Nigerian Journal of Theoretical and Environmental Physics (NJTEP)
ISSN Print: 3026-9601
ISSN Online: 1597-9849
DOI: https://doi.org/10.62292/njtep.v4i3.2026.156
Volume 4(3), September 2026

Development and Implementation of an IoT-Based Solar PV Monitoring and Sizing System Using Arduino and ThingSpeak

[1]Onifade Olanrewaju A and [2]Fasanu O. Iyintosoluwa

[1]Department of Mathematical and Physical Sciences, Osun State University, Osogbo, Osun State, Nigeria.
[2]Department of Electrical & Information Engineering, Covenant University, Ota, Ogun State, Nigeria.

*Corresponding author's email: aonifadeolanrewaju@gmail.com

ABSTRACT

Real-time performance monitoring and appropriate array sizing remain practical barriers to small-scale solar photovoltaic (PV) adoption, particularly for residential users in off-grid and weak-grid settings. Published monitoring platforms are typically either proprietary or priced for utility-scale plants, or research prototypes that report electrical parameters only; sizing, meanwhile, is frequently carried out by rule of thumb at the point of purchase. This paper reports the development and bench testing of a low-cost, microcontroller-based system that combines real-time PV monitoring with an on-device sizing aid. An Arduino Uno acquires PV-array and battery voltage and current, ambient temperature and relative humidity, and illuminance, and drives a character LCD for local display. An ESP-01 Wi-Fi module forwards the readings over HTTP to a ThingSpeak channel, and a lightweight custom web dashboard reads that channel through the ThingSpeak API so that the fields are presented to the user in a single browser view. A matrix keypad accepts a daily energy demand figure, from which the firmware estimates the number of panels required using the peak-sun-hour method with a fixed 150 W panel rating, five peak sun hours and a system efficiency factor of 0.8. Bench testing confirmed that every sensor channel returned readings of the expected form, that the sizing routine produced correct results across a range of demand values, and that data reached the cloud within a few seconds of acquisition. Quantitative accuracy characterisation against calibrated reference instruments was not undertaken, and is identified as the principal limitation of the present work.

Keywords:
Arduino,
ESP-01,
Peak Sun Hour Method,
Real-Time Monitoring,
Solar PV Sizing,
ThingSpeak.

INTRODUCTION

Energy underpins transport, agriculture, industry and domestic life, and is drawn upon in chemical, electrical, nuclear, thermal and radiant forms (Panwar et al., 2011). Because fossil reserves are finite and their combustion carries substantial environmental cost, renewable sources solar, wind, tidal, biogas and geothermal among them have grown steadily in importance. Solar has expanded fastest of these, assisted by the absence of moving parts, modest maintenance requirements, long service life and a sustained fall in unit cost. Conversion is achieved in the photovoltaic (PV) cell, a photosensitive p–n semiconductor junction in which incident photons are converted directly into electrical energy.

Electricity demand continues to rise while conventional generation capacity lags behind it. In Nigeria, rapid population growth combined with heavy dependence on fossil fuels has produced measurable economic, environmental and public-health costs, and a persistent gap between electricity supply and demand (Chanchangi et al., 2023). PV technology is therefore directly relevant to Nigeria's pursuit of Sustainable Development Goal 7 on affordable and clean energy. Adoption, however, is constrained not only by device efficiency — the subject of extensive international research but also by practical factors particular to developing economies, among them installation quality, financing and the availability of design expertise (Chanchangi et al., 2023).

Two stages dominate the practical success of a small PV installation. The first is sizing: determining how much generating capacity and storage a given load actually requires. Khatib et al. (2013) survey the range of techniques used for this, from intuitive rules of thumb through analytical and numerical methods to artificial-intelligence approaches, and observe that the simpler methods remain the ones most often applied in the field because the data and tooling that the more rigorous methods require are frequently unavailable. The second

is monitoring, which is what allows a system to be operated as it was designed. A monitoring system gathers the data needed to track generation performance indicators, detect fault events and schedule maintenance economically (Aghaei et al., 2020; Rahman et al., 2018).

Related Work

Low-cost monitoring of PV installations using open hardware and open IoT platforms is an established research theme. Rahman et al. (2018) survey modern PV monitoring systems and note a persistent trade-off between the measurement fidelity of instrument-grade systems and the cost at which smaller installations can be served.

