasit		06 DEC 2019
Helly Mohod		06 DEC 2019
Jaswanth Mandava		06 DEC 2019
Leona Michelle Pinto		06 DEC 2019
Manoj Reddy Bolledla		06 DEC 2019
Meera Unni		06 DEC 2019
Nikhil James		06 DEC 2019
Sanoop Thattampakil		06 DEC 2019

Appendix I

The PLC team worked on the calculation of metrics, configuration of registers for HMI team, HMI (Human Machine Interfaces) display and successful communication between PLCs and inverters.

Calculation of Metrics: The calculation of metrics includes different factors of solar panels that need to be fetched and modify as per the requirements. For instance, the power that has been fetched and calculated was in Kwh (Kilowatt-hour), this is later modified to BTU (British Thermal Unit). Other factors are Voltage, Current, Flow of Water, Temperature of In-Water and Temperature of Out-Water.

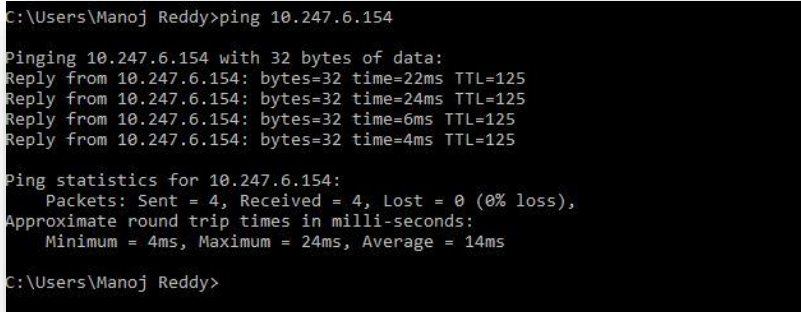
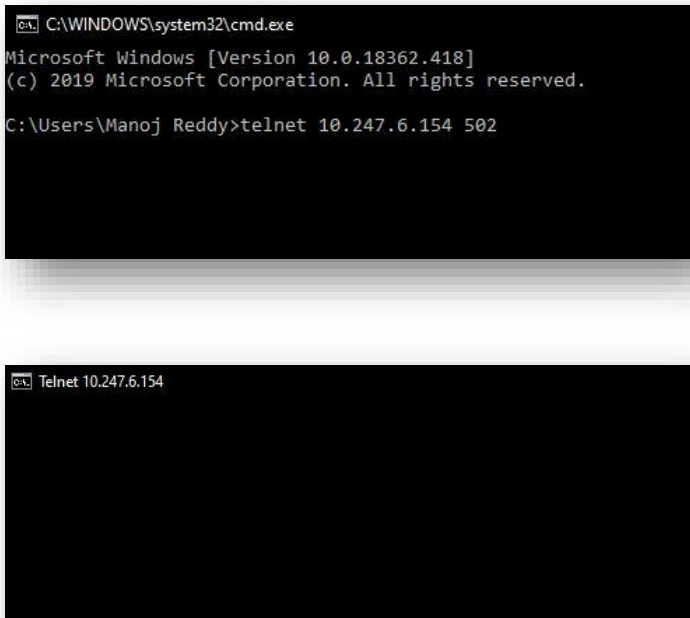
Configuration of Registers for HMI Team: Once we finish the calculation and conversions as required to be displayed, we move to configuration. In the configuration of registers, the PLC team has created registers and registers data lists that can be used by the HMI team to fetch the data that will be displayed. The registers that are created are locally available on the device and is also been attached to the data tags on crimson. The registers data list is created on the basis of the data addresses of the particular register and the addresses attached to the crimson under Data Tags of that register.

HMI (Human Machine Interfaces) Display: The display that had to exist already was not accurate for some of the factors i.e. Power. This has been evaluated and corrected by the PLC team. In accordance with the calculated and corrected values, the HMI display has been modified. This display is connected to the PLCs using Ethernet. The HMI display can be modified from crimson under Display Pages.

Successful communication between PLCs and Inverters: The PLC team had tried to ping the data of inverters but was not able to read any register data from inverters. This happened as it is already configured to another device, where it acts as a slave. Also, due to some limitations of crimson and limitation from the customer end i.e. Fanshawe College, the team was not able to set a successful communication between PLCs and Inverters.

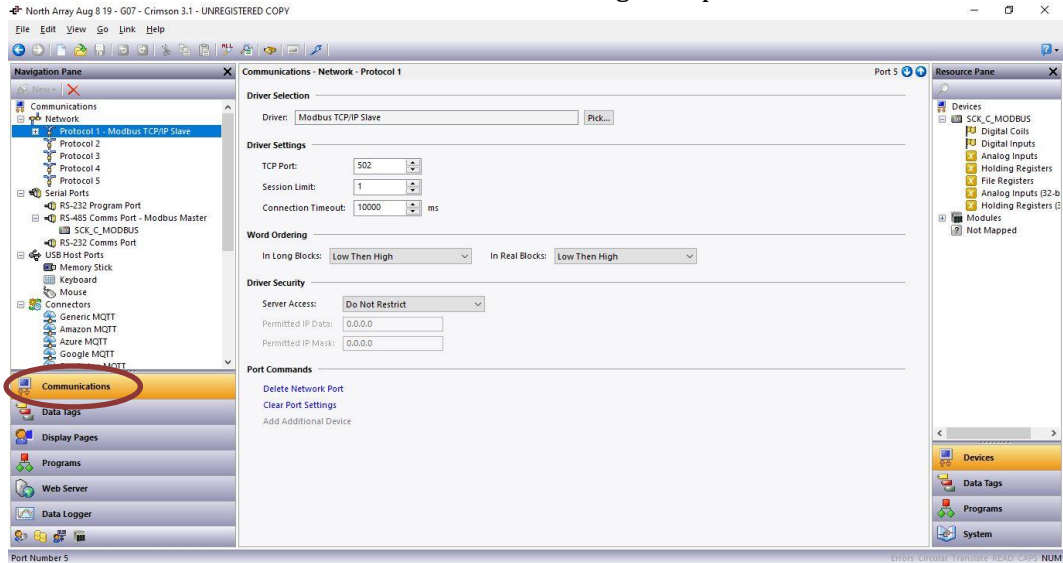
Test Code:

```
return ((Power_Array1_Watts + Power_Array2_Watts));  
/*-----*/  
int power =0;  
power = (int)(Power_Array1_Watts);  
  
return ((power*341)/100);  
/*-----*/  
return ((DC_Current_Array1 * DC_Voltage_Array1) / 100);  
/*-----*/
```

