**Lab 6: Finite State Machines and Controllers** **Name:** Luke Bachman

EGCP 281: Designing with VHDL **Date:** Friday, December 20, 2019

***You may discuss this assignment with others, but this assignment must be completed individually.***

*Please recall that academic dishonesty will not be tolerated. By submitting this assignment, you understand penalties will be assessed if you submit work for credit that is not your own.*

**Due Date: Upload this work sheet to Titanium by Friday, December 20, 2019 by 11:59 PM**

**No late submissions will be accepted**

**Grade** **Grade Item**

/3 Task 1: Complete Problem 3 of Project 10 (a single-digit stopwatch)

*/3 Demonstration of Task 1*

/3 Lab Recap Questions

**/6 Total Score (of deliverables and questions)**

**/3 Lab Demonstrations**

**Once you have completed the lab (or once you are done for the day), and after you close your project, remember to copy your project folder to a flash drive, your Dropbox or Google Drive, or your email in order to keep a copy of your files.**

**Recommended Reading:** *Digilent Real Digital Module 10*

**Objectives:**

* Gain experience designing, specifying, and implementing a finite state machine-based controller
* Create and use counters and clock dividers in the design of a digital system

**Lab Description:**

Follow the instructions in the lab task below to complete **Problem 3 of Project 10** from the [Digilent Real Digital website](https://reference.digilentinc.com/textbooks:real_digital). This design problem involves finite state machine design and interfacing with the seven-segment display.

First start by analyzing the block diagram for **Problem 3 of Project 10**. Then, use VHDL to design each of the system components. You will need to use four separate design modules and instantiate each of these within a fifth design module for the overall system.

**Lab Tasks:**

1. **Complete Problem 3 of Project 10 (a single-digit stopwatch)**
   1. Pay particular attention to the block diagram displayed for this problem. Create each of the four components to this system:
      1. **Seven-segment decoder**: You will be able to reuse your design from Lab 4
      2. **Clock divider**: You will be able to reuse your design from Lab 5  
         *Note: The stopwatch circuit will increment the digit once every second; this is the same output clock rate that you used in Lab 5.*
      3. **4-bit counter**: This is a new circuit. I recommend looking at the behavioral binary counter described in “Binary counters in VHDL” from **Digilent Real Digital Module 10**.
      4. **Controller**: This is a new circuit. This is the main component you will design using a finite state machine. You will want to carefully plan your finite state machine **before** you write the VHDL code for this controller.
   2. Use VHDL test benches to verify the correct operation of your 4-bit counter, controller, and overall system (I suggest you use a small clock divider value for simulating so you do not have to simulate for a long duration).
   3. [1 pt] Place your four-bit counter’s VHDL design module code here (copy-paste as text, not an image):

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

-- Company:

-- Engineer:

--

-- Create Date: 12/16/2019 10:06:51 PM

-- Design Name:

-- Module Name: counter - Behavioral

-- Project Name:

-- Target Devices:

-- Tool Versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.all;

use IEEE.STD\_LOGIC\_UNSIGNED.all;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx leaf cells in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

--Binary Counter Example was sourced from

--https://reference.digilentinc.com/\_media/textbooks/realdigital\_module\_10.pdf

entity counter is

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

B : inout STD\_LOGIC\_VECTOR (3 downto 0));

end counter;

architecture Behavioral of counter is

begin

process (clk, rst)

begin

if rst = '1' then B <= "0000";

elsif (rising\_edge(clk)) then

B <= B + 1;

end if;

end process;

end Behavioral;

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

-- Company:

-- Engineer:

--

-- Create Date: 12/16/2019 10:06:51 PM

-- Design Name:

-- Module Name: counter - Behavioral

-- Project Name:

-- Target Devices:

-- Tool Versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.all;

use IEEE.STD\_LOGIC\_UNSIGNED.all;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx leaf cells in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

--Binary Counter Example was sourced from

--https://reference.digilentinc.com/\_media/textbooks/realdigital\_module\_10.pdf

entity counter is

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

cen : in STD\_LOGIC; --Clock Enable

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end counter;

architecture Behavioral of counter is

signal count : STD\_LOGIC\_VECTOR(3 downto 0);

begin

process (clk, rst)

begin

if rst = '1' then count <= "0000";

if(cen = '1') then

if (rising\_edge(clk)) then

count <= count + 1;

end if;

end if;

end if;

end process;

b <= count;

end Behavioral;

* 1. [1 pt] Place your controller’s (finite state machine code) VHDL design module code here (copy-paste as text, not an image):

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

-- Company:

-- Engineer:

--

-- Create Date: 12/16/2019 10:23:22 PM

-- Design Name:

-- Module Name: controller - Behavioral

-- Project Name:

-- Target Devices:

-- Tool Versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx leaf cells in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity controller is

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

start : in STD\_LOGIC;

stop : in STD\_LOGIC;

inc : in STD\_LOGIC;

run : out STD\_LOGIC\_VECTOR(3 downto 0));

end controller;

architecture Behavioral of controller is

--define the states using a type statement

type myStates\_t is (

st\_stop,

st\_count,

st\_lap

);

--Create a signal that uses the different states

signal state : mystates\_t;

begin

process( clk, rst)

begin

if(rst = '1') then

state <= st\_stop;

run <= "0000";

elsif(rising\_edge(clk)) then

case state is

when st\_stop =>

if(start = '1') then

state <= st\_count;

else

state <= st\_stop;

end if;

run <= "0000";

when st\_lap =>

if(inc = '0') then

state <= st\_count;

else

state <= st\_lap;

end if;

run <= "1111";

when st\_count =>

if( stop = '1') then

state <= st\_stop;

elsif( inc = '1') then

state <= st\_lap;

else

state <= st\_count;

end if;

run <= "1111";

when others =>

state <= st\_stop;

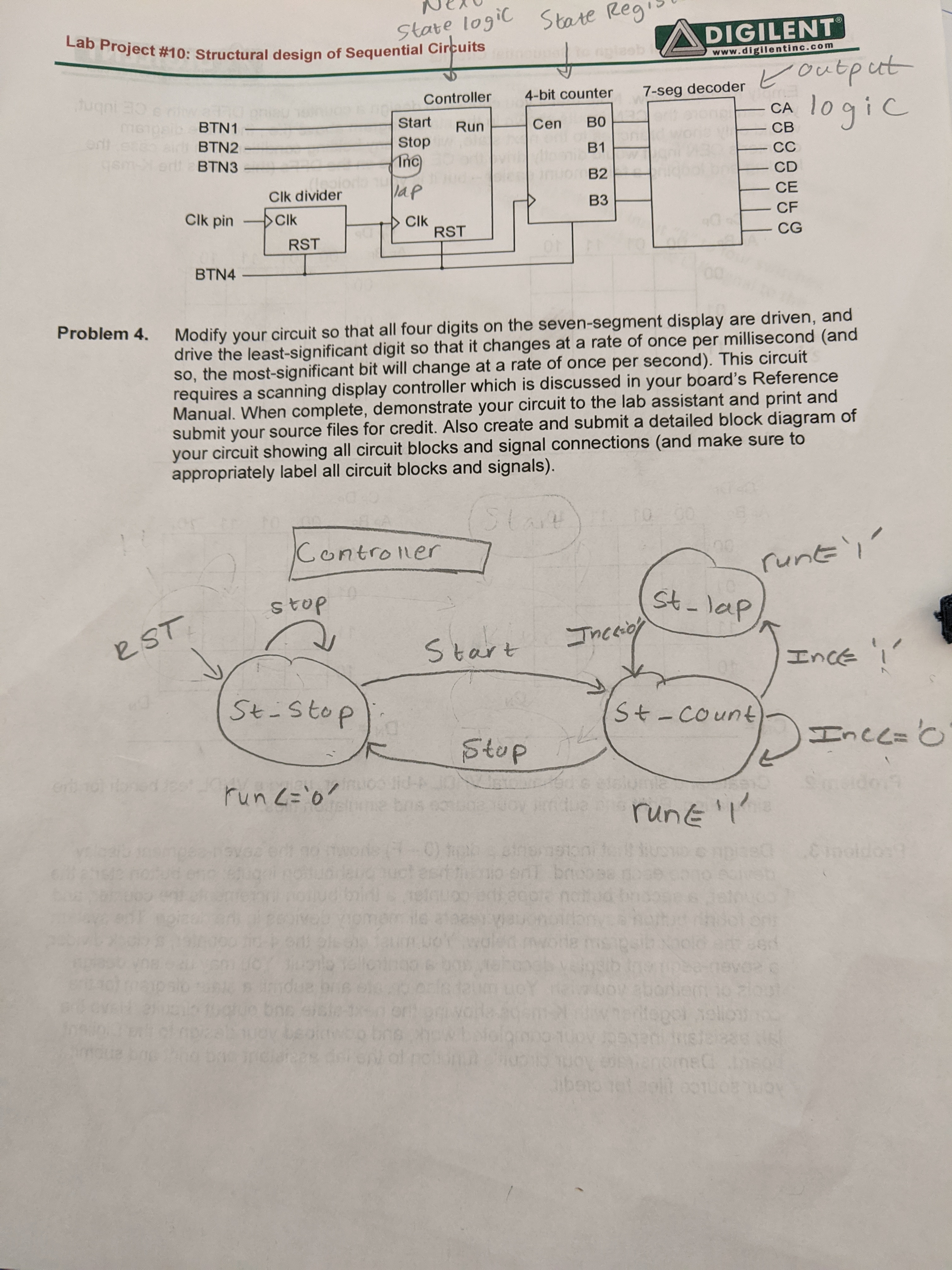
run <= "0000";

