The University of Alabama in Huntsville
ECE Department
CPE 526 01
Midterm Exam
March 3, 2020

Name: ______________________________

1. (20 points)
a. (12 points) Write a VHDL function that has inputs **`vect`** of type **`std_logic_vecto`**r, **`sub`** of type **`std_logic_vector`**, and **`ins_pos`** of type natural. You may assume that **`vect`** has the form **`std_logic_vector(vect'length-1 downto 0)`** and **`sub`** has the form **`std_logic_vector(sub'length-1 downto 0).`** The function creates a new vector that is the size of **vect** plus **sub**. If **ins_pos** = 0, **`sub`** is appended to **`vect`** and if **`ins_pos`** = 1, **`sub`** is prepended to **`vect`**. Output an error if **`ins_pos`** doesn't equal 0 or 1.
b. (8 points) Show an architecture that includes three calls to the function with the following properties. 1 – appends (inserts after), 2 – prepends(inserts before), 3 – triggers error message.

2. (11 points) Modify the following VHDL model by adding an enable input. When enable = ‘0’, the outputs are set to “ZZZZ’ and as specified when enable = ‘1’. Further, translate the process into concurrent signal assignment statements that include the new functionality.

```
library ieee;
use ieee.std_logic_1164.all;

entity TWO_TO_4_DEC is
  generic(DEL: TIME);
  port(I: in  std_logic_vector(1 downto 0);
       O: out std_logic_vector(3 downto 0));
end TWO_TO_4_DEC;

architecture ALG of TWO_TO_4_DEC is
begin
  process(I)
  begin
    case I is
      when "00" => O<= "0001" after DEL;
      when "01" => O<= "0010" after DEL;
      when "10" => O<= "0100" after DEL;
      when "11" => O<= "1000" after DEL;
    end case;
  end process;
end ALG;
```

3. (15 points) Write a VHDL description for a D-type latch with an asynchronous active low reset, **r**, and an active high clock, `clk`. Create three generics – TPDQ, TPCQ and TPRQ which represent the propagation delay from **d**, **clk**, and **r**, respectively.

   a. (5 points) Write an entity using std_logic inputs and outputs.

   b. (10 points) Write a behavioral architecture

4. (12 points) Construct an 8 bit ALU using two 4- bit ALUs (entity shown below).
   a. (4 points) Entity
   b. (8 points) Architecture

```
library ieee;
use ieee.std_logic_1164.all;

entity ALU is
  generic(DEL: TIME);
  port(A,B: in std_logic_vector(3 downto 0); CI: in std_logic;
       FSEL: in std_logic_vector(1 downto 0);
       F: out std_logic_vector(3 downto 0); COUT: out std_logic);
end ALU;
```

5. (15 points) Given the following VHDL, indicate all transactions and events. Give the values of A, B, C, D, E, and F each time a change occurs. Carry this out until no further change occurs.

```
entity prob is
  port (D : out bit);
end prob;

architecture PROB of PROB is
  signal A, B, C, E, F : bit;
begin
  process
    A <= '1' after 5 ns;
    wait;
  end process;
  P1: process (F, C)
  begin
    B <= F after 4 ns;
    E <= C after 6 ns;
  end process P1;
  C <= transport A or B
         after 3 ns;
  P2: process (C, E)
  begin
    F <= (C xor E) after 4 ns;
  end process P2;
  D <= F xor (B and C) after 2 ns;
end PROB;
```

Time	A	B	C	D	E	F
0 ns	0	0	0	0	0	0
5 ns	1	0	0	0	0	0

Time	Event	Processes Triggered	Scheduled Transactions	Event?

