ECE-124 Lab-3 Submission Form – Winter 2018			
GROUP NUMBER: 15	Lab3 Demo	Lab3 Report	
SESSION NUMBER: 205	Out of 10	Out of 10	

NAME: (Print)	UW User ID (not Student ID)	Signature
Partner A: Yeliang Shou	y2shou	yeliang
Partner B: Yun Han Xiao	yhxiao	

LAB3 DESIGN DEMO	Marks Allotted	A	B
Desired Temp (sw[7..4]) is displayed on Digit1?	1	1	1
Current Temp (sw[3..0]) is displayed on Digit2?	1	1	1
Doors/Windows (PB[2..0]) displayed on LEDs[6..4]?	1	1	1
Furnace, Blower, System At Temp, A/C, Blower Indicators LEDs?	1	1	1
A/C ON & Blower ON when Current Temp > Desired Temp?	1	1	1
Furnace ON & Blower ON when Current Temp < Desired Temp?	1	1	1
A/C, Furnace, Blower turn OFF when Doors/Windows Open?	1	1	1
DISCUSSION: Comment on your VHDL Implementation?	3	3	3

LAB3 DESIGN REPORT (see rubric on LEARN for details)	Marks Allotted	TEAM
All VHDL files (not the sevenseg_decoder or seg7_mux files). Structural VHDL design must be used at the Top Level. Comparator must have a **Boolean equation** per each output.	2	
Truth Table for 4-Bit Comparator Design with all entries inserted	2	
Part A Simulations of Comparator showing A>B, A=B, A<B	2	
RTL View of the Logic design (just of the Top Level)	2	
Total Design Logic Elements Used /8064	2	
Delay in Report Submission (-1 per day) x number of days:		
LAB3 REPORT MARK	**Out of 10**	

LogicalStep_Lab3.vhd

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


entity LogicalStep_Lab3_top is port (
   clkin_50    : in  std_logic;
   pb          : in  std_logic_vector(3 downto 0);
   sw          : in  std_logic_vector(7 downto 0);
   leds        : out std_logic_vector(7 downto 0);
   seg7_data   : out std_logic_vector(6 downto 0);
   seg7_char1  : out std_logic;
   seg7_char2  : out std_logic

);
end LogicalStep_Lab3_top;

architecture Energy_Monitor of LogicalStep_Lab3_top is

-- Components Used

    component Compx4 port(
        A       :   in  std_logic_vector(3 downto 0);
        B       :   in  std_logic_vector(3 downto 0);
        GT      :   out std_logic;
        EQ      :   out std_logic;
        LE      :   out std_logic
    );
    end component;

    component SevenSegment port (
       hex     :  in  std_logic_vector(3 downto 0);
       sevenseg :  out std_logic_vector(6 downto 0)
    );
    end component;

   component segment7_mux port(
      clk         :  in  std_logic := '0';
      DIN2        :  in  std_logic_vector(6 downto 0);
      DIN1        :  in  std_logic_vector(6 downto 0);
      DOUT        :  out std_logic_vector(6 downto 0);
      DIG2        :  out std_logic;
      DIG1        :  out std_logic
   );
   end component;
```

In this first part, we declared the component including the 4 bits comparators (Compx4) and the muxes necessary for Character Display.

```
    component Thermostat port(
       CT :  in  std_logic_vector(3 downto 0);
       DT :  in  std_logic_vector(3 downto 0);
       OP :  in  std_logic_vector(2 downto 0);
       OTP : out std_logic_vector(6 downto 0)
    );
    end component;


-- Create any signals, or temporary variables to be used

    signal thermo_otp : std_logic_vector(6 downto 0);

    signal char1 : std_logic_vector(6 downto 0);
    signal char2 : std_logic_vector(6 downto 0);


-- Here the circuit begins

begin

    --Thermostat instance will implement the logic
    --    it also incorporate the comparator inside
    --    and output directly to the LEDS

    thermostatInstance : Thermostat port map( sw(3 downto 0),
                                              sw(7 downto 4),
                                              pb(2 downto 0),
                                              leds(6 downto 0));

