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 1: “Counter and Register Project"

Jorge J. Jurado-Garcia

Date: 09/12/2020

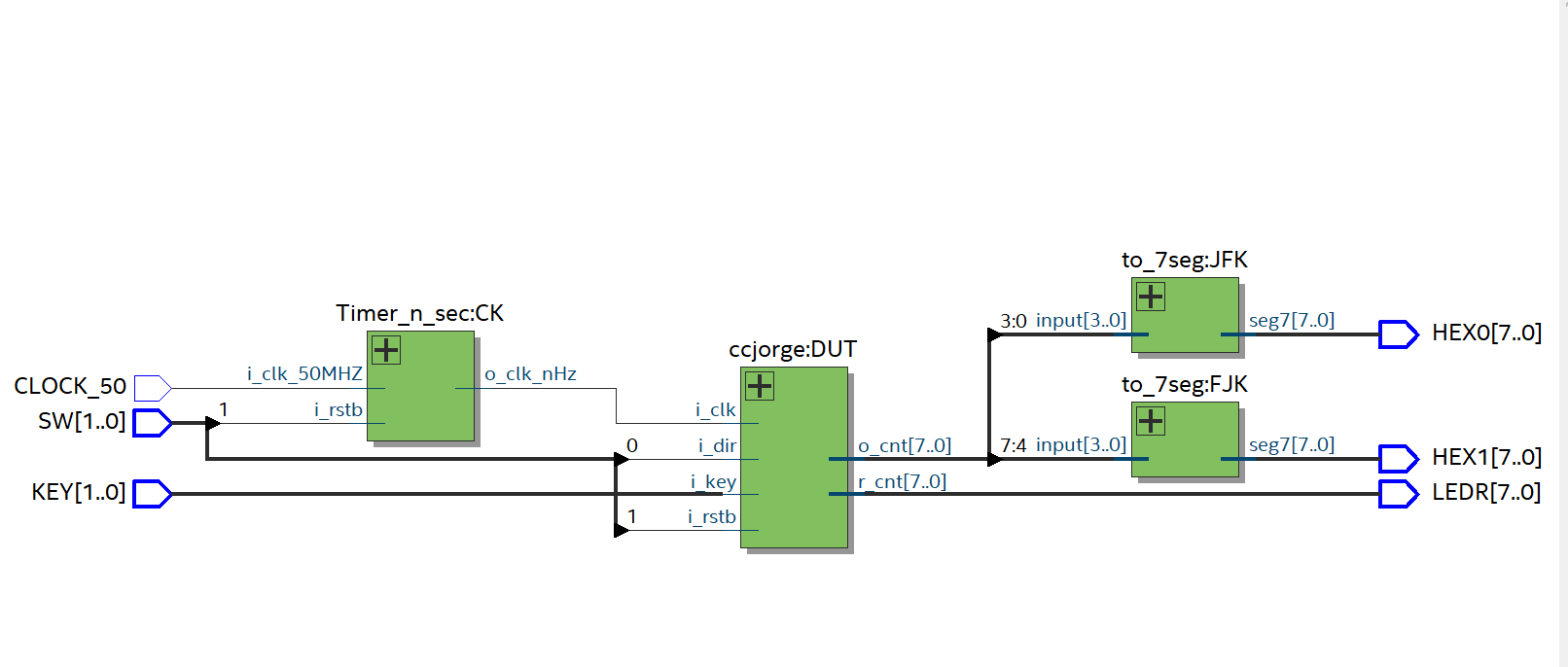
Level Attempted: Advanced

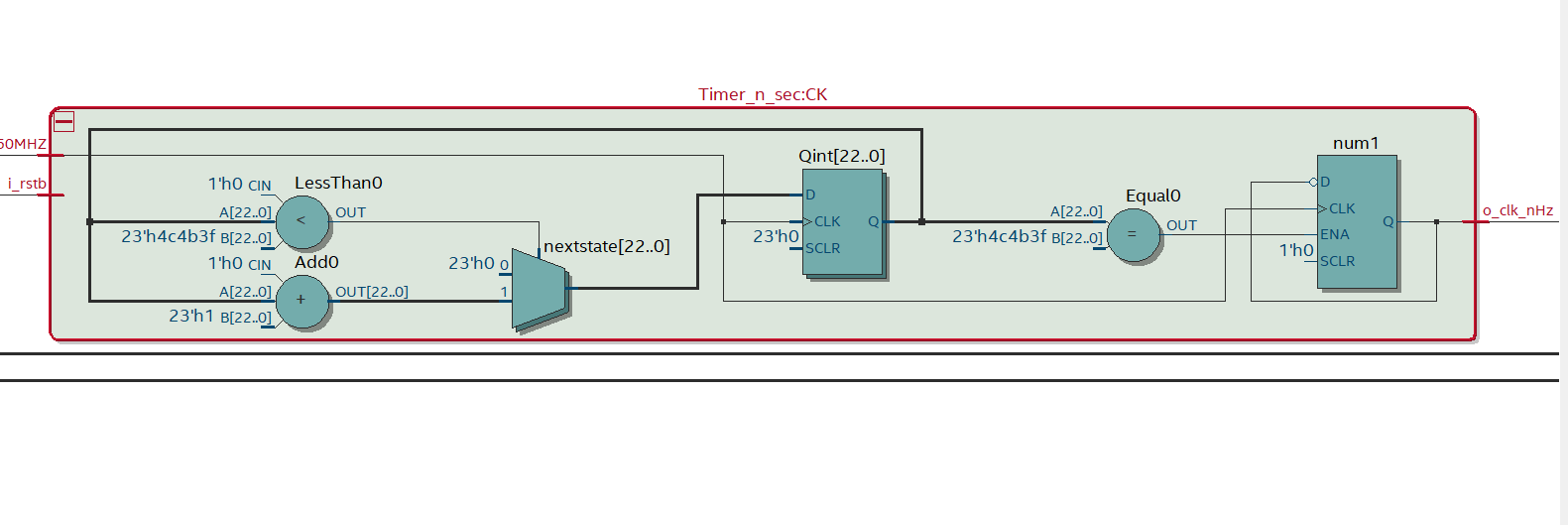
Objective: The objective of this lab was review VHDL Programming by creating a generic counter from two parameters. Variable M for to specify the number of seconds between each increment of the counter and variable N the to specify the maximum count value (in ranges of 1 to 255). This counter should be able to store a value and display it on the LEDR when KEY[1] is pressed and should have wrapping characteristics.

