Display Technology

- Images stolen from various locations on the web...

Cathode Ray Tube
Cathode Ray Tube

Raster Scanning
Electron Gun

Beam Steering Coils
Color

Shadow Mask and Aperture Grille
Liquid Crystal Displays
LCoS

- Liquid Crystal on Silicon
  - Put a liquid crystal between a reflective layer on a silicon chip
Grating Light Valve (GLS)

- lots (8000 currently) of micro ribbons that can bend slightly
  - Make them reflective
  - The bends make a diffraction grating that controls how much light where
  - Scan it with a laser for high light output
- 4000 pixel wide frame ever 60Hz
Digistar 3 Dome Projector

VGA

- Stands for Video Graphics Array
- A standard defined by IBM back in 1987
  - 640 x 480 pixels
  - Now superseded by much higher resolution standards...
- Also means a specific analog connector
  - 15-pin D-subminiature VGA connector
### VGA Connector

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Red out</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>Green out</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Blue out</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Unused</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>Ground</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>Monitor ID 0 in</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Monitor ID 1 in or data from display</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Horizontal Sync</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Vertical Sync</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Monitor ID 3 in or data clock</td>
<td></td>
</tr>
</tbody>
</table>

### Raster Scanning

![Raster Scanning Diagram]
Raster Scanning

VGA Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Dots</td>
<td>640</td>
</tr>
<tr>
<td>Vertical Scan Lines</td>
<td>480</td>
</tr>
<tr>
<td>Horiz. Sync Polarity</td>
<td>NEG</td>
</tr>
<tr>
<td>A (µs)</td>
<td>31.77 Scanline time</td>
</tr>
<tr>
<td>B (µs)</td>
<td>3.77 Sync pulse length</td>
</tr>
<tr>
<td>C (µs)</td>
<td>1.89 Back porch</td>
</tr>
<tr>
<td>D (µs)</td>
<td>25.17 Active video time</td>
</tr>
<tr>
<td>E (µs)</td>
<td>0.94 Front porch</td>
</tr>
</tbody>
</table>

60Hz vertical frequency

Diagram showing the timing and scanning process with labels for different time intervals and durations.
VGA Timing

Horiz. Sync Polarity: NEG

A (µs) 31.77 Scanline time
B (µs) 3.77 Sync pulse length
C (µs) 1.89 Back porch
D (µs) 25.17 Active video time
E (µs) 0.94 Front porch

25.17/640 = 39.33ns/pixel = 25.4MHz pixel clock

VGA Timing

Horiz. Dots: 640
Vert. Scan Lines: 480
Vert. Sync Polarity: NEG

O (ms): 16.68 Total frame time
P (ms): 0.06 Sync pulse length
Q (ms): 1.02 Back porch
R (ms): 15.25 Active video time
S (ms): 0.35 Front porch
VGA Timing Summary

### 60 Hz refresh and 25MHz pixel clock

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Vertical Sync</th>
<th>Horizontal Sync</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time</td>
<td>Clocks</td>
</tr>
<tr>
<td>$T_{s}$</td>
<td>Sync pulse time</td>
<td>16.7 ms</td>
<td>416,800</td>
</tr>
<tr>
<td>$T_{disp}$</td>
<td>Display time</td>
<td>15.36 ms</td>
<td>384,000</td>
</tr>
<tr>
<td>$T_{pw}$</td>
<td>Pulse width</td>
<td>64 μs</td>
<td>1,600</td>
</tr>
<tr>
<td>$T_{pp}$</td>
<td>Front porch</td>
<td>320 μs</td>
<td>8,000</td>
</tr>
<tr>
<td>$T_{bp}$</td>
<td>Back porch</td>
<td>928 μs</td>
<td>25,200</td>
</tr>
</tbody>
</table>

60 Hz refresh and 25MHz pixel clock

### Relaxed VGA Timing

- This all sounds pretty strict and exact...
- It’s not really... The only things a VGA monitor really cares about are:
  - Hsync
  - Vsync
  - Actually, all it cares about is the falling edge of those pulses!
  - The beam will retrace whenever you tell it to
  - It’s up to you to make sure that the video signal is 0v when you are not painting (i.e. retraction)
## Relaxed VGA Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Dots</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Vertical Scan Lines</td>
<td>?</td>
<td>60Hz vertical frequency</td>
</tr>
<tr>
<td>Horiz. Sync Polarity</td>
<td>NEG</td>
<td></td>
</tr>
<tr>
<td>A (µs)</td>
<td>30.0</td>
<td>Scanline time</td>
</tr>
<tr>
<td>B (µs)</td>
<td>2.0</td>
<td>Sync pulse length</td>
</tr>
<tr>
<td>C (µs)</td>
<td>10.7</td>
<td>Back porch</td>
</tr>
<tr>
<td>D (µs)</td>
<td>12.8</td>
<td>Active video time</td>
</tr>
<tr>
<td>E (µs)</td>
<td>4.50</td>
<td>Front porch</td>
</tr>
</tbody>
</table>

