buildings

MDPI

Article

BIoT Smart Switch-Embedded System Based on STM32 and Modbus RTU—Concept, Theory of Operation and Implementation

Ionel Zagan [1,2,*] and Vasile Gheorghiță Găitan [1,2,*]

1 Faculty of Electrical Engineering and Computer Science, Stefan cel Mare University, 720229 Suceava, Romania
2 Integrated Center for Research, Development and Innovation in Advanced Materials, Nanotechnologies, and Distributed Systems for Fabrication and Control (MANSiD), Stefan cel Mare University, 720229 Suceava, Romania
* Correspondence: zagan@usm.ro (I.Z.); vgaitan@usm.ro (V.G.G.)

Abstract: Considering human influence and its negative impact on the environment, the world will have to transform the current energy system into a cleaner and more sustainable one. In residential as well as office buildings, there is a demand to minimize electricity consumption, improve the automation of electrical appliances and optimize electricity utilization. This paper describes the implementation of a smart switch with extended facilities compared to traditional switches, such as visual indication of evacuation routes in case of fire and acoustic alerts for emergencies. The proposed embedded system implements Modbus RTU serial communication to receive information from a fire alarm-control panel. An extension to the Modbus communication protocol, called Modbus Extended (ModbusE), is also proposed for smart switches and emergency switchboards. The embedded smart switch described in this paper as a scientific and practical contribution in this field, based on a performant microcontroller system, is integrated into the Building Internet of Things (BIoT) concept and uses the innovative ModbusE protocol. The proposed smart lighting system integrates building lighting access control for smart switches and sockets and can be extended to incorporate functionality for smart thermostats, access control and smart sensor-based information acquisition.

Keywords: smart home; remote terminal unit (RTU); BIoT; smart cities; Modbus; acquisition cycle; Industrial Internet of Things

check for updates

Citation: Zagan, I.; Găitan, V.G. BIoT Smart Switch-Embedded System Based on STM32 and Modbus RTU—Concept, Theory of Operation and Implementation. *Buildings* **2024**, *14*, 3076. https://doi.org/10.3390/buildings14103076

Academic Editor: Apple L.S. Chan

Received: 26 August 2024
Revised: 17 September 2024
Accepted: 23 September 2024
Published: 26 September 2024

1. Introduction

The emergence of Industry 4.0 and the continuously expanding request for automation [1] have led to the necessity of efficient energy management and control systems based on intelligent buildings in an attempt to optimize energy consumption. Traditional buildings usually supply electricity to electrical devices using on/off switches from power sources to consumers. These control components need to be improved as smart buildings become more and more popular and therefore it is a necessity to have them in everyday life. Smart switches are those elements of a smart building that additionally implement visual and acoustic signals in cases of emergencies, ambient information display, smart thermostats and access control systems, i.e., information acquisition based on smart sensors in an efficient and non-invasive way. Smart switches are responsible for lighting system physical control in a building based on physical user interaction with switches present on them or commands communicated by a smart switchboard. Thus, data acquisition, as well as the connectivity of systems integrating smart switches, brings together integrated building automation and a modern Building IoT (BIoT) [2,3] framework. The benefits are increased energy efficiency, practical application of the smart switch function potential, intelligent control and a flexible BIoT system. Due to the connection of a large number of electrical systems, electronic and household appliances, and more, energy consumption tends to increase uncontrollably as their number increases. As a result, the operational cost in terms of energy consumption increases concerning the quality of service, implemented function redundancy, efficiency and sustainability.

With the rapid development of embedded systems, communication protocols, distributed systems [4], photovoltaic plants and solar-powered street lighting systems [5], smart building [6] automation is becoming increasingly interconnected. Thus, current research in intelligent building control and security systems can make a significant contribution to smart building control, integration of environmental monitoring functions, intelligent lighting, signaling and more. The continuous evolution in integrated circuits, computing and communication technologies has led to a new generation of sensors called smart sensors, which are capable of communicating wired or wirelessly, as well as automatically processing data related to certain events or physical parameters. These parameters are usually represented by analog signals and not in a digital form that microcontrollers operating on discrete data values can process. At the same time, smart switches, cyber-physical systems (CPSs) and the IoT concept [7], developed together with existing technologies and communication protocols, are successfully used in the smart city concept [8] and various BIoT applications [9] to design an energy-smart environment.

Smart switches are those devices that can bring additional benefits when designing smart lighting systems in the context of BIoT for sustainable energy and the environment. Detecting changes in light and environment is an important part of developing energy-efficient lighting for buildings. Thus, ambient light sensors can be used to switch off the main lights when there is sufficient illumination available. Smart switches and sensors can also provide real-time data for monitoring and controlling an energy-efficient building. A smart switch consists of a set of sensors, a processing module and a network communication module [10]. Thus, there are thousands of sensors produced by many different vendors using different communication interfaces and protocols, so data exchange and switch interoperability are major challenges in data-acquisition communication systems. The network communication module in a smart switch relies on a standard wired or wireless communication protocol. The interoperability of smart sensors is the ability to exchange information and to easily use information that has been exchanged through a standard communication protocol to achieve specific functions or goals in energy-efficient smart lighting. Smart switches can use different communication models [11], such as client–server and publisher–subscriber communication models. Thus, interoperability testing is used to verify whether two or more device implementations based on the same standard can interoperate using the communication technologies integrated into the system.

Popular communication interfaces designed for intelligent lighting systems include Digital Addressable Lighting Interface (DALI), Ethernet, ZigBee light link, Bluetooth and Wi-Fi [12,13], which can be integrated to organize different lighting zones with various settings [14]. Thus, smart lighting devices can autonomously determine the required light intensity according to prevailing conditions and allow flexible lighting for a variety of activities and users [15]. An add-on benefit can be the function of real-time energy metering. Smart lighting systems allow various types of luminaires to intercommunicate and synchronize [16]. An advantage is also the possibility of individually controlling the luminaire via a remote control, e.g., a web or mobile app.

As a novelty brought to this field, the project described in this paper mainly improves the lighting system and optical and acoustic signaling for emergencies. The scientific contributions concern the proposal of an embedded system based on the novel ModbusE RTU protocol extension. As a practical contribution, it can be highlighted that the proposed smart switch has been designed and implemented as an experimental laboratory device in the framework of industrial research activities. It is composed of an ARM microcontroller that facilitates the implementation of a well-structured graphical user interface (GUI) via touch display, which is useful and at the same time intuitive for lighting control in a smart building. The embedded system controls actuators that can be located remotely or locally, ensuring compatibility with existing electrical installations in a building. Thus, the ability to operate electrical switching elements is more efficient as it can be operated both locally via the smart switch and remotely via a mobile phone using wireless communication. This control strategy of the proposed system is favorable to people with disabilities in that the device is autonomous in operation and the control process is distributed throughout.

The rest of this paper is organized as follows. Section 2 reviews current research in smart switch systems and BIoT designs and Section 3 presents a smart switch-embedded distributed system architecture and ModbusE protocol integration. Sections 4 and 5 describe the ModbusE protocol and present experimental data on the proposed smart switch validation presenting the concept, and implementation results including contributions, and Section 6 presents the conclusions.

2. Related Work

Smart switches are responsible for physically controlling the supply of AC power to the lighting system in a building. Electrical or electronic devices perform this task based on the user's physical interaction with the switches or based on commands communicated by a smart switchboard. Thus, current implementations in the BIoT field include a wide range of products and prototypes based on a multitude of communication protocols and hardware configurations. When it comes to light control and lighting scenes, ZigBee Light Link [17] proposes a protocol integrated into a series of highly flexible yet robust devices [18].

In [19], to automatically minimize energy consumption, an intelligent switch control system is developed that is based on open-source software and can be set up with no physical configuration of the communication environment. In the Smart Switch control system proposed by the authors, Android application is utilized to control the switches remotely using Wi-Fi. The device consists of two main parts, an Android app and a device that includes a programmable Arduino board, ESP8266 module (produced by Espressif Systems), a socket and an SD card. Thus, the main objectives of the paper [19] were to investigate, develop, realize and validate an intelligent switch control system using Wi-Fi technology and to benchmark the system in terms of precision and timing of switch control. In [20], the authors provide a comprehensive literature review involving blockchain technology applied to smart cities. Then, they look at how blockchain technology is being applied to smart cities from the perspective of citizens, smart healthcare, smart grid, smart transportation, supply chain management and others.

The application proposed in [21] can help users control and monitor the use of lights remotely via a smartphone connected to the Internet. The method used in this research project is an experimental one carried out in several stages such as hardware design, software design, testing and evaluation. This project uses a microcontroller that can control the electrical switch connected to a networked mini server. As technology is advancing in every field, it has also advanced in making smarter homes where home devices can be controlled remotely [22]. Here, the word smart also helps to generate electricity bills in a way that reduces human effort in reading and calculating all the necessary parameters. Accordingly, through the research work presented in [22], the authors have developed a smart system to obtain a controlled and efficient billing system. The concept proposed in [23] is considered by the authors to be a very useful one, as our future world requires a more intelligent way to manage all electrical and electronic devices in the context of BIoT. The user interaction is realized through an Android application running on a smartphone that interfaces to the microcontroller using WiFi, where cameras can be inserted in the building, as well as break switches that can be included in each room as widgets, thus showing a live simulation of the network of switches in the house, indicating whether they are on or off. The app enables efficient energy management by showing the user which consumers are switched on unnecessarily. The "branch switch" concept is designed to make home automation feasible with very few parts and reduce costs relative to similar implementations.

In [24], the authors formulate a definition of the smart home concept, which is meant to represent the extension of residential building automation. The paper presents a project for the implementation of a low-cost smart home remotely managed by a phone running Android. Thus, control and automation of systems that provide such human comfort as lighting, heating, cooling, ventilation, air conditioning and related security were considered. This system is designed to control electrical devices throughout the home and is characterized by ease of installation, simplicity of use and design, and cost-effective deployment. The system design does not remove existing electrical switches but rather provides more secure control over the switches. In [25], the authors propose an LED Smart switch with lightweight middleware, which provides location-based services and pattern-based prediction services. Moreover, the smart switch locates the user and interconnects between other switches through a network connection.

