Research, Society and Development, v. 11, n. 10, e345111032542, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i10.32542

Development and Implementation of Hardware to trigger I/O of a PLC via Wireless Network

Desenvolvimento e Implementação de Hardware para Acionamento de I/O de um CLP via Rede Sem Fio

Desarrollo e Implementación de Hardware para Disparo de E/S de un PLC vía Red Inalámbrica

Received: 07/05/2022 | Reviewed: 07/23/2022 | Accept: 07/25/2022 | Published: 08/03/2022

Felipe Naressi Grandinetti
ORCID: https://orcid.org/0000-0002-6964-685X
Universidade Federal de Itajubá, Brazil
E-mail: felipe.ng.unifei@gmail.com
Luiz Felipe Pugliese
ORCID: https://orcid.org/0000-0003-1185-0863
Universidade Federal de Itajubá, Brazil
E-mail: pugliese@unifei.edu.br
Rodrigo Aparecido da Silva Braga
ORCID: https://orcid.org/0000-0002-5490-9944
Universidade Federal de Itajubá, Brazil
E-mail: rodrigobraga@unifei.edu.br
Tiago Gaiba de Oliveira
ORCID: https://orcid.org/0000-0001-5195-4179
Universidade Federal de Itajubá, Brazil
E-mail: tgaiba@unifei.edu.br
Diogo Leonardo Ferreira da Silva
ORCID: https://orcid.org/0000-0002-6038-3600
Universidade Federal de Itajubá, Brazil
E-mail: diogoleonardof@unifei.edu.br
Fadul Ferrari Rodor
ORCID: https://orcid.org/0000-0003-0591-2247
Universidade Federal de Itajubá, Brazil
E-mail: fadulrodor@unifei.edu.br

Abstract
This work aims to use ESP32 and ESP8266 microprocessors with built-in Wi-Fi modules to activate the digital inputs of a PLC via the Web, in addition to sending the command signal via wireless network to the respective actuators of the system. A local wireless network is created to allow control of inputs via a browser on a cell phone or computers, such as allowing the exchange of information between the controller and its actuators. To test the functionality of the proposed prototype, a three-phase motor was started using the browser of a cell phone connected to the ESP32 Wi-Fi network.
Keywords: Wireless network; Programmable logic controllers; Embedded systems; Arduino; Internet of things.

Resumo
O presente trabalho tem o objetivo de utilizar microprocessadores ESP32 e ESP8266 com módulo Wi-Fi embutido para efetuar o acionamento das entradas digitais de um CLP via Web, além de enviar o sinal de comando via rede sem fio para os respectivos atuadores do sistema. Uma rede sem fio local é criada para possibilitar o controle das entradas via browser no celular ou computador, como permitir a troca de informação entre o controlador e seus atuadores. A fim de testar a funcionalidade do equipamento proposto, foi efetuada a partida de um motor trifásico utilizando o navegador de um celular conectado à rede Wi-Fi do ESP32.
Palavras-chave: Rede sem fio; Controlador lógico programável; Sistemas embarcados; Arduino; Internet das coisas.

Resumen
El presente trabajo tiene como objetivo utilizar microprocesadores ESP32 y ESP8266 con módulo Wi-Fi incorporado para activar las entradas digitales de un PLC vía Web, además de enviar la señal de comando vía red inalámbrica a los respectivos actuadores del sistema. Se crea una red inalámbrica local para permitir el control de entradas vía navegador en el celular o computadora, además de permitir el intercambio de información entre el controlador y sus

actuadores. Para probar la funcionalidad del equipo propuesto se puso en marcha un motor trifásico utilizando el navegador de un celular conectado a la red wifi ESP32.
Palabras clave: Red inalámbrica; Controlador lógico programable; Sistemas embebidos; Arduino; Internet de las cosas.

1. Introduction

Technological and industrial advances provide a significant quest to improve industrial processes by reducing operating costs, greater safety for equipment and operators, and improving the efficiency of a process (Sacomano, et al., 2018). Currently, there is a methodology for digitalizing industrial processes called Industry 4.0, which makes this search even more incisive as it provides an integration that extends from the factory floor to the upper levels of the automation pyramid (Hermann, et al., 2016; Souza, et al., 2017; Santos, et al., 2020).

The data transmission carried out by cables is the simplest form of communication. However, its cost can increase due to the insertion of new clients into the network. Wireless networks are an alternative due to the flexibility of installation and operation. Therefore, there is the possibility of expansion without the difficulties of installing the project (Morimoto, 2008).

The term IoT (Internet of Things) can be treated as a concept that refers to the digital interconnection of equipment with the internet through a wireless network (Singh & Singh, 2015; Camargos, et al., 2022). In this context, the low-cost feasibility of a wireless system to drive the inputs and outputs of a PLC (Programmable Logic Controller) can provide an improvement in the industrial environment.

Using Wi-Fi technology and its security protocols, it is possible to make this service cheaper, making it even more robust in terms of controlling all equipment in a process. The connectivity between these devices provides better supervision of the process, allowing for more effective decision-making, both at an operational and managerial level (Cavalcante & Almeida, 2018; Almeida, 2019).

Despite having a higher energy consumption, Wi-Fi technology is superior in terms of communication as it uses the IEEE 802.11 protocol and operates in frequency bands of 2.4 GHz or 5 GHz (Lee, et al., 2007). Thus, it becomes possible to create local wireless networks with high transmission rates, depending only on the distance from the access point. Furthermore, it is an alternative for the communication between PLC and the actuators of a process. For such implementation, there is a need to use hardware compatible with embedded systems technology such as ESP32 and ESP8266 microcontrollers, for example (Almeida, et al., 2017).

