Virtual Conducting

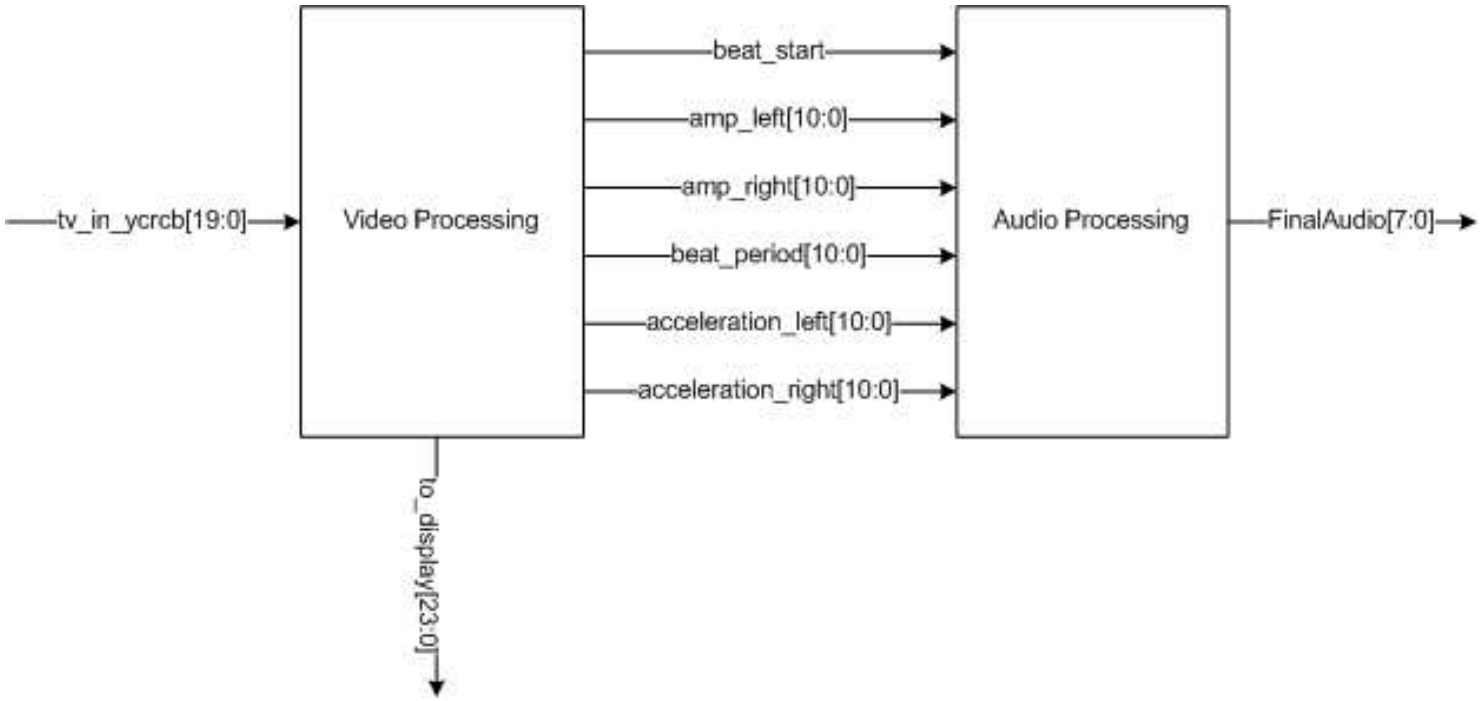
Andy Lin and Brandon Yoshimoto

Project Overview

- **Description:**
  - An interactive music player which allows the user to control the sound of a composition through hand movements.
  - The user conducts holding a blue LED in each hand
  - Movements are interpreted as beats and affect the playback of music
  - Music qualities controlled:
    - Volume: left hand controls low frequencies, right controls high frequencies
    - Articulation: left hand controls low frequencies, right controls high frequencies
    - Tempo: right hand controls the tempo of the piece
- **Inputs:**
  - Camera Video
  - Music
- **Outputs:**
  - Visualization on Monitor
  - Speakers

