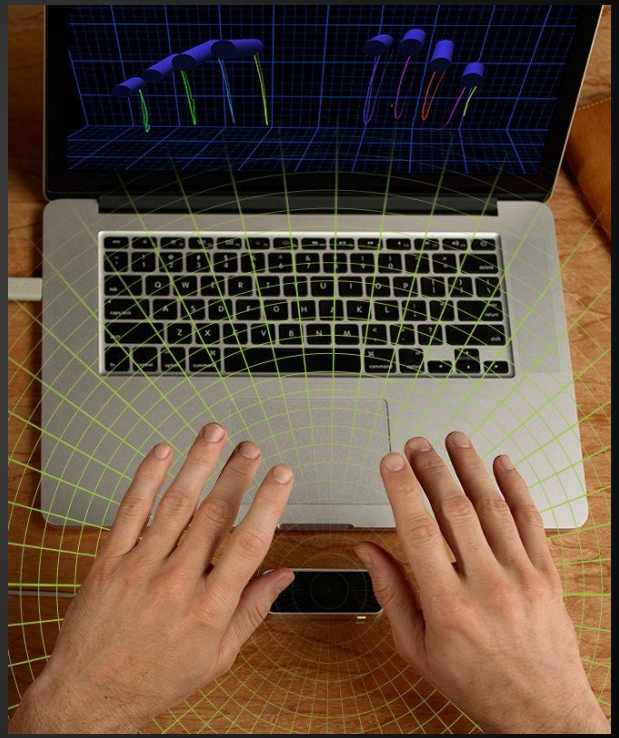
Leap Motion Piano

Patrice Liang
Matthew Patey
Vanshil Shah
Kevin Walters

Overview - Leap Motion

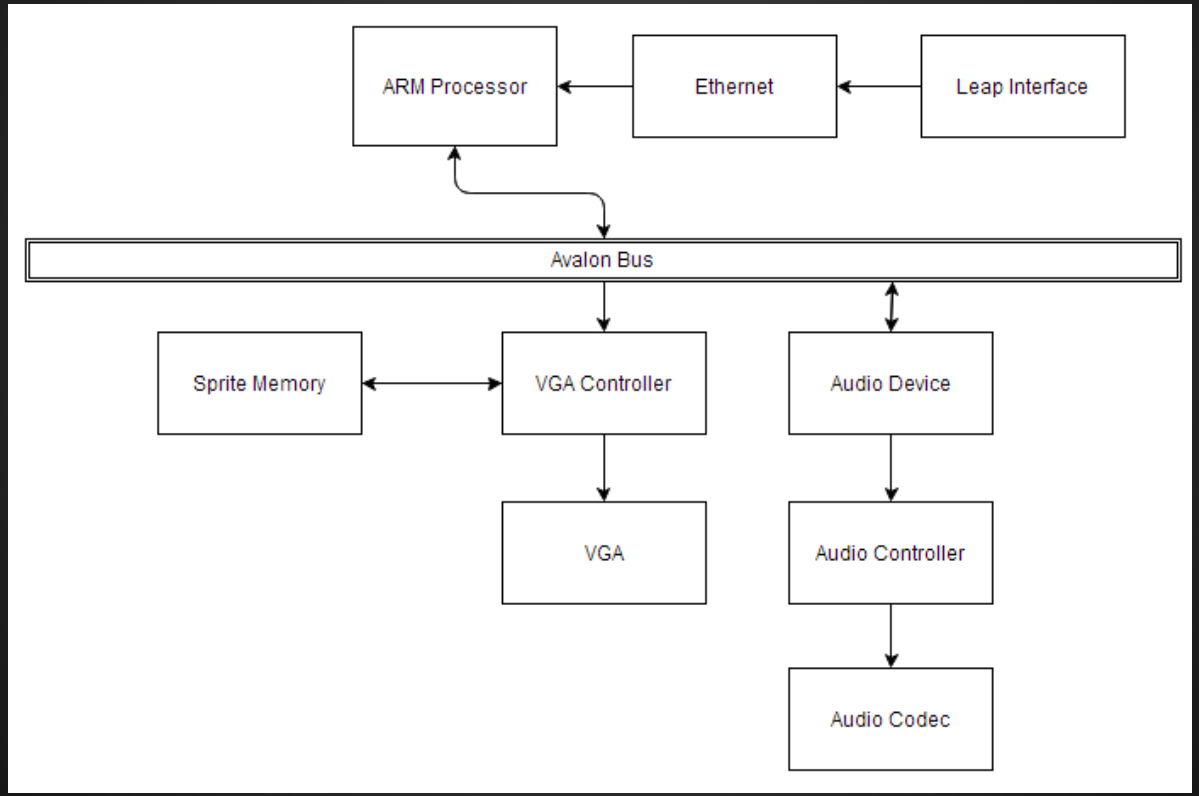
- 2 cameras, 3 infrared LEDs
- 8 ft^3 interactive space
- Leap Motion controller

software



Retrieved from *www.leapmotion.com*

Overview - High-Level Block Diagram



Architecture

- Hardware
  - custom VGA
  - cursor memory
  - audio
- Software
  - communication with Leap
  - drivers for the hardware peripherals
  - userspace programs

Hardware: VGA

- Connected to the Avalon Bus as a slave
- VGA monitor runs on 25MHz clock, created from the on-board 50Mhz clock
- Responsible for painting the cursor (retrieved from custom-built cursor memory) and piano (hardcoded)