In embedded systems, it is necessary to create an HMI (Human Machine Interface) so that the operator can operate clearly. One way to get a user interface is through web browsers, that is, a web page encoded in HTML (Hyper Text Markup Language). However, HTML elements alone cannot perform functions such as sending or receiving data. For this, it is necessary to have a program that can manipulate and process data referring to the server in web pages, such as PHP, ASP, Java, Ruby, Python, and ColdFusion, among others (Silva, 2010).

The IoT is already part of many people's lives through residential technologies (Sacomano, et al., 2018; Lima, et al., 2020). In the industrial sphere, there are already projects that show a transition to industry 4.0 concerning the activation of motors via wireless networks (Buoro, 2013; Silva, 2015; Rodrigues, et al. 2022). Given this scenario, the proposal of this work is the development of low-cost plug-and-play hardware that aims to provide the integration between shop floor elements with embedded systems using a Wi-Fi network.

In addition to the remote control of a PLC's inputs with a web page's visual interface, the output signal's transmission to their respective actuators is also performed. To exemplify the operation of the proposed hardware, the wireless energization of contactors that make a star-delta start of a three-phase motor will be presented.

The work is organized as follows: Section 2 will deal with the hardware components used in the prototype's

construction. Section 3 will describe the software elements used to develop the project schedule. Section 4 will present the architecture and construction of the prototype hardware and all the development and results of this work. Finally, the conclusions of the work are presented.

2. Methodology

Based on the methodological procedures presented by Köche (2016), this work has qualitative and quantitative characteristics. The development and implementation of the system illustrated in this work allows the application of concepts related to Industry 4.0 and the Internet of Things in disciplines related to industrial automation content and programming. For the development of the system, the four steps of the action research method were followed: plan, act, describe, and evaluate. Action research is a methodology for conducting applied research-oriented toward making diagnoses, identifying problems, and finding solutions (Thiollent, 1988; Tripp, 2005; Pugliese, et al., 2022).

As industrial technology advances, understanding PLC (Programmable Logic Controller) concepts, programming languages, and industrial communication devices are of utmost importance for a control and automation engineering student. For this reason, education in industrial automation has become a significant issue in universities' most diverse interdisciplinary engineering departments (Özerdem, 2016). Didactic benches and laboratory devices are essential tools for teaching, where theoretical content is consolidated with practice, especially in technology-oriented courses (Pinho, et al., 2021).

In this work, a device to aid in the learning of industrial automation and IoT and Industry 4.0 concepts is developed. The device, in addition to assisting in teaching industrial automation, covers advanced concepts of the current industry, allowing the activation of machines remotely, without the need for physical cabling. The device also provides a method to improve learning outcomes. It will enable the development of projects in the laboratory, allowing the student to acquire real-time skills and experiences in engineering education, such as electrical, electronics, mechatronics, and control engineering.

2.1 Hardware Elements

To fulfill the objective of this work, which is to develop a prototype for experimental validation, some electronic components were used in its creation. The LM2596 converter will be responsible for powering the boards, reducing the voltage from a range of 10 to 40 VDC to the voltage suitable for the microcontrollers, which is 5 VDC.

The ESP32 is a MCU (Multipoint Control Unit), which will be the main microcontroller of the system in which the server will be implemented and is responsible for performing both the human-machine interface and data transmission to the auxiliary microprocessor. ESP32 has 15 GPIO ports that can be safely used as digital input or output.

The ESP8266 will connect to the same Wi-Fi network and oversee requesting data from the outputs. This, in turn, will replicate the signal in relays that make it possible to energize loads and contractors. ESP8266 has 9 GPIO pins. However, only six can be used as digital inputs/outputs since the other 3 are for serial communication.

A relay module 8 channels will be responsible for receiving signals from operator commands and energizing the PLC input port. Other two modules (4 channels and 2 channels) will connect with the actuator elements. Resistances of 5.1 kΩ and 30 kΩ will be used, which will be part of a voltage divider circuit to adapt the voltage available in the PLC to the microcontrollers used in developing the prototype. The Allen Bradley SLC 500 PLC will be used for the application and validation of the prototype. It is worth mentioning that the PLC is not part of the prototype but is necessary for carrying out experimental tests.

The average price of each component that is part of the prototype is shown in Table 1.

Table 1 - Average price of the main components of the prototype.

Component	Unit price (R$)
NodeMCU ESP32	40,00
NodeMCU ESP8266	25,00
LM2596	12,00
Relay Module 8 channels	40,00
Relay Module 4 channels	22,00
Relay Module 2 channels	12,00
Resistance of 5.1 kΩ	0,10
Resistance of 30 kΩ	0,10

Source: Authors.

2.2 Software and Programming Elements

Two software will be used to implement the embedded system: the Arduino IDE and Visual Studio Code. Both will be important because they will provide all the integration between the hardware and software elements of the project, allowing the correct functioning of the final prototype.

2.2.1 Arduino IDE Setup

First, downloading the Arduino IDE (Integrated Development Environment) is necessary for the installation of the MCUs functions by the Arduino board manager. Installation takes place by opening the "Preferences" option (after initializing a blank Arduino program) and adding the URLs:

- http://dl.espressif.com/dl/package_esp32_index.json
- https://arduino.esp8266.com/stable/package_esp8266com_index.json

As shown in (Kurniawan, 2019), the installation is performed in the "Tools" tab of the main menu.

In both MCUs, it is possible to create their own Wi-Fi network; for that, additional libraries are necessary for communication between the boards. Among the libraries used in this work, there are "ESP8266WiFi.h", "WiFi.h", and "ESPAsyncWebServer.h". The programming language used is C++ at this stage of the work.

