الجمهورية الجزائرية الديمقراطية الشعبية

Peoples Democratic Republic of Algeria

وزارة التعليم العالي والبحث العلمي

Ministry of Higher Education and scientific research

المدرسة الوطنية العليا للتكنولوجيا

National Higher School of Technology

Department of Electrical Engineering and Industrial Computing

Final Year Project to Obtain the Diploma of Engineering

Field:

Electronics

Specialty:

Embedded Systems

- Subject -

Development of a Communication Module For Smart Grids and Electric Meters

By

GHOUMRASSI Hafsa

Examination Committee:

Ms Karima Boulaam	Chair
Ms Malika BOUTERFAS	Supervisor
M. Tarek CHERIFI	Co-supervisor
M. Toufik BOUDIAR	Examiner
Ms. Lamia HABANI	Examiner

Algiers, the 02/07/2023

Academic year 2022 –2023

Acknowledgements

First and foremost, I have to thank my research supervisors, Mme. Malika BOUTERFAS and M. Tarek CHERIFI. Without their assistance and dedicated involvement in every step throughout the process, this paper would have never been accomplished. I would like to thank you very much for your support and understanding over these past months.

I would also like to thank Mme. Karima Boulaam for taking the time to preside over the jury of the validation of this dissertation, and M. Toufik BOUDIAR and M. Lamia HABANI for evaluating the work and elevating it by their constructive criticism.

Without forgetting to appreciate the support of my family and friends who have kindly supported and encouraged me through this journey.

Table of contents

General Introduction 1

Chapter I: Introduction to Smart Grids and Intelligent Electrical Counters

I-1) Introduction 4

I-2) Smart Grid: History and Architectures 4

I-2-1) Advances in Grid Modernization 5

I-2-2) Importance of Communication in Smart Grids 6

I-3) Smart Electrical Meter: Concept and Components 7

I-3-1) Concept of a Smart Electrical Meter 7

I-3-2) Main Functions of a Smart Electrical Meter 8

I-3-3) Components of a Smart Electrical Meter 8

I-4) Communication Protocols in the Smart Grid 9

I-4-1) Overview of Communication Protocols 9

I-4-2) Architecture of a SCADA System 10

I-4-2-1) A Remote Terminal Unit (RTU) 10

I-4-2-2) A Master Terminal Unit (MTU) 11

I-4-3) Communication Protocols for Metering and Monitoring in a SCADA System 12

Chapter II : communication protocols in smart grids and overview of IEC 60870-5 standard

II-1) Introduction 14

II-2) Communication Protocols in Smart Grids: A Comparative Overview 14

II-2-1) Introduction to Communication Protocols in Smart Grids 14

II-2-2) Types of Communication Protocols 14

II-2-2-1) Wired Protocols 14

II-2-2-2) Wireless Protocols 15

II-2-2-3) Fieldbus Protocols 15

II-2-2-4) Utility-specific Protocols 15

II-3) IEC 60870-5 standard 16

II-3-1) Derivation of IEC 60870-5 16

II-3-2-1) IEC 60870-5-1 16

II-3-2-2) IEC 60870-5-2 17

II-3-2-3) IEC 60870-5-3 17

II-3-2-4) IEC 60870-5-4 17

II-3-2-5) IEC 60870-5-5 18

II-3-2) IEC 60870-5 10x Protocols 19

II-3-3) Enhanced Performance Architecture (EPA) 20

II-4) Architecture of IEC 60870-5-101 Protocol 21

III-5) Physical Layer of IEC 60870-5-101 21

III-5-1) Network Configurations 22
II-6) Data Link Layer 22
II-6-1) IEC 60870-5-101 Telegram Sttructure 23
II-6-1-1) Frame with fixed length 23
II-6-1-2) Frame With Variable Length 24
II-6-2) Primary And Secondary 26
II-6-3) Unbalanced And Balanced Transmission 27
II-6-4) Service Procedures 27
II-6-5) The Control Field 28
II-7) Application Layer 31

Chapter III: Implementation and tests

III-1) Introduction 33
III-2) Hardware and Software Setup 33
III-2-1) Hardware Setup 33
III-2-1-1) Raspberry Pi 33
III-2-1-2) Arduino 34
III-2-2) Software Setup 34
III-2-2-1) Python 34
III-2-2-2) Virtual Serial Port 35
III-2-2-3) Docklight 36
III-3) Code Structure And Organization 36
III-4) The Implemented iec_101_functions Python Library 37
III-5) Tests And Results 51
III-5-1) Link Reservation Results 51
III-5-2) Common Error Checking 53
III-5-3) Log Files 53
III-5-4) MTU And Data Extractor 54
III-5-5) Conversion Errors 56
III-5-6) Latency 57
Conclusion and Outlook 59

LIST OF TABLES

TABLE I-1: A BRIEF COMPARISON BETWEEN THE EXISTING GRID AND THE SMART GRID 5
TABLE II-2: Selected standard provisions of the defined telecontrol companionstandard 19
TABLE II-3: STANDARDS SELECTIONS FOR IEC 60870-5-101 21
TABLE II-4: THE RELATION BETWEEN LINK LAYER IEC 60870-5 32
TABLE II-5: FIXED FRAME STRUCTURE 23
TABLE II-6 : VARIABLE FRAME STRUCTURE 24
TABLE II-7 : CONTROL FIELD BIT MEANINGS – UNBALANCED TRANSMISSION 27
TABLE II-8 : PERMISSIBLE COMBINATIONS OF UNBALANCED LINK LAYER SERVICES 28
TABLE III-9 : AVERAGE AND MAXIMUM LATENCY OF DIFFERENT TYPES OF FRAMES 57

LIST OF FIGURES

FIGURE I.1 : SMART GRID EVOLUTION 6
FIGURE I-2: AN ACTUAL MODEL OF A SMART METER 8
FIGURE I-3: BLOCK DIAGRAM OF A SMART METER 9
FIGURE I-4: SCADA SYSTEM COMPONENTS 10
FIGURE II-5: EXAMPLE OF THE IEC TC 57 STANDARD 16
FIGURE II-6 : STRUCTURE OF IEC 60870 20
FIGURE II-7 : ENHANCED PERFORMANCE ARCHITECTURE (EPA) MODEL 20
FIGURE II-8: PRIMARY AND SECONDARY STATIONS 26
FIGURE II-9: PRIMARY AND SECONDARY STATIONS IN SCADA AND SMART GRID SYSTEMS 26
FIGURE II-10: BALANCED COMMUNICATIONS 27
FIGURE II-11 : LINK LAYER SERVICE 28
FIGURE II-12 : CONTROL FIELD – UNBALANCED TRANSMISSION 29
FIGURE III-13: RASPBERRY PI 4 MODEL B 34
FIGURE III-14: VIRTUAL SERIAL PORT 35
FIGURE III-15: DOCKLIGHT INTERFACE 36
FIGURE III-16: LOGIGRAM ILLUSTRATING THE ESTABLISHMENT OF COMMUNICATION 39
FIGURE III-17: MTU SEND FRAME AND COLLECTION DATA 40
FIGURE III-18: FRAME AND COLLECTION DATA SENDING FROM MTU 41
FIGURE III-19: THE REQUEST OF DATA_C1. 50
FIGURE III-20: RESULTS FROM TEST 1 ON LINK RESERVATION FROM MTU. 52
FIGURE III-21: RESULTS FROM TEST 1 ON LINK RESERVATION FROM RTU 52
FIGURE III-22: RESULTS FROM TEST 1 ON LINK RESERVATION FROM RTU 53
FIGURE III-23: FRAMES SAVED INTO LOG FILE 54
FIGURE III-24: DATA SAVED INTO LOG FILE 54
FIGURE III-25: MTU AND DATA EXTRACTOR GRAPHIC USER INTERFACE 56
FIGURE III-26: DISPLAY OF DATA EXTRACTOR 56
FIGURE III-27: CONVERSION ERROR EXPERIMENT 57

LIST OF ABBREVIATIONS

ACD: Access Demand Bit
ADC: Analog-to-Digital Converter
ASDU: Application Service Data Unit
DA: Distribution Automation
DER: Integration of Distributed Energy
DFC: Data Flow Control Bit
DNP3: Distributed Network Protocol
DSP: Digital Signal Processor
EPA: Enhanced performance architecture
FCB: Frame Count Bit
FCV: Frame Count Valid
IEC: International Electrotechnical Commission
IEEE: Institute of Electrical and Electronics Engineers
IoT: Internet of Things
IP: Internet Protocol
MTU: Master Terminal Unit
PLC: Power Line Communication
PRM: Primary Message
RES: Reserved
RTU: Remote Terminal Unit
SCADA: Supervisory Control and Data Acquisition
TCP: Transmission Control Protocol

General Introduction

Smart grid technology has revolutionized the field of power distribution and management, incorporating advanced communication and information technologies to improve efficiency, reliability, and sustainability. Effective communication protocols play a crucial role in enabling seamless data exchange and control among different components and subsystems in smart grids.

Among the most widely used protocols in smart grids and SCADA systems (Supervisory Control and Data Acquisition), the IEC 60870-5-101 protocol emerged as the most suitable one. Here are some of the several advantages that make it the preferred protocol choice:

- The IEC 60870-5-101 protocol has gained significant industry integration, making it a widely recognized and accepted standard in the field. This integration ensures compatibility and interoperability with existing smart grid infrastructure and devices.
- The IEC 60870-5-101 protocol offers advanced diagnostics and error reporting capabilities. This feature enables efficient monitoring and identification of potential issues within the smart grid system, providing detailed diagnostic information, thus, allowing efficient maintenance and troubleshooting.
- Real-time communication is a critical requirement in smart grid systems, and the IEC 60870-5-101 protocol excels in this aspect.With its robust and efficient data transmission capabilities, theprotocol enables rapid and accurate real-time communicationbetween various components and subsystems.

As mentioned above, the IEC 60870-5-101 protocol, the most widely used in smart grids, is also used in the Algerian industry infrastructure, mainly in energy data transmission. Nevertheless, the IEC 60870-5-101 protocol keys and maintenance are not available on the Algerian market, sinceit's not an Algerian solution. To tackle this problem, we first develop a Python library specifically for the IEC 60870-5-101 protocol as part of our work. This library enables all the needed rules and commands to implement the IEC 60870-5 standard, and establishes the communication between the Master Terminal Unit (MTU) and the Remote Terminal Unit (RTU).

The main properties of this library are

- To detect errors and handle them,
- To record communication logs,
- To fasten the process, since it does not have a significant latency,
- To enhance the process robustness: it does not break when used over long periods.

To test the functionality of our library, we use the "Virtual Serial Port" software, which establishes virtual ports to connect the MTU and RTU. Additionally, we employ the "Docklight" software as the MTU, but we encounter an issue during this part of work: "Docklight" software doesnot allow to display the data requested by the MTU when receiving the frame.

To solve this problem, we create an interface that plays the role of the MTU, and displays the data received from the RTU. This solution is implemented to enhanceuser readability. It provides a more user-friendly experience.

Our work is organized into four parts:

- The first chapter covers various topics on smart grids and SCADA systems. Firstly, we provide an overview of smart grids and their communication systems. Then, we delve into smart meters, discussing their components and primary functions. Lastly, we explore communication protocols used in smart grids and provide an understanding of SCADA system architecture.
- In the second chapter, we focus on communication protocols. We first introduce different types of communication protocols, with a special emphasis on the IEC 60870-5 standard and its derivations. We also present the IEC 60870-5-10x protocols. We provide a detailed explanation of the IEC 60870-5-101 protocol, exploring the functions of each layer within this specific variant.
- In the last chapter, we introduce the software and hardware tools used in our project. We explain the structure of our code and the library we create. Finally, we present the simulation results with some explanations and illustrations.

Finally, this work is brought to a close with a general conclusion. and We also discuss the potential avenues for enhancing the performance of the protocol in the future.

Chapter I: Introduction to Smart Grids and Intelligent Electrical Counters

I-1) Introduction

In recent times, there have been significant changes in power distribution and management due to fast technological advancements. Traditional electrical meters have evolved into intelligent electrical meters, known as smart meters, which have revolutionized the way power is produced, transferred, and used. These smart meters play a crucial role in accurately monitoring and regulating energy usage **[1]**. They provide a potential solution to rising energy consumption, environmental concerns, and the need for increased efficiency. Smart grid infrastructures incorporate cutting-edge technology, such as sensors, data analytics, and communication systems, enabling effective and sustainable energy management **[2]**. Communication protocols within smart grids facilitate seamless collaboration among various components, including smart meters. **[3]**.

This introductory chapter provides an overview of the key concepts and terminologies related to smart grids and smart electrical meters, emphasizing their significance in addressing energy management, sustainability, and efficiency challenges.

I-2) Smart Grid: History and Architectures

The demand for greater efficiency, less environmental effect, and improved dependability in the power industry have propelled the transformation of electrical networks into smart grids. To understand the current status and potential future of smart grids, it is essential to appreciate the historical backdrop and diverse architectural approaches.

- The concept of smart grids, which utilize cutting-edge technology, emerged in the early 2000s for grid modernization. This led to advancements in remote monitoring, intelligent control, and automation, transforming grid designs from centralized to more decentralized and distributed systems **[4]**.
- Centralized systems limitations prompted the development of distributed designs. These decentralized approaches distribute power generation across localized renewable sources and regional energy storage systems. This strategy promotes energy independence, reduces transmission losses, and enhances grid resilience. Smart grids, characterized by bidirectional power and information flow, utilize advanced sensing and communication technology for real-time data transmission and decision-making. They also facilitate demand-side management, empowering users to participate in load balancing and energy-saving efforts **[5]**.
- Smart grids utilize advanced communication networks and protocols to facilitate smooth communication between various components. These networks ensure interoperability, scalability, and secure data transmission. Protocol IEC 60870-5-101, known as IEC-101 or T101, is a commonly used communication protocol that provides a standardized framework for exchanging data in smart grids. It enables reliable and real-time communication for grid administration, control, and monitoring **[6]**.

The Table I-1 provides a comprehensive comparison between the existing grid and the smart grid

Table I-1: A BRIEF COMPARISON BETWEEN THE EXISTING GRID AND THE SMART GRID [7].

Existing Grid	Smart Grid
Electromechanical	Digital
One-way communication	Two-way communication
Centralized generation	Distributed generation
Few sensors	Sensors throughout
Manual monitoring	Self-monitoring
Manual restoration	Self-healing
Failures and blackouts	Adaptive and islanding
Limited control	Pervasive control
Few customer choices	Many customer choices

I-2-1) Advances In Grid Modernization

a) Integration of Advanced Sensing and Measurement Technologies

- Smart meters and sensors allow utilities (companies or organizations) to collect real-time data on energy usage, voltage levels, and power quality. This valuable information helps utilities monitor and enhance the grid's performance.

b) Grid Automation

- Distribution automation (DA) and supervisory control and data acquisition (SCADA) systems enable remote monitoring and control of the grid. This simplifies the response to failures, reduces downtime, and improves the reliability of the grid.

c) Advanced Communication Networks

- To enable real-time monitoring, load balancing, fault detection, and grid management, smooth data exchange is facilitated among grid components, control centers, and end users through the use of fiber optics, wireless networks, and Internet of Things (IoT) connections.

d) Grid Cybersecurity

- To ensure the security and resilience of critical grid infrastructure, smart grids employ encryption techniques, intrusion detection systems, access control measures, and robust authentication processes. These measures effectively protect against cyber threats targeting smart grids.

e) Integration of Renewable Energy Generation and Energy Storage

- To effectively manage the unpredictability and intermittency of advanced renewable energy technologies like solar and wind generation, complex grid

management solutions are necessary. Energy storage systems play a crucial role in reducing volatility and enhancing grid stability.

