CECS 460 Spring 2023 Project #1

Audio Processing

Team Members	**Student ID#**
Ethan Dixon	025861054
Eddie Hernandez	017205368
Abhishek Jasti	016784623
Anand Jasti	026851342

Table of Contents

Description

For this project we are tasked with designing a SOC that is able to perform multiple audio processing tasks, such as equalization, inputting audio and outputting processed audio, and volume control. To perform these tasks, we have installed external volume knobs with potentiometers, with the black knob controlling the volume of the board and the silver knob controlling the voltage the board takes. The board can also directly take audio from the user and repeat it through the speaker, with the user able to control the audio and equalization of the audio repeating to the user. The SOC will also include multiple modules to perform multiple tasks, modules such as a processor core, memory, and audio processing modules. The processor core is in charge of executing instructions and performing tasks, the memory is used to store data, such as when the user inputs audio for the board to store. Lastly, there are multiple audio processing modules, in charge of adjusting the frequency and volume of the system.

Code Analysis

The given tutorial for this project was not the best for us to get started with the Zybo z7-10 board, so we looked online for other tutorials that might help us. We found one that was very helpful for the development of this project. We haven't changed any C code from the tutorial we found online Zybo z7-10 DMA Audio Demo Vivado 2018.2. The only thing we changed from this tutorial code was changing the number of seconds to record/playback from five seconds to ten seconds.

```
// Audio constants
// Number of seconds to record/playback
#define NR_SEC_TO_REC_PLAY      10
```

Verilog Code

```
module volume_control(
    input wire [31:0] tdata,
    input wire [3:0] tkeep,
    input wire tlast,
    input wire tvalid,
    input wire tready,
    input wire [15:0] volume_control,
    input [1:0] switch,

    output reg [31:0] tdata_out,
    output wire [3:0] tkeep_out,
    output wire tlast_out,
    output wire tvalid_out,
    output wire tready_out

    );

//    reg [31:0] scaled_audio;

    assign tkeep_out = tkeep;
    assign tlast_out = tlast;
    assign tvalid_out = tvalid;
    assign tready_out = tready;


    always @(switch) begin
        case(switch)
            2'b00: tdata_out = tdata* volume_control[15:10];
            2'b01: tdata_out = (tdata[11:0]* volume_control[15:10]) + tdata[31:12];
            2'b10: tdata_out = (tdata[23:12]*volume_control[15:10]) + tdata[31:24] + tdata[11:0];
            2'b11: tdata_out = (tdata[31:24]* volume_control[15:10]) + tdata[23:0];
        endcase
    end

endmodule
```

This was the code we wrote for volume control and equalization.

```
22
23  module adc_led(
24      input clk,
25      input [15:0] dout,
26      input drdy,
27      output led
28      );
29
30      reg [1:0] _drdy = 0;
31      reg [7:0] data0 = 0;
32
33      reg [7:0] pwm_count; // shared pwm counter
34      reg [7:0] pwm_duty0; // duty cycles for the 4 pwm led brightness controllers
35
36      always@(posedge clk)
37          _drdy <= {_drdy[0], drdy};
38
39      always@(posedge clk) begin
40          if (_drdy == 2'b10) begin // on negative edge
41              data0 <= dout[15:8];
42          end
43      end
44
45      always@(posedge clk)
46          pwm_count <= pwm_count + 1;
47
48      always@(posedge clk)
49          if (pwm_count == 0) begin
50              pwm_duty0 <= data0;
51          end
52
53      assign led = (pwm_count <= pwm_duty0) ? 1 : 0;
54
55
56  endmodule
57
```

This is the code we wrote to show the level of volume control on the onboard LED.

We also added the XADC IP into the block design to take in the analog input. All the other modules/IP's in this project were created following the tutorial [Zybo z7-10 DMA Audio Demo Vivado 2018.2.]

Project Difficulties

For this project we are using a board that we had never used before, the Zybo-z7-10, and apart from Vivado we are also using a new software program, the Xilinx SDK. The Audio Zync Tutorial to set up an example project of basic audio recording and playback only got us so far since the step-by-step guide was missing the audio and audiotest header files and .c files. However, even though the tutorial was incomplete it was still helpful for the basic setup of the board and with Vivado and the SDK. That includes various useful links to useful downloads, incorporating the IPs needed, creating the block design wrapper, and using the Xilinx SDK. We found another tutorial that was a little more useful. After finishing this tutorial we were able to record a ten-second audio either from a mic input or a line in input and play back the recorded audio. This tutorial was very difficult to follow and took a long time to finish. After finishing the tutorial we had trouble figuring out where or how we could implement the volume control module, but after some more research, we figured it out.