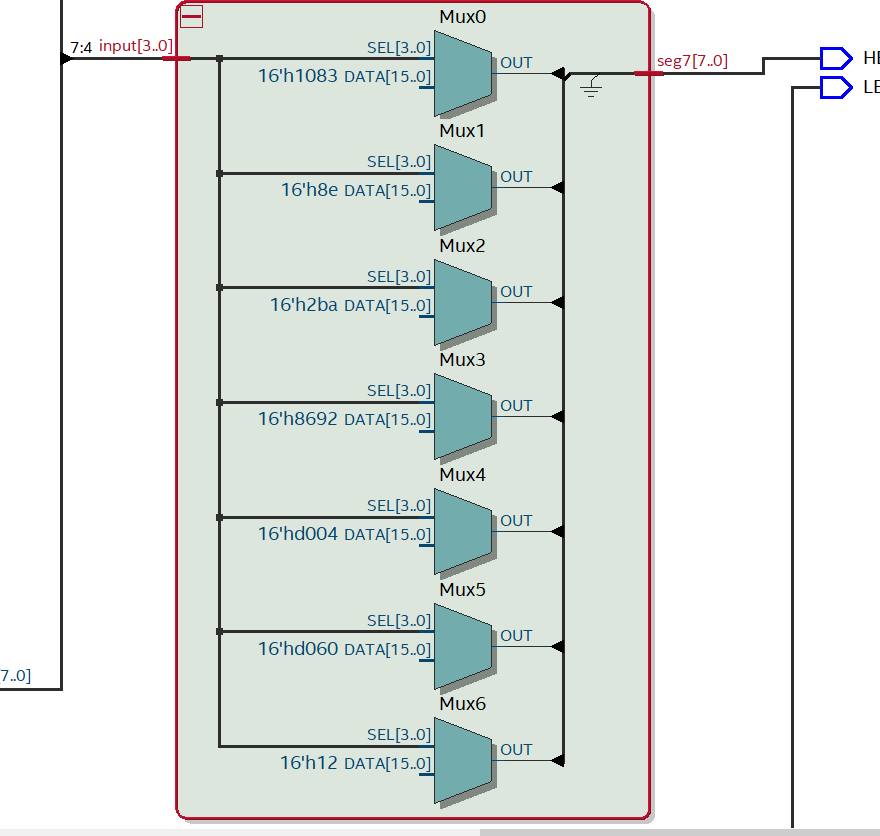
Description: In this lab I created 3 components for my final file. The first component is the “ccjorge” which is the VHDL file for my counter with implementations of the register. Second component is a timer named “TIMER\_N\_Secs” which used the 50MHZ crystal clock on the DE10 Lite and created a impulse every 1 seconds. Lastly, my last component is a conversion from binary to 7-seg conversion component in order to show the values in Hex [1:0].

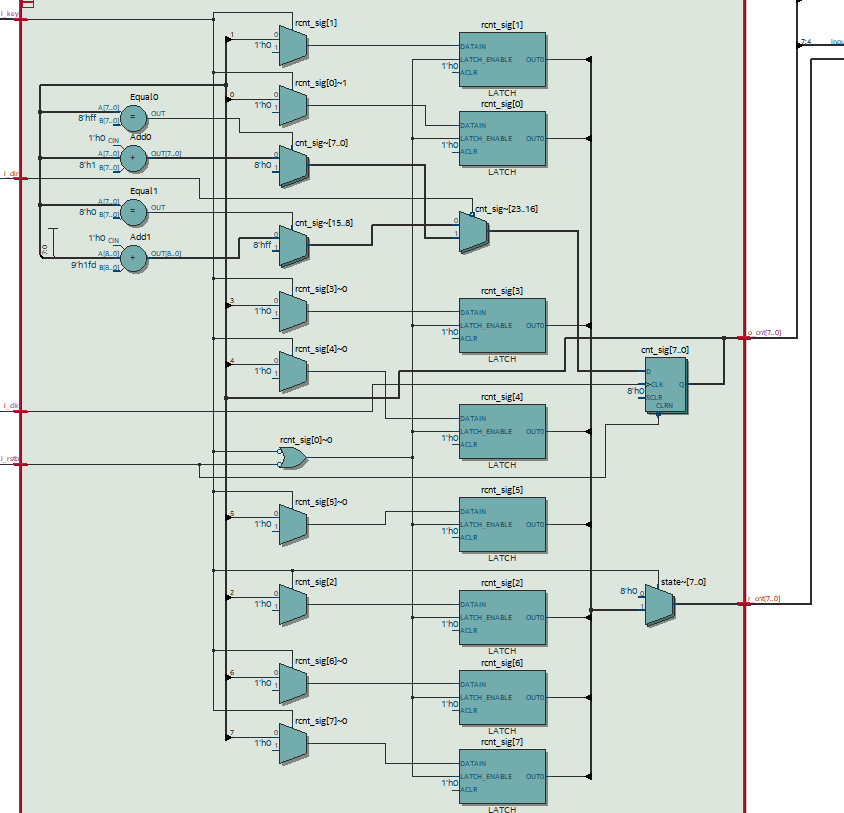
Conclusion: The results of this lab were better than what I was expected. I was able to implement all the required objectives for the advance part. The biggest difficulty I encountered when doing this lab is trying to implement all the components together. I was overthinking the storage component with my counter and was trying to use structural model for everything. Which would just make the project more complex and difficult to debug. What I did learned a lot is the use of sequential logic and how to implement in the language of VHDL. Things I would have done differently, would be the timer component. Ideally, this should have been implemented together with my counter component in order to reduce possible errors when trying to connect components together structural and save a lot of more time so. But however, the benefit of this error is that I can use this component in future projects that would be needed. Overall, this project was a success.

