المدرسة الوطنية للمهندسين بسوسة
École Nationale d'Ingénieurs de Sousse

National Engineering School of Sousse

UNIVERSITY OF SOUSSE

End of Studies Dissertation

Specialty: **Industrial Electronics**

Programmer and Debugger for ESP32

Realized by:

Mohamed GUENI

Academic Supervisor: Dr. Bouraoui MAHMOUD
Industrial Supervisor: Mr. Hamdi BEN AMOR

Defended on: **00/09/2019**

JURY

Dr. ZAIDI NOUR EDDAHR (President)
Dr. BOUALLEGUE ADEL (Rapporteur)

Academic Year: 2017/2018

Dedication

I dedicate my dissertation work to my family and many friends. A special feeling of gratitude to my loving mother, Halima, whose words of encouragement and push for tenacity ring in my ears.

To my sisters Imen, Shayma and my brother Wadid who first taught me the value of education and critical thought. They are the ultimate role models.

I dedicate this work and give special thanks to my Academic Supervisor Dr. Bouraoui MAHMOUD and Industrial Supervisor Mr. Hamdi BEN AMOR for being there for me throughout the entire process and helped develop and shape my entire skill set.

I consider this humble work as a tribute to all my friends and specially those who have been my greatest support in my academic years as well as my personal life.

Acknowledgments

This work would have never been possible without the help and guidance of many people with whom I have had the pleasure to work during this project.

First and foremost, I would like to express my greatest gratitude **to Mr. Hamdi BEN AMOR**, my industrial supervisor at Micro Device Tunisie, for providing guidance and feedback throughout this project and for teaching me a great deal along my path and that inspired me to reach my full potential.

I would like to sincerely thank to **Dr. Bouraoui MAHMOUD**, my academic supervisor for his guidance and support throughout this entire internship and specially for his confidence in me.

Finally, I would like to thank the jury members for putting the time and effort in evaluating this humble work.

Abstract

We're entering a brand-new era of computing technology known as the Internet of Things (IoT) and its potential is extremely important towards technological advancement and will eventually encompass every aspect of our lives. Its main foundation, in a way, is the simple yet becoming very essential, intelligence that embedded devices and embedded processing provides.

The Internet of Things represents a vast market for many world-wide known pioneers, among those we shall not pass without mentioning Espressif Systems, the multinational semiconductor company which have created the popular ESP series of chips and most importantly the ESP32. Powerful Wi-Fi and Bluetooth modules that target a wide variety of applications, ranging from low-power sensor networks to the most demanding tasks. while this series of chips are widely used, unfortunately there still no proper solution to how to program it and/or debug it considering professional and industrial tools.

The project presented in this document and proposed by the company Micro Device Tunisie is developed for Programming and debugging these chips in real time. Our focus is to deliver both a hardware and software solution not only capable of what is mentioned before but also well designed and suited for industrial purposes.

Keywords: ESP32, Hardware Design Flow, PyQt, Software Developement,3D Design.

Résumé

Nous entrons dans une toute nouvelle ère de technologie informatique appelée Internet des objets (IoT) et son potentiel est extrêmement important pour le progrès technologique et englobera tous les aspects de notre vie. D'une certaine manière, son fondement repose sur l'intelligence simple, mais essentielle, que fournissent les appareils et les systèmes embarqués.

L'Internet des objets représente un vaste marché pour de nombreuses entreprises mondialement connues, parmi ceux que nous ne passerons pas sans mentionner Espressif Systems, la multinationale des semi-conducteurs qui a créé la série de puces populaires ESP, et plus particulièrement l'ESP32. De puissants modules Wi-Fi et Bluetooth destinés à une grande variété d'applications, allant des réseaux de capteurs à faible consommation aux tâches les plus exigeantes. Bien que cette série de puces soit largement utilisée, malheureusement, il n'existe toujours pas de solution adéquate pour la programmer et / ou la corriger en tenant compte des outils professionnels et industriels.

Le projet présenté dans ce document et proposé par la société Micro Device Tunisie est développé pour la programmation et le débogage de ces puces en temps réel. Notre objectif est de fournir à la fois une solution matérielle et logicielle capable non seulement de ce qui a été mentionné précédemment, mais également bien conçue et adaptée aux besoins industriels.

Mots clés : ESP32, Flux de conception matérielle, PyQt, Développement de logiciels, Conception 3D.

Contents

List of Abbreviations

- **IoT**: Internet of Things
- **EEPROM**: Electrically Erasable Programmable Read Only Memory
- **FTDI: Future** Technology Devices
- MPSSE: Multi-Purpose Synchronous Serial Engine
-

List of Figures

List of Tables

Introduction

Chapter I

Bibliographic Research

Contents

Introduction

Discerning the remarkable synergy between the field of embedded systems and the elegant internet of things, was the essential stimuli behind the idea of this project.

This end of studies project, entitled “Programmer and Debugger for ESP32”, represents the backbone and a cornerstone of a much larger merger between both fields of embedded systems and the Internet of Things to design and create even more innovative products.

The subject covered in this chapter is required to understand the following chapters. It covers aspects related to the embedded systems, information about the chip (ESP32) which the project targets, as well as existing solutions on the market, followed by the host company’s work within the intersection of these two fields.

Finally, we’ll go through the project description and specifications as the last step in this chapter.

1 Preliminary Study

1.1 ESP32 Chip

ESP32 is a series of low-cost and low-power SoC microcontrollers . it integrates Wi-Fi and dual-mode Bluetooth technologies . Equipped with a Tensilica Xtensa LX6 microprocessor and includes in-built antenna switches ,power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company. ESP32 is capable of functioning reliably in industrial environments. Engineered for mobile devices, wearable electronics and IoT applications, ESP32 achieves ultra-low power consumption with a combination of several types of proprietary software. ESP32 can perform as a complete standalone system or as a slave device to a host MCU, reducing communication stack overhead on the main application processor [reférence]

1.2 Existing Solutions

IoT solutions have largely had to be built from scratch with a high degree of customization to specific requirements, which has driven up the cost and complexity of development and deterred many prospective entrants to the market.

What have been missing are developer tools that alleviate the costs associated with building. the foundational infrastructure the of their solutions so they can focus on
optimizing the core functionality and bring solutions to market more quickly with less cost.

Up until now, there has been all sorts of unprofessional and none practical techniques and hacks involved in programming an ESP32 chip. But never an actual hardware solution mainly designed for industrial purposes.

