ECE-124 Lab-2 Submission Form – Winter 2018

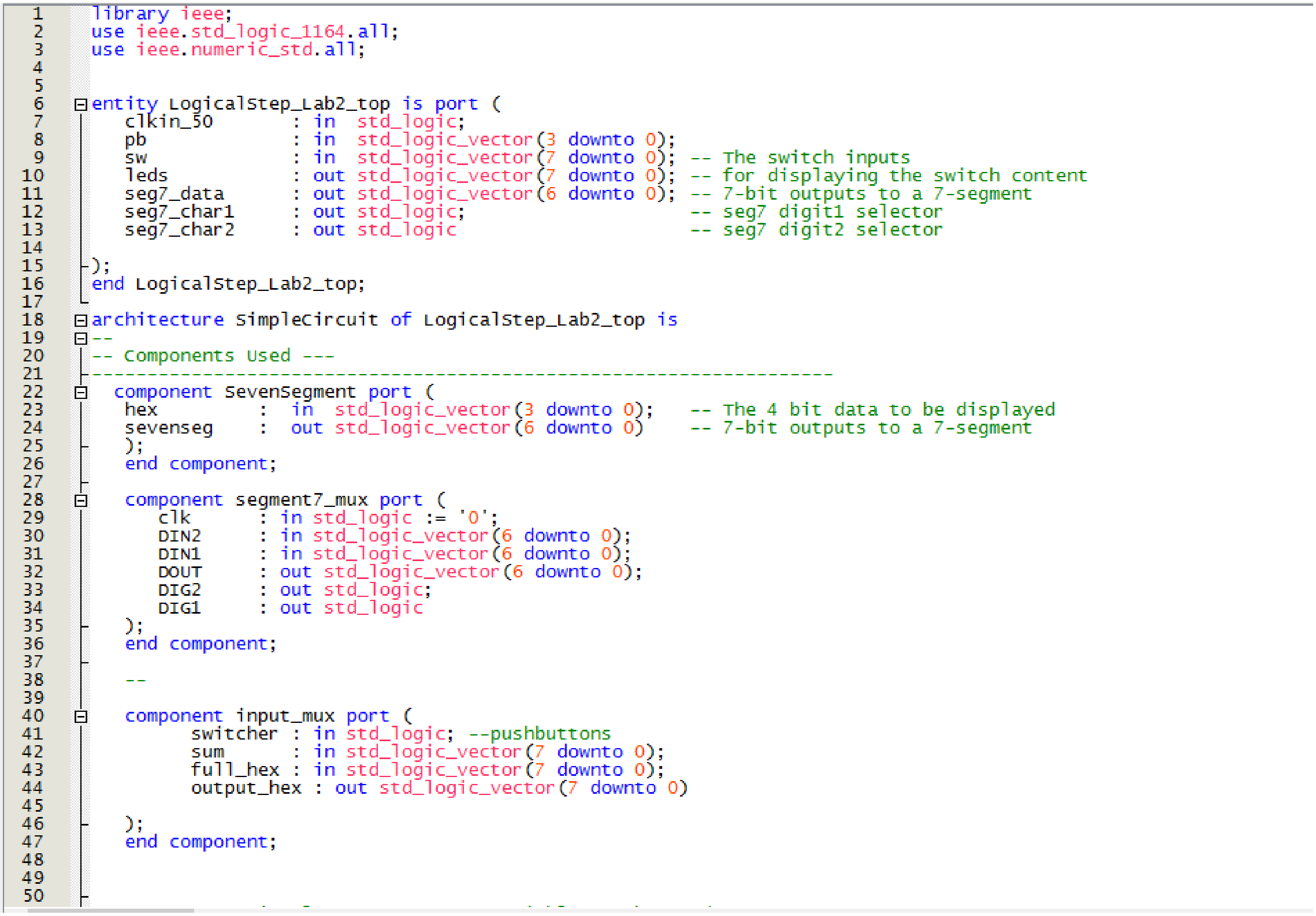
GROUP NUMBER: 20		Sushaant.
SESSION NUMBER: 201		

NAME: (Print)	UW User ID (not Student ID)	Signature
Partner A: Wesley Barton	whbarton	wesley Barton
Partner B: SidharthBaveja	sbaveja	Sid