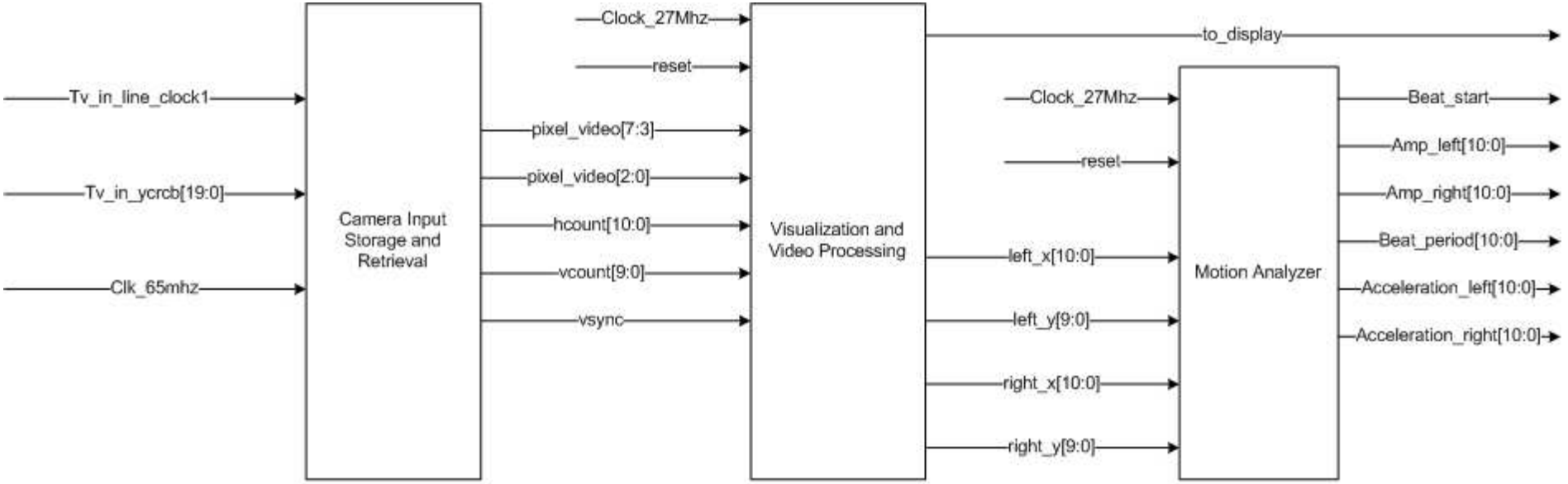
High-Level Description

- **Two main units:**
  - Video Processing
    - Inputs: Camera data
    - Outputs: Video display and movement qualities
  - Audio Processing
    - Inputs: Movement qualities and audio
    - Outputs: Processed audio

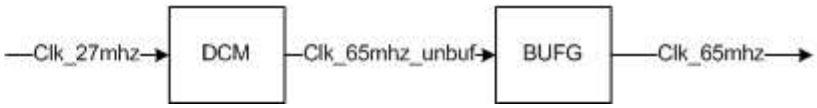


Video Unit

- **3 Main Parts:**
  - Camera Input Storage and Retrieval: Retrieves data from the camera
  - Visualization and Video Processing: Calculates position of the hands and displays on monitor
  - Motion Analyzer: Interprets hand movements

