Augmented Reality System for bats based on Acoustic Signal Manipulation

The study of bats requires the ability to record and generate sound waves in the ultrasonic ranges. The required bandwidth for such systems is up to 100kHz, which is much higher than the common acoustic systems designed for humans. Studying bat echolocation capabilities and generating real time acoustic responses requires a minimal latency acoustic system. The few available systems on the market aren`t designed for real time signal processing and manipulation. In order to meet those constrains, a full acoustic system was developed. The system includes a Smart microphone module embedded with a microprocessor, a Speaker-amplifier and a Switch box which is capable of signal redirection and real time processing (Figure 1).

The proposed system in this study allows signal manipulation and processing at up to 1 MHz sample rate with a bandwidth of 1 – 100 kHz. The system is capable of monitoring up to 5 microphones and directing a selected sound source to up to 5 speakers for simultaneous transmission. Several experiments are presented demonstrating the systems capabilities for signal manipulations such as digital filtering, automatic triggering and signal rerouting, inducing latencies and variable gains. In addition, future developments will allow more complex applications such as a real time doppler effects.

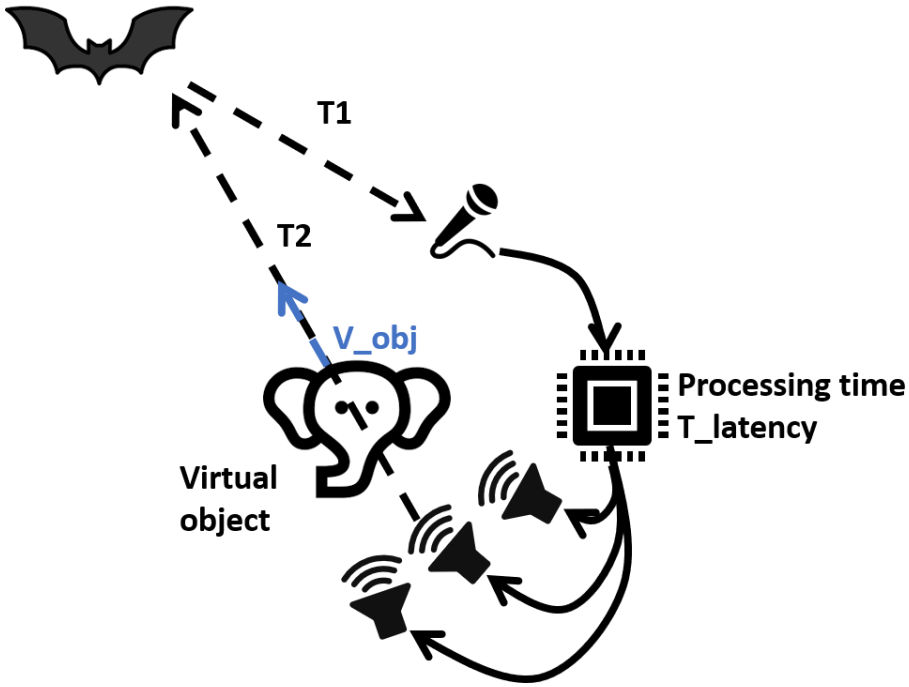

Figure 1. Augmented reality illustration: A bat call is received by a microphone, processed and transmitted through several speakers generating a virtual object to be perceived by a bat.

1 System Overview

The PlayBack setup consists of three main modules: Smart Microphones, a Switch and speaker amplifiers (Figure 2). It is designed to monitor audio signals from multiple microphone modules and direct a signal from a selected microphone to an array of speaker modules including DSP options for audio processing in real time. The setup is designed to work at an audio bandwidth of 1-100 kHz.

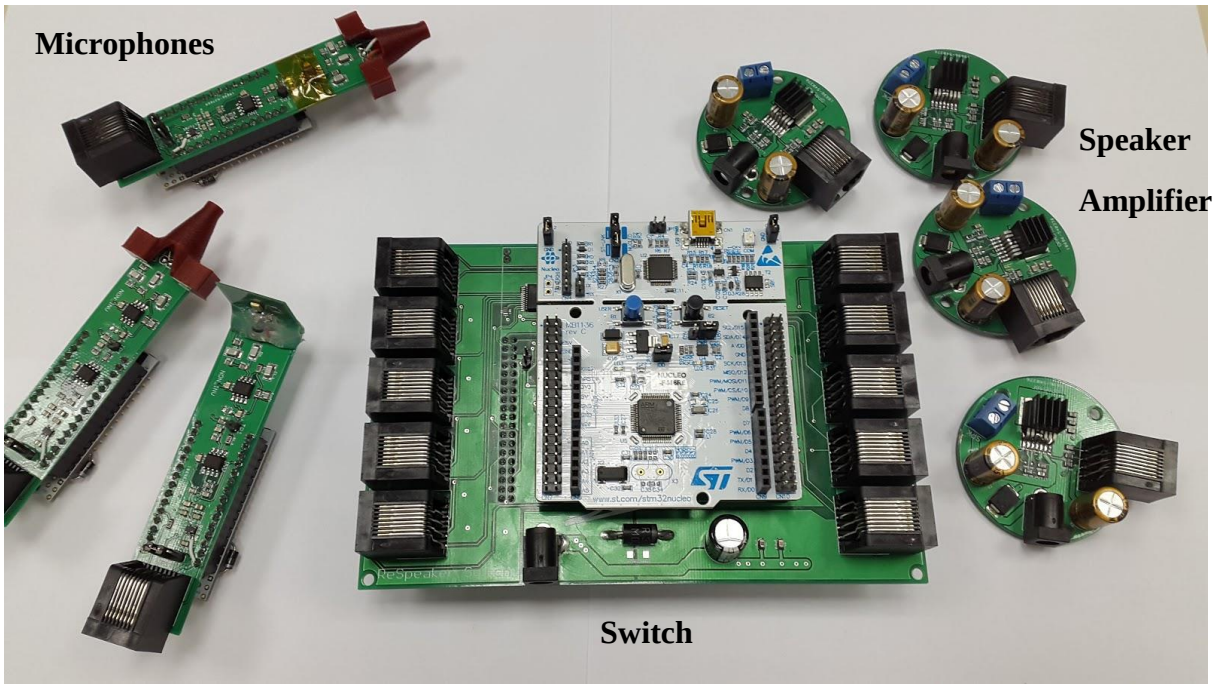

Figure 2. The Playback system – Smart Microphone on the left, Switch Module in the middle and Speaker amplifiers on the right.

The system's topology (Figure 3) is such that the microphone modules are connected to the switch module using a standard RJ25 cable. The 6 wires of the cable are used to pass the following between the Smart microphones and the Switch: power to the module, trigger signal, the audio signal and an analog "quality" signal. The Switch module consists of two MCU`s (Micro Controller Unit), one is used as the Switch interface, and routes the signals while the other is used as a DSP MCU to implement signal processing in real time to the acoustic signal. The speaker amplifiers are connected through an RJ45 cable which in turn is used to supply power from the switch to the amplifiers and transmit the audio signals and the enable signal to the amplifier. The Switch module is connected to a Matlab GUI application running on a PC using an Xbee adapter for wireless operation. In addition, a Saleae logic analyzer is connected to the Switch allowing to monitor the analog signals and the trigger signal (Figure 4).

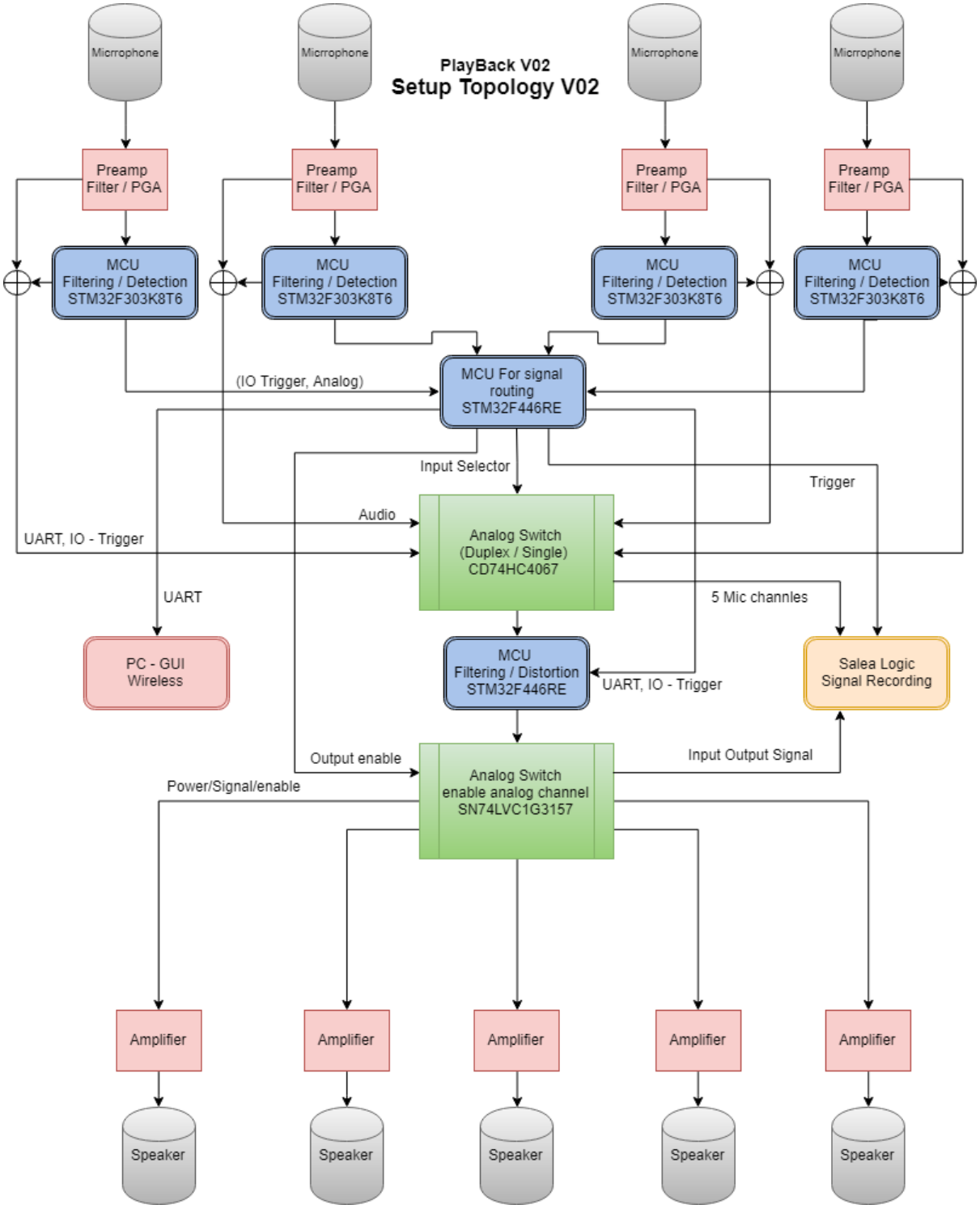

Figure 3. System topology diagram: An array of up to 5 microphone modules are connected to a switch module which connects to up to 5 speaker amplifier modules with ultrasonic speakers. The switch module is connected to a PC and the signals are monitored by a Saleae Logic analyzer.

Connection Diagram

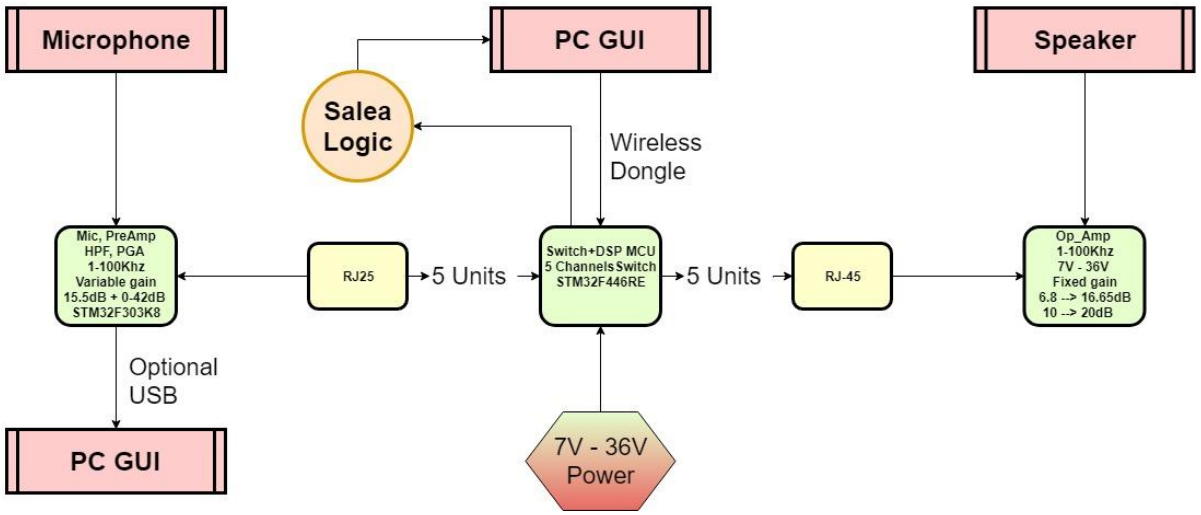

Figure 4. Playback System Connection diagram: Microphones connected through RJ25 cables to a switch module which is connected to speaker amplifiers through RJ-45 cables.

1.1 The smart microphone

The Smart microphone module (Figure 5) is designed to meet the required acoustic bandwidth of bat echolocation calls. In addition, since the system requirements are minimal latency for signal processing and signal manipulation, each of the microphones are embedded with a dedicated microprocessor which is responsible for the initial signal processing of the individual acoustic source. Such an approach allows spreading the computational power required by the system and simplifies the decision-making requirements of the Switch by reducing the acoustic information to a more simplified representation of an acoustic source. i.e. digital triggers / signal quality indication. This approach also allows future expending of the system since each of the microphones acts as an independent entity responsible for its own acoustic system. Adding more microphone to a system will not influence the required signal processing for the main unit.

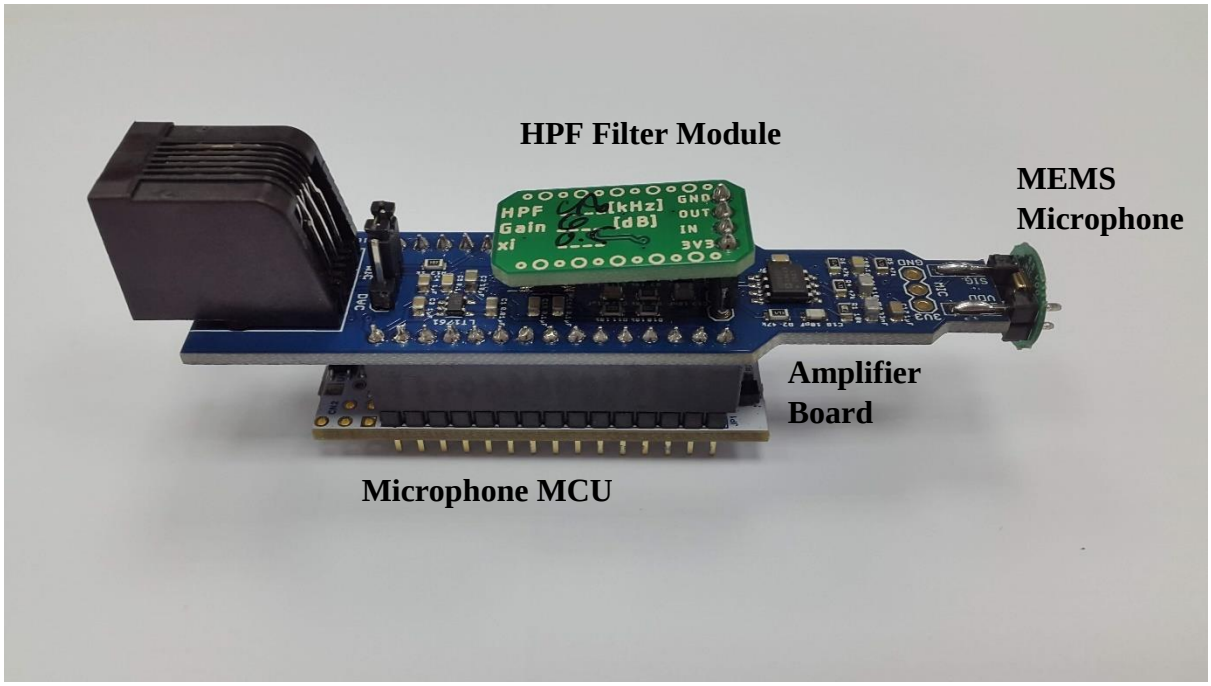

Figure 5. The Smart Microphone module: Round PCB populated with a mems microphone membrane, attached to an analog amplifier board (blue PCB). At the top attached an analog HPF filter module and at the button an MCU board.