Procedure:	
Step 1:	Download and Install Crimson 3.1 freeware software from Crimson 3.1 on laptop.
Step 2:	Download and Install Modbus Poll freeware software from Modbus Poll on laptop.
Step 3:	Connect to Fanshawe network (Wi-Fi/Ethernet) ¹
Step 4:	Click on the start button and search for cmd .
Step 5:	<p>Type the command ping 10.247.6.154 press enter.²</p>  <pre> C:\Users\Manoj Reddy>ping 10.247.6.154 Pinging 10.247.6.154 with 32 bytes of data: Reply from 10.247.6.154: bytes=32 time=22ms TTL=125 Reply from 10.247.6.154: bytes=32 time=24ms TTL=125 Reply from 10.247.6.154: bytes=32 time=6ms TTL=125 Reply from 10.247.6.154: bytes=32 time=4ms TTL=125 Ping statistics for 10.247.6.154: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), Approximate round trip times in milli-seconds: Minimum = 4ms, Maximum = 24ms, Average = 14ms C:\Users\Manoj Reddy> </pre>
Step 6:	Enable the telnet feature by following the guidelines on TELNET .
Step 7:	<p>Type the command telnet 10.247.6.154 502 and press enter.³</p>  <pre> C:\WINDOWS\system32\cmd.exe Microsoft Windows [Version 10.0.18362.418] (c) 2019 Microsoft Corporation. All rights reserved. C:\Users\Manoj Reddy>telnet 10.247.6.154 502 </pre> <pre> Telnet 10.247.6.154 </pre>
Step 8:	Open North Array Aug 8 19.cd31 crimson file ⁴ .

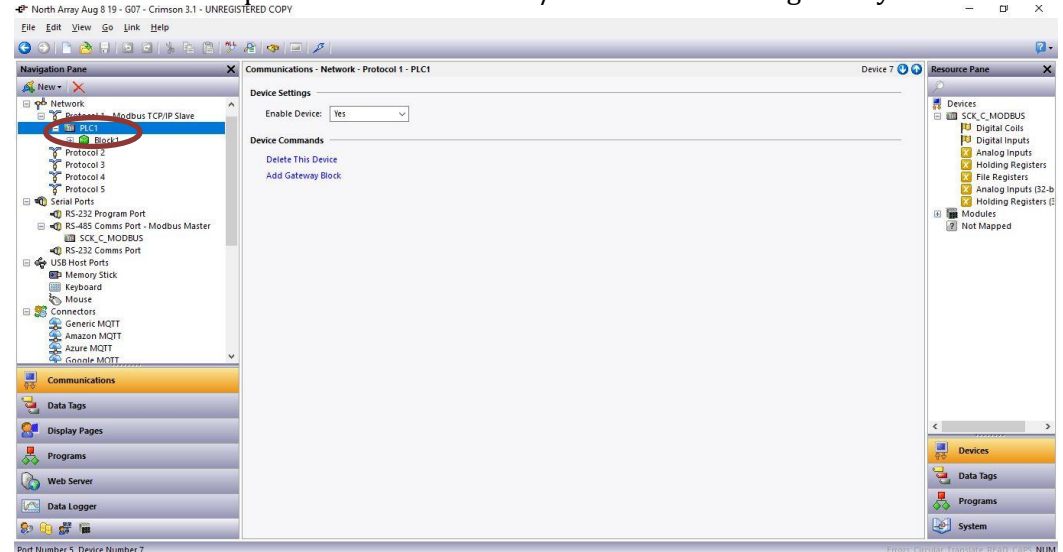
Step 9:

Click on the communications section from the navigation pane.



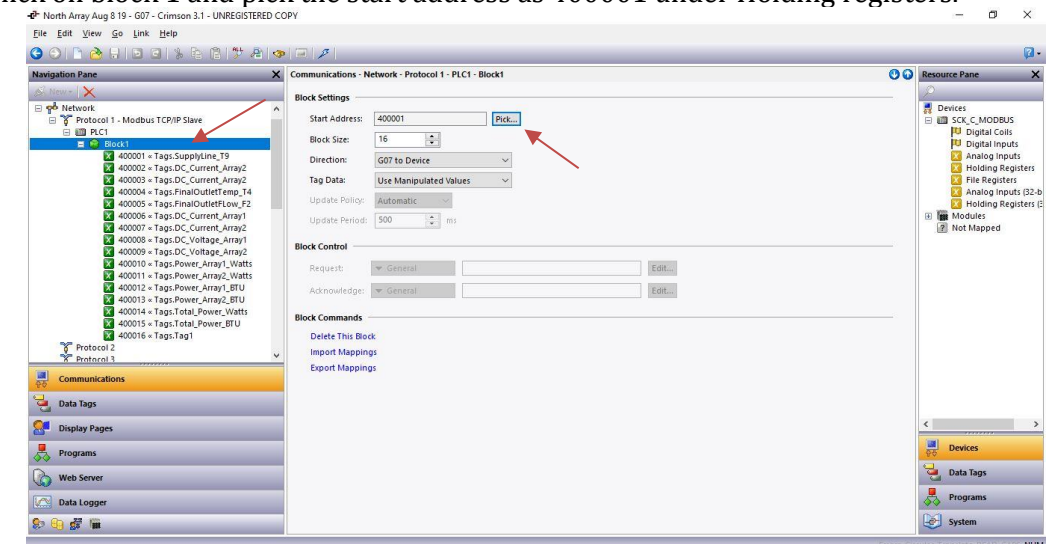
Step 10:

Click on PLC1 under protocol 1-Modbus TCP/IP Slave and add a gateway block.



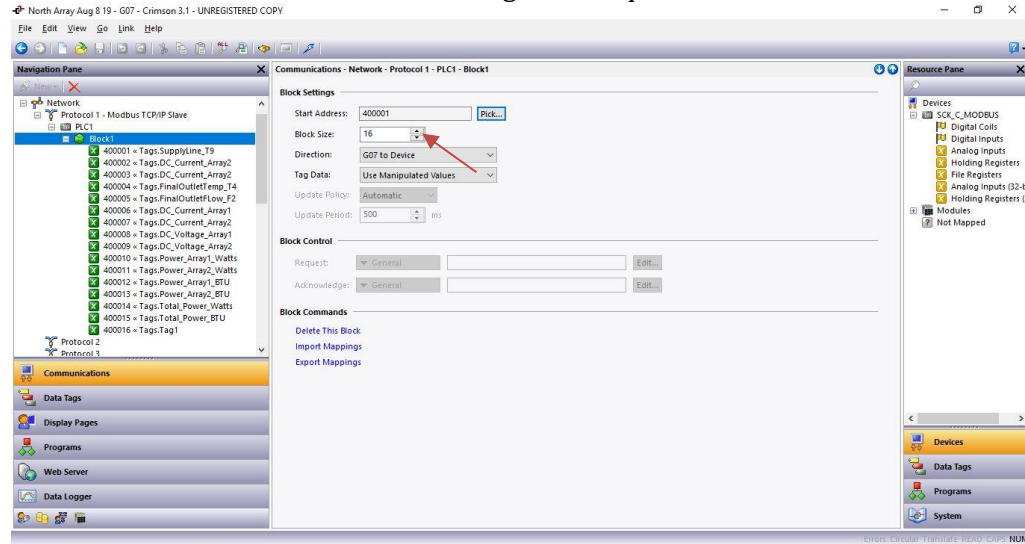
Step 11:

Click on block 1 and pick the start address as 400001 under Holding registers.