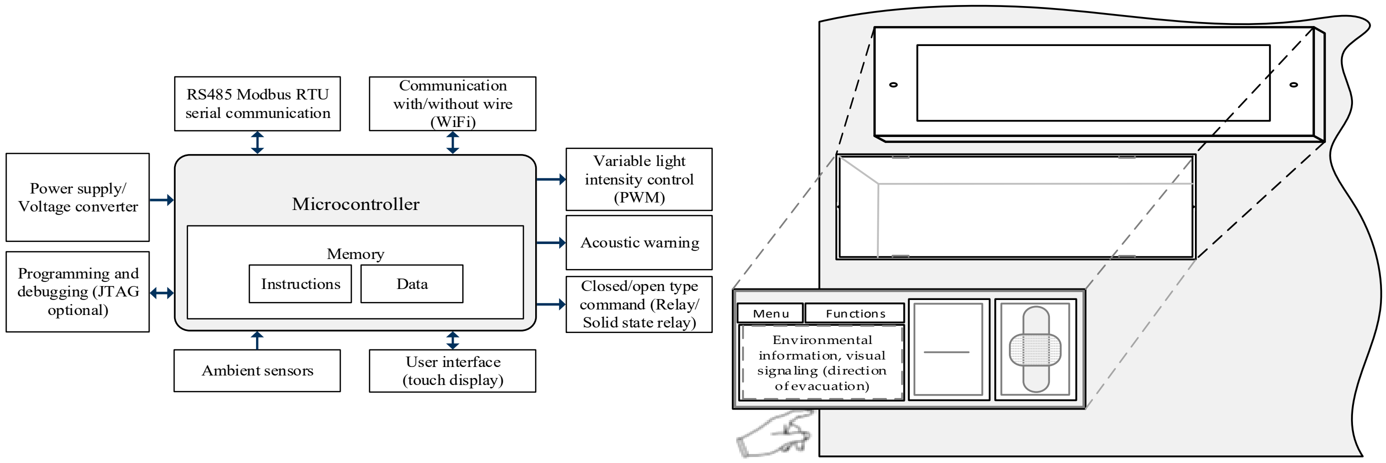
The work presented in [26] uses a "web application" and "cloud" to control the smart switch operation. A cloud server is created for the environment in which the switches are mounted. The smart switch uses a display to provide simple services and useful information such as weather forecasts and light status information using icons. The user communicates with the processor via the web application. The system described in [26] is, moreover, platform-independent, which means that any system with a web browser can use the application, thus avoiding the need for operating system-specific apps and minimizing development expenses. To enhance convenience in everyday life, the IoT paradigm is now a popular research topic. Different consumer appliances, however, provide various functions and services. In ref. [27], an IoT SH is proposed by utilizing a Cloud smart Tetris switch, including a Home as a Service (HaaS) Cloud server coupled with IoT appliances. Additionally, the dynamic extensible module is embedded so that IoT-based applications enable the identification service. The HaaS server is integrated by the authors to supply the user interface for the consumers, storing any information or data corresponding to the particular smart home and querying the operation information of each home device.

3. Smart Switch-Embedded Distributed System Architecture and ModbusE Protocol Integration

3.1. Smart Switch-Embedded System Architecture and STM32 Implementation

There is a growing requirement for low-cost smart switch-management solutions that can remotely control switches or home devices in residential areas utilizing apps or websites. The research in this project is centered on the patent request (OSIM A/00224 CBI from 2023) submitted by the inventors describing the smart switching device (DIGI-TOUCH) proposed in this research paper. This paper presents the design of an embedded application interface for the DIGI-TOUCH class B laboratory model using the touch screen TFT LCD. TouchGFX 4.21.1 Designer (X-CUBE-TOUCHGFX), STM32CubeIDE 1.11.2 and STM32CubeProgrammer 2.11.0, respectively, were used to achieve these objectives, the microcontroller used in this project is STM32F429ZIT with a maximum CPU clock of 180 MHz. Thus, the STM32F429I-DISCO kit uses SRAM1 (112 Kbytes) to handle the display on a 320 × 240 resolution LCD with 8 bpp (75 Kbytes). The STM32F429ZIT microcontroller architecture (produces by STMicroelectronics) consists mainly of a 32-bit multilayer Advanced High-Performance Bus (AHB) matrix interconnecting ten masters and eight slaves. The LCD-TFT display controller (LTDC) is one of ten AHB master devices on the AHB bus [28]. Distributed for free under the MIT open source license, FreeRTOS which was used in the smart switch project described in this paper includes a set of IoT libraries suitable for use in all industries [29]. FreeRTOS is based on examples that include all essential libraries to securely connect to the Cloud and provides methods for multiple threads of execution, mutexes, semaphores and software timers [30]. The RTOS core is based on the idea of parallel threads of execution. Similar to the real world, an application usually needs to perform several different tasks.

The smart switch with optical and acoustic signaling for emergencies is represented by the embedded system in Figure 1a based on the ARM Cortex-M4 microcontroller (STM32F429ZIT), integrated into the building structure and compatible with existing accessories (Figure 1b). According to the proposed embedded system, the IoT smart switch device consists of, as appropriate, a set of buttons rendered on a touch display. The actuators are controlled via an intuitive menu and the ModbusE communication bus is used to receive messages specific to emergencies. Thus, it is possible to display both evacuation directions and environmental information. In addition, there is a temperature and humidity sensor (HTU21D) connected to I2C3, PA8 (SCL) and PC9 (SDA) pins, the wireless communication module, the on/off relay module, the variable light intensity-control module, the power module, the integrated programming and debug connector, and the buzzer module (Figure 1a).


RS485 Modbus RTU serial communication
Communication with/without wire (WiFi)
Power supply/ Voltage converter
Microcontroller
Memory
Instructions
Data
Programming and debugging (JTAG optional)
Variable light intensity control (PWM)
Acoustic warning
Closed/open type command (Relay/ Solid state relay)
Ambient sensors
User interface (touch display)
Menu
Functions
Environmental information, visual signaling (direction of evacuation)

(**a**) (**b**)

Figure 1. (**a**) DIGI-TOUCH device block diagram corresponding to the smart switch with optical and acoustic signaling for emergencies. (**b**) Smart switch integration into the building structure and compatibility with existing accessories.

Several patent solutions (CN107980106B, US7535131, US9575587, US10615995) are known to increase the energy efficiency of a smart building, mainly consisting of a set of electronic systems, processing units, sensors and communication elements, so that they are used, where appropriate, with the related operating diagrams to ensure optimal environmental comfort and integrated automation of smart buildings in the context of the IoT paradigm. Improvements can be made through related functions such as energy saving, disability support, remote control and more. Analyzing current architectures, it can be stated that the control process of the lighting system and implicitly of the intelligent building is incomplete and partially inefficient when considering possible undesirable events (e.g., flood, intrusion, etc.) or emergencies (fire alarm, earthquake, etc.), or in the case of the possible evacuation of people using signaling access ways, so a complex control strategy is required.

As a novelty, the proposed concept refers to a smart switch with optical and acoustic signaling for emergencies, designed to control the lighting and signaling of emergencies in a building, so that the overall system is controlled both via the GUI rendered on a touch-sensitive display that facilitates easy operation of the switch and via ModbusE serial communication. The smart switch with optical and acoustic signaling for emergencies overcomes the disadvantages presented above by implementing acoustic and visual signaling functions in the embedded device for emergencies and ModbusE communication. The main advantages of the designed system are conceptual, so that the smart switch performs basic functions, i.e., switches on/off the lighting in a room via a button shown on the touchscreen display and controls the light intensity via a sliding button, respectively.

The distributed system allows the lighting control commands to be received via a local ModbusE, via a wireless phone, optionally via a gesture-recognition application or voice commands for people with disabilities, or remotely via ModbusE communication. Traditional buildings usually supply electricity to electrical devices using a closed/open contact means of electrical distribution from power sources to consumers. These control components need to be improved as smart buildings become more and more popular and therefore it is a necessity to have them in our daily lives. The BIoT smart switch-embedded system device (DIGI-TOUCH) based on STM32F429 and ModbusE RTU brings the following advantages:

- The smart switch introduces new optical and acoustic emergency signaling, display and switching capabilities;
- The device is simple, modern and compatible with traditional electrical installations, with a variety of actuation possibilities, so it takes the form of an ordinary switch;
- The embedded device can be integrated into the smart building and is compatible with its electrical components;
- The system has no negative impact on users' daily lives as it works in a non-invasive and ergonomic manner;
- The system can eliminate traditional electrical contacts, i.e., flashing, so it no longer encourages deflagration in the unlikely event of accidental methane gas leaks;
- The system provides a power supply for connected lighting devices and optionally can maintain the current state of the contact after the supply voltage has been interrupted;
- The system improves the energy efficiency of lighting and/or power supply systems because it is configurable and flexible, so it can be controlled in an on/off manner and can control the light intensity via a slider;
- The switch is operated locally by physically touching the buttons shown on the touch display or optionally via an app on your phone, a gesture-recognition app or voice commands for the disabled, or remotely via ModbusE communication.

3.2. BIoT-Embedded System Integration and Distributed System Architecture Using ModbusE Communication Protocol

The distributed system architecture integrating the proposed device, shown in Figure 2, can optionally include a Gateway (STM32H7B3I-DK) between Ethernet/WiFi and Modbus RTU using USB Virtual COM, the server stations (smart switches) provided with ModbusE server connected to the communication network based on RS485 line standard. DIGI-TOUCH architecture may also include a voice control interface (Google, Alexa, Siri) for persons with disabilities, as appropriate, fog and cloud computing for IoT connectivity and an auxiliary client with a configuration and testing application. All technological aspects are assembled so that the system configuration matches the final application in almost every aspect. When the touchscreen buttons are operated, the corresponding digital outputs are controlled (HAL_GPIO_TogglePin (GPIOG, GPIO_PIN_3)) according to the embedded device features and configuration. Specific advantages derived from the proposed smart switch device are that it provides optical and acoustic signaling for emergencies; i.e., it allows the measurement and display of temperature and humidity. At the same time, the ARM microcontroller, via the touch display, reproduces certain environmental information, and if a fire signal is received from the alarm-control unit, the microcontroller displays information on the evacuation direction and performs local acoustic signaling, respectively. The smart switch with optical and acoustic signaling for emergencies, according to the research project, can be manufactured with the same performance and characteristics whenever desired for any distributed lighting-control application compatible with it or its modules, which is an argument for meeting the industrial applicability criterion. The ModbusE protocol, with its RTU and TCP/IP variants, is one of the oldest and most widely used and is also partly defined. Since ModbusE also operates at high communication speeds, fast processing of communication frames is required, assembling all the necessary technology components for the implemented communication protocol. The different events represented in Figure 3 are summarized below. The event ev_sel_Comm is emitted by the GUI_Task to the ssThreadComm task, enabling the communication mode, and the events event event_receive as well as event_transmit are events produced by the callback functions when a message is completely received.

