Music Visualization with Audio Beat-matching
Concept and Motivation

Music visualization only commercial available in software implementation
  > Wanted hardware implementation
  > Wanted more interesting visualization

Beat-matching visualization more interesting and challenging than audio spectrum analyzer
  > Would have to implement beat detection algorithm to find tempo and phase of music
  > Would have to create moving image that updates in real-time and changes with music

Visualization can be used for parties, hanging out, or other social events
Pre-existing Software Visualization

Software from Pico Technology
Stevie Wonder
You Are the Sunshine of My Life

Audio Glow
Music Visualizer,
Android App
Implementation

Pre-existing beat-matching algorithm
> Create implementation in Verilog
> Provides tempo and phase, or time until next beat

Two part visualization
> Color changing and rotating nested shapes that update in real-time
> Stretch goal: background Mandelbrot fractal visualization and alpha-blending with moving image
Color-changing and rotating nested shapes

Mandelbrot fractal background
Overall Block Diagram
Audio Block
Image Generator Block

- Button Input
- Fractal Generator
- Movement Generator
- Frequency
- Phase
- VGA out
- Hsync
- Vsync
- Vclock
- Hcount
- Vcount
- Blank
- SVGA
Movement Generator

Coordinate Generator

CORDIC rotation block

Coordinate Controller

Address Generator

Address Generator

Current Image SRAM

Next Image SRAM

SRAM Arbiter

Frequency [9:0]

SVGA hcount, vcount, vsync, etc.

current [20:0]

pixel [24:0]

Phase

Pixel [24:0]

fractal pixel [24:0]
Fractal Generator Block

- C, U, D, L, R
  - Button press

- Reset

- Mandelbrot controller

- Mandelbrot calculation engine
  - Mandelbrot calculation engine

- Flash memory

- BRAM buffer

- SVGA output controller
  - pixel [24:0]

- SVGA in (hysnc, vsync, etc.)
<table>
<thead>
<tr>
<th></th>
<th>11/2</th>
<th>11/9</th>
<th>11/16</th>
<th>11/23</th>
<th>11/30</th>
<th>12/7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finalize block diagram and shared signals</td>
<td>![Red Box]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Writing low level modules of visual/audio blocks</td>
<td></td>
<td></td>
<td>![Orange Box]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing and integrating low level modules</td>
<td></td>
<td>![Yellow Box]</td>
<td></td>
<td>![Green Box]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing and integration of visual and audio blocks, working on stretch goals</td>
<td></td>
<td></td>
<td>![Green Box]</td>
<td></td>
<td>![Blue Box]</td>
<td></td>
</tr>
<tr>
<td>Debugging and stretch goals</td>
<td></td>
<td></td>
<td></td>
<td>![Blue Box]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Checkoff, demo, report</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>![Purple Box]</td>
<td></td>
</tr>
</tbody>
</table>