The Arduino Wi-Fi Shield libraries "ESP8266WiFi.h" and "WiFi.h" make it possible to connect a board compatible with the Arduino platform to the internet. This connection allows the card to be configured as a server or client and connect to other networks (Monk, 2013; Monk, 2015).

The "ESPAsyncWebServer.h" library has additional functions that make the communication between client and server asynchronous. Furthermore, it can offer the method of data transmission via HTTP, whose purpose is to allow the exchange of information between client and server in a standardized way (Kurniawan, 2019).

Except for the latter, all libraries are already included in the Arduino IDE. To install a library in the IDE, it is necessary to download the .zip file referring to the desired library, and in the compiler menu follow the steps: "Sketch"> Include library > add .zip library and select the downloaded file, so the program will include the library automatically.

An additional library "ArduinoOTA.h," must also be installed to use ESP32's Over the Air (OTA) function. Through it, it is possible to reprogram the MCU without the need to connect a cable, just being on the same network. In this way, initialization functions are created that, through a network port, allow the transfer of the code via Wi-Fi.

2.2.2 Visual Studio Code Setup

Visual Studio Code is an open-source code editor and compiler with several available languages. For this work, it will be necessary to install the NodeJS software (available at https://nodejs.org/en/), an application platform used to interpret and

compile the codes written in Javascript. Once installed, it will be possible to code in Visual Studio both the backend and the frontend of the interface.

To aid in the development, a standard project will be created to explain the necessary dependencies of the work. To do so, it is required to open the program's terminal, which can be opened using the shortcut "Ctrl + Shift + ' ", and type the command "npm create-react-app Project". This execution will create a standard "Project" application using the React framework.

After that, you must install the aWOT Scripts tool using the "npm install awot-scripts --save-dev" command, which will automatically generate the backend project according to the frontend settings. You must also install a button library called React Toggle Buttons, using the command "npm install react-toggle-button –save", which contains a class of toggle buttons to facilitate later implementation.

You need to generate the previously mentioned backend configuration file. For this, the command "npx awot-create backserver" will be used, which creates the base files for later implementation, including the .ino sketch with an example of coding a button and the functions necessary to work with the frontend.

The "package.json" file must be changed as described in Appendix C (available at https://tinyurl.com/bn9w9az4) to add the "npm run dist" command to the project. The purpose of this command is to translate the project files from the build folder to static files from the server in the "StaticFiles.h" file in the "backserver" directory.

Another essential configuration that must be performed is the server proxy, as shown in Figure 13, which must contain the IP assigned to ESP32 on the network. As in this case it is the access portal, the default address "192.168.1.4" will be kept.

3. Results and Discussion

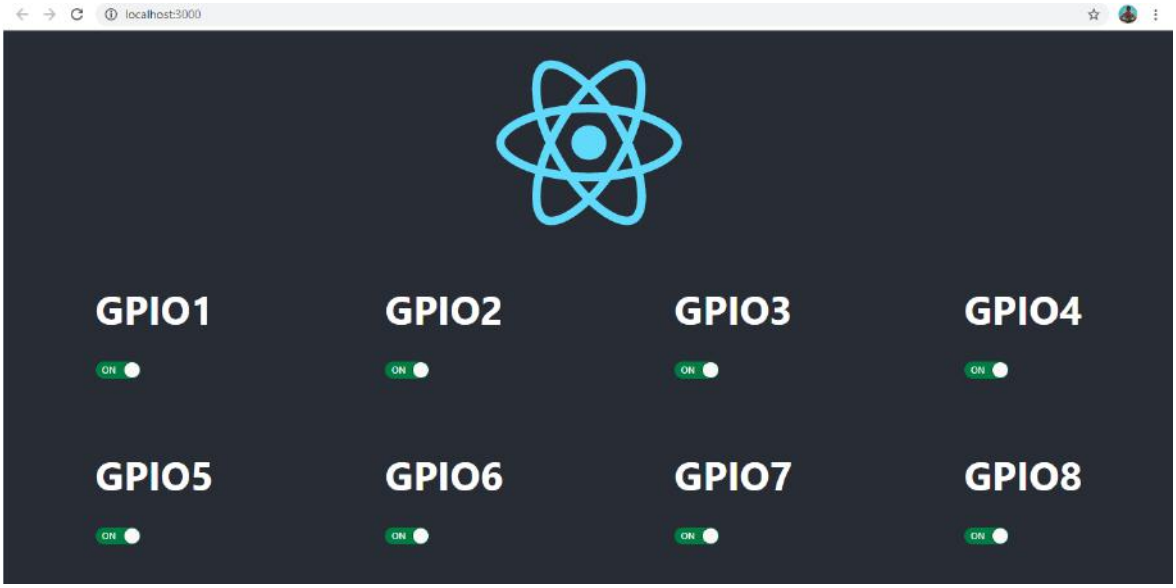
3.1 Prototype development

The prototype will be compatible, in the correct dimensions for fitting, with the bench available in the automation laboratory of the Unifei campus in Itabira. For energization, as a 24 VDC output is already available on the PLC bench, the LM2596 step-down CI will be used, it performs the voltage reduction to 5V. It was also defined that with this prototype, it will be possible to control 8 inputs and 6 outputs of a PLC.

Using the Visual Studio Code software, it is possible to implement the communication interface with the user (browser) in Javascript and generate the configuration file responsible for the direct connection of the page's visual elements with the internal configuration of an MCU. First, the browser interface will be coded by changing the "App.js" and "button.js" files as shown in Appendix D and E (available at https://tinyurl.com/bn9w9az4), respectively. With these codes, you have a page considered responsive. That is, it adapts to the size of the screen that is making your request. In addition, it contains 8 toggle buttons, which you must click on to toggle its state. After changing the files, it is necessary to regenerate the static files and create a new build, using the "dist" and "build" commands, respectively.