On the other hand, only the software part of the solution exists. Flash Download Tool, a desktop application made by Espressif Systems based on esptool libraries and wxpython GUI API. However, not taking advantage of the whole libraries unlike the project in this document which not only implements all important features of the esptool libraries but also focuses on the UI too making it more user friendly, modern and practical.

Micro Device Tunisie is addressing these challenges with new solution that have the potential to expand the market for IoT by organizing, simplifying and reducing the complexity of manufacturing and configuring ESP32 chips. This document outlines the challenges that IoT poses for industries, and how Micro Device Tunisie solution can help overcome them.

2 Host company: Micro Device Tunisie

Facing a great deal of challenges and seizing massive opportunities in the intersections of IoT and Embedded systems, MDT has always been thinking beyond the ordinary ways and providing precise and carefully tailored solutions.

Figure I.1 – Micro Device Tunisie's Logo

Micro Device Tunisie (MDT) is a young startup established in 2014, in Sousse, Tunisia. MDT provides service to the industrial sector such as studying, designing and prototyping product in the launch phase or improvement and optimization. The company's work aims at both Hardware and Software Solutions and the interaction between them.

3 Project description

This project was proposed by the host company, Micro Device Tunisie, and it mainly consists of designing and creating a prototype of an electronic device fully capable of connecting the ESP32 based modules to a computer in order to program its memory, flash a new firmware / Bootloader into it and/or perform some debugging procedures.

Meanwhile, the project also comprises a major part in which the development of a piece of software is needed. the software is supposed to simplify the firmware flashing procedure inside industrial locations by presenting two operating modes. First mode is the simplest and it's designed to be used by workers in production departments whereas an advanced mode is introduced for more complex configurations.

The figure I.2 shows the project diagram where the left block indicates the Hardware part and the right one shows the software and its main functionalities.

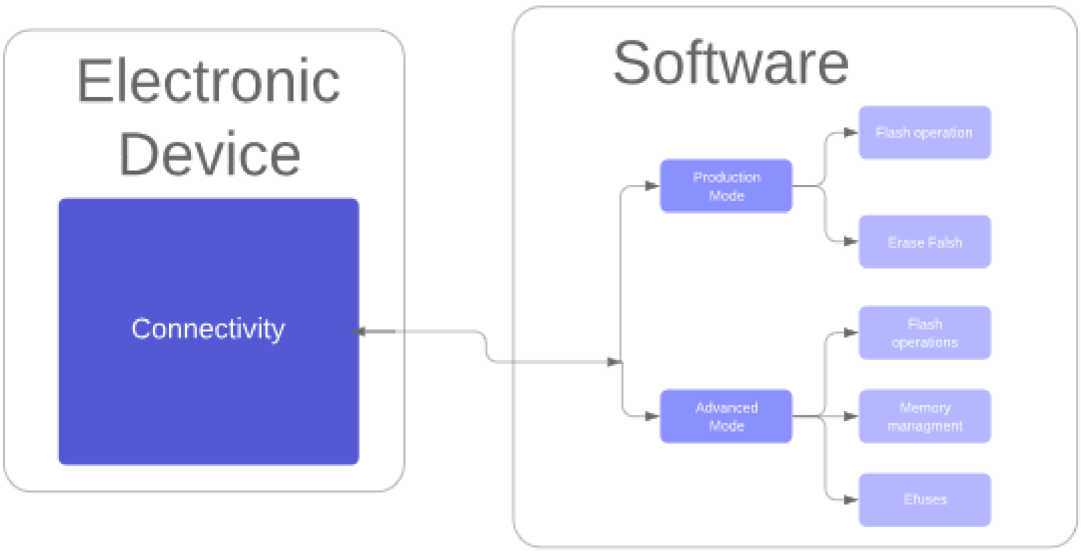


Figure I.2 – Project functionalities diagram

4 ~~End of studies~~ project specifications

For us to describe such large project efficiently, we need to beak it down into multiple parts. Therefore, we were requested to execute the following:

- Study the possible solution for the problem
- Draft system architecture (Blocks diagram)
- Choose Components, create and store them in database for further use.
- Create and optimize schematic
- Move to PCB layout and 3D Model hand in hand.
- Design user interface and implement all necessary functionalities.

Conclusion

This first chapter presented a brief introduction to the target chip of our work followed by shedding the light on the existing solutions available on the market. We’ve had the opportunity also to highlight the host company, Micro Device Tunisie, and its field of work and important technological contributions in those respected fields.

Last but not least, we went through a brief description of the project in hand followed by the specifications.

ChapterII

Hardware Solution Development & 3D Design

Contents

Introduction

Unlike the previous chapter, in which we had gone through the introductory phase and the main idea addressed by the project. During this second chapter will define the hardware architecture of the soon to be presented product and will eventually cover the 3-dimensional design for the product and the major alterations made throughout the process.

1 Hardware Solution Development

1.1 Electronic System Architecture

The Figure II.3 describes the general architecture of the desired electronic system. It consists of six major building blocks:

- **Power Supply and USB communication port:** for powering the device the most flexible and suitable way to do so is by using the USB Bus powered configuration which gets its power from the USB bus (+5V) and since no heavy load will be connected to the input/output of the system an external power supply won't be necessary, especially when it's more convenient and practical to use with a computer.

- **ESD Protection circuit:** Although the system is to be used inside industrial sites in most of which an ESD protection tools and protocols are already taken into consideration. We went an extra mile here to include an ESD protection circuit both at the input and the output of the system to limit any kind of electrostatic discharge from damaging the vital parts of the circuit from both ends.

- **USB to UART/JTAG Communication:** this by far is the most intuitive block in the system diagram. In order to go from USB communication protocol to UART/JTAG we need a chip to do so. We will discuss which chip we are going to choose for this task in the following section.

- **Complementary Blocks:** Meanwhile some blocks in the system would need complementary circuits in order to fully function with an extremely small chance of failing. Among those, we mention external oscillators, bypass capacitors, filters, pull-ups etc.…

- **Interaction Interface:** For us to switch between the ESP chip modes we need a two-switch circuit which will allow for entering the boot mode and reset functionalities by following a sequence of clicks. We will introduce a more sophisticated circuit for that purpose in the coming sections.

- **I/O Connectivity Ports:** as intuitive as it might seem the system needs a bidirectional port to be able to connect and communicate with the target ESP chip and in some cases to power it too.