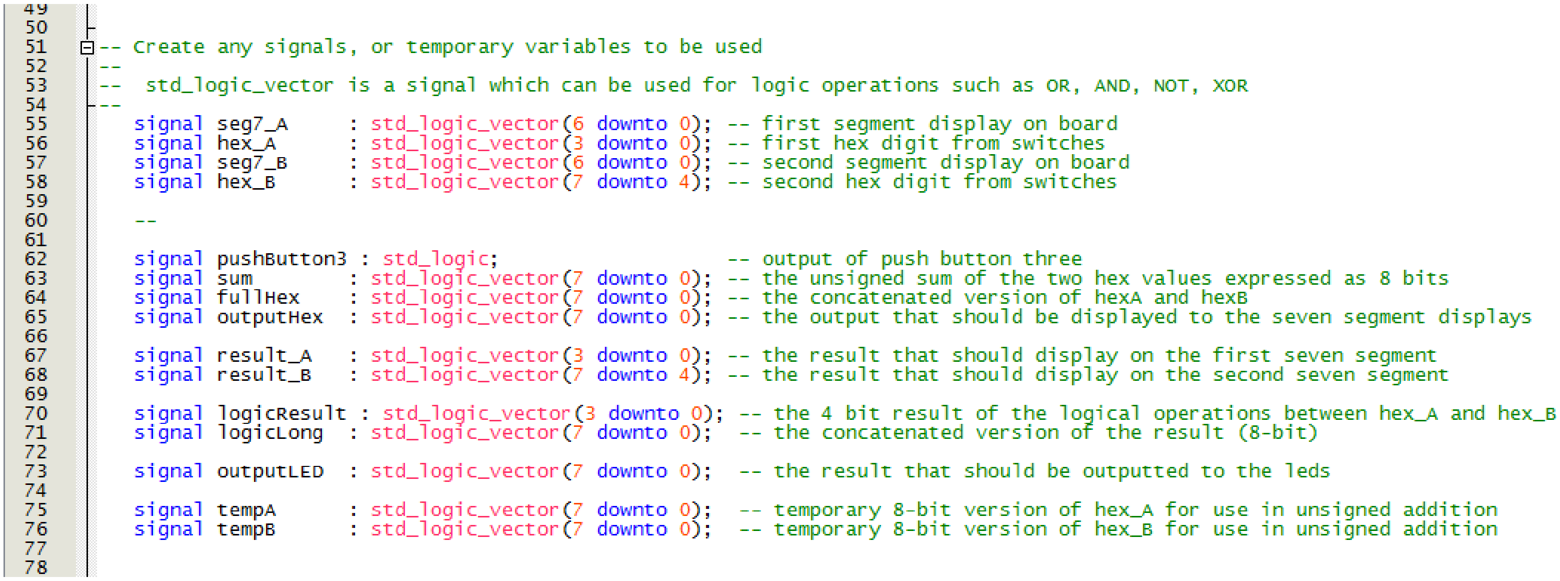
LAB2 DESIGN DEMO	Marks Allotted	A	B
Seven Segment Display bugs (quantity 3) corrected ?	1	1	1
Operands appear on Digit1 & Digit2 when PB's are OFF ?	1	1	1
Logical Results shown correctly on LEDs[3..0] when PB[2..0] ON ?	1	1	1
Arithmetic results shown on Digits and LED's when PB(3) ON ?	2	2	2
LEDs[7..4] OFF when Arithmetic result Less than or Equal to 1111	2	2	2
DISCUSSION: Describe how you implemented the VHDL coding.	3	3	3
LAB2 DEMO MARK		10	10

LAB2 DESIGN REPORT (see rubric on LEARN for details)	Marks Allotted	
Structural VHDL Used in top level VHDL design	2	
Sub-block VHDL files with good Coding Style	2	
Simulation of Logic functions showing the AND,OR,XOR modes	2	
Simulation of Arithmetic functions showing the ADD mode	2	
Total Design Logic Elements Used from Compilation Report	2	
Delay in Report Submission (-1 per day) x number of days:		
LAB2 Report MARK	**Out of 10**	

