DOI: https://doi.org/10.52436/1.jutif.2023.4.3.1027
p-ISSN: 2723-3863
e-ISSN: 2723-3871

IMPLEMENTATION AND ANALYSIS OF THE INTERNET OF THINGS SYSTEM FOR ELECTRICAL ENERGY MONITORING AT INSTITUT TEKNOLOGI TELKOM PURWOKERTO

I Ketut Agung Enriko*[1], Mas Aly Afandi[2], Herryawan Pujiharsono[3], Fikri Nizar Gustiyana[4], Hedi Krishna[5], Filbert H. Juwono[6]

[1,2,3]Faculty of Telecommunications Engineering and Electrical Engineering, Institut Teknologi Telkom Purwokerto, Indonesia
[4]Fakulty of Electrical Engineering, Telkom University, Indonesia
[5]Indonesia Telecommunication & Digital Research Institute, Indonesia
[6]Connected Intelligence Research Group, University of Southampton, Malaysia
Email: [1]enriko@ittelkom-pwt.ac.id

(Article received: May 03, 2023; Revision: May 22, 2023; published: June 26, 2023)

Abstract

Measurement of electric power usage is carried out using simple measuring instruments and the recording is still manual so that the data obtained is not real-time and accurate. This research aims to implement an electrical energy monitoring system using the Internet of Things (IoT) to obtain real-time information related to electrical energy in the education industry. This research uses an Industrial Grade Power Meter to get a more accurate measurement value. To connect the Power Meter device with the IoT system, this research uses Modbus RS485 communication and a mini PC to process data from the meter, so that the data can be sent to a server using the MQTT communication protocol, and displayed on the Dashboard. The test results of this study indicate that the monitoring system can be implemented and the system runs well with end-to-end measurement results. From the measurement results, the current value (3 phase average) has an average deviation of 0.001 Amperes, Voltage (3 phase average) has an average deviation of 0.519 V, Power factor has an average deviation of 0.012, Active power has a deviation average of 0.000 kW, reactive power with an average deviation of 0.000 kVAR, apparent power with an average deviation of 0.000 kVA and frequency with an average deviation of 0.124 Hz. Then the MQTT protocol has a quality of service with index 4 based on TIPHON standardization on delay, throughput, and packet loss parameters, and index 3 based on TIPHON standardization on jitter parameters.

Keywords: *Internet od Things, Monitoring, MQTT, Power, RS485.*

1. INTRODUCTION

Currently, technology is developing rapidly in various scientific fields. Humans continue to strive to develop and research the latest technologies to make it easier for humans themselves. One of them is in the field of technology regarding the Internet of Things (IoT). The Internet of Things has been widely applied in several fields of science and industry, such as in the fields of health, informatics, and geography, and in everyday life, it can also be applied. At the opening ceremony of the 2018 Indonesia Industrial Summit, the Ministry of Industry launched the Industry 4.0 Roadmap [1]. The main technologies supporting Industry 4.0 are artificial intelligence, 3D printing technology, robotic and sensor technology, human-machine interface, and the internet of things. Thus the use of IoT is in line with government policies to build a globally competitive manufacturing industry.

The use of electric power in the industrial sector so far can only be seen using the kWh meter measuring instrument distributed by PLN. One of the industrial sectors that consumes large amounts of electrical energy is the education sector which consumes 72% of electrical energy [2]. This tool does not provide detailed information about how much electricity is connected to the kWh meter. When viewed from historical data, sometimes a change in power usage suddenly drops due to machine failure or unknown maintenance in the education sector. This will result in a waste of electric power. Because of this, monitoring tools are needed for consumers who can show the use of electric power, so they can monitor the use of electric power [3]. To be able to view electricity usage data directly, a system is needed that is designed to display electricity usage data and its characteristics on devices such as computers and Android which can be accessed. anywhere, anytime as long as the device is connected to the Internet. In this way, historical data on electricity use can be recorded in an orderly manner. comparing the readings of the PLN meter system with the data obtained from the designed system [4]. In Indonesia itself, the development of smart meter devices has been used in several places, one of which is in Musi Banyuasin, Palembang. PT Muba Elektrik

Power, a Regional Owned Enterprise of the Musi Banyuasin Regency, makes it easy for 48,000 customers to monitor electricity usage through the Smart Electric Muba application [5]. With the application of smart meters, it is hoped that the Education sector can monitor the use of electrical energy through applications, especially in the Educational Area of the Telkom Purwokerto Institute of Technology.

Electric power is defined as the rate of delivery of electrical energy in an electric circuit. In AC power networks with sinusoidal waveforms, several types of power are known, including active power, reactive power and apparent power. [6].

Internet of Things is composed of 2 words, namely, Internet and Things. Internet itself means interconnection and networking [7]. Internet Of Things or commonly referred to as IoT is a concept that aims to be able to expand the benefits of internet connectivity that is connected continuously. [8].

Modbus is an application layer messaging protocol from Modicon, located at level 7 of the OSI model, that provides client/server communication between devices connected on any type of bus or network. [9].

Message Queuing Telemetry Transport is a lightweight publish¬/subscribe-based protocol specifically designed for communication on low power devices or lightweight messages. [10]. A lightweight publish-subscribe message-based protocol requires a message broker, where the broker is responsible for distributing messages to interested clients based on message topics [11].

Quality of Service (QoS) is the ability of a network to provide a better service for traffic services that pass through it. QoS is an end-to-end architectural system and is not a network feature [12].

Research on an Electric Power Usage Monitoring system using the Web-based MQTT Protocol. The monitoring system that is running has a small delay, which is 11 ms. In addition, the MQTT protocol can also be run in limited network conditions, namely in network conditions that have low bandwidth [13]. This study applies RS485 data communication with the main components of the microcontroller [14].

This research designed an IoT Gateway Using the MQTT Protocol on Modbus-RTU Based Control Devices. The use of the Modbus RTU protocol was chosen because it is a protocol that has been widely used in industrial environments, especially in Programmable Logic Control (PLC) communications.) [15]. This research will create a simple display for HMI using Node-red which aims to display sensor reading data to a Dashboard [16]..

2. RESEARCH METHOD

A. Tools and Materials Used

In this Thesis Report, several stages of the workings of the tool that will be used will be explained.

1) Tools

a. Computer/PCIn this study, the computer or PC was a PC with the HP 14s-dk0148AU. Which uses an AMD Reyzen 5 processor with DDR48GB memory. With SSD storage of 512GB NVme. A PC with this specification is used for report generation, system integration, and QoS measurement

Fig. 1. PC HP 14s-dk0148AU

b. Mikrotik Access point

With it, a router operating system can be created using home devices that can maintain 2GHz and 5GHz dBi. This proxy operates by connecting to the Winbox application speeds. Supported by wireless chip models QCA9880 and QCA9588 with antenna gain specifications of 2-2.5

The Mikrotik access point used in this study has a display with components that can be used, namely, there is a CPU space of 720 MHz and 128 MB of RAM and a flash size of 16 MB, has five Ethernet ports with SFP space, and an output port and a USB port for a 3G modem. /4G as shown in Figure .2. In this study the authors use a proxy Access Point as an Access Point Wifi IoT Gateway.

Fig. 2. Mikrotik Access Point

c. Node-Red

Node-RED is a browser-based tool for creating Internet of Things (IoT) applications whose visual programming environment makes it easy for users to create applications as "flow". This flow is formed from interconnected nodes where each node performs certain tasks.

Although Node-RED was designed for the Internet of Things (IoT), it can also be used for general purposes and various applications. It is named "Node" because it is implemented as a node application but from the customer's point of view, it is only the details of the internal implementation. In this study, the authors use node-red to integrate data from the power meter to the IoT Gateway and create a local. dashboard.