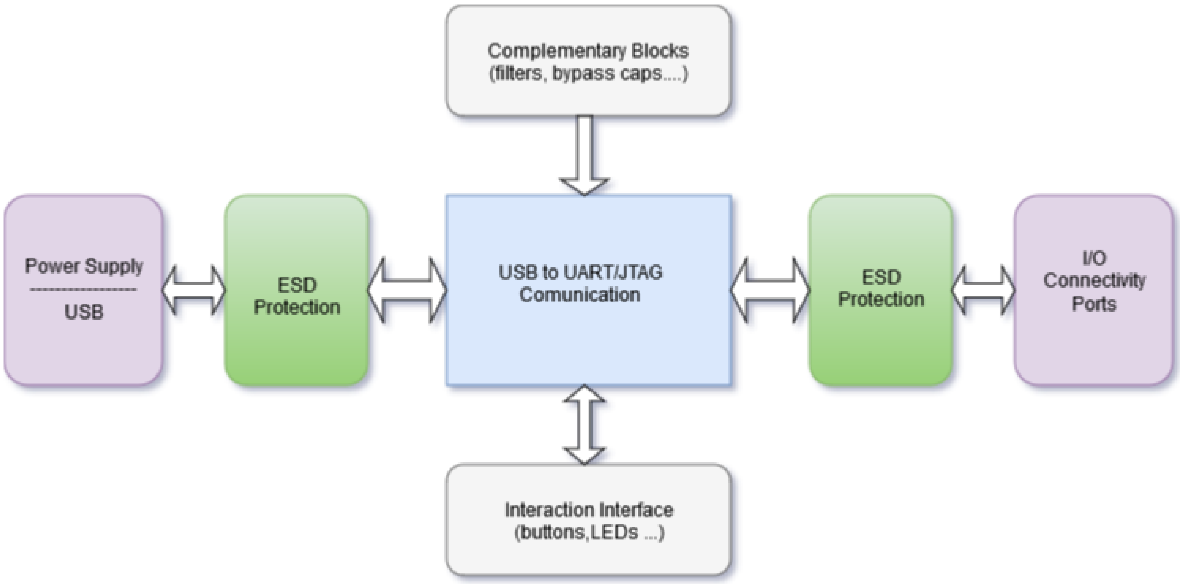


Figure II.3 – Electronic System Architecture

1.2 Component Selection

Aside from being the longest task during this project, it is also a critical step towards a well-designed electronic system. Throughout this phase we had gone through a variety of components and manufacturers in order to get to the most convenient and efficient set of components.

In this section, we shall not go through all the choices but only the ones involving the vital parts of the system.

- **FT2232:** The FT2232H may be used to convert one USB port to either 2 UARTs, 2 FIFOs, 2 MPSSE or a combination of these interfaces. We could have gone with just a microcontroller for the central block in the diagram in the Figure II.3. However, we were aiming at satisfying the duality of having a UART and a JTAG output/input. After going through many IC's, we picked FTDI's chip designated FT2232 which happens to fill the criteria mentioned above. The Block Diagram in the Figure II.4 bellow shows in fact two separate channels respectively designated A and B each of them is configurable in a variety of industry standard serial or parallel interfaces. In addition, it comprises a USB Protocol Engine hence the Entire USB protocol is handled on the chip, so No USB specific firmware programming is required. [référence]

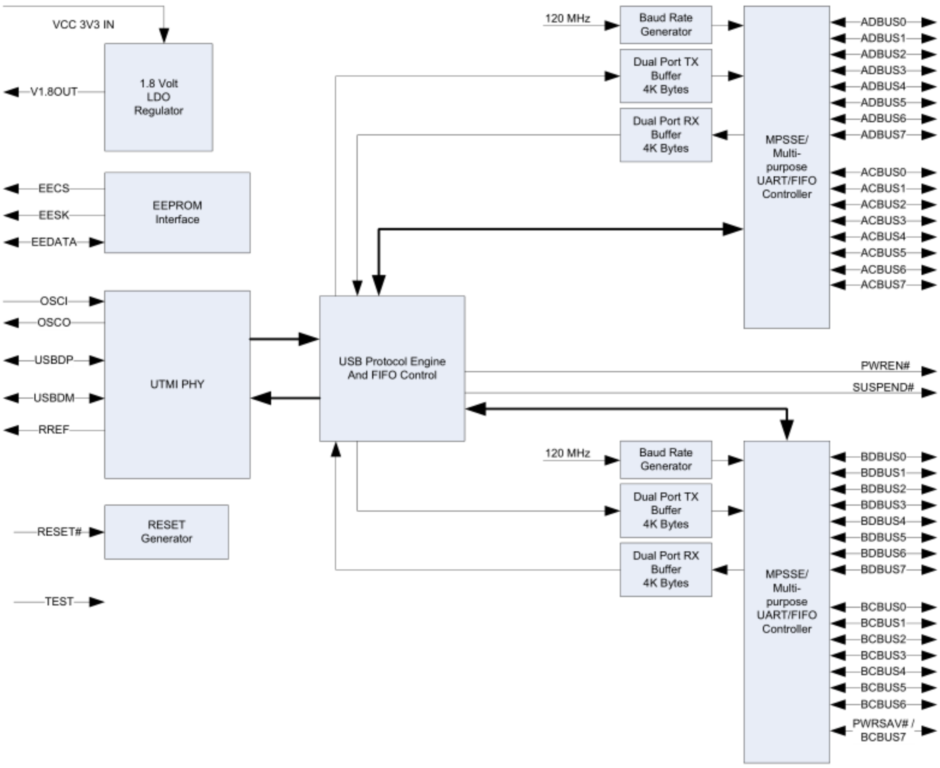


Figure I.4 – FT2232H Block Diagram [référence]

- **EEPROM 93C46:** The FTDI FT2232H chips are usually fitted with an EEPROM which is then used for configuring descriptors and operational parameters. When used without an external EEPROM interface, the FT2232H automatically defaults to a USB to dual asynchronous serial port device. Thus, adding an external EEPROM allows us to configure each of the chip's channels independently either as a serial UART (RS232 mode), parallel FIFO mode or fast serial. However, we can also exploit other aspects such as configuring the USB serial number, product description strings etc.…[référence]

- **Voltage Regulator:** As we mentioned before, we had gone with a USB bus powered configuration. Since the FT2232H requires the USB +5V to be regulated down to the necessary +3.3V, a voltage regulator should be added immediately after the power bus to satisfy this requirement. All other major blocks in the FT2232H are powered with either +3.3V from the external voltage regulator or from the on chip +1.8V regulator.[référence]

- **ESD Suppressors / TVS Diodes**: For ESD protection we went with a couple of ON Semiconductor's TVS diodes for every output / input signal instead of just using a single IC for maintenance purposes. As for the USB port we chose the STM’s USBLC6-2SC6 ESD suppressor IC for the same task. [reférence]

- **USB Port:** an SMT USB with through hole leads from Molex to be less exposed to any radial movement related fracture and to be intact with the PCB later on.