Figure I.1, illustrates the evolution of the smart grid, depicting the chronological progression of advancements.

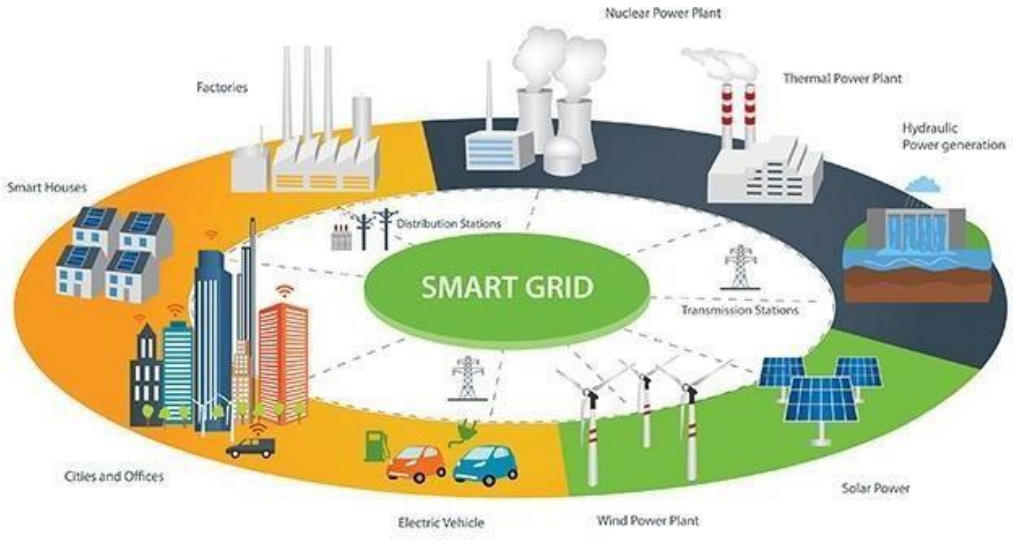


Figure I.1 : Smart grid evolution [8].

Grid modernization has transformed outdated networks into intelligent systems capable of integrating renewable energy sources, enhancing reliability, and optimizing energy distribution. Smart grids serve as the cornerstone for a more efficient, sustainable, and resilient energy future by incorporating advanced technologies such as sensing, automation, communication, cybersecurity, and renewable energy systems.

I-2-2) Importance of Communication in Smart Grids

For smart grids to operate and be efficient, communication is essential. Strong communication networks are necessary for allowing smooth data interchange, real-time monitoring, control, and coordination of grid operations as electrical grids changed into intelligent and linked systems **[9]**.

There are numerous major areas where the significance of communication in smart grids may be emphasized:

a) Grid Monitoring and Control

Effective communication networks enable continuous monitoring of grid metrics and real-time data gathering from sensors and smart meters, allowing utilities to identify abnormalities, optimize grid operations, and ensure stability and reliability.

b) Integration of Distributed Energy Resources (DERs)

Seamless communication networks are essential for integrating renewable energy sources, energy storage technologies, and electric vehicle infrastructure, enabling utilities to optimize grid stability, balance intermittent generation, and facilitate interactions between the grid and electric vehicles.

c) Demand Response and Energy Management

Communication networks enable utilities to deploy demand response systems, allowing customers to adjust their energy usage based on real-time pricing and grid conditions. This two-way communication empowers consumers to actively engage in energy management, to optimize usage, and to support grid resilience.

d) Grid Optimization and Efficiency

Effective communication between grid components, control centers, and intelligent devices enables utilities to optimize grid operations, identify energy loss areas, and improve overall system efficiency. This leads to cost savings, reduced environmental impact, and enhanced performance **[10]**.

e) Grid Resilience and Cybersecurity

Effective communication networks in smart grids should ensure grid resilience and cybersecurity by implementing robust protocols, encryption mechanisms, and intrusion detection systems to protect critical grid infrastructure from cyber threats and unauthorized access.

I-3) Smart Electrical Meter: Concept and Components

In order to fully understand the concept and components of a smart electrical meter, it is essential to explore its significance within the broader framework of smart grid technology and its role in facilitating efficient energy management for both utility companies and customers

I-3-1) Concept of a Smart Electrical Meter

A smart electrical meter is an important device in smart grid technology that keeps track of electricity usage. It is more advanced than regular meters because, it has special features and can communicate with other smart grid devices to collect useful information **[11]**. The main purpose of a smart meter is to provide real-time information about how much electricity is being used, which helps both the utility company and the customers better manage their energy usage. Smart meters are also helpful in programs that encourage people to use electricity wisely, improve how the electricity grid works, and make energy consumption more efficient. They allow people to communicate with the utility company and be more involved in managing their energy.

Figure I-2 represents an actual model of a smart meter, it showcases the physical appearance of a smart meter.



Figure I-2: An actual model of a smart meter [12].

I-3-2) Main Functions of a Smart Electrical Meter

A smart electricity meter becomes a vital component of the smart grid ecosystem through three key functions. In addition to precise and immediate energy monitoring, it also enables load characterization, data-driven decision-making, and interaction with other smart grid elements. Utility companies and customers may optimize energy usage, cut costs, increase grid reliability, and support environmental goals thanks to this multifunctionality. Here are some of the main functions of a smart electrical meter.

- Energy measurement and monitoring,
- Real-time data collection and reporting,
- Load profiling and analysis,
- Integration with other smart grid components.

I-3-3) Components of a Smart Electrical Meter

The components of a smart electrical meter consist of: a metering unit, data processing unit, user interface, power supply, and communication interface.

- For measuring electrical characteristics including voltage, current, and power, the metering unit has sensors or transducers.
- Additionally, it includes a real-time clock for timestamping data on energy usage, an analog-to-digital converter (ADC) for converting analog signals into digital data, a microcontroller or digital signal processor (DSP) for data processing and computation, non-volatile memory for storing energy consumption data, and an analog-to-digital converter (ADC) for converting analog signals into digital data.
- The communication interface supports a number of protocols, including Modbus, DNP3, and IEC 61850, and allows wired or wireless data exchange. It enables interaction with smart

grid infrastructure or utility communication networks, guaranteeing safe data transmission and encryption for privacy.

- A processing unit (microcontroller or DSP) for data analysis and computation, algorithms for load profiling, demand response, and energy management, and memory for storing temporary data and computations make up the data processing unit. Consumers may view information on energy use on a display panel or other interface, and buttons or a touch-screen is available for user engagement and navigation. It provides graphical representations, historical data, and a real-time display of energy use.
- The power supply, which also includes backup power choices like batteries or capacitors for continuous operation even during power outages, enables the operation of the smart electrical counter. If needed, energy collecting capabilities can also be added for self-powered operation.

Figure I-3 presents a block diagram illustrating the internal components and functional architecture of a smart meter, The diagram provides a clear visual representation of the various elements and their interconnections within a smart meter.

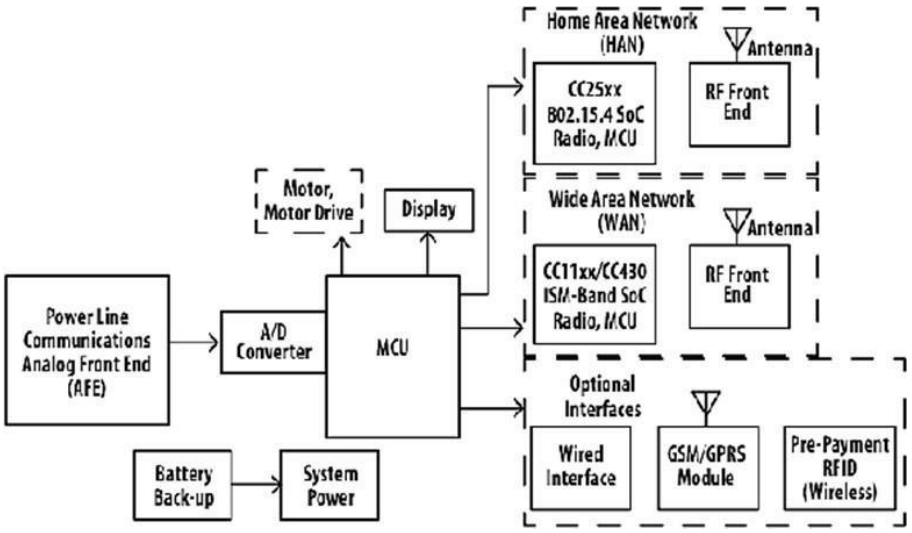


Figure I-3: Block Diagram of a Smart Meter [13].

I-4) Communication Protocols in the Smart Grid

Communication protocols in the smart grid are essential for enabling effective data sharing and collaboration between devices. Standardizing these protocols ensures interoperability, enhancing grid management and optimizing system performance.

I-4-1) Overview of Communication Protocols

In order for the smart grid to work effectively, it needs good communication between its different parts. Communication protocols are like rules that govern how data is shared and transferred between devices in the smart grid. These protocols make sure that the devices can understand each other and work together to monitor, control, and optimize the grid. They also

ensure that the data is delivered reliably and securely. Standardizing communication protocols is important to make sure that different devices and technologies in the smart grid can all work together smoothly. This helps improve grid management and makes the system more effective .

I-4-2) Architecture of a SCADA System

SCADA (Supervisory Control and Data Acquisition) plays a crucial role in the smart grid. It enables centralized control and monitoring of industrial equipment and operations.

SCADA typical hardware includes a Master Terminal Unit (MTU) placed at a control center, communications equipment (e.g., radio, telephone line, cable, or satellite), and one or more geographically distributed field sites consisting of either a Remote Terminal Unit (RTU) or a Pro….LC (PLC) which controls actuators and/or monitors sensors. Figure I.4 represents a typical architecture of a SCADA system.

Figure I-4 presents a visual depiction of the components comprising a SCADA system. The figure illustrates the key elements involved in SCADA, including the supervisory control center (SCADA operator terminal), field devices, communication networks, and data storage systems, highlighting their interconnectedness and role in monitoring and controlling various processes within the smart grid infrastructure.

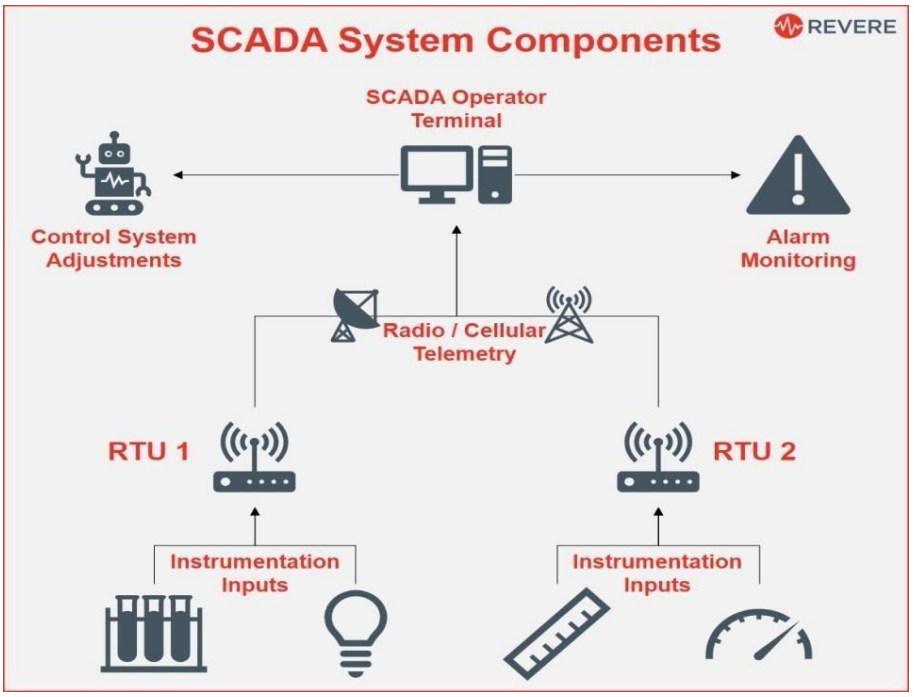


Figure I-4: SCADA System Components [14].

I-4-2-1) ~~A~~ Remote Terminal Unit (RTU)

RTU is a control device that helps the master device (SCADA) monitor and communicate with grid devices placed across different geographical locations. RTUs are used to collect data from equipment in the field and to transmit commands back from the master

device (SCADA). They can be used in a wide variety of electrical and process automation devices where SCADA is required to monitor and control equipment. This can be done through wired connections (telephone lines, cables, Ethernet) or wireless connections. The data is sent to the operator in real time so that they are aware of the situation and can carry out any necessary changes to the grid **[15]**.

- RTUs are used in substations to increase automation, leading to optimal distribution of power and correction of any anomalies that can result in loss of power during transmission and distribution, reduction in damage to equipment, and improvement in quality and reliability **[15]**.
- RTUs can be used to monitor and control water treatment and distribution plants, power transmission and distribution, and oil and gas pipelines **[15]**.
- RTUs are connected to field electrical equipment, such as sensors, which send signals to the RTUs. The basic function of the RTU is to convert the sensor signals into digital signals and then send the encoded digital data to the control center. At the same time, they also communicate control commands to the field equipment **[16]**.

I-4-2-2) ~~A~~ Master Terminal Unit (MTU)

The main purpose of an MTU is to receive different inputs from the remotely connected devices and transmit them over the network. The MTU stores and processes the information from RTU inputs and outputs transmitted via an appropriate physical support (wired or wireless). The field signals such as pump or motor, digital switches, or any other analog signals are terminated at the basic level to RTU **[17].**

As shown in figure I.5, multiple RTUs are connected to an MTU (central control). Each RTU has a point-to-point connectivity with the MTU on individual bases. Additionally, all RTUs are indirectly connected to each other via the central MTU. This results in an increase of the system reliability by reducing the chances of the network failure **[18].**

Figure I-5 showcases the architecture of a SCADA (Supervisory Control and Data Acquisition) system, specifically highlighting the inclusion of MTUs (Master Terminal Units) and RTUs (Remote Terminal Units). The figure provides a visual representation of how these components are integrated within the SCADA framework, demonstrating their respective functions and their role in collecting and transmitting data between the field devices and the supervisory control center for efficient monitoring and control of the smart grid operations.

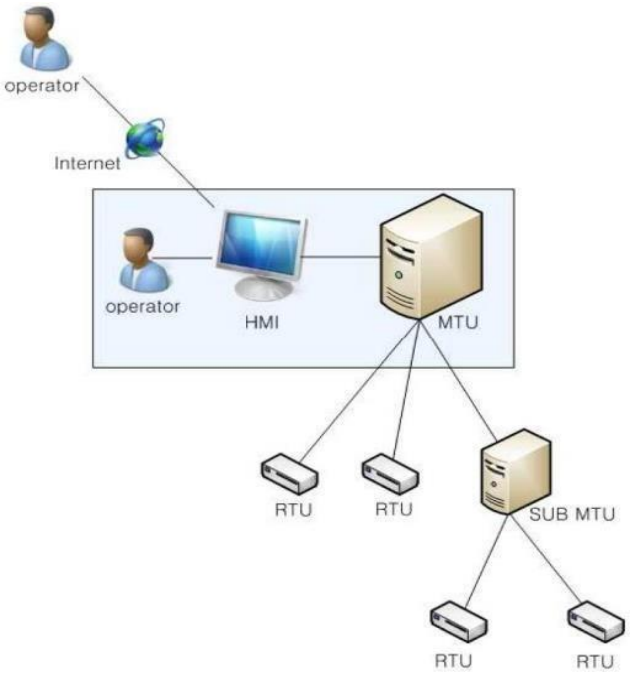


Figure I-5: SCADA system architecture including MTU and RTUs [19].

