

DESIGN (E) 314
TECHNICAL REPORT

Basic Audio Recording Device


Author:
Noah FOROMA

Student Number:
20687184

June 1, 2020

Plagiaatverklaring / Plagiarism Declaration

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
Plagiarism is the use of ideas, material and other intellectual property of another's work and to present it as my own.
2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
I agree that plagiarism is a punishable offence because it constitutes theft.
3. Ek verstaan ook dat direkte vertalings plagiaat is.
I also understand that direct translations are plagiarism.
4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelike aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism.
5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.


Handtekening / Signature

N. FOROMA
Voorletters en van / Initials and surname

20687184
Studentenommer / Student number

01/06/2020
Datum / Date

Abstract

The report documents the design and development of a basic audio recording device. The hardware design, decisions made and justifications for the implementation of the hardware are also discussed. Furthermore, the software design and structure are elaborated on, as well as the test methodology and the obtained results. The report concludes with the declaration that the required functionality is satisfied, and suggestions for future design are included.

Contents

1	Introduction	6
2	System description	6
2.1	System block diagram	6
2.2	Interface description	7
2.2.1	Power supplies	7
2.2.2	STM32F446RE board.....	7
2.2.3	ADC and Microphone	8
2.2.4	DAC and Audio Output	8
2.2.5	GPIO	8
2.2.6	SPI and SD card	8
2.2.7	Serial communication (UART)	8
3	Hardware design and implementation	8
3.1	Power supply	8
3.1.1	5 V Power Supply	9
3.1.2	3.3 V Power Supply.....	9
3.2	UART communications	10
3.3	Push-buttons and LEDs	10
3.4	Audio output.....	11
3.5	Microphone	12
3.6	SD Card	13
4	Software design and implementation	14
4.1	Layout of project	14
4.2	Control Logic.....	15
4.3	Button bounce handling.....	16
4.4	Data flow and processing.....	17
4.5	SD card interfacing	19
4.6	Peripheral setup	20
4.6.1	UART.....	20
4.6.2	GPIO	20
4.6.3	ADC.....	20
4.6.4	DAC.....	20
4.6.5	Timers	20
4.6.6	SPI.....	21
5	Measurements and Results	21
5.1	Power Supplies	21
5.1.1	5 V Power Supply	21
5.1.2	3.3 V Power Supply.....	21
5.2	UART communications	22
5.3	Buttons	23

5.4	LEDs.....	23
5.5	Audio output.....	24
5.6	Microphone	26
5.7	SD Card.....	26
6	Conclusions	27
6.1	Strengths.....	27
6.2	Weaknesses.....	27
6.3	Improvements and suggestions.....	27
7	Bibliography	28
8	Appendices	29
8.1	Appendix A - System schematics.....	29
8.2	Appendix B – STM32 module pinout table.....	32

List of Figures

1	System block diagram.....	7
2	5 V Power Supply Schematic.....	9
3	3.3 V Power Supply Schematic.....	10
4	Push-buttons and LEDs.....	11
5	Audio output schematic.....	12
6	Waveshare Sound Sensor Module Schematic	13
7	Flow diagram of system.....	15
8	State Machine Diagram	16
9	Flowchart for button bounce handling.....	17
10	File creation on SD card via SPI	18
11	Data flow and processing.....	19
12	Terminal program output.....	23
13	LED test result for record condition.....	24
14	Audio output for 440 Hz sine wave.....	25
15	Audio output for 523 Hz sine wave.....	25
16	Audio output for summed sine waves at 440 Hz and 523 Hz	25
17	Partial schematic of system.....	29
18	Microphone module schematic.....	30
19	SD card module schematic.....	31

List of Tables

20	Table 1: System components.....	6
21	Table 2: sd.c file.....	19
22	Table 3: 5 V power supply results	21
23	Table 4: 3.3 V power supply results.....	22
24	Table 5: STM32 module pinout.....	32

List of Abbreviations

ADC Analog to Digital Converter

DAC Digital to Audio Converter
DC Direct Current
DMA Direct Memory Access
HAL Hardware Abstraction Layer
FATFs File Allocation Table file system
GPIO General Purpose Input / Output
IC Integrated Circuit
LED Light Emitting Diode
MCU Microcontroller Unit
MISO Master In Slave Out
MOSI Master Out Slave In
PCB Printed Circuit Board
SCK Clock
SPI Serial Peripheral Interface
UART Universal Asynchronous Receiver / Transmitter
USB Universal Serial Bus

1 Introduction

The project required the creation of a basic audio recording device, which would be powered by a rechargeable 9 V battery, and would be ergonomic, aesthetically pleasing, and compact. In addition to this, it was required that the user interface be simple, that the device be able to record audio to an SD card, and that the recorded audio be played back through an audio socket. Furthermore, the device was to make use of an STM32 microcontroller, and have push buttons, LEDs, and volume control incorporated into the system [1].

The developed audio recorder makes use of an STM32F446RE MCU, which is responsible for interfacing with the various components of the device. The device is powered by an external 9 V supply, which is regulated down to 5 V and further to 3.3 V to power the various components of the device which have limited operating voltage ranges. The audio input is captured via a Waveshare microphone module, and is processed by the MCU and stored in an SD card. The following sections provide a detailed discussion on the design of the system.

2 System description

The table below lists the components used in the system and their operating voltages.

Table 1: System components

Component	Operating Voltage
STM32 module	2.0 V – 5.5 V
MCP1700 regulator	2.3 V – 6.0 V
Waveshare Sound Sensor	3.3 V – 5.3 V
Catalex MicroSD Card Adapter	4.5 V – 5.5 V
MCP602 Operational Amplifier	2.7 V – 6.0 V
LM7805 regulator	7 V – 35 V

2.1 System block diagram

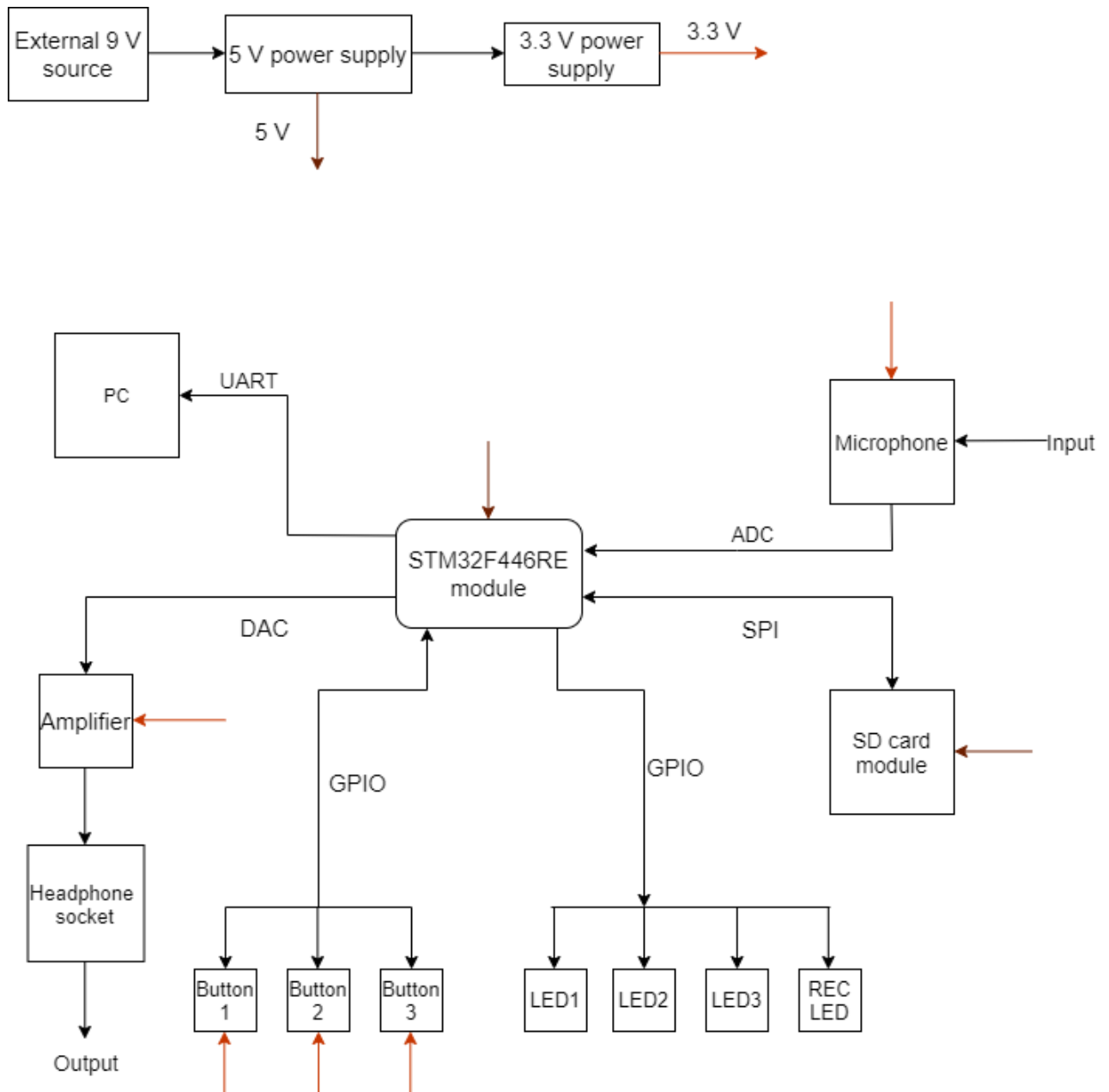


Figure 1: System block diagram

2.2 Interface description

2.2.1 Power supplies

The project PCB was powered by an external 9 V power supply, as shown in Figure 1. Regulation of the external power supply down to a 5 V supply was necessary, and this was further regulated down to a 3.3 V supply. The 5 V and 3.3 V supplies were then used to power various components of the board.

