

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'Hamed BOUGARA – Boumerdès



Institute of Electrical and Electronic Engineering
Department of Electronics

Project Report Presented in Partial Fulfilment of
the Requirements of the Degree of

‘Master’

In Computer Engineering

Title:

**Design and Implementation of a
Microcontroller Development Board with
Multi Add-on Compatibility**

Presented By:

- **ZIDANE Aymen**

Supervisor:

Dr. AZZOUGUI Yasmina

Co-Supervisor:

Dr. MEZZAH Ibrahim

Registration Number:...../2024

Abstract

The goal of this project is to develop an ESP32-based microcontroller that can directly communicate with many add-ons, including Feather, Pmod, and microBUS. This multi-interface microcontroller aims to eliminate the need to buy different microcontrollers for various peripherals or waste time interfacing, thereby lowering expenses and complexity. The study describes the design and implementation procedure, covering schematic design, PCB layout design, and the SMD soldering process. We tested the functionality of the designed board using the mikroBUS temp&Hum 12 click and also with a normal ESP32 and the results were matching and satisfactory.

Acknowledgments

First and foremost, I would like to praise Allah the Almighty, the Most Gracious, and the Most Merciful for His blessings given to me during my studies and in completing this project.

I would like to express my gratitude to **Mr. MEZZAH Ibrahim** for his invaluable support and guidance throughout this research. His expertise and insightful suggestions were a great help for the successful completion of this work. I am really appreciative of his time and effort in mentoring me during this journey.

I am also highly grateful to **Mrs. AZZOUGUI Yasmına** for her significant contributions, especially in the report writing and overall guidance. Her continuous attention and help during the hard times was a great motivation for finishing the work.

Also, I can never forget the huge support from my small family especially the financial one which led to the successful and full implementation of the project

Contents

Abstract	i
Acknowledgments	ii
General Introduction	x
1 Project Overview and Tools	1
1.1 Microcontrollers	1
1.1.1 ESP32	1
1.1.2 Feather	2
1.1.3 MikroElektronika Flip & Click SAM3X	2
1.2 ESP32-S3 Module	3
1.2.1 Features	4
1.2.2 Applications	5
1.3 The standard types of add-ons	5
1.3.1 Feather Wings	5
1.3.2 mikroBUS Click	6
1.3.3 Pmod	7
1.3.4 Qwiic	8
1.4 Arduino IDE	8
2 Printed Circuit Board (PCB) Design Theory	10
2.1 Prototyping	10
2.1.1 Breadboards	10
2.1.2 Prototype Soldering boards	11
2.1.3 Printed Circuit Boards	12
2.2 History of PCB Design	12
2.3 Evolution of PCB Boards	13
2.3.1 Manual Tracks	13
2.3.2 Printed Tracks	14
2.3.3 Two Layer PCBs	15
2.3.4 Multi-Layer PCBs	16
2.4 EDA Tools	17
2.4.1 Common PCB Design Software	18
2.4.2 Why Altium?	19

2.4.3	Altium Designer Features	19
2.4.3.1	Filters	19
2.4.3.2	Switching Layer views	19
2.4.3.3	Cross Probe selection	20
3	Hardware Design	21
3.1	Choice of the Microcontroller Unit (MCU)	21
3.2	Schematic Design	22
3.2.1	Accessing Components	22
3.2.2	Circuit Design	23
3.2.2.1	5V Power Supply	23
3.2.2.2	3.3V Power Supply	25
3.2.2.3	Additional Power Supply	26
3.2.2.4	Battery Charger	26
3.2.2.5	Push Buttons	28
3.2.2.6	LEDs	29
3.2.2.7	RGB LED	29
3.2.2.8	Add-Ons Headers	30
3.2.2.9	MCU pins	33
3.3	PCB Design	34
3.3.1	Golden Rules	34
3.3.2	Manufacturing Rules	35
3.3.3	Steps of the design	36
3.3.3.1	Component placement	36
3.3.3.2	Routing Signal Tracks	37
3.3.3.3	Power signals	37
3.3.3.4	Ground routing	37
3.3.3.5	Creating a Polygon	38
3.3.4	Final design	38
3.4	Ordering the PCB	39
3.4.1	PCB Specification	39
3.4.2	PCBA	40
3.4.3	The stencil	40
3.4.4	Components	41
3.5	Soldering the board	41
3.5.1	Tools Used	41
3.5.1.1	Microscope	41
3.5.1.2	Automatic Vacuum Pen	41
3.5.2	Applying Paste	42
3.5.3	Cleaning Paste	43
3.5.4	Soldering	43

4	Results and discussions	46
4.1	Pinout Diagram	46
4.2	Testing the board	47
4.2.1	GPIOs	47
4.2.2	User LED	47
4.2.3	RGB	48
4.2.4	mikroBUS add-on	48
4.3	Comparison	51
	General Conclusion	52
	Bibliography	53

List of Figures

1.1	ESP32 Microcontroller.	1
1.2	Feather Microcontrollers [1].	2
1.3	Flip & Click SAM3X [2].	2
1.4	ESP32 S3 Functional Block Diagram [3].	3
1.5	Assembled DC Motor with a stepper.	5
1.6	Adafruit Power Relay.	6
1.7	Adafruit FeatherWing OLED.	6
1.8	mikroBUS Standard [4].	7
1.9	Click example boards for : Display, Temperature and Storage.	7
1.10	Different Pmod Chips.	8
1.11	Qwiic Description [5].	8
1.12	Arduino Graphical User interface [6].	9
2.1	A prototype design using the breadboard.	11
2.2	A Manually soldered circuit.	11
2.3	A compact PCB design.	12
2.4	Board with manually soldered tracks.	14
2.5	Board with printed tracks.	14
2.6	THT Component.	15
2.7	SMD Component.	15
2.8	SMD components compared to a finger.	16
2.9	Physical Layers of Multi-Layer PCB. [7]	16
2.10	Examples of the three types of vias in a circuit board: (1) blind via, (2) buried via, and (3) through-hole via [8].	17
2.11	Filtering a track.	19
2.12	Different layer modes.	20
2.13	Cross Probe selection.	20
3.1	Different ESP32 Modules.	21
3.2	Real-World component with its footprint [9].	22
3.3	Manufacturer Part Search.	23
3.4	VBUS Power Supply.	24
3.5	Valid Upstream Facing Port (UFP) Rd Pull-down Resistor Values [19].	24
3.6	3.3V Power supply circuit.	25
3.7	Qwiic Power Supply.	26
3.8	Battery charger circuit.	27

3.9	Charge Current vs Programming resistor.	28
3.10	BOOT and RESET signals.	28
3.11	Power Led & User LED	29
3.12	An RGB LED circuit.	29
3.13	Using the multimeter with a testpoint.	30
3.14	The Add-ons headers.	31
3.15	Steps of creating headers in the schematic library.	32
3.16	Add-On Standard measurements.	32
3.17	The Headers' footprint.	33
3.18	MCU wired pins	33
3.19	JLCPCB Manufacturing Capabilities [10].	35
3.20	Altium Designer Rule wizard.	36
3.21	Component auto placement from Altium.	36
3.22	Qwiic-3.3V pin.	37
3.23	3D View of the power plane	37
3.24	Routing constraint solved.	38
3.25	3D view of the board.	38
3.26	Example of two gerber files.	39
3.27	PCB Specifications in JLCPCB.	39
3.28	A stencil example.	40
3.29	Solder paste layer.	40
3.30	A microscope with an integrated camera.	41
3.31	VAC-12000.	42
3.32	Usage of the stencil.	42
3.33	Process of cleaning the paste.	43
3.34	A clean PCB view for the board.	43
3.35	Altium & Camera views of the board.	44
3.36	The full microcontroller board	44
4.1	The microcontroller's pinout diagram.	46
4.2	Code for testing GPIOs.	47
4.3	LED Blinking.	48
4.4	RGB colors.	48
4.5	HDC2080 Arduino code.	49
4.6	Measurment results with the normal ESP32.	49
4.7	Interfacing the sensor with normal ESP32.	50
4.8	Measurment results with our MCU.	50
4.9	Interfacing the sensor with the implemented MCU.	51

List of Abbreviations

ADC Analog to Digital Converter

API Application Programming Interface

BGA Ball Grid Array

BOM Bill of Materials

ECME Electronic Circuit Making Equipment

EDA Electronics Design Automation

EMI Electromagnetic Interference

ERC Error Rule Check

GPIO General-Purpose Input/Output

HDI High Density Interconnect

HSD High Speed Design

I2C Inter-Integrated Circuit

IC Integrated Circuit

IDE Integrated Development Environment

IO Input Output

IoT Internet of Things

LDO Low Dropout

LED Light Emitting Diode

MCU Microcontroller Unit

NRND Not Recommended for New Designs

OTP One-Time Password

PCB Printed Circuit Board

PCBA Printed Circuit Board Assembly

PWM Pulse Width Modulation

RGB Red Green Blue

ROM Read Only Memory

SDK Software Development Kit

SIMD Single Instruction Multiple Data

SMD Surface Mount Device

SMT Surface Mount Technology

SoC System On Chip

SPI Serial Peripheral Interface

SRAM Static Random Access Memory

THT Through Hole Technology

UFP Upstream Facing Port

ULP Ultra-Low Power

General Introduction

Microcontrollers are compact integrated circuits designed to perform different general-purpose operations in embedded systems. They are essential in different fields, starting from simple applications to industrial automation. There are various types of them and each one has its pros and cons. The ESP32, for instance, is known for its great community and is widely used in IoT applications due to the built-in Wi-Fi and Bluetooth capabilities. The Feather series by Adafruit provides a flexible platform with integrated battery charging and compatibility with various wireless modules and wings. MikroElektronika's boards, on the other hand, combines the ease of Arduino with the power of ARM Cortex-M3 architecture, supporting a wide range of applications through its mikroBUS sockets.

The common problem with this rich set of options is that various design specifications are followed by the manufacturer companies like Feather, mikroBUS, and PMOD due to commercial reasons most of the time, where the developer buys the board and then the add-ons from the same company to ensure compatibility and the ease of interfacing. Because of this drawback, this project aims to address this challenge by designing a microcontroller board that supports multiple add-ons, including Feather, mikroBUS, and PMOD, on a single platform. This unified solution will improve the development process, reduce costs, and enhance flexibility for developers.

The report is organized as follows: *Chapter 1* provides an overview of the project, the existing market solutions and the standard add-ons that we will work on. *Chapter 2* discusses a theoretical background about the field of Printed circuit board (PCB) design diving through its history and developments and the main software used for the design. *Chapter 3* starts by including the choice of the ESP32 module then the details of the hardware design on both the schematic and PCB levels. Also, the ordering and soldering processes have been briefly shown. *Chapter 4* includes a results and discussion part where the board has been tested under different conditions including connecting a mikroBUS sensor to ensure having a correct implementation.

Chapter 1

Project Overview and Tools

This chapter provides an overview of the project by detailing existing market solutions, and outlining the primary hardware tools and software used for testing and debugging the final design.

1.1 Microcontrollers

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. The designed one in this project should be a balance between all the pros and capabilities of the boards discussed below.

1.1.1 ESP32

The ESP32 microcontroller, created by Espressif Systems, is a flexible platform known for its strong connection and speed. With Wi-Fi and Bluetooth capabilities, it is great for Internet of Things (IoT) apps like smart home devices, wearables, and industrial automation. The ESP32 supports programming environments like Arduino Integrated Development Environment (IDE), ESP-IDF, and MicroPython, catering to both beginners and experienced coders. Figure 1.1 shows that it is for general purpose interfacing and does not directly support any common add-ons.

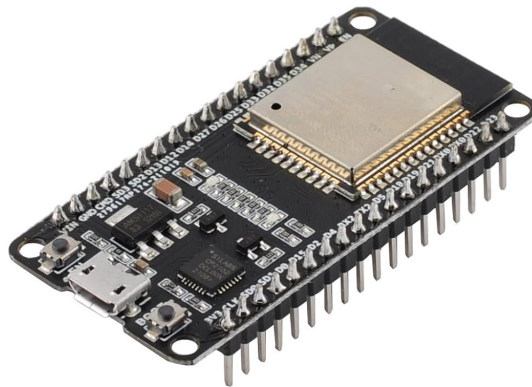


Figure 1.1: ESP32 Microcontroller.

1.1.2 Feather

The Adafruit Feather series is a flexible group of development boards that work with different microcontrollers, such as the ESP32, ATSAMR21, and nRF52. They are thin and light, and the batteries are charged inside. Generally, these boards also have Wi-Fi, Bluetooth, and LoRa modules for smart home and wireless communication. Feather boards work with the Arduino IDE and have a standard pinout that lets shields and other accessories be used with them. A group of them is shown in Figure 1.2.

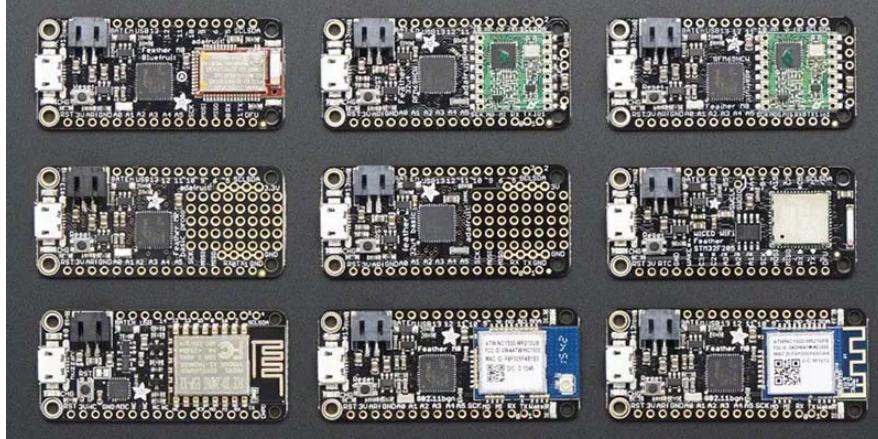


Figure 1.2: Feather Microcontrollers [1].

1.1.3 MikroElektronika Flip & Click SAM3X

The Flip & Click SAM3X shown in Figure 1.3 is a unique development board that combines the ease of use of Arduino with the power of ARM Cortex-M3 architecture. It is based on the Atmel SAM3X8E microcontroller and features four mikroBUS sockets for Click Board expansion. This dual-sided board can function as both an Arduino shield and a standalone development platform, making it really suitable for a wide range of applications, from simple hobby projects to complex industrial solutions.

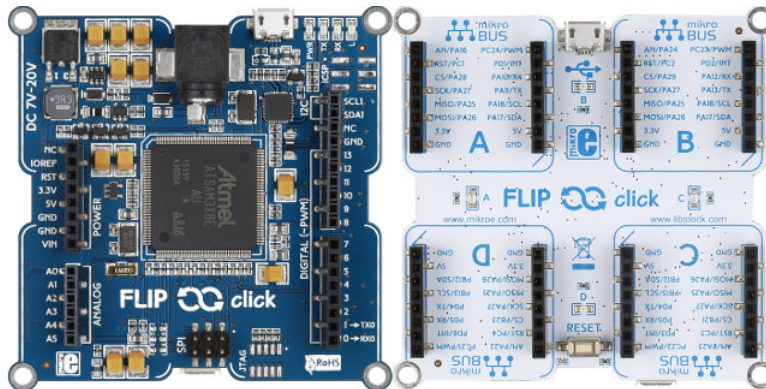
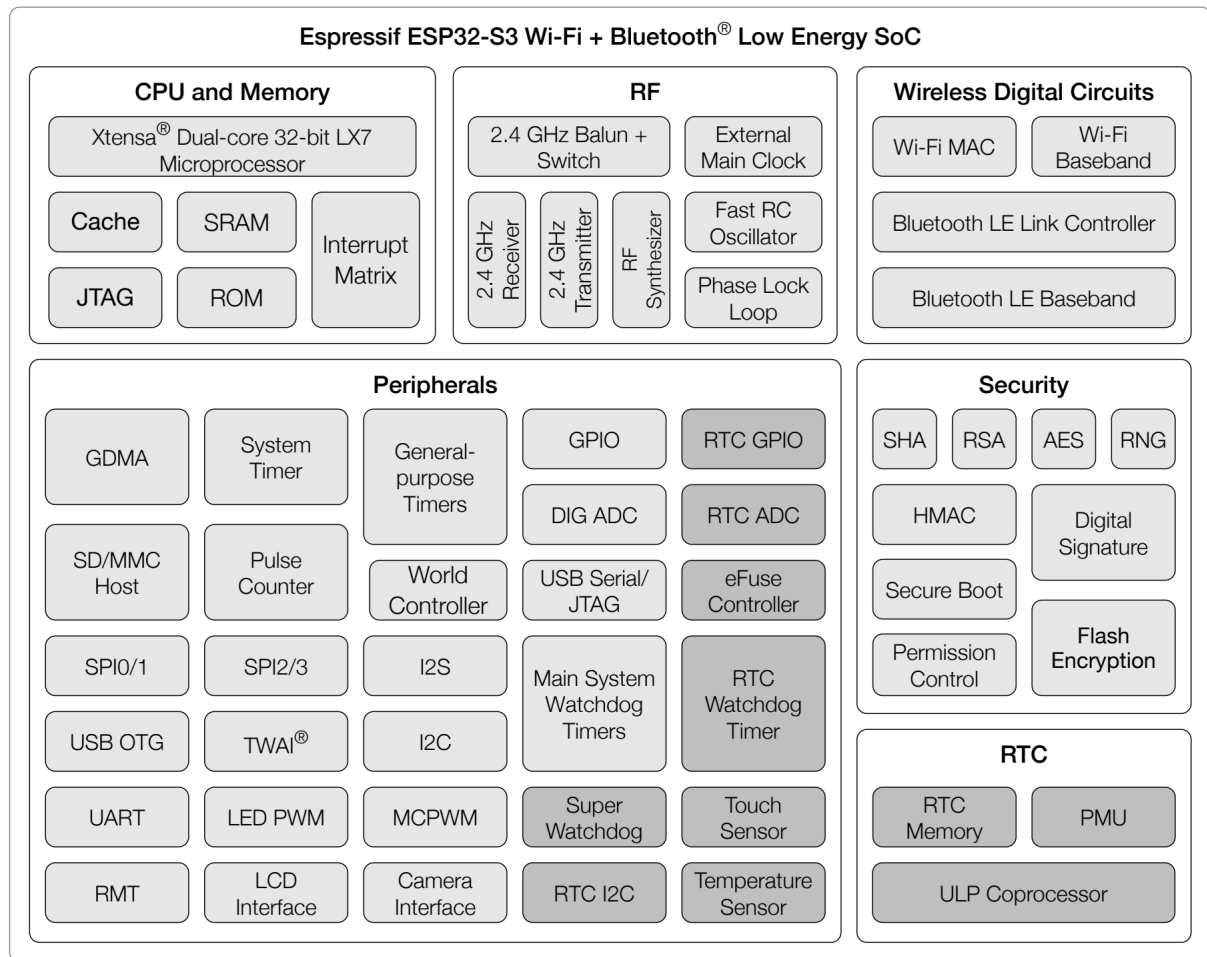


Figure 1.3: Flip & Click SAM3X [2].

