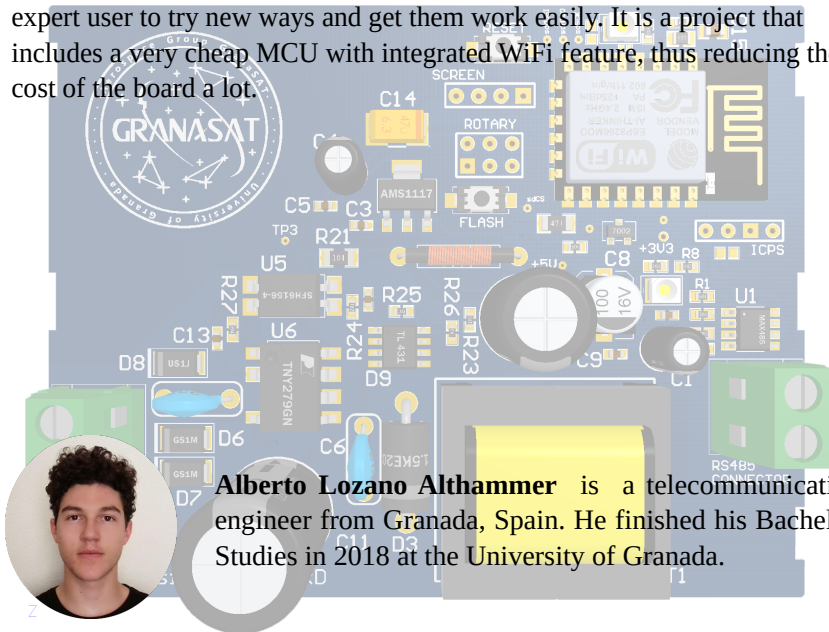




The aim of this project is to design a device which takes the measurements from a power meter and sends these to a remote server via WiFi. It should allow the new user to get involved into electronics, as well as for the more expert user to try new ways and get them work easily. It is a project that includes a very cheap MCU with integrated WiFi feature, thus reducing the cost of the board a lot.



Alberto Lozano Althammer is a telecommunications engineer from Granada, Spain. He finished his Bachelor's Studies in 2018 at the University of Granada.



Andrés María Roldán Aranda is the academic head of the present project, and the student's tutor. He is professor in Department of Electronics and Computer Technology at University of Granada.

Power Meter Telemeasurement based on ESP8266

Alberto Lozano
Althammer

TELECOMMUNICATIONS
ENGINEERING

2018/19



UNIVERSITY OF GRANADA Bachelor Degree in Telecommunications Technology Engineering



Bachelor Thesis
**Power Meter
Telemeasurement
based on ESP8266**
Alberto Lozano Althammer
Academic year 2018/2019

Tutor: Andrés María Roldán Aranda



UNIVERSIDAD DE GRANADA

GRADO EN INGENIERÍA DE TECNOLOGÍAS
DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

*Power Meter Telemeassurement based on
ESP8266*

CURSO: 2018/2019

Alberto Lozano Althammer



UNIVERSIDAD DE GRANADA

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE
TELECOMUNICACIÓN

Power Meter Telemeasurement based on ESP8266

REALIZADO POR:

Alberto Lozano Althammer

DIRIGIDO POR:

Andrés María Roldán Aranda

DEPARTAMENTO:

Electrónica y Tecnología de los Computadores

D. Andrés María Roldán Aranda, Profesor del departamento de Electrónica y Tecnología de los Computadores de la Universidad de Granada, como director del Trabajo Fin de Grado de Alberto Lozano Althammer,

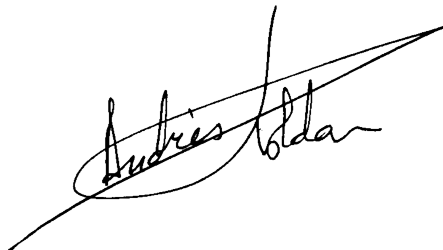
Informa:

que el presente trabajo, titulado:

Power Meter Telemeassurement based on ESP8266

ha sido realizado y redactado por el mencionado alumno bajo nuestra dirección, y con esta fecha autorizo a su presentación.

Granada, a 13 de noviembre de 2018

A handwritten signature in black ink, appearing to read 'Andrés Roldán', with a long, sweeping horizontal stroke extending to the right.

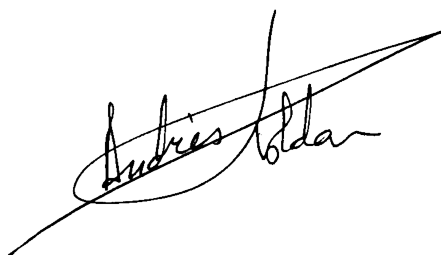
Fdo. Andrés María Roldán Aranda

Los abajo firmantes autorizan a que la presente copia de Trabajo Fin de Grado se ubique en la Biblioteca del Centro y/o departamento para ser libremente consultada por las personas que lo deseen.

Granada, a 13 de noviembre de 2018

A stylized handwritten signature in black ink, consisting of a large loop followed by several smaller, connected strokes.

Fdo. Alberto Lozano Althammer

A handwritten signature in black ink, featuring a large, sweeping loop that encompasses the first part of the name, followed by more fluid, connected strokes.

Fdo. Andrés María Roldán Aranda

Power Meter Telemeassurement based on ESP8266

Alberto Lozano Althammer

PALABRAS CLAVE:

ESP8266, WiFi, Medidor de potencia, [PCB](#), [Altium](#), [MODBUS](#).

RESUMEN:

El objetivo de este proyecto es diseñar un producto capaz de pedir y almacenar los datos de consumo de un medidor de potencia propio de una vivienda o edificio, y así llevar un registro de éste. Estará controlado por el chip ESP8266, integrado en un módulo con funcionalidad WiFi. Los datos se volcarán en un servidor remoto a través de una conexión WiFi. La intención es implementar los componentes necesarios en una [PCB](#) diseñada mediante el software [EDA](#) Altium Designer. El proyecto intenta abordar este diseño como si de un producto de *hardware* para uso comercial se tratase, poniendo enfoque sobre las diferentes partes del diseño de *hardware* y estudiando capa parte por separado.

Power Meter Telemeassurement based on ESP8266

Alberto Lozano Althammer

KEYWORDS:

ESP8266, WiFi, Power Meter, [PCB](#), [Altium](#), [MODBUS](#).

ABSTRACT:

The main purpose of this project is to develop a full product able to ask consumption data from a power meter of a house or building to have a record of power usage. It will have the ESP8266 chip as the controller of the device, embedded in a module with WiFi feature. Data will be dumped to a remote server via WiFi connection. It is intended to integrate all needed components on a [PCB](#), designed with the [EDA](#) software Altium Designer. This project tries to tackle the design as for a comercial hardware product, placing the focus on the several stages of hardware design, analysing each section independently.

Agradecimientos:

En primer lugar, quiero agradecer a mi familia. A mis padres, Andrea y Manolo, por estar siempre ahí para apoyarme y animarme en los momentos duros, y a mi hermano Sergio por destacar infinidad veces la parte tan interesante de lo que estudiamos, en momentos cuando solo veía lo menos bueno.

Por otro lado, este camino tan duro como es nuestro grado no hubiera podido soportarlo sin la compañía que he tenido durante estos años. A los Telecomos originales ivanzu, adri, el pisha, ramos y zapa que han estado todo este tiempo bien presentes y tan buenos momentos hemos pasado juntos. A bailon e irenilla porque los descansos de primero y segundo no hubieran sido comparables: el bailooooooooon. A los que luego fueron viniendo, joe, guz, mase, glo y el morenillo ese to gracioso de la tarde, el reda, dándome que pensar sobre lo increíble que hubiera sido haber juntado los de la mañana y la tarde desde el principio. A mi hermano elegido albert, porque hemos pasado tanto juntos que tuvimos que estudiar lo mismo, era irremediable. A ichi por darle ese aire tan suyo a todo lo que toca, y junto a ella al equipo de electrónicos valero, carmelo y los adoptados juani y pablo, que se dieron cuenta de lo bien que se estaba con los pequeños.

Quiero agradecer también esos profesores que contrarrestaban todo la pasividad que abunda en la carrera, e ibas con muchas ganas a sus clases. A Andrés, porque he sufrido mucho en el proyecto pero se estudia sin duda de lo más interesante en la carrera.

Acknowledgments:

First of all, I want to thank my family. My parents, Andrea and Manolo, for always being there to support me and cheer me on hard times, and my brother Sergio for the many times he highlighted the interesting part of what we study, at a time when I only saw the less good.

On the other hand, this road as hard as our grade could not have endured without the company I have had during these years. To the original Telecomos ivanzu, adri, the pisha, ramos and zapa that have been all this time well present and such good times we have spent together. To bailon and irenilla because the first and second years breaks would not have been comparable: el bailooooooooon. Those who came, joe, guz, mase, glo and el morenillo ese to gracioso de la tarde, el reda, giving me to think about how amazing it would have

been having joined the morning and evening mates from the beginning. My chosen brother albert, because we have spent so much together that we had to study the same thing, it was irremediable. To ichi for giving that air she gives to everything he touches, and along with her to the electronic team valero, carmelo and the adopted juani and pablo, who realized how well they were with the children.

I also want to thank those teachers who counteract all the passivity that abounds in the degree, and you went with mood to their classes. To Andrés, because I have suffered a lot in the project but he does undoubtedly the most interesting things in the degree.

INDEX

Autorización Lectura	iv
Autorización Depósito Biblioteca	vi
Resumen	viii
Agradecimientos	xi
Index	xv
List of Figures	xix
List of Videos	xxiii
List of Tables	xxv
Glossary	xxvii
Acronyms	xxix

1	Introduction	1
1.1	State of the Art	2
1.2	Motivation	2
1.3	Objectives	3
1.4	Methodology	3
1.5	Project Structure	4
2	System Requirements	9
2.1	Mechanical Requirements	9
2.2	Hardware Requirements	9
2.3	Software Requirements	10
3	System Analysis	11
3.1	Power Module	12
3.1.1	SMPS using optocoupler	12
3.1.2	SMPS using digital isolator	13
3.1.3	Comparison table	14
3.2	Communication Module	15
3.3	Storage Module	16
3.4	Interfacing Modules	17
3.4.1	Display	17
3.4.2	Scroll through menu	18
3.4.2.1	Rotary working principle	18
3.5	MCU	19
3.5.1	Arduino UNO	20
3.5.2	Raspberry Pi 3 & BeagleBone Black	20
3.5.3	ESP32	21
3.5.4	ESP12	21
3.5.5	Comparison table	22

3.6	Tests to fulfill	22
4	System Design	25
4.1	Electronic Design	25
4.1.1	PCB design	27
4.2	Software Design	37
4.2.1	Activity diagram	38
4.2.2	RS485 Communication	38
4.2.3	OLED Display + Rotary Encoder	41
4.2.4	microSD Card	44
4.2.5	ESP-12	44
4.3	Mechanical Design	48
5	Verification and Testing	51
5.1	Electronic testing	51
5.2	Software testing	52
5.2.1	microSD Card	53
5.2.2	Communication Module	53
5.2.3	Display + Rotary	55
5.2.4	MCU - ESP-12	55
6	Conclusions and future lines	61
6.1	Future lines	62
A	MODBUS Protocol Specifications	65
A.1	Transmission Modes: RTU	65
A.1.1	Frame description	66
A.1.2	MODBUS Function Codes	67
A.1.3	Example MODBUS RTU frame	67

B	SM101C Multi-function meter registers	69
C	ESP8266 pin list	85
D	Project Budget	87
D.1	Software	87
D.2	Human Resources	87
D.3	Components cost	88
	References	89

LIST OF FIGURES

1.1	Basic scheme of the system	1
1.2	Design Method	5
1.3	Product development Gantt's diagram	7
2.1	DIN Rail Box provided	10
3.1	Block model of the project	11
3.2	Block model of power system	12
3.3	Implemented SMPS based on TNY279G - TinySwitch [1]	13
3.4	SMPS based on <i>Si8642</i> digital isolator [2]	14
3.5	RS485 to TTL converter used [3]	15
3.6	MAX485 pin definitions [4]	16
3.7	Diagram justifying the interfacing modules usage	17
3.8	Different displays compared	18
3.9	Rotary Encoder used [5]	19
3.10	Rotary Encoder close-up [6]	19

3.11 Rotary Encoder signals [6]	20
3.12 Arduino UNO [7]	20
3.13 PC-functionality boards	21
3.14 Modules with integrated WiFi functionality	21
4.1 Final block model of the design	26
4.2 Explanation of PCB rules	27
4.3 PCB's 3D model	37
4.4 Main activity diagram	39
4.5 RS485-RS232 adapter + USB	40
4.7 Response from power meter through the python code	40
4.6 Implemented python code	41
4.8 Rotary-position-updating code	42
4.9 Screen + rotary flow diagram	42
4.10 setInputFlags function diagram	42
4.11 setInputFlags function code	43
4.12 resolveInputFlags function diagram	43
4.13 resolveInputFlags function code	44
4.14 Menu implemented: a) Front page, b) 2nd page, c) Selection of "WiFi Connection" item, d) Selection of "IP Server" item, e) Selection of "Version" item, f) Selection of "Language" item	45
4.15 Diagram of the storing process in microSD Card	45
4.16 microSD Card writing code	46
4.17 ESP-12 pinout [8]	46
4.18 ESP-12 uncovered close-up [9]	47
4.19 ESP-12 connections schema	47
4.20 Diagram representing the pin-sharing situation	48
4.21 Pictures of the 3D model prototype	49

5.1	Empty manufactured PCB	52
5.2	Full mounted PCB	52
5.3	Output voltage of the power module	53
5.4	File written on SD Card	54
5.5	First communication test with power meter	54
5.6	RS485 communication setup using Arduino	55
5.7	RS485 communication setup using NodeMCU breakboard	55
5.8	Display + rotary circuit setup	56
5.9	Flashing the ESP-12 module	57
5.10	Response from power meter through the python code	58
5.11	Full mounted device	58
6.1	Compact PCB 3D model	63
A.1	MODBUS communication stack	65
A.2	MODBUS frame over Serial communication	66
A.3	MODBUS RTU message description	66
A.4	MODBUS RTU transmission	67

LIST OF VIDEOS

4.1	3D video of the PCB*	38
4.2	Video of 3D model prototype	49
5.1	Scrolling display menu video	56
6.1	Compact PCB 3D video	63

LIST OF TABLES

3.1	SMPS comparison table [2], [1]	15
3.2	RS485 converterts comparison table [10] [11] [4]	15
3.3	SD memory cards analysis -, [12], [13]	17
3.4	Display comparison table [14], [15], [16]	18
3.5	MCU comparison table [17], [18], [19], [20], [21]	22
4.1	Design rules for chemical reaction PCB manufacture	27
4.2	SM101C - HAGER Multimeter settings	39
A.1	Format for each byte in RTU mode	66
A.2	MODBUS most relevant functions	67
A.3	MODBUS Request format	68
A.4	MODBUS Response format	68
D.1	Product software costs	87
D.2	Human cost of product	88
D.3	Component cost	88

GLOSSARY

Altium Paquete *software* [EDA](#) capaz de asistir el diseño de esquemáticos electrónicos y [PCB](#), así como la simulación de sistemas electrónicos.

API Una interfaz de programación de aplicaciones o API (del inglés Application Programming Interface) es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas (también denominadas comúnmente *librerías*).

eduroam eduroam (education roaming) is an international roaming service for users in research, higher education and further education. It provides researchers, teachers, and students easy and secure network access when visiting an institution other than their own..

IC Circuito compuesto básicamente de elementos semiconductores y pasivos que se fabrica en la superficie de una oblea semiconductora y posteriormente es encapsulado. Componen la mayoría de los productos electrónicos del mercado.

MODBUS Modbus is a serial communications protocol originally published by Modicon (now Schneider Electric) in 1979 for use with its programmable logic controllers (PLCs). It is now a commonly available means of connecting industrial electronic devices..

RS232 RS-232, Recommended Standard 232[1] is a standard introduced in 1960[2] for serial communication transmission of data. It formally defines the signals connecting between a DTE (data terminal equipment) such as a computer terminal, and a DCE (data

circuit-terminating equipment or data communication equipment), such as a modem. The RS-232 standard had been commonly used in computer serial ports. The standard defines the electrical characteristics and timing of signals, the meaning of signals, and the physical size and pinout of connectors..

RS485 RS485, also known as TIA-485(-A), EIA-485, is a standard defining the electrical characteristics of drivers and receivers for use in serial communications systems. Digital communications networks implementing the standard can be used effectively over long distances and in electrically noisy environments. It uses the same differential signaling over twisted pair as RS-422..

ACRONYMS

DIN Deutsches Institut für Normung.

DIO Dual Input/Output.

DIY Do it Yourself.

EDA Electronic Desing Automation.

EEPROM Electrically Erasable Programmable Read-Only Memory.

GPIO General Purpose Input Output.

GPRS General Packet Radio Service.

I2C Inter-Integrated Circuit.

JSON JavaScript Object Notation.

LCD Liquid Crystal Display.

MCU Micro-Controller Unit.

OLED Organic Light Emitting Device, Organic Light Emitting Diode.

PCB Printed Circuit Board.

PDU Protocol Data Unit.

QIO Quad Input/Output.

SBC Single Board Computer.

SCL Serial CLock Line.

SDA Serial Data Line.

SMPS Switched-mode Power Supply.

SPI Serial Peripheral Interface.

UART Universal Asynchronous Receiver-Transmitter.

UGR Universidad de Granada.

WPA Wi-Fi Protected Access.

CHAPTER

1

INTRODUCTION

The following Bachelor Thesis ends the studies of the Telecommunication Engineering degree at the University of Granada. The aim of this project is to create a WiFi-based device which stores power consumption of a house or building to have better evidences of day-to-day power usage.

The scenario of this project will be the implemented device being attached to a power meter, it will ask with a certain frequency for current, voltage and power measurements, and will send these via WiFi to a remote server.

A basic illustration of the operation would be the one shown in figure 1.1.

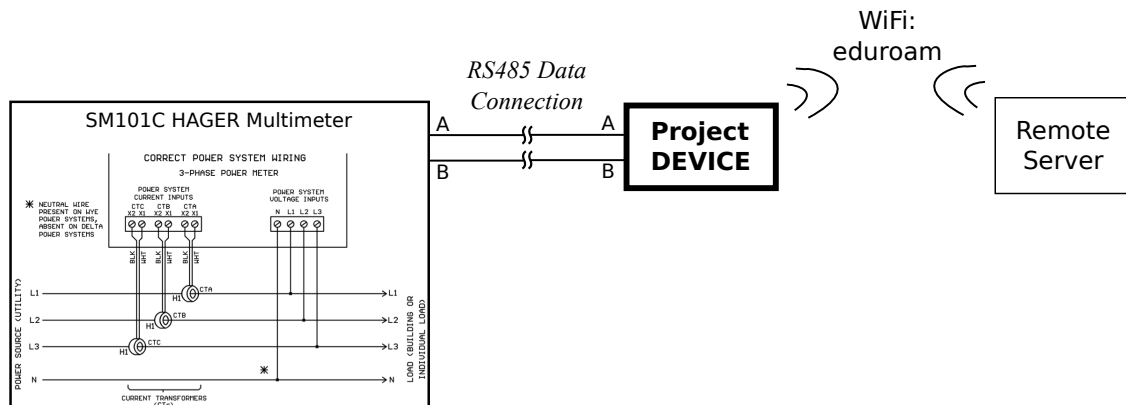


Figure 1.1 – *Basic scheme of the system*

1.1 State of the Art

1

This project covers a really present matter: power consumption. As technology develops, one key point of the near-future aims is to become more efficient regarding energy consumption. It is not only a matter of being more eco-friendly and enhance our planet condition, which is an enormously important point though; it goes highly beyond. It is the mere fact of avoiding the current waste of energy. To be energy-efficient is becoming the most important objective to reach to.

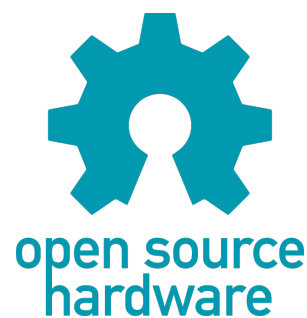
The path to follow is defined, but there is still a lot to do. Specially to enhance older equipment, for which this device would be a preliminary solution.

1.2 Motivation

The aim of this project is to create an appropriate device to enhance power usage in buildings, particularly for older generation constructions, and extend awareness of a more efficient energy consumption. This work could facilitate the testing for many new modules yet to come.

It also brings the student the opportunity of creating a useful [PCB](#) using modern techniques and learn about the manufacturing process of a current electronic system, subjects that are not easy to learn and usually require the student to learn while facing the situation of creating them.

The project is also part of the [DIY](#) community, since due to the use of mostly cheap components and not so hard electronics this project could be very easily modified and recreated by a third person with not much need of knowledge. All the schematics, gerbers, [PCBs](#) and firmware will be put online for the community as open source hardware/software. Open Source Hardware Association's logo will be shown in figure [1.2a](#) and Open Source initiative's logo will be shown in figure [1.2b](#).



(a) *Open Source Hardware Foundation's logo*



(b) *Open Source Initiative's logo*

Additionally, another motivation reason for this project is the possible application scenario for the developed device, the same University of Granada. Conscious about the present energy management in the university buildings, there is a need to gather all the energy use to have a real estimate and be able to centralize consumption. Many of the buildings are older constructions without digital meters, thus this work could be a preliminary solution in the way to achieve the objective.

1.3 Objectives

Even though some of the objectives were briefly introduced in the previous section 1.2, here the objectives will be described in a more clear, concise and organized way. For this project to be considered successful the following requisites shall be fulfilled:

1. Design a board where all required modules can be integrated.
2. Provide a firmware that makes possible to interact with all integrated modules and send the data to the remote server.
3. Provide a firmware that permits the main controller to interact and obtain data from the power meter.

Therefore, in order to fulfill the objectives, the student managing the project shall be able to:

1. Read technical documentation and be able to apply the specifications of each module to an electronic design.
2. Be able to read technical documentation to create a firmware able to interact with the module in a correct way, with 1 fulfilled.
3. Have the ability to apply critical reasoning and concepts learned during the degree.

1.4 Methodology

In order to successfully design a system, a method must be followed to ensure that time is dealt in an efficient way and solutions are achieved by critical thinking through qualitative and quantitative research. The use of such design method will lead more probably to:

- Reach the optimal solution.
- Ease of transformation of the design specifications, leading to a more competent product as time passes.

- Achieve solutions that otherwise would not be accessible.
- A better understanding of the project as a whole.

In the present design method for this project, the process will be divided in four blocks:

1. **System Requirements**, providing specifications of how the system shall behave without attending technical data.
2. **System Analysis**, breaking down the system into bigger blocks to gain a better understanding of the project. Also selecting the optimal solution.
3. **System Design**, the actual implementation of what was studied in the system analysis.
4. **Test and Verification**, apply the tests listed in the system analysis in order to validate the system requirements and check what was produced in the system design.

The design methodology followed during this project is presented in figure [1.2](#).

1.5 Project Structure

This project will be divided into 6 chapters and 4 appendices, which will describe the procedure of designing and manufacturing the project, from the very basic idea of what is expected to the in-depths of the software and hardware created to achieve it. The chapters forming this present document are:

1. **Introduction:** This chapter will introduce the project, from the conception of the idea to the planification of the work to complete the project. Figure [1.3](#) describes the Gantt diagram for the project planification.
2. **System Requirements:** This chapter will describe the essential features for the project to be considered valid.
3. **System Analysis:** During this chapter an overview of the system will be presented in big scale. This chapter will also approach the basic working principle of the components that the system will integrate.
4. **System Design:** In this chapter the design of the system will be tackled, trying to explain the technical decisions made and the in-depths of both the hardware and software implementations.
5. **Verification and Testing:** This chapter will cover the tests made to verify the functionality of the different modules and the [PCB](#) itself.

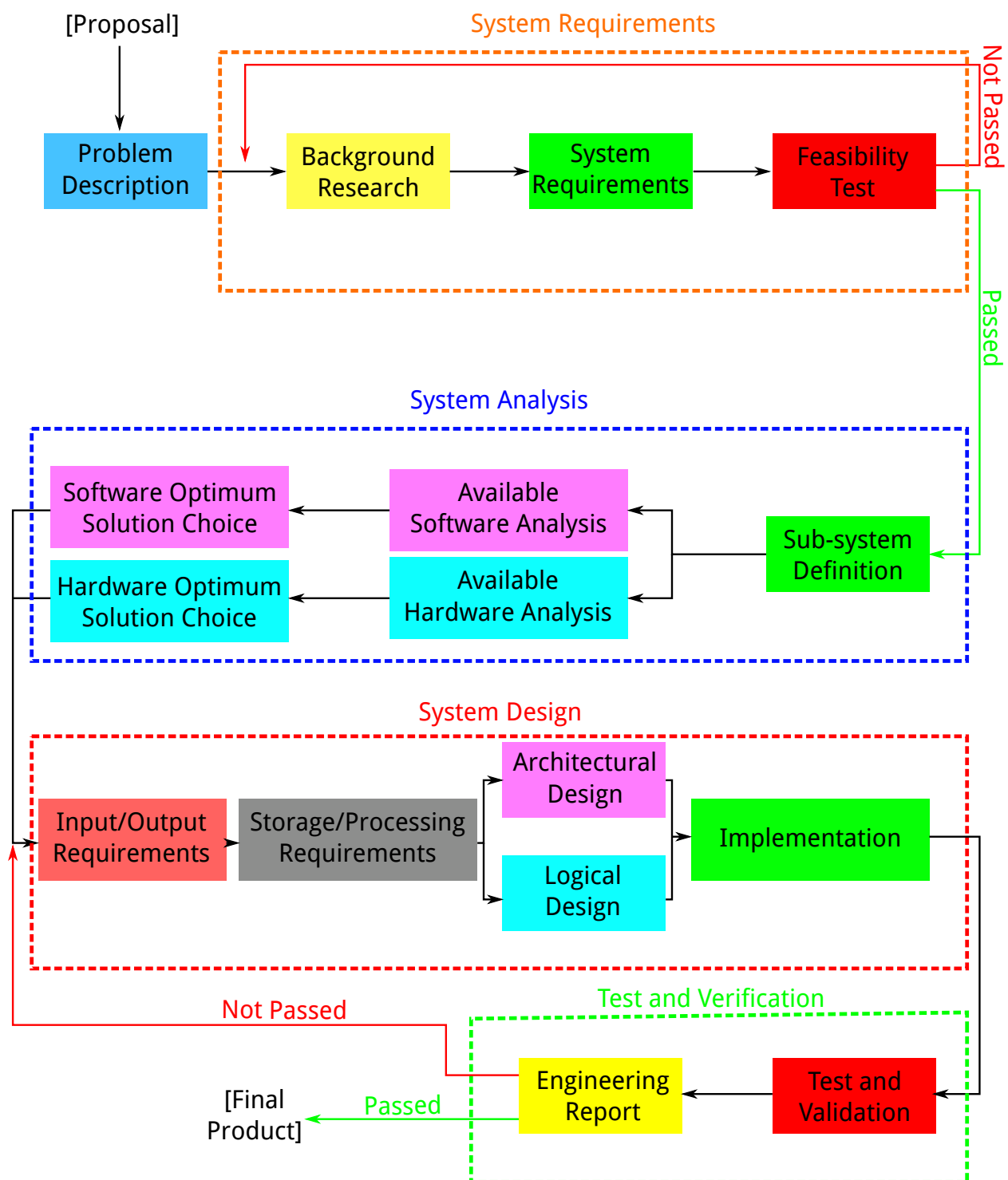


Figure 1.2 – *Design Method*

6. **Conclusion:** Summary of the things learnt and discovered during this work, along with some future improvements that may be applied to.