2.2.2 STM32F446RE board

The STM32F446RE Nucleo module was the primary component of the system, and was responsible for interfacing between the various components incorporated into the system. The module makes use of an ARM 32-bit Cortex M4 CPU, and has 512 kB of Flash memory, as well as 128 KB of SRAM [2]. In addition to this, the module also includes various peripheral interfaces (such as ADC, DAC, GPIO, SPI,

UART) that were used in the implementation of the system. The module was programmed to communicate with the various components of the board, and debugging of the system's communication was done via the module.

2.2.3 ADC and Microphone

The ADC was used to sample the audio signal from the microphone. The ADC converts a sample voltage to a value in the range 0 – 4095, corresponding to the range of allowable voltages (0 – 3.3 V) [2]. The microphone module ensures that the previously mentioned condition is satisfied through the use of an amplifier to amplify the signal output by the microphone.

2.2.4 DAC and Audio Output

An audio sub-circuit, consisting of an amplifier in a voltage follower configuration and a headphone socket, was used to output the stored audio file. This was achieved by connecting the sub-circuit to the DAC of the Nucleo module, which converted the stored audio file into an analog equivalent.

2.2.5 GPIO

To process inputs from the push-buttons, and to power the LEDs based on the conditions stipulated in [1], pins of the Nucleo module were configured in either output mode for the latter, or in external interrupt mode for the former.

2.2.6 SPI and SD card

To facilitate the storage of recorded audio signals for later playback, an external SD card was incorporated into the system. Communication between the Nucleo module and the SD card occurred via SPI. This allowed for the writing of files that contained audio data to the SD card, and for the reading of these files later by the Nucleo module. The file system used to achieve this was the FATFs.

2.2.7 Serial communication (UART)

The default UART2 channel of the module was used to communicate with the PC. This channel is linked to the ST-Link of the Nucleo module and enumerates as a virtual COM port on the PC [1]. The UART channel was used to send status messages and stream audio data to the PC, as well as to facilitate debugging. The UART communication was specified as 8 bit asynchronous, with no parity bit, and 1 start and stop bit. The bitrate was also specified as 500000 bits/sec [1].

3 Hardware design and implementation

A detailed discussion on the decisions, calculations and justifications for each individual hardware component that was incorporated into the design of the audio recorder device is provided in this section.

3.1 Power supply

As stipulated in [1] under Requirement 1, the system was to be powered by a nominal 9 V unregulated supply (in the range 8 V – 12 V). This unregulated supply would then be regulated down to stable 5 V and 3.3 V supplies. These supplies would then be used to power various components on the project PCB.

3.1.1 5 V Power Supply

The 5 V power supply was predesigned, and the schematic shown below was taken from [1].

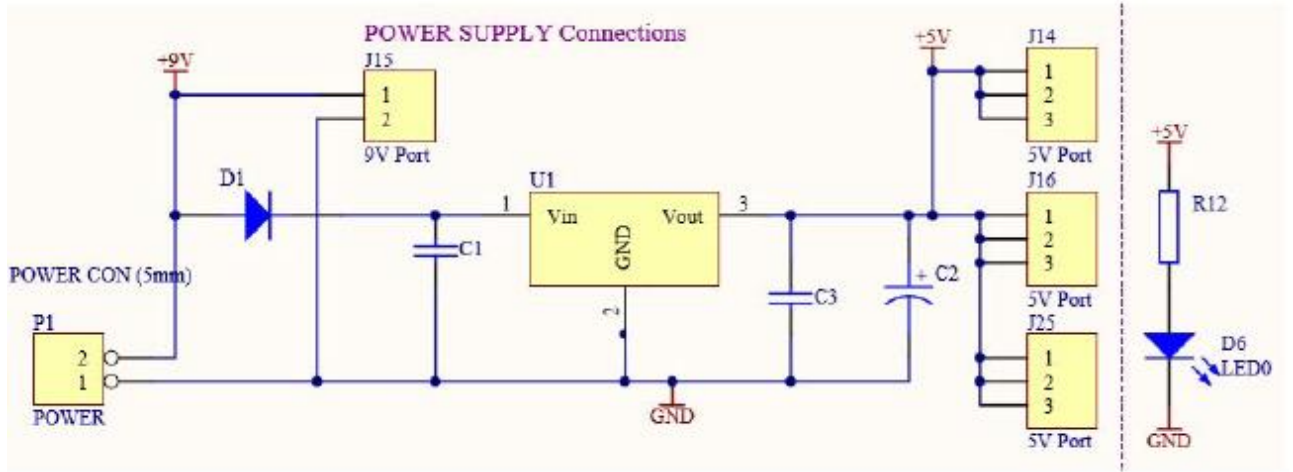


Figure 2: 5 V Power Supply Schematic

The maximum current that the system will draw from a 5 V source according to [1] is 300 mA. The 5 V power supply makes use of an LM7805 regulator in a TO-220 (U1) to regulate the 9 V input down to a stable 5 V output. This regulator satisfies the requirement for the current drawn as it can deliver a maximum of 1 A [3]. The diode D1 was incorporated into the supply to prevent damage to the components in the event that the input voltage polarity was reversed. For this, a 1N4007 diode was used, as it has a low forward voltage drop of 1 V for $I_F = 1$ A [4]. The input bypass capacitor C1 was included for stable operation under all load conditions, and in line with the recommendation in [3], was chosen as 0.1 μ F. The capacitors C2 and C3 improve stability and transient response of the regulator, and were chosen as an electrolytic 10 μ F capacitor and a ceramic 0.1 μ F capacitor respectively, as suggested in [3]. Also, to signify presence of a voltage on the 5 V line, the LED D6 was added, with a series resistor of size 1 k Ω to limit the current draw to 5 mA. From [3], the dropout voltage is 2 V, hence for correct operation, the minimum input voltage should be:

$$V_{in} = V_{out} + V_{dropout} = 7 \text{ V}$$

From the above result, it can be concluded that the unregulated input voltage should prove sufficient for the LM7805 regulator.

3.1.2 3.3 V Power Supply

A schematic of the 3.3 V power supply is provided below.

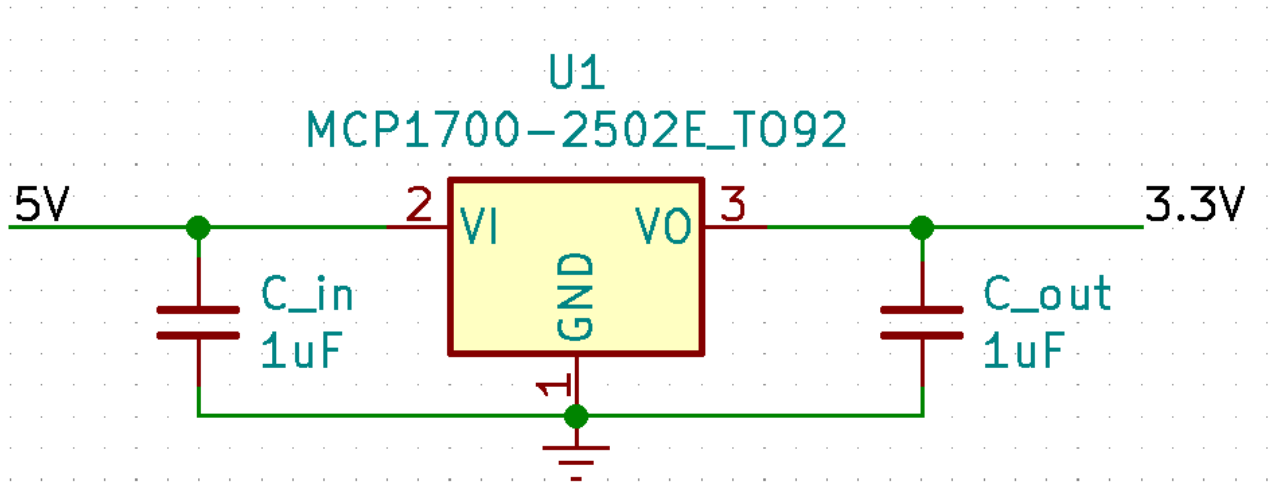


Figure 3: 3.3 V Power Supply Schematic

The 3.3 V supply makes use of an MCP1700 regulator in a TO-92 package to regulate the stable 5 V supply down to 3.3 V. This regulator was chosen since the 3.3 V power supply was required to deliver a maximum output current of 20 mA [1], and the regulator is capable of outputting a current of 250 mA [5]. To ensure circuit stability, the input capacitor C_{in} was included, and as recommended in [5], was chosen as a 1 μ F ceramic capacitor. The output capacitor was also chosen as the recommended value in [5], as it was required for small signal stability. From [5], the typical dropout voltage is 178 mV for $V_{IN} > 2.5$ V, hence the 5 V input is sufficient for correct operation of the 3.3 V regulator.