1.2 ESP32-S3 Module

”ESP32-S3 is a low-power MCU-based System On Chip (SoC) with integrated 2.4 GHz Wi-Fi and Bluetooth® Low Energy (Bluetooth LE). It consists of high-performance dual-core microprocessor (Xtensa® 32-bit LX7), a low power coprocessor, a Wi-Fi baseband, a Bluetooth LE baseband, RF module, and numerous peripherals” [3]. The functional block diagram of this specific SoC is shown in Figure 1.4.



Power consumption

- Normal
- Low power consumption components capable of working in Deep-sleep mode

Figure 1.4: ESP32 S3 Functional Block Diagram [3].

1.2.1 Features

Here is the list of the most important features that motivated the use of this specific chip for IoT applications:

Wi-Fi

- IEEE 802.11b/g/n-compliant
- Wi-Fi Multimedia (WMM)
- 4 virtual Wi-Fi interfaces
- Antenna diversity
- 1T1R mode with data rate up to 150 Mbps

CPU and Memory

- 128-bit data bus and SIMD commands
- 384 KB ROM
- 512 KB SRAMs
- Flash controller with cache is supported

Analog Interfaces

- 2 12-bit SAR ADCs, up to 20 channels
- 14 touch sensing IOs

Low Power Management

- Power Management Unit with five power modes
- Ultra-Low-Power (ULP) coprocessors.

Bluetooth

- Bluetooth LE: Bluetooth 5, Bluetooth mesh
- Speed: 125 Kbps, 500 Kbps, 1 Mbps, 2 Mbps
- Internal co-existence mechanism between Wi-Fi and Bluetooth to share the same antenna

Advanced Peripheral Interfaces

- 45 programmable GPIOs
- 1 USB Serial/JTAG controller
- 4 SPI
- 2 I2C
- 2 I2S
- 1 RMT (TX/RX)

Security

- Secure boot
- Flash encryption
- 4-Kbit OTP, up to 1792 bits for users
- Cryptographic hardware acceleration:
 - AES-128/256 (FIPS PUB 197)
 - Random Number Generator (RNG)
 - Digital signature

1.2.2 Applications

ESP32 is very powerful and widely used in different applications like:

- Smart Home
- Industrial Automation
- Cameras for Video Streaming
- Health Care
- Consumer Electronics and art Agriculture
- Service robot
- Speech Recognition
- Image Recognition
- Low-power IoT Sensor Hubs
- USB Devices
- Networking Cards
- Audio Devices

1.3 The standard types of add-ons

At this point, the standard types of sensors or even some other chips are going to be presented. The Micro-controller designed in this project can universally support them without any further breadboard usage or wiring.

1.3.1 Feather Wings

Adafruit offers a variety of modules that share a common form and pinout, allowing for easy interchangeability. Feather aims to provide various add-ons and wings available which are generally sensors suitable for IoT and wearable tech applications [1]. The most used ones are listed below:

Motors: Figure 1.5 shows an example of a fully assembled (with headers) DC Motor + Stepper FeatherWing which will provide the usage of 2 bipolar stepper motors or 4 brushed DC motors or 1 stepper and 2 DC motors.

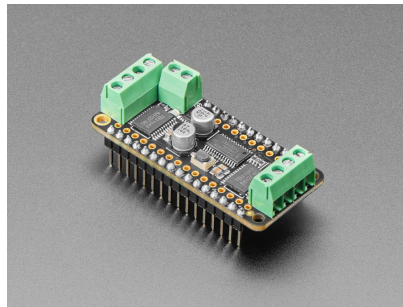


Figure 1.5: Assembled DC Motor with a stepper.

Relays: The relay shown in Figure 1.6 is a very famous electronic part to control lamps, fans, solenoids, and other small appliances.

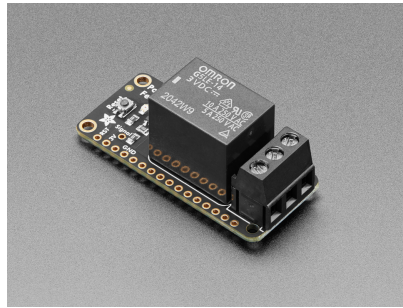


Figure 1.6: Adafruit Power Relay.

Display Screens: As an example in Figure 1.7, they have the 128x64 OLED which is small and very readable due to the high contrast of the OLED display.

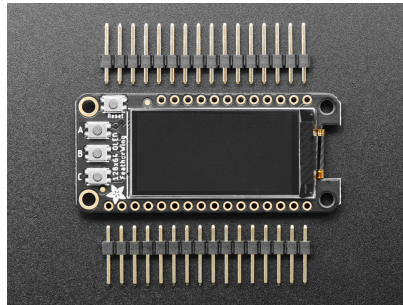


Figure 1.7: Adafruit FeatherWing OLED.

1.3.2 mikroBUS Click

The mikroBUS™ socket contains two 1×8 female headers with a proprietary pin configuration. The pinout (always laid out in the same order) consists of three groups of communications pins (SPI, UART and I2C), six additional pins (PWM, Interrupt, Analog input, Reset and Chip select), and two power groups (+3.3V and 5V). This enables access to a wide range of possibilities with sensors, wireless transceivers, audio amplifiers, LED displays, and more. Figure 1.8 shows the exact arrangement of pins [4] .

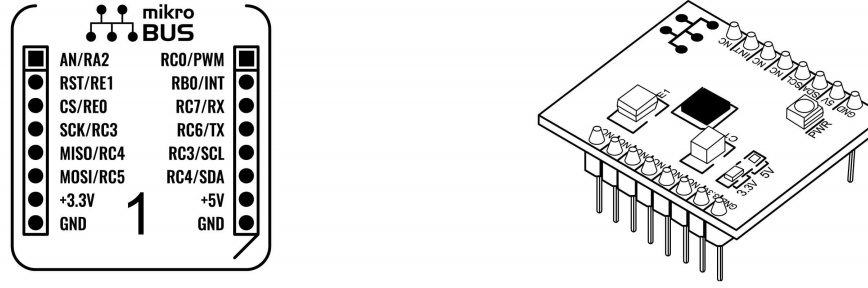
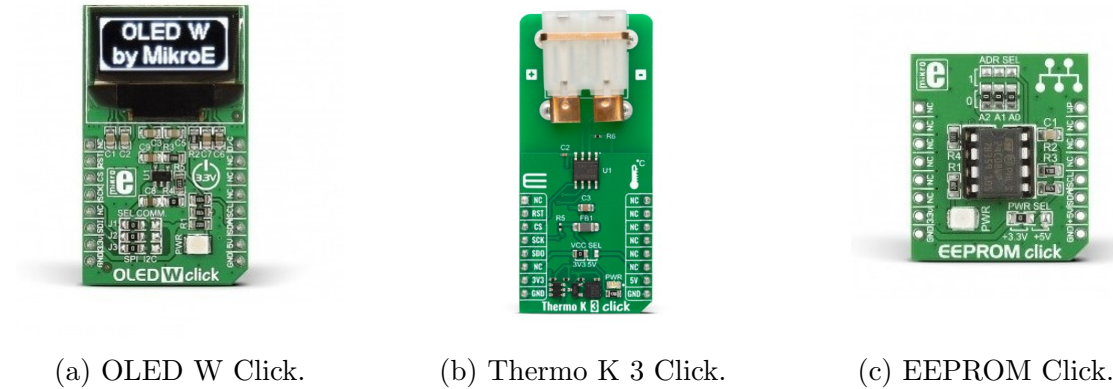


Figure 1.8: mikroBUS Standard [4].

Example Boards: They have a very wide range of sensors and general purpose chips which evaded the market because of the small size and accurate functioning. Some figures of them are shown in Figure 1.9.



(a) OLED W Click.

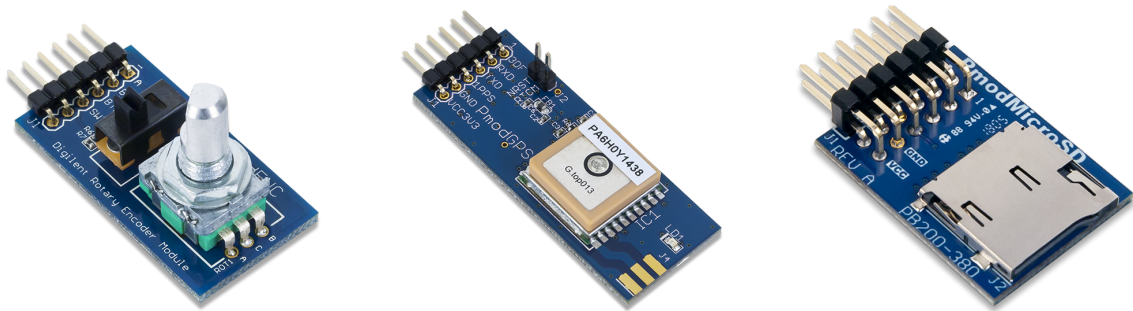
(b) Thermo K 3 Click.

(c) EEPROM Click.

Figure 1.9: Click example boards for : Display, Temperature and Storage.

1.3.3 Pmod

Pmods shown in Figure 1.10 are made by Digilent company which is very famous of manufacturing measurement devices, sensors and even FPGA development boards. They are low-cost modules that enhance the capabilities of embedded systems. They communicate via 6, 8, or 12-pin connectors using standard serial protocols and enable efficient designs by directing analog signals and power supplies where needed [11] .



(a) Pmod Enc.

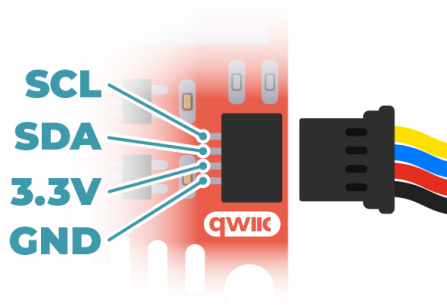
(b) Pmod GPS.

(c) Pmod MicroSD.

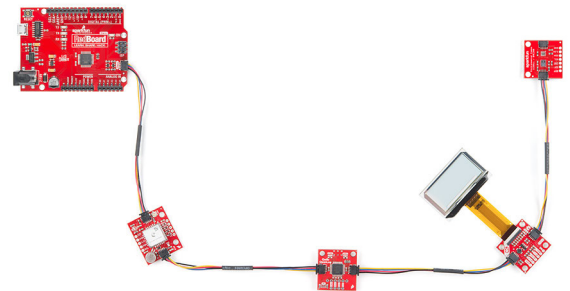
Figure 1.10: Different Pmod Chips.

1.3.4 Qwiic

”SparkFun’s Qwiic Connect System is an I2C Prototyping that uses 4-pin JST connectors to quickly interface development boards with sensors, LCDs, relays and more.” [5]. A lot of different chips can be easily connected in series I2C mode as clarified in Figure 1.11(b).



(a) Qwiic Pinout.



(b) Daisy chain connected chips.

Figure 1.11: Qwiic Description [5].

1.4 Arduino IDE

The Arduino Integrated Development Environment (IDE) is an excellent tool for testing software quickly and easily due to its user-friendly interface and extensive library support as illustrated in Figure 1.12. It simplifies the development process, allowing even beginners to get started with coding and prototyping efficiently. Arduino supports a wide range of microcontrollers, including the ESP32, as well as ATmega328, ATtiny85, SAMD21, and the ARM Cortex series. This flexibility makes it a great tool to test our designed board in the shortest possible time.



Figure 1.12: Arduino Graphical User interface [6].

Conclusion

After taking a detailed look into the hardware tools, the goal is to design a board to support those different add-ons. Thus, The next step is to prepare the prerequisites to achieve all of that by diving into the field of PCB Design .

Chapter 2

PCB Design Theory

This chapter introduces most of the theoretical background knowledge needed about Printed Circuit Boards (PCBs) to clearly understand how to design the hardware of the system passing through some history of the field, some techniques used and the software needed.

2.1 Prototyping

Before making a final proper design, it is generally necessary to do a prototype just to test the functionality of the system and to ease replacing components and fixing errors.

2.1.1 Breadboards

A breadboard is a basic tool used for the purpose of prototyping or constructing circuits. The process enables the placement of components and connections on the board, facilitating the creation of circuits without the need for soldering. The apertures within the breadboard facilitate the establishment of connections by physically securing components or wires at their designated locations and electrically linking them within the board. The simplicity and rapidity of the system make it very suitable for acquiring knowledge and rapidly creating basic circuits [12].

As can be seen in Figure 2.1, breadboards make it very simple to replace components, wires and use test equipment to determine if the circuit is well functioning.

However, they cannot be used as a final product for many reasons: the components are not stick or well attached to the board and with some movements, the wires will pop out and hence it needs a lot of maintenance periodically.

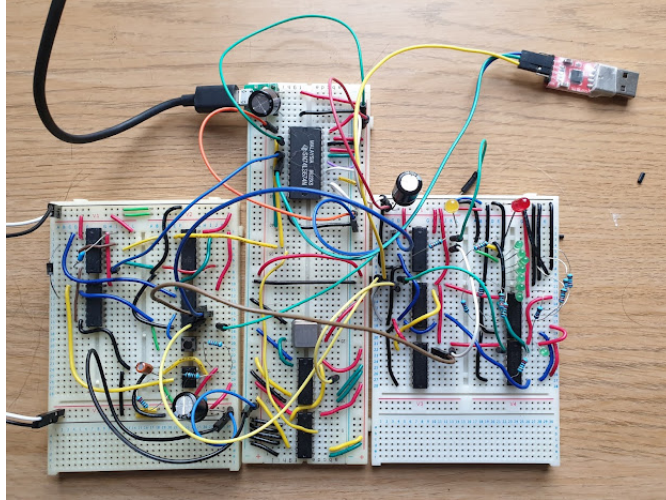


Figure 2.1: A prototype design using the breadboard.

2.1.2 Prototype Soldering boards

This prototyping method is a lot better in terms of checking the stability of systems for some long periods before making adjustments and designing a final proper circuit.

