**ECEN 323 – Winter 2020**

Lab 2: VGA Display

Johnson, Ryan

Section 01

Preliminary

What is the frequency of the pixel clock for the 640×480 VGA resolution?

25 MHz

What is the frequency of the horizontal sync signal (HS) for the 640×480 resolution?

31250 Hz

During the scan of one horizontal row, 640 pixels are displayed. However, additional time is needed during the horizontal scan for retracing. How many pixel clocks are needed during each horizontal sync for this retracing (i.e. how many pixel clocks are used when no pixel is displayed in a horizontal scan)?

160 Pixel Clocks

How many frames per second are generated with these timing specifications?

Approximately 60 (59.88)

How many lines are NOT displayed during a full frame (i.e. lines that are blanked during a vertical retrace)?

41 Lines

Exercise #1

Exercise #2

Submit your .tcl simulation script

restart

add\_force clk {0 0} {1 5ns} -repeat\_every 10ns

add\_force rst 1

run 300 ns

add\_force rst 0

run 300000000 ns

Submit your working VGA timing controller

`timescale 1ns / 1ps

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Module: vga\_timing

\*

\* Author: Ryan Johnson

\* Class: ECEN 323, Section 01, Winter 2020

\* Date: 13 January 2020

\*

\* Description: to correctly time a display on a vga screen

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

module vga\_timing(

input wire logic clk,

input wire logic rst,

output logic hs,

output logic vs,

output logic [9:0] pixel\_x,

output logic [9:0] pixel\_y,

output logic last\_column,

output logic last\_row,

output logic blank

);

logic pixel\_en;

logic [1:0] pixel\_cnt;

localparam CLOCKS\_PER\_PIXEL\_CLK = 4;

localparam HORIZONTAL\_PIXELS = 800;

localparam VERTICAL\_PIXELS = 521;

localparam LAST\_COLUMN = 639;

localparam LAST\_ROW = 479;

localparam HOR\_FRONT\_PORCH = 16;

localparam HOR\_PULSE\_WIDTH = 96;

localparam VER\_FRONT\_PORCH = 10;

localparam VER\_PULSE\_WIDTH = 2;

// incrementing pixel\_cnt to enable pixel\_en every fourth clock

always\_ff@(posedge clk)

begin

if (rst)

pixel\_cnt <= 0;

else

pixel\_cnt <= pixel\_cnt +1;

end

assign pixel\_en = (pixel\_cnt == CLOCKS\_PER\_PIXEL\_CLK - 1) ? 1'b1 : 1'b0;

//incrementing pixel\_x until it reaches max value HORIZONTAL\_PIXELS

always\_ff@(posedge clk)

begin

if (rst)

pixel\_x <= 0;

else if (pixel\_en == 1'b1)

begin

if (pixel\_x == HORIZONTAL\_PIXELS - 1)

pixel\_x <= 0;

else

pixel\_x <= pixel\_x +1;

end

end

assign hs = (pixel\_x <= LAST\_COLUMN + HOR\_FRONT\_PORCH) ? 1 :

(pixel\_x <= LAST\_COLUMN + HOR\_FRONT\_PORCH + HOR\_PULSE\_WIDTH) ? 0 : 1;

assign last\_column = (pixel\_x == LAST\_COLUMN) ? 1'b1 : 1'b0;

//incrementing pixel\_y after each row of pixel\_x is incremented until it reaches max value VERTICAL\_PIXELS

always\_ff@(posedge clk)

begin

if (rst)

pixel\_y <= 0;

else if (pixel\_en == 1'b1 && pixel\_x == HORIZONTAL\_PIXELS - 1)

begin

if (pixel\_y == VERTICAL\_PIXELS - 1)

pixel\_y <= 0;

else

pixel\_y <= pixel\_y +1;

end

end

assign vs = (pixel\_y <= LAST\_ROW + VER\_FRONT\_PORCH) ? 1 :

(pixel\_y <= LAST\_ROW + VER\_FRONT\_PORCH + VER\_PULSE\_WIDTH) ? 0 : 1;

assign last\_row = (pixel\_y == LAST\_ROW) ? 1'b1 : 1'b0;

assign blank = (pixel\_x <= LAST\_COLUMN && pixel\_y <= LAST\_ROW) ? 0 : 1;

endmodule

Exercise #3

How many distinct colors can be displayed with 12-bits of color information?

