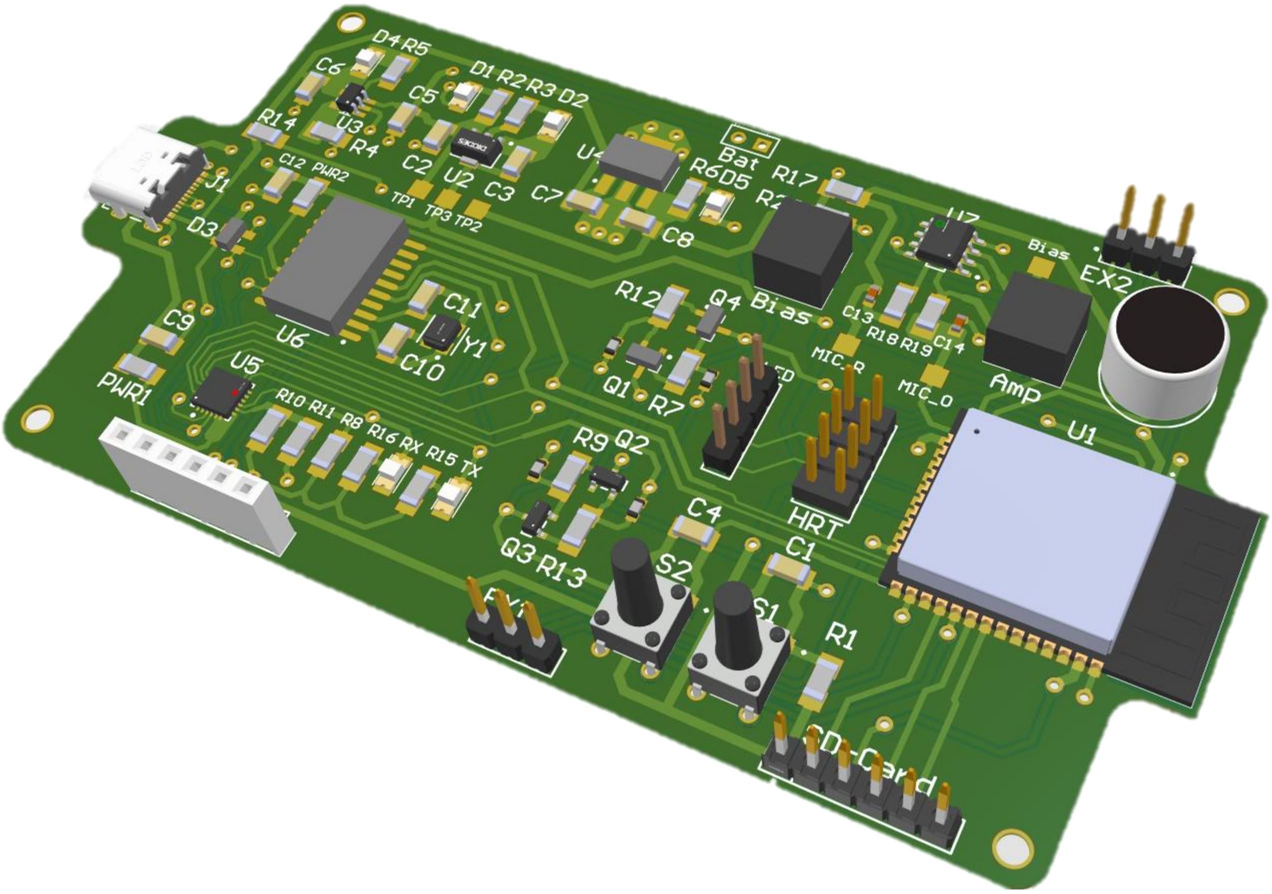
24 FEBRUARI 2023



ESP32 BREAKOUT BOARD

Personal Project Integration

DAAN HEETKAMP
SAXION UNIVERSITY OF APPLIED SCIENCES
M.H. Tromplaan 28, 7513 AB Enschede

Table of contents

Figures

Attachments

1. Introduction

This project is based around the ESP32 Wroom 32E. The assignment is as follows:

- Make a PCB in Altium using the ESP32.
- Let it use an output.
- Make a digital input.
- Make an analog input.

The ESP32 Wroom 32E is a small microcontroller that can connect to Wi-Fi and Bluetooth. This way it is ideal for building Internet of Things (IoT) applications. It's an Arduino/AT Mega that is also possible to connect to Wi-Fi and Bluetooth.

In this report the steps will be shown fast and clear.

2. ESP32 connections

The things that will be connected are:

- Microphone
- OLED screen
- Heart rate sensor
- SD Card module
- Extra connector pins
- Two buttons for programming
- Two programmers with automatic programmer circuits.
- Different voltage regulators and a BMS.

The first flowchart can be seen in:

3. Microphone

First, I searched for the microphone specs.

The microphone needs a certain voltage and that needs to be created. From the datasheet: 1,5Vavg = 1,67 Vrms. iS = 0,5mA.

To create iS the formula: $\frac{VCC-Vmic}{iS} = \frac{5-1,5}{0,5m} = 7K\ ohm$ was used.

A value of 6k8 is used from the E12 table.

3.1. Low pass filter

For the Op-amp a LM358 is used. From the Texas Site I found it was a good one for the application. Humans can hear from 20Hz to 20kHz.

High pass: 20Hz. R = 10kOhm because it is an easy value for amplifications.

$$R = \frac{1}{2 * Pi * f * C} => 10K = \frac{1}{2 * Pi * 20 * C} = 795nF$$

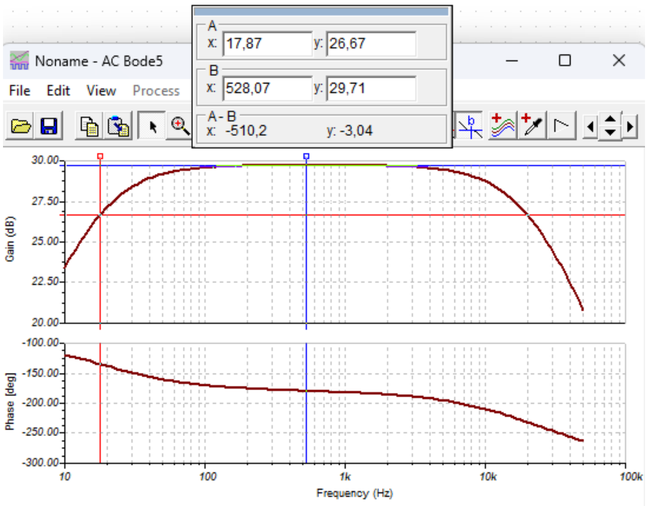


Figure 1 20 Hz filter

In figure 1 the low pass filter can be seen. At a frequency of 17,9Hz it is -3dB.