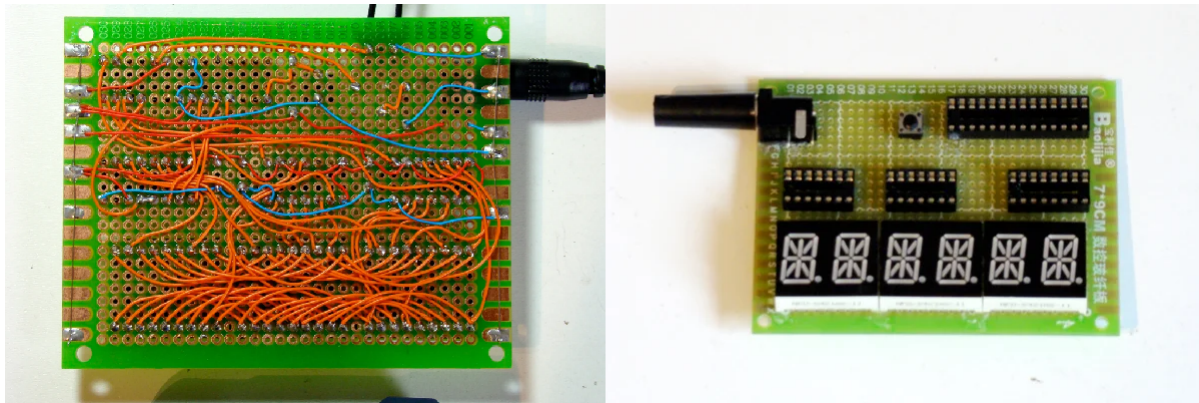


Figure 2.2: A Manually soldered circuit.

Figure 2.2 shows that the board is a simple insulator surface with a lot of evenly distributed holes throughout that plane. It just needed to plug all the devices inside those holes then flip the board and start connecting wires and soldering them in their proper place. The advantage of these boards is that it is very easy for beginners to make optimized size systems in a short time. But when looking for a high density and space optimum system then it causes a lot of problems because the wiring is very messy and random.

2.1.3 Printed Circuit Boards

PCB stands for Printed Circuit Board and it is a non-conductive material with conductive lines printed or etched. They are a very precise and effective way to make very robust systems in the smallest possible size. Figure 2.3 shows how PCBs can encapsulate large amount of different components in a one organized board.

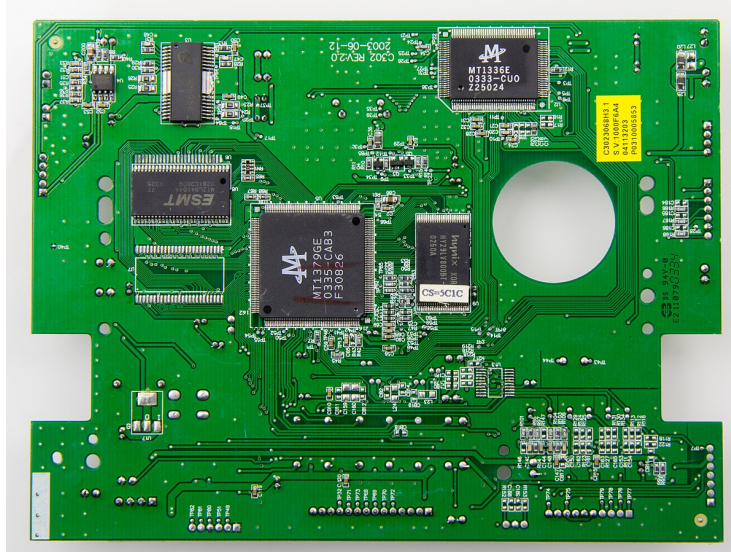


Figure 2.3: A compact PCB design.

Compared to the previous prototyping methods. They offer a very simple way for designing since all of the tracks and details are printed by the manufacturer company and not placed manually which decreases the chance of errors that can take long debugging time. However, unlike the previous methods, any mistake done in the design cannot be recovered from or fixed in the general case because it needs advanced equipment. Therefore, PCBs are a more suitable design approach to apply after verifying everything is correct and functional.

2.2 History of PCB Design

Like other great inventions, PCBs also have their history from past to present. The current technology that can be seen today was built on a foundation of advancements throughout history. [13]

Before printed circuit boards were developed, electrical circuits were wired directly on a chassis. These circuits were large, heavy, fragile, and expensive to produce. The development of modern printed circuit boards began in the early 20th century. In 1903, *Albert Hanson* described laminated conductors on an insulating board. *Thomas Edison* experimented with plating conductors on linen paper in 1904. *Arthur Berry* patented a print-and-etch method in the UK in 1913. *Charles Ducas* patented electroplating circuit patterns in 1925. In 1936, *John Sargrove* invented the Electronic

Circuit Making Equipment (ECME), which could produce three radio boards per minute on a Bakelite plastic board [14].

The printed circuit was invented by Austrian engineer *Paul Eisler* in the UK around 1936. It was later used in German naval mines in 1941. The United States began using the technology on a large scale in 1943 to make proximity fuses for World War II. The Centralab Division of Globe Union developed a proposal for electronic circuits that could withstand being fired from a gun and be produced in quantity. The technique involved screen-printing metallic conductors and resistors onto a ceramic plate, with capacitors and vacuum tubes soldered in place. In 1984, *Harry W. Rubinstein* was awarded the *Cledo Brunetti* Award for his key contributions to printed components and conductors on an insulating substrate. This invention also played a role in the development of integrated circuit technology [14].

In 1948, the USA commercialized printed circuits. The Auto-Semby process, developed by the United States Army in the mid-1950s, made printed circuits more common in consumer electronics. Similar work was done in the UK by *Geoffrey Dummer*. Motorola became an early leader in adopting plated circuits for home radios in 1952. Hallicrafters also introduced a printed circuit product that year. These boards were used to reduce the size, weight, and cost of electronic components. Initially, through-hole construction was used, but later, surface mount parts became more popular. In the 1990s, multi-layer and High Density Interconnect (HDI) PCBs emerged [14].

2.3 Evolution of PCB Boards

PCB Boards have gone through many stages and developments, the major transfers and advancements that happened are going to be listed briefly.

2.3.1 Manual Tracks

This board type is very similar to the prototype ones except that holes are not spread over the whole board but only where needed to solder the components as shown in Figure 2.4.

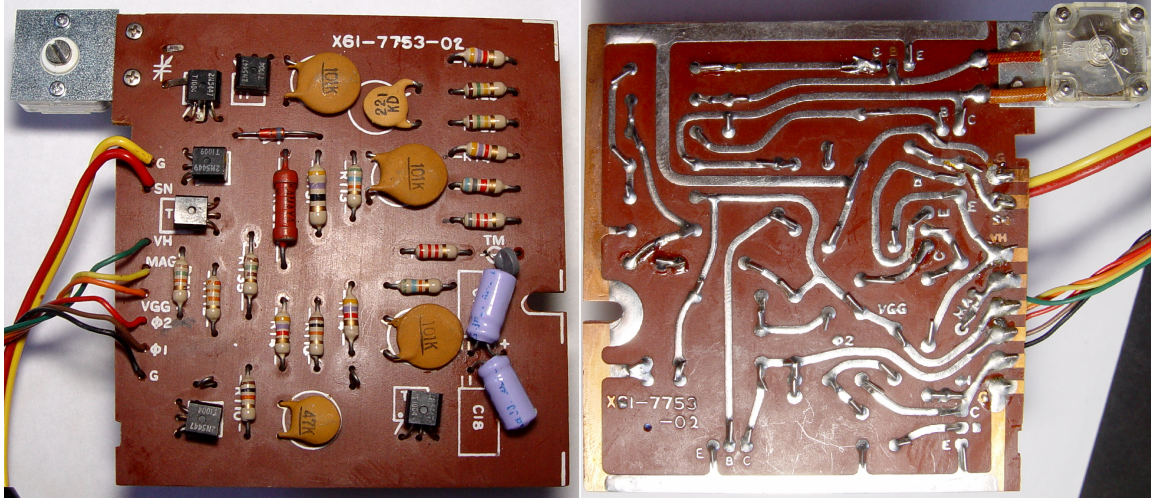


Figure 2.4: Board with manually soldered tracks.

This type has a lot of disadvantages, for example: the designer should make a very wise decision about the exact components' placement since there is no way that two level tracks can be made. Also, the designer is generally restricted and obliged to organize the design in some random way where he may fall in making analog circuits and digital ones very adjacent and create an unstable system.

2.3.2 Printed Tracks

Figure 2.5 shows that this version is slightly different from the previous one where tracks are printed with some special machines as pure copper. The design gets easier but it still has a lot of limitations as a being able to construct one layer circuits only.

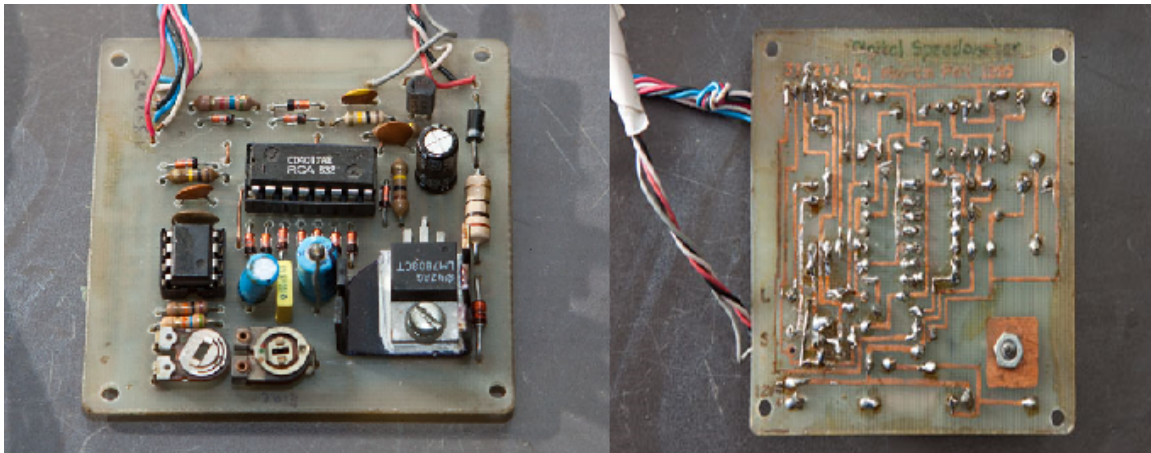


Figure 2.5: Board with printed tracks.

2.3.3 Two Layer PCBs

The technology of two layer PCBs was not very commonly used in the age of Through hole technology (THT) components because of the space constraints and the huge number of holes created by component placements. Surface mount devices (SMDs) were a great invention and an efficient replacement for them. These two types are more explained:

THT Components: They are the usual electric components that are used daily. As can be seen in Figure 2.6, the drawback of this type is that its leads generally should penetrate through the whole board which defines two sides of the PCB: component side and solder side, and it is not appropriate and hard to use the latter one as a new layer for signals.

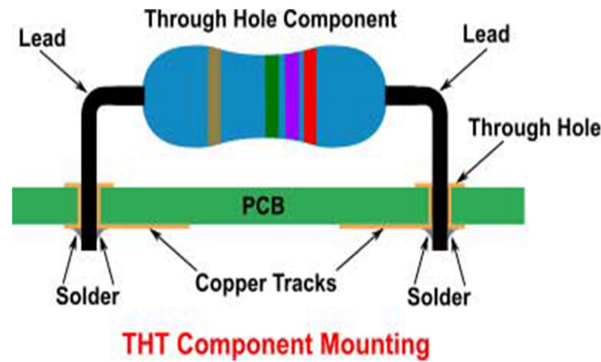


Figure 2.6: THT Component.

SMD Components: SMDs are the new invention which made two layer PCBs vastly used and easily manufactured. They do not penetrate through the board but only take a space of one layer as shown in Figure 2.7.

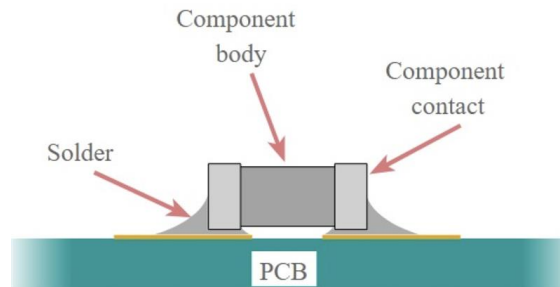


Figure 2.7: SMD Component.

They offer space efficiency through compactness on the PCB surface, allowing flexible circuit designs. Figure 2.8 shows how this technology developed to make packages that are even hard to see.

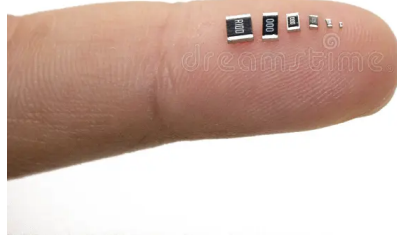


Figure 2.8: SMD components compared to a finger.

They also provide cost savings and efficiency with automated assembly processes. SMD components are mounted using Surface Mount Technology (SMT), increasing production efficiency compared to THT components that require manual soldering. Additionally, this technology allows for dual-sided placement on the PCB, enabling more advanced designs and improved heat dissipation with easier two-layer PCB layouts.

2.3.4 Multi-Layer PCBs

Multi-layer PCBs are circuit boards with more than two electrical layers bonded by resin layers. They are complex and costly to manufacture due to lower production yields and difficulty in re-working. The need for multi-layer boards arises from the high concentration of interconnecting lines in integrated circuits, which can cause design problems. To meet these requirements, single-sided and double-sided boards are insufficient, hence the concept of multi-layer boards. The primary purpose is to provide more routing options for complex and noise-sensitive electronic circuits [15].

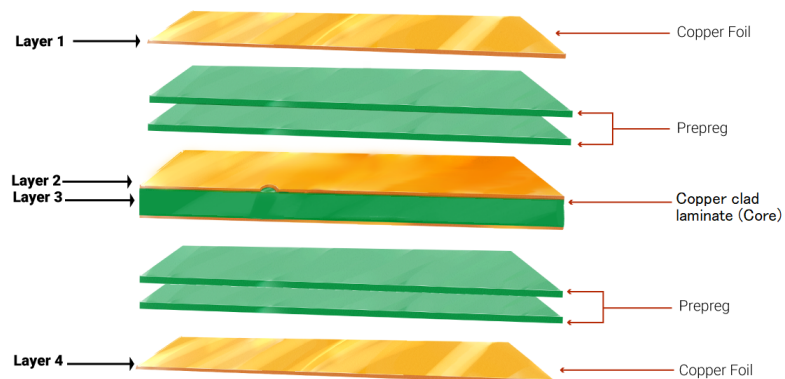


Figure 2.9: Physical Layers of Multi-Layer PCB. [7]

Figure 2.9 shows the main and common physical layers to understand how the

layout is organized. For *Copper foil*, it is an essential component in PCBs, providing electrical pathways for current flow. It is etched to create circuit traces and pads. While *Prepreg* is a composite material, consists of reinforcing fibers impregnated with resin. It acts as an insulating layer between copper layers in the PCB. Also, *The laminate* or core, is made up of one or more layers of prepreg. It is pressed, hardened, and cured with heat. The core is plated with copper foil on both sides, forming the central structure of the PCB [16]. The different copper layers of the boards are not totally independent but interconnected using vias.

Vias: Vias are required to make electrical connections between layers when there are multiple layers of signals, power, and ground. Based on their manufacturing process and the objects they connect, vias are classified into three types that are frequently used in boards. Figure 2.10 shows examples of these three types of vias [8].

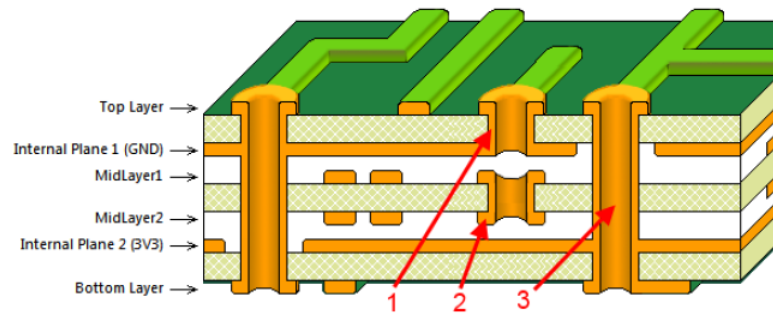


Figure 2.10: Examples of the three types of vias in a circuit board: (1) blind via, (2) buried via, and (3) through-hole via [8].

Blind via connect either the top or bottom layer of a PCB to an inner layer, and they are only visible from one outer layer. They optimize component placement in a high-density way since the normal via will take place from the other side of the board.

Buried via connect inner layers of the PCB without extending to the outer layers and are completely hidden within the board. They allow for complex multi-layer designs without affecting external dimensions or component placement, enhancing signal integrity and reducing a lot of Electromagnetic Interference (EMI) problems.

Through-hole via connect all the way through the PCB from one external layer to another. They are common in standard PCB designs but can affect overall board density in surface mount technology. They may not be suitable for high-density layouts.

2.4 EDA Tools

PCB Electronics Design Automation (EDA) tools are specialized software applications used by engineers and designers to create, simulate, and test the layout and

routing of PCBs. These tools help in designing the electrical connections and physical layout of the components on a PCB. It is very critical to master using them since they are the sole discussion language between the designer and the manufacture industry.

2.4.1 Common PCB Design Software

There are tens of EDA software tools available and the most famous one are going to be briefly described below.

Proteus:

It is a very famous software specially made for electronics beginners and hobbyists because they offer simple and friendly schematic capture design with various kinds of simulations to determine if the system is functioning properly before proceeding with the PCB design.

