From: Hojin Jeon (hojinj@mit.edu)
To: Prof. Gim P. Hom (gim@mit.edu)
Date: 2017 NOV 12
Subject: 6.111 Final Project Proposal

FPGA Audio Spectrum Analyzer

Overview

This project aims to implement an audio spectrum analyzer on an FPGA system. The basic function of a spectrum analyzer is to identify and measure frequency components of a discrete time signal using an FFT algorithm. FPGAs often supplement to their configurable logic array with specialized DSP hardware, and may be more capable than general-purpose processors for such a task.

Implementation involves separate modules for obtaining and digitally sampling a continuous audio signal, applying FFT to obtain the frequency components of the digital signal, taking measurements pertaining to frequency, and graphically representing the data via LEDs, 7-seg hex display, and/or VGA display. Typical spectrum analyzer functionality includes a configurable frequency range and resolution bandwidth, peak detection, total spectrum intensity measurements, and noise measurements. The project will begin a basic audio spectrogram, and add additional functionality with time and hardware permitting.

Hardware

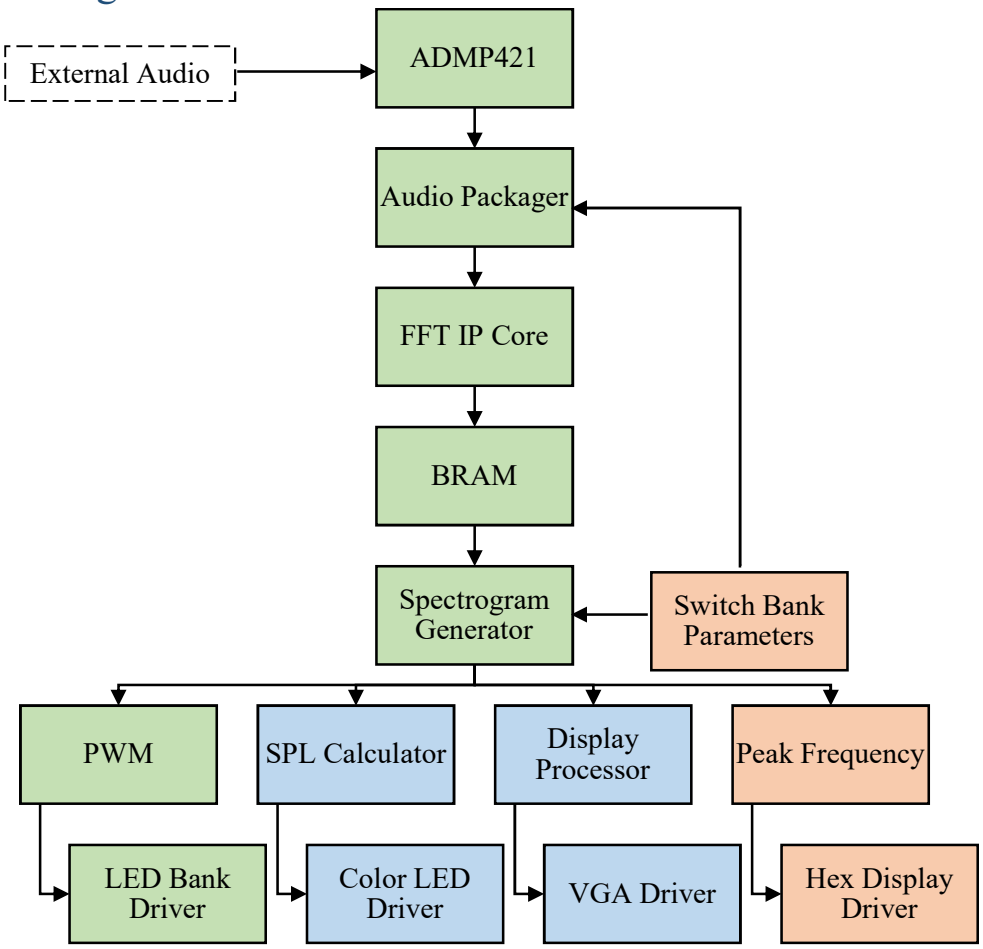
At the base of a Nexys-4 development board is the Artix-7 FPGA from Xilinx, specifically the 7A100T [1]. This particular chip features 15,850 CLBs (configurable logic blocks), 4,860Kb BRAM, 1 dual-channel XADC, and 240 DSP48E1 slices [2]. The most resource-intensive module of this project will be the FFT core itself, which depending on the architecture and bit-depth chosen, requires anywhere from 1e3 to 1e4 LUTs (look up tables) and FFs (flip flops), up to 50 DSP48s, and up to 100 18Kb BRAMs [3]. This is possible within the 7A100T [4], and an FFT architecture that leaves enough core and processing time for the other desired functionalities of measurement and display will be chosen for this project. For example, a VGA module alone requires at least 256 KB video memory.

Clocking and speed is generally not a concern for this project. The 7A100T includes 6 CMTs (clock management tiles) capable of frequencies exceeding 100 MHz [5], and the FFT core IP included in the Vivado suite can analyze frequencies of at least 300 Mhz [3]. This project is generally concerned with the human audio frequency range, which ranges from 20 Hz to 20 kHz. This speed overhead may allow for techniques such as oversampling, which will improve measurement noise and resolution.

