Phantom Sight Reader

Lance Collins, Jing Han
Dilini Warnakulasuriyarachchi
11/12/08
Agenda

• Overview
• Block Diagram
• Camera / User interface
• Note Decoder
• Audio Generation
• Timeline
• Conclusion
Overview

Camera/user interface → Note Detection → Audio Output

C half note, A whole note…
Camera/user interface - Jing

- Converting Y,Cr,Cb to RGB -> display
- Freeze image
- Filter – image B&W only, no grayscale
- Orientation of sheet music on camera
- User interface – underline note being played
Note Decoder Overview - Dilini

• Three main components:
  – The Staff Identifier
  – The Note Identifier
  – The Beat Identifier

• Minor FSM controls the overall function
• Interacts with the ZBT and stores data in a Memory Array to be used by the player module
Filter-pixel

- Cleans the image captured from the camera to reduce pixel errors

Flowchart:
- ZBT Memory
- Filter
- BRAM

8-bit pixel value → Filter
1 bit pixel value
Note Decoder Module

Master FSM

Minor FSM

Beat Identifier

Staff Identifier

Note Identifier

Note Decoder

BRAM

Music Memory

Stored Image

Decode_enable, pixel values

Memory Address

Find_staff

Find_note

Find_beat

Notes
Minor Finite State Machine (FSM)

- Receives Decode_enable signal from the master FSM
- Control the sub modules in the note decoder module
- Comprised of three states:
  - Staff
  - Note
  - Beat
- Request data from the BRAM via Master FSM depending on the state
Staff Identifier Module

- Locates where each staff is situated in the image
- Waits for the Find_staff enable signal from the Minor FSM
- Locates one staff at a time and notes the location
- Once a staff is found notifies the Minor FSM by setting Find_staff signal to low
Note Identifier Module

- Identifies each note by evaluating the number of black pixels in each block between staff lines
- Activated by the Find_note signal from the Minor FSM
- The notes on the score sheet must be evenly spaced
Beat Identifier

- Identify the duration of each note
- Waits for the Find_beat signal from the Minor FSM
- Counts the number of black pixels in each column of a single staff

[Diagram of Beat Identifier system with nodes and arrows]
Audio Generation Overview - Lance

• Initialization
  – Music Memory filled with music data
  – Play signal from Master FSM

• Operation
  – Ready pulse from AC97
    • Begin computing note data
    • Pipelined design
    • On next ready pulse, output audio data
  – Compute note tones (amplitude and phase)
  – Apply Attack-Delay-Sustain-Release (ADSR) envelope
  – Combine notes together
Player Module

- Gets play signal from Master FSM
- Gets music data for a beat from Music Memory
- Enables the tone selector and indicates which notes are on to the Amplitude Calculator
Tone Selector Module

- Passes note index and harmonic index down to other modules in the pipeline
Tone Parameters Module

- Outputs the tone’s initial theta (phase) and theta increment (frequency) to the Theta Memory

Master FSM

Music Memory

Player

Audio Generator

Tone Selector

Tone Params

Theta Memory

Sine Generator

Note Generator

Note Aggregator

Music Memory

Address

Beat Notes

Enable

Notes On

Note Index

Note Scale

Amplitude Calculator

AC97

Ready

Audio Out
Theta Memory Module

- Increments and stores the theta value in memory as well as passing it to the Sine Generator.
Sine Generator Module

- Lookup table for sine values for a given theta value
Note Amplitude Module

- Gets the relative amplitude for a harmonic in a note and scales it based on the ADSR envelope for that note.
- Passes the resulting scale value to the Note Generator.

Audio Generator

- Note Amplitude Calculator
- Note Generator
- Sine Generator
- Theta Memory
- Tone Params
- Tone Selector

Player

- Master FSM
- Music Memory

AC97

- AC97 ready
- AC97 audio_out
- AC97 address
- AC97 beat_notes

Note Memory

- Note Memory
- Note Memory
- Note Memory
- Note Memory
- Note Memory

Note Selector

- Note Selector
- Note Selector
- Note Selector
- Note Selector
- Note Selector

Tone Parameters

- Tone Parameters
- Tone Parameters
- Tone Parameters
- Tone Parameters
- Tone Parameters

Music Memory

- Music Memory
- Music Memory
- Music Memory
- Music Memory
- Music Memory

Tone Selector

- Tone Selector
- Tone Selector
- Tone Selector
- Tone Selector
- Tone Selector
Note Generator Module

- Scales and aggregates the tones for each note and passes the complete note sample to the note aggregator.
Note Aggregator Module

- Scales each note to prevent overflow and adds them together
- Outputs sound data to the AC97 on the next ready pulse
## Timeline

<table>
<thead>
<tr>
<th></th>
<th>Dilini</th>
<th>Jing</th>
<th>Lance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Week 1</strong></td>
<td>1. Basic Pixel Concentration Note Recognition</td>
<td>1. Filter Module</td>
<td>1. Multiple Notes</td>
</tr>
<tr>
<td></td>
<td>2. Basic Pattern Matching Note Recognition</td>
<td>2. Orientation Module</td>
<td>2. Player Module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Apply ADSR envelopes</td>
</tr>
<tr>
<td><strong>Week 2</strong></td>
<td>1. ZBT interface</td>
<td>1. Underline notes as played</td>
<td>1. Extend to Multiple Instruments</td>
</tr>
<tr>
<td></td>
<td>2. Choose note detection methodology and expand</td>
<td>2. UI Elements</td>
<td></td>
</tr>
<tr>
<td><strong>Week 3</strong></td>
<td></td>
<td>Integration</td>
<td></td>
</tr>
<tr>
<td><strong>Week 4</strong></td>
<td></td>
<td>Debugging</td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

• Baseline
  – Reads a few staves of preformatted sheet music
  – Plays the sheet music using piano or other
  – Shows notes being played

• Optimal
  – Reads a pages of preformatted sheet music
  – Users customizes which instruments play various parts (UI and Audio components)