6.111 Project Proposal

Liam Cohen[1], Sarah Flanagan[2], Zoe Klawans[3]

[1]Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

1 Overview

For our project, we plan to develop a digital system which projects a 1.5 octave piano, of fixed dimension, onto the floor, and is played using one's feet. This system will interface a projector, a line array of distance sensors, and an FPGA to accomplish this task. There are three main modules to this device: the *visual module*, the *sensor module*, and the *sound module*. Each module is designed such that they minimally rely on each other so that parallel development is possible. This will be made apparent in the block diagram found in (2.2). Furthermore, although a high level overview of each module is provided below, a more detailed account can be found in (3.1).

The first module - the visual module - will require the FPGA to interface with some form of VGA output. The main responsibility of this module will be to display, via a projector or a computer monitor, a dynamic image of a 1.5 octave keyboard. The image should be updated in response to information from the sensor module indicating which key was pressed. Zoe Klawans will be responsible for implementing this module.

The second module - the sensor module - will interface with an array of time-of-flight distance sensors. The main responsibility of this module is to use the information from the TOFs and have the FPGA generate a finite-state-machine which converts the given information to a number indicating which key was pressed. This information is updated periodically and made available to the other two modules. Liam Cohen willl be responsibile for implementing this module.

The third module - the sound module - will take as input from the sensor module a number, corresponding to some key press. This module will then use the FPGA to interface with the AC97 codec to generate a tone corresponding to the right note for the given key press. Sarah Flanagan will be responsible for implementing this module.

This represents the basic functionality of our project. The nuances of each module, again will be detailed further in (3.1).

2 Design

2.1 Goals and Scope

2.1.1 Baseline Goals

1. Have FPGA generate (make image out of drawn blocks on FPGA) 17 key piano image, from C to E, on computer monitor.

2. Have FPGA dynamically change color of keys to indicate key press.

3. Have FPGA read out 1 note at a time from the sensor module.
4. Have FPGA read out sensor module at 10Hz.
5. Have FPGA generate a 5-bit tone given some key press.

2.1.2 Expected Goals

1. Use FPGA to display image of the keyboard on a projector: may turn into stretch goal depending on difficulty arising from skew correction and changes to dynamic updating to not display image over one's foot.
2. Implement a recording feature which stores a played note sequence in FPGA memory
3. Implement a playback feature which plays back a stored recording
4. Implement octave modulation - i.e., changing center frequency of octave
5. Have FPGA recognize 2 notes simultaneously

2.1.3 Stretch Goals

1. Use images stored in memory to generate a more realistic looking piano
2. Convert FPGA button presses into 'buttons' as defined by the TOF sensors - i.e. convert hard interface to a soft interface
3. Display soft buttons on the projector
4. Display playback on the projector as if the keys were being played by the user
5. Have FPGA Recognize up to 17 notes simultaneously
6. Play instrument samples from memory instead of 5-bit tone
7. Implement button to change instrument sample type

