2025 33rd Southern African Universities Power Engineering Conference (SAUPEC)

A practical evaluation of an IEC 61499 standard-based Low-Cost Gateway for photovoltaic power plants

Emmanuel Luwaca and Senthil Krishnamurthy
Centre for Substation Automation and Energy Management Systems (CSAEMS)
Center for Intelligent Systems and Emerging Technologies (CSIET)
Department of Electrical, Electronics and Computer Engineering
Cape Peninsula University of Technology, PO BOX 1906, Bellville 7535, South Africa
e.luwaca@gmail.com and KrishnamurthyS@cput.ac.za

2025 33rd Southern African Universities Power Engineering Conference (SAUPEC) | 979-8-3315-3516-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/SAUPEC65723.2025.10944445

***Abstract*— Communication plays a significant role in an electrical grid that includes Distributed Energy Resources (DERs). Modern control and automation systems used by these systems are predominantly distributed using domain standards such as IEC 61131-3. However, the IEC 61131-3 standard does not address communication. The IEC 61499 standard, in contrast, introduces the use of the Service Interface Function Block (SIFB) concept and high-level communication patterns to achieve hardware-independent access to communication services. This paper recommends the application of the IEC 61499 standard for DER control and automation and the application of SIFB to facilitate the ease of system integration in a DER plant. A low-cost gateway application using open-source IEC61499 tools is implemented, and its simulation results are presented.**

Keywords—Automation, distributed control, distributed energy resource, IEC61499, Matlab

I. INTRODUCTION

Southern Africa's regional energy demand is predicted to increase ten to fourteen times between 2010 and 2070. There has been a drive to deploy renewable energy systems as an ecologically acceptable generating strategy in recent years. Over the past two decades, South Africa has embraced renewable energy to increase its generating capacity. As part of South Africa's Integrated Resource Plan (IRP) 2019, the country plans to roll out renewable power and decommission old coal power plants in phases.

However, the veritable nature of renewable energy resources as part of a developing country's energy mix presents technical and economic challenges. Renewable energy resources challenge power systems regarding stability, dependability, and interoperability. However, new DER systems are being deployed across the globe for social, economic, and environmental benefits. DER systems form part of the Smart Grid (SG) framework.

The objectives of an intelligent grid include improving energy efficiency, reducing the carbon footprint, and providing an open framework for information exchange between power system devices [1] given the dispersed nature of the devices in a DER power plant, communication is vital in controlling it. The currently installed power plant uses algorithms based on the well-established IEC 61131-3 standard with a central controller. Most DER systems commissioned predominantly use proprietary or basic protocols such as Modbus. There are no defined standardized interfaces, protocols, and services. Using legacy communication protocols has resulted in a firm reliance on the original equipment manufacturers for the plant owners during all project phases.

In contrast, the IEC 61499 standard is enunciated as a resource for providing a conceptual reference architecture for distributed control and automation systems. Since its publication in 2005, the IEC 61499 standard architecture has stimulated the development of new engineering technologies that offer portability, reusability, and interoperability for distributed control and automation applications [2]. IEC 61499 open-source tools such as the 4DIAC integrated development platform also support communication protocols such as TCP/IP, Modbus, and IEC 62541- OPC Unified Architecture (OPC UA) using open-source libraries. Through the sections in this document, this study explores how the existing open-source tools can be used to monitor and control a solar power plant. A solar plant is simulated using Matlab, and the various measurements that are key in monitoring and controlling a solar power plant, such as irradiance, temperature, power (active and reactive), and metering, are monitored using third-party tools.

The International Electrotechnical Commission (IEC) published the IEC 61499 standard in 2005. This standard presents a new, innovative paradigm for industrial automation systems. However, it is presently not widely supported and utilized within industry. [3] Highlights what the paper argues as unsubstantiated claims of the main features of the IEC 61499 standard that are promoted. Whilst the industry does not so widely support the IEC 61499 standard, it can form the basis for developing industry 4.0 applications as detailed by [4]. [5] Evaluate using third-party communication stack, i.e., Sisco IEC 61850 stack, System Corp stack, and on IEC 61499 compliant Integrated Development Environments (s) such as 4Diac and isaGraf. To make IEC 61499 more applicable to industry, Three main challenges with the transition to IEC 61499 are identified in [6], including industrial concerns and technical issues that research on transforming the modeling techniques and implementation tools is an ongoing activity, whilst [7] discusses a practical application of the IEC 61499 standard for structuring Cyber-Physical Systems. To ensure industry readiness, [8] encourages potential adopters to monitor developments in the domain closely. This document explores one possible use case for IEC 61499 within distributed energy resources.

