Jose Rubianes (jer2201), Tomin Perea-Chamblee (tep2116),
Eitan Kaplan (ek2928), Mikhail Belov (mb4560)
CSEEW4840 Embedded Systems
Project Proposal

Totally Not Shazam
Song Recognition

Summary

Use the built in microphone on the development board to recognize songs being played in the background. Song recognition will depend heavily on signal analysis, which will be accelerated by a custom FFT unit implemented on the FPGA.

Planned Implementation

Hardware

We will write SystemVerilog to access the built in microphone on the board. The time-domain digital signal from the microphone will be pipelined into a unit on the FPGA which will continuously calculate a discrete time Short-Time Fourier Transform (STFT) to create a spectrogram. The STFT circuit will be implemented by many parallel FFT circuits (the FFT is an operation which is amenable to being implemented in hardware). Some work will be necessary to determine the space requirements for the STFT. Ideally, if the data is small enough, it will be stored in the FPGA BRAMs. The STFT data (i.e., the spectrogram) will then be read by the processor from the RAM. We will implement ping pong buffers to facilitate data sharing between the microphone controller and the STFT circuit, and between the STFT circuit and the processor. We will also use the DSP hard IP units of the FPGA to aid in STFT calculation.

Software

Once we have the spectrogram, we will find the frequency peaks of the spectrogram and store the time-frequency coordinates of those peaks. Next step is to implement the hashing search strategy described by Froitzheim[1] to build a hashed song database. Finally, we are going to use that database to recognize short (~ten-second) samples of an unknown song and print the name of the song to the monitor.

We will use a switch to set the system to either training or classifying mode. A push button will enable the microphone and have it record a song. The monitor will display the name of the song soon after the push button is released or display an error message such as “too short of a sample” or “unknown song”.

[1] Froitzheim, S. (2017). A Short Introduction to Audio Fingerprinting with a Focus on Shazam. Retrieved from http://hpac.rwth-aachen.de/teaching/sem-mus-17/Reports/Froitzheim.pdf

Possible tweaks/modifications/optimizations

It may make sense to find the peaks of the STFT in hardware rather than software. Reducing the STFT to just the amplitude peaks would significantly reduce the amount of data that needs to be transferred from the FPGA to the processor. This can greatly decrease the amount of data that needs to be sent on the bus (since bus reads will likely be a bottleneck).

Ideally, we will find a database of songs that will allow us to build up a large hash table so we can classify many songs. That may not be practical however, in which case we will just train on a handful of songs (<50) that we will manually play for the system; of course, this would mean that we can only classify those few songs we teach the system.