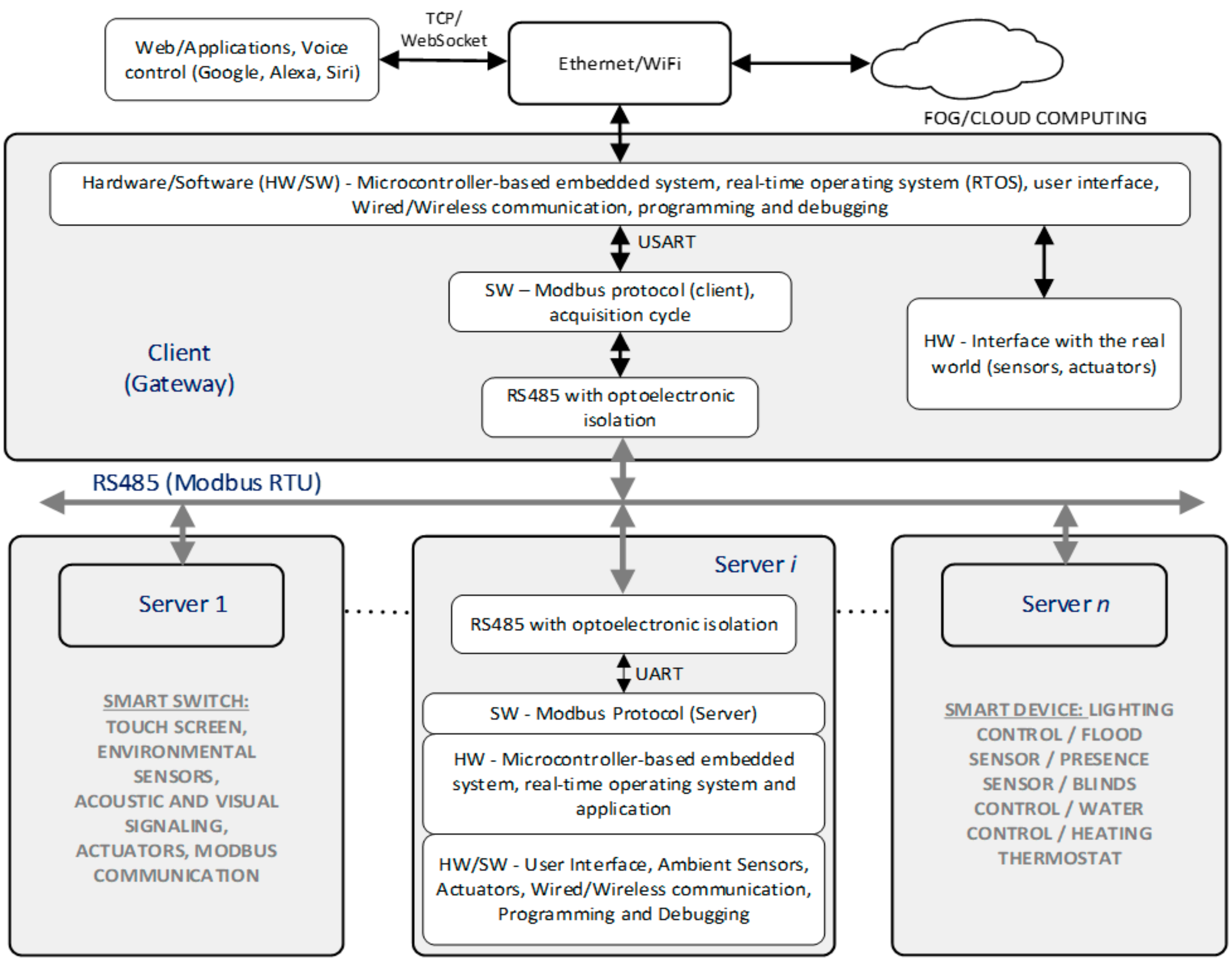

Web/Applications, Voice control (Google, Alexa, Siri)
TCP/ WebSocket
Ethernet/WiFi
FOG/CLOUD COMPUTING
Hardware/Software (HW/SW) - Microcontroller-based embedded system, real-time operating system (RTOS), user interface, Wired/Wireless communication, programming and debugging
USART
SW – Modbus protocol (client), acquisition cycle
HW - Interface with the real world (sensors, actuators)
Client (Gateway)
RS485 with optoelectronic isolation
RS485 (Modbus RTU)
Server 1
SMART SWITCH: TOUCH SCREEN, ENVIRONMENTAL SENSORS, ACOUSTIC AND VISUAL SIGNALING, ACTUATORS, MODBUS COMMUNICATION
Server *i*
RS485 with optoelectronic isolation
UART
SW - Modbus Protocol (Server)
HW - Microcontroller-based embedded system, real-time operating system and application
HW/SW - User Interface, Ambient Sensors, Actuators, Wired/Wireless communication, Programming and Debugging
Server *n*
SMART DEVICE: LIGHTING CONTROL / FLOOD SENSOR / PRESENCE SENSOR / BLINDS CONTROL / WATER CONTROL / HEATING THERMOSTAT

Figure 2. Distributed system architecture incorporating the DIGI-TOUCH smart switch with optical and acoustic signaling for emergencies.

The events ev_MODBUS_execute and ev_execute_MODBUS initiate the execution of a ModbusE command to the target task and a response to indicate the completion of this execution (from ssThreadComm to ssThreadAcq, GUI_Task and ssThreadControl, respectively). The ev_start_acquisition is an eveniment that causes the ssThreadAcq task to start the actuation procedure, and the ev_acquisition_ made is an event that informs the GUI_Task task of the finalization of this process acquisition. The action_event event requires the ssThreadControl task to update the output with the latest button state.

ModbusE maintains Modbus RTU compatibility. The acquisition cycle (AC) comprises a slot named SYNC, (optionally) which can be utilized by the MASTER (gateMBE station as Modbus RTU client) to trigger a broadcast command, for instance “start scanning inputs” or “started a new AC”. The cycle is optionally completed with a SEND end-of-cycle slot (Slot END), that marks the completion of the AC and can cancel some of the errors that caused certain slots to extend. Slots S1 and SN permit transactions that are multiples of one tick (θ) in length. Alternatively, the tick can be suppressed by expressing the interval explicitly, in microseconds for example. For simplification, the choice of equal slots can also be selected [31]. The AC period (t_{CA}) is computed according to Equation (1).

$$t_{CA} = t_{SYNC} + t_{S1} + \ldots\ldots + t_{SN} + t_{SEND} \quad (1)$$

The ModbusE communication protocol is designed to expand the classic Modbus protocol specification for the next goals:

- Transform the Modbus communication protocol [32] in a completely defined protocol by defining a flexible AC and description language for network devices;
- Implementing deterministic behavior by deploying the AC and ensuring temporal coherence of information when specific decisions need to be taken on a set of data generated by distributed systems and apps;
- Increased performance upon new microcontroller manufacturing technologies and improved data flow based on improved communication rates, character lengths and microcontroller implementations;
- ModbusE protocol structure is modeled on the ISO/OSI pattern for 3 levels, level 1, 2 and 7.

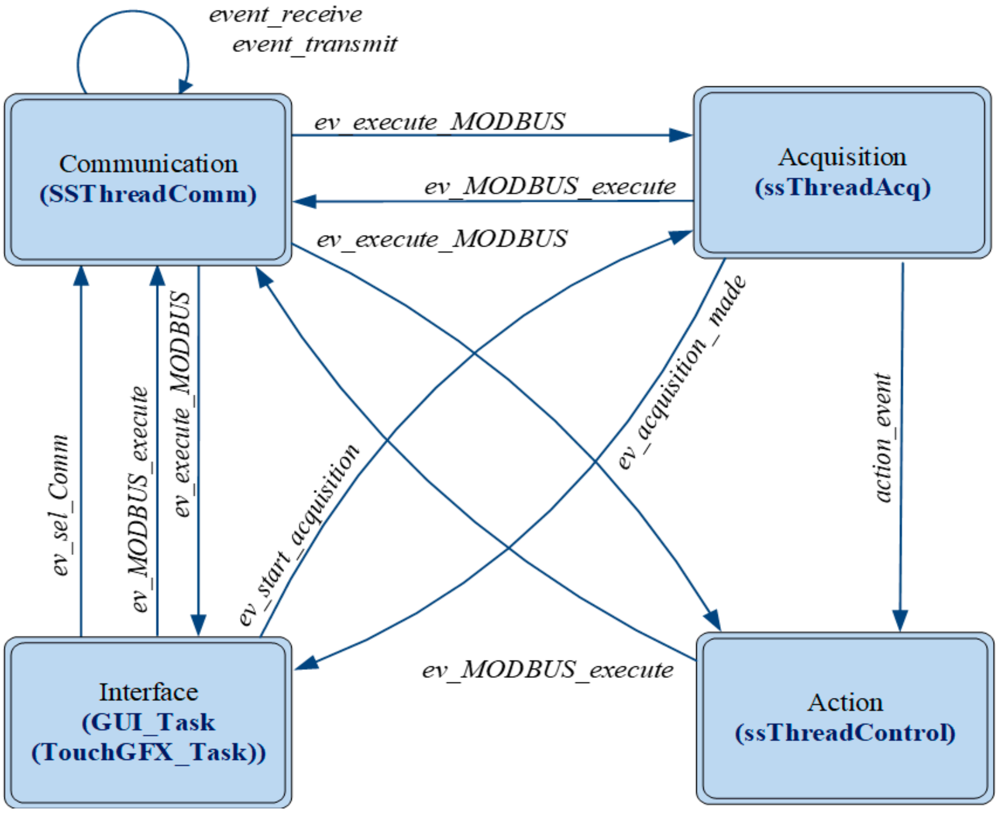


Figure 3. Tasks and events associated with the software tasks.

Slave stations that support ModbusE protocol will benefit from all the advantages of ModbusE, whereas slaves using Modbus_RTU, if it can support the communication rates, can work in normal Modbus_RTU communication mode. The communication speed for the 32 slots is 10.5 Mbps. If the gateMBEx is interfaced with an OPC UA application system that supports AMQP (Advanced Message Queuing Protocol) and supports Modbus_RTU and/or TCP-IP drivers [33,34], then the gateMBEx may be simply incorporated within an industrial IoT architecture. A higher level of integration may be obtained either by exploiting the new multi-core MCUs, in Cortex-Mx and Cortex-Ax schemes or other specialized architecture, by deploying the OPC UA server on the gateMBEx itself (gate_MBEx_UA). Server module called MBE has the following general characteristics:

- Deployment of Modbus RTU communication to RTU server terminals;
- Adding and defining new features and capabilities to the ModbusE at the MBE_RTU server unit, fully compliant with the Modbus RTU;
- The server-MBE module includes 100% compatibility with the RTU server module;
- A client module with a VirtualComm USB connection for PC connection and testing of operation with Modbus-specific applications will also be implemented for testing.

4. Formal Modeling of ModbusE Protocol

4.1. MODBUS Server

The information exchange is initiated between Modbus client–server devices as soon as the client station sends the server a command to exchange data, execute a service or perform one of the many potential functions. Once the Modbus server receives the incoming message, it performs the command and/or collects the necessary information. The system response time is constrained by two main limiting parameters, namely the period of time taken by the client to issue the query/response and the ability of the server to respond in a certain time range. For example, in ModbusPlus, either service is always available at the specific Modbus point and the client or server function is handled by a symbol passing scheme. A given device can support a Modbus client service, a Modbus server service or a combination of both, according to the needs of the device and, in some cases, the specific Modbus level. Depending on the lower level of communication, a client may trigger Modbus messaging queries to one or many servers, and a server may respond to queries from one or many clients. Such capabilities will be discussed in more detail later when the various transport mechanisms and underlying layers are discussed. A representative HMI or SCADA application implements a client service to trigger communication with PLCs and other devices to obtain data. Typically, a PLC deploys both the client service and the server service so that it can start communicating with other PLCs and I/O devices, i.e., it can respond to requests from other PLCs, SCADA applications, HMIs or other devices. The differentiation among the different instances is currently not important, since the Modbus application protocol is the identical in all scenarios. An I/O device performs a server service so that other peripherals can read and write I/O data. Since this I/O peripheral does not have to trigger the communication, it does not perform a client service.

The function of the Modbus server, according to [35], is to allow access to application objects and services to Modbus clients with remote access. Depending on the user application different types of access can be provided, namely:

- Simple access, such as setting and accessing attributes of application objects;
- Advanced access for the purpose of initializing application-specific services.
- The Modbus server must implement the following functions:
- Assigning application objects to Modbus objects that can be read and written in order to set or retrieve attributes of application objects;
- Provide a way to trigger services on application objects.