Goals

Basic	Expected	Stretch
• Implement FFT IP with digitally sampled audio data. • Display a live spectrogram on the Nexys-4 16 LED bank, with variable intensities via PWM.	• Implement sound pressure level meter using color LEDs (green/yellow/red). • Display a live spectrogram on VGA display.	• Configure FFT frequency range and/or resolution using switch bank. • Display peak frequency on hex display.

Block Diagram



Due to the modularity and low interdependency of this project, additional functions can be added without affecting the existing ones. The core project is highlighted in green, desired functionality in blue, and stretch functionality in orange. Certain modules may be omitted if hardware constraints do not allow them; the Display Processor and VGA Driver are of particular concern.

Implementation

1. **ADMP421**
   External audio is obtained via the ADMP421, package containing a microphone and sigma delta modulator. The Data output is in PDM format. Output data is made available on either the rising or falling edge of clock, depending on whether the L/R select pin on the chip is high or low. 2.4 MHz is typical for CLK. All these signals are connected and generated internally.

2. **Audio Packager**
   The Audio Packager converts single-channel PDM data to a format compatible with the FFT IP core. This may involve conversion to PCM data, output with a serial transmission scheme. As we desire information up to 20 kHz our sampling rate must be at least 40 kHz, due to the Nyquist theorem. This module will involve filtering and decimation of the incoming PDM data.

3. **FFT IP Core**
   The FFT Core will take a real-valued signal, and output the real and imaginary components of a frequency-spectrum. As we want the magnitude of frequency response and not the real and imaginary components separately, we will have to perform an "x = sqrt (a^2 + b^2)" calculation. It will be best if the IP core includes this functionality. If not, additional hardware will be required. The FFT core uses an AXI4-stream interface for its data inputs and outputs, which appears to be a serial-transmission scheme. Figuring out this interface is a hurdle for this project.

4. **BRAM**
   The output of the FFT core will be frequency domain data. The maximum transform size (# frequency bins) is 2^16 = 65536. Assuming a byte per bin (bit-depth of 8), storing the spectra of one frame at the maximum transform size will require ~525 Kb of memory. It is likely that the full size will not be used, as collecting data to for a spectrogram on the 16 LED bank only requires 32 bins.

5. **Spectrogram Generator**
   The results of FFT must be processed to suit a spectrogram. For example, half of the spectra generated by an FFT of a real valued signal is discarded due to symmetry. Sound is also typically measured in dB rather than absolute intensity by frequency. If the project warrants any additional processing, it will be completed in this module.

6. **PWM**
   Nexys 4 includes a bank of 16 LEDs that are set either on or off. We may give the illusion of varying brightness via PWM. PWM frequencies above 200 Hz are imperceptible as flicker to the human eye; this will be the minimum frequency of the signal generated by this module. The duty cycle of the PWM will be proportional to the value of each bin produced by the spectrogram generator.

7. **LED Bank Driver**
   This module will drive the 16 LEDs on the Nexys 4 with the PWM signal generated previously.

8. **SPL Calculator**
   Strictly speaking, SPL is a measure of the deviation in ambient pressure. A time-averaged measurement of SPL may be obtained by summing the energy contained in all the frequencies within a frame of the spectrogram. This module will calculate that value, and convert it to an RGB value acceptable for the Color LEDs.

9. **Color LED Driver**
   Commercial SPL meters typically use green to indicate low intensity, yellow for medium intensity, and red for high intensity or signal clipping situations. This module will drive the color LEDs to the appropriate color using the values derived from the SPL calculator.

10. **Display Processor**
    The display processor will generate some graphic representation of the spectrogram for the VGA display. The simplest spectrograms consist of rectangles of varying sizes, one per bin of the spectrogram. This module will generate the sprites and graphics required for the VGA driver.

11. **VGA Driver**

The VGA driver drives the VGA port of the Nexys 4. This code may be borrowed from previous labs.

12. **Peak Frequency Finder**
    Typical spectrum analyzers can show which frequency contains the most energy. This module involves finding the greatest value in the array storing the spectrogram, identifying the frequency bin it belongs to, and creating a hexadecimal string for the 7-seg display.

13. **Hex Display Driver**
    The hex display driver will display the frequency and corresponding intensity found by the previous module on the onboard 7-seg display.

14. **Switch Bank Parameters**
    The Nexys 4 includes a bank of 16 switches below the LEDs. One or more of these switches can be used to provide parameters to the previous modules, to vary the range and resolution of FFT.

Timeline

Week of…	**Task**
Nov. 13	Audio Packager / FFT IP Core Implementation
Nov. 20	BRAM / Spectrogram Analyzer / PWM / LED Bank Driver
Nov. 27	Complete Core Project (green), SPL Calculator / Color LED Driver / VGA Display
Dec. 4th	Complete Expected Goals (blue), Work on Stretch Goals
Dec. 11th	Finish

Sources

[1]: https://reference.digilentinc.com/_media/nexys:nexys4:nexys4_rm.pdf
[2]: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf
[3]: https://www.xilinx.com/support/documentation/ip_documentation/ru/xfft.html
[4]: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf
[5]: https://www.xilinx.com/support/documentation/user_guides/ug472_7Series_Clocking.pdf