Video

Prof. Stephen A. Edwards
sedwards@cs.columbia.edu

Columbia University
Spring 2008
Television: 1939 Du Mont Model 181

The Model 181 is a high console model which provides television sight and sound entertainment with a selection of four (4) television channels. The black and white picture of pleasing contrast is reproduced on the screen of the 14 inch teletron, and measures 8 inches by 10 inches. The beautifully grained walnut cabinet of pleasing modern design measures 48\% inches high, 23 inches wide and 26 inches deep. It is completely A.C., operated from standard 110 volt 60 cycle power lines. Twenty-two (22) tubes including the Du Mont Teletron are employed in the superhetrodynue circuit. A dynamic speaker is used for perfect sound reproduction. In addition, a three-band superhetrodynue all wave radio is provided for standard radio reception. This receiver employs 8 tubes, is completely A.C. operated from 110 volt 60 cycle power lines. Push button and manual tuning are provided. An individual dynamic speaker is used for broadcast sound reproduction.
Vector Displays

Game Over

Insert Coin

© Atari 1980
Raster Scanning
Raster Scanning
Raster Scanning
Raster Scanning
Raster Scanning
Originally black-and-white
60 Hz vertical scan frequency
15.75 kHz horizontal frequency

\[
\frac{15.75 \text{ kHz}}{60 \text{ Hz}} = 262.5 \text{ lines per field}
\]

White 1 V
Black 0.075 V
Blank 0 V
Sync − 0.4 V
A Line of B&W Video

White

Black

Blank

Sync

Front Porch: 0.02H
Sync: 0.08H
Back Porch: 0.06H
Blanking: 0.16H

H
Interlaced Scanning
Interlaced Scanning
Interlaced Scanning

- Diagram showing interlaced scanning with lines indicating the scanning process.
Interlaced Scanning

Video – p.
Interlaced Scanning
Color added later: had to be backwards compatible.

Solution: continue to transmit a “black-and-white” signal and modulate two color signals on top of it.

RGB vs. YIQ colorspaces

\[
\begin{bmatrix}
0.30 & 0.59 & 0.11 \\
0.60 & -0.28 & -0.32 \\
0.21 & -0.52 & 0.31 \\
\end{bmatrix}
\begin{bmatrix}
R \\
G \\
B \\
\end{bmatrix}
= 
\begin{bmatrix}
Y \\
I \\
Q \\
\end{bmatrix}
\]

Y baseband 4 MHz “black-and-white” signal
I as 1.5 MHz, Q as 0.5 MHz at 90°: modulated at 3.58 MHz
<table>
<thead>
<tr>
<th></th>
<th>lines</th>
<th>active lines</th>
<th>vertical res.</th>
<th>aspect ratio</th>
<th>horiz. res.</th>
<th>frame rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTSC</td>
<td>525</td>
<td>484</td>
<td>242</td>
<td>4:3</td>
<td>427</td>
<td>29.94 Hz</td>
</tr>
<tr>
<td>PAL</td>
<td>625</td>
<td>575</td>
<td>290</td>
<td>4:3</td>
<td>425</td>
<td>25 Hz</td>
</tr>
<tr>
<td>SECAM</td>
<td>625</td>
<td>575</td>
<td>290</td>
<td>4:3</td>
<td>465</td>
<td>25 Hz</td>
</tr>
</tbody>
</table>

PAL: Uses YUV instead of YIQ, flips phase of V every other line

SECAM: Transmits the two chrominance signals on alternate lines; no quadrature modulation
Computer Video: VGA

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Red</td>
<td>Green</td>
<td>Blue</td>
<td>ID2</td>
<td>GND</td>
</tr>
<tr>
<td>6</td>
<td>RGND</td>
<td>7</td>
<td>GGND</td>
<td>8</td>
<td>BGN D (+5V)</td>
</tr>
<tr>
<td>11</td>
<td>ID0</td>
<td>12</td>
<td>ID1</td>
<td>13</td>
<td>hsync</td>
</tr>
<tr>
<td>14</td>
<td>vsync</td>
<td>15</td>
<td>ID3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ID2**  **ID0**  **ID1**
---  ---  ---
-  -  GND  Monochrome, $< 1024 \times 768$
-  GND  -  Color, $< 1024 \times 768$
GND  GND  -  Color, $\geq 1024 \times 768$