1.1.1 Electronics

The Smart Microphone module is assembled by 4 independent electronic boards (Figure 5). The modules contain several parts, an analog mems membrane for the ultrasonic ranges and an analog amplifier board matched for the mems microphone. The amplifier board contains an adapter for an Analog high pass filter to allow the replacements of the board for various filter characteristics for the specific acoustic responses required. The whole system is mounted on a microcontroller board for signal detection and signal manipulation.

The circuit components were chosen to meet the system's requirements of an audio bandwidth of 1-100kHz. The mems microphone (SPU0410LR5H) chosen for the design has an acoustic characteristic at up to 100kHz. The signal operational amplifiers (Op Amp`s)

chosen for the design are an AD8655 which is a low noise high bandwidth operational amplifier, with 28MHz bandwidth, capable of a single supply design and rail to rail signals. The programmable gain amplifier (PGA - PGA112) module has similar analog characteristics of 10MHz bandwidth, rail to rail signals and is compatible for single supply designs. In addition, the PGA has an internal MUX of 2 channels, allowing a digital selection of the signal source. The chosen PGA is cable of an additional amplification stage in binary stages from 1 to 128 gain. The main board 3V3 LDO regulator LT1761 is selected for its good noise rejection for the circuit`s analog bandwidth.

In order to work with an analog microphone a matching amplification circuit is implemented. The amplification circuit has several stages each selected and implemented for its individual purpose and are illustrated in Figure 6. The first stage contains an amplifier circuit at a non-inverting configuration with a gain of 15.1dB and is called a preamplifier. It`s purpose is to amplify the “weak” signals generated by an analog microphone. It is designed with a high impedance input to match the high impedances of the analog microphone. The amplification is selected to meet the full bandwidth of the microphone up to its saturation point. Following the preamplifier is an analog high pass filter module (HPF module). The analog high pass filter is implemented on a separated board in order to be easily replace by a different high pass filter module with different analog characteristics. Following the HPF module is a programmable gain amplifier (PGA). The PGA is configured digitally by the on board MCU and as an internal mux allowing the selection of a filtered signal from the HPF module or the unfiltered signal fed directly by the preamplifier. The PGA acts a variable gain amplifier and is implemented in order to amplify the signals to the full scale of the microprocessor’s ADC levels (analog to digital converter) by that utilizing the full resolution of the MCU ADC which is limited to 12bit. The amplifier board has a physical jumper which can select the output source for the acoustic signal. Either the generated signals by the MCU, or the analog signal of the PGA unit.

The HPF module for the setup is designed as a second order analog high pass filter at a cutoff frequency of 48kHz with a damping factor of 0.5 in addition it adds a 6dB gain to the signal respectively to the dumping factor, The design was evaluated using a Simulink simulation and evaluated based on preliminary recording for best performance at ultrasonic ranges of a bat callings.

The amplifier board is mounted on top of a microprocessor which samples the amplified signals of the PGA unit and generates an analog signal through its digital to analog converter (DAC) at 12bit resolution and up to 1 MHz sample rates. The MCU can delay the signals and detect signals presence by amplitude threshold evaluation. When a signal is present an IO interrupt is set to notify the switch of a present signal. The software parameters of the MCU are configured by a Matlab GUI (Figure 9, Figure 6).

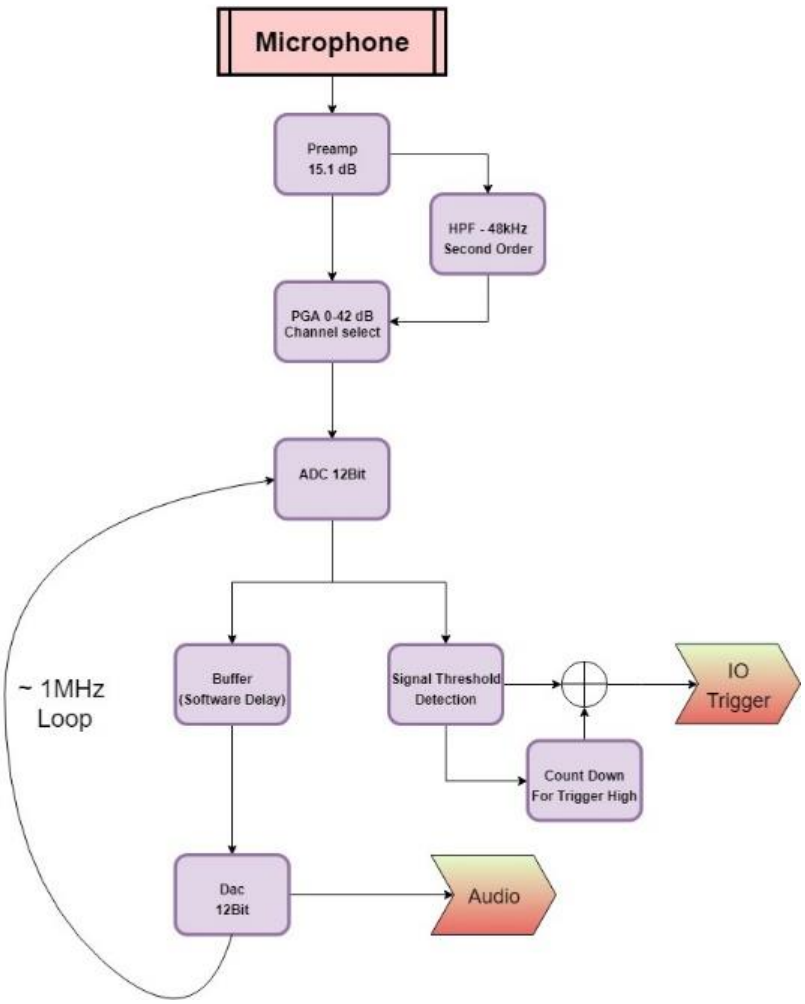

Figure 6. Smart Microphone diagram: Hardware & Software implementation diagram. Illustrated an amplifier circuit, with preamp, HPF and PGA implementation. Following sampling of the signal by a microcontroller, and software implementation for the sample. Signal detection, and delay implementation. With appropriate signal generation – i.e. Digital trigger and generated analog signal.

1.1.2 MCU code

The code for the Smart Microphone MCU (STM32F303) is implemented through an STM32CubeIDE environment, using C++ language. The MCU is responsible to the communication with the Matlab GUI and implements some signal processing in real time.

The main code algorithm is represented in Figure 6, The MCU performs an analog to digital conversion at 12 bits resolution (ADC), stores the sample into a circular buffer, checks if a bat signal is present by a simple threshold comparison to a stored variable, if a signal is present the MCU sets a trigger IO high to update the Switch MCU of a present signal and sets a countdown which ends with resetting the trigger IO state. A digital to analog conversion

(DAC) at 12 bits is than performed to a stored value in the circular buffer, defined by the set delay. The whole loop is performed at about 1MHz.

The communication with the Matlab GUI is done through a UART stream which triggers an interrupt routing in the MCU workflow. The incoming byte is checked for the correct values depending on the defined state for the transmission (Figure 7).

- State 1: compares the incoming byte to the defined header - failed comparison resets the buffer and transmission state
- State 2: stores the incoming bytes to a packet stream.
- State 3: compares the incoming byte to the defined footer - failed comparison resets the buffer and transmission state.

When a successful transmission is performed, the defined packet bytes are parsed in the format of a Json message. The variables are derived by the variable names and used to update the stored variables of the MCU in the virtual EEPROM memory of the MCU. The data exchange is completed by sending an acknowledgement message to the Matlab GUI, stating the microphone number using the Json format for message construction.

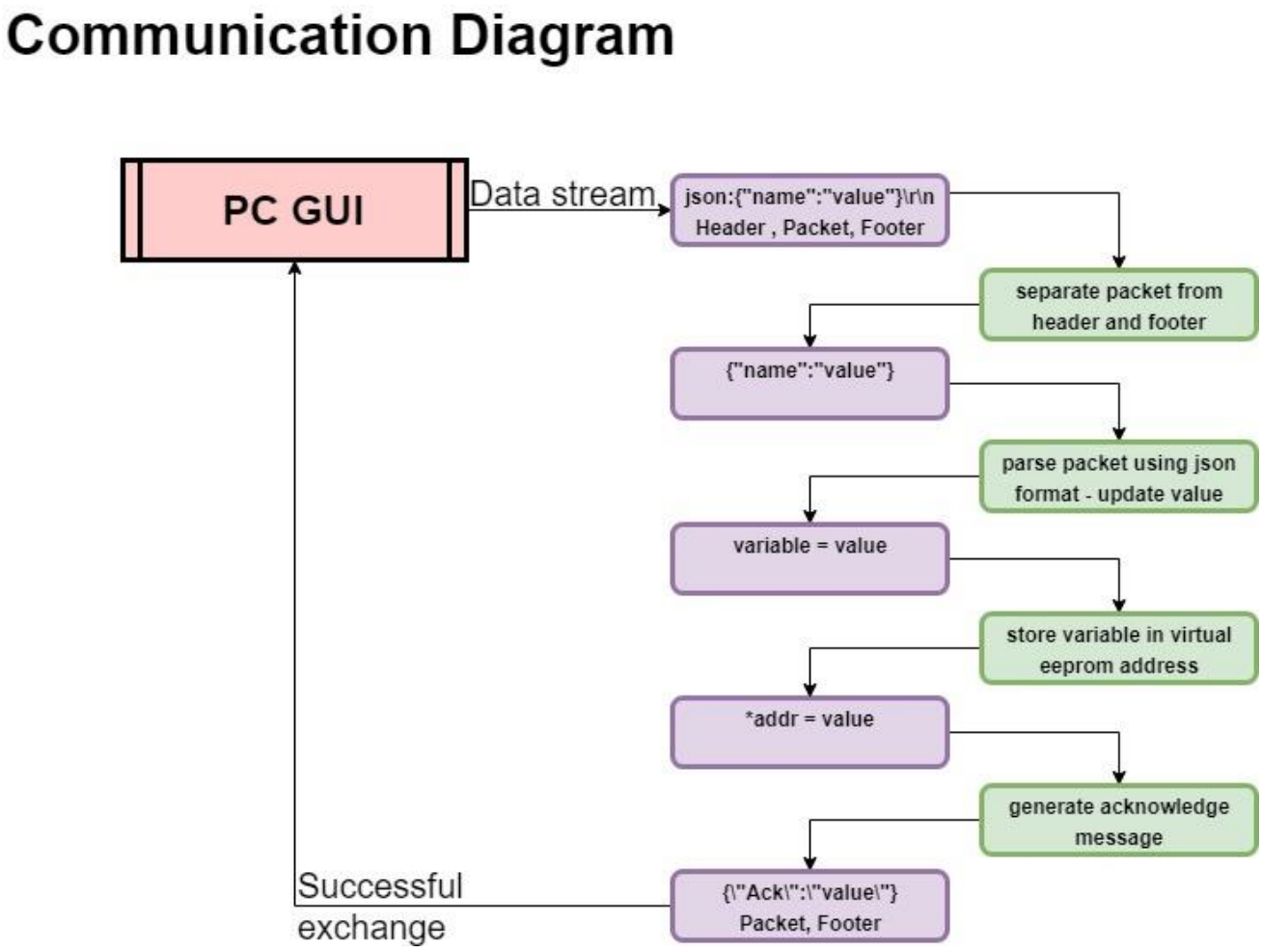

Figure 7. Smart Microphone communication diagram: The implementation of the communication protocols is based on packet handling and json formats.

Recoding of a signal is performed when a recording state is selected in the Matlab GUI. The transmission begins when a countdown for the trigger event is finished. The recorded signal is then transmitter in a binary form (Figure 8). Prior to sending the packet, an update is sent in a json format to notify the Matlab GUI of an upcoming binary packet, the size of the binary packet, and an index for the start of the circular buffer. The Matlab GUI is then switched to the binary packet retrieval state and the MCU sends the circular buffer in a binary format. After the binary packet is transmitted, the MCU sends a json message stating the end of a transmission and by that it acknowledges the Matlab GUI of a successful packet transfer. Using a binary format, reduces significantly the number of bytes required to be transmitted and the overhead for the communication protocol over UART, allowing a continuous capture of bat calls at up to 20 – 30 calls a second at a 3-4 ms duration each.

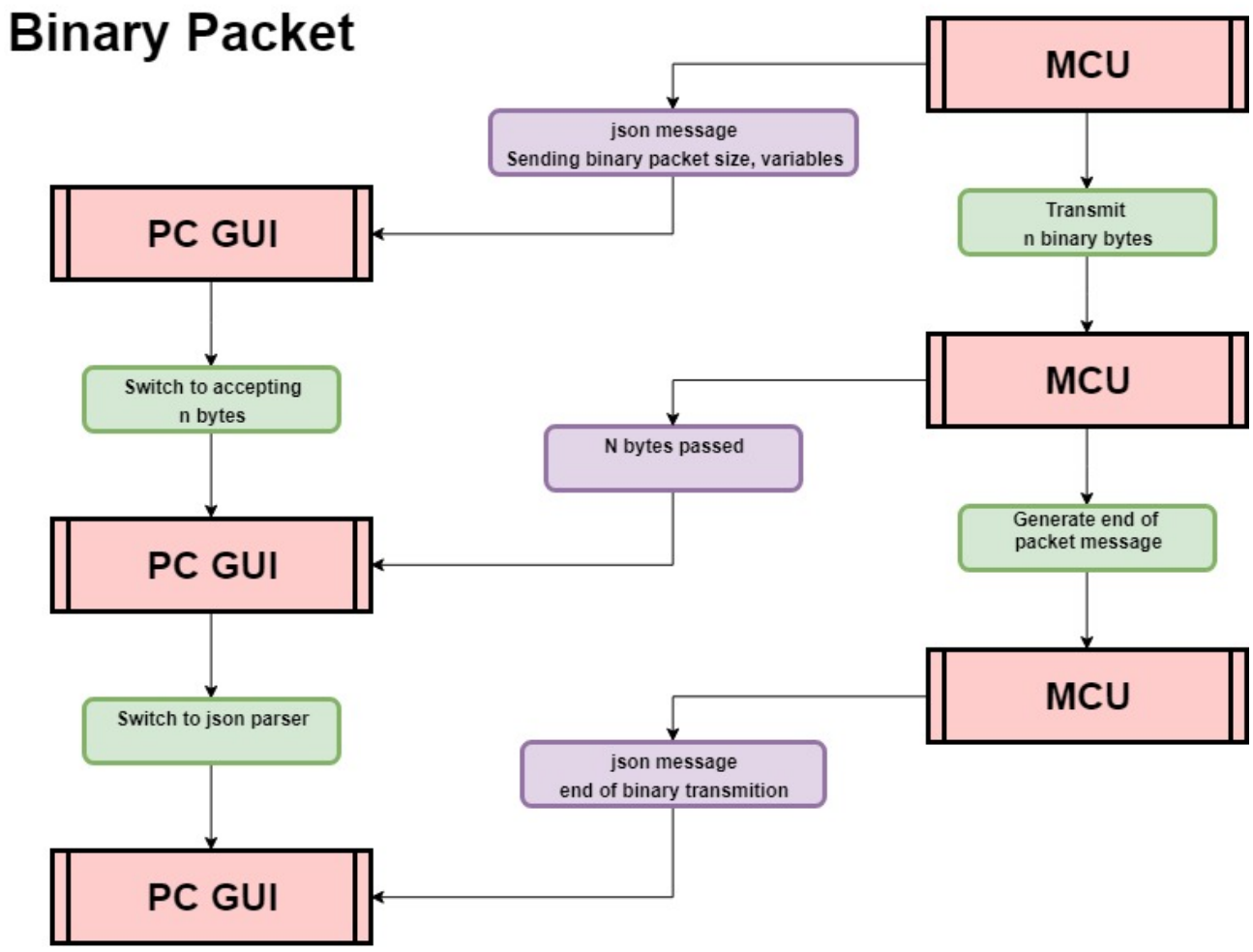

Figure 8. Binary packet transfer: Binary communication protocol allows large files with minimal bandwidth requirements, including an error detection mechanism implemented by a handshake implementation.