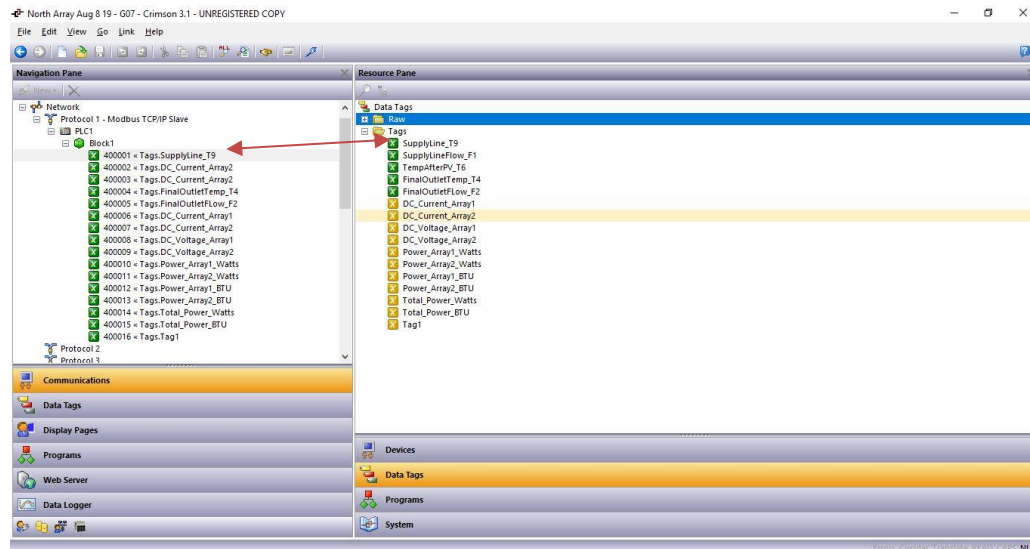


Step 12:

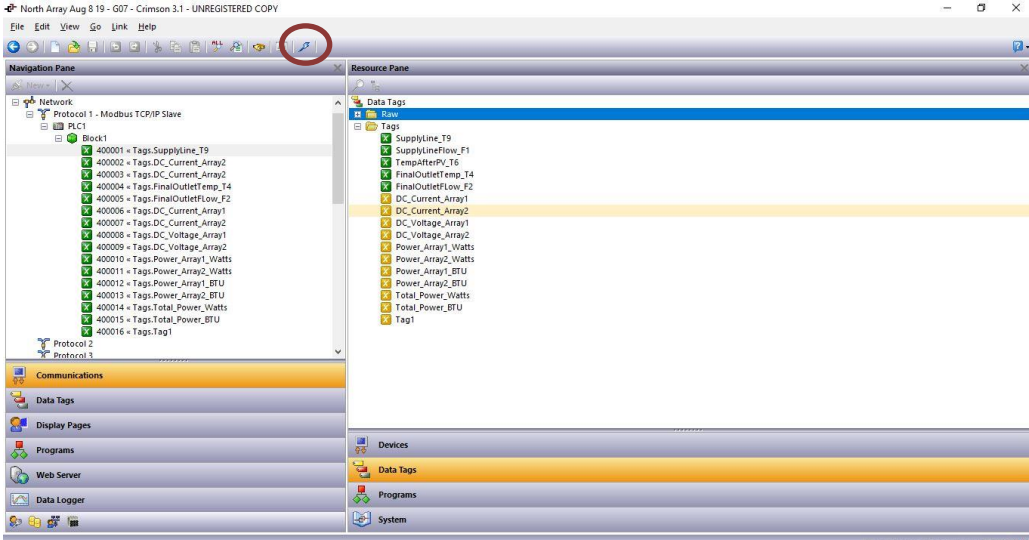
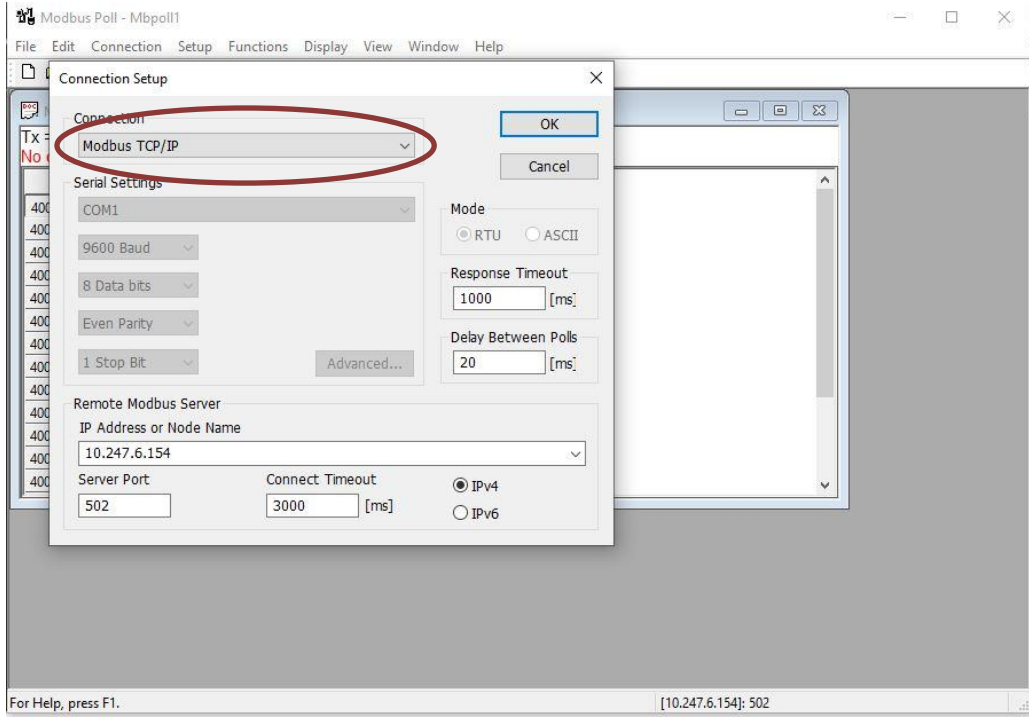
Increase the Block size to the number of registers required.