What each Individual did

Ethan Dixon has worked on the project report and helped with the creation of the project and getting set up with the new Zybo z7-10 board.

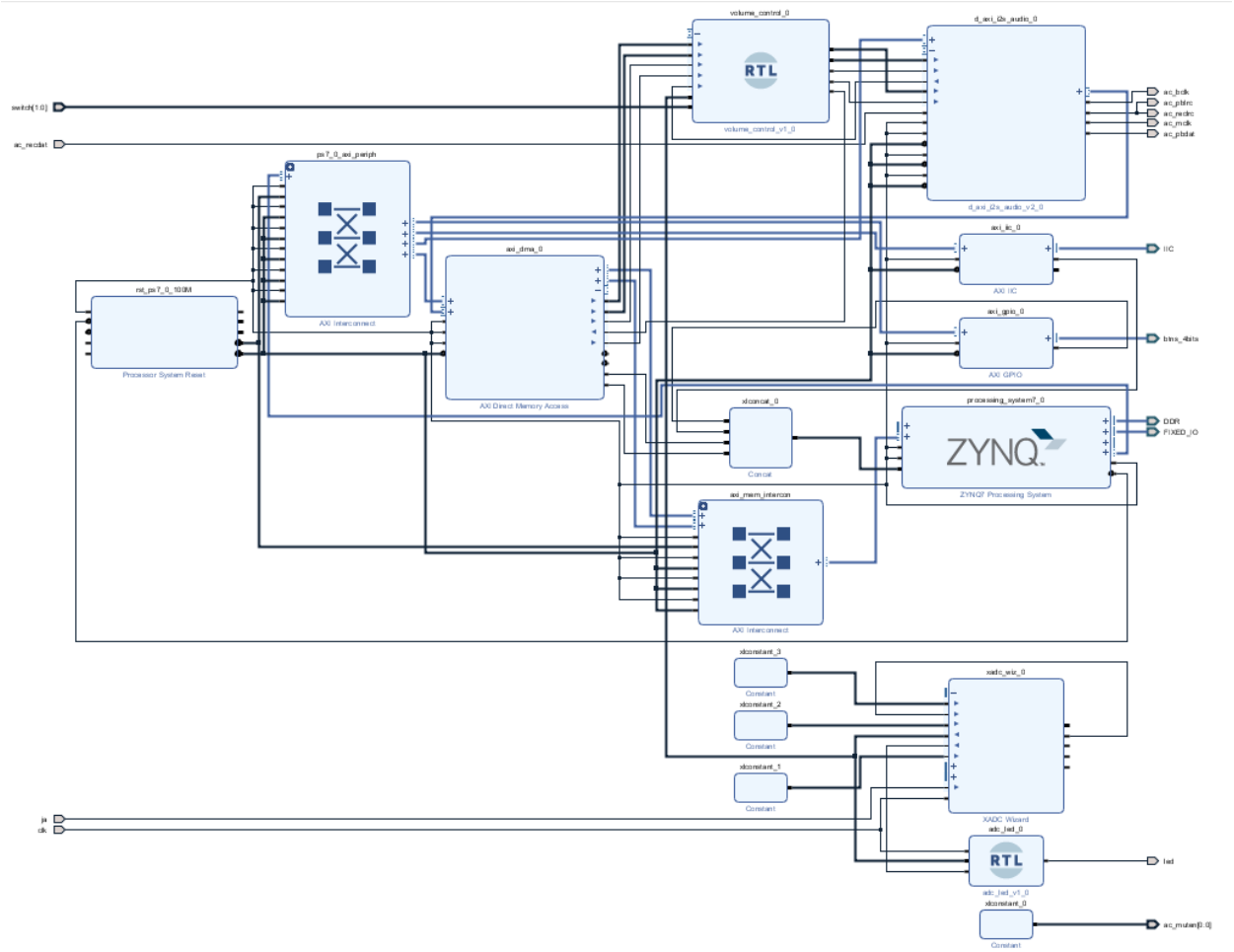
Eddie Hernandez has figured out how to take in an analog input into the Zybo z7-10 board, he also created a module to light up an LED using PWM to exactly show the level of volume using the analog input.