7. **MODBUS communication protocol:** An introduction of the transmission structure of this protocol.
8. **Registers map of SM101C multi-function meter from HAGER:** The various data types stored in this meter device's registers.
9. **ESP8266 pin release:** Pin assignment and functions from the ESP8266.
10. **Project Budget:** A calculation of the costs for developing this project.

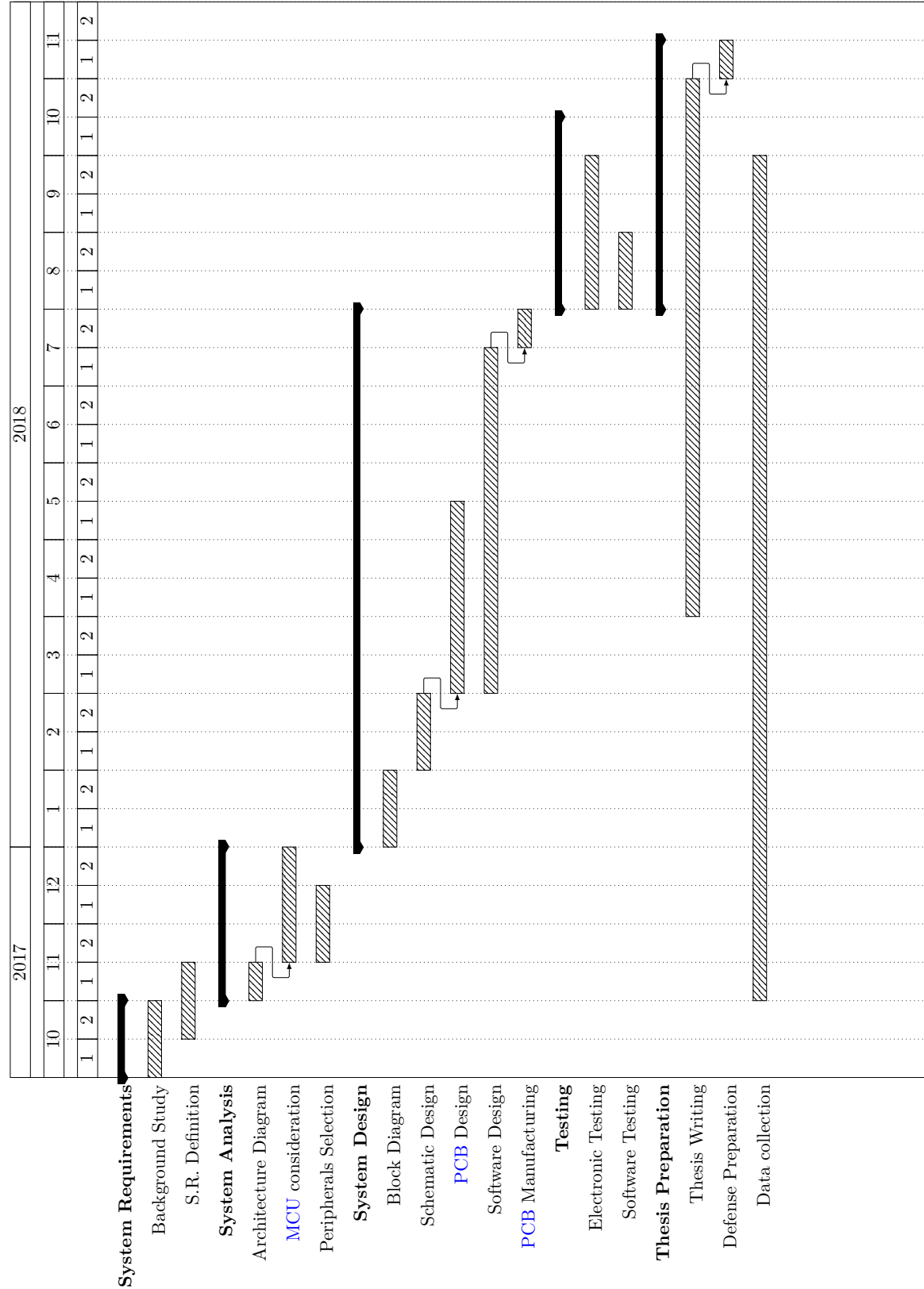


Figure 1.3 – Product development Gantt’s diagram

1

CHAPTER

2

SYSTEM REQUIREMENTS

To establish great system requirements is essential in the design and development of any complex system. These specify how a system should work and the tasks expected to accomplish. Three types of requirements will be presented: Mechanical requirements, Hardware requirements and Software requirements.

2.1 Mechanical Requirements

1. The system has to fit into the box provided, with its particular physical constraints. The corresponding [DIN](#) Rail Box is presented in figure [2.1](#).

2.2 Hardware Requirements

1. An [OLED](#) display will be included into the device, as a friendly user interface to display relevant information about the system.
2. A rotary encoder will be added to scroll through the menu which will be created.
3. An external microSD Card will be included to save the data asked to the power meter whenever server connection is not established, to prevent information loss.
4. A [RS485](#) converter will be included to be able to communicate with the power meter.
5. All data from the power meter must be transmitted via WiFi to a remote server.

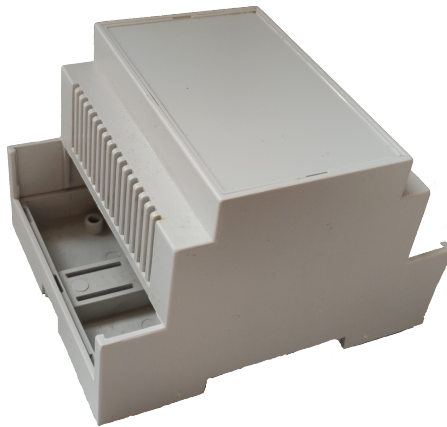


Figure 2.1 – *DIN Rail Box provided*

2.3 Software Requirements

The various objectives the project will have to fulfill are listed next:

1. Be able to read from/write to the SD memory card.
2. Whenever the remote server connection is not available, store data in SD card. When recovering connection, try again and send stored data to the server, besides the new data asked.
3. The screen shall be able to display the menu that will be implemented.
4. The rotary encoder shall ensure acceptable user experience on scrolling through the menu.
5. Establish communication with the power meter, firstly through a simple python code, to make sure the meter device works fine.
6. Be able to get the desired measurements from power meter via Arduino.
7. Be able to receive the data via the final microcontroller module to choose.
8. The software must be able to operate all the modules mentioned in 2.2.
9. The data format the device is going to work with is JSON files.
10. The software shall be able to poll consumption data frequently.
11. The WiFi connection to the server will be done through eduroam.

CHAPTER

3

SYSTEM ANALYSIS

In this chapter, the analysis of the system will be tackled. The various modules used will be described and justified in detail on the next pages. First of all, in figure 3.1 a block diagram made of the elements that will deal with the requirements from chapter 2 is shown.

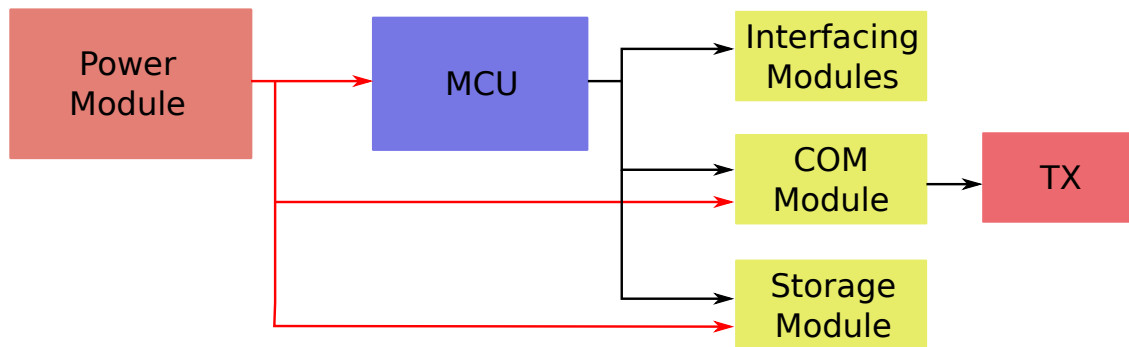


Figure 3.1 – *Block model of the project*

The project uses a **MCU** as the main controller of the board, directly connected to all modules. Which means that the code controlling each module can be tested independently and put them all together later. It will simplify the design considerably.

Each block will be discussed in the following sections.

3.1 Power Module

This block contains all power related electronics, from the power supply to the voltage converters used in this project. An illustration of how the various elements are distributed is depicted in figure 3.2.

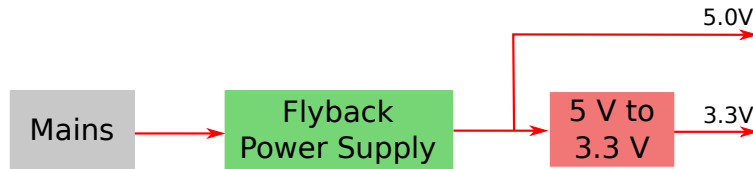


Figure 3.2 – Block model of power system

3

A switching power supply was chosen to feed the device due to the environment in which it will be integrated in. It will not need to be portable, so to take advantage of a nearby electrical outlet would be a clever decision. Apart from do not depend on a removable battery which will have to be changed every certain period of time. This will increase complexity of the project, but still a reasonable and interesting challenge to face up to.

Two implementations of **SMPS** will be analysed in order to study its pros and cons.

3.1.1 **SMPS** using optocoupler

This switched-mode power supply is based on the *TNY279G* chip, from the *TinySwitch-III* family, with an ON/OFF control for a flexible and quality design. The design is presented in figure 3.3, being an adaption of a more complete one [1]. An analysis of the operation principle will be discussed hereafter.

First of all, a two output voltage functionality can be noticed. However, the implemented one would hold only the 5 V concerning parts, because the objective is to get a single 5 VDC output.

The AC input is rectified by a full wave rectifier made up of D4, D5, D6 and D7. Then, the already rectified signal is filtered by C11, with 400 V of rated voltage.

This DC thread is connected to one of the terminals from the transformer and the drain pin from *TNY279G* to the other one, allowing the ability of cutting the current flow by the chip.

To maintain the Drain-Source voltage drop from *TNY279G* less than the (≈ 700 V) rated value, D3, D8, R20, R22 and C10 limit the peaks when the MOSFET is OFF. C13 capacitor supplies the necessary energy to U6 when MOSFET is ON and charges when is OFF.

The voltage generated in the secondary winding from TF1 is rectified by D2 and filtered by C7 to provide de 5 V output. The LC filter formed by L1, C8 and C9 right after is used

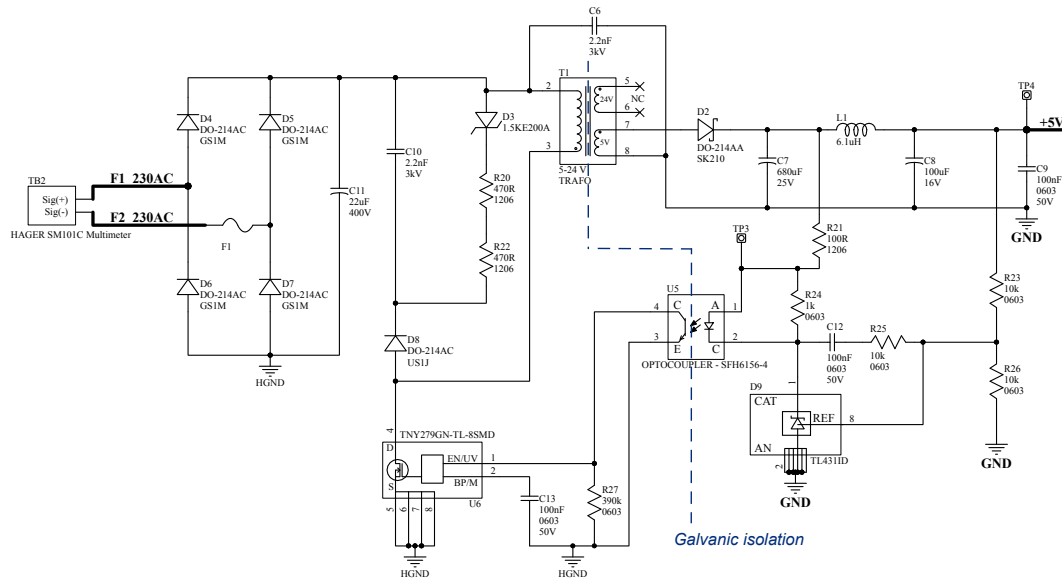


Figure 3.3 – Implemented *SMPS* based on *TNY279G* - *TinySwitch* [1]

to reduce the output ripple.

The reference voltage provided by D9 permits regulation of the output voltage. With a simple voltage divider the TL431 controls the output and the optocoupler gives feedback to the switcher. In addition, the optocoupler will permit electrical insulation between primary and secondary windings.

From the 5 V output, a voltage converter will provide the 3.3 V for the needed modules.

3.1.2 *SMPS* using digital isolator

Optocouplers are frequently used in *SMPS* for galvanic separation between the primary and secondary sides as well as from the feedback generator. However, there are several disadvantages, including performance and durability issues. The following implementation is an alternative to the optocoupler. It uses a digital isolator, instead.

SMPS power converter designs depend on feedback from the output voltage to maintain regulation. This feedback signal typically passes through an optocoupler to maintain galvanic isolation between the primary and secondary sides. One of the key concerns when using an optocoupler, however, is that it introduces an extra pole in the control loop. This pole reduces the feedback path's bandwidth. In addition, temperature and lifetime degradation can affect the calibration of the control loop and, therefore, long-term drift.

The design in figure 3.4 shows an alternative using the *Si8642* digital isolator from Silicon Labs to form the barrier between the converter's primary and secondary sides. It requires

generating a feedback signal based on the converter's output. Notice that V_{IN} denotes the signal at the full-wave rectifier diodes output.

The feedback generator starts with a 10 MHz clock signal sent over the isolation barrier to the secondary side, where the circuit R_2 - C_2 converts the clock signal into a triangular waveform. This waveform drives high-speed comparator U3's inverting input. The non-inverting one receives a scaled value of the converter's output voltage. Whenever the triangular signal is smaller than the scaled output voltage, the comparator's output goes high. The comparator output signal's duty cycle will thus be proportional to the converter's output voltage.

The comparator's output passes through the isolation barrier back to the primary side, where circuit R_1 - C_1 low-pass filters the signal and applies that result to the switching controller's feedback pin.

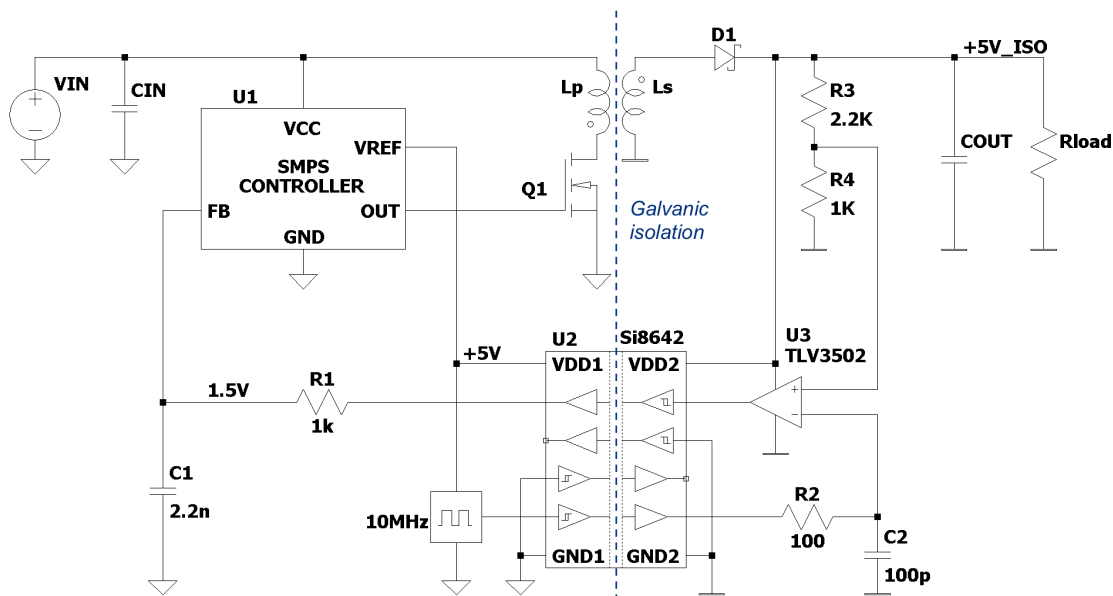


Figure 3.4 – *SMPS based on Si8642 digital isolator [2]*

3.1.3 Comparison table

In the next table 3.1, the comparison between both *SMPS* will be presented, defining also the chosen one.

As a conclusion, the *TinySwitch-III* *SMPS* option was available from the start and is the chosen one. It may not be the “best” option, but it cannot be forgotten, that this is a prototype project. So it was decided to take the closest choice. On the other hand, the up to 1 A this *SMPS* can provide is more than enough for the project consumption requirements.

	SMPS - Si8642 digital isolator	SMPS - TinySwitch
No. passive elements	9	19
Durability	better	worse
Availability	need to buy	available
Election	✓	✓

List of Tables 3.1 – SMPS comparison table [2], [1]

3.2 Communication Module

The HAGER power meter uses MODBUS communication protocol on top of RS485 communication. About this protocol, a brief introduction will be explained in A. After some research, the possible RS485 converter candidates found will be discussed in this section, in table 3.2.

	SN65HVD12D	THVD1500	MAX485
Mode	Half-duplex	Half-duplex	Half-duplex
Supply Voltage	3.3 V	5 V	5 V
Signaling Rates ^(*)	1 Mbps	500 Kbps	2.5 Mbps
Package	SOIC(8)	SOIC(8)	SOIC(8)
Price	3.18 €	1.11 €	1.71 €
Election	✗	✓	✓

^(*) The signaling rate of a line is the number of voltage transitions that are made per second expressed in the units bps (bits per second).

List of Tables 3.2 – RS485 converterts comparison table [10] [11] [4]

Finally, existing a general similarity among them, the chosen one was the MAX485 . This is the most extended one, with a large documentation throughout the internet, permitting a quite straightforward implementation. A picture of the converter module, with the integrated on it, can be seen in figure 3.5. The allocation and definition of the pins, taken from its datasheet, is shown in figure 3.6.

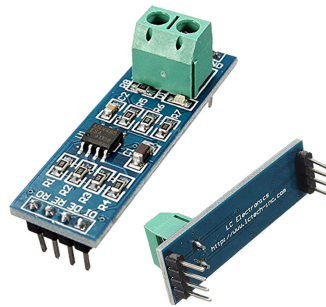


Figure 3.5 – RS485 to TTL converter used [3]

PIN					NAME	FUNCTION
MAX481/MAX483/ MAX485/MAX487/ MAX1487		MAX488/ MAX490		MAX489/ MAX491		
DIP/SO	μMAX	DIP/SO	μMAX	DIP/SO		
1	3	2	4	2	RO	Receiver Output: If A > B by 200mV, RO will be high; If A < B by 200mV, RO will be low.
2	4	—	—	3	\overline{RE}	Receiver Output Enable. RO is enabled when \overline{RE} is low; RO is high impedance when \overline{RE} is high.
3	5	—	—	4	DE	Driver Output Enable. The driver outputs, Y and Z, are enabled by bringing DE high. They are high impedance when DE is low. If the driver outputs are enabled, the parts function as line drivers. While they are high impedance, they function as line receivers if \overline{RE} is low.
4	6	3	5	5	DI	Driver Input. A low on DI forces output Y low and output Z high. Similarly, a high on DI forces output Y high and output Z low.
5	7	4	6	6, 7	GND	Ground
—	—	5	7	9	Y	Noninverting Driver Output
—	—	6	8	10	Z	Inverting Driver Output
6	8	—	—	—	A	Noninverting Receiver Input and Noninverting Driver Output
—	—	8	2	12	A	Noninverting Receiver Input
7	1	—	—	—	B	Inverting Receiver Input and Inverting Driver Output
—	—	7	1	11	B	Inverting Receiver Input
8	2	1	3	14	VCC	Positive Supply: 4.75V ≤ VCC ≤ 5.25V
—	—	—	—	1, 8, 13	N.C.	No Connect—not internally connected

NOTE: PIN LABELS Y AND Z ON TIMING, TEST, AND WAVEFORM DIAGRAMS REFER TO PINS A AND B WHEN DE IS HIGH. TYPICAL OPERATING CIRCUIT SHOWN WITH DIP/SO PACKAGE.

Figure 3.6 – MAX485 pin definitions [4]

3.3 Storage Module

To prevent having a loss of measured data from the power meter, a memory card will be included in the project. Various alternatives will be studied to look for the most adequate one: SD Card slot, microSD slot or using some kind of internal memory from the chip.

It is worth to point out the option of an **EEPROM** memory. **EEPROM** is a type of non-volatile memory that can be reprogrammed electrically. Taking this option could suppose to skip any external memory module, otherwise needed, if this would have enough capacity to store the information.

The more later chosen ESP-12 does not have **EEPROM**, through software is possible to configurate one though, taking part from the flash memory, with size between 4 and 4096 bytes[22].

	EEPROM	SD Card	microSD Card
Storage	4 - 4096 bytes	<i>desired</i>	<i>desired</i>
Dimensions ^(*) (length x width)	internal	14.2 x 14 mm	28 x 30.5 mm
Price	-	3.14€	3.15€
Election	✗	✓	✓

^(*) Size of card slot/holder, not the entire module.

List of Tables 3.3 – *SD memory cards analysis* -, [12], [13]

As a conclusion, it was decided to choose the option for an external memory card, rejecting the EEPROM due to a less than desired storage capacity. And between both cards, the smaller sized one will be the one incorporated to the project.

3.4 Interfacing Modules

The device will incorporate a small display with a simple menu showing relevant information to scroll through. It is relevant to point out, that both the screen and the scroll-through module feature will make no sense in the final product, they are only elements that have a place in this first prototype. For the final marketable product, some stages of refinement would have to be done, where these two components would make no sense. An illustration to show this point of view will be presented in figure 3.7.

Several options will be discussed to determine the best choice.

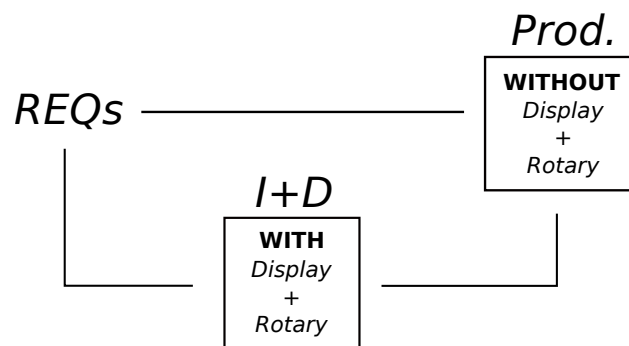


Figure 3.7 – *Diagram justifying the interfacing modules usage*

3.4.1 Display

The studied screens are listed in table 3.4. Each of them are shown in figure 3.8.

After each of them were analysed, the 128x64 OLED display will be the one incorporated to the project. I2C as unique communication interface is not really a restriction, as the ESP-12 chosen has an I2C bus that could be used for this screen. Dimensions are quite appropriate

	16x2 Display	128x64 Display	128x64 Display - two colors
Technology	LCD	OLED	OLED
Operating Voltage	5 V	3.3-5 V	3.3-5 V
Interface	SPI/I2C	I2C	I2C
Screen size	64.5 x 16 mm	21.7 x 11.2 mm	21.7 x 11.2 mm
Dimensions	80 x 35 x 11 mm	27.3 x 27.3 x 2 mm	27.3 x 27.3 x 2 mm
Consumption	~20 mA	LED dependent, avg. 15 mA	LED dependent, avg. 15 mA
Price	9.17€	2.21€	2.27€
	✗	✓	✓

List of Tables 3.4 – Display comparison table [14], [15], [16]

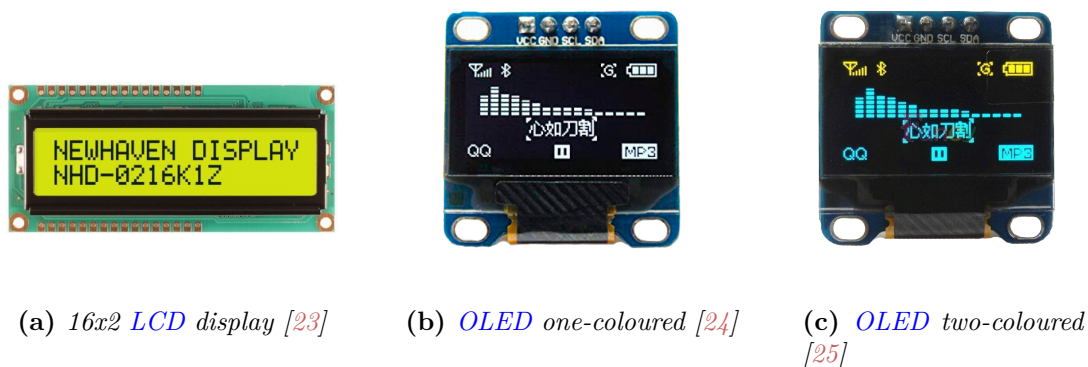


Figure 3.8 – Different displays compared

for the enclosure the product will be housed in and the two-coloured one provides a better-looking appearance than the just white option.

3.4.2 Scroll through menu

To be able to scroll through the implemented menu, there are two general ways: via push buttons, e.g. three differentiated buttons for going forwards, backwards and selection; or a rotary encoder.

Considering the scenario with both options placed on the device, the rotary encoder is concluded to be the preferred one, being more aesthetic and eventually taking less area than the three buttons. In addition, it will be a rotary with push feature, thus no extra button needed. An illustration of this module is shown in figure 3.9.

3.4.2.1 Rotary working principle

A rotary encoder is a variant of position sensor used for calculating the angular position of a rotating shaft. It generates a signal according to the rotational movement. There are

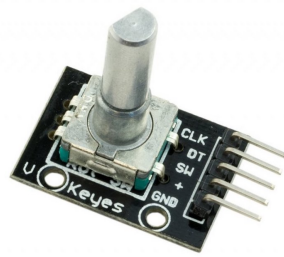


Figure 3.9 – *Rotary Encoder used [5]*

several types of rotary encoders, the one used here is an incremental rotary encoder whose output is a series of square wave pulses.