I-4-3) Communication Protocols for Metering and Monitoring in a SCADA System

The smart grid can gather correct data, perform real-time monitoring, and manage its resources effectively by using reliable communication protocols for metering and monitoring. These communication protocols make sure that data on energy use, grid health, and device health indicators are reliably sent, enabling efficient load management, billing, and maintenance operations.

Modbus and DNP3 (Distributed Network Protocol) are two of the most often utilized communication protocols in a SCADA system.

- Using a master-slave design, Modbus is a widely used protocol that enables communication between devices. It is adaptable for many metering and monitoring applications since it supports both serial and Ethernet connections.
- DNP3 is intended primarily for usage in utility automation systems. It has sophisticated features including event reporting, time synchronization, and data prioritizing, making it suitable for smart grid metering and monitoring.

Chapter II: Communication Protocols in Smart Grids and overview of IEC 60870-5 Standard

II-1) Introduction

In the world of smart grids, communication protocols are very important for exchanging data efficiently. This chapter provides an overview of communication protocols used in smart grid systems, with a focus on the IEC 60870-5 standard. These protocols enable the exchange of data and commands, allowing for optimization of the grid, effective energy management, and real-time monitoring. The use of established protocols ensures that smart grid systems and devices can work together smoothly.

II-2) Communication Protocols in Smart Grids: A Comparative Overview

Communication protocols are vital for smart grid systems as they enable the exchange of information and commands between components. In this section, we will delve into the basics of communication protocols in smart grids, highlighting their importance in facilitating seamless communication, component integration, and efficient energy resource management. Understanding the functioning of these protocols is key to comprehending their crucial role in grid operations.

II-2-1) Introduction to Communication Protocols in Smart Grids

Communication protocols are important for the smooth operation of smart grids. They help different parts of the grid, like smart meters and substations, share information and control commands effectively. These protocols allow the grid to manage events like faults, respond to changes in demand, optimize energy distribution, and monitor energy usage in real-time. They also define how devices communicate with each other and handle challenges like data transfer, compatibility, and security. Choosing the right protocol is crucial as the grid grows and becomes more complex. Understanding these protocols helps in building reliable communication systems.

II-2-2) Types of Communication Protocols

Depending on the type of technology or network architecture used in smart grids, communication protocols can be classified into different categories. One must understand various protocol categories to determine which one is best suited for specific Smart Grid applications **[20]**. These are a handful of common categories:

II-2-2-1) Wired Protocols

In a smart grid, devices use wired protocols to communicate with each other through physical connections like cables and wires. These protocols ensure reliable and secure data transfer. Wired protocols are preferred in smart grid systems. They are particularly useful for applications that require high-bandwidth communication, such as advanced metering, substation automation, and distribution management, as they can handle large amounts of data effectively. Wired protocols used in smart grids include Ethernet and Power Line Communication (PLC). However, selecting the right wired protocol depends on factors such as network limitations, communication distance, interference concerns, and specific smart grid

needs.**[21]**

II-2-2-2) Wireless Protocols

Wireless protocols facilitate the transfer of data between devices in a smart grid infrastructure without the need for physical connections. Instead, they utilize electromagnetic waves to establish communication links, providing flexible and mobile connectivity compared to wired protocols **[22]**. Wireless protocols offer several advantages in smart grid communication, including flexibility and mobility, wide area coverage and scalability.

Wireless protocols commonly used in smart grid communication include: Wi-Fi (IEEE 802.11), Zigbee (IEEE 802.15.4) and cellular networks.

Wireless protocols are vital for wireless connectivity in smart grids, facilitating fast data transfer and smooth integration of systems and devices. Factors such as range, data rates, power consumption, infrastructure accessibility, and compatibility play a role in choosing the appropriate wireless protocol **[21].**

II-2-2-3) Fieldbus Protocols

Fieldbus protocols are communication protocols specifically designed for industrial automation and control systems, including those used in smart grid applications. These protocols provide a standardized method for communication between devices within a field or industrial network.

Fieldbus protocols offer several advantages in smart grid communication, including: deterministic communication, robustness and reliability, integrated communication and control.

Fieldbus protocols commonly used in smart grid applications include: Profibus, Modbus and Foundation Fieldbus.

Fieldbus protocols are reliable and standardized communication infrastructures used in smart grid applications, providing real-time monitoring, control, and management capabilities. The selection of a specific fieldbus protocol depends on factors such as scalability, required data transmission speeds, compatibility with existing systems, and the unique requirements of the smart grid application **[23]**.

II-2-2-4) Utility-specific Protocols

Utility-specific protocols are communication protocols that are specifically designed and tailored for the unique requirements and characteristics of utility networks and smart grid applications. These protocols address the specific needs of utility companies, including advanced metering, grid monitoring, demand response, and distribution automation.

Utility-specific protocols offer several advantages in smart grid communication, including: enhanced security, scalability and interoperability, optimal performance and efficiency **[24]**.

Examples of utility-specific protocols commonly used in smart grid applications include: DNP3 (Distributed Network Protocol 3), IEC 61850, M-bus.

Utility-specific protocols are crucial for ensuring secure, scalable, and efficient communication within smart grid systems, catering specifically to the needs of utility networks.

Among IEC 60870-5 standards, the IEC-101 protocol is designed for the utility industry and plays a key role in smart grid connectivity. IEC 60870-5-101 offers a standardized framework for communication between utility devices, including sensors, meters, substations, and control centers. It enables real-time data exchange, event reporting, and remote control capabilities, empowering utility companies to effectively manage their smart grid assets and achieve seamless integration. When selecting a utility-specific protocol like IEC 60870-5-101, factors such as compatibility with existing infrastructure, adherence to industry standards, and alignment with the unique requirements of the utility network come into consideration.

II-3) IEC 60870-5 standard

The standard IEC 60870-5 was developed by the International Electrotechnical Commission (IEC), Technical Committee 57, Working Group 03, as shown in figure II.5 the Example of the IEC TC 57 standard. and published progressively starting from 1988 **[25]**. The hierarchy of the standard has been gradually developed and released. It includes five sections and several associated standards. The standard is structured hierarchically, and there are five companion standards that provide specific details for different application fields, as shown in figure II.5.

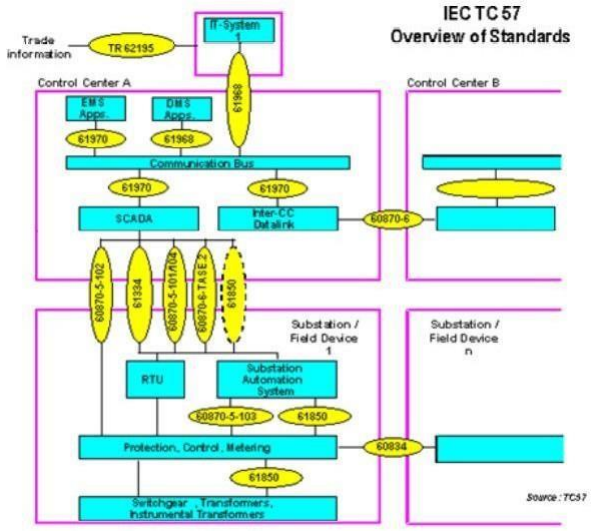


Figure II-5: Example of the IEC TC 57 standard [26].

II-3-1) Derivation of IEC 60870-5

The IEC 60870-5 standard series consists of several parts, each addressing different aspects of telecontrol communication. These parts define the protocols, data structures, and procedures for effective and reliable data exchange between control centers and remote devices. The key parts of the standard series include:

II-3-1-1) IEC 60870-5-1

IEC 60870-5-1 is a crucial component of the IEC 60870-5 standard series, focusing on the basic communication protocol for telecontrol systems. It defines the transmission media,

message formats, and error detection mechanisms necessary for reliable data transfer at the physical and data link layers. This standard promotes compatibility and interoperability, facilitating smooth communication between systems and devices that adhere to it **[27]**.

IEC 60870-5-1 prioritizes dependability, security, and standardization to ensure successful telecontrol communication across various sectors. The next part of the course will delve into the details of IEC 60870-5-1, providing comprehensive explanations of its usage and implementation **[27]**.

II-3-1-2) IEC 60870-5-2

IEC 60870-5-2 is an important part of the IEC 60870-5 standard series that focuses on the application layer of telecontrol communication. It defines information objects and data structures for exchanging control and monitoring data between control centers and field devices. This ensures consistency and interoperability of data representation **[28]** .

The main goal of IEC 60870-5-2 is to specify uniform application models and protocols for telecontrol systems. It provides a framework for data interchange, promoting interoperability and smooth communication in telecontrol systems through standardized protocols **[28]**.

IEC 60870-5-2 offers telecontrol application models for various scenarios, covering topics like data synchronization, event handling, control operations, and data gathering. It provides standardized protocols for communication between command centers and remote devices, including functions like error detection, recovery, and message acknowledgment **[28]**.

The standard also defines communication profiles for different media, considering factors like network latency, capacity limitations, and data integrity **[28]**.

II-3-1-3) IEC 60870-5-3

IEC 60870-5-3 is a part of the IEC 60870-5 standard series that focuses on transmission protocols in telecontrol systems. It defines procedures and formats for data transmission, ensuring efficient and reliable communication over different media **[29]**.

The standard creates uniform profiles for telecontrol devices, outlining their functionality, communication standards, and capabilities. It promotes compatibility and integration of telecontrol devices into larger control systems **[29]**.

The standard specifies different device profiles for various device types, facilitating interoperability among products from different vendors. It also covers communication specifications, device configuration, and setup protocols. In summary, IEC 60870-5-3 defines transmission protocols and establishes standard profiles for telecontrol devices to ensure reliable and standardized communication in telecontrol systems **[29]**.

II-3-1-4) IEC 60870-5-4

IEC 60870-5-4 is a part of the IEC 60870-5 standard series that specifies the application layer protocol for telecontrol communication using TCP/IP as the transport layer. It focuses on the fundamental communication architecture for supervisory control and data acquisition

(SCADA) systems **[30]**.

The goal of IEC 60870-5-4 is to provide a uniform communication framework for SCADA systems, enabling the transmission of control instructions, telemetry data, and status information between control centers and field devices.

The standard defines the format and semantics of information objects, fields, and groups to ensure consistent data interpretation and interchange. It also outlines communication protocols for reliable and secure data sharing, including error detection, error recovery, and synchronization techniques **[30]**.

Additionally, IEC 60870-5-4 covers topics related to SCADA system administration and organization, such as event management, time synchronization, addressing techniques, and data types. By providing a standardized communication framework, IEC 60870-5-4 promotes interoperability and facilitates efficient coordination and management of SCADA systems **[30]**.

II-3-1-5) IEC 60870-5-5

IEC 60870-5-5 is a part of the IEC 60870-5 standard series that provides guidelines for conformance testing of telecontrol systems. It focuses on transmission protocols specifically designed for use in supervisory control and data acquisition (SCADA) and smart grid systems **[31]**.

The main goal of IEC 60870-5-5 is to establish standardized transmission protocols that facilitate secure and efficient communication in SCADA and smart grid environments. The standard includes protocols such as TCP/IP, ISO/OSI-based protocols, and serial-based protocols, defining their structures, message formats, data encoding techniques, and error detection systems. It also outlines guidelines for data exchanges, session management, and communication initiation and termination **[31]**. By utilizing the transmission protocols specified in IEC 60870-5-5, SCADA and smart grid systems can achieve seamless integration, interoperability, and reliable data exchange for remote control, real-time monitoring, and status communication with field devices. The standard ensures effective and secure data transfer in various network contexts **[31]**.

Table II-2 displays a selection of standard provisions from the defined telecontrol companion standard. The figure defined the IEC 60870-5-101 layers and their relation with the five Derivation of IEC 60870-5.

Selected application fuctions of IEC 60870-5-5	User process
Selected application information elements of IEC 60870-5-4	Application
Selected application service data units of IEC 60870-5-3	
Selected link transmission procedures of IEC 60870-5-2	Link
Selected transmission frame formate of IEC 60870-5-1	
Selected ITU-T recommendations	physical

Table II-2 : Selected standard provisions of the defined telecontrol companionstandard [25]

II-3-2) IEC 60870-5 10x Protocols

In this section, we will explore the family of IEC 60870-5 protocols, which play a significant role in smart grid communication

- The IEC 60870-5-101, also known as T101 profile (T for telegram), is a significant protocol in the IEC 60870-5 standard, providing a comprehensive SCADA protocol for telecontrol applications across long distances. It includes essential functions, data structures, and communication formats for efficient and reliable remote control. IEC 60870-5-101 is frequently referenced for SCADA systems as it defines the complete transmission protocol for telecontrol tasks **[25]**.
- The IEC 60870-5-102 and IEC 60870-5-103 companion standards support electrical protection systems such as transformer differential protection and distance protection.
- The IEC 60870-5-104 companion standard focuses on using TCP and IP protocols for transmission over networks while maintaining compatibility with IEC 60870-5-101's application-level operations **[25].**

The IEC 60870-5 standard series is widely recognized and used in the utility industry. It ensures interoperability, reliability, and security in telecontrol communication, allowing seamless integration of various devices and systems. The standard provides a common language and framework for different vendors and manufacturers, facilitating the development and deployment of telecontrol solutions **[32]**.

Figure II-7 depicts the main standard, IEC 60870, along with its derived standards and the corresponding protocols.

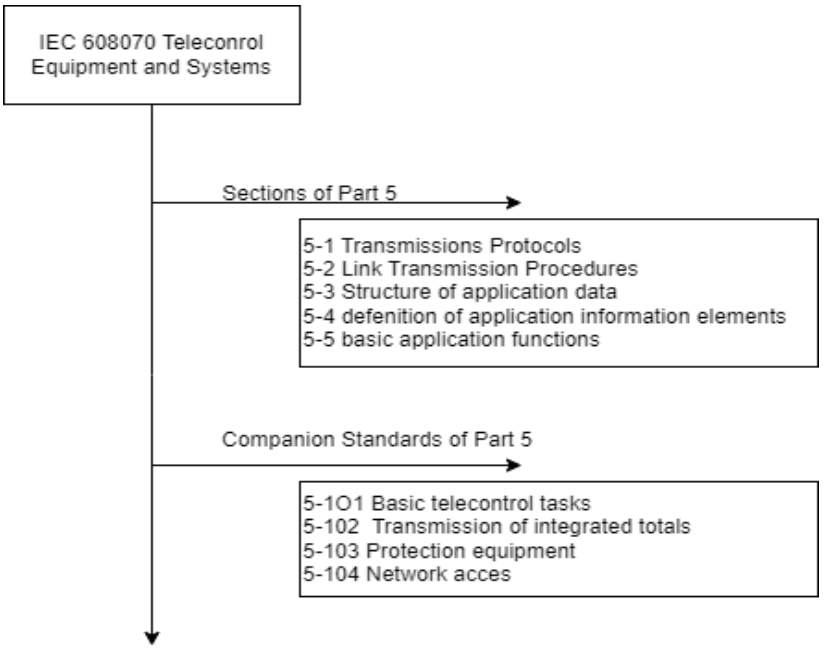


Figure II-6 : Structure of IEC 60870 [25].