Eagle:

Eagle from Autodesk is powerful PCB design software that allows to capture schematics and make changes to PCB layouts. It has an easy-to-use interface for designing complicated circuit boards, with features like advanced routing and checking for design rules. Eagle supports real-time synchronization, large libraries of components, and seamless integration with other Autodesk products to make working together and getting things done easier [17].

KiCad:

KiCad is an open-source PCB design software which makes it one the most famous and favorable ones . It has many tools for capturing schematics and laying out PCBs. It has a 3D viewer, an interactive router, a hierarchical schematic editor, a robust PCB editor, and an integrated SPICE simulator. KiCad has a lot of libraries that help both beginners and experts and make sure that the design is correct [18].

Altium Designer:

Altium Designer is one of the best PCB design programs and the most used one by companies because it has a unified design environment that brings together all the different parts of the PCB design process. This includes drawing schematics, laying out PCBs, designing FPGAs, making embedded software, and simulating. Its advanced routing tools, like differential pair routing and length tuning, make design more accurate. The software's 3D visualization and clearance checking make sure that parts are placed and fit correctly. Altium Designer also supports extensive libraries and component management, real-time collaboration, and data management features, making it ideal for complex, high-quality PCB projects.

2.4.2 Why Altium?

Altium Designer is the richest EDA software with its advanced tools and capabilities but yet still one of the most complex ones to learn and use. It was chosen for a simple principle "Learn the difficult things to master the simple ones". Also, using this specific software opens doors to learning the design of complex systems very efficiently like Ball Grid Array (BGA) technology, High speed designs ...etc.

2.4.3 Altium Designer Features

The main features of Altium that simplifies the designing process are briefly discussed in the following points.

2.4.3.1 Filters

As its name applies, filters are a really powerful way to choose specifically which parts are intended to use or select inside the schematic or PCB design as it may be very time consuming to do so when the design is compact and full of details.

In Figure 2.11, It can be seen that some track is above other pads of the MCU (although that they are on different layers) but when we press there , the software will be confused which one to choose. The solution is just to apply a filter to select tracks only.

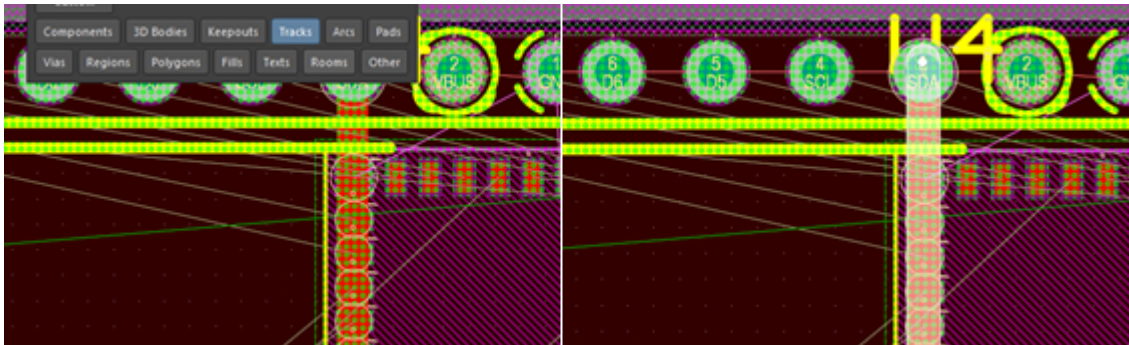


Figure 2.11: Filtering a track.

2.4.3.2 Switching Layer views

Two different layers are most of the time independent from each other and that is why generally messy to see them at the same time. CTRL + S can be used to switch from the *all layers view* to a *one layer view* where the focus is on it solely , as shown in Figure 2.12.

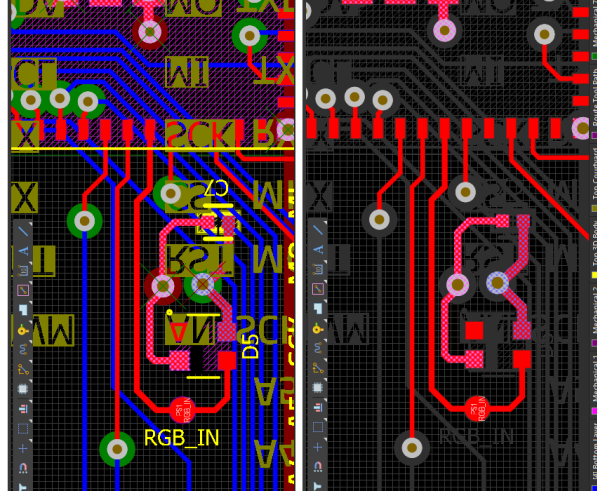


Figure 2.12: Different layer modes.

2.4.3.3 Cross Probe selection

As can be seen in Figure 2.13, Altium Designer's cross-probe selection function helps users choose and highlight related elements in various views, including schematic and PCB layout, so improving design efficiency. This feature guarantees that, when an element is chosen in the schematic, the PCB layout also highlights it, and vice versa. Before this feature's release, a long time should be spent reading the components names one by one to find the matching item.

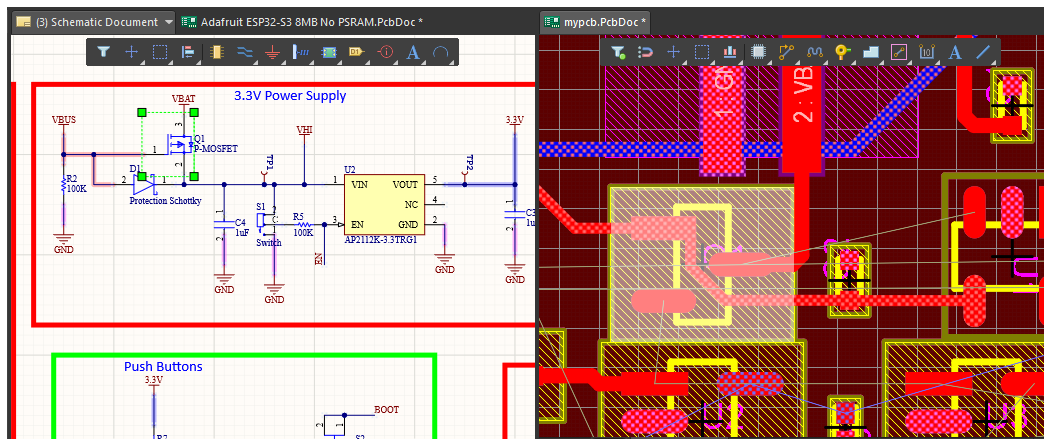


Figure 2.13: Cross Probe selection.

Conclusion

In this chapter, some foundations of the PCB design field were presented. Also, Altium Designer was picked to be the main EDA tool for the project. The next step is to start using those tools for the hardware design and manufacturing of the microcontroller.

Chapter 3

Hardware Design

This chapter's content will dive into the specific details and steps of the hardware design. It starts by exploring and explaining the different circuits used in the schematic design and then the PCB design with some tips and techniques used. Also, the soldering process will be briefly discussed. Finally, some code examples will be illustrated to test the functionality of the system.

3.1 Choice of the MCU

There are tens of ESP32 MCUs that gets released each year, The choice of a specific one to work with was based on the analysis resumed in the table below.

Module	Number of GPIOs	Flash	Size	Price	Status
ESP32-S2-SOLO-U	36	4 MB	18×19.2×3.2	\$2.15	Active
ESP32-S3-MINI-1	39	8 MB	15.4×20.5×2.4	\$3.1	Active
ESP8684-MINI-1	14	2 MB	13.2×16.6×2.4	\$1.82	Active
ESP32-C3-MINI-1	15	4 MB	13.2×16.6×2.4	\$1.9	Active
ESP32-WROOM-32	23	4 MB	18×25.5×3.1	\$3.2	NRND

Figure 3.1: Different ESP32 Modules.

In Figure 3.1, different variations of SoCs can be seen:

- One type like **ESP32-C3-MINI-1** that has many advantages except the very low number of GPIOs which is not practical at all for the project's design that is rich of pins.

- Another type like **ESP32-S2-SOLO-U** which is cheap and full of GPIOs but its large area and small memory size subject it to not being selected especially that IoT codes generally contains many APIs and are large of size.
- A third type like **ESP32-WROOM-32** is the most common ESP32 module which will ease programming it but considered as Not Recommended for New Designs (NRND) which means a lot of difficulties will be found to buy them in the market and also no longer software updates.
- That is why, the trade-offs were balanced and the decision was to work with **ESP32-S3-MINI-1**.

3.2 Schematic Design

Altium Designer's schematic design environment contains the blueprint of the circuit. It uses symbols to represent electronic components like resistors, transistors and MCUs. These symbols are then connected with lines, showing how they interact with each other. This schematic acts like a clear map, allowing to understand the overall functionality and connections before building the actual circuit

3.2.1 Accessing Components

The real-world component that will be mounted on the board is represented as a schematic symbol during design capture, and as a PCB footprint for board design.

Footprint: It is the land pattern of the component which contains all the details about the pads, their location and with the exact sizes. Footprints if not found can be drawn using the component's data-sheets which provides all the specific details. Figure 3.2 illustrates an example of a component with its footprint.

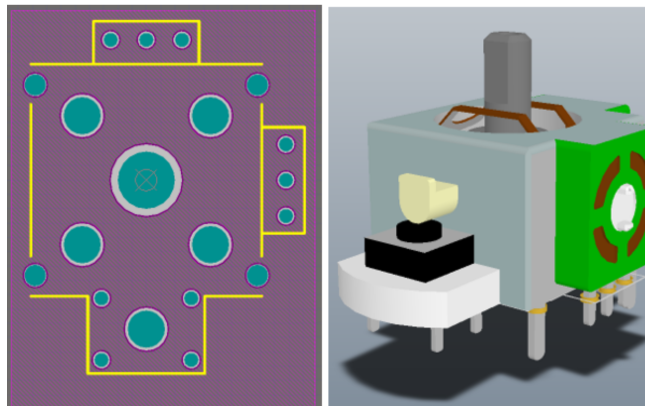


Figure 3.2: Real-World component with its footprint [9].

In Altium Designer, there are generally two ways to access components and use them in the schematic capture, standard components and Manufacturer Part Search.

Standard components:

It contains a library of standard components just like any other electronics simulation software. This method is the easiest and the fastest but it is very limited because each component will come with a one and specific footprint which may not fit the design requirements, e.g., it is larger than required, or may not be available on the market.

Manufacturer Part Search:

This tool is the best choice when working for professional designs but the most time consuming one. By finding any desired component with its exact reference number, the Manufacturer Part Search will give the matching one with the correct footprint to avoid any conflicts. Figure 3.3 shows how this tool is specific where it provides all the details including the price.

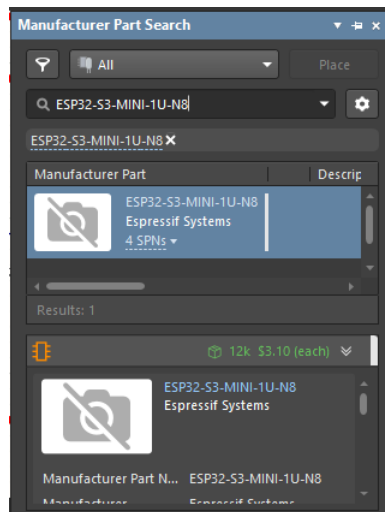


Figure 3.3: Manufacturer Part Search.

3.2.2 Circuit Design

Each microcontroller contains some common circuits to allow the proper and basic functioning of it. Those circuits will be discussed and explained in the following sections.

3.2.2.1 5V Power Supply

The port shown in Figure 3.4 is responsible for regulating and providing a 5V supply to be used later with the Low Dropout (LDO) circuit. The details are the following:

- USB Type C 2.0 has been used instead of the old MicroUSB standard which is outdated. It is a universal bus that can do various applications using one port.

- The A4B9 and B4A9 pins are the most critical ones and they provide the 5V regulated output. While A1B12 and B1A12 are the ground pins.
- B6, A6 and B7, A7 are the differential pairs D+ and D- respectively. They are crucial for data communication between the USB host and the MCU.

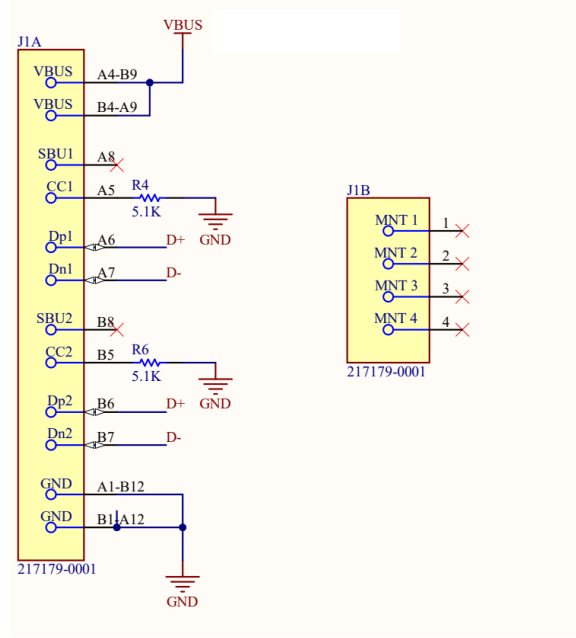


Figure 3.4: VBUS Power Supply.

- CC1 and CC2 channel pins are responsible for the cable orientation detection and power negotiation to provide the desired amount of current range. This side of the port is the recipient side so two 5.1K resistors should be used to get up to 1.5A current capability [19]. This specification is derived from Figure 3.5.

Rd Implementation	Nominal Value	Detect Power Capability?	Current Source to 1.7V - 5.5V
$\pm 20\%$ voltage clamp	1.1V	No	1.32V
$\pm 20\%$ resistor to GND	5.1k Ω	No	2.18V
$\pm 10\%$ resistor to GND	5.1k Ω	Yes	2.04V

Figure 3.5: Valid Upstream Facing Port (UFP) Rd Pull-down Resistor Values [19].

- SBU1 and SBU2 are the key features that allow the USB-C to be universal. They provide alternate modes and can be used for different protocols including display, TX/RX transmission ...etc. For the current design, those pins are not needed and therefore they are not even connected.
- MNT1-MNT4 are just mechanical pads that will be used later for soldering the port. It is highly recommended to add No Error Rule Check (ERC) cross marks

to the unused pins to signal Altium designer that they are not errors but done on purpose.

3.2.2.2 3.3V Power Supply

Figure 3.6 presents the main and sensible circuit responsible for delivering the accurate amount of power to the MCU to avoid burning it or providing insufficient current and thus power failures.

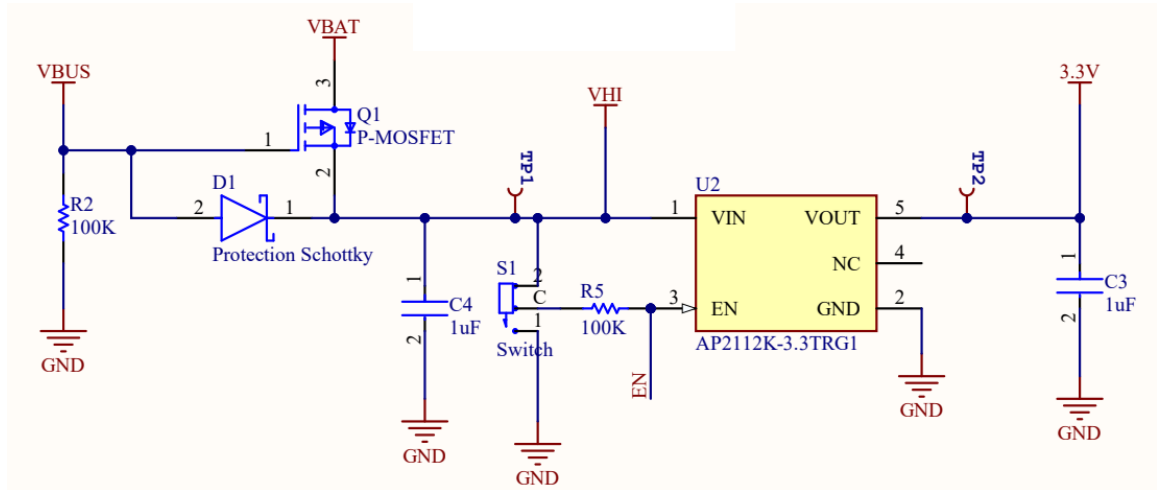


Figure 3.6: 3.3V Power supply circuit.