Aghenta and Iqbal (2019a) developed an open-source SCADA system for PV monitoring built from analogue current and voltage sensors, an Arduino Uno acting as a remote terminal unit, a Raspberry Pi running Node-RED as the communication channel and an Emoncms server for storage and visualization. In related work the same authors implemented a comparable architecture on an ESP32 with the Thinger.IO platform, reducing the hardware count and cost further (Aghenta & Iqbal, 2019b). Sutikno et al. (2021) reported a NodeMCU ESP8266 system measuring irradiance, ambient temperature, PV voltage and PV current using photodiodes, a DHT22, resistive dividers and an ACS712, publishing to ThingSpeak while retaining a microSD backup so that data would survive a network outage. Ali et al. (2024) described a further NodeMCU-based monitoring scheme in the same family, and Wibowo et al. (2024) used a PZEM004T energy meter with a NodeMCU to raise measurement accuracy on an on-grid installation.

Two observations follow from this body of work and motivate the present study. First, the architecture adopted here an ATmega-class controller for acquisition, an ESP8266-family radio for transport and ThingSpeak for storage and plotting is well established, so the contribution of this work does not lie in the monitoring path itself. Second, and more to the point, the systems cited above are monitoring systems only. They report what an installed array is doing; none of them assists the user with the prior question of how large that array should have been. Sizing tools, conversely, exist largely as desktop or web software divorced from the hardware, so the person installing a small system typically works from a rule of thumb at the point of purchase and never revisits it against measured performance. The system described here is an attempt to place both functions in the same low-cost device, so that a user with no design background can obtain a first-order sizing estimate at the keypad and then observe, through the same platform, how the installed system actually behaves.

Aim and Objectives

The aim of this study is to develop and implement a low-cost, web-accessible system that combines real-time monitoring with an on-device sizing aid for small standalone solar PV installations. The specific objectives are:

i. To select and integrate sensors for PV-array and battery voltage and current, ambient temperature and relative humidity, and illuminance with an Arduino Uno controller;
ii. To develop firmware that acquires these readings, presents them on a local character LCD, and accepts numeric input from a matrix keypad;
iii. To establish a cloud data path from the controller through an ESP-01 Wi-Fi module to a ThingSpeak channel, and to build a custom web dashboard that reads that channel and presents the fields in a single browser view;
iv. To implement a peak-sun-hour sizing routine on the controller and verify its output across a range of entered demand values; and
v. To evaluate the assembled system on the bench and to document its limitations explicitly.

Scope and Limitations

The study is confined to small standalone PV systems and does not address grid-tied or utility-scale installations. Remote monitoring depends on continuous internet connectivity, and interruptions suspend the cloud function, although local LCD display and sizing continue unaffected.

Three limitations of the sizing function should be stated at the outset, because they bound how far its output can be generalized. The routine uses fixed assumptions a 150 W panel rating, five peak sun hours and a system efficiency factor of 0.8 — which are reasonable for southern Nigeria but are not site-specific, are not adjusted seasonally, and do not account for tilt, shading, module temperature or days of autonomy. The peak-sun-hour method itself is among the simplest of the approaches surveyed by Khatib et al. (2013) and yields a first-order estimate rather than an optimized design. Finally, the firmware accepts a single numeric entry and treats it as a daily energy quantity in watt-hours; a user who enters an instantaneous load in watts will therefore obtain a result that is correct only if that load runs for one hour per day. This dimensional ambiguity is a defect of the present implementation rather than of the method, and is addressed in the recommendations.

A further limitation concerns measurement. Sensor outputs were checked for plausibility and for correct response to changing conditions, but no calibrated reference instrumentation was available to establish a formal uncertainty budget. The results reported below are accordingly presented as functional verification, not as a characterization of measurement accuracy.

MATERIALS AND METHODS

The work followed a structured hardware-then-software sequence: component selection and integration, firmware development, establishment of the cloud and web path, and bench testing of the assembled system.

System Architecture

The system is organized as a single acquisition node with two output paths. Sensors mounted on the PV array and battery circuit feed the controller, which drives a local LCD for immediate display and passes the same readings to a serial-attached Wi-Fi module for transmission to the cloud. Figure 1 shows this arrangement in block form.