1.3 Component Creation and Database

At the previous phase in the project’s life cycle, the electronic system design was rather blocks containing the essential components for the well-functioning of the system. In order to move to the next phase in the design, we needed to provide an adequate model and footprint for each component and store them along with their parameters. Thus, a components database is needed to arrange every component in its adequate category. The components database included different parameters such as library and footprint references, manufacturer, description and so on. This database is then used to generate instances of the project components whenever needed. This way the project documents can be moved without any change in the schematics and without any loss.

Using Altium Designer as the tool to design the electronic system was the ultimate choice.

Since it’s a very powerful and practical tool to work with and considered the one which most

professionals use for PCB design and such.

The Figure II.5 shows one of the components’ both symbol and footprint which will then be used to create both the circuit schematic and PCB.

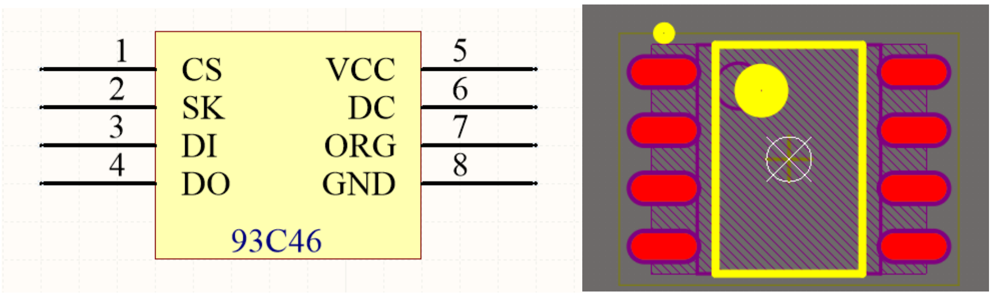


Figure II.5 – From left to right, component’s symbol and footprint

For the components’ 3D bodies, there was no need for us to reinvent the wheel, so we either used Altium’s 3D body generating tools or we acquired them from online sources since it’s not a critical part of the project. The Figure II.6 shows the 3D body of the EEPROM 93C46.

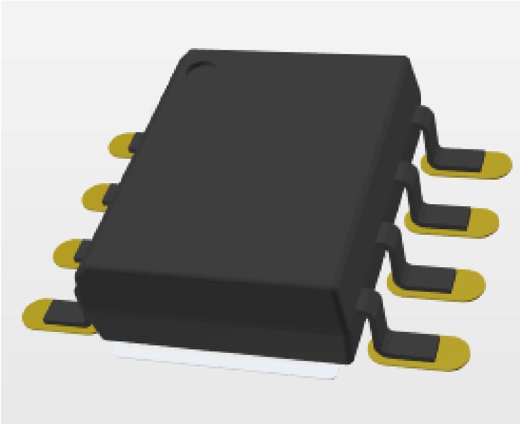


Figure II.6 – EEPROM 93C46 3D body

1.4 Electronic Circuit Schematic

Building the schematic follows a specific modular approach consisting in building separate sheets and then connecting them using ports. The schematic is then broken down into four separate sheets each of them serves a unique function.

- **USB/Power Connector circuit:** the USB connector and power supply is the least complicated circuit of the schematic. Yet it's a vital part of the design not to mention very practical also since it relies on USB power bus. The Figure II.7 shows the chosen SMT USB port from Molex where both its data and power connections are protected by an ESD suppressor IC from STMicroelectronics to minimise if not eleminate the risk of damaging both the voltage regulator and the FTDI IC for that matter.

  The voltage regulator, fitted on both its ends with two identical 0.1 uF decoupling capacitors, delivers on its end a stable +3.3 volts to the rest of the circuit while taking as an input the USB +5 volts bus. Meanwhile, to notify the user that the device is properly powered we added a red LED, as a state indicator, with a protection resistor pulled down to the ground.

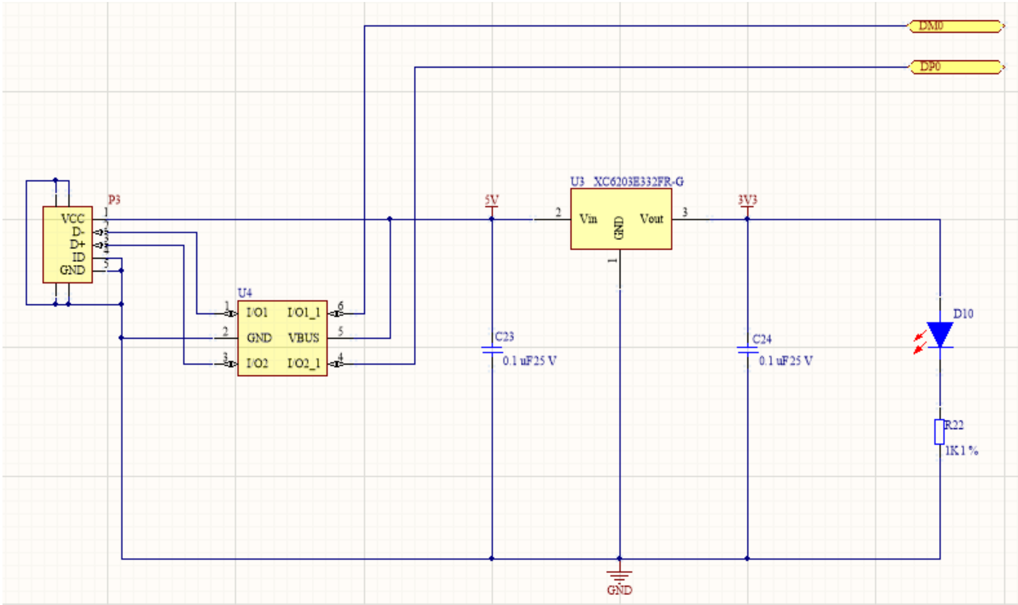


Figure I.7 – USB/Power Connector circuit schematic

- **FT2232 USB to UART/JTAG Circuit:** as shown in the Figure II.8, the FT2232 , placed at the center makes up an important part in the schematic along with the EEPROM IC at the bottom left of the sheet equiped with three 10K pull-up resistors connecting the chip select (CS) , clock (CLK) and the data (DI/DO) pins to the +3.3 volts. It is possible to connect the Data In and Data Out pins. The data in and data out pins are connected together as instructed in the Microchip's 93C46 datasheet however with this configuration it's obvious that a bus conflict bewtween both pins could occur hence a resistor should be connected between DI and DO to limit the current between these two.
  The number of bypass capacitors in this circuit is relatively noticeble for there are more power pins present in the FTDI CI not to mention the sensitivity of the chip twowards tiny voltage drops caused by fluctuation in the delivered power .