1.1.3 Matlab GUI

To control the software parameters of the Smart Microphone, a GUI was created in Matlab (Figure 9). The Microphone has a UART communication over USB with the PC through the MCU`s micro USB connection or through an Xbee adapter board connected to its mate at the PC - as a wireless configuration. The GUI is designed with several section, Connection settings section, Parameters setting and display section, Messages section and Microphone updates section.

The Matlab GUI buttons settings include:

Disconnect / Connect – Disconnect from currently connected USB COM Port. Or connect to the selected COM Port from the Select Port menu.

Update – Refreshes the Select Port menu with the available COM Ports connected to the MCU.

Select Port Menu – Displays the available COM Ports connected to the PC.

BaudRate – Baud Rate for the selected connection, for a Cable connection the configured baud rate is 921600, as for the wireless configuration it is 115200.

ComState – Displays the connection status with the MCU, Green if the connection is active, Red for an inactive connection.

Reset – Resets the GUI configuration and updates the MCU with the default configuration settings.

Sent Message – Displays the last sent messages to the MCU.

Received Message – Displays the last received message from the MCU.

Plot Frame – Displays the received buffer from the MCU.

Get Settings – Retrieves the setting stored at the MCU and updates the GUI configuration to match those settings.

Set Gain – Gain settings to the PGA module, from 1- 128 with binary scale. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Set Trig Tresh – Sets the trigger threshold values from 1-100 in %. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Trig On – Sets the number of samples used to maintain a trigger at high state after trigger event. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

ADC Latency – Sets the number of samples to delay the signal. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Filter On – Changes the Filter Status. Filter On – Green, Filter Off – Gray.

Reset Plot – Resets the Plot window from the accumulated received buffer.

Update All – Update whichever microphone that is connected to the GUI with the current settings displayed by the GUI. Turns Green for successful update, Red for a failed update attempt.

Update Mic 1-5, 10 – Updates the specifically numbered microphone if connected to the GUI with the current settings displayed by the GUI. Turns Green for a successful update, Red for a failed update attempt.

Record On – Sets the Recording state. Record On – Green, Record Off – Gray. When recording is enabled, recorded data transmitted to the GUI will be updated over the Plot window.

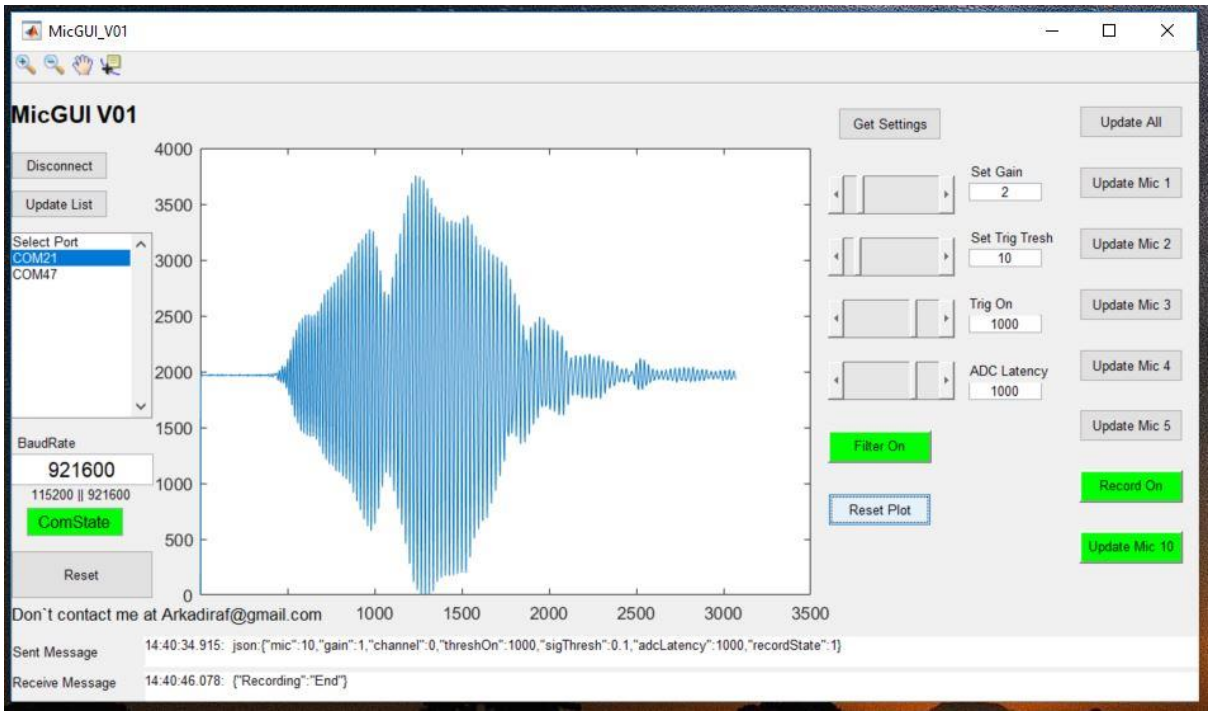

Figure 9. Smart Microphone Matlab GUI: includes various button and slides to manipulate the software behavior of the microphone MCU. With visual representations of the microphone configuration and plot of the acoustic signals as recorded by the Smart microphone MCU.

1.1.4 Casing

The casing for the microphone has been designed in Fusion 360 and 3d printed, Figure 10. The casing has a switchable cover for the microphone in order to test covers which may implement some signal direction. In addition, various adapters have been designed to accommodate various setup options.

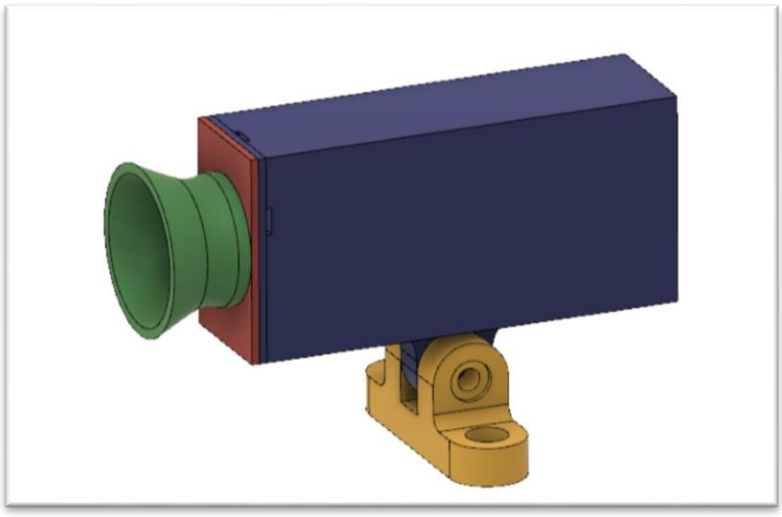

Figure 10. 3d model for the Smart Microphone module: Designed in Fusion 360 for 3d printing fabrications. Simple robust design to protect and ease the use of the electronics.

1.2 Switch module

The Switch Module is used as the center piece of the system. It connects all the Smart microphone modules and Speaker amplifiers, has an analog routing circuitry to route one of the microphone signals to several speakers. In addition, it acts as a power delivery system, converting a single supply source to several voltage levels for the various components of the system. The switch contains several status indicators such as active speaker LEDs, external trigger. Connectors for all the analog signals running through the system and generally configured status LEDs for future configurations. The switch module contains two MCUs, Switch MCU and DSP MCU, the Switch MCU is used as the system interface and responsible for routing the acoustic signals and monitor the trigger events from the Smart Microphones, while the DSP MCU is used as a DSP and implements various signal processing algorithms (Figure 11, Figure 12).

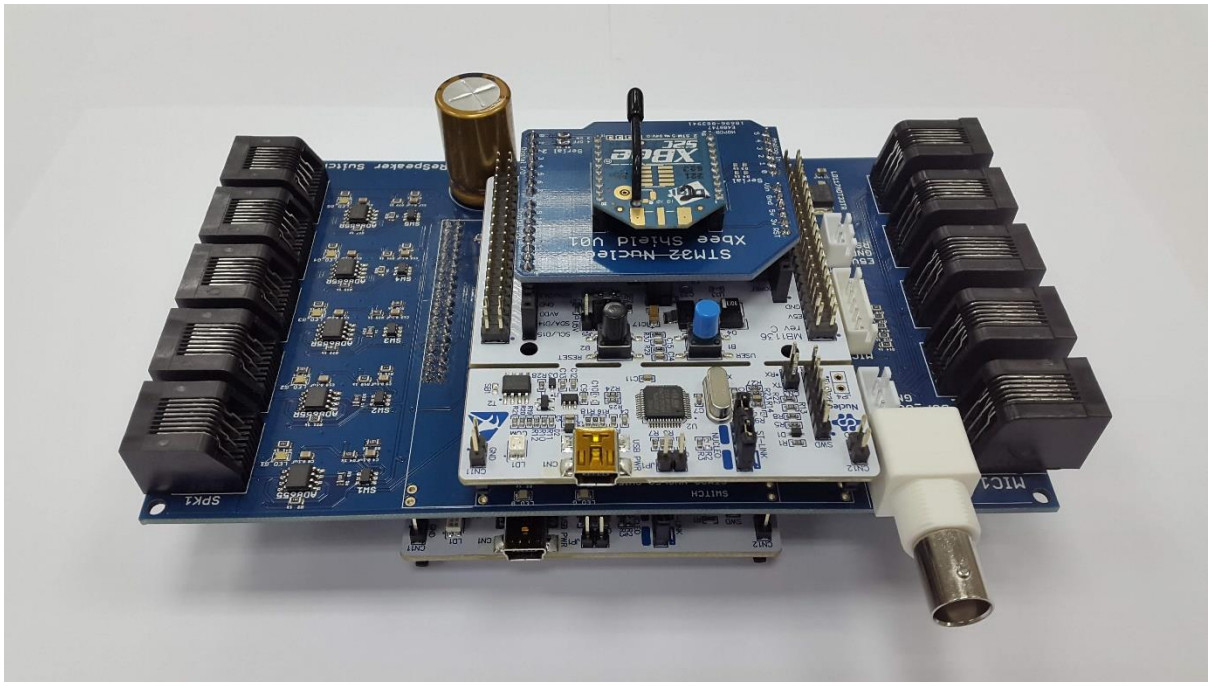

Figure 11. Switch Module: contains 2 MCU, DSP MCU for signal processing, and Switch MCU for signal routing. The base board is design for analog signal routing with several status indicators external triggers and power handling.

1.2.1 Electronics

The Switch Contains several sections on the board. Power delivery, Signal routing, Signal monitoring. The power delivery section is designed to work with supply voltages of 7 - 36V at 5A limit. It has a reverse polarity protection diode, a DC-DC converter module (D24V50F5) to generate 5V from the supply voltage. The generated 5V is used to feed the MCU`s present on the Switch and the connected Smart Microphones. In addition, the 5V is regulated to 3V3 level using a linear regulator (LD117ADT33TR) for better noise rejection. The 3V3 Regulated voltage is used for the analog circuit of the board. In addition, the supply voltage is directed to the Speaker amplifiers through the reverse polarity protection diode.

The acoustic signal is routed using a multiplexer integrated circuit (MUX IC), the MUX (CD74HC4067SM96) is controlled by the Switch MCU, it selects one of the acoustic signals arriving from the connected Smart Microphones and routes the signal to the DSP MCU. The generated signal from the DSP MCU is routed to the analog switches (741G3157DBVR) allowing the selection of multiple speakers for the same input signal. In order to eliminate cross talk between the speaker amplifiers. the signals are buffered by an operations amplifier (OP AMP, AD8655),

The Switch MCU is a Nucleo-STM32F446 it monitors the trigger events arrived from the Smart Microphones. In addition, it has trigger event monitor from the DSP MCU (Nucleo-STM32F446) which is used for DSP frequency loop calculation. Several status LED indicators are present on board, and an external trigger connector is present. Several of the signals are optionally connected to an external hardware for farther monitoring. Such as the Trigger Signal, Smart Microphones analog signal, selected microphone signal and generated signal.

The Switch MCU has an Xbee adapter allowing a wireless connection to the PC and has an internal UART connection with the DSP module for passing messages to and from the DSP MCU and the PC GUI. The externally monitored signals are connected to a SALEAE logic analyzer, capable of logging up to 8 analog signals at 12bit resolution (at -10V to 10V range).

Switch Diagram

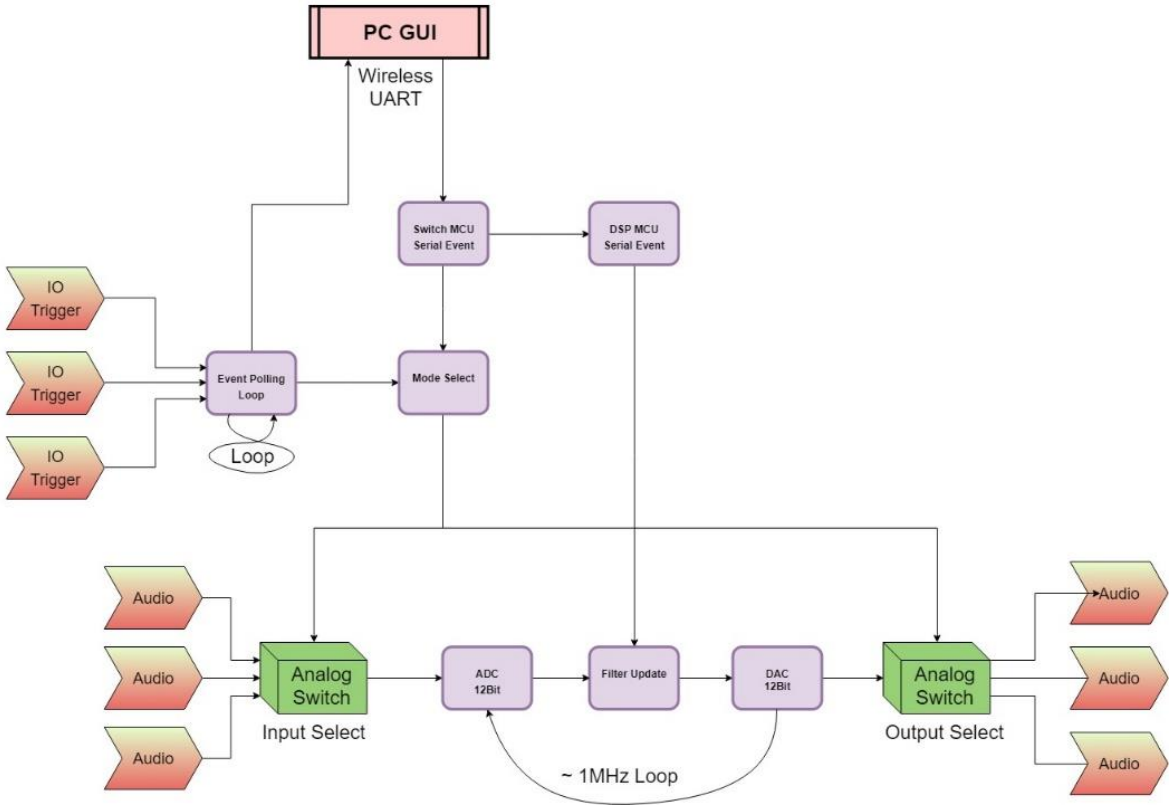

Figure 12. Switch Diagram: illustrates Software & Hardware implementations for the Switch board with both microcontrollers present on the Switch module.