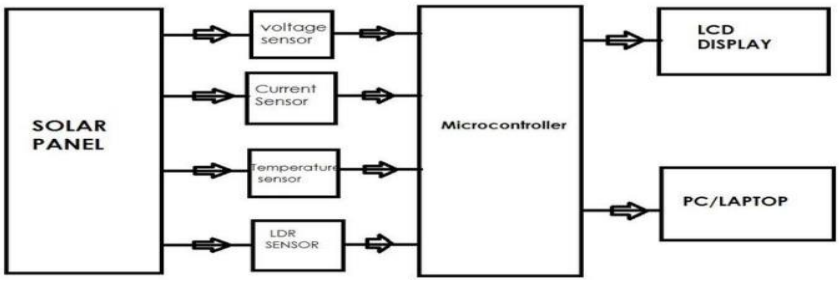

Figure 1: Block diagram of the solar PV monitoring and sizing system, showing the sensor inputs to the controller and the two output paths — local LCD display and remote viewing on a networked computer

Figure 2 sets out the corresponding firmware sequence. On power-up the controller initializes itself and the Wi-Fi module and verifies that a network association has been established, reporting an error to the display if it has not. Once connected, the controller enters its main loop: it reads the analogue and digital sensor channels, formats the values, forwards them to the Wi-Fi module over the serial link for upload, and refreshes the local display. Keypad entry is serviced within the same loop, so that a sizing request is handled without interrupting acquisition.

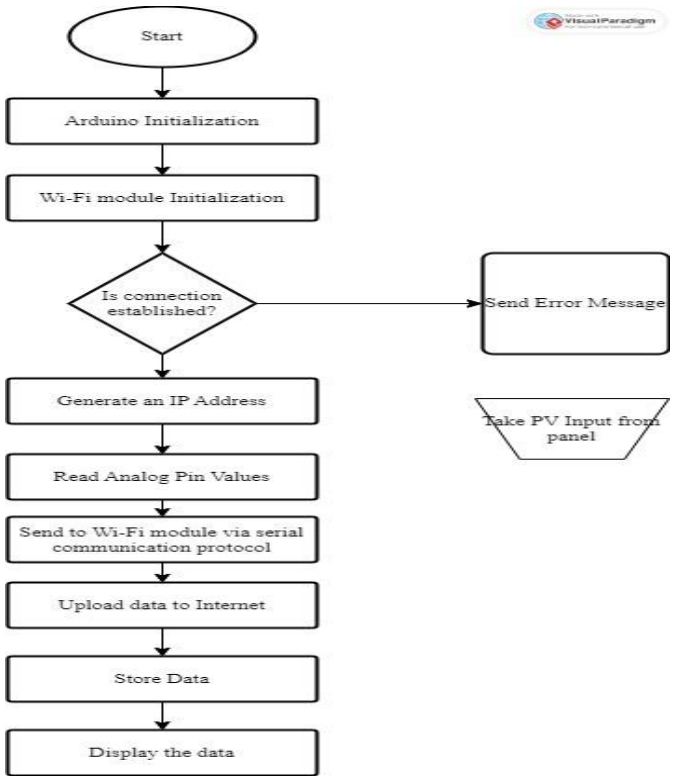

Figure 2: Firmware flowchart for the monitoring and sizing system, from controller and Wi-Fi initialization through connection verification to the acquisition, upload and display loop

Hardware

The controller is an Arduino Uno, chosen for the availability of its ATmega328P toolchain, the breadth of existing sensor libraries and its low unit cost. Analogue channels carry the scaled voltage measurements; the I²C bus carries the current monitor, the light sensor and the LCD backpack; and the remaining digital pins serve the keypad, the DHT11 and the serial link to the radio. Table 1 lists the components and the function each performs.

Table 1: System components and their functions.

Component	Function in the system
Arduino Uno (ATmega328P)	Central controller: sensor acquisition, keypad handling, sizing computation, LCD driving, serial link to Wi-Fi module
ESP-01 (ESP8266) Wi-Fi module	Serial-attached network interface; issues HTTP GET requests to the ThingSpeak channel
Resistive voltage dividers	Scale PV-array and battery terminal voltages into the 0–5 V analogue input range of the controller
INA219 current/voltage monitor	Shunt-based measurement of PV-side current and bus voltage over the I²C bus
ACS712 Hall-effect current sensor	Battery-side current measurement, providing galvanic isolation from the measured conductor
DHT11 sensor	Ambient temperature and relative humidity
BH1750 digital light sensor	Ambient illuminance in lux, used as a proxy indicator of irradiance conditions
20 × 4 character LCD (I²C backpack)	Local display of live readings and sizing results
4 × 3 matrix keypad	Numeric entry of the user's energy demand figure
Charge controller and battery pack	Energy storage and charge management for the standalone configuration
ThingSpeak channel	Cloud data store and time-series visualization of the six transmitted fields
Custom web dashboard	Browser front-end that reads the ThingSpeak channel through its HTTP API and presents the fields in a single view

Voltage measurement uses resistive dividers to bring the PV-array and battery terminal voltages within the 0–5 V analogue input range of the controller. Current is measured by two complementary devices: an INA219 monitor, which senses the drop across a low-value series shunt on the PV side and reports both current and bus voltage over I²C, and an ACS712 Hall-effect sensor on the battery side, which provides galvanic isolation from the measured conductor. Environmental conditions are captured by a DHT11 for temperature and relative humidity and a BH1750 digital light sensor for illuminance. Sensor placement and wiring were arranged to keep analogue runs short and separated from switching conductors, in order to limit noise pickup on the high-impedance divider nodes.

