Journal of Electronics and Electrical Engineering
https://ojs.wiserpub.com/index.php/JEEE

Research Article

Laboratory-Scale Design and Implementation of A Low-Cost IoT-Based Data Logger for A Remote DC Microgrid in Ghana

Godfred Atinkum*, M. Tariq Iqbal, John E. Quaicoe

Department of Electrical and Computer Engineering, Memorial University of Newfoundland, St. John's, NL, Canada
*Correspondence: godredatinkum14@gmail.com

Received: 14 August 2025; **Revised:** 28 January 2026; **Accepted:** 3 February 2026; **Published:** 9 March 2026

Abstract: The deployment of renewable and decentralized energy systems in remote locations is on the rise. As a result, the need for supervisory and data-logging frameworks to monitor distributed energy resources efficiently has increased. Robust logging of data forms the core of such systems, as it supports the continuous data acquisition, storage, data management, and analysis of the most critical operational parameters that define the system's operation, stability, and performance. Traditional data logging systems are typically expensive, complicated, and unsuitable for small-scale or remote microgrids. To address this challenge, this paper presents the design and implementation of a low-cost IoT-based data logger for monitoring and analyzing the performance of a remote DC microgrid in Ghana. The proposed system consists of a photovoltaic (PV) system, a wind turbine, a battery energy storage system, and a diesel generator, all connected to a DC bus. An Arduino UNO R4 Wi-Fi microcontroller serves as the RTU that collects voltage and current signals from sensors in the field and transmits them via an MQTT communication protocol to a cloud-based platform for monitoring and further analysis. ThingSpeak and MATLAB App Designer are used to provide real-time display, analysis, and supervision of the system's performance. Tests were conducted on a FESTO LabVolt hybrid power training system, and the results indicated that the data logger effectively captured the system's dynamic electrical parameters. It also enabled cloud-integrated real-time supervision.

***Keywords*:** Arduino, data logger, Hybrid Power System (HPS), Master Terminal Unit (MTU), Remote Terminal Unit (RTU), ThingSpeak, remote microgrid

1. Introduction

As the world continues to emphasize reducing greenhouse gas emissions, renewable energy integration has become a key focus for sustainable development. Hybrid Power Systems (HPS) based on photovoltaics, micro hydro, wind turbines, and battery storage systems are increasingly being used as alternatives to conventional power systems [1, 2]. Combining multiple renewable generation sources improves the overall reliability, security, efficiency, and resilience of HPS. Such hybrid configurations, particularly in off-grid and remote locations, help mitigate the intermittency issues associated with individual renewable energy sources. Despite the advantages, numerous challenges arise due to the distributed nature of the various power sources. Some of the challenges include voltage and frequency regulation imbalance, power flow management, communication between components, and data integrity issues [3]. One way to address this problem is to

DOI: https://doi.org/10.37256/jeee.5120269320

monitor, control, and coordinate the multiple generating sources. A data logging system can be utilized to achieve this objective. Data loggers enable real-time data collection and monitoring of field devices. They contribute to the timely detection of faults, performance optimization, and coordinated energy management. In addition, a data logger can balance generation, optimize load management, and reduce operational inefficiencies for remote HPS operations with reliability [4].

The use of multiple renewable energy generation sources to form hybrid systems requires the implementation of robust supervisory systems to manage data and control operations. In this regard, data logging systems have become indispensable. They are centralized systems for monitoring, acquisition, storage, and visualization of real-time operational data. With a data logger, system operators can remotely monitor generation, storage, and distribution assets, thereby minimizing the need for human intervention [3]. Data loggers are at the core of SCADA operations. These devices are designed to continuously measure and record electrical parameters, such as voltage, current, frequency, and power output, as well as environmental variables like irradiance, wind speed, and temperature [5]. The role of the data logger is not merely to store data. It also serves as a link between sensors and higher-level control systems, allowing for real-time analysis and decision-making [4]. In HPS, where generation and load requirements are dynamic and change with time, continuous logging provides the visibility required to achieve energy balance and stability. In the absence of robust data logging, operators would be unable to detect system abnormalities in a timely manner or optimize system behavior based on historical patterns. Furthermore, with more complex HPS coming into existence, the volume of data generated also increases exponentially. Hence, there is a need for an advanced logging infrastructure to handle not only real-time data but historical data as well [6]. Data loggers thus form the backbone of any SCADA system. They ensure that reliable and accurate data support all supervisory and control functions.

Aside from data loggers, the effective operation of SCADA in hybrid power systems also relies on Remote Terminal Units (RTUs) and Master Terminal Units (MTUs). RTUs generally act as intelligent devices that directly interface with sensors and actuators in the field. They collect real-time data, execute local control commands, and allow communication of information with higher supervisory layers [7]. MTUs, located at the control center, function as the supervisory hub of the SCADA system. They perform centralized processing and send control instructions back to the field [4].

In recent years, SCADA-based hybrid power systems have also been enhanced by the introduction of the Internet of Things (IoT). IoT-enabled sensors and gateways extend connectivity beyond the conventional RTU-MTU communications. IoT enables devices to communicate wirelessly and through cloud platforms. This supports real-time analytics and remote access from virtually anywhere. It eliminates the sole reliance on centralized control rooms. IoT also enables interoperability with smart meters, mobile applications, and advanced energy management systems, providing a more flexible and scalable supervisory framework. When combined with SCADA, IoT bridges the gap between field-level data collection, intelligent decision-making, and enterprise-level energy management. As hybrid systems grow in size and complexity, IoT integration ensures greater visibility, responsiveness, and efficiency, especially in remote systems.

There has been a considerable effort by researchers in developing and improving data acquisition systems. Kamyaba et al. reviewed HPS that employ SCADA and Phasor Measurement Unit (PMU) data [8]. They argued that SCADA systems suffer from low sampling rates, whereas PMUs provide high-frequency, time-synchronized data. However, PMUs are relatively expensive to use across entire networks. They proposed a Hybrid State Estimation (HSE) as an alternative. HSE integrates the two data sources to improve observability and reliability. Despite its advantages, HSE suffers from computational burden in centralized systems.