The entire directory of the build folder, together with the generated "StaticFiles.h" file, must be loaded into the MCU through the "Tools > ESP32 Sketch Data Upload" option in the Arduino IDE. Thus, the page in Figure 1 can be accessed in ESP32.

Figure 1 - Interface to control the PLC inputs.

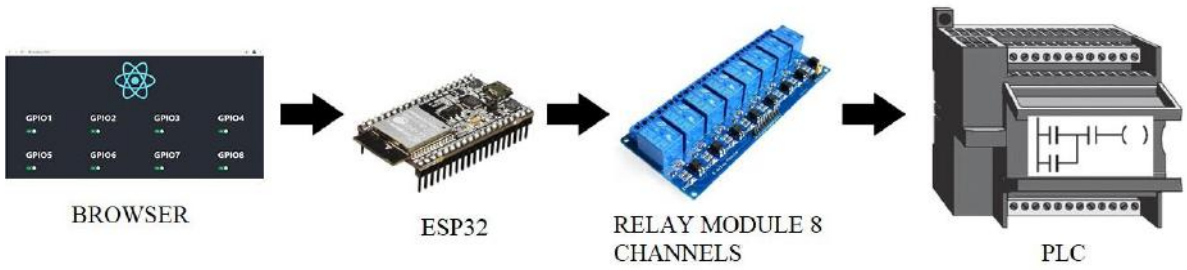


Source: Authors.

So that the interface can be accessed correctly, ESP32 will be responsible for creating its Wi-Fi network through the WiFi.softAP ("name", "password") function. This function allows the network to be created automatically from the desired name and password for access. Once started, add the "WiFi.h" library already built into the Arduino IDE to use it.

There is also a need to configure the pins that are the digital outputs, that is, the pins responsible for sending the signals indirectly to the PLC inputs and those that receive the signals from the outputs. In addition, OTA functions are implemented so that it is not necessary to connect a cable to reprogram the ESP32. It is essential to upload the code as shown in Appendix A (available at https://tinyurl.com/bn9w9az4), so a Wi-Fi network with the name "NEWCLP" and password "12345678" will be created when the board is powered up, the server will receive the commands from the buttons updating the outputs correctly. The pins configured as the ESP32 digital output will be controlled and must be connected to relays directly connected to the PLC inputs, enabling wireless activation through the browser, as shown in Figure 2.

Figure 2 - Example circuit with ESP32, Relays and PLC.

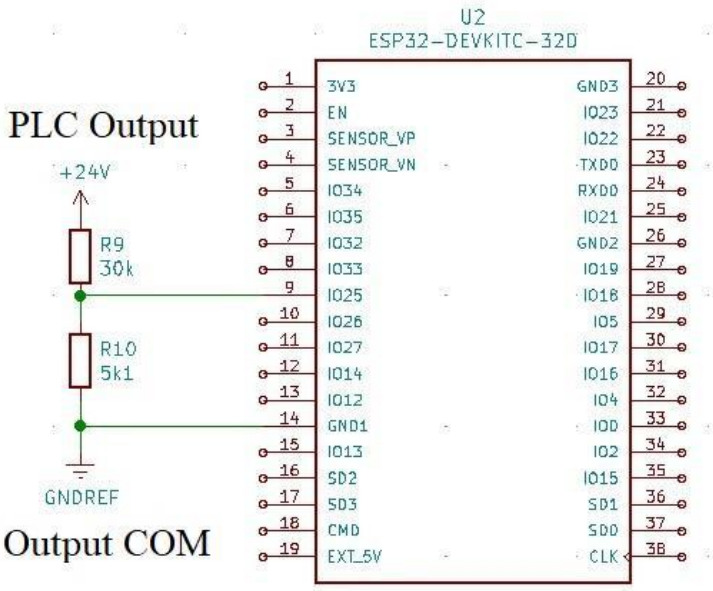


Source: Authors.

The PLC has a program related to the process logic, and its outputs will be activated according to the energization of the internal coils directly associated with its physical outputs. However, instead of having a physical connection between the output of the PLC and the actuator element, it is possible to connect the digital inputs of the ESP32 itself, for example, configured as a transmitter, and for this it is necessary to convert the output voltage 24 VDC to 3.3 VDC. A voltage divider is

designed to fulfill this objective, since the current required to implement a high logic level on the pin can be very low, and this is the cheapest way to perform the voltage signal attenuation, as illustrated by the schematic of Figure 3 referring to a PLC output and an ESP32 input.