The ESP-01 module is configured through it's AT command set and communicates with the controller over a serial link. Because the module is intolerant of supply sag during transmit bursts, it is powered from a regulated 3.3 V rail with local bulk decoupling rather than from the controller's own regulator.

Figure 3 gives the circuit schematic. It should be read alongside two notes. The schematic was drawn in a simulation environment in which the ATmega328P board is represented by its Arduino Nano symbol; the physical build uses an Arduino Uno carrying the same microcontroller and the same pin functions. The schematic likewise does not show the ESP-01 module, which was added to the physical build after the circuit was drawn and which attaches to the serial pins shown at the lower left of the controller symbol.

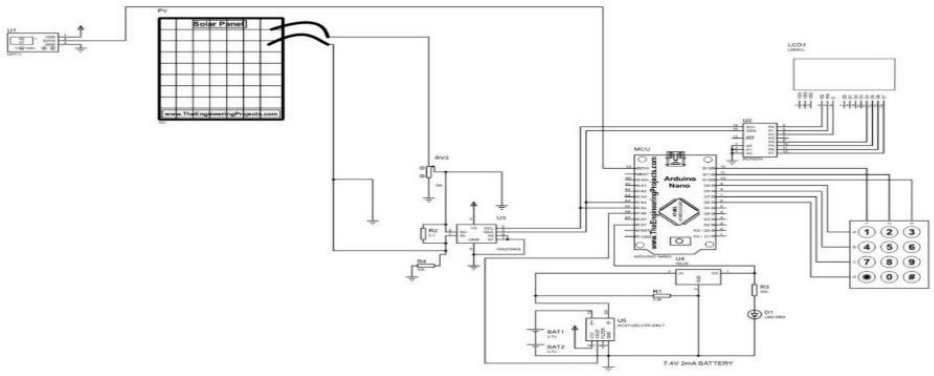

Figure 3: Circuit schematic of the monitoring and sizing system, showing the PV source, series shunt and current monitor, the Hall-effect sensor on the battery side, the DHT11, the regulated supply, and the I²C-driven character LCD and matrix keypad on the controller

No formal calibration procedure was carried out against traceable references. Divider ratios were computed from the nominal resistor values and the resulting scale factors written into the firmware as constants; the current sensors were used with their default sensitivity figures and a zero-current offset taken at start-up. Establishing calibration constants empirically, against a reference meter across the working range of each channel, is identified in Section 5 as necessary further work.

Firmware

The firmware was written in C/C++ and uploaded using the Arduino IDE. It is organized around four tasks: reading the sensor channels, servicing keypad input, computing the sizing estimate, and transmitting data to the cloud. Raw analogue counts are converted to engineering units using the stored scale factors before display; the I²C devices return calibrated quantities directly. Values are refreshed on the LCD on every loop iteration, and uploaded to the cloud on a slower cycle set by the channel's minimum update interval.

Cloud Path and Web Interface

Data reach the user through two layers, and the distinction between them matters for how the contribution of this work should be read. The first layer is ThingSpeak, a third-party IoT analytics platform, which serves as the data store and provides the time-series plots reproduced in Section 3. The ESP-01 writes to it by issuing HTTP GET requests carrying the six field values, using the module's AT command interface. No part of the storage or plotting layer was written for this project; ThingSpeak was adopted as an existing service.

The second layer is a lightweight web dashboard developed for this work. It runs in the browser, reads the same channel through the ThingSpeak HTTP API, and presents the six fields together in a single view rather than as the separate per-field charts that the platform renders by default. Its role is presentational: it does not perform acquisition, storage or analysis. Describing the system as "web-based" should therefore be understood in this restricted sense — a custom front-end over a third-party backend — and not as a claim to have implemented a full monitoring platform from first principles.