     -- Displaying current and desired temperatures
     sevenmux1 : SevenSegment port map(sw(7 downto 4), char1);
     sevenmux2 : SevenSegment port map(sw(3 downto 0), char2);

     display : segment7_mux port map(
                                  clkin_50,
                                  char1,
                                  char2,
                                  seg7_data,
                                  seg7_char1,
                                  seg7_char2
                                  );


end Energy_Monitor;
```

Here, we declared a Thermostat component which will be a sub-module responsible for handling the logic. We instantiate the Thermostat by mapping our input switches [7..4] as desired temperature and sw[3..0] as current temperature (each a 4 bit input). The output of the Thermostat will be directly mapped to the appropriate LED through the 7 bit LED vector. We have also instantiated the necessary components for the Seven Seg Display.

Compx4.vhd

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


entity Compx4 is port(
    A       :   in  std_logic_vector(3 downto 0);
    B       :   in  std_logic_vector(3 downto 0);
    GT      :   out std_logic;
    EQ      :   out std_logic;
    LE      :   out std_logic
);
end Compx4;

architecture logicCompx4 of Compx4 is

    component Compx1 port(
        ai  :   in  std_logic;
        bi  :   in  std_logic;


        go  :   out std_logic;
        eo  :   out std_logic;
        lo  :   out std_logic
    );
    end component;

    -- For each bit's output by index

    signal g3, g2, g1, g0 : std_logic;
    signal e3, e2, e1, e0 : std_logic;
    signal l3, l2, l1, l0 : std_logic;


begin

   inst0   : Compx1 port map(A(0), B(0), g0, e0, l0);
   inst1   : Compx1 port map(A(1), B(1), g1, e1, l1);
   inst2   : Compx1 port map(A(2), B(2), g2, e2, l2);
   inst3   : Compx1 port map(A(3), B(3), g3, e3, l3);

    EQ  <= e0 and e1 and e2 and e3;
    GT  <= g3 or
                (e3 and g2) or
                (e3 and e2 and g1)  or
                (e3 and e2 and e1 and g0);
    LE  <= l3 or
            (e3 and l2) or
            (e3 and e2 and l1) or
            (e3 and e2 and e1 and l0);


end architecture logicCompx4;
```

Here, we applied the 1-bit comparator to each of the 4 bits in input A and B, we then applied the necessary logical operation to obtain an overall result on Greater or Equal or Less.

Compx1.vhd

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

entity Compx1 is port(
    ai      :   in  std_logic;
    bi      :   in  std_logic;

    go      :   out std_logic;
    eo      :   out std_logic;
    lo      :   out std_logic
);
end Compx1;

architecture logicCompx1 of Compx1 is

begin

    eo <= (ai and bi) or (not(ai) and not(bi));
    go <= ai and not(bi);
    lo <= not(ai) and bi;