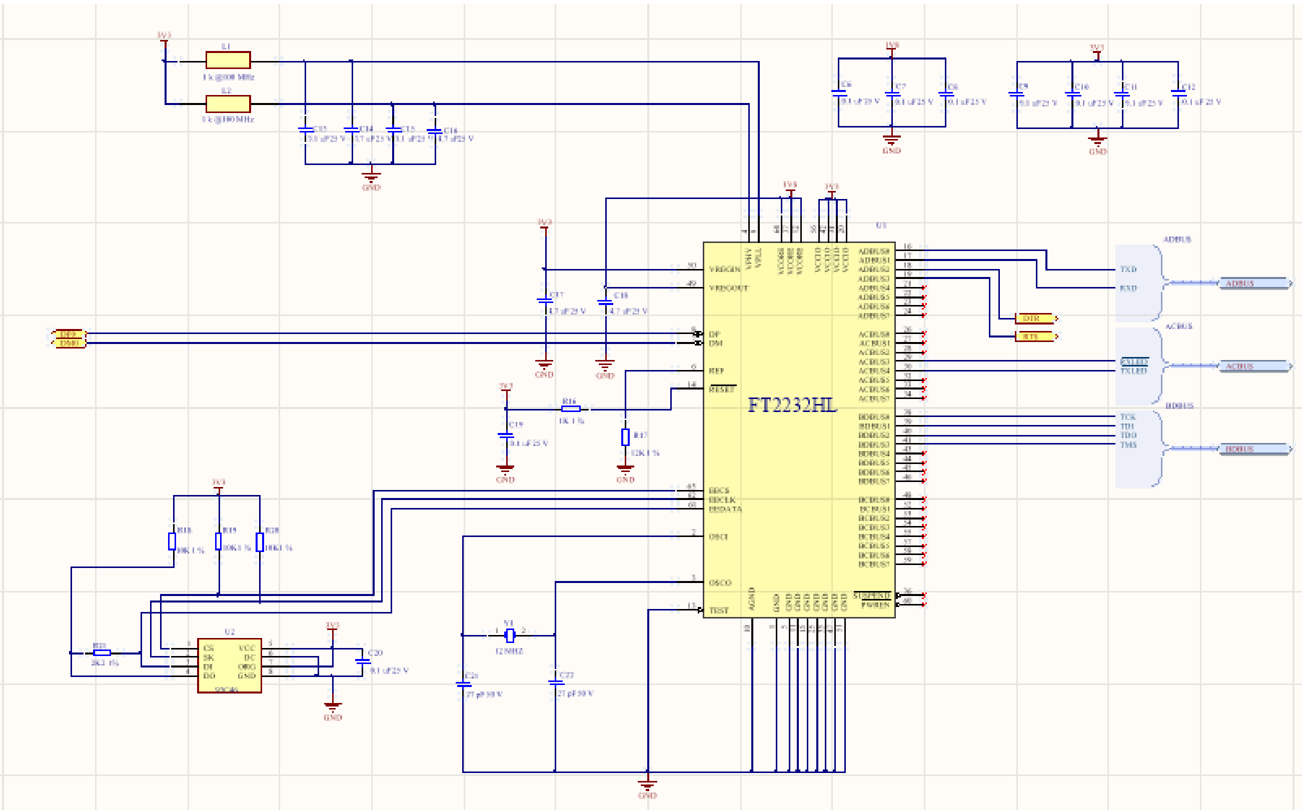
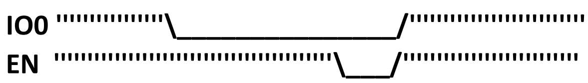


Figure II.8 – FT2232 USB to UART/JTAG Circuit schematic

According to FTDI the recommended FT2232H external crystal oscillator configuration .the Figure II.8 illustrates the connection between the FT2232H and a 12MHz ± 0.003% crystal along with two identical 27pF loading capacitors fitted between OSCI , OSCO and GND.

- **Auto program circuit:** RST and GPIO0 pins are used to reset and enter the bootload mode on target ESP device before programming starts. It's always hard to enter bootload mode by hand. Press FLASH button and hold, press and release RESET button, release flash as shown below. Then uploading the sketch.



The ESP32 flashing mode which should be automatically entered using a USB serial interface such as the one we're working with. For this purpose we added a resistor / transistor bridge circuit from DTR and RTS to properly toggle the ESP32 EN and IO0 pins. In order to enter serial flash upload mode, a specific sequence must be executed . IO0 needs to be held low, EN is pulsed low, then IO0 is brought high. The ESP32 will enter the serial bootloader when GPIO0 is held low on reset.Otherwise it will run the program in flash.

DTR	RTS	EN	IO0
1	1	1	1
0	0	1	1
1	0	0	1
0	1	1	0

Table II.1: logic table for mode selection

When GPIO0 pin is pulled low the ROM serial bootloader mode is activated whereas connecting it to VCC activates normal execution mode. If both DTR and RTS are pulled low, then neither GPIO0 nor EN are actively pulled down. Which means both DTR and RTS must be oppositely engaged in order for either output to be actively pulled down (one or the other, but not both).

