On-Grid Photovoltaic System Power Monitoring Based on Open Source and Low-Cost Internet of Things Platform

Hoedi Prasetyo
Politeknik ATMI Surakarta, Program Studi Teknologi Rekayasa Mekatronika

https://doi.org/10.5109/4372265

出版情報：Evergreen. 8 (1), pp.98-106, 2021-03. Transdisciplinary Research and Education Center for Green Technologies, Kyushu University
バージョン：

On-Grid Photovoltaic System Power Monitoring Based on Open Source and Low-Cost Internet of Things Platform

Hoedi Prasetyo*

Politeknik ATMI Surakarta, Program Studi Teknologi Rekayasa Mekatronika, Jl. Mojo/Jl. Adisucipto No.1, Karangasem, Laweyan, Surakarta, 57145, Jawa Tengah, Indonesia

*Author to whom correspondence should be addressed:
E-mail: hoedi.prasetyo@atmi.ac.id

(Received October 7, 2020; Revised March 18, 2021; accepted March 25, 2021).

Abstract: There is a lot of research that proposed PV monitoring systems based on the low-cost IoT technology. However, there is a lack of implementations in the On-grid PV systems. This research aims to fill this gap, by proposing a novel design of On-grid PV system power monitoring. The design utilizes a CT sensor, NodeMCU, MQTT, Node-RED, and Raspberry Pi. The hardware cost is affordable. SQLite is introduced to reduce the database complexity. A trial has been accomplished through a case study of 3.6 kWp On-grid PV system. The designed dashboard consists of two pages. The first page displays summary information, and the second page serves access to the historical data. Based on the test, the current readings accuracy is acceptable. The proposed design is applicable to residential On-grid PV systems in urban areas. This research may become a potential support for the public awareness of renewable energy, especially in the developing country, where financing becomes a key challenge.

Keywords: On-grid, Power Monitoring; Internet of Things; NodeMCU; Node-RED; SQLite

1. Introduction

IoT (Internet of Things) refers to a system of interrelated things (computer, machine, people, etc.) that can transfer data over a network automatically[1]. The utilization of internet has become an integral part of modern life. A study from Imansyah[2] predicts that internet penetration in Indonesia, as a developing country, is increasing exponentially. Many business activities are now going online. These include the field of renewable energy, especially PV (photovoltaic) system. PV system refers to a technology that able to convert sunlight to electricity. Due to the proximity to the equator, Indonesia has a profound potential solar energy. The solar intensity in Indonesia is very constant over the year.

In a PV system, IoT can be implemented primarily as a monitoring application. According to study by Kumar et al[3], implementation of IoT in the PV system delivers several benefits. IoT reduces the tedious job of visiting the plant site frequently. IoT enables the monitoring application to perform continuous record of performance and failure data for analytics, forecasting, predicting the future power, income generation, and even managing the controllable loads in the house autonomously and remotely[4]. Users can also easily identify the faults and reasons for low performance in effective manner without putting many efforts.

There are many IoT-based PV systems monitoring, which are commercially offered and sold in the market. Some of these are included in the purchasing package. Some are separately sold as add on accessories. All of these commercial technologies are coming with a full range of services. However, the costs associated with their installation, licensing, and maintenance have to be carefully considered[5]. There are PV users who decide not to procure these commercial technologies due to budget consideration. Fortunately, thanks to the open-source platform, plenty of research have attempted to provide low-cost solutions for implementing IoT in PV systems. Table 1 shows the list of some related research papers.

To perform monitoring, sensors are required to sense and measure the physical quantities, and then convert the data they receive into electrical signals. In a PV system, the major physical quantities required to sense are voltage and current. For the DC (Direct Current) voltage sensing, purchasing a simple and low-cost voltage divider-based sensor would be a reasonable option, as done by most of the papers on the list[6,7,8]. For the current sensing, the invasive type of current sensor, for example ACS712 [6,7] and INA219[9,10,11], is mostly used in the list. Once the voltage and the current data is acquired, the power quantity can be calculated. Beside the voltage and current, other sensors, such as temperature and humidity, can also be added[12].

Table 1. Related papers

Reference	Current sensor	Data acquisition	Server & Database	Dashboard	PV system
Adhya et al[17]	Invasive	PIC18F46K22, GSM-GPRS	Desktop PC	Local web	Off-grid
Kekre & Gawre[6]	Invasive	Arduino, GSM-GPRS	Desktop PC	Local web	Off-grid
Gupta et al[13]	Invasive	MCP3008-Rasp Pi, Serial	Desktop PC	LabView	Off-grid
Othman et al[7]	Invasive	MCP3008, Serial	Rasp Pi	Node-RED	Off-grid
Deshmukh & Bhuyar[8]	Invasive	MCP3008, Serial	Rasp Pi, Cloud	Ubidots	Off-grid
Choi et al[18]	Invasive	Arduino, LoRa	Rasp Pi, MongoDB	Local web	Off-grid
Hamied et al[19]	Invasive	Arduino-ESP, WiFi (HTTP)	Desktop PC	Local web	Off-grid
Priharti et al[23]	Invasive	Arduino-ESP, WiFi (HTTP)	Cloud	Thinkspeak	Off-grid
Aghenta & Iqbal[15]	Invasive	Arduino, Serial	Rasp Pi	EmonCMS	Off-grid
Fadel et al[16]	Invasive	Arduino, Serial	Rasp Pi	Cloud Node-RED	Off-grid
Oukennou et al[9]	Invasive	ESP, MQTT	Desktop PC	Node-RED	Off-grid
Rouibah et al[25]	Invasive	Arduino-ESP, WiFi (TCP/IP)	Desktop PC, MySQL	Local web	Off-grid
Ali & Paracha[12]	Invasive	Arduino-ESP, WiFi (HTTP)	Cloud	Adafruit	Off-grid
Cheddadi et al[10]	Invasive	ESP, WiFi (HTTP)	Desktop PC, InfluxDB	Grafana	Off-grid
This research	Non-invasive	ESP, WiFi (MQTT)	Rasp Pi, SQLite	Node-RED	On-grid