end architecture logicCompx1;
```

The one bit comparator is straightforward and trivial, if both bits are one or both are zero, then they are equal, if the A bit is one while the B is zero, we then have greater, conversely we will output "Less".

Table 1: Logic Table for 4-bit Magnitude Comparator

A3 < B3	A3 = B3	A3 > B3	A2 < B2	A2 = B2	A2 > B2	A1 < B1	A1 = B1	A1 > B1	A0 < B0	A0 = B0	A0 > B0	A < B	A = B	A > B
0	0	1	X	X	X	X	X	X	X	X	X	0	0	1
1	0	0	X	X	X	X	X	X	X	X	X	1	0	0
0	1	0	0	0	1	X	X	X	X	X	X	0	0	1
0	1	0	1	0	0	X	X	X	X	X	X	1	0	0
0	1	0	0	1	0	0	0	1	X	X	X	0	0	1
0	1	0	0	1	0	1	0	0	X	X	X	1	0	0
0	1	0	0	1	0	0	1	0	0	0	1	0	0	1
0	1	0	0	1	0	0	1	0	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0	0	1	0	0	1	0

Thermostat.vhd

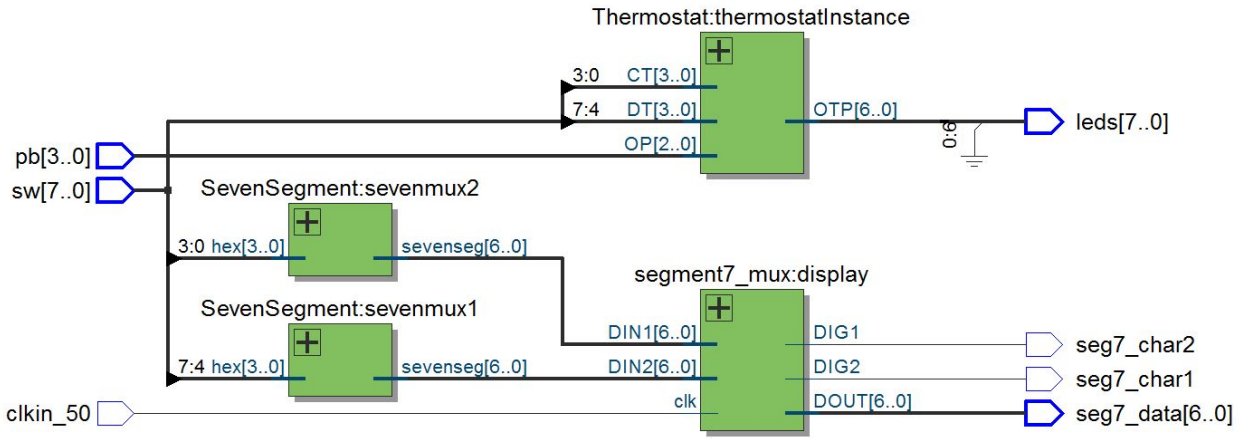
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;


entity Thermostat is port(
    CT :  in  std_logic_vector(3 downto 0);
    DT :  in  std_logic_vector(3 downto 0);
    OP :  in  std_logic_vector(2 downto 0);

    OTP : out std_logic_vector(6 downto 0)

);
end Thermostat;

architecture Logic of Thermostat is

    component Compx4 port(
         A       :   in  std_logic_vector(3 downto 0);
         B       :   in  std_logic_vector(3 downto 0);
         GT      :   out std_logic;
         EQ      :   out std_logic;
         LE      :   out std_logic
     );
     end component;

     signal gt, eq, le : std_logic;
     signal tmp : std_logic_vector(2 downto 0);
     signal all_closed : std_logic;
     signal ctrl: std_logic_vector(3 downto 0);
     signal doors: std_logic_vector(2 downto 0);

begin

    compx4inst : Compx4 port map(CT, DT, gt, eq, le);
    -- Comparing current temp with desired temp
    tmp <= gt & eq & le;
    doors <= not(OP(2)) & not(OP(1)) & not(OP(0));
    all_closed <= OP(2) and OP(1) and OP(0);
    -- Since PBs are active-low, we don't need to NAND it
    with tmp&all_closed select
              --LEDS[7..0]          tmp : gt&eq&le&ALL_CLOSED
    ctrl  <=     "1001"     when          "0011",  -- Furnace ON
                 "1100"     when          "1001",  -- A/C ON
                 "0010"     when          "0101",
                 "0010"     when          "0100",
                 "0000"     when          others;
    OTP <= doors & ctrl;
end architecture Logic;
```

The Thermostat logic first compares desired and current temperature (each a 4 bit vector) using the 4 bit comparator declared earlier. We also inspect if all the windows/door are closed, to do so is fairly trivial, consider that the PBs are active-low, by applying AND to all 3 PB inputs, if such returns True, then we know all doors/windows are closed. For

implementing the logic of Thermostat, we used a straightforward case by case approach with our select statements.

Wave Simulation for 4-Bits Comparator

RTL Overview of LogicStep_3.vhd



Number of Logical Elements Used: 47 /8064

Flow Status	Successful - Thu Mar 08 14:44:57 2018
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Standard Edition
Revision Name	LogicalStep_Lab3_top
Top-level Entity Name	LogicalStep_Lab3_top
Family	MAX 10
Device	10M08SAE144C8G
Timing Models	Final
Total logic elements	47 / 8,064 (< 1 %)
Total combinational functions	47 / 8,064 (< 1 %)
Dedicated logic registers	11 / 8,064 (< 1 %)
Total registers	11
Total pins	30 / 101 (30 %)
Total virtual pins	0
Total memory bits	0 / 387,072 (0 %)
Embedded Multiplier 9-bit elements	0 / 48 (0 %)
Total PLLs	0 / 1 (0 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 1 (0 %)