The working mechanism is shown in figure 3.10 in a precise way. It has a cogged disk with evenly spaced contacts connected to common pin C. As well as two other separated contacts A and B. The disk rotates step by step generating two output square waves due to the contact making of both pins with the common one.

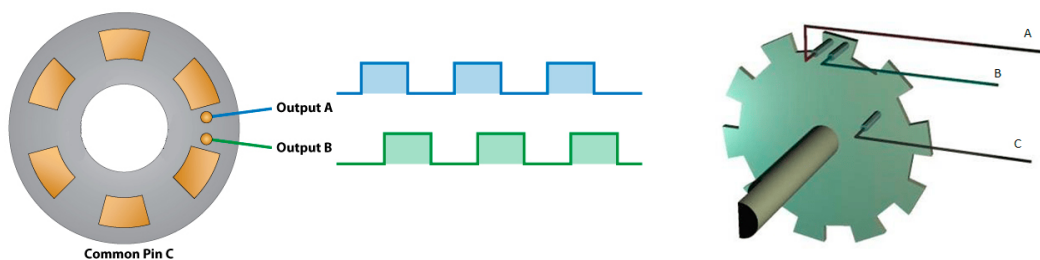


Figure 3.10 – *Rotary Encoder close-up [6]*

To determine the rotation direction as well (because the rotated steps can be determined just counting each pulse) both signals at the same time have to be considered. In figure 3.11, it can be noticed that both signals are 90° phase shifted with each other. When the encoder rotates clockwise, the output A is ahead, so both signals will have opposite values and the changes from *High* to *Low* or viceversa can be counted. In a similar way, on counter-clockwise rotation B will be ahead. So monitoring both the number of pulses and the relative phase of the signals, it is possible to track both the position and direction of rotation. An illustrating representation of this scenario can be appreciated in figure 3.11.

3.5 MCU

Various alternatives were analysed to determine the best MCU choice to adapt to the project requirements. Arduino UNO, Rapsberry Pi 3, BeableBone Black Board, ESP32 and ESP-12 will be the candidates to study about.

The right one will have to adapt to the number of ports to use, which are listed below:

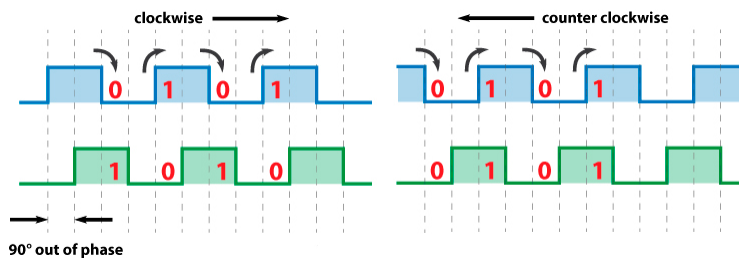


Figure 3.11 – Rotary Encoder signals [6]

- RS485 Communication: 1 [UART](#) bus + 1 I/O pin.
- Flash/Debug port: 1 [UART](#) bus.
- [OLED](#) Display: 2 I/O pins.
- Rotary Encoder: 3 I/O pins.
- microSD card: 1 HSPI bus.

3.5.1 Arduino UNO

Arduino is a nice and simple choice for [DIY](#) projects like this one. Its facilities to connect LEDs, sensors, motors, etc directly to the board makes it a really extended-use device. To program it, the company also provides the software needed, totally free and as simple as connect it to a PC via USB port and upload the code from their [API](#). An image of the Arduino is shown in [3.12](#). It is definitely a viable solution, with attractive characteristics in operation voltage, programmable pins, flash, RAM, oscillator frequency and size, among others.



Figure 3.12 – Arduino UNO [7]

3.5.2 Raspberry Pi 3 & BeagleBone Black

Both cases would be to choose a device with computer features, so more than enough resources for this project. The key point that put these devices as candidates was its WiFi functionality, without having to use external modules. Both boards are presented in figure [3.13](#).

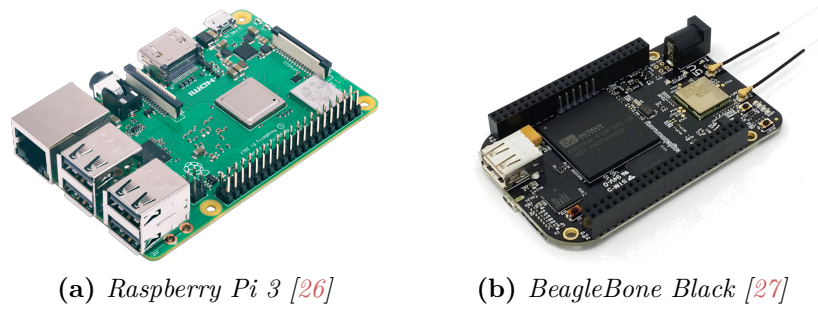


Figure 3.13 – *PC-functionality boards*

3.5.3 ESP32

The ESP32 is a chip with WiFi and Bluetooth integration at 2.4GHz , along with high efficiency dual cores, ultra-low power co-processor and several peripherals. It provides a robust and high integrated platform, ideal for currently demands on efficient consumption, security, high density and great performance. On figure 3.14a the module can be seen.

3.5.4 ESP12

The ESP-12 is a WiFi module controlled by the ESP8266 designed for mobile platforms with space and consumption limitations, with features as full TCP/IP stack. It is nowadays one of the leading WiFi chips in integration matter from the industry. It can act as a WiFi adapter to any connected device through its various [GPIOs](#), with [SPI](#), [I2C](#) and [UART](#) interfaces among others. The module is shown in figure 3.14b.

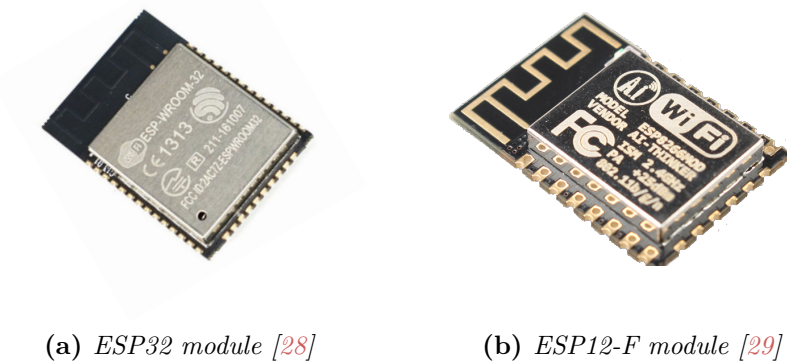


Figure 3.14 – *Modules with integrated WiFi functionality*

3.5.5 Comparison table

All candidates will be put together in this section, emphasizing its features in the table 3.5 down below.

	Arduino UNO	Raspberry & BeagleBone	ESP32	ESP12
Microcontroller	ATmega328	based on ATmega644 & 2x PRU	Tensilica Xtensa 32-bit Dual Core, chip ESP8266EX	Tensilica Xtensa 32-bit Single Core, chip ESP32-D0WDQ6
Operating Voltage	5 V	5 V	3.3 V	3.3 V
GPIO	14	40 & 66(3.3V) [30]	36	17
DC Current per I/O Pins	20 mA	16 mA & 4-6 mA	12 mA	12 mA
Flash Memory	32 KB - 0.5 KB bootloader	- & 2 GB	up to 16 MB	up to 4 MB
SRAM	2 KB	1024 MB & 512 MB	520 KB	80 KB
EEPROM	1 KB	- & 32 KB	-	-
Clock Speed	16 MHz	1.4 GHz & 1 GHz	up to 240 MHz	up to 160 MHz
WiFi integrated	No	Yes	Yes	Yes
Dimensions (length x width)	68.6 x 53.3 mm	85 x 56 mm & 86.4 x 53.3 mm	25.5 x 18 mm	24 x 16 mm
Price	20€	33.9€ & 70.95€	3.32€	3.32€
	X	X	✓	✓

List of Tables 3.5 – *MCU comparison table* [17], [18], [19], [20], [21]

Remarking table 3.5, the module selected will be the *ESP12*. It can be noticed that candidate ESP32 is superior in all characteristics to the ESP12, having as only drawback its price and size (which is very logical for getting better features). Thus, this option could have been the one elected. However, the ESP-12 was the chosen one for the main reason that there was already one available to start with. As the other characteristics were not essential (the number of pins was going to be a critical issue, it will be discussed later) and the facility of beginning right then, the ESP-12 was the choice opted for. Besides that, due to its recent release, the ESP32 could present more incompatibilities with the firmware, apart from having less information about the device available on the internet.

The Arduino or either of the *SBCs* Raspberry Pi and BeagleBone would not have been an efficient decision for the requirements this project has, being its price quite substantial, inappropriate dimensions and excessive characteristics comparing to the other two.

3.6 Tests to fulfill

In this section, the key points to analyse for a correct operation will be described. It would be the aspects to check in the verification section 5 later on.

The tests will be arranged according to the elements used:

1. **Power Module:** To check proper operation, the power supply shall:

- Be able to draw a proper 5 V output.
- Do not get strangely hot (i.e. do not burn oneself).
- Do not make any kind of noise due to wrong contacts.

2. **SD Card:** To ensure good software design, the memory module shall work this way:

- Write data only when connection establishment with the server is not possible.
- After writing on SD card due to connection loss, send the data that failed in the last try on next connection establishment, along with the new requested data.
- Take care of not overwriting data if connection loss lasts for a while.
- Data will have a certain format.

3. **Communication Module:** To be properly implemented, the COM module shall:

- Ask the power meter for new data in the period established.
- Drive the enable pin for communicating correctly, ensuring *real* HIGH and LOW values (in this case, 5 V and 0 V).
- Perform the requirements of the **MODBUS** communication protocol so the power meter understands.

4. **Display + Rotary:** An appropriate operation of the display and rotary encoder will be considered if it fulfills:

- The screen has to show up the menu correctly.
- When rotating the shaft, the selection box has to change accordingly to the new current menu item with a decent delay, permitting an appropriate user experience.

5. **MCU - ESP-12:** To confirm a correct implementation, the ESP-12 shall be able to:

- Coordinate all modules connected to it without crashing.
- Establish connection with the configured WiFi ID.
- Dump data properly to the server.

6. **Remote Server:** To ensure correct design, the remote server shall:

- Receive the data in a certain format.
- Save it to a local file.
- Interact with the client through “ACK” messages, for debugging .

3

CHAPTER

4

SYSTEM DESIGN

In this chapter the design and implementation will be faced, giving a solution to the system requirements mentioned in chapter 2. It will go through a detailed study about the hardware and software design.

The aim of this entire part is to materialize a system that fullfills the block diagram presented in figure 4.1.

4.1 Electronic Design

This section will deal with the interconnections between the different modules and the modules itself. The design was planned to eventually be implemented on a PCB, capable of housing all needed components and routing them to be a fully function circuit. In order to get to this point, *Altium Designer 2016* was used as the EDA software tool in this project, offering really important features concerning schematic and PCB design, as well as including a 3D visualization option of the board, allowing a relevant feedback of the appearance itself.

The board dimensions will be specified by the enclosure that will house it. The design will be developped using the *2-layer* technology, getting a taste of the manufacturing and working principle of PCBs, even if going on two layers wasn't necessarily a space matter.

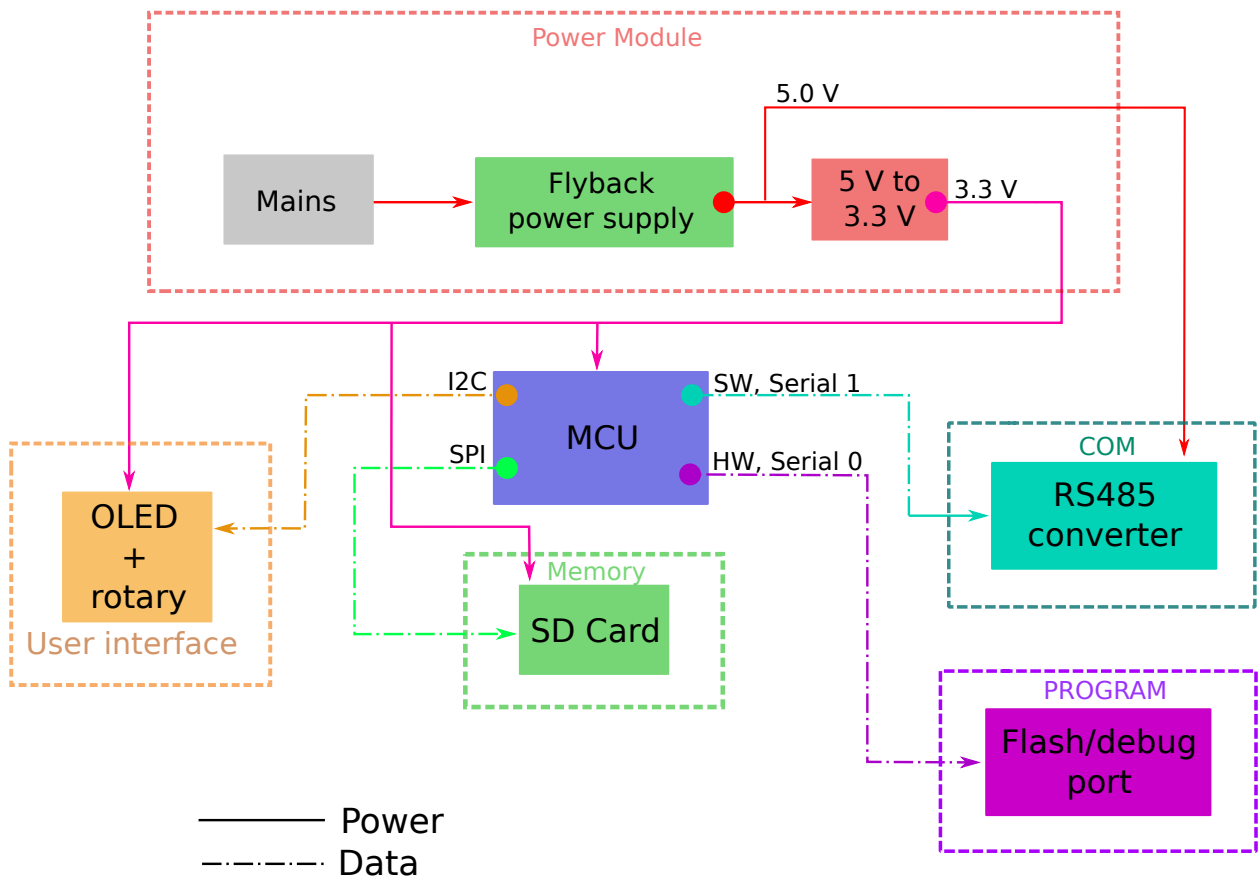


Figure 4.1 – *Final block model of the design*

4.1.1 PCB design

The complete design process as well as the design rules of the PCB will be presented in this section. More specific issues about module connections and their integration in the project will be addressed during the next sections.

The company JLCPCB [31] dealt with the board manufacturing. Just uploading the required files for the production on their website and for a reasonable price the order arrives after one and a half weeks or two. The design rules applied to the board are shown in table 4.1, keeping in mind the manufacturer's critical design rules provided on the website. For a clearer compression of the facts, a representative drawing can be seen in figure 4.2.

PCB design rules			
Default		Power	
Track width	0.254 mm	Track width	1.016 mm
Clearance	0.127 mm	Clearance	0.127 mm
Via hole size	0.4 mm	Via hole size	1 mm
Via diameter	0.7 mm	Via diameter	1.5 mm

List of Tables 4.1 – Design rules for chemical reaction PCB manufacture

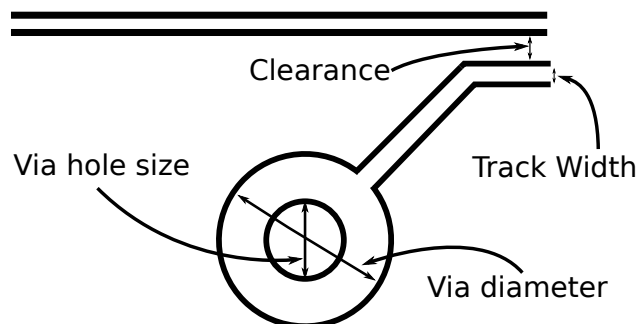


Figure 4.2 – Explanation of PCB rules

The final schematics of the project can be observed next:

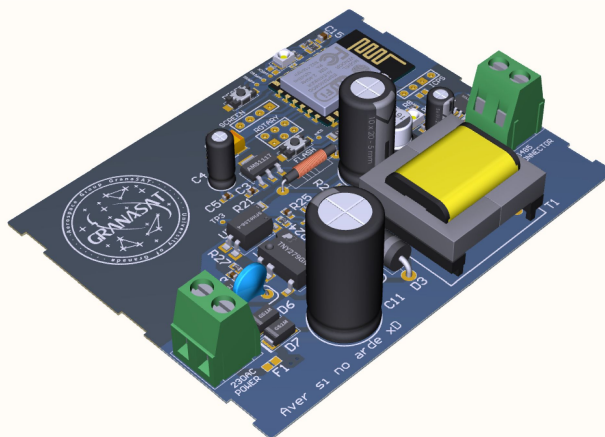
Power Meter Telemeassurement based on ESP8266

Variant: Prototype

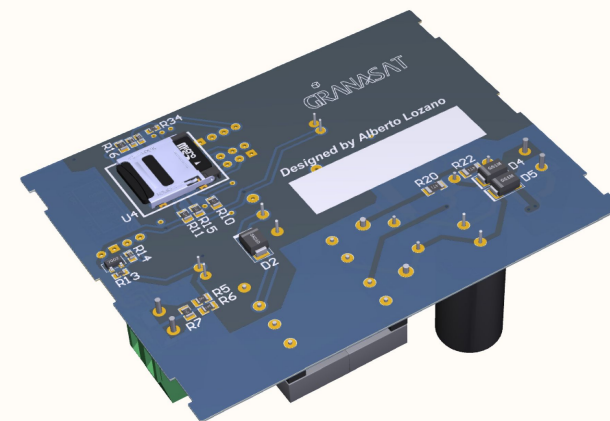
V.05

Page Index

Page	Index
1	COVER PAGE
2	COMMUNICATION MODULES AND RELATED
3	POWER SUPPLY
4	PERIPHERIALS
5	ESP-12 MODULE



TOP VIEW



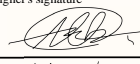


BOTTOM VIEW

DESIGN CONSIDERATIONS

DESIGN NOTE:
Example text for debug notes.

DESIGN NOTE:
Example text for explanatory notes.

DESIGN NOTE:
Example text for critical design notes.

Designer's signature 	Sheet title: COVER PAGE.SchDoc	<div>Dpto. Electrónica y Tecnología de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andrés Roldán Aranda</div> 
Supervisor's signature 	Project title: POWER METER TELEMEASSUREMENT BASED ON ESP8266	
	Designer: LOZANO ALTHAMMER, ALBERTO	
	Date: 05.11.2018	Revision: 05
	Sheet 1 of 5	

HAGER MULTIMETER

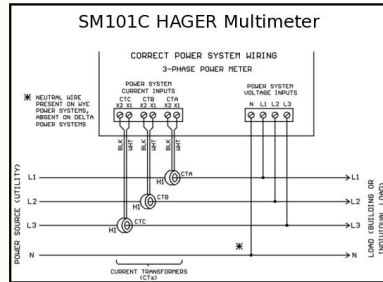
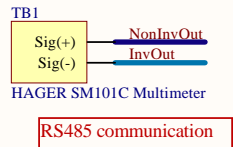
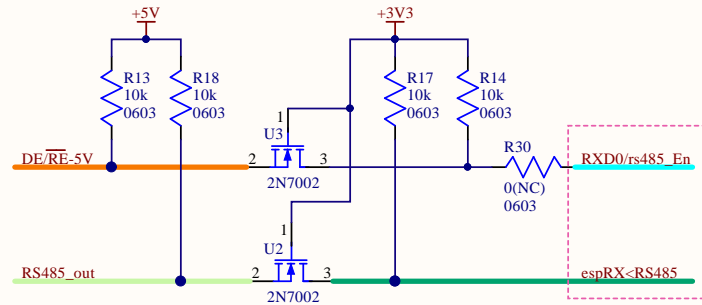


FIG 4. Power meter clarification diagram

LOGIC LEVEL SHIFTER 5V<=>3V3



DESIGN NOTE:
RS485 converter has 5V logic and ESP-12F
supports 3.3V (also be aware of the
DE/REneg pin)

VOLTAGE REGULATOR

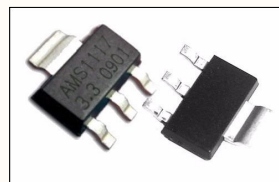
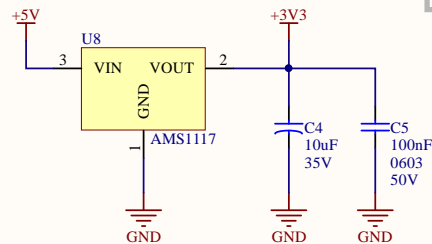
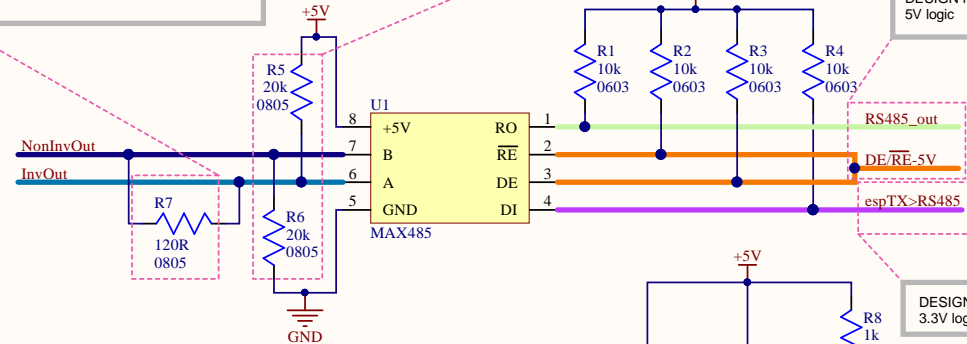


FIG.3 AMS1117 3.3V voltage regulator

RS485 TO TTL CONVERTER

DESIGN NOTE:
R7 for matching load with circuit impedance
(Rt in FIG.2).



DESIGN NOTE:
Resistors pull-up/down in connection with
HAGER Multimeter (R5 & R6) are bigger
because of the higher voltage values from
differential communication in rs485.

DESIGN NOTE:
5V logic

DESIGN NOTE:
3.3V logic

DESIGN NOTE:
Goes on in power-up.

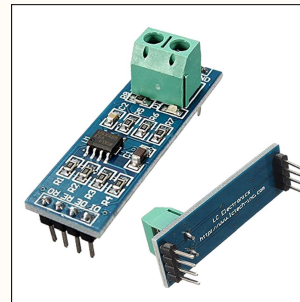


FIG 1. RS485 device appearance

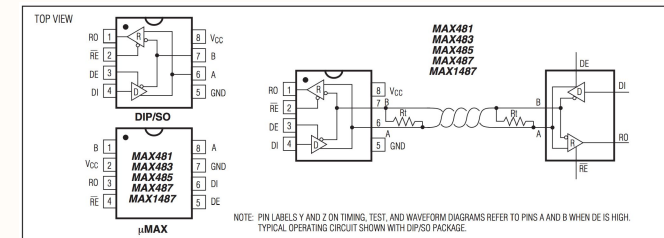
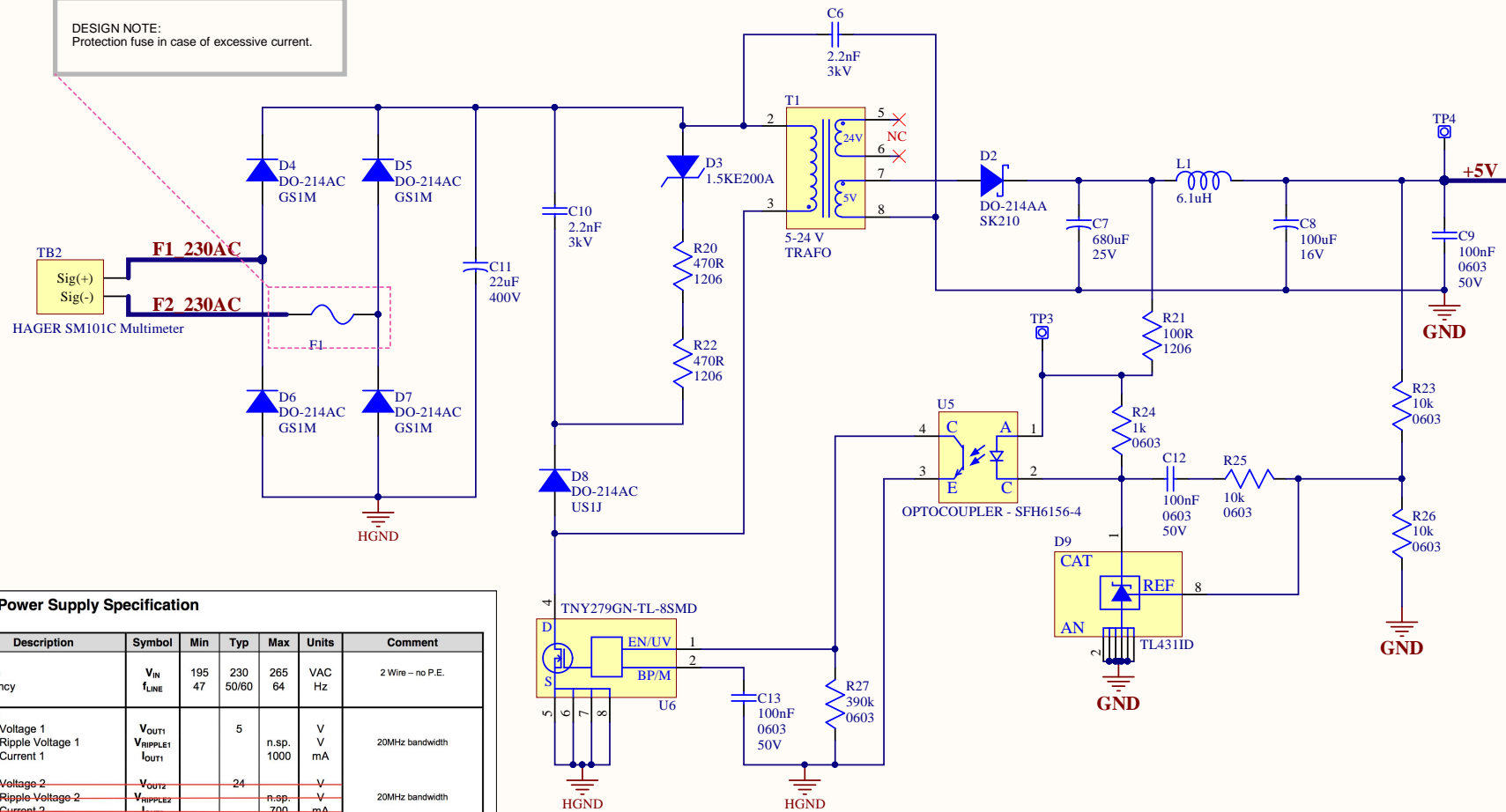


FIG 2. MAX481/MAX483/MAX485/MAX487/MAX1487 Pin Configuration and Typical Operating Circuit

Designer's signature 	Sheet title: COMMUNICATION MODULES AND RELATED.SchDoc	Dpto. Electrónica y Tecnología de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andrés Roldán Aranda
Supervisor's signature 	Project title: POWER METER TELEMEASUREMENT BASED ON ESP8266	
	Designer: LOZANO ALTHAMMER, ALBERTO	
	Date: 05.11.2018	Revision: 05
	Sheet 2 of 5	



DESIGN NOTE:
Protection fuse in case of excessive current.