The data generated by sensors is transmitted to a server. This requires data acquisition process. This process has two steps. The first step is to convert the analog signal from sensors to digital signal. The second step is to transmit the data. Some papers[7,8,13] propose a combination of Rasp Pi (Raspberry Pi) with MCP3008-ADC (Analog to Digital Converter). Rasp Pi is a low-cost, small, and portable size of computer board[14]. This combination seems to be bulky inasmuch as the utilization of Rasp Pi would be extravagant. The other papers[15,16] suggest the use of Arduino, since it is smaller than Rasp Pi, and already equipped with ADC. Either Rasp Pi or Arduino requires wired serial communication to transmit the data to a server. The wired communication certainly limits the distance between the sensors and the server. To deal with this limitation, some works recommend to combine Arduino with wireless modules, such as SIM900 GPRS module[6,17], LoRa modem[18], and NodeMCU[19,20]. According to Pereira et al[21], the coverage area of GPRS modules and LoRa modems are much better than NodeMCU, while the price, conversely, are more costly.

NodeMCU, such as ESP8266 and ESP32, is a microcontroller with a wireless (WiFi and Bluetooth) module inside. It can communicate with other devices wirelessly, and even be programmed remotely. Owing to the fact that its performance is improved considerably, a single NodeMCU connected to a WiFi is sufficient to perform data acquisition process with a coverage of 100 meters[21]. The protocol for data transmission can be performed using HTTP (Hyper Text Transfer Protocol)[10] or MQTT (Message Queuing Telemetry Transport)[9,22].

A server subsequently processes the transmitted data, so that the pertinent information can be delivered to users. The server shall be capable of handling and processing a large number of data. The server can be a desktop computer, Rasp Pi, or cloud service. Using a desktop computer as server can be a plausible option, since it has a high computation power[6,13,17,19]. However, the physical size, the cost, and the ability to deal with power outage should be considered. The other option for the server is a cloud service[11,23]. It completely does not require any space and has plenty of features to offer. A reliable internet connection is crucial to ensure the data transmission keeps running. Some premium features in a cloud service are often limited, and hence a paid subscription is required. Rasp Pi can be considered as an notable option to save computing space and costs[7,8].

One primary functionality of a server is to visualize the information on a dashboard. The information can be presented in the forms of text, number, gauge, or chart. A tool is required to design this dashboard. A conventional website locally installed on a computer can be used as the dashboard[19]. Its extensibility and customization are excellent, yet it requires a proficient web programming skill. Employing the cloud-IoT platform, such as Ubidots[8], Thingspeak[11,23], and Adafruit[12], is commonly considered as the simplest way to design the dashboard. However, the number of data and the displayed information are usually limited, except for the paid subscriptions. An emerging programming tool to create the dashboard is Node-RED[7]. Node-RED is an open-source flow-based programming tool. It is developed based on a run time implemented in JavaScript using the Node.js framework[24]. To extend the functionality of the dashboard, a database utilization can be employed. The options for the database vary from the traditional type, such as MySQL[25], to the more modern ones, such as MongoDB[18] and InfluxDB[10].

There are at least three research gaps in Table 1. The first gap is none of the listed papers implemented their IoT design in On-grid PV systems. It is hard to find any research that implemented the low-cost IoT design in On-grid PV systems. According to study by Mendu et al[26], based on the techno-economic comparative analysis, On-grid PV systems is preferable than Off-grid PV systems. On-grid PV systems are able to generate electricity that can be connected safely to the electric grid. The system generally consists of solar panels, an On-grid inverter, and electrical safety components. These systems do not require any battery, which is still considered as the most costly component in the photovoltaic systems. These systems are commonly installed in urban areas and have primary purpose to save electricity cost. The second gap is none of the listed papers considered to display a cost-saving information. The cost-saving information plays

important role in ensuring the income generation from PV systems. On top of that, it allows a user to carry out economic analysis of a PV investment, for example, a payback period analysis[27]. The third gap is none of the papers recommends the use of non-invasive electric current sensor[28]. Most of them utilized ACS712 current sensor. It has open screws connectors for the mains input-output, which are not covered, and requires pin connectors for the signal and DC supply. For high voltage application, its installation is tricky, and absolutely requires high caution due to safety reasons.

The motivation of this research is to fill the mentioned gaps. This research aims to propose a low-cost IoT design implemented in On-grid PV system. The proposed design presents cost-saving information and utilizes a non-invasive current sensor to improve safety and simplicity. The low-cost design is manifested in the use of open platform hardware, namely NodeMCU and Rasp Pi. For the software, this research utilizes free and lightweight applications, namely MQTT and Node-RED. In addition, this research introduces the use of SQLite[29] for the database. It features ample simplicity but is rarely used in IoT design. This research contributes to help the On-grid PV system users to build their own low-cost IoT power monitoring. This research follows the typical IoT waterfall methodology[30] that involves three steps, namely the design of the system; the system development; and the implementation. A case study is provided to show how the proposed design in this research is implemented.