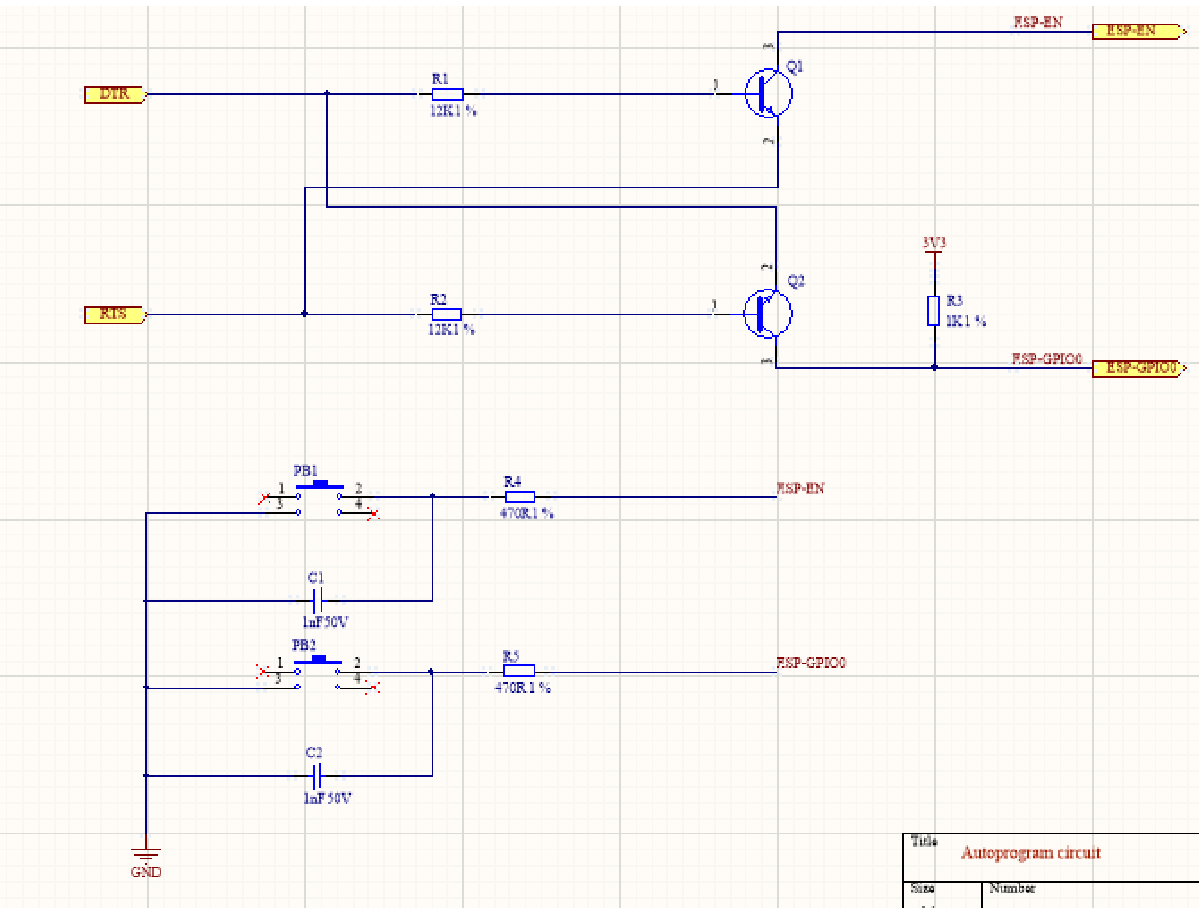


Figure II.10 Auto program circuit schematic

- **USB and JTAG Connectors circuit:** Figure I.11 illustrates the last part in the schematic is the two 2mm pitch male headers as the bidirectional data transfer ports fitted with a couple of ESD protection TVS diodes and an SMT bicolor LED module to indicate the receive and transfer operations (RX/TX).

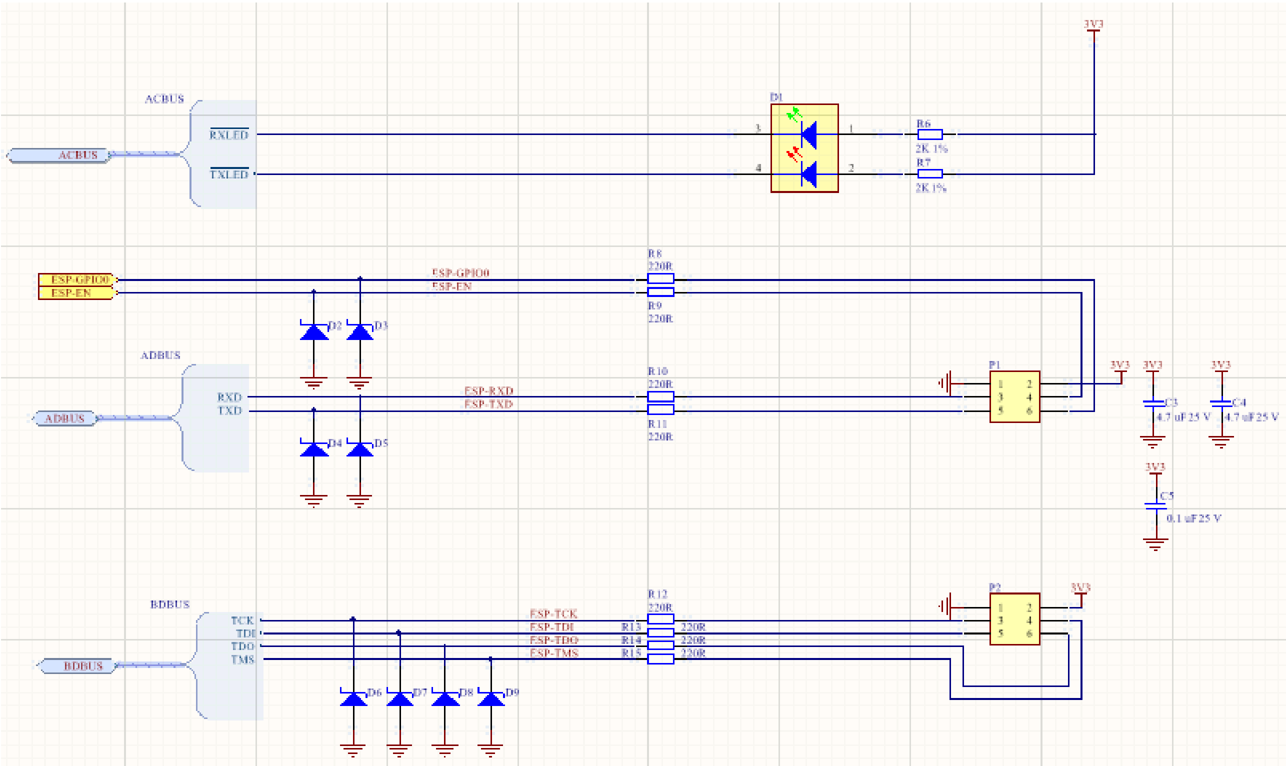


Figure II.11 – Input/output Connectors Circuit schematic

1.5 PCB Layout and Routing

At a preliminary stage of the PCB design phase, the board layout and dimensions were carefully tailored accordingly to match and fit inside the enclosure which we'll cover in the next section. The corners, top and bottom parts, as shown in the Figure II.12 , were shaped that specific way aiming to get the connectors in their respective places regarding the enclosure. The Figure II.12 also illustrates the different drills to be made in the PCB for the mounting holes and for placing the LED light pipes later on.