The paper presents an introduction in section 1 and a MATLAB TCP communication setup in section 2. In section 3, we discuss adding communication protocols to the 4Diac

979-8-3315-3516-2/25/$31.00 ©2025 IEEE

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.

Forte runtime, and in section 4, we discuss the Modbus to OPC gateway implementation using IEC 61499 service interface function blocks on 4Diac. Finally, we conclude the paper presented in this document and discuss ongoing activities in our research.

II. MATLAB TCP COMMUNICATION

This section describes the TCP/IP communication blocks in Matlab/Simulink. This study examines the Simulink TCP client and server communication block, i.e., how Matlab encodes and decodes the various datatypes transmitted over TCP/IP communication. Simulink can communicate with remote applications using the standard MATLAB Transport Communication Protocol (TCP) over Internet Protocol (IP) communication blocks. Client block enables sending live data from the Simulink model to an application; in this case study, the remote application is a Python code running on a Linux Ubuntu Virtual Machine (VM). The base element of the block is the S-function block, which uses the C MEX file. [9].

A. TCP/IP Communication

Using TCP/IP as data transport between applications is a cost-effective, easy, and flexible method of sharing data between systems. TCP is the underlying transport protocol that is used by most of the industrial communication protocols today. The TCP/IP stack is an open, standardized reference model to build a protocol suite, which means that two compatible end-system applications can communicate regardless of their underlying architectures. The RFC 1180 document provides a detailed TCP/IP communication model tutorial. With TCP/IP communication, similar to most layered communication protocols that utilize the Open-system Interconnect (OSI) model, the fundamental elements of sending and receiving data are collected into the groups that become the layers. Each layer provides a small set of specific services to the layer below and the layer above, thus providing independence between the layers whilst maintaining the service provided by one layer to another [10]. The protocols defined at each layer establish a peer-to-peer relationship with the corresponding layer of the receiving device. TCP/IP provides reliable communication and requires that the receiving devices acknowledge the reception of a packet. It also provides methods such as re-transmission, keep-alive, etc., that continuously ensure the integrity of the communication link between two devices.

B. Simulink PV Plant simulation with TCP client implementation.

To enable a more accurate depiction of a real-world plant, this section discusses the simulation of a PV plant in Matlab. A well-known control method is used in PV systems to provide the maximum available active power from the PV array to the grid. However, to prevent grid instability caused by the high penetration of renewable energy, the active power is controlled with the consideration of grid frequency variation and voltage stability in a similar manner to traditional power plants [11]To solve these problems, grid-connected PV inverter systems should supply both active and reactive power while delivering maximum power to the grid whenever possible. Maximum power point tracking ensures maximum power is delivered to the grid. Figure 1, shows the block diagram of the PV system model.

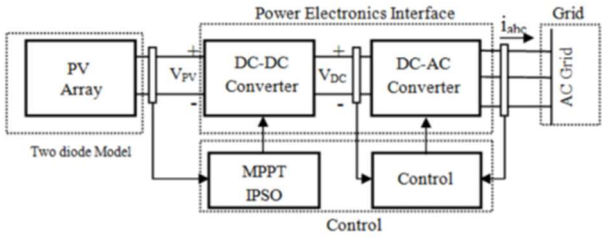


Figure 1: Block diagram of PV System Model ***Error! Reference source not found.***

The PV system depicted in Figure 1 is modelled in Matlab in Figure 2. The Matlab array is built of parallel strings of modules, each string consisting of modules connected in series. The PV Array block is a five-parameter model using a light-generated current source (IL), diode, series resistance (Rs), and shunt resistance (Rsh) to represent the irradiance- and temperature-dependent I-V characteristics of the modules