2. Design of the system

The case study for this research is a 3.6 kWp (kilowatt peak) On-grid PV system. It is located at the front yard campus of Politeknik ATMI Surakarta. The PV system has been running since 2007. It has no wireless monitoring application. As shown in Fig.1, the PV system has 3 arrays of solar panel. Each array generates up to 1.2 kWp of DC power. The power from the solar panels is connected to a Sunny Boy 3800 On-grid inverter, which can produce up to 3.8 kW of AC power. The nominal AC voltage is 220V. To aptly measure the current, the appropriate capacity of the current sensor should be determined by dividing 3.6 kWp by 220V, which results in 16 A. Because of the PV efficiency factor, the current capacity can be reduced up to 80%, which is 13 A.

Fig. 1: The PV panel arrays in the case study.

An interview with the campus building manager, which is the main customer for this research, was performed to identify the requirements. From this initial work, a set of dashboard requirements were identified. The dashboard should be displayed in a kiosk and contain the information of:

- power;
- highest power record;
- how long the system has been running;
- total of energy-saving;
- total of cost-saving in Indonesian Rupiah (Rp);
- average cost-saving;
- hourly energy records.

Fig. 2 shows the diagram of the proposed design. Solar panels generate DC power. The On-grid inverter converts this DC power into AC power, and then synchronizes the output waveform and frequency with the grid standard. A non-invasive current sensor measures the electric current, which flows from the inverter to the grid. The NodeMCU converts the sensor readings into digital data. The NodeMCU wirelessly transmits the data to the local server, which is Rasp Pi, by MQTT protocol. The use of MQTT protocol for data transmission is recommended, because it is remarkably accurate and reliable[31,32]. The database stores the data periodically. The Node-RED, which is installed in the Rasp Pi, process the data and then displays the required information on the dashboard. The combination of MQTT and Node-RED is selected, because its performance is already acceptable for industrial purposes[33]. The users can see the dashboard directly on a kiosk or access it wirelessly by their computers. To access the dashboard wirelessly, connecting to the local server's IP address through the Node-RED standard port (1880), is required.

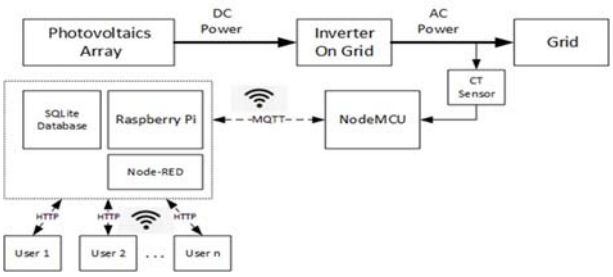

Fig. 2: The proposed design.

3. System development

The non-invasive current sensor used in this research is YHDC SCT-013-015[34]. It is a split core CT (Current Transformer) sensor. It has a capacity of current readings up to 15 A. This sensor costs around $ 5. Fig. 3 shows how the CT sensor is installed. ESP32 is selected as the NodeMCU due to its flexibility[35]. This NodeMCU costs only $ 10. The output from the CT sensor is AC signal, yet

the NodeMCU can only accept DC input signal. Therefore, a simple DC bias/offset circuit is required[34]. A $ 42 Rasp Pi 3B+ is selected as the local server, thus the total cost for the hardware is about $ 57.

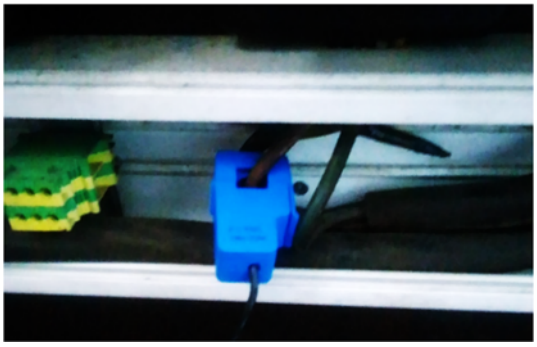

Fig. 1: Installation of CT sensor.

There is no software-cost in this research. This research uses MQTT broker application from Eclipse Mosquitto[36]. The Node-RED is commonly pre-installed in the Rasp Pi purchases. SQLite is selected for the database because of its simplicity. It works well in the devices that must operate without expert human support. SQLite is a good fit for use in the IoT[37].

3.1 NodeMCU algorithm

The NodeMCU functions to read, process, and send the data to the local server. The data consist of power, energy, and the highest power record. The power data is transmitted every second, whereas the others are transmitted every five minutes. The program algorithm can be seen in Fig. 4.

At the beginning of the program, all the required libraries and variables are initialized. The WiFi connection, MQTT server, and ADC pin are subsequently set up. The "millis ()" function is used to trigger the current sensor reading every second. Once the current data is acquired, Eq.1 is used to calculate the power in W. The voltage value is determined to be a constant value of 220V. This value is the standard for AC grid voltage in Indonesia. Through an MQTT topic entitled "pvsystem/realtime", the power data in JSON (JavaScript Object Notation) format, is transmitted.

$$power = current * voltage \quad (1)$$

If *power* is higher than *powerMax*, its recent value will be assigned to the *powerMax*. The variable *powerMax* functions to record the highest value of *power*. As shown in Eq.2, a discrete approximation is employed to compute energy. Energy is the summation of power (P) in a range of time (t).