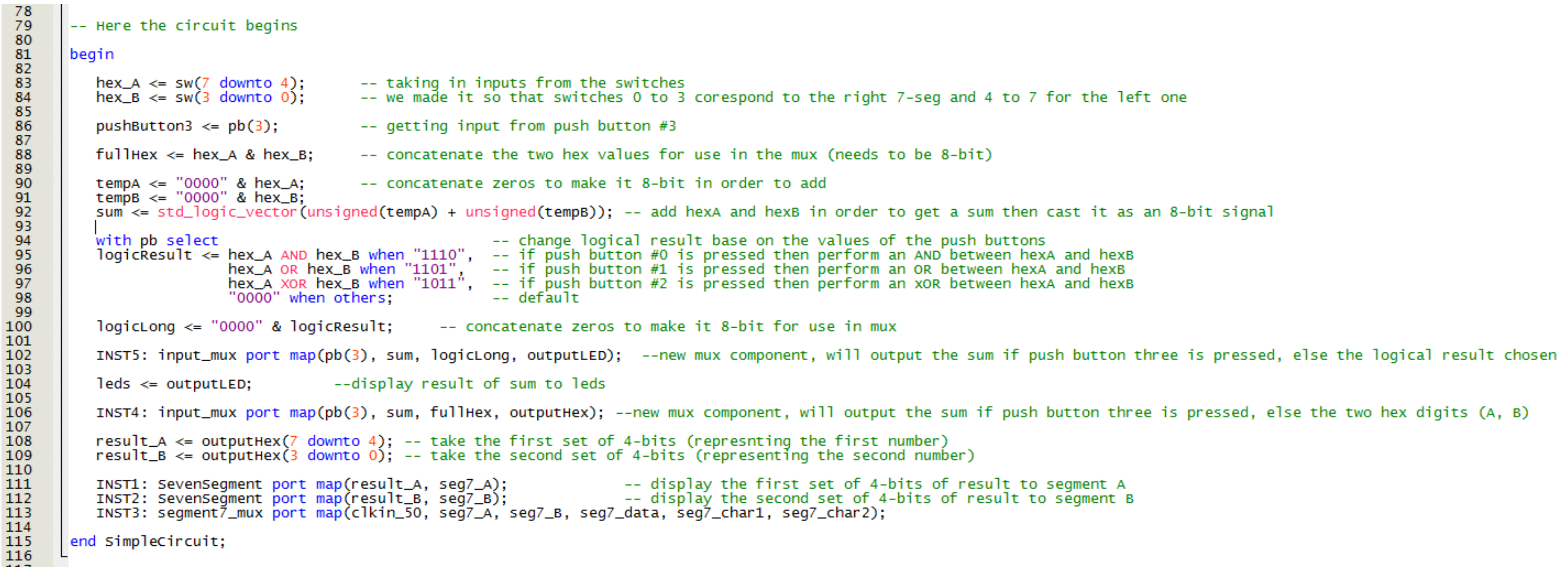
Structural VHDL in Top Level:



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


entity LogicalStep_Lab2_top is port (
   clkin_50        : in  std_logic;
   pb              : in  std_logic_vector(3 downto 0);
   sw              : in  std_logic_vector(7 downto 0); -- The switch inputs
   leds            : out std_logic_vector(7 downto 0); -- for displaying the switch content
   seg7_data       : out std_logic_vector(6 downto 0); -- 7-bit outputs to a 7-segment
   seg7_char1      : out std_logic;                    -- seg7 digit1 selector
   seg7_char2      : out std_logic                     -- seg7 digit2 selector

);
end LogicalStep_Lab2_top;

architecture SimpleCircuit of LogicalStep_Lab2_top is
--
-- Components Used ---
-------------------------------------------------------------------------------
  component SevenSegment port (
   hex          :  in  std_logic_vector(3 downto 0);   -- The 4 bit data to be displayed
   sevenseg     :  out std_logic_vector(6 downto 0)    -- 7-bit outputs to a 7-segment
   );
   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;

   --


   component input_mux port (
         switcher  : in std_logic; --pushbuttons
         sum       : in std_logic_vector(7 downto 0);
         full_hex  : in std_logic_vector(7 downto 0);
         output_hex : out std_logic_vector(7 downto 0)