The authors in [9] proposed an Open Platform Communications Uniform Architecture (OPC UA) for upgrading supervisory control in Concentrated Solar Power (CSP) plants. The system was a combination of an OPC UA, a Wi-Fi, and an open-source software. It also incorporated a Python-based gateway to ensure backward compatibility between the old Modbus systems and the new OPC UA server. The system was tested on a heliostat field. Long-term tests showed that the designed system could operate continuously without crashing, and hence, it is reliable for industrial applications. Additional tests also revealed that the new operator interface, which was built using Python, significantly simplified tasks and improved user satisfaction. The positive operator feedback confirmed the intuitive nature of the interface, which reduced cognitive load and training requirements.

In [10], an open-source, low-cost SCADA system was designed to monitor an offshore aquaculture HPS in Newfoundland. Five sensors were connected to measure the field voltages and currents. Equipped with a long-range, wide-area network, an Arduino Leonardo senses data from the field devices, processes the data, and communicates it to other applications via a Message Queuing and Telemetry Transport (MQTT) platform. Data from the field devices is stored in a cloud server as well. Similarly, reference [11] designed an IoT-based SCADA system for a renewable energy system. However, unlike in reference [10], where physical components were employed, the authors in [11] designed a web-based system. The Wonderware Intouch software was interfaced with MATLAB/Simulink through a server. By doing so, the HPS system simulated in Simulink can be accessed and processed by remote applications through the web. Although the Wonderware InTouch software was successfully interfaced with MATLAB/Simulink to enable remote access to HPS system data, the work remains limited to a simulated environment without hardware validation.

Orie et al. [4] presented an enhanced IoT-based optimization system for an HPS in Cartwright. The design was such that an Arduino-based data acquisition unit was used to measure current and voltage on the solar, wind, and battery components of a scaled FESTO training system of the original system. The collected sensor data were sent to a cloud storage through Wi-Fi for subsequent processing and display. This system is, however, limited by the lack of real-time monitoring capabilities, which restricts its practical applicability.

The authors in [12] implemented an open-source data logging system using Arduino Mega and Woo terminal as RTUs. To measure and record parameters such as battery voltage, battery current, solar panel current, generator current, and load currents in real-time, they used one voltage sensor and five current sensors. The Arduino collected sensor data and transmitted it to a laptop, which acts as the MTU, using a Fermata protocol. On the other hand, the Woo terminal operated as a standalone monitoring device. Node-Red was used as a processing platform, functioning as both analyzer and monitoring unit for data processing, analysis, storage, and visualization through a web-based dashboard. Similarly, He et al. [7] designed a Node-Red-based open-source SCADA system to monitor PV systems. Three voltage sensors and three current sensors were used to monitor the PV panel, battery, and load parameters. The RTU was an ESP32-E microcontroller that read analog data from the sensors. It also transmitted it to a Banana Pi via Wi-Fi. Unlike reference [12], an MQTT in a publish-subscribe framework was used for all communications. The Node-RED platform was installed locally on the Banana Pi to serve as the Human Machine Interface (HMI). A relay was also integrated for supervisory load control. The load could be switched remotely by the operator via the dashboard.

The authors in [11] designed an IoT-based data acquisition system for a hybrid power system using MATLAB/Simulink and the ThingSpeak platform. The model is built in Simulink, and a KEPServerEx client is used to interface the system with ThingSpeak. The effectiveness of the SCADA system is evaluated by comparing the simulation results with results from a hardware prototype.

In [13], Ndukwe et al. designed a Long-Range (LoRa) data acquisition system for communication and data transfer purposes in an MG. The system comprised voltage and current sensors, LoRa nodes equipped with microcontrollers, a gateway, and a Node-Red server for data visualization. The collected data were encoded in Base64 format for LoRa transmission. Tests were conducted in St. John's, Newfoundland, Canada, and achieved a 90% packet delivery for up to 4km. In a related work, Ferlito et al. [14] investigated an IoT-based SCADA system for large PV farms in which data is transmitted overhead. Their study used edge computing, microservices, and hybrid cloud storage to reduce latency and optimize real-time monitoring. Although the simulation results indicate that the proposed system significantly reduces communication overheads, improves system responsiveness, and enhances predictive maintenance capabilities, the proposed system may suffer from data breaches and cyber attacks.

A considerable amount of literature has been reviewed during the course of this study, with some of the findings highlighted. A comparison of several state-of-the-art approaches reported in the literature, highlighting their key features and limitations, is presented in Table 1. Building on the findings of previous studies, this work presents the design and implementation of a low-cost data logger for a remote DC microgrid in Ghana. The system is constructed based on the architecture and component parameters described in [17] and [18]. The primary objective of the proposed system is to capture, record, and monitor system voltage and current parameters, thereby supporting reliable performance assessment and informed decision-making. Accurate monitoring of a microgrid's voltage and current parameters is essential for evaluating its stability, detecting faults, identifying abnormal operating conditions, and preventing component damage. In remote

installations where technical support is limited, continuous parameter monitoring enables early detection of performance degradation and reduces system downtime. Furthermore, reliable data acquisition supports energy management, load analysis, and informed planning for system expansion or optimization. These capabilities are critical for ensuring the reliability and sustainability of remote microgrid systems. The study combines both local and remote data visualization and analysis, which, to the best of the author's knowledge, has not been comprehensively addressed in previous literature. Moreover, to ensure affordability without compromising performance, the proposed system utilizes low-cost components, including the ACS712 current sensor and the Arduino Uno R4 Wi-Fi microcontroller. The following are the contributions of this work:

1. The proposed system uniquely combines real-time local data visualization using MATLAB App Designer with cloud-based monitoring via ThingSpeak in a unified MQTT-based framework.

2. The proposed system offers empirical evidence of the feasibility and reliability of a lab-scale low-cost IoT-based monitoring architecture by implementing and testing it on a FESTO LabVolt hybrid power training platform.

The rest of the paper is organized as follows: Section 2 presents an overview of the system. It includes the description of the system and its components. The implementation methodology of the study is presented in Section 3. Section 4 discusses experimental results. Section 5 draws the conclusions.

