Move-O-Phone

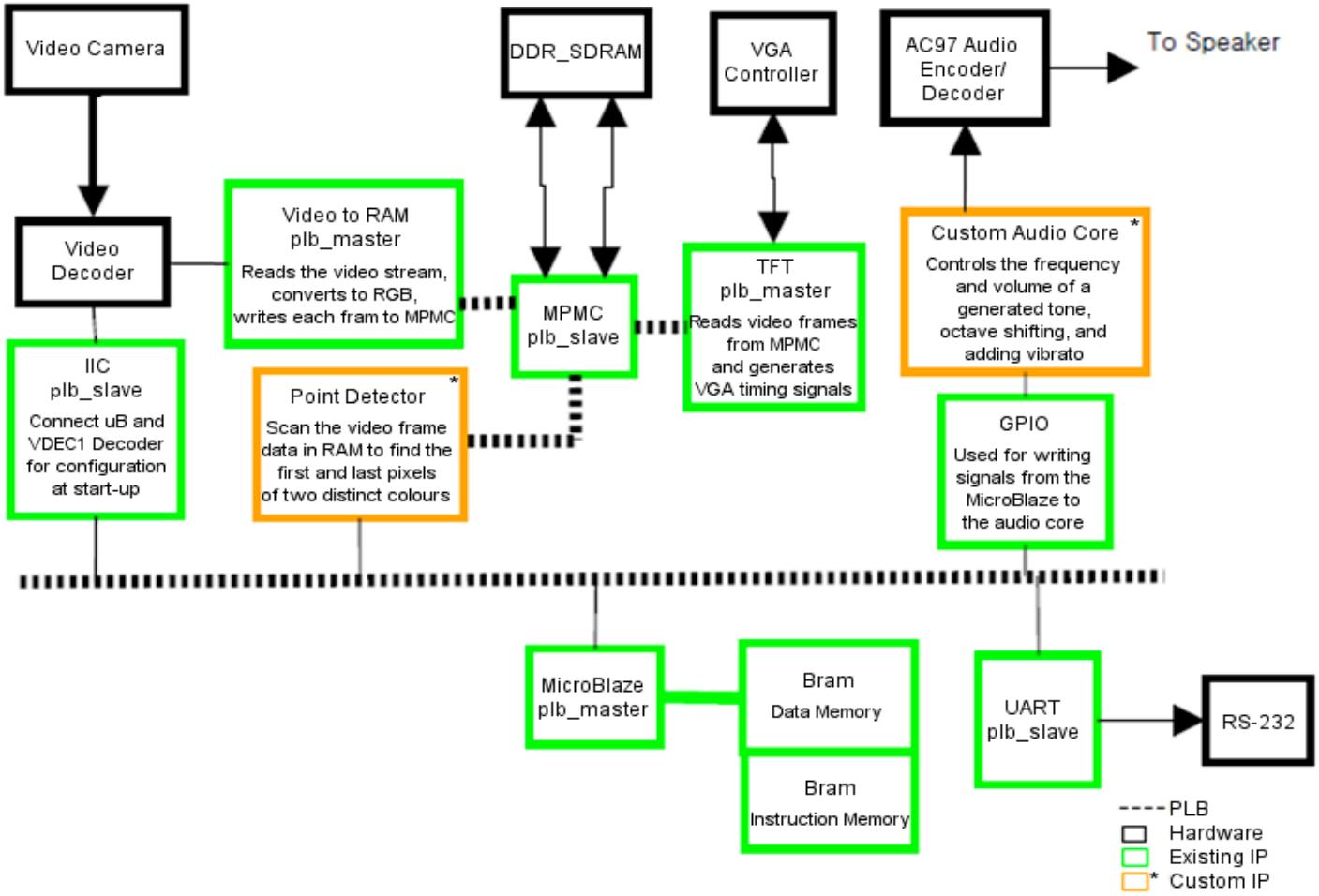
Motion Controlled Music

Cyu Yeol(Brian) Rhee
James Durst
Stuart Byma

Motivation and goals

- Creation of user interface based on movements
- Control a musical instrument, while removing a physical instrument from the equation.
- Initial goal: control musical tones by changing the angles between two points
- Modified goal: control tones of two different instruments with different sound effects.

System Block Diagram



Custom Hardware

- point_detector (modified from existing)
  - scan the video frame data in RAM to find the first and last instances (pixels) of two distinct colours over the entire screen
- audio core custom hardware logic block
  - control over the frequency and volume of a generated tone, shifting of the tone down one octave, and adding vibrato

Audio Core

- GPIO provides two memory mapped registers for control of left and right channels

Bit 31: Activate	Bit 30..24: Unused	Bit 23.. 4: Volume	Bit 3..0: Note

  - Activate – turn the channel on or off
  - Volume – level of attenuation
  - Note – 4 bit code for one of 16 semitones
- Board switches provide control of vibrato and octave range

Provided IP

- audio_0 AC97 interface
- video_to_ram (taken from previous project)
- video_out
- dlmb and dlmb_ctrl
- ilmb and ilmb_ctrl
- plb_v46
- DDR_SDRAM
- xps_gpio
- uart_uB

Software

- Video core initialization
  - Taken from previous project
- Reads coordinate data from RAM
- Translates coordinates to angles or distances, mapping those to musical notes
- Writes to audio core memory mapped registers to output sound

Design Process

- Each milestone completed every week
- Small steps taken rather than implementing large portions at once
- Backup copies
- Lots of testing the audio core with the ISE separately
- Used the LED debugger core in the project to check issues with the video processing.

Demo