To determine whether a heatsink would be required for the regulator, the following calculation was carried out:

From the datasheet [5]:

$$T_{J(MAX)} = P_{TOTAL} \times R\theta_{JA} + T_{A(MAX)}$$

$$T_{J(MAX)} = ((3.3 \times 250 \times 10^{-3}) \times 74) + 25 = 86.05 \text{ }^{\circ}\text{C}$$

The calculated junction temperature is less than the absolute maximum rated junction temperature, hence no heat sink would be required.

3.2 UART communications

For this project, no external UART module was incorporated into the design. Instead, the default UART2 channel of the STM32 Nucleo F446RE module was to be used. Since it enumerates as a virtual COM port when connected via USB cable to a PC, no hardware connections were required [1]. However, for evaluation at a test station, the UART2 had to be routed to the TIC, and this was accomplished with a wire.

According to Requirement 16 in [1], the baud rate of the UART channel was to be 500000. Hence, for 1 bit to be transmitted over the UART, the period would be:

$$period = \frac{1}{baudrate} = \frac{1}{500000} = 2 \text{ } \mu\text{s}$$

3.3 Push-buttons and LEDs

A graphical representation, taken from [1], of the implementation of the push-buttons and LEDs is shown below.

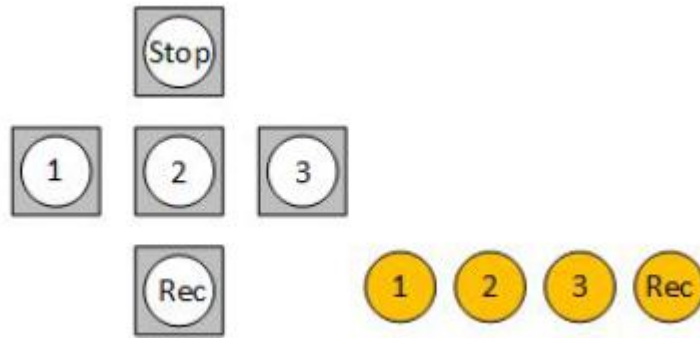


Figure 4: Push-buttons and LEDs

From [2], the maximum input voltage on FT and FTf pins is $V_{DD} + 4.0$ V, and the maximum current sunk/sourced by any I/O pins is ± 25 mA. The buttons were to be used as input to these pins, and so the input voltage was chosen to be 3.3 V (from the 3.3 V power supply designed earlier). Additionally, since the pins were chosen to be configured in pull-down mode, no additional resistors were connected to the push-buttons. This would still allow for the absolute maximum current rating to not be exceeded, since the resistance of a pull down resistor is 40 k Ω [2]. Hence for an input voltage of 3.3V, the input current would be:

$$I_{IN} = \frac{V_{IN}}{R_{pull-down}} = \frac{3.3}{40 \times 10^3} = 82.5 \mu A.$$

Suppression of mechanical bounce as stipulated in Requirement 13 [1] was to be carried out in software, and as a result no physical mechanism was implemented.

The LEDs used in this project were chosen to each consume a maximum of 2.5 mA of current, to give a total of 10 mA. From [2], the output voltage of an I/O pin is 3.3 V, hence the series resistor to be used with an LED had to have a resistance of:

$$R_{LED} = \frac{V}{I} = \frac{3.3}{2.5 \times 10^{-3}} = 1.32 k\Omega$$

As the closest standard resistor to the calculated value has a resistance of 1.5 k Ω , that was chosen to be used instead, resulting in a current of 2.2 mA. This resulted in a total current consumed of 8.8 mA, which is still within the range designed for.

3.4 Audio output

A schematic of the audio output sub-circuit is shown below:

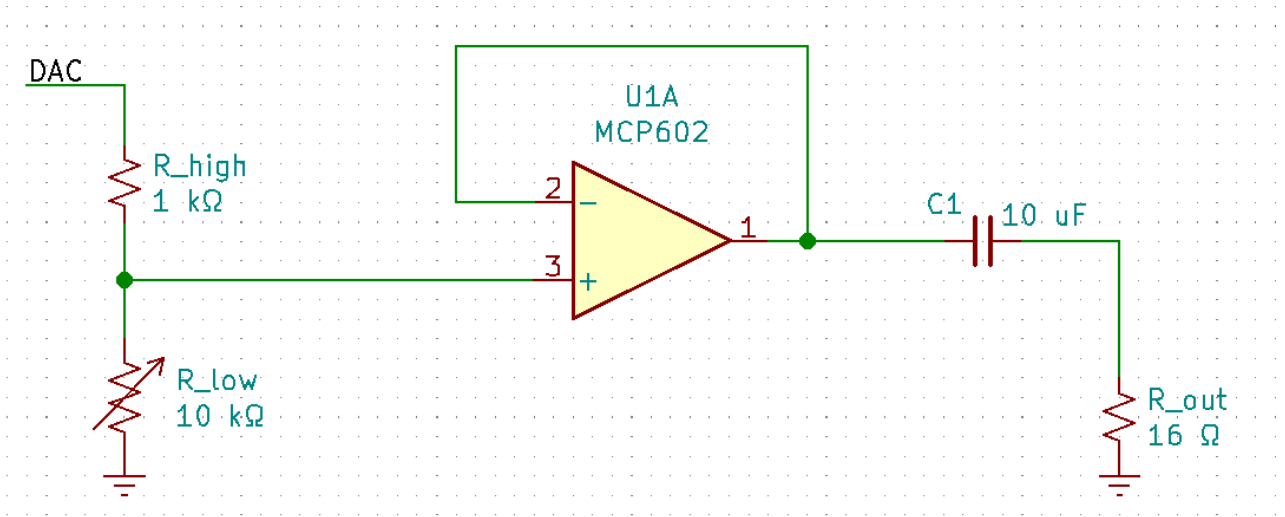


Figure 5: Audio output schematic

The above schematic was designed to satisfy Requirements 5 and 6 in [1], with an MCP602 operational amplifier which was provided. To power the operational amplifier, the 3.3V power supply designed earlier would be used. As it was required for the audio output to fall in the range ± 200 mV, it was necessary to connect an operational amplifier in the above configuration so that the output voltage of the DAC would not drop significantly when the load R_{out} was connected. This is because the effective resistance of the DAC + R_{load} when connecting the load directly to the DAC would be approximately equal to R_{out} . With this consideration in mind, the voltage divider at the input of the operational amplifier was designed as:

$$V_{in\ op-amp} = \frac{R_{low}}{R_{high} + R_{low}} \cdot V_{DAC}$$

The voltage ratio $\frac{V_{in\ op-amp}}{V_{DAC}}$ was chosen as ≈ 0.9 , so that the voltage divider output would closely resemble the DAC output, and with R_{high} chosen as 1 kΩ, R_{low} was then calculated as 9 kΩ. Since no such standard resistor exists, a 10 kΩ potentiometer was chosen instead, and this would be tuned until the required resistance was obtained.

C1 was included as a blocking capacitor, to filter out the DC component of the signal and center the voltage output around 0 V. As suggested by [6], the capacitor was chosen to have a capacitance of 10 μF, which was already provided with the project components. To hear the audio output, a 3.5 mm socket was included in the design that allows for the connection of headphones.

3.5 Microphone

The schematic of the microphone module, taken from [7] is shown below:

The SD card module used has two ICs on its breakout board, a 3.3 V regulator, and a quad buffer (which converts a 5 V signal for the MOSI, CS, and SCK lines to a 3.3 V signal for the SD card). From [10], the input voltage for the module lies in the range 4.5 – 5 V, which is regulated down by the 3.3 V IC, hence the module would be powered by the 5 V supply designed earlier. Communication between the SD card and the STM32 microcontroller was to occur via SPI. The SD card would then be mounted onto the module, and removed if necessary, thus allowing for multiple SD cards to be used. The SD card used in this implementation was a SanDisk 16GB SDHC card, which conforms to the specifications specified in [10].