During execution, the Modbus server has to process the incoming Modbus query, to analyze the required operation and to return a Modbus message. Commonly used Modbus data types include Discrete, Coil, Input Register and Holding Register. The record and tab Modbus data types are more rarely used and are supported by only a limited number of function codes. From the point of view of an application user (client), the Modbus record data type consists of a set of registers, distinguished by the address of the first register and its register number. Within the context of this specification, the associated registers have also been referred to as references.

The distinctions between inputs, outputs and data elements addressable at the bit or word level do not require any special implementation behavior. It is entirely permissible and usual to assume that all four tables overlap, if this is the most reasonable and natural behavior on the system in question. Discrete registers, coils, input and holding registers are often referred to as data references or data items. The situations encountered in practice offer some typical, but not completely exhaustive, interpretations when utilizing real tables, namely separated and overlapping memory tables. For an individual selection of 65,536 data items for each of the above-mentioned data item types, the communication protocol allows the respective 65,536 data items to be individually selected, and the operations to read or write these data items depend on the function code. The significance of the contents of any data reference depends entirely on whether the addresses of the client/server data references realized in the client/server service function codes are unsigned numbers greater than or equal to zero. The potential association of client/server data references, such as bits or registers, and the actual storage or logical meaning within devices is a particular local concern.

Client/server service identifiers are usually referred to as function codes (FC). FC is the encoding of the services invoked by a server. Some function codes are additionally characterized by a subcode, expressed as part of the data field. Such codes are divided into three groups, and since the subdivision may extend to subcodes even though they are part of the data field, they will be mentioned here. Officially assigned function codes are either assigned to a regular service or reserved for future assignment. The Modbus FCs typically assigned to a standard service and recommended for the implementation proposed in this paper are Read Holding Registers FC 03 (0×03), Write Multiple Registers FC 16 (0×10), Read/write multiple registers FC 23 (0×17) and Read Device Identification FC 43/14 (0×2B/0×0E). The use of these FCs allows the most efficient communication between Common Industrial Protocols (CIP) systems and a Modbus device. User-defined function codes can also be used for research in a user-defined controlled lab configuration, so they should not be used in an open system environment. FCs, usually assigned to a standard service, will be dealt with in another research paper. There are two ranges for this purpose: FC 65 (0×41)–FC 72 (0×48) and FC 100 (0×64)–FC 110 (0×6E). The function codes publicly assigned to a standard service are detailed extensively in the Modbus specification [35]. The Modbus protocol does not constrain the existence of a specific subset of these FCs, so a Modbus implementation can have any subset. Dedicated FCs are currently used by several companies for their proprietary products and are not freely available for public access. In terms of transmission order and data representation at the physical layer, Modbus uses a big-endian ordering convention for both addresses and data elements. This implies that when more than one byte of information is transferred, the most significant byte is transmitted first so that for example if we consider a 16-bit register of size 0×1234, the first byte sent is 0×12 and the next is 0×34. Note that the Remote Terminal Unit (RTU) cyclic redundancy check (CRC) expects first CRC low and then CRC high, which is the last byte of the frame in the RTU frame message.

The information provided in this chapter does not include all the particularities of the Modbus protocol, the licensed specifications are publicly available at the address given in [35]. In the following, we describe the Modbus protocol on the serial line, i.e., how the Modbus protocol is implemented at OSI level 7, i.e., at levels 2 and 1, respectively, level 1 being a serial line (EIA/TIA-485 (RS485) or EIA/TIA-232 E (RS232)). Addressing a server by a client is carried out completely using the unit ID. The network is a serial bus with only one client at a time, defined by the configuration. There can be up to 247 servers in the network, each unit requiring a unique ID between 1 and 247. RS232 is a point-to-point network where the unit ID, although not required, is still requested. For the client's finite state machine (FSM), an inappropriate response is one with a frame error and is therefore discarded. However, for the server's FSM, the mismatched request is either a frame error or a request that is not addressed to the server; in either case, the request is dropped. Messaging is accomplished by changing APDUs, but is supplemented with error checking.

4.2. Physical Level—ModbusE

Compatibility with ModbusRTU is maintained, so the following aspects can be stated:

- A data byte is always, when working with Modbus RTU compatibility, 11 bits long (one start bit, 8 data bits, one parity bit and one stop bit, or two stop bits without parity);
- Extended Modbus also uses characters of 9 (multiprocessor mode (MM)), 16, 17 (MM), 32, 33 (MM) with or without 8-bit CRC, 64, 645 (MM) with or without 8-bit CRC;
- Similar to Modbus RTU, frames have to be distinguished from each other with a silence interval of at least 3.5 characters;
- If there are more than 1.5 chars separating two sequential characters, the frame must be treated as incomplete and discarded;
- Typical communication speeds are in the range of 9.6 kbps up to 115.2 kbps Modbus RTU and up to 27 MBps ModbusE;

- Communication can be reported to UART transmit/receive indicators:
    a. Pooling;
    b. Interrupts;
    c. Multiprocessor mode interrupts, for ModbusE only;
    d. Interrupts and DMA (mandatory for ModbusE at high speeds: 1 Mbps);
- RS485 line standard is used, possibly CAN.

Facilities brought by new generations of microcontrollers embedded in UART/USART circuits that can be considered are the following:

- Increased communication speed (Mbps or even above 27 Mbps);
- Introduction of multi-microprocessor mode by using a ninth bit (feature inherited from the I8051 CPU family);
- Recognition of an 8-bit address in multi-microprocessor mode;
- DMA transfer between UART and microcontroller memory;
- Automatic direction control for RS485 line drivers, possibly with timers on change of communication direction;
- Timeout on receive (good for 1.5 and 3.5 character duration measurement);
- CRC computation.

4.3. Data Link Level—ModbusE

Modbus messages will be categorized in the protocol extension as SDO (Service Data Object) messages, classic Modbus messages and PDO (Process Data Object) messages, respectively. PDO messages have a simple structure and comprise the following elements:

- Slot address in the AC;
- Message data;
- CRC checksum.

The characteristics of ModbusE communication messages at the data link level are the following:

- The message length, at the station level, is fixed and cannot exceed 256 bytes;
- The message structure must be constructed using Modbus register mapping commands in PDOs (Figure 4);
- Each station may use one PDO for transmit and one PDO for receive, or a single PDO for both operations;
- A station may have multiple PDOs;
- Stations may subscribe to PDOs transmitted over the network;
- There may be stations that only subscribe without filling a PDO slot.
- ModbusE addresses have the following characteristics:
- Slots can also be addressed with a multi-microprocessor character if this communication mode is used;
- Only the first character in the slot is MM if this communication mode is used.
- The slot address always has bit 7 set to 0;
- Addresses with bit 8 set to 1 are SDO addresses. Optionally only these multi multiprocessor addresses can be used to communicate Modbus messages with classical structure;
- Address 0 corresponds to the SYNC slot and is broadcast and may or may not be followed by other characters;
- 0×70 through 0×7f are reserved addresses;
- Optionally, address 0×70 denotes a slot for SDOs;
- As an option, the slot address can be extended by an additional byte and can also be complemented with CRC8;
- ModbusE station addresses must have values between 0×81 and 0×ef, addresses 0×80, 0×f0—0×ff being reserved.
- A station is able to possess:
    a. SDO and PDO address;

b. Physical address;

c. With the exception of the 8th bit, the rest of the address bits may be identical.

- A PDO message may exceptionally be preceded by a 4-byte timestamp.

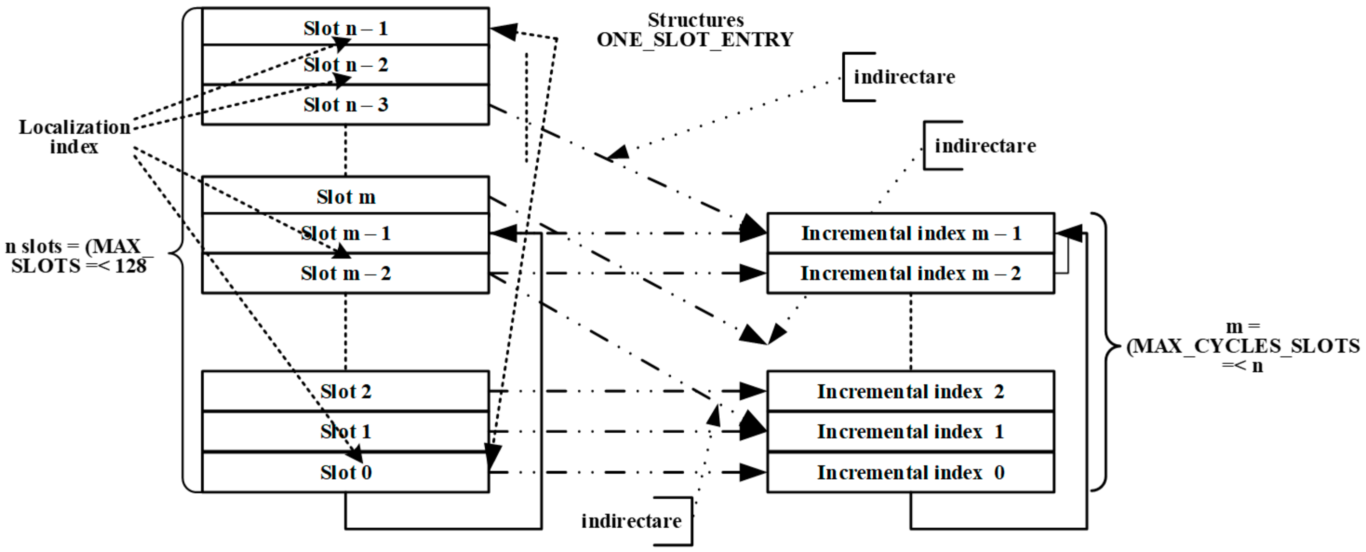


Figure 4. Slots and ModbusE acquisition cycle with the maximum number of slots and localization index.

4.4. Application Level

At least three commands must be used to deploy the ModbusE communication protocol, namely:

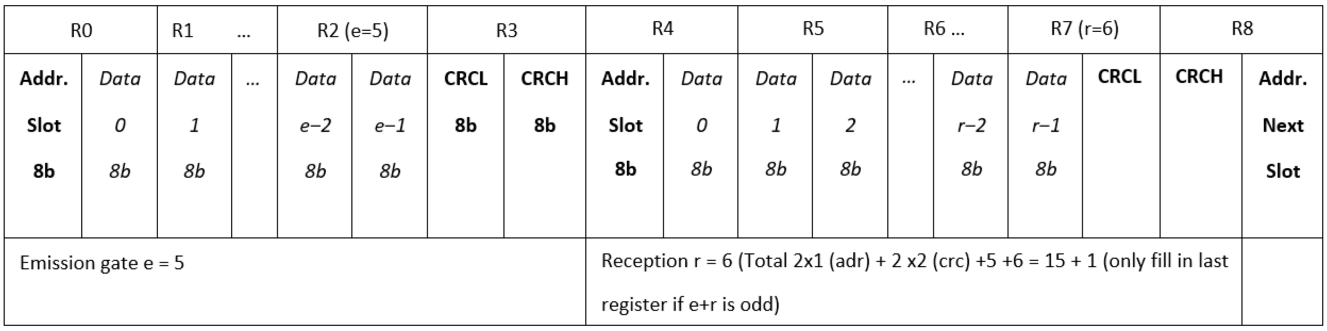
- PDO mapping command transmit/receive—100;
- Transmit/Receive PDO Read Command—101;
- PDO configuration command—102, physical, slot and classical Modbus command addresses;
- If the mapping addresses are 0, the command returns the current mapping.