- The LDO is the core IC of this circuit. It provides a stable 3.3V output from a higher input voltage (Either the 5V VBUS or VBAT) with low dropout voltage, which means it can regulate the output voltage even when the input voltage is very close to the output voltage.
- For the circuit providing input voltage to the LDO , there are three scenarios. Firstly, If **VBUS** is present only, the gate of the P-MOSFET transistor is pulled high which means $V_{GS} > 0$ and thus the transistor is off and VIN will be powered directly from VBUS. Secondly, if **VBAT** is present only, the gate of the P-MOSFET is pulled down due to the R2 resistor which means $V_{GS} = -VBAT < -V_{th}$ and thus the transistor is ON which implies that VIN will be equal to VBAT. Lastly, if both **VBAT** and **VBUS** are present then $V_{GS} > 0$ because VBAT comes from a LiPo battery which is around 3.7V, hence $VBAT < VBUS$. Again, the transistor is OFF and thus VIN will be fed from VBUS leaving the battery for emergency situations only.
- Capacitors C3, C4 are used for decoupling purposes to ensure fluent and highly stable voltages.
- The Schottky diode is put there for protection against reverse polarities that produce high currents which may burn the power sources. The first scenario

that can cause reverse current is the fact that **VBUS** is higher than **VBAT**, so current might attempt to flow back into **VBAT**. The second case is that if one of the sources is suddenly disconnected, then the remaining source might try to supply power to the inactive one leading to reverse current flow.

- **EN** pin controls whether the LDO is turned on or off. We controlled it using a *switch* which is a feature rare to find in other microcontrollers. It is a better alternative to plugging an unplugging the USB cable continuously.

3.2.2.3 Additional Power Supply

Figure 3.7 represents another LDO circuit which is a different 3.3V power supply used to supply FeatherWings or Qwiic devices. This is done generally for two main reasons: ensuring enough current for some sensors and avoiding any MCU power failures and reducing the risk of noise or interference from the microcontroller affecting the I2C protocol communication.

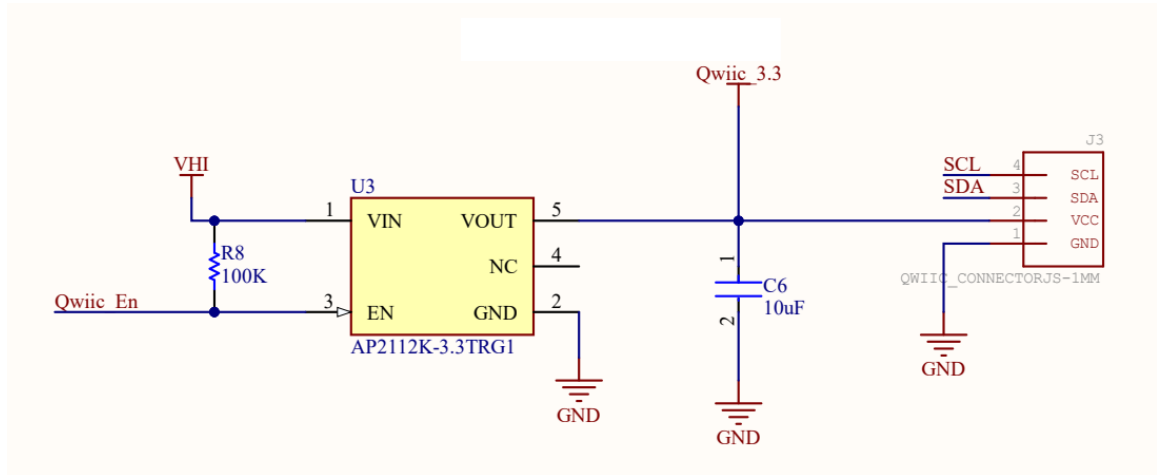


Figure 3.7: Qwiic Power Supply.

Figure 3.7 also shows the Qwiic connector used for I2C communications where SCL and SDA signals comes from the microcontroller. For the resistor R8, it is there to ensure the LDO is always active unless turned off manually using the *Qwiic-EN* signal.

3.2.2.4 Battery Charger

The circuit of Figure 3.8 is designed to charge a LiPo (Lithium Polymer) or Li-Ion (Lithium Ion) battery using the MCP73831T-2ACI/OT battery charger IC from Microchip.

- The MCP73831/2 devices are highly advanced linear charge management controllers for use in space-limited, cost-sensitive applications. The chip starts by outputting a constant current, once the battery voltage reaches a predefined

threshold, the controller switches to a constant voltage phase, maintaining the voltage and reducing the current until the battery is fully charged. This precise operation is done by the chip to ensure not damaging the battery or reducing its lifetime [20].

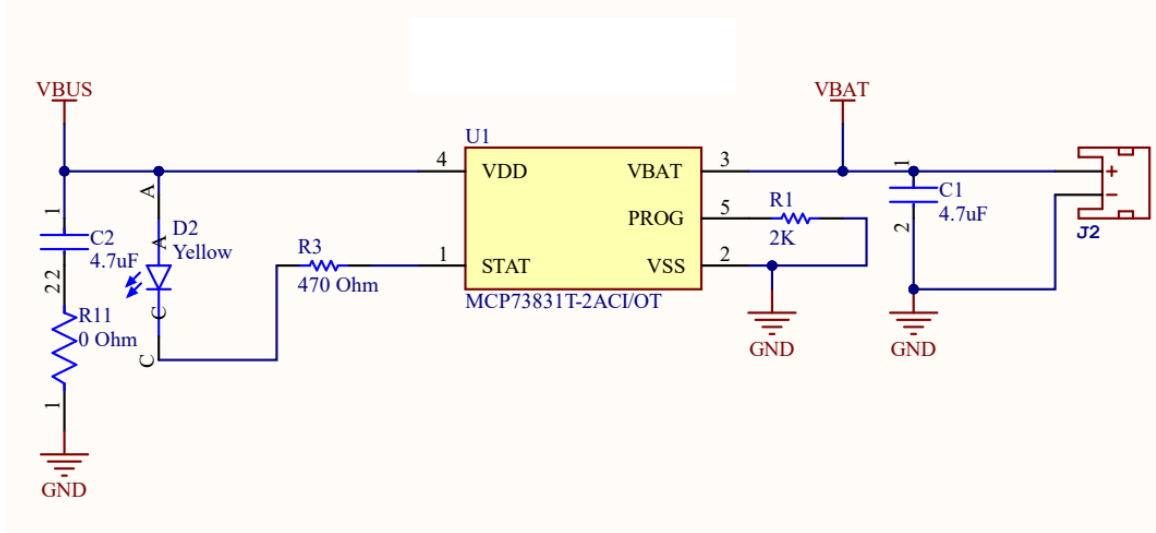


Figure 3.8: Battery charger circuit.

- VDD takes the power from VBUS and then feeds it properly to the battery through VBAT.
- The STAT pin is an open-drain output that indicates the charging status. When charging, the STAT pin is pulled low, turning on the LED (D2). When charging is complete, the STAT pin goes high, turning off the LED.
- PROG is the most important pin which should be used carefully since it is the one controlling how much current to be generated and this depends on the exact type of battery to be used. Figure 3.9 illustrates with a curve which resistance value should be used to get the desired amount of current, that is why 2K resistance has been chosen to get a 500 mA current capacity.

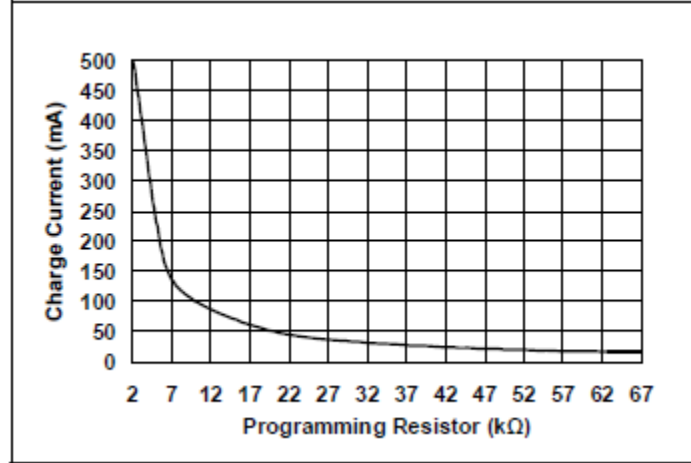


Figure 3.9: Charge Current vs Programming resistor.

3.2.2.5 Push Buttons

As can be seen in Figure 3.10, those two push buttons are used to control the signal BOOT and RESET.

- For the RESET signal, the resistor R7 pulls the RESET line to 3.3V by default, ensuring the ESP32 operates normally. When the button S3 is not pressed, the RESET line is held high. For the capacitor it is used for debouncing the key press and avoiding the pin to reset itself when near to environments with electric noise where unwanted resets can disrupt the operation of the microcontroller.

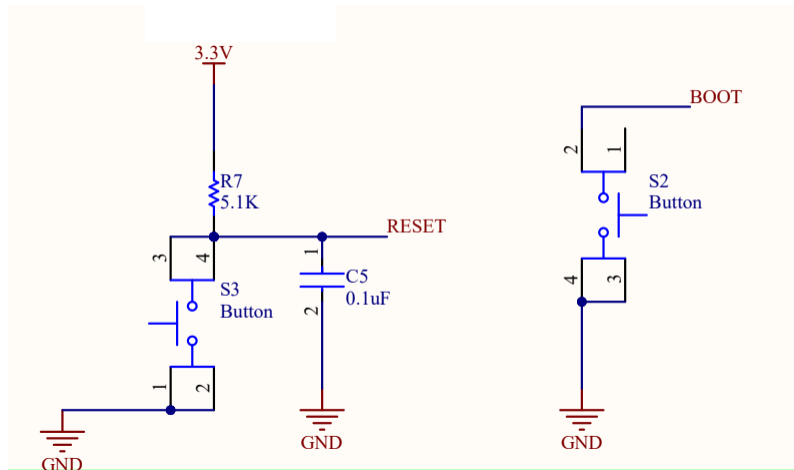


Figure 3.10: BOOT and RESET signals.

- S2 button is connected directly between the IO0 pin (BOOT) and ground. When the button is pressed, it pulls the IO0 pin low. For the ESP32-S3 to enter the bootloader mode, the IO0 pin must be held low during the power-up. Here, a pull up resistor is not needed since there is an internal one inside the MCU.

3.2.2.6 LEDs

The first RED LED shown in Figure 3.11 is a very common practice in microcontrollers design to indicate that it is on and outputting a valid voltage. The D4 is just an internal USER LED added for testing and debugging purposes.

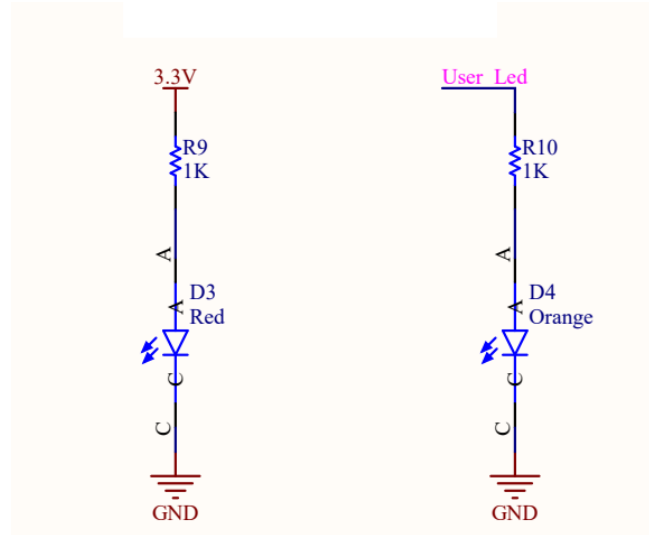


Figure 3.11: Power Led & User LED

3.2.2.7 RGB LED

The circuit in Figure 3.12 is designed to control a WS2812B RGB LED using the microcontroller.

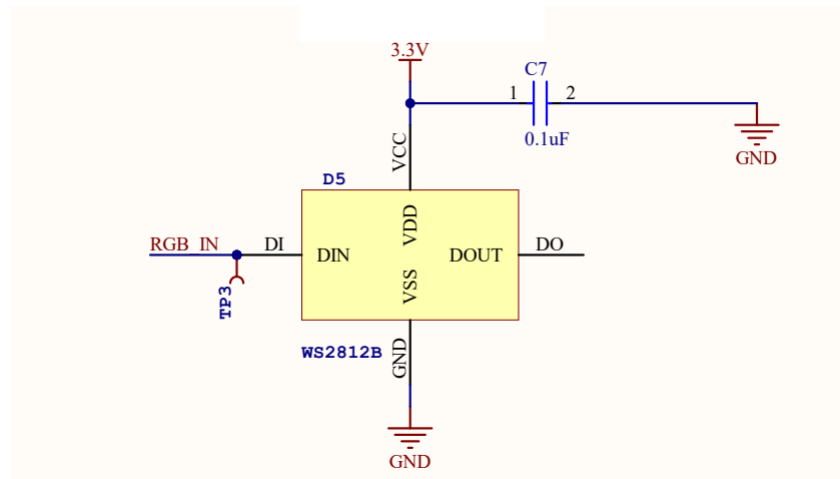


Figure 3.12: An RGB LED circuit.

- The DIN pin receives the data signal that controls the color and brightness of the LED. This signal is typically generated by the ESP32 and consists of a specific timing protocol that the WS2812B interprets to set the LED colors.

- An additional item that has been used before for the 3.3V output which is the **testpoint**. It is very crucial for debugging and testing that signals are outputted correctly. Since tracks are masked and many components are so small, a large testpoint will be the best solution as shown in Figure 3.13.

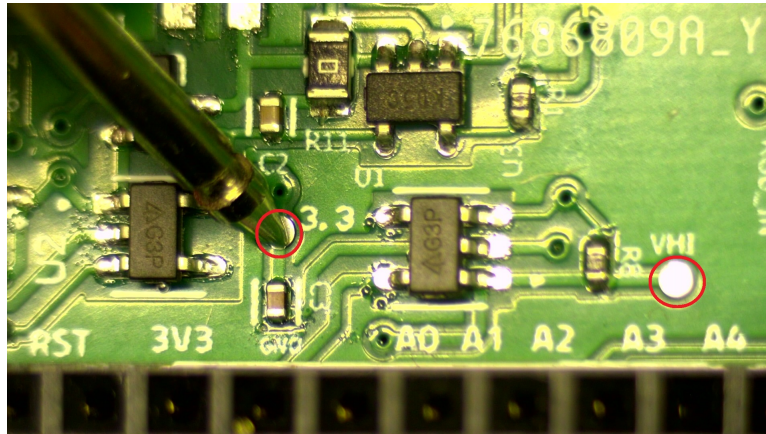


Figure 3.13: Using the multimeter with a testpoint.

3.2.2.8 Add-Ons Headers

The headers shown in Figure 3.14 is the key component that allows the microcontroller to support various types of add-ons. The *Blue dotted* pair of lines are the headers for the FeatherWings, while the *red dotted* ones are for mikroBUS and the *green spot* is for Pmod compatibility.

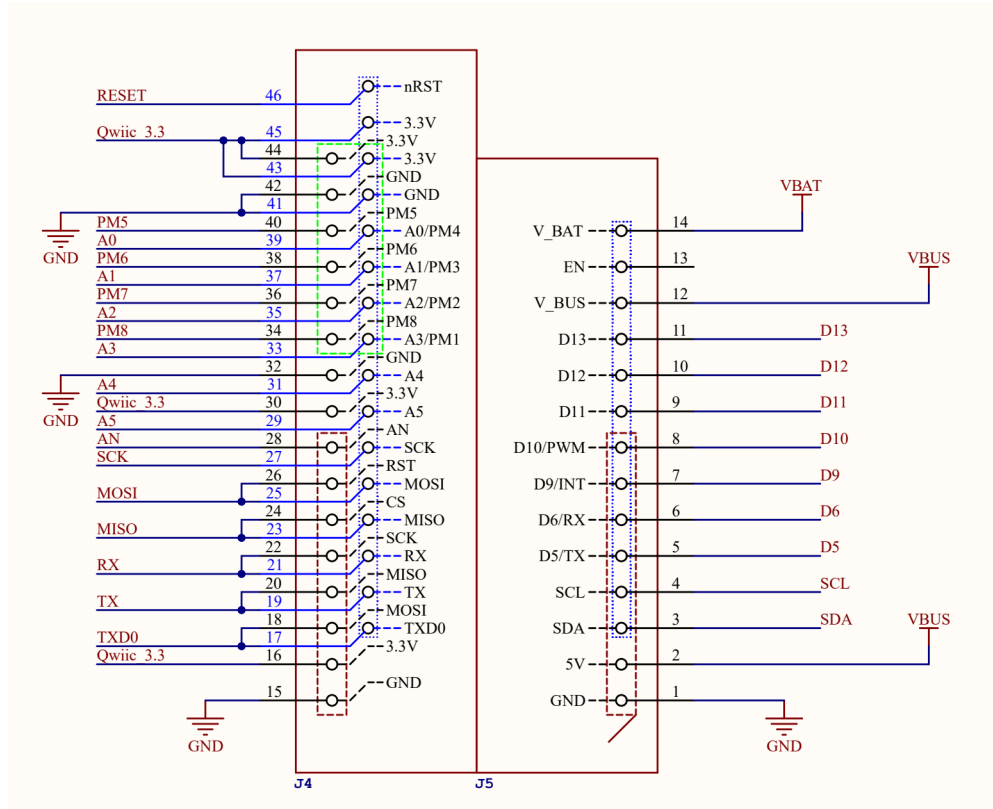


Figure 3.14: The Add-ons headers.