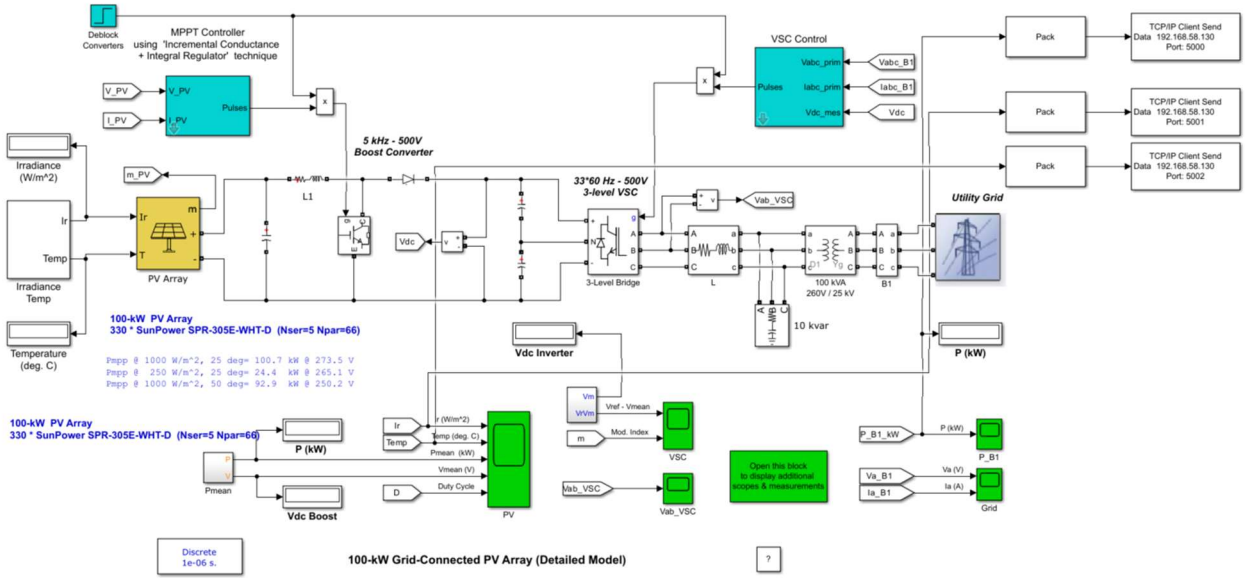


Figure 2: 100kW Grid-Connected PV Array

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.