Sizing Algorithm

The sizing routine applies the peak-sun-hour method, the most widely used of the intuitive techniques catalogued by Khatib et al. (2013). The number of panels required, N, is given by:

$$N = \frac{E_d}{P_p \times H_{psh} \times \eta} \quad (1)$$

Where E_d is the daily energy demand in watt-hours, P_p is the rated power of a single panel in watts, H_{psh} is the number of peak sun hours per day at the site, and η is a dimensionless system efficiency factor accounting for conversion, wiring and storage losses. The implementation fixes P_p at 150 W, H_{psh} at 5 h and η at 0.8, giving a denominator of 600 Wh per panel per day, and rounds the quotient up to the next whole panel.

Testing Procedure

Testing proceeded in two stages. Each sensor channel was first exercised individually on the bench, with the reading observed on the LCD and on the corresponding ThingSpeak field while the measured quantity was varied — the light sensor by changing the illumination on it, the DHT11 by breathing on the element, and the electrical channels by connecting and disconnecting load. The purpose of these checks was to confirm that each channel was wired correctly, returned values of the expected

magnitude and sign, and responded in the correct direction to a change in the measured quantity.

The assembled system was then run continuously and observed over several logging sessions spread across a number of days, during which the cloud charts were checked for gaps and the latency between a change at the sensor and its appearance in the channel was noted informally. The sizing routine was exercised separately by entering a series of demand values at the keypad and comparing the displayed panel count against the value computed by hand from Equation 1. The number of transmission cycles was not logged systematically, and the latency figures quoted in Section 3 are observational rather than measured; both are treated accordingly.

RESULTS AND DISCUSSION

Sensor Channel Verification

Figures 4 to 9 reproduce the ThingSpeak field charts recorded during the logging sessions. In each chart the horizontal axis is time, labelled by date and clock time, and the vertical axis carries the raw field value transmitted by the node; the platform plots the values as received and does not annotate units, so these are stated in the caption of each figure.

Figure 4: ThingSpeak field 1 — PV-array voltage in volts against time. The trace sits close to 10 V through the early evening period on the left of the plot and rises during the following logging session

The PV voltage channel (Figure 4) tracked the array terminal voltage throughout, sitting in the region of 10 V during the late-afternoon and evening portion of the session and rising when the array was returned to stronger illumination. The channel responded in the expected direction to changes in illumination and load.

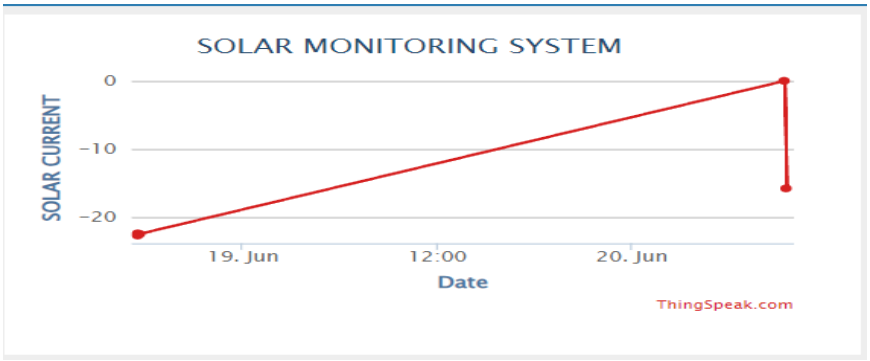

Figure 5: ThingSpeak field 4 — PV-array current in amperes against time. The negative sign convention reflects the orientation of the sensor with respect to the measured conductor

The PV current channel (Figure 5) registered current flow and followed load changes, but the trace is negative throughout, indicating that the sensor was installed with the reverse polarity relative to the assumed direction of conventional current. The magnitude information is usable, but the sign convention requires correction in firmware or by reversing the sensor connection.

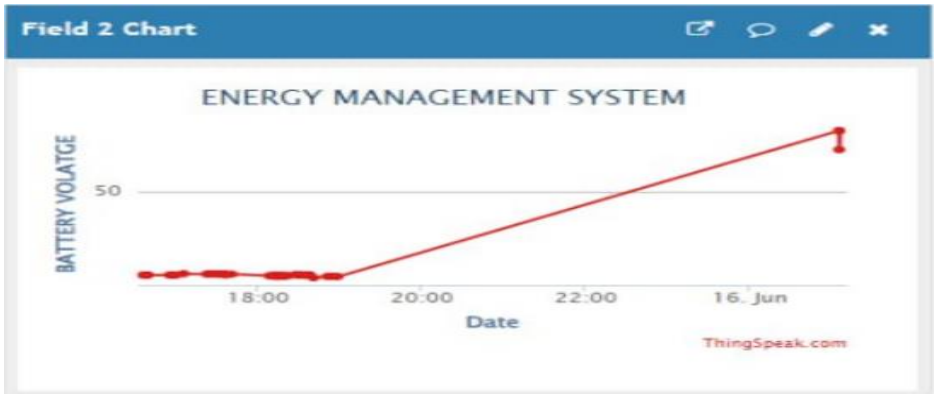

Figure 6: ThingSpeak field 2 — battery voltage in volts against time, over the same window as Figure 4

