The Project & Digital Video

EECS150 Fall2008 - Lab Lecture #7

Arjun Singh
Adopted from slides designed by
Greg Gibeling and Chris Fletcher

Today

- Project Introduction
- Good Design (Part 2)
  - Interfaces and Handshaking
- Video Encoder
  - Digital Video
  - ITU-R BT.601/ITU-R BT.656
  - Video Encoder
  - I2C Bus
  - More Information

The Project (1)

- Digital Storage Oscilloscope
  - Display audio as waveforms
  - Stream audio from network audio or from a microphone
    - Store audio stream
    - Playback audio stream
    - Trigger and freeze under different conditions
- Extra Credit
  - A major part of this project
  - Will augment checkpoints 3, 4 and 5
The Project (2)

- Checkpoints
  - Require more design work than labs
    - We’re not telling you exactly what to do
  - Part of your project
    - Design them well
    - Test them thoroughly!
    - Don’t lose your code
  - Require more time

The Project (3)

- Checkpoint Roadmap
  - Video Encoder
  - SDRAM in Simulation
  - SDRAM in Hardware + SDRAM Arbiter
  - Waveform Generator + OScope features
  - AC97 Audio
  - Extra Credit
  - Check calendar page and project spec for dates

Interfaces & Handshaking (1)

- Connect two modules with just wires
- No combinational logic
- Unidirectional Data-flow
  - Source → Sink
Interfaces & Handshaking (2)
- Handshaking Signals
  - Valid (from Source)
  - Ready (from Sink)
- Don’t rely on timing assumptions

Interfaces & Handshaking (3)
- Data Transfer
  - Synchronous
  - When Ready and Valid are both high

Checkpoint #1: Video Encoder
- Video Encoder
  - Sets up NTSC framing
    - Blanking, SAV, EAV
  - Request Data & Display it
Digital Video (1)

- **Pixel Array**
  - A digital image is represented by a matrix of pixels which include color information.

- **Frames**
  - Motion is created by flashing a series of still frames

Digital Video (2)

- **Scanning**
  - Images are generated on the screen by scanning pixel lines, left to right, top to bottom
  - Early CRTs required time to get from the end of a line to the beginning of the next. Therefore each line of video consists of active video portion and a horizontal blanking interval
  - To reduce flicker, each frame is divided into two fields: odd and even

Digital Video (3)

- **Colors**
  - Usually represented as red, green and blue
  - In the digital domain we could transmit 8 bits each for RGB.
  - Transition from B&W
    - Didn’t want to break old TVs
    - Added separate color or “Chroma” signals
      - Y: Luma (Black and White)
      - Cr: Chroma Red (New color signal)
      - Cb: Chroma Blue (New color signal)
Digital Video (4)

- Chroma Subsampling
  - Human eye is sensitive to Luma more than Chroma

Administrative Info (1)

- Project Partners
  - Talk to us ASAP if you don't have one
- SVN Repositories
  - Chris will give introduction next Tuesday, 3:30-5:00pm (his OH time)
  - Introduction will be audio-cast
    - (Audio-cast guaranteed this time)

Administrative Info (2)

- Design Reviews
  - Grading
    - You have it or you don't
  - Bring diagrams
    - Schematic
      - "On a napkin"
    - Bubble-and-arc
    - Block Diagrams
  - NO VERILOG
NO DESIGN
→ NO HELP

ITU-R BT.601

- Formerly, CCIR-601.
  - Designed for digitizing broadcast NTSC.
    - National Television System Committee.
- Variations:
  - 4:2:0 Chroma Subsampling
  - PAL (European) version
- Component streaming:
  - line i: C_{Y} Y C_{Y} Y C_{Y} Y
  - line i+1: C_{C_{R}} Y C_{C_{B}} Y C_{C_{B}} Y
- Effective Bits/Pixel:
  - 4 components / 2 pixels = 32/2 = 16 bits/pixel

ITU-R BT.656 (1)

- Details
  - Pixels/Line: 858
  - Lines/Frame: 525
  - Frames/S: 29.97
  - Pixels/S: 13.5M
- Active
  - Pixels/Line: 720
  - Lines/Frame: 487
- Blanking
  - SAV/EAV: 48/48
  - Black filter

FIGURE 1
Composition of interface data stream

Active Frame

Size

720 x 507

Frame Rate

29.97/sec

Scan

Interlaced

Chroma subsampling

4:2:2

2:1 in X only

Coincident

Bits per component

8

Effective bits/pixel

16
ITU-R BT.656 (2)

- Odd Field (262 Lines)
  - Total: 262 Lines
  - 16 Vertical Blanking
  - 244 Active
  - 2 Vertical Blanking
- Even Field
  - Total: 263 Lines
  - 17 Vertical Blanking
  - 243 Active
  - 3 Vertical Blanking

ITU-R BT.656 (3)

<table>
<thead>
<tr>
<th>P9</th>
<th>P8</th>
<th>P7</th>
<th>P6</th>
<th>P5</th>
<th>P4</th>
<th>P3</th>
<th>P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
<td>1'b1</td>
</tr>
<tr>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
</tr>
<tr>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
<td>1'b0</td>
</tr>
</tbody>
</table>

- F: Field Select (0: Odd, 1: Even)
- V: Vertical Blanking Flag
- H: EAV/SAV Flag (0: SAV, 1: EAV)

Video Encoder (1)

