

FPGA Homework - 1

Parham Alvani (9231058)

March 5, 2016

1 PROBLEM 4

1.1 PART 1

```
-----  
-- Author:      Parham Alvani (parham.alvani@gmail.com)  
--  
-- Create Date:  04-03-2016  
-- Module Name:  p4-1.vhd  
-----
```

```
library IEEE;  
use IEEE.std_logic_1164.all;  
  
entity four_bit_comparator is  
    port (a, b : in std_logic_vector(3 downto 0);  
          l, g, e : in std_logic;  
          eq, gt, lt : out std_logic);  
end entity four_bit_comparator;  
  
architecture structural of four_bit_comparator is  
    signal a0_b0_eq, a0_b0_gt, a0_b0_lt : std_logic;  
    signal a1_b1_eq, a1_b1_gt, a1_b1_lt : std_logic;  
    signal a2_b2_eq, a2_b2_gt, a2_b2_lt : std_logic;  
    signal a3_b3_eq, a3_b3_gt, a3_b3_lt : std_logic;  
begin
```

```

a0_b0_gt <= a(0) and (not b(0));
a0_b0_lt <= (not a(0)) and b(0);
a0_b0_eq <= a(0) xor b(0);

a1_b1_gt <= a(1) and (not b(1));
a1_b1_lt <= (not a(1)) and b(1);
a1_b1_eq <= a(1) xor b(1);

a2_b2_gt <= a(2) and (not b(2));
a2_b2_lt <= (not a(2)) and b(2);
a2_b2_eq <= a(2) xor b(2);

a3_b3_gt <= a(3) and (not b(3));
a3_b3_lt <= (not a(3)) and b(3);
a3_b3_eq <= a(3) xor b(3);

eq <= a3_b3_eq and a2_b2_eq and a1_b1_eq and a0_b0_eq;
gt <= a3_b3_gt or (a2_b2_gt and a3_b3_eq) or (a1_b1_gt and a2_b2_eq and a3_b3_eq)
    or (a0_b0_gt and a1_b1_eq and a2_b2_eq and a3_b3_eq);
lt <= a3_b3_lt or (a2_b2_lt and a3_b3_eq) or (a1_b1_lt and a2_b2_eq and a3_b3_eq)
    or (a0_b0_lt and a1_b1_eq and a2_b2_eq and a3_b3_eq);
end architecture structural;

```

1.2 PART 2

```

-----
-- Author:      Parham Alvani (parham.alvani@gmail.com)
--
-- Create Date:  04-03-2016
-- Module Name:  p4-2.vhd
-----

library IEEE;
use IEEE.std_logic_1164.all;

entity sixteen_bit_comparator is
    port (a, b : in std_logic_vector(15 downto 0);
          l, g, e : in std_logic;
          eq, gt, lt : out std_logic);
end entity sixteen_bit_comparator;

architecture structural of sixteen_bit_comparator is
    signal a0_b0_eq, a0_b0_gt, a0_b0_lt : std_logic;
    signal a1_b1_eq, a1_b1_gt, a1_b1_lt : std_logic;
    signal a2_b2_eq, a2_b2_gt, a2_b2_lt : std_logic;