$$energy = \sum_{i=1}^{t} P_i \quad (2)$$

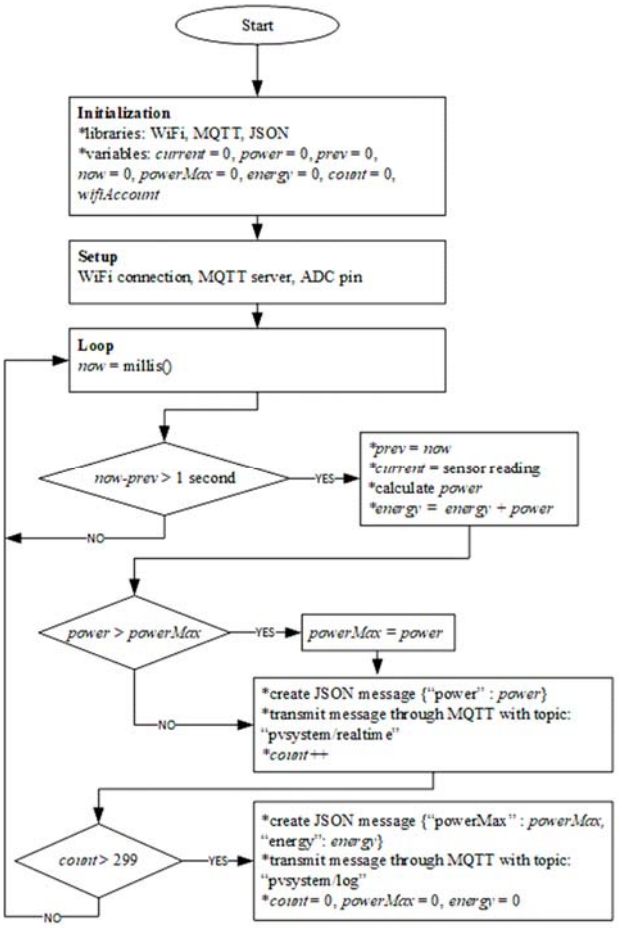

Fig. 2: NodeMCU algorithm flowchart.

In the algorithm, an increment operator can facilitate the computation. Every second, the *power* value is added to *energy*. In this research, the range of time is set for 300 seconds (5 minutes). Thus, the data of *energy* and *powerMax* is transmitted to the local server, through an MQTT topic entitled "pvsystem/log" every five minutes. There is no standard for how long the range of time should be set. Any range of time can be opted. However, two factors, which are the data storage utilization and the possibility of data loss, should be considered. If the range is too short, it will potentially burden the data storage with insignificant copious data. Conversely, if the range is too long, any occurrence of the data loss during transmission may affect the accuracy of information. Based on those considerations, a five-minute intermittent record is arguably reasonable.

3.2 Node-RED flow algorithm

A program in Node-RED is merely connections of a predefined code block, known as "node". A group of connected nodes is called a "flow". It is commonly a combination of input, processing, and output nodes. Based on the functions, the Node-RED algorithm in this research is divided into six parts.

The first part is the algorithm to display power. The power is presented into two formats, a numeric metric and a line chart. The input data comes from the MQTT topic "pvsystem/realtime". The data is then forwarded to two function nodes. A function node allows a customized JavaScript code to be added to the flow. The first function node process the data, and then sends it to a numeric dashboard node. The second function node generates an array data structure. The array is required to display the power data in a line chart. The algorithm can be seen in Fig. 5.

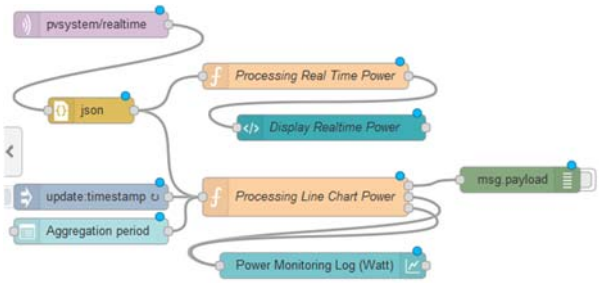

Fig. 3: Algorithm to display the power data.

An SQLite database, named "PV System DB", was created. It consist of two tables, the "datalog" and the "data_hourly". Fig. 6 shows the second part of the algorithm, which functions to store the data into table "datalog". This table contains three data elements, namely *highest power record*, *energy*, and *timestamp*. The input data comes from the MQTT topic "pvsystem/log". A function node containing SQL (Structured Query Language) codes is required to store the data.

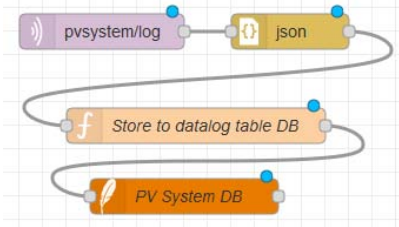

Fig. 4: Algorithm to update the table "datalog".

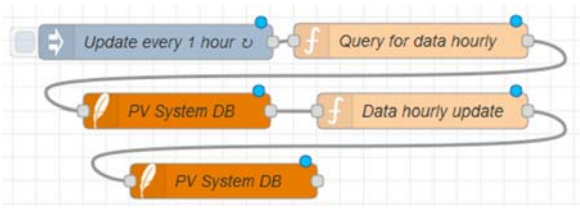

Fig. 5: Algorithm to update the table "data_hourly".

The third part is the algorithm to generate data for table "data_hourly". The table contains the hourly-highest power record and energy data. The first function node performs a query from the table "datalog". The second node forwards the query result to the table "data_hourly". This algorithm is triggered every hour by an inject node. The algorithm can be seen in Fig.7.