Table 1. Comparison of data logging systems

Reference	Microcontroller	Communication	Local monitoring	Cloud monitoring	Hardware validation	Data logging
[10]	Arduino Leonardo	LoRa	No	Yes	Yes	Yes
[11]	Simulink (Web-based)	HTTP	Yes	Yes	No	Yes
[15]	Raspberry Pi Zero	Wi-Fi	No	Yes	Yes	Yes
[7]	ESP32 + Banana Pi	MQTT/Wi-Fi	No	Yes	Yes	Yes
[3]	ESP32-E	HTTP TCP/IP	No	Yes	Yes	Yes
[16]	Arduino UNO Arduino Mega	UART + Wi-Fi	Yes	Yes	Yes	Yes
Proposed System	Arduino UNO R4 Wi-Fi	MQTT/Wi-Fi	Yes	Yes	Yes	Yes

2. System overview

2.1 *System description*

A schematic of the HPS with the proposed data acquisition components is illustrated in Figure 1. The system comprises PV, a wind turbine, battery storage, and a diesel generator, all connected to a DC bus. An inverter converts the DC voltage and current to supply an AC load. In Figure 1, there are six sensors, including two voltage sensors: one for measuring the system's DC voltage and one for measuring the battery voltage level. It also features four current sensors for measuring currents from PV, wind turbine, diesel generator, and load. The PV system is controlled by a Maximum Power Point Tracker (MPPT) to maximize its output. The diesel and battery storage systems serve as backup power and are manually switched on when there is no output power from the PV and wind turbine systems. Additionally, the battery serves as an energy storage component. The voltage sensors are connected in parallel to the DC bus and battery, while one current sensor is connected in series with each component. An Arduino R4 Wi-Fi microcontroller serves as the system's RTU, while an HP Victus Ryzen 5 laptop serves as the MTU. The load in this system is a 13W LED bulb.

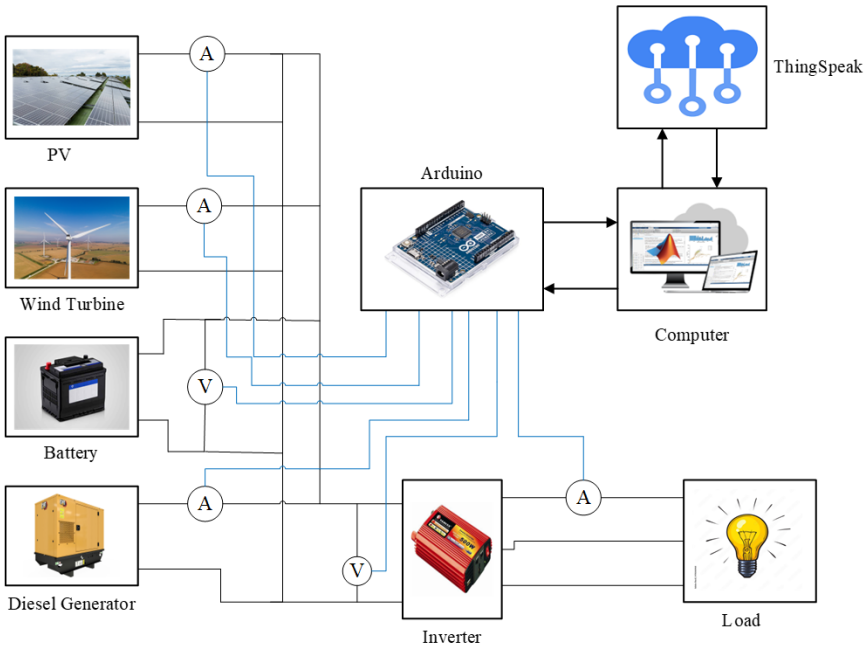


Figure 1. System architecture

2.2 *System components*

1. FESTO LabVolt Training System: The Festo didactic training system used in this study is composed of PV and wind turbine simulators. The PV simulator features a 600W AC floodlight that serves as the light source. A switch on the PV plate's cover regulates the timing of the 85W PV energy generation. The wind simulator uses a DC motor to regulate the speed of the 400W turbine. In addition to the PV and wind simulators, the system uses a 500W converter to invert the DC power to AC.

2. Arduino UNO R4 Wi-Fi: The Arduino UNO R4 Wi-Fi is the RTU used in this study. It uses the Renesas RA4M1 32-bit ARM Cortex-M4 microcontroller. The microcontroller has a clock speed of 48 MHz with 256 KB of flash memory and 32 KB of SRAM. The microcontroller also incorporates an Espressif ESP32-S3 module, thus making the board compatible with built-in Wi-Fi and Bluetooth applications and suitable for IoT applications. The UNO R4 Wi-Fi has 14 digital pins, 6 analog inputs, higher current capability on the 5V pin, and improved power efficiency. The Arduino UNO R4 Wi-Fi board used in this study is shown in Figure 2. In this study, the Arduino will communicate with the field sensors using its analog input terminals.

3. Current Sensor: The ACS712 current sensor is used in this work to measure the component currents. It is designed for precise measurement, and it operates on the Hall-effect principle [19]. The sensor is available in different versions with current ranges of 5A, 20A, and 30A, with sensitivities of 185 mV/A, 100 mV/A, and 66 mV/A, respectively. This study uses the ACS712 5A. It has three main pins, namely VCC for the 5V supply, GND for ground, and OUT, which provides an analog voltage proportional to the current. The sensor is biased at VCC/2, so that zero current is offset at 2.5 V on the sensor. Figure 3 gives a schematic of the ACS712 current sensor. Within the given operating range, the output current is linearly proportional to the primary current (IP) being measured. A filtering capacitor C is also included in order to reduce electrical noise, and this value is chosen depending on the type of application.

4. Voltage Sensor: Figure 4 shows the F031-06 Voltage Sensor Module, which is used in this study to measure DC voltage [12, 20]. It employs a voltage divider to scale down the input voltage, allowing microcontrollers with 3.3 V or 5 V analog inputs to safely read higher voltages. The divider contains a 7.5 kΩ resistance and a 30 kΩ resistance. Similar to the ACS712, this module has three major connections: a positive input, a ground, and an analog signal output that provides a voltage relative to the measured input.