The slot-configuration algorithm can be run on the Base Station Gateway (BSG) or on a host computer. The PDOs are built and executed based on the mappings by executing classical Modbus functions at the application level, and the data structures in the gateway are useful to define the behavior of the MBE server stations (slaves in the classical Modbus). Figure 4 shows how to consider the slots in an AC, so the following important statements can be made:

- It can be observed that the state of a slot can be either IN_CYCLE OUT_OF_CYCLE;
- Slots in IN_CYCLE cycle have incremental index (Figure 4);
- The AC evolves from 0 to the maximum number of slots in the cycle minus 1 (m − 1), after which it returns to 0, etc;
- The maximum number of slots in cycle can be less than or at most equal to the total number of slots that the access gate can support, which number in turn must be less than the maximum number of possible slots, i.e., (128);
- The slots outside the slots in the cycle are called OUT_OF_CYCLE (with localization index ($\geq$m) and m < 128) and can send the corresponding information for execution by indirection (Figure 4);
- Commands for IN_CYCLE slots (PDO type commands) are considered synchronous compared to OUT_OF_CYCLE slots which are considered asynchronous (SDO type commands). The latter are transmitted on the asynchronous slot (last in the cycle) only once. The asynchronous slot may also have synchronous behavior with the specification that it must have at least one idle state at the AC level;
- The slots (with address < 0×80) only support Modbus functions (3, 17 and 23) on the data in the transmit and receive buffers (which must cover a continuous memory area).

If the memory space is of odd length, the last byte not belonging to this space shall be taken into account when reading the entire space;

- The registers may also be read from another address, but the length must not exceed the length of the data area 6 + e + r, adjusted by +1 if e + r is odd (Figure 5);
- Both transmit and receive messages have a similar structure (Figure 6);
- The slot address can also be used as a classical Modbus address;
- OUT_OF_CYCLE slots >= 0×80 are considered classical Modbus commands that implement most Modbus functions. They are executed as asynchronous commands, which can be queued in a circular queue on the last slot in the cycle (with respect to its parameters).



R0		R1	...	R2 (e=5)		R3		R4		R5		R6 ...		R7 (r=6)		R8	
Addr. Slot 8b	*Data 0 8b*	*Data 1 8b*	...	*Data e−2 8b*	*Data e−1 8b*	**CRCL 8b**	**CRCH 8b**	**Addr. Slot 8b**	*Data 0 8b*	*Data 1 8b*	*Data 2 8b*	...	*Data r−2 8b*	*Data r−1 8b*	**CRCL**	**CRCH**	**Addr. Next Slot**
Emission gate e = 5								Reception r = 6 (Total 2x1 (adr) + 2 x2 (crc) +5 +6 = 15 + 1 (only fill in last register if e+r is odd)									

Figure 5. The data area of a slot starts with R0 and has length 6 (address +crc + e + r).

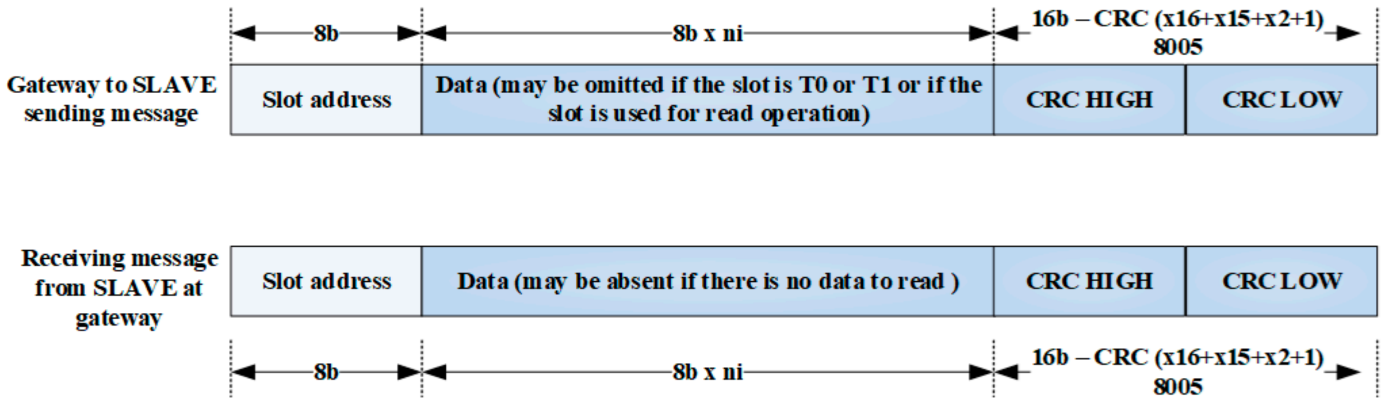

8b
8b x ni
16b – CRC (x16+x15+x2+1) 8005
Gateway to SLAVE sending message
Slot address
Data (may be omitted if the slot is T0 or T1 or if the slot is used for read operation)
CRC HIGH
CRC LOW
Receiving message from SLAVE at gateway
Slot address
Data (may be absent if there is no data to read)
CRC HIGH
CRC LOW
8b
8b x ni
16b – CRC (x16+x15+x2+1) 8005

Figure 6. The message structure within a slot (ModbusExtension—ModbusE, slots T0 and T1 do not send read messages to the gateway and neither do the empty slots).

Slot-level highlights for the ModbusE deployment are presented in Figure 7. For compatibility with the original implementation, the additional position 1 has been inserted. Time moment 1 indicates the handler duration for the interrupt generated by the USART that says it has finished transmitting the last character. This handler also prepares, if necessary (without the 0, 1 slots), reception of the response. Additional numbers from 2 to 4 for SLAVE have been inserted next, for the same reason, despite temporarily being placed before the 5, the position on the figure being important.

Time moments 2, 3 and 4 indicate the CRC calculation time for the message received by the slave for the current slot. To this time shall be added the preparation for sending the response. From Figure 7 one can see the degree of overlap of operations during a slot, as shown in Figure 8. Time moment 5 indicates the time at which the slot finishes. Table 1 shows the times measured with the implementation of the ModbusE protocol for the AC intervals on slave and master, the results obtained are performing well in terms of ModbusE AC performance, as well as other implementations proposed in the literature. With tsCRC we noted the CRC computing time for the message received by the slave for the current slot (to this time is added the preparation for sending the reply) and tmCRC represents the CRC computing time for the message received in the previous slot at master. On the slave, tsmove indicates the time period for the slave to move the message from the receiving buffer to the final buffer and tsthd the jitter while the received message processing task is active. At the master station the parameter tmfosli indicates when the master terminates the operations related to the old slot and tscommSi, respectively, tmcommi is the time period of sending the message through the DMA by the master to the slave. tmswitchi is the time interval of switching to the mbeThreadCycleRTU(thd) task that handles the slot which is the highest priority in the system.

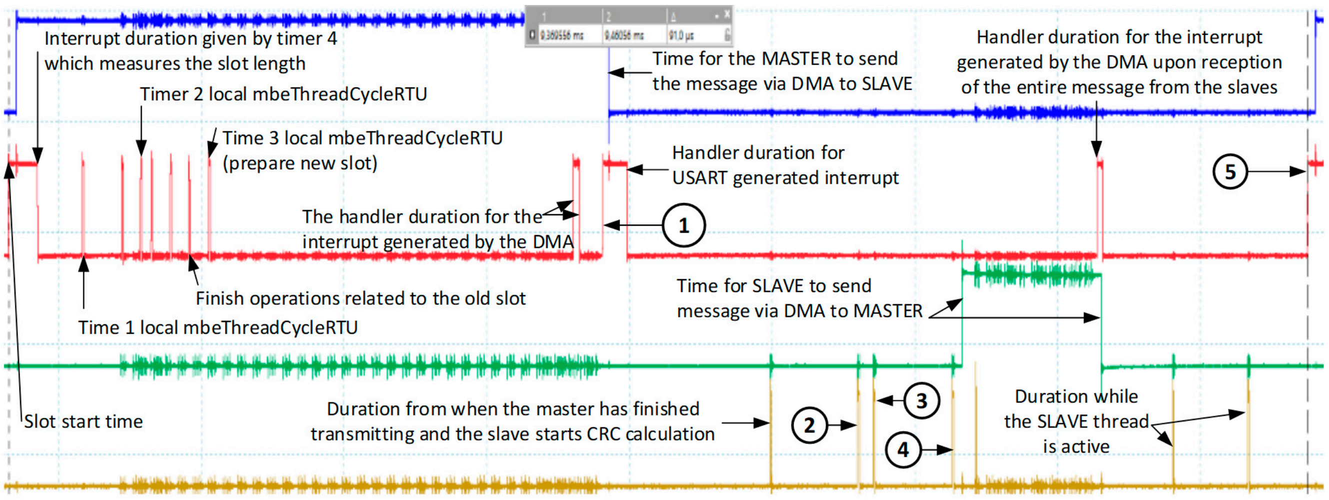


Figure 7. Important events for the communication periods in a tSi slot based on ModbusE implementation.

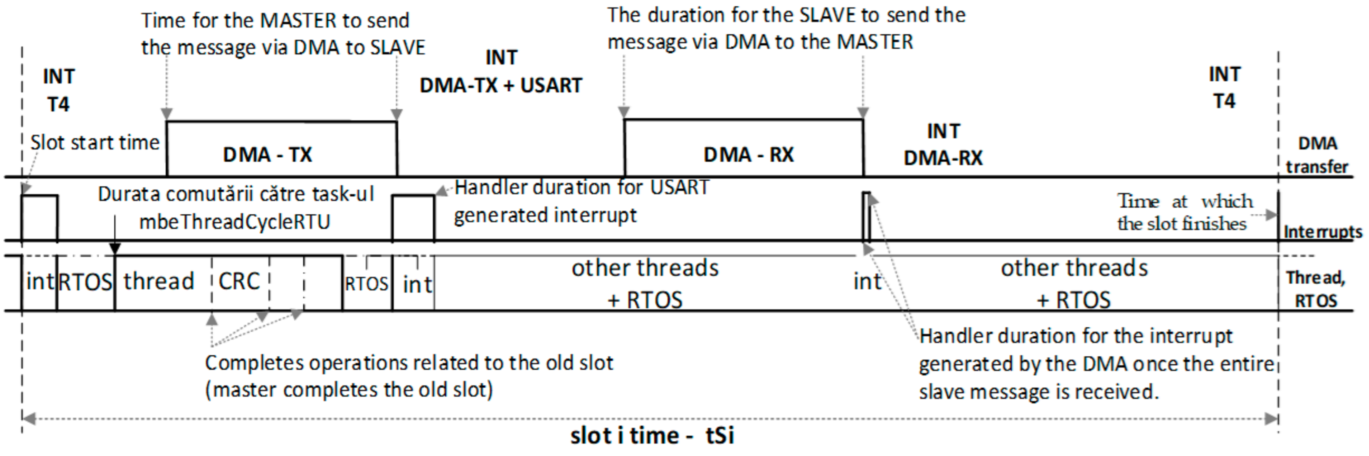

Time for the MASTER to send the message via DMA to SLAVE
The duration for the SLAVE to send the message via DMA to the MASTER
INT T4
INT DMA-TX + USART
INT T4
Slot start time
DMA - TX
DMA - RX
INT DMA-RX
DMA transfer
Durata comutării către task-ul mbeThreadCycleRTU
Handler duration for USART generated interrupt
Time at which the slot finishes
Interrupts
int
RTOS
thread
CRC
RTOS
int
other threads + RTOS
int
other threads + RTOS
Thread, RTOS
Completes operations related to the old slot (master completes the old slot)
Handler duration for the interrupt generated by the DMA once the entire slave message is received.
slot i time - tSi

Figure 8. Operations within a slot in ModbusE client–server communication.

