6.111 Final Project Proposal

iSing Voice Harmonizer

Cyril Lan, Jessie Li, Darren Yin

November 9, 2009

1 Overall System Description

Our basic system consists of three main components: pitch detection of the sung note using a FFT, pitch shifting the note several times to match each of the keyboard inputs, and adding the pitch shifted signals into one signal that is sent to the speakers. A block diagram of our system is in Figure 1.

When the user presses x keys simultaneously on the keyboard, the channel controller updates a memory that stores all possible midi frequencies. The CPU reads from the memory and sends the pressed midi frequencies to the pitch shifter module one at a time. The pitch shifter module uses this midi frequency as the target frequency to shift to and uses the data from the forward FFT to pitch shift the true frequencies of the sung note. The pitch shifter module then sends the pitch shifted frequencies to the FFT module which performs an inverse FFT to create the time domain signal. The CPU controls when the Adder should add up all the time domain signals.

2 Description of Each Module

2.1 FFT (Cyril)

The FFT module uses a butterfly circuit to implement the Discrete FFT. We plan to reuse the FFT module to perform both the forward and inverse FFT. To accomplish that, we will have the FFT module take a selector bit for whether to perform a forward or reverse transform. When the FFT is processing input from the microphone, the selector bit will be low, and the FFT will transform from the time domain to the frequency domain. When the FFT is processing input from the pitch shifter, the selector bit will be high, and the FFT will transform from the frequency domain to the time domain.

The frame size and frame offset of the FFT will be determined based on how much frequency domain resolution we require and how quickly the FFT can be performed.
A memory module will be used to store output from the FFT module. The memory module will also serve as the input from the FFT to the pitch shifter module. The FFT will serially write data to the memory module, and the pitch shifter will serially read data from the module. Read and write phases will be controlled by the CPU.

The module will be tested first on signals with known frequency such as a simple sine wave and then will be tested on more complex signals with variable frequencies.

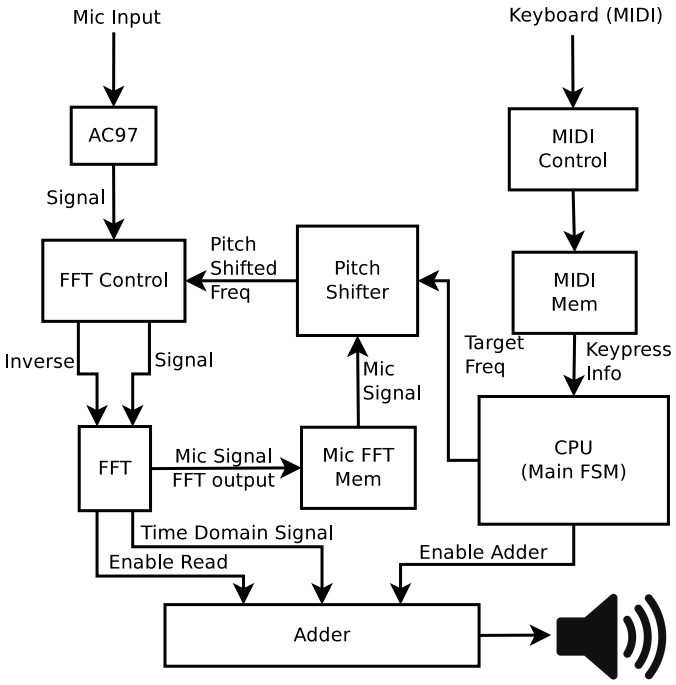

Figure 1: Block diagram of entire system

2.2 Channel Controller (Darren)

The Channel Controller Module takes as input the keyboard midi inputs and writes to a memory with the number of locations equal to the number of keys on the keyboard. If the key corresponding to a particular midi frequency is pressed, there is a 1 in the memory slot. Otherwise, there is a 0.

We plan to test the channel controller by playing a series of keys on the keyboard and viewing the contents of the memory using a logic analyzer.

2.3 Pitch Shifter (Jessie)

The pitch shifter module takes as inputs the target pitch from the channel module, the microphone signal, the frequency, magnitude, and phase outputs from the FFT, and returns as output a shifted signal.

The pitch shifter uses the short time fourier transform which breaks up the time domain signal into different frames and applies the DFT to each frame. From the DFT, we obtain a series of discrete frequencies known as bin frequencies, and each frequency has a corresponding magnitude and phase.

If we think of our time domain signal as a sum of partial sinusoids of different frequencies, we need to find some way of obtaining the frequencies of these sinusoids using the bin frequencies, magnitudes, and phases. If a partial sinuosoid has a frequency that is different from a bin frequency, a phase offset may result from one frame to the next. This phase offset information allows us to compute the true frequency of the partial sinusoid using the following formula

$$F_i = \frac{R}{N}(i + P_i \frac{S}{2\pi}) \tag{1}$$

R is the sampling rate of the signal; N is the DFT frame size in number of samples, S is the overlap factor which tells to what extent two successive frames overlap, P_i is the phase of bin i and F_i is the true frequency for the ith partial sinusoid. Overlapping frames is necessary because we want to have a wide range over which the true frequency of each partial sinusoid can vary. A typical overlap factor is at least 4, which means that two frames overlap by at least 75%.