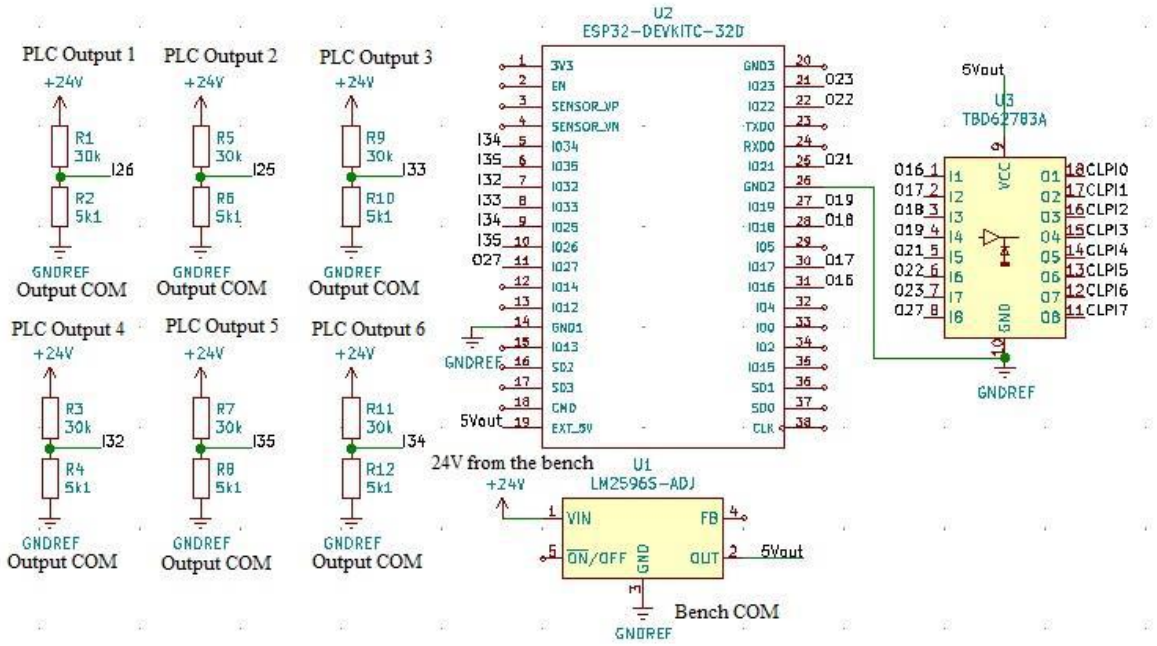
Figure 3 - Connection between PLC output and ESP32.



Source: Authors.

The ESP32 connection schematic with the PLC can be seen in Figure 4, in which the relay module was represented by the U3 block because in the program used to make the schematic the module does not exist.

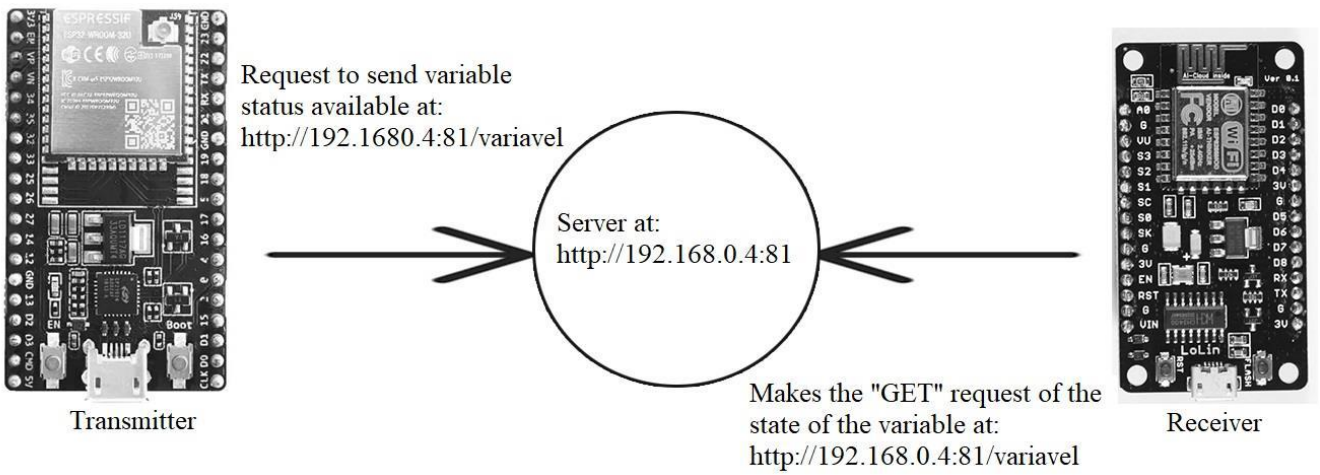
Figure 4 - ESP32 and PLC connection diagram.



Source: Authors.

The external library called "ESPAsyncWebServer.h" must be used to enable the creation of an asynchronous process on a port separate from the main network, which exchanges data through HTTP methods that return the Request and Response variables. The Request variable requests the sending or receiving of information to the server, while the Response variable shows the server's response to the request, confirming whether it was successful. The transmitter continuously sends the desired information, i.e., the current state of its ports. A variable is created in a process on the server, accessed at http://ipdoservidor:porta/variavel, as shown in Figure 5.

Figure 5 - Simplified communication scheme between server and client.



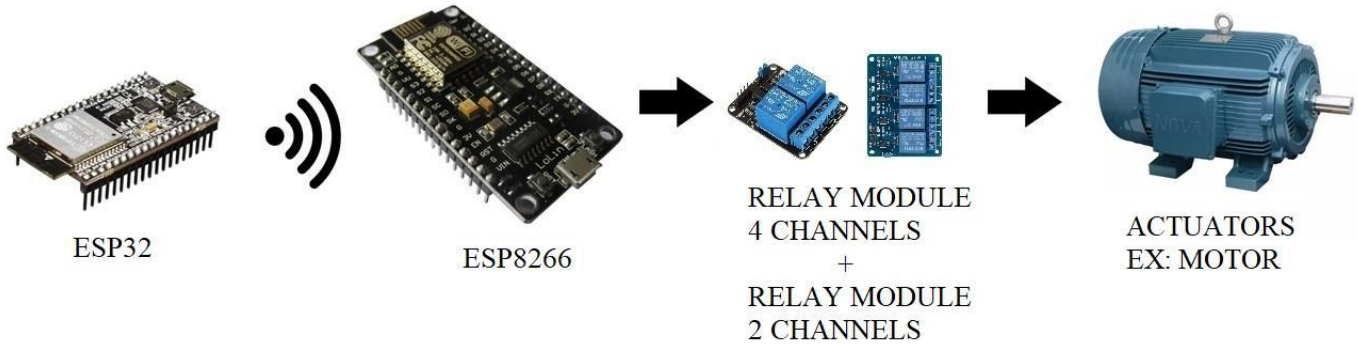
Source: Authors.