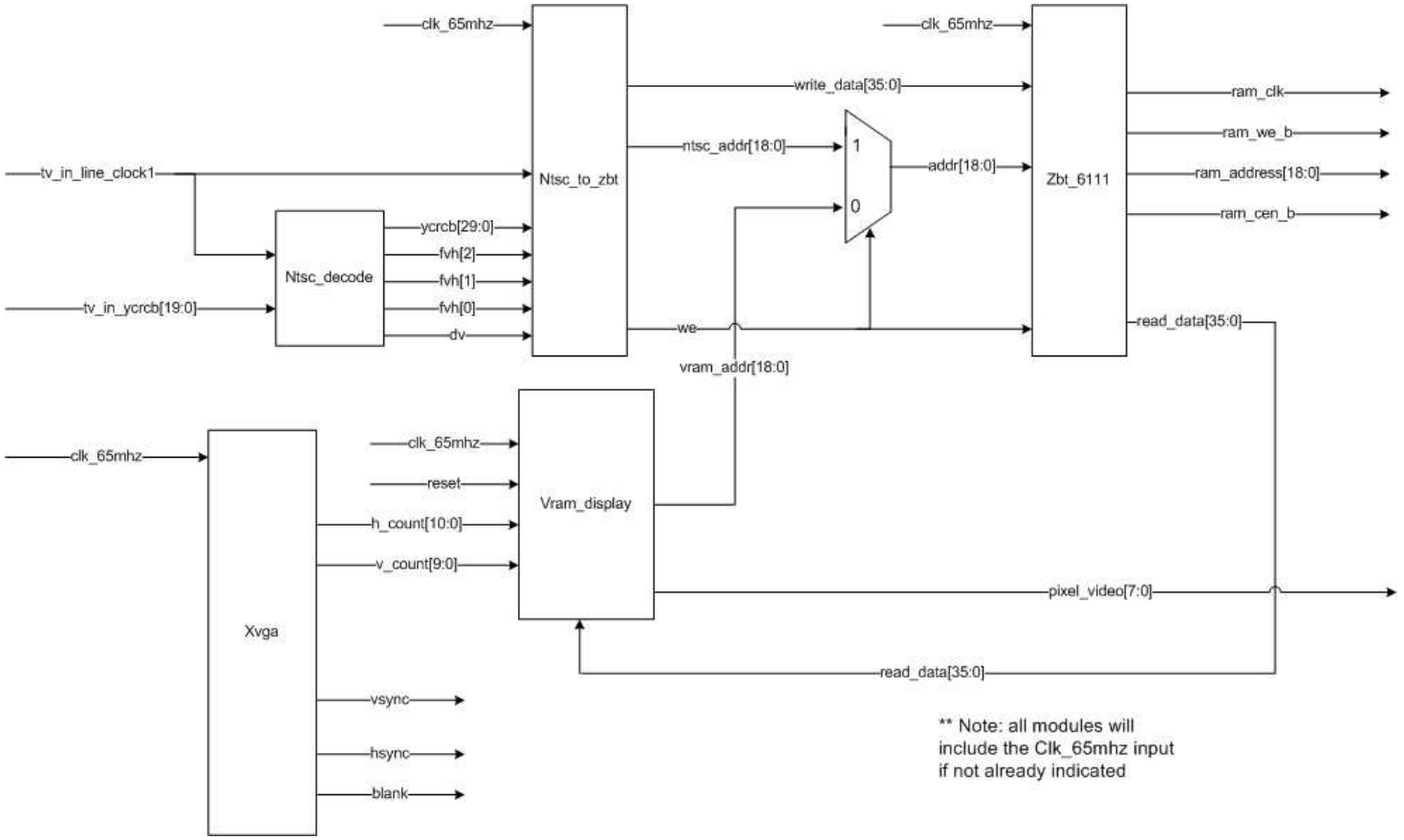


27Mhz to 65Mhz Clock Conversion

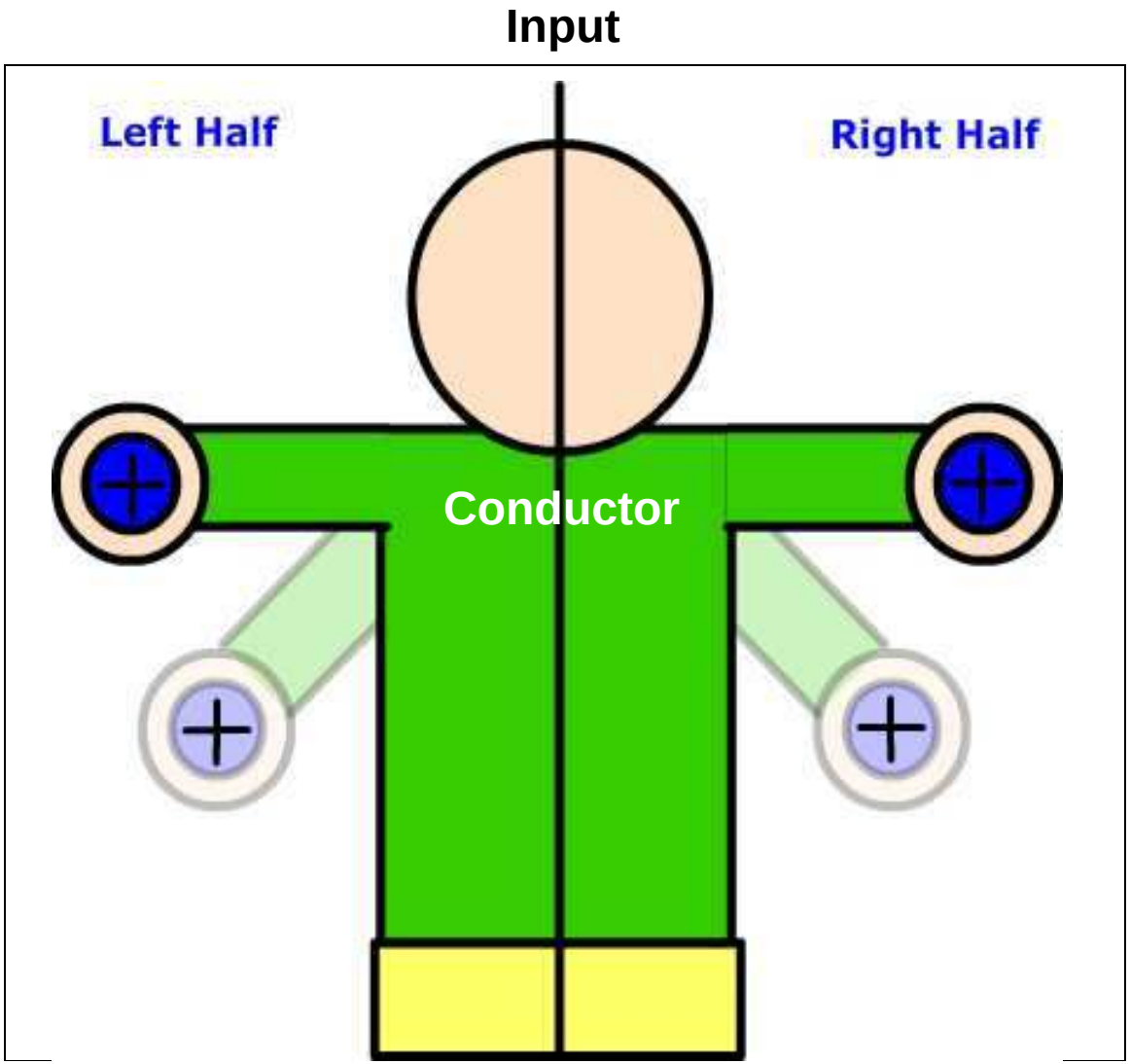


Camera Input Storage & Retrieval

- Each pixel stored as 8-bits: 5 bits for Y, 3 bits for Cb
- Store four pixels per location in ZBT
- ZBT Memory usage: (729 wide x 487 tall)/4 = 88755 locations per frame
- 65 Mhz clock