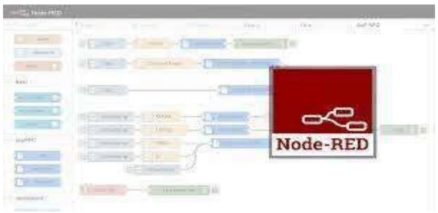
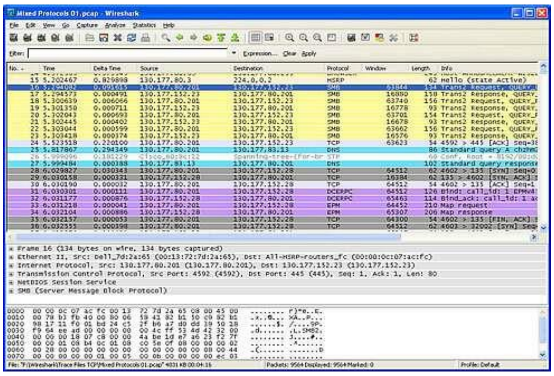


Fig. 3. Node-Red

d. Software Wireshark

Wireshark software that functions as a network protocol analyzer. In this measurement methodology, Wireshark, which is useful as a network analyzer, is useful for connecting clients to communicate with each other, and packets are sent from client1 (sender) to client2 (recipient) and vice versa, which is done via a server. In this study, the authors used Wireshark to measure the quality of service on the MQTT protocol. MQTT is easy to deploy and can communicate IoT data efficiently. MQTT supports messaging between devices to the cloud and cloud to devices.



```
No. -   Time       Delta Time  Source            Destination           Protocol  Window  Length  Info
     15 5.202467   0.829898    130.177.80.3      224.0.0.2             HSRP                 62 Hello (state Active)
     16 5.294082   0.091615    130.177.80.201    130.177.152.23        SMB       63844     134 Trans2 Request, QUERY_
     17 5.294573   0.000491    130.177.152.23    130.177.80.201        SMB       16080     158 Trans2 Response, QUERY_
     18 5.300639   0.006066    130.177.80.201    130.177.152.23        SMB       63740     156 Trans2 Request, QUERY_F
     19 5.301350   0.000711    130.177.152.23    130.177.80.201        SMB       16778      93 Trans2 Response, QUERY_
     20 5.302043   0.000693    130.177.80.201    130.177.152.23        SMB       63701     154 Trans2 Request, QUERY_F
     21 5.302445   0.000402    130.177.152.23    130.177.80.201        SMB       16678      93 Trans2 Response, QUERY_
     22 5.303044   0.000599    130.177.80.201    130.177.152.23        SMB       63662     156 Trans2 Request, QUERY_F
     23 5.303418   0.000374    130.177.152.23    130.177.80.201        SMB       16576      93 Trans2 Response, QUERY_
     24 5.523518   0.220100    130.177.80.201    130.177.152.23        TCP       63623      54 4592 > 445 [ACK] Seq=3E
     25 5.817867   0.294349    130.177.80.201    130.177.83.13         DNS                  86 Standard query A ch2hm
     26 5.999096   0.181229    Cisco_80:9c:12    Spanning-tree-(for-br STP                  60 Conf. Root = 8192/00:d
     27 5.999484   0.000388    130.177.83.13     130.177.80.201        DNS                 102 Standard query response
     28 6.029827   0.030343    130.177.80.201    130.177.152.28        TCP       64512      62 4602 > 135 [SYN] Seq=0
     29 6.030158   0.000331    130.177.152.28    130.177.80.201        TCP       16384      62 135 > 4602 [SYN, ACK] S
     30 6.030190   0.000032    130.177.80.201    130.177.152.28        TCP       64512      54 4602 > 135 [ACK] Seq=1
     31 6.030301   0.000111    130.177.80.201    130.177.152.28        DCERPC    64512     126 Bind: call_id: 1 EPMv4
     32 6.031177   0.000876    130.177.152.28    130.177.80.201        DCERPC    65463     114 Bind_ack: call_id: 1 ac
     33 6.031218   0.000041    130.177.80.201    130.177.152.28        EPM       64452     210 Map request
     34 6.032104   0.000886    130.177.152.28    130.177.80.201        EPM       65307     206 Map response
     35 6.032157   0.000053    130.177.80.201    130.177.152.28        TCP       64300      54 4602 > 135 [FIN, ACK] S
     36 6.032555   0.000398    130.177.80.201    130.177.152.28        TCP       64512      62 4603 > 32002 [SYN] Seq

+ Frame 16 (134 bytes on wire, 134 bytes captured)
+ Ethernet II, Src: Dell_7d:2a:65 (00:13:72:7d:2a:65), Dst: All-HSRP-routers_fc (00:00:0c:07:ac:fc)
+ Internet Protocol, Src: 130.177.80.201 (130.177.80.201), Dst: 130.177.152.23 (130.177.152.23)
+ Transmission Control Protocol, Src Port: 4592 (4592), Dst Port: 445 (445), Seq: 1, Ack: 1, Len: 80
+ NetBIOS Session Service
+ SMB (Server Message Block Protocol)

0000  00 00 0c 07 ac fc 00 13  72 7d 2a 65 08 00 45 00   ........ r}*e..E.
0010  00 78 b3 fb 40 00 80 06  58 41 82 b1 50 c9 82 b1   .x..@... XA..P...
0020  98 17 11 f0 01 bd 24 c5  2f b6 a7 d0 dd 19 50 18   ......$. /.....P.
0030  f9 64 ee ad 00 00 00 00  00 4c ff 53 4d 42 32 00   .d...... .L.SMB2.
0040  00 00 00 18 07 c8 00 00  4a be 1d e7 a6 23 f2 7f   ........ J....#..
0050  00 00 01 08 b4 0c 01 08  c0 5e 0f 08 00 00 00 02   ........ .^......
0060  00 28 00 00 00 00 00 00  00 00 00 00 00 08 00 44   .(...... .......D
0070  00 00 00 00 00 01 00 05  00 0b 00 00 00 00 ec 03   ........ ........

File: "F:\Wireshark\Trace Files TCP\Mixed Protocols 01.pcap" 4831 KB 00:04:16   Packets: 9564 Displayed: 9564 Marked: 0   Profile: Default
```

Fig. 4. Software Wireshark

2) Material Used

a. Schneider PM2100

This research used the PM2100 as an implemented power meter in this study. The PM2100 is a digital meter that offers comprehensive 3-phase electrical instrumentation and load management facilities in a compact and rugged package. The PM2100 offers energy monitoring and electricity cost management application features. All meters in the PM2100 series comply with Class 1, or Class 0.5S accuracy standards and deliver high quality, reliability, and affordability in a compact, easy-to-install format. The PM2100 series meters support many features, some of the features are listed below:

a) LED display screen: Intuitive self-navigation using three display LED buttons, with three rows of values together. Two LED columns light up on both sides of the gauge's front panel indicating its parameter name is displayed.
b) Energy calculation and balancing
c) Measurement of True Power Factor and Displacement Power factor
d) Modbus RTU protocol support with RS485 interface
e) And others, which are more complete, can be seen in the PM2100 datasheet in the Appendix [17].