The previous blocks were designed manually and the steps of doing that start by creating a personal integrated library which contains a schematic library and a PCB Library.

Schematic Library: Figure 3.15 summarizes the steps of creating the headers. (a) Firstly, it starts by creating two rectangles and putting them adjacent. (b) Secondly, this is the disturbing and time consuming part where lines and pins should be created and drawn one by one keeping the proper spacing to produce a professional work. (c) Lastly, all the names should be added based on the specifications and datasheets of the supported add-ons.

to ease moving it as a block and set the horizontal distances between them. Finally, silkscreen markings are added to enhance the readability. Figure 3.17 contains the final footprint.

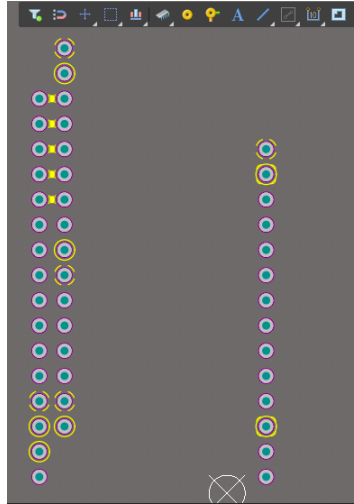


Figure 3.17: The Headers' footprint.

3.2.2.9 MCU pins

After finishing with the design of all the circuits, the last part in this section is to wire everything with the MCU pins. This step should be studied carefully by looking to both the ESP32 SoC datasheet and the other Add-Ons requirements in order to wire digital pins, analog pins and serial protocols to their proper places to guarantee a fully functional system. Figure 3.18 contains the MCU pins after wiring.

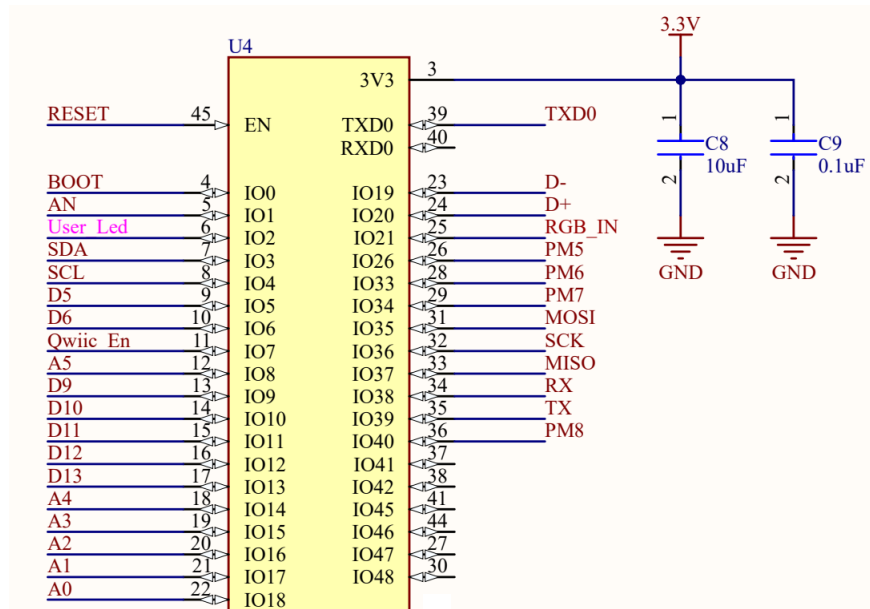


Figure 3.18: MCU wired pins

3.3 PCB Design

After finishing the schematic design part and studying well the proper components to be used. The next and the most important step in this project is achieving a correct and fully functional PCB design which may take a long time to bypass all the possible problems and limitations.

3.3.1 Golden Rules

Designing a potential PCB design can be made simpler by observing some basic rules that ensure the PCBs are not only in good working condition but are also easily manufacturable. The following guidelines must be respected in order not to make common mistakes and in order to get a good PCB design quality. Two types of rules will be presented, basic rules and High Speed Design (HSD) rules.

Basic rules:

Those are common rules that should be considered in any PCB design and they are listed below.

- **Component placement:** Large components must be placed first in specific locations due to mechanical enclosure constraints or their size like the MCU, USB-C and the headers. In a PCB layout, the number of crossing nets should be reduced the maximum to avoid layer transitions through vias and make routing guidelines easier to follow and this is done by trying multiple positions and orientations of the components until the unrouted nets look less messy and crossed.
- **Power Planes:** making a proper and efficient ground routing can cause a lot of headaches since it needs some experience, the best alternative to avoid most of these details is to create two inner planes for both ground and power [23].
- **Trace width:** There is some trade-off when setting the trace width. Each signal should be studied based on how much current will it carry and the frequency that will pass. Large trace width increases the capacitance between the tracks while thin tracks may heat or burn due to the current limitation [23].
- **Separating things:** Separating analog and digital components on a PCB is essential to prevent interference. Analog and digital circuits operate at different voltage levels and frequencies, making them subject to different types of noise. By physically isolating them, designers can minimize interference and maintain signal accuracy.

HSD rules

Those rules are not strictly required to apply in this project since the range of frequencies is medium but they should be taken into consideration whenever possible.

- **Avoid reflections:** Reflections occur when signals encounter impedance mismatches, typically at discontinuities like connectors or changes in trace width. To avoid them, controlled impedance routing must be used, matching impedances at signal transitions and minimizing sudden changes in trace geometry [24].
- **Minimize Crosstalks:** Cross talk happens when signals from one trace interfere with adjacent traces due to capacitive or inductive coupling. To avoid this phenomenon, space between traces should be increased, usage of differential signaling where possible, and employing ground planes or shielding to isolate sensitive signals from interference [24].
- **Minimize EMI:** EMI results from radiated or conducted emissions from electronic devices, causing interference with nearby circuits or external equipment. To reduce EMI, employ proper grounding techniques, use shielding for sensitive components, filter high-frequency signals with specialized capacitors, and comply with EMI standards during PCB layout and design [24].

3.3.2 Manufacturing Rules

Each PCB manufacturer company have rules that ensure PCB layouts are manufacturable and perform well. These rules cover trace width, clearance, hole size, solder mask expansion and more. Many beginner designers start working with the default Altium standard rules that are maybe be not manufacturable and then they will receive a malfunctioning system. In this project, JLCPCB company was chosen and all the constraints and capabilities can be found in their official site, a sample of those are illustrated in Figure 3.19.

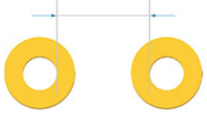
Pad to Pad clearance(Pad with hole, Different nets)	0.5mm	
Via to Track	0.254mm	
PTH to Track	0.33mm	
NPTH to Track	0.254mm	
Pad to Track	0.2mm	

Figure 3.19: JLCPCB Manufacturing Capabilities [10].

There are around 40 rules considering all the aspects of the design. The next step is that they have to be manually entered to the Altium designer software so that it signals you whenever you make a violation and you ensure that your design can be properly manufactured. Figure 3.20 provides an example for clearance constrain rules inside Altium where each pair of objects represents the maximum allowed spacing between them.

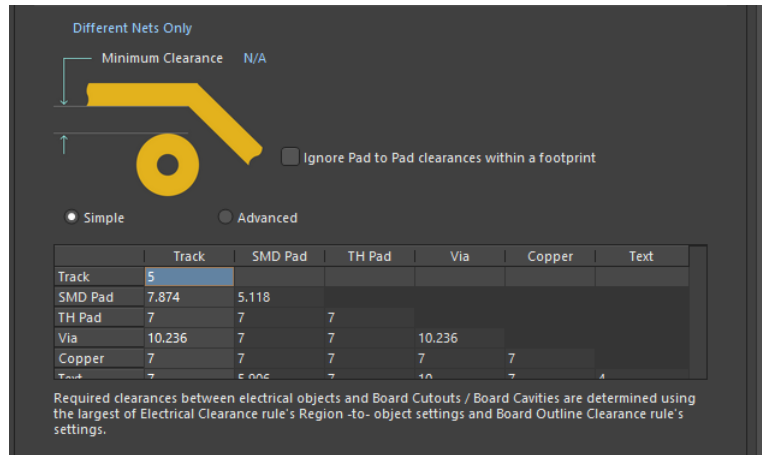


Figure 3.20: Altium Designer Rule wizard.

3.3.3 Steps of the design

This section provides a general overview about the steps done to achieve the final PCB design covering some problems and limitations faced and how they were solved.

3.3.3.1 Component placement

The first step to face after finishing the schematic design is that a large number of components are distributed everywhere, figure 3.21 clarifies this situation. It is preferable to take a long amount of time trying and figuring out how to find the most optimal way for component placement because it will decide whether the design will be simple as 4 layers or complex up to 8 layers.

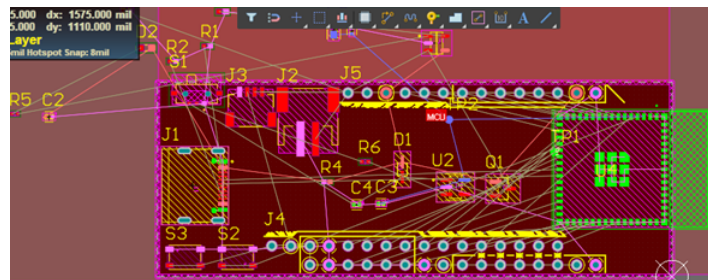


Figure 3.21: Component auto placement from Altium.

3.3.3.2 Routing Signal Tracks

Since a special two inner layers were dedicated for ground, it will be left as the last one to route. The first priority is to start routing the signals from the headers to the MCU because there are a plenty of them and they would take most of the space. After that, the tracks between the inner components are made and come orientations can be made during this stage since a better view will be provided.

3.3.3.3 Power signals

As seen before in the schematic design, there are two power sources which are the 3.3V and the Qwiic-3.3V. Due to this division, a serious time consuming problem happened here where no way was found to route the Qwiic-3.3V pin shown in Figure 3.22 which is extremely necessary to feed the mikroBUS sockets.

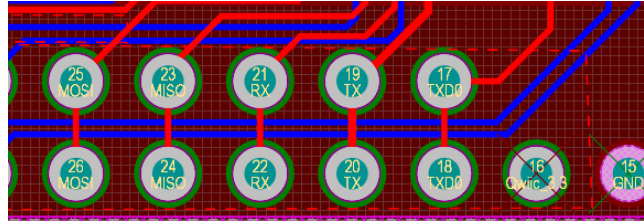


Figure 3.22: Qwiic-3.3V pin.

The solution to this problem was a recent technique which is the manufacturers' capability to divide the power layer into distinct copper pieces that are totally unrelated. Figure 3.23 illustrates how this technique was used to bypass the previous problem and avoid going further to 6 layers. The green square clarifies that the pin now is directly attached to the plane and no need for routing.

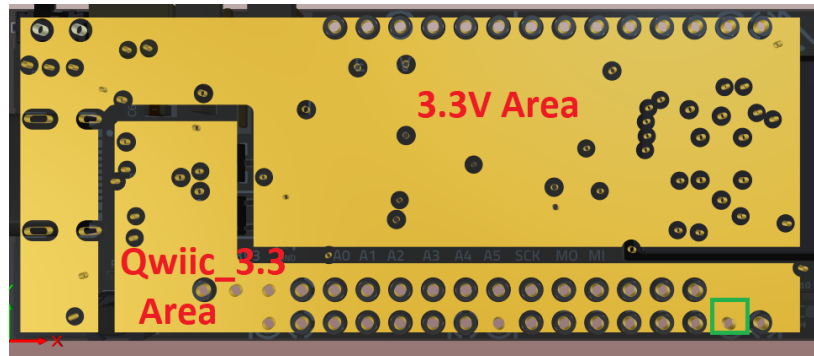


Figure 3.23: 3D View of the power plane

3.3.3.4 Ground routing

This step is simple since there is a dedicated ground plane and a vias can be inserted whenever a routing constraint is found, but another challenge was encountered here.

Figure 3.24a clarifies the inability to route those two ground pins nor create a vias since D+ and D- are passing through the bottom layer as shown in black. The solution to this problem is presented in Figure 3.24 and it is a trivial component called **Zero Ohm Resistance** which is just like an SMD wire that created another dimension and again avoiding high costs with 6 layer design.

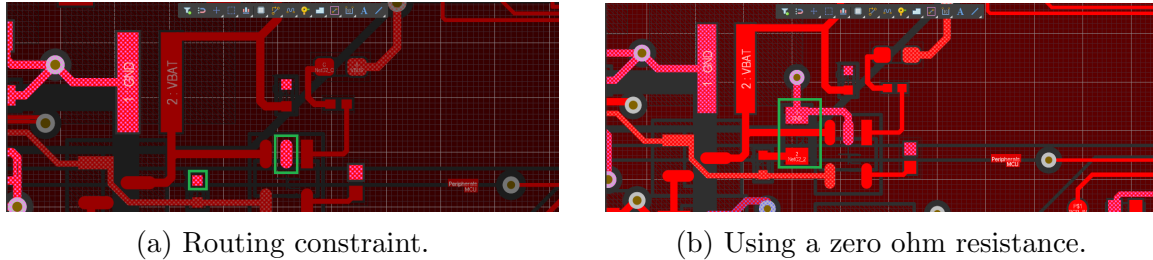


Figure 3.24: Routing constraint solved.

3.3.3.5 Creating a Polygon

Polygon pours, or copper planes in PCB design offer several advantages. By filling empty areas with copper, they improve signal integrity, reduce electromagnetic interference, and minimize ground loops. But the most important reason is to decrease manufacturing costs by reducing etching time and material usage.

They can be created automatically by Altium, after that just some spaces should be manually cut and enhanced especially around the ESP32 antenna since any adjacent high power currents will create interferences and Wi-Fi might fail to work.

3.3.4 Final design

The last step is just to add the top and bottom overlay layers and fill them with the texts and designators of proper sizes to ease the usage of the board. The board shape after finishing is shown in Figure 3.25.

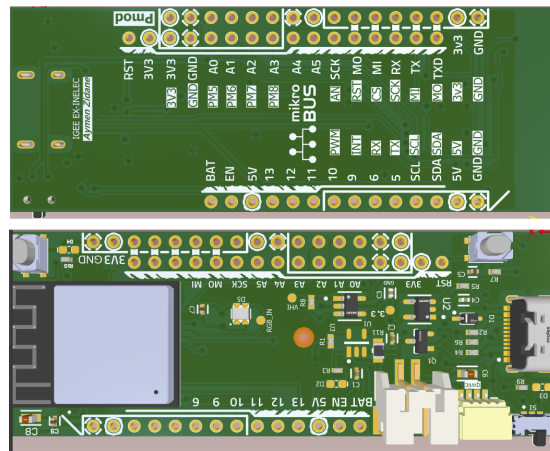


Figure 3.25: 3D view of the board.

3.4 Ordering the PCB

After finishing everything with the design and making sure it is correct, **gerber** files should be generated for the manufacturer company. They are the standard file format used in the electronics industry to describe the PCB designs. They contain information about the layers of a PCB, including copper layers, solder mask, silkscreen, and drilling data. Gerber files contain only the necessary data for manufacturing the PCB, without revealing the full schematic, netlist, or design logic. An example of them is clarified in Figure 3.26.

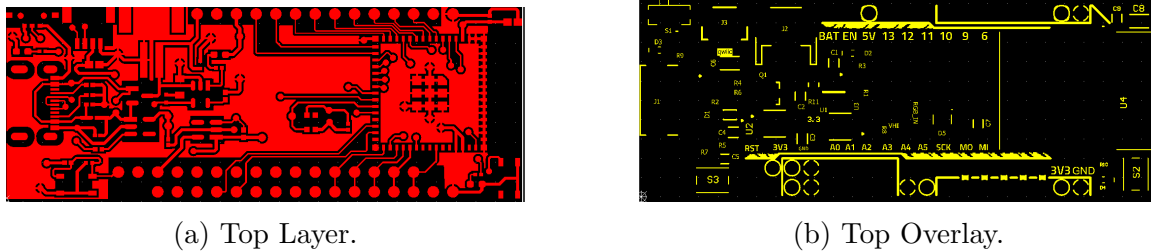
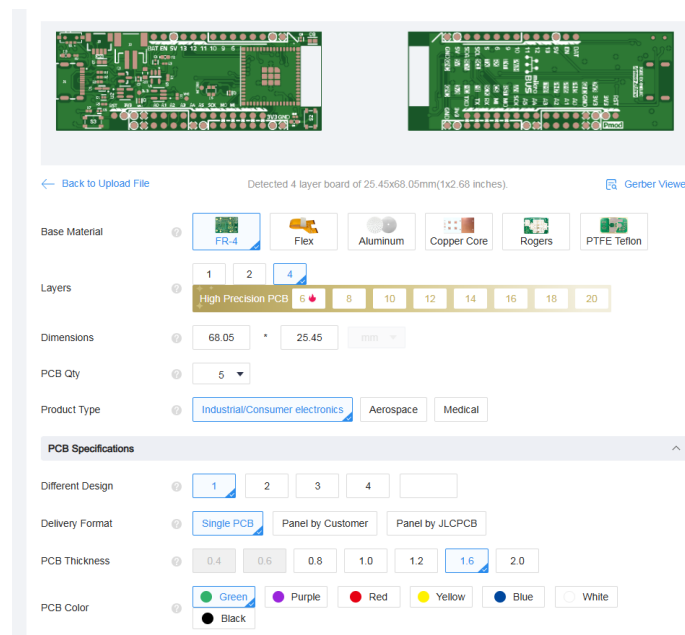


Figure 3.26: Example of two gerber files.