2 Power Supply Specification

Description	Symbol	Min	Typ	Max	Units	Comment
Input						
Voltage	V_{IN}	195	230	265	VAC	2 Wire - no P.E.
Frequency	f_{LINE}	47	50/60	64	Hz	
Output						
Output Voltage 1	V_{OUT1}		5		V	20MHz bandwidth
Output Ripple Voltage 1	$V_{RIPPLE1}$			n.sp.	V	
Output Current 1	I_{OUT1}			1000	mA	
Output Voltage 2	V_{OUT2}		24		V	20MHz bandwidth
Output Ripple Voltage 2	$V_{RIPPLE2}$			n.sp.	V	
Output Current 2	I_{OUT2}			700	mA	
Total Output Power			5		W	
Continuous Output Power	P_{OUT}		47		W	
Efficiency					%	Measured at full load 25 °C
Full Load	η	n.sp.			%	
Environmental						
Conducted EMI			Meets CISPR22B / EN55022B			
Ambient Temperature	T_{AMB}	0		50	°C	Adapter, no air flow, sea level

FIG.1 Power supply specifications

Designer's signature 	Sheet title: POWER SUPPLY.SchDoc	Dpto. Electrónica y Tecnología de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andrés Roldán Aranda
Supervisor's signature 	Project title: POWER METER TELEMEASUREMENT BASED ON ESP8266	
	Designer: LOZANO ALTHAMMER, ALBERTO	
	Date: 05.11.2018	Revision: 05
	Sheet 3 of 5	



microSD CARD

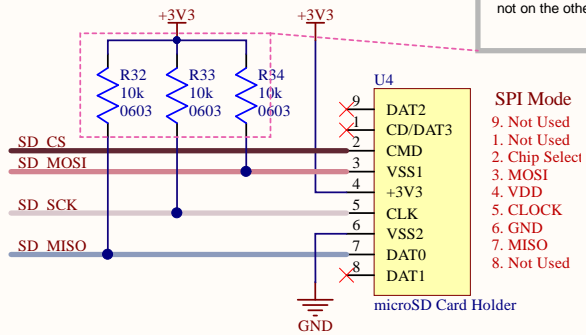


FIG.1 microSD Card holder

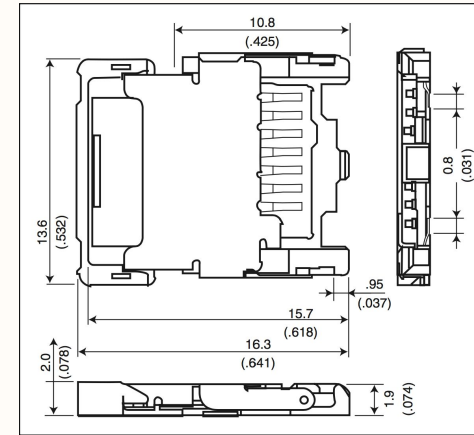


FIG.2 microSD Card holder dimensions - mm (in)

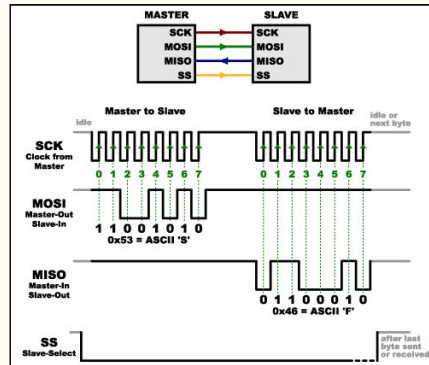


FIG.3 Read/write working principle in SPI interface

DESIGN NOTE:
Just before data is sent to the slave, the line is brought low, which activates the slave. When it's done using the slave, the line is made high again.

DESIGN NOTE:
To determine either increment and direction of rotation both signals have to be analysed.

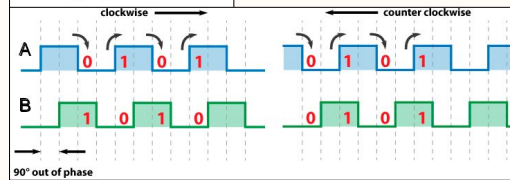
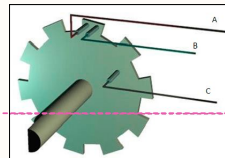
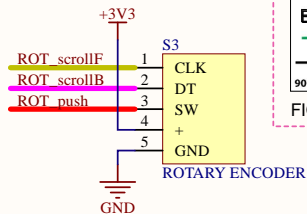


FIG.5 & 6 Rotary encoder close up and output signals

ROTARY



FIG. Rotary encoder module



LCD DISPLAY - I2C

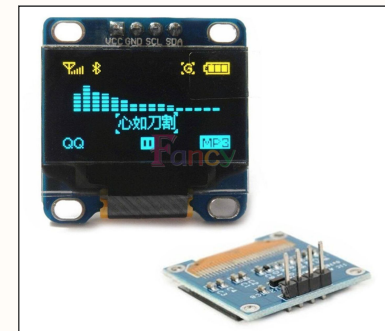
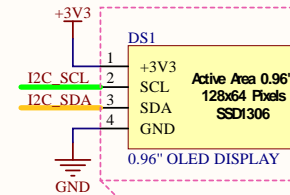


FIG.2 microSD Card holder dimensions - mm (in)

DESIGN NOTE:
(from a similar display, due to minimal info from actual seller)

- Can be powered either by 3.3V or 5V.
- Compatible I/O level: 3.3V / 5V.
- Driver: SSD1306

<https://www.hotmcu.com/ssd1306-096-128x64-oled-display->

Designer's signature 	Sheet title: PERIPHERALS.SchDoc	Dpto. Electrónica y Tecnología de Computadores University of Granada C/ Fuente Nueva, s/n, 18001 Granada, Granada, Spain Sr. Andrés Roldán Aranda
Supervisor's signature 	Project title: POWER METER TELEMESUREMENT BASED ON ESP8266	
	Designer: LOZANO ALTHAMMER, ALBERTO	
	Date: 05.11.2018	Revision: 05
	Sheet 4 of 5	



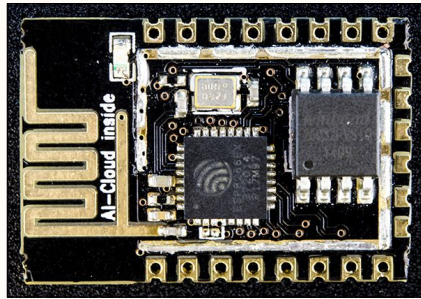


FIG.1 Internal look-up of ESP-12 module

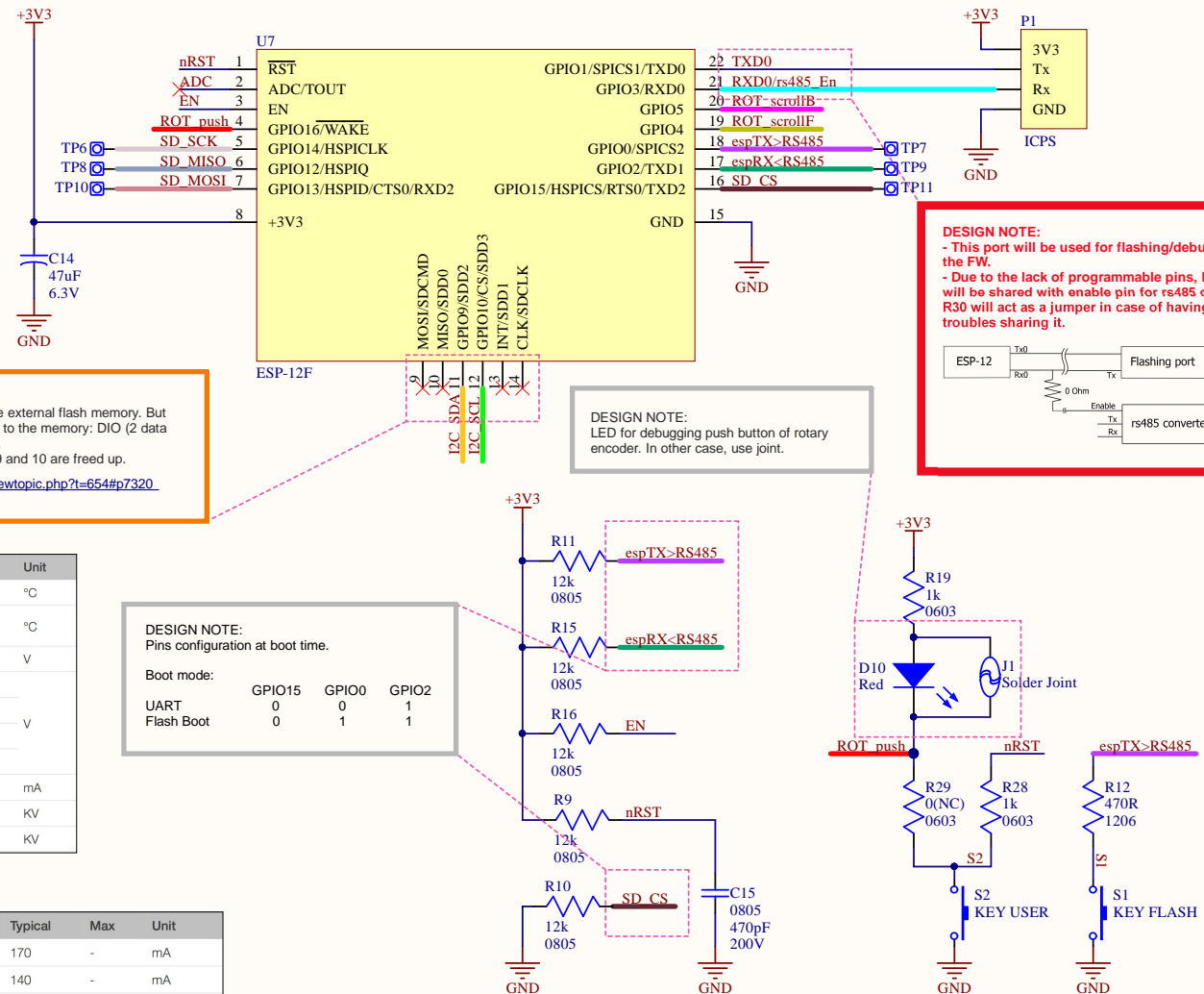
DESIGN NOTE:
GPIOs 6-11 used to interface external flash memory. But there are 2 modes of access to the memory: DIO (2 data lines) and QIO (4 data lines).
If selected DIO mode GPIO9 and 10 are freed up.
<https://bbs.espressif.com/viewtopic.php?t=654#p7320>

Parameters	Conditions	Min	Typical	Max	Unit
Operating Temperature Range	-	-40	Normal	125	°C
Maximum Soldering Temperature	IPC/JEDEC J-STD-020	-	-	260	°C
Working Voltage Value	-	2.5	3.3	3.6	V
I/O	V_{IL}	-0.3	-	0.25V _{IO}	
	V_{IH}	0.75V _{IO}	-	3.6	
	V_{OL}	-	-	0.1V _{IO}	
	V_{OH}	0.8V _{IO}	-	-	
	I_{MAX}	-	-	12	mA
Electrostatic Discharge (HBM)	TAMB=25°C	-	-	2	KV
Electrostatic Discharge (CDM)	TAMB=25°C	-	-	0.5	KV

FIG.2 Electrical characteristics

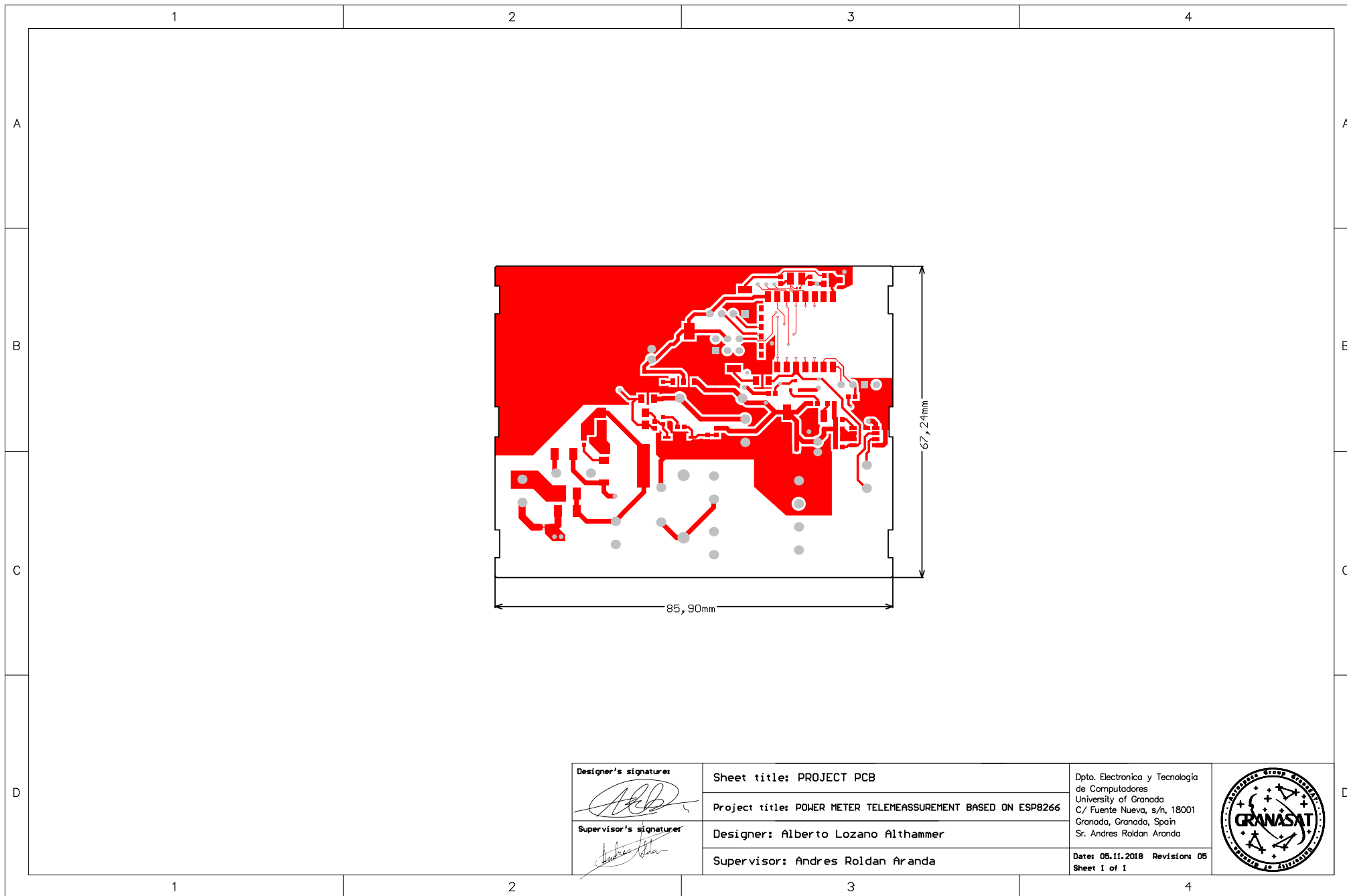
Parameters	Min	Typical	Max	Unit
TX 802.11b, CCK 11Mbps, P _{OUT} =+17 dBm	-	170	-	mA
TX 802.11g, OFDM 54Mbps, P _{OUT} =+15 dBm	-	140	-	mA
TX 802.11n, MCS7, P _{OUT} =+13dBm	-	120	-	mA
Rx 802.11b, 1024 bytes packet length, -80 dBm	-	50	-	mA
Rx 802.11g, 1024 bytes packet length, -70 dBm	-	56	-	mA
Rx 802.11n, 1024 bytes packet length, -65 dBm	-	56	-	mA

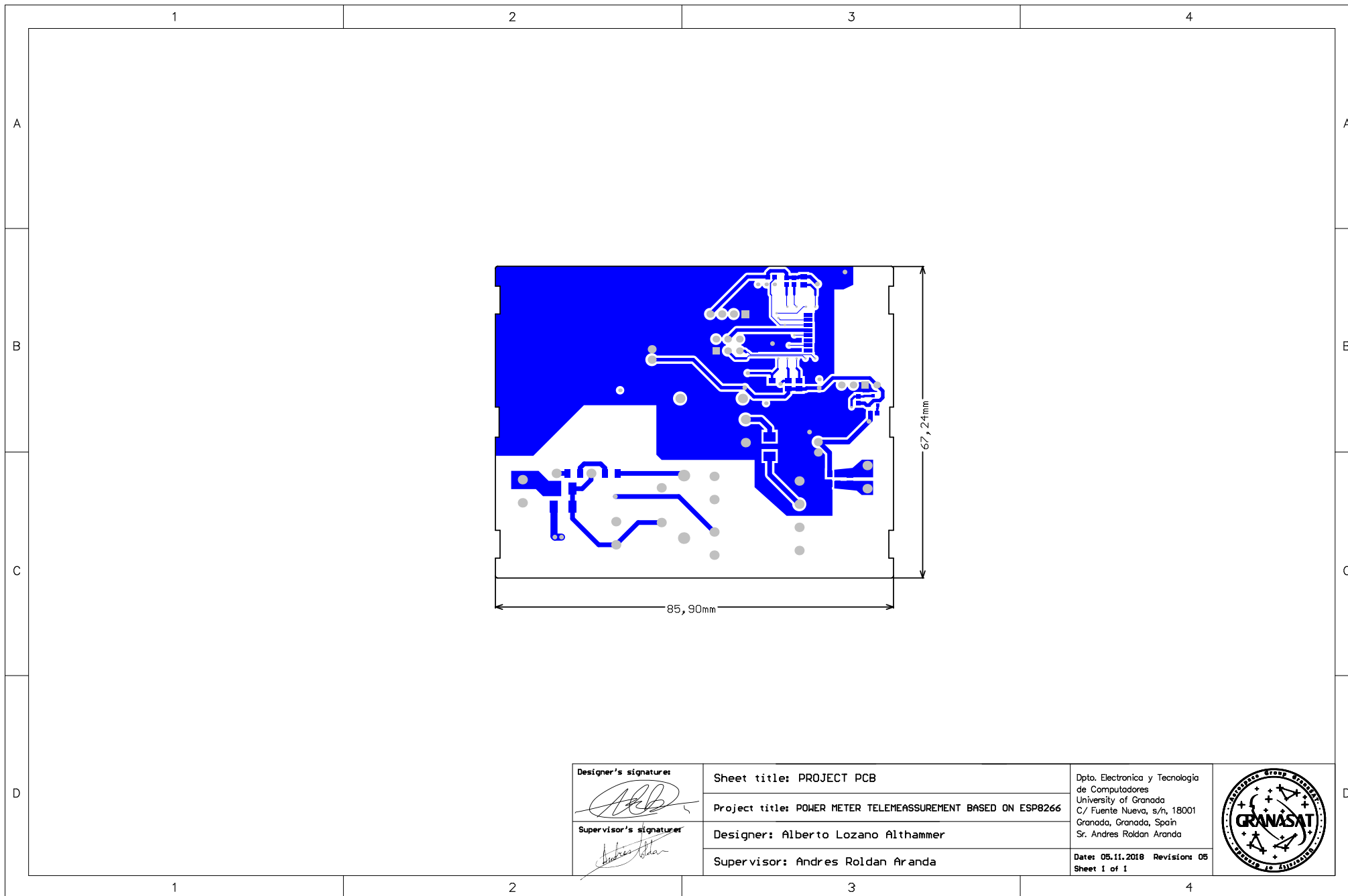
FIG.3 Power consumption active



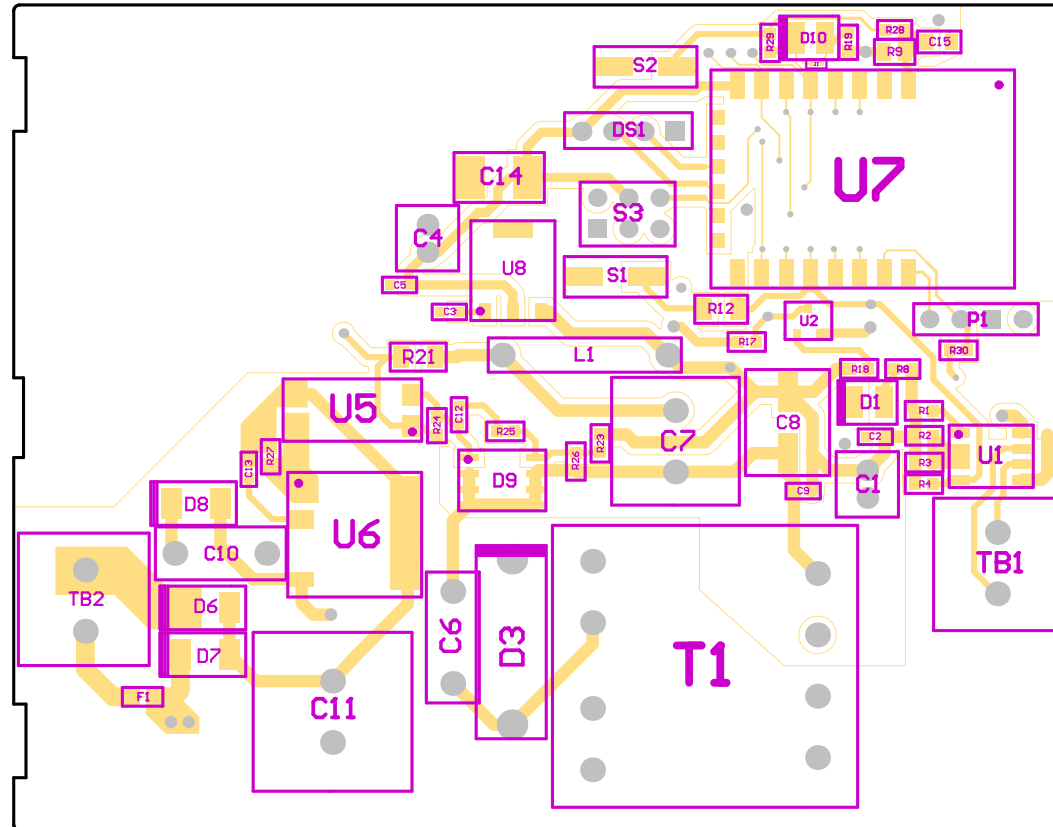
Designer's signature	Sheet title: ESP-12F MODULE CONNECTIONS.SchDoc	Dpto. Electrónica y Tecnología de Computadores
Supervisor's signature	Project title: POWER METER TELEMEASUREMENT BASED ON ESP8266	University of Granada
	Designer: LOZANO ALTHAMMER, ALBERTO	C/ Fuente Nueva, s/n, 18001
	Date: 05.11.2018	Granada, Granada, Spain
	Revision: 05	Sr. Andrés Roldán Aranda
	Sheet 5 of 5	



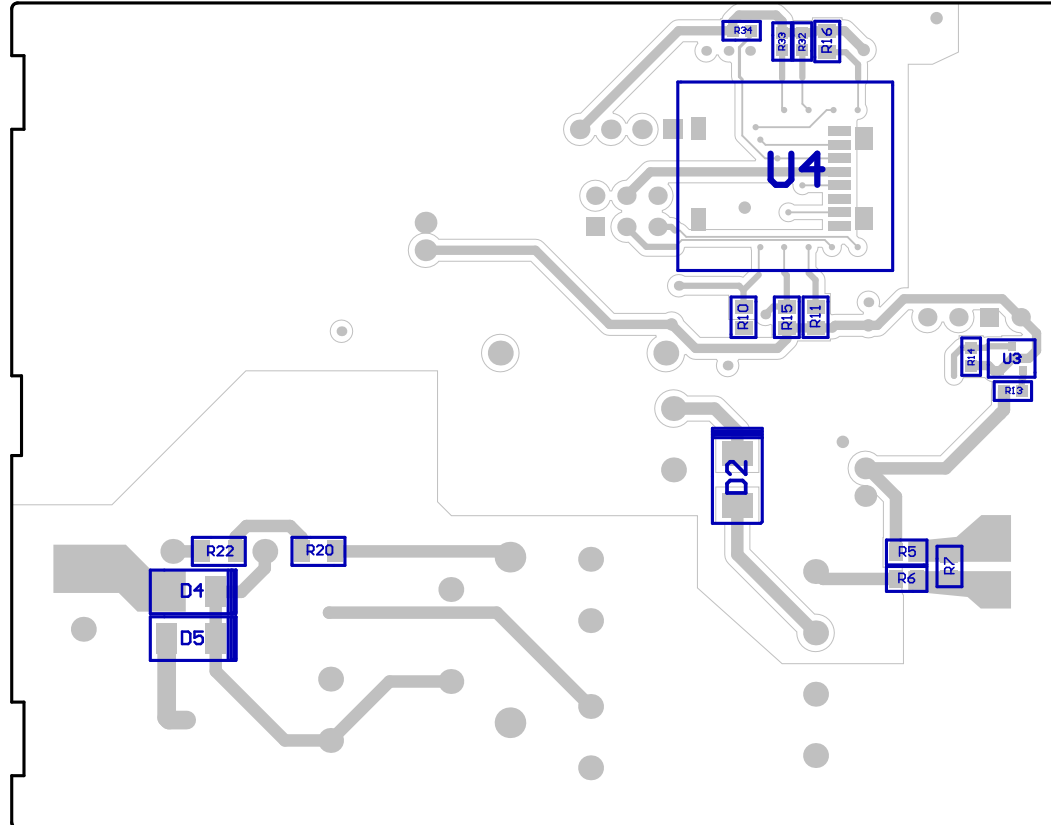


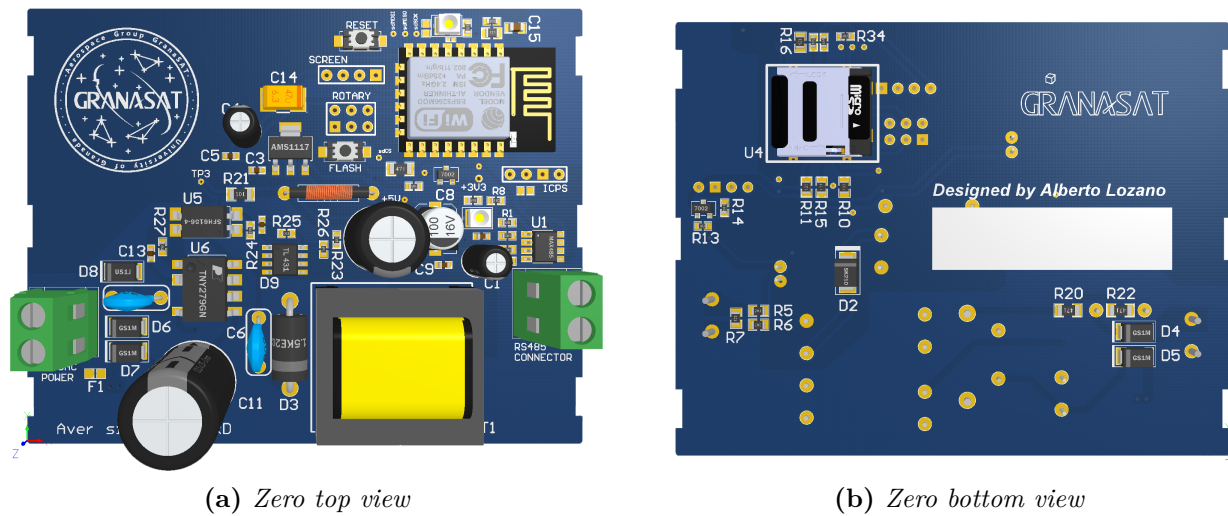


TOP LAYER ASSEMBLY



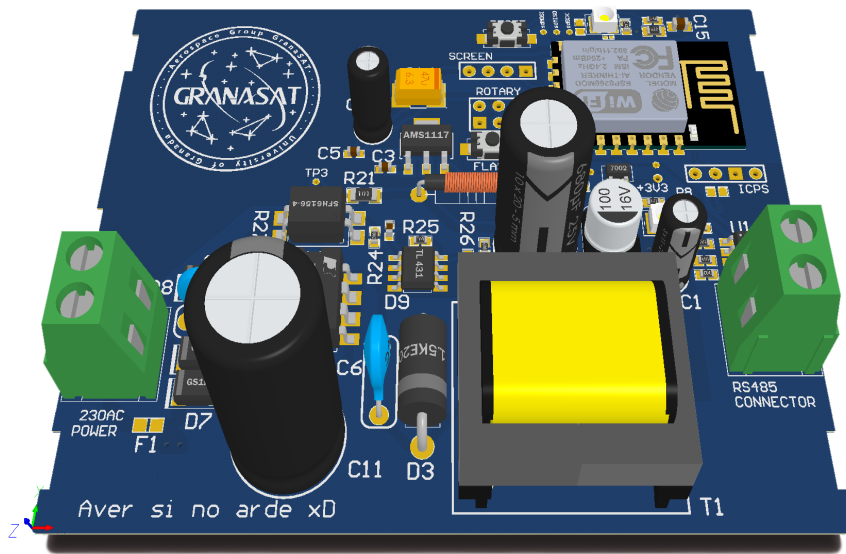
BOTTOM LAYER ASSEMBLY



Figure 4.3 – *PCB's 3D model*

4.2 Software Design

In this section, the software built will be analysed. It will be faced in a modular approach, studying the code implemented for each module. For flashing the device the Arduino IDE



- * Open with [Acrobat Reader](#) for playing the video.
- * To stop video: click and go out of frame holding.

