EE3921 Digital System Design

Section 031, Fall 2020

Professor: Kerry R. Widder, Ph.D.

Electrical Engineering and Computer Science Department

Milwaukee School of Engineering

Laboratory 5: “Nios II Switch Selector for VGA"

Jorge J. Jurado-Garcia

Date: 10/28/2021

Level Attempted: Medium

Objective: The objective of this lab was to create a Nios II microprocessor with a GPIO peripherals to add color selection functionality to my design. While using pixel buffers to output graphical information to a VGA monitor. This program should draw a rectangle using the selected color and check the switches value at least twice per second. And draw an x in the rectangle.

Description: In this lab I ended up created my Nios II with the described peripherals of

**include** "altera\_up\_avalon\_video\_pixel\_buffer\_dma.h"

**#include** "altera\_avalon\_pio\_regs.h"

Where a pointer was created to access the pixel\_buffer\_dma functions and reference in the dma functions. Afterwards I set a variable equal to the input read switches for my SW\_PIO\_BASE and uses if nested statements to select the colors. In terms of making the rectangle and X value inside my rectangle. I used two main functions. alt\_up\_pixel\_buffer\_dma\_draw\_rectangle()

alt\_up\_pixel\_buffer\_dma\_draw\_line()

Conclusion: Overall, this lab was a success, the toughest part I had with this lab would be

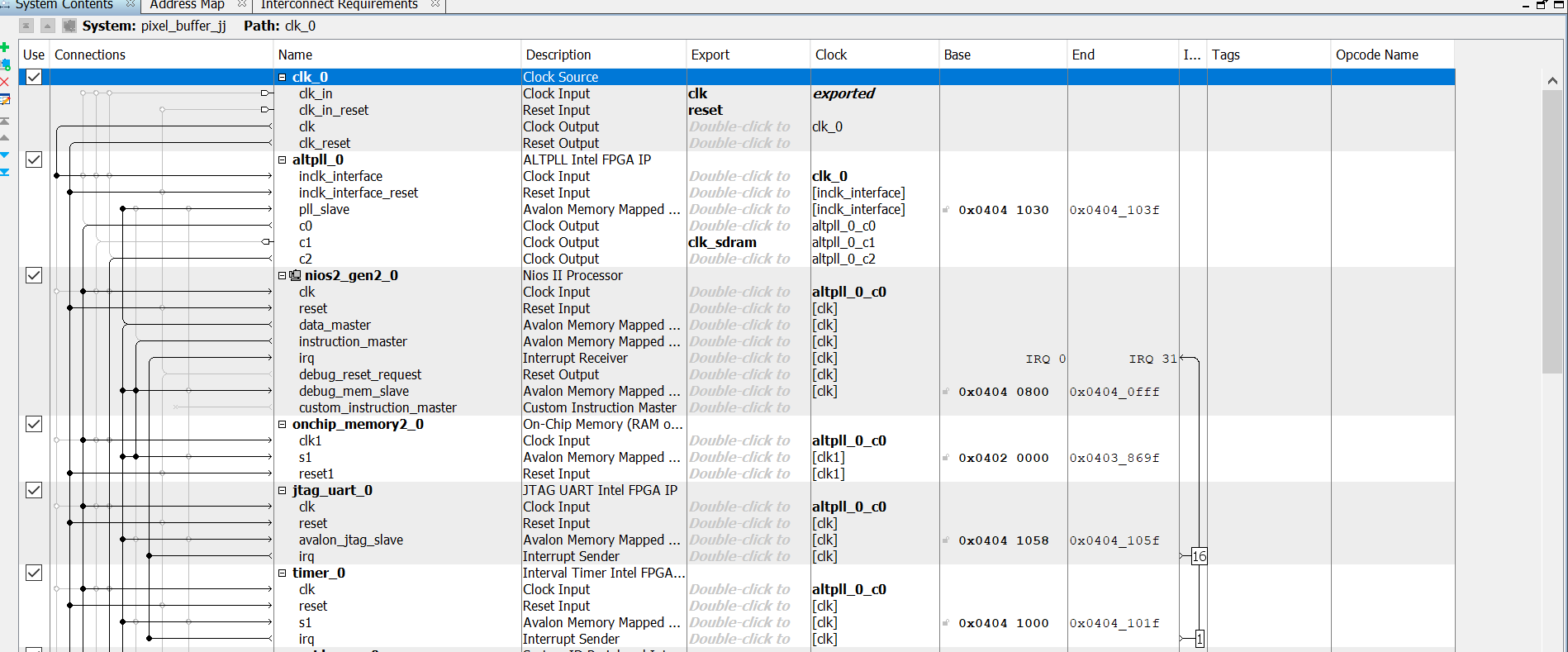
Debugging my hardware portion of my NIOS to display the written information on the monitor. I had a lot of trouble in figuring out which component was the culprit and after spending about 1-2 hour on going back on my work I was able to find the reason I couldn’t see anything is something to do with VGA controller width/heigh set by pixel was off by a factor of 10. Once I fixed that problem all I had to do is reload my program. Generate HDL, download spoc, QIP file, and gender bsp for my program. After doing all these my program worked well. Adding the GPIO pins to my noise processor was super easy to implement and so was written the code.

