Image Buddy: Voice-controlled Slideshow

Group 9
Cameron Buttazzoni
Shuran Xu
Shuyi Wu

Project Overview

High-level Description

FPGA

- User presses a button, then says one of three words: “alligator”, “baboon”, or “cow”
- The built-in microphone records the data
- The data is pre-processed using the MFCC algorithm to extract features

Server

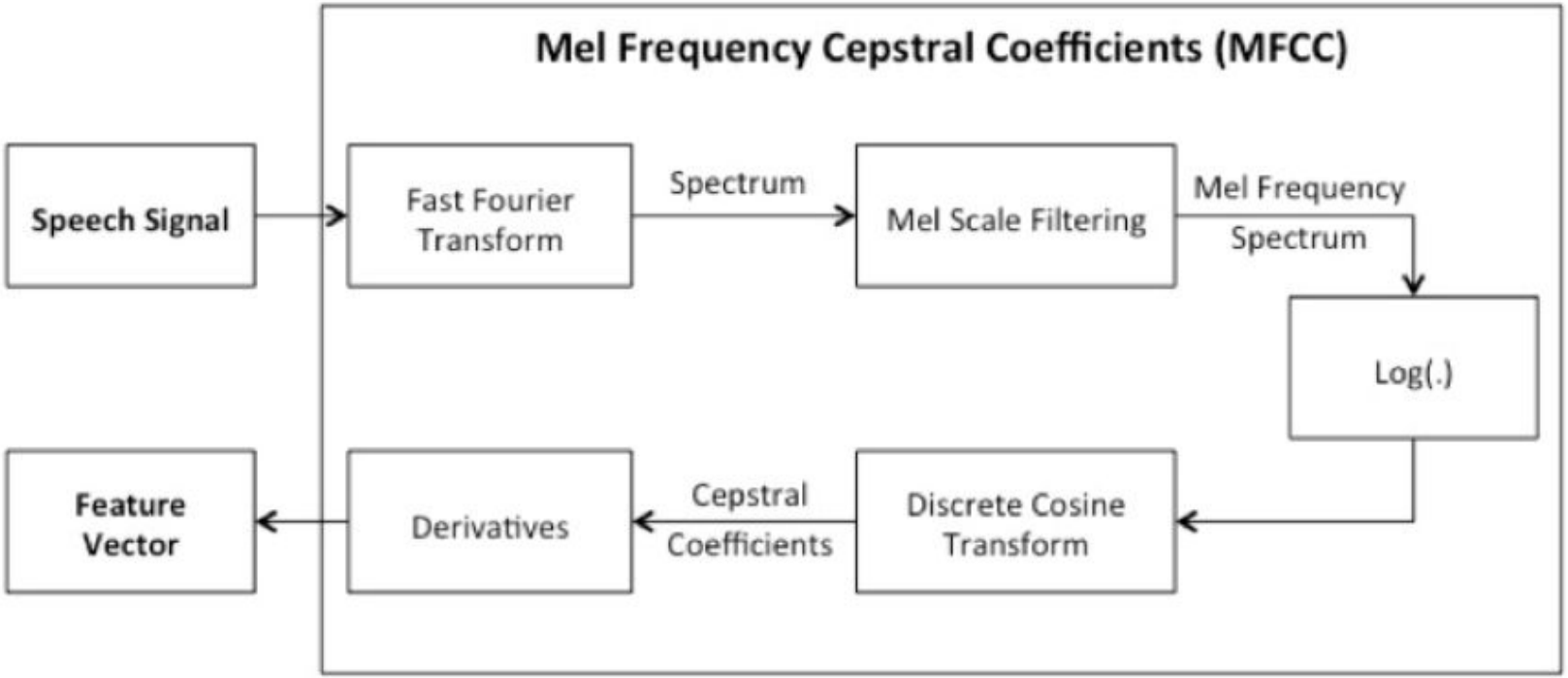
- The speech sample features are sent to a computer server over Ethernet
- A pre-trained neural network classifies the speech sample into one of the three categories of animals
- An image of that animal will be sent back to the FPGA over Ethernet, where it will be displayed on the PMOD screen