4

Video 4.1 – 3D video of the *PCB**

will be used, being an easy-to-work and really extended platform in this kind of DIY projects.

4.2.1 Activity diagram

Activity diagrams are useful to split a complex process into several simpler parts. The device is introduced in figure 4.4.

The device will first initialize all the corresponding modules. After that, new data measured by the power meter will be asked and try to connect to the server. If confirmed, data will be sent. If not, it will be written to the SD card to prevent information loss.

4.2.2 RS485 Communication

Firstly, the power meter used has to be configured in order to know the settings on how to establish communication. To do so, configuration mode is selected from the little user screen interface from meter and set the needed values for the RS485 communication to know packet structure and be able to encode and decode correctly. The set values are shown in table 4.2.

To ensure communication is possible with the power meter, a simple python *script* was written due to the ease of this programming language, put some values and wait to responses

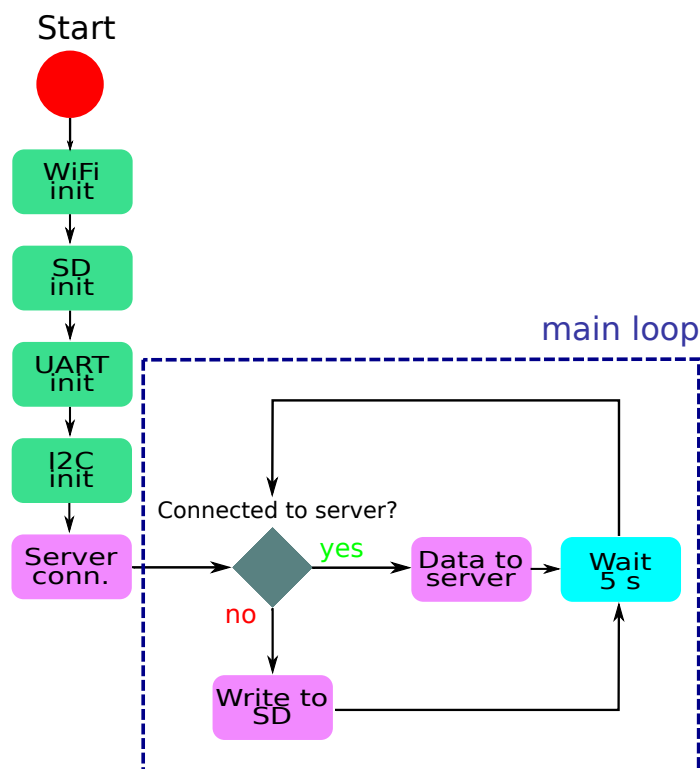


Figure 4.4 – Main activity diagram

SM101C - HAGER Power Meter	
Address nr.	1
Baud rate	9600 bps
Parity	None
Stop bits	1

List of Tables 4.2 – SM101C - HAGER Multimeter settings

(remember that ESP-12 didn't support python, it is just to receive responses from the meter for the first time). A [RS485-RS232](#) converter along with an [RS232-USB](#) adapter was used to connect the device to the PC. The adapters are presented in figure 4.5.

To compile the code, the python package has to be installed on the computer. To check if already installed, it can be done by typing the following code in a terminal window:

If the answer message gives the version correctly back, it is ready. If error messages come up, it may be needed to download and install the package [32].

In addition, the package *pip* is required to install the communication library during next step. For this, run the code below:

Now, it would be ready to go and type the desired code on any text editor, saving the file as *name.py* and running it from terminal. The implemented code is shown in figure 4.6.



Figure 4.5 – *RS485-RS232 adapter + USB*

```

1 | $ python -V
2 |
3 | Python 3.6.4 :: Anaconda, Inc.

1 | $ sudo easy_install pip
2 |
3 | $ pip install minimalmodbus

```

To get the information for the functions needed, the [API](#) of MinimalModbus [\[33\]](#) was visited. For simplicity, the settings set to the power meter in [table 4.2](#) above were the values that this library had by default. In this case, only the `read_registers` function will be used.

To figure out which registers store the desired values, the meter datasheet has to be looked up. It can be found in [chapter B](#).

Next step will be to run the code in terminal, seen in [figure 4.7](#).

```

1 | $ python test.py
2 |
3 | MinimalModbus debug mode. Writing to instrument (expecting 7
   | bytes back): (01 03 C9 01 00 01 EA 56)
4 | MinimalModbus debug mode. No sleep required before write. Time
   | since previous read: 1542035370151.1 ms, minimum silent period:
   | 4.01 ms.
5 | MinimalModbus debug mode. Response from instrument: (01 03 02 00
   | 18 B8 4E) (7 bytes), roundtrip time: 25.8 ms. Timeout setting:
   | 50.0 ms.
6 |
7 | 24 degrees Celsius

```

Figure 4.7 – *Response from power meter through the python code*

Because of the `instrument.debug = True` command from code in [figure 4.6](#), detailed information will be printed about the request and response, shown in [figure 4.7](#), which is really useful for checking if everything is right. After confirming on the LCD screen, integrated in the power meter itself, the answer was correct, the first step from the

```

1 #!/usr/bin/env python
2 # modbus RTU python code for talking to HAGER power meter
3 import minimalmodbus
4 # consider an instrument (slave) with addr. 1 to which communicate via
   serial port '/dev/ttyUSB1'
5 instrument = minimalmodbus.Instrument('/dev/tty.usbserial', 1) # port
   name, slave address (in decimal)
6 instrument.debug = True # communication details are printed
7 instrument.serial.baudrate = 9600 # Baud
8 # read_registers(registeraddress, numberOfRegisters, functioncode=3)
9 value = instrument.read_registers(51457, numberOfRegisters = 1)
10 # for checking it will be used reg. 51457 = TEMPERATURE, as it is
   pretty const.
11 print(value[0], "degrees Celsius") # read_registers returns a list of
   int, take wanted

```

Figure 4.6 – *Implemented python code*

‘debugging process’ that will be followed is done.

Next stop will be to write the code in the language the MCU supports: *C Programming Language*. At this point, it will be taken advantage of the information the python code from figure 4.6 printed about the packets. As the targets of the requests will be always the same ones (the same measurements will be constantly asked to the power meter), the packet will be directly implemented into the C code, replacing the need of libraries to form the request.

The UART port used for the RS485 communication will be set on pins GPIO 0(Tx) and GPIO 2(Rx). As it is not the ESP-12 hardware serial port, it will be configured via software (Software Serial Port). Hardware UART0 will be left free for flash/debugging purposes.

4.2.3 OLED Display + Rotary Encoder

The little screen attached to the device will be connected to the ESP-12 via I2C interface. GPIO pins 9 and 10 will be set as SDA and SCL respectively, changing the default allocation of these pins. The purpose of the display will be to show relevant information about the requesting data process such as WiFi signal strength or remote server IP address. To make it more user-friendly a simple menu will be implemented to present this information, along with a rotary encoder to scroll through it.

Considering interesting the way information will be printed on the display, the relevant parts of the code will be next exposed.

The rotary-updating-position code is based on the two functions stated in figure 4.8.

```

1 //in the main loop, after conditional for requesting data from
  power meter
2 // ...
3   setInputFlags(); //sets flag HIGH for the input pin that
    changed state
4   resolveInputFlags(); //checks flags array and updates
    accordingly menu current state
5 // ...
6 }

```

Figure 4.8 – Rotary-position-updating code

This piece of code will be placed in the main loop, right after the conditional block to ask new data to the power meter. It will always run on each cycle to ensure decent user experience when rotating the rotary (meaning that the scrolling functionality will have “real-time” feeling). In figure 4.9 is shown this scenario.

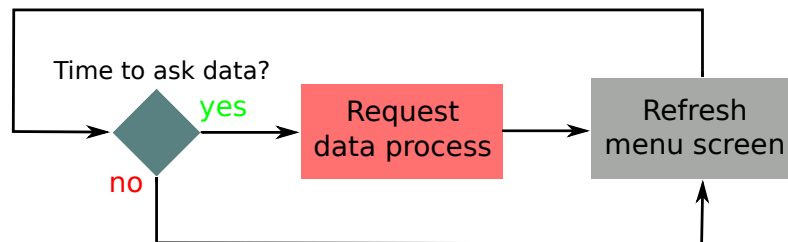


Figure 4.9 – Screen + rotary flow diagram

The first function *setInputFlags*, whose code is listed in figure 4.11, reads the state of the three input pins which will control the printed menu and checks if a change occurred since last state. An array will store the state changes of the input pins, called flags array. Accordingly, the variables from the array will be set to HIGH in the position that matches the pin which changed state. In figure 4.10 an explanatory representation for the code is shown.

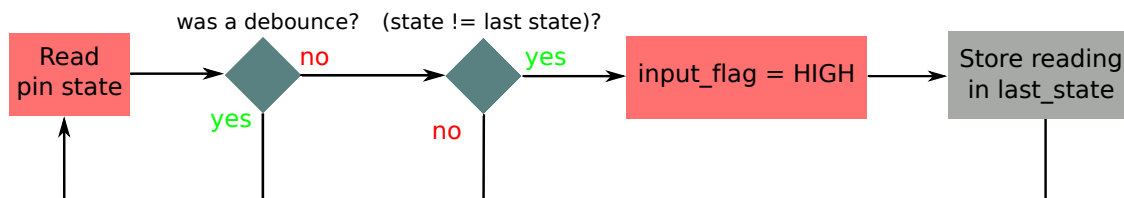


Figure 4.10 – *setInputFlags* function diagram


```

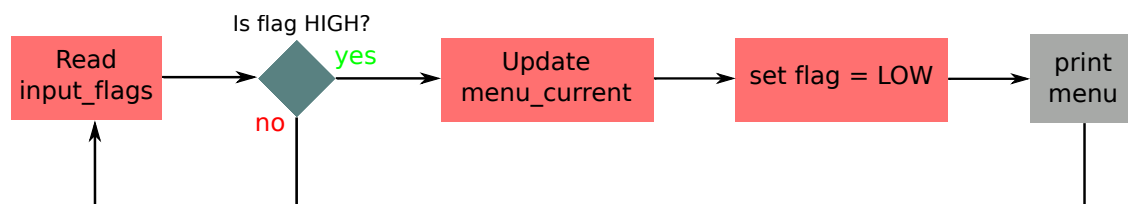
1 void setInputFlags() {
2   for (int i = 0; i < numOfInputs; i++) {
3     int reading = digitalRead(inputPins[i]); //stores reading
4     state
5     if (reading != lastInputState[i]) {
6       lastDebounceTime[i] = millis();
7     }
8     if ((millis() - lastDebounceTime[i]) > debounceDelay) { //
9       ensures it is not a debounce
10      if (reading != inputState[i]) {
11        inputState[i] = reading; //stores reading in state array
12        if (inputState[i] == HIGH) {
13          inputFlags[i] = HIGH; //set flags HIGH for changed pins
14          only
15        }
16      }
17    }
18    lastInputState[i] = reading; //stores for comparing in next
19    cycle
20  }
21 }

```

Figure 4.11 – *setInputFlags* function code

4

After code in 4.11 is executed, *resolveInputFlags* mission is to cover the flags array and act according the changed pin: it will increase or reduce the current-menu-item variable to refresh the display and resize the selection box onto the menu item the user just scrolled to. The code which performs these actions is shown in figure 4.13 and a descriptive diagram is presented in figure 4.12.

Figure 4.12 – *resolveInputFlags* function diagram

```

1 void resolveInputFlags() {
2     for (int i = 0; i < numOfInputs; i++) {
3         if (inputFlags[i] == HIGH) {
4             inputAction(i); //simple function that updates variable
4                 controlling the selection box on screen: depending on
4                 which pin flag is HIGH (scrollForw, scrollBa, push)
4                 scrolls graphical selection box up/down
5             inputFlags[i] = LOW;
6             if (printInfo != 0) { //checking if push pressed
7                 u8g2.clearBuffer(); //library process, clear screen for
7                 sending new info later
8                 printSubMenu(); //printing corresponding submenu if
8                 button pressed
9                 printInfo = 0; //clearing variable again
10                u8g2.sendBuffer(); //sends updated representation
11            } else {
12                u8g2.clearBuffer();
13                printMenu(); //updating menu appearance, placing
13                selection box on current menu item
14                u8g2.sendBuffer();
15            }
16        }
17    }
18 }

```

Figure 4.13 – *resolveInputFlags* function code

The implemented menu will be depicted in figure 4.14.

Notice the little yellow box laying on the first menu item. This marker will denote the current menu item selected by the user, the one which will scroll through the whole menu.

4.2.4 microSD Card

The microSD Card will be used to store data in case of connection loss with the remote server. Diagram in figure 4.15 summarizes this process.

The writing process into the SD Card is very simple and its code is listed below in figure 4.16. Once it is done, the file will be correctly stored in the memory card.

4.2.5 ESP-12

This is the tricky stage. After the analysis from chapter 3, the *ESP-12* module was the chosen one, with its pros and cons. In this section, unexpected problems that came up during the software designing process will be described.

Initially, looking at figure 4.17, the module has 16 GPIO pins for potentially usage (see



Figure 4.14 – Menu implemented: a) Front page, b) 2nd page, c) Selection of “WiFi Connection” item, d) Selection of “IP Server” item, e) Selection of “Version” item, f) Selection of “Language” item

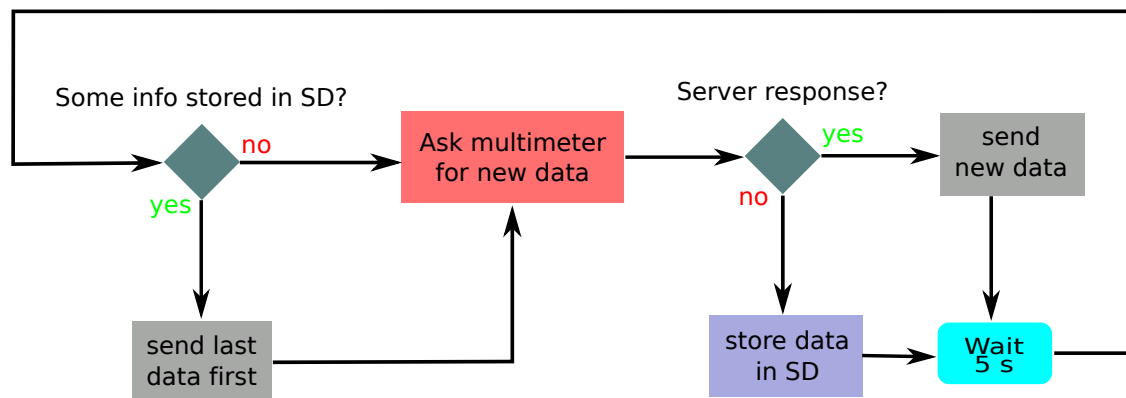


Figure 4.15 – Diagram of the storing process in microSD Card

pin release in appendix C). But after a quick search on internet, sources revealed that GPIOs 6 to 11 were used by the *esp8266* microcontroller to interface the external flash memory chip. They weren't actually connected to the header pins, only internally attached. In figure 4.18, once the metal cover is cleared, the microcontroller and the 8-lead flash memory chip are easily observed. Thus, the available pins had been reduced to 10. The module was slowly starting to be a tough nut to crack.

Some research later, it was found that depending on the flashing mode chosen, something called **DIO** and **QIO**, the chip would be using 2 or the 4 data lines to interface the flash memory[34]. So, going back to figure 4.17, from the 4 data lines (*DATA0*, *DATA1*, *DATA2* and *DATA3*) it was possible to free the two last ones if **DIO** interfacing mode was selected:

```

1 //after initializing SD in setup() -> SD.begin(CS_pin)
2 void loop() {
3   // ...
4   if (client.connect(host, port) == 0) { //if client doesn't
      connect
5     myFile = SD.open("test.txt", FILE_WRITE); //opens file called
      'test.txt' or creates it if not exists yet
6     if (myFile) { //ensure if file opened right
7       myFile.print("blablabla"); //writes what is indicated in
      brackets
8       myFile.flush(); //Ensures that any bytes written to the
      file are physically saved to the SD card. This is done
      automatically when the file is closed, done anyway xD
9       myFile.close(); //note that only one file can be opened at
      a time, so have to close one before opening another.
10    }
11  }
12  // ...
13 }

```

Figure 4.16 – microSD Card writing code

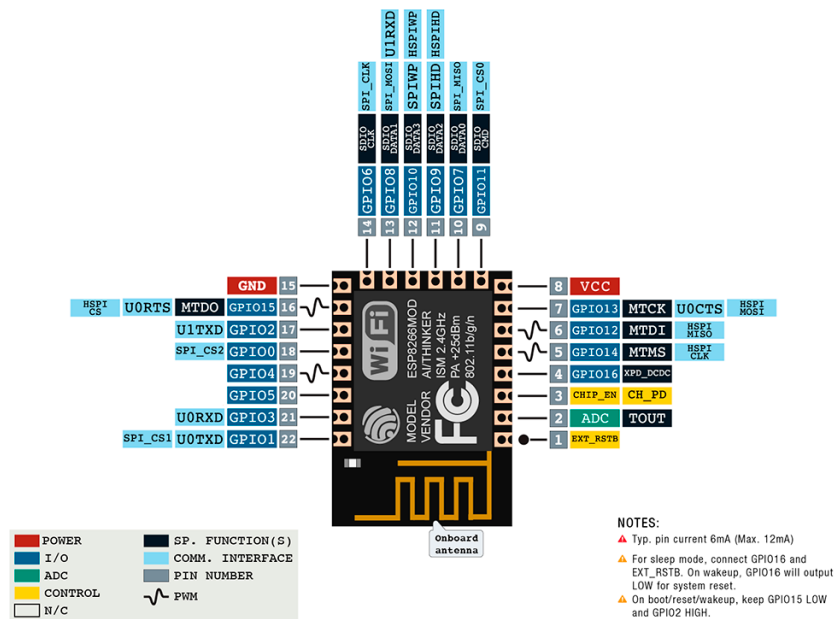


Figure 4.17 – ESP-12 pinout [8]

GPIO9 and GPIO10.

The next step was to determine which pins use for which module. After connecting each module separately and assuring proper operation, all modules were put together and debugged the issues that came up. In figure 4.19 is presented how all the pins were finally

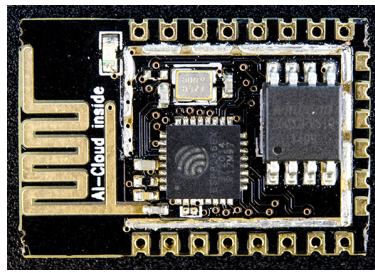


Figure 4.18 – ESP-12 uncovered close-up [9]

allocated.

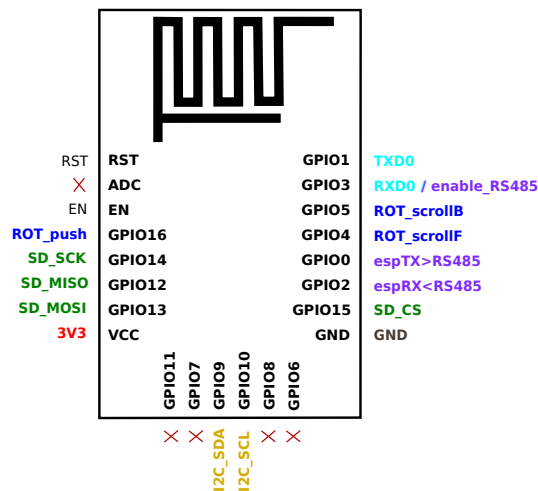


Figure 4.19 – ESP-12 connections schema

It is worth to discuss the pin distribution showed in previous figure 4.19:

- Hardware transmitter and receiver port ($Rx0$ and $Tx0$) will be used for flashing/debugging.
- microSD Card will be interfaced with the hardware SPI bus.
- I2C bus for the OLED display will be allocated via software to pins different from the default ones.
- A software UART will be set for communicating with the power meter.
- Rest of the pins will go to control the rotary connections.
- The clever idea comes now: there is still the read/write enable pin from the RS485 communication left but no more usable pins free. The proposal was to share the $Rx0$. It will next explained how.

In figure 4.20 a representation of this sharing-scenario is presented: on one hand, this port will be used to flash the module, but during operation mode this bus will be practically unused. Even if extra information would be needed for debugging the program, just the Tx0 pin would be critical to receive the debugging data on the computer. Rx0 pin would still be free.

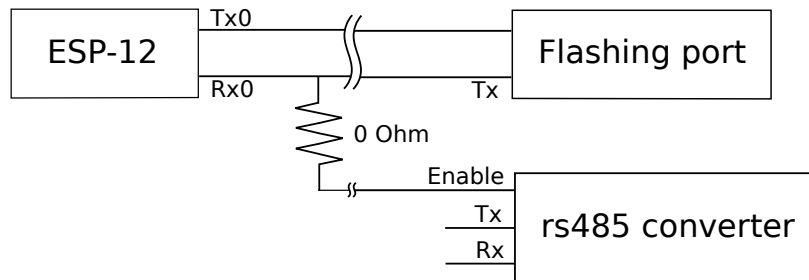


Figure 4.20 – *Diagram representing the pin-sharing situation*

4

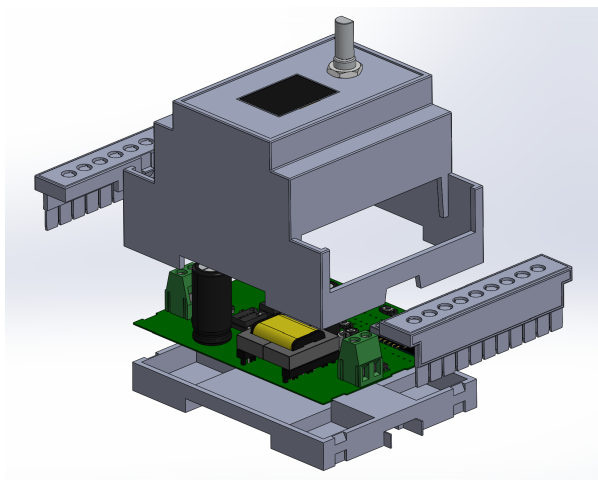
The only drawback would be the continuous switches on the enable pin due to data transfer during upgrading of the firmware, which could affect on the differential communication and might damage in some way the multimeter. It shouldn't be a real problem, because the RS485 communication bus in uploading mode should be empty and nothing should be sent to the meter device. Just in case, a 0Ω resistor¹ will be connected to separate both lines if trouble occurs.

4.3 Mechanical Design

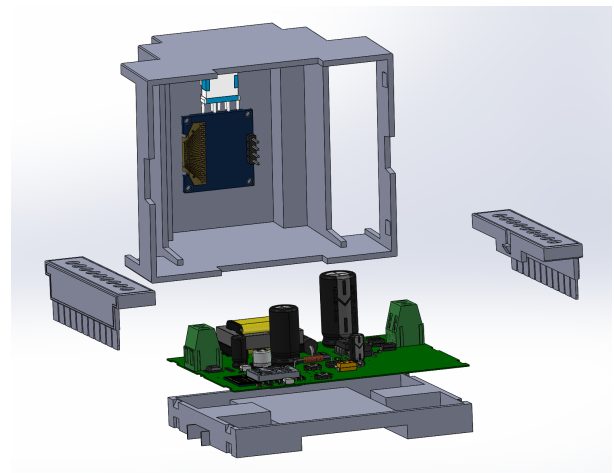
In this section, the appearance of the prototype will be presented. As mentioned in the requirements of chapter 2, the board will be placed inside a DIN rail box, along with the display and rotary encoder.

The 3D implementation of the design is shown in figure 4.21. Figure 4.2 is a video included in the document.

¹Meaning that the corresponding footprint of the resistor will serve as a bridge between the lines. A small resistor will be placed to share the pin or left the contact open if something goes wrong.



