Sound Localization: Design Document

Matheu Campbell (mgc2171), Dawn Yoo (dy2486), Peiran Wang (pw2593)
Elvis Wang (yw4082)

March 30, 2024

1 Introduction

In this project, we are aiming to implement a sound localization system. The microphone array will lie stationary in a row in the center of the room and the audio source will be a wireless speaker that we manually move around. The design will take place in three steps. First, the FPGA will acquire data from the microphones and store it in its memory. Second, the FPGA will calculate the angular offset of the audio source with respect to the microphone array. Finally, we will gather the display data.

2 System Block Diagram

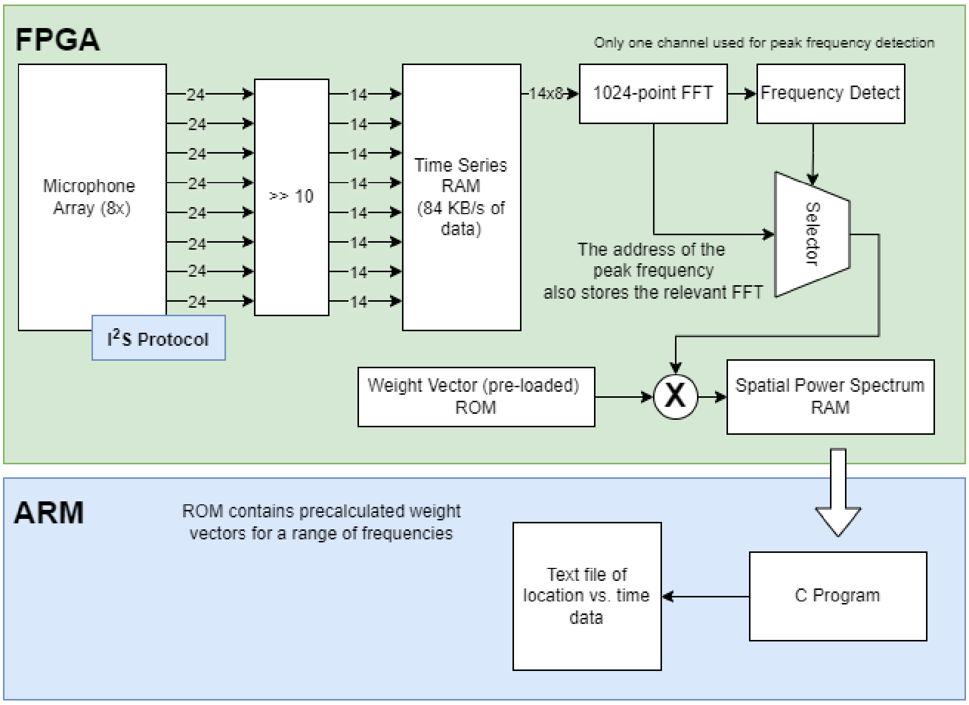
FPGA
Only one channel used for peak frequency detection
Microphone Array (8x)
24
I²S Protocol
>> 10
14
Time Series RAM (84 KB/s of data)
14x8
1024-point FFT
Frequency Detect
Selector
The address of the peak frequency also stores the relevant FFT
Weight Vector (pre-loaded) ROM
X
Spatial Power Spectrum RAM
ARM
ROM contains precalculated weight vectors for a range of frequencies
Text file of location vs. time data
C Program

Figure 1: System Block Diagram

Figure 1 shows the system block diagram of our design. It shows the general flow of data among blocks on the FPGA and in software.

3 Algorithms

Many algorithms exist for localizing a sound source using a microphone array. At their core, all of the known methods involve combining the spatial information from the microphone arrangement with the temporal information from the signal's arrival at the microphone array. More advanced methods achieve higher resolution (at the cost of higher computational effort) by calculating more obscure wave properties than amplitude and phase and optimizing on functions of those properties. More advanced methods still may use machine learning to implement an adaptive filtering technique. Due to memory and computational constraints, we will rely on the delay-and-sum technique to perform direction-of-arrival (DOA) analysis.

3.1 Bartlett Beamforming Algorithm for determining direction of arrival

Given a matrix representing the audio signals from a microphone array, beamforming is a method of weighting each sensor's signal vector such that audio signals originating from a target direction interfere constructively, and those from other directions interfere destructively. In ideal conditions, signals in the non-target direction are nulled by the interference.