3.2. High pass filter

High pass: 20KHz. C = 4,7nF because it will get a R value in the kilos. Not too small or big.

$$R = \frac{1}{2 * Pi * f * C} => R = \frac{1}{2 * Pi * 20K * 4{,}7n} = 1{,}7kOhm$$

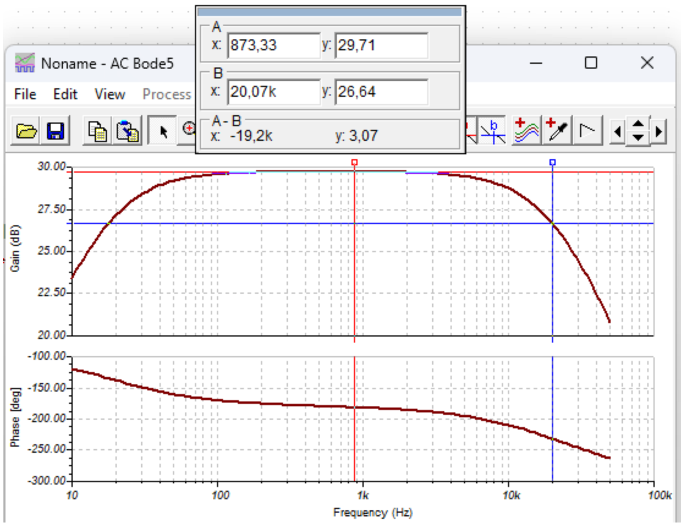


Figure 2 20kHz filter

In figure 2 the high pass pilter can be seen. At a frequency of 20,07kHz it is -3dB.

These two values are good enough.

3.3. Amplification

When simulating Vi is now 100mVPP. So from 0V to 100mVPP

The ADC from the ESP32 is from 0 to 3,3V. We amplify the 100mV 33 times.

Vin in the middle is 1,65V. Vbias should be 1,65V to make it go from -1,65V -> 1,65V to 0 -> 3.3V

This is done using a basic formula:

$$Vo = \frac{R2}{R2 + R1} * Vi = \frac{5k}{5k + 10k} * 5V = 1{,}66V$$

For the 10k a potentiometer is chosen. This is so finetuning is possible. This is also done for the 330k for the amplification. For that a 500k potentiometer is used.

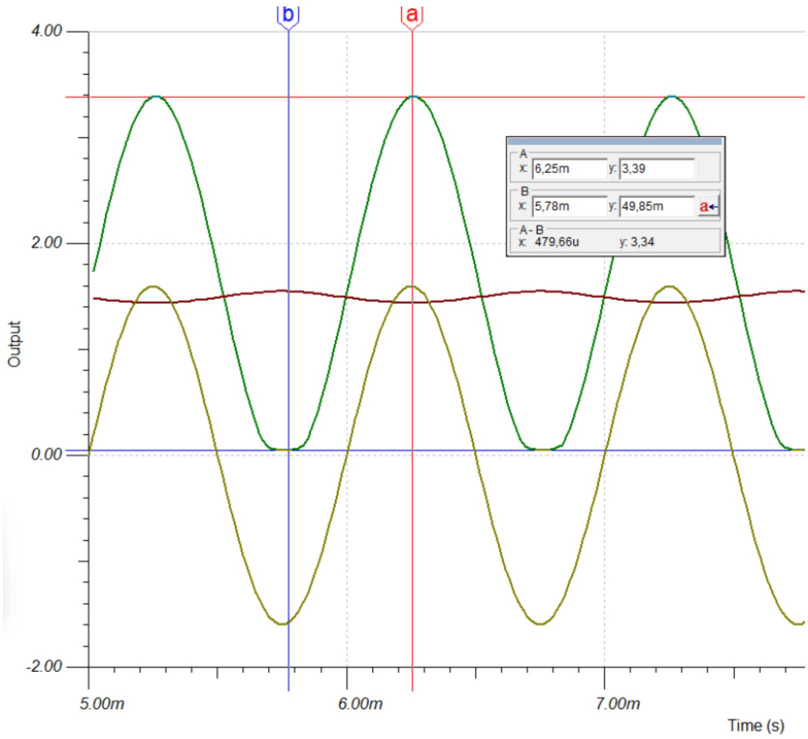
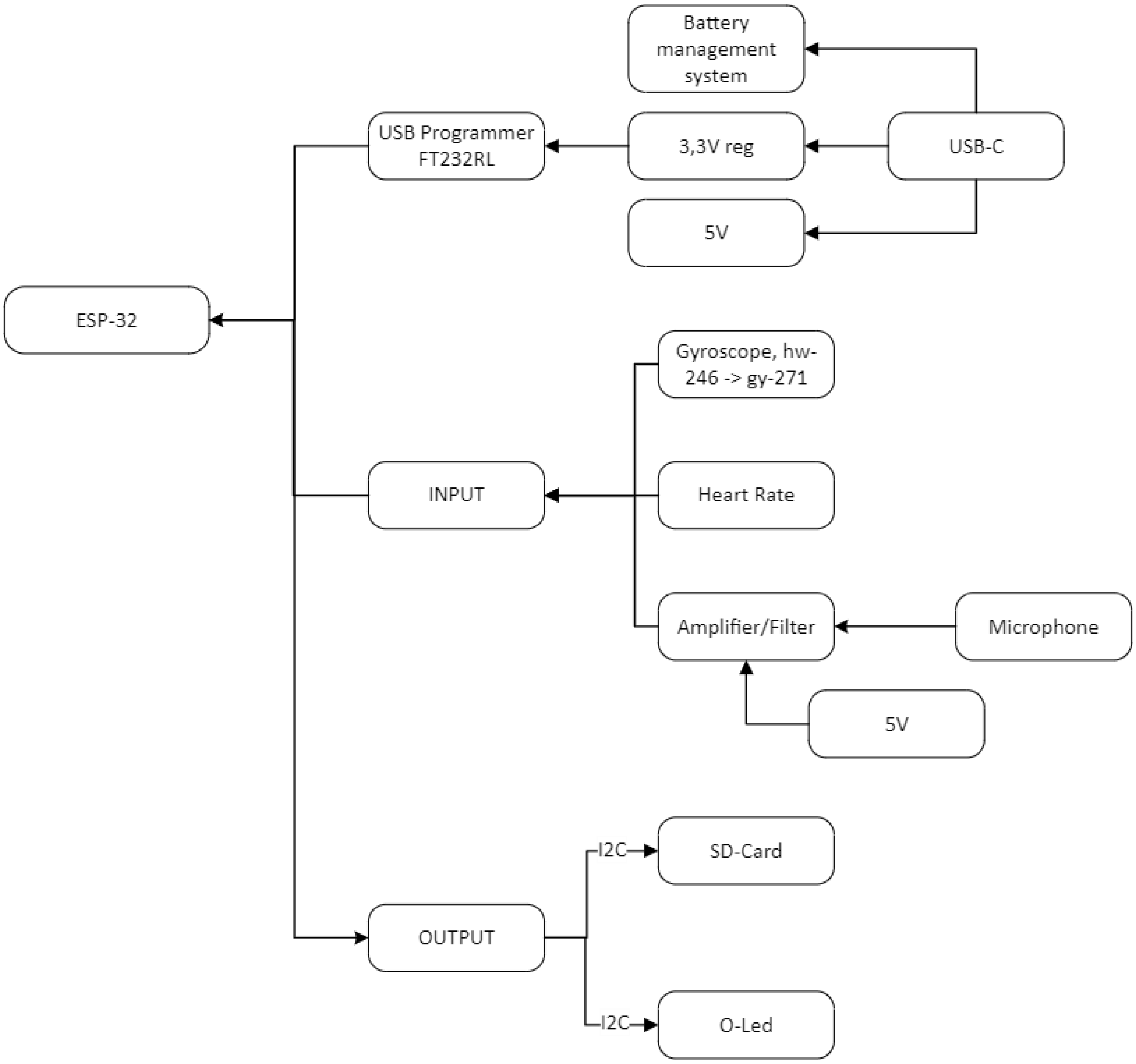