To not overload the server, the constantly updated variable will be a String containing 6 binary values (0 or 1),

referring to their respective outputs from the PLC.

Through the server, the receiver first connects to the same network as the transmitter, requests the value of the variables referring to the outputs, and receives it quickly (around 0.2 seconds) through the HTTP get method. Upon receiving the variable, the values must be separated, and the state of the respective port must be changed, which will be connected to the relay responsible for sending the command to the actuator in the field. The last step of the integration process can be seen in Figure 6, whose implemented code is in Appendix B (available at https://tinyurl.com/bn9w9az4).

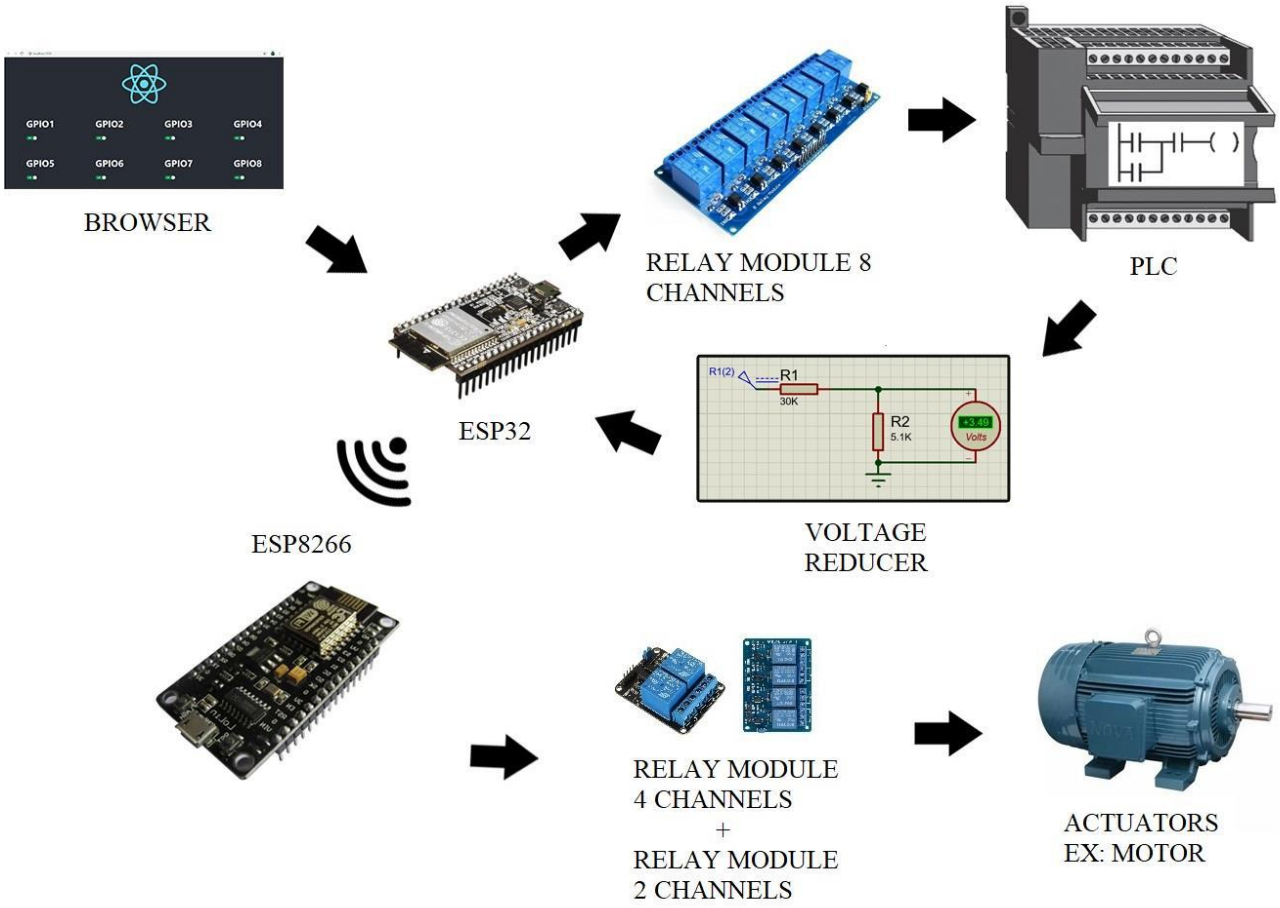
Figure 6 - Connections between transmitter, receiver, and actuators.



Source: Authors.

Finally, the complete diagram of the simplified integration between the design prototype elements is presented in Figure 7.

Figure 7 - Simplified full integration.