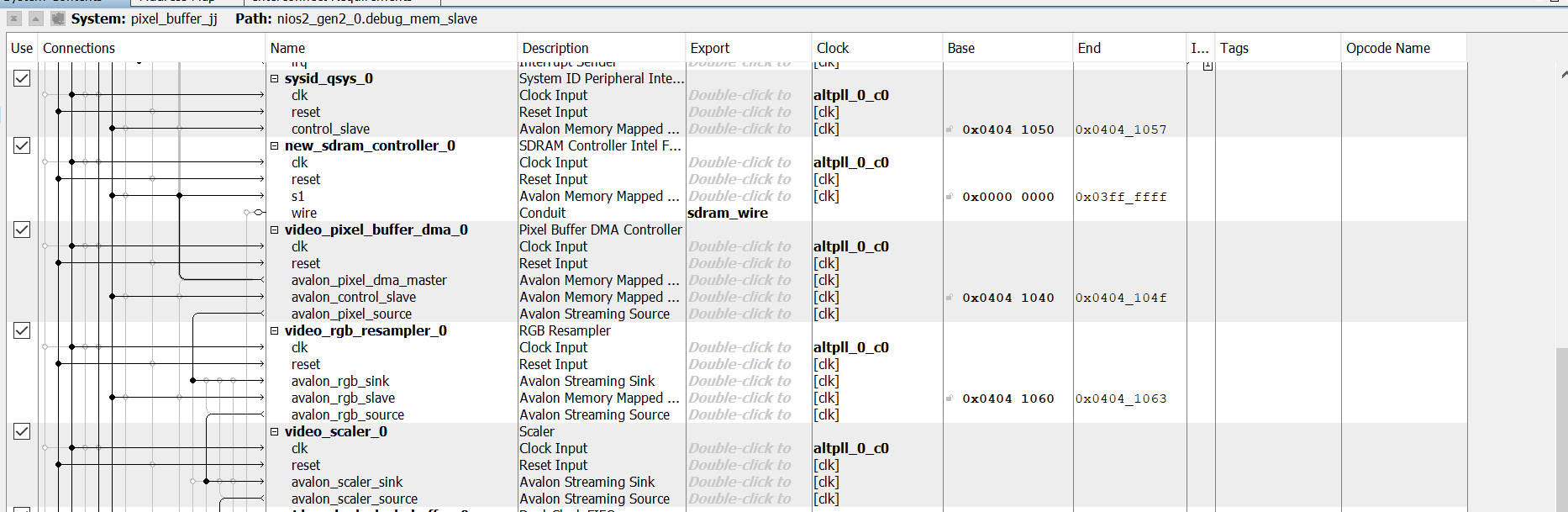
A part that I do like about this project is the fact I can use this template for final project specifically on the 1st prototype we have to implement so it is noice knowing that all the VGA monitor stuff and gpio switch is working fine.