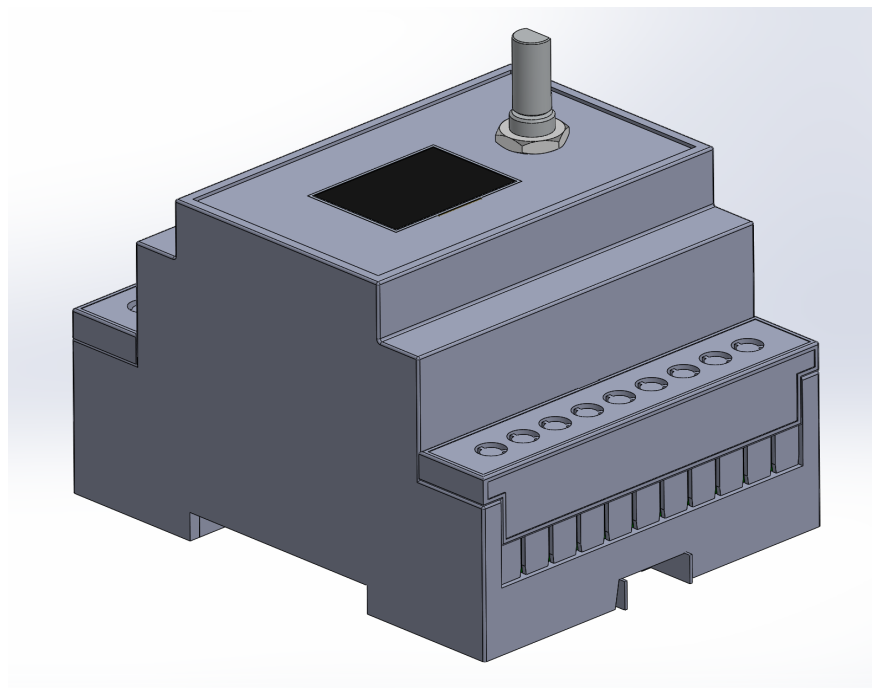
(a) View 1



(b) View 2

Figure 4.21 – *Pictures of the 3D model prototype*

4

**Video 4.2** – *Video of 3D model prototype*

4

CHAPTER

5

VERIFICATION AND TESTING

This chapter details the last step in a product development. During this stage it will be checked that the final results correspond to what was first established in the System Requirements 2. In order to verify the correct functioning of the system tests will be carried out, the ones listed at the end of the system analysis chapter 3, taking parts of the system independently. For this purpose this chapter will be subdivided in electronic/[PCB](#) and software testing.

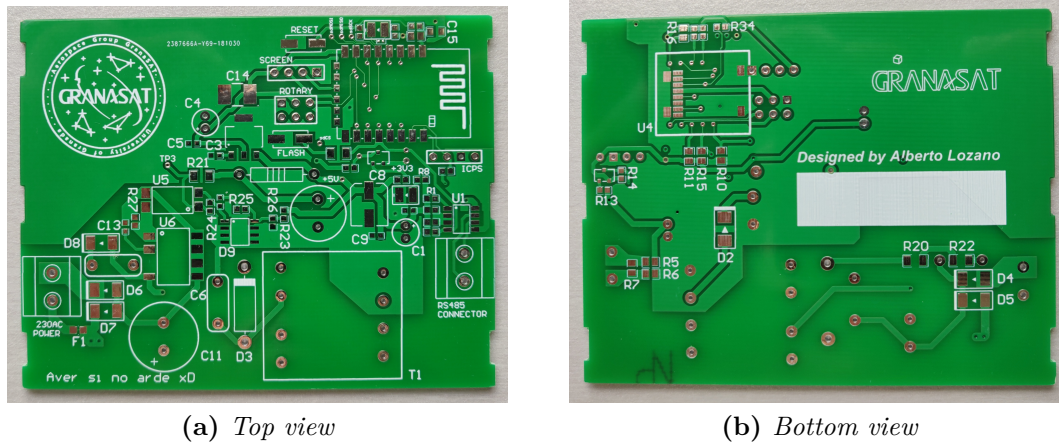
It is necessary to point out, that the following tests carried out will be a sort of preliminary ones, due to delays with the [PCB](#) implementation. It is interesting to document how each step was achieved, still even in the modular stage, arriving lastly to the final tests.

5.1 Electronic testing

The aim of this section is to test that every part of the hardware works and that the interconnection are done in a satisfactory way. The way to check the [PCB](#) was to be very careful revising the design before sending it to the manufacturer and then ensure all connections one by one on the board itself.

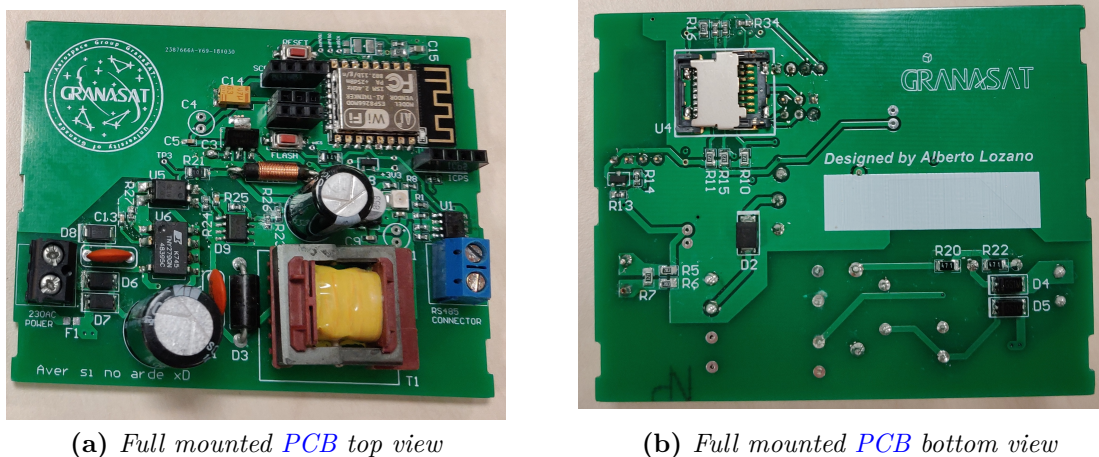
The manufactured board is shown in figure 5.1. And the full mounted [PCB](#) is depicted in figure 5.2.

To ensure a proper 5 V output, the [PCB](#) will be connected to the mains through its corresponding terminal block and measure the output. A picture showing the DC output



(a) Top view

(b) Bottom view

Figure 5.1 – Empty manufactured *PCB*(a) Full mounted *PCB* top view(b) Full mounted *PCB* bottom viewFigure 5.2 – Full mounted *PCB*

voltage along with its AC output, confirmig a good quality output, is depicted in figures 5.3a and 5.3b: the power module gives a consistent 5 V output.

Additionally, the *PCB* does not get strangely hot nor make any inappropriate noise.

5.2 Software testing

This testing will be divided into different sections, one for each module, in order to make independent tests and face any problems that may occur.

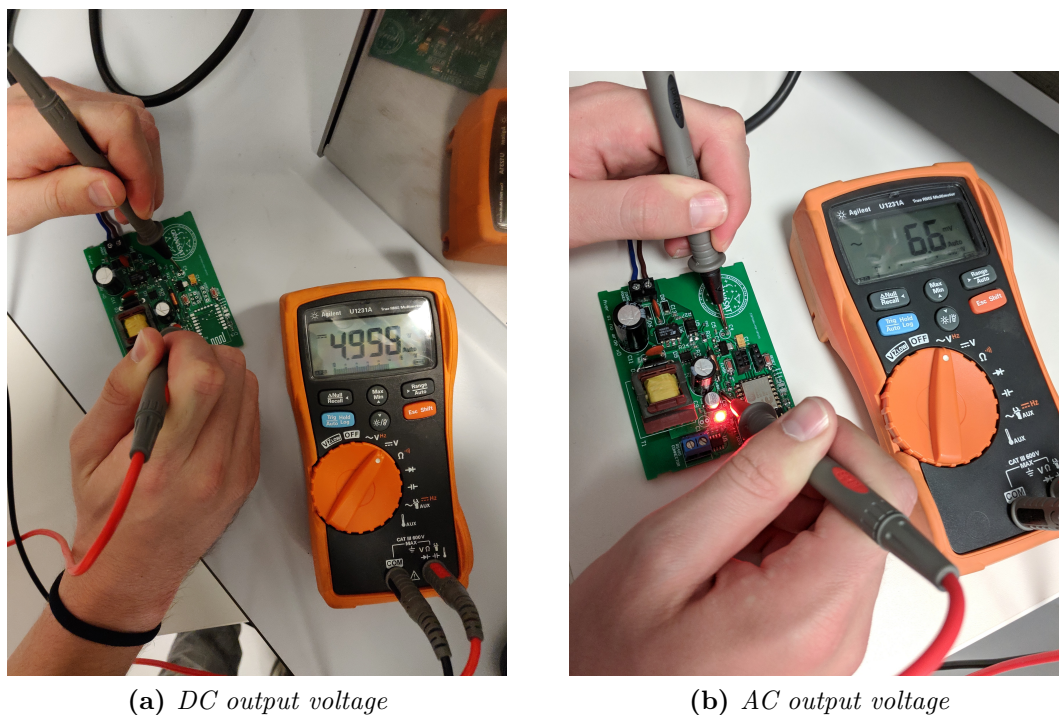


Figure 5.3 – *Output voltage of the power module*

5.2.1 microSD Card

To check if the data is properly saved to the SD card, a simple script with random variables to write on it will be created. Taking the structure code from figure 4.16 but including a loop to write multiple times, the data will be set to a file on the memory card.

After running the code and connecting the card to the computer, the data can be seen correctly written to the file. In this example, what is listed in figure 5.4.

5.2.2 Communication Module

First of all, the attempt to connect to the power meter by the RS485-RS232 adapter along with an USB adapter, the one shown in 4.5, will be next tackled. The setup circuitry is presented in figure 5.5. Notice the white cable connected to the RS485 converter, which goes to the HAGER multimeter differential outputs. This test will be just to ensure correct operation of the multi-function meter.

The corresponding response was the one shown in last chapter 4.6. It was already claimed to be correct.

Right after, the approach through the Arduino UNO was faced. This time, the implemented code had to be in *C Programming Language*. Applying the advantage provided by the python code about the transmission packets, it was able to establish

```

1 |
2 | {"serialNum":1111,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
3 | {"serialNum":1111,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
4 | {"serialNum":1112,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
5 | {"serialNum":1112,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
6 | {"serialNum":1112,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
7 | {"serialNum":1113,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
8 | {"serialNum":1113,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}
9 | {"serialNum":1113,"power":4000.00,"current":20.00,"timeStamp":"18
   | h30m20s_08.08.2018"}

```

Figure 5.4 – *File written on SD Card*

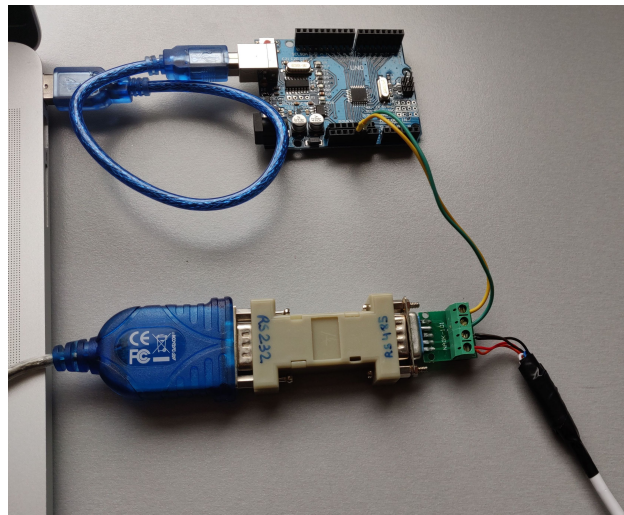


Figure 5.5 – *First communication test with power meter*

communication with the power meter. The circuitry setup for this situation is presented in figure 5.6.

Next step was to make it work with the main controller of the project: the ESP-12. It is important to point out, that for the development stage, the *NodeMCU* breakboard was the one used for the tests. It integrates the module used in this project. The corresponding setup is depicted in figure 5.7.

This time, the response was also received, and comparing the values to the correct ones, a good operation is confirmed. Concluding, it can be affirmed that the RS485 communication code designed is adequate to the presented scenario.

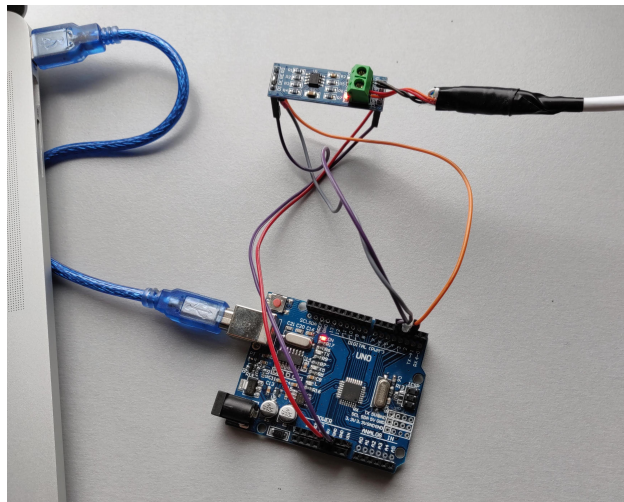


Figure 5.6 – *RS485 communication setup using Arduino*

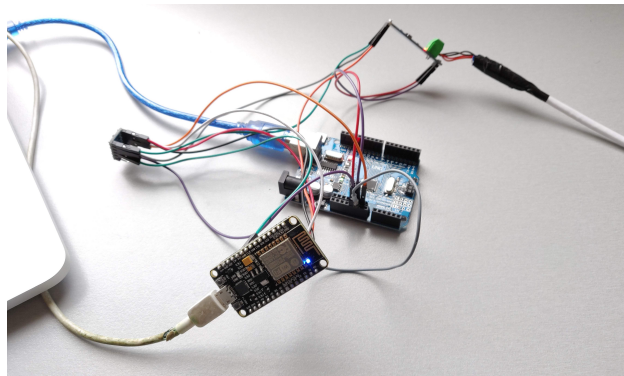


Figure 5.7 – *RS485 communication setup using NodeMCU breakboard*

5.2.3 Display + Rotary

To check a correct operation of the menu implemented on the display, the setup presented in figure 5.8 will be implemented, still using the NodeMCU breakboard.

The corresponding evidence of its operation will be shown in figure 5.1. By means of this video, a proper operation of the menu created is confirmed.

5.2.4 MCU - ESP-12

Last verification step would be to check the communication between the MCU module and the rest of components. To test the ESP-12, the entire connections will be attached, firstly, to the NodeMCU breakboard to ensure that the chip do not crash driving all code modules simultaneously. The pin allocation was so defined, that it works correctly.

Next, the program communication with the the module implemented on the manufactured

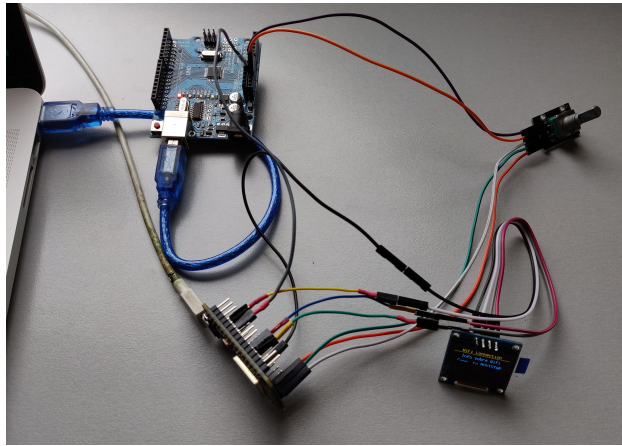
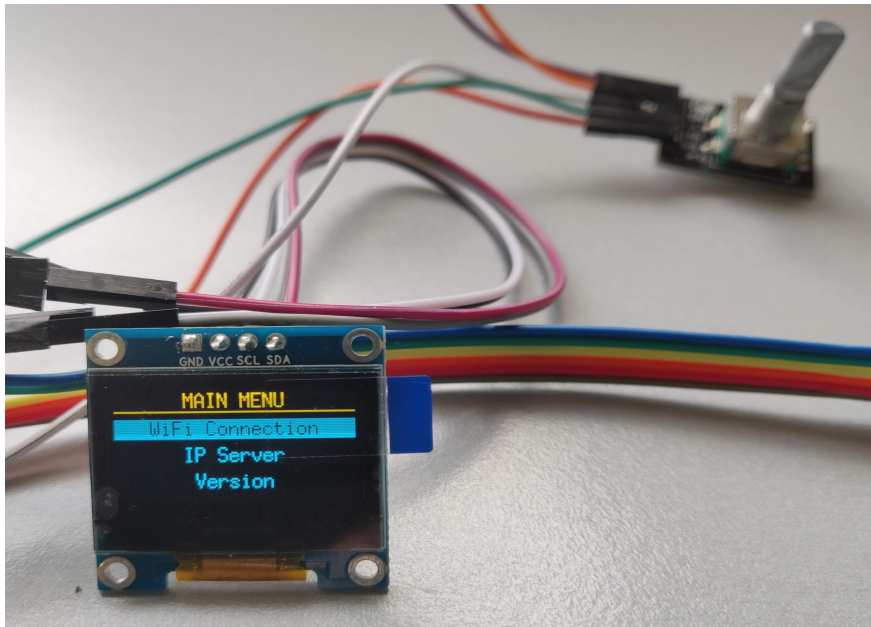


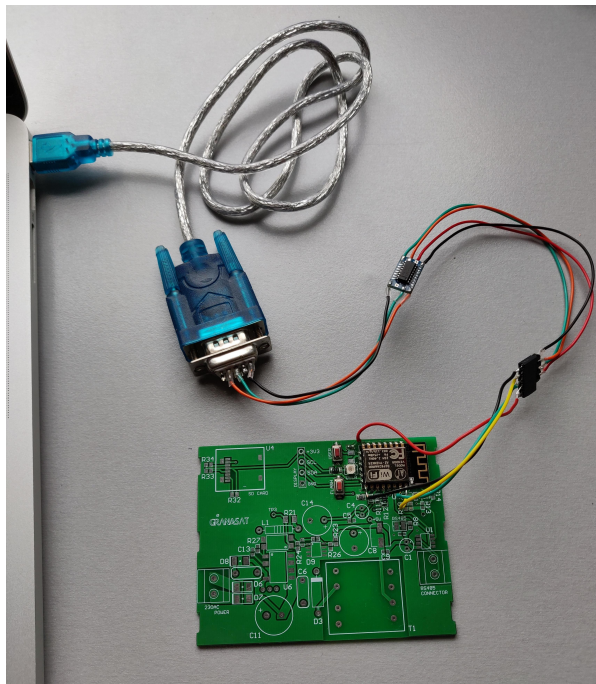
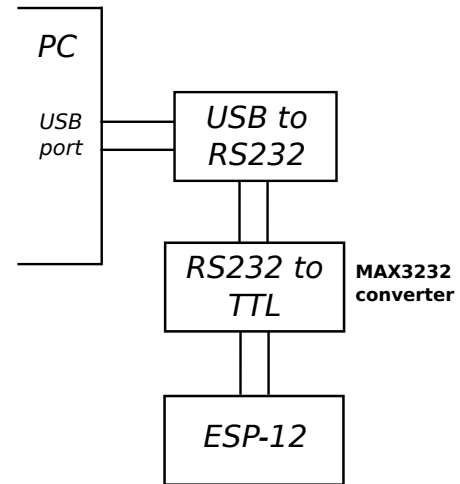
Figure 5.8 – *Display + rotary circuit setup*



Video 5.1 – *Scrolling display menu video*

PCB has to be checked. In figure 5.9a, the ESP-12 along with the needed components to enable flashing the chip are implemented on the board. At this stage, yet without the power module to power the chip. An external power supply was used to directly provide the 5 V and ensure correct design of the **PCB** tracks. Next to it, in 5.9b, a diagram clarifies the situation.

The *esptool.py* serial bootloader utility will be used to flash the firmware to the ESP8266. For this, the python package is needed [32], in order to install the flashing tool later, running the command below. With some Python installations this may not work and an error will be received: check [36].

(a) *ESP-12 module integrated on the final PCB*(b) *Explicative diagram of the flashing scenario [35]***Figure 5.9** – *Flashing the ESP-12 module*

```
1 | $ pip install esptool
```

Straightaway, flashing the firmware to the module is achieved, having revised the available commands and line options from the bootloader through its following help command. The right answer from the module can be seen in figure 5.10.

```
1 | $ esptool.py -h
```

```

1      $ esptool.py --chip esp8266 --port /dev/tty.wchusbserial1410
      --before no_reset --after no_reset write_flash 0x00000000
      prog.bin
2
3      esptool.py v2.3.2-dev
4      Connecting.....
5      Chip is ESP8266EX
6      Features: WiFi
7      Uploading stub...
8      Running stub...
9      Stub running...
10     Configuring flash size...
11     Auto-detected Flash size: 4MB
12     Compressed 251040 bytes to 183164...
13     Wrote 251040 bytes (183164 compressed) at 0x00000000 in 96.2
      seconds (effective 20.9 kbit/s)...
14     Hash of data verified.
15
16     Leaving...

```

Figure 5.10 – *Response from power meter through the python code*

After achieving communication with the module, this method is confirmed to be the one to flash the chip. Female header pins for the flash port will be provided to the [PCB](#) in order to facilitate the process, avoiding the need of soldering and desoldering wires to the pads.

Finally, the entire project will be mounted on the [PCB](#) and check its operation. It would be the final step to conclude the functionality test of the proposed prototype. The full device, along with the enclosure is shown in figure ??.



Figure 5.11 – *Full mounted device*

The mounting holes for the display and rotary encoder placing will be left due to complications in order to properly cut the enclosure. As all previous steps were satisfactory, this stage also worked.

CHAPTER

6

CONCLUSIONS AND FUTURE LINES

During this document it has been presented the design and development of a telemeassurement device, from the first stage to a final product. It was a challenge to code this device at the same time that the hardware platform was created, inducing many failures due to the lack of understanding of a particular module or the tight time schedules. But at the end, it all come to a conclusion.

During this project a lot has been learned about hardware development and manufacturing. From knowing the very basics digital communication protocols such as [UART](#) or [SPI](#), to understand a datasheet from a digital device and apply this concepts to the board. As long as software goes, many struggles have been defeated, such as finding out how to communicate with the power meter through a lot of trial and error attempts, or implementing the display menu properly along with a rotary encoder, assuming the fact that software development was not my focus point in this degree. In addition, the first contacts with relevant software in the engineering industry, as Altium Designer and Solidworks, was really motivating in the same level as hard, making possible to get a taste from the “real” world outside the student’s environment.

Several external modules were integrated on the [PCB](#). A memory card module and a display together with a rotary encoder to scroll through the implemented menu, which was not super hard but not trivial either. The probably most difficult stage was with the power module. The complexity in the implementation of a SMPS, taking care of many the different

parts like the two isolated grounds or the circuit tracks which would carry a lot of power. Many details that had to be properly thought to get it work. Regarding the MODBUS communication protocol, on top of [RS485](#), it was also challenging, since the protocol time constraints had to be really precise. For that reason, the appendix about this communication protocol was included.

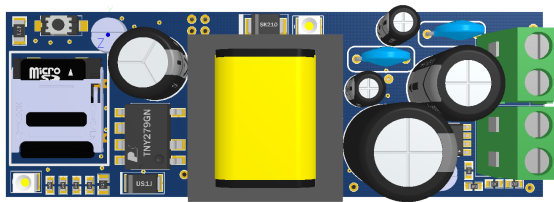
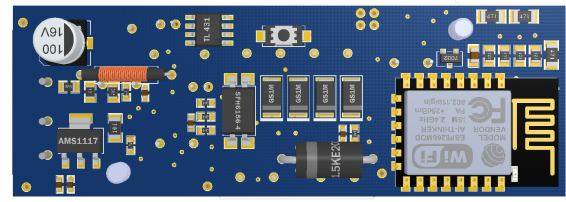
This project also involved a lot of manual soldering since the whole project had to be mounted, that also took quite some time. All of these difficulties along with their solutions have made this project possible and entertaining. It can be said that this project was an introduction to the student of how a hardware company works, what is needed and what are the steps to design a successful hardware product. This itself can be considered as a challenge, since normally the working routine is very different from the previous one the student may have, forcing her/him to collect information and technical data from a very early stage or produce a lot of work that may be useless. This also provides the student with days where nothing is achieved and days where everything works, which is part of what an engineer should be prepared to face up to.

With respect to the system requirements, the desired points to reach to, regarding the JSON data treatment and connection establishment through eduroam, these were not really successfully achieved. It will be left for a future refinement stage.

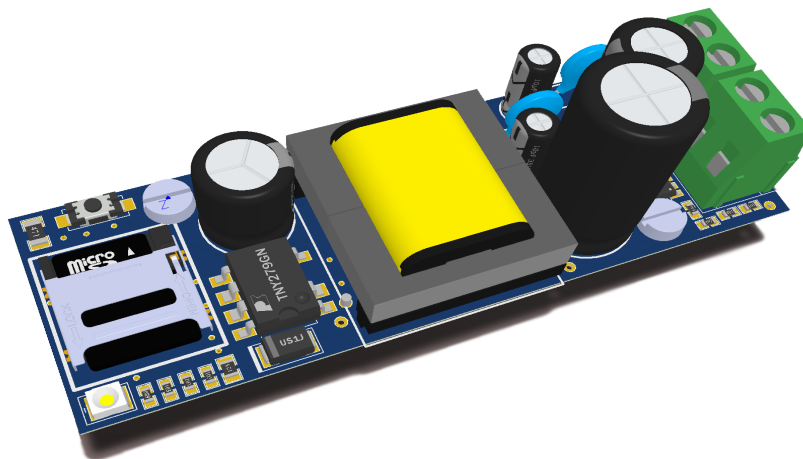
6.1 Future lines

Along with the creation of this project, many paths were open to improve the device, but since time is limited, just a certain number of features were selected to be implemented. Some of the upgrades reserved for future improvements are listed below.

- Be able to send the data through the university WiFi *eduroam*, with [WPA2](#) Enterprise mode, using specific encryption and authentication standards, since one possible scenario to apply the device for will be to gather the energy measured data from old power meters without wireless features amongst university buildings.
- Improve graphical interface if desired, making possible to change configuration settings like server IP Address or WiFi connection directly from the display, without forgetting that this will be redundant as a final product.
- Make the data treatment more robust and improve usage of [JSON](#) files.
- Add the alternative [SMPS](#) analysed [\[2\]](#), as it should give better features.
- Upgrade the [MCU](#) to a more powerful ESP32, permitting more pins to connect other components as [GPRS](#) module to enlarge features with no WiFi possibility.
- Reduce the device dimensions, due to what was mentioned in system design [4](#) stage about the plenty spare surface in the finally implemented board.

(a) Compact *PCB* design top view(b) Compact *PCB* design bottom view**Figure 6.1** – Compact *PCB* 3D model

Concerning the last point, the intention was thought from the beginning of the project, having tried with a smaller and more space efficient version already. However, during the design, a lot of difficulties were found to integrate all components while do an efficient trace routing for proper signal integrity. The intended design is presented in figure 6.1, where it is easily appreciated the much higher density. Also, an illustrative video of the board is shown in figure 6.1.