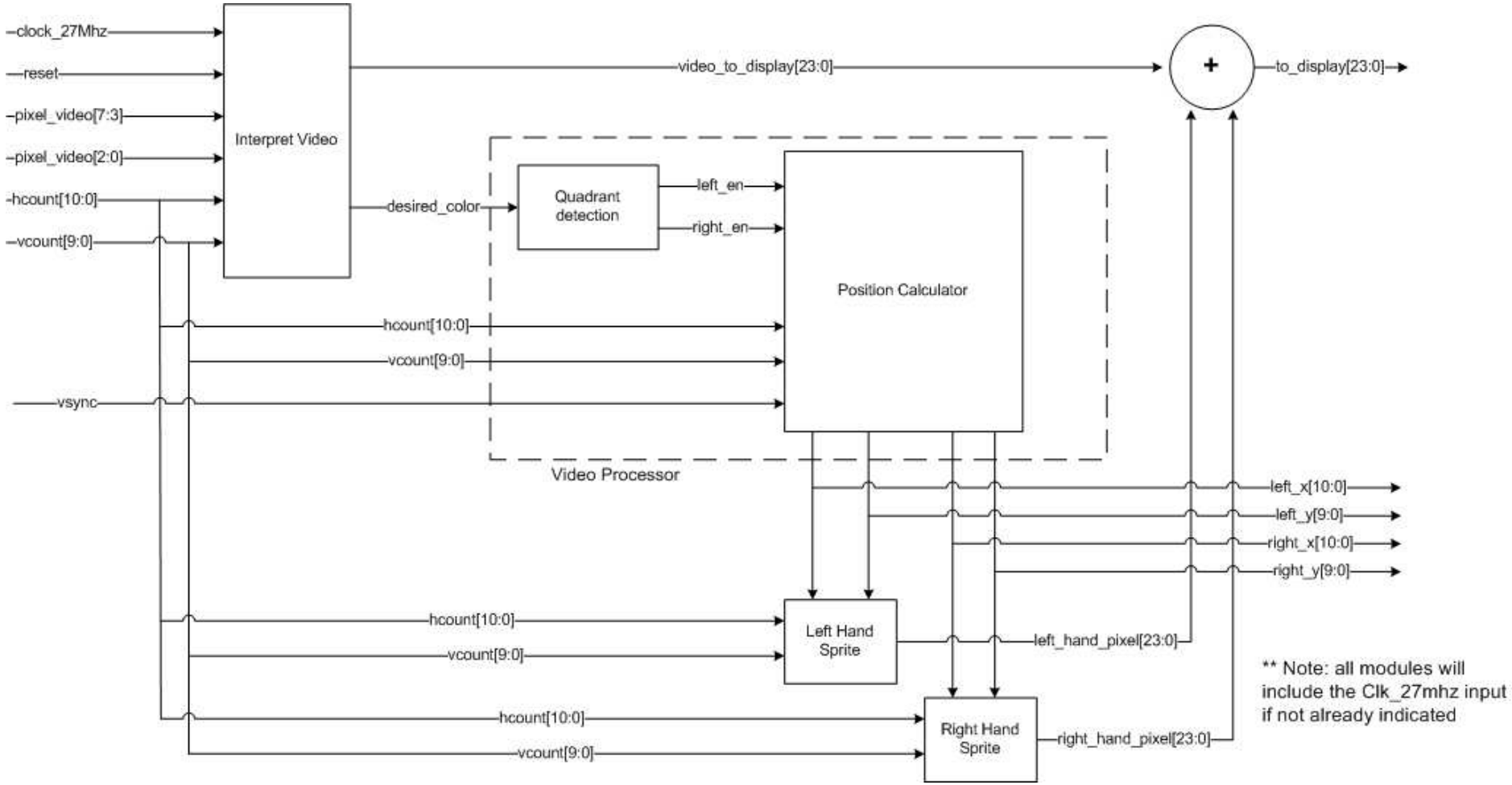


Camera Image



Visualization and Video Processing

- Calculates position of the user's hands: left in the left half plane, and right in the right half
- Displays hand positions