12.8/128 = 100ns/pixel = 10 MHz pixel clock

```
| VIDEO | | VIDEO (next line) |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
|        | |                   |
----------|---|-------------------|
```

## VGA Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Dots</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>Vertical Scan Lines</td>
<td>255</td>
<td></td>
</tr>
<tr>
<td>Vert. Sync Polarity</td>
<td>NEG</td>
<td></td>
</tr>
<tr>
<td>Vertical Frequency</td>
<td>60Hz</td>
<td></td>
</tr>
<tr>
<td>O (ms)</td>
<td>16.68</td>
<td>Total frame time</td>
</tr>
<tr>
<td>P (ms)</td>
<td>0.09</td>
<td>Sync pulse length (3x30µs)</td>
</tr>
<tr>
<td>Q (ms)</td>
<td>4.86</td>
<td>Back porch</td>
</tr>
<tr>
<td>R (ms)</td>
<td>7.65</td>
<td>Active video time</td>
</tr>
<tr>
<td>S (ms)</td>
<td>4.08</td>
<td>Front porch</td>
</tr>
</tbody>
</table>

```
| VIDEO | | VIDEO (next frame) |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
|        | |                    |
----------|---|--------------------|
```
VGA Voltage Levels

- Voltages on R, G, and B determine the color
  - Analog range from 0v (off) to +0.7v (on)
  - But, our pads produce 0-3.3v outputs!

For B&W output, just drive RGB together and let 0v=black and 3.3v=white
  - overdrives the input amps, but won’t really hurt anything

For color you can drive R, G, B separately
  - Of course, this is only 8 colors (including black and white)
  - Requires storing three bits at each pixel location
VGA on Spartan3e Starter

Series resistors limit output voltage to 0-0.7v

Table 6-1: 3-Bit Display Color Codes

<table>
<thead>
<tr>
<th>VGA_RED</th>
<th>VGA_GREEN</th>
<th>VGA_BLUE</th>
<th>Resulting Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Black</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Blue</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Green</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Cyan</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Magenta</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Yellow</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>White</td>
</tr>
</tbody>
</table>
Raster Scanning

![Diagram of Raster Scanning]

VGA on Spartan3e Starter

### Table 6-2: 640x480 Mode VGA Timing

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Vertical Sync</th>
<th>Horizontal Sync</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time</td>
<td>Clocks</td>
</tr>
<tr>
<td>$T_g$</td>
<td>Sync pulse time</td>
<td>16.7 µs</td>
<td>416,800</td>
</tr>
<tr>
<td>$T_{disp}$</td>
<td>Display time</td>
<td>13.36 µs</td>
<td>394,000</td>
</tr>
<tr>
<td>$T_{pw}$</td>
<td>Pulse width</td>
<td>64 µs</td>
<td>1,600</td>
</tr>
<tr>
<td>$T_{fp}$</td>
<td>Front porch</td>
<td>320 µs</td>
<td>8,000</td>
</tr>
<tr>
<td>$T_{bp}$</td>
<td>Back porch</td>
<td>925 µs</td>
<td>25,280</td>
</tr>
</tbody>
</table>

#### Figure 6-3: VGA Control Timing

The signal timings in Table 6-2 are derived for a 640-pixel by 480-row display using a 25 MHz pixel clock and 60 Hz ± 1 refresh. Figure 6-3 shows the relation between each of the timing symbols. The timing for the sync pulse width ($T_{pw}$) and front and back porch intervals ($T_{fp}$ and $T_{bp}$) are based on observations from various VGA displays. The front and back porch intervals are the pre- and post-sync pulse times. Information cannot be displayed during these times.
Figure 6-4 provides the UCF constraints for the VGA display port, including the I/O pin assignment, the I/O standard used, the output slew rate, and the output drive current.

```
SET "VGA_RED" LOC = "H14" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
SET "VGA_GREEN" LOC = "H15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
SET "VGA_BLUE" LOC = "G15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
SET "VGA_HSYNC" LOC = "F15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
SET "VGA_VSYNC" LOC = "F14" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
```