MFCC Algorithm

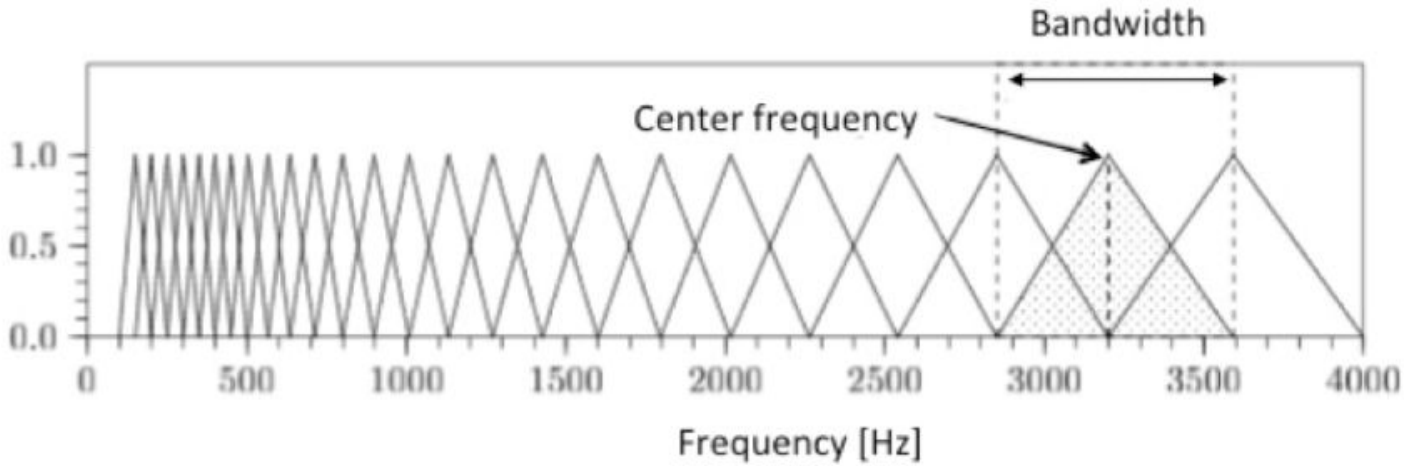
Steps:

1. FFT
2. Mel-Scale Filtering (Bandpass Filters)
3. Logarithm
4. Cepstral Coefficients (optional: remove speaker dependent characteristics)
5. Derivatives (optional)

MFCC Algorithm



Mel-Scale Filtering



Goals (from mid-project presentation)

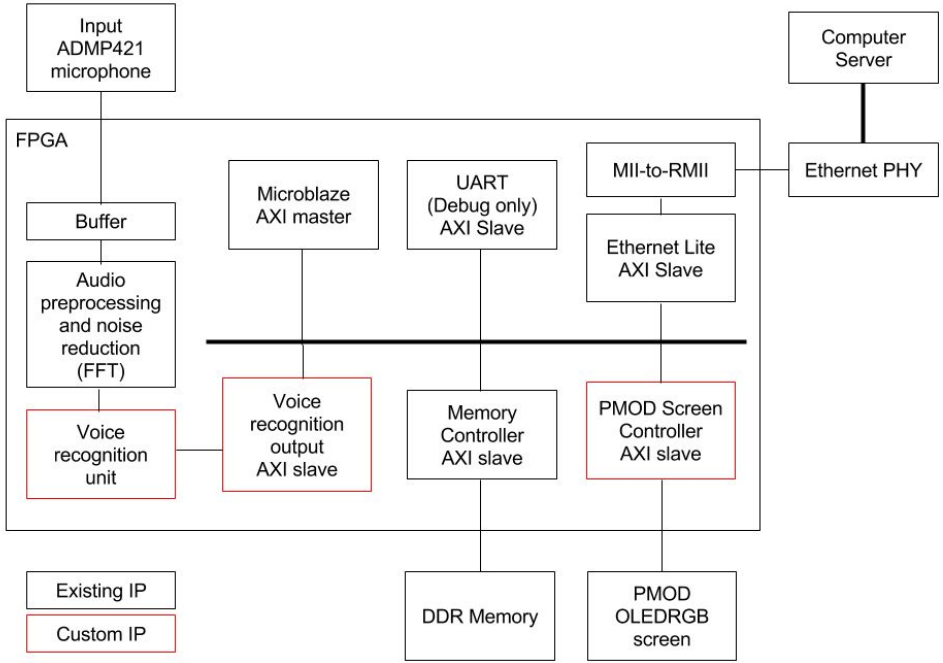
Primary goals:

- ✓ Implement a functional IoT voice recognition system
- ✓ Achieve over 50% recognition accuracy and < 3 second latency

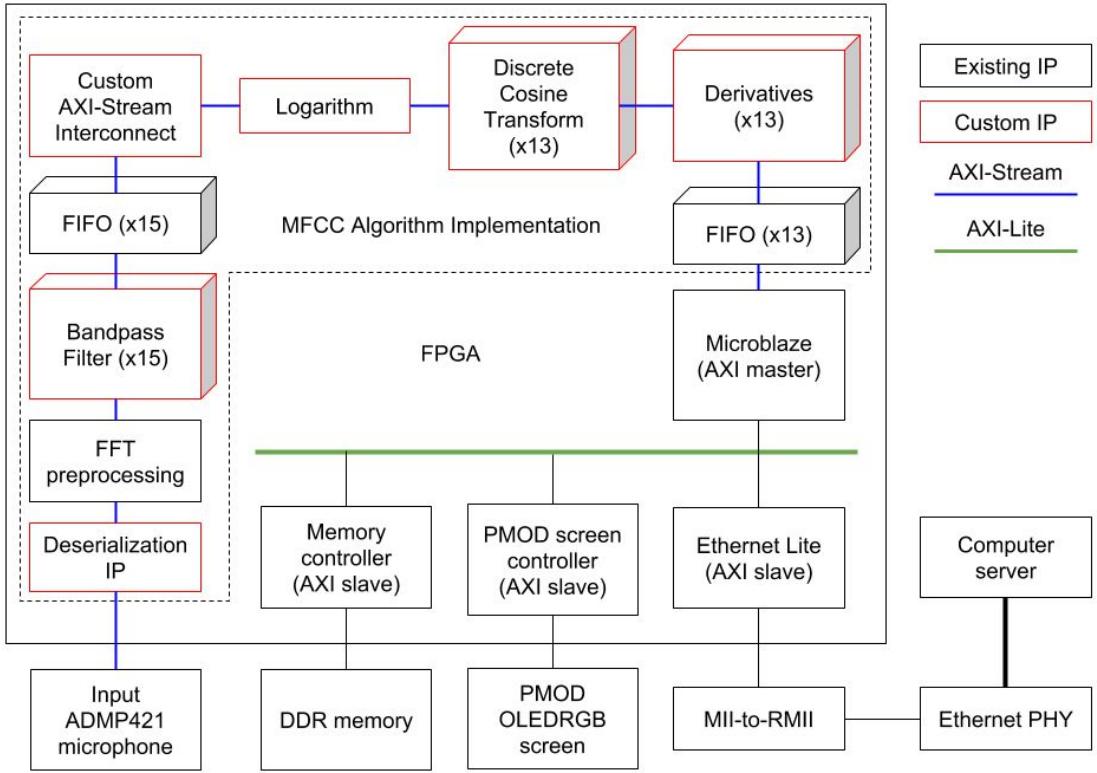
Stretch goals:

- ✓ Send images from the server back to the FPGA in an appropriate format, and use the PMOD screen to display the images
- ✗ Expand the number of commands recognized by the system (e.g., add additional animal categories, add commands to control the slideshow such as “next”)
- ✓ Add optional steps from MFCC algorithm

Initial Block Diagram



Final Block Diagram



Problems

- After adding the Derivatives modules, the design was too large to be placed and routed
- In the Discrete Cosine Transform module, the accumulator that we made did not have enough output bits and would often overflow (this was not discovered until the neural network did not perform very well)
- The 0th AXI-Stream port on the MicroBlaze does not work
- The Vivado AXI-Stream Interconnect IP outputted garbage values (we probed its inputs and outputs with the ILA), and so we had to write our own interconnect
- Network is not very robust (server does not receive all values sometimes)

Project Management

Design Methodology

- Main design philosophy: do everything on server first, and then slowly move parts over from software to hardware
- The design's individual components (deserializer, FFT, filters, networking, etc.) are individually tested in simulation, and then integrated one at a time and tested in hardware
- Sub-components are kept as separate IPs/blocks so integration has less code

What We Learned

- Signal processing algorithms (MFCC) and theory
- How to write good Verilog code and testbenches
- How to build a complex project out of smaller pieces
- The AXI-Stream interface
- Testing components thoroughly before integration

Questions?

Thank you!