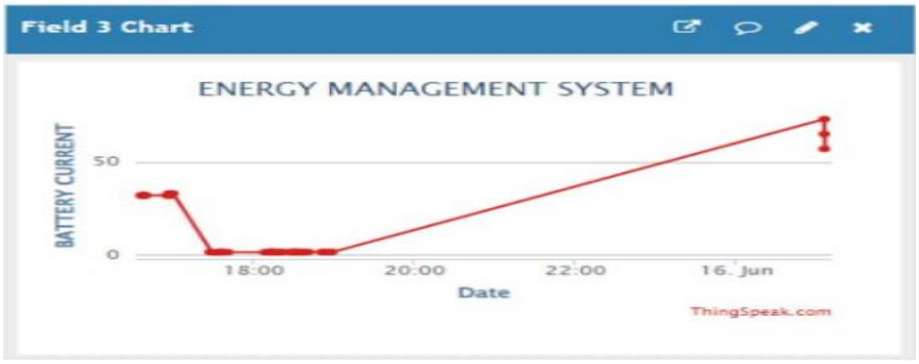

Figure 7: ThingSpeak field 3 — battery current in amperes against time, over the same window as Figure 4

The battery-side channels (Figures 6 and 7) recorded values through charge and discharge and responded to the connection and disconnection of load. The battery current trace in Figure 7 shows a step to near zero early in the session, corresponding to the point at which load was removed, and a subsequent rise. The absolute values on both battery channels are larger than the physical quantities they represent, which points to an incorrect scale factor in the conversion constants rather than to a fault in the sensors themselves; this is discussed in Section 4.

Figure 8: Local LCD display showing simultaneous temperature and relative humidity readings from the DHT11 sensor, at 30.00 °C and 89.00 % RH

The DHT11 channel (Figure 8) returned stable and mutually consistent temperature and humidity readings and responded promptly when warm, moist air was directed at the element. The values shown, 30 °C and 89 % relative humidity, are consistent with indoor conditions in Osogbo at the time of testing.

Figure 9: ThingSpeak field 6 — illuminance in lux against time. The narrow vertical range, 60 to 62 lux, reflects the low and nearly constant indoor lighting level during this short session

The BH1750 channel (Figure 9) reported illuminance in lux and scaled proportionally when the light falling on the sensor was varied. The session shown covers a short indoor interval in which lighting was almost constant, hence the narrow vertical range of the plot; larger excursions were observed when the sensor was shaded and uncovered by hand.

System Integration

With the individual channels verified, all subsystems were operated together. Acquisition, LCD refresh, keypad servicing and cloud upload ran concurrently without the loop stalling or the display becoming unresponsive. Data appeared in the ThingSpeak channel within a few seconds of the corresponding change at the sensor, an interval observed to be of the order of three to five seconds; no systematic latency measurement was performed and no count of transmission cycles was kept, so this figure is reported as an observation rather than a characterized result. Visible gaps in the recorded charts were not observed during the sessions reproduced in Figures 4 to 9, but the absence of gaps in a chart is weak evidence of reliable delivery and no claim of loss-free transmission is made here.

Sizing Verification

The sizing routine was exercised across a range of demand values spanning the small residential systems the device is intended for. Table 2 compares the panel count reported by the system against the value obtained by hand from Equation 1.

Table 2: Sizing Routine Output Across A Range of Entered Demand Values, With P_p = 150 W, H_{psh} = 5 H and H = 0.8

Entered demand (Wh/day)	Unrounded result	Panels reported by system
300	0.50	1
600	1.00	1
900	1.50	2
1200	2.00	2
1500	2.50	3
2400	4.00	4

The system reproduced the hand calculation at every point tested, including the cases at 900Wh and 1500Wh where the unrounded result falls between whole panels and the routine must round upward. Entry, computation and display completed without perceptible delay on repeated trials. The routine was not tested above 2400Wh per day, which lies at the upper end of the range the fixed assumptions are appropriate for.