   );
   end component;
```

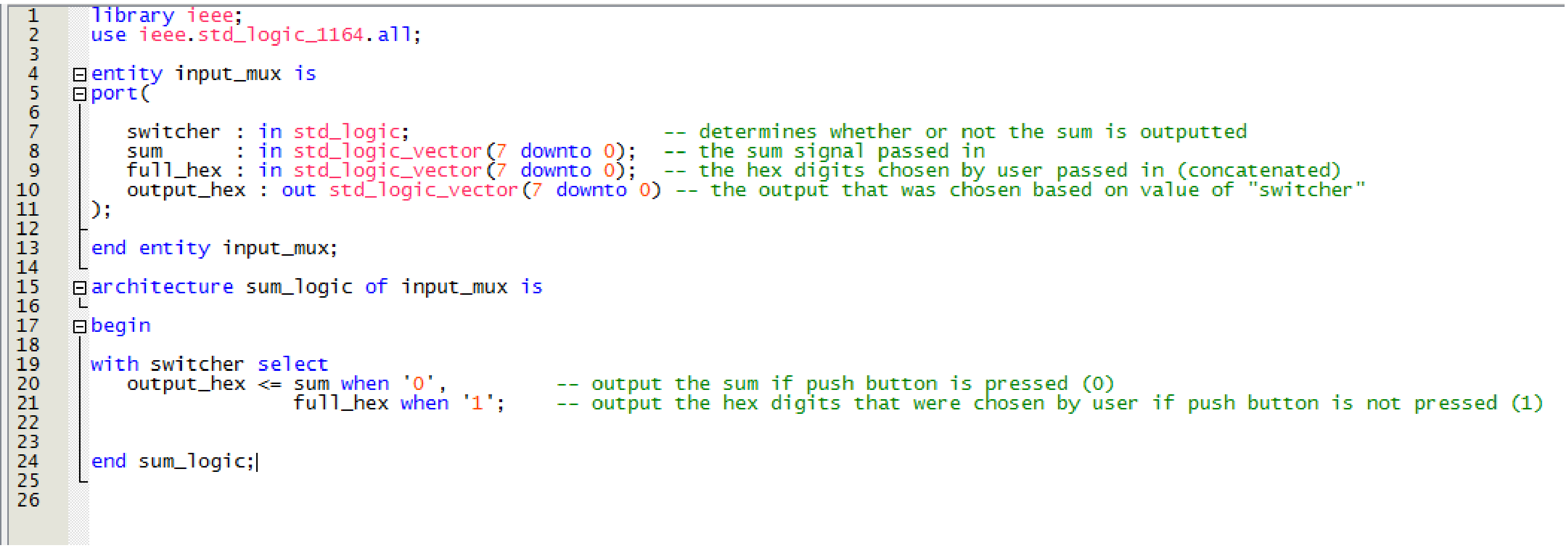


```

-- Create any signals, or temporary variables to be used
--
--  std_logic_vector is a signal which can be used for logic operations such as OR, AND, NOT, XOR
--
   signal seg7_A      : std_logic_vector(6 downto 0); -- first segment display on board
   signal hex_A       : std_logic_vector(3 downto 0); -- first hex digit from switches
   signal seg7_B      : std_logic_vector(6 downto 0); -- second segment display on board
   signal hex_B       : std_logic_vector(7 downto 4); -- second hex digit from switches

   --

   signal pushButton3 : std_logic;                    -- output of push button three
   signal sum         : std_logic_vector(7 downto 0); -- the unsigned sum of the two hex values expressed as 8 bits
   signal fullHex     : std_logic_vector(7 downto 0); -- the concatenated version of hexA and hexB
   signal outputHex   : std_logic_vector(7 downto 0); -- the output that should be displayed to the seven segment displays

   signal result_A    : std_logic_vector(3 downto 0); -- the result that should display on the first seven segment
   signal result_B    : std_logic_vector(7 downto 4); -- the result that should display on the second seven segment

   signal logicResult : std_logic_vector(3 downto 0);  -- the 4 bit result of the logical operations between hex_A and hex_B
   signal logicLong   : std_logic_vector(7 downto 0);  -- the concatenated version of the result (8-bit)

   signal outputLED   : std_logic_vector(7 downto 0);  -- the result that should be outputted to the leds

