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 4: “Nios II Counter and Switch board"

Jorge J. Jurado-Garcia

Date: 10/17/2020

Level Attempted: Advance

Objective: The objective of this lab was to create a Nios II microprocessor with 3 PIO peripherals in the microcontroller. With 16-bit output, 8-bit input, and 16-bit output. Each peripheral will control (HEX0 & HEX1), SW[7,0], and (HEX2 & HEX3) respectively. Using the software tool create a c project and BSP and program your board.

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

led\_pio\_external\_connection\_export (15 downto 0)

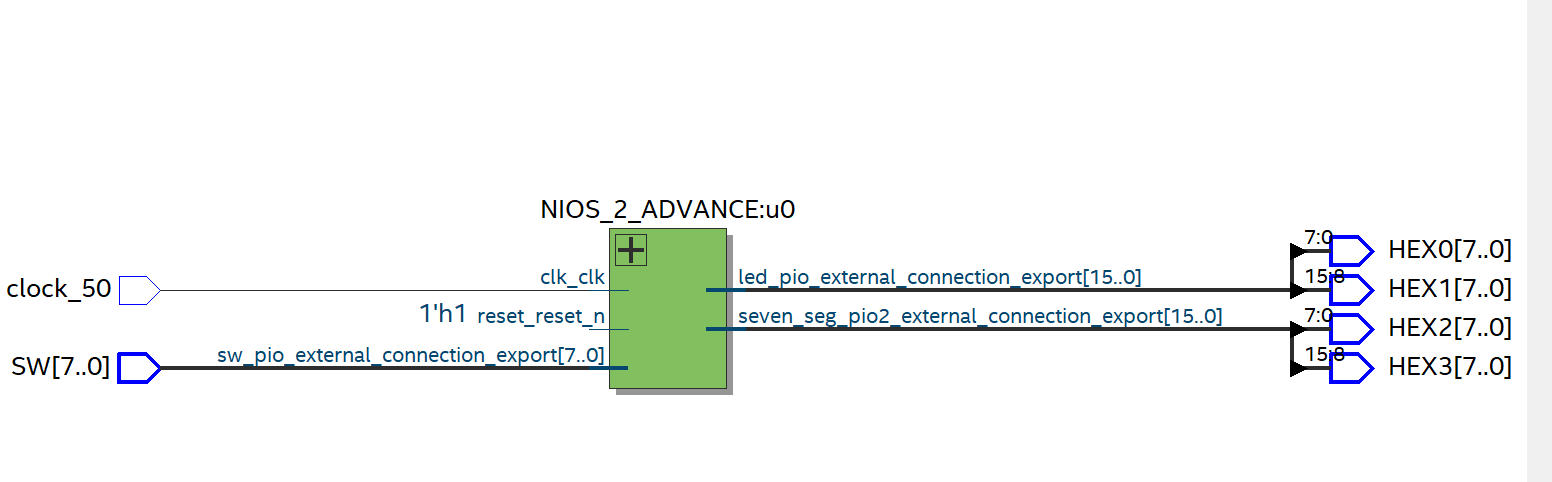
sw\_pio\_external\_connection\_export (7 downto 0)

seven\_seg\_pio2\_external\_connection\_export (15 downto 0)