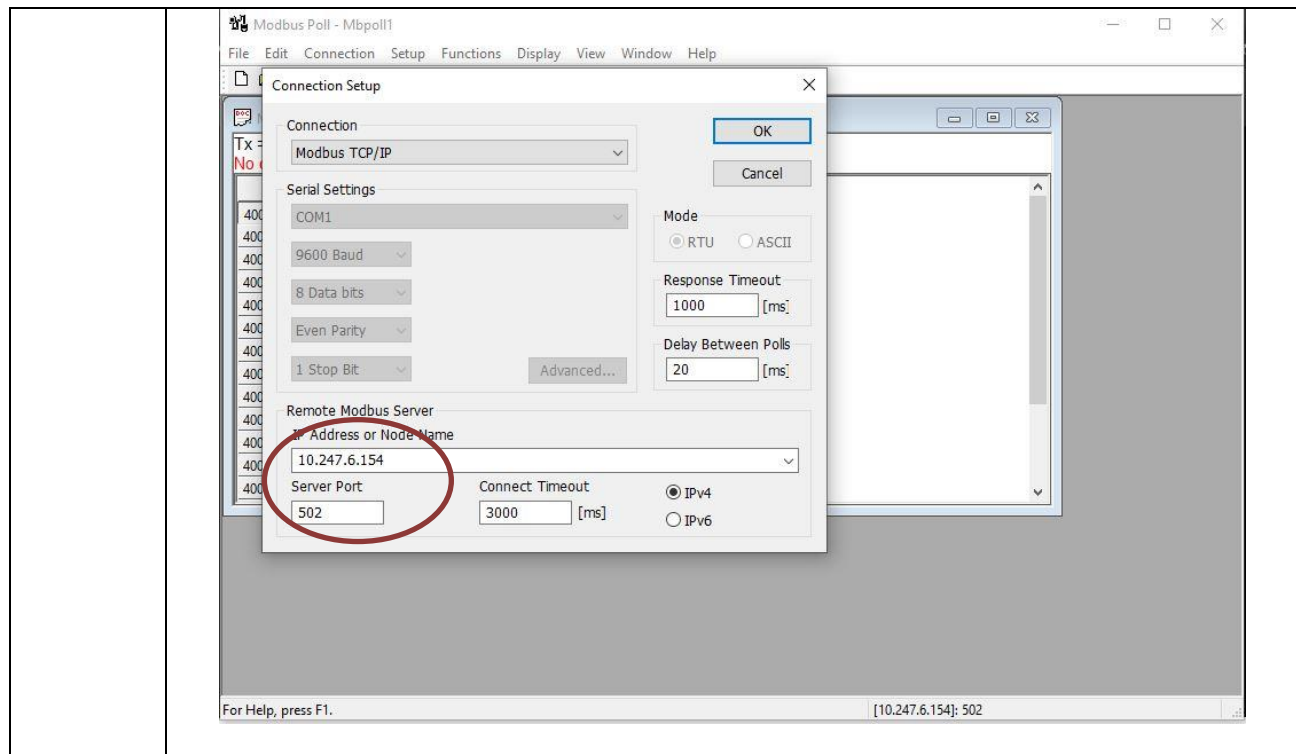
**Step 13:**

From the drop-down list under block 1 map each register to the data tags (from resource pane) that are to be read by the Modbus Master.

**Step 14:**

Click on the update (thunder symbol) from the toolbar and let the changes being done are updated onto the PLC through the webserver.

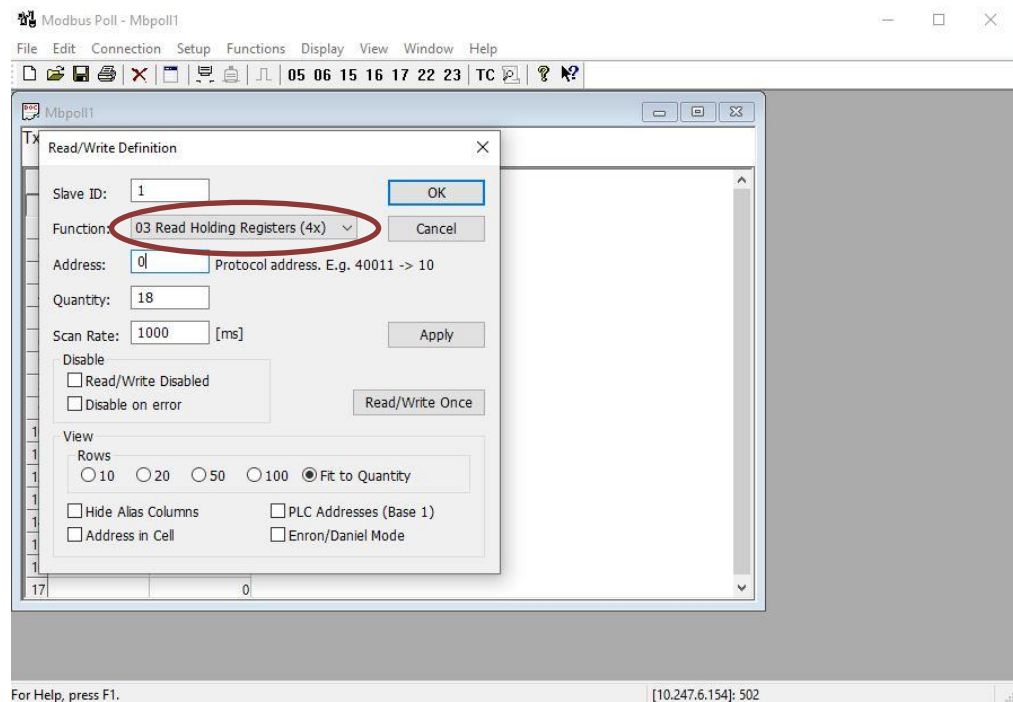
	
Step 15:	Open Modbus Poll Software.
Step 16:	<p>Connection setup window pops up, select Modbus TCP/IP from Connection drop-down list.</p> 
Step 17:	Enter 10.247.6.154 and 502 in the IP address field and server port respectively. Click Ok.

**Step 18:**

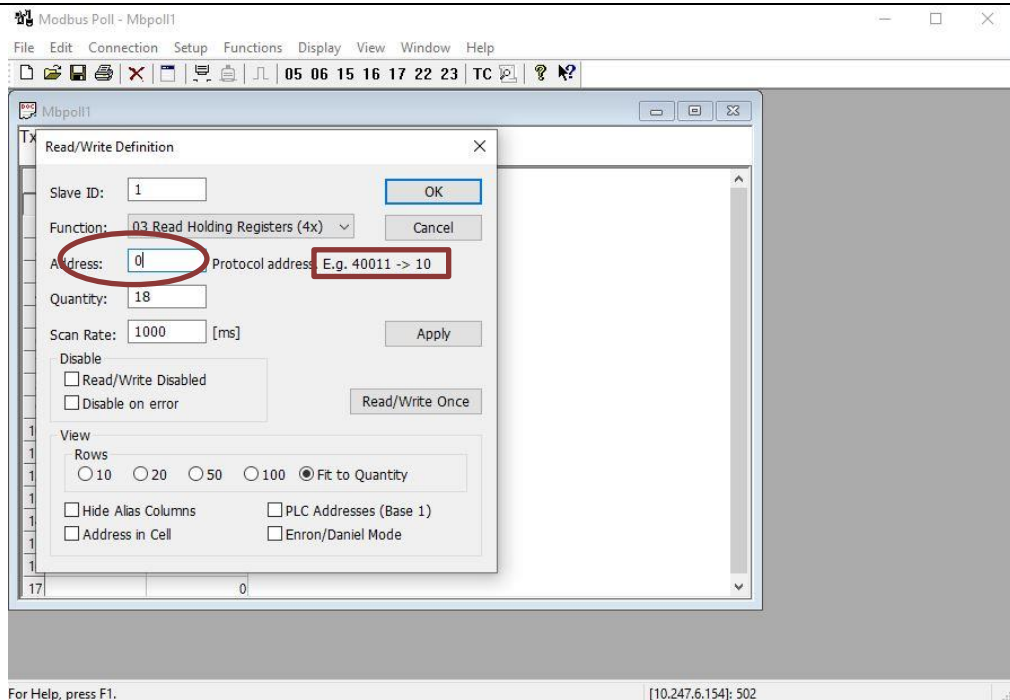
Hover the pointer to setup located in the navigation bar and click on Read/Write Definition.