Discussion

The results establish that the system functions as an integrated whole: sensor acquisition, local display, keypad-driven sizing and cloud transmission operate concurrently on a single low-cost controller, and the sizing routine reproduces the peak-sun-hour calculation correctly across the range tested. Placing a sizing aid alongside a monitoring node is, on the evidence of the literature reviewed in Section 1.1, the aspect of this work with least precedent. The monitoring architecture itself follows a pattern already well established by Aghenta and Iqbal (2019a, 2019b), Sutikno et al. (2021) and others, and no advance over those systems is claimed on that front.

Several results warrant a more careful reading than a functional pass. The PV current trace in Figure 5 is negative throughout, which is a sensor orientation error rather than a measurement failure; the magnitude is informative but the sign must be corrected before the channel can be interpreted directly by a user. More seriously, the battery voltage and current traces in Figures 6 and 7 carry values well above the physical range of the pack and sensors in question. Because the traces respond correctly to load being applied and removed, the most probable explanation is an error in the scale factors applied to those two channels a divider ratio or sensor sensitivity entered incorrectly rather than a hardware fault. This is precisely the class of error that the absence of a calibration step against a reference meter leaves undetected and it demonstrates the practical cost of that omission rather than merely its methodological untidiness.

For the same reason, no accuracy figure is quoted anywhere in this paper. Establishing that a channel responds in the right direction to a change is a necessary condition for correct measurement but not a sufficient one, and without paired readings from a calibrated multimeter, thermometer and lux meter across the working range of each sensor, any percentage accuracy claim would be unsupported by the data presented. The same reasoning applies to the transmission performance reported in Section 3.2: an observed latency of a few seconds and an absence of visible gaps in the charts are consistent with reliable delivery, but establishing a delivery rate would require the node to log its own transmission attempts and outcomes over a defined period, which the present firmware does not do.

The sizing function has a bounded but real utility. Its output is a first-order estimate produced by the simplest of the methods surveyed by Khatib et al. (2013), and it inherits the corresponding limitations: fixed peak sun hours take no account of season or site, a fixed efficiency factor absorbs tilt, shading, temperature derating and battery round-trip losses into one number, and no allowance is made for days of autonomy. Against that, the alternative available to most small buyers is not a rigorous optimization but no calculation at all, and a transparent estimate that states its assumptions is an improvement on that baseline. The dimensional ambiguity noted in Section 1.3 a single numeric entry treated as watt-hours per day is the most pressing defect, since a user entering an instantaneous wattage will under-size the array by the number of hours the load actually runs.

On cost, the system is assembled entirely from commodity development-board components and open-source software, and its bill of materials is accordingly small in absolute terms. No comparative cost analysis against commercial monitoring products was carried out, however and none of the published systems reviewed in Section 1.1 reports a directly comparable cost breakdown, so this paper makes no claim that the system is cheaper than any specific alternative. A component-level cost analysis alongside quoted prices for entry-level commercial monitoring hardware would be required to support such a comparison and is left to further work.

Taken together, the system is best characterized at this stage as a working integration prototype. It demonstrates that monitoring and sizing can be combined in one inexpensive device and that the combination is usable, which was the question this study set out to answer. It does not yet demonstrate metrological adequacy, and the two channels with evidently incorrect scaling show why that distinction should not be elided.

CONCLUSION

This study developed and tested a low-cost system that combines real-time monitoring of a small solar PV installation with an on-device sizing aid. An Arduino Uno acquires PV and battery voltage and current, ambient temperature and relative humidity, and illuminance; a 20 × 4 character LCD provides local display; a matrix keypad accepts a daily energy demand figure; and an ESP-01 module publishes the readings to a ThingSpeak channel, over which a custom web dashboard presents the fields in a single browser view. The sizing routine, based on the peak-sun-hour method, returned results identical to hand calculation at every demand value tested between 300 and 2400Wh per day.

The contribution of the work lies in the integration rather than in any individual subsystem. The monitoring path follows established practice; what is less common in the reviewed literature is the presence of a sizing aid on the same device, which allows a user without design training

to obtain a transparent first-order estimate and then observe the installed system's behavior through the same interface.