end case;

end if; --rising-edge of CLK

end process;

end Behavioral;



* 1. [1 pt] Place your overall system’s (the entire single-digit stopwatch) VHDL design module code here (copy-paste as text, not an image):

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

-- Company:

-- Engineer:

--

-- Create Date: 12/17/2019 10:48:00 PM

-- Design Name:

-- Module Name: stopwatch - Behavioral

-- Project Name:

-- Target Devices:

-- Tool Versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

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

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

--use IEEE.NUMERIC\_STD.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx leaf cells in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity stopwatch is

Port ( watch\_start : in STD\_LOGIC;

watch\_stop : in STD\_LOGIC;

watch\_inc : in STD\_LOGIC;

watch\_clk : in STD\_LOGIC;

watch\_rst : in STD\_LOGIC;

watch\_output : out STD\_LOGIC\_VECTOR (6 downto 0));

end stopwatch;

architecture Behavioral of stopwatch is

component clk\_divider

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

clkout : out STD\_LOGIC);

end component;

component controller

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

start : in STD\_LOGIC;

stop : in STD\_LOGIC;

inc : in STD\_LOGIC;

run : out STD\_LOGIC\_VECTOR (3 downto 0));

end component;

component counter

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

cen : in STD\_LOGIC;

b : out STD\_LOGIC\_VECTOR (3 downto 0));

end component;

component decoder

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

Output : out STD\_LOGIC\_VECTOR (6 downto 0));

end component;

signal output\_signal\_controller: STD\_LOGIC\_VECTOR(3 downto 0);

signal output\_signal\_counter: STD\_LOGIC\_VECTOR(3 downto 0);

signal output\_signal\_clkdiv: STD\_LOGIC;

signal input\_signal\_decoder: STD\_LOGIC\_VECTOR(3 downto 0);

begin

module\_a\_inst: clk\_divider

port map(

clk => watch\_clk,

rst => watch\_rst,

clkout => output\_signal\_clkdiv

);

module\_b\_inst: controller

port map(

start => watch\_start,

stop => watch\_stop,

inc => watch\_inc,

clk => output\_signal\_clkdiv,

rst => watch\_rst,

run => output\_signal\_controller

);

module\_c\_inst: counter

port map(

cen => output\_signal\_controller,

clk => output\_signal\_clkdiv,

rst => watch\_rst,

b => input\_signal\_decoder

);

