2022 Congreso de Tecnología, Aprendizaje y Enseñanza de la Electrónica (XV Technologies Applied to Electronics Teaching Conference (TAEE) | 978-1-6654-2161-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/TAEE54169.2022.9840723

M3CVME: Supporting Tool for Electric Vehicle Speed Controller Supervision

Juan Cerezo Sánchez
Inst. Univ. de Microelectrónica Aplicada (IUMA). Univ. Las Palmas de G.C. (ULPGC).
L.P. de Gran Canaria, Spain.
ORCID: 0000-0002-2914-170X

Sonia León del Rosario
Inst. Univ. de Microelectrónica Aplicada (IUMA). Univ. Las Palmas de G.C. (ULPGC).
L.P. de Gran Canaria, Spain.
ORCID: 0000-0001-8998-455X

José Cabrera Peña
Inst. Univ. de Microelectrónica Aplicada (IUMA). Univ. Las Palmas de G.C. (ULPGC).
L.P. de Gran Canaria, Spain.
ORCID: 0000-0001-6557-2294

Aurelio Vega Martínez
Inst. Univ. de Microelectrónica Aplicada (IUMA). Univ. Las Palmas de G.C. (ULPGC).
L.P. de Gran Canaria, Spain.
ORCID: 0000-0002-4154-8799

***Abstract*—A teaching tool, M3CVME, is presented that makes it easier for students to design and implement a data communication system. This tool was developed with the aim of assisting lab works of the Industrial Informatics subject. Using it, the student can select what entity of the communication system they want to simulate: the supervisory application, the system controller, or a remote TCP client. As M3CVME allows to simulate each communication system entity, the student can use each to check his own developed entity. The main advantage achieved is that the student can develop a moderately complex and almost real communication entity during the lab-session time, due to the support of the entities simulated by the tool, that are used to test their works. Later, a collaboration experience provided the opportunity to use M3CVME to integrate session labs with another subject of the same course level. The tool was then applied to control and supervise an electric vehicle (EV) prototype in the Power Electronics subject. With M3CVME, from the point of view of Industrial Informatics, the student can integrate systems using communication media and protocols. During several academic years (since 2018) it has been possible to verify that the availability of the M3CVME tool has been totally determinant for the students to be able to finish practices with this level of deepness and complexity.**

Keywords— Serial Communication, Master-Slave Protocol, Client-Server Connection, Supervision Application, Subject Coordination

I. Introduction

A coordination experience between two fourth-course subjects of "Automatic and Electronic Industrial Engineering" studies was carried out in the Civil Engineering School of the University of Las Palmas de Gran Canaria, Canarias, Spain. It consisted of a collaborative project regarding a speed controller for an electric vehicle. From the point of view of the Power Electronics subject, it was studied the operation of the controller based on an Arduino module [1]. From the Industrial Informatics subject's point of view, the students learnt about communications and the development of a supervisory application with remote and local connections. The experience has been already documented in [2]

The present article shows the developed software tool, M3CVME, provided to the student during the cooperation experience, that supports the design of the application supervising the controlled EV. This application must communicate locally with the controller through a master-slave serial protocol over the RS-232/RS-485 standard [3] and allow remote communication using the TCP/IP standard [4].

The tool is presented and explained from the side of the Industrial Informatics subject since it was initially developed to improve its own lab-practises and to provide larger and deeper range of knowledge to the students.

This work, as well, continues the work line started with [5], where first developments of communication tools were produced to expand the possibilities of the devices in the Industrial Informatics laboratory and offer the students a richer knowledge.

First, the requirements needed by the Electronics Power Lab and the EV are described in II. The protocol developed to communicate the controller with the supervisory application is showed in III. In IV, the task requirements of the Industrial Informatics Lab are explained. The M3CVME tool is described in V. And finally some results and conclusions are explained in VI and VII.

II. Work in the Electronics Power Lab

A. Physical Model of the Electric Vehicle

For the study of an electric vehicle (EV) from the point of view of Power Electronics, a physical model is available in the Power Electronics laboratory; this will be called from now on VEModule.