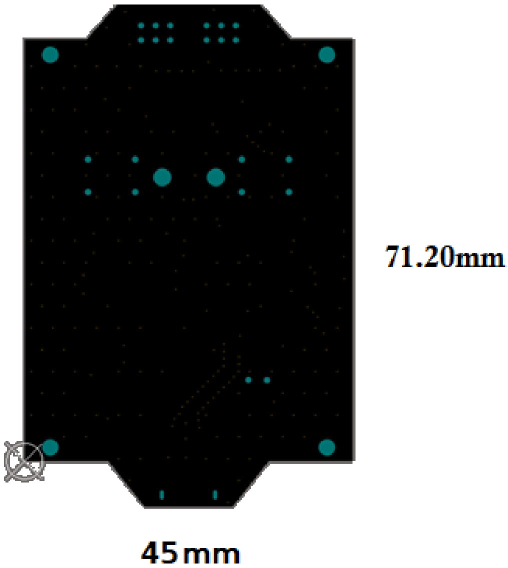


Figure II.12 – PCB Layout and drills

Another important point that revealed necessary through the PCB design, is the components placement. At this stage of the PCB layout, the components should be moved into their functional blocks so that associated components are close to each other and the circuit can be routed easily later. Therefore, we defined a placement methodology to work with. Like the schematic we placed the components as groups defined by their global function. The FT2232 IC is placed relatively at the center of the board since it's the one with the most connections and signals to almost every component in the design. The ESD suppressors are placed immediately after both connectors since they represent the higher risk of an electrostatic discharge. The Figure II.13 represents the final disposition for the component placement.

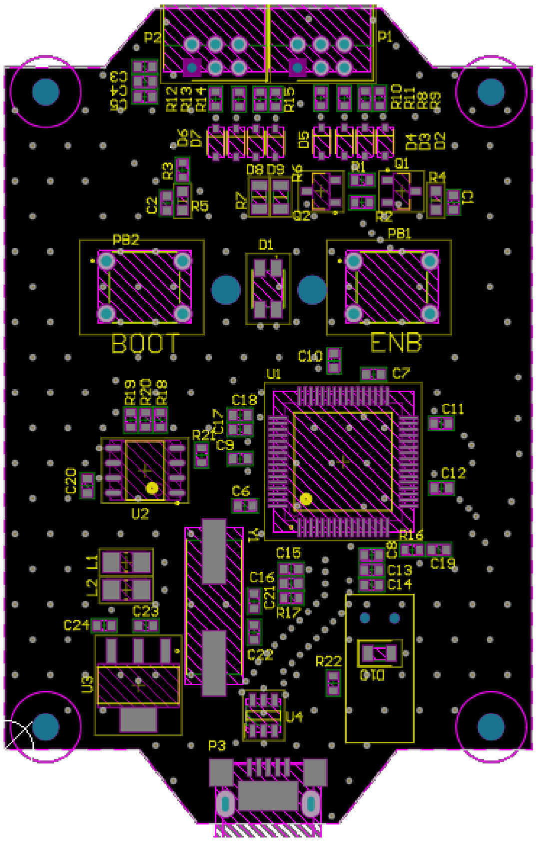


Figure II.13 – Components placement

Most of the bypass capacitors were placed as to be the closest possible to their respective pins without interfering with any other component at the correspondent area. Especially the two capacitors fitted with the crystal oscillator in order to not add to any unwanted parasitic capacity to the circuit. As for the switches and the LEDs were conveniently placed close to the center of the board facing upward for it's easier to use this way. Last but not least, the disposition of the ports illustrates how intuitive and clear would be to use the product for it has a clear input and output disposed at each end of the board.

After we had gone with the disposition as shown in the Figure II.14, setting up the footprint library and then perfecting the placement of the components on the board and to properly connect the placed components while obeying all design rules we need specify the size and topology rules for power, differential pairs like USB high speed data pairs. Before we start laying down copper traces on the board, first we need to fixate all requirement on minimum trace width 0.3mm, spacing (clearance 0.2mm) and more importantly the number of layers (two layers in this case) because that's what's going to determine the price at the manufacturing stage.

Meanwhile, if you take a closer look at the figure II.15 you'll notice that we deliberately avoided using 90-degree trace angles and the reason why we did so is to avoid having a sharp right angle turn on our board
Because the outside corner of that 90-degree angle has the likelihood of being etched narrower than your standard trace width. And at its worst, you might get a bunch of 90-degree traces back that aren't fully etched, resulting in short circuits. As a solution to this problem we used 45-degree angle traces which also produce an esthetically appealing PCB layout.

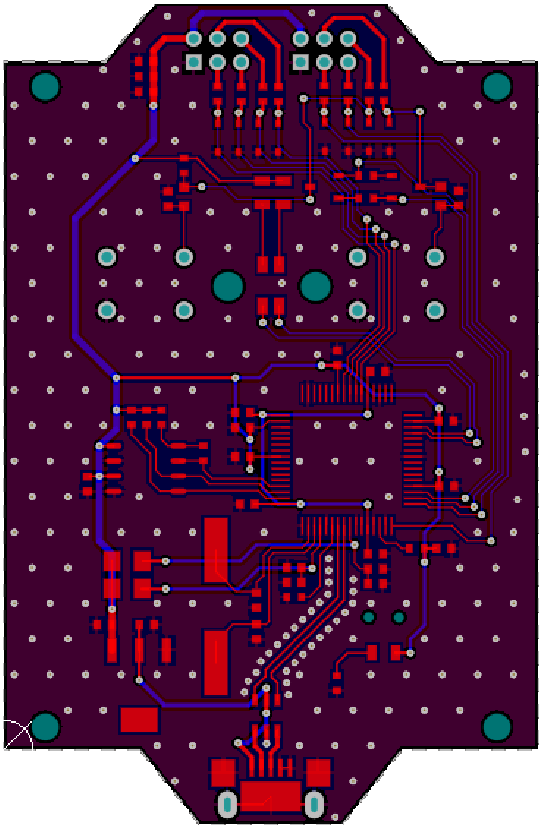


Figure II.15 Final routing for the PCB

Another important aspect we come across during this phase is having a common ground on our PCB which is imperative as it gives all of the traces the same point of reference. Thus, different trace resistance and voltage drops can be somewhat frustrating. Therefore, we dedicated a perfectly continuous ground plane on both layers of the PCB.