**DDC1**
ID2  Data from display
vsync  also data clock

**DDC2**
ID1  I^2C SDA
ID3  I^2C SLC
### VGA Timing

<table>
<thead>
<tr>
<th>Mode</th>
<th>Resolution</th>
<th>Vertical</th>
<th>Horizontal</th>
<th>Pixel Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGA</td>
<td>640×350</td>
<td>70 Hz</td>
<td>31.5 kHz</td>
<td>25.175 MHz</td>
</tr>
<tr>
<td>VGA</td>
<td>640×400</td>
<td>70 Hz</td>
<td>31.5 kHz</td>
<td>25.175 MHz</td>
</tr>
<tr>
<td>VGA</td>
<td>640×480</td>
<td>59.94 Hz</td>
<td>31.469 kHz</td>
<td>25.175 MHz</td>
</tr>
<tr>
<td>SVGA</td>
<td>800×600</td>
<td>56 Hz</td>
<td>35.2 kHz</td>
<td>36 MHz</td>
</tr>
<tr>
<td>SVGA</td>
<td>800×600</td>
<td>60 Hz</td>
<td>37.8 kHz</td>
<td>40 MHz</td>
</tr>
<tr>
<td>SVGA</td>
<td>800×600</td>
<td>72 Hz</td>
<td>48.0 kHz</td>
<td>50 MHz</td>
</tr>
<tr>
<td>XGA</td>
<td>1024×768</td>
<td>60 Hz</td>
<td>48.5 kHz</td>
<td>65 MHz</td>
</tr>
<tr>
<td>SXGA</td>
<td>1280×1024</td>
<td>61 Hz</td>
<td>64.2 kHz</td>
<td>110 MHz</td>
</tr>
<tr>
<td>HDTV</td>
<td>1920×1080i</td>
<td>60 Hz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UXGA</td>
<td>1600×1200</td>
<td>60 Hz</td>
<td>75 kHz</td>
<td>162 MHz</td>
</tr>
<tr>
<td>UXGA</td>
<td>1600×1200</td>
<td>85 Hz</td>
<td>105.77 kHz</td>
<td>220 MHz</td>
</tr>
<tr>
<td>WUXGA</td>
<td>1920×1200</td>
<td>70 Hz</td>
<td>87.5 kHz</td>
<td>230 MHz</td>
</tr>
</tbody>
</table>
Detailed VGA Timing

640 x 480, “60 Hz”

25.175 MHz  Dot Clock  
31.469 kHz  Line Frequency  
59.94 Hz  Field Frequency

<table>
<thead>
<tr>
<th>pixels</th>
<th>role</th>
<th>lines</th>
<th>role</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Front Porch</td>
<td>2</td>
<td>Front Porch</td>
</tr>
<tr>
<td>96</td>
<td>Horizontal Sync</td>
<td>2</td>
<td>Vertical Sync</td>
</tr>
<tr>
<td>40</td>
<td>Back Porch</td>
<td>25</td>
<td>Back Porch</td>
</tr>
<tr>
<td>8</td>
<td>Left border</td>
<td>8</td>
<td>Top Border</td>
</tr>
<tr>
<td>640</td>
<td>Active</td>
<td>480</td>
<td>Active</td>
</tr>
<tr>
<td>8</td>
<td>Right border</td>
<td>8</td>
<td>Bottom Border</td>
</tr>
<tr>
<td>800</td>
<td>total per line</td>
<td>525</td>
<td>total per field</td>
</tr>
</tbody>
</table>

Active-low Horizontal and Vertical sync signals.
Let’s build a VHDL module that displays a $640 \times 480$ VGA raster with a white rectangle in the center against a blue background.
Video on the DE2

VGA_R[0..9]
VGA_G0
VGA_G1
VGA_G2
VGA_G3
VGA_G4
VGA_G5
VGA_G6
VGA_G7
VGA_G8
VGA_G9
VGA_SYNC
VGA_BLANK
VGA_HS
VGA_VS

VGA_R[0..9]
VGA_G[0..9]
VGA_B[0..9]
VGA_SYNC
VGA_CLOCK
VGA_HS
VGA_VS

VGA_VCC5
GND

ADV7123
LQFP48-0.5

U34

VGA_VCC5
GND

BC119 0.1U
BC120 0.1U
BC121 0.1U

R85
560

R84
4.7K

BC117 0.1U
BC118 0.1U

R95

R90
47

R88
75

R86
75

R87
75

R89
47

J13

DB15-RA-F2
For a 25.175 MHz pixel clock,

- **HSYNC**: 96 pixels
- **BACK_PORCH**: 48
- **HACTIVE**: 640
- **FRONT_PORCH**: 16
- **HTOTAL**: 800
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity de2_vga_raster is

port (reset : in std_logic;
clk : in std_logic; -- Should be 25.125 MHz
VGA_CLK, -- Clock
VGA_HS, -- H_SYNC
VGA_VS, -- V_SYNC
VGA_BLANK, -- BLANK
VGA_SYNC : out std_logic; -- SYNC
VGA_R, -- Red[9:0]
VGA_G, -- Green[9:0]
VGA_B : out unsigned(9 downto 0) -- Blue[9:0]
);

end de2_vga_raster;
architecture rtl of de2_vga_raster is

--- Video parameters

constant HTOTAL : integer := 800;
constant HSYNC : integer := 96;
constant HBACK_PORCH : integer := 48;
constant HACTIVE : integer := 640;
constant HFRONT_PORCH : integer := 16;

constant VTOTAL : integer := 525;
constant VSYNC : integer := 2;
constant VBACK_PORCH : integer := 33;
constant VACTIVE : integer := 480;
constant VFRONT_PORCH : integer := 10;

constant RECTANGLE_HSTART : integer := 100;
constant RECTANGLE_HEND : integer := 540;
constant RECTANGLE_VSTART : integer := 100;
constant RECTANGLE_VEND : integer := 380;
-- Signals for the video controller