Scheduling Rules	Transport	Inertial
New before existing	Overwrite existing	Overwrite existing
New after existing	Append new	If $v_{new} = v_{existing}$, append new Elsif t_{new}-$t_{existing}$ > reject append new Else overwrite existing

6. (12 points) Draw the state diagram for the following state machine. You may omit reset from all the arcs.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity TAPE_PLAYER is
  port(PL, RE, FF, ST, M : in std_logic;
       P, F, R : out std_logic;
       COUNT : out unsigned (15 downto 0);
       RESET, CLK : in std_logic);
end TAPE_PLAYER;

architecture BEHAV of TAPE_PLAYER is
  type STATE_TYPE is (INIT, PUSHPLAY, PP_RE1, PP_REW, PP_FF1, PP_FFOR, PLAY);
  signal STATE : STATE_TYPE;
  signal INT_COUNT : unsigned (15 downto 0);
begin
  process (RESET, CLK)
  begin
    if (RESET = '1') then
      STATE <= INIT;
    elsif (CLK'event and CLK = '1') then
      if (STATE = INIT) then
        if (PL = '1') then
          STATE <= PUSHPLAY;
        end if;
        INT_COUNT <= "0000000000000000";
      elsif (STATE = PUSHPLAY) then
        if (PL = '0' or ST = '1') then
          STATE <= INIT;
        else
          if (RE = '1') then
            STATE <= PP_RE1;
          elsif (FF = '1') then
            STATE <= PP_FF1;
          else
            STATE <= PLAY;
          end if;
        end if;
        INT_COUNT <= "0000000000000000";
      elsif (STATE = PP_RE1) then
        if (PL = '0' or ST = '1') then
          STATE <= INIT;
        else
          if (RE = '0') then
            STATE <= PP_REW;
          else
            STATE <= PP_RE1;
          end if;
        end if;
        INT_COUNT <= "0000000000000000";
      elsif (STATE = PP_REW) then
        if (ST = '1') then
```

```
          STATE <= INIT;
        else
          if (M = '1') then
            STATE <= PP_REW;
          else
            STATE <= INIT;
          end if;
        end if;
        INT_COUNT <= INT_COUNT + 1;
      elsif (STATE = PP_FF1) then
        if (PL = '0' or ST = '1') then
          STATE <= INIT;
        else
          if (FF = '0') then
            STATE <= PP_FFOR;
          else
            STATE <= PP_FF1;
          end if;
        end if;
        INT_COUNT <= "0000000000000000";
      elsif (STATE = PP_FFOR) then
        if (ST = '1') then
          STATE <= INIT;
        else
          if (M = '1') then
            STATE <= PP_FFOR;
          else
            STATE <= INIT;
          end if;
        end if;
        INT_COUNT <= INT_COUNT + 1;
      elsif (STATE = PLAY) then
        if (ST = '1') then
          STATE <= INIT;
        else
          STATE <= PLAY;
        end if;
        INT_COUNT <= INT_COUNT + 1;
      end if;
    end if;
    COUNT <= INT_COUNT;
  end process;

  P <= '1' when STATE = PLAY else '0';
  F <= '1' when STATE = PP_FFOR else '0';
  R <= '1' when STATE = PP_REW else '0';

end BEHAV;
```

7. (1 point) _______________ delay represents wire delay.

8. (1 point)_______Multiple Choice: Which of the following cannot occur inside a process?

   a. Signal Assignment b. Variable Declaration c. Signal Declaration

9. (1 point) A ________________________ is used when you have multiple return values.

10. (1 point) A process is triggered whenever a signal in its ____________________________ has an event on it.

11. (1 point)._______ (True or False) All sequential statements are synthesized into sequential circuits.

12. (10 points) In order to add floating point numbers, the numbers must first have the same exponent, so combinational shifting is required so that floating point add can be accomplished in a reasonable time. Design a 4 bit barrel shifter that has 4 data inputs and 4 data outputs, and 2 control inputs that determine the shift amount (0-3). Use the truth table below as the functionality and use the entity declaration given.

s1	s0	c3	c2	c1	c0
0	0	a3	a2	a1	a0
0	1	a0	a3	a2	a1
1	0	a1	a0	a3	a2
1	1	a2	a1	a0	a3

```
entity barrel is
  port (a : in std_logic_vector (3 downto 0); s1, s0 : in std_logic;
    c : out std_logic_vector (3 downto 0));
end entity barrel;
```