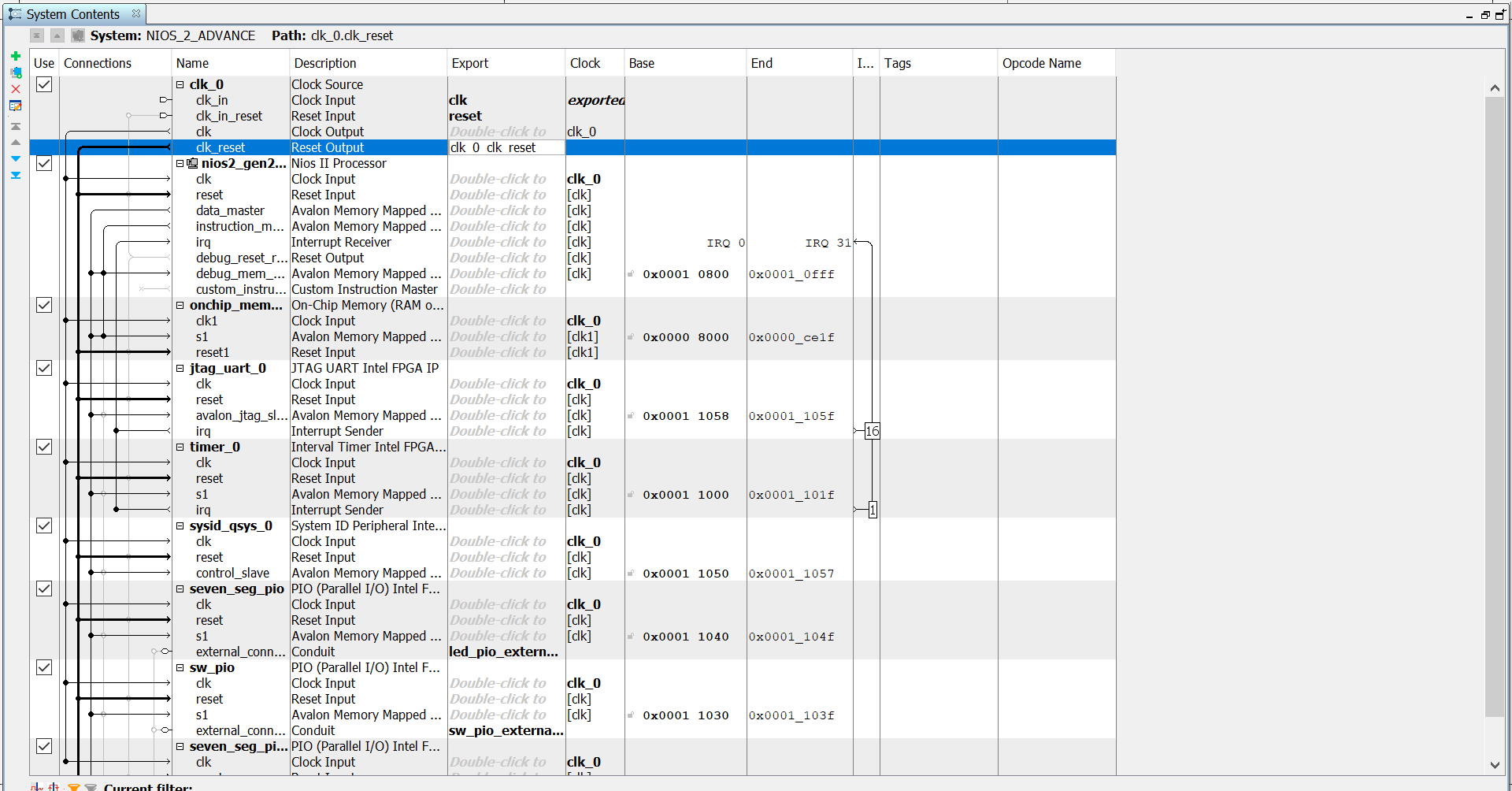
And connected them to by external connection by DE10-LITE board by my VHDL file named LAB4\_DE10LAB. Afterwards, using Nios II software tool build. I created code where it would store the value of SW and count value and store the first 4-bits into one variable and start the last 4-bits on another variable. Pass it through a function that convert the value into a 2-bit hex value which controls what signals stays on and off. For example, if the passed value is a 0 then in order to display 0 on my Hex display, I would need to output a 8-bit value that would be 0x40. This conversion is down for both variables and then values are added together and re-shifted back to display the correct values. For example, if the count value is 0x1F, x=1, y=F, convert x and y to 8-bit value. X’ = 0x79 and y’ = 0x0E. shift X’ to the left by 8 bits. Making X’ = 0x7900 then add y’ which means that the resulting value is 0x790E. Which will be the output of my HEX display. The same logic and design were implemented to the switches for this project.