Consider a linear array of 8 microphones, each separated by a distance d. For a plane-wave signal originating at angle θ, we define a steering vector $a(\theta)$ as follows:

$$\begin{aligned} a(\theta) &= [a_1(\theta), a_2(\theta), ..., a_M(\theta)]^T \qquad (1) \\ &= [e^{-j\varphi_1}, e^{-j\varphi_2}, ..., e^{-j\varphi_M}]^T \qquad (2) \end{aligned}$$

The vector defines the phase shifts that are to be applied to each sensor's signal to isolate sound from the target direction. The angle φ_i is dependent on the distance between sensors and the signal's wavelength λ as follows:

$$\varphi_i = 2\pi \frac{d(i-1)\sin\theta}{\lambda} \qquad (3)$$

The array's response to a signal $s(t)$ from a given angle θ can be expressed as follows, where $\mathbf{n}(t)$ is noise:

$$\mathbf{x}(t) = \mathbf{a}(\theta)s(t) + \mathbf{n}(t)$$

The power of the entire array's output is calculated as

$$\max E\{\mathbf{w}^H\mathbf{x}(t)\mathbf{x}^H(t)\mathbf{w}\} \qquad (4)$$

and the optimal weighting vector $\mathbf{w}$ is given by:

$$\mathbf{w_0} = \frac{\mathbf{a}(\theta)}{\sqrt{\mathbf{a}^H(\theta)\mathbf{a}(\theta)}}$$

Finally, the spatial power spectrum $P(\theta)$ can be generated.

$$\begin{aligned} P(\theta) &= \frac{\mathbf{a}^H(\theta)\mathbf{R}\mathbf{a}(\theta)}{\mathbf{a}^H(\theta)\mathbf{a}(\theta)} \qquad (5) \\ \mathbf{R} &= \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}(t)\mathbf{x}^H(t) \qquad (6) \end{aligned}$$

Note that equations (4) and (5) are simplified if $a(\theta)$ is normalized.

Once the spatial power spectrum is computed, the angle corresponding to its maximum must be the signal's direction of arrival.

3.2 FFT to determine frequency of interest

Before estimating the direction of arrival, we need to find the frequency of interest. While we'll restrict the signal's frequency to a range in which it can be resolved by the array, we don't want the system to be too sensitive to deviations from the nominal frequency of the input sound. We will use a Fourier transform on the incoming input to determine what the peak frequency of the signal is, and we'll assume that to be the frequency of interest. For each processing chunk, we will compute a 1024-point DFT with fixed-point arithmetic on just one of the input channels, assuming the peak frequency will be roughly the same for all channels. Once the FFT is complete, the bin corresponding to the maximum frequency will be found and stored for later use by the DOA estimating part of the system. The DE1-SoC's on-board multipliers for FFT will be used to implement the Cooley-Tukey FFT algorithm, decimating the signal in time and iteratively solving for the transform.