II-3-3) Enhanced Performance Architecture (EPA)

The enhanced performance architecture model developed by IEC Technical Committee 57 is a 3 layer sub-set of the OSI 7 layer model. The layers used in this model are the two hardware layers and the top software layer, the application layer

Figure II-8 presents the Enhanced Performance Architecture (EPA) model. The figure illustrates the architectural framework of the EPA,

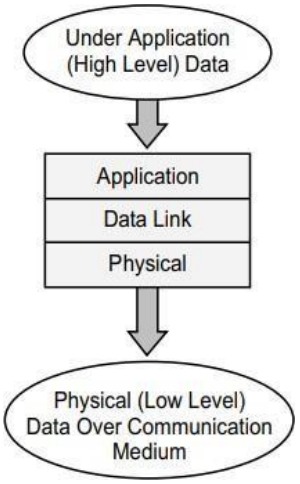


Figure II-7 : Enhanced performance architecture (EPA) model [25].

The Enhanced Performance Architecture (EPA) aims to optimize communication systems in SCADA and smart grid environments. IEC 60870-5 standards, including derivatives like IEC 60870-5-1 to -5, are crucial components of EPA. They define communication

protocols, message structures, and data encoding methods. EPA utilizes these standards to achieve enhanced performance, reliability, and interoperability. Adhering to IEC 60870-5 ensures compatibility between control centers and field devices, enabling optimized communication in SCADA and smart grid systems.

II-4) Architecture of IEC 60870-5-101 Protocol

The architecture of the IEC 60870-5-101 protocol is designed to ensure reliable and efficient communication in telecontrol applications. Its structure allows functional components to work seamlessly together, ensuring the effectiveness of the system. The protocol clear structure manages complex telecontrol applications and provides secure data delivery. It enhances efficiency in telecontrol operations by providing reliable communication between control centers and remote equipment. The standardized architecture promotes interoperability and accommodates various telecontrol applications, enabling an efficient and safe transmission of data across system components.

Table II-2 provides a selection of standards for IEC 60870-5-101. It illustrates the relationship between the protocol layers, the derived standards, and the type of transmission used in each layer.

Table II-3: Standards selections for IEC 60870-5-101 [25], [33].

Layer	Type of transmission	Source	Selections	utility
User Process	Data	IEC 60870-5-5	Application function	User-specific application logic
Application	ASDUs	IEC 60870-5-4	Application information elements	Data transmission and protocol management
		IEC 60870-5-3	ASDUs	
Link	Frame	IEC 60870-5-2	Transmission procedures	Data synchronization and framing
		IEC 60870-5-1	Frame formats	
Physical	Bytes	ITU-T	Interface specification	Electrical, optical, or wireless signals

III-5) Physical Layer of IEC 60870-5-101

The Physical layer is the lowest layer in the protocol stack, and deals with the actual transmission of signals over the chosen medium, such as electrical, optical, or wireless environment.

The Physical layer of a network is responsible for transmitting and receiving digital data as physical signals. Its primary goal is to convert digital data into signals suitable for transmission, and to decode received signals back into digital data **[25]**.

This layer considers factors like signal attenuation, noise, interference, and signal integrity to ensure reliable transmission. Modulation, demodulation, equalization, and error correction

techniques are employed to enhance signal quality.

The Physical layer operates independently of the higher layers in the protocol stack, it focuses solely on the reliable transmission of signals regardless of the specific data or functionality of the upper layers.

III-5-1) Network Configurations

Considering the ways in which devices are connected and organized within a telecontrol system is important. The network topology plays a crucial role in determining how communication is established and maintained between these devices. The IEC 60870-5-101 supports the following network configurations or topology **[25]**:

- Point-to-point,
- Multiple point-to-point,
- Multi-point-star,
- Multi-point-party line,
- Multi-point-ring.

IEC 60870-5-101 protocol allows for redundant network setups to improve reliability and to handle faults. Redundancy can be implemented at different levels, such as having backup master devices, communication channels, or slave devices. These redundant setups ensure uninterrupted communication even if there are network problems. The choice of a given network setup depends on factors like system size, device distance, reliability needs, and operational requirements. It is important to carefully plan the network layout to ensure efficient and reliable communication between devices. Additionally, intermediate RTUs can serve as local master stations for linked RTUs, creating a hierarchical network structure **[25]**.

II-6) Data Link Layer

The link layer of the IEC 60870-5-101 protocol is crucial for ensuring reliable communication between devices in a telecontrol system. It handles tasks like framing the data, controlling the flow of information, and establishing a logical and physical connection between the sender and receiver.

The data link layer is responsible for sending data across the communication channel and ensuring it is received correctly without any errors. It achieves this by using a data structure called a frame, along with protocols that govern its transmission and reception. The frame contains important control information, such as the destination address and error detection codes, and can also carry a payload of data if needed. The frame size is designed to be manageable, so that transmission failures or timing issues between devices don't lead to significant data loss or synchronization problems

The link layer in the IEC 60870-5-101 protocol has a direct relationship with the IEC 870-5-1 and IEC 870-5-2 standards. These standards define the transmission frame formats and link transmission procedures, which serve as the basis for the link layer functionality in IEC 60870-5-101 **[34].**

The link layer in IEC 60870-5-101 incorporates several essential concepts for efficient

communication:

- Frame Structure,
- Frame Synchronization,
- Error Detection and Correction.

The link layer in IEC 60870-5-101 is closely related to the IEC 870-5-1 standard's transmission frame formats and the IEC 870-5-2 standard's link transmission procedures. These standards provide the foundational concepts, formats, and protocols that shape the functionality and behavior of the link layer in IEC 60870-5-101, ensuring compatibility and interoperability with other systems that adhere to the IEC 870 series, The table II-3 showcases the relationship between the link layer and the two divided standards of IEC 60870-5, namely IEC 60870-5-1 and IEC 60870-5-2[**33]**.

Table II-4: the relation between link layer IEC 60870-5 [25],[28],[33].

Link Layer	IEC 60870-5-2	Transmission procedures
	IEC 60870-5-1	Frame formats

II-6-1) IEC 60870-5-101 Telegram Structure

The frame format typically includes various fields and information elements. Some common elements found in the frame format of IEC 60870-5-101 include:

- **Start Character:** Marks the beginning of the frame and helps with frame synchronization.
- **Control Field:** Contains control information, such as frame type, sequence number, and error detection.
- **Address Field:** Specifies the source and destination addresses of the communicating devices.
- **Data Field:** Carries the actual data payload, which may include measurement values, control commands, or other relevant information.
- **Checksum Field:** Provides error detection capabilities by calculating a checksum value based on the frame content.
- **End Character:** Indicates the end of the frame.

In the IEC 60870-5-101 protocol, there are two important frame format concepts: fixed frame and variable frame. These concepts define different ways of organizing and managing the payload data. To effectively utilize telecontrol applications, it is crucial to comprehend the characteristics and implications of both fixed frame and variable frame architectures **[25]**.

II-6-1-1) Frame with fixed length

In a fixed frame structure, the size of the frame remains the same for every transmission.

It has a set length, even if there is less data to be sent. This makes the communication process

simpler and helps with synchronization and error detection. However, when transmitting small amounts of data, it can be inefficient as the frame size is constant [25].

This type of frame is commonly used for link layer services and sometimes used for confirmation purposes.

Table II-4 presents the fixed frame structure utilized in a specific context or standard. The table outlines the components and characteristics of the fixed frame structure

Table II-5: fixed frame structure [35].

Byte\bit	7	6	5	4	3	2	1	0			
0	Start byte 1 (**0x10**)										LPCI
1	RES	PRM	FCB / ACD	FCV / DFC	Function code				Control field		
2	Link address										
	(0, 1 or 2 octets)										
n-1	Checksum										
N	Stop byte (0x16)										

II-6-1-2) Frame With Variable Length

In a variable frame structure, the size of the frame can vary depending on the amount of data being transmitted. The frame adjusts its length dynamically to accommodate the size of the payload. This flexibility allows for more efficient use of bandwidth when transmitting smaller amounts of data, as it eliminates the need for transmitting unused space in the frame. However, variable frame structures may introduce additional complexity in terms of synchronization and error detection, as the frame size changes dynamically.

This frame type as shown in Table II-5 is used to transmit user data between controlling and controlled station.

The following points are noted for the data link frames:

- Only the variable length frame can include user data,
- The variable length frame can hold up to 253 octets of link user data,
- The length L is repeated twice, and the two values of L must be equal for the frame to be accepted as valid,
- The maximum frame length is 261 octets. However, a lower maximum frame length may be specified by a manufacturer or by the system user as a system parameter,

- The fixed length frame is 5 or 6 octets long,
- Address field A may be 1 or 2 octets, determined by a fixed system parameter,
- A broadcast address is defined as 0xFF or 0xFFFF for 1 and 2 octet addresses respectively,
- The checksum is the modulo 256 sum of the frame user data (not the link user data). This is the data between the last start character and the checksum, L octets for the variable frame
- Rules state that no more than one bit-time idle interval is allowed between, characters within the frame, and that an idle interval of 33 bit-times must be allowed after detection of a frame error by the receiver

Table II-6 : variable frame structure [35].

Byte\bit	7	6	5	4	3	2	1	0					
0	Start byte 1 (0x10)								Header		LPCI		LPDU
1	Block length												
2	Block length (copy)												
3	Start byte 2 (0x68)												
4	RES	PRM	FCB / ACD	FCV / DFC	Function code				Control field				
5	Link address fields												
	(0, 1 or 2 octets)												
	Type identification								DATA UNIT IDENTIFIER		ASDU		
	SQ	Number of objects											
	T	P/N	Cause of transmission (COT)										
	Originator address (ORG, 0 or 1 octets)												
	ASDU address fields												
	(1 or 2 octets)												
	Information object address fields (IOA)								Info-object				
	(1,2 or 3 octets)												
	Object information												
n-1	Checksum								Tail		LPCI		
N	Stop byte (0x16)												

II-6-2) Primary And Secondary

In IEC 60870-5-101, the primary device is the active initiator of communication, while secondary devices are passive responders. The primary device takes charge by requesting actions or data from secondary devices, establishing communication channels, and coordinating operations. Secondary devices respond to the primary device's commands, providing the requested information and performing necessary actions.

Figure II-9 depicts the concept of primary and secondary stations within a specific context or system. The figure illustrates interconnection between the primary station and multiple secondary stations.

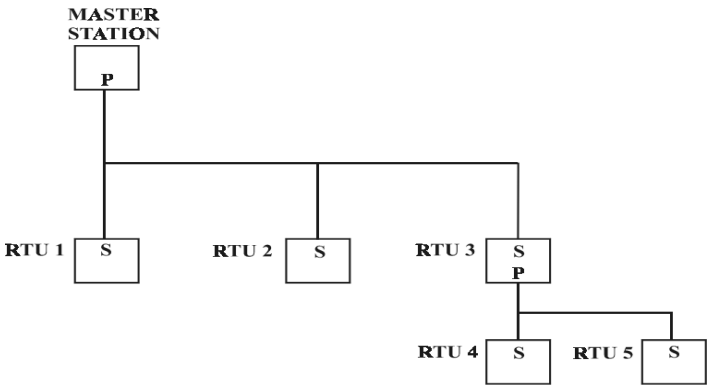


Figure II-8: Primary and secondary stations [25].

The primary and secondary roles are essential in maintaining structured communication in telecontrol systems. The primary device acts as the central authority, issuing commands and coordinating operations, while secondary devices respond to these commands and provide necessary data.

In SCADA and smart grid systems, the MTU serves as the primary station, as shown in figure II-10, it controls the system and gathers data from RTUs located in the field. The MTU uses its processing capabilities to issue commands and monitor the system, while the RTUs interface with sensors, collect data, and execute commands from the MTU, creating an efficient hierarchical system for communication and control **[19].**

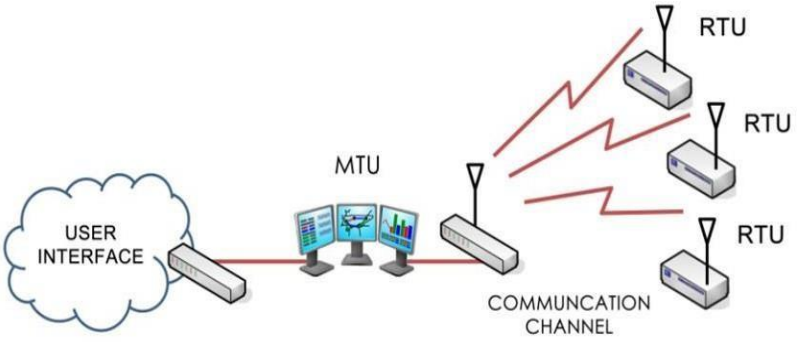


Figure II-9: Primary and secondary stations in SCADA and smart grid systems

[36].

II-6-3) Unbalanced And Balanced Transmission

Unbalanced and balanced transmission are two different approaches in IEC 60870-5-101 for communication. Unbalanced transmission involves a primary station controlling secondary stations, while balanced transmission allows multiple secondary stations to connect to a single primary station, simplifying communication and reducing infrastructure complexity [19].

A master station connected to a sub-master station over a point-to-point link is an example of a balanced communications link, as shown in figure II-11. At the ports for this link, each station can function as both a main and a secondary. As they are logically separated inside the stations, these may really be thought of as two different processes within each station. For such link, Station A has a primary process and a secondary process running concurrently, and Station B has the same [22].



Figure II-10: Balanced communications [25].

Both unbalanced and balanced transmission offer advantages and are selected based on the specific requirements of the SCADA or smart grid system. The choice depends on factors such as the system architecture, number of RTUs, communication distance, and desired level of scalability. In our study, we will focus on unbalanced transmission due to its suitability for collision prevention and enhanced protection measures. This choice allows us to investigate the unique characteristics and challenges associated with unbalanced transmission, enabling us to design efficient and robust communication solutions for our specific application.

II-6-4) Service Procedures

The link layer of the IEC 60870-5-101 protocol provides specific services to the application layer, enabling the transmission of data. These services can be categorized into three main types

- **Send/no reply :** The send/no reply service is used to send messages or commands that do not require a reply from the addressed station. It is commonly used for broadcast messages or messages where receipt confirmation is not essential.

- **Send/confirm:** The send/confirm service, on the other hand, is employed for transmitting commands or data that must be reliably transmitted. With this service, a confirmation response is required to ensure successful delivery.

- **Request/respond:** The request/respond service allows for the retrieval of data from the controlled station. In this case, the controlled station responds with the required data, rather than a confirmation.

Figure II-12 illustrates the link layer of two stations on a communication channel. The time-sequence diagram showcases the interactions between the service user (application layer) and the link layer, represented by the commands, data, and responses exchanged. These interactions are referred to as "service primitives." The transmission procedure name is indicated in the message transmission.

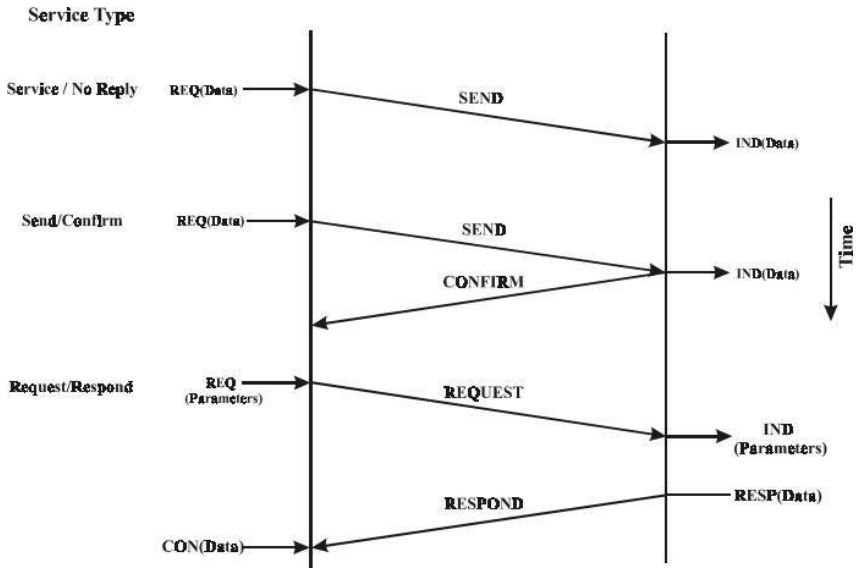


Figure II-11 : Link layer service [25].