RTL Viewer









VHDL CODE

CCJORGE

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

--ccjorge.vhdl

--

--created 9/10/21

--Jorge Jurado-Garcia

--rev 0

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

--

--Inputs: rstb, clk, dir

--outputs: cnt

--

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

--

--counts up when dir = 0

--counts down when dir = 1

--

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

**library** ieee**;**

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

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

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

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

---ENtity

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

--created a 8 bit generic counter with clk, rstb, and direction as inputs with a 4 bit output

**entity** ccjorge **is**

**generic(**

n**:** natural **:=** 8 --this parameters controls the parameter of are counter

**);**

**port** **(**

i\_clk**:** **in** std\_logic**;**

i\_rstb**:** **in** std\_logic**;**

i\_dir**:** **in** std\_logic**;**

i\_key**:** **in** std\_logic**;**

o\_cnt**:** **out** std\_logic\_vector**(**n**-**1 **downto** 0**);**

r\_cnt**:** **out** std\_logic\_vector**(**n**-**1 **downto** 0**)**

**);**

**end** **entity;**

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

--architecture

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

**architecture** behavioral **of** ccjorge **is**

--

--internal signals, signals are unsigned in order to start from 0 to 2^(n-1)

--

**signal** cnt\_sig**:** unsigned**(**n**-**1 **downto** 0**);**

**signal** rcnt\_sig**:** unsigned**(**n**-**1 **downto** 0**);**

**signal** state**:** unsigned**(**n**-**1 **downto** 0**);**

**begin**

**process(**i\_clk**,** i\_rstb**)** --sequential processing

**begin**

--

--reset

--

--when rstb is 0 signal is 0

**if(**i\_rstb **=** '0'**)** **then**

cnt\_sig **<=** **(others** **=>** '0'**);** --make sure all signals are zero when rstb is zzero

rcnt\_sig **<=** **(others** **=>** '0'**);**

state **<=** **(others** **=>** '0'**);**

--rising clk edge

**elsif(rising\_edge(**i\_clk**))** **then**

**if(**i\_dir **=** '0'**)** **then**

cnt\_sig **<=** cnt\_sig **+** 1**;** --adding one

**if(**cnt\_sig **=** "11111111"**)** **then** --0xFF

cnt\_sig **<=** "00000000"**;**

**end** **if;**--end if of wrapping up

**else**

cnt\_sig **<=** cnt\_sig **-** 1**;** --subtracting one

**if(**cnt\_sig **=** "00000000"**)** **then**

cnt\_sig **<=** "11111111"**;**

**end** **if;** --end if of wrapping down

**end** **if;** --end of if statement for i\_dir

--

**end** **if;** --end of if statement for i-rstb

**if(**i\_key **=** '1' **)** **then** --button logic

state **<=** rcnt\_sig**;** --when state rcnt\_sig

**else**

rcnt\_sig **<=** cnt\_sig**;** --will start at zero but rcnt\_sig wil constantly

--change then when button is not pressed.

state **<=** **(others** **=>** '0'**);** --state will be shown as zero on LEDR

**end** **if;** --end of if statement for i-rstb

**end** **process;**

--

--Output logic

--

o\_cnt **<=** std\_logic\_vector**(**cnt\_sig**);**

r\_cnt **<=** std\_logic\_vector**(**state**);**

**end** behavioral**;** --end of counter architerial

TIMER\_N\_SEC.