1.2.2 MCU code

The code for the Switch module is implemented through Mbed IDE environment, using C++ language. The Swith MCU (STM32F446) is responsible of the communication with the Matlab GUI and performs signal routing and monitoring. While the DSP MCU (STM32F446) performs the signal processing in real time Figure 12.

The main loop of the DSP MCU performs an analog to digital conversion (ADC) at 12 bits resolution. Following a signal manipulation based on the configured modes:

Off – Sets the acoustic signal to output 0.

Passthrough – Outputs the sample signal without any modification using a 12 bits DAC.

High Pass – Performs a high pass filter using the configured coefficients and the defined order, in the format of IIR Butterworth Filter, implemented as a direct form 1 configuration. Applies a digital gain to the signal and outputs the signal using a 12 bits DAC.

High Pass + Trigger – Performs a high pass filter like the high pass mode and checks for a present bat call based on a simple threshold detection comparison with a pre-defined value. When a successful event is detected a countdown is set with a predefined length which ends with a period of disabling the output signals in order to disable resonance effects caused by the signal echoes with itself. While no event is detected, or the countdown hasn't ended the output signal is generated using a 12 bits DAC.

High Pass + Trigger + Gains - performs similar operations to the High pass + Trigger mode but with a difference of changing the digital gain between signal detection based on a predefine gain vectors.

The main loop frequency is changed by the amount of computation is required at each of the applied modes, the most significant influence is the filter order. The frequency may change as far from 500kHz to 5MHz. The frequency of the loop is measured from the toggle of an event IO shared by the Switch MCU during each 10e6 iteration of an event loop.

The Switch MCU main function is to monitor the signal events supplied by the Smart Microphones and rout the signals as set by the configured mode.

- Manual Operation – Set the MUX by the to the define microphone by the Matlab GUI and enables the defined speaker amplifiers as defined by the Matlab GUI.
- Auto Operation – Switches the MUX automatically from the events triggered by the Smart Microphone and enables the speaker amplifiers by the same event. I.E – for an event from Smart Microphone on channel 1. The signal is routed from channel 1 and speaker amplifier 1 is set to on while the rest are disabled.

In addition, the Switch MCU is responsible to the communication with the Matlab GUI as represented in Figure 13. The communication with the Matlab GUI is done through a UART stream which triggers an interrupt routing in the MCU workflow. The incoming byte is checked for the correct values depending on the defined state of the transmission.

- State 1: compares the incoming byte to the defined header - failed comparison resets the buffer and transmission state
- State 2: stores the incoming bytes to a packet stream.

- State 3: compares the incoming byte to the defined footer - failed comparison resets the buffer and transmission state.

When a successful transmission is performed, and a defined packet byte are parsed in the format of a Jason message. The variables are derived by the variable names and used to configure the Switch operation. The data exchange is completed by sending an acknowledgement message to the Matlab GUI. IF the destination packet is set to the DSP MCU the message is streamed over the shared UART with the DSP MCU and in turn the DSP MCU performs stream detection and packet parsing with acknowledge message generation which in turn is stream back to the Matlab GUI through the Switch MCU.

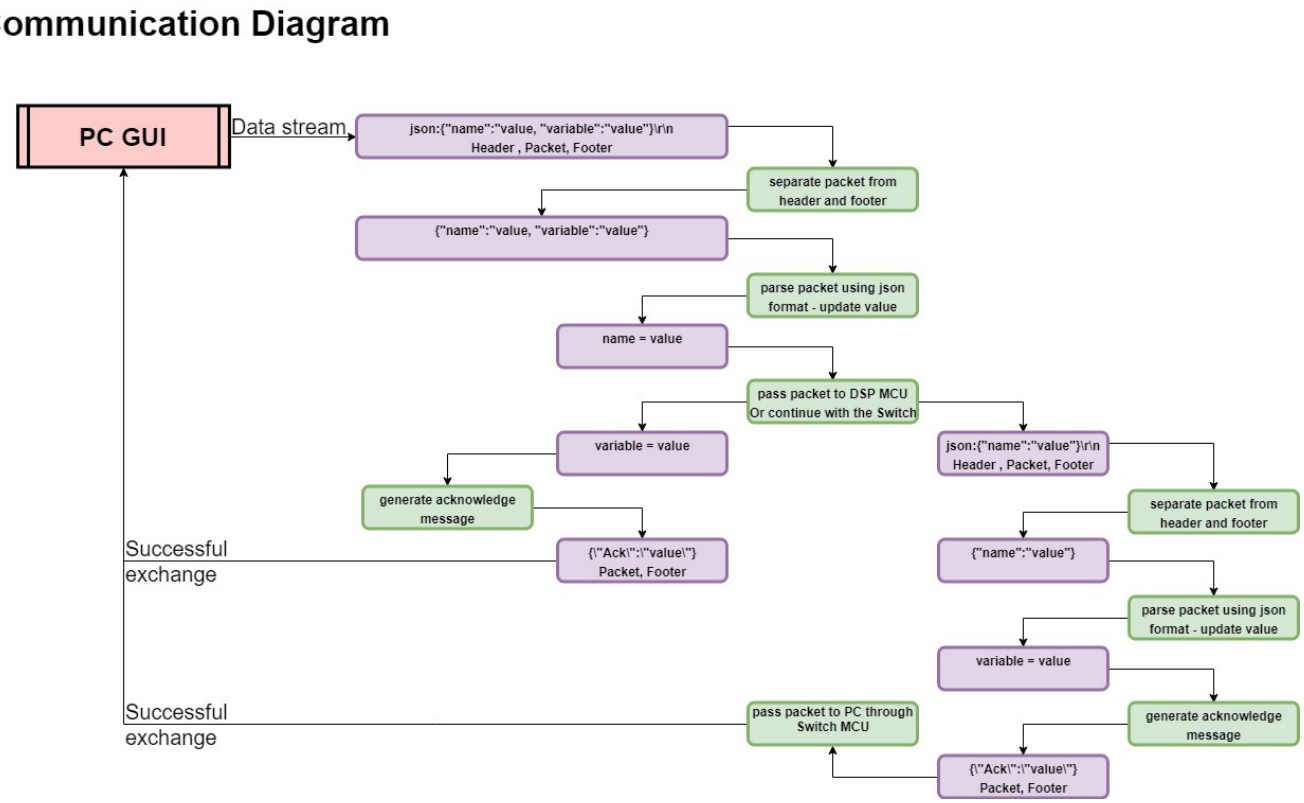

Figure 13. Switch Communication diagram: Illustrates the implemented communication protocol between the Matlab GUI and both microcontrollers. Demonstration of packet handling and routing.

1.2.3 Matlab GUI

To control the software parameters of the Switch a GUI was created in Matlab (Figure 14). The Switch MCU is connected to the GUI using UART communication over Xbee with an Xbee adapter connected to a PC as a wireless configuration. The GUI is designed with several section, Connection settings section, Messages section, Switch control settings section, DSP setting control section and events monitoring section.

The Matlab GUI buttons settings:

Disconnect / Connect – Disconnect from currently connected USB COM Port. Or connect to the selected COM Port from the Select Port menu.

Update – Refreshes the Select Port menu with the available COM Ports connected to the MCU.

Select Port Menu – Displays the available COM Ports connected to the PC.

BaudRate – Baud Rate for the selected connection, default at 57600 for the wireless configuration.

ComState – Displays the connection status with the MCU, Green if the connection is active, Red for an inactive connection.

Reset – Resets the GUI configuration and updates the MCU with the default configuration settings.

Debug Mode – Enables / Disable debug messages.

Sent Message – Displays the last sent messages to the MCU.

Received Message – Displays the last received message from the MCU.

Debug Message – Displays the last received debug message from the MCU.

Microphone Select – Allows manual selection of the microphone, for Autonomous mode the selection automatically updates based on the Automatically selected mode by the MCU.

Speaker Select – Allows manual selection of the enabled speakers, for Autonomous mode the selection automatically updates based on the Automatically selected mode by the MCU.

Mode Select – Allows the selection of the DSP operation mode.

Off Mode – Sets the output signal of the DSP MCU to VCC/2.

Passthrough – Sets the output signal of the DSP MCU to its input.

Highpass – Applies IIR HPF filter to the input signal based on the latest HPF parameters.

HPF + Trigger – Applies IIR HPF filter to the input signal based on the latest HPF parameters and implement signal trigger effect.

Gains + Trigger – Applies IIR HPF filter to the input signal based on the latest HPF parameters and implement signal trigger effect with variable gains between the triggers.

TBD – Future mode.

Loop Freq [kHz] – Displays the measured frequency loop of the DSP MCU

Off On (Loop Freq) – Enables / Disables the measuring of the frequency loop of the DSP MCU

Set Gain – Applies a digital gain to the signal, Valid to modes implementing HPF.

Set Trig Tresh – Sets the trigger threshold detection, valid values 1-100 in %. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Set Trig Pass – Sets the number of samples to pass after trigger event. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Set Trig Pause – Sets the pause time in ms after trigger event implementation. Connected to the slide bar next to it and refreshes the value / slide bar based on whichever is used.

Update Param – Sends an update to the DSP MCU with the updated setting.

Get Settings – Retrieves the settings of the GUI from memory.

Save Settings – Saves the settings of the GUI to memory.

Microphone Events – Displays the received event from the Smart Microphones. Mic column displays the Microphone number which has triggered the event, time column displays the time event for the trigger as measured by the Switch MCU.

Save Log – Saves the microphone events log to memory using a file name written below the button.

Play Sound – Plays a Stored audio file in the DSP MCU memory and generates a trigger event prior playing the file.

Operation Mode – Switches the operation mode of the Switch MCU between Manual and Auto

Filter order – Sets the IIR HPF filter order for the filter design.

Cut Off Freq [kHz] – Sets the IIR HPD filter cut off frequency in kHz for the filter design.

Design Filter – Designs an IIR high pass filter using the defined Filter Order and Cut Off frequency parameters and displays the frequency response plot using the Matlab DSP toolbox. The filter data stored as a SOS matrix.

Update DSP – Sends the latest filter coefficients to the DSP MCU designed by the filter design and stored as a SOS matrix.

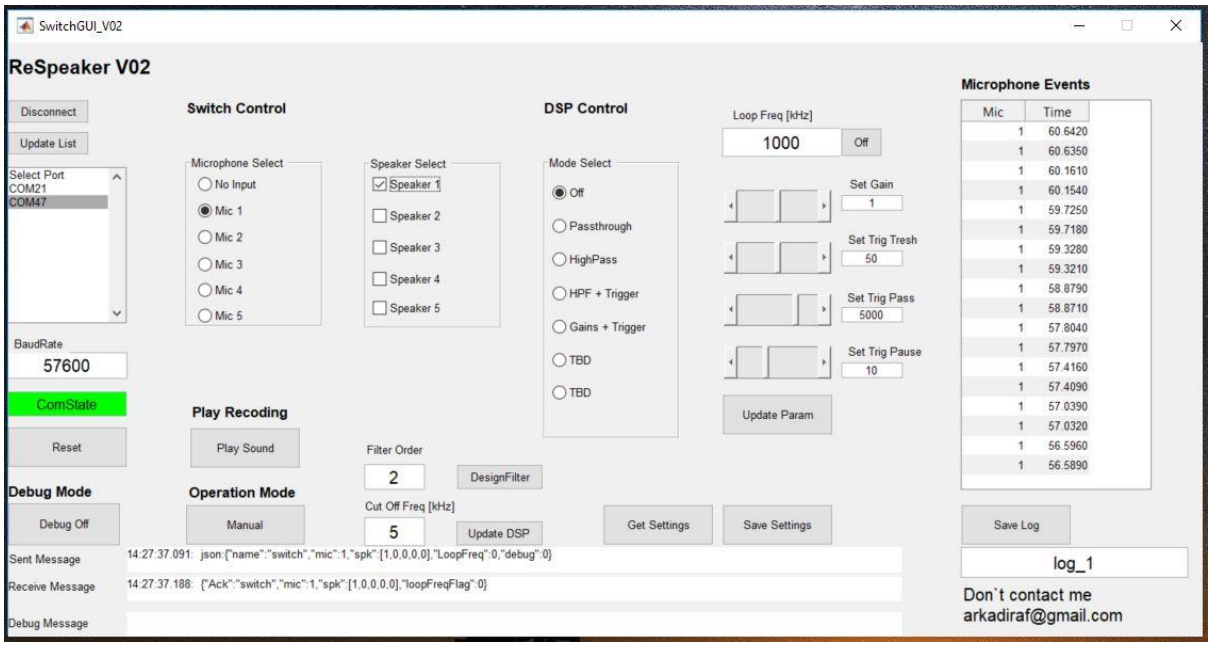

Figure 14. Switch Matlab GUI: Includes various buttons and status indicators. Allows easy manipulation of the software implemented on the Switch microcontrollers.

1.2.4 Casing

The casing for the Switch has been designed in Fusion 360 and 3d printed (Figure 15). The casing has various embedded text in order to indicate important information to the user such as power connection, iteration, connectors, led indicators, flashing ports etc.

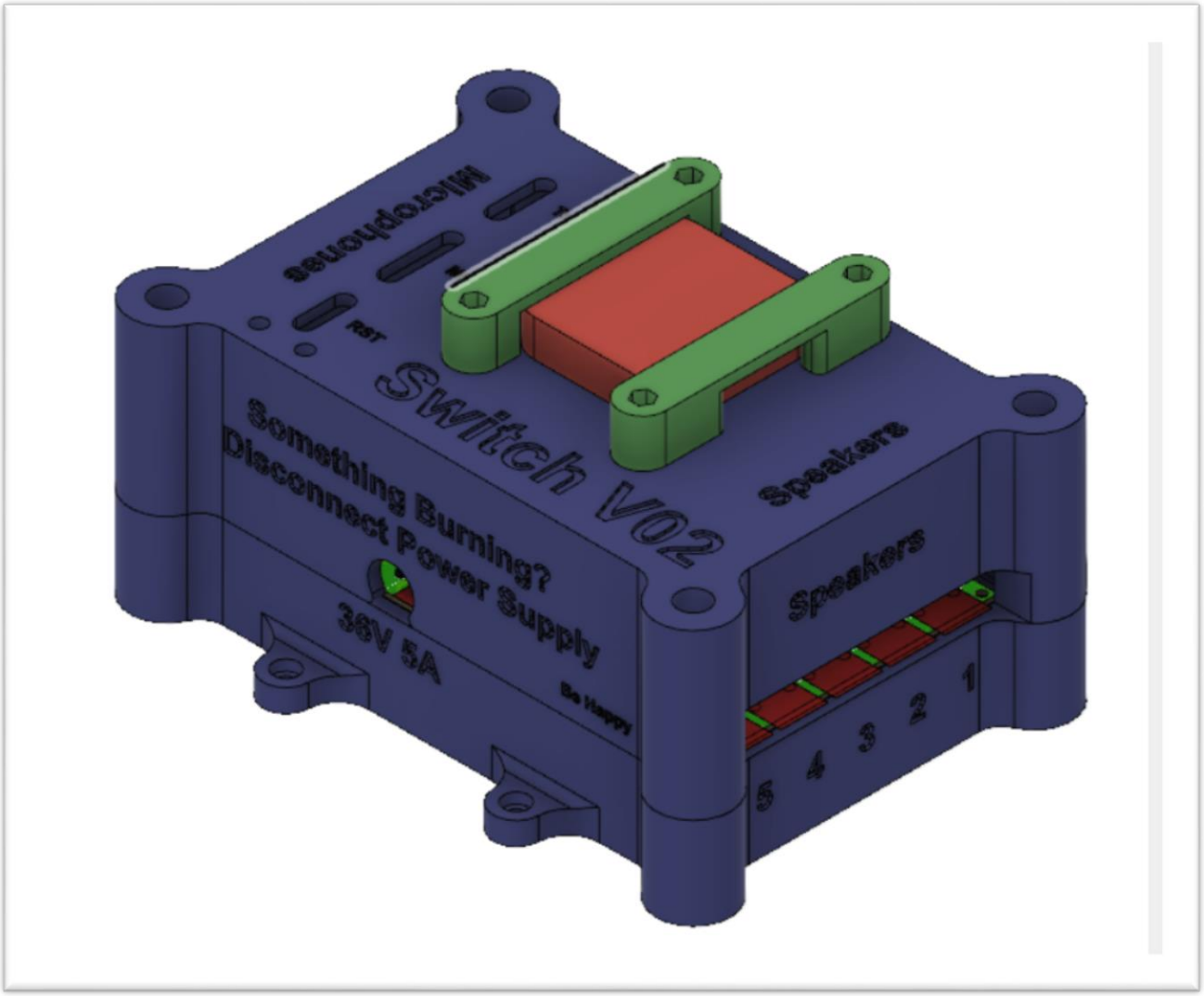

Figure 15. 3D model of the Switch module: Designed in fusion 360 for 3D printing fabrication. The design allows protection and eases the handling of the designated electronics.