Figure 3 Microphone Output

Now all the components have been chosen. The circuit is shown in: *Attachment 1 Hardware Flowchart*



Attachment 2

4. Main ports

For the components looking at the datasheet and the ports was enough.

The following pins have been used:

- SDA -> IO21
- SCL -> IO22
- MISO -> IO19
- MOSI -> IO23
- CS -> IO17
- CLK -> IO18
- TX -> TXD0
- RX -> RXD0

All the components have been connected like they should.

5. Programming

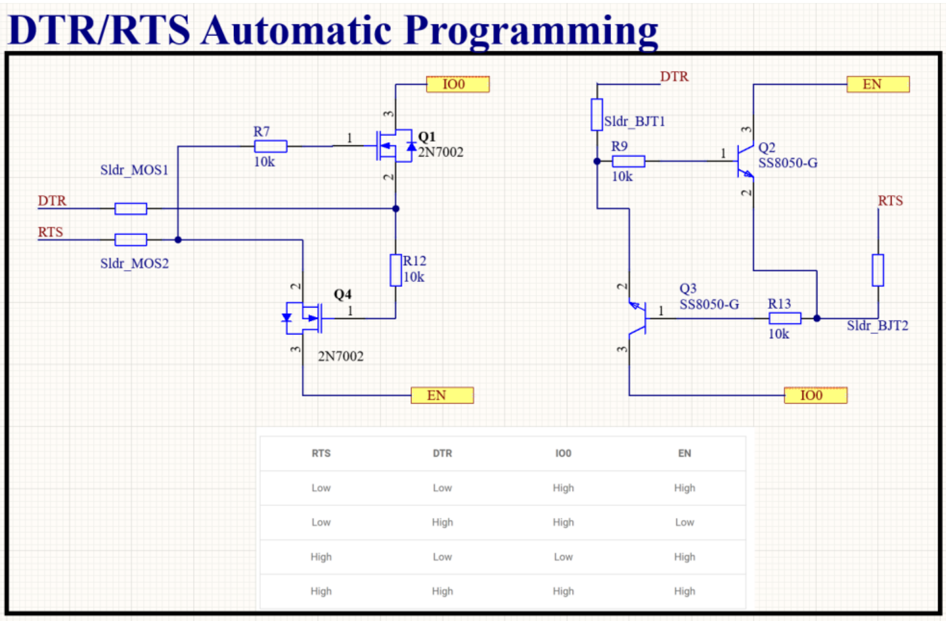
For the USB to UART two programmers have been used.

The CP2104 because it has built in DTR and RTS ports. This is for automatically setting the ESP32 in boot loading mode.

The MCP2200 also is used because this has been done before and the soldering is a little bit easier. I wanted to use two so that if one does not work maybe the other one will work.

Using soldering pads, the two will be split so not both will start programming.

For the CP2104 also a BJT and mosfet circuit have been created. This is because some people say the one wont work and the other will. Hopefully one of the two does work.



RTS	DTR	IO0	EN
Low	Low	High	High
Low	High	High	Low
High	Low	Low	High
High	High	High	High

Figure 4 DTR/RTS auto programmer

For the MCP2200 buttons have been added to set the ESP32 in boot mode manually.

The programmers have been connected using the datasheet data.

CTS and RTS for flow control have not been connected. This is because a baud rate of 9600 will do fine without it.

There is also an external programmer. This is just some headers where it can be connected.

6. Voltage regulators and BMS

The ESP32 works on a voltage of 3,3V.

The battery is 3.7V this is why the following stack up is chosen:

VBUS -> BMS and 5V. BMS -> 3.3V. The BMS charges the battery. The 5V supplies the microphone. The battery also goes through a 3.3V regulator.

Without the USB C connected the microphone circuit will not work. There was a possibility to use a boost converter, but it is chosen to not do this. Maybe in a future design.

7. PCB

The schematics have been broken up into three parts.

1. IC part
2. IC programmer part
3. Microphone part

This can be seen in:

The PCB is designed so the flow goes from left to right. See:

Also, the power regulators and signal components have been placed at different places of the PCB.