The fourth algorithm principally functions to display the numeric metrics. As shown in Fig. 8, all of the output nodes are the dashboard nodes. The input data comes from the "data_hourly" query result. Eq.3 is used to compute the *Total Energy* in kWh. It is the summation of energy (E) from the first row to the last row k in the table "data_hourly". This can be performed using "SUM()" function. Subsequently, as shown in Eq.4, the *Total Cost Saving* in Indonesian Rupiah (Rp), can be calculated by multiplying the *Total Energy* by the electricity price (*Rates*). Lastly, the the *Daily Cost Saving* is calculated using Eq.5.

$$Total\ \ Energy = \sum_{j=1}^{k} E_j \tag{3}$$

$$Total\ \ Cost\ \ Saving = Total\ \ Energy * Rates \tag{4}$$

$$Daily\ \ Cost\ \ Saving = \frac{Total\ \ Cost\ \ Saving}{Elapsed\ \ Time\ \ (in\ \ day)} \tag{5}$$

To format the numeric metrics, a function node precedes every dashboard node. This node makes the displayed metrics more readable. For example, adding a thousand and a million separator to the displayed number will prevent the user from misreading.

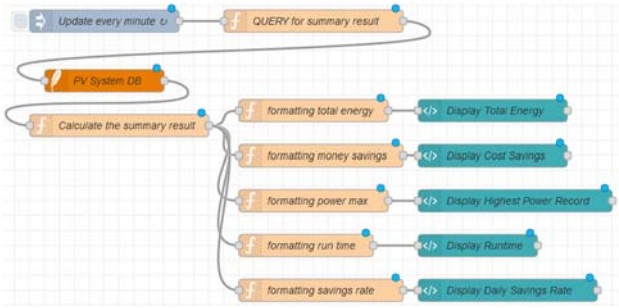

Fig. 8: Algorithm to process the numeric displays.

Fig.9 shows the fifth part of algorithm, whose purpose to display the hourly energy records in a bar chart. The input data comes from the table "data_hourly" query result. A function node process the query result to generate an array data structure, which is required by the dashboard node. This algorithm is triggered every hour by an inject node.

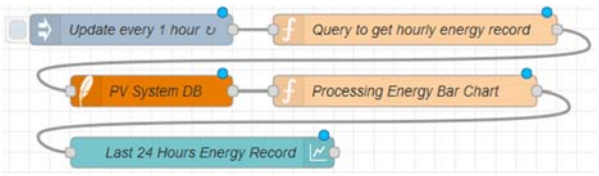

Fig. 9: Algorithm to display the hourly energy records.

The last part of the algorithm allows user to access the historical data. As shown in Fig.10, when the user inputs a specific date, it will trigger a database query. Before being forwarded to the output nodes, the data from the query result will be processed and formatted by several function nodes. There are three outputs for this algorithm, namely energy saving, cost saving, and hourly energy records on that specific date. The energy record is displayed in a bar chart, while the others are displayed in numeric metrics.

Fig. 10: Algorithm to access the historical data.

4. Implementation

The proposed power monitoring in this research has been running for 290 days. The stored data number in table "datalog" and "data_hourly" are around 79,000 and 7,000 entries, respectively. It has been consuming around 4 MB (Mega Bytes) of data storage. Based on the proposed power monitoring, the On-grid PV system has been saving 2,720 kWh of electricity with an average cost-saving around IDR 14,000 per day.

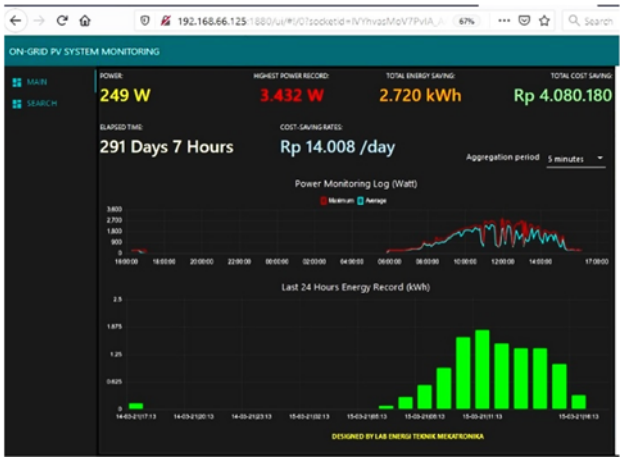

Fig. 10: The main page.

4.1 Dashboard

To fulfil the kiosk display requirement from the customer, a screen connected to the Rasp Pi is installed on a public space, in the campus building. Anyone can see the dashboard on this kiosk. The dashboard can be seen remotely on a desktop computer as well, by accessing the Rasp Pi IP (Internet Protocol) address. The computer should connect to the same WiFi as the Rasp Pi. The dashboard consists of two pages, the main page and the search page. As shown in Fig. 10, the display will automatically go to the main page, once a connection is established. The upper part of the main page displays the numeric metrics, namely *power*, *highest power record*, *total energy saving*, *total cost saving*, *elapsed time*, and *daily cost saving*. The *power* is updated every second and the others are updated every 5 minutes. The lower part displays two charts. The first chart shows the average and the highest power data, while the second chart shows the hourly energy data. Both are capable of visualizing the last 24 hours of data records.

Fig. 11: The search page.

The search page, as shown in Fig.11, serves the user to access the historical data. On this page, a date-picker allows the selection of a specific date and year. It displays a date input field by default. A dropdown calendar will appear when the user taps on the input field. Once a date is selected, two numeric metrics (energy and cost-saving) and an hourly energy bar chart for that selected date, are displayed.