   signal tempA       : std_logic_vector(7 downto 0);  -- temporary 8-bit version of hex_A for use in unsigned addition
   signal tempB       : std_logic_vector(7 downto 0);  -- temporary 8-bit version of hex_B for use in unsigned addition

```



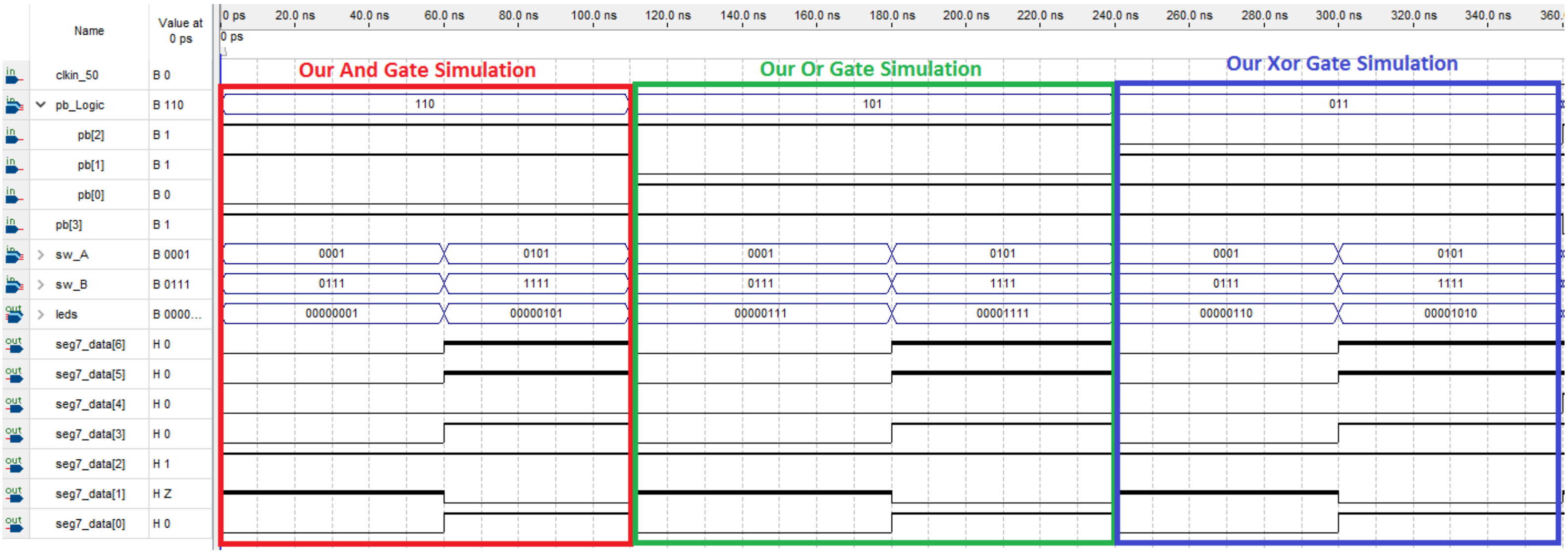
```vhdl

-- Here the circuit begins

begin

   hex_A <= sw(7 downto 4);       -- taking in inputs from the switches
   hex_B <= sw(3 downto 0);       -- we made it so that switches 0 to 3 corespond to the right 7-seg and 4 to 7 for the left one

   pushButton3 <= pb(3);          -- getting input from push button #3

   fullHex <= hex_A & hex_B;      -- concatenate the two hex values for use in the mux (needs to be 8-bit)

   tempA <= "0000" & hex_A;       -- concatenate zeros to make it 8-bit in order to add
   tempB <= "0000" & hex_B;
   sum <= std_logic_vector(unsigned(tempA) + unsigned(tempB)); -- add hexA and hexB in order to get a sum then cast it as an 8-bit signal

   with pb select                                -- change logical result base on the values of the push buttons
   logicResult <= hex_A AND hex_B when "1110",   -- if push button #0 is pressed then perform an AND between hexA and hexB
                  hex_A OR hex_B when "1101",    -- if push button #1 is pressed then perform an OR between hexA and hexB
                  hex_A XOR hex_B when "1011",   -- if push button #2 is pressed then perform an XOR between hexA and hexB
                  "0000" when others;            -- default

   logicLong <= "0000" & logicResult;     -- concatenate zeros to make it 8-bit for use in mux

   INST5: input_mux port map(pb(3), sum, logicLong, outputLED);  --new mux component, will output the sum if push button three is pressed, else the logical result chosen

   leds <= outputLED;          --display result of sum to leds

   INST4: input_mux port map(pb(3), sum, fullHex, outputHex); --new mux component, will output the sum if push button three is pressed, else the two hex digits (A, B)

   result_A <= outputHex(7 downto 4); -- take the first set of 4-bits (represnting the first number)
   result_B <= outputHex(3 downto 0); -- take the second set of 4-bits (representing the second number)

   INST1: SevenSegment port map(result_A, seg7_A);              -- display the first set of 4-bits of result to segment A
   INST2: SevenSegment port map(result_B, seg7_B);              -- display the second set of 4-bits of result to segment B
   INST3: segment7_mux port map(clkin_50, seg7_A, seg7_B, seg7_data, seg7_char1, seg7_char2);

end SimpleCircuit;

```

Our Mux:



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

entity input_mux is
port(

    switcher : in std_logic;                        -- determines whether or not the sum is outputted
    sum      : in std_logic_vector(7 downto 0);     -- the sum signal passed in
    full_hex : in std_logic_vector(7 downto 0);     -- the hex digits chosen by user passed in (concatenated)
    output_hex : out std_logic_vector(7 downto 0) -- the output that was chosen based on value of "switcher"
);

end entity input_mux;

architecture sum_logic of input_mux is

begin

with switcher select
    output_hex <= sum when '0',          -- output the sum if push button is pressed (0)
                  full_hex when '1';     -- output the hex digits that were chosen by user if push button is not pressed (1)


end sum_logic;