The conclusions of the authors' findings in accordance with the ModbusE refer to the fact that in a given slot only data and checksum are transferred, maximizing the bandwidth utilization of the communication channel, since headers are no longer transferred. Data significance is specified using configuration or classical Modbus commands in device initialization. Also based on ModbusE, data-acquisition devices can be implemented modularly using digital I/O, analog inputs such as voltage, current, 4–20 mA unified units, RTD, resistance and analog outputs. Such modules are controlled via BSGs, having a predefined AC. This allows adding a timestamp according to the acquisition period or a timestamp according to when the data is read from BSG. Together with the BSG, these modules are suitable for utilization in any industry sector where data from different sensors needs to be monitored. Interfacing the BSG to a PC or similar industrial workstation is accomplished via the TCP/IP Modbus protocol.

Table 1. Measured times for slave and master AC slots using ModbusE protocol.

tsCRC (µs)	tmCRC (µs)	tsmove (µs)	tmfosli (µs)	tsthd (µs)	tscommSi (µs)	tmcommi (µs)	tmswitchi (µs)
0.982	0	0.780	2.565	8.092	0	3.721	2.998
1.189	0	0.881	2.058	8.677	0	4.792	3.101
2.775	0	3.083	1.41	14.316	5.594	17.37	3.101
5.09	1.24	5.419	3.326	19.837	9.778	34.05	3.101
11.54	2.03	10.57	4.059	30.302	18.1	67.54	3.157
11.19	3.383	10.53	5.52583	30.122	34.93	67.66	3.157
41.32	5.788	140.5	8.156	92.676	268.5	267.4	3.165
11.61	43.09	10.5	43.35	30.22	68.35	67.66	3.061
11.72	11.57	10.66	13.33	30.392	26.16	67.60	3.142
1.321	0	1.057	14.88	8.633	0	4.879	3.177

5. Smart Switch Class B Laboratory Model Design Methodology, Experimental Results and GUI Testing

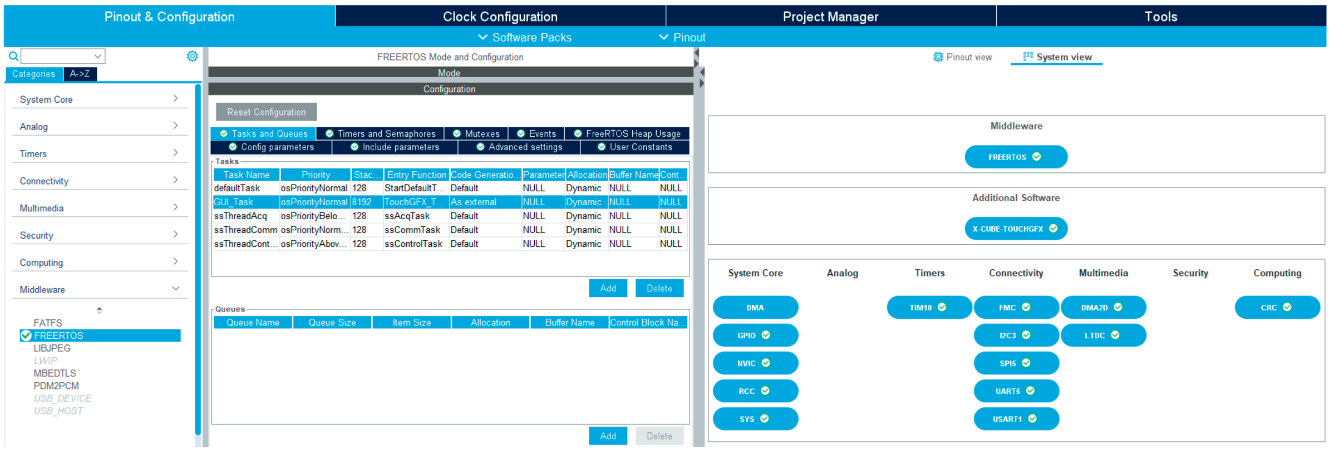
5.1. Smart Switch Class B Laboratory Model Design Methodology and SW-HW Validation

The DIGI-TOUCH Class B laboratory model consists of the STM32F429I-DISCO kit [36] with STM32F429ZIT microcontroller, LCD-Touch and JTAG, RS485 module (LTM2881) for ModbusE RTU communication, temperature and humidity sensor (HTU21D click MIKROE module), buzzer, relay output (RELAY click MIKROE module) or solid-state relay (SSR), and WiFi communication (ESP8266). The microcontroller pins used are configured as follows: PF6 is used for PWM output (TIM10: sConfigOC.OCMode = TIM_OCMODE_PWM1); PA13—buzzer output; PG3—relay output; PC1—test access/alarm output; PA0—user key input (set/reset relay/alarm test); PD11—fire alarm key input; PA8—SCL HTU21D (I2C3), PC9—SDA HTU21D (I2C3); PD2—RX (UART5) connected to TX—ESP8266 and PC12—TX (UART5) connected to RX—ESP8266; PA9—TX (USART1); and PA10—RX (USART1) connected to ModbusE RTU.

Figure 9 illustrates part of the steps performed to configure FreeRTOS, the tasks created and the system architectural view using the STM32F429I-DISCO development kit. The embedded system and user interface was designed with STM32CubeMX 6.7.0, TouchGFX 4.21.1 Designer and STM32CubeIDE 1.11.2. Listing 1 shows the interaction handling logic corresponding to the Main screen for an alarm event associated with the displayed graphics. Thus, the screens added via the TouchGFX environment are Splash, Main, Settings, Functions and Info. In the class "class SettingsView: public SettingsViewBase" the necessary code for controlling the displayed graphical elements was created, and the designed interactions (arrow_right.setAlpha(255); arrow_right.invalidate()) are mainly based on user actions and hardware events triggered from asynchronous digital inputs. The pulse-width modulation (PWM) signal for variable light intensity adjustment is controlled both remotely or using the slider displayed on the Main screen (pwm_value = slider1.getValue()). Figure 10 illustrates the user interface designed in TouchGFX for testing the laboratory model. The smart switch with optical and acoustic signaling for emergencies is represented by the DIGI-TOUCH embedded system, with a rectangular form for easy integration into the building structure (Figure 1a), consisting in this version of an on/off button and a sliding button displayed on the touchscreen, respectively, a relay or SSR. The proposed device is an embedded system based on a high-performance microcontroller from the Cortex-M4 family. GUI has an intuitive menu, a button for quick access to system functions, ModbusE communication for reading and writing certain internal parameters. The system emits acoustic signals, with the possibility to display evacuation direction information, in case a fire signal is received from the Gateway, as well as environmental information.

Listing 1. GUI interaction handling in TouchGFX 4.21.1 Designer and STM32CubeIDE 1.11.2.

```
void MainViewBase::handleTickEvent()
{
if (interaction4Counter > 0)
{
interaction4Counter--;
if (interaction4Counter == 0)
{//Interaction5;//When Interaction4 completed fade arrow_right;
//Fade arrow_right to alpha:0 with CubicIn easing in 100 ms (6 Ticks)
arrow_right.clearFadeAnimationEndedAction();
arrow_right.setFadeAnimationDelay(6);
arrow_right.startFadeAnimation(0, 6, touchgfx::EasingEquations::cubicEaseIn);
}
}
```



Tasks

Task Name	Priority	Stac...	Entry Function	Code Generatio...	Parameter	Allocation	Buffer Name	Cont...
defaultTask	osPriorityNormal	128	StartDefaultT...	Default	NULL	Dynamic	NULL	NULL
GUI_Task	osPriorityNormal	8192	TouchGFX_T...	As external	NULL	Dynamic	NULL	NULL
ssThreadAcq	osPriorityBelo...	128	ssAcqTask	Default	NULL	Dynamic	NULL	NULL
ssThreadComm	osPriorityNorm...	128	ssCommTask	Default	NULL	Dynamic	NULL	NULL
ssThreadCont...	osPriorityAbov...	128	ssControlTask	Default	NULL	Dynamic	NULL	NULL

Queues

Queue Name	Queue Size	Item Size	Allocation	Buffer Name	Control Block Na...

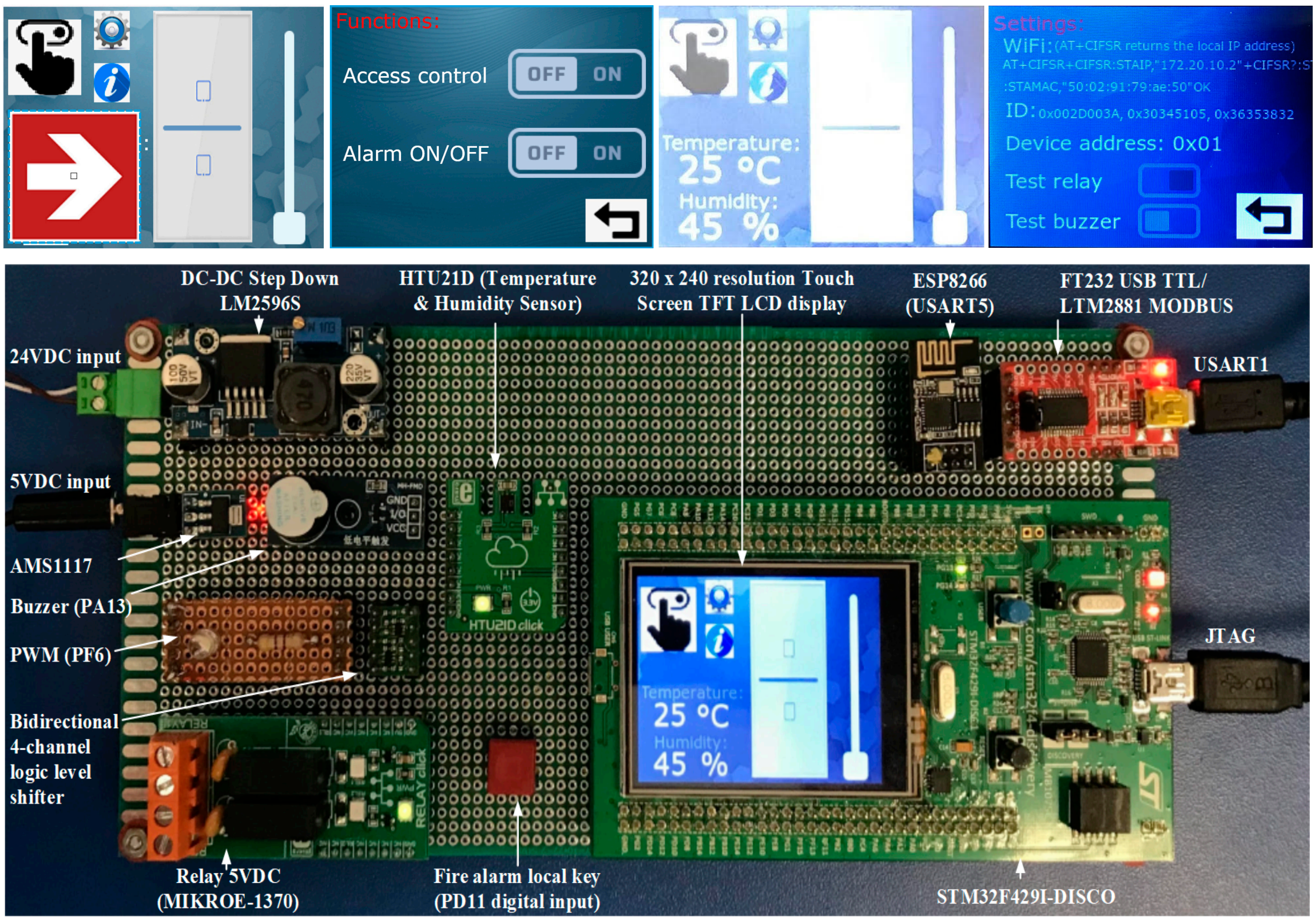
Figure 9. FreeRTOS configuration, tasks and an architectural view of the system using the STM32F429I-DISCO development kit and STM32CubeMX 6.7.0.