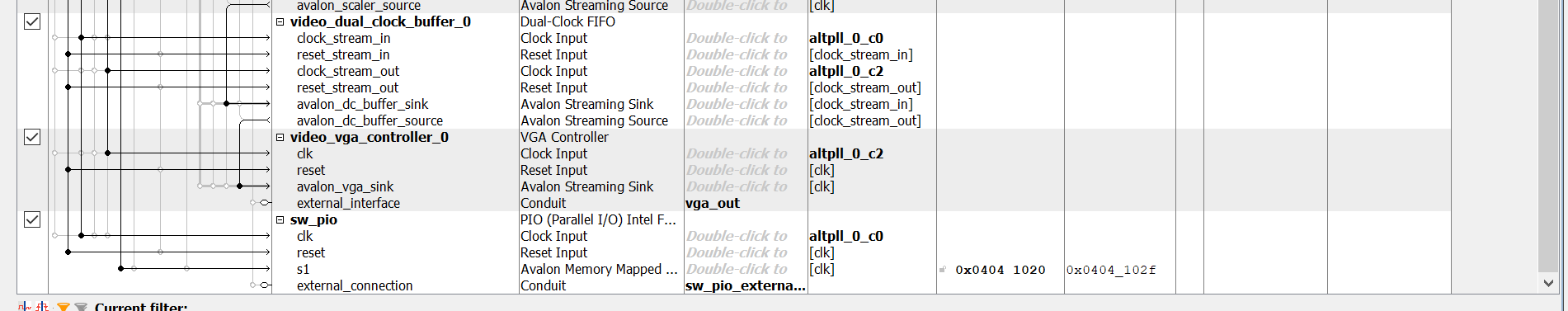
Counter\_RTL\_Viewer

A picture containing diagram

Description automatically generated







VHDL CODE

Niose\_pixel\_buffer\_with\_color\_DE10\_.vhd

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

--

--Niose\_pixel\_buffer\_with color\_de10\_lite.vhdl

--

--created: 10/23/2021

--By: Jorge Jurado-Garcia

--Rev: 0

--

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

**library** ieee**;**

**use** ieee**.**std\_logic\_1164**.all;**

**use** ieee**.**numeric\_std**.all;**

**entity** nios\_pixel\_de10 **is**

**port(**

CLOCK\_50 **:** **in** std\_logic**;**

SW**:** **in** std\_logic\_vector**(**2 **downto** 0**);**

DRAM\_ADDR**:** **out** std\_logic\_vector**(**12 **downto** 0**);**

DRAM\_BA**:** **out** std\_logic\_vector**(**1 **downto** 0**);**

DRAM\_CAS\_N**:** **out** std\_logic**;**

DRAM\_CKE**:** **out** std\_logic**;**

DRAM\_CS\_N**:** **out** std\_logic**;**

DRAM\_RAS\_N**:** **out** std\_logic**;**

DRAM\_WE\_N**:** **out** std\_logic**;**

DRAM\_DQ**:** **inout** std\_logic\_vector**(**15 **downto** 0**);**

DRAM\_UDQM**:** **out** std\_logic**;**

DRAM\_LDQM**:** **out** std\_logic**;**

VGA\_HS**:** **out** std\_logic**;**

VGA\_VS**:** **out** std\_logic**;**

VGA\_R**:** **out** std\_logic\_vector**(**3 **downto** 0**);**

VGA\_G**:** **out** std\_logic\_vector**(**3 **downto** 0**);**

VGA\_B**:** **out** std\_logic\_vector**(**3 **downto** 0**);**

DRAM\_CLK**:** **out** std\_logic

**);**

**end** **entity;**

**architecture** behavioral **of** nios\_pixel\_de10 **is**

--

--no signals

**component** pixel\_buffer\_jj **is**

**port** **(**

clk\_clk **:** **in** std\_logic **:=** 'X'**;** -- clk

clk\_sdram\_clk **:** **out** std\_logic**;**

reset\_reset\_n **:** **in** std\_logic **:=** 'X'**;**

sdram\_wire\_addr **:** **out** std\_logic\_vector**(**12 **downto** 0**);**

sdram\_wire\_ba **:** **out** std\_logic\_vector**(**1 **downto** 0**);**

sdram\_wire\_cas\_n **:** **out** std\_logic**;** -- cas\_n

sdram\_wire\_cke **:** **out** std\_logic**;** -- cke

sdram\_wire\_cs\_n **:** **out** std\_logic**;** -- cs\_n

sdram\_wire\_dq **:** **inout** std\_logic\_vector**(**15 **downto** 0**)** **:=** **(others** **=>** 'X'**);** -- dq

