CPE 487: Digital System Design

HW 3

Due: 2/16/18

1. (10 points): Which of the following identifiers are not legal basic identifiers? Why?

Cntl7, y[3], VHDL, A_1_2_3, begin5, test_, port, max_value, 17years, init___guess, iftrue

2. Consider the following concurrent signal assignment statements:

A <= B **or** C **after** 9 ns;
D <= A **nand** mux **after** 4 ns;

Does the order of these statements matter? Why? How does this differ from assignment statements in conventional procedural programming languages? (5 points)

Suppose initially (at t=0ns), B='1', C='0' and mux='1'. What will be the values of A and D at time t=20ns? At time t=100ns, an event on B causes it to go from '1' to '0'. Describe the sequence of events (signals, values and times) that will follow from this transition on B. (5 points)

What if there were no explicit delay clause in both of the above statements (in other words, the delay of each gate was zero). What would the event sequence (signals, values and times) look like now? Would you get a different logical result? (5 points)

3. Is it OK to write?

X <= s1 **AND** S2 **AND** s3;

How about?

X<= s1 **NAND** S2 **NAND** s3;

What is different about these two statements (apart from the fact that one uses AND while the other uses NAND)? (5 points)

What happens when you enter the NAND statement into the simulator? What result do you get? (5 points)

How would you code a 3-input NAND gate (i.e. a gate whose output is '0' only when all of its inputs are '1')? (5 points)

4. A priority encoder is a device that takes a 2^n-bit input and encodes it to produce an n-bit result. If exactly one input bit is set, the output will be the bit position of that input bit. If more than one input bit is set, the encoder will prioritize the set input bits to determine the correct output.

The following code is a dataflow model of a 4-to-2 bit priority encoder using conditional signal assignment in which priority is given to the lowest numbered input. Enter this code into the simulator and test it with all possible input combinations. Make sure that your test-bench tests the cases when multiple inputs are set and also the case when none of the inputs are set. Show your test-bench code and the graphical simulation results (cut & paste screenshot of waveform outputs into your homework solution). (5 points)

```
library IEEE;
use IEEE.std_logic_1164.all;
entity pr_encoder is
port(S0, S1, S2, S3: in std_logic;
        Z:out std_logic_vector (1 downto 0));
end entity pr_encoder;

architecture dataflow of  pr_encoder is
begin
Z<=   “00” after 5 ns when S0=’1’ else
        “01” after 5 ns when S1=’1’ else
        “10” after 5 ns when S2=’1’ else
        “11” after 5 ns when S3=’1’ else
        “00” after 5 ns;
end architecture dataflow;
```

Rewrite the model using if-then-else statements within a process. Change the priority order to 1, 3, 2, and 0 with 1 being the highest priority input. Re-run the simulation. Show your code and the graphical simulation results. (10 points)

Delete (or comment out) the “default else clause” (that covers the case when none of the inputs are set). How does this change the result? Why? (5 points)