5.2. Experimental Results

The smart switch consists of a temperature and humidity sensor connected to I2C3 (HAL_I2C_Mem_Read_IT(&hi2c3, HTU21D_Adress, HTU21D_Temp_Cmd, I2C_MEMADD_SIZE_8BIT, HTU21D_RX_Data, 3)), a wireless communication interface ("AT+CWMODE=1" to select WIFI mode as station mode, "AT+CIPMUX=1" settings allowing multiple connections, AT+CIFSR for get a local IP address (172.20.10.2), "AT+CIPSERVER=1,8080" to configure device as server with 8080 port number), an on/off relay module, a variable light intensity-control module controlled with PWM and a building-automation network-specific power module. Device ID represent the STM32 factory-programmed UUID memory (#define STM32_UUID ((uint32_t *)0×1FFF7A10)).

The DIGI-TOUCH also has an optional integrated connector used for programming and debugging the designed microcontroller device. The smart switch consists of the following architectural and constituent components: microcontroller STM32F429ZIT based on ARM instruction set and Harvard architecture, touch display, power module, environmental sensors, acoustic warning, digital output on/off, variable light intensity-control module, RS485 module for ModbusE RTU communication and wireless communication. FreeRTOS was chosen to be used in this project because it is built with an accent on reliability and ease of use.

In terms of a comparison with most existing systems that use only wireless communication, this paper presents the experimental laboratory system that was used to test the 2 problems approached, namely, AC and ModbusE. Slave terminals can operate in multiprocessor mode for serial communication, with a DMA controller that can be attached directly to the serial port. Upon identification of the slot attached to the slave station, data read and/or write transactions are automatically carried out by the DMA controller, in the ModbusE protocol the data block sizes are known.



```
WiFi:(AT+CIFSR returns the local IP address)
AT+CIFSR+CIFSR:STAIP,"172.20.10.2"+CIFSR?:S
:STAMAC,"50:02:91:79:ae:50"OK
ID:0x002D003A, 0x30345105, 0x36353832
Device address: 0x01
```

Figure 10. Smart switch GUI with the following screens: main (view from the TouchGFX design environment and real view on the lab model), functions and settings (IP: 172.20.10.2; MAC: 50:02:91:79:ae:50; device ID: 0×002D003A, 0×30345105, 0×36353832) and laboratory model for the DIGI-TOUCH-embedded device (STM32F429I-DISCO), AMS1117 voltage step-down module, LM2596S, FT232 USB TTL/LTM2881, HTU21D, bidirectional 4-channel logic level shifter (Pololu), ESP8266, PF6 PWM output, relay 5VDC (MIKROE-1370), buzzer (PA13 digital output), fire alarm local key (PD11 digital input).

5.3. Discussion

Fieldbus networks have evolved over a relatively long period of time from the first tentative industrial networks to complex, highly advanced, highly specialized automation networks. However, there is sufficient potential for further evolution and from a technological point of view, the communication environment allows continuous innovation. The most potentially prospective research horizon for technology advancement is incompletely defined and wireless protocols, so that the advantages are underpinned by high versatility as well as the lack of expensive, fault-prone cables. The findings have important contributions to innovations in industrial automation networking, as increasingly distributed services or installations may require the use of heterogeneous networks of wired and wireless communication devices. Today, fieldbus systems comprising networks and communication devices can also be considered as network-integrated systems. Simpler fieldbus communication protocols reflect the current situation in typical automation applications. It should be noted that not all fieldbus devices comprise only a physical layer, a datalink layer and an application layer. Initially, fieldbus systems were not intended to be very sophisticated, so the network and transport layers, which include end-to-end routing and control functions, are not necessary. Also, the session and presentation layers are not fully implemented, although some features in layers 3–6 may be needed in a limited form, so it may be necessary to consider the specific message encoding scheme that is more appropriate for the limited resources typically available in fieldbus nodes. In such instances, these functions are often incorporated in layer 2 or 7. There are a few examples in which other layers have been specifically defined, and for the IEC 61158 standard [37], layer 3 and 4 functions can be placed in either layer 2 or layer 7., while layer 5 and 6 functionalities are always covered in layer 7. In this context, in building automation the scenario is changed because, given the possible large number of nodes, fieldbus systems have to support a hierarchically structured network, so a three-layer implementation is not feasible. Specifically, the Building Automation and Control Network (BACnet) utilizes the network layer, which is particularly critical given that this protocol was built at a higher layer to operate over various lower layer protocols and links such as Ethernet, Master Slave/Token Passing (MS/TP) and LonTalk. The European Installation Bus (EIB) and KNX, an abbreviation for Konnex, also utilize, network and transport layers to implement hierarchical network routing along with connection-oriented and connectionless end-to-end communication functions, so for such a heterogeneous design approach, a uniform network layer is fundamental. The LAN (LonWorks) is closer to a LAN than to a high-performance fieldbus. The most advanced protocol structure in the fieldbus field is proposed by LonWorks. Although currently mainly used in building automation, the protocol was actually conceived as a general control network without a dedicated scope. Within the LonTalk protocol, all seven OSI layers are well-defined, although layer 6 is quite minimal in functionality, and the specific features of layer 3 support a variety of addressing schemes and enhanced routing capabilities. Market pressures and the demand for flexibility have forced the eventual inclusion of higher layer protocols, and the increasing capability of network controller hardware has facilitated their implementation. In this context, CAN is a perfect example, as a plethora of protocols such as CANopen, Smart Distributed System (SDS) and DeviceNet have emerged in the native CAN layers 1 and 2. The application layer, which provides abstract functionality to real applications, requires substantial software implementation time, which can negatively influence the runtime and protocol cost for fieldbus interfaces; therefore, in some cases, such as Interbus, Process Field Bus—Decentralized Peripherals/Process Automation (PROFIBUS-DP/PA) or Controller Area Network (CAN), the application layer was initially omitted. Among the fieldbus systems mainly used in industrial process automation, ControlNet and P-NET distinguish themselves by also implementing layers 3 and 4. A remarkable characteristic of the P-NET design is its capability for multi-network layouts, in which so-called multi-port clients can attach multiple segments to any arbitrary structure.

In terms of security coverage, smart switch type A implementations are based on MQTT. This is due to the fact that MQTT is recognized as among the top IoT protocols because of its characteristics and specific capabilities adapted to the particular requirements of IoT applications. MQTT also supports Transport Layer Security (TLS) and Secure Sockets Layer (SSL) encryption, ensuring data confidentiality during transmission. MQTT provides authentication and authorization mechanisms through username/password credentials or client certificates, protecting network access and resources. Regarding other technologies and protocols used in smart building systems, LoRaWAN is known for its ability to provide long-distance communications, often far beyond that of traditional wireless networks. Data rates in LoRaWAN typically range from 0.3 kbps to 50 kbps. The disadvantages of LoRaWAN are given by the limitation for large payloads, bidirectional management and latency as these times can be significant depending on the type of sensor and can limit certain applications. BACnet is also a communication protocol designed for intelligent buildings, so it should be noted that BACnet/IP uses the UDP protocol for data transmission, with a client–server communication approach, where devices typically act as servers, and the default port is 47808.

6. Conclusions

The significant technological progress that has been demonstrated in the Smart Home is also linked to the evolution of Industry 4.0, symbolizing the emergence of the "Fourth Industrial Revolution", the actual trend in industriay. This mainly involves technologies such as CPS, IoT and Cloud Computing, which connect the virtual space with the physical world. The scientific and technical contributions are related to the proposal and use of the ModbusE extension within the smart switch, as well as the implementation DIGI-TOUCH design with innovative functions within a BIoT system. Thus, it is possible to highlight the architectural elements of the embedded system component realized in this project, for which a patent application has also been filed by the authors. The proposed digital device integrates ModbusE communication for reading/writing certain internal parameters.

Thus, using the ModbusE type communication bus, the smart switch will be able to emit acoustic signals and will be able to display evacuation indications in case a fire signal is received from the alarm-control panel. Due to additional operational costs, today there is a need for an alternative means of reducing energy consumption due to smart devices and beyond. However, a proportion of global energy generation is still taken up by renewable energy sources, with hydroelectric power being the most popular, followed by wind and then solar. Despite advances in microelectronics, the typical efficiency of a solar panel, considering the ratio of sunlight to electricity, is still low. Therefore, to maximize the energy produced by the solar panel, it needs to be used with other electrical devices to help it maintain a reasonably constant power output throughout the day.

The proposed smart switch has been developed as a laboratory model mainly considering the research on ModbusE server. In terms of modularity of the hardware and source code as well as user friendly GUI, the proposed embedded system can be easily adapted and integrated for larger buildings or smart city deployments. For future research, an interface to indicate other metrics such as energy savings or system latency and reliability in different operational cases will be considered. User-centered features such as gesture recognition and voice commands for people with disabilities are implemented on the mobile device and the communication with the smart switch, is realized over WiFi or ModbusE. In the laboratory research phase, tests were conducted to examine the ease of use of the system from the end-users' perspective, especially if they had accessibility issues, both visually and functionally. The present tests were mainly focused on the ModbusE communication protocol. In terms of scalability, the code is a very modular and flexible system, with the possibility to indicate on the LCD display the number of buttons set by the parameters transmitted through the Modbus communication. For the tests corresponding to the final prototype, a real world evaluation of the system will be considered in order to make it more efficient and user friendly. As an optional feature the system can indicate how much energy has been consumed and, respectively, a prediction algorithm of user behavior.

