Voice Controlled Car System

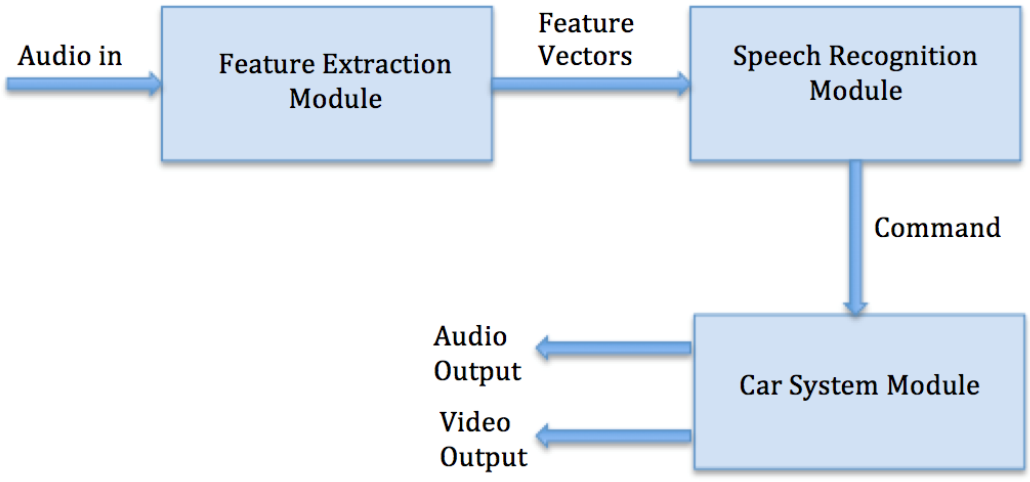
6.111 Project Proposal
Ekin Karasan & Driss Hafdi
November 3, 2016

1. Overview

Voice controlled car systems have been very important in providing the ability to drivers to adjust the controls of the car without any distractions. These systems are continuously improving providing drivers more control over the internal car system.
Our main goal in this project is to create a voice controlled car system, which has the ability to detect specific voice commands. According to these commands the system adjusts both the display on a computer screen and the audio output.
The system will consist of a microphone that detects input audio signals. The video output and the audio output will be controlled using digital outputs from the Nexys4. The car system will be able to play music from a certain library, adjust the volume, make calls to a certain list of contacts and display maps.

2. Design

The overall system will consist of three main modules: feature extraction module, speech recognition module and car system module. The brief overview of the general system is described in the schematic below:



The three modules are explained in more depth below:

1. Feature Extraction Module

   The main task of this module is to take as input the audio input from the microphone and produce a feature vector that will be compared to feature vectors collected from trained samples in order to determine the command. We are considering two different options for the design of this module. One option will be to extract peaks from the FFT of the audio input and produce a feature vector based on this. Another option is to use an algorithm called MFCC that will be explained in more depth in the next section. This decision will be determined based on testing in software (MATLAB).