Once we obtain the true frequencies of each of the partial sinusoids, we simply multiply each true frequency by a pitch-shift factor to obtain the pitch-shifted frequencies. We then synhesize the real and imaginary parts of the fourier coefficients. Finally, we apply the inverse DFT to construct the partial sinusoids, and we add them together to obtain our pitch shifted time domain signal. A diagram of the main logic pieces of our module is in Figure 2.

The pitch shifter will primarily be tested in Modelsim on mock inputs and later tested on real inputs during integration testing.

2.4 CPU (All of us)

The CPU is the main FSM that controls all of the slave modules such as the fft and the pitch shifter. The CPU reads from the midi frequency memory and sends the midi frequency to the pitch shifter. The CPU also controls when the adder should add up all of the time domain signals from the FFT.

2.5 Adder (Darren)

The Adder module takes as input the time domain signals from the FFT module when it's in inverse mode and combines them into one signal when the CPU sends the Adder an enable signal.

The Adder module will be tested in Modelsim and later integrated into the system.

3 List of Parts

- Microphone

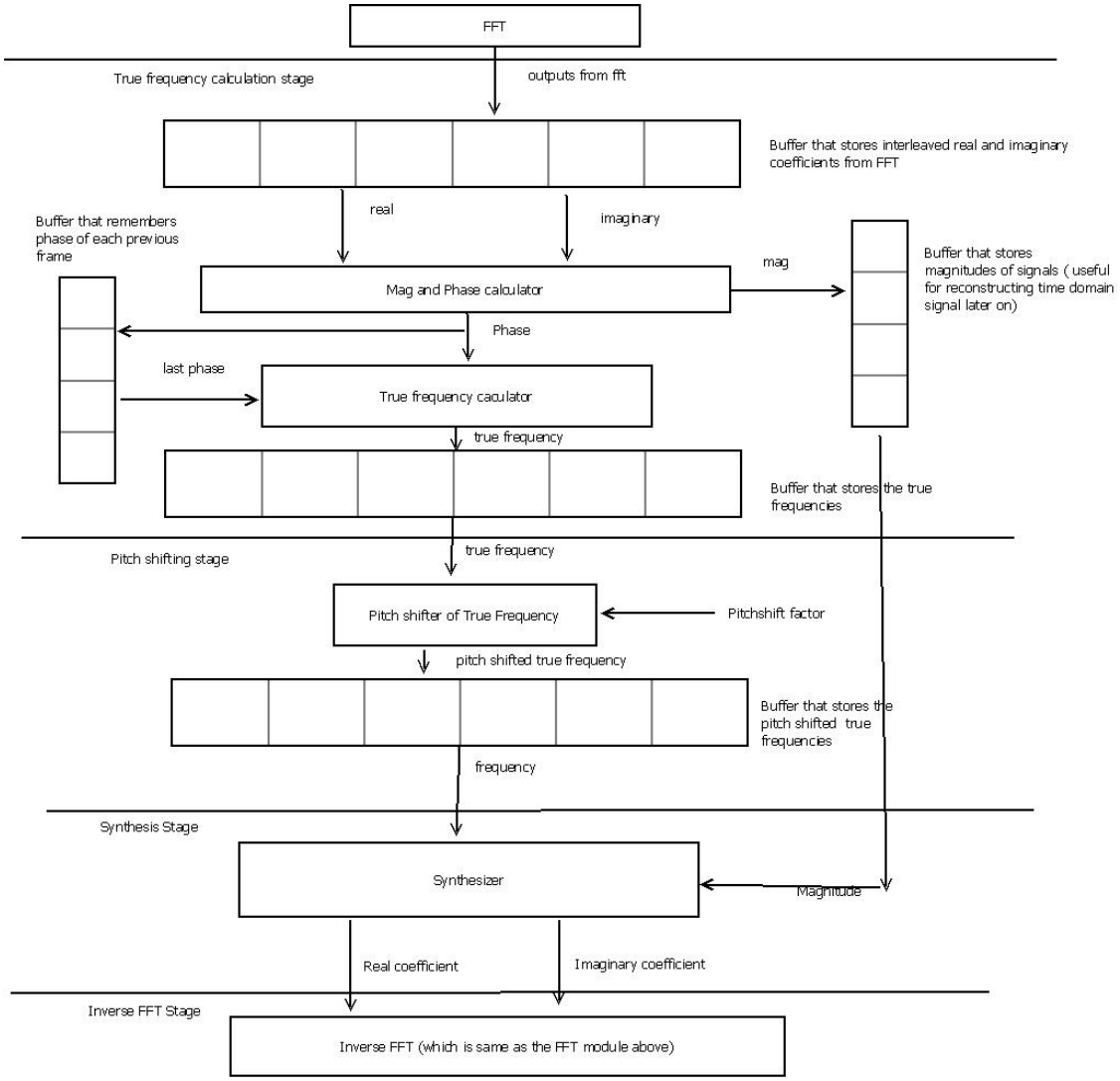

Figure 2: Block Diagram of Pitch Shifter

- Microphone stand
- Speakers
- Keyboard