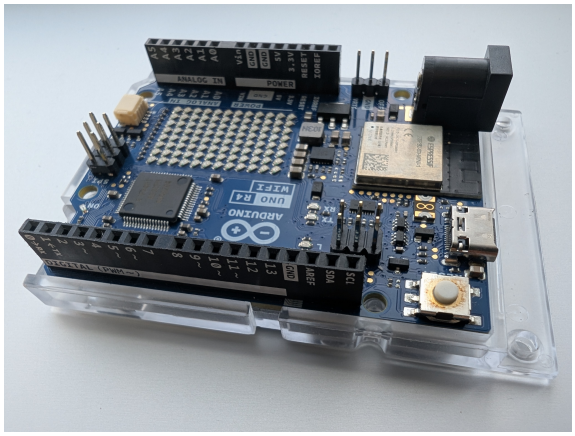


Figure 2. Arduino Uno R4 Wi-Fi

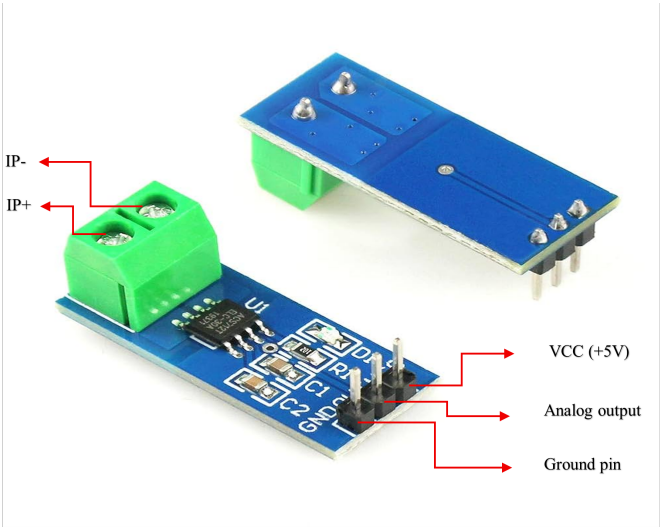


Figure 3. ACS712 current sensor

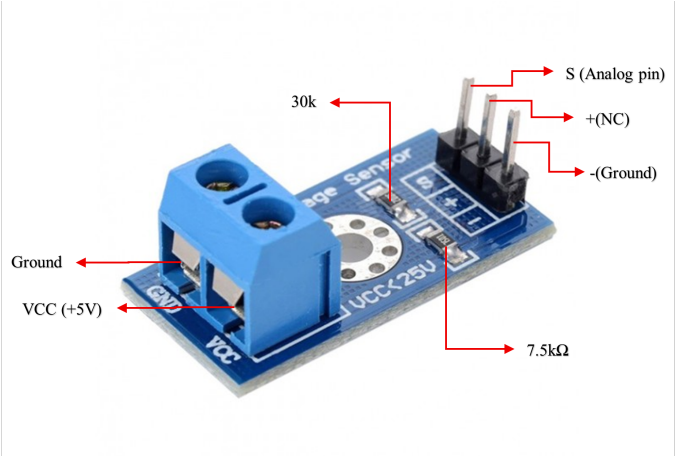


Figure 4. Voltage sensor

5. HP Victus Laptop: The MTU used for this research is an AMD Ryzen 5 laptop. It has a built-in Radeon Graphics featuring a 3.30 GHz processor that provides strong multi-core performance for handling data acquisition, processing, and control tasks. The system supports a 64-bit operating system running on an x64-based processor architecture, ensuring compatibility with modern SCADA software and systems. Equipped with 16.0 GB of RAM, the MTU can efficiently manage real-time monitoring, data logging, and communication with remote field devices. It should be noted that the Arduino Uno R4 Wi-Fi is capable of connecting directly to the cloud and, therefore, does not require a laptop to function as an MTU. However, in this research, the laptop is still used to support local measurement, data visualization, and system interaction during testing, in addition to its function as an MTU.

6. Communication Protocol: This study employs the Message Queuing Telemetry Transport (MQTT) as the communication platform over which Arduino UNO R4 Wi-Fi, MATLAB App Designer, and ThingSpeak interact [21]. MQTT is a lightweight communication protocol that runs over TCP/IP. As a publish/subscribe service, MQTT works by having a broker publish a message and a client subscribe to the message. In general, the broker routes messages to clients who have subscribed to a specific topic.

In this work, the Arduino UNO R4 Wi-Fi functions as the data publisher, transmitting sensor measurements to the MQTT broker hosted by the ThingSpeak cloud platform, while the ThingSpeak dashboard acts as a subscriber for data visualization and analysis. The ThingSpeak platform uses HTTPS and Transport Layer Security (TLS) to encrypt sensor data transferred from the Arduino to the MQTT broker. This ensures that sensor data is not intercepted or eavesdropped on while it is being transmitted. The system uses unique channel identifiers and Write API keys assigned by ThingSpeak to authenticate connections, ensuring that only authorized devices can publish data to the designated channels. These credentials are embedded in the Arduino firmware and MATLAB scripts and are accessible only within the authorized user environment.

It is worth noting that continuous internet connectivity is assumed for the cloud-based data transmission to ThingSpeak. The Arduino UNO R4 Wi-Fi does not perform edge-level buffering; however, MATLAB App Designer logs and stores acquired data locally on the host computer. This provides supervisory-level data buffering during operation.

The table below (Table 2) summarizes the estimated costs of the individual components incorporated in this work. Component selection was guided by both technical suitability and cost-effectiveness. The resulting cost structure demonstrates the design's affordability and scalability for real-world applications.

Table 2. Cost of data logging components

Component	Part Number	Quantity	Total price before taxes (in Canadian dollars)
Arduino UNO R4 Wi-Fi	ABX00087	1	37.75
Current sensor	ACS712ELCTR-05B-T	4	11.56
Voltage sensor	RB-Oel-88	2	20.06

3. Methodology

The proposed data logger uses an Arduino microcontroller, four current sensors, and two voltage sensors to measure and monitor circuit parameters of a FESTO training system. The current sensors are connected in series with the PV panel, wind turbine, and DG system, as well as the connected load. The two voltage sensors are connected across the DC bus and battery systems to measure the system's DC and battery voltages, respectively. The Arduino's VCC supplies all the sensors with 5V of power.