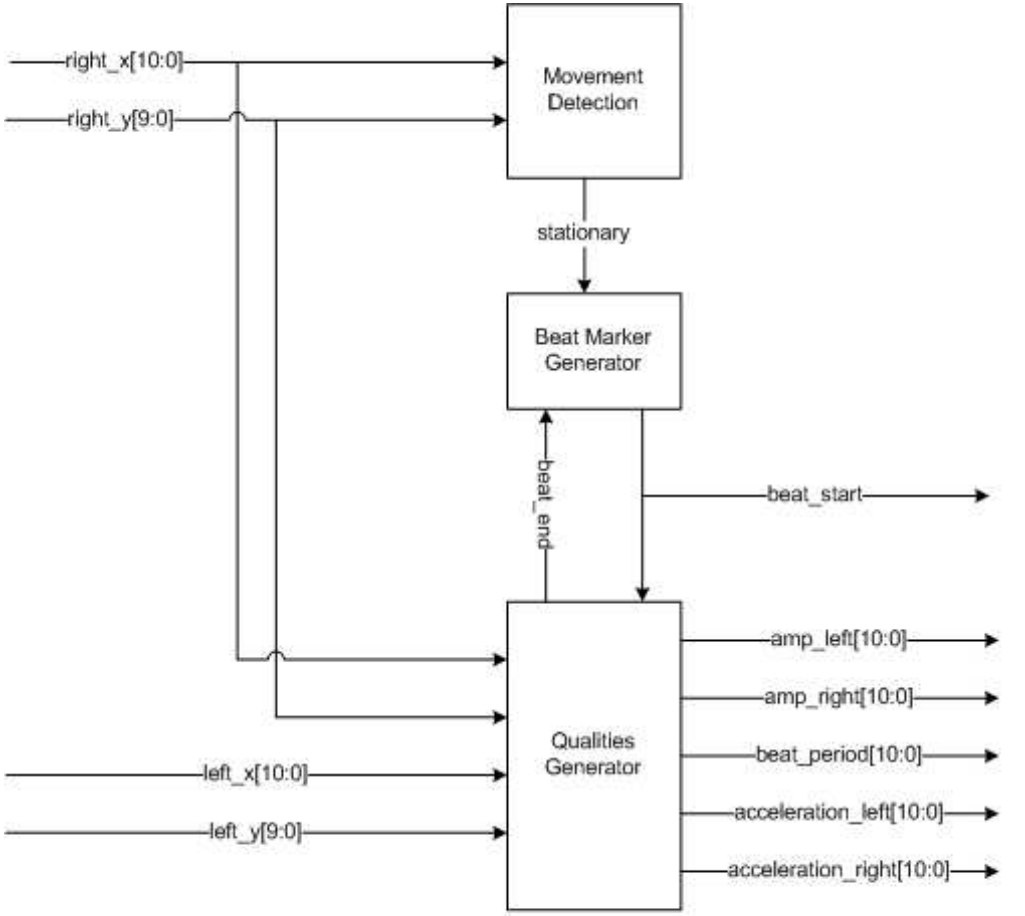


Details

- **Video Interpretation:**
  - For noise reduction: Requires at least three successive pixels to be blue before registering a pixel as part of the hand
- **Position Calculator:**
  - Calculates a running sum of x and y positions for the blue pixels in each half of the screen.
  - Uses Xilinx Pipelined Divider v3.0 to divide this sum by the count of pixels of the desired color to get the average coordinates of the hand
- **Display Output: 3 components**
  - Displays blue pixels detected by the camera, leaving other colors as black
  - Sprite to follow left hand movement
  - Sprite to follow right hand movement

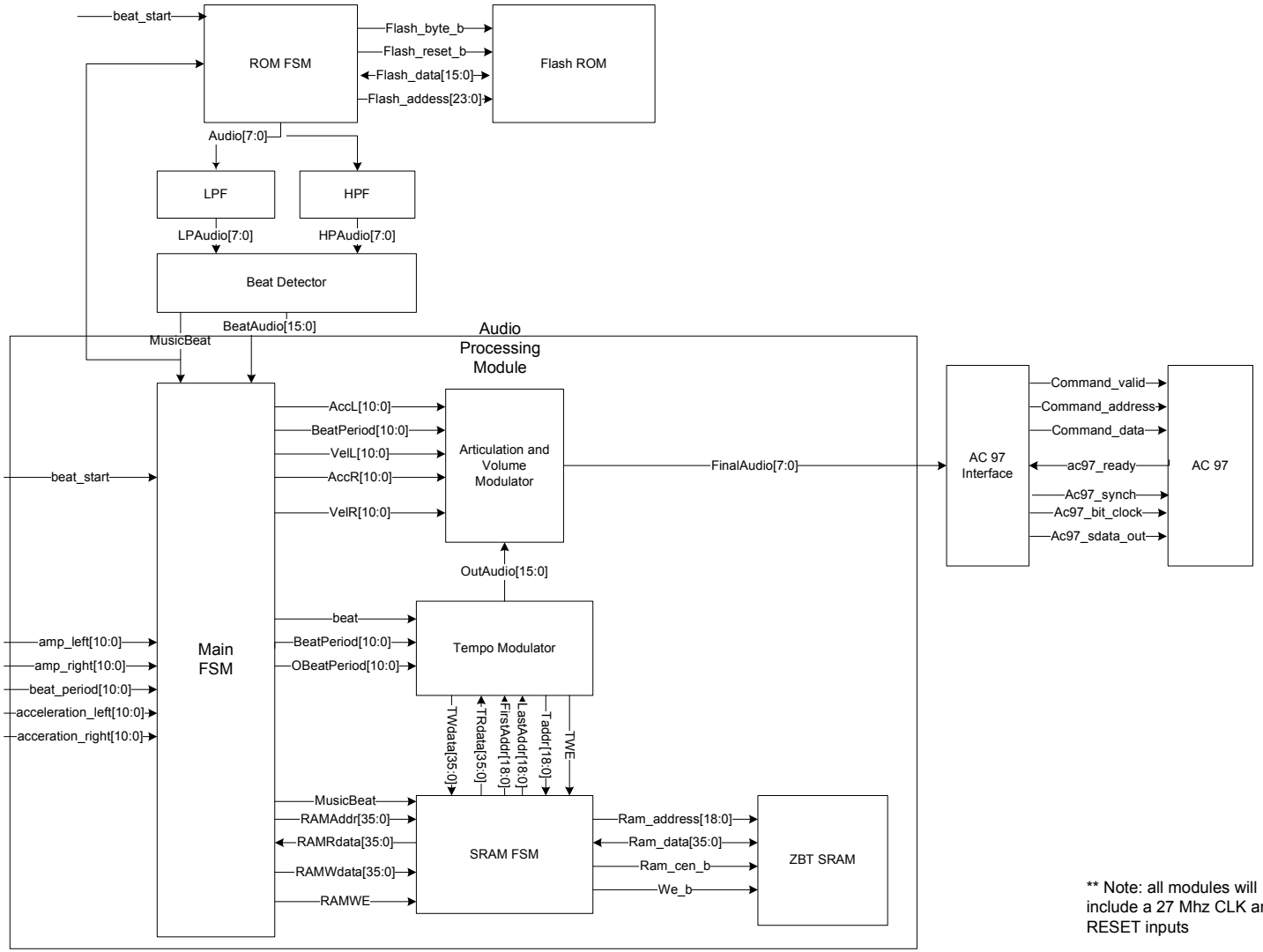
Motion Analyzer

- Determines the start and end of a beat
  - When coordinates stay within a certain distance for more than 10 frames, beat ends
  - After a beat ends, when movement starts again, a new beat starts
- Methods for calculating qualities:
  - **Amplitude:** difference in the x and y coordinates of successive beat starts
  - **Acceleration:** average second difference of the 10 frames following a beat start
  - **Beat period:** number of samples counted between two beat starts.



