```vhdl
-----------------------------------------
-- Block code:  hex2sevseg_w_control.vhd
-- History:     24.Sep.2013 - 1st version for lab5 (dqtm)
--              24.Sep.2013 - add ctrl inputs,lab6 (dqtm)
-- Function: Hexa to seven-seg converter
--            plain functionality plus control inputs
--            implemented with comb logic inside process
-----------------------------------------

-- Library & Use Statements
LIBRARY ieee;
use ieee.std_logic_1164.all;

-- Entity Declaration
ENTITY hex2sevseg_w_control IS
  PORT(
        blank_n_i:          IN  std_logic;
        lamp_test_n_i:      IN  std_logic;
        ripple_blank_n_i:   IN  std_logic;
        hexa_i:             IN  std_logic_vector(3 downto 0);
        ripple_blank_n_o:   OUT std_logic;
        seg_o:              OUT std_logic_vector(6 downto 0)); -- Sequence is
        "gfedcba" (MSB is seg_g)
END hex2sevseg_w_control ;


-- Architecture Declaration
ARCHITECTURE rtl OF hex2sevseg_w_control IS

    -- Signals & Constants Declaration
    CONSTANT display_0      : std_logic_vector(6 downto 0):= "0111111";
    CONSTANT display_1      : std_logic_vector(6 downto 0):= "0000110";
    CONSTANT display_2      : std_logic_vector(6 downto 0):= "1011011";
    CONSTANT display_3      : std_logic_vector(6 downto 0):= "1001111";
    CONSTANT display_4      : std_logic_vector(6 downto 0):= "1100110";
    CONSTANT display_5      : std_logic_vector(6 downto 0):= "1101101";
    CONSTANT display_6      : std_logic_vector(6 downto 0):= "1111101";
    CONSTANT display_7      : std_logic_vector(6 downto 0):= "0000111";
    CONSTANT display_8      : std_logic_vector(6 downto 0):= "1111111";
    CONSTANT display_9      : std_logic_vector(6 downto 0):= "1101111";
    CONSTANT display_A      : std_logic_vector(6 downto 0):= "1110111";
    CONSTANT display_B      : std_logic_vector(6 downto 0):= "1111100";
    CONSTANT display_C      : std_logic_vector(6 downto 0):= "0111001";
    CONSTANT display_D      : std_logic_vector(6 downto 0):= "1011110";
    CONSTANT display_E      : std_logic_vector(6 downto 0):= "1111001";
    CONSTANT display_F      : std_logic_vector(6 downto 0):= "1110001";
    CONSTANT display_blank  : std_logic_vector(6 downto 0):= (others =>'0');


-- Begin Architecture
BEGIN

    -----------------------------------------
    -- Process for combinatorial logic
    -----------------------------------------
    sevseg_comb: PROCESS (ALL)
    BEGIN
    -- Default statement for ripple_blank output= inactive
    ripple_blank_n_o <= '1';

        -- Control Inputs from high to low priority:
        --           blank; lamp_test; ripple_blank
        --------------------------------
        IF (blank_n_i='0') THEN
            seg_o <= NOT(display_blank);

        ELSIF (lamp_test_n_i = '0') THEN
            seg_o <= NOT(display_8);

        ELSE
        -- Hexa input values 1-F not affected by ripple_blank
```

```
71          -- For hexa input x0, check status of ripple_blank ctrl
72              CASE hexa_i IS
73                  WHEN x"0" =>
74                      IF (ripple_blank_n_i = '0') THEN
75                          seg_o <= NOT(display_blank);
76                          ripple_blank_n_o <= '0';
77                      ELSE
78                          seg_o <= NOT(display_0);
79                      END IF;
80
81                  WHEN x"1" =>   seg_o <= NOT(display_1);
82                  WHEN x"2" =>   seg_o <= NOT(display_2);
83                  WHEN x"3" =>   seg_o <= NOT(display_3);
84                  WHEN x"4" =>   seg_o <= NOT(display_4);
85                  WHEN x"5" =>   seg_o <= NOT(display_5);
86                  WHEN x"6" =>   seg_o <= NOT(display_6);
87                  WHEN x"7" =>   seg_o <= NOT(display_7);
88                  WHEN x"8" =>   seg_o <= NOT(display_8);
89                  WHEN x"9" =>   seg_o <= NOT(display_9);
90                  WHEN x"A" =>   seg_o <= NOT(display_A);
91                  WHEN x"B" =>   seg_o <= NOT(display_B);
92                  WHEN x"C" =>   seg_o <= NOT(display_C);
93                  WHEN x"D" =>   seg_o <= NOT(display_D);
94                  WHEN x"E" =>   seg_o <= NOT(display_E);
95                  WHEN x"F" =>   seg_o <= NOT(display_F);
96                  WHEN OTHERS => seg_o <= NOT(display_blank);
97              END CASE;
98          END IF;
99      END PROCESS sevseg_comb;
100
101 -- End Architecture
102 END rtl;
103
104
```