Step 19:

Choose function as 03 Read Holding Registers (4x) from the drop-down list.

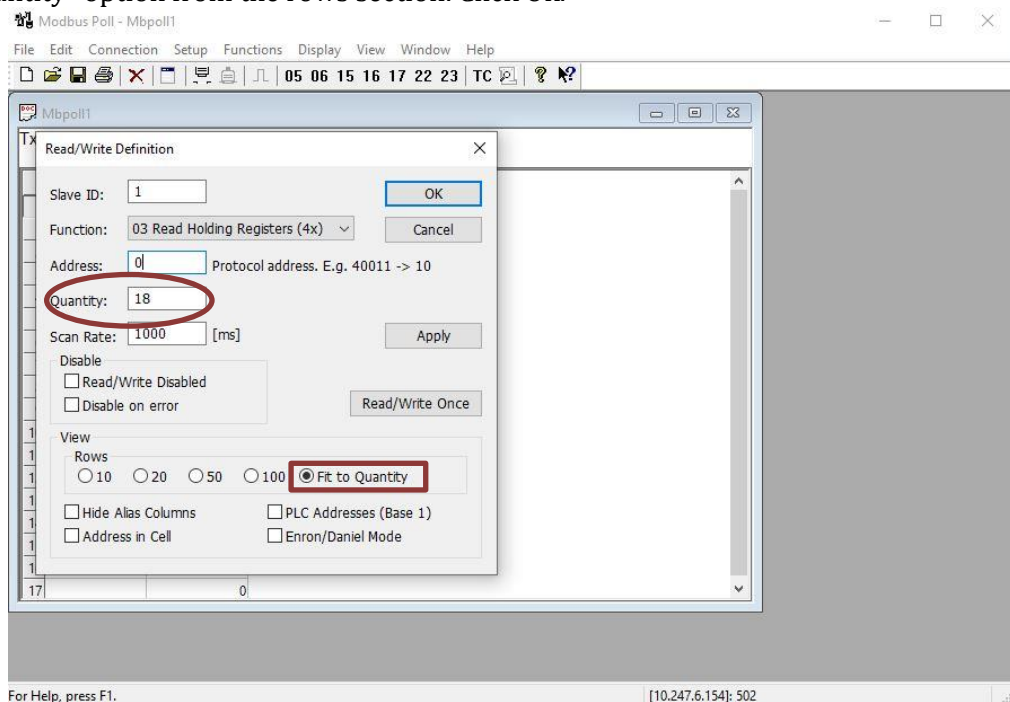
**Step 20:**

Enter the starting address in the address field⁵.




Enter the value in quantity field equivalent to the number of registers and choose “fit to quantity” option from the rows section. Click Ok.

Step 21:



Step 22:

Examine the values being displayed on Modbus Poll with values on webserver⁶.

	 <p>The screenshot shows a web browser window displaying a remote HTML page. The page is titled 'Fanshawe College' and 'German Solar Corporation'. It displays various data points related to a solar power monitoring system. The data is organized into two columns: 'Fanshawe College' and 'German Solar Corporation'. The 'Fanshawe College' column lists 'North Penthouse Domestic Hot Water Storage System' and 'SupplyLine-T9: 30.1'. The 'German Solar Corporation' column lists 'D Courtyard Sloped PV Solar Array(s)' and various DC current, voltage, and power values. A Modbus Poll application window is also visible on the right, showing a table of data points with columns for 'Alias' and 'Value'.</p>
Note 1 :	In order to access PLC through a web server, the laptop should be on the same network as PLC.
Note 2:	Upon successful connection, the plc will respond with data packets.
Note 3:	If the port 502 is open, you will notice the navigation bar will change to telnet 10.247.6.154
Note 4:	The name of the crimson file should be the same as mentioned or else the PLC will reject the file.
Note 5:	Follow the example mentioned in the window while entering the start address.
Note 6:	Decimal values cannot be transferred over the Modbus protocol, so the software will display integral values.

Appendix II

Install Raspbian on Raspberry Pi

Raspbian is one of the common operating systems of Raspberry Pi. Raspbian is a convenient and versatile platform that operates similarly to a PC. It has got a browser, a command line, and several other programs as that of a PC. In addition, several other software applications like Python, Java, SonicPi, Scratch and many more. The SD card of Raspberry Pi has an operating system installation manager called NOOBS which is preinstalled. NOOBS stands for New Out of Box Software. Insert the SD card on Raspberry Pi and connect it to the monitor. The upcoming screen will show numerous operating systems like Raspbian, LibreELEC, and OSMC.