II-6-5) Control Field

The control field of the data frame plays a crucial role in the transmission procedures of the IEC 60870-5-101 protocol. It is a significant component that governs the communication process between the primary and secondary stations. Interestingly, the control field in IEC 60870-5-101 is highly similar to the one used in the DNP3 protocol, as both protocols have originated from the same source document, IEC 60870-5-2 1992.

For our specific focus on unbalanced transmission, figure II-13 serves as a valuable visual representation of the control field's composition, specifically tailored for unbalanced transmission procedures **[33]**.

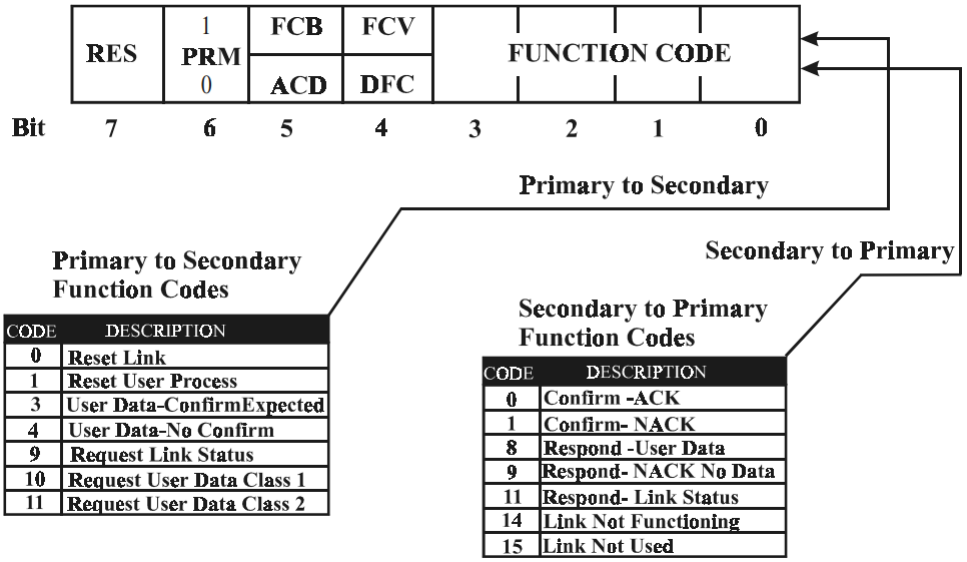


Figure II-12 : Control field – unbalanced transmission [25].

By studying the control field and its associated function codes, one can gain a comprehensive understanding of how primary and secondary stations communicate in unbalanced transmission scenarios **[33]**.

Table II-7 : Control field bit meanings – unbalanced transmission [34].

Code	Meaning	Descrioption
RES	Reserved	=0
PRM	Primary Message	0 = message from a secondary (controlled) station. 1 = message from a primary (controlling) station.
FCB	frame count bit	Alternates between 0 and 1 for sequential frames
FCV	Frame Count Valid	0 = alternating function and FCB bit is invalid 1 = alternating function and FCB bit is valid
DFC	Data Flow Control Bit	Set to 1 by secondary station if further send of user data will cause buffer overflow
ACD	Access Demand Bit	Set to 1 if there is Class 1 data available

The control field's function codes serve as essential indicators that enable the primary and secondary stations to exchange information effectively and perform their respective roles within the communication process. They provide a standardized framework for defining the purpose and intent of each message, facilitating seamless and structured communication in SCADA and smart grid systems.

Table II-7 showcases the function codes assigned to both primary and secondary messages, providing clear and concise descriptions that outline their distinct meanings and purposes.

Table II-8 : Permissible combinations of unbalanced link layer services [34]

Function codes and services in the primary direction	**Permitted function codes and services in the secondary direction**
<0> Reset of remote link	<0> CONFIRM: ACK or <1> CONFIRM: NACK
<1> Reset of user process	<0> CONFIRM: ACK or <1> CONFIRM: NACK
<3> SEND/CONF user data	<0> CONFIRM: ACK or <1> CONFIRM: NACK
<4> SEND/NO REPLY user data	No reply
<8> REQUEST for access demand	<11> RESPOND: status of link
<9> REQUEST/RESP request status of link	<11> RESPOND: status of link
<10> REQUEST/RESP request user data class 1	<8> RESPOND: user data or <9>RESPOND: requested data not available
<11> REQUEST/RESP request user data class 2	<8> RESPOND: user data or <9>RESPOND: requested data not available

PRM (Primary/Secondary): This bit indicates whether the message is sent by a primary station (PRM = 1) or a secondary station (PRM = 0). In unbalanced transmission, the primary station initiates communication, while the secondary station responds.

FCV (Function Code Validity): The FCV bit ensures the integrity of the function code within the control field. When FCV = 1, it indicates that the function code is valid and corresponds to a defined operation within the protocol.

Function Code (FC): The function code represents the specific operation or command to be performed. It defines the purpose of the message, such as requesting data, issuing commands, or indicating specific control actions**.**

Sequence of Control (SC): The SC bit is used for sequence numbering in the transmission of messages. It ensures the correct order of messages and helps detect any missing or out-of-sequence messages**.**

More Follows (FR): The FR bit indicates whether additional frames are expected to follow within a message sequence. FR = 1 indicates more frames, while FR = 0 indicates the last frame.

Test (T): The T bit is used for testing purposes. When T = 1, it indicates that the message is a test message, typically used for diagnostic or verification purposes**.**

Negative Acknowledge (NAK): The NAK bit is used to indicate a negative acknowledgment, signaling that an error or failure occurred in the previous message transmission.

User Data (UD): The UD bits contain the actual user data or information being transmitted. This can include measurements, status information, commands, or any other relevant data for the application.

II-7) Application Layer

The content on the application layer will cover the following topics:

1. **Overall Message Structure:** This section will explain the structure of the messages exchanged within the application layer, highlighting the different components and fields involved in transmitting data.
2. **ASDU (Application Service Data Unit) Structure:** The ASDU structure defines the format and content of the data units exchanged between stations. It will delve into the fields and information elements within an ASDU that carry the actual data and control information.
3. **Message Addressing and Routing:** The mechanisms for addressing and routing messages within the application layer will be discussed, including how messages are directed to their intended recipients and how routing decisions are made for efficient communication.
4. **Information Elements:** This section will focus on the information elements contained within an ASDU, explaining their types and roles in conveying specific data and control information. Examples of commonly used information elements will be provided.
5. **Set of ASDUs:** An overview of the various ASDU types defined by the IEC 60870-5 standard will be presented, highlighting their purposes and functionality in supporting different smart grid applications and services.

Chapter III: Implementation and tests

III-1) Introduction

In the implementation part of the present chapter, we focus on presenting the software and code structure that defines the protocol used for communication between the Master Terminal Unit (MTU) and the Remote Terminal Unit (RTU). The implementation details include the configuration rules and guidelines necessary for establishing effective communication between these two devices. The development of the iec-101 library represents the main core of our work. We choose Python language for developing and implementing the IEC 60870-5-101 protocol.

III-2) Hardware and Software Setup

To successfully implement the protocol, we need to establish a well-defined hardware and software setup. This section provides an overview of the key components and software tools used in our implementation.

III-2-1) Hardware Setup

The hardware setup forms the physical foundation of our communication system. It includes the Master Terminal Unit (MTU), the Remote Terminal Unit (RTU), and necessary peripheral devics.

The MTU serves as the central control unit responsible for managing and initiating communication with the RTU. It typically consists of a microcontroller or a single-board computer like the Raspberry Pi or Arduino. We choose the Raspberry Pi for its versatility, computational power, and community support.

The RTU is the device that interacts with the physical world or the target system being monitored or controlled. It may include sensors, actuators, and other relevant components. We select the Arduino board for its user-friendly interface, affordability, and extensive library support.

III-2-1-1) Raspberry Pi

We choose Raspberry Pi 4 Model B shown in figure III-14 because of low cost, availability and the main reason; its specifications are close enough to the project requirements. The specifications are enumerated bellow **[37]**:

- CPU: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- RAM: 1GB LPDDR2 SDRAM
- Network: 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- GPIO: Extended 40-pin GPIO header • Full-size HDMI and 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading the operating system and storing data
- 5V/2.5A DC power input



Figure III-13: Raspberry Pi 4 model B.

III-2-1-2) Arduino

We choose Arduino Uno due to its availability, extensive community support, and compatibility with the project requirements. The specifications of Arduino Uno are as follows:

- Microcontroller: ATmega328P
- Flash Memory: 32KB (of which 0.5KB used by bootloader)
- SRAM: 2KB
- EEPROM: 1KB
- Clock Speed: 16MHz
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- Operating Voltage: 5V

III-2-2) Software Setup

The software setup involves the programming languages, libraries, and development tools used in our implementation.

III-2-2-1) Python

Python is a versatile and widely adopted programming language. Its rich library ecosystem, ease of use, and strong community support make it an ideal choice for developing the protocol. Python's networking libraries enable seamless communication between the MTU and RTU.

Our program consists of multiple modules, each serving a specific functionality. We use two main libraries to support our implementation: the serial library and the time library. Additionally, we develop a T101-library named < *iec_101_functions*> to address specific requirements and enhance the functionality of our program.

- The serial library in Python is a module that simplifies serial communication between a computer and external devices. It enables programmers to interact with serial ports and establish communication using serial communication protocols. The library provides a range of functions and classes to facilitate this process. A notable feature of the serial library is the ability to create virtual ports. These virtual ports act as software-based emulations of physical serial ports. Through these virtual ports, the MTU and RTU can establish a communication channel, exchanging data and commands as if they were connected via physical cables. The virtual ports act as intermediaries for transmitting data between the two devices. This feature offers flexibility and convenience, especially in situations where physical connections are not practical or feasible.
- The time library in Python provides functions and methods for working with time-related operations. It allows programmers to retrieve the current time, introduce time delays, format and manipulate time values, and measure code execution time. With its versatile capabilities, the time library simplifies time management tasks in Python programs, enhancing efficiency and accuracy.
- The *iec_101_functions* library we develop, is a specialized library that sets the specific rules and commands of the T101 protocol.

III-2-2-2) Virtual Serial Port

A virtual serial port, also known as an emulated serial port or a software-based serial port, is a virtual representation of a physical serial port on a computer. The virtual serial port behaves similarly to a physical serial port, allowing communication between applications and devices as if they were connected via physical cables.

Figure III-16 illustrates the software interface of the Virtual Serial Port (VSP). The figure provides a control panel of the VSP software, showcasing the various options, settings, and functionalities available to users. It may include elements such as port configuration, data flow control, and communication parameters.

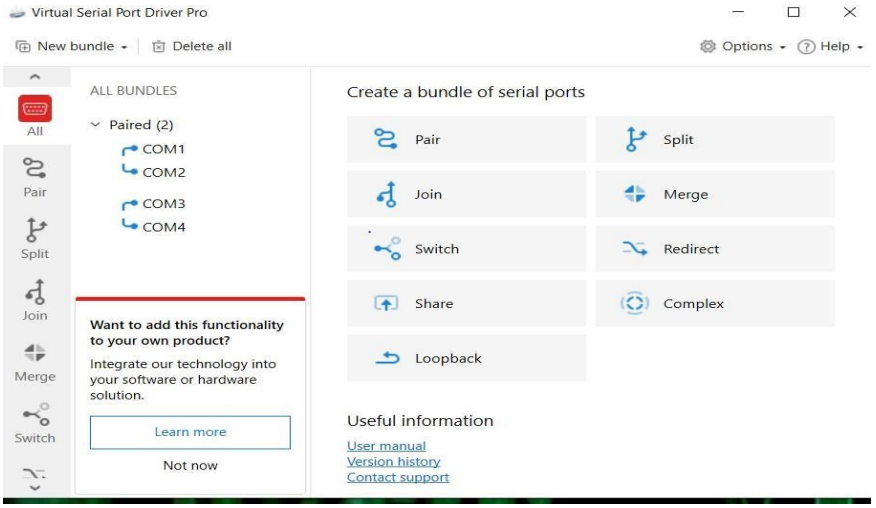


Figure III-14: Virtual Serial Port..

III-2-2-3) Docklight

Docklight is a software tool that serves as a Master Terminal Unit (MTU) in serial communication systems. It enables sending and receiving of commands, data, and frames between the MTU (Docklight) and the Remote Terminal Unit (RTU) or other devices. With Docklight, users can create and configure custom frames, define the communication parameters, and simulate the behavior of an MTU in a serial communication setup. It provides a user-friendly interface for designing and executing test scenarios, allowing users to analyze and monitor the communication flow, validate the performance of the RTU, and troubleshoot any issues that may arise.

Figure III-17 provides a visual representation of the user interface of Docklight, showcasing its various features, tools, and functionalities. It may include elements such as the main window, toolbars, menus, and specific panels or tabs for configuring communication settings, monitoring data, and analyzing serial communication.

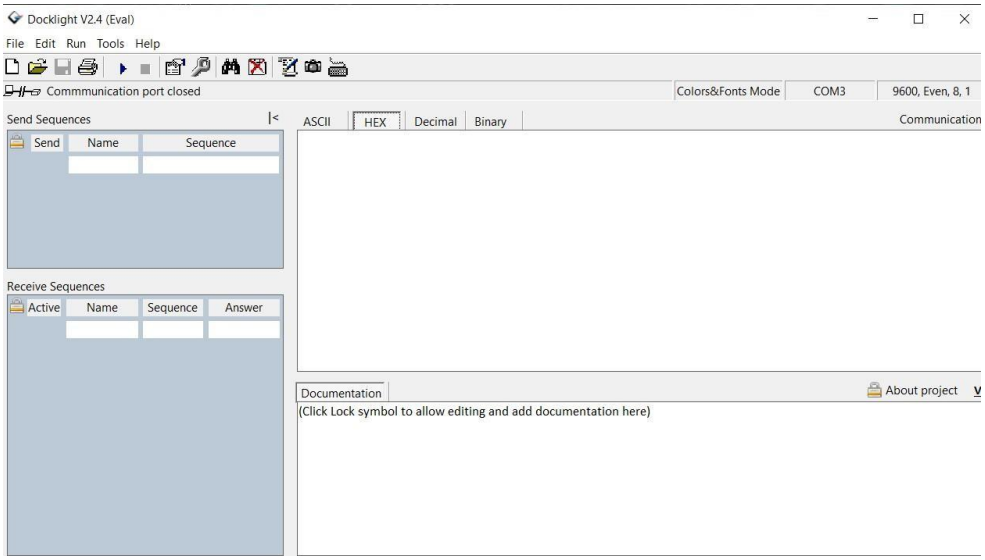


Figure III-15: Docklight interface.

III-3) Code Structure And Organization

The code structure and organization play a crucial role in the successful implementation of the communication protocol between the Master Terminal Unit (MTU) and the Remote Terminal Unit (RTU). In this section, we delve into the key aspects of our code structure and explain how it facilitates effective communication and maintainability.

Basically our work consists of three main codes, each plays a critical role to established reliable communication between the MTU and the RTU. Each code with its principal functions and utilities are described as follow:

- We create a library for handling the T101 protocol. The code includes functions for constructing ASDUs, calculating checksums, and implementing various function codes defined by the T101 protocol. We add two particular codes to improve communication and make it more reliable.