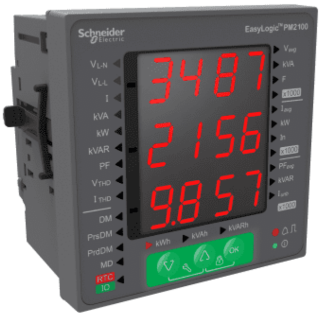


Fig. 5. Power Meter Schneider PM2100

b. Schneider Current Transformer 100/5 METSECT5CC010

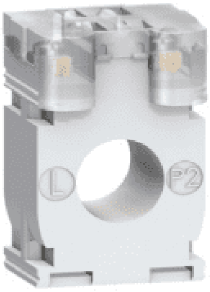


Fig. 6. Schneider Current Transformer 100/5 METSECT5CC010

To be able to measure the current flowing, the author uses a tool called Current Transform. Schneider Current Transformer METSECT5CC010 is a measuring tool that functions to convert the magnetic field into an electric current that can be processed by PM2100 with specifications reading Current 5A, Maximum voltage rating 720V, Frequency 50/60Hz with IEC Standard 61869-2 and CT 100/100 ratio 5 [18].

c. Mini Circuit Breaker (MCB) 3 Phase

MCB has a func as a circuit breaker towards the load direction. Disconnecting the current can be done manually or automatically. To do this manually, that is by changing the toggle switch that is in front of the MCB after that the MCB will cut off the electric current. Meanwhile, it will automatically occur if an overcurrent is detected due to an overload of electricity usage or a short circuit. In this study, the authors used the MCB as a safety device in the circuit.

Fig. 7. MCB 3 Phase

d. Raspberry Pi 3B+

Fig. 8. Raspberry Pi 3B+

Raspberry Pi, often abbreviated as Raspi, is a single-board computer (SBC) the size of a credit card that can be used to run office programs, and computer games, and as a media player for high-resolution video. The Raspberry Pi was developed by a non-profit foundation, the Rasberry Pi Foundation, which is fronted by a number of developers and computer experts from the University of Cambridge, England. Raspberry Pi 3 Model B+ is the latest version of the mini-computer released by the Raspberry Pi as a development of the previous Raspberry Pi version and as a successor to the Raspberry Pi 3 Model B which has already been released. Physically, the Raspberry Pi 3 Model B and Raspberry Pi 3 Model B+ are the same so that for the case the Raspberry Pi 3 Model B version can be used. Some of the advantages and improvements to the Raspberry Pi 3 Model B+ compared to the previous ones are as follows:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz.
- 1GB LPDDR2 SDRAM.
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE.
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO headers.
- Full-size HDMI.
- 4 USB 2.0 ports.

Based on the specifications above, the author uses raspberry pi as an IoT gateway in this study.

e. RS485 to USB Converter

This device aims to convert the RS485 communication interface to a USB port that is applied to the Raspberry Pi so that it can be compatible with existing USB 2.0 ports and there is no need to add a RS 485 shield to the Raspberry Pi. The author uses this device to convert RS485 communication to USB on IoT Gateway.

Fig. 9. Converter RS485 to USB

B. Research Flow

The research was carried out in several stages, namely the stages starting from the study of the determination of the literature study, system design, system testing, and analysis of the results of the tests that had been carried out.

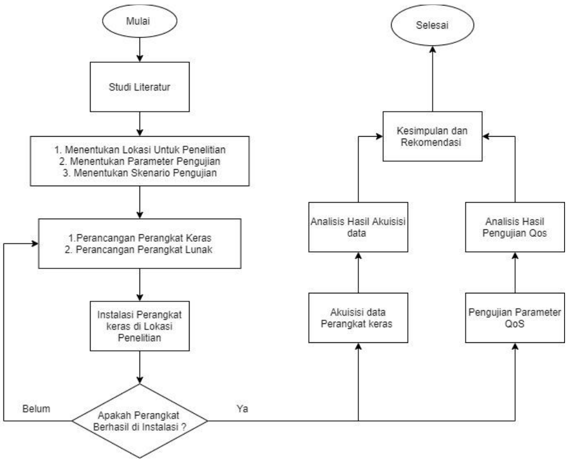

Mulai
Studi Literatur
1. Menentukan Lokasi Untuk Penelitian
2. Menentukan Parameter Pengujian
3. Menentukan Skenario Pengujian
1.Perancangan Perangkat Keras
2. Perancangan Perangkat Lunak
Instalasi Perangkat keras di Lokasi Penelitian
Apakah Perangkat Berhasil di Instalasi ?
Belum
Ya
Akuisisi data Perangkat keras
Analisis Hasil Akuisisi data
Pengujian Parameter QoS
Analisis Hasil Pengujian Qos
Kesimpulan dan Rekomendasi
Selesai

Fig. 10. Research Flowchart

This research requires a series of stages, from collecting literature studies to the conclusion and recommendation stages. A series of research stages is necessary to achieve the performance of the research process. The first stage is starting with finding sources from research that has been done before. The sources used are sources based on scientific journals, proceedings, and books. Literature study functions as a reference provider so that new concepts can be found that can be developed in research.

Next determine the research location that will be tested, after the test location is determined, the author determines what parameters will be tested related to the system that is made. Then determine the research scheme or scenario, research must determine how the research will be carried out later, so it is necessary to design a research scenario. After the research testing scenario is determined. In this study several parameter tests will be carried out, starting from testing PM2100 data acquisition for each parameter, and testing QoS performance with QoS standards 0, 1, and 2 on MQTT, to find out how good the quality of the protocol used on the system is. get, then do the Hardware and Software design with the tools and materials previously mentioned.

After the hardware and software have been designed, the next stage is the installation of the device at the test.

C. System Planning

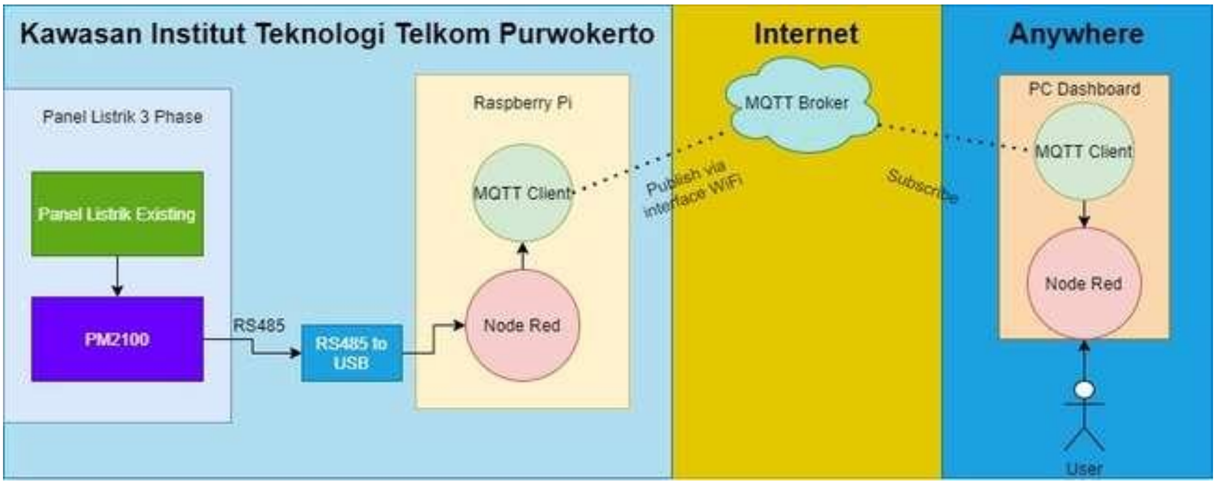


Fig. 11. System Block Diagram

In Figure 11 it can be explained that the design of this tool begins with installing the metering device to the Existing Electrical panel, then the metering device is connected to the IoT Gateway with the RS 485 interface which is converted to USB. The Raspberry Pi uses the Modbus RTU Protocol to acquire data from the PM2100. After the data has been acquired, the Raspberry pi then forwards the data to the MQTT broker using the publishing method via a WiFi interface that is connected to the Internet using Node-red. After that, PC users connected to the Internet can access the data by subscribing to Node-Red and displaying it on a simple Dashboard.