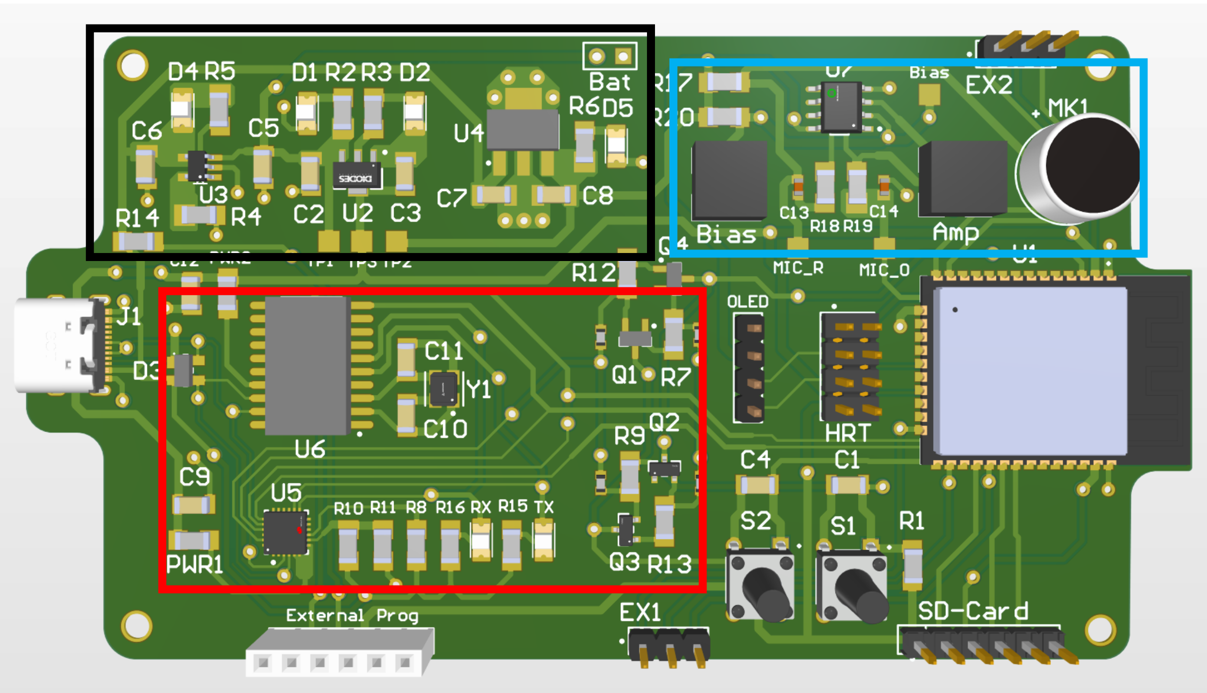


Figure 5 PCB layout

Black is the power management; red is the programmer place and blue the microphone.

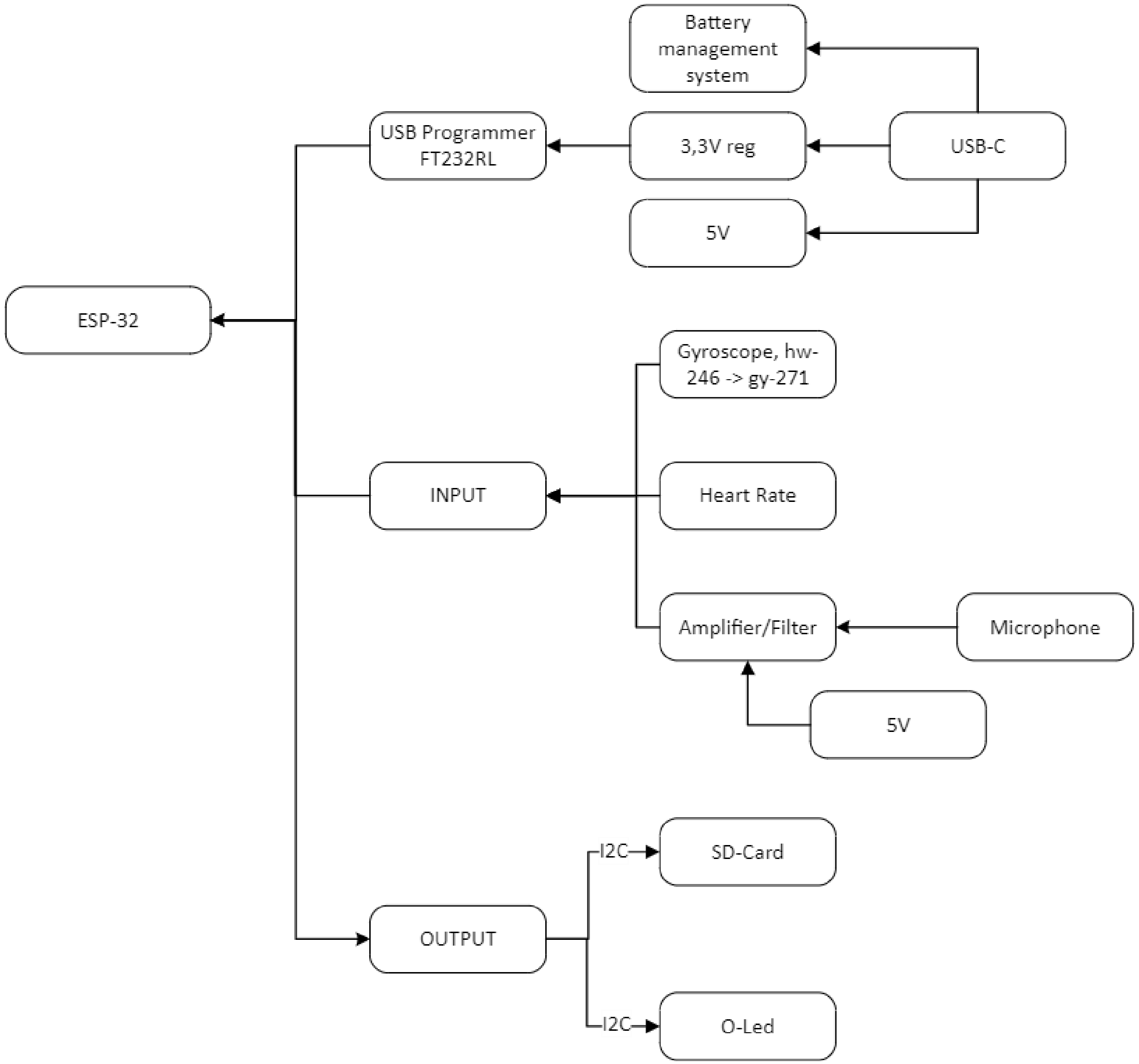
Sources

chatGPT [Online] / auth. OpenAI // openia.com. - https://chat.openai.com/.