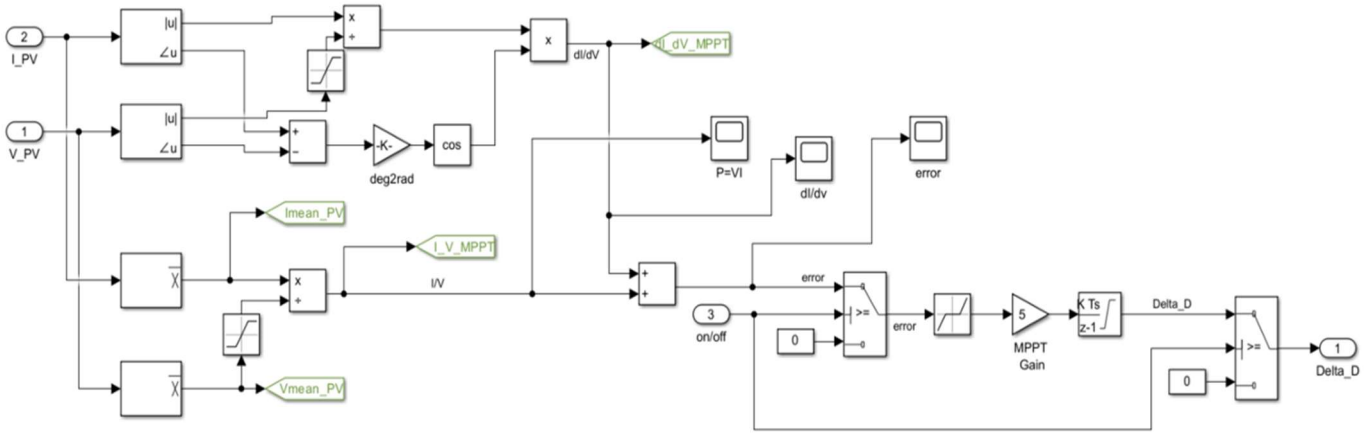


Figure 3: Maximum power point tracking by incremental conductance method

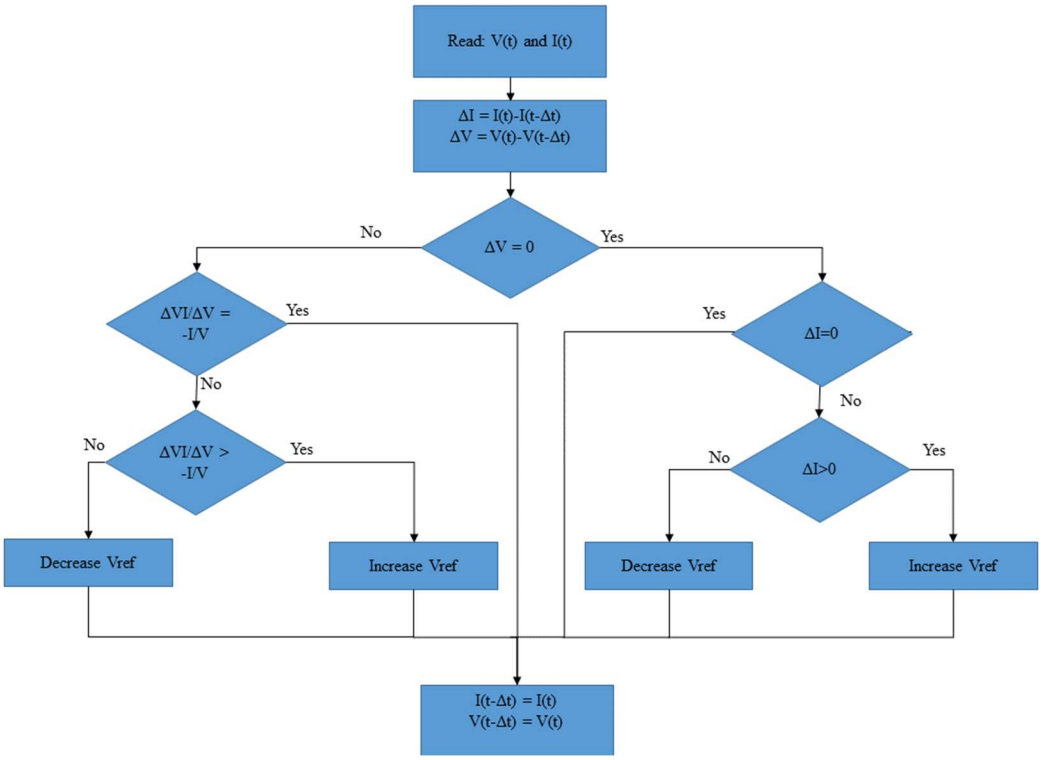


Figure 4: Flowchart of the IC Algorithm [13]

In Figure 2, the PV array is grid connected using a DC-DC converter, three-phase, three-level Voltage Source Converter, and Controllers [13]. The modeled PV plant has Maximum Power Point Tracking (MPPT) implemented using the 'Perturb & Observe' technique. The MPPT controller is based on the IC+IR (Incremental Conductance + Integral Regulator) technique in this model, shown in Figure 4. Incremental conductance looks to increasingly contrast the ratio of the derivative of conductance with the instantaneous conductance and is based on the fact that the power slope of a photovoltaic is null at the maximum power point (dP/dV = 0), and can be found in terms using (1) and (2) [13].

$$\frac{\delta P_{PV}}{\delta V_{PV}} = \frac{\delta(V_{PV} * I_{PV})}{V_{PV}} = V_{PV} * \frac{\delta I_{PV}}{\delta V_{PV}} + I_{PV} = 0 \quad (1)$$

Where the change in photovoltaic current is ∂IPV, the change in photovoltaic voltage is ∂VPV, and ∂PPV is the change in photovoltaic power, respectively. Equation (1) can be written as:

$$\frac{\delta I_{PV}}{\delta V_{PV}} = -\frac{I_{PV}}{V_{PV}} \approx \frac{\Delta I}{\Delta V} \quad (2)$$

Where photovoltaic current and voltage ΔI and ΔV are the increments, respectively.

The IC+IR technique depicted in Figure 4 adjusts (increases and decreases) Vref and Iref until I(t-Δt) = I(t) and V(t-Δt) = V(t) so that the modeled PV array delivers a maximum power of 100 kW at 1000 W/m^2 sun irradiance and 25˚C temperature.

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.

