

Fall 2007 CS460 Test One

Name :

ID:

1. Define the meaning of the following terms and abbreviations in the context of embedded systems and FPGAs.

IOB

PLD

NRE

generic

LED

tristate

2a. Explain the function of a two std-logic input, four std-logic output decoder.

2b. Draw a circuit diagram (using NOTs, ANDs and ORs) for the decoder described above.

3. Identify three logical errors in the following VHDL fragment.

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

entity S is
  generic (
    bitwidth : integer := 4);
  port (
    clk : in std_logic;
    rst : in std_logic;
    Xin : in std_logic_vector(bitwidth-1 downto 0);
    Xout : out std_logic_vector(bitwidth-1 downto 0));
end S;

architecture behavior of S is
  signal P : std_logic_vector(bitwidth+1 downto 0);

begin -- behavior

  bla : process (clk, rst, Xin)
  begin
    if rst = '1' then
      P <= (others => '0');
    elsif clk'event and clk = '1' then
      P <= Xin xor P;
    end if;
  end bla;

end behavior;
```

The following VHDL fragment defines the entity “test”.

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

entity test is
  port (
    Clock : in  std_logic;
    Reset  : in  std_logic;
    A      : in  std_logic;
    B      : in  std_logic;
    Q      : out std_logic;
  end test;

  architecture play of test is

    signal C : std_logic;
    signal I : std_logic_vector(1 downto 0);
    signal J : std_logic_vector(3 downto 0);

  begin

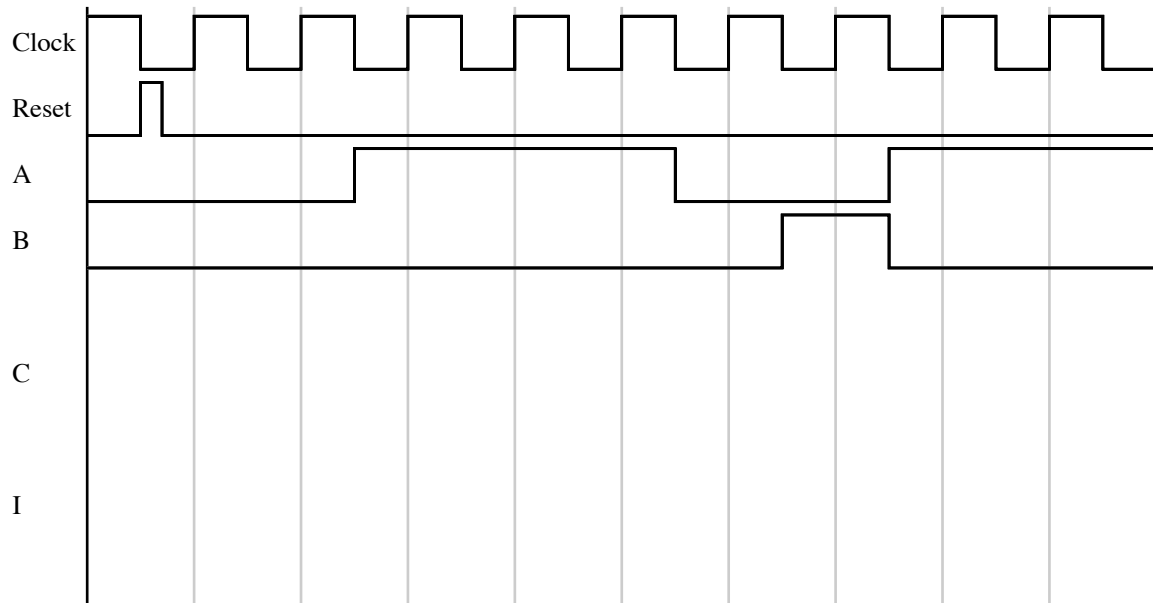
    C <= A and (not B);
    J <= "0111";

    Sally : process (Clock, Reset)
    begin
      if Reset = '1' then
        I <= "00";
      elsif Clock'event and Clock = '1' then
        if C = '1' then
          I <= J(1 downto 0);
        else
          I <= J(3 downto 2);
        end if;
      end if;
    end process Sally;

    Q <= not C;

  end play;
```

4. Fill in the Timing-diagram for “test” below:



The following VHDL fragment defines the entity “thingy”:

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

entity thingy is

    generic (
        W : integer := 8;
        D : integer := 3);

    port (
        Clock : in  std_logic;
        Reset  : in  std_logic;
        S      : in  std_logic;
        VI     : in  std_logic_vector(W-1 downto 0);
        VO     : out std_logic_vector((W * D)-1 downto 0));

end thingy;

architecture behavior of thingy is

    signal LVO : std_logic_vector((W * D)-1 downto 0);

begin

    MakeRegs : process (Clock, Reset)
    begin
        if Reset = '1' then
            LVO <= (others => '0');
        elsif Clock'event and Clock = '1' then
            if(S = '1') then
                -- assgn1
                LVO(W-1 downto 0) <= VI;

                -- assgn2
                LVO((W * D)-1 downto W) <=
                    LVO((W * (D-1))-1 downto 0);

            end if;
        end if;
    end process MakeRegs;

    VO <= LVO;

end behavior;
```

5a. Is the signal LVO needed? Explain

5b. What happens if the order of the signal assignments commented “assgn1” and “assgn2” is reversed?

5c. Describe in high level terms the function of thingy.