The VEModule contains the main elements of this type of vehicle (Fig.1): AC/DC converter, batteries, DC-DC converter (Buck-Boost), battery-charge controller (BMS), and a BLDC motor, with its controller, a three-phase inverter, and with rotor position sensor (*Fig. 2*)

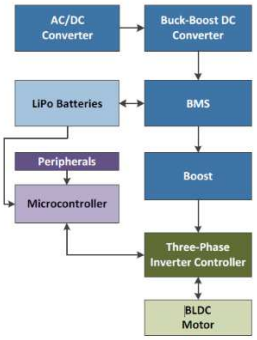

Fig. 1 – VEModule Block Diagram [2]

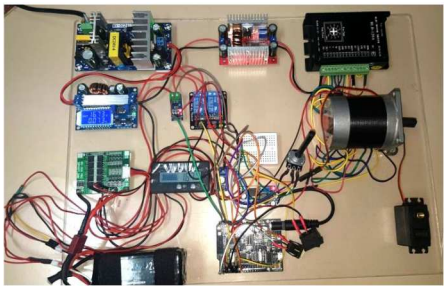

Fig. 2 - VEModule Elements [2]

978-1-6654-2161-4/22/$31.00 ©2022 IEEE

B. EVModule Controller.

To control this *EVModule*, control electronics based on an *Arduino*-type device is used. The control is based on the monitoring of some measurements of the physical model, and the user is allowed to establish some operating values (setpoints).

Table 1 - Measures and Setpoints

Name	*Type*
Input voltage and current	Measures
Battery Status	Measures
Motor Start up	Setup
Motor speed (PWM)	Setup
Brake on/off	Setup
Acceleration or deceleration slope	Setup
Boost disconnection	Setup

The controller was first designed to offer a menu of commands, shown to the user through the serial terminal. The execute options allow to visualize the monitored data and stablish the setpoint values.

In this laboratory, the student is provided with the material needed to analyse and connect the *EVModule* elements. The controller is provided already programmed. Anyway, it is possible as well to adapt the program to the singularities of each work group

C. EVModule supervision.

The operation of the *VEModule* is to be monitored from a PC connected to the controller. The objective is to monitor and assign values from a supervision application.

To allow an effective and real *VEModule* monitoring, the controller and the PC visual interface were modified. Orders form the user are finally managed as integrated commands in a *Modbus*-type serial protocol.

The aim is that the controller is going to be supervised through a serial connection type *RS-232* (or *RS-485* to allow the connection of several modules to the same application). (*Fig. 3*).

A Modbus-type serial protocol called *MEB* (Basic Master/Slave) was designed, which allows the master device, the supervision PC, to send input reading orders and setpoint writing commands to the slave, that is, the device controller.

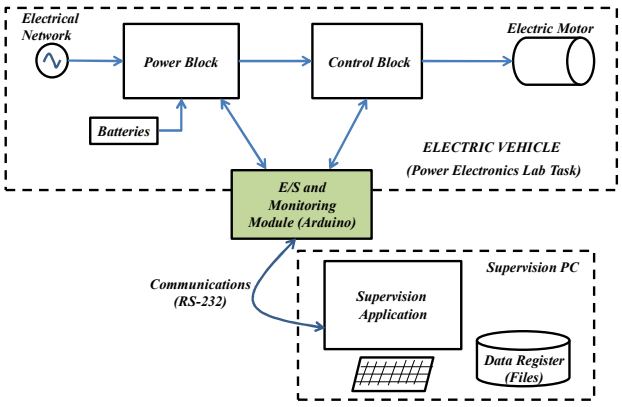

Fig. 3 – VEModule Supervision

Each *VEModule* is assigned a device number on the RS-485 network and implements the slave network function, using the *MEB protocol*. At the other side, the supervision application is in charge of generating the orders according to this protocol, to request the status of the monitored measures and to establish the setpoints.

III. MEB Protocol

A. Features

The Basic Master-Slave (*MEB, Maestro-Esclavo Básico*) protocol was initially meant to be used in the laboratory activities of the Industrial Informatics subject. It implements a communication method based on the Master-Slave model for managing a network connecting devices over RS-485.

This protocol is Modbus type (from Modicom) [6] and similar to Host-Link (from Omron) [7]. It is based on request and answer frame handshake.

The frame format follows a general pattern, although, depending on the commands (orders) included in the requests, the frame will contain different fields. The format of the response also depends on the type of commands used.

B. Data Model

MEB protocol is based on a data model that divides the information address area into different types of data blocks with their own characteristics (Table 2). In turn, each block is divided into analog and digital addresses.

Table 2 – Data Block Types

- **Inputs**
  Contain the copy of the **Input data** of the hardware device. This block only allows data reading (Read Only, RO).
- **Outputs**
  Contain the information to be updated in the **Outputs** of the hardware device. This block only allows data writing (Write Only, WO)
- **Variables**
  Contain information about data in **memory** of the hardware device. This block only allows data writing (Write Only, WO)
- **Flags**
  Contains information of **the status** of the hardware device. This block only allows data reading (Read Only, RO).

