EE 414/514
VHDL DESIGN

Dr. J.A. Starzyk

MIDTERM Tuesday, October 23, 2008

THIS IS A CLOSED BOOK EXAM

Name:_____ Box #:

Note:

1) All scratch and problem papers must be turned in.
2) Estimated times required to complete problems are indicated.

Problem	Weight (points)	Estimated Time (minutes)	Examiner' Use only
1	15	10(8)	
2	10	15(10)	
3	10	10(10)	
4	10	15(10)	
5	15	25 (15)	
TOTAL	60	75	

GOOD LUCK!!!

Problem 1
Answer True or False next to each statement:

1. All parameters in an entity declaration are generics. (**True**)
2. The structural design is easier to synthesize than behavioral because it specifies what the system is expected to do. (**False**)
3. A package is a compact way of writing an entity-architecture pair to disk. (**False**)
4. Internal signals require mode (in, out or inout) declarations. (**False**)
5. The order of bits specified in a vector is not important. (**False**)
6. All signals of a system are defined in the systems entity. (**False**)
7. A port must be either input or output. (**False**)
8. VHDL is a case-sensitive language. (**False**)
9. It is allowed to specify initial value of a port. (**True**)
10. A constant can be assigned a new value if the new value is equal to the previous one. (**False**)
11. A Boolean true is equivalent of bit '1'. (**False**)
12. Logical operators can be applied to single bits only. (**False**)
13. The Boolean condition in a while loop is checked at the end of each iteration. (**False**)
14. Execution of a process stops when the end process clause is reached. (**False**)
15. Signals are assigned values only when the process suspends. (**True**)

Problem 2

Dilbert wrote a VHDL code for his entity named *"test2"*.

a) **Determine the functionality of the part he intended to build with this code.**
b) **Complete this code to implement the intended functionality. Please specify the reason for each code insertion.**

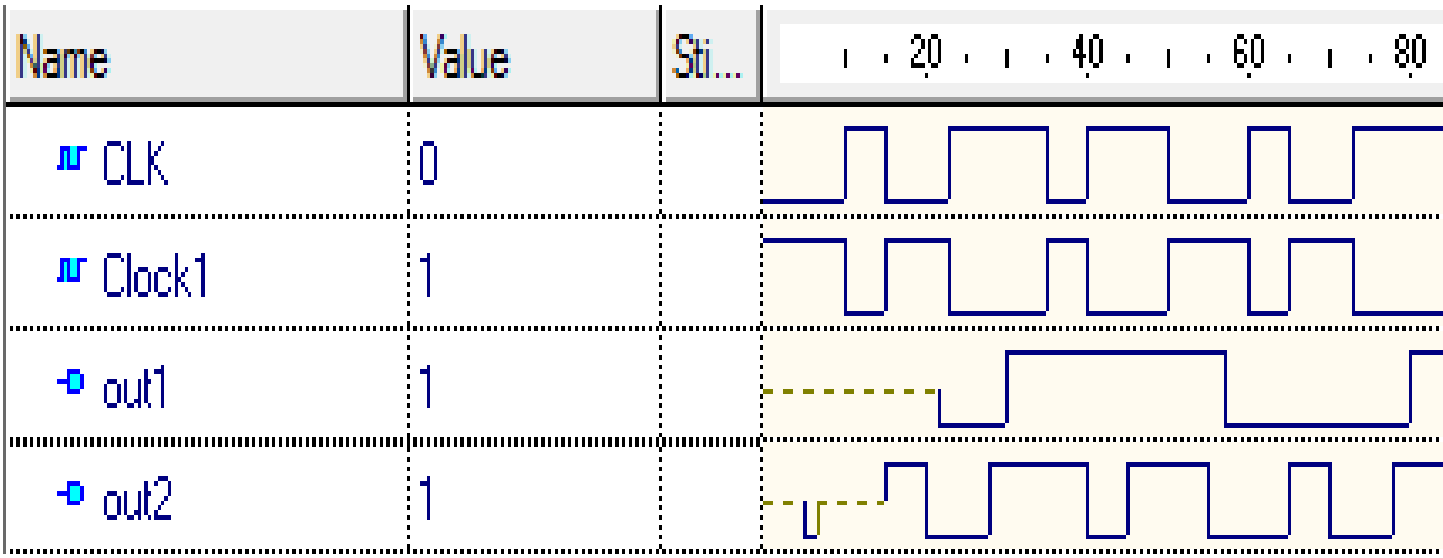
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity test2_a is
  port(C, CLR, up_down : in std_logic;                    (Mode decleration)
       Q : out std_logic_vector(3 downto 0));             (Missing brackets)
end test2_a;