Audio Processing

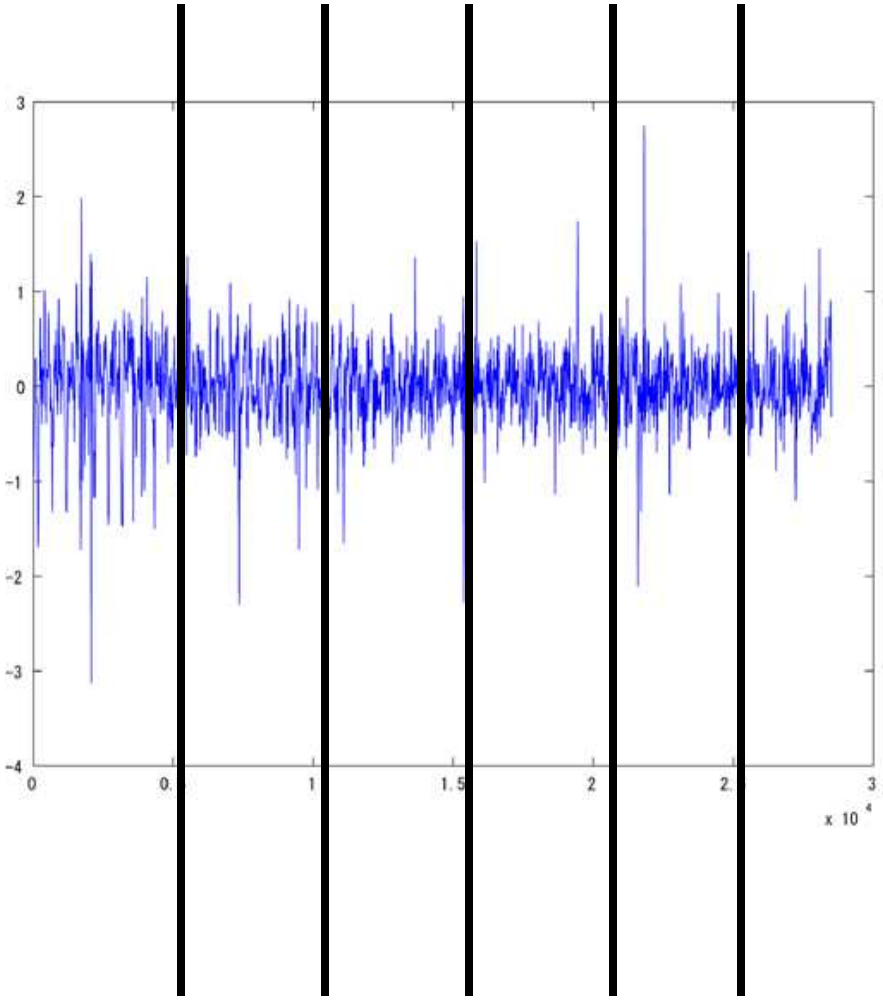
- **Beat-by-beat processing**
  - One beat stored in SRAM at a time
  - End/start of beat in RAM identified by Beat detector
  - Beat signal instructs system to move onto next beat.
- **Timings**
  - Audio read in from ROM every clock period (27 Mhz)
  - Final audio output at 48 Khz
- **LP/HP Filter**
  - 15 segment convolution
- **Beat detector**
  - LPAudio[7:0] exceeding a threshold amplitude signifies a beat
  - BeatAudio[15:0] contains LP signal in BeatAudio[15:8] and LP signal in BeatAudio[7:0]
- **Flash ROM**
  - Data fed in directly through RS232 interface on PC