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

--

--Timer\_n\_sec.vhdl

--

--created 9/11/21

--jorge jurado-garcia

--

--rev 0

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

--

-- Timer\_n\_sec for 1 seconds using a clock cycles

--

--assummes a 50 MHZ external clock

--

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

--

--Inputs: rstb, clock\_50MHZ

--Outputs: clk\_out

--

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

**library** ieee**;**

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

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

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

**entity** Timer\_n\_sec **is**

**generic(**

k**:** natural **:=** 1

**);**

**port(**

i\_clk\_50MHZ**:** **in** std\_logic**;**

i\_rstb**:** **in** std\_logic**;**

o\_clk\_nHz**:** **out** std\_logic

**);**

**end** **entity;**

**architecture** behavioral **of** Timer\_n\_sec **is**

--internal signals

--

**SIGNAL** nextstate**,** Qint**:** INTEGER **RANGE** 0 **TO** **(**50000000**\***k**):=** 0**;**

**SIGNAL** en**,** clk**,** num1**,** nextnum1 **:** STD\_LOGIC**;**

**begin**

o\_clk\_nHz **<=** num1**;**

clk **<=** i\_clk\_50MHZ**;**

-- generate clock enable - active for one CLOCK\_50 pulse every 100 ms

en **<=** '1' **WHEN** Qint **=** **(**49999999**\***k**)** **ELSE**

'0'**;**

nextstate **<=** Qint **+** 1 **WHEN** Qint **<** **(**49999999**\***k**)** **ELSE**

0**;**

**process(**clk**)**

**begin**

--

**if(rising\_edge(**clk**)** **)** **then**

Qint **<=** nextstate**;**

**end** **if;**

**end** **process;**

--end of gneratate clock enabled

nextnum1 **<=** num1 **WHEN** en **=** '0'

**else**

**not** num1**;**

**process** **(**clk**)**

**BEGIN**

**IF** **(rising\_edge(**clk**))** **THEN** -- rising clock edge

num1 **<=** nextnum1**;**

**END** **IF;**

**END** **PROCESS;**

--output logic

**end** behavioral**;**

TO\_7Seg

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

--

-- to\_7seg.vhdl

--

--created 9/11/21

--jorge jurado-garcia

--

--rev 0

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

-- takes a for bit bit value and then translates into a seven segment display

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

--Inputs: input

--Outputs: seg7

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

**library** ieee**;**

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

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

**entity** to\_7seg **is**

**Port** **(** input **:** **in** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**

seg7 **:** **out** STD\_LOGIC\_VECTOR **(**7 **downto** 0**)**

**);**

**end** to\_7seg**;**

**architecture** Behavioral **of** to\_7seg **is**

**begin**

--'input' corresponds to MSB of seg7 and 'g' corresponds to LSB of seg7.

-- when a bit is low the led on the seg7 will be on else '0'

