6.111 Final Project Proposal

Rhian Chavez & Miles Johnson

October 2019

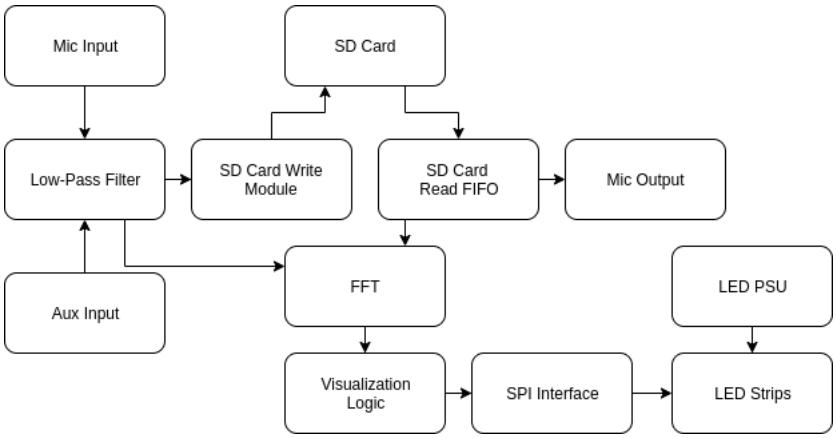

Figure 1: Full System Block Diagram

1 Introduction

For our project we will be creating a sound-interactive LED display. Our system will take as input either sound from a microphone, aux input, or audio data saved on an SD Card. We also supply a module for recording an audio input onto an SD Card to be played later. When the input is from an SD Card, we allow concurrent audio output with the LED display.

Our LED display will consist of eight led strips, each with 100 individual leds, oriented horizontally and stacked vertically on a wooden platform. We will control these lights by processing audio data and extracting information, especially from the fourier transform and significant beats, to be displayed in an aesthetically pleasing manner on the strips. As an example, we may have the horizontal direction along the strips correspond to frequency, the vertical direction across strips correspond to intensity, and the variation of color of the lights correspond to significant beats.

The details of each module in our system are outlined below, followed by a short analysis of possible performance limitations.

2 Filter - Rhian

The filter module will take streamed 1 byte data at 1MHz from the ADC (generated by Vivado IP) and remove the higher frequency noise associated with microphone sound recording. We may also have to use a filter when taking audio data from an aux input but it is unlikely. The filter module will take in only a clock, reset, and streamed 8 bit data in, and output streamed 8 bit data out. This will be accomplished using a 31 tap low-pass FIR filter designed in MATLAB with a specified cutoff frequency and roll-off. The internal calculations will be signed multiplication and addition of 8 bit values. Testing of this module can easily be accomplished using a simply software test-bench in Vivado.

3 SD Interface - Miles

3.1 SD Card

This module interfaces with the built in SD card slot on the FPGA, letting us write and read to the SD card. This allows us to store a sound sample to be played back at will. Additionally, we may load a sound sample on the SD card outside the FPGA and play that sample back.

3.2 SD Card Write Module

This module allows us to write data to the SD Card, and thus interfaces with the SD Card module. We will be using this to record either data from a microphone or data from an aux input. Specifically, this module should convert data from the Low-Pass Filter into a suitable format for input to the SD Card interface, and send it along as such. From some preliminary research, it seems the SD Card may use an SPI protocol like out LEDs, so we maybe able to reuse that module here or vice versa. However, the interface may depend on the SD Card we use, and more research is required to determine exactly how this module will function.

3.3 SD Card Read FIFO

This module, similar to the previous module, will interface with the SD Card module, this time with the purpose of reading. From a previous year's 6.111 project, it seems reading from the SD Card is not a reliably continuous project. Thus in order for our lights to work smoothly it will likely be advantageous to send read data to some memory block buffer which can then send the data to the FFT in a smooth manner. This module will be responsible for formatting the data from the SD card into inputs for the sound output and the FFT, as

well as maintining continuity in the data stream and synchronisity between the microphone and the lights (at least when we are playing music from memory).

4 FFT - Rhian

The FFT will take in streamed 8 bit data at a 1Mhz rate and periodically produce a 128 bin, 1 byte magnitude DFT. This will require an 8 bit wide data input, clock input, reset input, and will have 128 8 bit wide output ports (1 byte for each bin). The desired throughput will be at least 100Hz in order to visualize dynamic audio quickly. The module will be tested by supplying the FFT with various signals with a known DFT (likely using a text file and test bench) and comparing the DFT of the result.

5 Visualization Logic - Both

The visualization logic module will be responsible for translating the FFT data into the desired color and brightness of each LED in out 8 strips of 60 leds. This module will take data from the FFT at 100Hz (in parallel, allowing for every frequency bin to be interpreted at once, and will also be connected to a reset button. Output will be 8 bit stream at 100kHz to the SPI interface. Visualizations will include displaying frequency as a function of horizontal position on the led strips, amplitude as a function of brightness and vertical range, as well as sweeping color through a color wheel and drastically changing color when a significant beat has been calculated.

6 SPI Interface - Rhian

The SPI interface module will be responsible for translating information from the visualization logic into language that the LED strips can understand. Since we plan on having 8 led strips (each with a signal and clock line) the SPI interface will need to have 8 outputs (one for each LED strip). Inputs will likely be few depending on how many properties we would like to visualize, but we will likely stream data from the visualization logic over an 8 bit bus at a 100kHz rate or so and have a couple other inputs specifying other audio events such as beat detection. The output will update at the same rate as the input (100kHz). Since the LED strips can be controlled at any clock rate, 100kHz is reasonable and will allow for very quick updating of the entire strip (containing 60 LEDs per strip).

7 LED Hardware - Both

The LED hardware, although not digital logic, is important. The 8 1 meter long LED strips will be glued to a rectangular slab and their control wires will

be routed to digital output on the DDR board. 5V DC power with significant current capability will need to be connected to the LED strips as well. The total number of outputs needed will be 10 (8 data signals, common clock, common grounds). Wires will be kept short to avoid loss. We may also need to step up the output signals of the DDR to 5V using an IC.

8 Potential Performance/ Hardware Limitations

(i) Computational memory space:

For our project we will need to process data from the output of the FFT, and convert that data to an input for eight strips of lights. The most obvious implementation has us calculating the FFT with 100 data points, and sending all such data points to a processing module together. This processing module would then perform all necessary calculations with the data points, which involves specifying 24-bit RGB values for 800 leds. It is possible that these calculations will use too much memory, so we will need to optimize in some way, perhaps by spreading the computation over multiple clock cycles.

(ii) Computational speed:

This limitation is related to the memory space as well. The main goal of our project is to make the lights as aesthetically pleasing as possible, and much of this is reliant on a quick refresh rate. Our computational speed is limited by how much memory we are able to use as well as the nature of the computation itself. We will need to strike a balance between functionality, computational speed, and memory usage.

(iii) Power output

Especially given that we would like to be able to control a speaker while controlling the leds, the maximum output current may be a limiting factor. One way we may address this if it becomes a problem is by using time division multiplexing so we only have to control a single strip at a time.