** Note: all modules will include a 27 Mhz CLK and RESET inputs

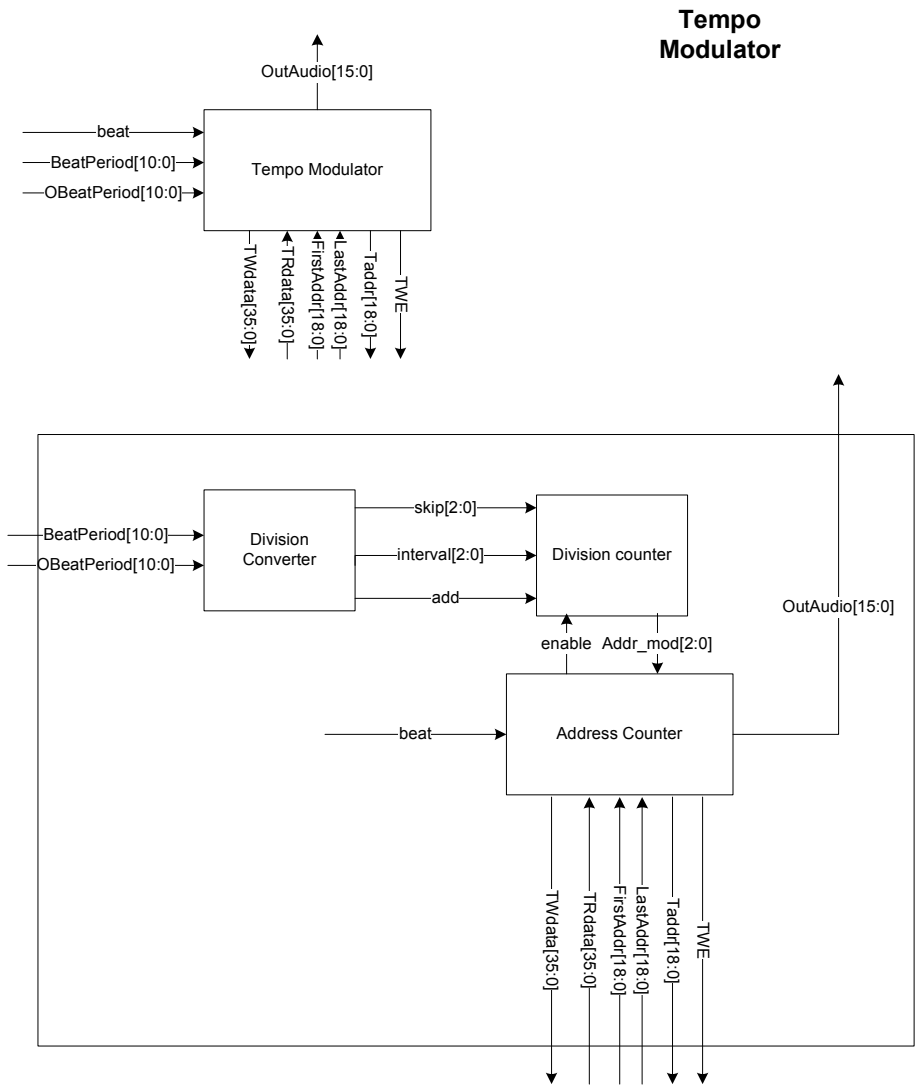
Tempo Modulator Theory

- Divides Audio signal into indivisible “divisions” whose time period is greater than that of the lowest audible sound.
- Scientifically, sounds less than 20hz are inaudible
- We will use 15 Hz divisions – 3200 samples (48 kHz)
- Divisions removed or added to change tempo
- Has been tested on Matlab

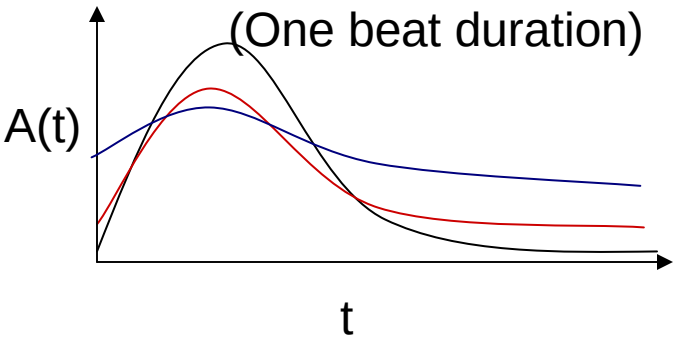


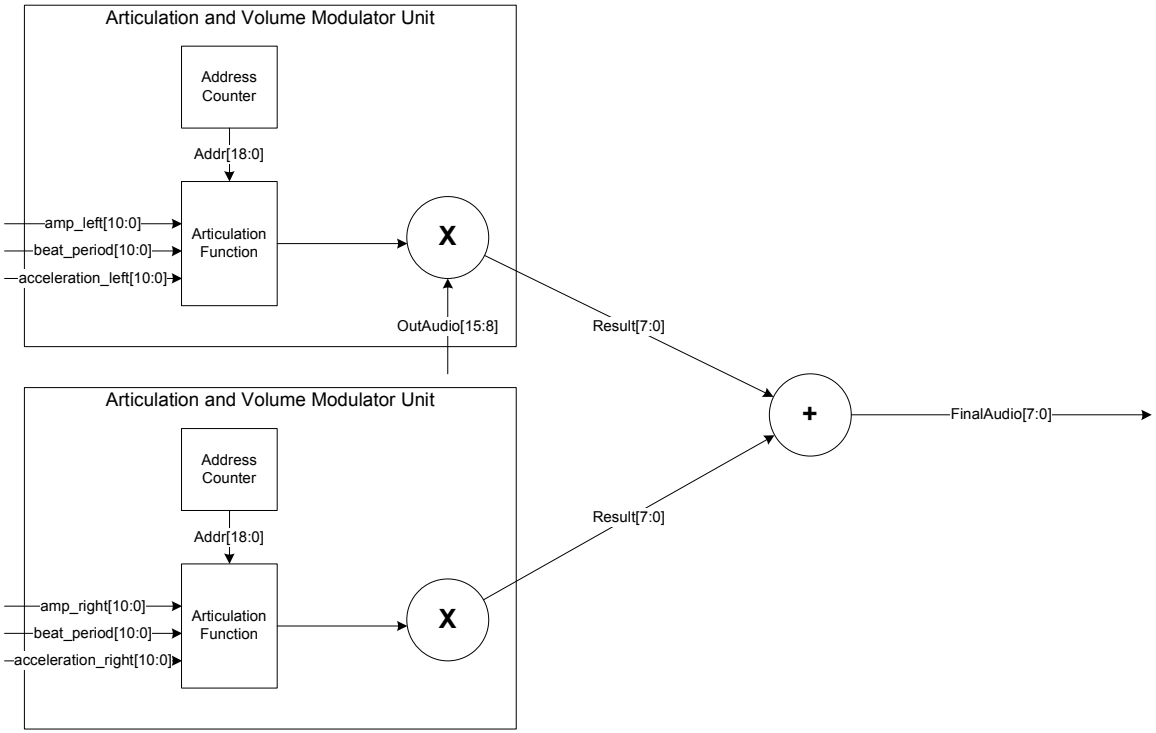
Tempo Modulator

- **Division Converter**
  - Rounding by truncation
  - Simplifies original signals into 3-bit representations
  - Interval[2:0] will be the truncated version of OBeatPeriod[10:0]
  - Skip[2:0] will be the positive difference between the two truncated versions of beat periods
  - Add <= (BeatPeriod > OBeatPeriod)
- **Division counter**
  - Counts the number of divisions.
  - Takes an enable signal from Address Counter which is triggered every 3200 address counted.
  - If division count = interval[2:0], skip[2:0] is added or subtracted from the accessing address via Addr_mod[2:0]
- **Special Cases**
  - Problem when approximated speed increase is inaccurate, or the beat period suddenly changes.
  - If beat arrives prematurely, rest of beat that needs to be played is cut off
  - If beat arrives late, repeat last few divisions until beat arrives.