C. Frame Format

The typical frame of the MEB protocol has a fixed structure. There are two types of frames: a.- Request Frame (Fig. 4 –), and b.- Response Frame (Fig. 5).

1	2		3		4		5		6						7		8	9
BOF	ID		CMD		ADDR		CNT		DATOS						FCS		EOF	LF
@	$x10^1$	$x10^0$	C1	C2	$x10^1$	$x10^0$	$x10^1$	$x10^0$	$x16^3$	$x16^2$	$x16^1$	$x16^0$	...		$x16^1$	$x16^0$	*	\n

Field description:

1. ***BOF (Begin Of Frame)***
   Contains the INITIAL frame character.
   *Type: **alphanumeric**. Fixed value: '@' (0x40). Size: 1 character (1 byte)*
2. ***ID.***
   Contains the identification number of the slave device, frame origin.
   Type: numeric (base 10). Size: 2 digits (2 bytes)
3. ***CMD***
   Contains ORDER code.
   *Type: **alphanumeric**. Size: 2 characters (2 bytes)*
4. ***ADDR***
   Contains the initial ADDRESS
   *Type: **numeric** (base 16). Size: 2 digits (2 bytes)*
5. ***CNT***
   Contains the DATA AMOUNT to transfer
   *Type: **numeric** (base 10). Size: 2 digits (2 bytes)*
6. ***DATOS (*)***
   Contains the data sequence to transfer.
   *Each data: **numeric** (base 16) and 4 digits (4 bytes)*
7. ***FCS***
   Contains the result numeric value for error checking
   *Type: **numeric** (base 16). Size: 2 digits (2 bytes)*
8. ***EOF (End Of Frame)***
   Contains the END of frame character
   *Type: **character**. Fixed value: '*' (0x2A). Size: 1 character (1 byte)*
9. ***LF (Line Feed)***
   *Contains the **LF character***
   *Type: **character**. Fixed value: '\n' (0x0A). Size: 1 character (1 byte)*

Fig. 4 – Request Frame

1	2		3		4		5		6						7		8	9
BOF	ID		CMD		ADDR		CNT		DATOS						FCS		EOF	LF
@	$x10^1$	$x10^0$	C1	C2	$x10^1$	$x10^0$	$x10^1$	$x10^0$	$x16^3$	$x16^2$	$x16^1$	$x16^0$	...		$x16^1$	$x16^0$	*	\n

Field description:

1. ***BOF (Begin Of Frame)***
   Contains the INITIAL frame character.
   *Type: **alphanumeric**. Fixed value: '@' (0x40). Size: 1 character (1 byte)*
2. ***ID.***
   Contains the identification number of the slave device, frame origin.
   Type: numeric (base 10). Size: 2 digits (2 bytes)
3. ***CMD***
   Contains ORDER code.
   *Type: **alphanumeric**. Size: 2 characters (2 bytes)*
4. ***ADDR***
   Contains the initial ADDRESS
   *Type: **numeric** (base 16). Size: 2 digits (2 bytes)*
5. ***CNT***
   Contains the DATA AMOUNT to transfer
   *Type: **numeric** (base 10). Size: 2 digits (2 bytes)*
6. ***DATOS (*)***
   Contains the data sequence to transfer.
   *Each data: **numeric** (base 16) and 4 digits (4 bytes)*
7. ***FCS***
   Contains the result numeric value for error checking
   *Type: **numeric** (base 16). Size: 2 digits (2 bytes)*
8. ***EOF (End Of Frame)***
   Contains the END of frame character
   *Type: **character**. Fixed value: '*' (0x2A). Size: 1 character (1 byte)*
9. ***LF (Line Feed)***
   *Contains the **LF character***
   *Type: **character**. Fixed value: '\n' (0x0A). Size: 1 character (1 byte)*

Fig. 5 – Respond Frame

IV. Work in the Industrial Informatics' Lab

A. Lab Task

The task to be carried out in the Industrial Informatics subject's laboratory consists of developing an integration application using the *VEModule* as the system been controlled and supervised.