**Video 6.1** – Compact *PCB* 3D video

APPENDIX

A

MODBUS PROTOCOL SPECIFICATIONS

The basics of **MODBUS** communication protocol will be next introduced. It will focus on the message structure, so CRC (*Cyclical Redundancy Checking*) forming or *Parity Check* will not be treated in this appendix.

MODBUS is an application layer messaging protocol, positioned at level 7 of the OSI model, that provides master/slave communication between devices connected on different types of buses or networks. An illustration is shown in figure A.1.

MODBUS is a request/reply protocol and offers services specified by function codes. The objectives of this appendix is to describe the frames structure within the framework of **MODBUS** transactions.

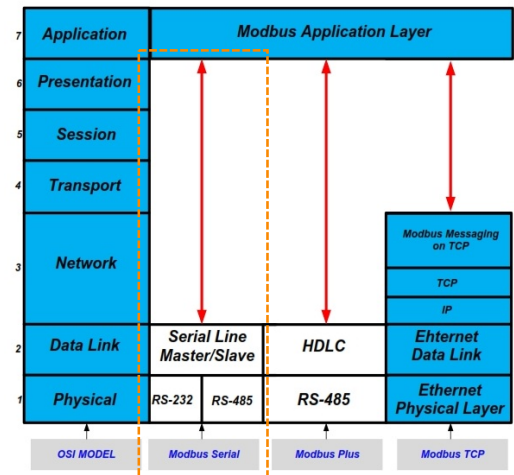


Figure A.1 – **MODBUS** communication stack

A.1 Transmission Modes: RTU

One serial transmission mode is defined as *RTU mode*, which is the one applied to this project.

When devices communicate on a MODBUS serial line using the *Remote Terminal Unit* mode, each 8-bit byte in a message contains two 4-bit hexadecimal characters. Each message must be transmitted in a continuous stream of characters. Each character or byte is sent left to right: Least Significant Bit (LSB) . . . Most Significant Bit (MSB)

The format for each byte in RTU mode is:

Bits per Byte:	1 start bit
	8 data bits, least significant bit sent first
	1 bit for even/odd parity, no bit for no parity
	1 stop bit if parity is used, 2 bits if no parity

List of Tables A.1 – Format for each byte in RTU mode

A.1.1 Frame description

The MODBUS application protocol defines a simple PDU independent of the underlying communication layers. To map the protocol on a specific bus or network, some additional fields has to be introduced on the Protocol Data Unit. An clarifying ilustration is presented in figure A.2.

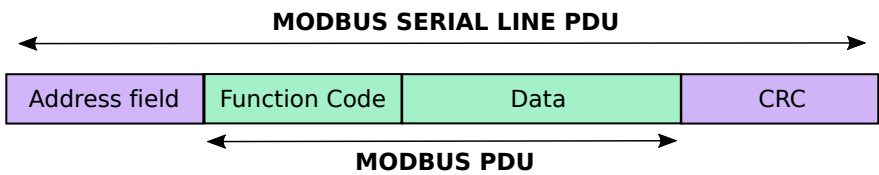


Figure A.2 – MODBUS frame over Serial communication

So the RTU frame looks as shown in figure A.3. The maximum size is 256 bytes.

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 to 252 byte(s)	2 bytes <small>CRC Low CRC High</small>

Figure A.3 – MODBUS RTU message description

Until here, a device can transmit a frame with known beginning and ending point. During transmission, to be able to identify the different messages, frames are separated by silent intervals of at least 3.5 character times. The behaviour is depicted in figure A.4. In addition, a silent interval of more than 1.5 character times has to occur between two characters, otherwise the message frame is declared incomplete and should be discarded by the receiver.

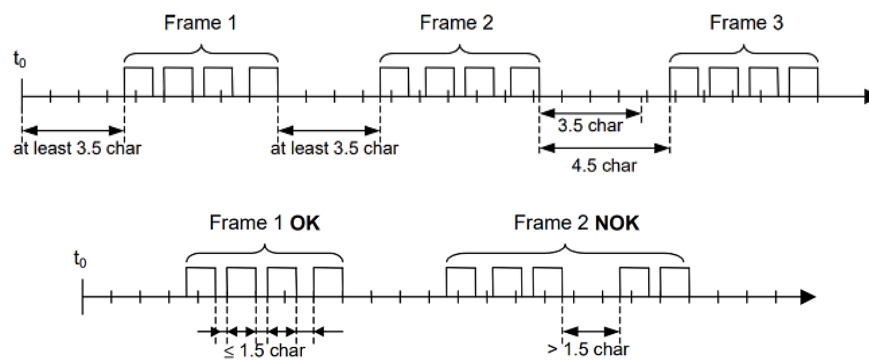


Figure A.4 – MODBUS RTU transmission

A.1.2 MODBUS Function Codes

MODBUS protocol uses some standard function codes to provide access to measurement and information registers. There are multiples function codes available. This appendix will focus on the one which will be used in this project: *Read Holding Registers, 03*. This function is for reading contents of a contiguous block of holding registers in the remote device. These are internal registers with 16 bits values stored in them, so 2 bytes of information.

For each register, the first byte contains the high order bits and the second contains the low order bits. The same procedure is applied for other functions. The most used ones will be listed in table A.2.

Data Access	Bit Access	Internal bits or Physical Coils		Code	(hex)
			Read Coils	01	01
16 bit Access		Physical Input Registers	Write Single Coil	05	05
			Write Multiple Coils	15	0F
			Read Input Register	04	04
		Internal Registers or Physical Output Registers	Read Holding Registers	03	03
			Write Single Register	06	06
			Write Multiple Registers	16	10

List of Tables A.2 – MODBUS most relevant functions

Coils are usually associated with relay outputs, so it would be requesting the ON/OFF status of a discrete coil. In *Input Registers*, analog content is requested, as temperature or pressure values.

A.1.3 Example MODBUS RTU frame

Representations of a typical request and response are presented in tables A.3 and A.4.

REQUEST		
Function code	1 byte	0x03
Starting Address	2 bytes	0x0000 to 0xFFFF
Quantity of Registers	2 bytes	1 to 125 (0x7D)

List of Tables A.3 – MODBUS Request format

RESPONSE		
Function code	1 byte	0x03
Starting Address	1 bytes	2 x N*
Quantity of Registers	N* x 2 bytes	

* Quantity of registers.

List of Tables A.4 – MODBUS Response format

APPENDIX

B

SM101C MULTI-FUNCTION METER REGISTERS

The datasheet from SM101C multi-function meter, concerning the available registers, will be included in this appendix.

The relevant registers will be highlighted in blue:

- The most important registers values for the project objectives.
- The register that was used to debug the code for requesting data to the multimeter, which was the *Module Temperature*. It was possible to check the temperature also on the user display from the device, and it is a value that didn't change constantly, unlike voltage or current measurements. So it was the ideal one to send a request for and check if the received value matched the one on the multimeter display.

The tables are listed next:

GENERALITIES

Generalities

The JBUS/MODBUS used by the product involves a dialogue using a master-slave hierarchical structure.

There are two possible dialogues :

- The master communicates with a slave and waits for its reply
- The master communicates with all the slaves without waiting for the reply

Hager products implement functions :

- **3** : to read n Words (Maximum 125 words)
- **6** : to write 1 word
- **16** : to write n Word

The register type used are :

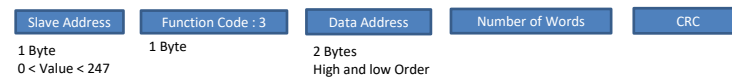
long	2 Words (4 bytes)
int	1 Word (2 bytes)

If the register value is not available, according to its type and its signed, the indicated value is :

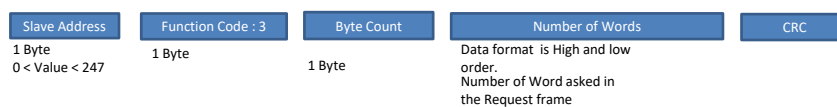
unsigned long	FFFF FFFFhex
signed long	7FFF FFFF hex
unsigned Int	FFFF hex
signed int	7FFF hex

Function 3 : Read n Words

Function 3 : Request frame format

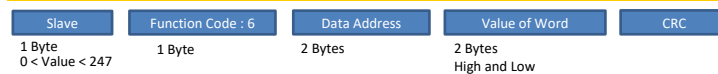


Function 3 : Answer frame format



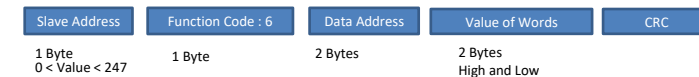
Function 6 :Write 1 Word

Function 6 : Request frame format



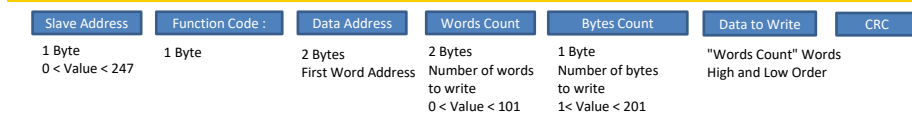
When selecting slave address 0, the message is sent to all devices (Hager or not) present on the network (No answer is sent by the device)

Function 6 : Answer frame format



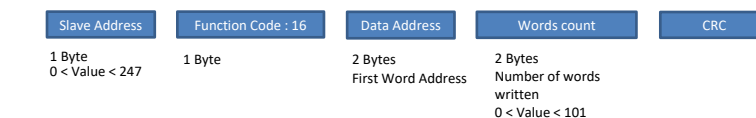
Function 16 :Write n Words

Function 16 : Request frame format



When selecting slave address 0, the message is sent to all devices (Hager or not) present on the network : No answer is sent by the device

Function 16 : Answer frame format



Product name : SM101C

C550 Hex : Metrology Affected by current and voltage transformers

Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
50512	C550	2	Hour Meter	1/100 h	3	0xFFFFFFFF
50514	C552	2	Phase to Phase Voltage: U12	V/100	3	0xFFFFFFFF
50516	C554	2	Phase to Phase Voltage: U23	V/100	3	0xFFFFFFFF
50518	C556	2	Phase to Phase Voltage: U31	V/100	3	0xFFFFFFFF
50520	C558	2	Simple voltage : V1	V/100	3	0xFFFFFFFF
50522	C55A	2	Simple voltage : V2	V/100	3	0xFFFFFFFF
50524	C55C	2	Simple voltage : V3	V/100	3	0xFFFFFFFF
50526	C55E	2	Frequency : F	Hz/100	3	0xFFFFFFFF
50528	C560	2	Current : I1	mA	3	0xFFFFFFFF
50530	C562	2	Current : I2	mA	3	0xFFFFFFFF
50532	C564	2	Current : I3	mA	3	0xFFFFFFFF
50534	C566	2	Neutral Current : In	mA	3	0xFFFFFFFF
50536	C568	2	Σ Active Power +/- : P	kW/100 (Signed)	3	0x7FFFFFFF
50538	C56A	2	Σ Reactive Power +/- : Q	kvar/100 (Signed)	3	0x7FFFFFFF
50540	C56C	2	Σ Apparent Power : S	kVA/100	3	0xFFFFFFFF
50542	C56E	2	Σ Power Factor : - : leading et + : lagging : PF	0,001 (Signed)	3	0x7FFFFFFF
50544	C570	2	Active Power phase 1 +/- : P1	kW/100 (Signed)	3	0x7FFFFFFF
50546	C572	2	Active Power phase 2 +/- : P2	kW/100 (Signed)	3	0x7FFFFFFF
50548	C574	2	Active Power phase 3 +/- : P3	kW/100 (Signed)	3	0x7FFFFFFF
50550	C576	2	Reactive Power phase 1 +/- : Q1	kvar/100 (Signed)	3	0x7FFFFFFF
50552	C578	2	Reactive Power phase 2 +/- : Q2	kvar/100 (Signed)	3	0x7FFFFFFF
50554	C57A	2	Reactive Power phase 3 +/- : Q3	kvar/100 (Signed)	3	0x7FFFFFFF
50556	C57C	2	Apparent Power phase 1 : S1	kVA/100	3	0xFFFFFFFF

	50558	C57E	2	Apparent Power phase 2 : S2	kVA/100	3	0xFFFFFFFF
	50560	C580	2	Apparent Power phase 3 : S3	kVA/100	3	0xFFFFFFFF
	50562	C582	2	Power Factor phase 1 -: leading and + : lagging : PF1	0,001 (Signed)	3	0x7FFFFFFF
	50564	C584	2	Power Factor phase 2 -: leading and + : lagging : PF2	0,001 (Signed)	3	0x7FFFFFFF
	50566	C586	2	Power Factor phase 3 -: leading and + : lagging : PF3	0,001 (Signed)	3	0x7FFFFFFF
Not applicable	50568	C588	2	System value I Sys = (I1+I2+I3) / 3	mA	3	0xFFFFFFFF
Not applicable	50570	C58A	2	System value U Sys = (U12 + U23 + U31) / 3	V/100	3	0xFFFFFFFF
Not applicable	50572	C58C	2	System value V Sys = (V1 + V2 + V3) / 3	V/100	3	0xFFFFFFFF
	50512	C550	62				

C650 Hex : Energies

	Address Dec.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
	50768	C650	2	Hour meter	1/100 h	3	0xFFFFFFFF
Not applicable	50770	C652	2	Total	kWh	3	0xFFFFFFFF
Not applicable	50772	C654	2	Total	kvarh	3	0xFFFFFFFF
Not applicable	50774	C656	2	Total	kVAh	3	0xFFFFFFFF
Not applicable	50776	C658	2	Total	kWh	3	0xFFFFFFFF
Not applicable	50778	C65A	2	Total	kvarh	3	0xFFFFFFFF
	50780	C65C	2	Partial	kWh	3	0xFFFFFFFF
	50782	C65E	2	Partial	kvarh	3	0xFFFFFFFF
Not applicable	50784	C660	2	Partial	kVAh	3	0xFFFFFFFF
Not applicable	50786	C662	2	Partial	kWh	3	0xFFFFFFFF
Not applicable	50788	C664	2	Partial	kvarh	3	0xFFFFFFFF
Not applicable	50790	C666	2	Number of	-	3	0xFFFFFFFF
Not applicable	50792	C668	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50794	C66A	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50796	C66C	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50798	C66E	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50800	C670	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50802	C672	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50804	C674	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50806	C676	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50808	C678	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50810	C67A	2	Total pulse	-	3	0xFFFFFFFF
Not applicable	50812	C67C	2	Predictive	kWh	3	0xFFFFFFFF
Not applicable	50814	C67E	2	Predictive	kvarh	3	0xFFFFFFFF
Not applicable	50816	C680	2	Predictive	kVAh	3	0xFFFFFFFF
Not applicable	50818	C682	2	Mean	0,01 kWh	3	0xFFFFFFFF
Not applicable	50820	C684	2	Mean	0,01 kWh	3	0xFFFFFFFF
Not applicable	50822	C686	2	Mean	0,01 kVARh	3	0xFFFFFFFF
Not applicable	50824	C688	2	Mean	0,01 kVARh	3	0xFFFFFFFF
Not applicable	50826	C68A	2	Last date for	s	3	0xFFFFFFFF
Not applicable	50828	C68C	1	Last average	W (no	3	0xFFFFFFFF
Not applicable	50829	C68D	1	Last average	var (no	3	0xFFFFFFFF
Not applicable	50830	C68E	1	Last average	W (no	3	0xFFFFFFFF
Not applicable	50831	C68F	1	Last average	var (no	3	0xFFFFFFFF

Not applicable	50832	C690	1	Last average	VA (no	3	0xFFFFFFFF
	50768	C650	65				

C750 Hex : Statistique Affected by current and voltage transformers

	Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
Not applicable	51024	C750	2	Avg U12	V/100	3	0xFFFFFFFF
Not applicable	51026	C752	2	Avg U23	V/100	3	0xFFFFFFFF
Not applicable	51028	C754	2	Avg U31	V/100	3	0xFFFFFFFF
Not applicable	51030	C756	2	Avg V1	V/100	3	0xFFFFFFFF
Not applicable	51032	C758	2	Avg V2	V/100	3	0xFFFFFFFF
Not applicable	51034	C75A	2	Avg V3	V/100	3	0xFFFFFFFF
Not applicable	51036	C75C	2	Avg F	Hz/100	3	0xFFFFFFFF
Not applicable	51038	C75E	2	Avg I1	mA	3	0xFFFFFFFF
Not applicable	51040	C760	2	Avg I2	mA	3	0xFFFFFFFF
Not applicable	51042	C762	2	Avg I3	mA	3	0xFFFFFFFF
Not applicable	51044	C764	2	Avg In	mA	3	0xFFFFFFFF
Not applicable	51046	C766	2	Avg P+ (Σ	kW/100	3	0xFFFFFFFF
Not applicable	51048	C768	2	Avg P- (Σ	kW/100	3	0xFFFFFFFF
Not applicable	51050	C76A	2	Avg Q+ (Σ	kvar/100	3	0xFFFFFFFF
Not applicable	51052	C76C	2	Avg Q- (Σ	kvar/100	3	0xFFFFFFFF
Not applicable	51054	C76E	2	Avg S (Σ	kVA/100	3	0xFFFFFFFF
Not applicable	51056	C770	2	Max/avg	V/100	3	0xFFFFFFFF
Not applicable	51058	C772	2	Max/avg U23	V/100	3	0xFFFFFFFF
Not applicable	51060	C774	2	Max/avg U31	V/100	3	0xFFFFFFFF
Not applicable	51062	C776	2	Max/avg V1	V/100	3	0xFFFFFFFF
Not applicable	51064	C778	2	Max/avg V2	V/100	3	0xFFFFFFFF
Not applicable	51066	C77A	2	Max/avg V3	V/100	3	0xFFFFFFFF
Not applicable	51068	C77C	2	Max/avg F	Hz/100	3	0xFFFFFFFF
	51070	C77E	2	Max/avg I1	mA	3	0xFFFFFFFF
	51072	C780	2	Max/avg I2	mA	3	0xFFFFFFFF
	51074	C782	2	Max/avg I3	mA	3	0xFFFFFFFF
	51076	C784	2	Max/avg In	mA	3	0xFFFFFFFF
	51078	C786	2	Max/avg P+	kW/100	3	0xFFFFFFFF
	51080	C788	2	Max/avg P-	kW/100	3	0xFFFFFFFF
	51082	C78A	2	Max/avg Q+	kvar/100	3	0xFFFFFFFF
	51084	C78C	2	Max/avg Q-	kvar/100	3	0xFFFFFFFF
	51086	C78E	2	Max/avg S	kVA/100	3	0xFFFFFFFF
Not applicable	51088	C790	1	Minimum	0.10%	3	0xFFFF
Not applicable	51089	C791	1	Minimum	0.10%	3	0xFFFF
Not applicable	51090	C792	1	Minimum	0.10%	3	0xFFFF
Not applicable	51091	C793	1	Maximum	0.10%	3	0xFFFF
Not applicable	51092	C794	1	Maximum	0.10%	3	0xFFFF
Not applicable	51093	C795	1	Maximum	0.10%	3	0xFFFF
	51024	C750	70				

C850 Hex : Metrology No Affected by current and voltage transformers

	Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
	51280	C850	1	Hour Meter	h	3	0xFFFF
	51281	C851	1	Phase to	V/100	3	0xFFFF
	51282	C852	1	Phase to	V/100	3	0xFFFF
	51283	C853	1	Phase to	V/100	3	0xFFFF
	51284	C854	1	Simple	V/100	3	0xFFFF
	51285	C855	1	Simple	V/100	3	0xFFFF
	51286	C856	1	Simple	V/100	3	0xFFFF
	51287	C857	1	Frequency :	Hz/100	3	0xFFFF
	51288	C858	1	Current : I1	mA	3	0xFFFF
	51289	C859	1	Current : I2	mA	3	0xFFFF
	51290	C85A	1	Current : I3	mA	3	0xFFFF
	51291	C85B	1	Neutral	mA	3	0xFFFF
	51292	C85C	1	Σ active	kW/100	3	0x7FFF
	51293	C85D	1	Σ aeactive	kvar/100	3	0x7FFF
	51294	C85E	1	Σ apparent	kVA/100	3	0xFFFF
	51295	C85F	1	Σ power	0.001	3	0x7FFF
	51296	C860	1	Active Power	kW/100	3	0x7FFF
	51297	C861	1	Active Power	kW/100	3	0x7FFF
	51298	C862	1	Active Power	kW/100	3	0x7FFF
	51299	C863	1	Reactive	kvar/100	3	0x7FFF
	51300	C864	1	Reactive	kvar/100	3	0x7FFF
	51301	C865	1	Reactive	kvar/100	3	0x7FFF
	51302	C866	1	Apparent	kVA/100	3	0xFFFF
	51303	C867	1	Apparent	kVA/100	3	0xFFFF
	51304	C868	1	Apparent	kVA/100	3	0xFFFF
	51305	C869	1	Power Factor	0.001	3	0x7FFF
	51306	C86A	1	Power Factor	0.001	3	0x7FFF
	51307	C86B	1	Power Factor	0.001	3	0x7FFF

Not applicable	51308	C86C	1	System	mA	3	0xFFFF
Not applicable	51309	C86D	1	System	V/100	3	0xFFFF
Not applicable	51310	C86E	1	System	V/100	3	0xFFFF
	51311	C86F	1	Total	MWh	3	0xFFFF
Not applicable	51312	C870	1	Total	Mvarh	3	0xFFFF
	51313	C871	1	Total	MWh	3	0xFFFF
Not applicable	51314	C872	1	Total	Mvarh	3	0xFFFF
51280 C850 35							

C900 Hex : Temperatures

Address Déc.	Address Hex.	Word Count	Description	Unit	Function	
	51456	C900	1	Internal module Temperature present	yes(1) / no (0)	3
	51457	C901	1	module Temperature	celsius degre	3
Not applicab	51458	C902	1	Number of external Temperature sensors to read (from 0 to 5 max)	-	3
Not applicab	51459	C903	1	Temperature extern 1	celsius degre	3
Not applicab	51460	C904	1	Temperature extern 2	celsius degre	3
Not applicab	51461	C905	1	Temperature extern 3	celsius degre	3
Not applicab	51462	C906	1	Temperature extern 4	celsius degre	3
Not applicab	51463	C907	1	Temperature extern 5	celsius degre	3
51456 C900 8						

C6A0 Hex : Energies per tariff

	Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
	50848	C6A0	1	0 >= Tariff number <= 8		3	0
	50849	C6A1	1	Tariff number in progress (1 to 8)		3	0
	50850	C6A2	16	8 * Positive Active Energies	kWh	3	0xFFFFFFFF
	50866	C6B2	16	8 * Positive Reactive Energies	kvarh	3	0xFFFFFFFF
Not applicable	50882	C6C2	16	8 * Time Counter	1/100 h	3	0xFFFFFFFF
	50848	C6A0	50				

C950 Hex : Harmonic

Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
THD						
51536	C950	1	thd U12	0.10%	3	0xFFFF
51537	C951	1	thd U23	0.10%	3	0xFFFF
51538	C952	1	thd U31	0.10%	3	0xFFFF
51539	C953	1	thd V1	0.10%	3	0xFFFF
51540	C954	1	thd V2	0.10%	3	0xFFFF
51541	C955	1	thd V3	0.10%	3	0xFFFF
51542	C956	1	thd I1	0.10%	3	0xFFFF
51543	C957	1	thd I2	0.10%	3	0xFFFF
51544	C958	1	thd I3	0.10%	3	0xFFFF
51545	C959	1	thd In	0.10%	3	0xFFFF
51536 C950		10				
I : harmonic						
Not applicable	51546	C95A	1	Number of	3	0
Not applicable	51547	C95B	1	harmonic I1	0.10%	3 0xFFFF
Not applicable	51548	C95C	1	harmonic I2	0.10%	3 0xFFFF

Not applicable	51549	C95D	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51550	C95E	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51551	C95F	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51552	C960	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51553	C961	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51554	C962	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51555	C963	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51556	C964	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51557	C965	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51558	C966	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51559	C967	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51560	C968	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51561	C969	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51562	C96A	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51563	C96B	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51564	C96C	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51565	C96D	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51566	C96E	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51567	C96F	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51568	C970	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51569	C971	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51570	C972	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51571	C973	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51572	C974	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51573	C975	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51574	C976	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51575	C977	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51576	C978	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51577	C979	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51578	C97A	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51579	C97B	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51580	C97C	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51581	C97D	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51582	C97E	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51583	C97F	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51584	C980	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51585	C981	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51586	C982	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51587	C983	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51588	C984	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51589	C985	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51590	C986	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51591	C987	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51592	C988	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51593	C989	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51594	C98A	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51595	C98B	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51596	C98C	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51597	C98D	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51598	C98E	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51599	C98F	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51600	C990	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51601	C991	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51602	C992	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51603	C993	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51604	C994	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51605	C995	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51606	C996	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51607	C997	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51608	C998	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51609	C999	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51610	C99A	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51611	C99B	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51612	C99C	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51613	C99D	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51614	C99E	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51615	C99F	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51616	C9A0	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51617	C9A1	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51618	C9A2	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51619	C9A3	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51620	C9A4	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51621	C9A5	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51622	C9A6	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51623	C9A7	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51624	C9A8	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51625	C9A9	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51626	C9AA	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51627	C9AB	1	harmonic I1	0.10%	3	0XFFFF

Not applicable	51628	C9AC	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51629	C9AD	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51630	C9AE	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51631	C9AF	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51632	C9B0	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51633	C9B1	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51634	C9B2	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51635	C9B3	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51636	C9B4	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51637	C9B5	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51638	C9B6	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51639	C9B7	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51640	C9B8	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51641	C9B9	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51642	C9BA	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51643	C9BB	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51644	C9BC	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51645	C9BD	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51646	C9BE	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51647	C9BF	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51648	C9C0	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51649	C9C1	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51650	C9C2	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51651	C9C3	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51652	C9C4	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51653	C9C5	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51654	C9C6	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51655	C9C7	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51656	C9C8	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51657	C9C9	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51658	C9CA	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51659	C9CB	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51660	C9CC	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51661	C9CD	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51662	C9CE	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51663	C9CF	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51664	C9D0	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51665	C9D1	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51666	C9D2	1	harmonic IN	0.10%	3	0XFFFF
Not applicable	51667	C9D3	1	harmonic I1	0.10%	3	0XFFFF
Not applicable	51668	C9D4	1	harmonic I2	0.10%	3	0XFFFF
Not applicable	51669	C9D5	1	harmonic I3	0.10%	3	0XFFFF
Not applicable	51670	C9D6	1	harmonic IN	0.10%	3	0XFFFF