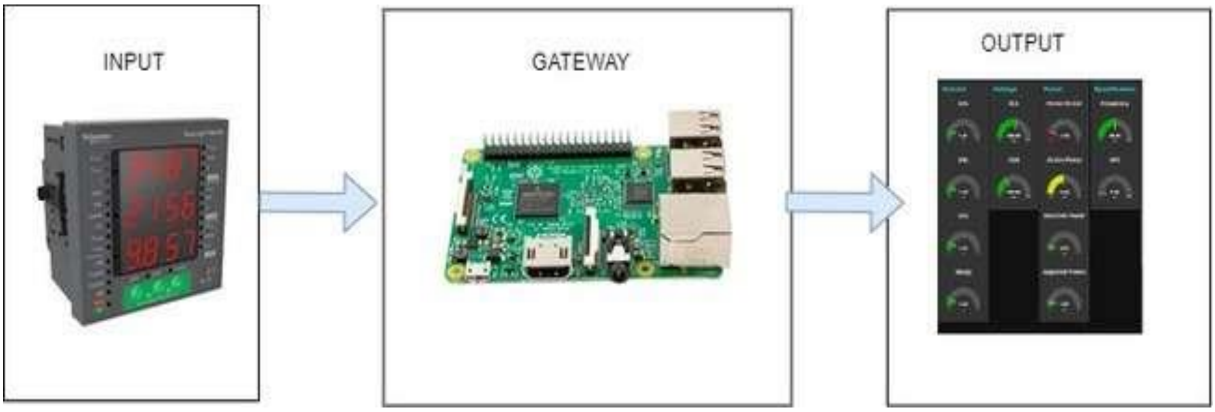


Fig. 12. Block Diagram of System Input and Output

From Figure 12 above it can be seen that the input from this research is the value of the quantity read by PM2100 from the existing electrical panel. After all, parameters can be read by the PM2100, then the data will be acquired by the IoT Gateway. The IoT Gateway here acts as a bridge that delivers data parameter values and sends them to the internet via a WiFi network. The data will then be displayed in the form of a simple dashboard which is the output of this research.

Here are some explanations of the components in the Block Diagram above

a) PM2100 functions to acquire electrical parameter data in the form of Current,
b) Voltage, Power (Active, Reactive, Apparent), Power Factor, and Frequency.
c) Raspberry Pi is a device that acquires data from PM2100 and forwards the data to the MQTT broker to be displayed on the Dashboard using Node-Red.

d) Data acquisition from PM2100 to Raspberry Pi uses RS485 interface and Modbus RTU communication protocol

e) Sending data from the Raspberry Pi to the Broker via the internet using WiFi.

f) The communication protocol used when sending data from the Raspberry Pi to the Dashboard uses the MQTT protocol.

g) The dashboard functions to display real-time data sent by the IoT Gateway.

3) Hardware Design

The design of the entire device is a hardware design and software design on the system created. The hardware design of this system consists of all schematic drawings of the interface design between the PM2100 and the Existing Panel and the PM2100 interface with the Raspberry Pi.

a. The PM2100 interfaces with the Existing Panel

The interface of the PM2100 with the existing panel can be seen in Figure 13

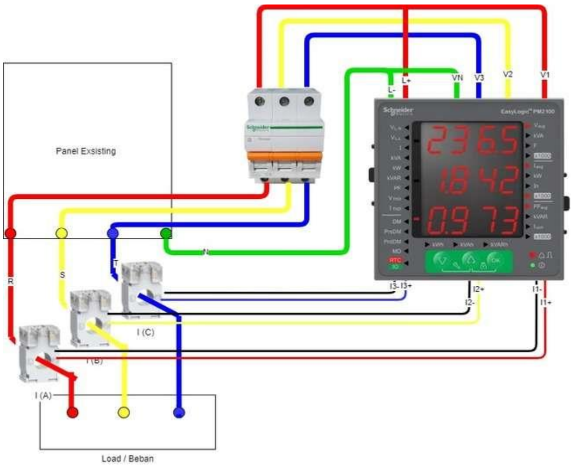


Fig. 13. PM2100 Interface Wiring Diagram with Existing Panel

Figure 13 is an illustration of the wiring connectivity that connects Existing panel devices to PM2100, the placement of Current Transformers, and the Current Transformer connectivity to PM2100. The port for each connection can be seen in Table 1.

Table. 1. Existing and PM2100 Panel Connection Tables

PM2100	Panel Exsisting	CT
V1	R (Trough MCB)	-
V2	S (Trough MCB)	-
V3	T (Trough MCB)	-
VN	N	-
L+	R (Trough MCB)	-
L-	N	-
I1+	-	I(A) +
I1-	-	I(A) -
I2+	-	I(B) +
I2-	-	I(B) -
I3+	-	I(C) +
I3-	-	I(C) -

b. The Interface between PM2100 with Raspberry Pi

The interface between PM2100 and Raspberry Pi. The interface description of the PM2100 with the Raspberry Pi is shown.

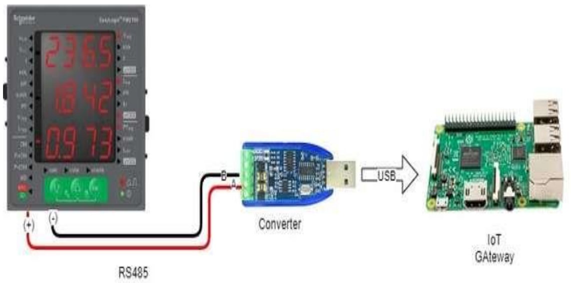


Fig. 14. The PM2100 interfaces with the Raspberry Pi

To connect the interface between PM2100 and Raspberry Pi, a conversion device is needed to convert the RS485 interface to USB as shown in Figure 3.14. For detailed port connections, see.

Table.2.

PM2100	Converter Rs485 to USB	Raspberry Pi
RS485 +	A	-
RS485 -	B	-
-	USB Male	USB Female

c. Software Design

Figure 15 describes the flowchart of the software planning that is made on this system. There are 3 main parts in software planning including IoT gateway, MQTT Broker, and PC user as Dashboard. The flow starts from the IoT Gateway device that requests data transmission to the PM2100 device, if the PM2100 sends a response to a data request, then the next data is coded in the form of String data or sentences. The data sent from PM2100 is data in the form of a buffer. After the data for each parameter is obtained and encoded into the form of the String data type, then the data is combined so that it becomes a message package that will be sent to the MQTT Broker. Simultaneously with data acquisition from PM2100, IoT Gateway also configures MQTT connectivity to connect to the intended Broker. If the IoT Gateway is connected, then the data packets that were previously sent directly to the MQTT Broker. On the Dashboard side, the PC device connects to MQTT Broker. If the device is connected, the device retrieves the data that has been sent by the IoT gateway to the MQTT Broker, then displays it in the form of a Dashboard.