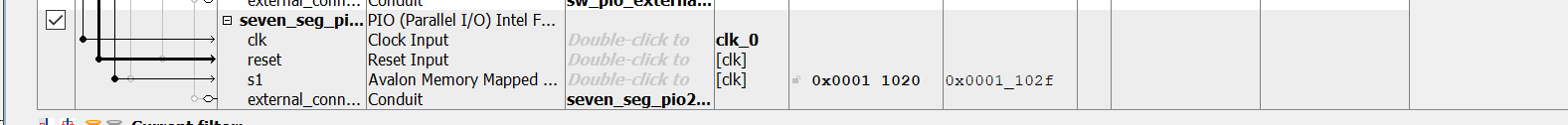
Conclusion: Overall, this lab was a success, the toughest part I had with this lab would be finding how to convert my counted value into values that can be displayed onto the DE10-LITE. At first, I was thinking of hardcoding this by doing it for all values of 0xFF. But this would be least efficient way and would waste to much time and memory. Instead, I decided to use the bit operators to shift my values in different direction where it would be the most useful for my project. In the end I created a function that takes care of this conversion and in my main function I just had two independents if statements in my main. Thinks I would have done differently would be differently would have to create on whole function that takes care of the shifting and conversion together instead of having three independent ones that do that.

RTL Viewer



Counter\_RTL\_Viewer





VHDL CODE

Nios2\_IO\_DE10\_.vhd

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

--

--NIOS2\_IO\_DE10\_VHDL

--

--CREATED: 10/1/2021

--BY: JORGE JURADO-GARCIA

--REV: 0

--

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

--

--ADVANCE NIOSE SYSTEM -WITH HEX0-3 WITH SW/PERIPHERALS

--

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

**LIBRARY** IEEE**;**

**USE** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**USE** IEEE**.**NUMERIC\_STD**.ALL;**

**ENTITY** LAB4\_DE10LAB **IS**

**PORT(**

clock\_50 **:** **IN** STD\_LOGIC**;**

SW **:** **IN** STD\_LOGIC\_VECTOR**(**7 **DOWNTO** 0**);**

HEX0**,**HEX1**,**HEX2**,**HEX3 **:** **OUT** STD\_LOGIC\_VECTOR**(**7 **DOWNTO** 0**)**

**);**

**END** **ENTITY;**

**ARCHITECTURE** behavioral **OF** LAB4\_DE10LAB **is**

**component** NIOS\_2\_ADVANCE **is**

**port** **(**

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

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

led\_pio\_external\_connection\_export **:** **out** std\_logic\_vector**(**15 **downto** 0**);**

sw\_pio\_external\_connection\_export **:** **in** std\_logic\_vector**(**7 **downto** 0**)** **:=** **(others** **=>** 'X'**);** -- export

seven\_seg\_pio2\_external\_connection\_export **:** **out** std\_logic\_vector**(**15 **downto** 0**)** -- export

**);**

**end** **component** NIOS\_2\_ADVANCE**;**

**BEGIN**

u0 **:** **component** NIOS\_2\_ADVANCE

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

clk\_clk **=>** clock\_50**,** --

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

led\_pio\_external\_connection\_export**(**7 **downto** 0**)** **=>** HEX0**(**7 **DOWNTO** 0**),** -- led\_pio\_external\_connection.export

led\_pio\_external\_connection\_export**(**15 **downto** 8**)** **=>** HEX1**(**7 **DOWNTO** 0**),** -- led\_pio\_external\_connection.export

sw\_pio\_external\_connection\_export **=>** SW**(**7 **DOWNTO** 0**),** -- sw\_pio\_external\_connection.export

seven\_seg\_pio2\_external\_connection\_export**(**7 **downto** 0**)** **=>** HEX2**(**7 **DOWNTO** 0**),** -- seven\_seg\_pio2\_external\_connection.export

seven\_seg\_pio2\_external\_connection\_export**(**15 **downto** 8**)** **=>** HEX3**(**7 **DOWNTO** 0**)**

**);**

**END** **ARCHITECTURE;**

C SOURCE CODE

/\*

\* counter.c

\*

\* Created on: Oct 1, 2021

\* Author: jurado-garciaj

\*/

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

#include "system.h"