**process** **(**input**)**

**BEGIN**

**case** input **is**

**when** "0000"**=>** seg7 **<=**"01000000"**;** -- '0'

**when** "0001"**=>** seg7 **<=**"01111001"**;** -- '1'

**when** "0010"**=>** seg7 **<=**"00100100"**;** -- '2'

**when** "0011"**=>** seg7 **<=**"00110000"**;** -- '3'

**when** "0100"**=>** seg7 **<=**"00011001"**;** -- '4'

**when** "0101"**=>** seg7 **<=**"00010010"**;** -- '5'

**when** "0110"**=>** seg7 **<=**"00000010"**;** -- '6'

**when** "0111"**=>** seg7 **<=**"01111000"**;** -- '7'

**when** "1000"**=>** seg7 **<=**"00000000"**;** -- '8'

**when** "1001"**=>** seg7 **<=**"00011000"**;** -- '9'

**when** "1010"**=>** seg7 **<=**"00001000"**;** -- 'A'

**when** "1011"**=>** seg7 **<=**"00000000"**;** -- 'b'

**when** "1100"**=>** seg7 **<=**"01000110"**;** -- 'C'

**when** "1101"**=>** seg7 **<=**"00000000"**;** -- 'D'

**when** "1110"**=>** seg7 **<=**"00000110"**;** -- 'E'

**when** "1111"**=>** seg7 **<=**"00001110"**;** -- 'F'

**when** **others** **=>** **NULL;**

**end** **case;**

**end** **process;**

**end** Behavioral**;**

COUNTER\_DE10\_N\_BITS

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

--Counter\_DE10\_n\_bits

--created 9/11/21

--Jorge Jurado-Garcia

--rev 0

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

--Inputs: rstb, clk, dir

--outputs: cnt

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

--counts up when dir = 0

--counts down when dir = 1

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

**library** ieee**;**

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

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

**entity** Counter\_DE10\_n\_bits **is**

**port(**

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

SW**:** **in** std\_logic\_vector**(**1 **downto** 0**);**

HEX0**:** **out** std\_logic\_vector**(**7 **downto** 0**);**

HEX1**:** **out** std\_logic\_vector**(**7 **downto** 0**);**

KEY**:** **in** std\_Logic\_vector**(**1 **downto** 0**);** --key 1

LEDR**:** **out** std\_logic\_vector**(**7 **downto** 0**)** **);**

**end** **entity** Counter\_DE10\_n\_bits**;**

**Architecture** behavioral **of** Counter\_DE10\_n\_bits **is**

---signals that will be used

**signal** CLK\_SIG**:** std\_logic**;** --intermediate value

**signal** INPUT\_SIG**:** std\_logic\_vector**(**3 **downto** 0**);**

**signal** INPUT\_SIG2**:** std\_logic\_vector**(**3 **downto** 0**);**

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

--component declarations

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

**component** Timer\_n\_sec **is**

**generic(**

k**:** natural **:=** **(**1**)**

**);**

**port(**

i\_clk\_50MHZ**:** **in** std\_logic**;**

i\_rstb**:** **in** std\_logic**;**

o\_clk\_nHz**:** **out** std\_logic

**);**

**end** **component;**

**component** ccjorge **is**

**generic(**

n**:** natural **:=** 8

**);**

**port** **(**

i\_clk**:** **in** std\_logic**;**

i\_rstb**:** **in** std\_logic**;**

i\_dir**:** **in** std\_logic**;**

i\_key**:** **in** std\_logic**;**

o\_cnt**:** **out** std\_logic\_vector**(**n**-**1 **downto** 0**);**

r\_cnt**:** **out** std\_logic\_vector**(**n**-**1 **downto** 0**)**

**);**

**end** **component;**

**component** to\_7seg **is**

**Port** **(** input **:** **in** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**

seg7 **:** **out** STD\_LOGIC\_VECTOR **(**7 **downto** 0**)**

**);**

**end** **component;**

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

**begin**

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

--Device under test (DUT)

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

CK**:** Timer\_n\_sec

**port** **map(**

i\_clk\_50MHZ **=>** CLOCK\_50**,**

i\_rstb **=>** SW**(**1**),**

o\_clk\_nHz **=>** CLK\_SIG

**);**

DUT**:** ccjorge

**port** **map(**

i\_clk **=>** CLK\_SIG**,**

i\_rstb **=>** SW**(**1**),**

i\_dir **=>** SW**(**0**),**

i\_key **=>** KEY**(**1**),**

o\_cnt**(**3 **downto** 0**)** **=>** INPUT\_SIG**,**

o\_cnt**(**7 **downto** 4**)** **=>** INPUT\_SIG2**,**

r\_cnt **=>** LEDR**(**7 **downto** 0**)**

**);**

JFK**:** to\_7seg

**port** **map(**

input **=>** INPUT\_SIG**,**

seg7 **=>** HEX0**(**7 **downto** 0**)**

**);**

FJK**:** to\_7seg

**port** **map(**

input **=>** INPUT\_SIG2**,**

seg7 **=>** HEX1**(**7 **downto** 0**)**

**);**

**end** **architecture;**