```

```

    signal a3_b3_eq, a3_b3_gt, a3_b3_lt : std_logic;

    entity 4_bit_comparator is
        port (a, b : in std_logic_vector(3 downto 0);
              l, g, e : in std_logic;
              eq, gt, lt : out std_logic);
    end entity 4_bit_comparator;

    for all:4_bit_comparator use entity work.4_bit_comparator;

begin
    c0: 4_bit_comparator port map (a(3 downto 0), b(3 downto 0), open, open, open,
        a0_b0_eq, a0_b0_gt, a0_b0_lt);

    c1: 4_bit_comparator port map (a(7 downto 4), b(7 downto 4), open, open, open,
        a1_b1_eq, a1_b1_gt, a1_b1_lt);

    c2: 4_bit_comparator port map (a(11 downto 8), b(11 downto 8), open, open, open,
        a2_b2_eq, a2_b2_gt, a2_b2_lt);

    c3: 4_bit_comparator port map (a(15 downto 12), b(15 downto 12), open, open, open,
        a3_b3_eq, a3_b3_gt, a3_b3_lt);

    eq <= a3_b3_eq and a2_b2_eq and a1_b1_eq and a0_b0_eq;
    gt <= a3_b3_gt or (a2_b2_gt and a3_b3_eq) or (a1_b1_gt and a2_b2_eq and a3_b3_eq)
        or (a0_b0_gt and a1_b1_eq and a2_b2_eq and a3_b3_eq);
    lt <= a3_b3_lt or (a2_b2_lt and a3_b3_eq) or (a1_b1_lt and a2_b2_eq and a3_b3_eq)
        or (a0_b0_lt and a1_b1_eq and a2_b2_eq and a3_b3_eq);
end architecture structural;

```

1.3 PART 3

```

-----
-- Author:      Parham Alvani (parham.alvani@gmail.com)
--
-- Create Date:  03-03-2016
-- Module Name:  p4-3.vhd
-----

```

```

library IEEE;
use IEEE.std_logic_1164.all;

entity decoder_2_4 is
    port (i0, i1 : in std_logic;
          o0, o1, o2, o3 : out std_logic);

```

```

end entity decoder_2_4;

architecture structural of decoder_2_4 is
begin
    out0 <= (not i0) and (not i1);
    out1 <= i0 and (not i1);
    out2 <= (not i0) and i1;
    out3 <= i0 and i1;
end architecture structural;

```

1.4 PART 4

```

-----
-- Author:          Parham Alvani (parham.alvani@gmail.com)
--
-- Create Date:     03-03-2016
-- Module Name:     p4-4.vhd
-----

library IEEE;
use IEEE.std_logic_1164.all;

entity carry_look_ahead_adder is
    generic (N : natural := 4);
    port (a, b : in std_logic_vector(N - 1 downto 0);
          s : out std_logic_vector(N - 1 downto 0);
          cin : in std_logic;
          cout : out std_logic);
end entity carry_look_ahead_adder;

architecture structural of carry_look_ahead_adder is
    signal P, G : std_logic_vector(N - 1 downto 0);
    signal C : std_logic_vector(N downto 0);
begin
    C(0) <= cin;
    cout <= C(N);

    carry: for I in 1 to N generate
        C(I) <= G(I - 1) or (P(I - 1) and C(I - 1));
    end generate carry;

    p_and_g: for I in 0 to N - 1 generate
        P(I) <= a(I) xor b(I);
        G(I) <= a(I) and b(I);
    end generate p_and_g;

```

```

        sum: for I in 0 to N - 1 generate
            s(I) <= a(I) xor b(I) xor C(I);
        end generate sum;
end architecture structural;

```

1.5 PART 5

```

-----
-- Author:      Parham Alvani (parham.alvani@gmail.com)
--
-- Create Date:  04-03-2016
-- Module Name:  p4-5.vhd
-----

library IEEE;
use IEEE.std_logic_1164.all;

entity counter is
    generic (N : natural := 4);
    port (clk : in std_logic;
          d : out std_logic_vector(N - 1 downto 0));
end entity counter;

architecture structural of counter is
    component t_flipflop is
        port( t, clk : in std_logic;
              q, q_bar : out std_logic);
    end component;

    signal C : std_logic_vector(N - 1 downto 0);
    signal B : std_logic_vector(N - 1 downto 0) := (others => '0');

    for all:t_flipflop use entity work.t_flipflop;
begin
    C(0) <= '1';
    c0: t_flipflop port map ('1', clk, B(0), open);

    cs: for I in 1 to N - 1 generate
        C(I) <= C(I - 1) and B(I - 1);
        cI: t_flipflop port map (C(I), clk, B(I), open);
    end generate;

    Bs: for I in 0 to N - 1 generate
        d(I) <= B(I);
    end generate;
end architecture structural;

```

```
        end generate;  
end architecture structural;
```