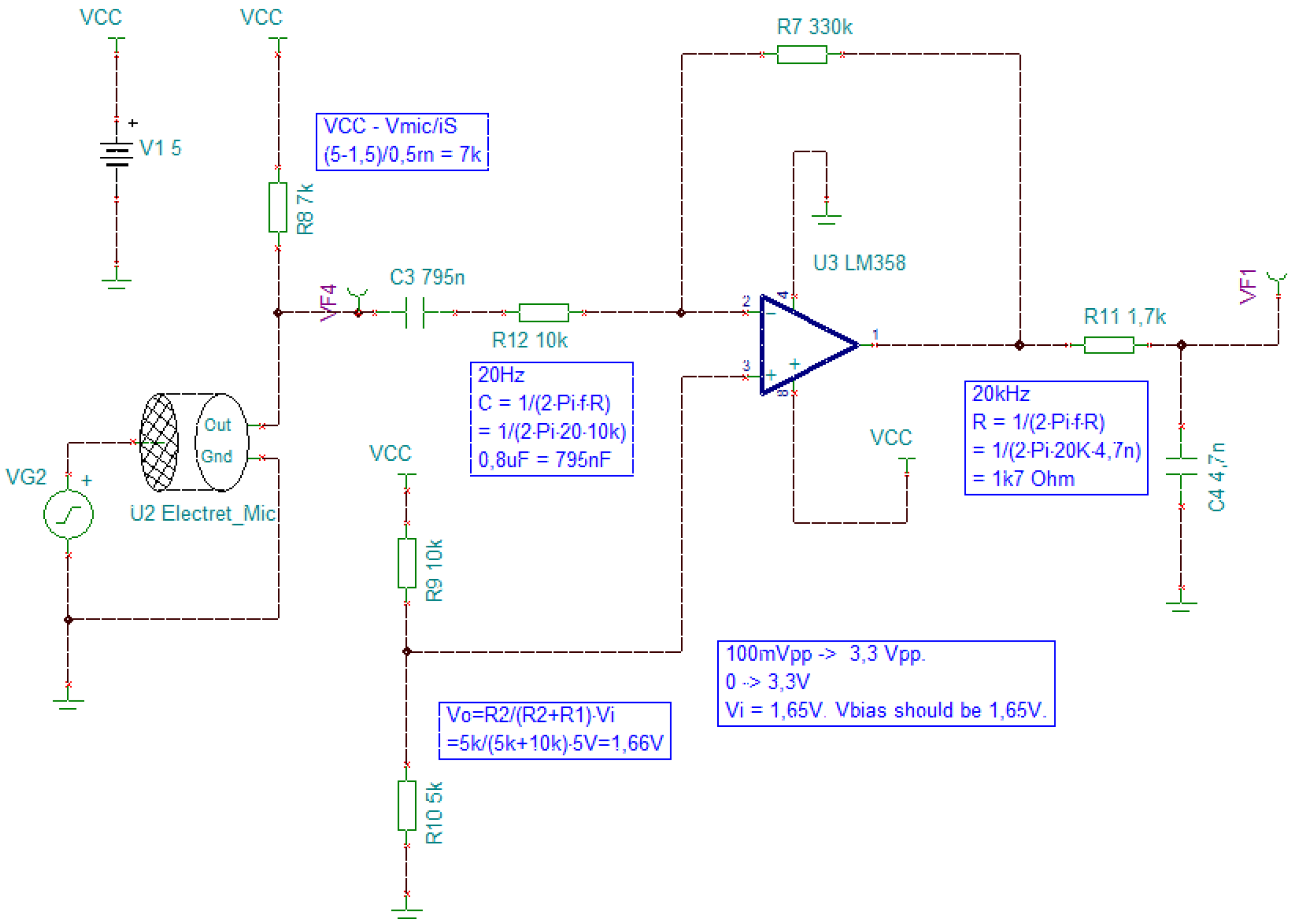
esp32-pinout-reference [Online] / auth. S Sara // randomnerdtutorials.com. - https://randomnerdtutorials.com/esp32-pinout-reference-gpios/.

esp32-wroom-32e datasheet [Online] / auth. Expressif // expressif.com. - https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf.