sdram\_wire\_dqm **:** **out** std\_logic\_vector**(**1 **downto** 0**);** -- dqm

sdram\_wire\_ras\_n **:** **out** std\_logic**;** -- ras\_n

sdram\_wire\_we\_n **:** **out** std\_logic**;** -- we\_n

vga\_out\_CLK **:** **out** std\_logic**;** -- CLK

vga\_out\_HS **:** **out** std\_logic**;** -- HS

vga\_out\_VS **:** **out** std\_logic**;** -- VS

vga\_out\_BLANK **:** **out** std\_logic**;** -- BLANK

vga\_out\_SYNC **:** **out** std\_logic**;** -- SYNC

vga\_out\_R **:** **out** std\_logic\_vector**(**3 **downto** 0**);** -- R

vga\_out\_G **:** **out** std\_logic\_vector**(**3 **downto** 0**);** -- G

vga\_out\_B **:** **out** std\_logic\_vector**(**3 **downto** 0**);**

sw\_pio\_external\_connection\_export **:** **in** std\_logic\_vector**(**2 **downto** 0**)** **:=** **(others** **=>** 'X'**)** -- export

**);**

**end** **component** pixel\_buffer\_jj**;**

**begin**

u0 **:** **component** pixel\_buffer\_jj

**port** **map** **(**

clk\_clk **=>** CLOCK\_50**,** -- clk.clk

clk\_sdram\_clk **=>** DRAM\_CLK**,** -- clk\_sdram.clk

reset\_reset\_n **=>** '1'**,** -- reset.reset\_n

sdram\_wire\_addr **=>** DRAM\_ADDR**,** -- sdram\_wire.addr

sdram\_wire\_ba **=>** DRAM\_BA**,** -- .ba

sdram\_wire\_cas\_n **=>** DRAM\_CAS\_N**,** -- .cas\_n

sdram\_wire\_cke **=>** DRAM\_CKE**,** -- .cke

sdram\_wire\_cs\_n **=>** DRAM\_CS\_N**,** -- .cs\_n

sdram\_wire\_dq **=>** DRAM\_DQ**,** -- .dq

sdram\_wire\_dqm**(**1**)** **=>** DRAM\_UDQM**,** -- .dqm

sdram\_wire\_dqm**(**0**)** **=>** DRAM\_LDQM**,**

sdram\_wire\_ras\_n **=>** DRAM\_RAS\_N**,** -- .ras\_n

sdram\_wire\_we\_n **=>** DRAM\_WE\_N**,** -- .we\_n

--vga\_out\_CLK => CONNECTED\_TO\_vga\_out\_CLK, -- vga\_out.CLK

vga\_out\_HS **=>** VGA\_HS**,** -- .HS

vga\_out\_VS **=>** VGA\_VS**,** -- .VS

--vga\_out\_BLANK => CONNECTED\_TO\_vga\_out\_BLANK, -- .BLANK

--vga\_out\_SYNC => CONNECTED\_TO\_vga\_out\_SYNC, -- .SYNC

vga\_out\_R **=>** VGA\_R**,** -- .R

vga\_out\_G **=>** VGA\_G**,** -- .G

vga\_out\_B **=>** VGA\_B**,** -- .B

sw\_pio\_external\_connection\_export **=>** Sw**(**2 **downto** 0**)** -- sw\_pio\_external\_connection.export

**);**

**end** **architecture;**

C SOURCE CODE

/\*

\* Medium\_pixel.c

\*

\* Created on: Oct 28, 2021

\* Author: jurado-garciaj

\*/

//includes

#include "altera\_up\_avalon\_video\_pixel\_buffer\_dma.h"

#include "altera\_avalon\_pio\_regs.h"

#include <stdio.h>

#include <unistd.h>

#include "system.h"

int main**(**void**){**

alt\_u8 sw**;**

alt\_u32 c1**,**c2**,**c3**,** color**;**

printf**(**"Processor started sucessfully - Entered Main of rectangle.c\n\n"**);**

//define a pointer of type buffer...