Figure 5 illustrates the wiring connections between the sensors and the Arduino, showing how each sensor is interfaced with the microcontroller. Table 3 lists the components, the types of sensors used, and the corresponding Arduino input pins to which each sensor is connected. Algorithm 1 summarizes the data acquisition algorithm, outlining the steps the Arduino follows to read sensor values, process the data, and store or display the measurements for monitoring purposes.

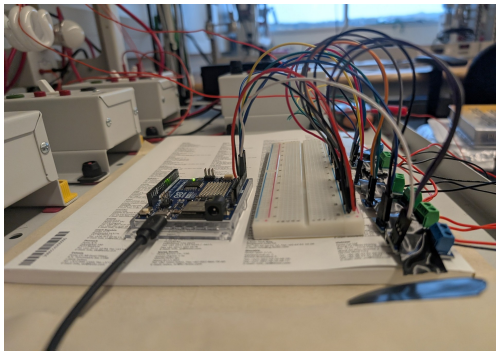


Figure 5. Connecting sensors to Arduino at MUN ECE laboratory

Table 3. Sensors and their connections with UNO R4 Wi-Fi

Equipment	Sensor Used	UNO R4 Wi-Fi pin#
DC Bus	Voltage Sensor	A0
PV system	Current Sensor	A1
Wind turbine	Current Sensor	A2
Diesel Generator	Current Sensor	A3
Load	Current Sensor	A4
Battery Storage	Voltage Sensor	A5

In Algorithm 1 below, ThingSpeak, a cloud-based IoT analytics platform, is employed to receive and visualize sensor readings transmitted by the UNO R4 Wi-Fi microcontroller. The UNO R4 Wi-Fi serves as the data acquisition node, utilizing its built-in Wi-Fi capabilities to transmit real-time measurements to ThingSpeak via an MQTT platform. MATLAB App Designer communicates directly with ThingSpeak to process and analyze the acquired data.

Algorithm 1 Data acquisition pseudocode

Initialization
Establish a connection between Arduino, App Designer, and ThingSpeak
if the START button is pressed **then**
    Set the STOP flag to false
    Initialize six graphs: one for each parameter to be recorded
    Record start time
    Set up ThingSpeak parameters: channel ID, writeAPIkey, upload interval
    Upload data to ThingSpeak
    while STOP flag is false **do**
        Read sensor values
        Compute actual currents using calibration equations
        Compute elapsed time
        Update graphs
        Adjust x-limits and refresh plots
        Append data to storage file
        Check stop condition: STOP = True
    end while
end if

In the MATLAB App Designer, custom scripts are also developed to handle data acquisition and publication, as well as to generate graphical interfaces for local transient data visualization. A one-to-one mapping is defined between each measured system parameter in MATLAB App Designer and a dedicated ThingSpeak channel field. This enables a structured cloud-based monitoring. Table 4 summarizes the configuration of the ThingSpeak channel, explicitly mapping each monitored system parameter to its corresponding cloud field and update interval. The dashboards in ThingSpeak enable remote monitoring of the PV, wind, battery, and diesel generator system parameters. These include PV current, battery voltage, wind current, load current, system DC voltage, and diesel generator current. A benchtop DC power supply is used to represent the diesel generator in this study, as an actual diesel generator is not available in the laboratory.

Table 4. ThingSpeak channel configuration

Field	Parameter	Description	Update Interval
Field 1	Voltage	System DC voltage	15 seconds
Field 2	Current	Solar output current	15 seconds
Field 3	Current	Wind turbine output current	15 seconds
Field 4	Current	Diesel generator output current	15 seconds
Field 5	Current	Load-side current consumption	15 seconds
Field 6	Voltage	Battery terminal voltage	15 seconds

Inside the MATLAB scripts, the various sensor readings are scaled and calibrated to read accurate measurements. The calibrations involved taking an initial reading and adjusting the sensors' output by accounting for the 'offset reading' effect and scaling up the voltage reading using the resistances of the internal voltage divider of the voltage sensor. The formulae for calibrating the current and voltage sensors are given in equations (1) and (2) below:

$$\text{Current reading} = \frac{V_{\text{out}} - VCC/2}{\text{Sensitivity}} \tag{1}$$

$$\text{Voltage reading} = V_{\text{out}} * \frac{R1 + R2}{R2} \tag{2}$$

In the above equations, V_{out} is the Arduino-measured voltage, VCC is Arduino's 5 V supply, and R1 = 30 kΩ and R2 = 7.5 kΩ are the resistance values of the voltage divider.

4. Experimental results and discussion

The developed IoT-based data acquisition system is configured to collect and monitor current and voltage parameters from a Festo Lab Training System and a benchtop DC power supply, which serves to represent a diesel generator, at the Renewable Energy Lab at Memorial University of Newfoundland. The experimental setup of the proposed data acquisition system is shown in Figure 6.

For the purpose of this study, the outputs from the solar, wind, and battery systems of the Festo Lab System, as well as the benchtop DC power supply, are connected to a DC bus. The 500 W, 12VDC/120VAC inverter converts the DC power from the DC bus to AC power, which feeds the light bulbs. The sensors, which serve as field instrumentation devices, are connected to the Arduino microcontroller, which is in turn connected to the MTU. Real-time measurements are obtained from the setup using the above system configuration. Figure 7 illustrates the local simulated current (A) output profiles from the PV, wind turbine, and diesel generator systems, as well as the local load current, for the simulations carried out in this study. As shown in Figure 7, the solar and wind turbine systems operate concurrently, while the diesel generator

remains in standby mode. For this reason, the output current profiles for both the solar and wind systems are similar. The solar and wind systems are powered shortly after the simulation begins, and each reaches its peak current approximately 4 seconds later. At the same time, the diesel generator remains in the off position. As the simulation progresses, the solar system is turned off while the wind system continues to operate under regulation. A variable-speed motor is used to regulate the speed of the wind turbine. Consequently, the output current is regulated such that it does not drop to zero altogether, unlike the solar system, which lacks an irradiation control mechanism. It can therefore be seen that while the solar output current is zero at approximately 36 s, 76 s, and 96 s, the wind output current is about 1.2 A, 0.8 A, and 0 A, respectively, at the end of the simulation. In the graph, the diesel generator does not produce a significant current because the battery partially supports the load when the two renewable sources are off, reducing the operating time and output required from the diesel generator.

