

```

1  -----
2  -- Company:
3  -- Engineer:
4  --
5  -- Create Date: 11/16/2017 02:45:02 PM
6  -- Design Name:
7  -- Module Name: top - Behavioral
8  -- Project Name:
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
21
22 library IEEE;
23 use IEEE.STD_LOGIC_1164.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 leaf cells in this code.
31 --library UNISIM;
32 --use UNISIM.VComponents.all;
33
34 entity top is
35     Port (
36         sw : in std_logic_vector(15 downto 0);
37         clk : in std_logic;
38         seg : out std_logic_vector(6 downto 0);
39         dp : out std_logic;
40         an : out std_logic_vector(3 downto 0);
41         led : out std_logic_vector(15 downto 0) );
42 end top;
43
44 architecture Behavioral of top is
45     component ssd_muxer is
46         Port (
47             a_in : in std_logic_vector(3 downto 0);
48             b_in : in std_logic_vector(3 downto 0);
49             c_in : in std_logic_vector(3 downto 0);
50             d_in : in std_logic_vector(3 downto 0);
51             e_in : in std_logic_vector(3 downto 0);
52             f_in : in std_logic_vector(3 downto 0);
53             g_in : in std_logic_vector(3 downto 0);
54             decp0_in : in std_logic;
55             decp1_in : in std_logic;
56             decp2_in : in std_logic;
57             decp3_in : in std_logic;
58             seg_out : out std_logic_vector(6 downto 0);
59             dp_out : out std_logic;
60             an_out : out std_logic_vector(3 downto 0);
61             clk : in STD_LOGIC
62         );
63     end component;
64     signal a_in, b_in, c_in, d_in, e_in, f_in, g_in : std_logic_vector(3 downto 0);
65     signal seg_out : std_logic_vector(6 downto 0);
66     signal dp_out : std_logic;

```

```

67     signal an_out : std_logic_vector(3 downto 0);
68
69     component encoder
70     Port (
71         hex_in : in  STD_LOGIC_VECTOR(3 DOWNT0 0);
72         a      : out STD_LOGIC;
73         b      : out STD_LOGIC;
74         c      : out STD_LOGIC;
75         d      : out STD_LOGIC;
76         e      : out STD_LOGIC;
77         f      : out STD_LOGIC;
78         g      : out STD_LOGIC
79     );
80     end component;
81
82     -- signal st, ed : integer := 0;
83
84 begin
85     seg <= seg_out;
86     dp <= dp_out;
87     an <= an_out;
88     led <= sw;
89
90     SSD_MUX: ssd_muxer port map(
91         a_in => a_in,
92         b_in => b_in,
93         c_in => c_in,
94         d_in => d_in,
95         e_in => e_in,
96         f_in => f_in,
97         g_in => g_in,
98
99         decp0_in => '0',
100        decp1_in => '0',
101        decp2_in => '0',
102        decp3_in => '0',
103
104        seg_out => seg_out,
105        dp_out => dp_out,
106        an_out => an_out,
107
108        clk => clk
109    );
110
111    --      -- https://www.ics.uci.edu/~jmoorkan/vhdlref/generate.html
112    --      REG_ENC: for I in 0 to 3 generate
113    --      --
114    --      https://stackoverflow.com/questions/10375858/how-to-slice-an-std-logic-vector-in-vhdl
115    --      REG_PMP: encoder port map(
116    --          hex_in => sw((I * 4) downto (((I + 1)) * 4 - 1)),
117    --          a => a_in(I),
118    --          b => b_in(I),
119    --          c => c_in(I),
120    --          d => d_in(I),
121    --          e => e_in(I),
122    --          f => f_in(I),
123    --          g => g_in(I)
124    --      );
125    --      end generate;
126
127    REG1: encoder port map(
128        hex_in => sw(15 downto 12),
129        a => a_in(3),
130        b => b_in(3),
131        c => c_in(3),

```

```
132         d => d_in(3),
133         e => e_in(3),
134         f => f_in(3),
135         g => g_in(3)
136     );
137
138     REGE2: encoder port map(
139         hex_in => sw(11 downto 8),
140         a => a_in(2),
141         b => b_in(2),
142         c => c_in(2),
143         d => d_in(2),
144         e => e_in(2),
145         f => f_in(2),
146         g => g_in(2)
147     );
148
149     REGE3: encoder port map(
150         hex_in => sw(7 downto 4),
151         a => a_in(1),
152         b => b_in(1),
153         c => c_in(1),
154         d => d_in(1),
155         e => e_in(1),
156         f => f_in(1),
157         g => g_in(1)
158     );
159
160     REGE4: encoder port map(
161         hex_in => sw(3 downto 0),
162         a => a_in(0),
163         b => b_in(0),
164         c => c_in(0),
165         d => d_in(0),
166         e => e_in(0),
167         f => f_in(0),
168         g => g_in(0)
169     );
170
171     end Behavioral;
172
```