2.2 Block Diagram

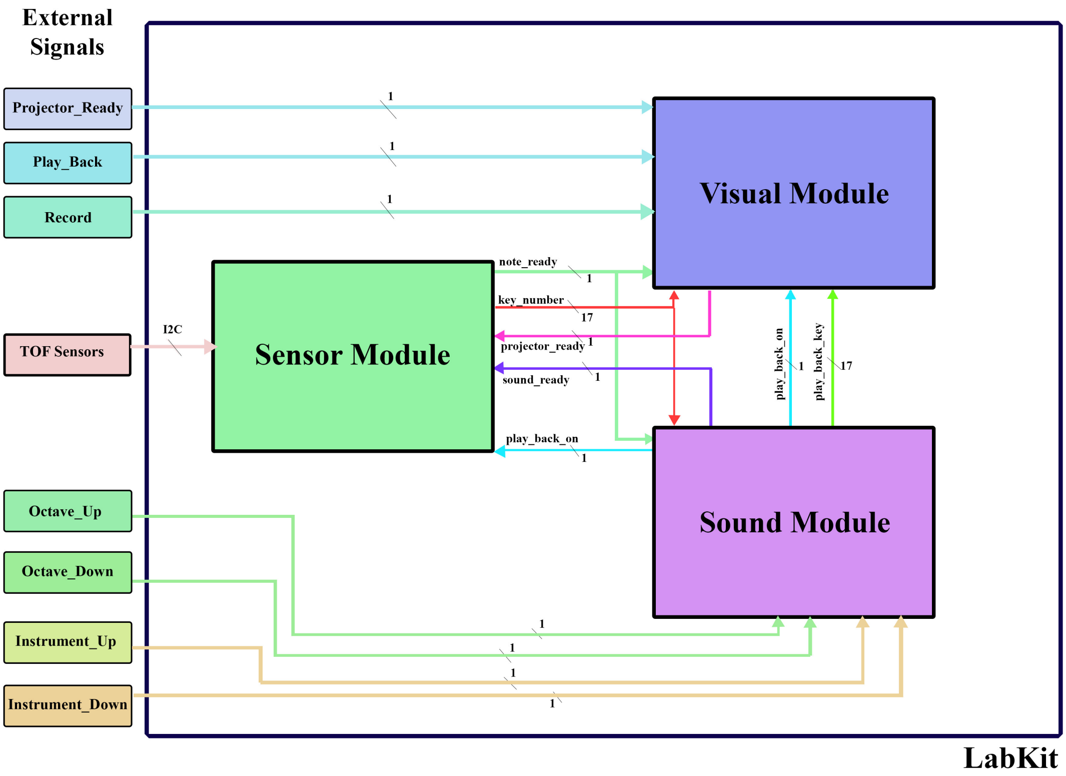
3 Implementation

3.1 Module Descriptions

3.1.1 Visual Module

The visual module will display images of a 1.5 octave keyboard, updating the image to show which key(s) are being pressed. It will receive the information about which note(s) are being played either from the sound module (during playback mode), or from the sensor module (during user or record mode). The visual module will use images of a keyboard stored in memory to update the VGA output based on key press information, leading to memory size limitations. The visual module will interface with the projector to create the floor piano. Depending on projector update rates, this may require skew correction for dynamic image updating. The projected image will also include buttons which can be pressed for functionalities such as recording, playback, and changing instrument sample type.

Figure 1: Projected Piano Block Diagram



Testing the visual module will involve both writing test jigs and conducting physical tests. Since there is a lot of imagery involved, physical tests will involve examining the piano images to make sure they look correct and to see if there is noticeable delay in image changes after key presses, and measuring projected image sizes to see if they match our expected dimensions. Test jigs will test that all internal signals are getting sent at the correct clock edges.

3.1.2 Sensor Module

The sensor module will gather information from seven time of flight distance sensors via I2C protocol. Taking advantage of the fact that each TOF sensor has a 25 degree cone of sight, using both the distance away from the sensor and conic intersection of two sensors, we can get all 17 keys on the piano using only 7 sensors.

The time of flight sensors take about 50ms to integrate for a single high accuracy distance measurement. This implies that any sensor query over I2C will be of a much smaller timescale than the integration time. What this really means from a design perspective is that, at a 400Kbs I2C communication speed, addressing seven slaves will take only a fraction of the time to integrate each measurement. Having a 20Hz refresh rate for updating note values to the visual and sound modules is okay because 20Hz corresponds to a song being played at 1200 beats per minute, which a human is unable to play.

The internals of the sensor module will include an I2C IP core taking care of the communication protocol. It will also contain logic for converting sensor readings into a 17 bit number indicating which keys are pressed. In addition, the sensor module will internally have modules for using the I2C IP core to specifically talk to the TOF sensor array to both configure the individual sensors and acquire the raw sensor data.

Each sensor will be acquired from adafruit or poulolu for roughly 15 dollars a sensor. For a total rough cost of 105 dollars. Adafruit supplies the VL53L0X in a breakout board which automatically implements a MOSFET based level shifter to allow the labkit I2C voltage to communicate with the 2.8V I2C voltage on the TOF sensor. This implies that no additional hardware should be needed to interface the labkit with the TOF sensor array.

This module is physical in nature and requires very minimal input from the other two modules. All inputs to the sensor module from other modules are single bit busses and can be imitated by switches onboard the FPGA. The details of this module can tested in steps. The first step is to guarantee the FPGA can use the I2C IP core to accurately address the sensor. The second step would be to guarantee the FPGA can multiplex between reading all seven sensors in the TOF array. The third step would be to make sure that the conversion between sensor readings and note values is accurate. This can be tested by printing out a mock version of the piano and aligning the TOF sensor array appropriately and physically testing whether or not the correct key is detected. Other aspects of the module, such as output signals to other modules, can be tested in simulation. Furthermore, the main aspect of the sensor module, the conversion from distance and sensor number to key press, can also be tested in simulation at least for functionality. This means that simulation can guarantee that the module is up to spec and is processing the information correctly. If an error occurs in real hardware, simulation can rule out an error in the module coding, and reveal an error in the "conversion" methodology.

3.1.3 Sound Module

The sound module will have 3 modes, user mode, recording mode, and playback mode. While in user mode the module will take inputs from the sensor module indicating note ready and what key is pressed. Based on this input, it will determine what frequency of tone should be generated. It will send this frequency to a module that produces the appropriate tone. This will be similar to the tone750hz module from Lab5. The tone will be sent to the ac97 module, and will continue to play as long as it is receiving the same KeyNumber and the NoteReady signal from the sensor. This is part of our baseline goal for the project.

As an expected goal, there will also be a recording mode. In this mode, as a key input and note ready input is received the note will both be played and stored into memory. This will be able to record a couple seconds of the note being played. There will also be a playback mode. In this mode, the memory will be accessed to replay the recorded notes.

One major limitation is memory, which will affect how long a recording can be. This will limit a recording to a couple seconds. In order to test the module, I will have a test module that sends mock NoteReady and KeyNumber inputs to the module and will see if the module can play the appropriate notes in user mode as well as testing the functionality of recording mode and playback mode.