4096

Submit your .tcl simulation script

restart

add\_force clk {0 0} {1 5} -repeat\_every 10

add\_force btnc 1

run 20ns;

add\_force btnc 0;

run 30000000ns;

Submit your top-level VGA test circuit

`timescale 1ns / 1ps

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Module: vga\_test

\*

\* Author: Ryan Johnson

\* Class: ECEN 323, Section 01, Winter 2020

\* Date: 13 January 2020

\*

\* Description: a top module to test our vga display module

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

module vga\_test(

input wire logic clk,

input wire logic btnc,

output logic hs,

output logic vs,

output logic [3:0] vgaRed,

output logic [3:0] vgaGreen,

output logic [3:0] vgaBlue

);

localparam BLACK = 79;

localparam BLUE = 159;

localparam GREEN = 239;

localparam CYAN = 319;

localparam RED = 399;

localparam MAGENTA = 479;

localparam YELLOW = 559;

localparam MAX = 4'b1111;

localparam MIN = 4'b0000;

logic blue\_color;

logic red\_color;

logic green\_color;

logic blue;

logic red;

logic green;

logic hsreg;

logic vsreg;

logic [9:0] pixel\_x;

logic [9:0] pixel\_y;

logic last\_column;

logic last\_row;

logic blank;

//instantiating our vga\_timing module to get the various values through

vga\_timing v1(.clk(clk), .rst(btnc), .hs(hsreg), .vs(vsreg), .pixel\_x(pixel\_x), .pixel\_y(pixel\_y), .last\_column(last\_column), .last\_row(last\_row), .blank(blank) );

//register for hs to avoid glitches in timing signals

always\_ff@(posedge clk)

hs <= hsreg;

//register for vs to avoid glitches in timing signals

always\_ff@(posedge clk)

vs <= vsreg;

assign red\_color = (pixel\_x <= CYAN) ? MIN : MAX;

assign green\_color = (pixel\_x <= BLUE) ? MIN :

(pixel\_x <= CYAN) ? MAX :

(pixel\_x <= MAGENTA)? MIN : MAX;

assign blue\_color = (pixel\_x <= BLACK) ? MIN :

(pixel\_x <= BLUE) ? MAX :

(pixel\_x <= GREEN) ? MIN :

(pixel\_x <= CYAN) ? MAX :

(pixel\_x <= RED) ? MIN :

(pixel\_x <= MAGENTA)? MAX :

(pixel\_x <= YELLOW) ? MIN : MAX;

assign red = blank ? 4'd0 : red\_color;

assign green = blank ? 4'd0 : green\_color;

assign blue = blank ? 4'd0 : blue\_color;

//register for vga colors

always\_ff@(posedge clk)

begin

vgaRed <= red;

vgaGreen <= green;

vgaBlue <= blue;

end

endmodule

Exercise #4

Include a copy of your .xdc file in your laboratory report

###################################################################

#

# nexys4\_323.xdc

#

# This is a master constraints file for laboratory assignments used

# at BYU for ECEN 323. This file is based on the nexys\_4\_master.xdc

# file provided by Digilent. Many of the signals that are not used for

# ECEN 323 labs have been removed for simplicity.

#

# You should uncomment those lines that define ports that you used

# in your top-level design. You should also change the name of the

# port in the .xdc file to match your corresponding top-level port.

#

###################################################################

# Clock

set\_property -dict { PACKAGE\_PIN E3 IOSTANDARD LVCMOS33 } [get\_ports { clk }]; #IO\_L12P\_T1\_MRCC\_35 Sch=clk100mhz

create\_clock -add -name sys\_clk\_pin -period 10.00 -waveform {0 5} [get\_ports {clk}];