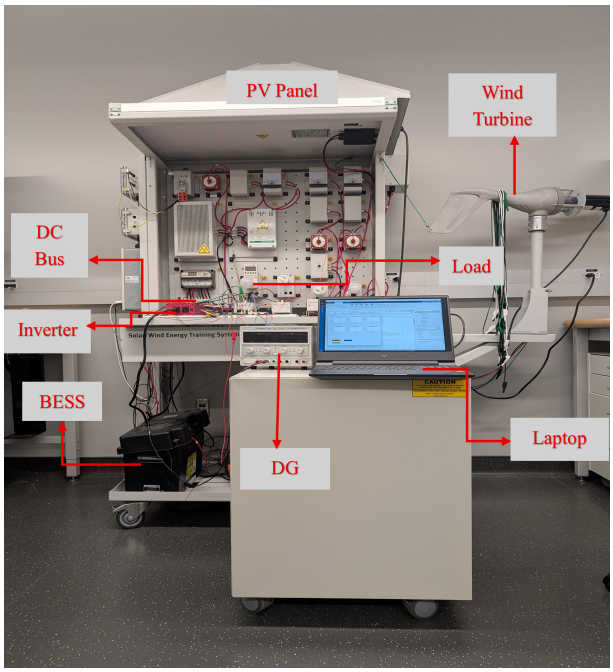


Figure 6. System setup

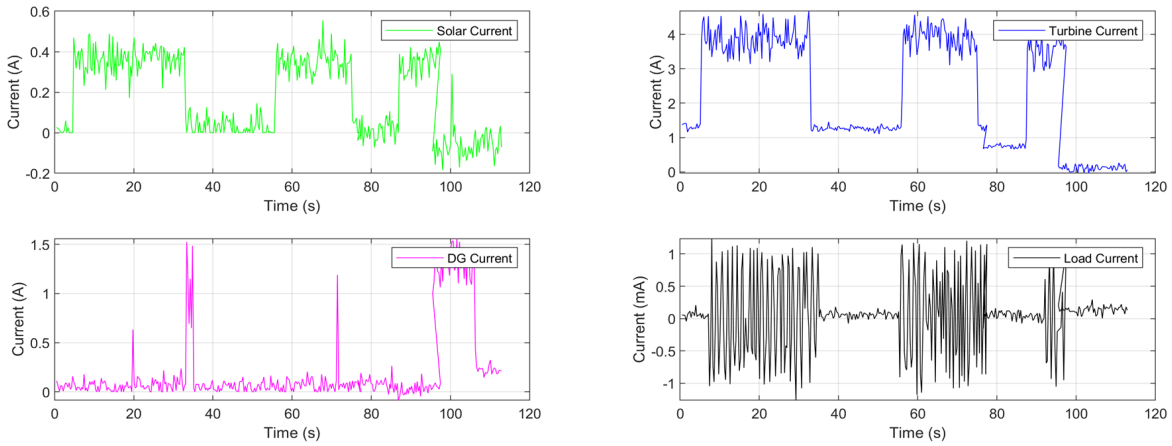


Figure 7. App Designer dashboard showing components' current waveforms

The load is also switched on and off at different stages of the simulation, as seen in Figure 7 above. This is done to emulate the varying load conditions and assess the effectiveness of the proposed systems in maintaining a stable DC voltage under varying load conditions.

Figure 8 shows the system DC voltage and battery voltage during the simulation. The system DC voltage remains almost constant at 12.6 V during the simulation, irrespective of the switching mechanisms of the solar, wind, diesel, and load systems. This proves the system's resilience under intermittent conditions. The battery, used in conjunction with the diesel generator, ensures stability and resilience.

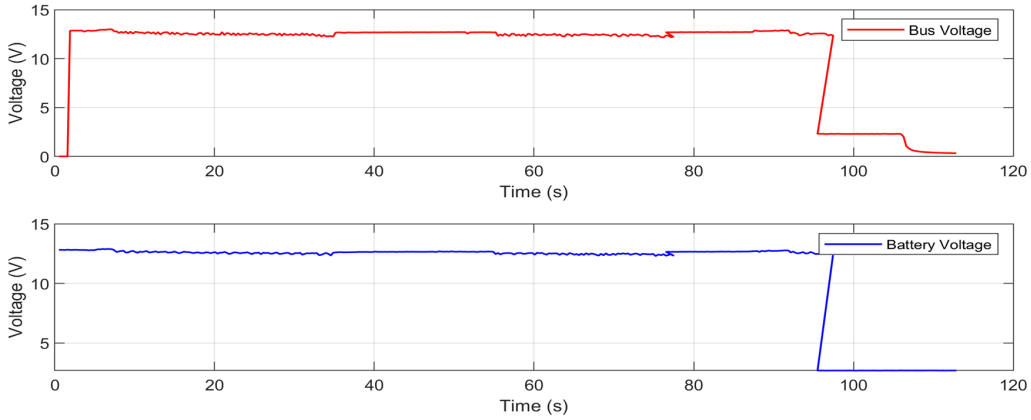
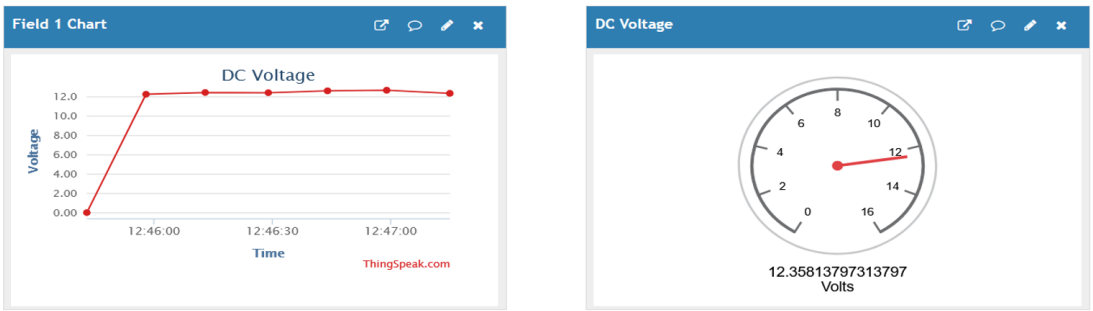