Figure 6-4: UCF Constraints for VGA Display Port

---

**VGA Assignment**

- **vgaControl**
  - Generate timing pulses at the right time
  - hSync, vSync, bright, hCount, vCount

- **bitGen**
  - Based on bright, hCount, vCount, turn on the bits
3 Types of bitGen

- Bitmapped
  - Frame buffer holds a separate rgb color for every pixel
  - bitGen just grabs the pixel based on hCount and vCount and splats it to the screen
  - Chews up a LOT of memory

- Character/Glyph-based
  - Break screen into nxm pixel chunks (e.g. 8x8)
  - For each chunk, point to one of k nxm glyphs
  - Those glyphs are stored in a separate memory
  - For 8x8 case (for example)
    - glyph number is hCount and vCount minus the low three bits
    - glyph bits are the low-order 3 bits in each of hCount and vCount
    - Figure out which screen chunk you’re in, then reference the bits from the glyph memory
3 Types of bitGen

- Direct Graphics
  - Look at hCount and vCount to see where you are on the screen
  - Depending on where you are, force the output to a particular color
  - Tedious for complex things, nice for large, static things

```verilog
parameter BLACK = 3'b 000, WHITE = 3'b111, RED = 3'b100;
// paint a white box on a red background
always@(*)
  if (~bright) rgb = BLACK; // force black if not bright
  // check to see if you're in the box
  else if (((hCount >= 100) && (hCount <= 300)) &&
            ((vCount >= 150) && (vCount <= 350))) rgb = WHITE;
  else rgb = RED; // background color
```

VGA Memory Requirements

- Remember, Spartan3e has 20 18kbit Block RAMs
  - i.e. 20k addresses where each address is a 16-bit (or 18 bit) word
  - But, 16 bits of address = 64k addresses
  - So, you can’t use all the address space with just Block RAMs
VGA Memory Requirements

- 640x480 VGA (bitmapped)
  - 307,200 pixels
  - 3 bits per pixel
  - 6 pixels per 18-bit word
  - 50k locations for 640x480
  - Oops – we only have 20k, and you need some space for code and other data…

VGA Memory Requirements

- 320x240 VGA (bitmapped)
  - 76,800 pixels
  - Each stored pixel is 2x2 screen pixels
  - 3 bits per pixel
  - 6 pixels per 18-bit word
  - 12.5k 18-bit words needed
  - Much more realistic… 7.5k left over for code/data
VGA Memory Requirements

- 80 char by 60 line display (8x8 glyphs)
  - 4800 locations
  - Each location has one of 256 char/glyphs
  - 8-bits per location – 2 locations per word
  - 2400 addresses for frame buffer
  - Each char/glyph is (say) 8x8 pixels
    - results in 640x480 display…
  - 8x8x256 bits for char/glyph table
    - 16kbits (1k words) for char/glyph table

Character Example…

64 characters
each 8x8 pixels
Character Example…

The Character ROM contains the 64 member ASCII upper-case character set. The characters are addressed with a 5-bit binary address \(A[4:0]\) and a 16-bit unary decoded address, \(nOE0\) to \(nOE120\). The Character ROM outputs a single row of the selected character at a time on the signals \(T[7:0]\).

\(A[4:3]\) decodes one of the four rows of 16 characters in the ROM.
\(A[4:3] = 0\) - first row
\(A[4:3] = 1\) - second row
\(A[4:3] = 2\) - third row
\(A[4:3] = 3\) - fourth row

Remember the skier/chicken/tron example?
That used character/glyph graphics similar to this…
Tbird VGA Assignment

- Get VGA working
  - Start with full-screen flood
  - then play around with direct VGA graphics
- Take the Tbird state machine
  - outputs are six lights
- Define six regions of the screen
  - Make those regions change color when the state machine says the lights should be on

Other I/O (more details later)

- LCD display
  - 2-line 16-char display
  - Reasonably easy to use, once you can do it under program control!
  - Reading and writing memory-mapped 8-bit registers
- PS/2 mouse/keyboard port
- RS323 connector and level converter
- DAC
  - 12 bit unsigned resolution – four outputs
- ADC
  - Dual-channel – 14 bit resolution
- Seven-segment LCDs
  - Already in your kits…
  - See the Starter Board users guide for more details!