1.3 Speaker amplifier

Generating sounds in the ultrasonic frequencies (above 20kHz) requires a dedicated amplifier. Since commercial amplifiers are usually designed for the human hearing range and are commonly implemented through power efficient amplifiers (Class D). These speakers work as digital switches at high frequencies above the human hearing thresholds. The high frequency digital switching is converted to an analog signal using the natural low pass filtering characteristics of the speaker coils. The attempt to use such an amplifier at an ultrasonic frequency will result in high frequency distortions if any sound is generated at all since the common amplifiers are implemented with filters to cut the ultrasonic ranges from the input signals.

To overcome these constrains, a speaker amplifier module has been developed (Figure 16). The speaker amplifier board contains a power amplifier circuit designed for high frequency audio signals at the range of 1-100 kHz which supports up to 36V and 3A. It is designed around a Class A/B amplifier (Analog power amplifier) for a 4ohm tweeter speaker such as a ScanSpeak R2004 or a Tymphany XT25SC90 speaker.

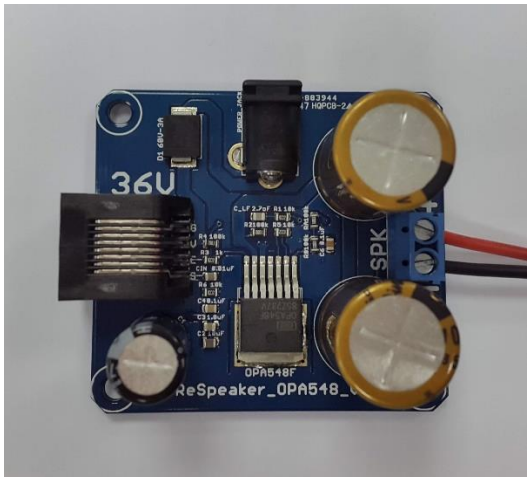

Figure 16. Speaker amplifier module: designed for single supply and meets the required bandwidth of the system.

1.3.1 Electronics

The speaker amplifier module is designed around the OPA548F power amplifier, which has a bandwidth of 1MHz, and can work at up to 5A. The amplifier implementation is for a single supply operation, valid for 5-36V supply voltages, the power can be supplied through a Power jack present on board or through the RJ45 cable connected to the Switch. The circuit has a constant gain setting implemented as an inverting amplifier and is set based on the desired operation. In addition, a current limit is set to 3A, and the circuit has an enable optioned which is controlled by the switch module.

1.3.2 Casing

The casing for the Speaker and speaker amplifier has been designed in Fusion 360 and 3d printed (Figure 17). The Speaker casing has a removable cover in order to implement sound directionality. In addition, various adapters have been designed to accommodate various setup options.

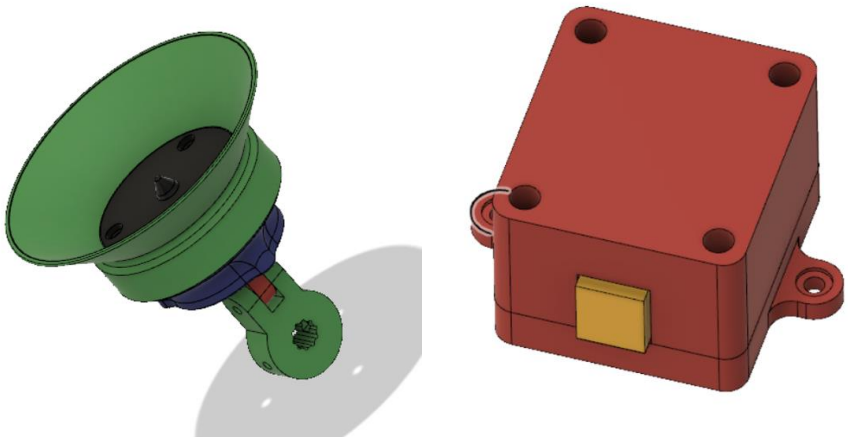

Figure 17. Speaker and speaker amplifier casings: designed in fusion 360 for 3d fabrication. The casing protects the circuit boards and allows mounting of various 3d printed acoustic focus – cone element.

2 Results

To evaluate the performance of the system and test its fitness to work with bats, a series of experiments were conducted. Ex.1 Evaluates the system`s frequency response. While Ex.2 though Ex.7 Evaluates various modes of operations. To conduct the experiments two experimental setups were established. One is a simplified setup consisting of 2 microphones and one speaker (Figure 18); and the other is a full-scale setup consisting of 5 microphones and 5 speakers (Figure 19). For the experiments, the system was powered by a Lithium polymer battery pack at 11V (3s Battery). The system's output signals were recorded by a Saleae logic analyzer. The Saleae logic analyzer was set to initiate recordings by a trigger pulse from the Switch MCU at a recording frequency 3.125MHz. The Saleae Logic analyzer recorded the 5 Microphone channels, the DSP MCU input and output and the trigger event channel from the Switch MCU.

In addition to the system setup, a Chirper box was used. The Chirper box, which was designed for this project, is a simplified version of the playback system, based on one MCU (STM32F466) connected to an amplifier, with a state button to set the output on and off, and a potentiometer to set the repeatability of the recorded signal on the MCU flash memory. The chirper box acts as a virtual bat box allowing the simulation of a bat call without using an actual bat for the performance test experiments. For the Chirper box, two recordings were prepared, one of a bat call recorded by a GRAS calibrated microphone, and the other is a simulated chirp signal consisting a frequency sweep from 1-100kHz. The chirper box is powered by an external Power bank battery for mobility, allowing to freely move it in the room.

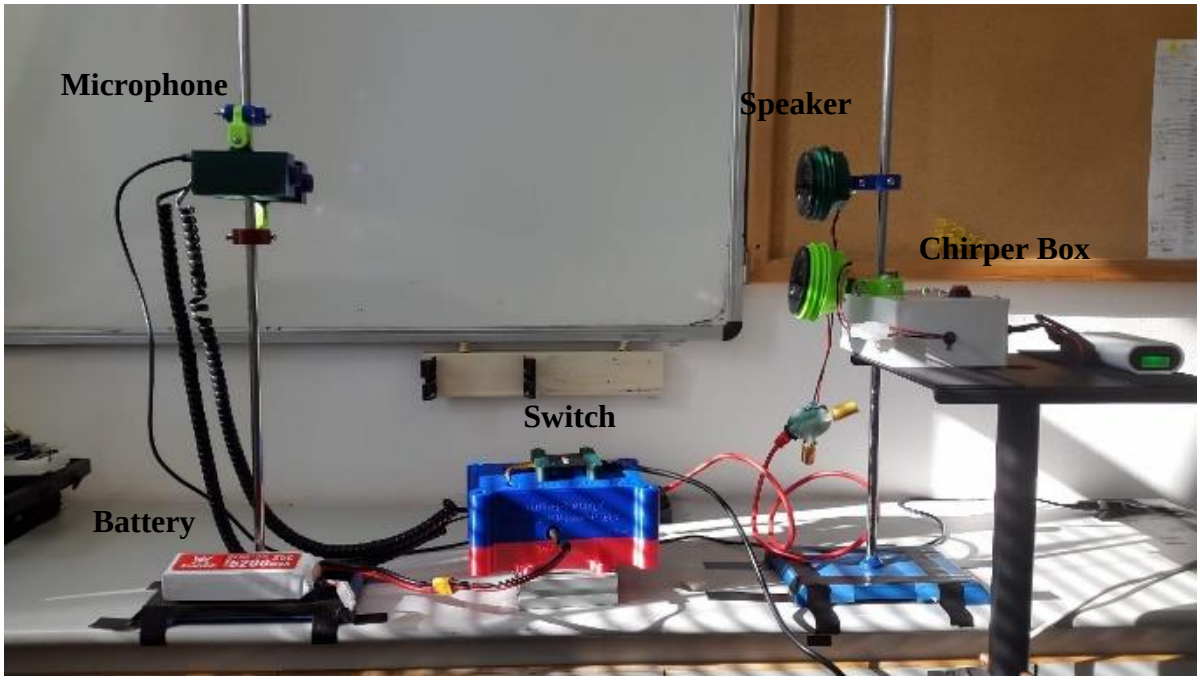

Figure 18. Experiments setup 1: 2 microphones on the left connected to a switch box in the middle, with 1 speaker module on the right. Battery powered setup. On the right a Chirper box. Right figure:

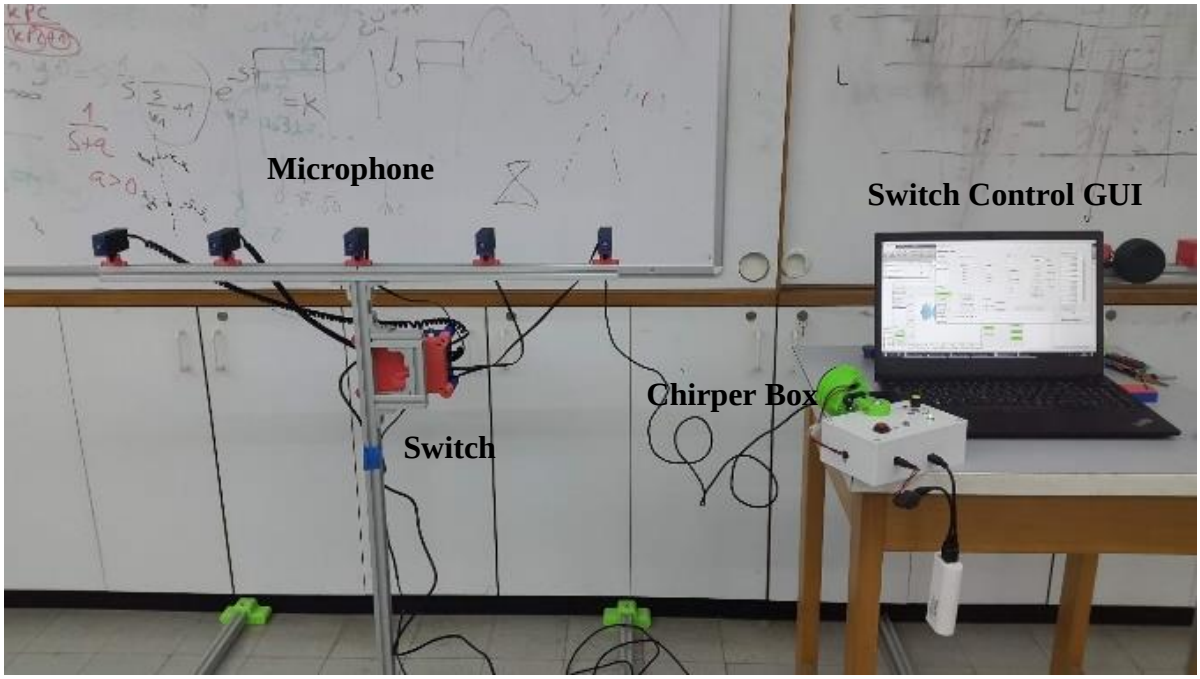

FIGURE 19. Experiment setup 2: 5 microphone modules mounted on a linear rail and connected to a switch box, a laptop running a microphone control GUI, Switch control GUI, Saleae logic analyzer recording GUI. Chirper Box.

2.1 Exp. 1 – The system's frequency response

In order to measure the system's frequency response, the microphone was positioned directly in front of the system`s speaker at 0.5m. The Smart Microphone gain was configured to 4 v/v (12dB) in order to get the maximum resolution of the ADC and the microphone threshold detection was set to 50%. A linear chirp signal was generated from the systems DSP MCU memory ranging from 1kHz to 100kHz and back to 1 kHz with a trapezoid windowing for the 5% of the signal. The generated chirp to the speaker and the received signal from the microphones were recorded by the Saleae logic analyzer. The experiment was conducted once with the analog HPF module of the microphone enabled and once with the HPF disabled (Figure 20).

In order to retrieve the system's frequency response, the signals from the microphone and the generated signal to the speaker were aligned by compensating for the time delay generated by the signal's propagation in air, between the speaker and the microphone. The time delay was retrieved using a synchronization pulse generated in the beginning of the chirp, and by cross correlating this pulse with the received pulse. The system's frequency response was estimated using "tfestimate" function in Matlab on the aligned data set.