Figure 8. Components' voltage waveforms from App Designer dashboard

In addition to the local simulation analysis, all measured variables were transmitted to ThingSpeak. A ThingSpeak account is created using an existing MathWorks account. A new channel with fields and widgets for measurement is set up, and each field, along with its widget, is assigned to the specific variable being uploaded. The ThingSpeak platform provides a unique Write API Key, which is inserted into the MATLAB code to enable data transmission. Once data begins uploading, the dashboard is edited by adding visual elements such as plots, indicators, or numeric displays, allowing real-time visualization of system performance. Figure 9 displays the ThingSpeak results for the proposed system. There is a separate chart and gauge for each component, as shown in the figure below. This allows remote monitoring and real-time visualization of the system's performance. Additionally, users can track the operational behavior of each component at any time and from any location with internet access. Combining ThingSpeak with MATLAB App Designer enabled deeper insights through data processing and visualization. System abnormalities can be detected with the help of the two dashboards, thereby improving the system's reliability.

(a)



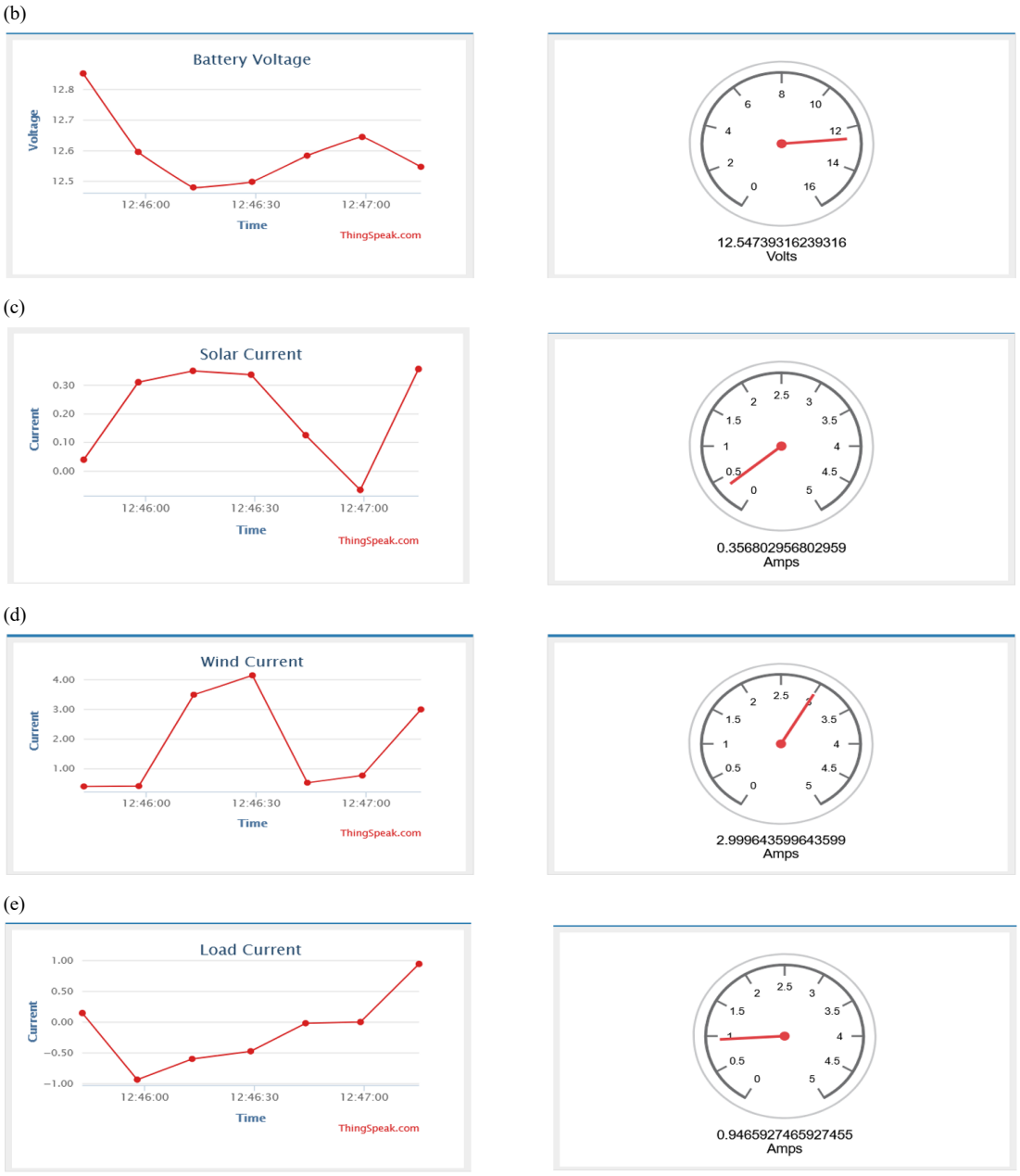


Figure 9. ThingSpeak dashboard of various components: (a) System DC voltage; (b) Battery voltage; (c) PV current; (d) Wind current; (e) Load current

5. Conclusion

The need for affordable and efficient data acquisition systems in remote microgrids has driven the development of innovative, IoT-enabled SCADA alternatives. In this study, a low-cost data acquisition system is successfully designed and implemented. The system incorporates a PV, a wind turbine, a diesel generator, and a battery storage system, all connected to a DC bus. An inverter inverts the DC voltage to supply AC loads. An Arduino UNO R4 Wi-Fi microcontroller is used as the RTU and sends data to the MTU, a personal laptop, for analysis. The study utilizes MATLAB App Designer for local data monitoring and visualization, and ThingSpeak for remote data monitoring and visualization, respectively. Communication between MATLAB App Designer and ThingSpeak is via the MQTT protocol. Experimental results

confirmed the system's ability to accurately capture and monitor the remote microgrid. In all, the data logging methodology presented in this work represents a significant step towards intelligent monitoring and management of remote microgrids.

Conflicts of interest

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

[1] E. Tsiaras, D. N. Papadopoulos, C. N. Antonopoulos, V. G. Papadakis, and F. A. Coutelieris, "Sustainable off-grid power supply for small settlements," in *Hybrid Technologies for Power Generation*, Academic Press, 2022, pp. 219-247. https://doi.org/10.1016/B978-0-12-823793-9.00007-3.