4 Software design and implementation

The software development tool used for this project is STM32CubeIDE. This section provides a discussion on the software design and implementation.

4.1 Layout of project

The flow diagram of the system is shown in the Figure 6:

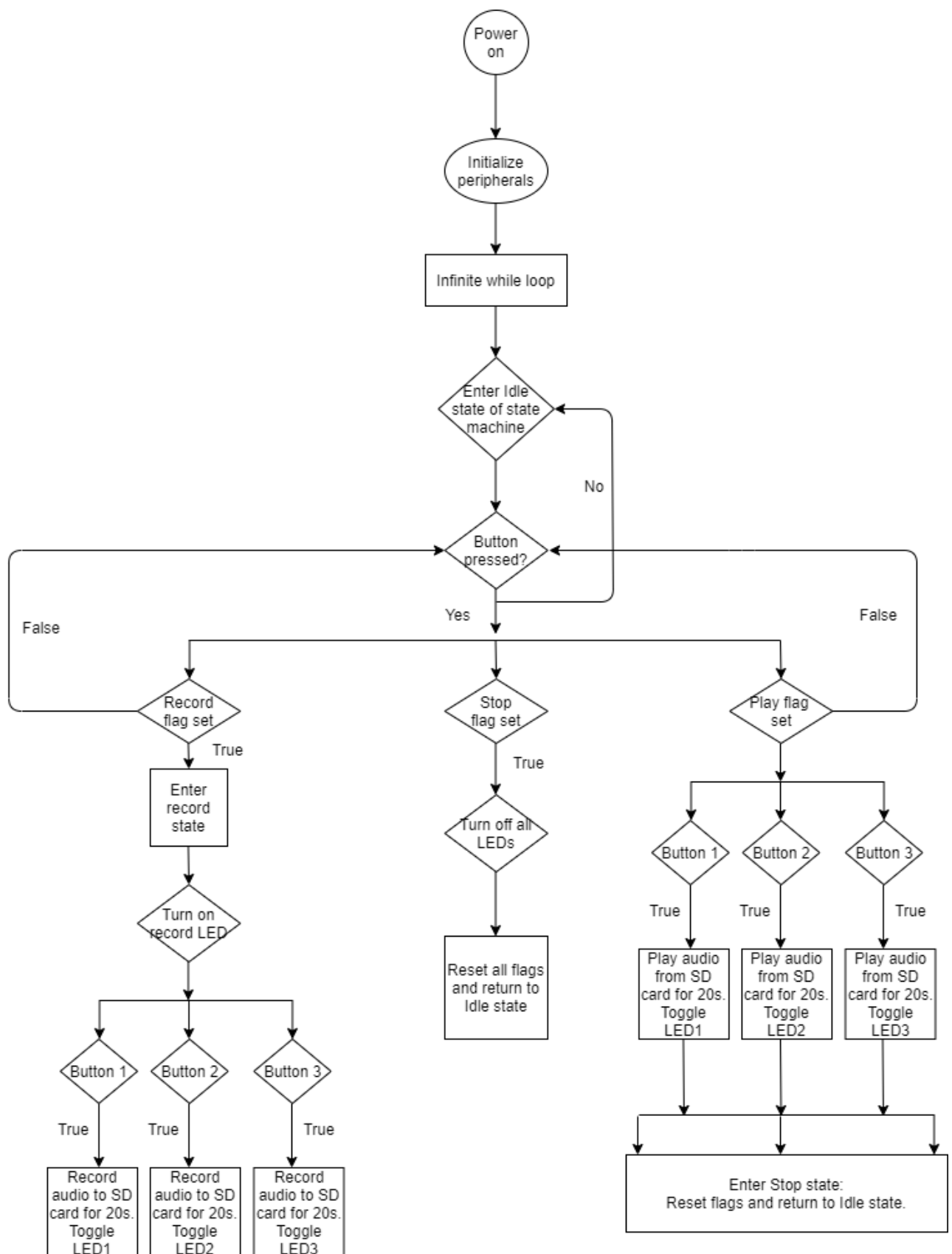


Figure 7:Flow diagram of system

4.2 Control Logic

The state machine diagram for the state machine that was implemented in software is shown in Figure 8.

To transition between states in the state machine, flags were set to correspond to each state input. These flags would be set based on the interrupts generated from button presses, and reset whenever a valid stop condition was processed. The states of the state machine are as shown in Figure 6, with Idle being the default state. For validity of the stop condition, either of the following had to occur:

- The stop button was pressed,
- The time elapsed for recording was 20 s,
- The time elapsed for playback of the recording was 20 s.

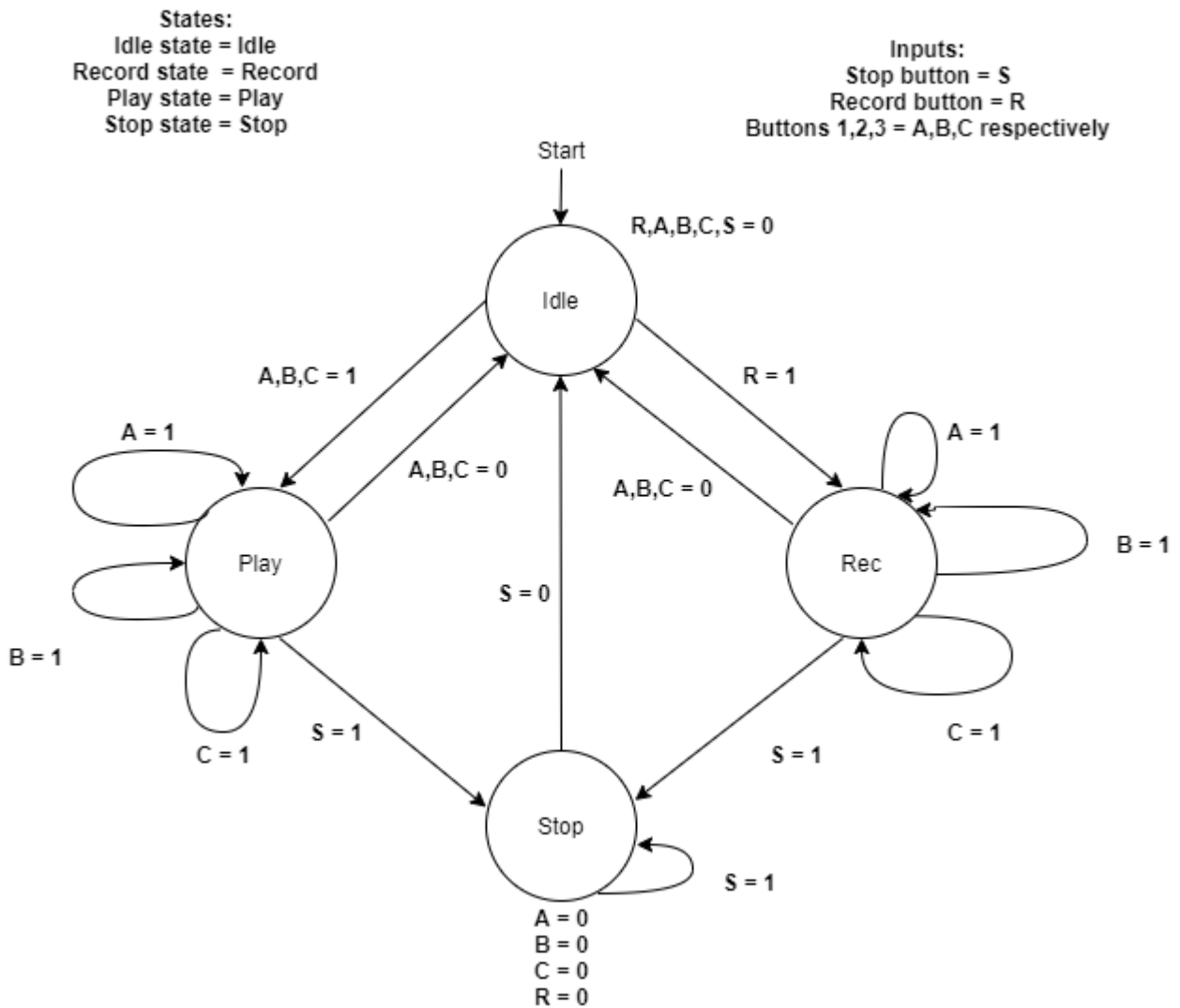


Figure 8: State Machine Diagram

4.3 Button bounce handling

As stipulated in Requirement 13 [1], mechanical bounce was to be suppressed for 10 ms. The following flow diagram illustrates how the mechanical bounce suppression was implemented:

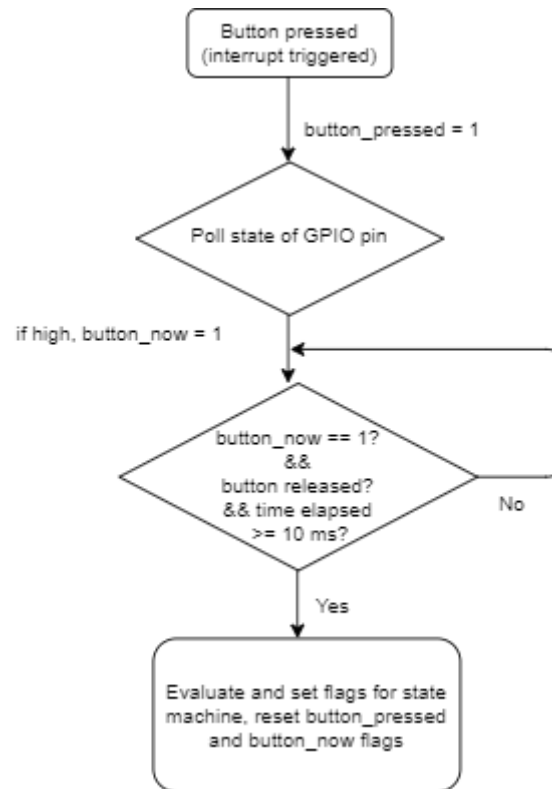


Figure 9: Flowchart for button bounce handling

4.4 Data flow and processing

As shown in Figure 11, the microphone output was transferred to memory through the use of ADC and DMA. This structure allows for the CPU to continue to perform other tasks, while allowing the ADC direct access to memory. The memory buffer was chosen as an array of 1024 bytes, since each sector of the SD card has a size of 512 bytes [11], and also since the DMA was set to operate in circular mode, hence continuous overwriting of the memory buffer.

The method used to create and store a file on the SD card using the FATFs system Is shown in Figure 10.

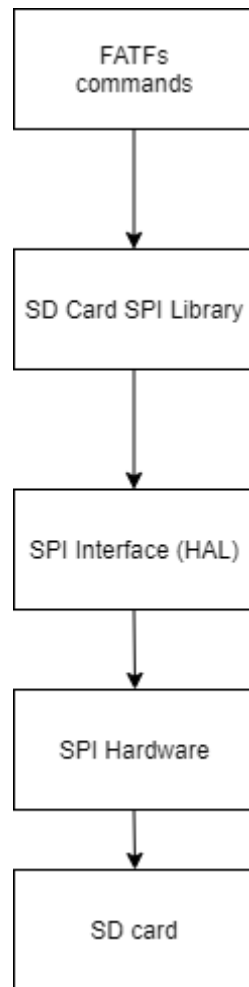


Figure 10: File creation on SD card via SPI

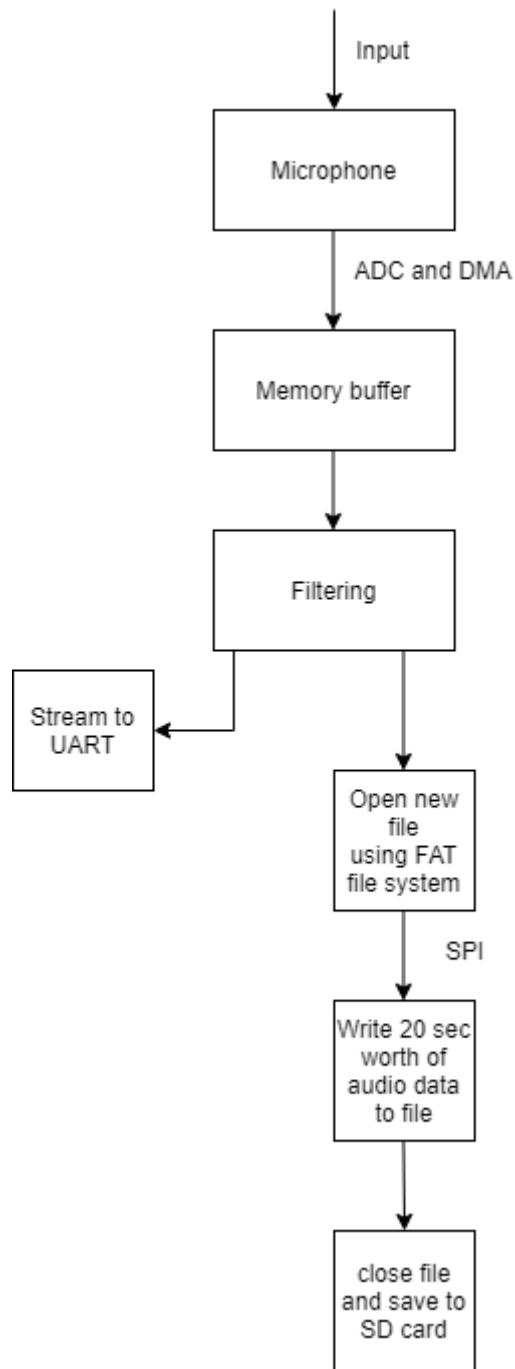


Figure 11: Data flow and processing

4.5 SD card interfacing

To interface with the SD card, the source files [12] were used, and these contain helper functions that were used to implement the commands detailed in [11] to the SD card. The following table lists the functions and their descriptions:

Table 2: sd.c file

Function	Description
uint8_t SD_SendCommand (uint8_t cmd, uint32_t arg)	Sends desired command to the SD card.
uint8_t SD_Init ()	Initializes the SD card.
uint8_t SD_Read (uint8_t* rxbuffer, uint32_t address, uint32_t numblocks)	Read 512 byte block from SD card.

uint8_t SD_Write (uint8_t* txbuffer, uint32_t address, uint32_t numblocks)	Write 512 byte block to SD card.
---	----------------------------------

4.6 Peripheral setup

To interface with the microcontroller, and to satisfy specified functionality, the following peripherals were setup using the Device Configuration Tool of STM32CubeIDE. Communication with the peripherals was facilitated by the already defined functions from the HAL library [13].

4.6.1 UART

Channel 2 of the UART was configured to transmit data in blocking mode. The baud rate was set to 500000 bits/sec, and the format of the data was 8N1 (8 data bits, 1 start bit, 1 stop bit and no parity bit).

4.6.2 GPIO

The GPIO pins were either set to GPIO Output mode to toggle the LEDs, or External Interrupt mode with rising edge trigger detection to process the input from the push buttons. The pins configured in External Interrupt mode were also set up in pull down mode, to prevent them from being in a floating state.

4.6.3 ADC

The ADC was used to sample the input from the microphone module. To do this, ADC channel 2 was selected, and configured to have a resolution of 8 bits. In addition to this, a timer update event was configured as the trigger conversion source. Furthermore, DMA transfers were enabled for the ADC, with the DMA controller set to transfer a byte to memory (data width of a byte), since the ADC has a resolution of 8 bits. Finally, continuous requests were enabled, and the DMA was set to circular mode to enable continuous overwriting of the sampling buffer.

4.6.4 DAC

The DAC was used to convert the signal sampled by the ADC to an analog output. To do this, DAC channel 1 was selected, and as with the ADC, a timer update event was configured as the trigger conversion source. Additionally, DMA transfers were enabled for the DAC, and so the DMA was set to normal mode. The data width was left unaltered, with a standard configuration of half-word.

4.6.5 Timers

The trigger conversion source for the ADC and DAC required the use of a timer, and to achieve this timer 8 was used for both the peripherals. Since the recording and playback were to happen at a rate of 44.1 kHz [1], the counter period was calculated as:

$$T = \frac{N(R + 1)}{f_{clk}}$$

Where N is the prescaler value, R the reload value and f_{clk} the frequency of the clock source. From the above formula, the reload value was calculated as 1905. Furthermore, for the trigger event selection, the update event option was selected for timer 8.

It was also required that a timer be used to toggle the LEDs as required in [1]. To do this, timer 3 was selected to generate an interrupt every 250 ms, and from the formula above, with the prescaler value

chosen as 8400, the reload value was calculated as 2499. The remaining settings were kept in their default configuration. The clock source for all the clocks (f_{clk}) was set as the internal clock (which is 84 MHz) [2].

4.6.6 SPI

To communicate with the SD card, the SPI peripheral would have to be used. To achieve this, SPI1 was configured in Full-Duplex Master mode. In addition to this, the prescaler was set to 8, resulting in a baud rate of 10.5 Mbits/s. All other settings were left unchanged.

5 Measurements and Results

5.1 Power Supplies

5.1.1 5 V Power Supply

To evaluate the performance of the 5 V regulator, two tests were conducted, namely:

- No-load test
- Load test under various loads

The following procedure was used to determine the performance of the 5 V power supply:

Requirement(s):

- The output voltage of the 5 V regulator shall be $5\text{ V} \pm 0.1\text{ V}$, for loads in the range 0 – 300 mA.