3.4.1 PCB Specification

After entering to the **JLCPCB** website, all the gerber files have to be uploaded in one packed zip file. After that, a list of options appear where all the details can be entered (like color, quantity, thickness and more) based on the design and the preferences, some of the options are shown in Figure 3.27.



← Back to Upload File

Detected 4 layer board of 25.45x68.05mm(1x2.68 inches).

Gerber Viewer

Base Material: FR-4, Flex, Aluminum, Copper Core, Rogers, PTFE Teflon

Layers: 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20

Dimensions: 68.05 x 25.45 mm

PCB Qty: 5

Product Type: Industrial/Consumer electronics, Aerospace, Medical

PCB Specifications

Different Design: 1, 2, 3, 4

Delivery Format: Single PCB, Panel by Customer, Panel by JLCPCB

PCB Thickness: 0.4, 0.6, 0.8, 1.0, 1.2, 1.6, 2.0

PCB Color: Green, Purple, Red, Yellow, Blue, White, Black

Figure 3.27: PCB Specifications in JLCPCB.

3.4.2 PCBA

Printed Circuit Board Assembly (PCBA) is the process of soldering electronic components onto a PCB. It involves placing resistors, capacitors, and integrated circuits on designated areas and soldering for electrical connections. This step is crucial for device manufacturing, ensuring secure attachment and proper functionality.

Due to some constraints in our project, only the MCU was requested to be assembled from JLCPCB since its pads are very small and impossible to manually solder. In the other hand, JLCPCB did not have all the components required for assembling the microcontroller.

3.4.3 The stencil

A PCB stencil is a thin, flat sheet or foil, typically made of stainless steel or sometimes polyimide, used in the SMT process to apply solder paste accurately onto the PCB. The stencil has precisely cut openings that correspond to the pads on the PCB where components will be placed. Figure 3.28 clarifies an example of them.

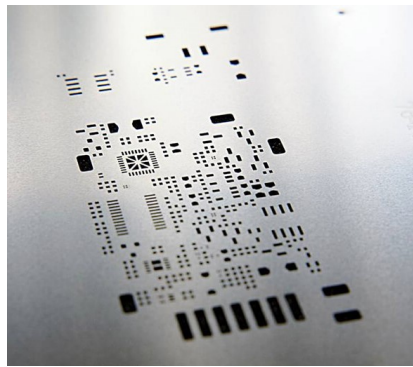


Figure 3.28: A stencil example.

Since the PCB will be assembled with the ESP32 module only, the default stencil provided by JLCPCB could not be used but instead we made a modified version of the design to keep the MCU space empty and can use it later on the soldering process. Figure 3.29 shows the solder paste layer sent to the manufacturers.

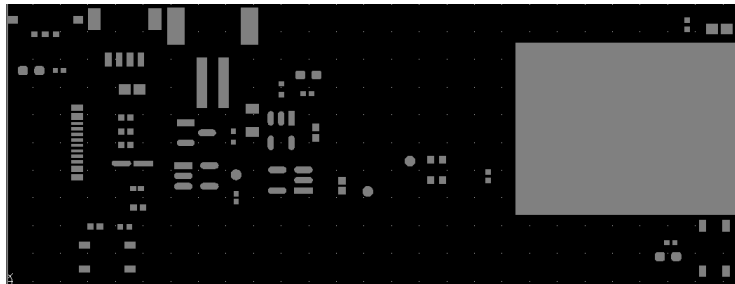


Figure 3.29: Solder paste layer.

3.4.4 Components

In Altium designer, there is an option to generate a file called **Bill of Materials (BOM)** which contains all the components used inside the project and their reference number. This file can be used and entered to international selling sites like *Mouser* to order and buy everything needed.

3.5 Soldering the board

After both the PCB and components arrived, another critical stage to achieve in the project which is soldering everything altogether.

3.5.1 Tools Used

The board size is very compact and most of the components are of 0402 package. Therefore, some professional tools are needed to achieve the objective.

3.5.1.1 Microscope

The camera-equipped microscope shown in Figure 3.30 has been used during the soldering process. It allows to see components and solder joints clearly at high magnification. Also, the camera provides the capability of watching and recording in real time, which helps with quality control and fixing problems.

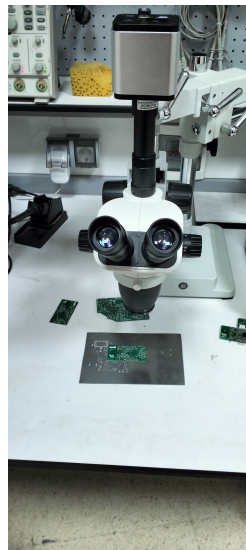


Figure 3.30: A microscope with an integrated camera.

3.5.1.2 Automatic Vacuum Pen

Figure 3.31 shows the VAC-12000 automatic vacuum pen which is a special tool for precise handling of electronic components. It uses a soft vacuum to pick up very small

surface SMDs that are totally impossible to hold with hands or normal tweezers. With its automatic operation, it improves productivity and accuracy, making it ideal for high-density PCB designs.



Figure 3.31: VAC-12000.

3.5.2 Applying Paste

The process starts by putting many boards of the same thickness adjacent to each other and the desired one in the middle to provide a stable area for the stencil, this is shown in Figure 3.32. Then, some solder paste is applied all across the edges of some solid card. Finally, the card is positioned at a 45° angle with the stencil and the process of passing it evenly starts.

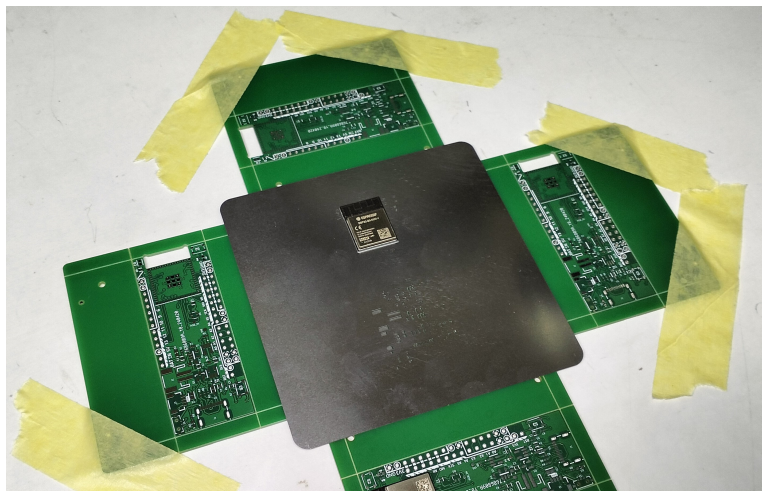


Figure 3.32: Usage of the stencil.

This part is generally optional, especially that the stencil made the work very clean and easy but the paste used was expired and of a medium quality which could probably cause solder bridges. It is required to use the camera and the smallest tool possible to be able to clean only the required spots without messing with other areas. Figure 3.33 provides two examples of the process.

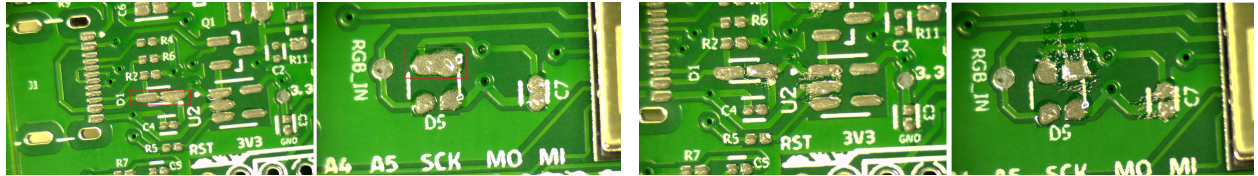


Figure 3.34: A clean PCB view for the board.

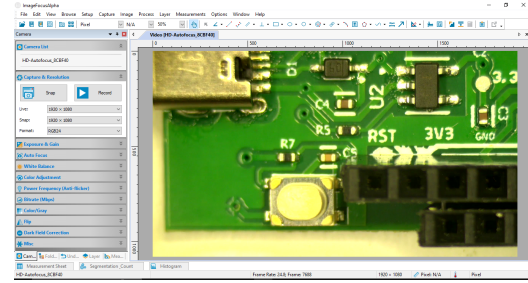
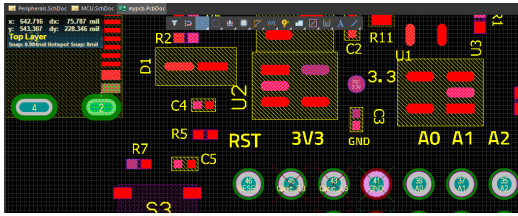
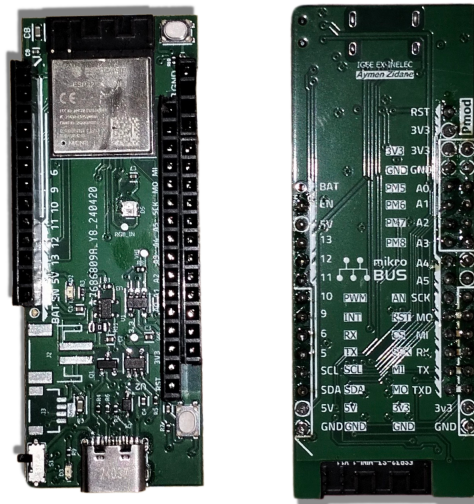


Figure 3.35: Altium & Camera views of the board.

- Finally, after all the devices are placed in their designated positions, the hot air gun is used to heat each spot. The process continues until the solder paste melts and then solidifies, ensuring secure connections for all components. Before powering the microcontroller on, the Multimeter is set to **beep continuity mode** to check each component individually. This ensures that no solder bridges or short circuits are present.

The final design after finishing all the steps is presented in Figure 3.36.



Conclusion

In conclusion, this chapter detailed the process of designing the hardware for the microcontroller, including choosing the MCU and using Altium Designer for schematic and PCB design. Also, The most important tips and tricks used in both the design

and implementation phases were illustrated. The next step is to test everything is working correctly and it will be discussed in the next chapter.

Chapter 4

Results and discussions

To examine the microcontroller design proposed in this project4.1A series of different tests have been done to ensure the board’s proper functioning and capability to be used for real world projects.

4.1 Pinout Diagram

Figure 4.1 shows the pinout diagram of the board which will be used for the programming purpose and Arduino IDE pin assignments.

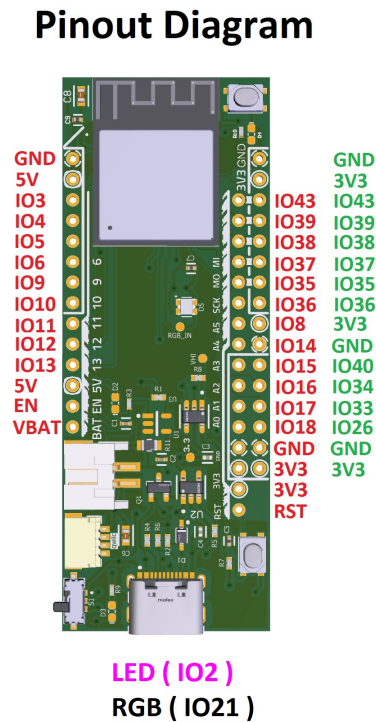


Figure 4.1: The microcontroller’s pinout diagram.

4.2 Testing the board

Different things will be tested inside this section starting from primary operations until a real world sensor application.

4.2.1 GPIOs

General-Purpose Input/Output (GPIO) is a very important key component in the board since it signals all the intelligent decisions done by the MCU, therefore, it is crucial to test that they are all compatible with the pinout diagram and functioning as expected. Figure 4.2 illustrates a simple way proposed to test them in a short time. The code defines an array containing all the pin numbers that exists classified exactly as the real design. After that, loop iterates through the GPIOs one by one turning them ON then OFF for one second and the multimeter is used during those 2 seconds to test that all of them are outputting the required voltage and at the right pin.

```
const int pins[] = {3, 4, 5, 6, 9, 10, 11, 23, 13, //leftmost pins
                   43, 39, 38, 37, 35, 36, 8, 14, 15, 16, 17, 18, //right pins 1
                   43, 39, 38, 37, 35, 36, 40, 34, 33, 26 // right pins 2
                   }; //Array of pins

const int numPins = sizeof(pins) / sizeof(pins[0]); // number of pins

void setup() {
    //Initialize pins
    for (int i = 0; i < numPins; i++) {
        pinMode(pins[i], OUTPUT);
    }

    // Loop through each pin in the array
    for (int i = 0; i < numPins; i++) {
        digitalWrite(pins[i], HIGH);
        delay(1000);
        digitalWrite(pins[i], LOW);
        delay(1000);
    }
}

void loop() {
}
```

Figure 4.2: Code for testing GPIOs.

4.2.2 User LED

The board's built-in user led is generally used for all applications to signal different events. Also, testing its proper working proves that current rating requirements are met and the MCU can feed some basic low power loads. Figure 4.3 shows the result brightness status for a simple blinking code. Also, the analog pin has been tested here and it resulted in a smooth brightness change thus an accurate Pulse Width Modulation (PWM) behavior.

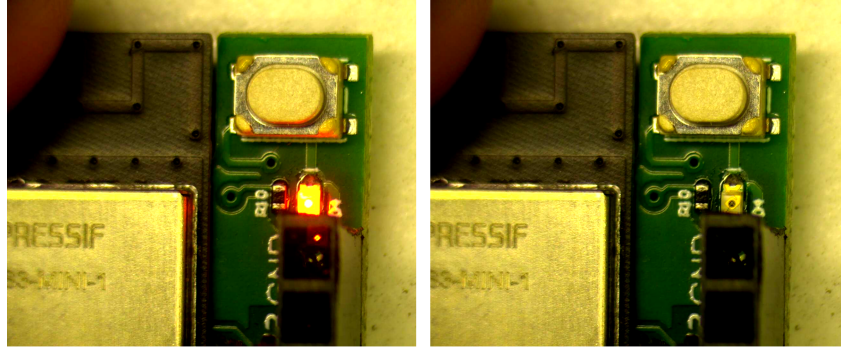


Figure 4.3: LED Blinking.

4.2.3 RGB

RGB LEDs are commonly used in IoT applications to indicate different states of outputs. The Adafruit RGB was tested with their library codes on different colors as clarified in Figure 4.4 to verify it was soldered in the correct orientation and performing as expected.

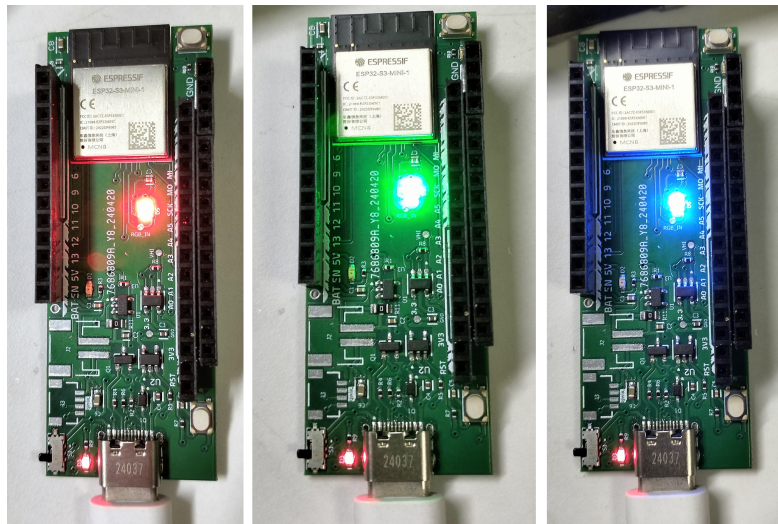


Figure 4.4: RGB colors.

4.2.4 mikroBUS add-on

The core of the project is to ease the work with the supported add-ons. Temp&Hum 12 click board was taken as an example, the sensor IC used on the board is HDC2080, a Low Power Humidity and Temperature Digital Sensor from Texas Instruments that works with the I2C protocol.