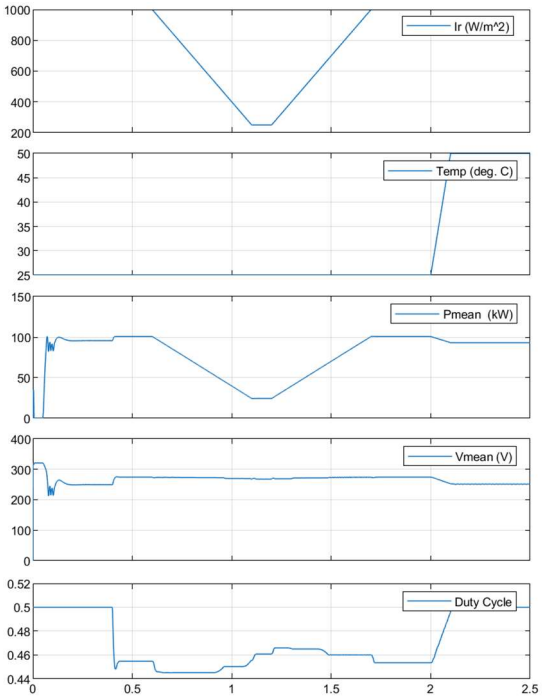


Figure 5: Output Waveform of Simulated PV plant

Using the Simulink model of the PV plant presented in Figure 5 has enabled us to observe the behavior of a PV plant. The model allows us to test the simulated PV plant's stability and build an understanding of I-V and PV output characteristics under various conditions. The measured temperature on the PV panel shows a gradual increase based on the simulated conditions, from a cold panel through the shading period and on the steady-state conditions. In the study simulation, we also enabled communication from Simulink with external devices using TCP/IP. A python and IEC 61499 4Diac TCP server are used later to receive the transmitted data for external consumption and can be used by other applications. Different datatypes can be packed for transmission, such as "uint," "double," "bool," etc. The application has an interest in "Real" floating-point numbers, so the data is packaged for transmission as a double.

C. TCP Server using Python

Python provides two levels of access to network services, and through the "socket" library, it gives the essential socket support in the underlying operating system, facilitating the implementation of client (s) or server (s) connections. Python supports the creation of custom messages by interacting with the operating system kernel using sockets such as Socket.SOC_STREAM. These sockets can create a TCP Server that listens for connections on a specified communication port.

The data received by the Python TCP server can be logged onto an SQL database or written to a file for storage and validation purposes. Matching the encoding mechanism the sending device uses when decoding the packet is essential, as using the in-correcting decoding mechanism results in an inaccurate representation of the data from the sending device. The Python 'struct library' was used during this test, a module that converts between Python values and C structs represented as Python bytes objects. The library can use the format string's first character to indicate the packed data's byte order, size, and alignment.

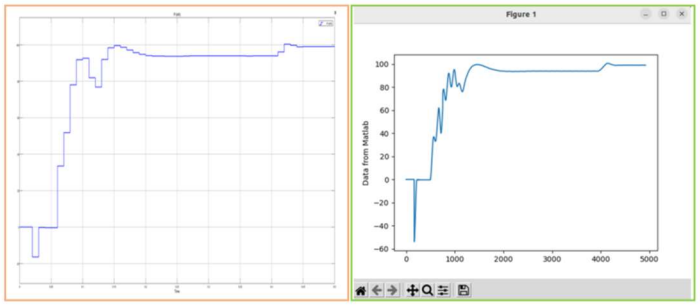


Figure 6: Data captured by the Python script compared with Matlab scope data.

The right-hand side of Figure 6 validates that the Python script receives the sent data; however, the script drops packets more significantly than the hard-coded payload size because of challenges with retransmitted packets. The challenge experienced with retransmission is what would seem like variable, un-deterministic payload lengths. In this investigation, python was utilized to understand how Matlab publishes the TCP data and how the payload data is encoded and decoded. The understanding developed is then utilized when implementing the TCP server using IEC 61499, as shown in Figure 7Using the developed function blocks in IEC 61499 and data exchange, the output of the PV plant is controlled by varying the duty cycle on the Insulated-Gate Bipolar Transistor (IGBT) of the power inverter. Section IV discusses how the data exchanged by the 4Diac application is transferred to upstream systems such as the Supervisory and Data Acquisition (SCADA) system.