[2] E. Tsiaras, D. N. Papadopoulos, C. N. Antonopoulos, V. G. Papadakis, and F. A. Coutelieris, "Planning and assessment of an off-grid power supply system for small settlements," *Renewable Energy*, vol. 149, pp. 1271-1281, 2020. https://doi.org/10.1016/j.renene.2019.10.118.

[3] M. Waqas and M. Jamil, "Smart IoT SCADA system for hybrid power monitoring in remote natural gas pipeline control stations," *Electronics*, vol. 13, no. 16, p. 3235, 2024. https://doi.org/10.3390/electronics13163235.

[4] R. Orie, L. Otabil, J. Agorua, and M. T. Iqbal, "Enhanced IoT-based optimization for a hybrid power system in Cartwright, Labrador," *Energies*, vol. 18, no. 7, p. 1566, 2025. https://doi.org/10.3390/en18071566.

[5] L. Chalal, A. Saadane, and A. Rachid, "Unified environment for real time control of hybrid energy system using digital twin and IoT approach," *Sensors*, vol. 23, no. 12, p. 5646, 2023. https://doi.org/10.3390/s23125646.

[6] L. Ahsan, M. J. A. Baig, and M. T. Iqbal, "Low-cost, open-source, emoncms-based SCADA system for a large grid-connected PV system," *Sensors*, vol. 22, no. 18, p. 6733, 2022. https://doi.org/10.3390/s22186733.

[7] W. He, M. J. A. Baig, and M. T. Iqbal, "An open-source supervisory control and data acquisition architecture for photovoltaic system monitoring using ESP32, Banana Pi M4, and Node-RED," *Energies*, vol. 17, no. 10, p. 2295, 2024. https://doi.org/10.3390/en17102295.

[8] L. Kamyabi, T. T. Lie, S. Madanian, and S. Marshall, "A comprehensive review of hybrid state estimation in power systems: challenges, opportunities and prospects," *Energies*, vol. 17, no. 19, p. 4806, 2024. https://doi.org/10.3390/en17194806.

[9] J. A. Carballo, J. Bonilla, J. Fernández-Reche, A. L. Avila-Marin, and B. Díaz, "Modern SCADA for CSP systems based on OPC UA, Wi-Fi mesh networks, and open-source software," *Energies*, vol. 17, no. 24, p. 6284, 2024. https://doi.org/10.3390/en17246284.

[10] M. N. Asgher and M. T. Iqbal, "Development of a low-cost, open-source LoRA-based SCADA system for remote monitoring of a hybrid power system for an offshore aquaculture site in Newfoundland," *European Journal of Electrical Engineering and Computer Science*, vol. 7, no. 6, pp. 65-73, 2023. https://doi.org/10.24018/ejece.2023.7.6.589.

[11] M. O. Qays, M. M. Ahmed, M. A. P. Mahmud, A. Abu-Siada, S. M. Muyeen, M. L, Hossain, et al., "Monitoring of renewable energy systems by IoT-aided SCADA system," *Energy Science and Engineering*, vol. 10, no. 6, pp. 1874-1885, 2022. https://doi.org/10.1002/ese3.1130.

[12] S. A. Omidi, M. J. A. Baig, and M. T. Iqbal, "Design and implementation of node-red based open-source SCADA architecture for a hybrid power system," *Energies*, vol. 16, no. 5, p. 2092, 2023. https://doi.org/10.3390/en16052092.

[13] C. Ndukwe, M. T. Iqbal, X. Liang, J. Khan, and L. Aghenta, "LoRa-based communication system for data transfer in microgrids," *AIMS Electronics and Electrical Engineering*, vol. 4, no. 3, pp. 303-325, 2020. https://doi.org/10.3934/ElectrEng.2020.3.303.

[14] S. Ferlito, S. Ippolito, C. Santagata, P. Schiattarella, and G. Di Francia, "A study on an IoT-based SCADA system for photovoltaic utility plants," *Electronics*, vol. 13, no. 11, p. 2065, 2024. https://doi.org/10.3390/electronics13112065.

[15] A. Soetedjo and E. Hendriarianti, "Development of an IoT-based SCADA system for monitoring of plant leaf temperature and air and soil parameters," *Applied Sciences*, vol. 13, no. 20, p. 11294, 2023. https://doi.org/10.3390/app132011294.

[16] A. J. Moshayedi, A. S. Roy, L. Liao, and S. Li, "Raspberry Pi SCADA zonal based system for agricultural plant monitoring," In Proc. 6th International Conference on Information System and Computing Engineering, Shanghai, China, Dec. 20-22, 2019, pp. 427-433. https://doi.org/10.1109/ICISCE48695.2019.00092.

[17] G. Atinkum, M. T. Iqbal, and J. E. Quaicoe, "Dynamic simulation of a DC microgrid for a remote community in Ghana," *Journal of Electronics and Electrical Engineering*, vol. 4, no. 2, pp. 195-212, 2025. https://doi.org/10.37256/jeee.4220257851.

[18] G. Atinkum, M. T. Iqbal, and J. E. Quaicoe, "Techno-economic design and sensitivity analysis of a DC microgrid for a remote community: a case study in Ghana," *Journal of Electronics and Electrical Engineering*, vol. 4, no. 1, pp. 357-378, 2025. https://doi.org/10.37256/jeee.4120256509.

[19] Đ. Lazarević, M. Živković, Đ. Kocić, and J. Ćirić, "The utilizing hall effect-based current sensor ACS712 for true RMS current measurement in power electronic systems," *Scientific Technical Review*, vol. 72, no. 1, pp. 27-32, 2022. https://doi.org/10.5937/str2201027l.

[20] Components101, "Voltage sensor module pinout, features, specifications and arduino circuit," [Online]. Available: https://components101.com/sensors/voltage-sensor-module. [Accessed Nov. 7, 2025].

[21] A. A. O. Bahashwan, and S. Manickam, "A brief review of messaging protocol standards for internet of things (IoT)," *Journal of Cyber Security and Mobility*, vol. 8, no. 1, pp. 1-14, 2018.