The University of Alabama in Huntsville
ECE Department
CPE 526 01
Midterm Exam Solution
Spring 2009

1. (15 points) (a) (4 points) Create a VHDL entity named `32_bit_adder`.(b) (11 points) Create a VHDL architecture representing a structural model of the 32 bit adder using as many `8_bit_adder` components as are needed. You do not need to write an entity or an architecture for `8_bit_adder`. You may also assume that a component has already been declared and that no configuration statement is required.

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

entity ADDER_32_BIT is
  port (A, B : in std_logic_vector(31 downto 0);
        CIN : in std_logic;
        COUT : out std_logic;
        C : out std_logic_vector(31 downto 0));
end ADDER_32_BIT;

architecture EXAMPLE of ADDER_32_BIT is
  component ADDER_8_BIT is
    port (A, B : in std_logic_vector(7 downto 0);
          CIN : in std_logic;
          COUT : out std_logic;
          C : out std_logic_vector(7 downto 0));
  end component;
  signal CARRY : std_logic_vector(2 downto 0);
begin
  U0: ADDER_8_BIT
      port map (A => A(7 downto 0), B => B(7 downto 0),
                C => C(7 downto 0), CIN => CIN,
                COUT => CARRY(0));
  U1: ADDER_8_BIT
      port map (A => A(15 downto 8), B => B(15 downto 8),
                C => C(15 downto 8), CIN => CARRY(0),
                COUT => CARRY(1));
  U2: ADDER_8_BIT
      port map (A => A(23 downto 16), B => B(23 downto 16),
                C => C(23 downto 16), CIN => CARRY(1),
                COUT => CARRY(2));
  U3: ADDER_8_BIT
      port map (A => A(31 downto 24), B => B(31 downto 24),
                C => C(31 downto 24), CIN => CARRY(2),
                COUT => COUT);
end EXAMPLE;
```

2. (1 point) _False_ (True/False) There is no difference between CLK’event and not CLK’stable.

3. (20 points) (a) (12 points) Write a VHDL function that takes two std_logic_vectors. The function searches the first argument to see whether the second argument appears as a subvector of the first. If the second argument is found, the function returns the bit position of the first match, otherwise it returns -1. You may assume that both arguments have the form std_logic_vector(x'length-1 down to 0) where x is the name of the argument. Output an error if the length of the second argument exceeds that of the first. (b)(8 points) Show an architecture that includes three calls to the function with the following properties. 1 - returns a value, 2 – triggers an error message, 3 – returns -1.

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

package MINE is
  function SEARCH(VECTOR, SUBVECTOR : std_logic_vector) return integer;
end MINE;

package body MINE is
  function SEARCH(VECTOR, SUBVECTOR : std_logic_vector) return integer is
    variable POSITION : integer;
    variable FOUND : boolean := FALSE;
  begin
    POSITION := -1;
    if (VECTOR'length < SUBVECTOR'length) then
      report "Incorrect usage of SEARCH function"
      severity error;
      return POSITION;
    end if;
    for I in VECTOR'length - 1 downto SUBVECTOR'length - 1 loop
      if not FOUND then
        if (VECTOR(I downto I - (SUBVECTOR'length -1)) = SUBVECTOR) then
          POSITION := I;
          FOUND := TRUE;
        end if;
      end if;
    end loop;
    return POSITION;
  end SEARCH;
end MINE;

use work.MINE.all;
library ieee;
use ieee.std_logic_1164.all;

entity TEST_SEARCH is
end TEST_SEARCH;

architecture TEST of TEST_SEARCH is
  signal A : std_logic_vector(2 downto 0) := "000";
  signal B : std_logic_vector(7 downto 0) := "11111111";
  signal C : std_logic_vector(32 downto 0) :=
            "110010101111000111111111100101011";
  signal D, E, F : integer;
begin
  D <= SEARCH(C, A) after 1 ns;
  E <= SEARCH(B, A) after 2 ns;
  F <= SEARCH(A, B) after 3 ns;
end TEST;
```

4. (1 point) _True_ (True/False) Operators may be overloaded in VHDL.

5. (1 point) _Transport_ delay is the delay which represents wire delay in VHDL.

6. (1 point) _False_ (True/False) Functions are primary design units.

7. (8 points) Design a majority gate that outputs a ‘1’ if two or more of the three inputs are ‘1’. (a) (3 points) Write a VHDL entity that includes a generic for delay. (b) (5 points) Use concurrent signal assignments to implement the architecture.

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

entity MAJ3 is
  generic(DEL : time);
  port (X : in std_logic_vector(2 downto 0);
        F : out std_logic);
end MAJ3;

architecture MAJ3 of MAJ3 is
begin
  F <= '1' after DEL when (X(2) = '1' and X(1) = '1') OR
                          (X(1) = '1' and X(0) = '1') OR
                          (X(2) = '1' and X(0) = '1') else
       '0' after DEL;
end MAJ3;
```

8. (3 points) (a) (2 points) Specify a CLASSIFICATION enumeration data type that spells out the various classifications for undergraduate students.(b) (1 point) Write a signal declaration MY_CLASS that has a value equal to the rightmost element of the type.

```
  type CLASSIFICATION is (FRESHMAN, SOPHOMORE, JUNIOR, SENIOR);
  signal MY_CLASS : CLASSIFICATION := CLASSIFICATION'right;
```

9. (1 point) _b_Multiple Choice: Which of the following cannot occur outside a process?
   (a) Signal Assignment (b) Variable Declaration (c) Signal Declaration

10. (4 points) (a) (3 points) Write a declaration of an array that can be used to hold the email addresses of the students in this class. (b) (1 point) Initialize the first element of this array with your email address.