2. Speech Recognition Module

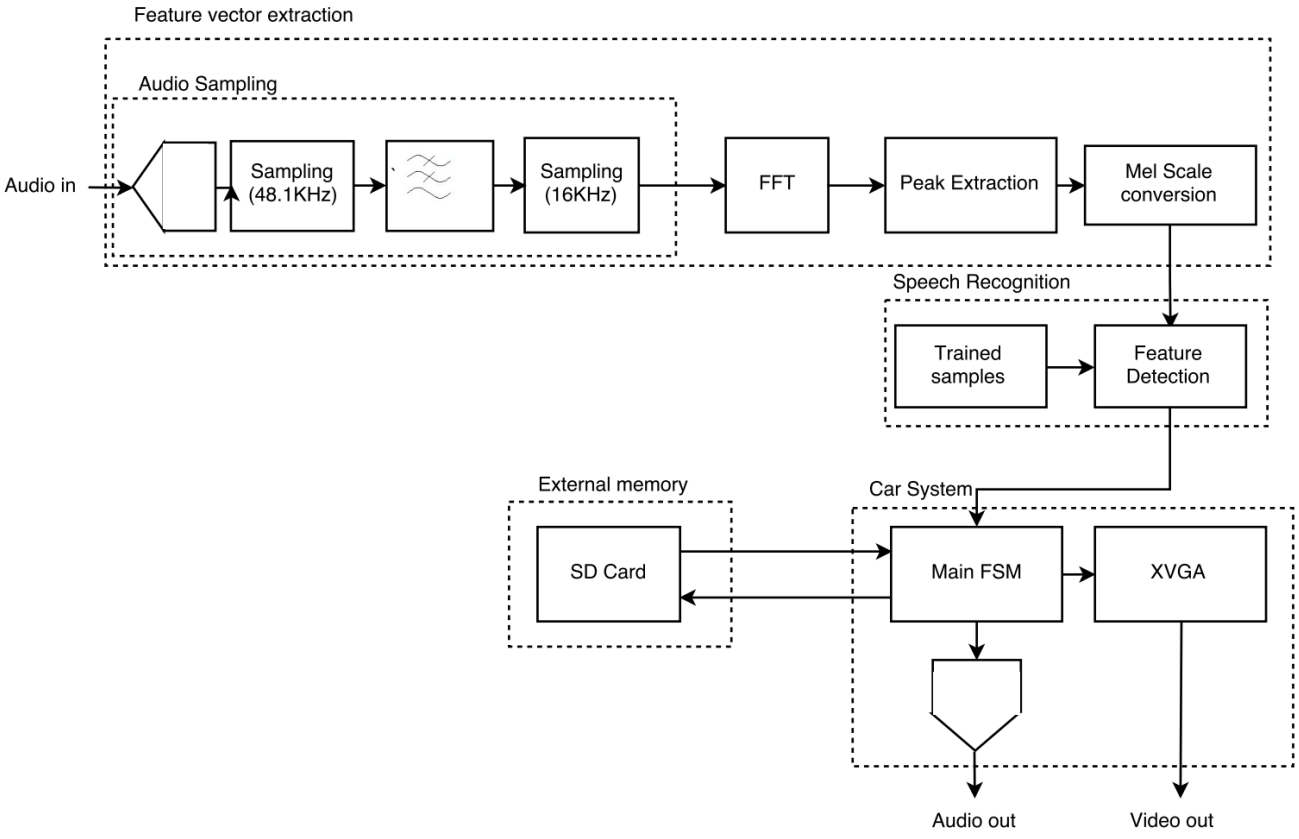
   This module will take a feature vector as input and compare it to the feature vectors of trained samples in order to produce a command as output.

3. Car System Module

   This module will behave as a finite state machine that takes in a command as input. Depending on the command it will output both a video output and an audio output provided to module through an external SD card.

3. Implementation and Testing

1. Block Diagram



2. Feature Extraction Module

This module has an audio_en signal as input, which controls whether this module will be performed. The audio_en signal is only high if the enable button is pressed (by the user) and the audio input is above a certain threshold (the user is speaking).

a. Audio Sampling Module (Driss)

This module will use the 12-bit XADC of the Nexys4 in order to convert the analog audio input into a digital signal. The digital signal will be down-sampled to a rate of 48kHz. This is possible without an anti-aliasing filter because the nyquist rate given by the highest frequency component of our signal is lower than the

sampling rate. This will be the input to a combination of an anti-aliasing filter and sampling module, which will combine to produce a further down-sampled signal. The signal will then be store at a buffer until all of the input has passed through the module. Two alternative ways to test this module will be to display the signal on a logic analyzer or inputting a known signal to the module and connecting the output to a speaker.

b. FFT Module (Driss)

For this portion of our system, we will use the FFT module provided in the Vivado library. In order to test this we will input a single frequency sine wave and write a test bench to see the output, which should only be nonzero in only one frequency value.

c. Peak Extraction and Mel Scale Conversion (Ekin)

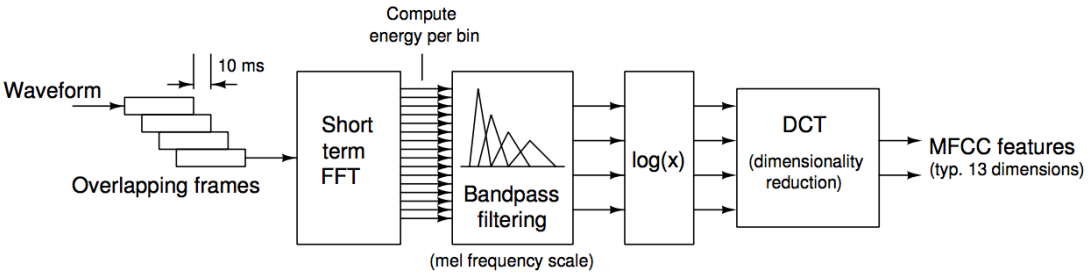
This module will take the Fourier transform of our input signal as an input and output a feature vector. The feature vector will be extracted by identifying the 10 frequency components in the FFT that have the highest power (the square of the $X(w)^2$). These frequencies will the be converted into the Mel scale calculated with the following formula (using the Mel lookup table):

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

The calculated mel values will be ordered in increasing value creating the feature vector. In order to test this module we will compare the result on the internal logic analyzer with pre-computed values in Matlab.

d. MFCC Module (Driss and Ekin)

In our second design option, an MFCC module replaces the FFT module and the peak extraction and mel scale conversion module. Here is the block diagram of the MFCC module:



MFCC Block Diagram(courtesy of Michael Price)

The MFCC module in the following steps:

1. The incoming signal is divided into 20-40ms frames with a 10ms gap between the starting points of the frames. This allows for some overlap between frames. Lets assume that the incoming signal was called $s(n)$. Then the signal with frames will be in the form of $s_i(n)$ where i denoted the frame number and n refers to the elements inside the frames.