For power traces we increased the width of the traces to 0.7mm because power traces will have more current flowing through them and not making them wider will generate a lot of heat which can end up burning wires and damaging the whole board.

Last but not least, we used vias for providing connectivity between both layers but also to sink heat whenever present from under ICs such as transistors. One more trick that could make our design more reliable in the long run, is using vias as a Faraday cage. after finishing up with the whole design we managed to dispatch a considerable number of vias around the surface of the board and around the USB data wires to protect the whole board from EMI.

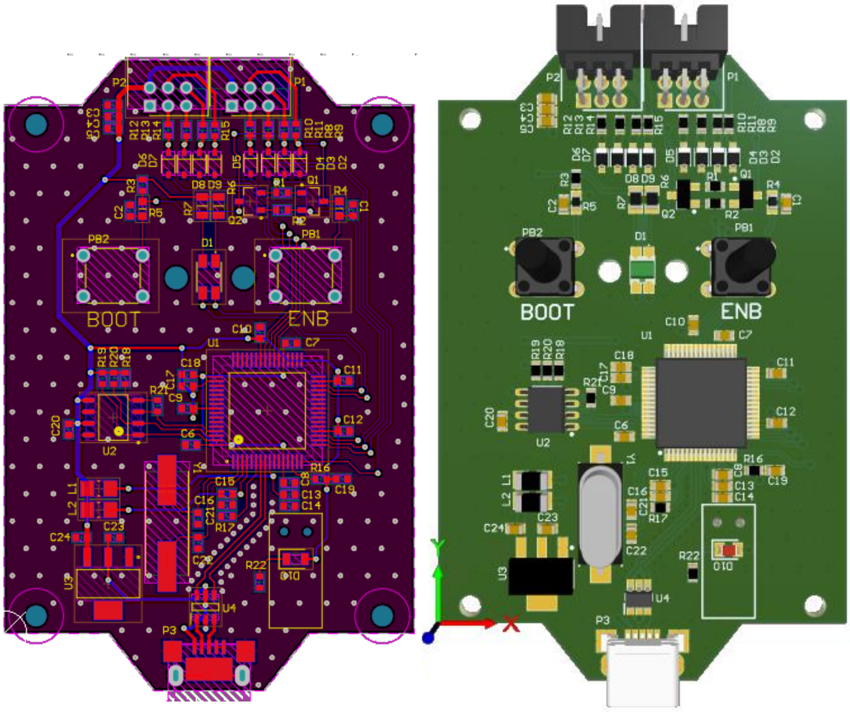


Figure II.16 final PCB and 3D model

2 Product Enclosure 3D Design

At an early stage when working on the enclosure for the Board designed in the previous section, we started working on Autodesk Fusion 360 due to its simplicity and also it could get the job done. However, later on we switched to the famous Solidworks because it’s the most commonly known tool for 3D design and because of its vast community. Not to mention also the ability to realize the needed rendering for product visualization before diving into manufacturing.

Since we’re not aiming for mas market production considering it’s only a prototype. We selected a ready 3D model from OKW vast catalogue for electronic devices enclosures. Then we altered the model and customized it to meet our specific needs.

Figure II.17 shows the enclosure 3D final design before rendering it with the rendering tool made available by Solidworks.

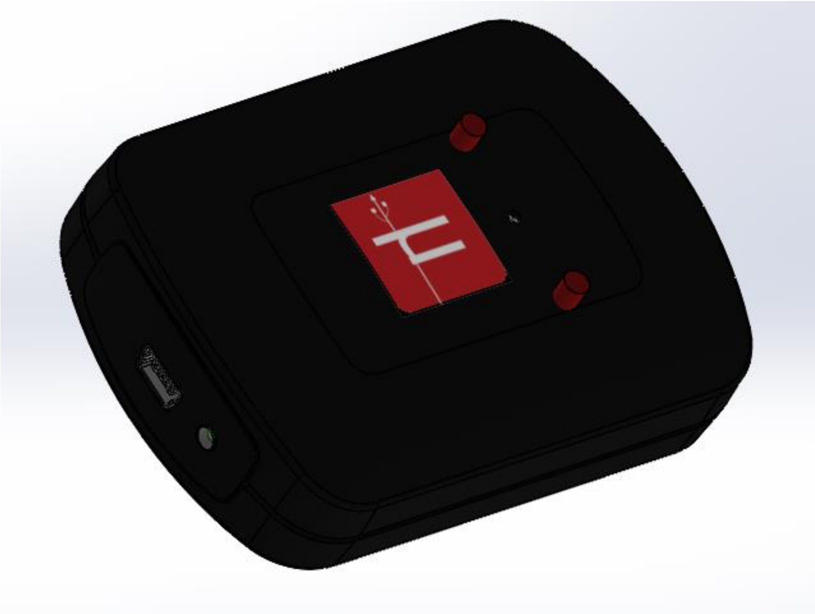


Figure II.17 3D design for the enclosure before rendering

The modification made were a couple of drills for the tactile buttons' heads and both light pipes. In Addition, we modified both ends of the model to fit our connectors. The final result can be observed clearly in the Figure II.18 below. After conducting the rendering procedure.

Figure II.18 enclosure design after rendering

The figure II shows the final product as a whole after exporting the 3D model of the PCB from Altium Designer and importing it in Solidworks to fit perfectly inside the enclosure.

Figure II.19 Final product design after adding PCB

Conclusion

During this second chapter we covered a deeper description of the different building blocks of the hardware part of this project. we then specified most of the components we chose and moved to the realization phase where we talked about the schematic and PCB design. Last but not least, we stepped by the 3D design of the enclosure as the last part of this chapter.

Chapter III

Software Solution Development

Contents

Conclusion and Outlook

Bibliography