Procedure:

A range of loads that would draw current in the range specified above were connected to the 5 V regulator output. In addition to this, a stress test was carried out to assess if the regulator could provide more than what it was officially required to [1]. The voltage and current measurements were then obtained using a multimeter. The test was considered successful if the output voltage was within the required tolerance, and unsuccessful if the output voltage exceeded this tolerance.

Results:

Table 3: 5 V power supply results

Range	Calculated load / Ω	Actual load / Ω	Voltage / V	Current / mA	Expected current / mA
No load	0	0	5.11	0	0
Mid-load	33	33	5.11	149	150
Full load	16.67	18	5.10	292	300
Stress test	15.87	15	5.11	335	315

The above results show that the output of the regulator is within the 5% tolerance range as required.

5.1.2 3.3 V Power Supply

To evaluate the performance of the 3.3 V regulator, two tests were conducted, namely:

- No-load test
- Load test under various loads

The following procedure was used to assess the performance of the 3.3 V power supply:

Requirement(s):

- The output voltage of the 3.3 V regulator shall be $3.3 \text{ V} \pm 0.1 \text{ V}$ for loads in the range 0 – 25 mA.

Procedure:

A range of loads that would draw current in the range specified above were connected to the 3.3 V regulator output. In addition to this, a stress test was carried out to assess if the regulator could provide more than what it was officially required to [1]. The voltage and current measurements were then obtained using a multimeter. The test was considered successful if the output voltage was within the required tolerance, and unsuccessful if the output voltage exceeded this tolerance.

Results:

Table 4: 3.3 V power supply results

Range	Calculated load / Ω	Actual load / Ω	Voltage / V	Current / mA	Expected output current / mA
No load	0	0	3.33	0	0
Mid-load	264	265	3.33	12.5	12.5
Full load	132	150	3.33	22.6	25
Stress test	125.8	120	3.33	27.8	26.25

The above results show that the output is within the 5% tolerance region as required.

5.2 UART communications

Requirement(s):

- The bitrate of the UART output shall be 500000 bits/sec.

Procedure:

The best procedure would involve connecting an oscilloscope probe to the RX pin of the UART and sending a character from the microcontroller. The period would then be evaluated using the oscilloscope. The measured bitrate would then be calculated from the formula:

$$\text{bitrate} = \frac{1}{\text{period}}$$

However, due to unforeseen circumstances, the above test procedure could not be carried out. An alternative method to test that the configured bitrate satisfies the requirement would be to configure a terminal program (such as Termit or TeraTerm) to receive serial data at the required bitrate. A character, or string of characters could then be sent from the microcontroller. The test would be considered successful if the output on the terminal program matches the character/string transmitted via UART, and unsuccessful if the output does not match the character/string transmitted via UART.

Results:

The terminal output of the string sent via UART at a bitrate of 500000 bits/sec is shown below.

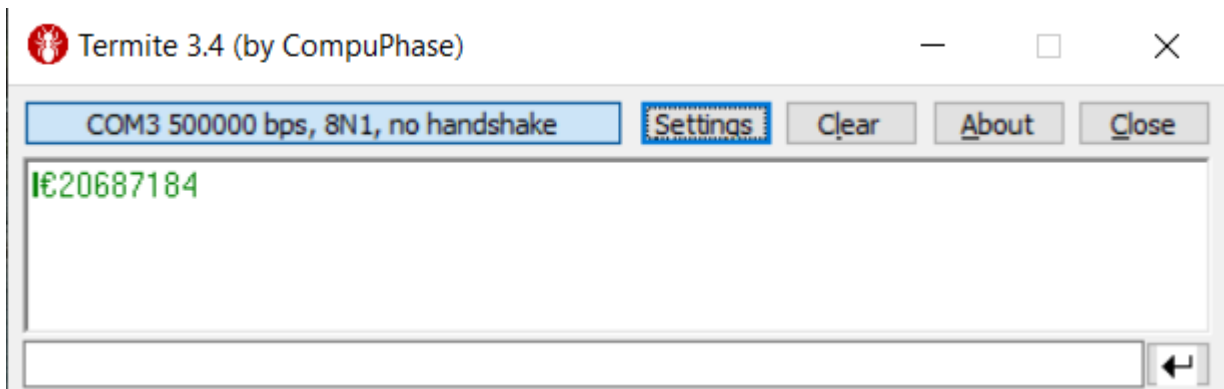


Figure 12: Terminal program output

5.3 Buttons

Requirement(s):

- Mechanical bounce shall be suppressed for a duration of 10 ms.

Procedure:

As the mechanical debouncing was implemented in software, and since aspects of the device such as flashing of the LEDs were dependent on the button inputs, the following procedure was devised to test that the requirement was satisfied.

The push buttons were each pressed and held in for a duration longer than 10 ms, then released and the LED output observed. The test was considered successful only if the LED turned on after the button was released, and unsuccessful if the LED turned on during the time that the push button was held in.

5.4 LEDs

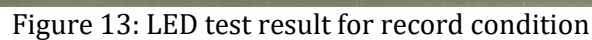
Requirement(s):

- The LEDs shall either turn on / off after the relevant button has been pressed and released as stipulated in [1].

Procedure:

A push button would be pressed and released, and the corresponding LED would be inspected to see if its response matched that specified in [1]. The test was considered successful if the LED turned on if it was off (for either the record or play conditions), and off if the LED was on (stop condition). The test was considered unsuccessful if the response of the LED did not match the expected output in [1].

The result for one of the record conditions [1] is shown below:



5.5 Audio output

The following requirements are as specified in [1]:

- Using an oscilloscope and an oscilloscope probe, the voltage output of the Nucleo board DAC is measured. The test was considered positive if the frequency of the sinusoidal output matched that stipulated in the requirements, and if the output voltage was within the range stipulated above. The test was considered unsuccessful if either of the above requirements did not hold in the output signal.

The results of the above procedure are shown below:

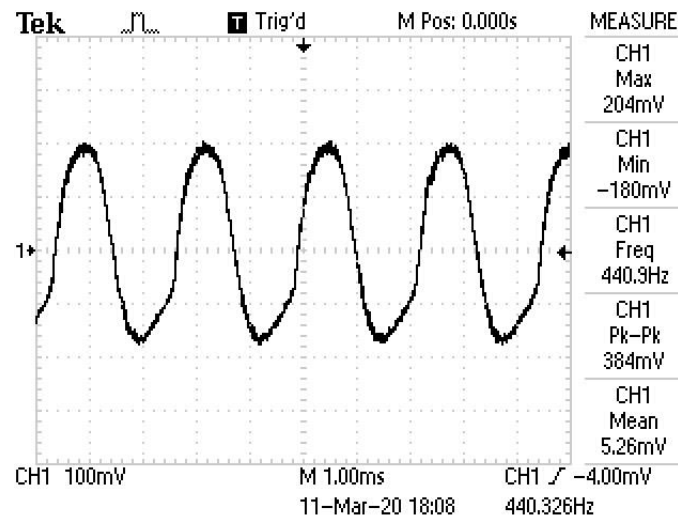


Figure 14: Audio output for 440 Hz sine wave

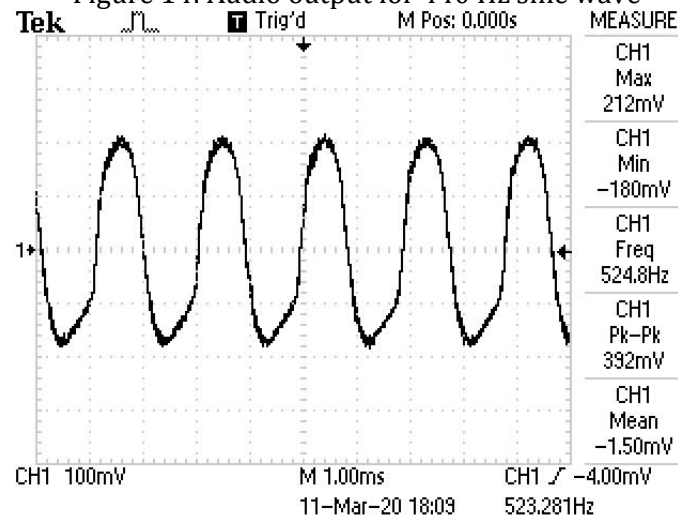


Figure 15: Audio output for 523 Hz sine wave

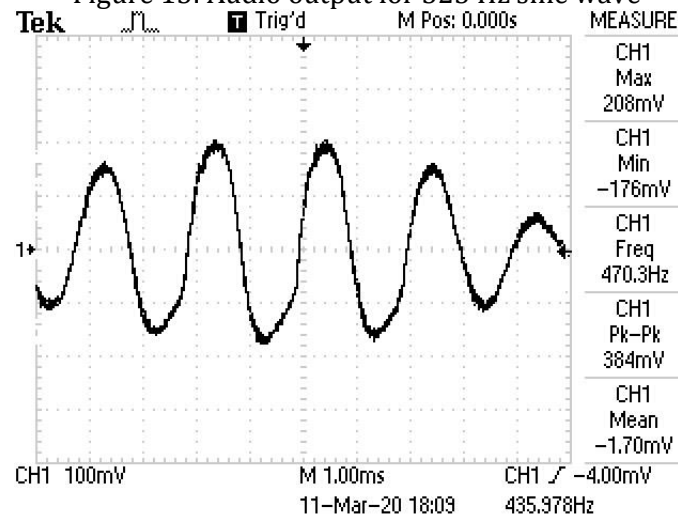


Figure 16: Audio output for summed sine waves at 440 Hz and 523 Hz

The output of the sum of the two sinusoids was expected to have a frequency of:

$$f_{sum} = \frac{440 + 523}{2} = 481.5 \text{ Hz}$$

As can be seen from the results above, the voltage output lies within the range specified in the requirements, as well as the frequencies. The slight deviation in the frequencies is as a result of

imperfections in the DAC, however, the error for all the frequencies is less than 5%, hence the test was successful.

5.6 Microphone

Requirement(s):

- The microphone module shall output a (sinusoidal) signal appropriate for ADC sampling (in the range 0 – 3.3 V).

Procedure:

Using an oscilloscope probe connected to an oscilloscope, the microphone module output voltage pin is probed, and the oscilloscope is observed.

Expected results:

When there is no input to the microphone, the signal level on the oscilloscope should hover around the 0 level of the microphone module output. When there is an input to the microphone module, a sinusoidal waveform should be observed on the oscilloscope. The test is unsuccessful if either of these conditions are not met.

5.7 SD Card

Requirement(s):

- The SD card shall interface with the Nucleo module via SPI, and allow for the reading of and writing to files on the SD card.

Procedure:

The following procedures can be used to verify the requirement(s):

- An oscilloscope probe is connected to the SCK, MOSI, and MISO pins of the SD card module, and the oscilloscope is observed.
- After recording, the SD card is unmounted from the module and connected to a PC. The file is then analyzed with Audacity to assess conformity with the stipulations in [1]. Alternatively, with the SD card still mounted onto the module, a button press conforming to the playback condition can be performed and the output at the headphone socket observed.
- The initialization process documented in [11] is performed to verify that the SD card is responsive when commands are sent to it. The reply from the SD card is then transmitted via UART for further analysis on a terminal program.

Results:

For the first procedure above, a successful test is indicated by the presence of SPI signals on the oscilloscope screen when communication occurs between the Nucleo module and the SD card module. An unsuccessful test is indicated by the absence of any signal on the oscilloscope screen.

For the second procedure, a successful test is indicated by the presence of a file on the SD card, and by the file data being interpreted as audio output by Audacity. For the alternative case, a successful test is indicated by the presence of audio output. For both cases, an unsuccessful test is indicated by the absence of any audio output on either the SD card or headphone socket.

For the third procedure, a successful test is indicated by a single byte with the value of 1 being sent to the Nucleo module by the SD card. An unsuccessful test is indicated by the absence of any data when observing the terminal program.

6 Conclusions

After testing the system to establish satisfaction of the required functionality, it was concluded that the system was compliant with the required functionality.

6.1 Strengths

- The system conformed to the requirements stipulated in [1].
- The coding approach was modular, hence extra functionality could be easily added and the debugging procedure was made easier.

6.2 Weaknesses

- Due to damage caused by soldering, the project board had to be taken in for maintenance.
- There initially was distortion for the output of playback condition 3 [1], however this was corrected after being identified.
- No logic was implemented to handle button input during the occurrence of playback or recording.

6.3 Improvements and suggestions

- Use higher quality PCBs to minimize the damage to solder pads caused by soldering.
- Add functionality for audio manipulation and effects.
- Add functionality to include real-time audio processing (i.e. concurrent playback as audio data is being recorded).

7 Bibliography

- [1] D. A. Barnard and D. L. Visagie, "E-Design 314 -2020 Project Definition Document," Stellenbosch, 2020.
- [2] STMicroelectronics, "STM32F446xC/E Datasheet," 2016.
- [3] Fairchild Semiconductor Corporation, *LM78XX / LM78XXA 3-Terminal 1A Positive Voltage Regulator*, 2006.
- [4] Diodes Incorporated, "1N4001 - 1N4007 Datasheet".
- [5] Microchip Technology Inc., *MCP1700 Low Quiescent Current LDO*, 2018.
- [6] D. A. Barnard, *Capacitor Suggestion (Lab Announcement)*, 2020.
- [7] Waveshare, *Sound Sensor Schematic*.
- [8] Texas Instruments, "LM386 Low Voltage Audio Power Amplifier," 2017.
- [9] V. M. Aiea, "CATALEX Micro SD Card Module," 2017.
- [10] Catalex, *Micro SD Card Micro SDHC Mini TF Card Adapter Reader Module for Arduino*.
- [11] SD Card Association, "SD Specifications Part 1 Physical Layer Simplified Specification Version 7.10," 2020.
- [12] D. L. Visagie, *sd.c and sd.h source files*, 2020.
- [13] STMicroelectronics, "Description of STM32F4 HAL and LL drivers, Rev 5," 2017.
- [14] Microchip Technology Inc, *MCP601/1R/2/3/4 2.7V to 6.0V Single Supply CMOS Op Amps*, 2007.
- [15] STMicroelectronics, "AN3126 Application note: Audio and waveform generation using the DAC in STM32 microcontrollers," 2017.

8 Appendices

8.1 Appendix A - System schematics

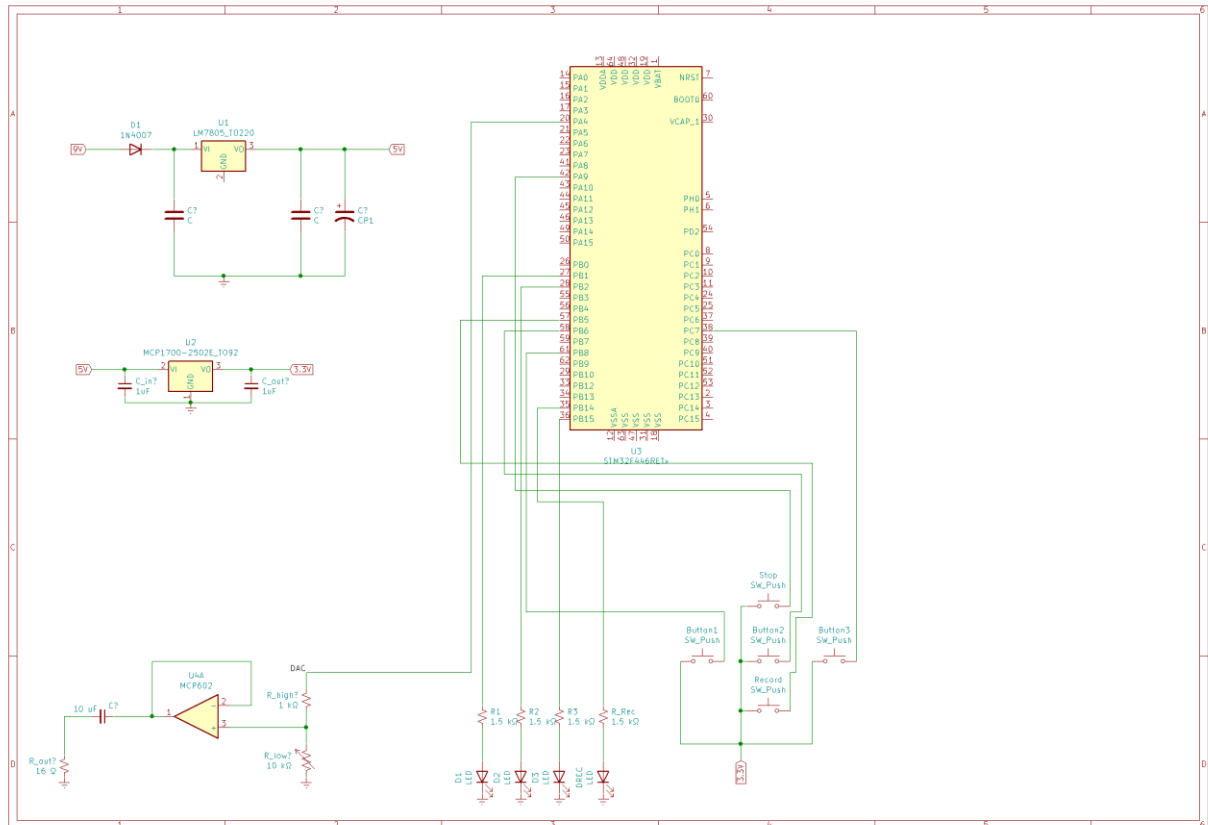


Figure 17: Partial schematic of system

The following schematic was taken from [7].

