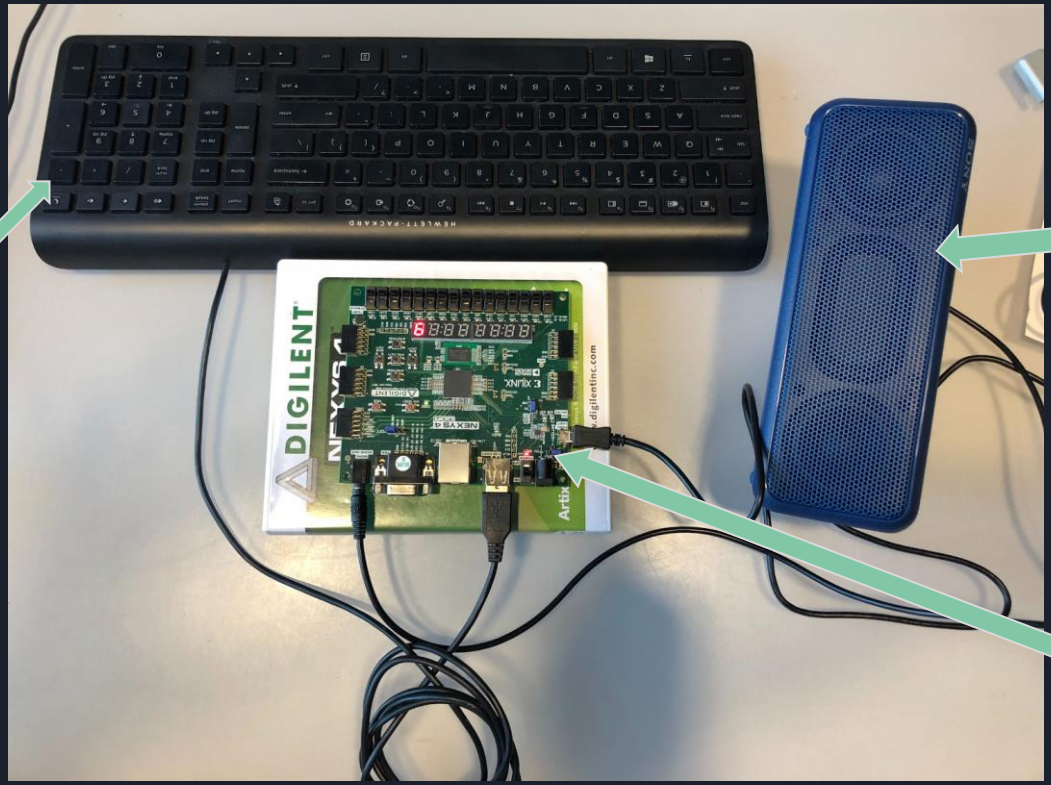
Piano Simulator with Keyboard and Accelerometer Interface