2. Afterwards, and FFT is computed for each of the frames, producing $S_i(k)$ for each of the frames. Afterwards these values are converted into power spectrum using the following conversion:
$$P_i(k) = \frac{1}{N}|S_i(k)|^2$$

3. The next step is bandpass filtering. A set of triangular filters is calculated using the Mel-spaced filterbank. These filters are applied the each of the power spectrums calculated in the previous part. The filterbank energies are calculated by summing the coefficients of the previous result. This results in a set of filterbank energies for each of the frames.

4. The filterbank energies calculated in the previous step are converted into a log scale.

5. A discrete cosine transform is performed on the log filterbank energies creating a set of MFCC feature vectors for each of the frames. These MFCC feature vectors will be the output of the feature extraction module.

In order to test this module we will compare the result on the internal logic analyzer with pre-computed values in Matlab.

3. Speech Recognition Module

The speech recognition module consists of a command detection module and the trained samples module.

a. Trained Samples Module (Driss and Ekin)

The feature vectors of our trained samples will be computed on MATLAB from prerecorded audio samples. These will then be hardcoded into our system and stored in a ROM.

b. Feature Detection Module (Driss)

This module has two inputs: the feature vector obtained from the current audio input and the set of trained samples. The main way we will match an audio input to a command will be to compute the Euclidian distance between the feature vector of the audio input and that of the trained samples. The command that is the closest to the audio input will be chosen. Our reach goal would be to instead using the MFCC of the audio input and compare it with that of the trained samples using the Dynamic Time Warp (DTW) algorithm. Another alternative to using the Euclidian Distance would be to instead use a statistical model to compare the feature vectors such as the Gaussian Mixture model. In order to test this module, we will use one of the training samples as input. We will connect each command to and led and allow the led to blink when that command is executed.

4. Car System Module

a. Car System FSM (Ekin)

The car system FSM will take the output of the speech recognition module, which is a command as input. This command might cause the state machine to transition from one state to another. Each state in the FSM produces a different audio output and video output through the XVGA. The FSM will be tested separately from any of the other modules. The commands will be inputted using switches and the states will be displayed on the LCD display.

b. SD Card Interface (Driss)

The car system FSM will communicate with an SD card. The SD card will contain the pictures to be displayed in a bit format. These will be computed using MATLAB. It will also contain songs in a

digital format. A DAC is needed in order to play these songs on speakers.

4. Goals and Objectives

Our main goal in this project is to implement a common and widely used software algorithm for speech recognition on hardware.

1. Base Goal:

   Our base goals will be to create a car system that can be controlled by switches. This car system will have many functions and many different VGA displays.

2. Expected Goal:

   Our expected goal is to implement a speech recognition system that can identify basic commands consisting of one word spoken by a single speaker. The system should at the least be able to identify the training samples played back to it. This speech recognition system will be combined with the car system FSM.

3. Stretch Goal:

   Our stretch goal is to implement a speech recognition system that can identify basic commands spoken by a group of speakers or any speaker.

5. Project Timeline

	Implementation	**Testing**
Week of Nov 7	Implement both designs on Matlab. (Ekin and Driss)	Test the Implementation on Matlab with different training and test samples. (Ekin and Driss)
Week of Nov 14	Audio sampling module and FFT module (Driss) Car System FSM (Ekin) Graphics for VGA (Ekin and Driss)	Car System FSM (Ekin and Driss)
Week of Nov 21	Feature Extraction Module (Ekin) Speech Recognition Module (Driss)	Feature Extraction Module (Driss)
Week of Nov 28	MFCC Module-Reach Goal (Ekin and Driss)	Debugging for all system
Week of Dec 5	MFCC Module (if not finished)	Debugging with MFCC module
Week of Dec 12		Debugging for Checkoff

6. Conclusion

Implementing a voice controlled car system is a very interesting project because it allows us to explore our areas of interest and also create a system that is very useful and widely used. This also allows us to combine a well-known software algorithm and implement it on hardware. Although it is a very challenging project, we hope that we will gain a lot of experience and improve our knowledge in many aspects at the end of this project.