- The second code snippet appears to be a script that imports the serial library and a custom module called <*raw_data_generator*>. It includes a function called <*data_wrapper*> and a main function for sending data via a serial port.
- The third code snippet is a script that includes imports of the <*serial*> library and a custom module < *iec_101_functions* > which is the library we create. It contains functions for data manipulation and communication with serial ports.

III-4) The Implemented iec_101_functions Python Library

In order to establish a connection between the MTU and RTU, after deciding to use protocol 101, there was a lack of python libraries related to this implementation. So, we create our proper *iec_101_functions* library, as described in the first part of this work (Introduction).

As making the full library could take months or years to fully code, the following few points represent the scope of our library and its uses:

- Since the goal of the present work is to build an infrastructure for a smart meter, we only implement an unbalanced type of transmission (One Master Multiple Slaves)
- Detect common errors will be done by checking the important bytes and bits:
    - Start Byte
    - Checksum
    - End Byte
- The RTU needs to ignore any frames that aren't meant for it (not its address) and it also should ignore frames that are sent by other RTU in the system.
- To deal with the goal of minimal latency, we only use simple logic, and limit the use of time library only when necessary,
- The MTU should respect the link reservation protocol, as to limit errors.
- The RTU can only respond to 4 types of frames:
    - Link Reset
    - Link Statuts
    - Request Data class 1
    - Request Data class 2
- When data is requested from the RTU, only two types of data can be sent, integer or float, that can either be time tagged or not. So, in order to reduce the work and complexity of the program (there are 253 ASDUs in total). The Types of ASDU that the RTU can expect to send are limited to only 4 types:
    - The data is double and not time tagged 0x03
    - The data is double and time tagged 0x04
    - The data is float and not time tagged 0x0D
    - The data is float and time tagged 0x0E

- Each RTU need to build the frame accordingly to each ASDU type.
- When sent from the RTU to the MTU, data can be easily extracted and converted back by the MTU, from its hexadecimal into readable format (decimal). Data should have negligent or tolerable levels of errors. We create helper functions to do that
    - *float_to_hex* (used in the RTU) and *hex_to_float* (used in the MTU),

- *double_to_hex* (used in the RTU) and *hex_to_double* (used in the MTU).

As mentioned above, the MTU and RTU need link reservation, such as:

- If the LINK is not open, communication can't happen between MTU and RTU,
- The MTU checks if the LINK by sending a frame that has a function code of 0x9,
- If The MTU doesn't receive any response it means the LINK is closed and the MTU needs to establish a communication by sending a frame that includes a function code 0x0,
- When the RTU receives that frame, it will respond to the MTU then it will start to accept other frames,
- The MTU can check again by sending a frame that has a function code of 0x9 but this time it will receive a response from the RTU,
- The RTU checks for different kinds of errors from frames sent by the MTU and if and when that happens, the RTU closes the LINK and the MTU needs to restart the process all over again,
- The RTU wait for the MTU to start the link, and only when that happens, the RTU will start collecting data (to reduce the energy consumption).

Figure III-18 provides an overview of how the code works. It illustrates that the RTU accepts and reads frames only after receiving the reset link frame, which resets the communication channel. Once the link established, the RTU initiates communication based on the state of the function code byte in the received frame. It generates a response to the commands specified in the received frame. Then, it applies the main code. If an error occurs, the code will automatically restart itself, closing the link.

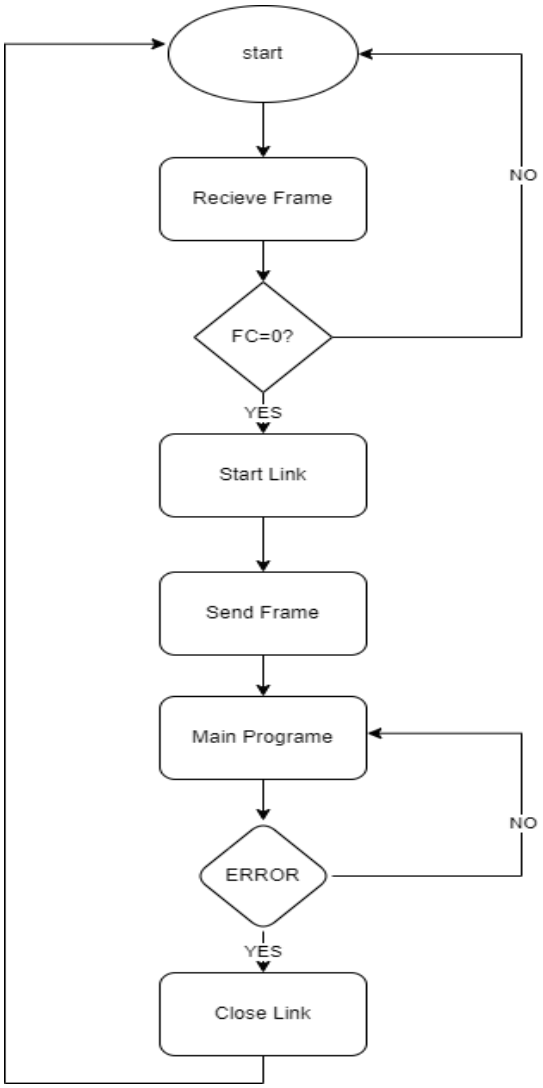


Figure III-16: Logigram Illustrating the Establishment of Communication

As seen in the logigram:

- The RTU waits indefinitely for link reset frame,
- When it receives it, it responds with a frame that acknowledges the link reset frame has been reserved,
- The RTU will keep receiving and responding to the MTU as long as there are no errors in those frames in the “Main program”,
- When the errors arise, the RTU close the LINK and start the process all over again.

The results of the process are shown in details in section III-5-2) Common Error Checking bellow.

While the link state is opened, the RTU start requesting data from the data source (in our case an Arduino). The data source can output 9 values of data, which we classifiy as Type 1 (class 1) data. The RTU then saves the data in a file. If data doesn’t exist or there were errors of transmission between the RTU and Arduino, the data would be labeled as None.

The RTU also responds to the frames sent by the MTU, and when the MTU requests data, it responds by either “data not available” FC 0x9 (the primary and secondary function codes have different meanings as shown in Figure II-9), or by sending a data frame, depending on the availability of data. The logigram in figure III-19 illustrates this process.

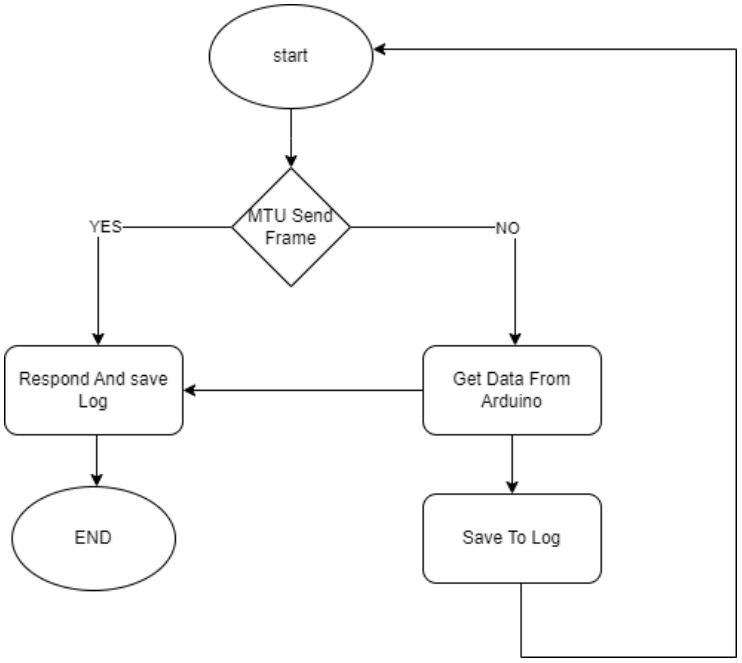


Figure III-17: MTU send frame and collection data.

The main function of our program, is the function *respond*, we create in our library. Its main task is not to make the RTU communicate directly with the MTU, but to find an appropriate response from the frames sent by the MTU.

The following logigram sketched in figure III-20 illustrates the main logic behind the *respond* function

- The *respond* function takes 4 arguments into consideration:
  - *Input_frame*
  - *Our_address*
  - *Data_class_1*
  - *Data_class_2*
- The *respond* fun starts by checking common error types:
  - Start Byte: compares the first byte of the frame and compares it with 0x10.
  - Checksum: Calculates the checksum of the *input_frame* and compares it with the checksum present in that frame.
  - End Byte: compares the last byte of the frame and compares it with 0x16.

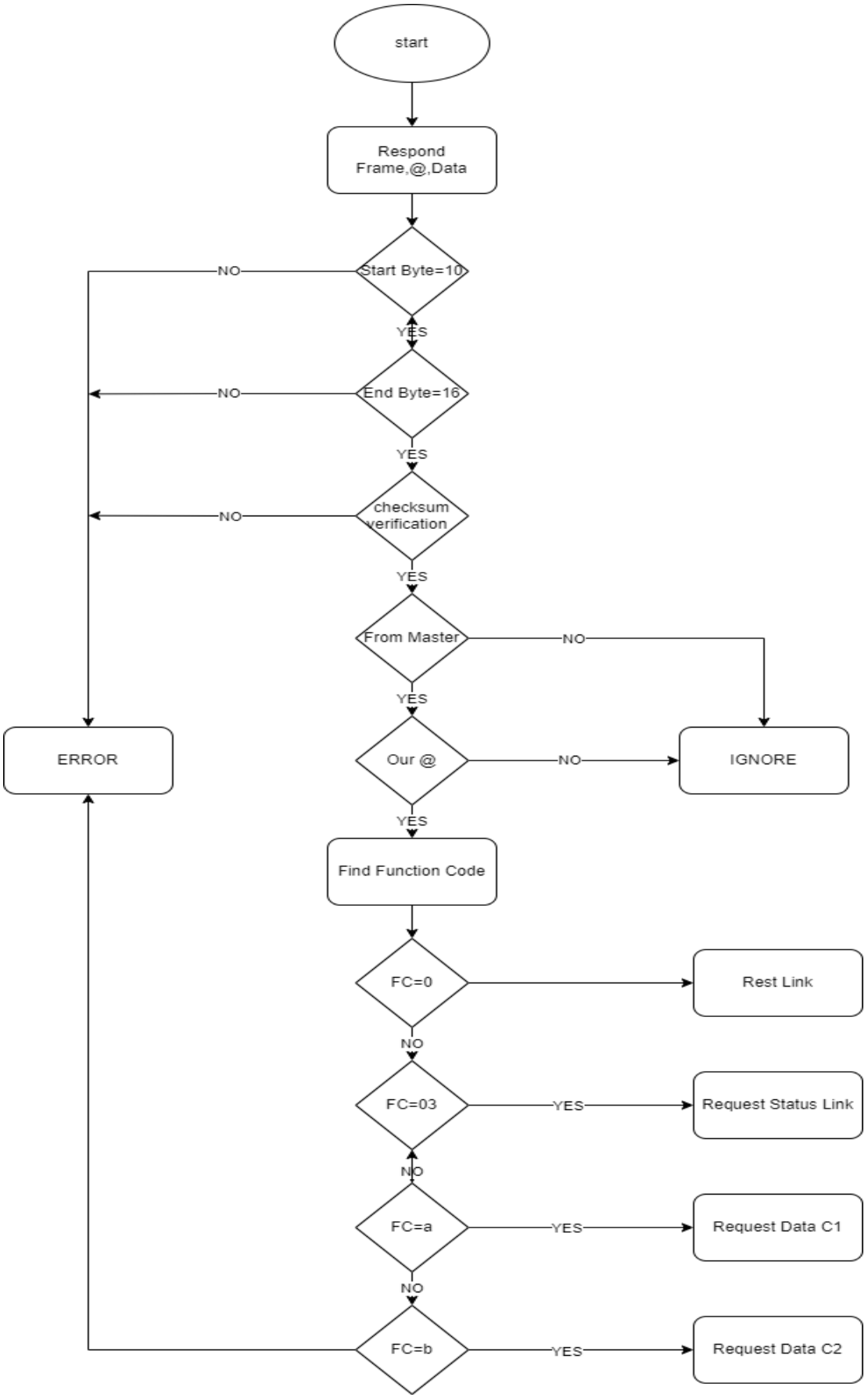


Figure III-18: frame and collection data sending from MTU

.

- If any of these parameters are False, an error raises and the link is closed. But in the case where they are all true, the RTU checks if the frame is from the master and destined to its addresss, so it performes the following chekings
    - Address check: compare the address present on the frame with "*Our_address*", the RTU address,
    - Goes to the PRM bit (the 6th bit in the control byte = 1, whether it is a frame from the primary)
- If one of these is True, the function just ignores these frames and does not continue any further, but if they are all False, it checks the FC part of the control byte (bits from 0 to 3). Due to time constraints, we code only 4 out of the 7 possible cases:
    - If FC = `0x0`, the RTU calls the function *reset_link* to build a fixed frame. The only thing that changes in the response frame, in our implementation, is *Our_address* (in this example *Our_address*= `0x09`). It builds the frame as: start byte 0x10, control field `0x00`, our address `0x09`. Then, it concatenates them and calculates the checksum using the function *calc_chacksum.* In the present case, it iss equal to `0x08`. It finally concatenates everything including the end byte `0x16`, so it returns the frame "`10 00 09 08 16`" that should adhere to protocol 101.
    - If FC = `0x9`, the RTU calls the function *request_link_status.* This function is similar to *reset_link,* the only difference is the control field which is equal to 0x0B. Assuming *our_address* is `0x09`, the response frame is "`10 0B 09 3A 16`".
    - If FC = `0xA`, the RTU calls the function *request_user_data_c1*. This function as explained below, receives data type 1 (*data_class_1)* only.
    - If FC = `0xB`, the RTU calls the function *request_user_data_c2.* This function has a similar function behavior as *request_user_data_c1,* but receives data type 2 (*data_class_2)* only .

The logigram in figure III-21 illustrates the behavior of the functions *request_user_data_c1* and *request_user_data_c2*. Figure III-21 shows also how functions *request_user_data_c1* and *request_user_data_c2* builds the output frames:

- First, the RTU checks if the data is None (not available) or not,
- If it is not available, it sends a fixed frame just like reset_link but with a control field equal to `0x09`, for example "`10 09 09 11 16`". The MTU reads the frame and figures out that the requested data is not available.
- If the data is available, the RTU tries to find the ASDU type and call *find_asdu.*
- Each data point has two main properties [value, time_tag]: value can only be float or double (int); time_tag can only be float (Exists) or None (doesn't exist).

- The RTU l checks the type of each value, and if it is time tagged according to the following priority, chooses one out of the following options

– The value is float and time tagged `0x0E`.
– The value is float and not time tagged `0x0D`.
– The value is double and time tagged `0x04`.
– The value is double and not time tagged `0x03`.

The reason for this priority list, is that all the values must share the same converting method (only one ASDU in the frame) and a float value is rounded to the nearest integer, thus, there will be a loss of information, but a double value can be written as a float value without losing. Time tagging is also more important than no time tagging, but less important than the actual value.

- After finding the ASDU type, the function builds a frame depending on that ASDU.