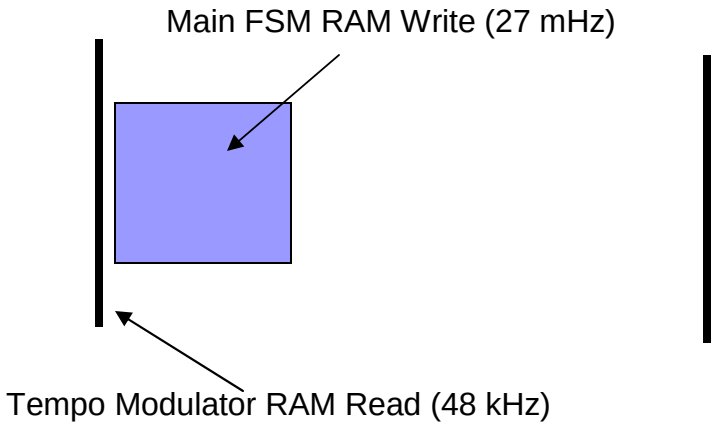
Articulation and Volume Modulator

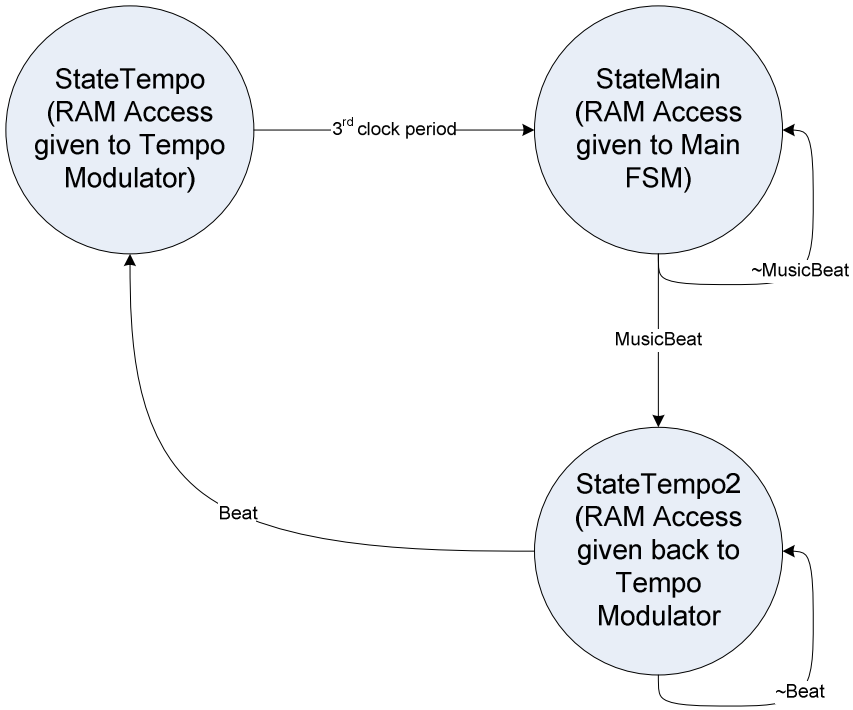


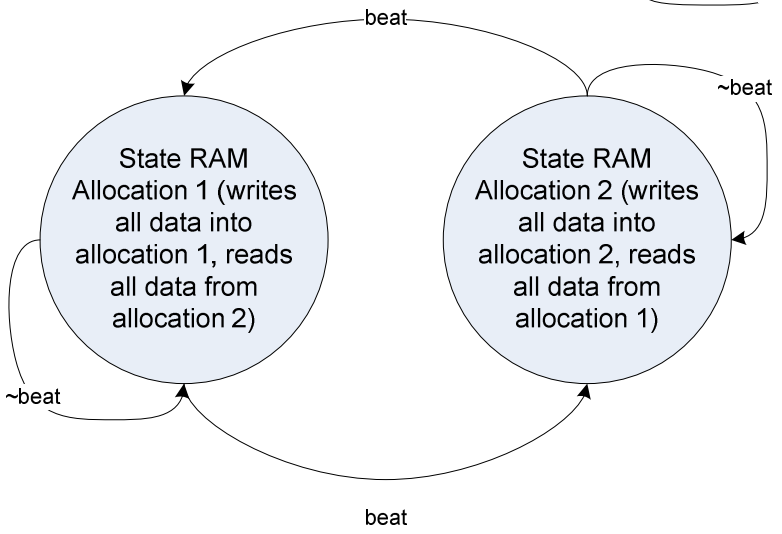


- **Mathematical multiplication in time domain of tempo-modified audio and "Articulation Function"**
- **Separate Articulation and Volume Modulating for LP and HP signals**
- **Articulation Function**
  - 3rd degree polynomial

RAM FSM







- **Access Control**
  - Tempo Modulator and Main FSM share access to ZBT RAM
  - Tempo Modulator at 48 kHz, Main FSM at 27 mHz
  - Data from Main FSM delayed 3 clock cycles
- **Allocation Alternator**
  - Insures data from previous beat is not overwritten by next beat
  - Addresses fed into SRAM FSM are “virtual addresses.” SRAM FSM adjusts these addresses to correspond to actual addresses.