Matthew Bayer
Nick Deneau
James Khoury
Logan Verstraete

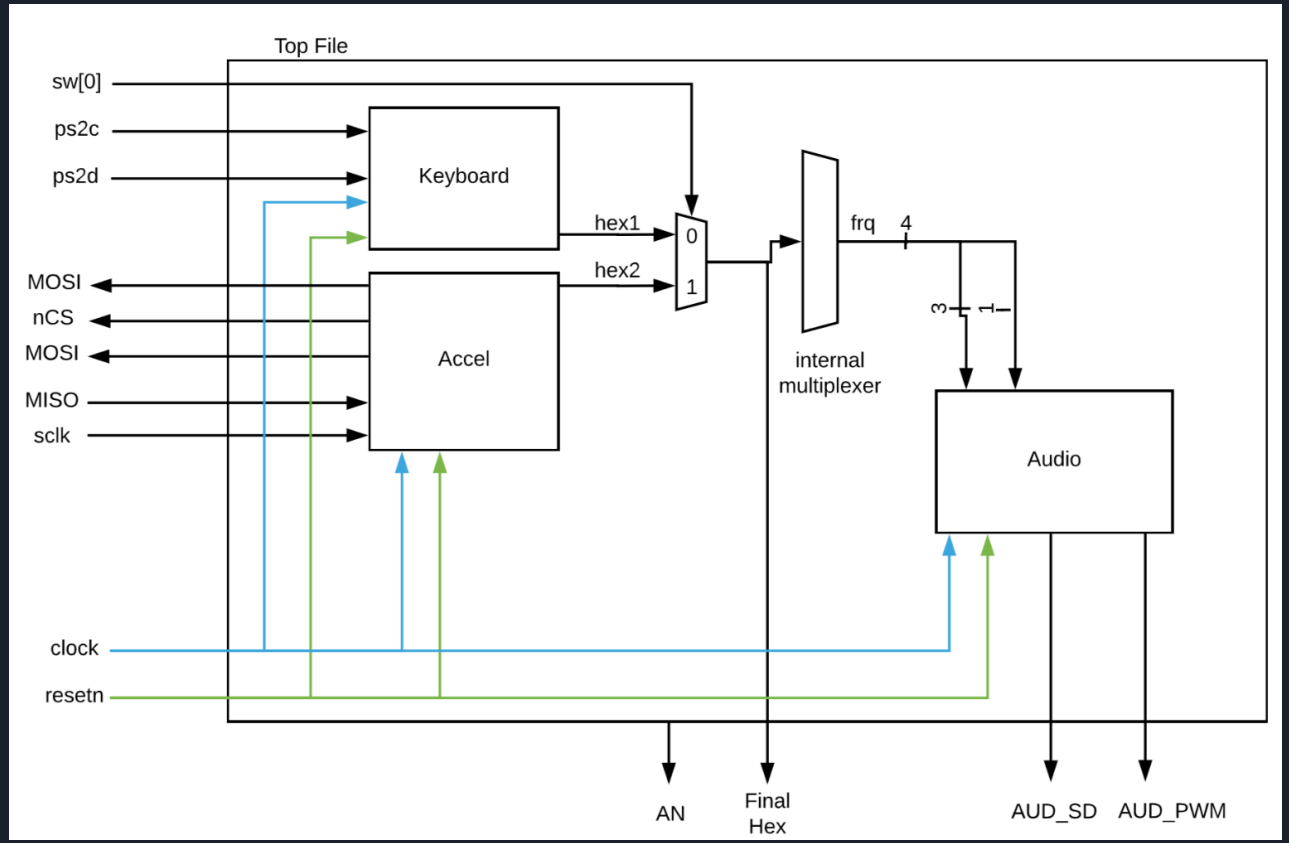
Hardware Setup



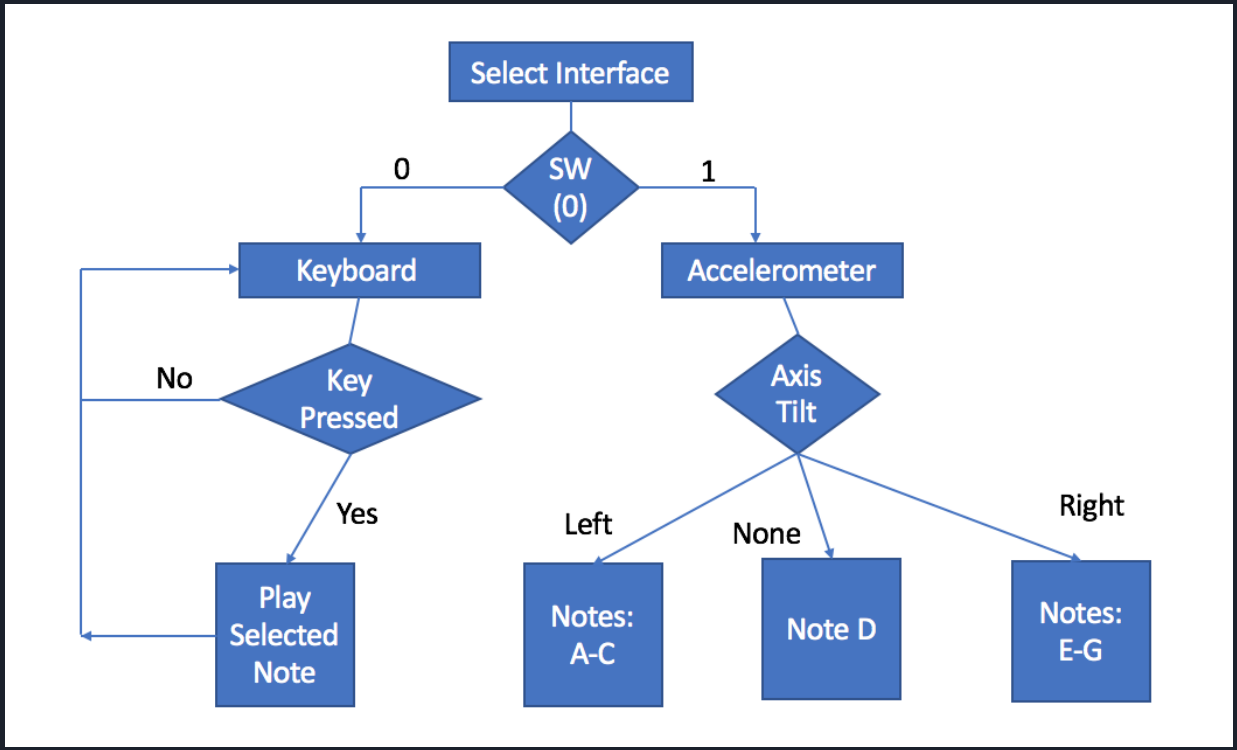
Purpose

- Create a piano simulator using the Nexys4 board’s audio output and a ps2 interface keyboard
- Use the Nexys4 board’s built in accelerometer to control the audio output
- Can choose between either keyboard mode or accelerometer mode with switches

Block Diagram



Flow Chart


Select Interface
SW (0)
0
1
Keyboard
Accelerometer
No
Key Pressed
Yes
Play Selected Note
Axis Tilt
Left
None
Right
Notes: A-C
Note D
Notes: E-G