# Buttons

set\_property -dict { PACKAGE\_PIN N17 IOSTANDARD LVCMOS33 } [get\_ports { btnc }]; #IO\_L9P\_T1\_DQS\_14 Sch=btnc

# set\_property -dict { PACKAGE\_PIN M18 IOSTANDARD LVCMOS33 } [get\_ports { btnu }]; #IO\_L4N\_T0\_D05\_14 Sch=btnu

# set\_property -dict { PACKAGE\_PIN P17 IOSTANDARD LVCMOS33 } [get\_ports { btnl }]; #IO\_L12P\_T1\_MRCC\_14 Sch=btnl

# set\_property -dict { PACKAGE\_PIN M17 IOSTANDARD LVCMOS33 } [get\_ports { btnr }]; #IO\_L10N\_T1\_D15\_14 Sch=btnr

# set\_property -dict { PACKAGE\_PIN P18 IOSTANDARD LVCMOS33 } [get\_ports { btnd }]; #IO\_L9N\_T1\_DQS\_D13\_14 Sch=btnd

#VGA Connector

set\_property -dict { PACKAGE\_PIN A3 IOSTANDARD LVCMOS33 } [get\_ports { vgaRed[0] }]; #IO\_L8N\_T1\_AD14N\_35 Sch=vga\_r[0]

set\_property -dict { PACKAGE\_PIN B4 IOSTANDARD LVCMOS33 } [get\_ports { vgaRed[1] }]; #IO\_L7N\_T1\_AD6N\_35 Sch=vga\_r[1]

set\_property -dict { PACKAGE\_PIN C5 IOSTANDARD LVCMOS33 } [get\_ports { vgaRed[2] }]; #IO\_L1N\_T0\_AD4N\_35 Sch=vga\_r[2]

set\_property -dict { PACKAGE\_PIN A4 IOSTANDARD LVCMOS33 } [get\_ports { vgaRed[3] }]; #IO\_L8P\_T1\_AD14P\_35 Sch=vga\_r[3]

set\_property -dict { PACKAGE\_PIN C6 IOSTANDARD LVCMOS33 } [get\_ports { vgaGreeb[0] }]; #IO\_L1P\_T0\_AD4P\_35 Sch=vga\_g[0]

set\_property -dict { PACKAGE\_PIN A5 IOSTANDARD LVCMOS33 } [get\_ports { vgaGreen[1] }]; #IO\_L3N\_T0\_DQS\_AD5N\_35 Sch=vga\_g[1]

set\_property -dict { PACKAGE\_PIN B6 IOSTANDARD LVCMOS33 } [get\_ports { vgaGreen[2] }]; #IO\_L2N\_T0\_AD12N\_35 Sch=vga\_g[2]

set\_property -dict { PACKAGE\_PIN A6 IOSTANDARD LVCMOS33 } [get\_ports { vgaGreen[3] }]; #IO\_L3P\_T0\_DQS\_AD5P\_35 Sch=vga\_g[3]

set\_property -dict { PACKAGE\_PIN B7 IOSTANDARD LVCMOS33 } [get\_ports { vgaBlue[0] }]; #IO\_L2P\_T0\_AD12P\_35 Sch=vga\_b[0]

set\_property -dict { PACKAGE\_PIN C7 IOSTANDARD LVCMOS33 } [get\_ports { vgaBlue[1] }]; #IO\_L4N\_T0\_35 Sch=vga\_b[1]

set\_property -dict { PACKAGE\_PIN D7 IOSTANDARD LVCMOS33 } [get\_ports { vgaBlue[2] }]; #IO\_L6N\_T0\_VREF\_35 Sch=vga\_b[2]