```

Logic Simulations:

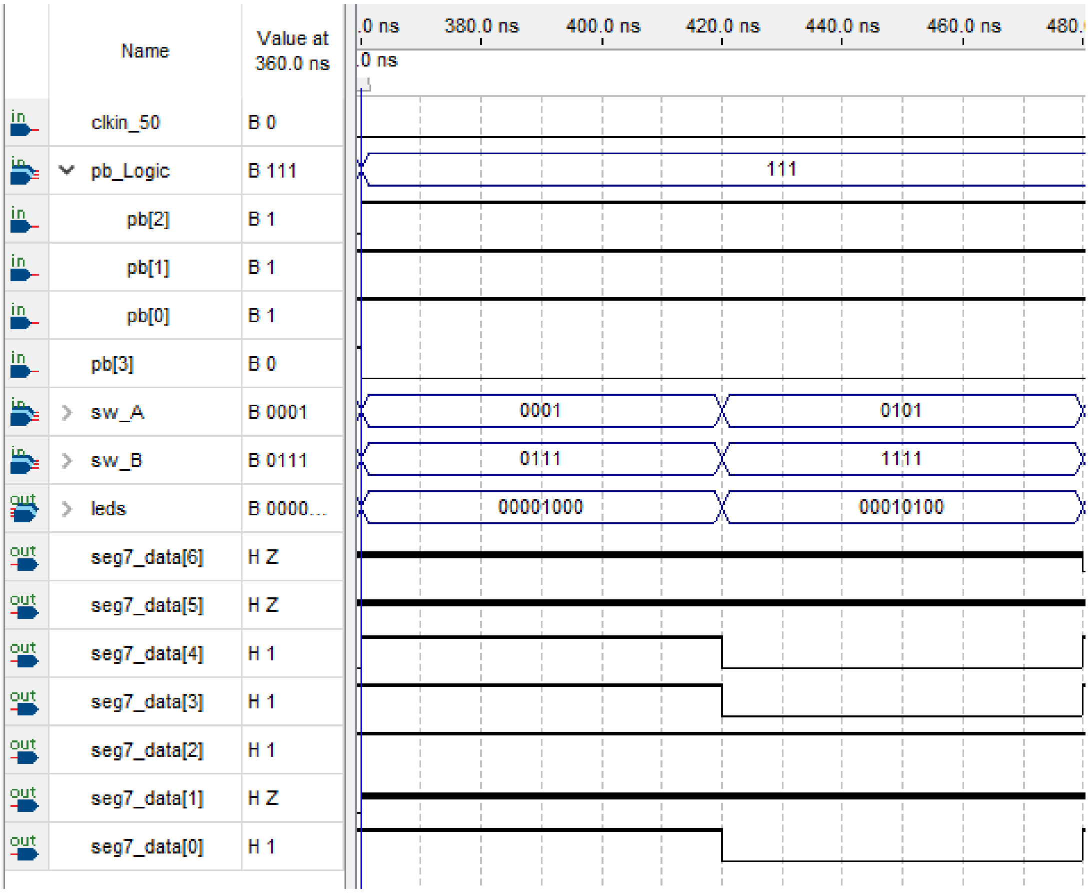


Name	Value at 0 ps
clkin_50	B 0
pb_Logic	B 110
pb[2]	B 1
pb[1]	B 1
pb[0]	B 0
pb[3]	B 1
sw_A	B 0001
sw_B	B 0111
leds	B 0000...
seg7_data[6]	H 0
seg7_data[5]	H 0
seg7_data[4]	H 0
seg7_data[3]	H 0
seg7_data[2]	H 1
seg7_data[1]	H Z
seg7_data[0]	H 0