DOI: https://doi.org/10.52436/1.jutif.2023.4.3.1027
p-ISSN: 2723-3863
e-ISSN: 2723-3871

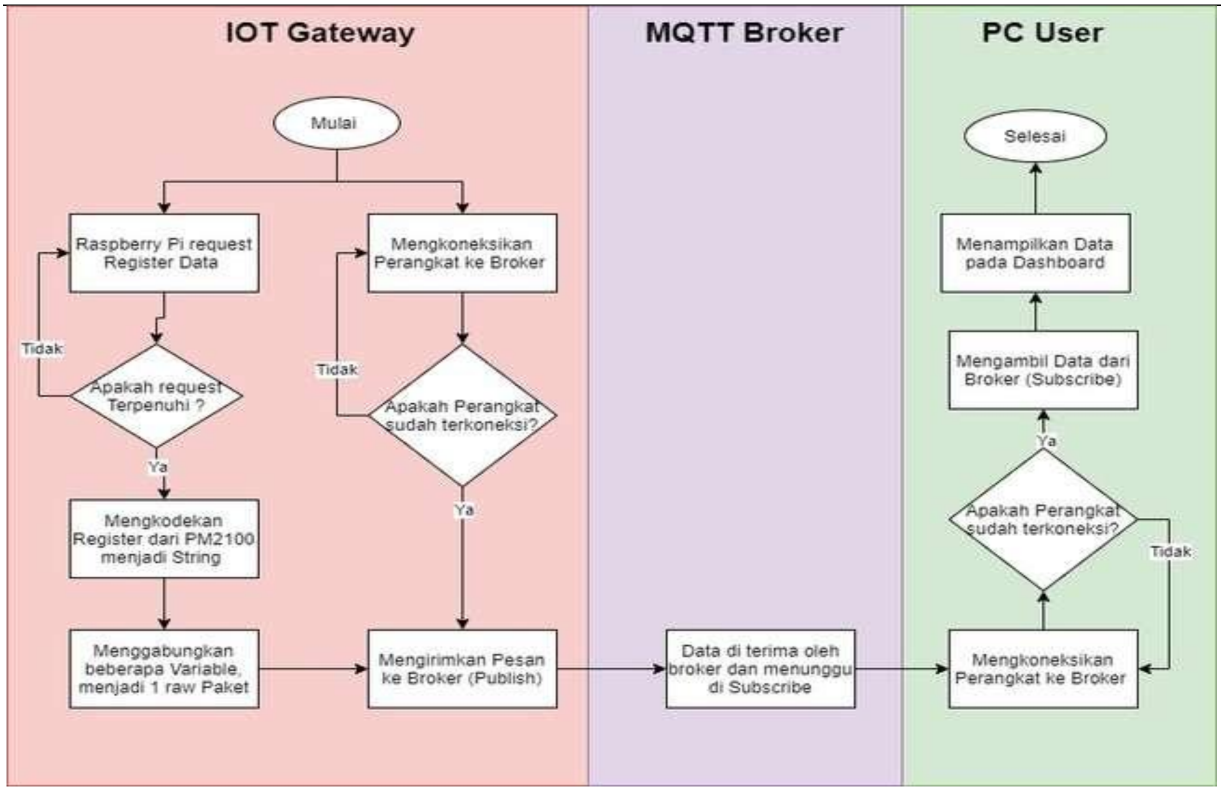

IOT Gateway
MQTT Broker
PC User
Mulai
Raspberry Pi request Register Data
Mengkoneksikan Perangkat ke Broker
Tidak
Apakah request Terpenuhi ?
Tidak
Apakah Perangkat sudah terkoneksi?
Ya
Mengkodekan Register dari PM2100 menjadi String
Ya
Menggabungkan beberapa Variable, menjadi 1 raw Paket
Mengirimkan Pesan ke Broker (Publish)
Data di terima oleh broker dan menunggu di Subscribe
Selesai
Menampilkan Data pada Dashboard
Mengambil Data dari Broker (Subscribe)
Ya
Apakah Perangkat sudah terkoneksi?
Tidak
Mengkoneksikan Perangkat ke Broker

Fig. 15. Software flowchart

3. RESULT AND DISCUSSION

A. SYSTEM DESIGN RESULT

In this study, there were several designs including the results of hardware design including meter installation to existing panels, IoT Gateway installation, and integration between Meter and IoT Gateway. Then the design of the software includes the display on the Dashboard. The installation results can be seen in Figure .16

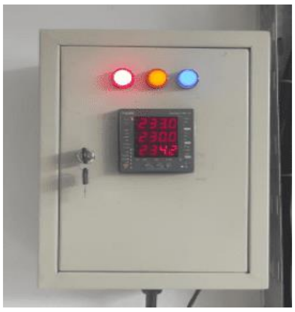


Fig. 16. Power Meter Panel Installation Results

combined so that it becomes a message package that will be sent to the MQTT Broker. Simultaneously with data acquisition from PM2100, IoT Gateway also configures MQTT connectivity to connect to the intended Broker. If the IoT Gateway is connected, then the data packets that were previously sent directly to the MQTT Broker. On the Dashboard side, the PC device connects to MQTT Broker. If the device is connected, the device retrieves the data that has been sent by the IoT gateway to the MQTT Broker, then displays it in the form of a Dashboard.

a) PM2100 Interface Installation Results with Existing Panels

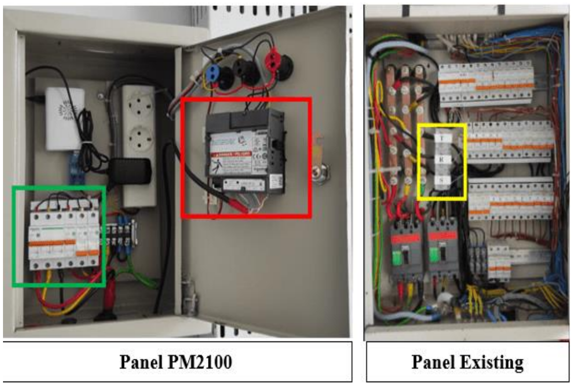


Fig. 17. Installation results of the PM2100 interface and the Existing Panel

In Figure 17 there are 3 devices for the Pm2100 interface and the existing panel, each of which functions as:

a. The green box is an MCB that functions as a scavenger and breaker of electricity which is measured from the existing panel
b. The red box is PM2100 which functions as a measuring instrument for electrical energy on the existing panel
c. The yellow box is a CT device that functions to read the current on each 3-phase cable line (R,S, T)

b) PM2100 Interface Installation Results with Raspberry Pi

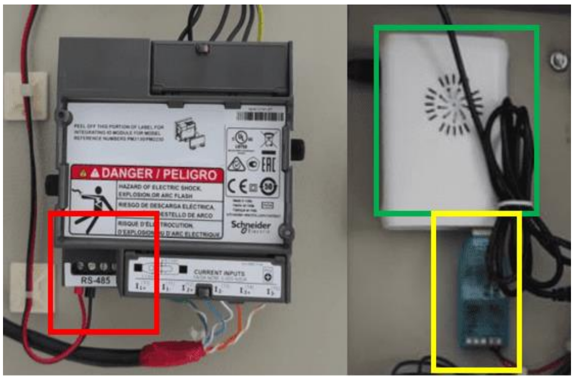


Fig. 18. Installation results of PM2100 interface with Raspberry Pi

In Figure .18 there are 3 devices for the Pm2100 interface and the existing panel, each of which functions as:

1. Raspberry Pi green box that functions as an IoT Gateway.
2. The red box is the RS485 communication interface on the PM2100.
3. The yellow box is the Rs485 to USB Converter device which functions to convert the RS485 interface to a USB port.

c) Flow Node-Red on IoT Gateway

The flow design and configuration of Node-Red on the IoT Gateway can be seen in Figure .19