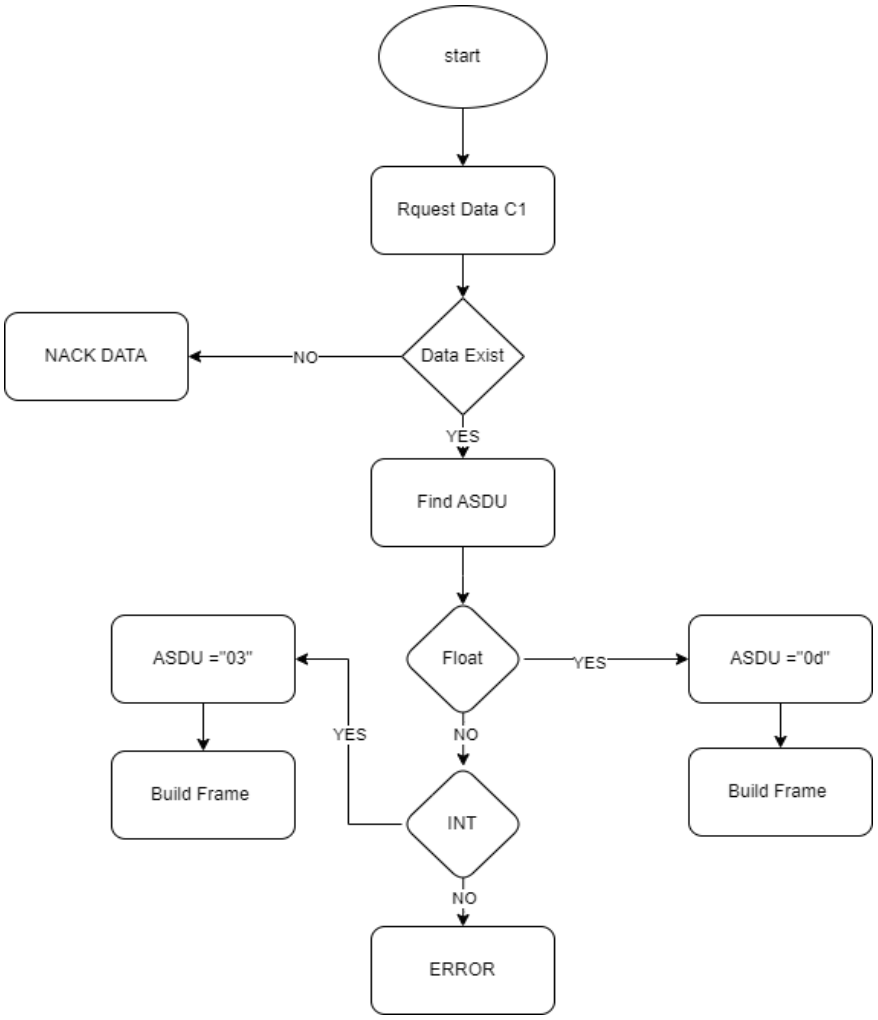


Figure III-19: The request of data_c1.

- Example we: have only 1 data point [1 , None] the response frame will be "`68 0D 0D 68 08 09 04 01 03 09 3F F0 00 00 00 00 00 8B 16`" since it follows the variable length format we will explain what each part means:

- 68: start byte 1
- 0D 0D: Two length bytes of data frame (Not the full frame) in this case the data frame is "08 09 04 01 03 09 3F F0 00 00 00 00 00"
- 68: start byte 2
- 08: control field with FC = 0x8 (data available)
- 09: our_address 1.
- 04: ASDU type (int and not time tagged).
- 01: The number of data points (in this case 1).
- 03: the cause of transmission, it was fixed in the full library as our scope was limited
- 09: our address 2.
- 3F F0 00 00 00 00 00: The value of the data coded on 7 bytes of information using *double_to_hex* converts the value 1 to 3F F0 00 00 00 00 00. Using the inverse function *hex_to_double* on this value returns a value of 1.0.
- 8B: the checksum of "68 0D 0D 68 08 09 04 01 03 09 3F F0 00 00 00 00 00".
- 16: end byte.

III-5) Tests And Results

In this part we show the results of our algorithms and the different types of errors that can occur. We also explain them and how they affect our algorithm. Some of the tests we prorm are, but not limited to,

- Link reservation from The MTU and RTU point of views,
- Error Checking,
- Showcase the frame_log and the data_log files,
- MTU & Data Extractor,
- Conversion errors,
- Latency.

III-5-1) Link Reservation Results

In this part, we run one crucial test as it allows checking whether the link is open or not. Without this test, there won't be any communication between the MTU and RTU. The test consists of three parts as follow:

- MTU sends a frame that has FC 0x9 (link status).
- MTU sends a frame that has FC 0x0 (link reset).
- MTU sends a frame that has FC 0x9 (link status).
- MTU sends a Faulty frame for example "00 00 00 00 00".
- MTU sends a frame that has FC 0x9 (link status).

The expected results are:

- RTU doesn't respond because link is not open yet.
- RTU opens LINK and responds FC 0x0.
- RTU returns a confirmation that the link is open FC 0xB.
- RTU detects the error and closes the link.
- RTU doesn't respond because link was closed.

For better understanding and view the outputs, we only keep outputs needed for this test, as shown in figure III-22.

```
[TX] - 10 59 09 0C 16
[TX] - 10 50 09 03 16
[RX] - 10 00 09 08 16
[TX] - 10 59 09 0C 16
[RX] - 10 0B 09 3A 16
[TX] - 00 00 00 00 00
[TX] - 10 59 09 0C 16
```

Figure III-20: Results from test 1 on Link Reservation from MTU.

As you shown in figure III-23, the MTU only receives a response from frames 2 and 3, while other frames are ignored.

```
Waiting for Link to open...
Waiting for Link to open...
Waiting for Link to open...
Recieved frame: 1059090c16 but Link is closed..
Waiting for Link to open...
Waiting for Link to open...
Waiting for Link to open...
Waiting for Link to open...
LINK OPENED:: Recieved frame: 1050090316, Sent Frame: 1000090816
Recieved frame: 1059090c16, Sent Frame: 100b093a16
Recieved frame: 0000000000 due to error link is going to be closed
Waiting for Link to open...
Waiting for Link to open...
Waiting for Link to open...
Recieved frame: 1059090c16 but Link is closed..
Waiting for Link to open...
Waiting for Link to open...
```

Figure III-21: Results from test 1 on Link Reservation from RTU.

– It keeps printing "waiting for link to open…", and every time it receives frames, it ignores them since the link is closed, and keeps waiting for the link reset command.

– Once the link is opened, via the Received frame: `1059090c16`, the RTU keeps responding to the MTU with a Sent Frame: `100b093a16`.

– When the RTU receives a faulty frame, it closes the connection (Received frame: `0000000000` due to error link is going to be closed).

As expected, these results mean that the link reservation algorithm is implemented successfully.

III-5-2) Common Error Checking

In this part, we test the behavior of the program when it receives some faulty frames and frames that it should ignore.

- Start byte is wrong "`11 59 09 0D 16`".
- End byte is wrong "`10 59 09 0C 17`".
- Checksum is wrong "`10 59 09 01 16`".
- Not from master "`10 00 09 08 16`".
- Not our address "`10 50 10 5F 16`".

Expected results:

- Whenever it receives either a wrong start byte or end byte or checksum, we expect an error to raise and the link to close.
- The program should ignore frames that are not from the master and that aren't meant for us (wrong address).

In figure III-24, we can see that the RTU responds by closing the link whenever it received either a wrong sart byte, end byte or checksum (lines 2, 4, 6) : we need to open the link each time after that to continue the test. However, it ignores the frames and does not close the link when it receives a frame from another RTU (line 8) or when the address is meant for it.

```
LINK OPENED:: Recieved frame: 1050090316, Sent Frame: 1000090816
Recieved frame: 1159090d16 due to error link is going to be closed
LINK OPENED:: Recieved frame: 1050090316, Sent Frame: 1000090816
Recieved frame: 1059090c17 due to error link is going to be closed
LINK OPENED:: Recieved frame: 1050090316, Sent Frame: 1000090816
Recieved frame: 1059090116 due to error link is going to be closed
LINK OPENED:: Recieved frame: 1050090316, Sent Frame: 1000090816
Recieved frame: 1000090816 it was ignored
Recieved frame: 1050105f16 it was ignored
```

Figure III-22: Results from test 1 on Link Reservation from RTU.

As we can see, the test validates the expected results. As a result, the program successfully detects faulty or ignorable frames, and deals with them accordingly.

III-5-3) Log Files

As the RTU receives both data from the Arduino and frames from the MTU, it is necessary for the RTU to keep logs of this information, as shown in figure III-25 and figure III-

26. The Log files are tools to make the system recover, in case it runs into unforeseen errors that may break the smooth running of the code.

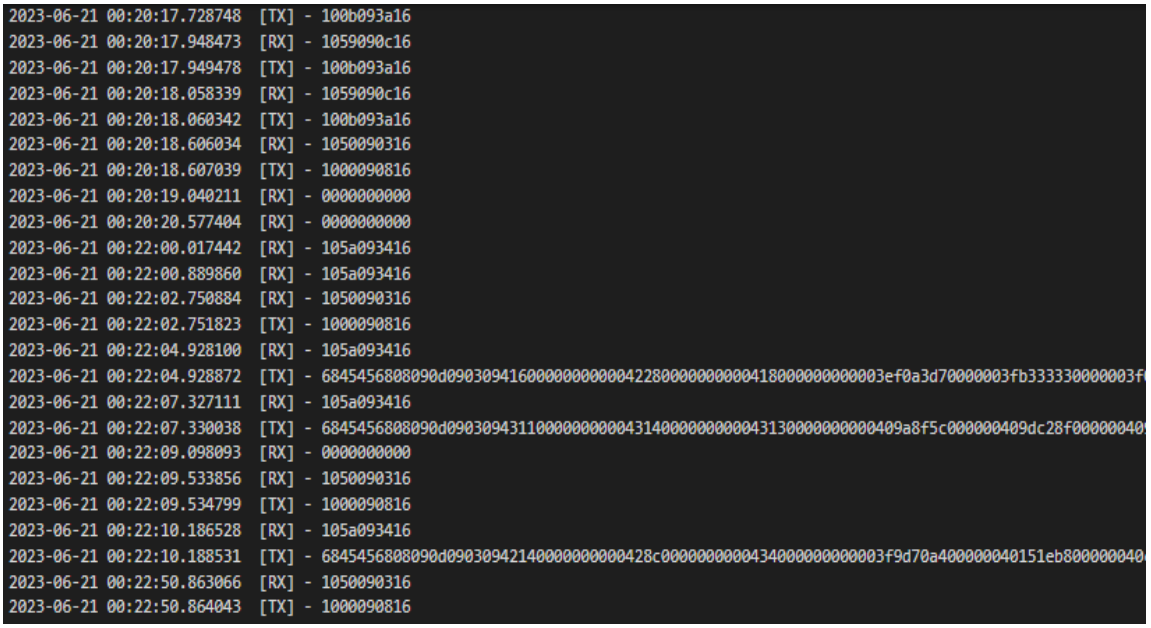


Figure III-23: Frames saved into Log file.

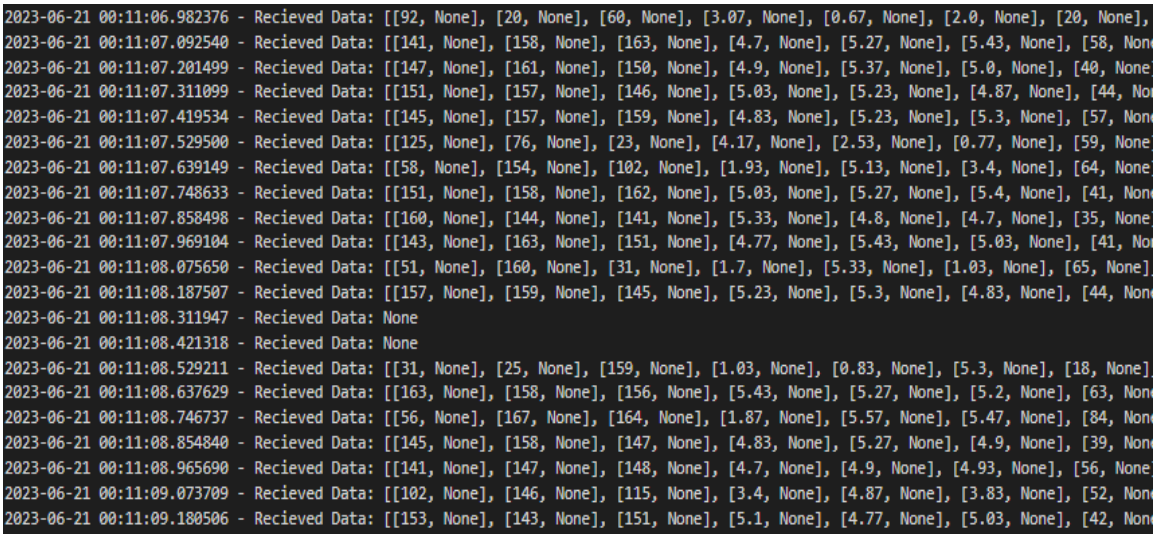


Figure III-24: Data Saved into Log file.

III-5-4) MTU And Data Extractor

While using docklight is very important throughout the development of our solution, it is rather limited since it is a free version. It does not fulfill all the needs of the MTU, it could only send and receive frames to and from RTU, but when the RTU sends data, it can not extract and show them, from their frames to a more readable format.

For this reason, we create a GUI (Graphic User Interface) that fulfills our needs:

- Send and receive protocol 101 frames,
- Does basic error checks (Start Byte, End Byte, Checksum Error),
- Extract The data available in frames that contain data,
- Easy to use.

As seen in figure III-27, a simple and streamline GUI was created for the purposes of our application, the user interface will be used as follows:

- After entering the name of the port in the *Port* textbox we press the *Connect* button to connect to the RTU.
- The connection will be made in a couple milliseconds and we can start sending frames and command the RTU from our interface.
- We can see in figure III-27 textbox with buttons connected to them, the button sends the frame of the connected text box, in the example picture:
    - 1st frame is a primary *reset_link* command
    - 2nd frame is a primary *status_link* command
    - 3rd frame is a primary *request_data_calss_1* command
    - Keep in mind the user can change the input frame and sends different kinds of frames
- The last important area to discuss and explore can be organized into two output areas:
    - Communication History: this output area deals with keeping the history and organizing the history of communication between the MTU and RTU and colors were used to improve the user experience, the color code goes as follows:
        - Green for time.
        - Blue for frames sent.
        - Red for frames received.
    - Extracted Data: This output area is only usable for two cases:
        - When there's an error in the frame. (error checking is the same as the RTU so there won't be a need to repeat the tests here)
        - Or more importantly when the MTU receives a variable length frame that doesn't have an error and extracts the data within it, it first reads the ASDU of the frame and depending on what kind of ASDU (0x03, 0x04, 0x0d and 0x0e) it applies the *hex_to_float* or *hex_to_double* on the the data, if the data is time tagged it will also extract the time tag. The figure III-28show the results for when there's a *time_tag* inside the frame received and when there isn't.

As shown figure III-28, the MTU can easily extract the data (and time tags) if available on the frame, and then communicate about the way it acts.