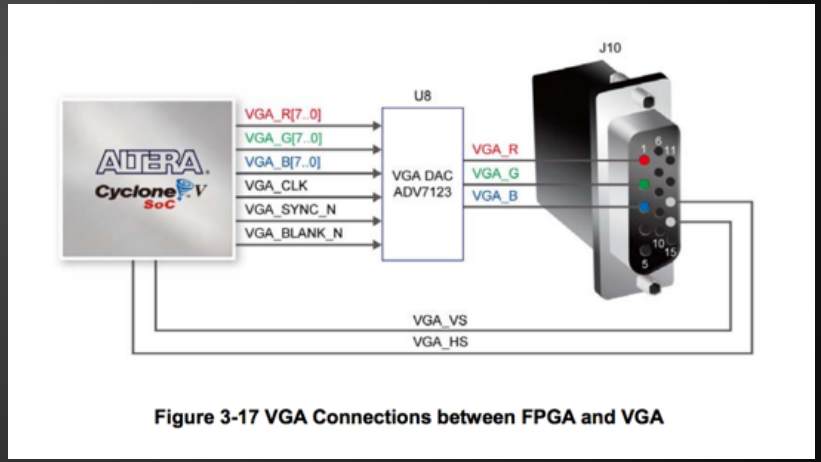


Figure 3-17 VGA Connections between FPGA and VGA

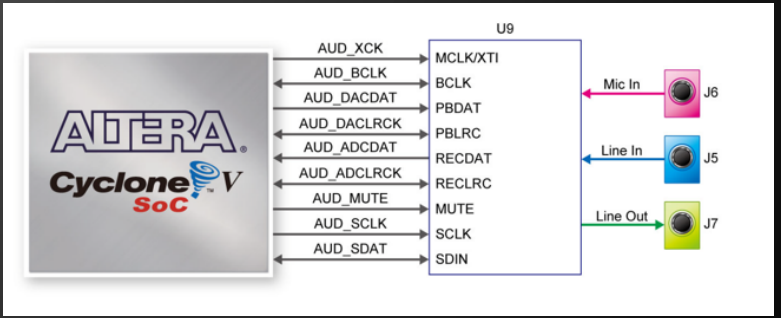
http://www.rocketboards.org/pub/Documentation/ArrowSoCKitEvaluationBoard/SoCKit_User_manual.pdf

Hardware: Audio

- I2C bus controller and configuration (FPGA master, Audio codec slave)
  - I2C_SCLK set to ~390kHz
- I2C data sent in through I2C_SDAT line; sends Start signal, addresses 0x34 for the SSM2603 codec, then configures data
- I2C configuration done in hardware, 44.1kHz sample rate, 16 bit samples

Hardware: Audio

- Audio codec input clock: 11.2896MHz; cannot be derived from main 50MHz clock, thus created a precision clock generator
- Audio codec controller responsible for sending data from the HPS to the codec
- HPS connected to the main audio hardware, which has two 2048 byte buffers



Software

- Leap cannot run on ARM instruction set so had to use external, x86 architecture and send over data through UDP
- Receive this data on the HPS and send it to VGA controller as the cursor position
- Send information about key presses to hardware
- Send audio data over the Avalon bus by sending 2048 byte chunks of the pre-downloaded audio files
- Audio files converted to raw amplitude data using “sox”

Challenges - VGA Cursor

- Cursor image stored in small memory
- When raster scan is within bounds of cursor, read correct pixel from memory and paint it
- Requires two cycles
- Though 50 MHZ clock is double 25 MHZ VGA clock, each point only has one board cycle before VGA clock rises

VGA - continued

- Problem: painting is behind reading
- Paints column at left side
  - for each scan, first cursor cycle sees pixel at 0,0 the address sent to memory during rest of scan
- Solution: start painting one cycle after start requesting pixels

Challenges - Control Audio Buffers

- Two buffers, alternate between writing to and playing from
- Writing happens on Avalon/HPS clock, playing happens on audio clock
- Use flags in registers to control when each buffer is accessed
- Can’t set flags in both sections of hardware

Challenges - Audio Software

- Filling audio buffer presents time constraints for processor
- Processor clock is a lot faster than audio clock, but difficult to guarantee timing on processor
  - context switches, kernel traps, IO latency
  - (somewhat) out of programs control, cause significant delays

Challenges - Audio Software

- Maximize CPU time with separate thread
- Use thread-safe queue for communication between main and audio threads
- Minimize traps, send an entire frame in one driver call

Summary - Lessons Learned

- Hardware compilation is LONG
  - double-check all changes
  - be smart about it
- Have a backup plan
  - workspace unavailability and faulty boards
- Front-load as much as possible
- Expect the unexpected
- Divide and conquer

Summary - Future Implementations

Multiple Fingers

- Sending the data
- Displaying multiple fingers
  - duplicate logic for each finger
- Drawing keypresses (software)
- Playing audio from multiple inputs
  - simultaneous playing
  - note cancellation on a per-finger basis

Future Implementations (cont.)

- Incorporate interrupts instead of polling
  - interrupt when available to send data
- Condition variables instead of popping queue
  - prevent unnecessary looping; wake on a queue push
  - better multi-threaded practice
- Continuous key playing
  - prolong the last part of the data sent

Thank you!