This application has a double functionality. On the one hand, it must perform monitoring tasks by communicating with the controller through a serial connection and managing communications according to the *MEB protocol*. On the other hand, it must work as a server to allow TCP/IP connections so that another remote client application can send commands and monitor the status of the *VEModule.* (Fig. 6)

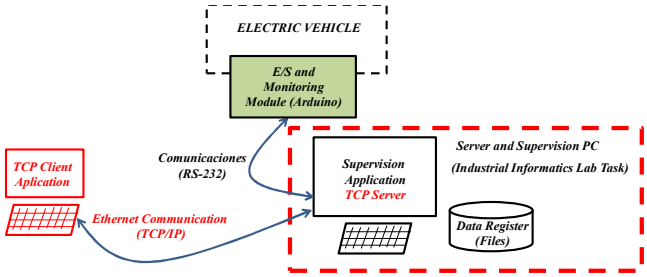

Fig. 6 - Task at the Industrial Informatics laboratory

Orders sent from Client to Server must meet the *MEB protocol format,* encapsulated in the TCP frames that are transferred from the local equipment to the remote one.

Table 3 – Specifications of the Integration Application

- Application developed in C# with Microsoft's Visual Studio
- Configurable application (number of connected slave device, mapping of inputs/outputs and ranges)
- Master of the RS-485 network fitting the MEB protocol
- TCP connection server fitting the MEB protocol
- Show the evolution of communications (show the frames)
- Graphically display the status of Inputs and Outputs depending on the type (analog or digital)
- Allow the user to set the output values (setpoints)

The student, in the laboratory, is provided with all the information about the MEB protocol, the list of Inputs/Outputs with their meaning and ranges (Fig. 7), as well their mapping in the protocol. (Fig. 8).

#Ord	Type	TAG	Description	Comment
1	EA	V2	Input Voltage to the Controler-Inverter	Min=0V. Max=5V. ADC=12 bits
2	EA	I	Input current to the Controler-Inverter	ADC=12 bits
3	EA	V1	Battery Voltage	Min=0V. Max=5V ADC=12 bits
4	SA (SD PWM)	VEL	ME Speed set	Min=0V. Max=5V DAC=12 bits
5	SD	EN	VC module enable	0=Enabled 1=Disabled
6	SD	BRK	CV brake enable	0=Enabled 1=Disabled
7	SD	PRO	BOOST activation	0=Enabled 1=Disabled
8	SD	DES	Ciruit Discharge	0=Enabled 1=Disabled
9	SD	FR	ME rotation direction	0=Clockwise 1=Counterclockwise

Fig. 7 - List of Inputs/ Outputs

Type	*ADDRESS*	TAG
EA	0	V2
EA	1	I
EA	2	V1
SA	0	VEL
SD	0 (Bit 0)	EN
SD	0 (Bit 1)	BRK
SD	0 (Bit 2)	PRO
SD	0 (Bit 3)	DES
SD	0 (Bit 4)	FR

Fig. 8 - I/O mapping

V. M3CVME: Supporting Tool

A. Objectives

The need of this supporting tool arises due to the next requirements for the lab task:

1. Provide a training and educational environment to demonstrate the operation of the requested integration application.
2. Provide utilities that allow the student to check the different functional parts of their application while developing it.
3. Do not depend on the hardware of the controller device to check the developed application functionality.
4. Provide the possibility of developing and testing the application using non-class time, that is, to work from home.

Given this need, the M3CVME tool (Monitoring and Control of a Speed Control Module of an Electric Motor) was created. It is an application developed in C# with Microsoft's Visual Studio [8] (Fig. 9).

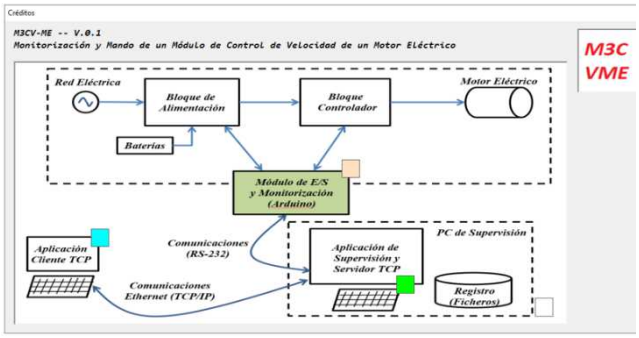

Fig. 9 - Herramienta M3CVME

M3, as students and teachers call it, is used by teachers as a working example, and by students as an assisting tool for the creation of their integration application. Apart of achieving the requirements listed above, it offers additional utilities:

- Supervisor and Server Utility (SUP SRV). It behaves like the integration application the student must develop.
- Arduino Simulator Utility (SIM ARD). This behaves as the *VEModule* Controller.
- TCP Client utility (TCP CLI). It works as the remote application.