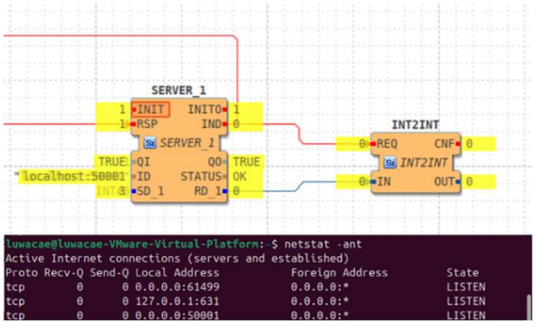


Figure 7: TCP Server implemented using IEC 61499

III. TESTING COMMUNICATION PROTOCOLS ON 4DIAC

4DIAC is an open-source project that contains an Integrated Development Environment (IDE) for developing IEC 61499 standard-compliant distributed control applications [9]. The 4DIAC IDE contains standard elements for IEC 61499 projects such as System, Application, Device, Function Blocks (Basic, Composite, and SIFB), etc.,

The 4diac runtime (FORTE) is available in two versions; there is a pre-built runtime environment for both Windows and Linux operating systems. A source code of the runtime is also available to develop custom Function Blocks and to build additional functionality onto the 4diac (Forte) runtime. Using the source-code version, users can also build communication libraries as part of the 4diac runtime, enabling communication functionality without the need to programmatically interface with the communication libraries. The 4Diac Modbus communication interface requires the libmodbus library to

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.

work [15]. Figure 8, shows how the Forte library can be compiled with additional functionality of custom function blocks and communication libraries using CMAKE software. Once the libmodbus library is compiled as part of the 4Forte runtime, the IEC 61499 standard defined generic CLIENT/SERVER SIFB for bi-directional data exchange and PUBLISH/SUBSCRIBER SIFB for unicast data exchange are utilized to facilitate communication with external devices. The standard CLIENT/SERVER SIFB is defined in Extensible Markup Language (XML) and is easily customizable based on the number of inputs and output required within the application. Figure 9, illustrates Modbus TCP communication between the 4Diac IDE and a substation Intelligent Electronic Device (IED). The Client SFIB block in this illustration has been modified to receive and send eight tags of the "ANY" data type. The "ANY" data is a placeholder for defining data objects. This is a feature of C++ that provides a type-safe container to store a single value of any type. The "ANY" data gets associated during runtime based on the datatype connected to the input or output.

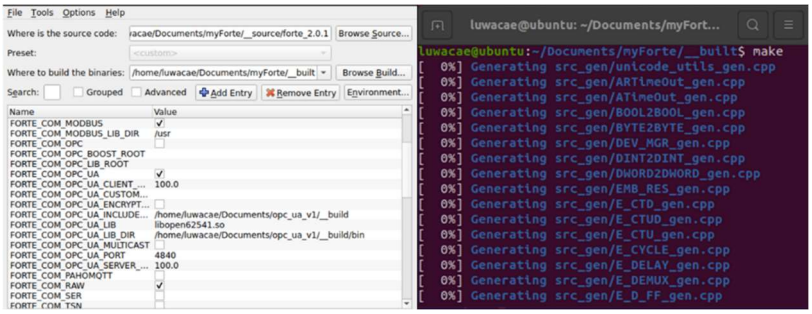


Figure 8: Building additional functionality onto the Forte Runtime

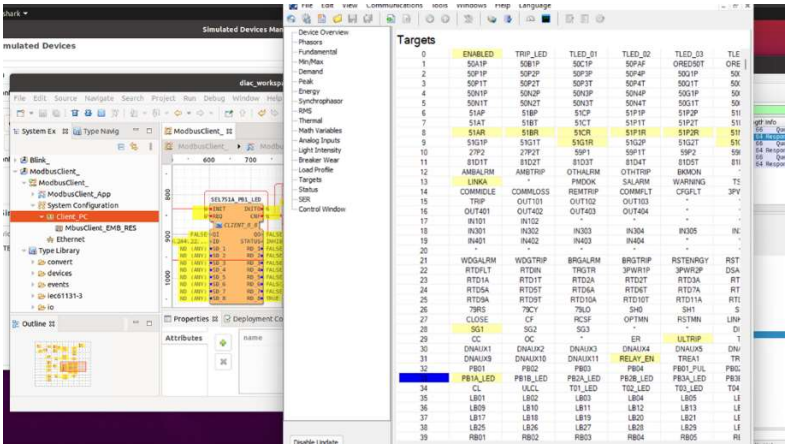


Figure 9: Using 4diac client SIFB to communicate with the SEL751A using Modbus protocol.

In Figure 9, the 4diac (FORTE) runtime is used to interface with the physical IED using Modbus to retrieve data and serve this data to upstream systems. Modbus is a widely used communication protocol. In Figure 9, the SIFB connects to the SEL751A relays and reads the Modbus values of row 33, which are the pushbutton LEDs on the relay. The values returned are represented in a structured data type {FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE} highlighted in green in Figure 9, and are a representation of the relay LED status highlighted in red. Highlighted in yellow. The Modbus response packet to a poll of the input registers is shown in Figure 10.