Abhishek Jasti has researched and scoured the internet to find resources that could help develop this project. He also helped with creating the volume control and equalization module.

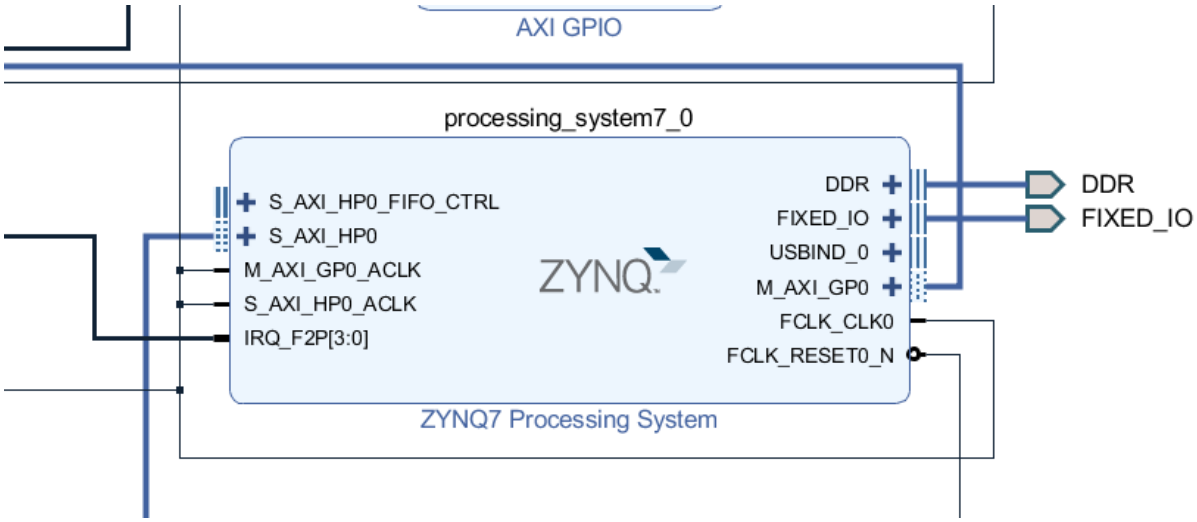
Anand Jasti has created the interface between the analog input and the volume control module, he has also helped create the volume control module and interface it with the internal memory module, and the audio output module.

Diagram

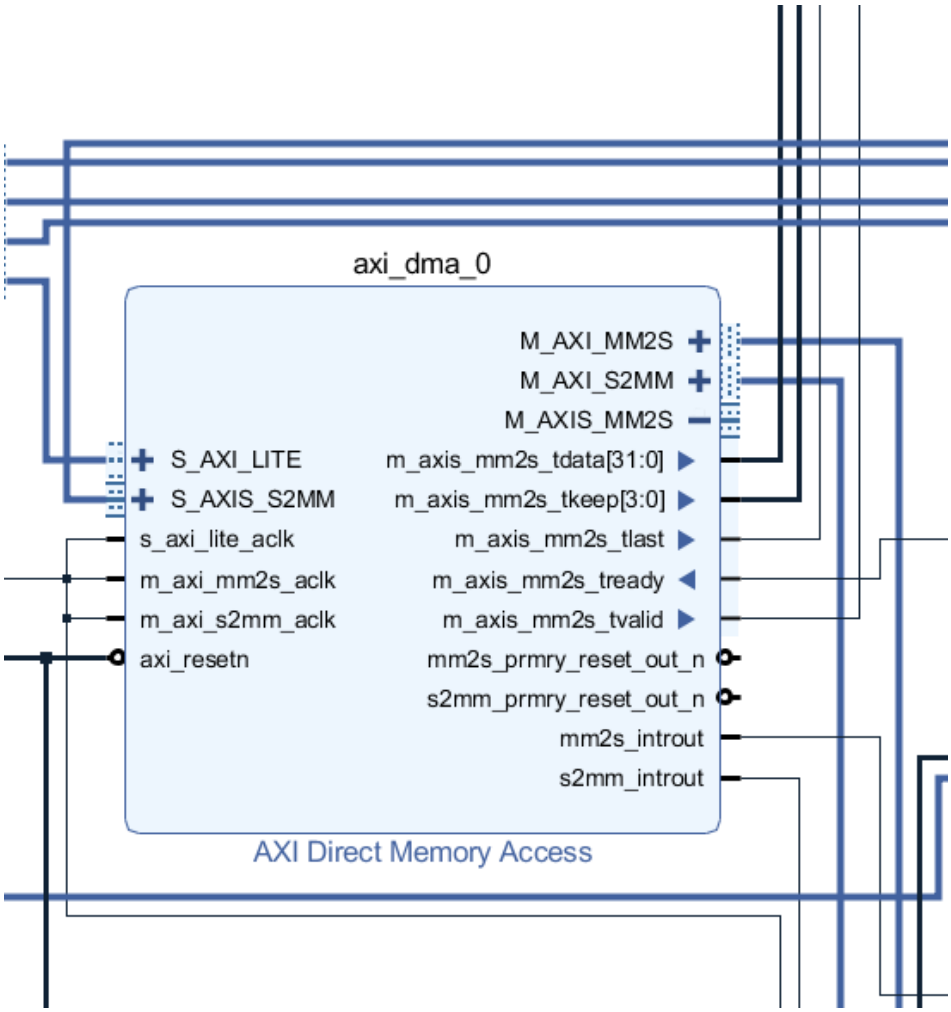
Full Design Diagram:



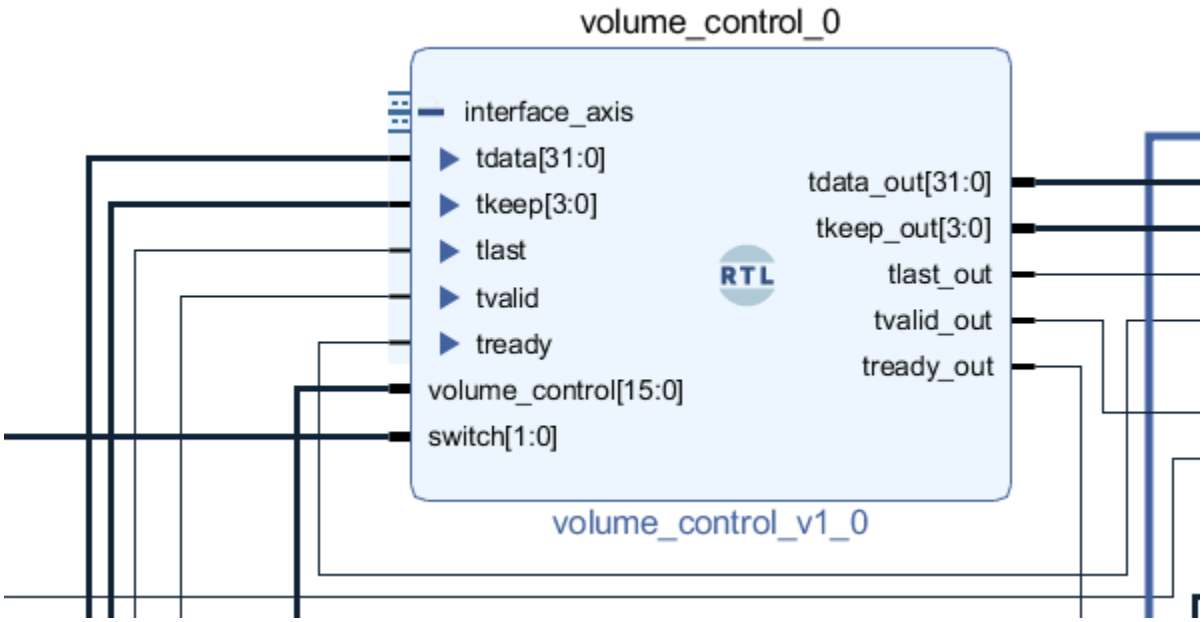
Processor Core Design Diagram:



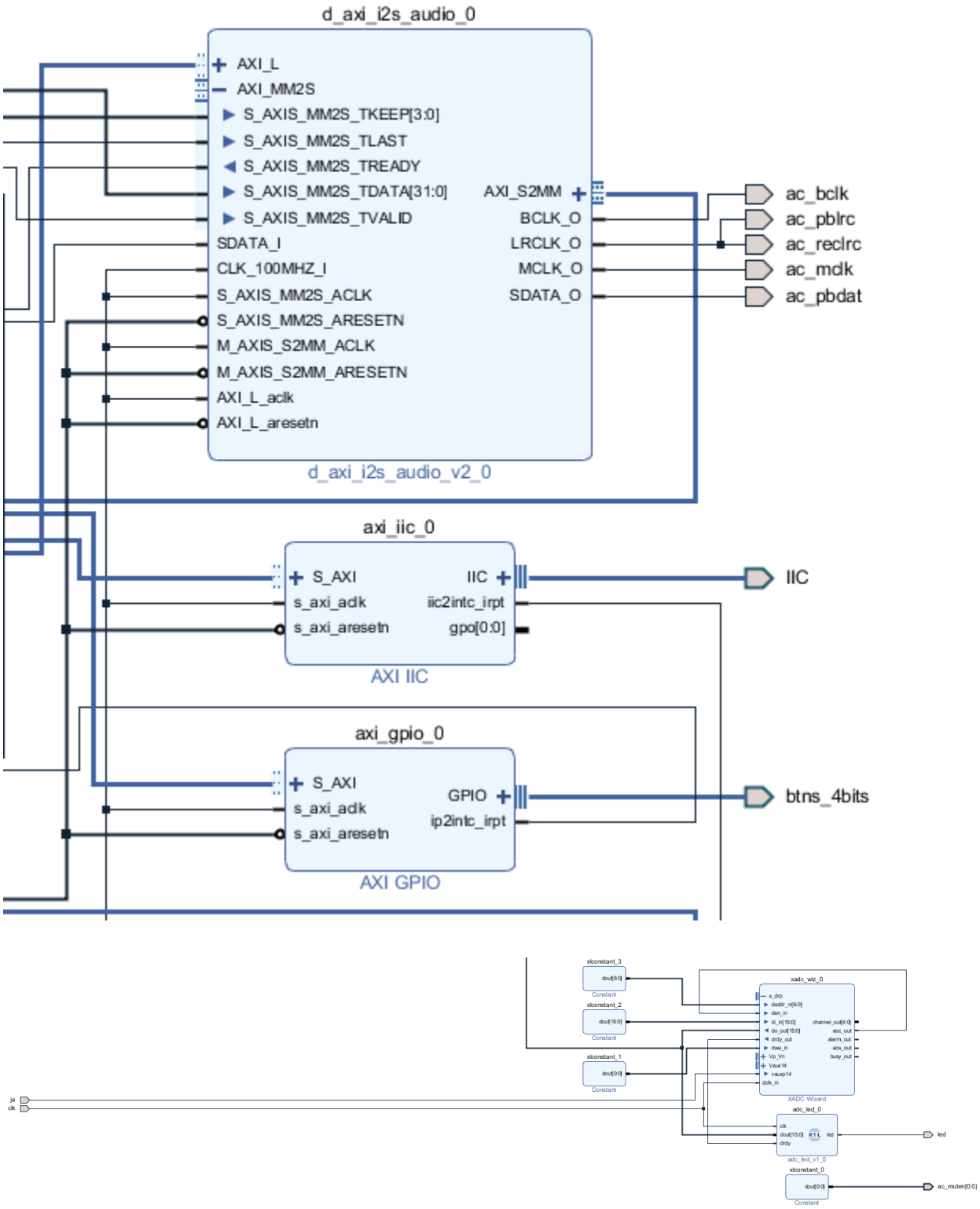
Memory Design Diagram:



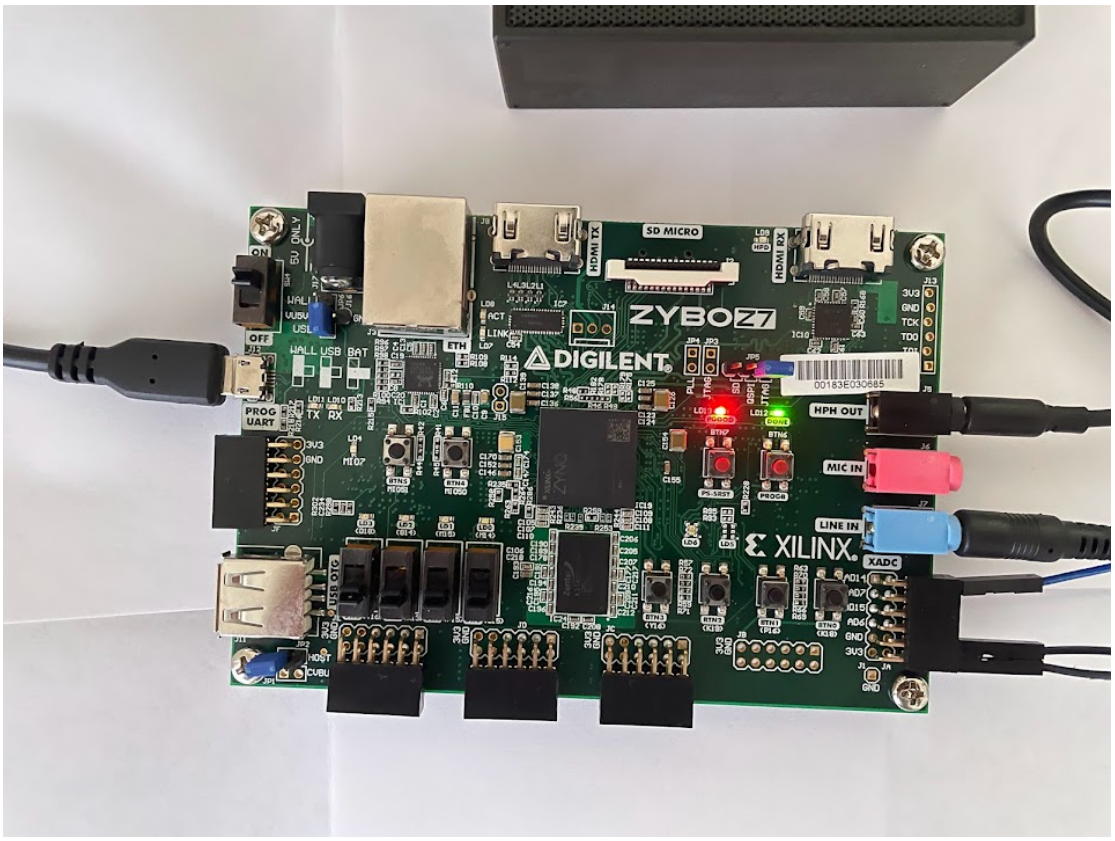
Audio Processing Module:

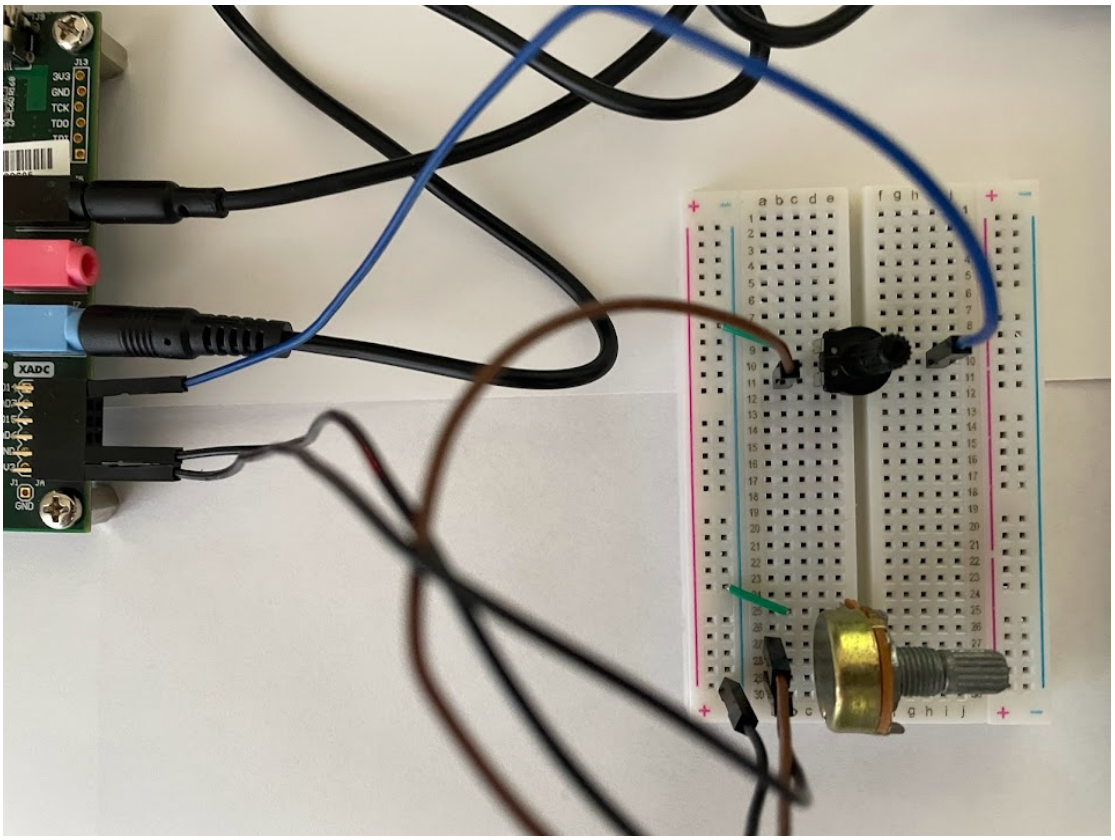


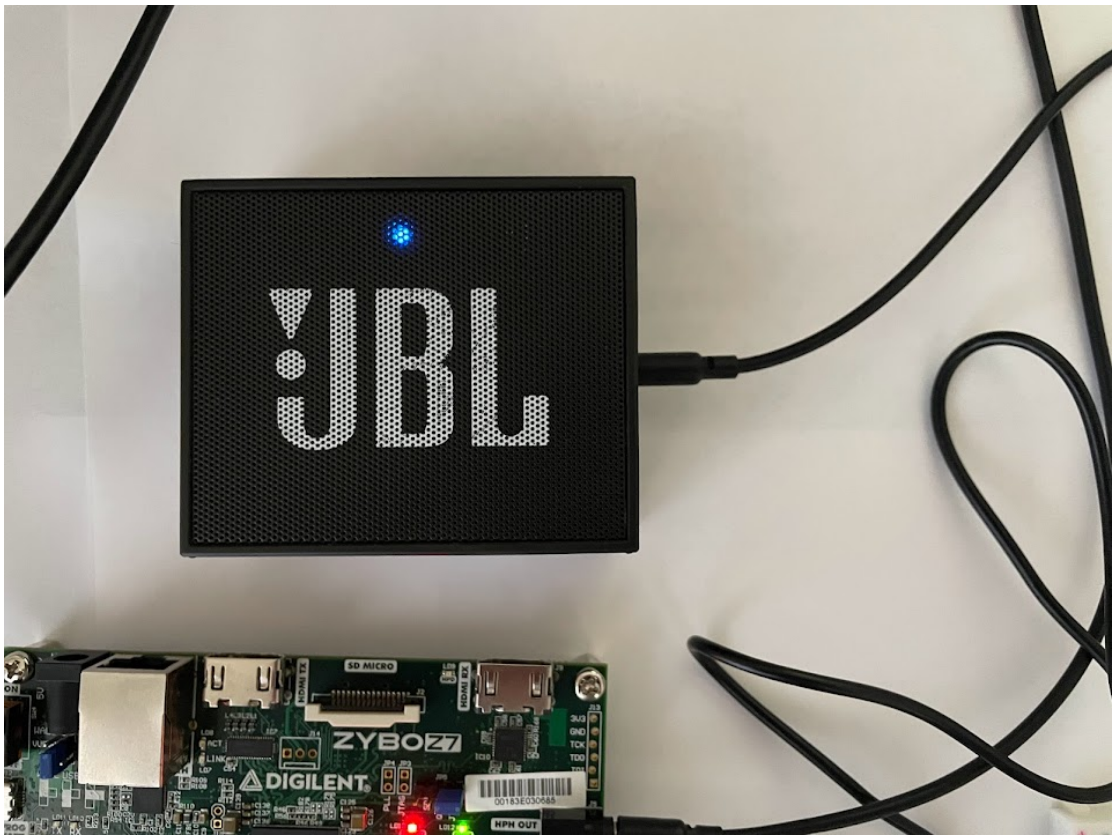
Input/Output interfaces:

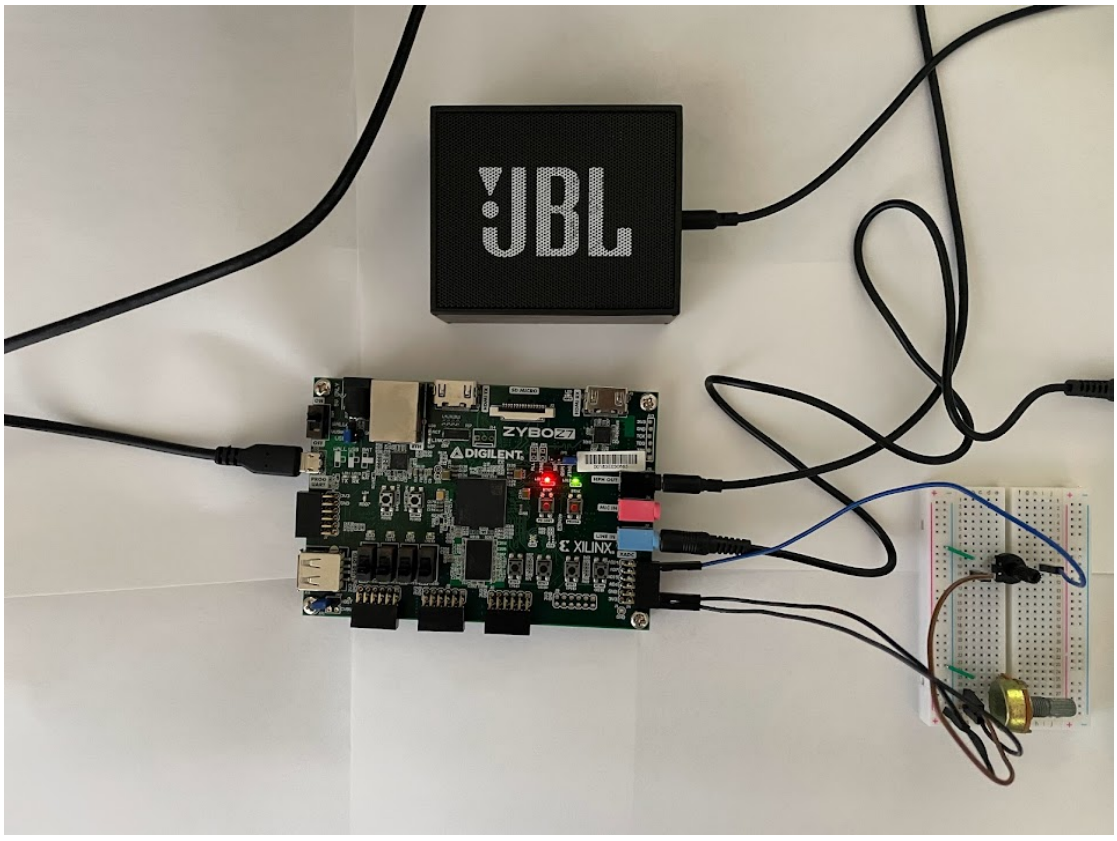


Picture of Board






JBL
SD MICRO
ZYBO Z7
DIGILENT
HPH OUT


JBL
ZYBO Z7
DIGILENT
XILINX

Demonstration of Project (Video)

Link to Demonstration Video:

https://youtu.be/MMlikNwXhaM

Overall Outcome

After completing this project we learned that processing audio is much more complicated than we had anticipated. We had to take in an analog signal from a microphone and convert it to digital, then process the audio and convert it back to analog so we could output it on the speaker. The equalization aspect of this project was the most difficult, as we initially could not understand all of the methods of equalization, and proved difficult to get it working. Despite these difficulties, our SOC was still able to process audio at our expected outcome.

Lessons Learned

- Zybo z7-10
    - Learned the basics of how a Zybo z7-10 board works.
- SDK
    - Learned how to set up and use the SDK for Vivado.
- Arm Cortex-A7 chip on the Zybo z7-10 board
    - Learned how to implement programmable logic using a processing system.
- Analog Input
    - Learned how to take in analog input on the Zybo z7-10 board.
- Volume Control

    - Learned how to control volume via an analog volume circuit and vary the voltage.
- Equalization
    - Implemented four levels of EQ: Gain, Low, Mid, and High.