Supervisor and Server Utility

It is the utility that behaves as the integrating application that students must develop. It has the following functionalities:

- It is an application that implements the MEB protocol, playing the role of master device of the RS-485 network. It makes requests to the *VEModule controller using* commands sent through the serial port.
- It lets configuring the device number it is to be connected to, as well as the mapping of the I/O signals to the MEB protocol.
- It allows the user to control the values outputs, both analog and digital, through graphic elements. Graphical controls are also used to show the inputs values.
- It displays all the transmitted and received communication frames, both those of the MEB and TCP protocol.
- It provides as well, communications status indicators.

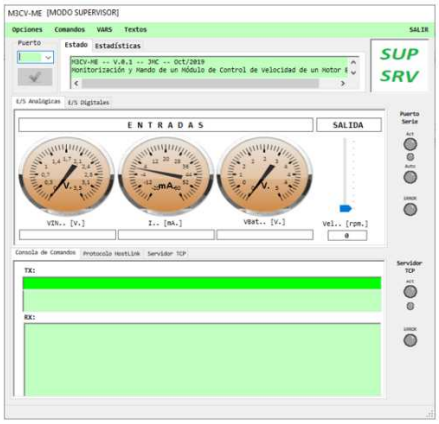

Fig. 10 - SUP SRV Utility

B. Arduino Simulator Utility

This is the utility that behaves as the *VEModule controller*. It has the listed functionalities:

- It is an application that implements the MEB protocol, working as a slave device of the RS-485 network. It responds to master requests.
- It allows configuring the device number of the simulated slave, as well as the mapping of the I/O signals to the MEB protocol.
- It lets the user to establish the values of all inputs, both analog and digital, through graphic elements. It also shows through graphical controls the values of all outputs.
- It provides as well, communications status indicators.

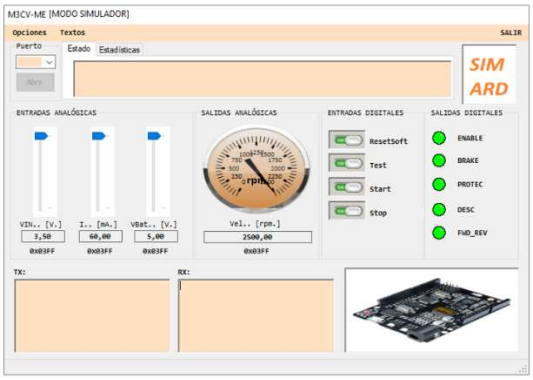

Fig. 11 -Utilidad SIM ARD

C. TCP Client Utility

This is the utility that behaves as the remote application requesting data from the integrating application. It provides the next functionalities:

- It is an application that performs as a TCP client and encapsulates the MEB protocol within transport layer frames (TCP) as master device.
- It lets configuring the device number to which it is going to connect, as well as the mapping of the I/O signals to the MEB protocol.
- It allows the user to control the outputs values, both analog and digital, through graphic elements. It also shows, by means of graphical controls, all inputs values.
- It monitors all the communication frames transmitted and received from the TCP protocol.
- It provides as well, communications status indicators.

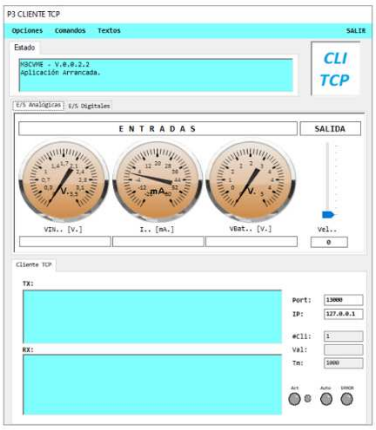

Fig. 12 - CLI TCP Utility

D. Client application for Android mobile

As a complement to the M3 tool, a TCP client application was also developed to run on Android-type mobile devices. It was developed in C# language with the Xamarin package [9] on Microsoft's Visual Studio environment.

With this application, students can check the performance of their integration application using their own mobile and a WIFI connection.

Fig. 13 - Mobile client application

Its main functionalities are:

- It implements the MEB protocol over TCP frames.
- Performs the TCP client function and connects to the supervisor and server application.
- Allows to request input values and set output values.
- Allows to configure the device number it is going to be connected to, as well as the mapping of the I/O signals on the MEB protocol.
- It allows the user to control the values outputs, both analog and digital, through graphic elements. Graphical controls are also used to show the inputs values.
- It monitors all the TCP communication frames transmitted and received.
- Dispone de indicadores del estado de las comunicaciones.