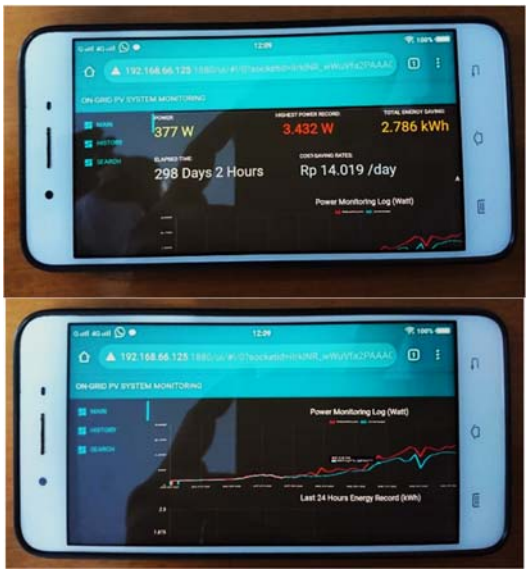

Fig. 12: The dashboard displayed on a smartphone.

As shown in Fig. 12, accessing the dashboard from a smartphone is possible. However, it would be less user-friendly, since the dashboard design is not mobile responsive[38]. The official Node-RED dashboard nodes have not yet supported the responsive design. To improve this limitation, an alternative dashboard node can be employed[39]. Using this alternative node requires adequate HTML (Hypertext Markup Language) and Bootstrap

programming skill[40].

4.2 Accuracy of the current readings

To check the current measurement validity, this research performed an accuracy test. Accuracy refers to the closeness of the current sensor reading to the actual value. The sensor reading was performed by a serial monitor, and at the same time, compared to the actual reading. As shown in Fig.13, the actual current reading was performed using a Kyoritsu 2033 digital clamp meter.

Fig. 13: Current reading using a clamp meter.

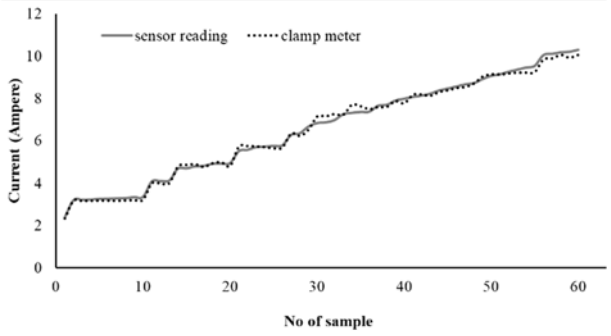

Fig. 14: Sensor vs clamp meter readings.

There are 60 samples taken from the test. Fig.14 shows the data plot from the samples. It can be seen that the two curves are approximately coincident. The accuracy was determined using percent error formula, as shown in Eq.6. Notation s and c are the sensor and the clamp meter readings, respectively, while n is the number of samples. The greater the error, the lower the accuracy.

$$percent\ \ error = \sum_{m=1}^{n} \frac{|s-c|}{c} * 100\% \qquad (6)$$

Table 2 shows the result of the accuracy test. There are three levels of current range. The level starts from 2A and has an interval range of 3A. The number of samples was distributed equally. This division into levels was intended to observe the relation between accuracy and the value of current. The error for the first, second, and third level are 2.48%, 2.25%, and 1.49%, respectively. It can be seen that error tends to be smaller when current gets higher. It presumably occurred because the absolute error was approximately constant, which is around 0.13A, among the levels of current. This finding indicates that the higher the current, the better the accuracy.

Table 2. Accuracy test result

Level	Range	Number of Samples	Mean of Absolute error	Percent Error
I	2 A – 5 A	20	0.09 A	2.48 %
II	5 A – 8 A	20	0.15 A	2.25 %
III	8 A – 11 A	20	0.14 A	1.49 %
Overall			0.13 A	2.07 %

The overall error is 2.07 %. It is slightly higher than the standard error of a CT sensor[34], which is 2.00 %. The overall error may be primarily influenced by the CT sensor accuracy. According to Alexe[28], the accuracy of the CT sensor is sufficient for the residential scale of power monitoring. Study from Sutisna et al[41] states that a microcontroller-based of power measuring device, with an average error below 5%, is reasonable. Thus, by considering the test result, the CT sensor accuracy, and the justification from other research, it can be said that the accuracy of the current readings in this research is acceptable.

5. Conclusion

There is a lot of research that proposed the PV monitoring systems based on the low-cost IoT technology. Most of them utilized the invasive current sensor, which is less safe for high power PV systems. In urban areas, On-grid PV system is preferable for its feasibility. However, there is a lack of research number in the On-grid PV system implementations. Availability of a simple cost-saving monitor would be an additional advantage for the PV system user. There is hardly any research works which considered to display the cost-saving information.

This research aims to fill those gaps, by proposing a novel design of On-grid PV system monitoring, based on the low-cost IoT platform. The hardware cost is plausibly affordable. Using Node-RED, for creating the dashboard with various forms of display, is uncomplicated[42]. SQLite, which is rarely used in the previous research, is introduced to reduce the database complexity. For residential purposes, the accuracy is satisfactory. This research may become a potential support for the public awareness of renewable energy, especially in the developing country, where financing becomes a key challenge[43].

There are some potential improvements for the future works. The dashboard can be remade to be mobile friendly. For those who prefer to promote the environmental influence, rather than the economic result, adding a carbon emission reduction display would be more meaningful[44]. The structure of the database can be expanded by adding a daily data record. This will allow the user to trace the data record with a wider range of time, for example in weeks or months.