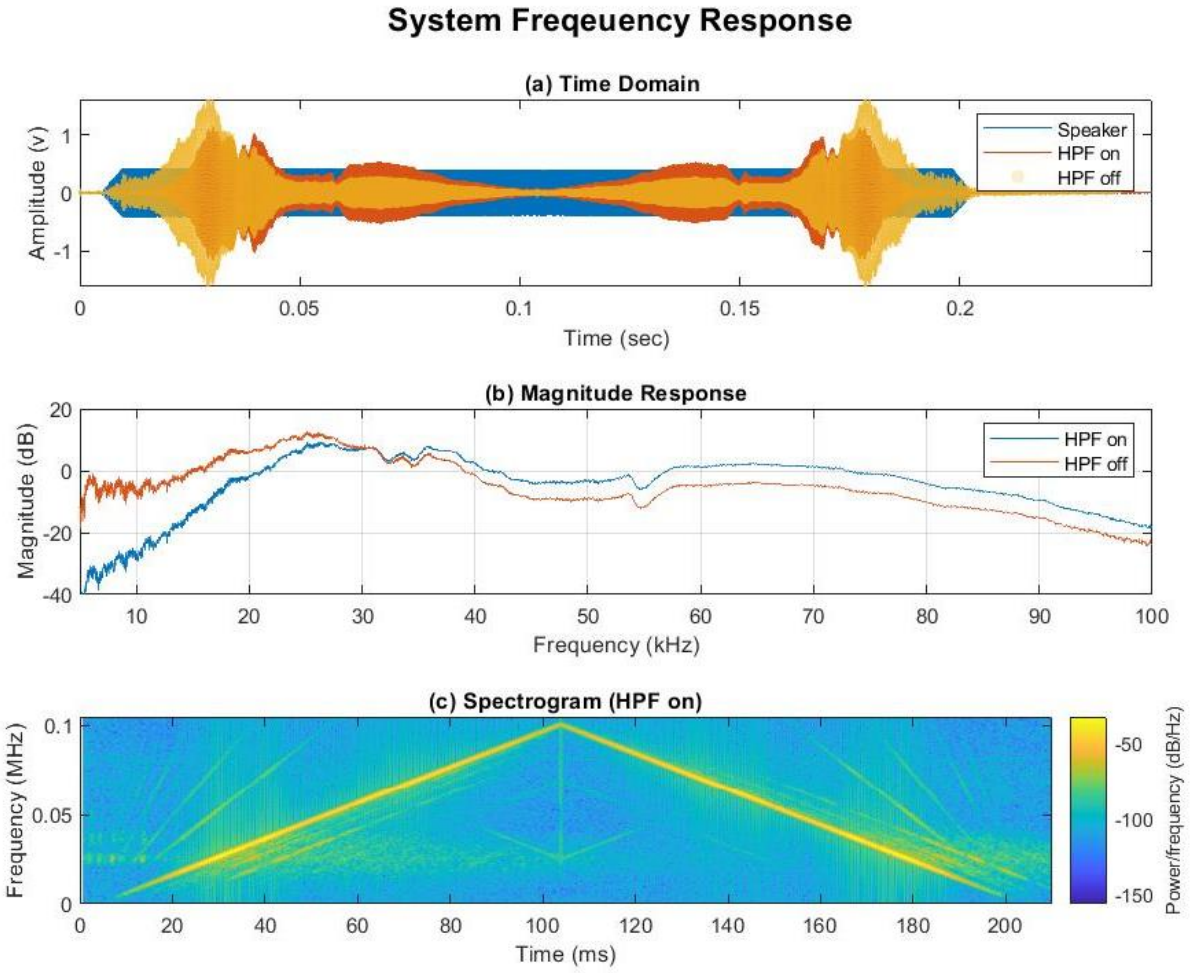

Figure 20. System's frequency response, microphone switch and speaker: (a) Frequency response in time domain, (b) Magnitude response of the system, (c) Spectrogram showing

that the system has some harmonic distortions (frequencies at double and higher than the main frequency but at a much lower density), In addition it has an echo signal, probably derived from a close by wall, since the echoes have the same delay on the rising side of the chirp and the decreasing side.

The results reveal that the implemented analog high pass filter module significantly improves the frequency response for the 25kHz – 50kHz range, while filtering out the frequencies lower than 20kHz. In addition, from frequency response curve it is noted that the HPF module adds approximately 6dB gain corresponding to the design parameters of the module as a second order high pass filter at a cutoff frequency of 48kHz with a damping factor of 0.5.

2.2 Exp. 2 – The switch pass-through mode

The simplest operation mode for the DSP MCU is passing the signal from its input to its output without any modifications to the signal and with minimum latency. The setup used for this experiment is the same as in the frequency response experiment.

Figure 21 presents the recorded signals for the pass-through mode, the output has a barely noticeable delay compared to the input, at the order of 1uS. The delay is generated by the analog characteristics of the microcontrollers ADC / DAC. Corresponding to its frequency response of up to 1Mhz.

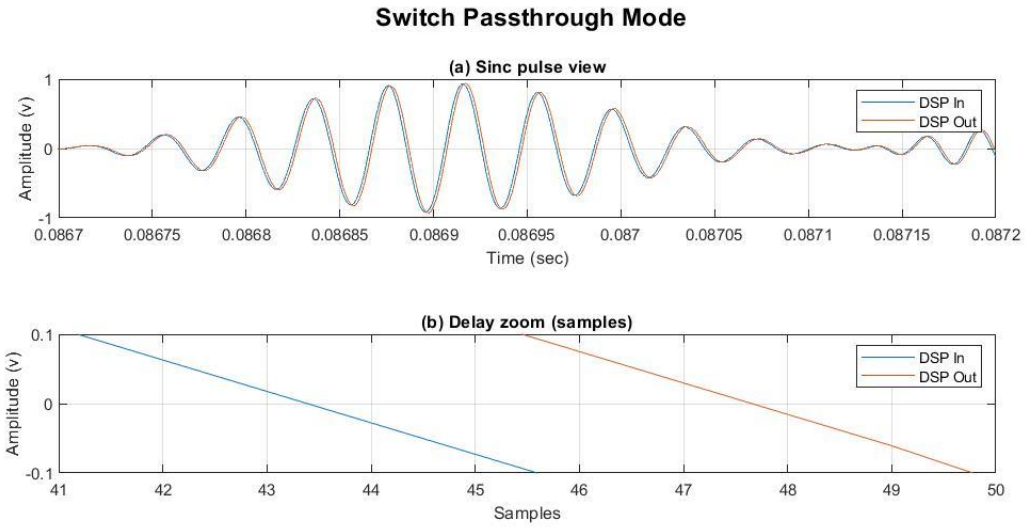

Figure 21. Switch Pass-through Mode Experiment: (a) Sinc pulse, demonstrates that the generated signal is the same as the input signal. (b) Delay zoom, shows the induced latency, which is accumulated to 4 samples of the Saleae logic analyzer which is set to 3.15 MHz sample rate. And correspond to an order of 1uS latency for the system.

2.3 Exp. 3 – Generating delays

In order to simulate an object at various distances for the same setup position an optional software delay to the acoustic signal was implemented using the microphone's MCU. For a bat such a delay would be equivalent to moving the object backwards. The maximal delay depends on the memory size of the MCU, for the STM32F303 the memory limit is at about 3ms. To demonstrate this capability, both Smart microphone modules were recorded using the setup shown in Figure 18. The Smart Microphone gain was configured to 2 v/v (6dB), the HPF module was set to on and the microphone threshold detection set to 50%. The acoustic signal was generated by the Chirp Box. The signal was a recorded signal of an actual bat call, as recorded by a GRAS microphone (PD40, GRAS). Both microphones are recorded simultaneously while each has a different delay setting.

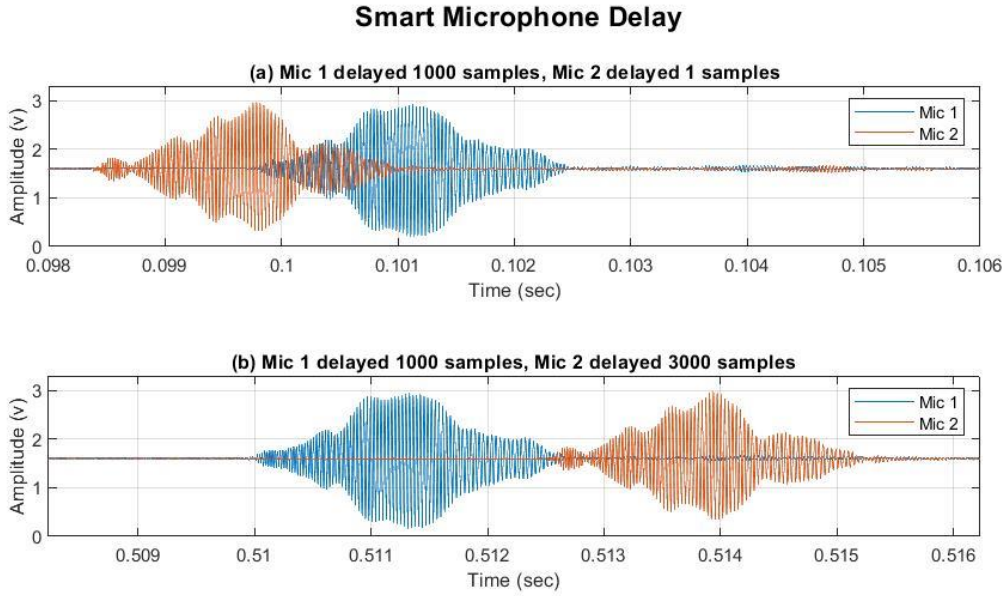

Figure 22. **Smart Microphone delay experiments**: (a) Mic 1 Delayed, (b) Mic 2 Delayed. The plots demonstrate the influence of the software delay induced for each microphone and how one signal can be delay compared to the other with the appropriate settings described in the titles for the graphs.

An additional use for the delay feature is to delay the acoustic signal generated by the microphone while the Switch MCU switches between the microphone sources, allowing fluent transition between sources. This is accomplished since the Smart microphone that generates a trigger when a signal is present, but the trigger is generated only after the signal level passes a pre-defined threshold those missing a fraction of the beginning of the signal. By delaying the signal, with an appropriate amount. The trigger will be generated before the actual acoustic signal has reached the Switch.

2.4 Exp. 4 – Testing the switch filter mode

To test the bats' perception and response to various effects, it is required to implement real time signal processing's to the signal. One of the most basic implementations is a standard digital filter. In this experiment the performance of a high pass filter was evaluated. For optimized computational power the filter is implemented as an infinite impulse response filter (IIR) using the direct form 1 configuration. Several Butterworth high pass filters were tested at various orders.

The Switch filter mode is implemented on the DSP MCU. In this mode the MCU performs a high pass filter using the configured coefficients and the defined order by the Matlab GUI. The MCU samples the signal using an analog to digital converter (ADC) at 12bit resolution, applies the predefined filter and adds a digital gain afterwards it outputs the signal using embedded hardware digital to analog converter (DAC) of the microcontroller at a resolution of 12 bits. The setup used for this experiment is the same as in the system frequency response experiment. Figure 23 presents two recordings of the HPF filter mode. One for an HPF at 50kHz with 2 order, and the other is a 50kHz with 8 order.

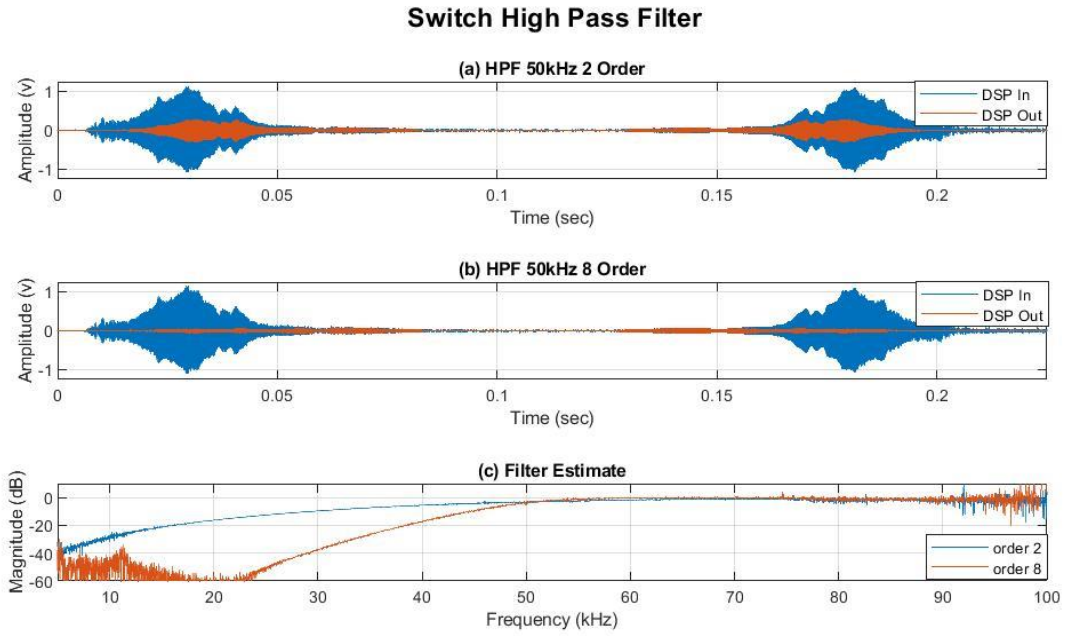

Figure 23. Switch High Pass Filter Mode: (a) HPF 2 order filter, (b) HPF 8 order filter, (c) Filter estimation. The figure demonstrates the estimated frequency response for the implemented filters derived from the recorded data, using the "tfestimate" function in Matlab and they correlate to the implanted filters for the DSP MCU.

In addition to evaluating the filter implementation the filter computation load was also tested. The sampling frequency of the setup is directly influenced by the number of calculations implemented at every operating mode and since implementing high order filters requires more operations to implement the sampling frequency decreases for higher filter orders, the values are summed up in **Table 1.**.

Filter order	Sampling frequency (kHz)
2	1135
4	885
6	725
8	614
10	533

Table 1. Filter order vs Sampling frequency for Filter Mode experiment: The higher the filter order the lower the sampling frequency.

2.5 Exp. 5 – Testing the trigger HPF mode

One of the most difficult issues with the playback system is the direct feedback from the speakers to the microphones. Since the system transmits the acoustic signals received by the microphone in real time, a positive feedback loop may occur, resulting in self-resonance at the resonance frequencies of the system. From the frequency response of the system and actual resonance experiments those frequencies are at the range of 25kHz-35kHz. Depending on the speaker angle and the microphone filter implementation.

In order to overcome this challenge a trigger mode has been implemented. The trigger mode acts as a simple signal presence detector based on a threshold detection and it silences the system after the signal has been received and transmitted using a countdown technique – after a signal presence is detected a countdown is initiated and sets the number of samples to be passed before the system silences the channel. The number of samples to be transmitted after a signal is detected is set by the Trigg pass value. After the countdown is completed a silencing period is implemented at the length of the 'Trig pause' value. The silencing effect causes any residue acoustic signals to dissipate and reduces the chances for the system resonating. In addition to the trigger detection, this mode implements an HPF using the same coefficients used in the HPF mode.

In order to test the performance of this mode the speaker was positioned directly in front of the microphone (Figure 18) which is the worst configuration possible because the microphone picks up any emission from the speaker. Bat sound signals were generated by the chirper box, and the signals recorded by the Saleae logic. The results are seen in Figure 24. The top plot demonstrates how the system enters self-resonance just from background noise even with a short silencing period of 10ms, the middle plot demonstrates how extending the silencing period to 20ms fixes this issue. In the bottom plot it is possible to see how the system starts resonating from a direct feedback loop.

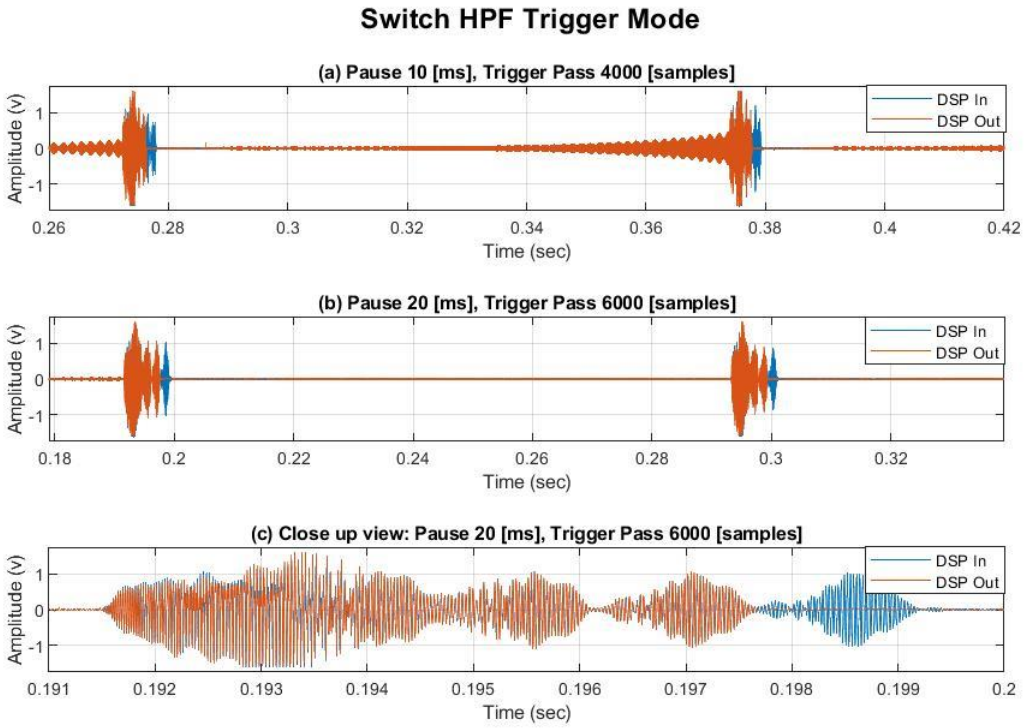

Figure 24. Switch HPF Trigger Mode, Resonance handling: (a) HPF Trigger mode with the ms pause and 4000 samples to pass, notable the self-resonance generated by background noise. (b) HPF Trigger mode with 20 ms pause and 6000 samples to pass, the self-resonance is disabled. (c) Zoom on the signal of HPF Trigger mode with 20 ms pause and 6000 samples pass. The repetitive pulse length directly correlated to the distance from the speaker to the microphone. Which is approximately 0.5m, and the repetitive pulse is approximately 1.5ms which is at the speed of sound of 343m/s covers 0.51m. In addition, the figure demonstrates the trig pass implementation and that the microphone has recorded the last generated pulse after the system has been silenced – I.e. the signal that has covered the distance between the speaker and the microphone.