- sw_A: switches 7 downto 4
- sw_B : switches 3 downto 0

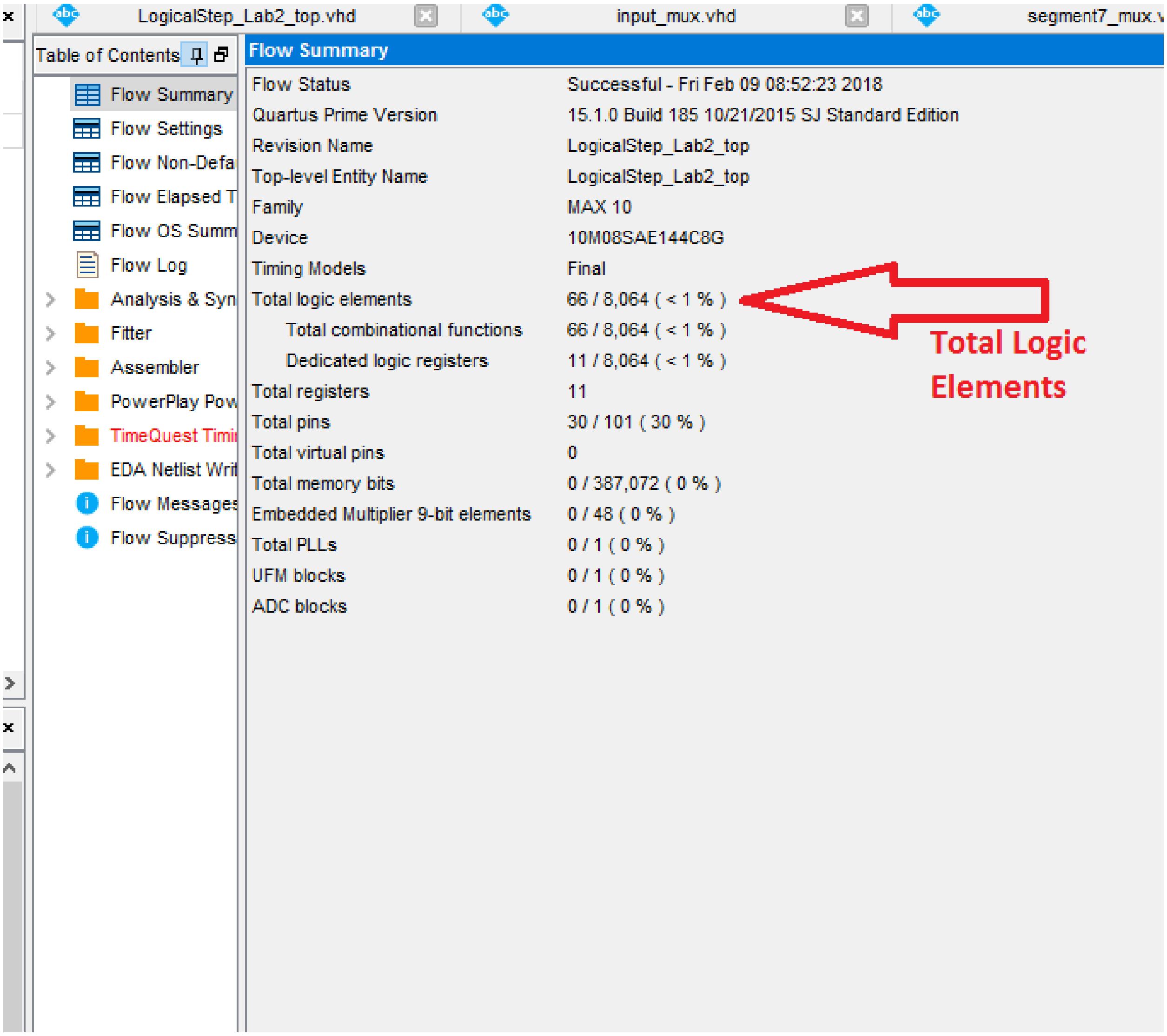
Arithmetic Simulations:

- Pushbuttons are active-low



Name	Value at 360.0 ns
clkin_50	B 0
pb_Logic	B 111
pb[2]	B 1
pb[1]	B 1
pb[0]	B 1
pb[3]	B 0
sw_A	B 0001
sw_B	B 0111
leds	B 0000...
seg7_data[6]	H Z
seg7_data[5]	H Z
seg7_data[4]	H 1
seg7_data[3]	H 1
seg7_data[2]	H 1
seg7_data[1]	H Z
seg7_data[0]	H 1

Compilation Report:

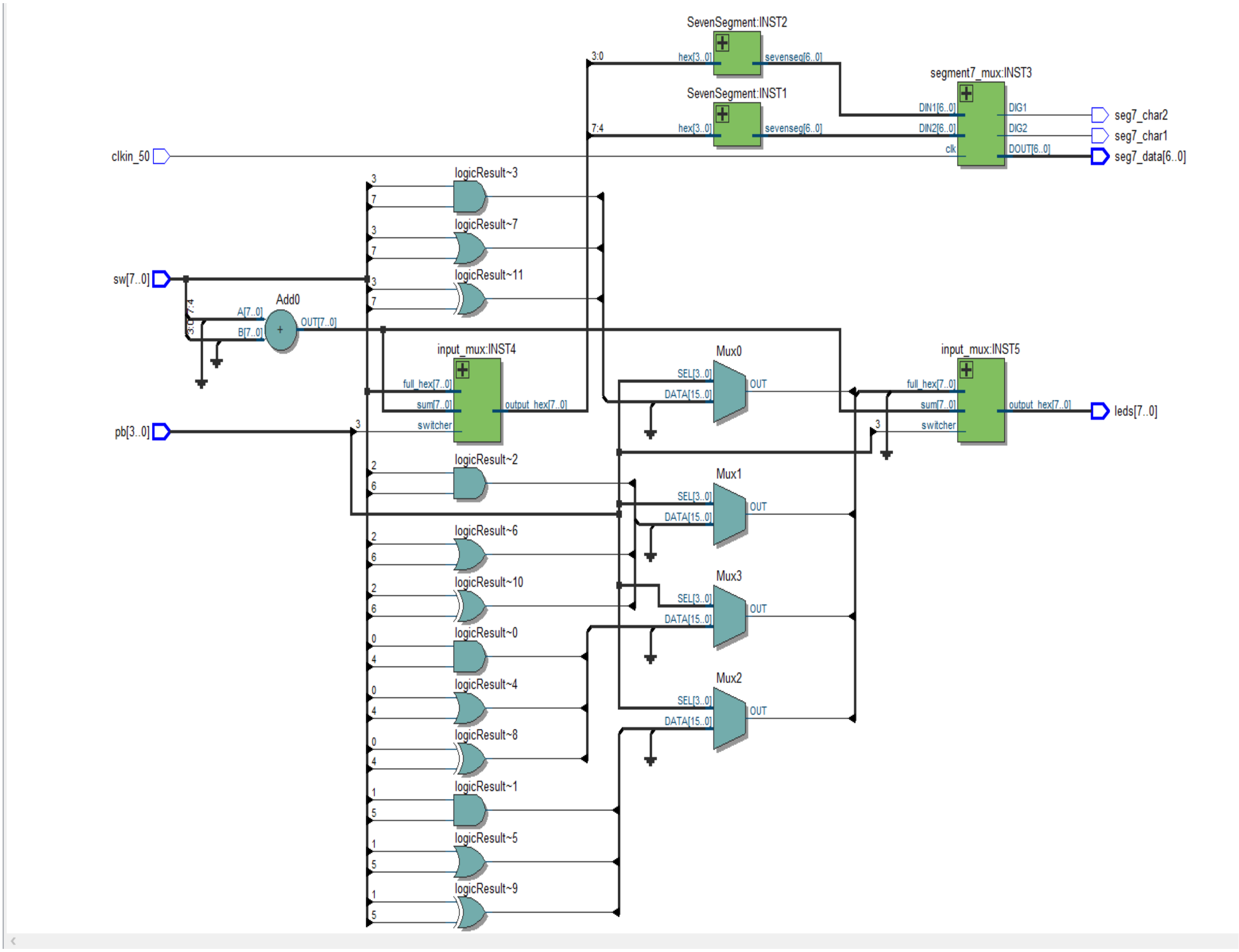


Flow Summary

Flow Status	Successful - Fri Feb 09 08:52:23 2018
Quartus Prime Version	15.1.0 Build 185 10/21/2015 SJ Standard Edition
Revision Name	LogicalStep_Lab2_top
Top-level Entity Name	LogicalStep_Lab2_top
Family	MAX 10
Device	10M08SAE144C8G
Timing Models	Final
Total logic elements	66 / 8,064 (< 1 %)
Total combinational functions	66 / 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 %)

Total Logic Elements

RTL Diagram:


SevenSegment:INST2
3:0
hex[3..0]
sevenseg[6..0]
segment7_mux:INST3
SevenSegment:INST1
DIN1[6..0]
DIG1
seg7_char2
7:4
hex[3..0]
sevenseg[6..0]
DIN2[6..0]
DIG2
seg7_char1
clkin_50
clk
DOUT[6..0]
seg7_data[6..0]
logicResult~3
logicResult~7
logicResult~11
sw[7..0]
Add0
A[7..0]
B[7..0]
OUT[7..0]
3:0
7:4
input_mux:INST4
full_hex[7..0]
sum[7..0]
switcher
output_hex[7..0]
Mux0
SEL[3..0]
DATA[15..0]
OUT
input_mux:INST5
full_hex[7..0]
sum[7..0]
switcher
output_hex[7..0]
leds[7..0]
pb[3..0]
logicResult~2
Mux1
logicResult~6
logicResult~10
Mux3
logicResult~0
logicResult~4
Mux2
logicResult~8
logicResult~1
logicResult~5
logicResult~9