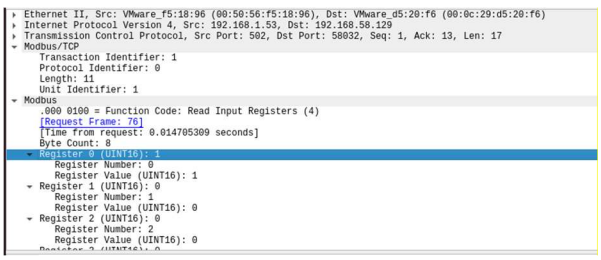


Figure 10: Modbus TCP Input register response

Section II discussed simulating a PV plant using Matlab Simulink, which would allow access to other Photovoltaic (PV) plant data. We also discussed TCP/IP and Modbus communication. Section IV details the low-cost communication gateway application implemented using 4Diac.

IV. Implementing a Low-Cost Gateway Application using 4diac – OPC UA

OPC UA can be included in FORTE runtime, similar to the Modbus protocol in the section III. Another alternative for adding OPC UA support discussed in papers is to have SIFBs that provide at least part of the functionality or the development of an OPC wrapper that can interact with the runtime to give the protocol support [16]. The OPC UA open source library (open6254) can be compiled into the Forte runtime in 4diac and used as a standard service interface function block defined in the IEC 61499 standard. A prototype gateway device has been developed on a Linux Ubuntu using 4Diac, as shown in Figure 11. Data is sourced from Matlab using TCP/IP client/server communication and from a Modbus server. This data is then available to upstream streams such as SCADA master using OPC UA. Depicted in Figure 11

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.

is the Modbus TCP server simulation highlighted in orange; the IEC 61499 gateway application reads the Modbus registers. The collected values are served to the OPC client (UA Expert).

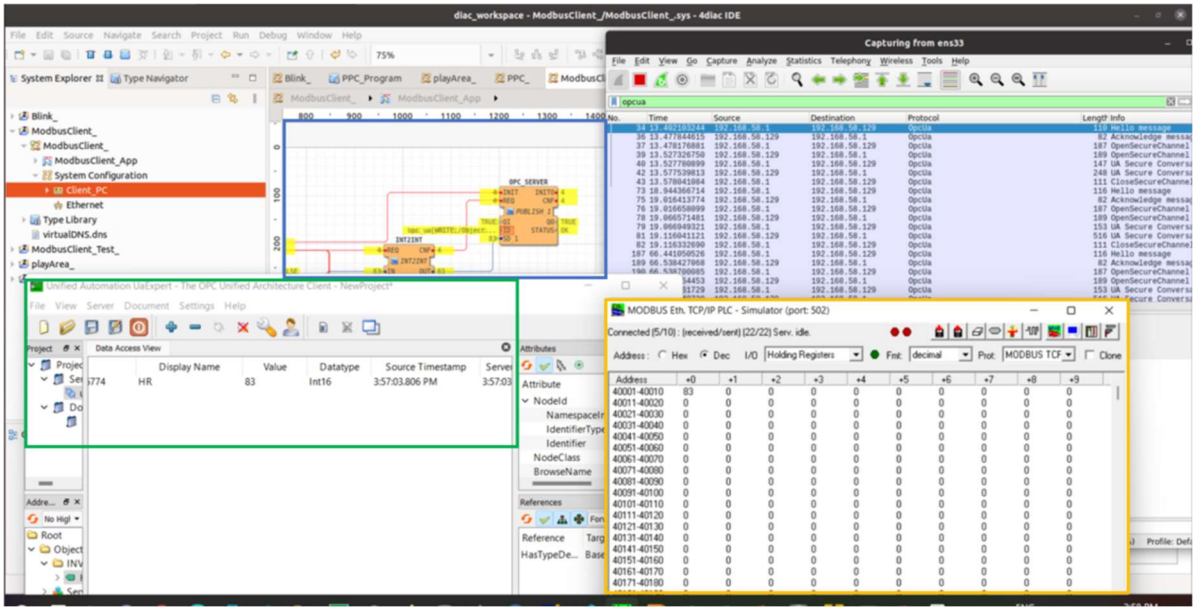


Figure 11: OPC Server implementation and Wireshark capture

CONCLUSION AND FUTURE WORK

The IEC 61499 standard holds great potential, as it proposes a distributed architecture for designing and implementing industrial automation systems. However, the industry has not fully adopted it. Research into practical applications and further work into implementation is required to highlight the standard's benefits.