Code

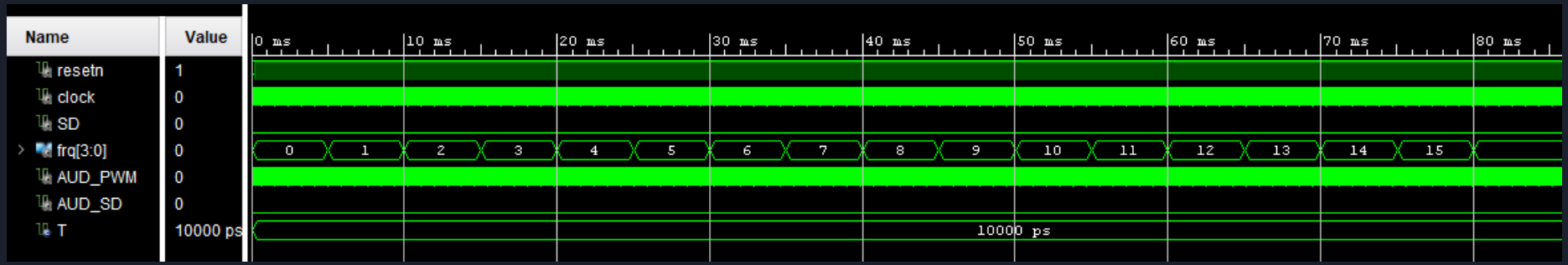
Four main files

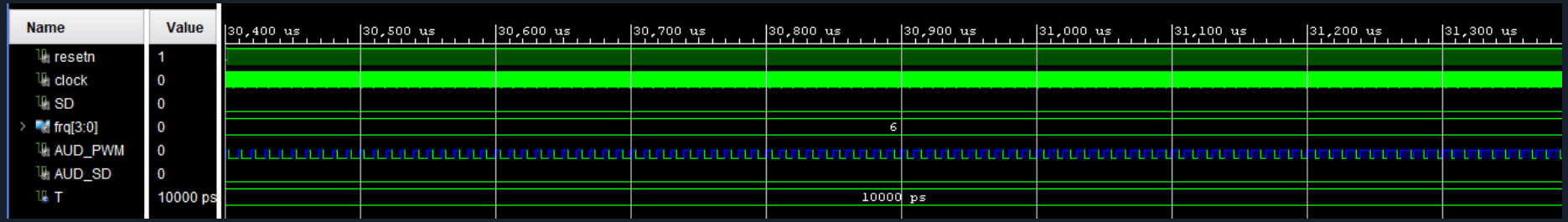
1. pwm_audio: generates the note frequencies and outputs the sound
2. ps2keyboard_top: takes user input and outputs a sound based on what key was pressed and displays the corresponding piano key on the seven segment display
3. top_accel: using only the x-axis accelerometer the Nexys board will output sounds based on the movement in the x direction. The 0 position is note d.
4. Project_top: connects the three files above in order to make the project work as one.

Sound

- The code produces tones by changing the time it takes for the duty cycle to go from 0 to 255
  - An equation dependent on the variable “frq” determines the time from 0-255
  - Changing the equation will produce a different frequency range
- The code was modified to have 16 tones instead of the original 8, then the frequency range of the tones was adjusted to produce better sounding notes
  - A variable “frq” from 0-15
  - Only 7 of the 16 tones were selected due to overall code design
- The duration of the note is determined by the number of times the duty cycle value reaches 255
  - This causes the lower frequency notes to be longer
- Notes can be played before another ends
  - The 0-255 count will reset if the current frq does not match the previous frq

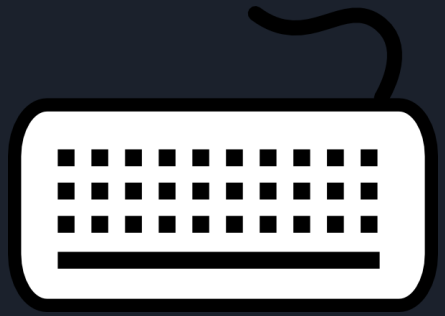
Sound Simulation

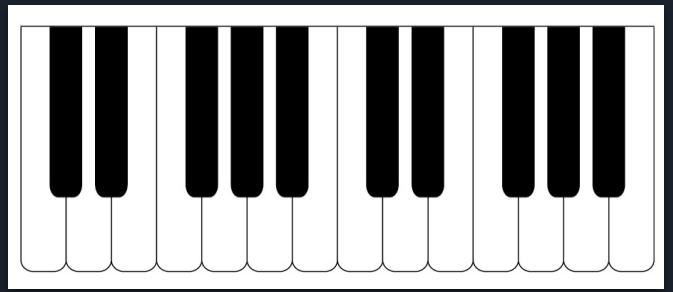




Keyboard

- Keystonotes: Inside this file the team set seven 8-bit numbers to a 7-bit number used for the seven segment display as well as for the frequency
- Ps2keyboard: Shifts 1 bit from ps2d until that data has 8 bits. Those 8 bits correlate to certain key on the actual keyboard





Accelerometer

1. FSM_accel: Used from Lab 3, be able to give correct input data into decoder
2. Wr_reg_axl362: Used to interface with from FPGA and accelerometer chip
3. Decoder: Used from Lab 3, be able to enable the correct register(data of x-axis)

Improvements

- Tuning the synthesizer to produce a full octave of notes from C-B, like on a piano
- Adding more keys, having 2 or 3 full octaves
- Having the notes be the same length, instead of the lower frequencies being longer

Demonstration!!!

Questions?