Four items of further work follow directly from the limitations documented above. First, each measurement channel should be calibrated against traceable reference instruments across its working range, with paired readings and residuals reported, so that a defensible uncertainty figure can replace the functional verification presented here; the incorrect scale factors evident on the battery channels should be resolved as part of this exercise, and the PV current sensor orientation corrected. Second, the firmware should distinguish explicitly between an instantaneous load in watts and a daily energy demand in watt-hours, and should expose the panel rating, peak sun hours and efficiency factor as user-settable parameters rather than fixed constants, so that the estimate can be adapted to site and season. Third, the node should log its own transmission attempts and outcomes so that delivery reliability and latency can be reported quantitatively over a defined observation period. Fourth, a component-level cost analysis set against quoted prices for commercial monitoring hardware would allow the affordability of the approach to be assessed on evidence rather than asserted.

REFERENCES

Aghaei, M., Kumar, N. M., Eskandari, A., Ahmed, H., De Oliveira, A. K. V., & Chopra, S. S. (2020). Solar PV systems design and monitoring. In Photovoltaic Solar Energy Conversion: Technologies, Applications and Environmental Impacts (pp. 117–145). Academic Press. https://doi.org/10.1016/B978-0-12-819610-6.00005-3

Aghenta, L. O., & Iqbal, M. T. (2019a). Development of an IoT based open source SCADA system for PV system monitoring. 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), 1–4. https://doi.org/10.1109/CCECE.2019.8861827

Aghenta, L. O., & Iqbal, M. T. (2019b). Low-cost, open source IoT-based SCADA system design using Thinger.IO and ESP32 Thing. Electronics, 8(8), 822. https://doi.org/10.3390/electronics8080822

Ali, A. H., El-Kammar, R. A., Hamed, H. F. A., Elbaset, A. A., & Hossam, A. (2024). Smart monitoring technique for solar cell systems using internet of things based on NodeMCU ESP8266 microcontroller. International Journal of Electrical and Computer Engineering, 14(2), 2322–2333. https://doi.org/10.11591/ijece.v14i2.pp2322-2333

Allegro MicroSystems. (2023). ACS712: Fully integrated, Hall-effect-based linear current sensor IC with 2.1 kVRMS isolation and a low-resistance current conductor [Datasheet]. https://www.allegromicro.com/en/products/sense/current-sensor-ics/zero-to-fifty-amp-integrated-conductor-sensor-ics/acs712

Aosong Electronics. (2019). DHT11 digital-output relative humidity and temperature sensor/module [Datasheet]. https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

Chanchangi, Y. N., Adu, F., Ghosh, A., Sundaram, S., & Mallick, T. K. (2023). Nigeria's energy review: Focusing on solar energy potential and penetration. Environment, Development and Sustainability, 25(7), 5755–5796. https://doi.org/10.1007/s10668-022-02308-4

Energy Internet: Systems and Applications [1st ed.] 9783030454524, 9783030454531 - DOKUMEN.PUB. (n.d.). Retrieved June 15, 2024, from https://dokumen.pub/energy-internet-systems-and-applications-1st-ed-9783030454524-9783030454531.html

Khatib, T., Mohamed, A., & Sopian, K. (2013). A review of photovoltaic systems size optimization techniques. Renewable and Sustainable Energy Reviews, 22, 454–465. https://www.sciencedirect.com/science/article/abs/pii/S1364032113001251

Panwar, N. L., Kaushik, S. C., & Kothari, S. (2011). Role of renewable energy sources in environmental protection: A review. Renewable and Sustainable Energy Reviews, 15(3), 1513–1524. https://ideas.repec.org/a/eee/rensus/v15y2011i3p1513-1524.html

Rahman, M. M., Selvaraj, J., Rahim, N. A., & Hasanuzzaman, M. (2018). Global modern monitoring systems for PV based power generation: A review. Renewable and Sustainable Energy Reviews, 82(3), 4142–4158. https://doi.org/10.1016/j.rser.2017.10.111

ROHM Semiconductor. (2011). BH1750FVI: Digital 16-bit serial output type ambient light sensor IC [Datasheet]. https://www.mouser.com/datasheet/2/348/bh1750fvi-e-186247.pdf

Sutikno, T., Purnama, H. S., Pamungkas, A., Fadlil, A., Alsofyani, I. M., & Jopri, M. H. (2021). Internet of things-based photovoltaics parameter monitoring system using NodeMCU ESP8266. International Journal of

Electrical and Computer Engineering, 11(6), 5578–5587. https://doi.org/10.11591/ijece.v11i6.pp5578-5587

Wibowo, L., Wahyusari, R., Yuwono, T., & Shofia, A. (2024). IoT-based high-accuracy monitoring system for on-grid photovoltaic power system using NodeMCU ESP8266 and PZEM004T. Journal of Mechatronics, Electrical Power, and Vehicular Technology, 15(2). https://mev.brin.go.id/mev/article/view/823