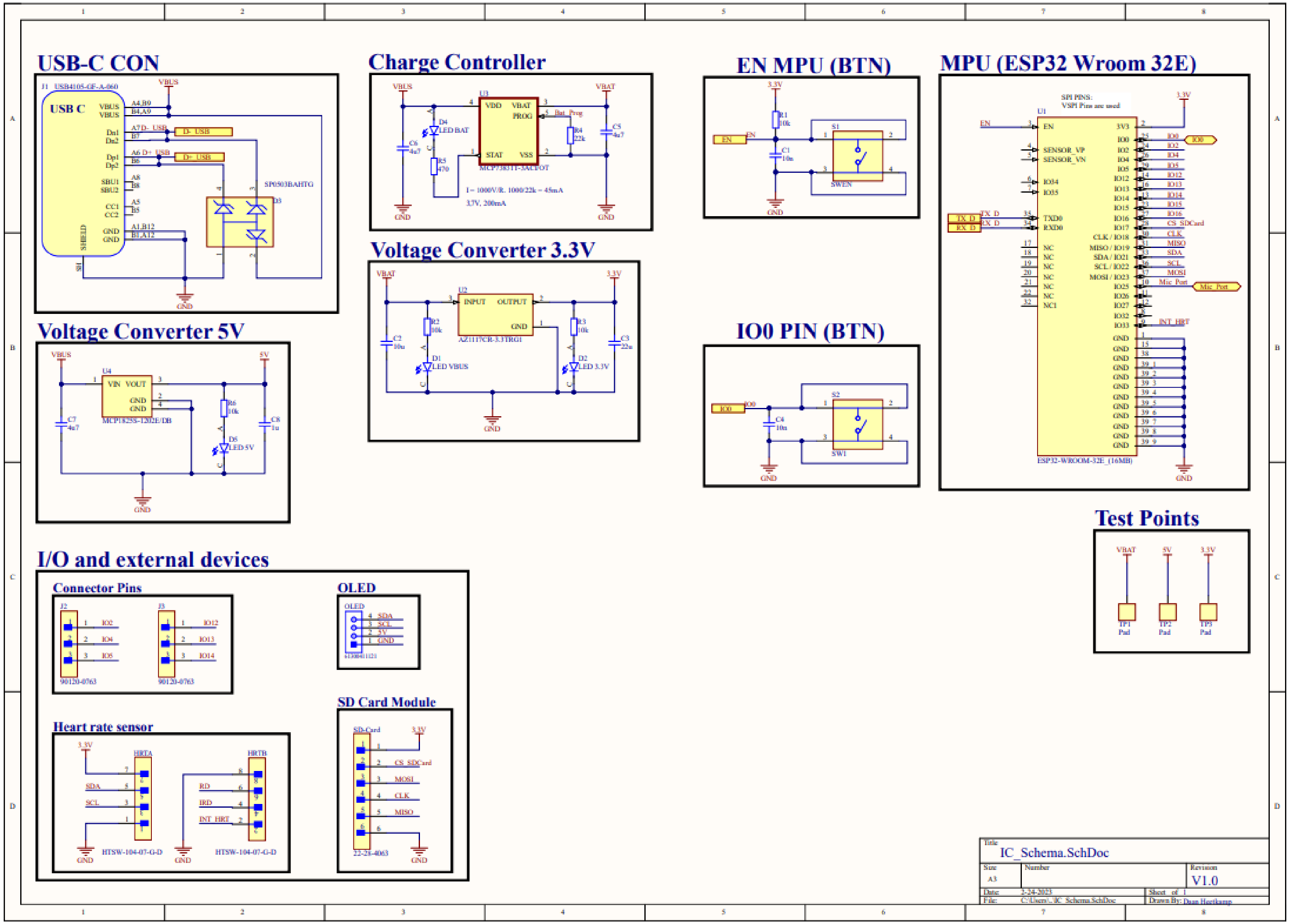
Attachment 1 Hardware Flowchart

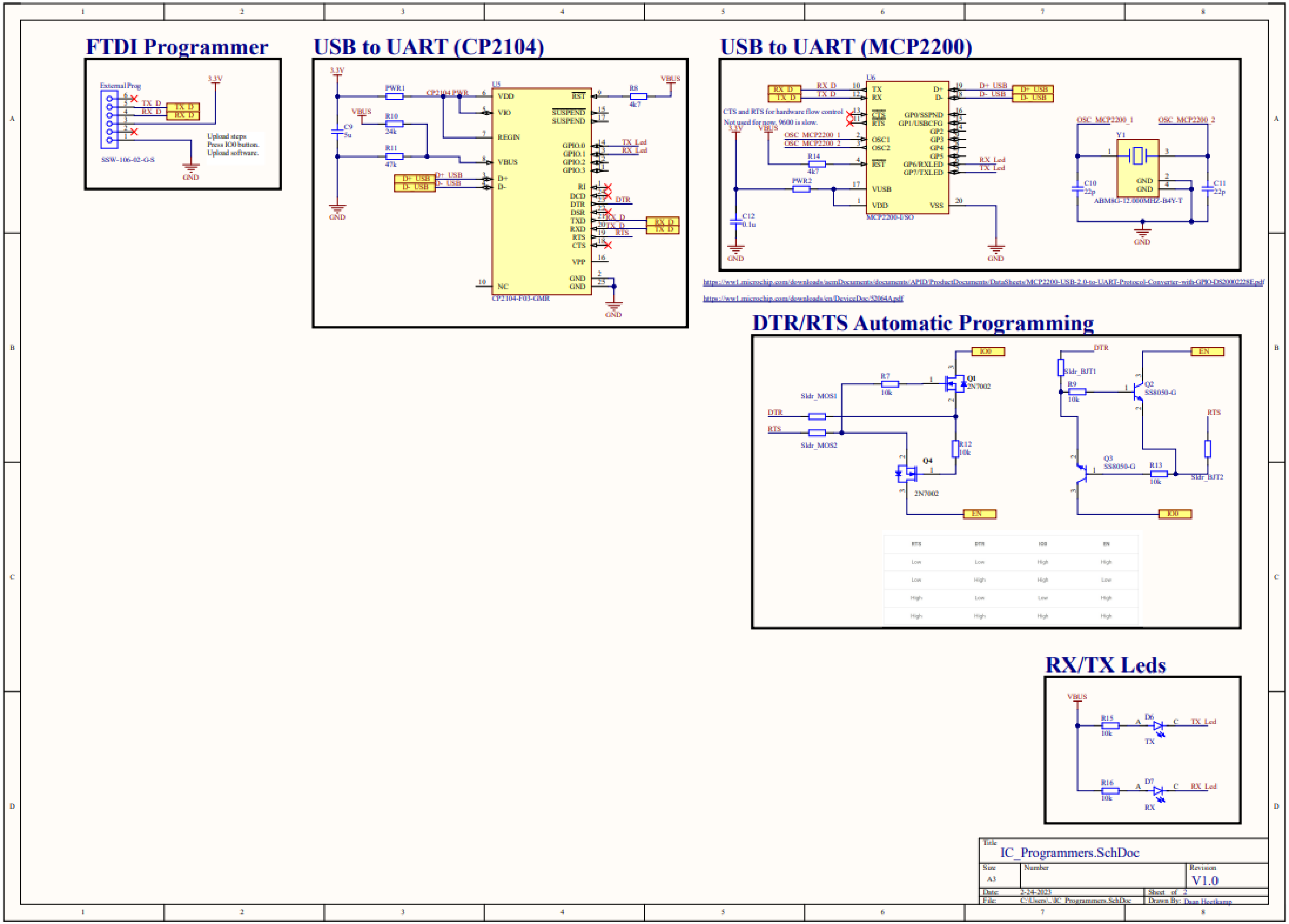


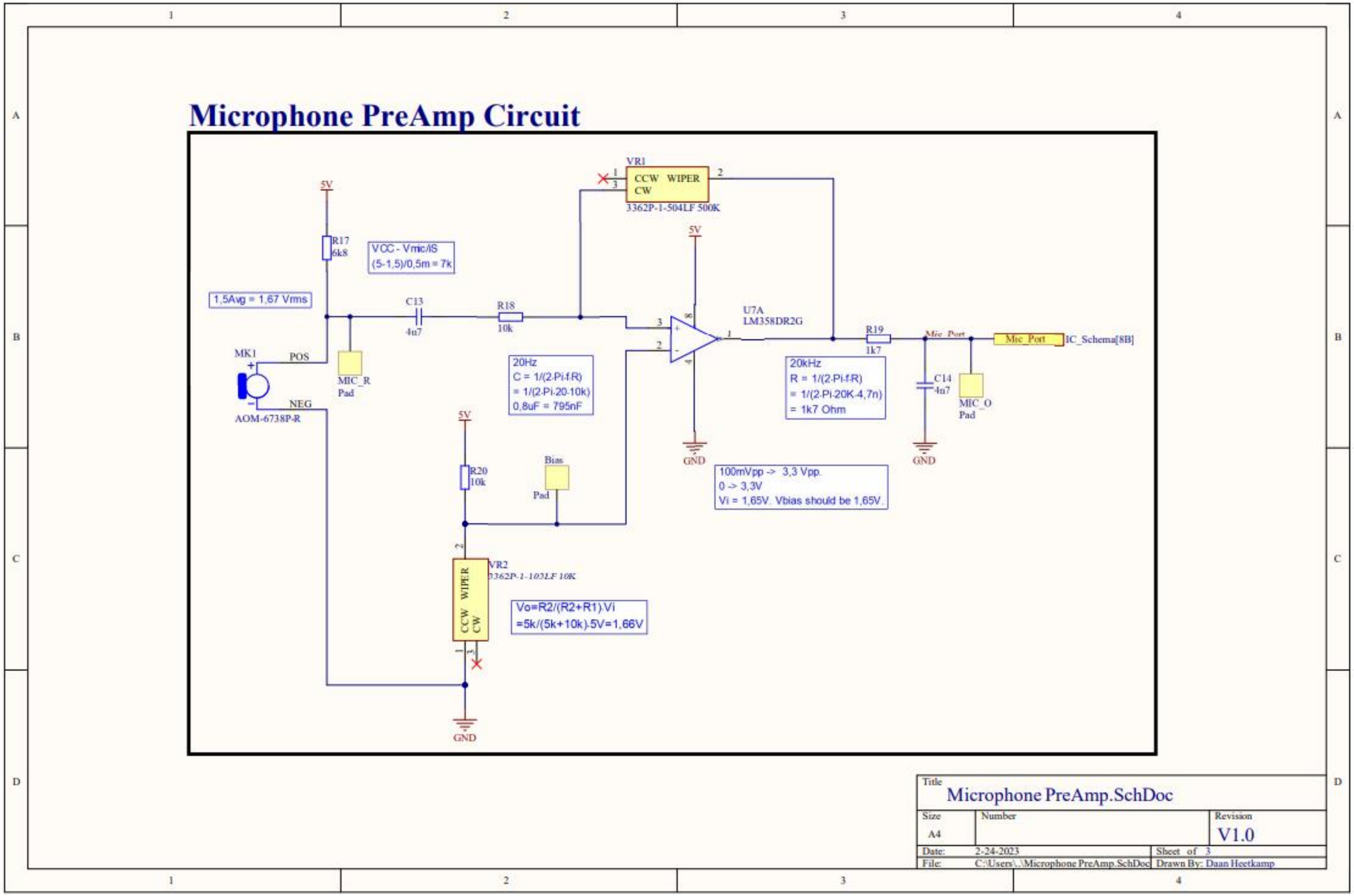
Attachment 2 Microphone Circuit - TINAspice