```
  type EMAILS_TYPE is array(0 to 15) of string(17 downto 1);
  signal EMAILS : EMAILS_TYPE := (0 => "gaede@ece.uah.edu",
                        others => "                 ");
```

11. (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 : in    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 (D, C)
  begin
    B <= D after 2 ns;
    E <= C after 7 ns;
  end process P1;
  C1: C <= transport A or E
         after 6 ns;
  P2: process (C, E)
  begin
    F <= (C and E) after 4 ns;
  end process P2;
  C2: D <= A xor B xor C after 1 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
6 ns	1	0	0	1	0	0
8 ns	1	1	0	1	0	0
9 ns	1	1	0	0	0	0
11 ns	1	0	0	0	0	0
18 ns	1	0	0	0	1	0
22 ns	1	0	0	0	1	1

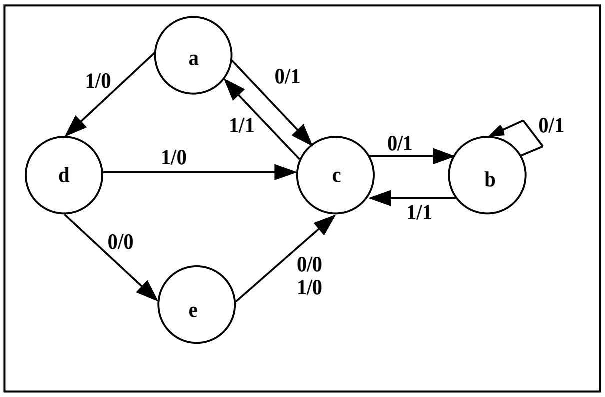
Time	Event	Processes Triggered	Scheduled Transactions	Event?
5 ns	A→1	C1	C ‘1’ 11 ns	Y
		C2	D ‘1’ 6 ns	Y
6 ns	D→1	P1	B ‘1’ 8 ns	Y
		P1	E ‘0’ 13 ns	N
8 ns	B→1	C2	D ‘0’ 9 ns	Y
9 ns	D→0	P1	B ‘0’ 11 ns	Y
		P1	E ‘0’ 16 ns	N
11 ns	B→0	C2	D ‘0’ 12 ns	N
	C→1	P1	B ‘0’ 13 ns	N
		P1	E ‘1’ 18 ns	Y
		P2	F ‘0’ 15 ns	N
18 ns	E→1	C1	C ‘1’ 24 ns	N
		P2	F ‘1’ 22 ns	Y
22 ns	F→1	none		

12. (15 points) Design a new type of positive-edge-triggered flip-flop called the LH flip-flop. It has a clock C, a data input D, and a load input L. If, at the positive edge of C, L equals 1, then the data on D is stored in the flip-flop. If, at the positive edge of C, L equals 0, then the current stored value in the flip-flop is held. (a) (3 points) Write a VHDL entity. (b) (6 points) Use concurrent signal assignments to implement the architecture. (c) (6 points) Use sequential statements to implement the architecture. Include any necessary library references.

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

entity LHFF is
  port (C, L, D : in std_logic;
        Q, QB : out std_logic);
end LHFF;

architecture CON of LHFF is
  signal TEMP : std_logic;
begin
  A: block (not C'STABLE and C = '1')
  begin
    TEMP <= GUARDED D when (L = '1');
  end block A;
  Q <= TEMP;
  QB <= not TEMP;
end CON;

architecture SEQ of LHFF is
  signal TEMP : std_logic;
begin
  process(C)
  begin
    if (C'EVENT and C = '1') then
      if (L = '1') then
        TEMP <= D;
      end if;
    end if;
  end process;
  Q <= TEMP;
  QB <= not TEMP;
end SEQ;
```

13. (10 points) Draw the state diagram for the following state machine. Is it a Moore machine or a Mealy machine? Mealy

```
ENTITY state_machine IS
   PORT (sig_in ; IN BIT; clk, rst : IN BIT;
          sig_out : OUT BIT);
END state_machine;

ARCHITECTURE state_machine OF state_machine IS
   TYPE state_type IS (a, b, c, d, e);
   SIGNAL current_state, next_state : state_type;
BEGIN
   PROCESS (sig_in, current_state)
   BEGIN
      sig_out <= ‘0’;
      next_state <= c;
      CASE current_state
      WHEN a =>
         IF sig_in = ‘0’ THEN
            next_state <= c;
            sig_out <= ‘1’;
         ELSE
            next_state <= d;
         END IF;
      WHEN b =>
         IF sig_in = ‘0’ THEN
            next_state <= b;
         ELSE
            next_state <= c;
         END IF;
         sig_out <= ‘1’;
     WHEN c =>
       IF sig_in = ‘1’ THEN
           sig_out <= ‘1’;
           next_state <= a;
       ELSE
          next_state <= b;
       END IF;
       sig_out <= ‘1’;
     WHEN d =>
       IF sig_in = ‘0’ THEN
           next_state <= e;
       END IF;
     WHEN e =>
       IF sig_in = ‘1’ THEN
          next_state <= c;
       END IF;
      END CASE;
   END PROCESS;
   PROCESS (clk)
   BEGIN
      IF (rst = ‘0’) then
         current_state <= a;
      ELSIF (clk’EVENT AND clk = ‘1’) THEN
         current_state <= next_state;
      END IF;
   END PROCESS;
END state_machine;
```


a
1/0
0/1
1/1
0/1
0/1
d
1/0
c
b
1/1
0/0
0/0
1/0
e