References

1) O. Elijah, T.A. Rahman, I. Orikumhi, C.Y. Leow, and M.N. Hindia, "An overview of Internet of Things (IoT) and data analytics in Agriculture: Benefits and Challenges," *IEEE Internet of Things Journal*, **5**(*5*), 3758-3773 (2018). doi: 10.1109/JIOT.2018.2844296.
2) R. Imansyah, "Impact of Internet Penetration for the Economic Growth of Indonesia", *Evergreen*, **5**(*2*), 36-43 (2018). doi: 10.5109/1936215.
3) N. M. Kumar, K. Atluri, and S. Palaparthi, "Internet of Things (IoT) in photovoltaic systems", in *2018 National Power Engineering Conference (NPEC)*, Madurai, India, 1-4 (2018). doi: 10.1109/NPEC.2018.8476807.
4) G. D. Nugraha, B. Sudiarto, and K. Ramli, "Machine learning-based energy management system for prosumer", *Evergreen*, **7**(*2*), 309-313 (2020). doi: 10.5109/4055238.
5) N. Stroia, D. Moga, and Z. Barabas, "Web based monitoring of solar power systems", *IFAC Proceedings Volumes*, 46(6), 131-136 (2013). doi: 10.3182/20130522-3-RO-4035.00046.
6) A. Kekre and S. K. Gawre, "Solar photovoltaic remote monitoring system using IOT", in *2017 International conference on recent innovations in signal processing and embedded systems (RISE)*, Bhopal, India, 619-623 (2017). doi: 10.1109/RISE.2017.8378227.
7) N.A. Othman, M.R. Zainodin, N. Anuar, and N.S. Damanhuri, "Remote monitoring system development via Raspberry-Pi for small scale standalone PV plant," in *7th IEEE International Conference on Control System, Computing and Engineering*, Penang, Malaysia, 360-365 (2017) . doi: 10.1109/ICCSCE.2017.8284435.
8) N. S. Deshmukh and D. L. Bhuyar, "A Smart Solar Photovoltaic Remote Monitoring and Controlling", in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 67-71 (2018). doi: 10.1109/ICCONS.2018.8663127.
9) A. Oukennou, A. Berrar, I. Belbhar, and N. El Hamri, "Low Cost IoT System for Solar Panel Power Monitoring," in *Colloque sur les Objets et systèmes Connectés, Ecole Supérieure de Technologie de Casablanca (Maroc), Institut Universitaire de Technologie d'Aix-Marseille*, Casablanca, Morocco, (2019).
10) Y. Cheddadi, H. Cheddadi, F. Cheddadi, F. Errahimi, and N. Es-sbai, "Design and implementation of an intelligent low-cost IoT solution for energy monitoring of photovoltaic stations", *SN Applied Sciences*, **2**, 1-11 (2020). doi: 10.1007/s42452-020-2997-4.
11) M. Katyarmal, S. Walkunde, A. Sakhare, and M. U. Rawandale, "Solar power monitoring system using IoT", *Int Res J Eng Technol (IRJET)*, **5**(*3*), 3431-3432 (2018).
12) M. Ali and M. K. Paracha, "An IoT Based Approach for Monitoring Solar Power Consumption with ADAFRUIT Cloud", *Int. J. Eng. Appl. Sci. Technol*, **4**(*9*), 335-341 (2020).
13) A. Gupta, R. Jain, R. Joshi, and R. Saxena, "Real time remote solar monitoring system", in *2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA)(Fall)*, Dehradun, India, 1-5 (2017). doi: 10.1109/ICACCAF.2017.8344723.
14) C. W.Zhao, J. Jegatheesan, and S. C. Loon, "Exploring IoT Application Using Raspberry Pi", *International Journal of Computer Networks and Applications*, **2**(*1*), 27-34 (2015).
15) L. O. Aghenta and M. T. Iqbal, "Development of an IoT Based Open Source SCADA System for PV System Monitoring," in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, Edmonton, AB, Canada, 1-4 (2019). doi: 10.1109/CCECE.2019.8861827.
16) A. M. Fadel, A. M. Ibrahim, A. K. Tamazin, R. A. Hamdy, and A. S. Abdel-Khalik, "IoT-Based Power Management for DC Microgrids," in *2019 21st International Middle East Power Systems Conference (MEPCON)*, Cairo, Egypt, 1107-1111 (2019). doi: 10.1109/MEPCON47431.2019.9008216.
17) S. Adhya, D. Saha, A. Das, J. Jana, and H. Saha, "An IoT based smart solar photovoltaic remote monitoring and control unit", in *2016 2nd international conference on control, instrumentation, energy & communication (CIEC)*, 432-436 (2016). doi: 10.1109/CIEC.2016.7513793.
18) C. S. Choi, J. D. Jeong, I. W. Lee, and W. K. Park, "LoRa based renewable energy monitoring system with open IoT platform", in *2018 international conference on Electronics, Information, and Communication (ICEIC)*, Hawaii, USA, 1-2 (2018). doi: 10.23919/ELINFOCOM.2018.8330550.
19) A. Hamied, A. Mellit, M. A. Zoulid, and R. Birouk, "IoT-based experimental prototype for monitoring of photovoltaic arrays", in *2018 International conference on applied smart systems (ICASS)*, 1-5 (2018). doi: 10.1109/ICASS.2018.8652014.
20) A. C. Bento, "IoT: NodeMCU 12E X Arduino Uno, Results of an experimental and comparative survey", *International Journal of Advance Research in Computer Science and Management Studies*, **6**(*1*), 46-56 (2018).
21) R. I. Pereira, S. C. Jucá, and P. C. Carvalho, "IoT embedded systems network and sensors signal conditioning applied to decentralized photovoltaic plants", *Measurement*, **142**, 195-212 (2019). doi: 10.1016/j.measurement.2019.04.085.
22) U. Hunkeler, H. L. Truong, and A. Stanford-Clark,

"MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks", in *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*, Bangalore, India, 791-798 (2008). doi**:** 10.1109/COMSWA.2008.4554519.