Attachment 3 ESP32 Breakout Schematics

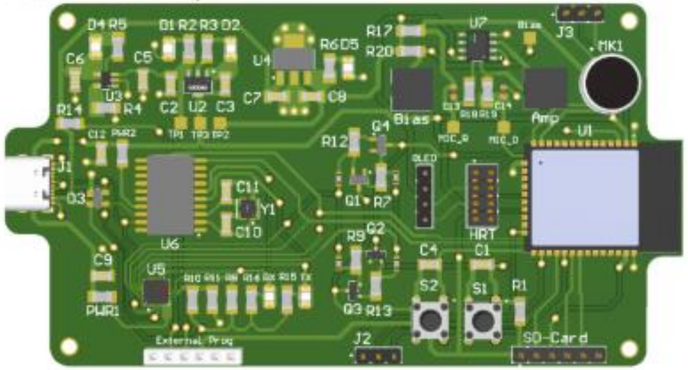



FTDI Programmer
USB to UART (CP2104)
USB to UART (MCP2200)
DTR/RTS Automatic Programming
RX/TX Leds
IC_Programmers.SchDoc
V1.0

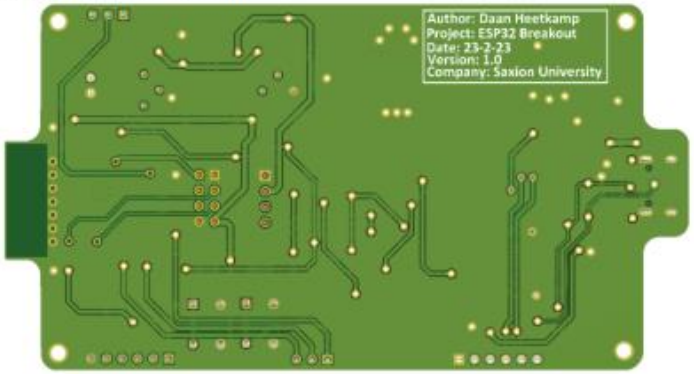

Microphone PreAmp Circuit
VR1
CCW WIPER
CW
3362P-1-504LF 500K
5V
R17
6k8
VCC - Vmic/iS
(5-1,5)/0,5m = 7k
1,5Avg = 1,67 Vrms
C13
4u7
R18
10k
U7A
LM358DR2G
R19
1k7
Mic Port
IC_Schema[8B]
MK1
POS
NEG
AOM-6738P-R
MIC_R
Pad
20Hz
C = 1/(2·Pi·f·R)
= 1/(2·Pi·20·10k)
0,8uF = 795nF
20kHz
R = 1/(2·Pi·f·R)
= 1/(2·Pi·20K·4,7n)
= 1k7 Ohm
C14
4n7
MIC_O
Pad
GND
R20
10k
Bias
Pad
100mVpp -> 3,3 Vpp
0 -> 3,3V
Vi = 1,65V. Vbias should be 1,65V
VR2
3362P-1-103LF 10K
Vo=R2/(R2+R1)·Vi
=5k/(5k+10k)·5V=1,66V
Title
Microphone PreAmp.SchDoc
Size
A4
Number
Revision
V1.0
Date: 2-24-2023
Sheet of 3
File: C:\Users\..\Microphone PreAmp.SchDoc
Drawn By: Daan Heetkamp

Attachment 4 ESP32 Break-out PCB

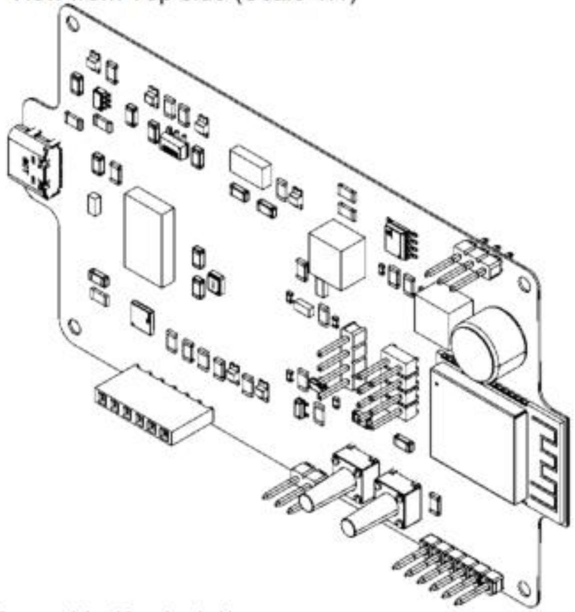
Realistic View Top



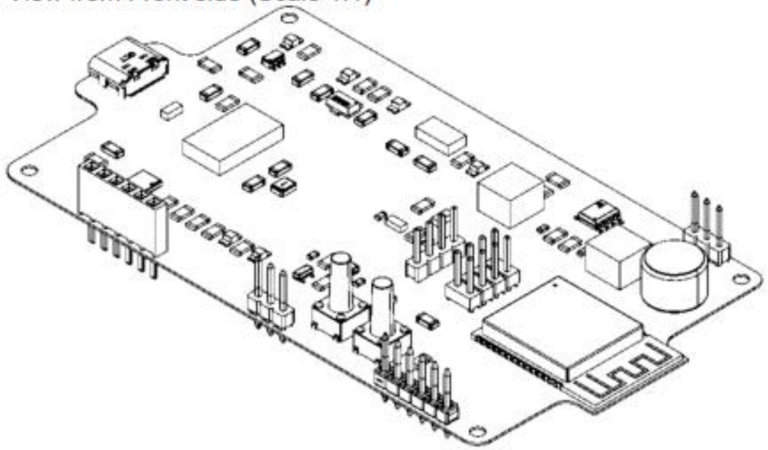
Realistic View Bottom



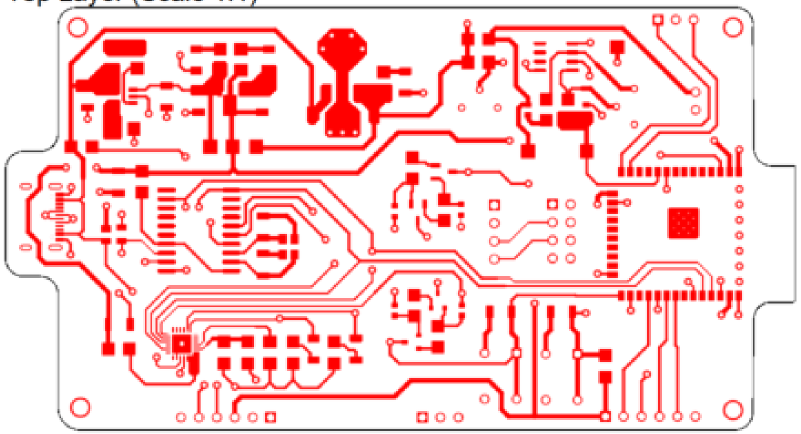
View from Top side (Scale 1:1)