A portion of the code used for testing the previous sensor is shown in Figure 4.5. This code sets up the ESP32 to read temperature and humidity data from an HDC2080 sensor and serve the data via both the serial output and a local web page. The ESP32 connects to a Wi-Fi network and starts a server on port 80. In the main loop,

it triggers a measurement on the HDC2080, waits for it to complete, then reads the temperature and humidity data. This data is printed to the serial monitor and served dynamically on a webpage.

```
void loop() {
  WiFiClient client = server.available();
  if (client) {
    // Trigger new measurement
    Wire.beginTransmission(HDC2080_ADDR);
    Wire.write(MEASUREMENT_CONFIG);
    Wire.write(0x01); // Start measurement
    uint8_t error = Wire.endTransmission();
    if (error != 0) {
      Serial.print("Error in I2C communication: ");
      Serial.println(error);
      delay(1000); // Wait before retrying
      return;
    }

    delay(100); // Wait for measurement to complete

    // Read temperature and humidity
    float temperature = readTemperature();
    float humidity = readHumidity();

    // Print temperature and humidity
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.println(" °C");
  }
}
```

Figure 4.5: HDC2080 Arduino code.

For this part, we proposed a comparison between the interfacing of the temperature and humidity sensor with a normal ESP32 then with our proposed module.

Step1: Using a normal ESP32

After running the code discussed earlier on the ESP32, we can see in Figure 4.6 the display of updated temperature and humidity values on both the serial monitor and the webpage.

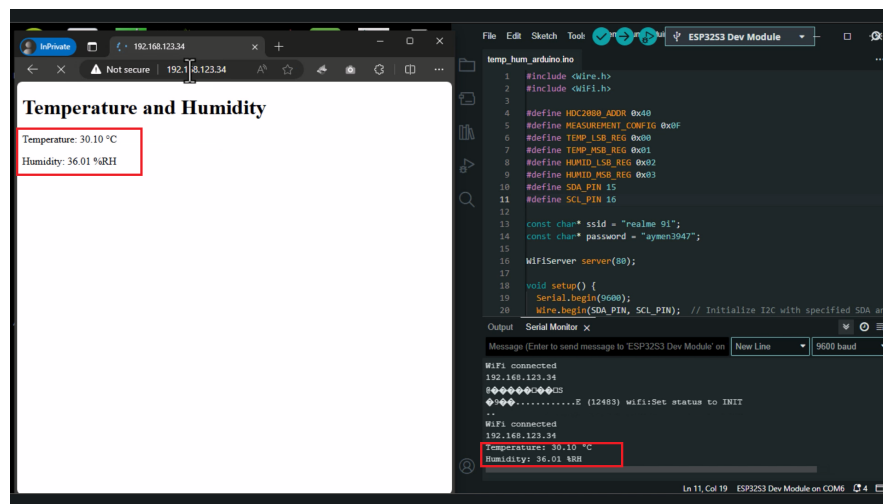


Figure 4.6: Measurement results with the normal ESP32.

The **first** approach which is interfacing the mikroBUS chip with a normal ESP32 requires the usage of a breadboard breadboard as illustrated in Figure 3.39. It takes a large size with an inflexible wiring method.

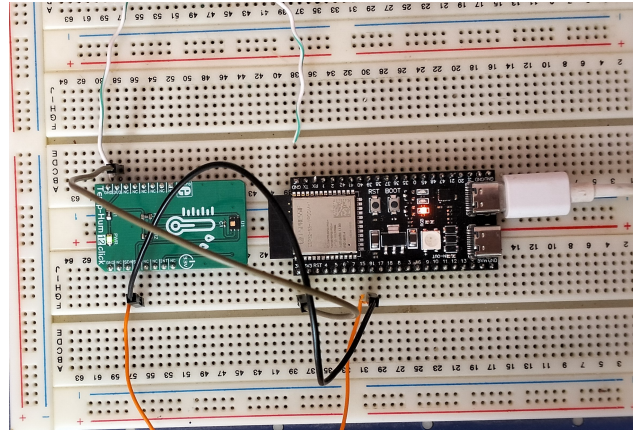


Figure 4.7: Interfacing the sensor with normal ESP32.

Step2: Using the designed board

A small thing to change before running the code is the SDA and SCL pin locations based on the pinout diagram presented before. Figure 4.8 shows very similar results to the **first** design which validates that the board is operating efficiently. Another remark to notice here is that the MCU can print the data in the webpage *only* and not in the serial monitor since it doesn't have a built in USB to Serial converter.

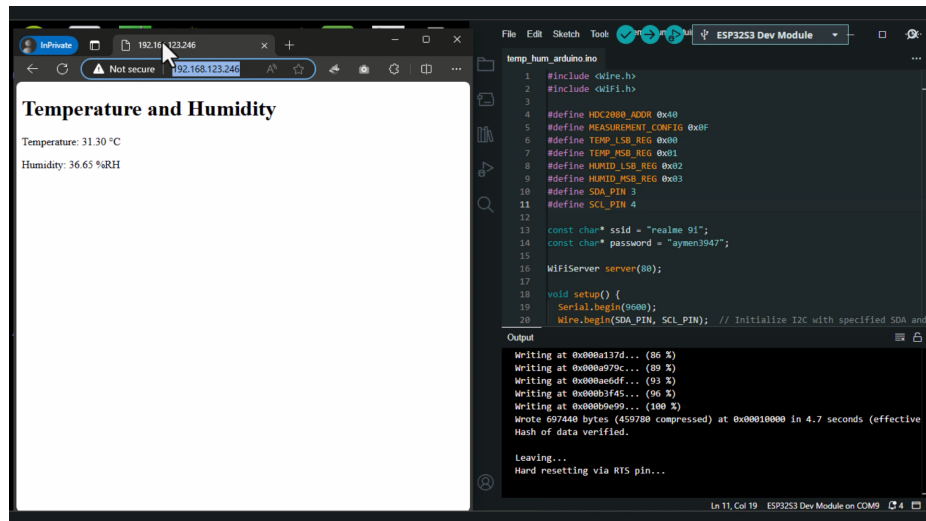


Figure 4.8: Measurement results with our MCU.

The **second** approach used is plugging the click board with the MCU. Figure 4.9 illustrates a big difference in terms of size compactness and being able to use the system as a final design since it is robust and well attached to the board.

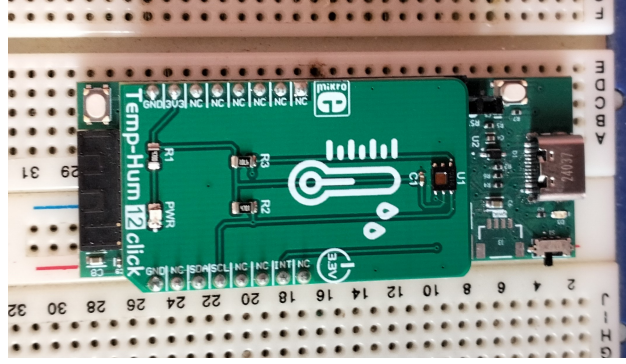


Figure 4.9: Interfacing the sensor with the implemented MCU.

4.3 Comparison

In this section, a detailed comparison will be done between the implemented board and the ESP32-S3-DevKitM-1 from Espressif since both use the exact same ESP32 module.

Size:

Our development board is of size $6.8\text{cm} \times 2.5\text{cm}$ while the Espressif DevKitM one is $6.2\text{cm} \times 2.54\text{cm}$. The size is very similar between the two.

Price:

ESP32-S3-DevKitM-1 is sold for around \$. Our board costed \$72 for components ($12\$ + 60\$$ for shipping) and \$77.52 for the PCB ($\$7\text{ PCB} + \$42.71\text{ PCBA} + \$7\text{ Stencil} + \20.81 shipping). The shipping for one board is too expensive and also the design was a first trial and not fully optimized, so after other enhancements in the future, the price may become cheap.

Compatibility:

The designed board can directly support 4 add-ons (3 at the same time) and it is better for less space and time, while using the normal ESP32 is for general purpose applications and for prototyping only.

Conclusion

This chapter presented a comprehensive analysis of the microcontroller design through a series of tests including GPIOs, user LED, RGB LED, and the mikroBUS add-on usage. Also, a comparison was done between our MCU and a normal ESP32. The successful implementation of these tests demonstrates the effectiveness and reliability of the designed MCU.

General Conclusion

In conclusion, the project done focused on developing an ESP32-based microcontroller design. The work began with creating a detailed schematic diagram using Altium Designer, incorporating key circuits such as power supply regulation and filtering, Adafruit battery charging and GPIO configurations with setup. The PCB layout followed, ensuring optimal component placement and efficient routing techniques to minimize noise and enhance signal integrity while maintaining a 4 layer design avoiding high costs. After that, soldering steps were presented which involved both manual techniques for normal components and SMT methods for the module. Finally, testing included verifying GPIO functions, LEDs, Wi-Fi connectivity and I2C communication via the temperature and humidity sensor application.

The microcontroller operated efficiently and delivered accurate results, making it a versatile and cost-effective solution for developers

Future Work

Although that the board provided satisfying results, some limitations were faced. One significant limitation was the absence of a built-in USB to serial converter, which restricted the microcontroller's ability to output data to the serial monitor and that is a great drawback making the debugging part complex and time consuming. Additionally, the board cannot handle plugging the Feather and mikroBUS sockets simultaneously except with the usage of additional stack-up headers, and this interfacing combination is required in many applications.

Future work should focus on addressing these limitations and expanding the capabilities of the microcontroller. Adding a USB to Serial chip will significantly ease the debugging process for the programmers. Also, Providing complete support within the Arduino IDE and creating specialized software development kits (SDKs) for the board will improve the platform's usability and accessibility. The previous improvements may give a chance for the board to be fully supported and widely used in the embedded systems community.

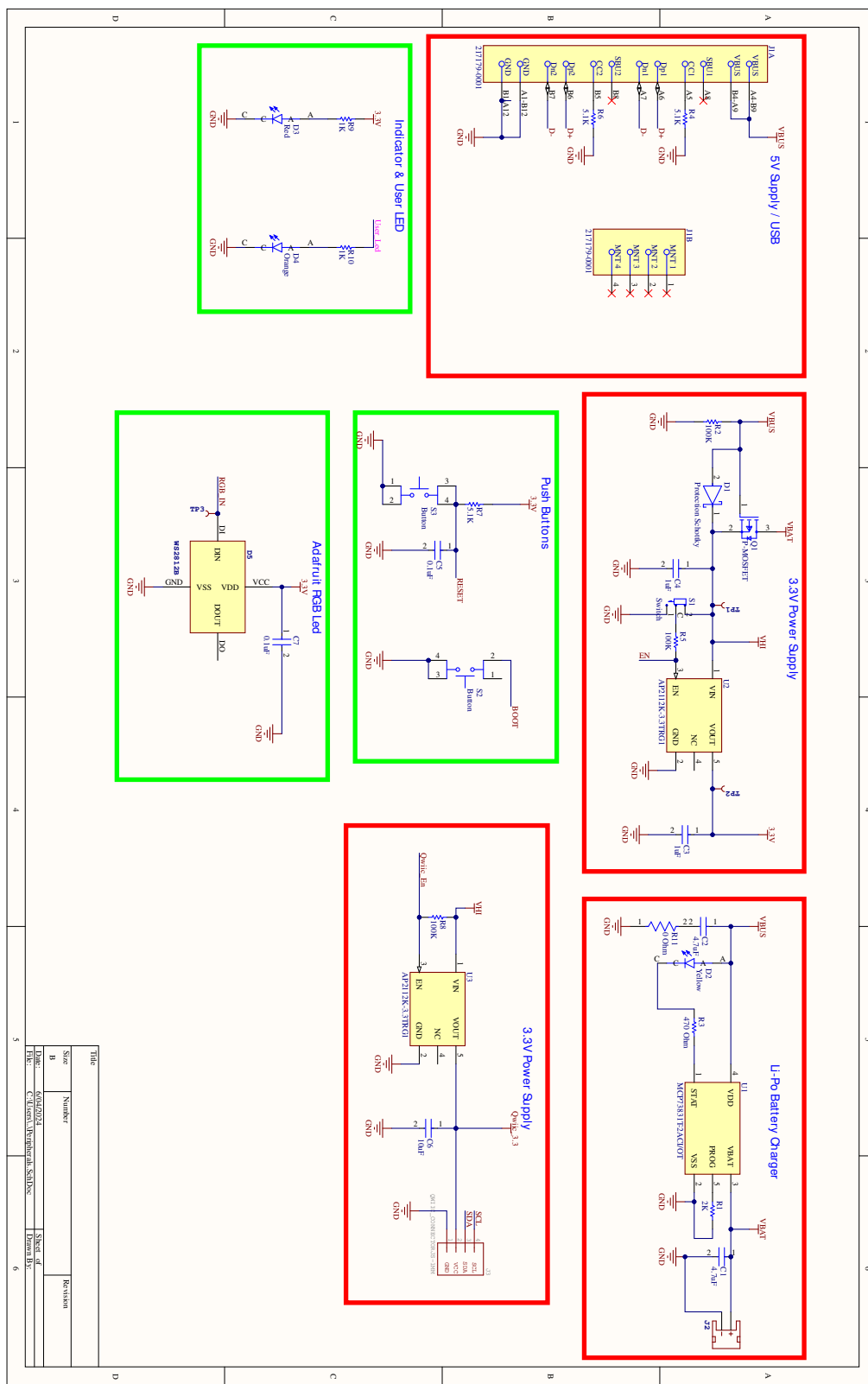
Bibliography

- [1] L. Ada, *Feather History*. <https://learn.adafruit.com/adafruit-feather/feather-history>, May 2017.
- [2] J. L. Aufranc, *Flip Click Board with Atmel SAM3X8E MCU*. cnx-software.com/2015/10/22/, Oct 2015.
- [3] Espressif Systems, *ESP32-S3 Series*, version 1.8 ed., 2023. Available at https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf.
- [4] *The World's fastest growing add-on Board Standard*. <https://www.mikroe.com/mikrobus>.
- [5] *Prototyping with I2C has never been easier*. <https://www.sparkfun.com/qwiic>.
- [6] K. S. Jacob Hylen, *Getting Started with Arduino IDE 2*. <https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/>, Jan 2024.
- [7] M. Yogendrappa, *How to Build a Multilayer PCB Stack-up*. <https://www.protoexpress.com/blog/build-multilayer-pcb-stack-up/>, Mar 2021.
- [8] E. Bogatin, *Bogatin's Practical Guide to Prototype Breadboard and PCB Design*. Artech House, 2021.
- [9] Z. Peterson, *4 Steps on How to Create a Component Footprint in Altium Designer*. <https://resources.altium.com/p/how-use-altium-designer-quickly-create-component-footprint>, Mar 2018.
- [10] *PCB Manufacturing Assembly Capabilities*. <https://jlcpcb.com/capabilities/pcb-capabilities>.
- [11] Digilent, *Digilent Pmod™ Interface Specification*, version 1.3.1 ed., Oct 2020. Available at <https://digilent.com/reference/pmod/start>.
- [12] G. Crowell, *What is a Breadboard?* <https://www.circuitbread.com/ee-faq/what-is-a-breadboard>, Apr 2019.

- [13] O. Brampton, *The History of Printed Circuit Board PCB 1880 – Present*. <https://how2electronics.com/history-printed-circuit-board-pcb-nextpcb/>, Aug 2023.
- [14] *Printed circuit board*. https://en.wikipedia.org/wiki/Printed_circuit_board, Apr 2024.
- [15] R. Khandpur, *Printed Circuit Boards: Design, Fabrication, and Assembly*. McGraw Hill professional, McGraw Hill LLC, 2005.
- [16] K. P. L.-G. K. Tord Johnson, Mattias Andersson, *Design For Excellence: Design and Process Correlations*. MTEK CONSULTING, 2011.
- [17] *Eagle Features*. <https://www.autodesk.com/products/eagle/features>.
- [18] *A Cross Platform and Open Source Electronics Design Automation Suite*. <https://www.kicad.org/>.
- [19] A. Rogers, *Introduction to USB Type-C*. Microchip Technology Inc, 2015. Available at <https://ww1.microchip.com/downloads/en/AppNotes/00001953A.pdf>.
- [20] Microchip Technology Inc, *Miniature Single-Cell, Fully Integrated Li-Ion, Li-Polymer Charge Management Controllers*, 2008. Available at <https://ww1.microchip.com>.
- [21] L. Ada, *Introducing Adafruit Feather*. <https://learn.adafruit.com/adafruit-feather/feather-specification>, May 2017.
- [22] MikroElektronika, *mikroBUS™ standard specifications*, version 2.00 ed., May 2015. Available at <https://download.mikroe.com/documents/standards/mikrobus>.
- [23] Z. Peterson, *Top 5 PCB Design Rules You Need to Know*. <https://resources.altium.com/p/pcb-layout-guidelines>, Feb 2017.
- [24] I. NAGY, *Complex Digital Hardware Design*. CRC Press, 2024.

Appendices

Appendix A: Schematic of the circuit



Appendix B: Schematic of the MCU and headers