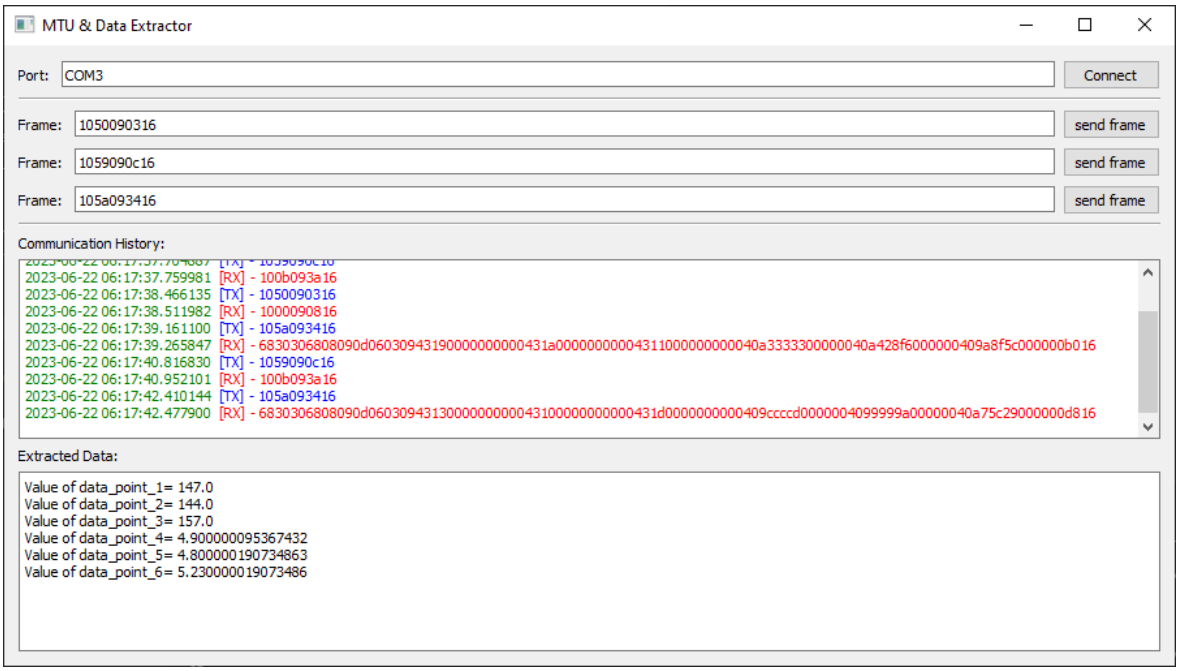


Figure III-25: MTU and Data Extractor graphic user interface

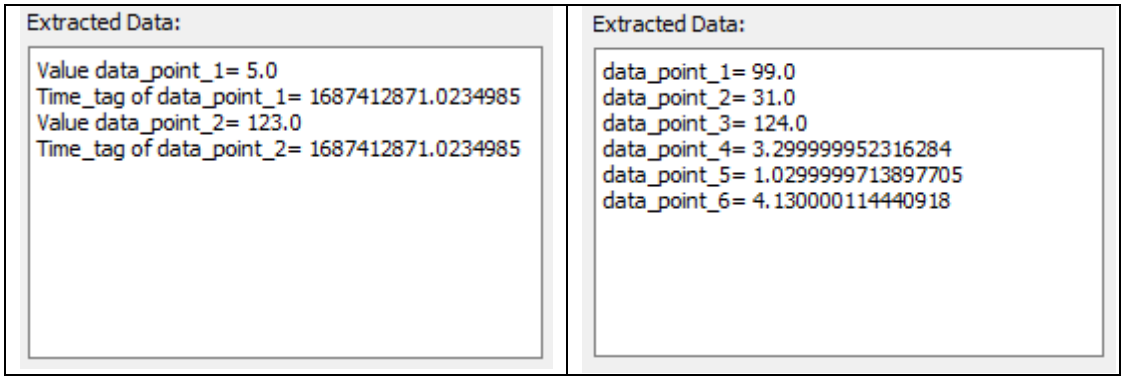


Figure III-26: Display of Data Extractor

III-5-5) Conversion Errors

In this part, we test the conversion mechanism implemented when sending variable length frames. This allows us to conclude on the validity of this method. If it has a large error rate, a better conversion method should be implemented.

In this test, we calculate the numeric error that can raise from converting data (both float and double) as shown in figure III-29. This can be done by creating a list of float and double values, using the *float_to_hex* (in case of float values) and *double_to_hex* (in case of double values), then converting them back using *hex_to_float* and *hex_to_double* respectively.

```
Float to Float conversion:
0.5 ******* 3f000000000000 ******* 0.5
5000.01 ******* 459c4014000000 ******* 5000.009765625
9956.153 ******* 461b909d000000 ******* 9956.1533203125
88.5 ******* 42b10000000000 ******* 88.5
6 ******* 40c00000000000 ******* 6.0
Average of error:  0.00011093749999417924, Maximum Error:  0.00032031249975261744
-------------------------
Double to Double conversion:
0 ******* 00000000000000 ******* 0.0
1 ******* 3ff00000000000 ******* 1.0
2 ******* 40000000000000 ******* 2.0
3 ******* 40080000000000 ******* 3.0
4 ******* 40100000000000 ******* 4.0
Average of error:  0.0, Maximum Error:  0.0
```

Figure III-27: Conversion Error Experiment

In the first part of the test, we convert values from float to hex and back to float again. The maximum conversion error is 0.0003 with an average error of 0.0001. These results are acceptable since our data doesn't have to be that precise.

In the second part, we convert values from double to hex and back to double again with an error of 0.0 This means double conversion methods are also valid.

III-5-6) Latency

In this part, we calculate the latency of the system, the time period elapsed for the MTU between sending a frame and receiving a response. In this test, we manually calculate the latency of different types of function codes, and see whether there is a connection between function codes and latency. The test is run 10 times, for each frame. We calculate then, the following parameters.

AOL = Average Of Latency.
MOL = Maximum Of Latency.
ACPS = Average commands per second.

Table III-9 : Average and maximum latency of different types of frames.

Function code	AOL (ms)	MOL (ms)	ACPS

Link Reset	56.2	99	17
Link status	**44.1**	**88**	**22**
Request data while data is unavailable	70.9	119	14
Request data while data is available	76.9	122	13

The table above shows the relationship between latency and the type of frame the MTU sent. Link status has the lowest latency, while request data frames generally have higher latency which makes them slower commands. This is probably related to a more complicated algorithm in those functions compared to the other simpler functions.

While the maximum Latency is 122 ms, the average is only 76.9 ms. This results in 13 commands treated per second. 13 full data frames per second seem to be enough for a stable, fast and robust communication. Generally, a MTU sends a request data frame each 5 seconds to each RTU it controls.

Conclusion and Outlook

Our study has shown that the IEC 60870-5-101 protocol is the most widely used in smart grids and SCADA systems, because of the many advantages it offers.

In Algeria, this protocol is not available on the market, so in our project we developed this protocol using the python language.

We proceeded by developing a library named <*iec_101_functions*> that includes the various standards and rules of this protocol. This library performs various functions, including:

- *reset_link*: this function tests the function code of the frame received by the RTU and if FC=0x00 the RTU rests the link and sent frame with FC=0x00. If the we don't reset the link, the RTU always demands it and the communication cannot be established.
- *request_link_status*: this function tests the function code (FC) of the MTU, if it is equal to 0x09, the *reset_link* is established and it sends a confirmation frame included on the value 0x0B, on the FC of the respond frame.
- *request_user_data_c1*: this function asks if there is a data class 1 by sending a frame with FC = 0xA, and if the data is None (note available, it sends a fixed frame length included FC=0x09 . If the data is available, the RTU builds a variable frame length which includes this data.

We test the elaborated library by creating virtual serial ports using the "Virtual Serial Port" software, to set the connection between the MTU and the RTU, and the "Docklight" software, which plays the role of the MTU, but does not display the data sent by the RTU.

Finally, we develop our own graphical interface to display the data and make it easier for the user to read.

We successfully conduct multiple tests on the various functions of our library, and the obtained results align with the expected operation, validating the reliability of our system.

Creating this protocol requires a significant amount of time, and due to time constraints, we couldn't cover all aspects of the IEC 60870-5-101 protocol. Therefore, we are exploring different possibilities for further developing this protocol. Some of the prospects we are considering include:

- The list of ASDU is quite bigger than the implemented 4 (0x03, 0x04, 0x0D and 0x0E). Although it can satisfy our needs for the current work it is still limited.
- While coding and creating the library we assumed and fixed a lot of variables, knowing where and why they were fixed such as
    - Transmission type was fixed to Unbalanced,
    - Cause of transmission was fixed to spontaneous,
    - Development time restraints made it impossible to implement the "frame count validity" mechanism.
- We could not get the authentic algorithm of how the checksum function works. Thus, an algorithm is arbitrarily implemented to fulfill this task. The main goal is "the same frame should always have the same checksum", it does not harm the smooth running of the program, but it is not the actual checksum function that the protocol uses.

Even the implemented code is not a fully completed library for the IEC-101communication protocol, it is still a good starting point for further development.

References

[1] R. Morello et al. "A Smart Power Meter to Monitor Energy Flow in Smart Grids: The Role of Advanced Sensing and IoT in the Electric Grid of the Future." IEEE Sensors Journal (2017).

[2] Jeannie Oliver et al. "The Energy Trilemma and the Smart Grid: Implications Beyond the United States." Renewable Energy eJournal (2017).

[3] Jung-Yoon Kim et al. "An efficient and versatile key management protocol for secure smart grid communications." 2012 IEEE Wireless Communications and Networking Conference (WCNC) (2012).

[4] Anthony P. Johnson et al. "The history of the Smart Grid evolution at Southern California Edison." 2010 Innovative Smart Grid Technologies (ISGT) (2010).

[5] B. Alagoz et al. "A user-mode distributed energy management architecture for smart grid applications." Energy (2012).

[6] Yakubu Tsado et al. "Resilient communication for smart grid ubiquitous sensor network: State of the art and prospects for next generation." Comput. Commun. (2015).

[7] Fang, Xi, et al. "Smart grid—The new and improved power grid: A survey." *IEEE communications surveys & tutorials* 14.4 (2011): 944-980.

[8] https://www.eolasmagazine.ie/ (https://www.eolasmagazine.ie/smart-grid-evolution/)

[9] Jeannie Oliver et al. "The Energy Trilemma and the Smart Grid: Implications Beyond the United States." Renewable Energy eJournal (2017).

[10] Zhichao Xu et al. "The Construction of Interconnected Communication System among Smart Grid and a variety of Networks." 2010 Asia-Pacific Power and Energy Engineering Conference (2010).

[11] Fang, Xi, et al. "Smart grid—The new and improved power grid: A survey." *IEEE communications surveys & tutorials* 14.4 (2011): 944-980.

[12] Zheng, Jixuan, David Wenzhong Gao, and Li Lin. "Smart meters in smart grid: An overview." *2013 IEEE Green Technologies Conference (GreenTech)*. IEEE, 2013.

[13] Ajenikoko, Ganiyu A., and Anthony A. Olaomi. "Hardware design of a smart meter." Int. Journal of Engineering Research and Applications 4.9 (2014): 115-119.

[14] https://www.reverecontrol.com/scada-basics-what-are-scada-and-telemetry/

[15] https://www.technavio.com/report/global-remote-terminal-unit-rtu-smart-grid-market-2015-2019

[16] https://www.prnewswire.com/news-releases/global-remote-terminal-unit-rtu-in-smart-grid-market-2014-2018-key-vendors-are-abb-group-general-electric-schneider-electric-and-siemens-ag-269197231.html

[17] https://www.allinterview.com/showanswers/102879/what-is-the-difference-between-rtu-mtu-in-scada-system.html

[18] https://www.sciencedirect.com/topics/engineering/remote-terminal-unit

[19] Donghyun Choi et al. "Advanced Key-Management Architecture for Secure SCADA Communications." IEEE Transactions on Power Delivery (2009).

[20] A. Singh et al. "Stability Analysis of Networked Control in Smart Grids." IEEE Transactions on Smart Grid (2015).

[21] Stefano Galli et al. "For the Grid and Through the Grid: The Role of Power Line Communications in the Smart Grid." Proceedings of the IEEE (2010). https://doi.org/10.1109/JPROC.2011.2109670.

[22] Rani Kumari et al. "Performance comparison of various routing protocols in WSN and WBAN." 2016 International Conference on Computing, Communication and Automation (ICCCA) (2016).

[23] D. Choi et al. "Design and Implementation of Wireless Fieldbus for Networked Control Systems." 2006 SICE-ICASE International Joint Conference (2006). https://doi.org/10.1109/SICE.2006.315745.

[24] Dan Li et al. "Advanced metering standard infrastructure for smart grid." 2012 China International Conference on Electricity Distribution (2012). https://doi.org/10.1109/CICED.2012.6508429.

[25] Clarke, Gordon, Deon Reynders, and Edwin Wright. Practical modern SCADA protocols: DNP3, 60870.5 and related systems. Newnes, 2004.

[26] ELMANSOUR, TARIG MOHAMED. *SIMULATION OF STANDARD TELECONTROL PROTOCAL IEC 6087O-5-101*. Diss. Sudan University of Science & Technology, 2006.

[27] INTERNATIONAL STANDARD CEI IEC 870-5-1: 1990-02

[28] INTERNATIONAL STANDARD CEI IEC 870-5-2: 1992-04

[29] INTERNATIONAL STANDARD CEI IEC 870-5-3: 1992-09

[30] INTERNATIONAL STANDARD CEI IEC 870-5-4: 1993-08

[31] INTERNATIONAL STANDARD CEI IEC 870-5-5: 1995-06

[32] Hodo, Ersi, et al. "Anomaly detection for simulated iec-60870-5-104 trafiic." Proceedings of the 12th international conference on availability, reliability and security. 2017.

[33] Cherifi, Tarek. *Développement d'un protocole SCADA temps réel sécurisé à base de l'IEC 60870-5-101*. Diss. 2019.

[34] INTERNATIONAL STANDARD CEI IEC 870-5-101: 1995-11

[35] https://infosys.beckhoff.com/index_en.htm

[36] Lu, Xie. "Supervisory Control and Data Acquisition System Design for CO2 Enhanced Oil Recovery." Master of Engineering Thesis, Technical Report No. UCB/EECS-2014-123. EECS Department, University of California at Berkeley (2014).

[37] TerryWarwick. "Raspberry Pi 2 & 3 Pin Mappings - Windows IoT." docs.microsoft.com. N.p., n.d. Web.

[38] "General Python FAQ — Python 3.8.4 Documentation." docs.python.org. N.p., n.

ملخص

تستعمل العدادات الـ الذكية البروتوكول 101 حصة -5-60870 IECلإرسال المعلومات بطريقة موثوقة وفعالة، رغم أن البروتوكول حاصل على كبيرة السوق العالمية، لا توجد حالة استخدام في السوق الجزائرية، ولهذا في إطار المشروع النهائي قررنا تطوير مكتبة برمجية تحتوي على بهدف استعمالها ي الشبكات الذكية وأنظمة5-60870 IECخصائص ووظائف البروتوكول SCADA

كلمات مفتاحية: *عدادات ذكية، 5 -60870 iec، SCADAشبكات ذكية*

Summary

Smart meters transfer data in a very reliable and efficient way using the IEC 60870-5-101 protocol. This protocol is the most used in industries worldwide, but it is unavailable in the Algerian market. Inour final year project, we have developed an IEC 60870-5-101 communication protocol for smart grids and SCADA systems by creating a library that includes the different rules and functionalities of this protocol, as well as an interface that displays the data for the users.

Keywords: smart meters, IEC 60870-5-101, SCADA, Smart Grid.

Résumé

Les compteurs intelligents transfèrent les données d'une manière très fiable et efficace en utilisant le protocole IEC 60870-5-101. Ce protocole est le plus utilisé dans les industries mondiale, mais, il est indisponible dans le marché algérien. Nous avons développé dans notre projet de fin d'études, un protocole de communication IEC 60870-5-101 pour les smart grids et les systèmes SCADA en créant une libraire qui inclue les différentes règles et fonctionnalités de ce protocole, ainsi qu'une interface qui affiche les données pour les utilisateurs.

Mots clés : compteurs intelligents, IEC 60870-5-101, SCADA, Smart Grid.