- Analog Devices ADV7194
  - Supports ITU-R BT.601/656
  - S-Video and Composite Outputs
  - I²C Control (We will give this to you)
### Video Encoder (2)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Dir</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE_P</td>
<td>10</td>
<td>O</td>
<td>Outgoing NTSC Video (Use {Data, 2'b00})</td>
</tr>
<tr>
<td>VE_SCLK</td>
<td>1</td>
<td>O</td>
<td>I2C Clock (For Initialization)</td>
</tr>
<tr>
<td>VE_SDA</td>
<td>1</td>
<td>O</td>
<td>I2C Data (For Initialization)</td>
</tr>
<tr>
<td>VE_PAL_NTSC</td>
<td>1</td>
<td>O</td>
<td>PAL/NTSC Mode Select (Always 1'b0)</td>
</tr>
<tr>
<td>VE_RESET_B_</td>
<td>1</td>
<td>O</td>
<td>Active low reset (~Reset)</td>
</tr>
<tr>
<td>VE_HSYNC_B_</td>
<td>1</td>
<td>O</td>
<td>Manual Control (Always 1'b1)</td>
</tr>
<tr>
<td>VE_VSYNC_B_</td>
<td>1</td>
<td>O</td>
<td>Manual Control (Always 1'b1)</td>
</tr>
<tr>
<td>VE_BLANK_B_</td>
<td>1</td>
<td>O</td>
<td>Manual Control (Always 1'b1)</td>
</tr>
<tr>
<td>VE_SCRESET</td>
<td>1</td>
<td>O</td>
<td>Manual Control (Always 1'b0)</td>
</tr>
<tr>
<td>VE_CLKIN</td>
<td>1</td>
<td>O</td>
<td>Clock (27MHz, Just send Clock)</td>
</tr>
</tbody>
</table>

### Video Encoder (3)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Dir</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td>1</td>
<td>I</td>
<td>Clock input (27MHz)</td>
</tr>
<tr>
<td>Reset</td>
<td>1</td>
<td>I</td>
<td>Reset input</td>
</tr>
<tr>
<td>Data</td>
<td>32</td>
<td>I</td>
<td>Requested Data from ROM</td>
</tr>
<tr>
<td>Captured</td>
<td>1</td>
<td>I</td>
<td>Data is valid this cycle</td>
</tr>
<tr>
<td>DataReady</td>
<td>1</td>
<td>O</td>
<td>The Video Encoder is ready to receive more data. &amp; DataReady and DataValid are both high, the VideoEncoder should latch in data on the next rising edge.</td>
</tr>
<tr>
<td>AddressLine</td>
<td>9</td>
<td>O</td>
<td>Line of Video (Line[7:0], Field) The ROM will return a pixel pair from this line.</td>
</tr>
<tr>
<td>AddressPair</td>
<td>9</td>
<td>O</td>
<td>Pair of Pixels. The line will return data for this pixel pair.</td>
</tr>
<tr>
<td>AddressValid</td>
<td>1</td>
<td>O</td>
<td>AddressLine and AddressPair are valid this cycle.</td>
</tr>
<tr>
<td>AddressReady</td>
<td>2</td>
<td>O</td>
<td>The sink connected to AddressLine/AddressPair is ready to receive those signals.</td>
</tr>
</tbody>
</table>

### Video Encoder (4)

General Video Encoder Block Diagram

- Address Counter
- H FSM
- V FSM
- PC Control
- Blank Gen
- Data Clip
- Test ROM
- Video Encoder
Video Encoder (5)

- Basic Design
  - Stream EAV, Blank, SAV, Active Lines
    - Generate EAV/SAV/Blank using a mux
    - Register output data (Timing reasons)
  - Request Incoming Data
    - Request it the cycle before you need it
    - Must be clipped
      - Minimum data is 0x10
      - Maximum data is 0xF0
      - Otherwise it will appear to be blanking signals

Video Encoder (6)

- Testing
  - Test thoroughly
    - Simulation is difficult with test ROM
    - Try using values which count, so you can see it
  - Design your testbench early
    - Perhaps one partner should design the module, one should design the testbench
    - Ensure that you test corner cases
      - First and last lines
      - Off-by-one errors in counters

I2C

- ADV7194 Initialization using I2C
  - Requires only 2 wires
    - Serial Data (Bidirectional)
    - Clock (Driven by master)
  - Runs at up to 400kHz
  - Bidirectional Communication
- Given to you
  - Complicated to get right
  - Hard to debug
**I²C (2)**

- **Physical Protocol**
  - **Data**
    - Open collector bidirectional bus
    - Driven by sender
  - **Clock**
    - Open collector unidirectional bus
    - Driven by master
    - May be pulled low to stall transmission

**Bidirectional Open Collector Bus**

10/10/2008

**I²C (3)**

- **Protocol**
  - Start Condition
  - Address
  - Address Acknowledge
  - Data Transfer
  - Data Acknowledge
  - Stop Condition

**I²C (4)**

- **Arbitration**
  - Anyone can drive bus at any time
    - No central arbiter
    - No short circuits (Impossible in open collector)
  - Decentralized Arbitration
    - Check data bus against value you’re sending
    - Mismatch means someone else is transmitting
      - So let them finish, and then try again
    - Inherently gives preferences to accesses with more 1'b1s in them

10/10/2008
More Information

- Checkpoint Writeup
- Documents Page of the Website
  - Video in a Nutshell
  - ADV7194 Datasheet
    - Complete ADV7194 reference
  - ITU-R BT.656 & ITU-R BT.601 Standards
    - Complete video standards
  - I²C Bus Specification

- READ THE DATASHEETS!