//to use a reference in the dma functions

//

alt\_up\_pixel\_buffer\_dma\_dev**\***pixel\_buf\_dma\_dev**;**

//open the pixel buffer port

//-command is in drivers/inc/alter

//name reference is in the system.h

pixel\_buf\_dma\_dev **=** alt\_up\_pixel\_buffer\_dma\_open\_dev **(**VIDEO\_PIXEL\_BUFFER\_DMA\_0\_NAME**);**

//check for errors and output to the console

**if(** pixel\_buf\_dma\_dev **==** **NULL)**

printf**(**"Error: could not open pixel buffer devion \n"**);**

**else**

printf**(**"Opened pixel buffer device \n"**);**

//clear the screen

//-command is in drivers/inc/alter....video\_pixel\_buffer\_dma.h

//-wait until done before continuing

//

sw **=** 0**;**

**while(**1**){**

sw **=** IORD\_ALTERA\_AVALON\_PIO\_DATA**(**SW\_PIO\_BASE**);**

c1**=**0**;**

c2**=**0**;**

c3**=**0**;**

color**=**0**;**

alt\_up\_pixel\_buffer\_dma\_clear\_screen**(**pixel\_buf\_dma\_dev**,** 0**);**

alt\_up\_pixel\_buffer\_dma\_clear\_screen**(**pixel\_buf\_dma\_dev**,** 1**);**

**if(**sw **==** 1**){** //sw[0]

c1 **=** 0xF800**;** //red

c2 **=** 0**;**

c3 **=**0**;**

**}**

**if((**sw **==** 2**)){** //sw[1]

c2 **=** 0x07E0**;** //green

c1 **=** 0**;**

c3 **=**0**;**

**}**

**if((**sw **==** 3**)){** //sw[1]

c1 **=** 0xF800**;**

c2 **=** 0x07E0**;** //green

c3 **=** 0**;**

**}**

**if((**sw **==** 4**)){** //sw[1]

c2 **=** 0**;** //green

c1 **=** 0**;**

c3 **=** 0x001F**;**

**}**

**if((**sw **==** 5**)){** //sw[1]

c1 **=** 0xF800**;**

c2 **=** 0**;** //green

c3 **=** 0x001F**;**

**}**

**if((**sw **==** 6**)){** //sw[1]

c2 **=** 0x07E0**;** //green

c1 **=** 0**;**

c3 **=** 0x001F**;**

**}**

**if((**sw **==** 7**)){** //sw[1]

c1 **=** 0xF800**;**

c2 **=** 0x07E0**;** //green

c3 **=** 0x001F**;**

**}**

color **=** c1**+**c2**+**c3**;**

printf**(**"color is: %2X \n"**,** color**);**

printf**(**"Sw value is: %d \n"**,** sw**);**

alt\_up\_pixel\_buffer\_dma\_draw\_rectangle**(**pixel\_buf\_dma\_dev**,** 80**,** 75**,** 269**,** 180**,** color**,** 0**);**

alt\_up\_pixel\_buffer\_dma\_draw\_line**(**pixel\_buf\_dma\_dev**,** 80**,**75**,** 269**,**180**,** color**,** 0**);**

alt\_up\_pixel\_buffer\_dma\_draw\_line**(**pixel\_buf\_dma\_dev**,** 80**,**180**,** 269**,**75**,** color**,** 0**);**

usleep**(**2000000**);**

printf**(**"project is complete \n"**);**

sw**++;**

**}**//end while

**return** 0**;**

**}**//end main