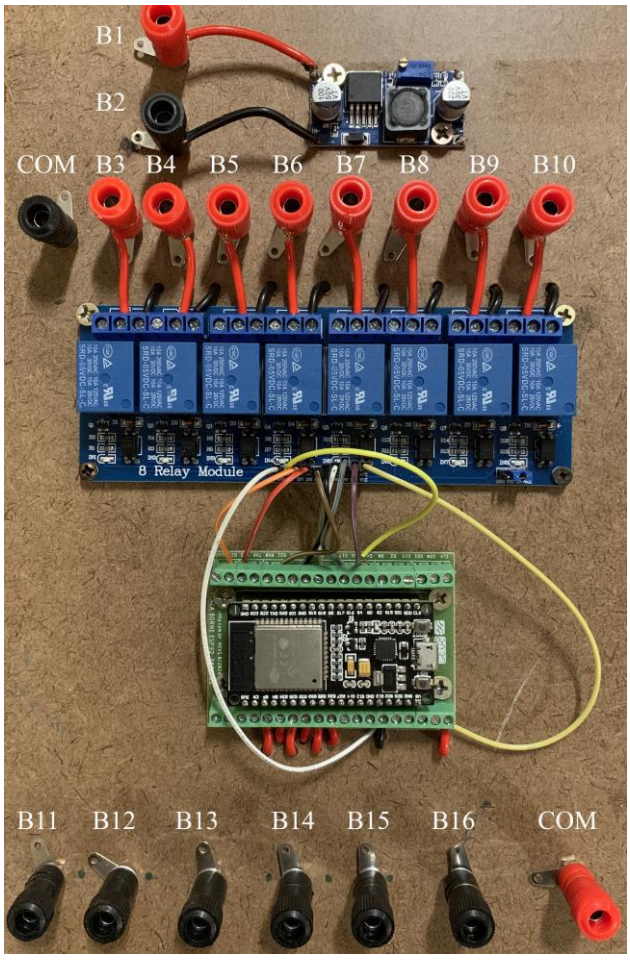


Source: Authors.

3.2 Results

To verify and validate the functioning of the proposal of this work, a prototype was built in thin wood, according to Figure 8, whose size is necessary to fix the plate on the didactic bench of the automation laboratory of Unifei campus in Itabira.

Figure 8 - Prototype of the board that controls the PLC.



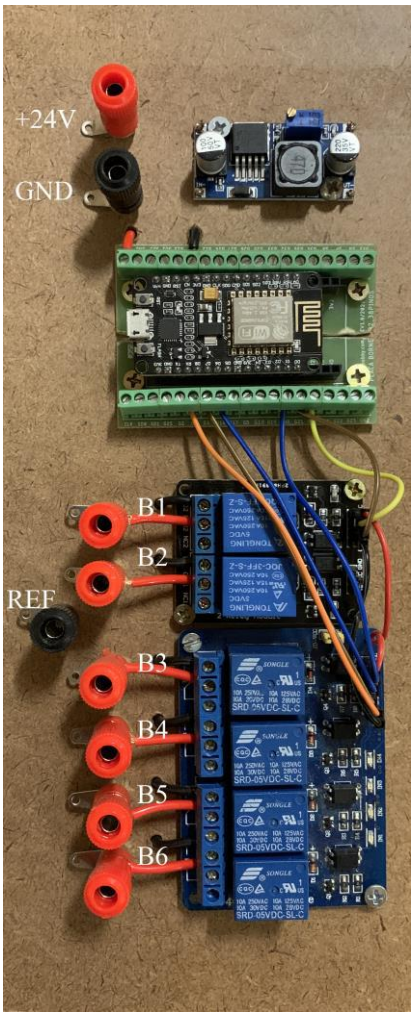
Source: Authors.

The entire circuit is energized by terminals B1 and B2, responsible for the connection, respectively, with 24 VDC and OV of the bench. In the red terminals (B3 to B11), it is possible to connect up to eight PLC inputs that will be controlled through the browser, the common pin of the inputs must be connected to COM (black), which is the board's reference ground.

The COM (red) terminal must be connected to the common outputs. On terminals B11 to B16, connect the outputs you want to control, whose voltage must be 24 VDC so that the voltage divider works correctly and applies the correct voltage to the GPIO pins of the MCU. Once all connections are made, it is possible to observe the status of the outputs through the address "http://192.168.1.4:81/Saida", which shows a String containing the binary values of each of the respectively connected outputs.

The board developed in Figure 9 follows the same standards for the bench, also powered by 24 VDC converted to 5 VDC through the LM2596. On the REF terminal, one of the phases (R, S, or T) must be connected, and this will be the phase that will leave the terminals B1 to B6, referring to the PLC outputs. The distance for the ESP8266 to connect to the ESP32 network is up to 50 meters outdoors and about 20 meters indoors, but this distance can be reduced according to the number of objects or walls that can attenuate the signal. Wi-Fi.

Figure 9 - Prototype of the board that receives the output signals from the PLC.



Source: Authors.

The star-delta start of a three-phase motor was carried out for the functioning test. As the purpose is not to implement the PLC logic, the ladder diagram shown in Figure 10 was used.

Figure 10 - Ladder diagram star-delta starting of a motor.

Source: Authors.

The T4:1 timer was used only to prevent the de-energization time of the contactors referring to the O:2/1 (star close, K2) and O:2/2 (delta close, K3) outputs from being too short, and a short circuit occurs. It follows that the O:2/0 output refers to the motor energizing contactor K1, connected to terminal B1 of the receiving board, while K2 and K3 are connected to terminals B2 and B3 of the same receiving board, respectively. The verification of the entire execution of the star-delta starter of a motor can be consulted by the link https://tinyurl.com/yxsoqqma.

4. Conclusion

Although the star-delta starting of a three-phase motor is simple, it allowed the validation and correct functioning of the prototype built and configured in this work. No significant delay in the commands was noticed when activating the input initiating the motor start. The only delay present is the time it takes for the ESP32 to update the server's output, which in this case is configured to be done every 200 ms. However, this time can be shorter as the board clock is adjustable for frequencies between 80 at 240 MHz. The prototype can be extended to any PLC that supports input and output voltages of 24 VDC.