architecture archi of test2_a is                          (Entity Name is required)
  signal tmp: std_logic_vector(3 downto 0);               (It is 'Signal' and not 'Variable')
  begin
    process (C, CLR)
      begin
        if CLR='1' then                    (It has to be CLR='1' and not  CLR =="1")
          tmp <= "0000";                                  (<= represents signal)
        elsif (C'event and C='1') then
          if (up_down='1') then
            tmp  <= tmp + 1;
          else
            tmp  <= tmp - 1;
          end if;
        end if;                                           (Missing End if)
    end process;
    Q  <= tmp;
end archi;
```

Functionality : - 4-bit unsigned Up/Down counter with asynchronous clear.

Problem 3: -

Draw the output waveforms for the following code. Assume that the entity "*prob3*" is simulated with a clock, "*CLK*", shown in the figure below. Use the plot on the next page.

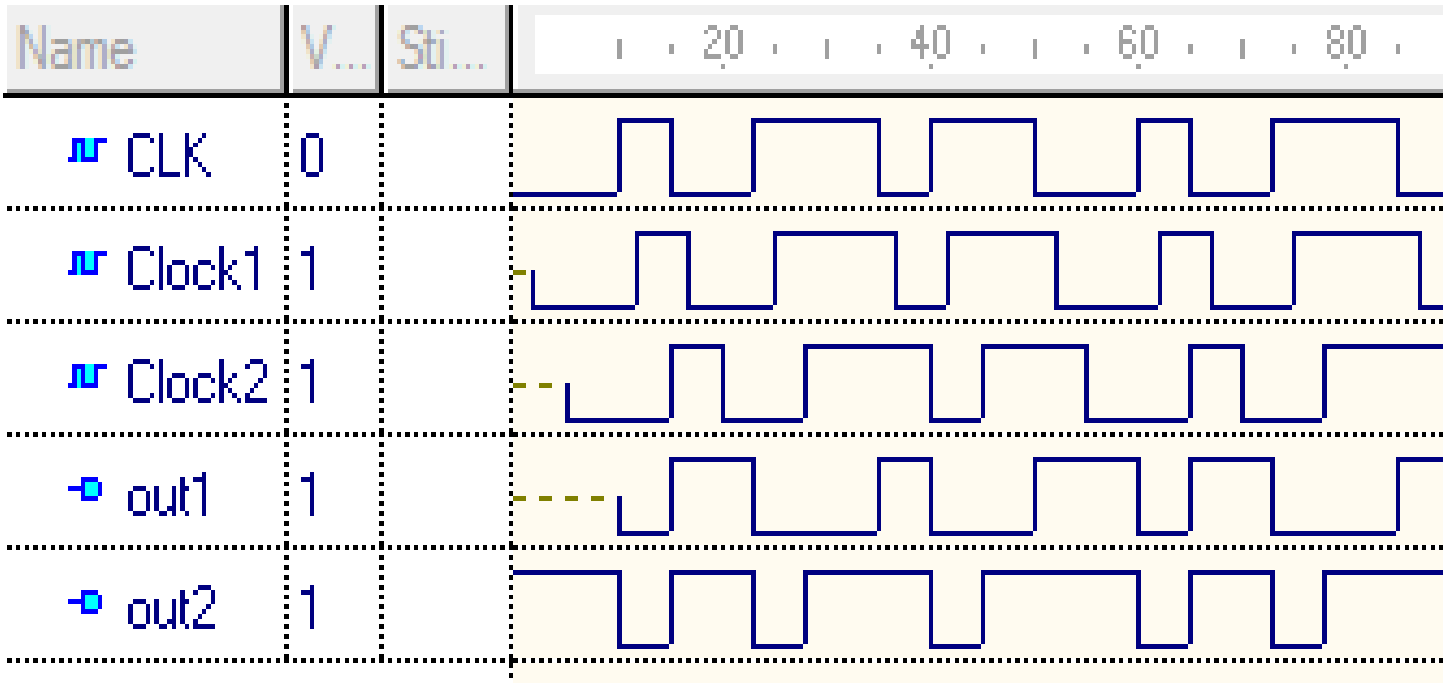
```
library IEEE;
use IEEE.std_logic_1164.all;
entity prob3 is
        port (
        out1 : out std_logic;
        out2 : out std_logic);
end entity;
architecture behv of prob3 is
signal CLK :std_logic;
signal Clock1 :std_logic;
begin
        CLOCK: process
        begin
                CLK<= '0','1' after 10ns,'0' after 15ns,'1' after 23ns,'0' after 35ns, '1' after 40ns;
                wait for 50 ns;
        end process;
proc1: process(CLK)
        begin
                Clock1 <= not CLK;
                out1<= Clock1 after 7ns;
                out2 <= transport CLK after 5ns;
                out2 <= transport Clock1 after 5ns;
        end process;
end behv;
```



Problem 4

Assume that an entity "*test4*" is simulated with an input clock signal equal to CLK from the problem 3 and shown in the figure below. Draw the signal waveform for the following code. Use the plot shown below. CLK, Clock1 and Clock2 are signals in the architecture, and out1, and out2 are ports.