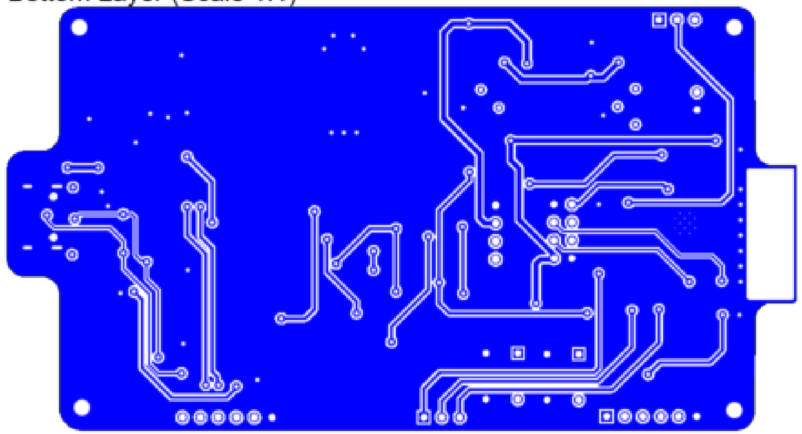
View from Front side (Scale 1:1)



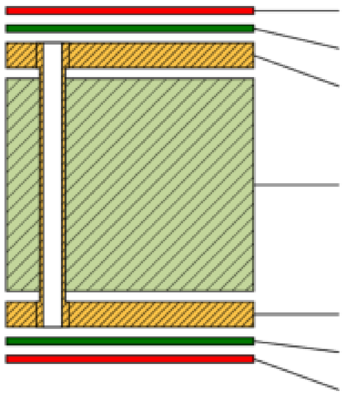
Top Layer (Scale 1:1)



Bottom Layer (Scale 1:1)



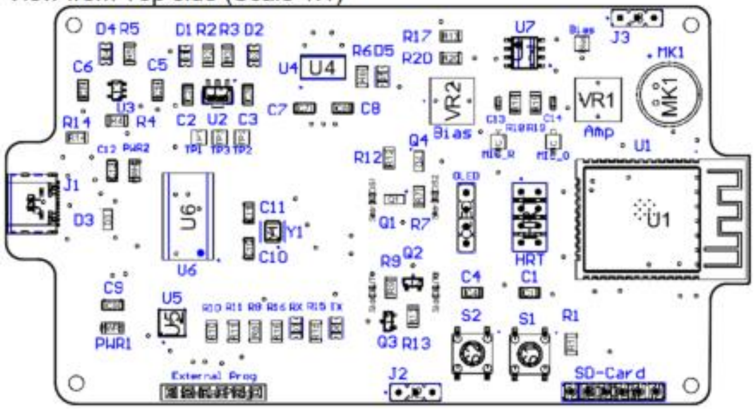
Layer Stack Legend



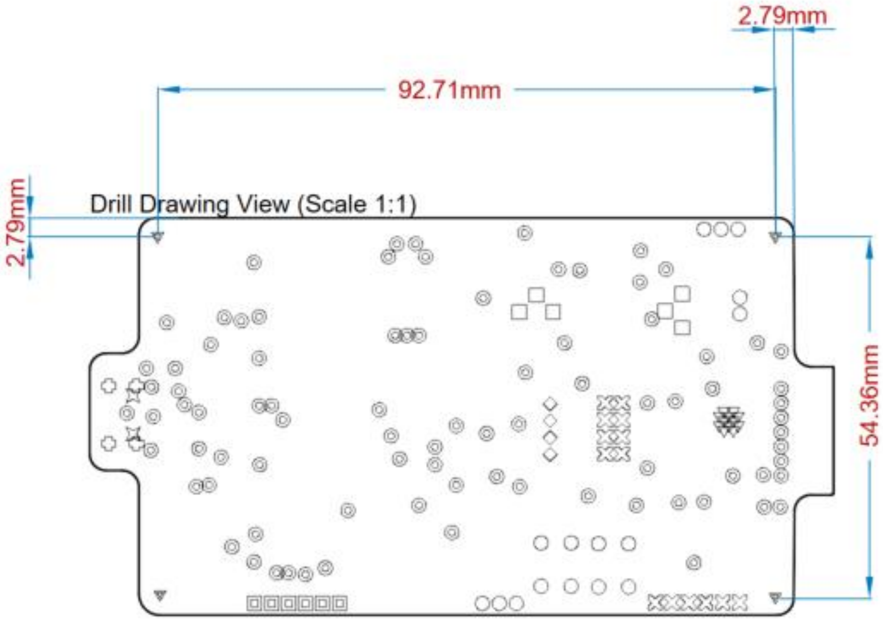
Material	Layer	Thickness	Dielectric Material	Type	Gerber
	Top Overlay			Legend	GTO
Surface Material	Top Solder	0.01mm	Solder Resist	Solder Mask	GTS
Copper	**Top Layer**	**0.04mm**		**Signal**	**GTL**
		0.32mm	*FR-4*	*Dielectric*	
Copper	**Bottom Layer**	**0.04mm**		**Signal**	**GBL**
Surface Material	Bottom Solder	0.01mm	Solder Resist	Solder Mask	GBS
	Bottom Overlay			Legend	GBO

Total thickness: 0.41mm

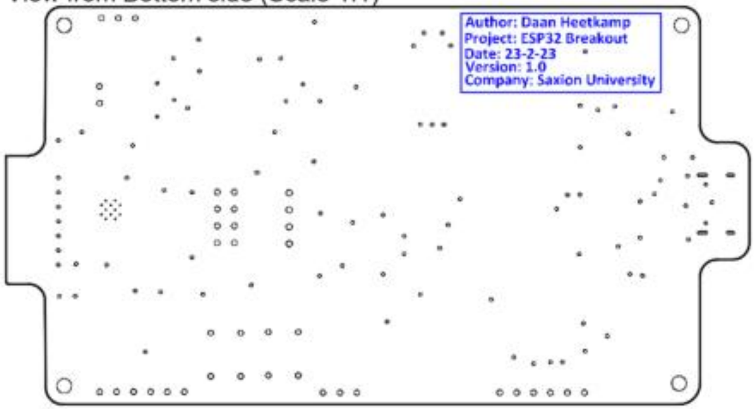
View from Top side (Scale 1:1)



Drill Drawing View (Scale 1:1)



View from Bottom side (Scale 1:1)



SAXION HOGESCHOOL

Assembly