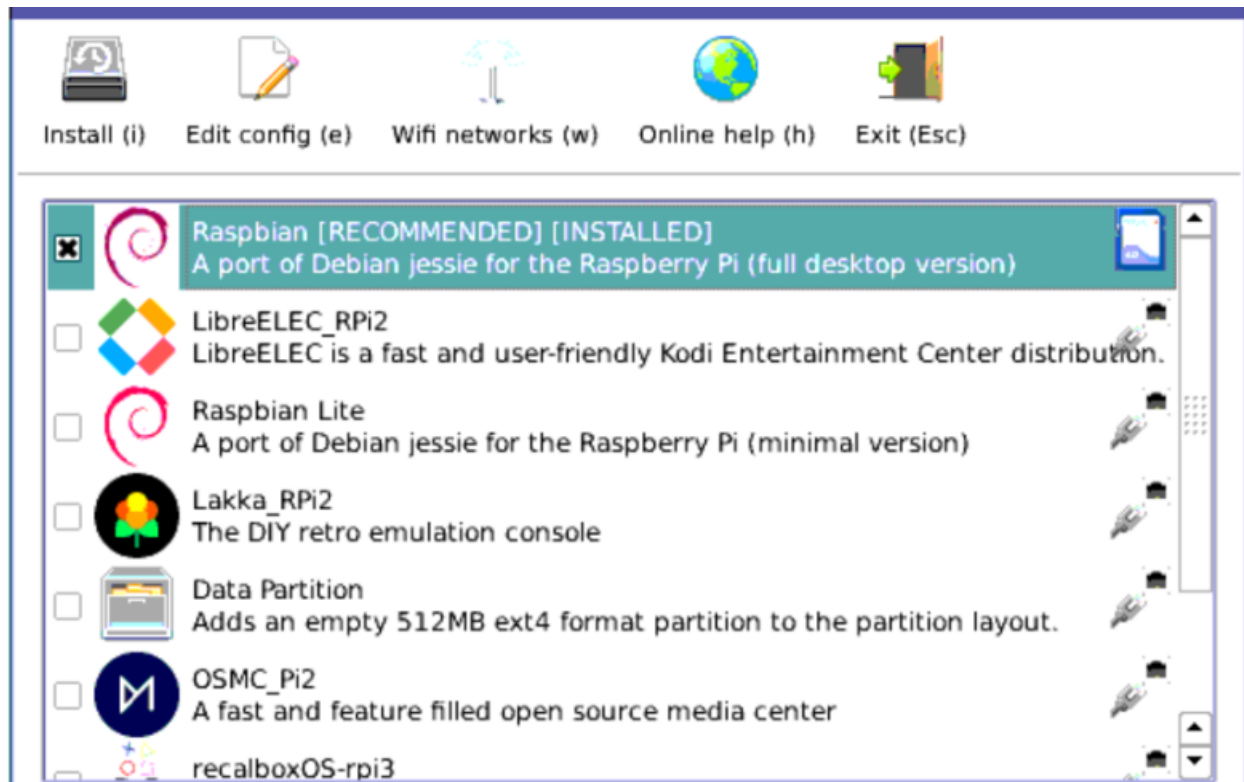


Figure 14. Choosing the operating system Raspbian

Since the required operating system is Raspbian, the first option must be chosen and installed. Once Raspbian has been installed, restart the Raspberry Pi to boot up the Raspbian.

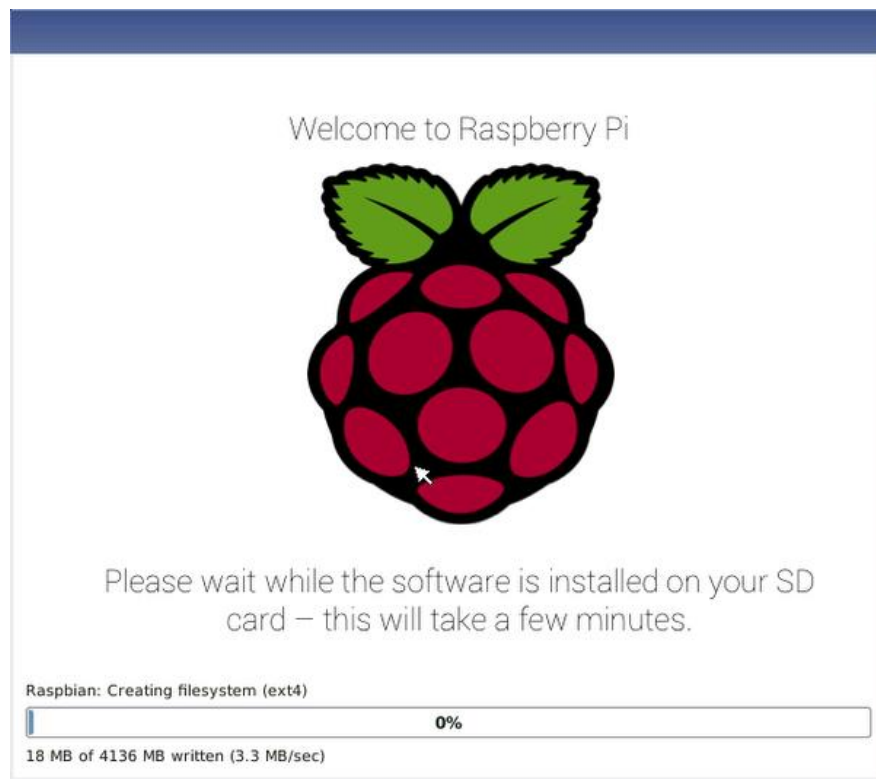


Figure 15 Installing Raspbian OS

Install Node-RED on Raspberry Pi

To install Node-RED on Raspbian OS, the following command should be given. This command will download and run the script to install Node-RED.

```
bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)
```

Once installed, it can be run as a service in the background and will be automatically started up once the system has been booted. There are several commands that are provided with the installation of the application. They are as follows:

nod-red-start: This command is used to start the Node-red service and run it on the background.

nod-red-stop: this command stops the Node-RED service.

nod-red-restart: This command stops whatever work progressing on Node-RED and immediately restarts the Node-RED.

In addition, the command “*sudo systemctl enable nodered.service*” allows Node-RED to auto-start when the Raspberry Pi is turned on.

Install nodes on Node-RED

Node-Red comes with a set of pre-installed nodes. The nodes can be installed directly from the editor by choosing the option “Manage Palette” from the main menu. From the Manage Palette option, choose the Palette Manager option. Then, the screen will direct to the nodes tab which will show the list of all nodes installed. The install tab on the screen helps the user to search any node required and to install it.