2.6 Exp. 6 – Trigger mode gain adjustments

In order to simulated variable echo gains which in nature can be generated by the flapping wings of a butterfly. The Trigger mode gains was implemented. The Trigger mode gains is an extension mode to the Trigger mode, in addition to the implemented in the Trigger mode, this mode adds a variable gain to the detected signals. The gains are set by a predefined value and switched after each detected signal. The results are seen in Figure 25.

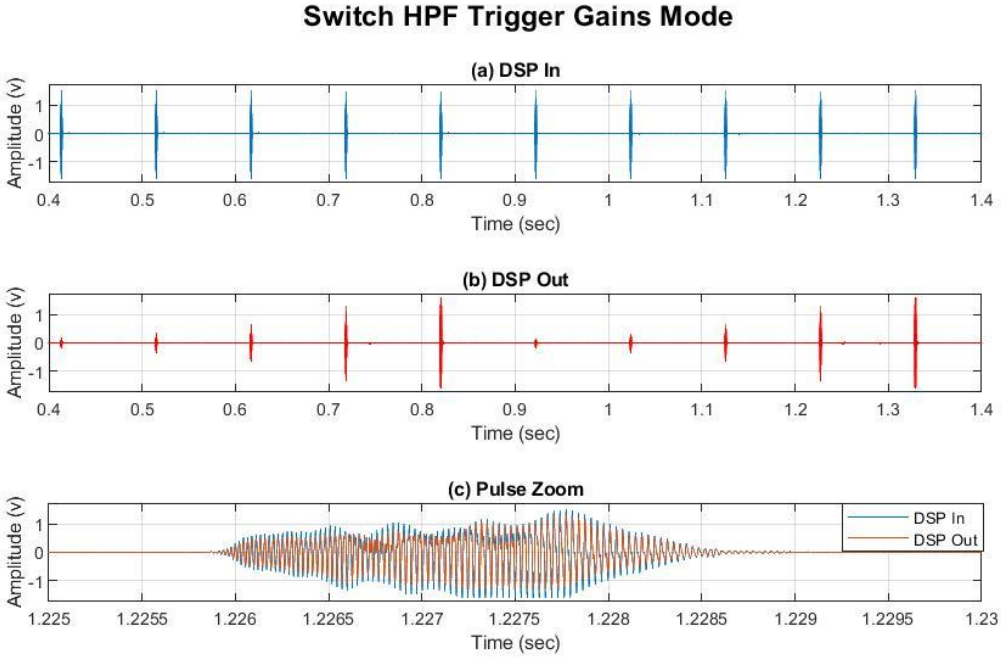

Figure 25. Switch HPF Trigger Gains Mode: (a) DSP in Signal, (b) DSP Out signal - notable the variable gains for the pulses recorded at (a) , (c) Zoom on a single pulse with a slightly lower gain.

2.7 Exp. 7 – Auto mode

One of the most innovative features of the system is the ability to redirect sound from various channels to several speakers in real time and automatically. This capability can be used to generate obstacles near the bat by detecting the closest source to the bat. In addition, this mode can be used to generate a virtual predator by generating a “chasing” effect.

To demonstrate this capability, the Auto Mode was implemented and tested using the experiment setup 2 described in Figure 18, The Chirper box was moved in front of a microphone array to a distance of 1 meter, playing a prerecorded bat call stored in its memory and at a repetition of 10 pulses per second. The received signals from each microphone were logged by a Saleae logic including the trigger pulses. The results of the experiment are demonstrated in Figure 26. The figure shows how the system switches automatically from the nearest microphone and generates a sound wave in real time.

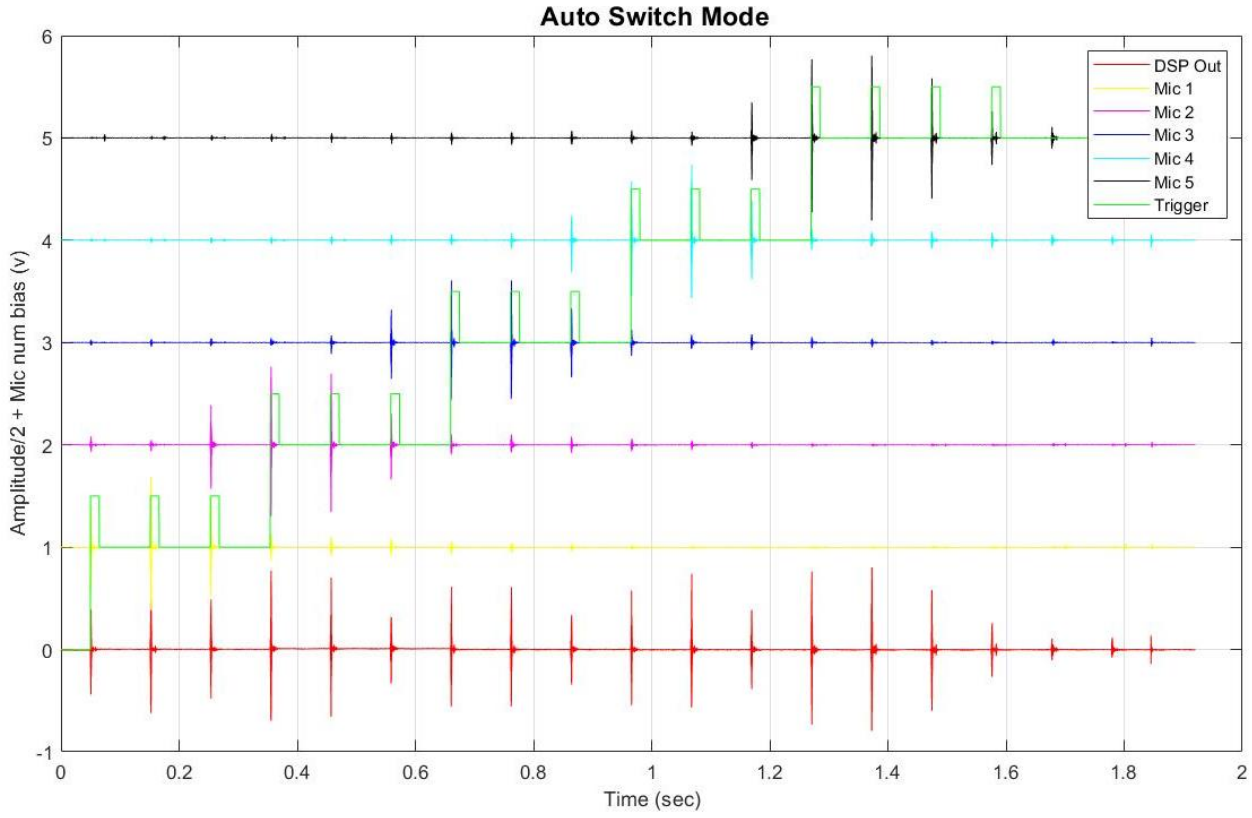

Figure 26. Auto switch mode: the plot demonstrates the signals for each microphone, the generated signal and the trigger pulse generated by the microphone corresponding to the microphone number as logged by the Switch GUI. Each trigger the signal is rerouted from the triggered microphone to a corresponding speaker.

The ability to switch between microphones in real time is mainly achieved by the system’s architecture. Each of the microphones contains its own microprocessor which is responsible for its own signal detection. When a microphone detects a signal, it generates a trigger which the Switch MCU receives and processes. Since a trigger is a simplified representation of a signal presence the workload for the Switch MCU is very low and it can handle multiple

microphones simultaneously. The generated pulses are recorded by the Switch Matlab GUI and presented in **Table 2**.

Trigger	Mic	Time (sec)
1	1	379.71
2	1	379.813
3	1	379.916
4	2	380.019
5	2	380.122
6	2	380.226
7	3	380.329
8	3	380.432
9	3	380.535
10	4	380.639
11	4	380.742
12	4	380.845
13	5	380.948
14	5	381.051
15	5	381.155
16	5	381.258
17	5	381.465
18	5	381.532

Table 2 Generated table by the Switch GUI: Records the Trigger number, Trigger source and time stamp of the trigger.

3 Future directions

Although the system has been developed extensively tested thoroughly and implemented in actual experiments, a lot of improvements are still optional, which can induce interesting experiments and demonstration.

- Upgrade to Differential signaling – The use of a differential signaling can significantly improve noise rejection in the wiring between modules as opposed to the single ended implementation.
- Applying an improved external signal recording oscilloscope for a better resolution ADC
- Upgrading the DSP MCU to a better ADC resolution and higher performance such as the NUCLEO-H743ZI which has 2.5 the current DSP performance, and able to run at 480 MHz, with better ADC. The new version MCU also incorporates a high-speed USB at 480 Mbit, which can allow real time recordings of several microphone signals.
- Upgrading Smart Microphone MCU to a newer better version such as the NUCLEO-G431KB which has 2.5 the current MCU performance, and able to run at 170 MHz, with hardware IIR filtering which can offload some of the MCU work load.
- Adding a PGA module to the output signal will allow better analog performance to the generated signals.
- General improvements to the software.
- Adding real time signal triangulation for position estimation and enabling position-based interactions.
- Adding an optional DMA mode. Although using a DMA implements some delay to the signal, the use will allow better ADC DAC timings and release resources for more complex signal processing.
- Adding an analog switch for the Smart Microphone module to switch between signal sources – analog signal from microphone or the generated signals from the MCU DAC.
- Implementing better signal detection – based on bandpass filtering or cross correlation to prerecorded signals.