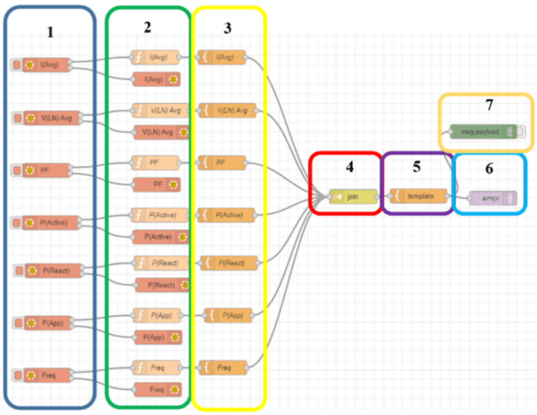
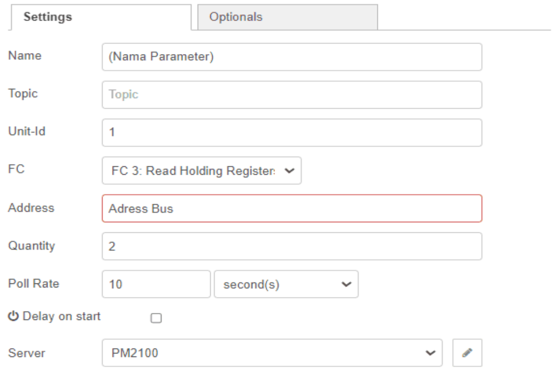


Fig. 19. Installation results of PM2100 interface with Raspberry Pi

Based on Figure .19 there are 7 flow segments which can be explained as follows:

1. Segment 1: using the Modbus Node which functions to Request data parameters that are measured using the Modbus RTU protocol from the RS485 interface connected to PM2100. Modbus RTU communication configuration can be seen in Figure 4.5

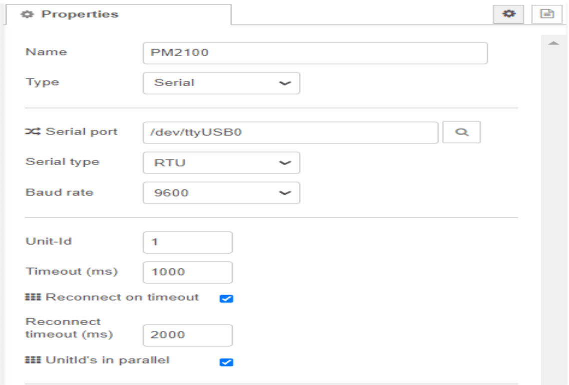


Settings | Optionals

Name: (Nama Parameter)
Topic: Topic
Unit-Id: 1
FC: FC 3: Read Holding Registers
Address: Adress Bus
Quantity: 2
Poll Rate: 10 second(s)
Delay on start: □
Server: PM2100

Fig. 20. Modbus Node Configuration

The Modbus node configuration consists of parameter names, Unit id (ID number on PM2100), Function Code (Using a Holding Register according to PM2100 Specifications), Address Bus (parameter address code measured according to specifications), Quantity (amount of data buffer taken from PM2100), Poll Rate (request Data on PM2100), and Modbus RS485 server configuration.



Properties

Name: PM2100
Type: Serial
Serial port: /dev/ttyUSB0
Serial type: RTU
Baud rate: 9600
Unit-Id: 1
Timeout (ms): 1000
Reconnect on timeout: ☑
Reconnect timeout (ms): 2000
UnitId's in parallel: ☑

Fig. 21 Server configuration on Modbus Nodes

Based on Figure. 21, the server configuration consists of the server name, communication type, Serial Port (port connected to the USB converter on the Raspberry Pi), Serial Type, Baud rate, Unit Id (Number Id on PM2100), Time Out, and Request Time Out.

2. Segment 2: use the Node function which functions to convert data from the buffer data type to a decimal number data type
3. Segment 3: using the Node template which functions to make the data form for each parameter into a JavaScript Object Notation (JSON) data format. The JSON data format has a "key": "value" structure, where the key value is configured on the template node, and the value is obtained from the value of each parameter requested.
4. Segment 4: uses Node join which functions to combine 7 parameters with the JSON data type.
5. Segment 5: use the Node template which functions to organize JSON data resulting from the merger of segment 4.

6. Segment 6: using the MQTT Publish Node which functions to send data from segment 5 using the MQTT publish protocol with server configuration.
7. Segment 7: uses a debug node that functions to display data connected to this node, namely data from segment 5.

B. SYSTEM TEST RESULT

a) Validation Results of Electrical Parameter Values on the IoT Gateway

Validation is carried out by comparing the value of the reading value on the IoT Gateway with the value on the Display Power Meter for 1 minute with the duration of sending data every second. The value on the Power Meter display can be seen in Figure. 22

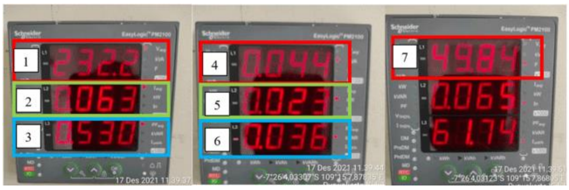


Fig. 22 Server configuration on Modbus Nodes

In Figure .22 it can be seen that the parameter values are based on box number:

1. Voltage (Average 3 phase)
2. Current (Average 3 phase)
3. Power Factor (Average 3 phase)
4. Apparent Power
5. Active Power / Real power
6. Reactive Power / Reactive power
7. Frequency

Based on the measurement results, the researcher obtained 50 data on measurements with a duration of 1 minute as in Appendix 1, and produced an average deviation between the data on the Power Meter and the IoT Gateway as shown in Table 4.5.

Table. 3. Average deviation of each Validation parameter (PM2100 and IoT Gateway)

Iavg (A)	VLN Avg (A)	PF	P Act (kW)	P React (kVAR)	P App (kVA)	Freq (Hz)
0,006	0,221	0,088	0,005	0,000	0,004	0,072

Based on Table 4.5, the current value (3-phase average) has an average deviation of 0.006 Amperes, Voltage (3-phase average) has an average deviation of 0.221 V, Power factor has an average deviation of 0.088, Active power has an average deviation of 0.005 kW, Reactive power has an average deviation of 0.000 kVAR, Apparent power has an average deviation of 0.004 kVA and frequency has an average deviation of 0.072 Hz. The results of these deviations can be caused by several factors such as the difference in time when collecting data between the Power meter and the IoT Gateway on a scale of seconds, fluctuations in electrical parameters that cannot be controlled, and human error when measuring.

b) End to End Electrical Parameter Value Validation Results

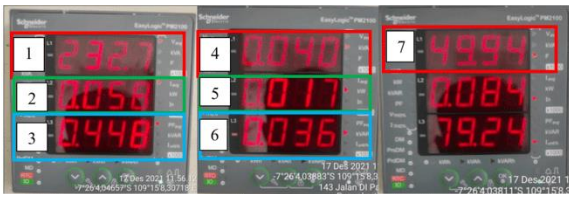


Fig. 23 Electrical value on Display Power Meter (End to End Measurement)

In Figure.23 it can be seen that the parameter values are based on the box number:

1. Voltage (Average 3 phase)
2. Current (Average 3 phase)
3. Power Factor (Average 3 phase)
4. Apparent Power
5. Active Power / Real power
6. Reactive Power / Reactive power
7. Frequency

The data on the dashboard is displayed in the form of a gauge meter for each measured parameter. The dashboard can only be accessed on a computer or PC that has node-red software installed and also has a flow that has been configured for the needs of the User Interface (user interface) so that it makes it easier for users to view data from power meter readings.