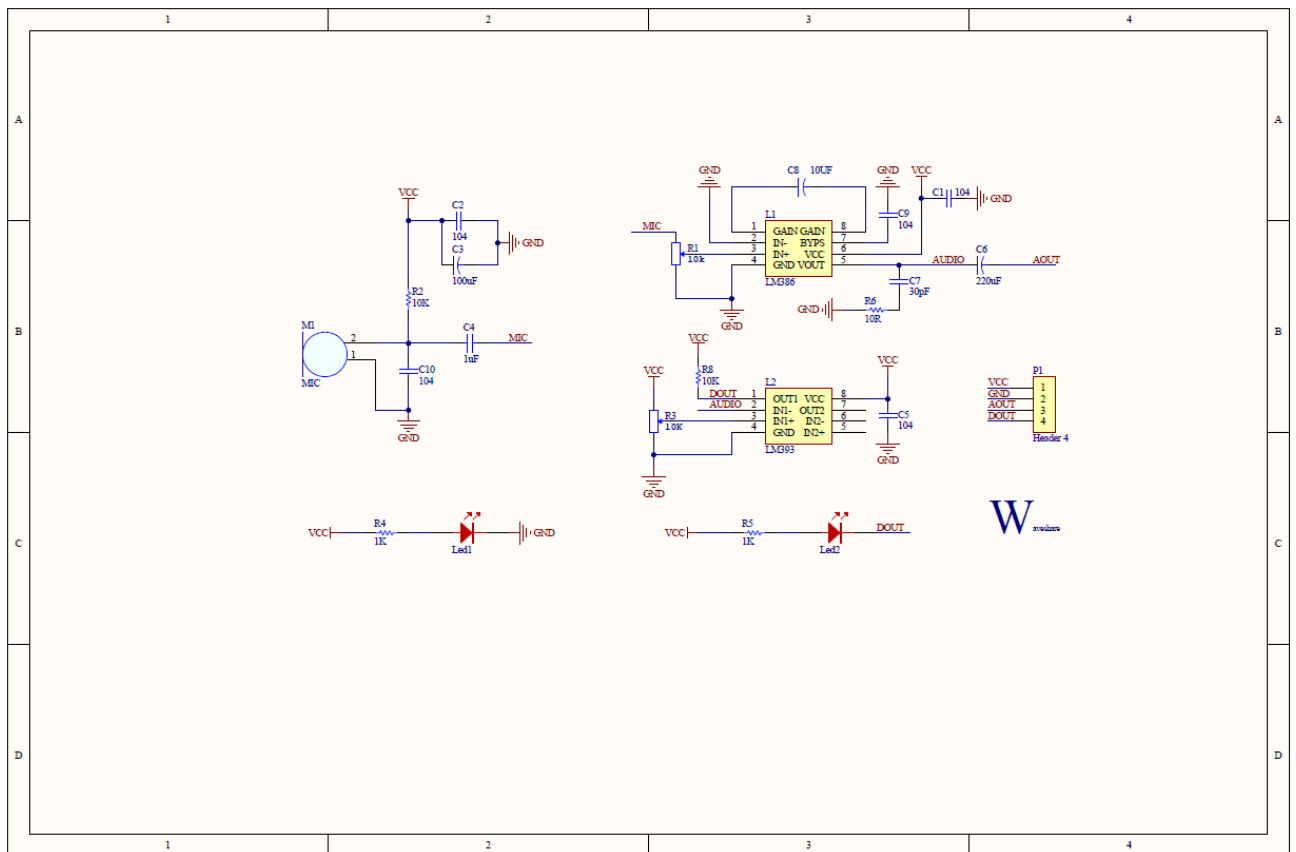


Figure 18: Microphone module schematic

The following schematic was taken from [9].

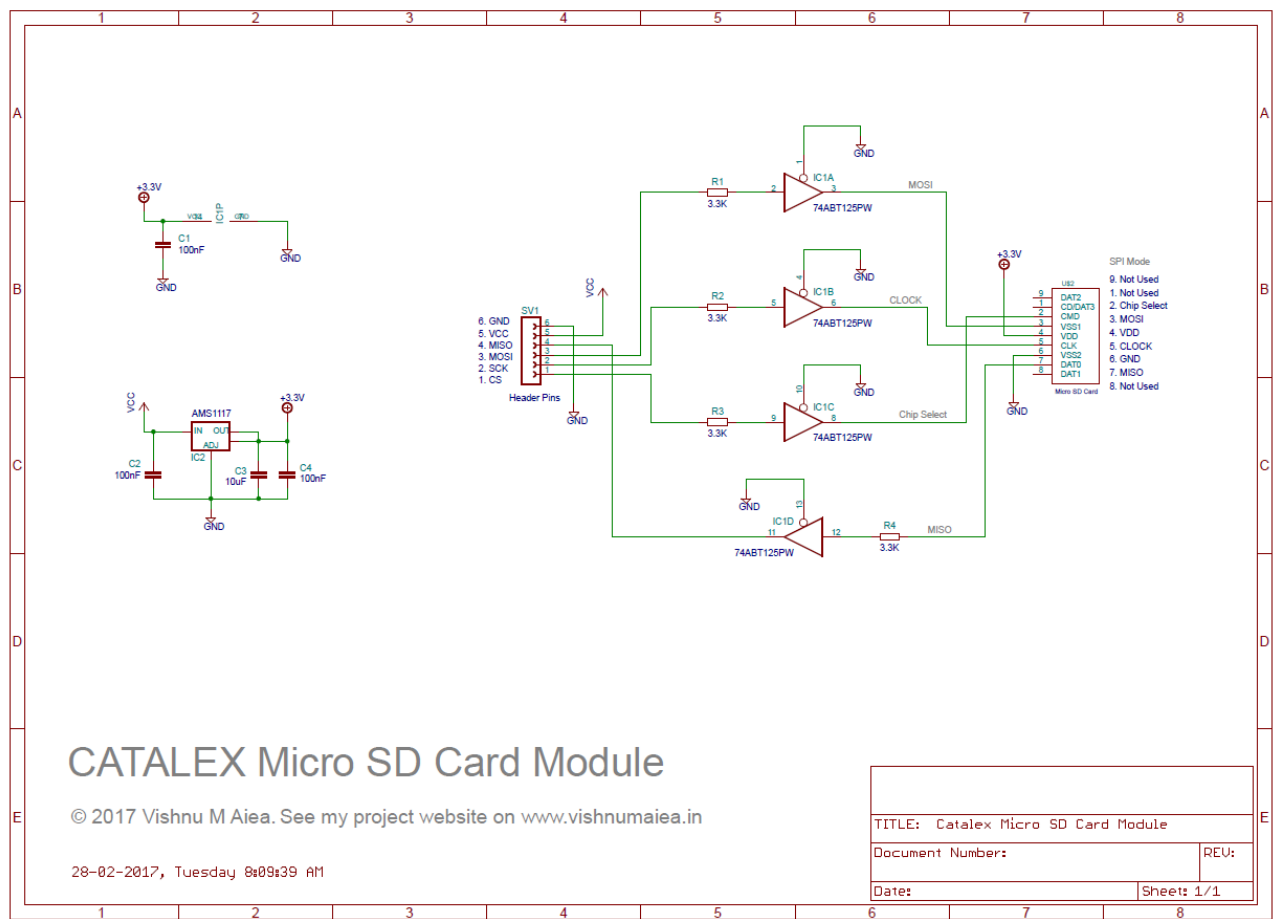


Figure 19: SD card module schematic

8.2 Appendix B – STM32 module pinout table

Table 5: STM32 module pinout

Pin	Configuration	Connection to hardware element
PA4	DAC_OUT1	Amplifier sub-circuit
PA6	SPI1_MISO	SD card module (MISO Pin)
PA7	SPI1_MOSI	SD card module (MOSI Pin)
PA9	GPIO_EXTI9	Stop button
PB0	GPIO_Output	SD card module (CS Pin)
PB1	GPIO_Output	LED1
PB2	GPIO_Output	LED2
PB3	SPI1_SCK	SD card module (SCK Pin)
PB5	GPIO_EXTI5	Record button
PB6	GPIO_EXTI6	Button 2
PB8	GPIO_EXTI8	Button 1
PB14	GPIO_Output	Record LED
PB15	GPIO_Output	LED3
PC4	ADC2_IN14	Microphone module (AOUT Pin)
PC7	GPIO_EXTI7	Button 3