23) W. Priharti, A. F. K. Rosmawati, and I. P. D. Wibawa, "IoT based photovoltaic monitoring system application", *Journal of Physics: Conference Series*, **1367**(*1*), p. 012069 (2019). doi: 10.1088/1742-6596/1367/1/012069.
24) M. Blackstock and R. Lea, "FRED: A Hosted Data Flow Platform for the IOT Built Using Node-RED," in *Proceedings of the 1st International Workshop on Mashups of Things and APIs*, Trento, Article No. 2, 1–5 2016. doi: 10.1145/3007203.3007214.
25) N. Rouibah, L. Barazane, A. Mellit, B. Hajji, and A. Rabhi, "A low-cost monitoring system for maximum power point of a photovoltaic system using IoT technique", in *2019 International conference on wireless technologies, embedded and intelligent systems (WITS)*, Fez, Morocco, 1-5 (2019). doi: 10.1109/wits.2019.8723724.
26) S. S. Mendu, P. Appikonda, A. K. Emadabathuni, and N. Koritala, "Techno-Economic Comparative Analysis between Grid-Connected and Stand-Alone Integrated Energy Systems for an Educational Institute", *Evergreen*, **7**(*3*), 382-395 (2020). doi: 10.5109/4068616.
27) B. P. Numbi and S. J. Malinga, "Optimal energy cost and economic analysis of a residential grid-interactive solar PV system-case of eThekwini municipality in South Africa", *Applied Energy*, **186**, 28-45 (2017). doi: 10.1016/j.apenergy.2016.10.048.
28) V. M Alexe, "Comparative study regarding measurements of different AC current sensors", in *2016 International Symposium on Fundamentals of Electrical Engineering (ISFEE)*, Bucharest, Romania, 1-6 (2016). doi: 10.1109/ISFEE.2016.7803152.
29) K. Yue, L. Jiang, L. Yang, and H. Pang, "Research of Embedded Database SQLite Application in Intelligent Remote Monitoring System," in *2010 International Forum on Information Technology and Applications*, Kunming, China, 96-100 (2010). doi: 10.1109/IFITA.2010.241.
30) D. Santos and J. C. Ferreira, "IoT power monitoring system for smart environments", *Sustainability*, **11**(*19*), p. 5355 (2019). doi: 10.3390/su11195355.
31) S. S. Prayogo, Y. Mukhlis, and B. K. Yakti, "The Use and Performance of MQTT and CoAP as Internet of Things Application Protocol using NodeMCU ESP8266", in *2019 Fourth International Conference on Informatics and Computing (ICIC)*, Semarang, Indonesia, 1-5 (2019). doi: 10.1109/ICIC47613.2019.8985850.
32) M. Kashyap, V. Sharma, and N. Gupta, "Taking MQTT and NodeMcu to IOT: Communication in Internet of Things," *Procedia Computer Science*, **132**, 1611-1618, (2018). doi: 10.1016/j.procs.2018.05.126.
33) M. Tabaa, B. Chouri, S. Saadaoui, and K. Alami, "Industrial Communication Based on Modbus and Node-RED", *Procedia computer science*, **130**, 583-588 (2018). doi: 10.1016/j.procs.2018.04.107.
34) Y. Albert, 2018, "Non-Invasive Sensor: YHDC SCT013-000 CT used with Arduino", https://www.poweruc.pl/blogs/news/non-invasive-sensor-yhdc-sct013-000-ct-used-with-arduino-sct-013, (accessed March 17, 2021).
35) J. C. D Lara, S. Gutierrez, and F. Rodríguez, "Low Cost Greenhouse Monitoring System Based on Internet of Things", in *2019 IEEE International Conference on Engineering Veracruz*, Veracruz, Mexico, **1**, 1-6 (2019). doi: 10.1109/ICEV.2019.8920502.
36) R. A. Light, "Mosquitto: server and client implementation of the MQTT protocol", *Journal of Open Source Software*, **2**(*13*), 265 (2017). doi: 10.21105/joss.00265.
37) Hipp, Wyrick & Company, Inc, "Appropriate Uses For SQLite", https://www.sqlite.org/whentouse.html, (accessed March 15, 2021).
38) F. Shahzad, "Modern and responsive mobile-enabled web applications", *Procedia Computer Science*, **110**, 410-415 (2017). doi: 10.1016/j.procs.2017.06.105.
39) Julian Knight, "node-red-contrib-uibuilder", https://flows.nodered.org/node/node-red-contrib-uibuilder, (accessed March 17, 2021).
40) BootstrapVue, https://bootstrap-vue.org/, (accessed March 17, 2021).
41) I. U. Sutisna, I. Usrah, N. Hiron, and A. Andang, "Power analyzer based arduino-uno validation using Kyoritsu KEW 6315 and Hioki 328-20", *IOP Conference Series: Materials Science and Engineering*, ***550***(*1*), 012024 (2019). doi: 10.1088/1757-899x/550/1/012024.
42) P. Degreef, D. Van Merode, G. Tabunshchyk, "Low-Cost, Open-Source Automation System for Education, with Node-RED and Raspberry Pi," in *International Conference on Remote Engineering and Virtual Instrumentation*, Düsseldorf, Germany, 458-465 (2018). doi: 10.1007/978-3-319-95678-7_51.
43) K. Marzia, M. F. Hasan, T. Miyazaki, B. B. Saha, and S. Koyama, "Key Factors of Solar Energy Progress in Bangladesh until 2017", *Evergreen*, **5**(*2*), 78-85 (2018). doi: 10.5109/1936220.
44) T. Hanada,"Modifying the feed-in tariff system in Japan: an environmental perspective", *Evergreen: joint journal of Novel Carbon Resource Sciences & Green Asia Strategy*, **3**(*2*), 54-58 (2016). doi: 10.5109/1800872.