Table. 4. Average deviation of each parameter Validation (PM2100 and dashboard)

Iavg (A)	VLNAvg (A)	PF	P Act (kW)	P React (kVAR)	P App (kVA)	Freq (Hz)
0,001	0,519	0,012	0,000	0,000	0,000	0,124

Based on Table 4., the current value (3-phase average) has an average deviation of 0.001 Amperes, Voltage (3-phase average) has an average deviation of 0.519 V, Power factor has an average deviation of 0.012, Active power has an average deviation of 0.000 kW, Reactive power has an average deviation of 0.000 kVAR, Apparent power has an average deviation of 0.000 kVA and frequency has an average deviation of 0.124 Hz. The results of these deviations can be caused by several factors such as the time difference when collecting data between the Power meter and the dashboard on a scale of seconds, uncontrollable fluctuations in electrical parameters, end-to-end packet travel delays, and human error during measurement. Based on the test results, it can be concluded that the electricity monitoring system was successfully implemented because the data from the power meter was successfully sent to the internet with values that almost match the data on the dashboard with deviations as shown in Table 4.

c) Quality of Service (QoS) Test Results

1) Delay

From the data captured on Wireshark that has been exported in CSV form, then the data is processed by performing calculations. Calculations are made using the delay formula (5) which has been explained in the previous chapter, namely:

The following is an example of calculating the delay in the MQTT protocol using formula (5), to calculate the delay:

Table. 5. MQTT Protocol Data Sample :

NO	Time	Source	Destination	Protocol
1	0	52.202.162.140	100.100.100.253	TCP
2	2,009989	34.236.123.240	100.100.100.253	TCP
3	2,009893	34.236.123.240	100.100.100.253	TCP
4	2,009917	34.236.123.240	100.100.100.253	TCP
5	2,010304	34.236.123.240	100.100.100.253	TCP

Table. 6. Calculation of Delay on Ms. Excel

t2	t1	Delay
2,009989	0	2,00999
2,009893	2,009989	-0,00010
2,009917	2,009893	0,00002
2,010304	2,009917	0,00039
Delay Avg :		0,50258

The first step is to copy the time data into t1 and t2, then do the calculation by reducing the value of t2 with t1. We call the result of the difference a delay per data. Then calculate the average delay from the existing data, namely using the average function on Ms. Excel. Then the data will automatically appear what is the average value of the delay.

a. Publish Method

Table. 7. Calculation of Delay on Ms. Excel

Range (m)	Publish Delay (ms)			
	44 Byte	56 Byte	80 Byte	128 Byte
5	1,942	1,942	1,942	1,949
10	1,943	1,942	1,942	1,942
15	1,942	1,950	1,942	1,942
20	1,940	1,942	1,945	1,942

When compared with the TIPHON standard, the quality of service results for the delay parameter produced by the MQTT publish protocol is very good. It can be seen in Table .7 that the average delay value generated by the MQTT protocol in the publishing method has a value below 150 ms, which indicates that the delay is very good.

b. Subscription Method

Based on the results of measuring the average delay on the subscribe method with variations in data size and distance, it can be seen in the table

Table. 8. Average delay of the subscribe method

Range (m)	Subscribe Delay (ms)			
	44 Byte	56 Byte	80 Byte	128 Byte
5	1,943	1,943	1,945	1,941
10	1,941	1,942	1,944	1,942
15	1,941	1,956	1,964	1,960
20	1,958	1,942	1,942	1,957

When compared with the TIPHON standard, the results of the delay parameter quality of service produced by the MQTT subscribe protocol are very good. It can be seen in Table .8 that the average delay value generated by the MQTT protocol on the subscribe method has a value below 150 ms, which indicates that the delay is very good.

2) Jitter

Table. 9. Average delay of the subscribe method

delay2	delay1	Jitter
2,08102	0,088535	1,99249
0,088535	0,010317	0,07822
0,010317	0,019283	0,00897
Total Jitter :		2,07967
Jitter Avg :		0,693223

From the jitter calculation sample above, it can be seen that the formula for finding a jitter is the value of delay 2 - delay 1. Then after looking for the delay value between packets, then do the calculation of the total delay. Only then can the jitter value be searched by entering formula (6), namely the total delay divided by the packet received minus 1.

a. Publish Method

Table. 10. Average delay of the subscribe method

Range (m)	Publish Jitter (ms)			
	44 Byte	56 Byte	80 Byte	128 Byte
5	0,071	0,071	0,071	0,073
10	0,128	0,130	0,133	0,135
15	0,072	0,077	0,072	0,073
20	0,081	0,268	0,265	0,136

When compared with the TIPHON standard, the results of the quality of service parameter jitter produced by the MQTT publish protocol are good. It can be seen in Table .10 that the average jitter value generated by the MQTT protocol in the publishing method has a value below 1 ms, which indicates that the jitter is good.

b. Subscription Method

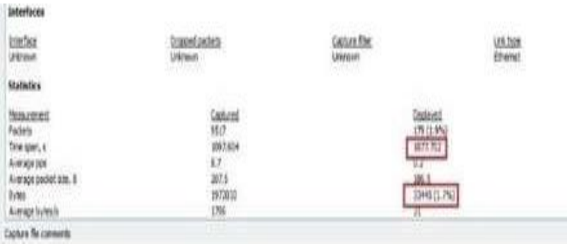
Table. 11. Average jitter of subscribe method

Range (m)	Subscribe Jitter (ms)			
	44 Byte	56 Byte	80 Byte	128 Byte
5	0,111	0,131	0,148	0,139
10	0,099	0,137	0,178	0,097
15	0,118	0,136	0,112	0,103
20	0,094	0,105	0,100	0,096

When compared with the TIPHON standard, the results of the quality of service parameter jitter produced by the MQTT subscribe protocol are good. It can be seen in Table .11 that the average jitter value generated by the MQTT protocol in the Subscribe method has a value below 1 ms, which indicates that the jitter is good.

3) Throughput

The throughput value can be seen in the average bytes in Wireshark because the value of the average byte is the result of the division between the bytes and the TimeSpan. However, the results obtained from average bytes are still in bytes, so if you want to get results in bits per second (bps), you can perform calculations using equation (8).



Interfaces

Interface	Dropped packets	Capture filter	Link type
Unknown	Unknown	Unknown	Ethernet

Statistics

Measurement	Captured	Displayed
Packets	9517	179 (1.9%)
Time span, s	1097.604	1077.752
Average pps	8.7	0.2
Average packet size, B	207.5	186.1
Bytes	1972010	33440 (1.7%)
Average bytes/s	1796	31

Capture file comments

Fig. 22 Throughput Calculations in Wireshark

a. Publish Method

Table. 12. Average jitter of subscribe method

	Publish Throughput (bps)			
Range (m)	**44 Byte**	**56 Byte**	**80 Byte**	**128 Byte**
5	519,012	568,470	667,262	779,909
10	518,912	568,426	667,255	782,763
15	519,045	566,232	667,287	782,719
20	519,592	568,512	666,268	782,733

It can be seen in Table .12 that at each distance, the larger the data packet sent, the greater the measured throughput. When compared with the TIPHON standard, the results of the quality of service parameter throughput produced by the MQTT publish protocol are very good. It can be seen in Table .12 that the throughput value generated by the MQTT protocol in the publishing method has a value above 100 bps, which indicates that the throughput is very good.

b. Subscription Method

Table. 13. Subscribe method throughput

	Subscribe Throughput (bps)			
Range (m)	**44 Byte**	**56 Byte**	**80 Byte**	**128 Byte**
5	469,316	518,694	616,932	733,533
10	469,780	519,128	617,158	733,400
15	469,800	515,327	611,004	726,685
20	465,877	519,010	617,818	727,818

It can be seen in Table .13 that at each distance, the larger the data packet sent, the greater the measured throughput. When compared with the TIPHON standard, the results of the quality of service parameter throughput produced by the MQTT subscribe protocol are very good. It can be seen in Table .13 that the throughput value generated by the MQTT protocol on the subscribe method has a value above 100 bps, which indicates that the throughput is very good.

