The University of Alabama in Huntsville
ECE Department
CPE 426 01
Midterm Exam
February 27, 2013

Name: _____

1.. (1 point) A(n) _____ is an example of a design unit.

2. (1 point) A _____ is a signal used in describing the interface of a VHDL model.

3. (10 points) Create a VHDL architecture representing a structural model of an 8-bit odd-parity checker using the entity given and instances of the exclusive-or gate entity given.

```
entity ODD_PARITY_8 is
  Port ( I : in std_logic (7 downto 0);
         P : out std_logic) ;
end entity ODD_PARITY_8;

entity XOR is
  port ( A, B : in std_logic;
         F : out std_logic);
end entity XOR;
```

The logic equation describing the parity checker is

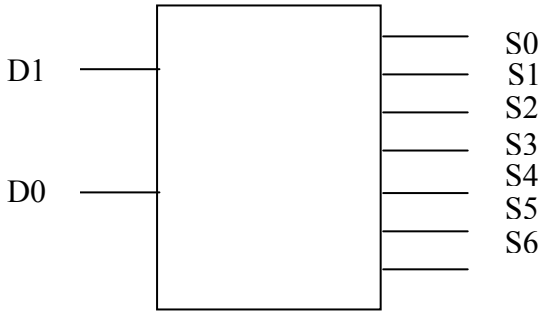
$$P = ((I0 \oplus I1) \oplus (I2 \oplus I3)) \oplus ((I4 \oplus I5) \oplus (I6 \oplus I7))$$

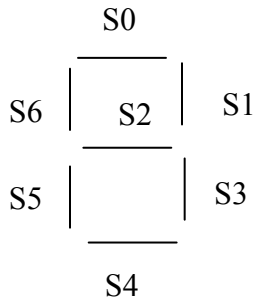
4. (7 points) Write the equivalent process for the conditional signal assignment statement

```
mux_logic:
  z <= a and not b after 5 ns when enable and not sel else
       x or y after 6 ns when enable and sel else
      '0' after 4 ns;
```

5. (15 points) (a) (10 points) Write a VHDL procedure called align_address that aligns a binary encoded address in a bit-vector variable parameter. The procedure has a second parameter that indicates the alignment size. If the size is 1, the address is unchanged. If the size is 2, the address is rounded to a multiple of 2 by clearing the least significant bit. If the size is 4, two bits are cleared, and if the size is 8, three bits are cleared. The default alignment size is 4. (b)(5 points) Show an architecture that includes two calls to the function with the following properties. 1 - returns a bit_vector with size of 1 length, 2 – passes only an address, not a size.

6. (8 points) Consider the following combinational digital system, called a light-emitting diode (LED) driver. The LED driver converts a 2-bit binary number (D_1D_0) into an LED-displayed numeral. For example, $D_1D_0 - 10_2$ is displayed by asserting S_0, S_1, S_2, S_4, and S_5.






```
entity LED_DRIVER is
  port (D : in bit_vector (1 downto 0);
        S : out bit_vector(6 downto 0));
end entity LED_DRIVER;
```

Use concurrent signal assignments to model the LED driver.

7. (1 point) 'RANGE is an example of a VHDL _____.

8. (1 point) VHDL is a strongly typed language _____ (True/False)

9. (2 points) Create a COLLEGE enumeration data type that has the values of the colleges at UAH.

10. (1 point) A function is a primary design unit. (True/False) _____.

11. (3 points) Write a declaration of a two-dimensional table, TABLE_2D, with index values and table entries all of type bit (which has been declared elsewhere and is visible). Initialize all elements of the array to '1';

12. (20 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 : inout bit);
end prob;

architecture PROB of PROB is
  signal A, B, C, E, F : bit;
begin
  process
    A <= '1' after 5 ns,
         '0' after 12 ns;
    wait;
  end process;
  P1: process (A, C)
  begin
    B <= A after 2 ns;
    E <= C after 7 ns;
  end process P1;
  C1: C <= A and B after 6 ns;
  P2: process (C, E)
  begin
    F <= C and E after 4 ns;
  end process P2;
  C2: D <= A or B or C or F;
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

13. (15 points) Develop a behavioral model for a D-latch with tristate output. The entity declaration is

```
entity d_latch is
  port ( latch_en, out_en, d : in std_logic;
         q : out std_logic );
end entity d_latch;
```

When `latch_en` is asserted, data from the `d` input enters the latch. When `latch_en` is negated, the latch maintains the stored value. When `out_en` is asserted, data passes through to the output. When `out_en` is negated, the output has the value 'Z' (high-impedance). The propagation delay from `latch_en` to `q` is 3 ns and from `d` to `q` is 4 ns. The delay from `out_en` asserted to `q` active is 2 ns and from `out_en` negated to `q` high-impedance is 5 ns.

14. (15 points) Draw the state diagram for the following state machine. Is it a Moore machine or a Mealy machine?

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

entity THUNDERBIRD is
  port (L, R, H, CLK : in std_logic;
        LC, LB, LA, RA, RB, RC : out std_logic);
end THUNDERBIRD;

architecture BEHAVE of THUNDERBIRD is
  type STATE_TYPE is (S0, S1, S2, S3, S4, S5, S6, S7);
  signal CURRENT_STATE, NEXT_STATE : STATE_TYPE;
begin
  process(CURRENT_STATE, L, R, H)
    variable INPUTS : std_logic_vector(2 downto 0);
  begin
    INPUTS := L&R&H;
    case CURRENT_STATE is
      when S0 => if (INPUTS = "000") then
                   NEXT_STATE <= S0;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S1;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S4;
                 else
                   NEXT_STATE <= S7;
                 end if;
      when S1 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S2;
                 else
                   NEXT_STATE <= S0;
                 end if;
      when S2 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "100") then
                   NEXT_STATE <= S3;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S3|S6 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S4 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S5;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S5 => if (INPUTS(0) = '1') then
                   NEXT_STATE <= S7;
                 elsif (INPUTS = "010") then
                   NEXT_STATE <= S6;
                 else
                   NEXT_STATE <= S0;
                 end if;
       when S7 => NEXT_STATE <= S0;
     end case;
   end process;
```

```
  process (CLK)
  begin
    if (CLK'event and CLK = '1') then
      CURRENT_STATE <= NEXT_STATE;
    end if;
  end process;
  process(CURRENT_STATE)
  begin
    LC <= '0'; LB <= '0'; LA <= '0';
    RA <= '0'; RB <= '0'; RC <= '0';
    case CURRENT_STATE is
      when S0 => null;
      when S1 => LA <= '1';
      when S2 => LA <= '1'; LB <= '1';
      when S3 => LA <= '1'; LB <= '1'; LC <= '1';
      when S4 => RA <= '1';
      when S5 => RA <= '1'; RB <= '1';
      when S6 => RA <= '1'; RB <= '1'; RC <= '1';
      when S7 => LA <= '1'; LB <= '1'; LC <= '1';
                 RA <= '1'; RB <= '1'; RC <= '1';
    end case;
  end process;
end BEHAVE;
```