6.111 Final Project: Digital Debussy-A Hardware Music Composition Tool

Jordan Addison and Erin Ibarra
November 6, 2014
Purpose

- Professional music composition software is expensive
  - $150-$600, typically between $300-$600
  - Amateurs, beginners do not need the power and functionality of professional software

- Very little information on internal details of music composition software
Outline of Minimum Features

- Single note melodies with up to 64 individual notes and playback
- Bass and treble clefs
- Two time signatures: 3/4, 4/4
- Transcribe pitches from C3-C5
  - One octave below Middle C to one octave above Middle C
- 1 Key Signature (C major) but all flats and sharps available as accidentals
- Note duration from whole to sixteenth notes
- Dynamic volume from piano to forte
User Interface Block Diagram

Labkit Buttons → UI Control FSM → Display Storage

Display Storage → Note BRAM Storage

Note BRAM Storage → Graphics Display

Graphics Display → VGA Output

Note- 16 bits

~Pitch~ Octave Duration Dot Accidental 5 Open
User Interface: Transcription Process

- **User Interactions**
  - Music transcribed from left to right using labkit buttons
  - Select parameters on toolbar
  - Switch between “Staff” and “Toolbar” modes using “Enter” button
  - Place a note based on chosen parameters by pressing “1” button in staff mode

- **Display**
  - Entered notes are stored in note memory
  - Addresses of notes and duration values for corresponding notes are stored in measure counter
    - Measures are divided accordingly on display
  - Pitches are displayed as sprites and mapped to appropriate location on staff
UI Testing

- Using generated note files
- Test benches with simulated button pushes
- Visual inspection
Button Functionality

**Toolbar**
- **Up/Down**: cycle through attributes (pitch, duration, etc.)
- **Left/Right**: cycle through values of a specific attribute (i.e. pitch: A,B,C, etc.)
- **Enter**: switch to staff mode
- **3**: playback (press and hold)

**Staff**
- **Left/Right**: scroll through existing notes on the staff
- **0**: erase the highlighted note (turns into a rest)
- **Enter**: switches to toolbar mode
- **3**: playback (press and hold)
- **1**: place a note
Pitch and Speed

- Seem to be inherently linked
- Higher Speed $\rightarrow$ Higher Pitch, Slower Speed $\rightarrow$ Lower Pitch

Standard Solution:
- Window audio sample
- Analyze pitch in frequency domain
- Analyze tempo in time domain
- “Phase Vocoder”
Controlling Speed

- To change playback speed, change amount of overlap between windowed segments
- No effect on pitch
- Can have other effects on the sound
Pitch Shifting: Concept

440Hz ‘A4’ → FFT → Impulse at 440hz

Impulse at 440hz → Shift by X2 → Impulse at 880hz

Impulse at 880hz → IFFT → 880Hz ‘A5’
Controlling Pitch: Issues

FFT Mapping

Frequency Bins
Controlling Pitch: Solutions

- Audio signals are “short-time stationary”
- Adjacent signals are NEARLY identical
- But differ enough to use the phase difference to extract a very precise estimate of the “true frequency” of a bin
- Without phase information, looks like a 215Hz signal (5Hz error)
- With phase info, estimated frequency is 220.0219! ($\Delta t = 3ms$)
Implementation Scheme

- Compute FFTs of sound clips ahead of time in MATLAB
- Store FFT data in FPGA BRAM
- Implement pitch shifting/time dilation on FPGA

Other considerations:
- Proper note shaping (Attack, sustain, release), will likely use simple shaping filters if needed
- Pitch shifting only has a 2 octave range
- Time scaling is quite limited with this implementation
Playback Architecture Overview

- **Note BRAM Storage**
  - [15:0]
  - Play Button

- **Playback Control**
  - Sound Clip FFT BRAM

- **Pitch Shifter**
  - IFFT

- **Time Stretching/Compression**
  - ac97
Playback Testing

- MATLAB generated test files to put in BRAM
- Use these for unit testing
  - Time Shift
  - Pitch Shift
  - Volume Control
## Timeline

<table>
<thead>
<tr>
<th>Week 1</th>
<th>Week 2</th>
<th>Week 3</th>
<th>Week 4</th>
<th>Week 5</th>
<th>Week 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/2</td>
<td>11/5</td>
<td>11/9</td>
<td>11/12</td>
<td>11/16</td>
<td>11/19</td>
</tr>
<tr>
<td>11/16</td>
<td>11/19</td>
<td>11/23</td>
<td>11/26</td>
<td>11/30</td>
<td>12/3</td>
</tr>
<tr>
<td>12/3</td>
<td>12/7</td>
<td>12/10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **User Interface**
  - FSMs/Internal Structure
  - GUI/External Structure
  - Troubleshooting/Debug

- **Playback**
  - MATLAB Prototyping
  - Verilog Time Shifting
  - Verilog Pitch Shifting

- **Integration**
  - Integration
  - Integration