set\_property -dict { PACKAGE\_PIN D8 IOSTANDARD LVCMOS33 } [get\_ports { vgaBlue[3] }]; #IO\_L4P\_T0\_35 Sch=vga\_b[3]

set\_property -dict { PACKAGE\_PIN B11 IOSTANDARD LVCMOS33 } [get\_ports { VGA\_HS }]; #IO\_L4P\_T0\_15 Sch=vga\_hs

set\_property -dict { PACKAGE\_PIN B12 IOSTANDARD LVCMOS33 } [get\_ports { VGA\_VS }]; #IO\_L3N\_T0\_DQS\_AD1N\_15 Sch=vga\_vs

Summarize any synthesis warnings you received. Indicate “none” if you had no warnings.

No synthesis warnings.

Summarize the resource utilization of your design providing the following 1) Total number of Slices, 2) Total number of Bonded IOBs, and 3) summarize the type and number of primitives in the design (search for “Primitive” in the utilization report)

1. Slice Logic

--------------

+-------------------------+------+-------+-----------+-------+

| Site Type | Used | Fixed | Available | Util% |

+-------------------------+------+-------+-----------+-------+

| Slice LUTs\* | 32 | 0 | 63400 | 0.05 |

| LUT as Logic | 32 | 0 | 63400 | 0.05 |

| LUT as Memory | 0 | 0 | 19000 | 0.00 |

| Slice Registers | 27 | 0 | 126800 | 0.02 |

| Register as Flip Flop | 27 | 0 | 126800 | 0.02 |

| Register as Latch | 0 | 0 | 126800 | 0.00 |

| F7 Muxes | 0 | 0 | 31700 | 0.00 |

| F8 Muxes | 0 | 0 | 15850 | 0.00 |

+-------------------------+------+-------+-----------+-------+

4. IO and GT Specific

---------------------

+-----------------------------+------+-------+-----------+-------+

| Site Type | Used | Fixed | Available | Util% |

+-----------------------------+------+-------+-----------+-------+

| Bonded IOB | 16 | 0 | 210 | 7.62 |

| Bonded IPADs | 0 | 0 | 2 | 0.00 |

| PHY\_CONTROL | 0 | 0 | 6 | 0.00 |

| PHASER\_REF | 0 | 0 | 6 | 0.00 |

| OUT\_FIFO | 0 | 0 | 24 | 0.00 |

| IN\_FIFO | 0 | 0 | 24 | 0.00 |

| IDELAYCTRL | 0 | 0 | 6 | 0.00 |

| IBUFDS | 0 | 0 | 202 | 0.00 |

| PHASER\_OUT/PHASER\_OUT\_PHY | 0 | 0 | 24 | 0.00 |

| PHASER\_IN/PHASER\_IN\_PHY | 0 | 0 | 24 | 0.00 |

| IDELAYE2/IDELAYE2\_FINEDELAY | 0 | 0 | 300 | 0.00 |

| ILOGIC | 0 | 0 | 210 | 0.00 |

| OLOGIC | 0 | 0 | 210 | 0.00 |

+-----------------------------+------+-------+-----------+-------+

7. Primitives

-------------

+----------+------+---------------------+

| Ref Name | Used | Functional Category |

+----------+------+---------------------+

| FDRE | 27 | Flop & Latch |

| LUT6 | 17 | LUT |

| OBUF | 14 | IO |

| LUT5 | 8 | LUT |

| LUT2 | 6 | LUT |

| LUT4 | 5 | LUT |

| LUT3 | 5 | LUT |

| IBUF | 2 | IO |

| LUT1 | 1 | LUT |

| BUFG | 1 | Clock |

+----------+------+---------------------+

How many hours did you work on the lab?

4 hours

Please provide any suggestions for improving this lab in the future:

No suggestions. It was great.