7. Patents

The proposed smart switch and ModbusE server are based on patent proposal A/00224 (OSIM, 2023).

Author Contributions: Conceptualization, I.Z. and V.G.G.; methodology, I.Z. and V.G.G.; software, I.Z. and V.G.G.; validation, I.Z. and V.G.G.; formal analysis, I.Z. and V.G.G.; investigation, I.Z. and V.G.G.; resources, I.Z. and V.G.G.; data curation, I.Z. and V.G.G.; writing—original draft preparation, I.Z. and V.G.G.; writing—review and editing, I.Z. and V.G.G.; visualization, I.Z. and V.G.G.; supervision, V.G.G.; project administration, I.Z.; funding acquisition, I.Z. and V.G.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work is also supported by the NetZeRoCities Competence Center, funded by European Union–NextGenerationEU and Romanian Government, under the National Recovery and Resilience Plan for Romania, contract no. 760007/30.12.2022 with the Romanian Ministry of Research, Innovation and Digitalisation through the specific research project P2-Sustainable energy & environments.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the editor and the anonymous reviewers for reviewing our manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Leccisi, M.; Leccese, F.; Moretti, F.; Blaso, L.; Brutti, A.; Gozo, N. An IoT Application for Industry 4.0: A New and Efficient Public Lighting Management Model. In Proceedings of the 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, Roma, Italy, 3–5 June 2020; pp. 669–673. [CrossRef]
2. Khan, I.; Zedadra, O.; Guerrieri, A.; Spezzano, G. Occupancy Prediction in IoT-Enabled Smart Buildings: Technologies, Methods, and Future Directions. *Sensors* **2024**, *24*, 3276. [CrossRef] [PubMed]
3. Havard, N.; McGrath, S.; Flanagan, C.; MacNamee, C. Smart Building Based on Internet of Things Technology. In Proceedings of the 2018 12th International Conference on Sensing Technology (ICST), Limerick, Ireland, 4–6 December 2018; pp. 278–281. [CrossRef]
4. Still, L.; Oispuu, M.; Koch, W. Optimal Sensor Placement for Shooter Localization within a Surveillance Area. In Proceedings of the 2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), Karlsruhe, Germany, 23–25 September 2021; pp. 1–6. [CrossRef]
5. Belloni, E.; Massaccesi, A.; Moscatiello, C.; Martirano, L. Implementation of a New Solar-Powered Street Lighting System: Optimization and Technical-Economic Analysis Using Artificial Intelligence. *IEEE Access* **2024**, *12*, 46657–46667. [CrossRef]
6. Burgio, A.; Cimmino, D.; Dolatabadi, M.; Jasinski, M.; Leonowicz, Z.; Siano, P. Virtual energy storage system for peak shaving and power balancing the generation of a MW photovoltaic plant. *J. Energy Storage* **2023**, *71*, 108204. [CrossRef]
7. Zagan, I.; Gaitan, V.G.; Petrariu, A.-I.; Brezulianu, A. Healthcare IoT m-GreenCARDIO Remote Cardiac Monitoring System—Concept, Theory of Operation and Implementation. *Adv. Electr. Comput. Eng.* **2017**, *17*, 23–30. [CrossRef]
8. Paiva, S.; Amaral, A.; Pereira, T.; Barreto, L. Conceptual Architecture for an Inclusive and Real-Time Solution for Parking Assistance. In *Smart Energy for Smart Transport. CSUM 2022*; Nathanail, E.G., Gavanas, N., Adamos, G., Eds.; Lecture Notes in Intelligent Transportation and, Infrastructure; Springer: Cham, Switzerland, 2023. [CrossRef]
9. Li, L.; Zhou, Z.; Hu, H.; Ning, H.; Zhang, J. Design of Power Environment Monitoring System for Central Computer Room Based on Internet of Things. In Proceedings of the 2024 Asia-Pacific Conference on Software Engineering, Social Network Analysis and Intelligent Computing (SSAIC), New Delhi, India, 10–12 January 2024; pp. 1–5. [CrossRef]
10. Song, E.Y.; FitzPatrick, G.J.; Lee, K.B.; Gopstein, A.M.; Boynton, P.A. Interoperability testbed for smart sensors in smart grids. In Proceedings of the 2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), Washington, DC, USA, 19–22 February 2018; pp. 1–5. [CrossRef]
11. Zungeru, A.M.; Gaboitaolelwe, J.; Diarra, B.; Chuma, J.M.; Ang, L.M.; Kolobe, L.; David, M.; Zibani, I. A Secured Smart Home Switching System based on Wireless Communications and Self-Energy Harvesting. *IEEE Access* **2019**, *7*, 25063–25085. [CrossRef]
12. Lun, Y.Z.; Rinaldi, C.; D'iNnocenzo, A.; Santucci, F. Co-Designing Wireless Networked Control Systems on IEEE 802.15.4-Based Links Under Wi-Fi Interference. *IEEE Access* **2024**, *12*, 71157–71183. [CrossRef]
13. Gozuoglu, A.; Ozgonenel, O.; Gezegin, C. Design and Implementation of Controller Boards to Monitor and Control Home Appliances for Future Smart Homes. *IEEE Trans. Ind. Inform.* **2024**, *20*, 11458–11465. [CrossRef]
14. You, M. Research on the Management of Smart Home Control System Based on ZigBee. In Proceedings of the 2023 International Conference on Applied Physics and Computing (ICAPC), Ottawa, ON, Canada, 27–29 December 2023; pp. 207–211. [CrossRef]
15. Draganova-Zlateva, I.; Kolev, V. Smart Lighting in the House. In Proceedings of the 2020 12th Electrical Engineering Faculty Conference (BulEF), Varna, Bulgaria, 9–12 September 2020; pp. 1–3. [CrossRef]
16. Gokhale, P.; Deshpande, S.; Page, S.; Bagul, M.; Bundele, A. Street Light Monitoring and Controlling System Using RaspberryPi and Zigbee Protocol. In Proceedings of the 2023 7th International Conference on Computing, Communication, Control and Automation (ICCUBEA), Pune, India, 18–19 August 2023; pp. 1–4. [CrossRef]
17. Available online: https://new.abb.com/low-voltage/products/wiringaccessories/remote-control/zigbee-light-link (accessed on 25 July 2023).
18. Du, Y.; Tan, Y.; Lim, Y. RF-Switch: A Novel Wireless Controller in Smart Home. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Taichung, Taiwan, 19–21 May 2018; pp. 1–2. [CrossRef]
19. Makhanya, S.P.; Dogo, E.M.; Nwulu, N.I.; Damisa, U. A Smart Switch Control System Using ESP8266 Wi-Fi Module Integrated with an Android Application. In Proceedings of the 2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 12–14 August 2019; pp. 125–128. [CrossRef]
20. Xie, J.; Tang, H.; Huang, T.; Yu, F.R.; Xie, R.; Liu, J.; Liu, Y. A Survey of Blockchain Technology Applied to Smart Cities: Research Issues and Challenges. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2794–2830. [CrossRef]

21. Efendi, A.; Siswanto, A.; Sudarman, A. Application Control and Monitoring of Light Usage in Smart Home Environment. In Proceedings of the 2018 Third International Conference on Informatics and Computing (ICIC), Palembang, Indonesia, 17–18 October 2018; pp. 1–5. [CrossRef]
22. Vivek, P.S.; Rahul, P.V.S.; Dyuthy, E.; Yadav, S. Arduino based Smart System for Control and Effective Billing. In Proceedings of the 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), Coimbatore, India, 25–27 March 2021; pp. 1706–1709. [CrossRef]
23. Ramachandran, S.S.; Sivaraman, K.; Veeraraghavan, A.K.; Yuvaraj, R.; Azhagumurgan, R. Design and development of smart branched switch for home automation systems. In Proceedings of the 2017 International Conference On Smart Technologies for Smart Nation (SmartTechCon), Bengaluru, India, 17–19 August 2017; pp. 622–625. [CrossRef]
24. Okorie, P.U.; Ibraim, A.A.; Auwal, D. Design and Implementation of an Arduino Based Smart Home. In Proceedings of the 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 26–27 June 2020; pp. 1–6. [CrossRef]
25. Hwang, Z.; Uhm, Y.; Kim, Y.; Kim, G.; Park, S. Development of LED smart switch with light-weight middleware for location-aware services in smart home. *IEEE Trans. Consum. Electron.* **2010**, *56*, 1395–1402. [CrossRef]
26. Reddy, V.M.; Vinay, N.; Pokharna, T.; Jha, S.S.K. Internet of Things enabled smart switch. In Proceedings of the 2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN), Hyderabad, India, 21–23 July 2016; pp. 1–4. [CrossRef]
27. Jian, M.-S.; Wu, J.-Y.; Chen, J.-Y.; Li, Y.-J.; Wang, Y.-C.; Xu, H.-Y. IOT base smart home appliances by using Cloud Intelligent Tetris Switch. In Proceedings of the 2017 19th International Conference on Advanced Communication Technology (ICACT), PyeongChang, Republic of Korea, 19–22 February 2017; pp. 589–592. [CrossRef]
28. Yiu, J. *System-on-Chip Design with Arm® Cortex®-M Processors, Reference Book*; Arm Education Media: Cambridge, UK, 2019; ISBN 978-1-911531-19-7.
29. Available online: https://www.freertos.org/ (accessed on 12 January 2024).
30. Available online: https://www.freertos.org/FreeRTOS-Plus/FreeRTOS_Plus_POSIX/index.html (accessed on 12 January 2024).
31. Gaitan, V.G.; Zagan, I. Modbus Protocol Performance Analysis in a Variable Configuration of the Physical Fieldbus Architecture. *IEEE Access* **2022**, *10*, 123942–123955. [CrossRef]
32. de Brito, I.B.; de Sousa, R.T. Development of an Open-Source Testbed Based on the Modbus Protocol for Cybersecurity Analysis of Nuclear Power Plants. *Appl. Sci.* **2022**, *12*, 7942. [CrossRef]
33. Reyes, N.A.; Cerrato, H.I. Modbus TCP Bridging for Interconnecting Non-Compatible Devices in the Energy Sector Using Node-RED and Edge Computing. In Proceedings of the 2023 IEEE 41st Central America and Panama Convention (CONCAPAN XLI), Tegucigalpa, Honduras, 8–10 November 2023; pp. 1–4. [CrossRef]
34. Zahran, B.; Abu Zahra, F. IT/OT Convergence Protocols: DNP3, Ethernet/IP, and Modbus. In Proceedings of the 2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE), Las Vegas, NV, USA, 24–27 July 2023; pp. 1743–1748. [CrossRef]
35. Available online: http://www.modbus.org (accessed on 11 July 2023).
36. Available online: https://www.st.com/en/evaluation-tools/32f429idiscovery.html (accessed on 24 April 2024).
37. Felser, M.; Sauter, T. The fieldbus war: History or short break between battles? In Proceedings of the 4th IEEE International Workshop on Factory Communication Systems, Vasteras, Sweden, 28–30 August 2002; pp. 73–80. [CrossRef]