With the design presented in this work, there is the possibility of extending the logic and codes to applications that involve a more significant number of inputs and outputs, limited to the number of inputs and outputs of the built prototype. In addition, it is possible to increase the operating distance if the equipment used is in an environment with a Wi-Fi network instead of using its network.

In future work, we intend to apply the same project idea for systems and processes that involve analog variables and not only digital ones. There is also the possibility of developing a supervision system to improve the monitoring of process variables. There is also the idea of building larger boards being possible applications with a more significant number of inputs and outputs. Finally, the visual and physical connections can be improved through PCB designs for the final board.

References

Almeida, P. S. D. (2019). *Indústria 4.0: princípios básicos, aplicabilidade e implantação na área industrial*. Érica.

Almeida, R. M. A., de Moraes, C. H. V., & Seraphim, T. D. F. P. (2017). *Programação de Sistemas Embarcados: Desenvolvendo Software para Microcontroladores em Linguagem C*. Elsevier.

Buoro, A. S. (2013). *Controle dos motores e acionamento sem fios de uma pequena embarcação*. Projeto de Graduação, Universidade do Estado do Rio de Janeiro, Departamento de Engenharia Eletrônica e Telecomunicações, Rio de Janeiro, Brasil.

Camargos, A. F. P., Santos, C. R. B., Silva, F. D., Kai, B. H. D., & Vieira, V. (2022). Produto educacional: automação residencial com uso de Arduino e IoT. *Research, Society and Development*, 11(6), e8311628882-e8311628882.

Cavalcante, C. G. S., & Almeida, T. D. D. (2018). Os benefícios da Indústria 4.0 no gerenciamento das empresas. *Journal of Lean Systems*, 3(1), 125-152.

Hermann, M., Pentek, T., & Otto, B. (2016). Design Principles for Industrie 4.0 Scenarios. *In 2016 49th Hawaii International Conference on System Sciences (HICSS)*, (pp. 3928-3937). IEEE.

Köche, J. C. (2016). *Fundamentos de metodologia científica*. Editora Vozes.

Kurniawan, A. (2019). *Internet of Things Projects with ESP32: Build exciting and powerful IoT projects using the all-new Espressif ESP32*. Packt Publishing Ltd.

Lee, J. S., Su, Y. W., & Shen, C. C. (2007). A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi. *In IECON 2007-33rd Annual Conference of the IEEE Industrial Electronics Society*, (pp. 46-51). IEEE.

Lima, C. C., Schlemmer, E., & Morgado, L. (2020). Internet das Coisas e Educação: uma revisão sistemática da literatura. *Research, Society and Development*, 9(11), e6039119674-e6039119674.

Monk, S. (2013). *Programação com Arduino: começando com Sketches*. Bookman Editora.

Monk, S. (2015). *Programação com Arduino II: Passos avançados com sketches*. Bookman Editora.

Morimoto, C. E. (2008). Redes, guia prático. *Porto Alegre: Sul Editores*, 4.

Özerdem, Ö. C. (2016). Design of two experimental setups for programmable logic controller (PLC) laboratory. *International Journal of Electrical Engineering Education*, 53(4), 331-340.

Pinho, A. G., Olímpio, E. J. S., Cabral, L. M., de Oliveira Filho, R. M., Silva, B. C. R., Furriel, G. P. & de Melo Junior, G. (2021). Desenvolvimento de bancada didática contendo múltiplos sensores e atuadores. *Research, Society and Development*, 10(13), e222101321165-e222101321165.

Pugliese, L. F., de Oliveira, T. G., da Silva, D. L. F., Rodor, F. F., da Silva Braga, R. A., & Amorim, G. F. (2022). Modeling and development of a low-cost didactic plant for teaching in multivariable systems. *Research, Society and Development*, 11(7), e33011730249-e33011730249.

Rodrigues, T. R., Massoca, J. M., Neto, M. M., da Silva Rodrigueiro, M. M., Oliveira, K. S. M., & dos Santos, P. S. B. (2022). Automação e IoT nos processos de produção de biodiesel: revisão sistemática. *Research, Society and Development*, 11(3), e31311326509.

Sacomano, J. B., Gonçalves, R. F., Bonilla, S. H., da Silva, M. T., & Sátyro, W. C. (2018). *Indústria 4.0*. Editora Blucher.

Santos, J. P., Andrade, A. A., Facó, J. F. B., Santos, E. B., & Thimoteo, A. C. A. (2020). Industry 4.0-Efforts to adjust man the Revolution 4.0. *Research, Society and Development*, 9(4), 2.

Silva, G. S. C. (2015). *Acionamento remoto de motores elétricos trifásicos de indução: utilização da plataforma microcontrolada arduino e comando via celular com interfaceamento wi-fi*. Monografia (Lato Sensu) – Pós-graduação em Sistemas de Telecomunicações, Universidade Aberta do Brasil (ESAB), Vila Velha, ES.

Silva, M. S. (2010). *JavaScript-Guia do Programador: Guia completo das funcionalidades de linguagem JavaScript*. Novatec Editora.

Singh, S., & Singh, N. (2015). Internet of Things (IoT): Security challenges, business opportunities & reference architecture for E-commerce. *In 2015 International conference on green computing and internet of things (ICGCIoT)*, (pp. 1577-1581). IEEE.

Souza, P. D., Junior, S. J., & Neto, G. G. (2017). Indústria 4.0: contribuições para setor produtivo moderno. *Encontro Nacional de Engenharia de Produção*, Joinvile, 4.

Thiollent, M. (1988). Metodologia da pesquisa-ação. *In Metodologia da pesquisa-ação*. (pp. 108-108).

Tripp, D. (2005). Action research: a methodological introduction. *Educação e pesquisa*, 31(3), 443-466.