51546 C95A 125

V PH/Ph

Not applicable	51671	C9D7	1	Number of		3	0
Not applicable	51672	C9D8	1	harmonic	0.10%	3	0XFFFF
Not applicable	51673	C9D9	1	harmonic	0.10%	3	0XFFFF
Not applicable	51674	C9DA	1	harmonic	0.10%	3	0XFFFF
Not applicable	51675	C9DB	1	harmonic	0.10%	3	0XFFFF
Not applicable	51676	C9DC	1	harmonic	0.10%	3	0XFFFF
Not applicable	51677	C9DD	1	harmonic	0.10%	3	0XFFFF
Not applicable	51678	C9DE	1	harmonic	0.10%	3	0XFFFF
Not applicable	51679	C9DF	1	harmonic	0.10%	3	0XFFFF
Not applicable	51680	C9E0	1	harmonic	0.10%	3	0XFFFF
Not applicable	51681	C9E1	1	harmonic	0.10%	3	0XFFFF
Not applicable	51682	C9E2	1	harmonic	0.10%	3	0XFFFF
Not applicable	51683	C9E3	1	harmonic	0.10%	3	0XFFFF
Not applicable	51684	C9E4	1	harmonic	0.10%	3	0XFFFF
Not applicable	51685	C9E5	1	harmonic	0.10%	3	0XFFFF
Not applicable	51686	C9E6	1	harmonic	0.10%	3	0XFFFF
Not applicable	51687	C9E7	1	harmonic	0.10%	3	0XFFFF
Not applicable	51688	C9E8	1	harmonic	0.10%	3	0XFFFF
Not applicable	51689	C9E9	1	harmonic	0.10%	3	0XFFFF
Not applicable	51690	C9EA	1	harmonic	0.10%	3	0XFFFF
Not applicable	51691	C9EB	1	harmonic	0.10%	3	0XFFFF
Not applicable	51692	C9EC	1	harmonic	0.10%	3	0XFFFF
Not applicable	51693	C9ED	1	harmonic	0.10%	3	0XFFFF
Not applicable	51694	C9EE	1	harmonic	0.10%	3	0XFFFF
Not applicable	51695	C9EF	1	harmonic	0.10%	3	0XFFFF
Not applicable	51696	C9F0	1	harmonic	0.10%	3	0XFFFF
Not applicable	51697	C9F1	1	harmonic	0.10%	3	0XFFFF
Not applicable	51698	C9F2	1	harmonic	0.10%	3	0XFFFF
Not applicable	51699	C9F3	1	harmonic	0.10%	3	0XFFFF
Not applicable	51700	C9F4	1	harmonic	0.10%	3	0XFFFF
Not applicable	51701	C9F5	1	harmonic	0.10%	3	0XFFFF
Not applicable	51702	C9F6	1	harmonic	0.10%	3	0XFFFF
Not applicable	51703	C9F7	1	harmonic	0.10%	3	0XFFFF
Not applicable	51704	C9F8	1	harmonic	0.10%	3	0XFFFF

Not applicable	51705	C9F9	1	harmonic	0.10%	3	0xFFFF
Not applicable	51706	C9FA	1	harmonic	0.10%	3	0xFFFF
Not applicable	51707	C9FB	1	harmonic	0.10%	3	0xFFFF
Not applicable	51708	C9FC	1	harmonic	0.10%	3	0xFFFF
Not applicable	51709	C9FD	1	harmonic	0.10%	3	0xFFFF
Not applicable	51710	C9FE	1	harmonic	0.10%	3	0xFFFF
Not applicable	51711	C9FF	1	harmonic	0.10%	3	0xFFFF
Not applicable	51712	CA00	1	harmonic	0.10%	3	0xFFFF
Not applicable	51713	CA01	1	harmonic	0.10%	3	0xFFFF
Not applicable	51714	CA02	1	harmonic	0.10%	3	0xFFFF
Not applicable	51715	CA03	1	harmonic	0.10%	3	0xFFFF
Not applicable	51716	CA04	1	harmonic	0.10%	3	0xFFFF
Not applicable	51717	CA05	1	harmonic	0.10%	3	0xFFFF
Not applicable	51718	CA06	1	harmonic	0.10%	3	0xFFFF
Not applicable	51719	CA07	1	harmonic	0.10%	3	0xFFFF
Not applicable	51720	CA08	1	harmonic	0.10%	3	0xFFFF
Not applicable	51721	CA09	1	harmonic	0.10%	3	0xFFFF
Not applicable	51722	CA0A	1	harmonic	0.10%	3	0xFFFF
Not applicable	51723	CA0B	1	harmonic	0.10%	3	0xFFFF
Not applicable	51724	CA0C	1	harmonic	0.10%	3	0xFFFF
Not applicable	51725	CA0D	1	harmonic	0.10%	3	0xFFFF
Not applicable	51726	CA0E	1	harmonic	0.10%	3	0xFFFF
Not applicable	51727	CA0F	1	harmonic	0.10%	3	0xFFFF
Not applicable	51728	CA10	1	harmonic	0.10%	3	0xFFFF
Not applicable	51729	CA11	1	harmonic	0.10%	3	0xFFFF
Not applicable	51730	CA12	1	harmonic	0.10%	3	0xFFFF
Not applicable	51731	CA13	1	harmonic	0.10%	3	0xFFFF
Not applicable	51732	CA14	1	harmonic	0.10%	3	0xFFFF
Not applicable	51733	CA15	1	harmonic	0.10%	3	0xFFFF
Not applicable	51734	CA16	1	harmonic	0.10%	3	0xFFFF
Not applicable	51735	CA17	1	harmonic	0.10%	3	0xFFFF
Not applicable	51736	CA18	1	harmonic	0.10%	3	0xFFFF
Not applicable	51737	CA19	1	harmonic	0.10%	3	0xFFFF
Not applicable	51738	CA1A	1	harmonic	0.10%	3	0xFFFF
Not applicable	51739	CA1B	1	harmonic	0.10%	3	0xFFFF
Not applicable	51740	CA1C	1	harmonic	0.10%	3	0xFFFF
Not applicable	51741	CA1D	1	harmonic	0.10%	3	0xFFFF
Not applicable	51742	CA1E	1	harmonic	0.10%	3	0xFFFF
Not applicable	51743	CA1F	1	harmonic	0.10%	3	0xFFFF
Not applicable	51744	CA20	1	harmonic	0.10%	3	0xFFFF
Not applicable	51745	CA21	1	harmonic	0.10%	3	0xFFFF
Not applicable	51746	CA22	1	harmonic	0.10%	3	0xFFFF
Not applicable	51747	CA23	1	harmonic	0.10%	3	0xFFFF
Not applicable	51748	CA24	1	harmonic	0.10%	3	0xFFFF
Not applicable	51749	CA25	1	harmonic	0.10%	3	0xFFFF
Not applicable	51750	CA26	1	harmonic	0.10%	3	0xFFFF
Not applicable	51751	CA27	1	harmonic	0.10%	3	0xFFFF
Not applicable	51752	CA28	1	harmonic	0.10%	3	0xFFFF
Not applicable	51753	CA29	1	harmonic	0.10%	3	0xFFFF
Not applicable	51754	CA2A	1	harmonic	0.10%	3	0xFFFF
Not applicable	51755	CA2B	1	harmonic	0.10%	3	0xFFFF
Not applicable	51756	CA2C	1	harmonic	0.10%	3	0xFFFF
Not applicable	51757	CA2D	1	harmonic	0.10%	3	0xFFFF
Not applicable	51758	CA2E	1	harmonic	0.10%	3	0xFFFF
Not applicable	51759	CA2F	1	harmonic	0.10%	3	0xFFFF
Not applicable	51760	CA30	1	harmonic	0.10%	3	0xFFFF
Not applicable	51761	CA31	1	harmonic	0.10%	3	0xFFFF
Not applicable	51762	CA32	1	harmonic	0.10%	3	0xFFFF
Not applicable	51763	CA33	1	harmonic	0.10%	3	0xFFFF
Not applicable	51764	CA34	1	harmonic	0.10%	3	0xFFFF
51671 C9D7 94							
V PH/N :							
Not applicable	51765	CA35	1	Number of		3	0
Not applicable	51766	CA36	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51767	CA37	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51768	CA38	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51769	CA39	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51770	CA3A	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51771	CA3B	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51772	CA3C	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51773	CA3D	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51774	CA3E	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51775	CA3F	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51776	CA40	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51777	CA41	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51778	CA42	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51779	CA43	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51780	CA44	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51781	CA45	1	harmonic V1	0.10%	3	0xFFFF

Not applicable	51782	CA46	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51783	CA47	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51784	CA48	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51785	CA49	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51786	CA4A	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51787	CA4B	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51788	CA4C	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51789	CA4D	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51790	CA4E	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51791	CA4F	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51792	CA50	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51793	CA51	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51794	CA52	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51795	CA53	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51796	CA54	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51797	CA55	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51798	CA56	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51799	CA57	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51800	CA58	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51801	CA59	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51802	CA5A	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51803	CA5B	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51804	CA5C	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51805	CA5D	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51806	CA5E	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51807	CA5F	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51808	CA60	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51809	CA61	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51810	CA62	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51811	CA63	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51812	CA64	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51813	CA65	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51814	CA66	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51815	CA67	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51816	CA68	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51817	CA69	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51818	CA6A	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51819	CA6B	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51820	CA6C	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51821	CA6D	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51822	CA6E	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51823	CA6F	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51824	CA70	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51825	CA71	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51826	CA72	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51827	CA73	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51828	CA74	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51829	CA75	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51830	CA76	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51831	CA77	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51832	CA78	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51833	CA79	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51834	CA7A	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51835	CA7B	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51836	CA7C	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51837	CA7D	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51838	CA7E	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51839	CA7F	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51840	CA80	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51841	CA81	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51842	CA82	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51843	CA83	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51844	CA84	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51845	CA85	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51846	CA86	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51847	CA87	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51848	CA88	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51849	CA89	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51850	CA8A	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51851	CA8B	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51852	CA8C	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51853	CA8D	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51854	CA8E	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51855	CA8F	1	harmonic V3	0.10%	3	0xFFFF
Not applicable	51856	CA90	1	harmonic V1	0.10%	3	0xFFFF
Not applicable	51857	CA91	1	harmonic V2	0.10%	3	0xFFFF
Not applicable	51858	CA92	1	harmonic V3	0.10%	3	0xFFFF

51765 CA35 94
51536 C950 323

E000 Hex : Network Setting

	Address Déc.	Address Hex.	Word Count	Description	Unit	Function	Value if not Available
	57344	E000	1	Network Type : 0 : 1BL 1 : 2BL 2 : 3BL 3 : 3NBL 4 : 4BL	—	3,6,16	0xFFFF
	57345	E001	1	Current	A	3,6,16	0xFFFF
	57346	E002	1	Current	A	3,6,16	0xFFFF
Not applicable	57347	E003	1	Neutral	—	3,6,16	0xFFFF
Not applicable	57348	E004	1	Neutral	A	3,6,16	0xFFFF
Not applicable	57349	E005	1	Neutral	A	3,6,16	0xFFFF
Not applicable	57350	E006	1	Voltage	—	3,6,16	0xFFFF
	57351	E007	1	Voltage Transformer secondary : 60 : 60 V 100 : 100 V 110 : 110 V 115 : 115 V 120 : 120 V	V	3,6,16	0xFFFF
Not applicable							
Not applicable	57352	E008	2	Voltage	V	3,6,16	0xFFFFFFFF
Not applicable	57354	E00A	1	Internal/Exter	0:internal /	3,6,16	0xFFFF
	57355	E00B	1	Synchronisat ion Top for P+/- Q+/- : time in seconds (2mn, 5mn, 8mn, 10mn, 15mn, 20mn, 30mn, 60mn)	secondes	3,6,16	0xFFFF
Not applicable							
	57344	E000	12				

E200 Hex : Action system (Write Function only)

Address Déc.	Address Hex.	Word Count	Description	Unit	Function	
57856	E200	1	Action : 0xA1 : Product Configuration storage 0xB2 : Produit reboot		6	
57856	E200	1				

SM101C specific tables**8D50 Hex : Alarme**

Address Dec.	Address Hex.	Word Count	Description	Unit	Function
--------------	--------------	------------	-------------	------	----------

36176	8D50	1	Current alarm on lower threshold cause: 0 : No Alarm / 1 : I1 / 2 : I2 / 3 : I3 / 4 : IN / 5 : U12 / 6 : U23 / 7 : U31 / 8 : $\Sigma P+$ / 9 : $\Sigma Q+$ / 10 : ΣS / 11 : F / 12 : ΣPFL / 15 : thdI1 / 16 : thdI2 / 17 : thdI3 / 18 : thdU12 / 19 : thdU23 / 20 : thdU31 / 21 : Hour / 22 : V1 / 23 :	0 : - 1 / 2 / 3 / 4 : mA 5 / 6 / 7 : mV 8 : mW 9 : mVAr 10 : mVA 11 : Hz/1000 12 : - 15 / 16 / 17 : /1000 18 / 19 / 20 : /1000 21 : Hour/100 22 / 23 / 24 : mV 31 : - 32 : °C/10	3
36177	8D51	2	Current alarm on lower threshold : min value	-	3
36179	8D53	1	alarm on upper threshold cause: 0 : No Alarm / 1 : I1 / 2 : I2 / 3 : I3 / 4 : IN / 5 : U12 / 6 : U23 / 7 : U31 / 8 : $\Sigma P+$ / 9 : $\Sigma Q+$ / 10 : ΣS / 11 : F / 12 : ΣPFL / 15 : thdI1 / 16 : thdI2 / 17 : thdI3 / 18 : thdU12 / 19 : thdU23 / 20 : thdU31 / 21 : Hour / 22 : V1 / 23 : V2 / 24 : V3 /	0 : - 1 / 2 / 3 / 4 : mA 5 / 6 / 7 : mV 8 : mW 9 : mVAr 10 : mVA 11 : Hz/1000 12 : - 15 / 16 / 17 : /1000 18 / 19 / 20 : /1000 21 : Hour/100 22 / 23 / 24 : mV 31 : - 32 : °C/10	3
36180	8D54	2	Current alarm on upper threshold : max value	-	3

36182	8D56	1	Current alarm duration	s	3
36176	8D50	7			

8E00 Hex : Table Setup

Address Dec.	Address Hex.	Word Count	Description	Unit	Function
			Network : 0 : 1BL 1 : 2BL 2 : 3BL 3 : 3NBL 4 : 4BL 5 : 4NBL		3,6,16
36352	8E00	1		/	
36353	8E01	1	Current Trans	A	3,6,16
36354	8E02	1	Current Trans	A	3,6,16
36355	8E03	1	Reserved		3,6,16
			Synchronisation of I AVG/MAX: 2 : 2 seconds 10 : 10 seconds 300 : 5 minutes 480 : 8 minutes 600 : 10 Minutes 900 : 15 minutes 1200 : 20 minutes 1800 : 30 minutes 3600 : 60 minutes		3,6,16
36356	8E04	1		/	
			Synchronisation of P/Q/S AVG/MAX 10 : 10 seconds 300 : 5 minutes 480 : 8 minutes 600 : 10 Minutes 900 : 15 minutes 1200 : 20 minutes 1800 : 30 minutes 3600 : 60 minutes		3,6,16
36357	8E05	1		/	
			OUT 1 : pulse output allocation : 0 : kWh+ 1 : kvarh + 2 : Alarm 3 : Command		3,6,16
36358	8E06	1		/	

36359	8E07	1	OUT 1 : pulse output value : 0 : 0,1 kWh/kvarh 1 : 1 kWh/kvarh 2 : 10 kWh/kvarh 3 : 100 kWh/kvarh 4 : 1000 kWh/kvarh 5 : 10000 kWh/kvarh		3,6,16
36360	8E08	1	OUT 1 : pulse output duration : 1 : 100ms - 2 : 200ms 3 : 300ms - 4 : 400ms 5 : 500ms - 6 : 600ms 7 : 700ms - 8 : 800ms 9 : 900ms		3,6,16
36361	8E09	1	Hour meter allocation 1 : Auxiliary power supply 2 : Currents 3 : phase to phase voltage		3,6,16
36362	8E0A	1	Hour meter trigger threshold	A/V	3,6,16
36363	8E0B	1	Alarm Type : 1 : I 2 : In 3 : U 4 : V 5 : $\Sigma P+$ 6 : $\Sigma Q+$ 7 : $\Sigma S+$ 8 : ΣPFC 9 : ΣPFL 5 : P 6 : Q 7 : S 8 : CPF 9 : LPF 10 : THDU 11 : THDV 12 : THDI 13 : HOUR 14 : F 15 : Internal temperature		3,6,16
36364	8E0C	1	Alarm Specified time (0-999)		3,6,16
36365	8E0D	1	Alarm upper Threshold		3,6,16
36366	8E0E	1	Alarm Lower Threshold		3,6,16
36367	8E0F	1	Alarm Hysteresis(0- 99)		3,6,16
36368	8E10	1	Relay State : 0 : Open 1 : Closed		3,6,16
36352	8E00	17			

400 Hex : Reset (These values could be combine all)

Address Dec.	Address Hex.	Word Count	Description	Unit	Function
1024	400	1	Reset : Max 4I : 0x1 Max P+,Q+,S : 0x2 kWh+ : 0x80 kvarh+ : 0x100 tous les paramètres : 0x1000	/	6
1024	400	1			

C691 Hex : Energy meters residual values (current kWh & kvarh)

Address Dec.	Address Hex.	Word Count	Description	Unit	Function
50833	C691	1	Ea+ Residual	0,01 kWh	3
50834	C692	1	Er+ Residual	0,01 kvarh	3
50835	C693	1	Dummy		3
50836	C694	1	Dummy		3
50837	C695	1	Dummy		3
50833	C691	5			

1000 Hex : Customization specific data

Address Dec.	Address Hex.	Word Count	Description	Unit	Function
4096	1000	1	Customisation specific data loaded 0x0000 : FALSE 0x0001 : TRUE		3
4097	1001	1	Retrofit activated 0x0000 : FALSE 0x0001 : TRUE		3
4098	1002	8	TC list : TC 1 / TC2 / TC3 / TC4 / TC5 / TC6 / TC7 / TC8	A	3
4106	100A	16	TC gain correction X1 2 words = 1 TC correction		3
4122	101A	16	TC gain correction X2 2 words = 1 TC correction		3

2

APPENDIX

C

ESP8266 PIN LIST

The pin list release from ESP8266 microcontroller will be next included:

	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
	Inst Name	Side	Coordinate	X	Function1	Type	Function2	Type	Function3	Type	Function4	Type	Function5	Type	At Reset	After Reset	Sleep
1																	
2	MTDI_U				MTDI	I	I2SI_DATA	I/O/T	HSPIQ_MISO	I/O/T	GPIO12	I/O/T	U0DTR	O	oe=0, wpu	wpu	oe=0
3	MTCK_U				MTCK	I	I2SI_BCK	I/O/T	HSPIQ_MOSI	I/O/T	GPIO13	I/O/T	U0CTS	I	oe=0, wpu	wpu	oe=0
4	MTMS_U				MTMS	I	I2SI_WS	I/O/T	HSPICLK	I/O/T	GPIO14	I/O/T	U0DSR	I	oe=0, wpu	wpu	oe=0
5	MTDO_U				MTDO	O/T	I2SO_BCK	I/O/T	HSPICS	I/O/T	GPIO15	I/O/T	U0RTS	O	oe=0, wpu	wpu	oe=0
6	U0RXD_U				U0RXD	I	I2SO_DATA	I/O/T		O	GPIO3	I/O/T	CLK_XTAL	O	oe=0, wpu	wpu	oe=0
7	U0TXD_U				U0TXD	O	SPICS1	I/O/T		O	GPIO1	I/O/T	CLK_RTC	O	oe=0, wpu	wpu	oe=0
8	SD_CLK_U				SD_CLK	I	SPICLK	I/O/T		O	GPIO6	I/O/T	U1CTS	I	oe=0		oe=0
9	SD_DATA0_U				SD_DATA0	I/O/T	SPIQ	I/O/T		O	GPIO7	I/O/T	U1TXD	O	oe=0		oe=0
10	SD_DATA1_U				SD_DATA1	I/O/T	SPID	I/O/T		O	GPIO8	I/O/T	U1RXD	I	oe=0		oe=0
11	SD_DATA2_U				SD_DATA2	I/O/T	SPIHD	I/O/T		O	GPIO9	I/O/T	HSPIHD	I/O/T	oe=0		oe=0
12	SD_DATA3_U				SD_DATA3	I/O/T	SPIWP	I/O/T		O	GPIO10	I/O/T	HSPIWP	I/O/T	oe=0		oe=0
13	SD_CMD_U				SD_CMD	I/O/T	SPICS0	I/O/T		O	GPIO11	I/O/T	U1RTS	O	oe=0		oe=0
14	GPIO0_U				GPIO0	I/O/T	SPICS2	I/O/T		O		I/O/T	CLK_OUT	O	oe=0, wpu	wpu	oe=0
15	GPIO2_U				GPIO2	I/O/T	I2SO_WS	I/O/T	U1TXD	O		I/O/T	U0TXD	O	oe=0, wpu	wpu	oe=0
16	GPIO4_U				GPIO4	I/O/T	CLK_XTAL	O							oe=0		oe=0
17	GPIO5_U				GPIO5	I/O/T	CLK_RTC	O							oe=0		oe=0
18																	
19	XPD_DCDC				XPD_DCDC	O	RTC_GPIO0	I/O/T	EXT_WAKEUP	I	DEEPSLEEP	O	BT_XTAL_EN	I	oe=1, wpd	oe=1, wpd	oe=1

ANNOTATIONS

1. INST_NAME indicate the IO_MUX REGISTER , for example MTDI_U refers to PERIPHS_IO_MUX_MTDI_U
2. NET NAME accords with the pin name in schematic.
3. FUNCTION says the multifunction of each pin pad.
4. Function numbers 1-5 in this table correspond to FUNCTION 0-4 in SDK.

APPENDIX

D

PROJECT BUDGET

D.1 Software

This section will describe the prices of the software licenses used, listed in table [D.1](#).

Software	Owner	Cost €
Altium designer 14.3	UGR	0
FreeCAD	Open Source	0
Arduino IDE	Open Source	0
Inkscape	Open Source	0
Texmaker	Open Source	0
TOTAL		0

List of Tables D.1 – *Product software costs*

D.2 Human Resources

Since this work is part of a Bachelor Thesis, we will consider the case where a junior engineer was contracted for this project, working 9 months full time, for about 10 €/ hour. Another senior engineer was contracted as a supervisor for 50 €/ hour. This is presented in table [D.2](#).

Charge	Hours	Cost €
Junior Engineer	1440	14400
Senior Engineer	180	9000
TOTAL		23400

List of Tables D.2 – Human cost of product

D.3 Components cost

This section will list all needed components as well as the manufactured PCBs costs. It is presented in table D.3.

Item	Description	Units	Cost/ Units(€)	Used Cost (€)
ESP-12F	Microcontroller Module	1	3.32	3.32
Arduino UNO	Debug Module	1	20	20
OLED Display	Interface Module	1	2.27	2.27
microSD Card Holder	Storage Module	1	3.15	3.15
MAX485 converter	MODBUS-TTL converter	4	1.71	6.84
SMPS	Power Module components	28		8
Designed Boards	Manufactured PCBs	2	5	10
			Total	53.58

List of Tables D.3 – Component cost

REFERENCES

- [1] Eurotronix, “Reference smps implemented.” Datasheet.
- [2] G. P. E. Network, “Comparative power supply.” [Design Idea](#). Accessed 2018.07.02.
- [3] altronix, “Max485 module image.” [Product Info](#). Accessed 2018.08.02.
- [4] M. Integrated, “Max485 datasheet.” [Datasheet](#). Accessed 2018.08.02.
- [5] Ebay, “Rotary encoder module.” [Product Info](#). Accessed 2018.07.25.
- [6] HowToMechatronics, “Rotary encoder working principle images.” [Online Documentation](#). Accessed 2018.07.25.
- [7] iberobotics.com, “Arduino uno rev3 image.” [Product Info](#). Accessed 2018.07.05.
- [8] Acrobotic, “Esp-12 module pinout.” [Online Documentation](#). Accessed 2018.06.10.
- [9] Acrobotic, “Esp-12 module close-up image.” [Online Documentation](#). Accessed 2018.06.10.
- [10] T. Instruments, “Snx5hvd1x 3.3v rs-485 transceivers.” [Datasheet](#). Accessed 2018.08.05.
- [11] T. Instruments, “Thvd1500 rs-485 transceivers.” [Datasheet](#). Accessed 2018.08.05.
- [12] M. Electronics, “Sd memory card holder specifications.” [Datasheet](#). Accessed 2018.08.10.
- [13] R. Components, “microsd memory card holder specifications.” [Datasheet 1](#) and [2](#). Accessed 2018.08.11.

References

- [14] M. Electronics, “Display lcd 16x2 specifications.” [Datasheet](#). Accessed 2018.07.22.
- [15] Ebay, “Display oled 128x64 i2c white specifications.” [Product Info](#). Accessed 2018.07.23.
- [16] Ebay, “Display oled 128x64 two-coloured i2c specifications.” [Product Info](#). Accessed 2018.07.23.
- [17] store.arduino.cc, “Arduino uno rev3 technical specifications.” [Product Info](#). Accessed 2018.07.05.
- [18] raspberry.org, “Raspberry pi 3b+ specifications.” [Datasheet](#). Accessed 2018.07.01.
- [19] adafruit, “Beaglebone black board specifications.” [Reference Manual](#). Accessed 2018.06.27.
- [20] Espressif, “Esp32 module datasheet.” [Datasheet](#). Accessed 2018.07.06.
- [21] elecrow, “Esp-12 wifi module datasheet.” [Datasheet](#). Accessed 2018.06.11.
- [22] “Esp8266wifi library - eeprom.” [Online documentation](#). Accessed 2018.08.10.
- [23] M. Electronics, “Display lcd 16x2 image.” [Product Info](#). Accessed 2018.07.22.
- [24] amazon.com, “128x64 oled display white coloured image.” [Product Info](#). Accessed 2018.07.20.
- [25] amazon.com, “128x64 oled display two-coloured image.” [Product Info](#). Accessed 2018.07.20.
- [26] reichelt.com, “Raspberry pi 3b+ image.” [Product Info](#). Accessed 2018.07.01.
- [27] ameridroid.com, “Beaglebone black board image.” [Product Info](#). Accessed 2018.06.27.
- [28] ebay, “Esp32 module image.” [Product Info](#). Accessed 2018.07.06.
- [29] electrodragon.com, “Image about the esp-12f.” [Product Info](#). Accessed 2018.06.10.
- [30] beaglebone.cameon.net, “Beaglebone black board gpios.” [Online Documentation](#). Accessed 2018.06.28.
- [31] JLPCB, “Jlpcb company web page.” [Web Page](#). Accessed 2018.08.20.
- [32] “Python official web page.” [Data Package](#). Accessed 2018.08.08.
- [33] “Minimalmbus library documentation.” [Online Documentation](#). Accessed 2018.06.20.
- [34] H. Helfawi, “Spi flash modes.” [Repository Documentation](#). Accessed 2018.07.08.
- [35] T. Instruments, “Max3232 line driver/receiver.” [Datasheet](#). Accessed 2018.08.22.
- [36] espressif, “Esp8266 bootloader utility installation.” [GitHub Repository](#). Accessed 2018.08.22.