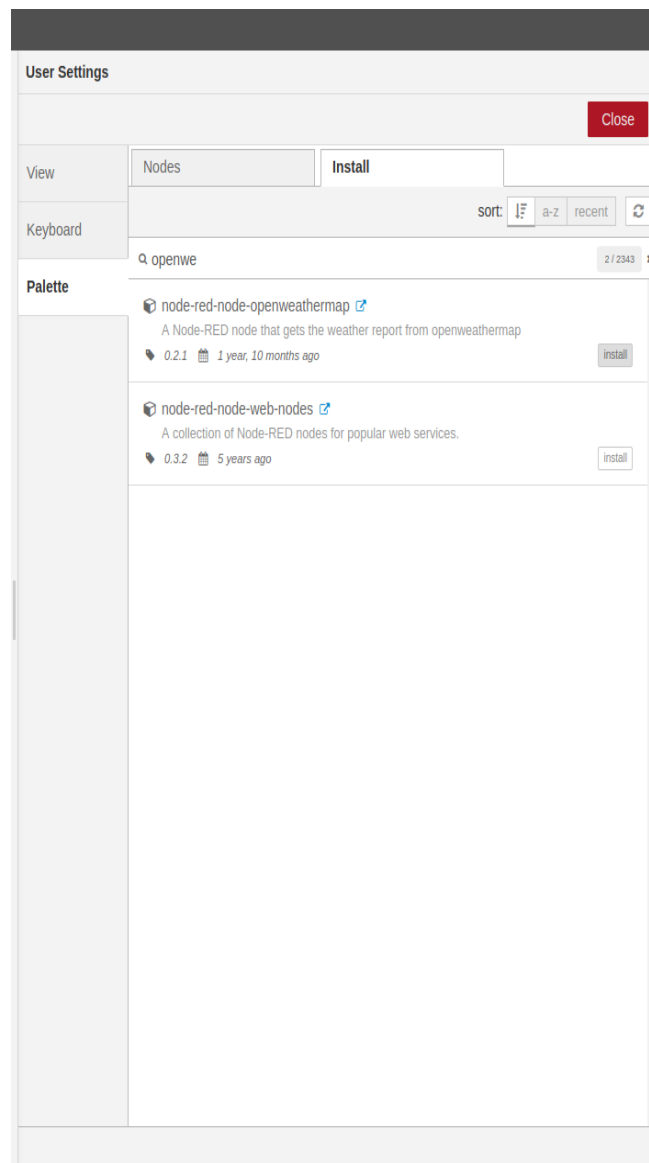


Figure 16 Installing nodes on Node-RED

The node called dashboard provides numerous nodes and helps to create a live data dashboard. This can be quickly installed by searching the nod-red-dashboard in the palette option tab. This dashboard can fetch data from any server and display it using gauges, graphs or any other user required widgets. After installing the node, the Node-RED should be restarted using the corresponding command to get the UI nodes as well as the new dashboard tab.

The layout of the tab can be customized using the layout tab in the dashboard. The tabs can be used to change the order of the tab, groups, and widgets. In addition, the theme and font of the UI can also be customized. Every widget should have a specific label as well as a value must be given to it. Few of the widgets are a button, chart, slide, switch and many more.



Figure 17 Nodes in the dashboard

Once all the nodes are customized and placed in the correct position, the deploy option should be enabled to run the flow. To open the user interface of Node-RED, the IP address should be typed in the browser followed by 1880/ui like `http:// IP_address:1880/ui`.

Arranging the nodes

The nodes are arranged in order to create a dynamic display for the solar power monitor. The nodes used are timestamp, msg. payload, plc, split input, functions, and gauges as required. The gauges are then customized to fetch the data from the corresponding registers of the PLC. Each register has a specific address and it was given in the address bar of the gauge while customizing it. Corresponding address of the registers were given to each gauge and they included the address of the temperature of water from supply line, temperature of water after heating, final outlet temperature, dc output current, dc output voltage, the graph depicting the power calculation, the gauges representing the reduction of CO2 emitted and oil saved in gallons.

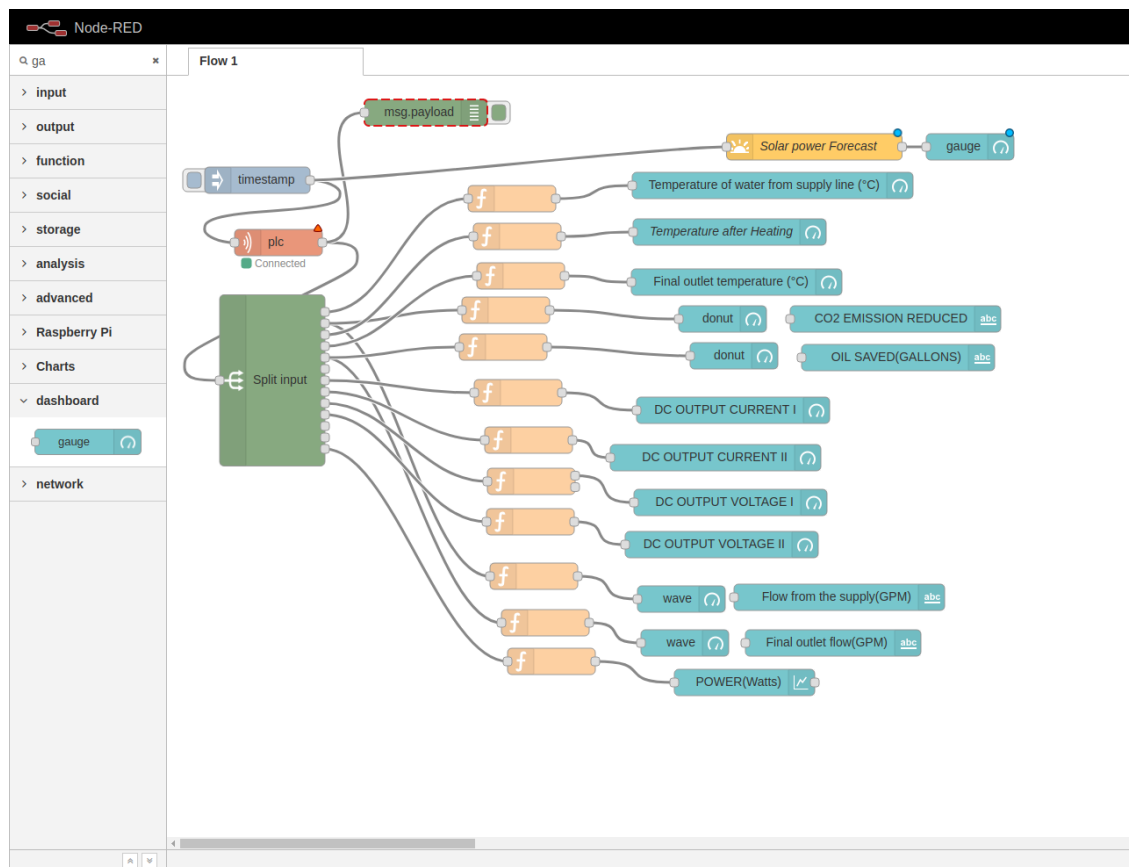


Figure 18. Nodes arranged to create dynamic display

Configuration of Modbus TCP and data fetching

Modbus TCP protocol is used to fetch the data from the PLC using the node called plc by giving the IP address of the PLC. The fetched data from the PLC will be in the form of an array. The array should be parsed to allocate the corresponding values to each node. Parsing can be done using a node called split input. Each pin of the split input node is connected to each gauge node as represented above. As a result, the fetched data will be displayed according to the gauge node and the input given to it. The node called msg.payload helps to display the data.