E. Functionality Testing

In order to check the functionality of the application developed by the student and the M3 tool on the same laptop, it is necessary to install an additional utility called Virtual Serial Port (VSP).

The VSP allows crossed virtual serial ports, that is, to connect the input of a port to the output of another and conversely. This way, you can connect the supervision application to a virtual port, the Arduino simulator application to another virtual port, and the two utilities will communicate with each other; no external hardware is needed to be connected to the computer.

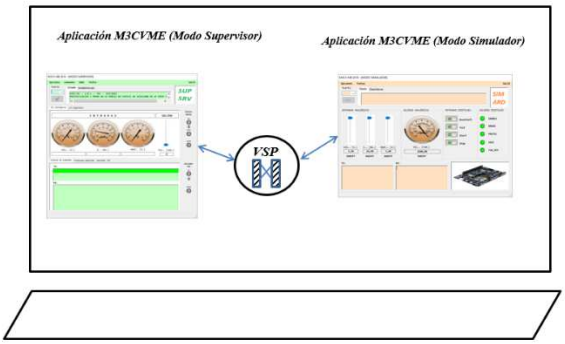

Fig. 14 – VSP Utility

VI. Results

The results obtained after the use of the M3 tools for several academic years was very satisfactory

M3 tool provided the student with a very broad and clear vision of the usefulness of the concepts studied in theory, since it showed the details of the evolution of communications between the systems integrated. To support this point, the protocol devised, MEB, was based on ASCII characters, thus providing full visibility of the frames transferred between devices.

In addition, since the tool was used in collaboration with another subject that provided the system to be controlled, students already knew the whole functionality of the equipment to be supervised. This way, the lab practise provided the students with a more realistic view of the developed application.

During several academic years (since 2018) it has been possible to verify that the availability of the M3 tool has been totally determinant for the students to be able to finish practices with this level of richness and complexity.

Each academic year approximately 35 students studied the subject till the end. All of them assured the usefulness of M3 and ware grateful to be able to carry out this type of integration practises.

VII. Conclusions

This article shows the tool called M3CVME. It is a tool created to facilitate the realization of laboratory practices based on the development of communication system integration and supervisory applications.

The tool allows several operating modes, each of them, allows to simulate a communication system entity: Controller mode based on an Arduino, local supervisor mode, and remote mode, which allows to remotely supervise the controlled system. These three modes let the students to test their own developed designs that in turn may be: a controller, a remote client, or a supervisor application.

Thanks to the availability of M3CVME, students can face the development of practices with a high level of richness and obviously also complexity, but nonetheless, students are able to finish them in the time course due the aid of the tool.

Tough these integration tasks are very useful for the students to develop a lot of important skills, the level of complexity they present can only be overcome thanks to the availability of utilities such as M3CVME that provide total support during the development process.

Collaboration with other subjects should be more commonly used. Students have in their hands the possibility of connecting a lot of knowledge which produces a deeper understanding of what they are doing, more contact with real systems, and big mental changes, arising visible enthusiasm to go on studying.

Referencias

[1] «UNO R3 + WiFi ATmega328P + ESP8266 (32Mb de memoria) USB-TTL CH340G para Arduino Uno NodeMCU WeMos ESP8266 One,» [En línea]. Available: shorturl.at/eruxL.

[2] S. O. E. Q. H. F. G. M. C. José Cabrera Peña, «Modular Battery Management System for Power Electronics Practical Laboratory Lessons,» de *2020 XIV Technologies Applied to Electronics Teaching Conference (TAEE)*, 2020.

[3] T. Instruments, «RS-485 Reference Guide,» 2014.

[4] T. Socolofsky, «A TCP/IP Tutorial,» Spider Systems Limited, 1991.

[5] S. L. E. V. a. A. V. Juan M. Cerezo, *CQMIH PLC simulator host-link communications protocol experience,* 2016 Technologies Applied to Electronics Teaching (TAEE), 2016.

[6] «Modicon Modbus Protocol Reference Guide,» MODICON, Inc., Industrial Automation Systems, 1996.

[7] OMRON, «HOST LINK UNITS. System Manual,» 1995.

[8] S. Cleary, Concurrency in C\# Cookbook: Asynchronous, Parallel, and Multithreaded Programming, O'Reilly Media, 2019.

[9] *Xamarin,* https://docs.microsoft.com/es-es/xamarin/.