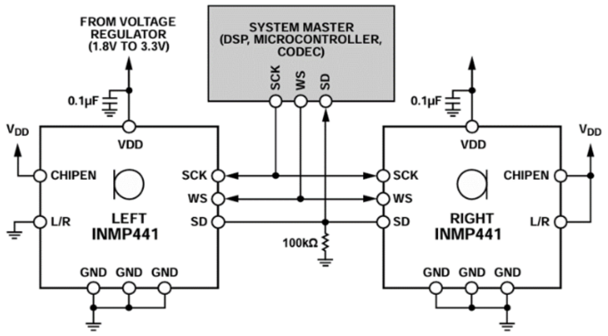

Figure 2: Microphone Block Design

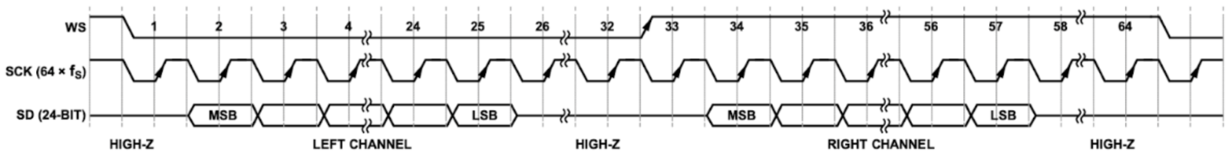

Figure 3: Stereo-Output I2S Format

4 Hardware/Software Interface

4.1 Microphone Array

4.1.1 Introduction

INMP441 MEMS microphones are used to collect audio signals. The digital microphones communicate with FPGA through I2S signal with three buses: serial data clock (SCK), serial data word select (SW) and serial data output (SD). To best utilize GPIO resources on board, two microphones are configured together as shown in Figure 2. Their timing diagram is shown in Figure 3. Notice that the output word length is 24 bits per channel, but a stereo pair always has 64 clock cycles for every data word ($f_{SCK} = 64 \times f_{WS}$).

A system is designed to test the microphones in advance, as in Figure 4. This test system bypasses the FFT module, directly passing the sound data to the Arm Core, where wave files are generated for algorithm simulation.

4.1.2 Pin Assignment

The board has two 40-pin expansion headers with 3.3V DC and GND pins. In the stereo design, three data wires are used: SCK, WS, and SD. SCK and WS can be reused in all four pairs. Thus, a total number of 8 pins will be occupied, two of which are VCC and GND pins.

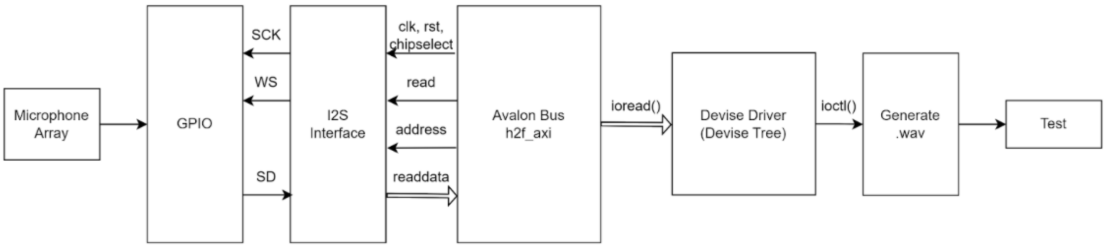

Figure 4: Test Block Diagram

4.1.3 Pseudocode

The idea of retrieving the I2S data is making use of a 5-bit width counter, which counts to 31 and returns to 0 two times in a period of the WS signal, and outputs the data when the counter number is smaller than 24. Additionally, a mealy state machine is adopted to change the state between idle (channel), left (channel), and right (channel). Here is the pseudocode of the SystemVerilog style.

```
// SCK and WS are generated by 50MHz clk with respect to the sampling rate
module project(input clk, input rst, input SD, output [23:0] left_data, output [23:0] right_data, outpu
logic [23:0] right, left;  // Temp memory
logic [8:0] clk_cnt;       // Clock counter
logic [4:0] cnt;           // Count to 32

always @(posedge clk) begin // Generate SCK
 if (rst)
   SCK <= 0
 else if (clk_cnt == SCK_NUM)
   SCK = ~SCK
   clk_cnt = 0
 else
  clk_cnt++
end

always @(posedge clk) begin // Generate WS
 if (rst)
   WS <= 0
 else if (clk_cnt == WS_NUM)
   WS = ~WS
   clk_cnt = 0
 else
  clk_cnt++
end

always @(posedge SCK) begin // Decode I2S SD signal
  if (rst)
    left <= 0
    right <= 0
  else
    cnt++
    case (state)
      IDLE:
      LEFT: if (cnt < 24) left <= {left[22:0], SD}   // SD is MSB first.
      RIGHT: if (cnt < 24) right <= {right[22:0], SD}
    endcase
end

endmodule
```

4.2 FFT

24-bit data that every microphone collected per sample is truncated to 14 bits and is sent to time series RAM. After that, all eight flows of data go through a 1024-point FFT.

4.3 Selector

The output of the FFT module will be compared to the pre-calculated result saved in the FPGA to get the angle and distance.

4.4 Display

We plan to display the location of the sound source in a .png format from the angle and distance data we acquired.

5 References

A new direction-of-arrival estimation method using automotive radar sensor arrays FPGA Implementation of a Bartlett Direction of Arrival Algorithm for a 5.8GHz Circular Antenna Array