-- Horizontal position (0-800)
signal Hcount : unsigned(9 downto 0);

-- Vertical position (0-524)
signal Vcount : unsigned(9 downto 0);

signal EndOfLine, EndOfField : std_logic;

signal vga_hblank, vga_hsync,
    vga_vblank, vga_vsync : std_logic;  -- Sync. signals

-- rectangle area
signal rectangle_h, rectangle_v, rectangle : std_logic;

begin
Counters

HCounter : process (clk)
begin
    if rising_edge(clk) then
        if reset = '1' or EndOfLine = '1' then Hcount <= (others => '0');
        else Hcount <= Hcount + 1;
    end if; end if;
end process HCounter;

EndOfLine <= '1' when Hcount = HTOTAL - 1 else '0';

VCounter: process (clk)
begin
    if rising_edge(clk) then
        if reset = '1' then Vcount <= (others => '0');
        elsif EndOfLine = '1' then
            if EndOfField = '1' then Vcount <= (others => '0');
            else Vcount <= Vcount + 1;
        end if; end if; end if;
end process VCounter;

EndOfField <= '1' when Vcount = VTOTAL - 1 else '0';
Horizontal signals

HSyncGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' or EndOfLine = '1' then
      vga_hsync <= '1';
    elsif Hcount = HSYNC - 1 then
      vga_hsync <= '0';
    end if;
  end if;
end process HSyncGen;

HBlankGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' then
      vga_hblank <= '1';
    elsif Hcount = HSYNC + HBACK_PORCH then
      vga_hblank <= '0';
    elsif Hcount = HSYNC + HBACK_PORCH + HACTIVE then
      vga_hblank <= '1';
    end if;
  end if;
end process HBlankGen;
Vertical signals

VSyncGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' then vga_vsync <= '1';
    elsif EndOfLine = '1' then
      if EndOfField = '1' then vga_vsync <= '1';
      elsif Vcount = VSYNC - 1 then vga_vsync <= '0';
      end if;
    end if;
  end if;
end process VSyncGen;

VBlankGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' then vga_vblank <= '1';
    elsif EndOfLine = '1' then
      if Vcount = VSYNC + VBACK_PORCH - 1 then
        vga_vblank <= '0';
      elsif Vcount = VSYNC + VBACK_PORCH + VACTIVE - 1 then
        vga_vblank <= '1';
      end if; end if; end if;
  end if;
end process VBlankGen;
The Rectangle

RectangleHGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' or Hcount = HSYNC + HBACK_PORCH + RECTANGLE_HSTART then
      rectangle_h <= '1';
    elsif Hcount = HSYNC + HBACK_PORCH + RECTANGLE_HEND then
      rectangle_h <= '0';
    end if; end if;
  end if;
end process RectangleHGen;

RectangleVGen : process (clk)
begin
  if rising_edge(clk) then
    if reset = '1' then rectangle_v <= '0';
    elsif EndOfLine = '1' then
      if Vcount = VSYNC + VBACK_PORCH - 1 + RECTANGLE_VSTART then
        rectangle_v <= '1';
      elsif Vcount = VSYNC + VBACK_PORCH - 1 + RECTANGLE_VEND then
        rectangle_v <= '0';
      end if; end if; end if;
  end if;
end process RectangleVGen;

rectangle <= rectangle_h and rectangle_v;
Output signals

VideoOut: **process** (clk, reset)

begin

  if reset = '1' then
      VGA_R <= "0000000000"; VGA_G <= "0000000000"; VGA_B <= "0000000000";
  elsif clk'event and clk = '1' then
      if rectangle = '1' then
          VGA_R <= "1111111111"; VGA_G <= "1111111111"; VGA_B <= "1111111111"
      elsif vga_hblank = '0' and vga_vblank = '0' then
          VGA_R <= "0000000000"; VGA_G <= "0000000000"; VGA_B <= "1111111111"
      else
          VGA_R <= "0000000000"; VGA_G <= "0000000000"; VGA_B <= "0000000000"
      end if;
  end if;
end if;
end process VideoOut;

VGA_CLK <= clk;
VGA_HS <= not vga_hsync;
VGA_VS <= not vga_vsync;
VGA_SYNC <= '0';
VGA_BLANK <= not (vga_hsync or vga_vsync);

end rtl;