```
proc1: process(CLK)
begin
        Clock1 <= CLK after 2ns;
        out1<= Clock1;
end process;

Clock2 <= CLK after 5ns;
out2 <= (not Clock2) nand CLK;
```



Problem 5

Figure 1 shows the logic diagram of the blocks 1 and 2 of figure 2. The inputs a_1 and a_2 in figure 2 refer to a of figure 1. Similarly b_1 and b_2 refer to b and z_1 and z_2 refer to z. If all the signals are of type "std_logic",

a) **Implement the entity Design1 shown in figure 1.**
b) **Implement the entity Design2 shown in figure 2 in structural VHDL by using circuit of figure 1 as a component.**
c) **Identify the circuit function of Design1 by writing logical expression for the outputs.**

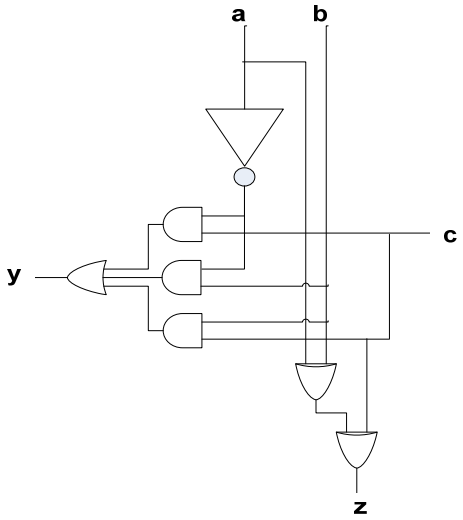


Figure1: - Design1

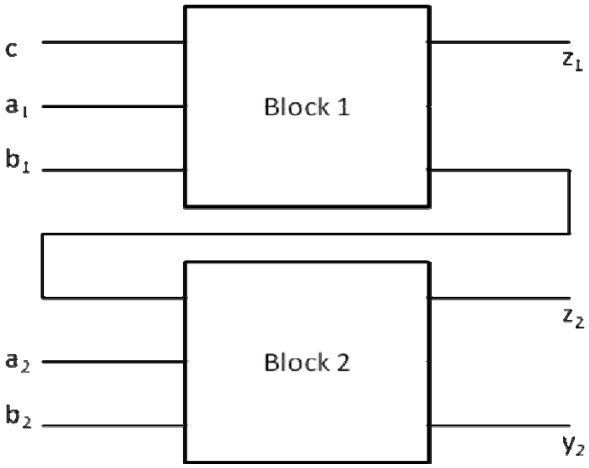


Figure2: - Design2

Implementation of Design1:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity Design1 is
        port(
                a : in STD_LOGIC;
                a : in STD_LOGIC;
                c : in STD_LOGIC;
                y : out STD_LOGIC;
                z : out STD_LOGIC
            );
end Design1;
```

```
architecture Design1 of Design1 is
begin
        y <= ((not a)and c) or((not a)and b) or (b and c);
        z <= (a xor b) xor c;
end Design1;
```

Implementation of Design2:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity Design2 is
        port(
                a1,a2 : in STD_LOGIC;
                b1,b2 : in STD_LOGIC;
                c : in STD_LOGIC;
                y2 : out STD_LOGIC;
                z1,z2 : out STD_LOGIC
            );
end Design2;

architecture Design2 of Design2 is
component Design1
        port(
                a : in STD_LOGIC;
                a : in STD_LOGIC;
                c : in STD_LOGIC;
                y : out STD_LOGIC;
                z : out STD_LOGIC
            );
end component;

signal b_int : std_logic;

begin
U1: Design1
port map (a1, b1, c, b_int, z1);

U2: Design1
port map (a2, b2, b_int, y2, z2);

end Design2;
```

Functionality:- Full subtractor
y = a'.c + a'.b + b.c;
z = ((a xor b) xor c);