This paper represents the steps to model a Low-Cost Gateway for PV Plants. The focus is predominantly on communication and implementing a low-cost communication gateway to ease system integration. This paper provides a use-case scenario using 4Diac, which enables users to extend the runtime's functionality, enabling industrial communication protocols to support the development of industrial applications. This paper's proposed research methodology and simulation results provide system integration that can be achieved using open-source tools.

ACKNOWLEDGEMENT

The authors acknowledge Cape Peninsula University of Technology for offering the facilities to conduct this research at CSAEMS and CISET research centers. This research was partly funded by the NRF Thuthuka Grant, Number 138177, Eskom Tertiary Education Support Programme (TESP) funding.

REFERENCES

[1] Kumar, R., Srinivasan, S., Indumathi, G. and Dzitac, S. (2018), A Communication Viewpoints of Distributed Energy Resource, in 'Soft Computing Applications: Proceedings of the 7th International Workshop Soft Computing Applications (SOFA 2016), Volume 1 7', pp. 107--117.

[2] Vyatkin, V. (2011), 'IEC 61499 as an enabler of distributed and intelligent automation: State-of-the-art review', IEEE Transactions on Industrial Informatics 7(4), 768--781.

[3] Thramboulidis, K. (2013), 'IEC 61499 vs. 61131: A comparison based on misperceptions', arXiv preprint arXiv:1303.4761.

[4] Andrйn, F., Strasser, T., Zoitl, A. and Hegny, I. (2012), A reconfigurable communication gateway for distributed embedded control systems, in 'IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society', pp. 3720--3726.

[5] Abdulkadir, M., A. H. M. Yatim, and S. T. Yusuf. "An improved PSO-based MPPT control strategy for photovoltaic systems." International Journal of Photoenergy 2014.1 (2014): 818232

[6] Lyu, G. and Brennan, R. W. (2020), 'Towards IEC 61499-based distributed intelligent automation: A literature review', IEEE Transactions on Industrial Informatics 17(4), 2295--2306.

[7] Monroy Cruz, E.,Garcia C., Luis R., and Cruz Salazar, L. (2023). 'Structuring Cyber-Physical Systems for Distributed Control with IEC 61499 Standard. IEEE Latin America Transactions 21. 251-259. 10.1109/TLA.2023.10015217.

[8] Christensen, J. H., Strasser, T., Valentini, A., Vyatkin, V. and Zoitl, A. (2012), 'The IEC 61499 function block standard: Overview of the second edition', ISA auto. Week 6, 6--7.

[9] Sysel, M. (2011), Matlab/Simulink TCP/IP communication, in 'Proceedings of the 15th WSEAS international conference on Computers', pp. 71--75.

[10] Socolofsky, T. J. and Kale, C. J. (1991), 'TCP/IP tutorial'.

[11] Krishnamurthy, S. and Mohlwini, E. X. (2016), Voltage stability index method for optimal placement of capacitor banks in a radial network using real-time digital simulator, *in* '2016 International Conference on the Domestic Use of Energy (DUE)', pp. 1-8.

[12] https://ch.mathworks.com/help/sps/ug/detailed-model-of-a-100-kw-grid-connected-pv-array.html

[13] Dighore, V. and Welankiwar, A. (2021), 'Design and Simulation of 100 MW Photovoltaic Power Plant Using Matlab Simulink', 258-262.

[14] De Brito, Moacyr & Sampaio, Leonardo & Galotto, Luigi & Melo, Guilherme & Canesin, C.A.. (2011). 'Comparative analysis of MPPT techniques for PV applications'. 3rd International Conference on Clean Electrical Power: Renewable Energy Resources Impact, ICCEP 2011. 10.1109/ICCEP.2011.6036361.

[15] Online:https://eclipse.dev/4diac/en_help.php?helppage=html/communication/modbus.html

[16] Seilonen, I., Vyatkin, V. and Atmojo, U. D. (2019), OPC UA Information Model and a Wrapper for IEC 61499 Runtimes, *in* , pp. 1008-1013.

Authorized licensed use limited to: Cape Peninsula University of Technology. Downloaded on November 23,2026 at 08:24:33 UTC from IEEE Xplore. Restrictions apply.