module\_d\_inst: decoder

port map(

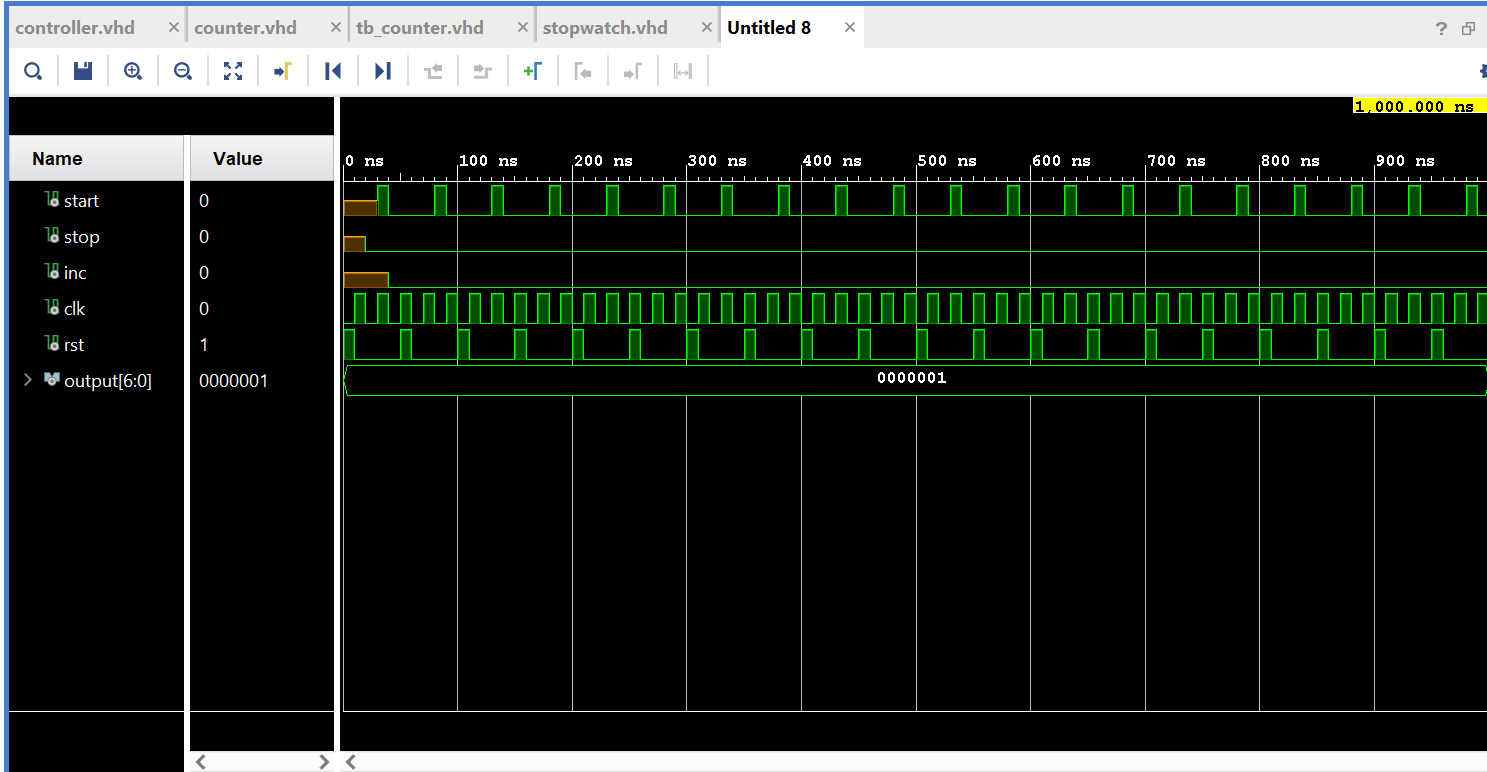
Input => input\_signal\_decoder,

Output => watch\_output

);

end Behavioral;

* 1. Now implement this system on the FPGA board. Connect the Start, Stop, Increment, and Reset inputs to four buttons, the CLK signal to the board’s clock, and the seven-segment display output to the seven-segment display.
  2. [3 pts] Ask the instructor to check the FPGA board implementation of your system. If you are unable to successfully demonstrate this by the lab deadline, then place your (1) simulation screenshots of your overall system and write (2) a description of any problems/issues with your design here:



I was unable to get the overall stopwatch output to work. I was however able to get my individual components working. I adjusted and fixed my clock. I tried my very best on this lab, but I couldn’t figure it out.

Now complete the lab recap questions below and submit the worksheet on Titanium by the due date.

**Lab Recap Questions:**

Type your answers to these questions once you finish the lab (do not print and write these):

1. [1 pt] Describe the design of your controller circuit for the single-digit stopwatch of lab task 1. Why did you design your state machine the way you did? Please be specific.

For my design, I created 3 states, had one 4 bit output to the counter, and 5 inputs to the controller. The 3 states are st\_stop, st\_count, and st\_lap. I used start, stop, reset, and inc, to go in between these states. So if you hit the start button, the state will go to count, if you hit the stop button, the state will go to stop, and if you hit the inc button, the state will go to lap.

1. [1 pt] Summarize your design process for creating the 4-bit counter:

For my design process, I started with the inputs and decided that I needed a clk, rst, and another 4 bit input and output. I then created a process statement and passed it the clk and rst into the sensitivity list. Then I defined what rst would do and that our 4bit input/output would increment on every rising edge of the clock.

1. [1 pt] Did you make changes to your design of the single-digit stopwatch during the course of this lab? Why (or why not)? Please explain.

I made a couple of changes to the design of my stopwatch during this lab. I changed the states a few times and the buttons that triggered the states because I redesigned my state diagram to be more readable.

1. List the references you used for this lab assignment (e.g. sources/websites used or students with whom you discussed this assignment)

The only sources that I used for this lab were the lecture notes and Digilent Real Digital

1. Do you have any comments or suggestions for this lab exercise?

The only suggestion I have to make this lab exercise a little bit better is to include a little bit more information about the functions of the finite state machine.