```
1  ----------------------------------------------------------------------------------
2  -- Company: HSRW
3  -- Engineer: Andy Stamm
4  --
5  -- Create Date:    08:49:12 12/17/2015
6  -- Design Name:
7  -- Module Name:    7 segment 4bit to decimal - Behavioral
8  -- Project Name:     Lab 2.1
9  -- Target Devices:
10 -- Tool versions:
11 -- Description:
12 --
13 -- Dependencies:
14 --
15 -- Revision:
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 ----------------------------------------------------------------------------------
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22 use IEEE.STD_LOGIC_ARITH.ALL;
23 use IEEE.STD_LOGIC_UNSIGNED.ALL;
24
25 -- Uncomment the following library declaration if using
26 -- arithmetic functions with Signed or Unsigned values
27 --use IEEE.NUMERIC_STD.ALL;
28
29 -- Uncomment the following library declaration if instantiating
30 -- any Xilinx primitives in this code.
31 --library UNISIM;
32 --use UNISIM.VComponents.all;
33
34 library IEEE;
35 use IEEE.STD_LOGIC_1164.ALL;
36 use IEEE.STD_LOGIC_ARITH.ALL;
37 use IEEE.STD_LOGIC_UNSIGNED.ALL;
38
39 entity seven_segment is
40        Port (  clk, SW0, SW1, SW2, SW3  : in     STD_LOGIC;                      --
   inputs to the entity
41                COM0, COM1 : out  STD_LOGIC;
                                                --ouput anode of the seven segment display
42                SEGA0, SEGB0, SEGC0, SEGD0, SEGE0, SEGF0, SEGG0 : out  STD_LOGIC;
   -- ouputs of the first seven segment dislay
43                SEGA1, SEGB1, SEGC1, SEGD1, SEGE1, SEGF1, SEGG1 : out  STD_LOGIC  --
   ouputs of the second seven segment dislay
44              );
45 end entity seven_segment;
46
47 architecture Behavioral of seven_segment is
48 signal temp: std_logic_vector(13 downto 0);  -- vector to drive the 7-segment display
49 signal btn: std_logic_vector(3 downto 0);    -- vector to save the switch state
50 signal number: integer;                      -- variable to save the integer number
51     begin
52        process(clk)
53            begin
```

```
54              if clk'event and clk='1' then -- wait for the rising edge of the clock
    then assign SW state to vector
55                 btn(0)<=SW0;
56                 btn(1)<=SW1;
57                 btn(2)<=SW2;
58                 btn(3)<=SW3;
59              end if;
60          end process;
61
62       process
63          begin
64             -- comparing if the number is larger then 9
65             number <= conv_integer(unsigned(btn)); -- converting the bit vector to an
    integer
66             if number > 9 then   -- check if we need the second 7-segment display
67                COM1<='0';           -- enable the second 7-segment display
68             else
69                COM1<='1';           -- disable the second 7-segment display
70             end if;
71
72             -- assign the decoded output to the first seven segment display outputs
73             SEGA0 <= temp(0);
74             SEGB0 <= temp(1);
75             SEGC0 <= temp(2);
76             SEGD0 <= temp(3);
77             SEGE0 <= temp(4);
78             SEGF0 <= temp(5);
79             SEGG0 <= temp(6);
80             COM0<='0';        -- enable the first seven segment display
81
82             -- assign the decoded output to the second seven segment display outputs
83             SEGA1 <= temp(7);
84             SEGB1 <= temp(8);
85             SEGC1 <= temp(9);
86             SEGD1 <= temp(10);
87             SEGE1 <= temp(11);
88             SEGF1 <= temp(12);
89             SEGG1 <= temp(13);
90       end process;
91
92             -- decoding the four bit vector to the first seven segment display
93             with number Select
94                temp<="00000000111111" when 0,
95                      "00000000000110" when 1,
96                      "00000001011011" when 2,
97                      "00000001001111" when 3,
98                      "00000001100110" when 4,
99                      "00000001101101" when 5,
100                     "00000001111101" when 6,
101                     "00000000000111" when 7,
102                     "00000001111111" when 8,
103                     "00000001101111" when 9,
104                     "00001100111111" when 10,
105                     "00001100000110" when 11,
106                     "00001101011011" when 12,
107                     "00001101001111" when 13,
108                     "00001101100110" when 14,
```

```
109                     "00001101101101" when 15,
110                     "10000001000000" when others;
111
112
113   end Behavioral;
114
```