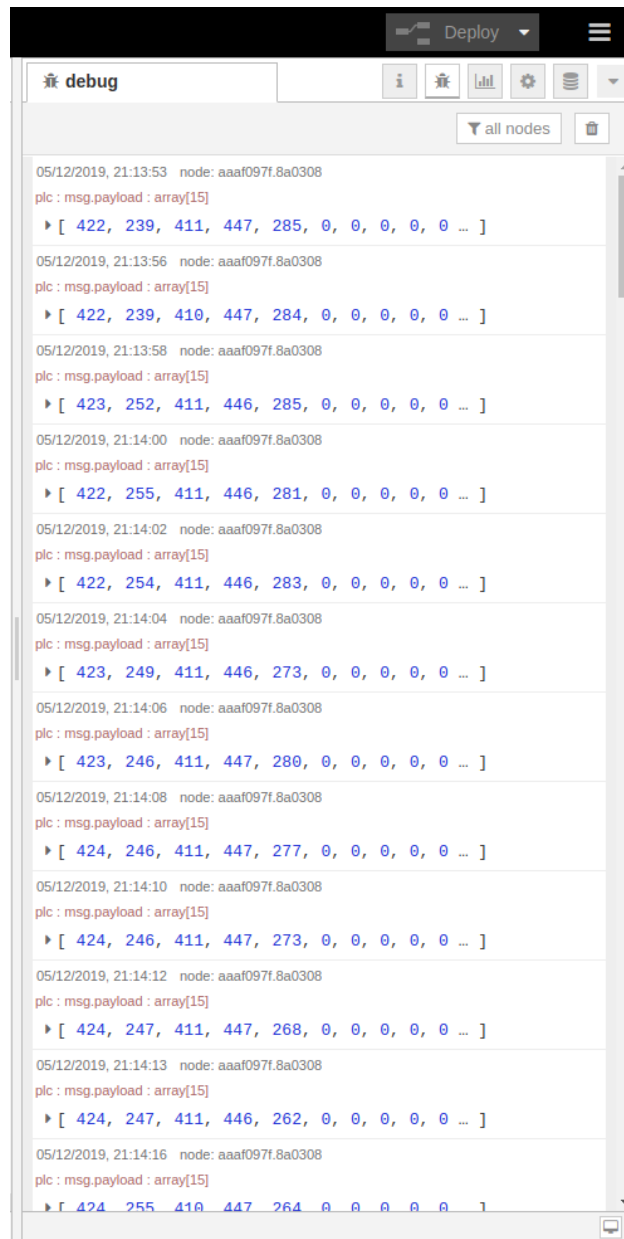


Figure 19. Data fetched from the registers

There are 15 registers associated with PLC which contain several data like temperature, current, voltage, cumulative power generated and many more. Thus, the array in the above image shows the size of the array as 15; array [15]. The address of the registers starts at 400001 and ends at 400015. This information must be mentioned at the properties of the plc node. The data is fetched every 2 seconds and thus the poll rate was modified to 2 seconds. The IP address of the PLC is mentioned at the server bar and the quantity is given as 15.

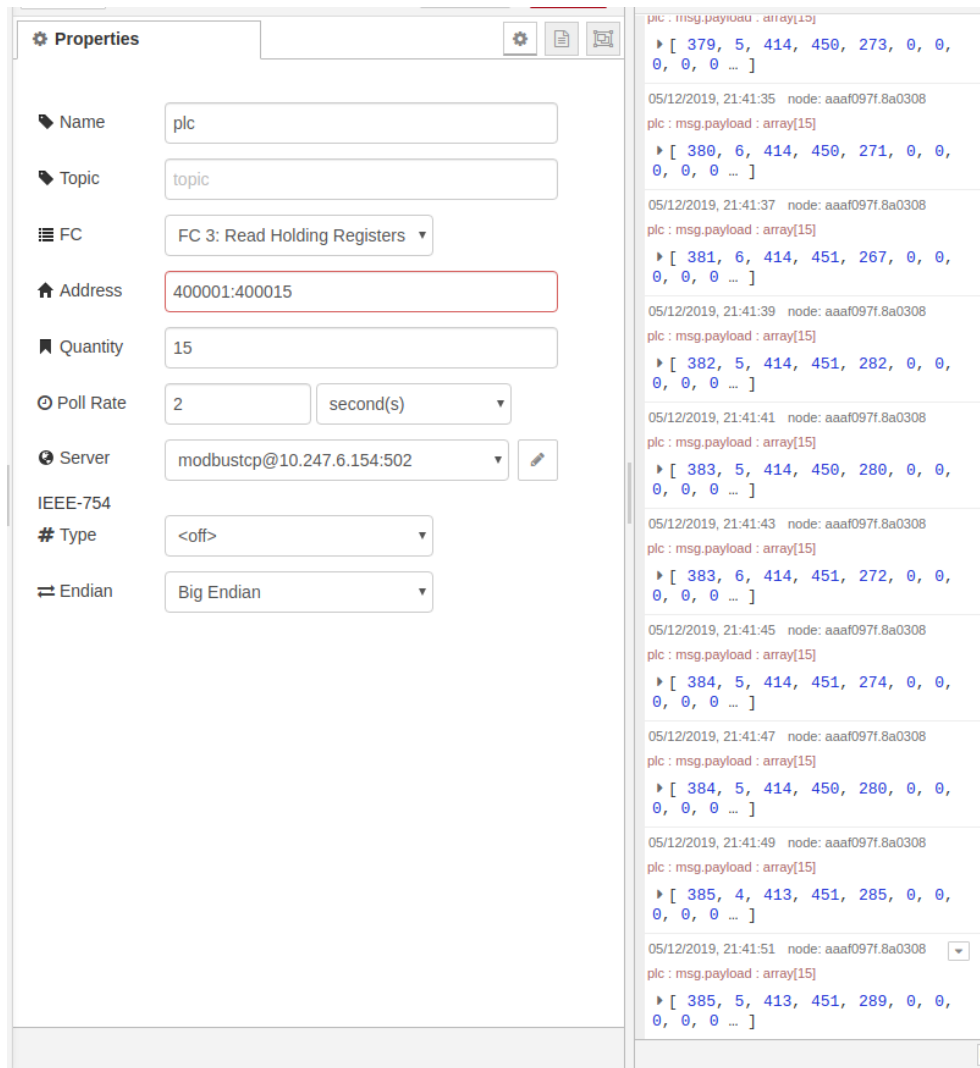


Figure 20. Initializing the address

Label	Register Address	Signed/Unsigned	Scaling Value	Register Type	Description
Supplyline-T9	400000	U	10	Holding	Temperature of water from supply line (°F)
SupplyLineFlow_F1	400001	U	10	Holding	flow from the supply(GPM)
TempAfterPV_T6	400002	U	10	Holding	temperature after heating using solar cells (°F)
FinalOutletTemp_T4	400003	U	10	Holding	Final outlet temperature (°F)
FinalOutletFlow_F2	400004	U	10	Holding	Final outlet flow(GPM)
DC_Current_Array1	400005	U	100	Holding	DC current output from pv array 1
DC_Current_Array2	400006	U	100	Holding	DCcurrent output from pv array 2
DC_Volatge_Array1	400007	U	100	Holding	DC voltage output from pv array 1
DC_Voltage_Array2	400008	U	100	Holding	DC voltage output from pv array 2
Power_Array1_Watts	400009	U	100	Holding	Output power from pv array 1 in watts
Power_Array2_Watts	400010	U	100	Holding	Output power from pv array 2 in watts
Power_Array1_BTU	400011	U	100	Holding	Output power from pv array 1 in BTU
Power_Array2_BTU	400012	U	100	Holding	Output power from pv array 2 in BTU
Total_Power_Watts	400013	U	100	Holding	Total output power in watts
Total_Power_BTU	400014	U	100	Holding	Total output power in BTU
PowerForecast	400015	U	100	Holding	Estimated generated power(to be updated)

Figure 21. Register addresses and label