#include <stdio.h>

#include <unistd.h>

int convert**(**alt\_u8 c**);**

int shift\_4bits**(**alt\_u8 b**);**

int clear\_MS4B**(**alt\_u8 a**);**

int main**(){**

printf**(**"My counter program!\n"**);**

alt\_u8 count **=** 0**;** //up to 0xff

alt\_u16 resultt **=** 0b0000000000000000**;**

alt\_u8 sw**;**

alt\_u16 output **=** 0b0000000000000000**;**

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

//output the count to hex0, hex1

IOWR\_ALTERA\_AVALON\_PIO\_DATA**(**SEVEN\_SEG\_PIO\_BASE**,** resultt**);**

//output the sw value to hex1, hex2

IOWR\_ALTERA\_AVALON\_PIO\_DATA**(**SEVEN\_SEG\_PIO2\_BASE**,** output**);**

//read for the switches

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

//place-holder value

alt\_u8 x**,**y**;**

**if(**sw **>=** 0**){**

printf**(**"switch value: %2X,"**,** sw**);**

x **=** shift\_4bits**(**sw**);** //make x equal to sw

x **=** convert**(**x**);** //convert to 7-seg value

y **=** clear\_MS4B**(**sw**);**

y **=** convert**(**y**);**

output **=** x**<<**8**;**

output **=** output **+** y**;**

printf**(**"converted switch value %2X, "**,** output**);**

**}** //end if

**if(**count **==** 0xFF**){**

count **=** 0x00**;**

**}**

x **=** shift\_4bits**(**count**);** //make x equal to sw

x **=** convert**(**x**);** //convert to 7-seg value

y **=** clear\_MS4B**(**count**);**

y **=** convert**(**y**);**

resultt **=** x**<<**8**;**

resultt **=** resultt **+** y**;**

usleep**(**600000**);** //sleep for 3 second

count**++;**

**}**//end while loop

**return** 0**;**

**}**//end main

int shift\_4bits**(**alt\_u8 b**){**

alt\_u8 x **=** b**;**

x **=** x**>>**4**;**

**return** x**;**

**}**//end function

int clear\_MS4B**(**alt\_u8 a**){**

alt\_u8 x **=** a**;**

x **&=** 0x0F**;**

**return** x**;**

**}**//end function

/\*function returning the count value from decimal to hex value \*/

int convert**(**alt\_u8 c**){**

alt\_u8 result**;**

int LEDarray**[]** **=** **{**0x40**,**0x79**,**0x24**,**0x30**,**0x19**,**0x12**,**0x02**,**0xD8**,**0x00**,**0x18**,**0x08**,**0x83**,**0x46**,**0xA1**,**0x06**,**0x0E**};**

**switch(**c**){**

**case** 0**:**

result **=** **(**LEDarray**[**0**]);**

**break;**

**case** 1**:**

result **=** **(**LEDarray**[**1**]);**

**break;**

**case** 2**:**

result **=** **(**LEDarray**[**2**]);**

**break;**

**case** 3**:**

result **=** **(**LEDarray**[**3**]);**

**break;**

**case** 4**:**

result **=** LEDarray**[**4**];**

**break;**

**case** 5**:**

result **=** **(**LEDarray**[**5**]);**

**break;**

**case** 6**:**

result **=** **(**LEDarray**[**6**]);**

**break;**

**case** 7**:**

result **=** **(**LEDarray**[**7**]);**

**break;**

**case** 8**:**

result **=** **(**LEDarray**[**8**]);**

**break;**

**case** 9**:**

result **=** **(**LEDarray**[**9**]);**

**break;**

**case** 10**:**

result **=** **(**LEDarray**[**10**]);**

**break;**

**case** 11**:**

result **=** **(**LEDarray**[**11**]);**

**break;**

**case** 12**:**

result **=** **(**LEDarray**[**12**]);**

**break;**

**case** 13**:**

result **=** **(**LEDarray**[**13**]);**

**break;**

**case** 14**:**

result **=** **(**LEDarray**[**14**]);**

**break;**

**case** 15**:**

result **=** **(**LEDarray**[**15**]);**

**break;**

**}**//end switch statement

**return** result**;**

**}**//end function