4 Appendix A

Smart Microphone schematics

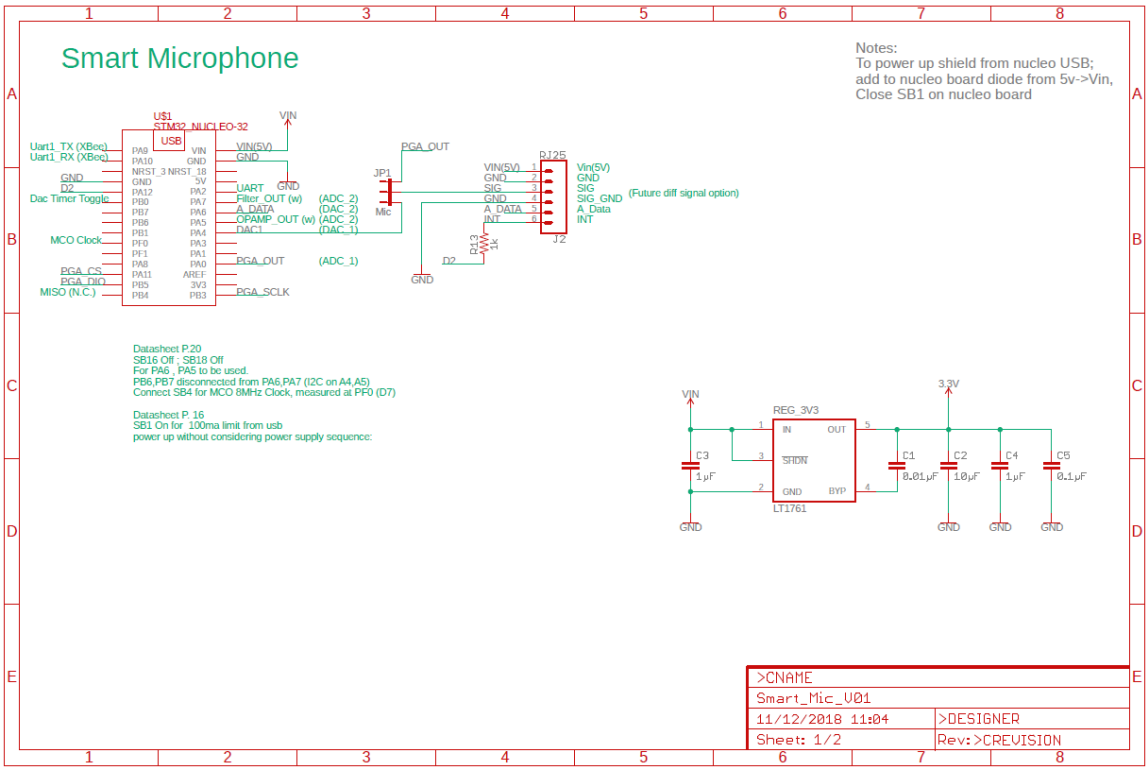

Figure 27. Smart Microphone schematics 1/2

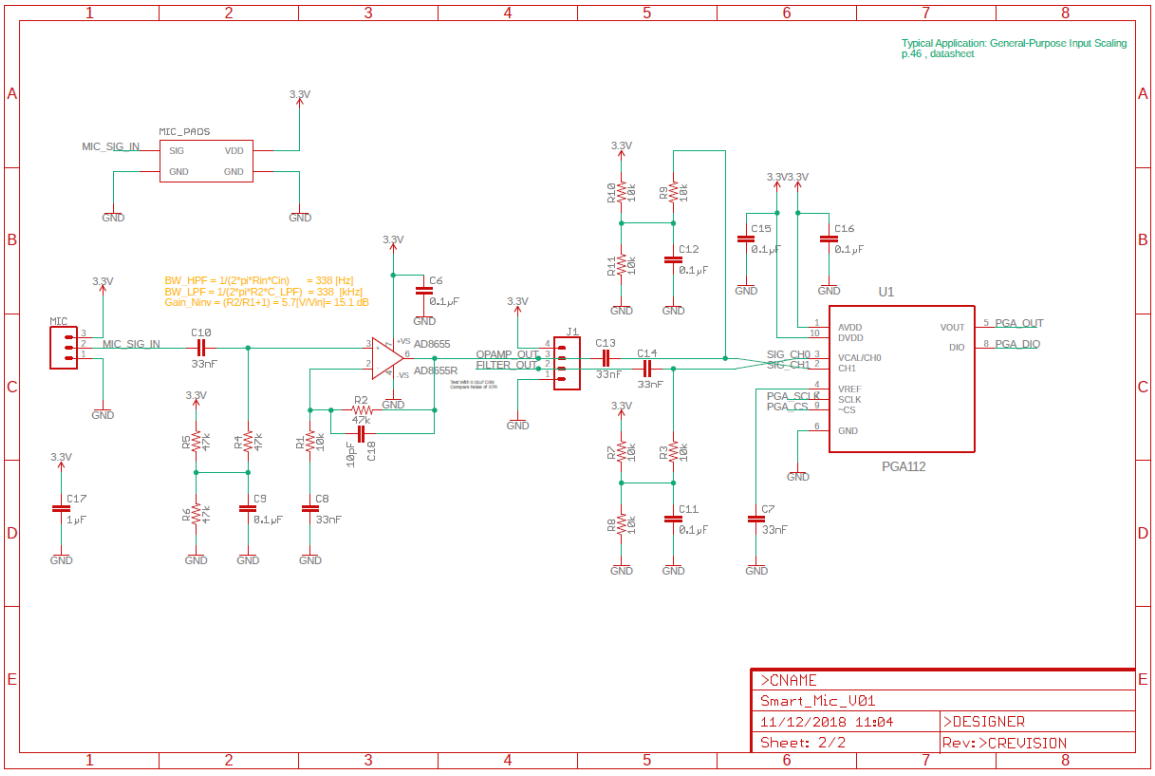

Figure 28. Smart Microphone schematics 2/2

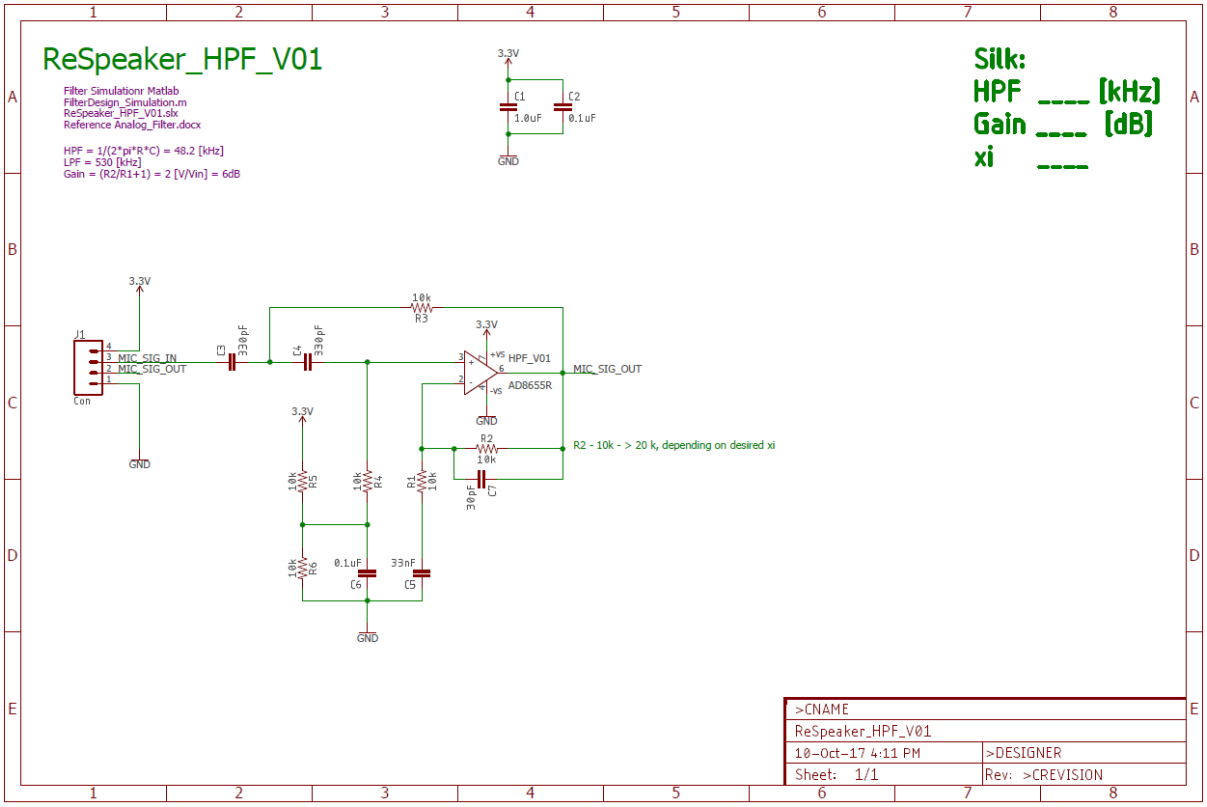

Figure 29. HPF Module

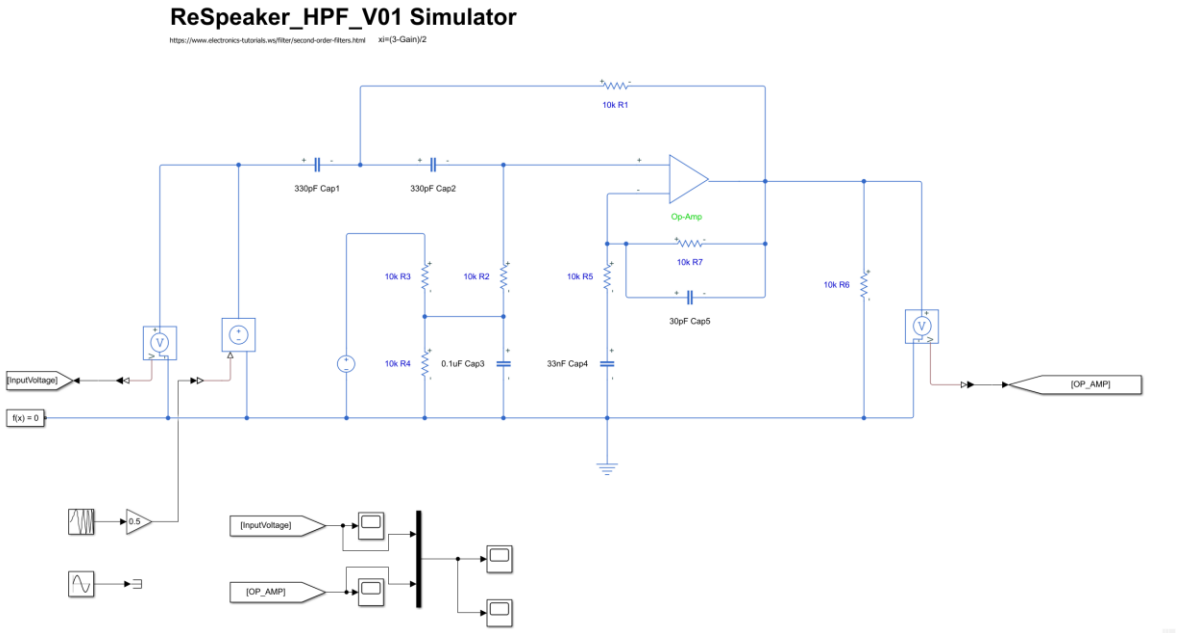

Figure 30. HPF Module simulation

Switch schematics

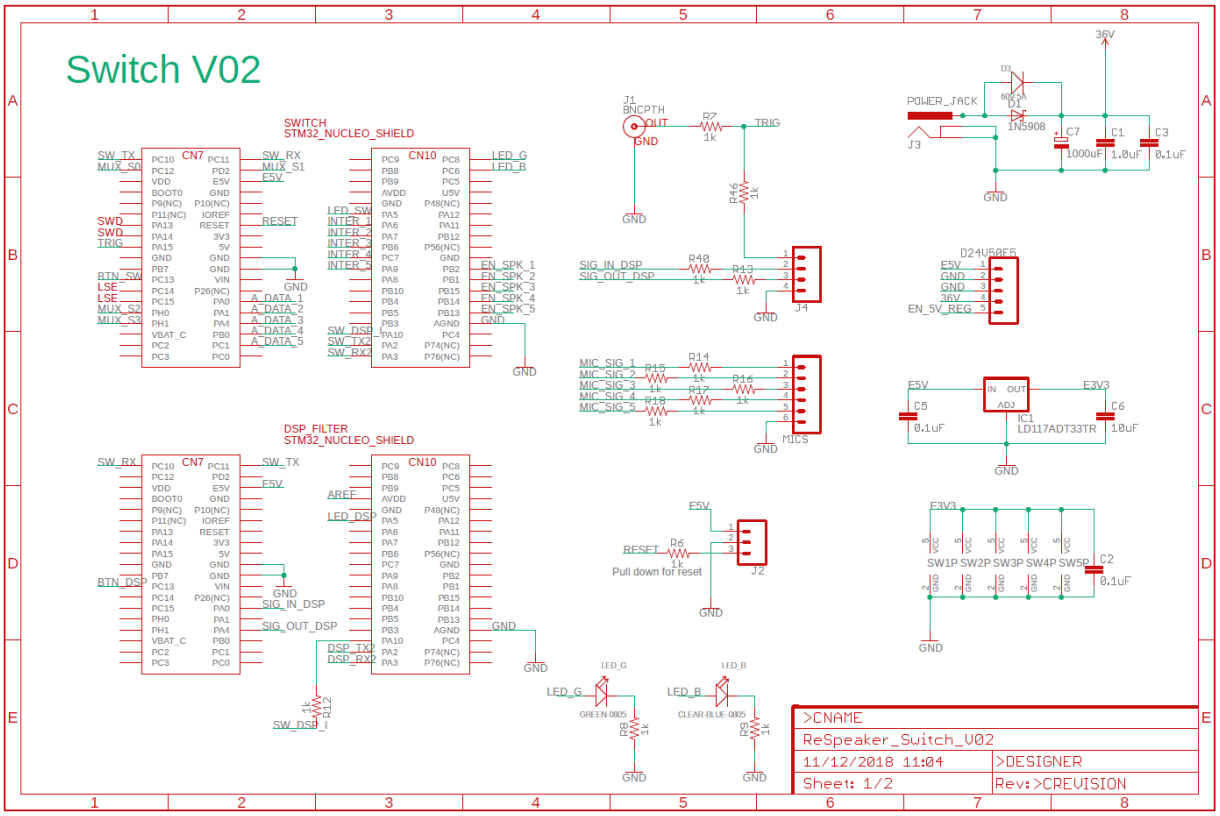

Figure 31. Switch schematics 1/2

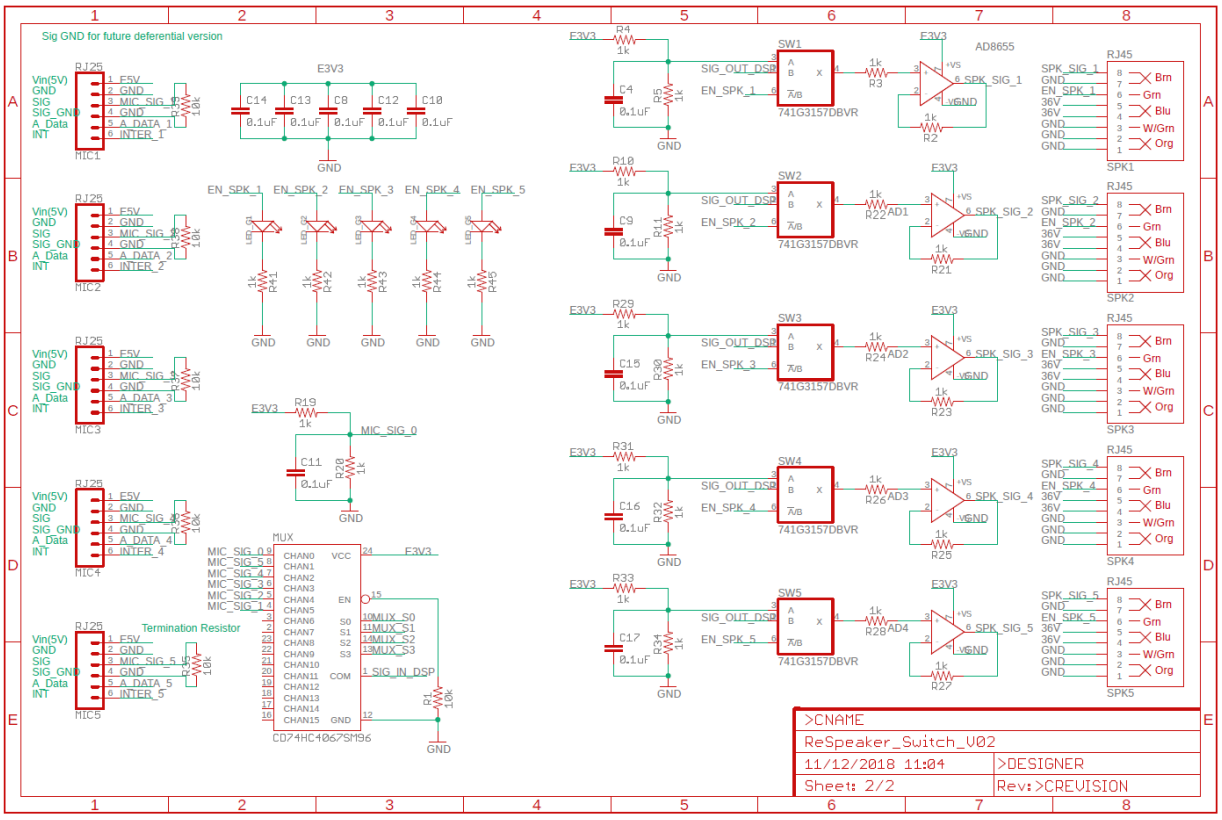

Figure 32. Switch schematics 2/2

Speaker amplifier schematics

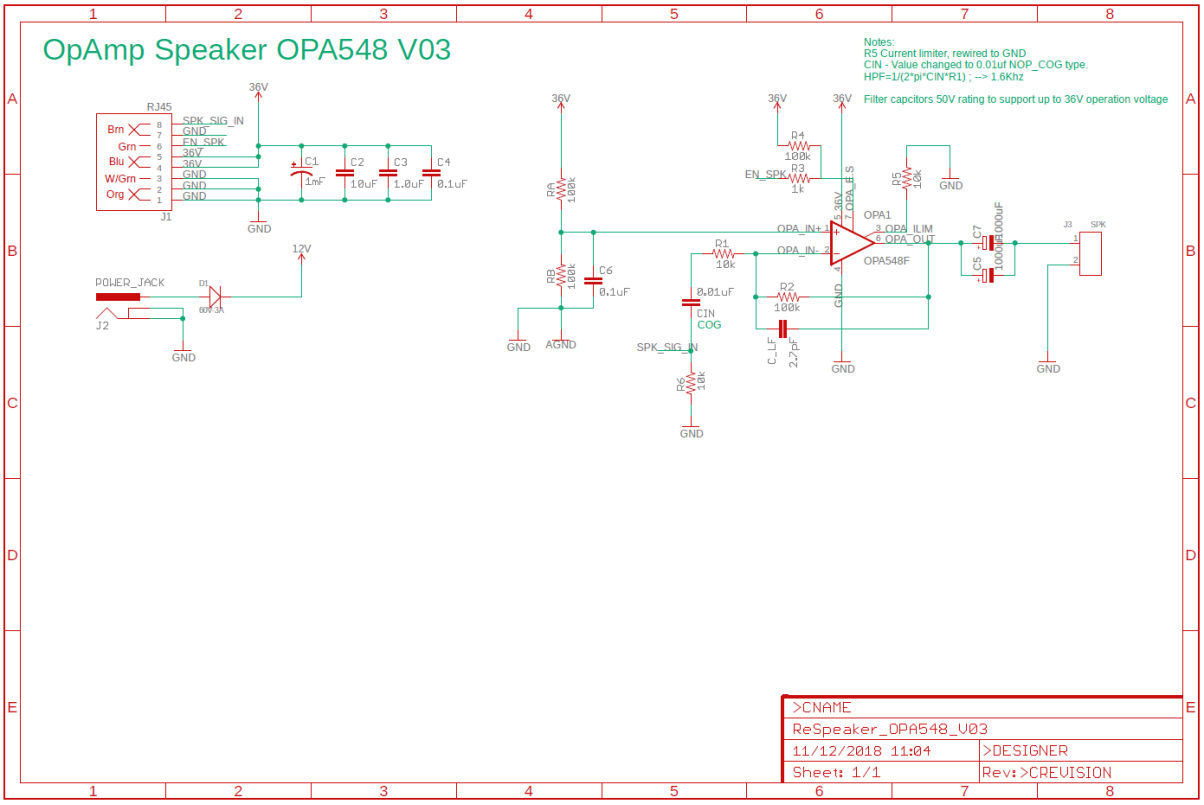

Figure 33. Speaker amplifier schematics