4) Packet Loss

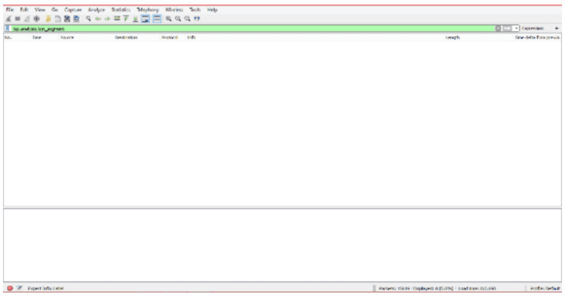


Fig. 23 Packet Loss

The picture above shows that there are no lost packets. Whereas previously in the throughput, jitter, and delay tests, each test of the publish and jitter methods continued to experience increases and decreases in network quality, but in each of these tests there was no packet loss or lost packet delivery. This is due to the fact that in data transmission there is no reciprocal ACK data from the platform to Wireshark, so the resulting data packet loss is 0%. In addition, packet loss also usually occurs in sending data with large data sizes, this will only reduce packet size, not packet loss.

a. Publish Method

Table. 14. Packet loss publish method

	Publish Packet Loss (%)			
Range (m)	**44 Byte**	**56 Byte**	**80 Byte**	**128 Byte**
5	0	0	0	0
10	0	0	0	0
15	0	0	0	0
20	0	0	0	0

When compared with the TIPHON standard, the results of the quality of service parameter packet loss produced by the MQTT publish protocol are very good. It can be seen in Table .14 that the packet loss value generated by the MQTT protocol in the publishing method has a value of no more than 0%, which indicates that the packet loss is very good.

b. Subscription Method

Table. 15. Packet loss subscribe method

	Subscribe Packet Loss (%)			
Range (m)	**44 Byte**	**56 Byte**	**80 Byte**	**128 Byte**
5	0	0	0	0
10	0	0	0	0
15	0	0	0	0
20	0	0	0	0

When compared with the TIPHON standard, the results of the quality of service parameter packet loss generated by the MQTT subscribe protocol are very good. It can be seen in Table .15 that the packet loss value generated by the MQTT protocol on the subscribe method has a value of no more than 0%, which indicates that the packet loss is very good.

4. CONCLUSION

Based on the discussion regarding the Implementation and Analysis of the Internet of Things System for Monitoring Electrical Energy at the Telkom Purwokerto Institute of Technology, several conclusions can be obtained as follows:

1. Researchers have succeeded in designing an Internet of Things System for Monitoring Electrical Energy at the Telkom Purwokerto Institute of Technology because electrical parameter data can be viewed via the Dashboard via the Internet.
2. The results of the design of the Internet of Things System for Monitoring Electrical Energy

at the Telkom Institute of Technology Purwokerto can be implemented properly which is marked by the successful transmission of End to end data with a current value (3 phase average) have an average deviation of 0.006 Amperes, Voltage (average 3 phase) has an average deviation of 0.221 V, Power factor has an average deviation of 0.088, Active power has an average deviation of 0.005 kW, Reactive power has an average deviation of 0.000 kVAR, Apparent power has an average deviation of 0.004 kVA and the frequency has an average deviation of 0.072 Hz.

3. Service quality in the delay, throughput, and packet loss categories for the publish and subscribe methods has an index value of 4 according to the TIPHON standard, while the jitter category for the publish and subscribe method has an index value of 3 according to the TIPHON standard.

REFERENCES

[1] Jawa Pos, “No Title,” Jalan Industri 4.0, 2018. https://www.pressreader.com/indonesia/jawa-pos/20180406/282870846385944 (accessed Apr. 06, 2021).

[2] G. B. C. Indonesia, “Konsumsi Energi di Sektor Pendidikan,” 2017. http://www.gbicindonesia.org/19-green-at-your-school.

[3] R. T. Hudan, Ivan Safril, “Rancang Bangun Sistem Monitoring Daya Listrik Pada Kamar Kos Berbasis Internet of Things (Iot),” J. Tek. ELEKTRO, vol. 08, no. 01, pp. 91–99, 2019.

[4] J. Lianda, D. Handarly, and A. Adam, “Sistem Monitoring Konsumsi Daya Listrik Jarak Jauh Berbasis Internet of Things,” JTERA (Jurnal Teknol. Rekayasa), vol. 4, no. 1, p. 79, 2019, doi: 10.31544/jtera.v4.i1.2019.79-84.

[5] D. Wulandari, “Warga Muba bisa Cek dan Isi Token Listrik via Aplikasi Muba Listrik Pintar,” Bisnis .com, 2020. .

[6] I. Nurdin, "Sistem Tenaga Listrik: Teori, Konsep, dan Aplikasi," Yogyakarta, Deepublish, 2019.

[7] S. Tyagi and P. C. Jain, “Internet of Things using LPWAN,” no. January 2019, [Online]. Available: https://www.researchgate.net/publication/330352019_Internet_of_Things_using_LPWAN.

[8] M. Mehta, “ESP8266 : A Breakthrough in Wireless Sensor Networks and Internet of Things,” Int. J. Electron. Commun. Eng. Technol., vol. 6, no. 8, pp. 7–11, 2015, [Online]. Available: www.iaeme.com/IJECET/index.asp.

[9] A. Nugroho, “Mengenal Protokol MODBUS,” 2020. https://automation.or.id/2020/01/05/mengenal-protokol-modbus-bagian-1/ (accessed Apr. 06, 2021).

[10] A. Banks, " MQTT Essentials - A Lightweight IoT Protocol," MQTT, 2019. [Online]. Available: https://mqtt.org/. [Accessed 20 Mei 2023].

[11] S. A. Malik, I. A. Khan and S. Zeadally, "Internet of Things (IoT): Architecture, Protocols, and Security Issues.," 2020.

[12] A. Bahadori and M. Uddin, "Quality of Service for 5G and Beyond Networks: Innovations and Challenges," IEEE Communications Magazine, vol. 58, no. 1, pp. 40-47, 2020.

[13] R. Z. Pratama and H. Nurwarsito, “Monitoring Penggunaan Daya Listrik menggunakan Protokol MQTT berbasis Web,” J. Pengemb. Teknol. Inf. dan Ilmu Komput., vol. 3, no. 11, pp. 10820–10826, 2019.

[14] B. Cahyono, A. Budijanto, and Y. Alif, “Prototipe Panel Monitoring Lampu Listrik Terpusat Menggunakan Komunikasi Rs485,” pp. 1–6, 2017.

[15] I. Harjanto, “IoT Gateway Menggunakan Protokol MQTT pada Perangkat Kendali Berbasis Modbus-RTU,” vol. VI, no. 1, pp. 12–19, 2020.

[16] E. H. Wiguna and A. Subari, “Rancang Bangun Sistem Monitoring Ketinggian Air Dan Kelembaban Tanah Pada Penyiram Tanaman Otomatis Dengan Hmi (Human Machine Interface) Berbasis Raspberry Pi Menggunakan Software Node-Red,” Gema Teknol., vol. 19, no. 3, p. 1, 2017, doi: 10.14710/gt.v19i3.21878.

[17] S. Electric, “EasyLogic TM PM2100 series,” pp. 1–78, 2016.

[18] Schneider Electric, “PowerLogic Electrical network management,” pp. 79–93, 2018.

[19] H. Arijuddin, A. Bhawiyuga, and K. Amron, “Pengembangan Sistem Perantara Pengiriman Data Menggunakan Modul Komunikasi LoRa dan Protokol MQTT Pada Wireless Sensor Network,” Pengemb. Teknol. Inf. dan Ilmu Komput., vol. 3, no. 2, pp. 1655–1659, 2019.