```
--*  History:
--*
--*  04.02.2016    First Version
--*************************************************************************


-- ************************************************************************
-- *  Library                                                             *
-- ************************************************************************

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;


-- ************************************************************************
-- *  Entity                                                              *
-- ************************************************************************

entity segment7_mux is
   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 entity segment7_mux;

-- ************************************************************************
-- *  Architecture                                                        *
-- ************************************************************************

architecture syn of segment7_mux is

   signal toggle        : std_logic;
   signal DOUT_TEMP     : std_logic_vector(6 downto 0);

begin

   -----------------------------------------
   -- Register File
   -----------------------------------------
```

```
 77
 78
 79      clk_proc:process(CLK)
 80      variable COUNT       :unsigned(10 downto 0) := "00000000000";
 81      begin
 82         if (rising_edge(CLK)) then
 83            COUNT := COUNT + 1;
 84         else
 85            COUNT := COUNT;
 86         end if;
 87      toggle <= COUNT(10);
 88      end process clk_proc;
 89      DIG1 <= NOT toggle;
 90      DIG2 <= toggle;
 91
 92      DOUT_TEMP(0) <= (DIN2(0)) WHEN (toggle = '1')   ELSE (DIN1(0));
 93      DOUT_TEMP(1) <= (DIN2(1)) WHEN (toggle = '1')   ELSE (DIN1(1));
 94      DOUT_TEMP(2) <= (DIN2(2)) WHEN (toggle = '1')   ELSE (DIN1(2));
 95      DOUT_TEMP(3) <= (DIN2(3)) WHEN (toggle = '1')   ELSE (DIN1(3));
 96      DOUT_TEMP(4) <= (DIN2(4)) WHEN (toggle = '1')   ELSE (DIN1(4));
 97      DOUT_TEMP(5) <= (DIN2(5)) WHEN (toggle = '1')   ELSE (DIN1(5));
 98      DOUT_TEMP(6) <= (DIN2(6)) WHEN (toggle = '1')   ELSE (DIN1(6));
 99   -- DOUT_TEMP(7) <= (DIN2(7)) WHEN (toggle = '1')    ELSE (DIN1(7));
100
101      DOUT(0) <= '0' WHEN (DOUT_TEMP(0) = '0')  ELSE '1';
102      DOUT(1) <= '0' WHEN (DOUT_TEMP(1) = '0')  ELSE 'Z'; --open drain
103      DOUT(2) <= '0' WHEN (DOUT_TEMP(2) = '0')  ELSE '1';
104      DOUT(3) <= '0' WHEN (DOUT_TEMP(3) = '0')  ELSE '1';
105      DOUT(4) <= '0' WHEN (DOUT_TEMP(4) = '0')  ELSE '1';
106      DOUT(5) <= '0' WHEN (DOUT_TEMP(5) = '0')  ELSE 'Z'; --open drain
107      DOUT(6) <= '0' WHEN (DOUT_TEMP(6) = '0')  ELSE 'Z'; --open drain
108   -- DOUT(7) <= '0' WHEN (DOUT_TEMP(7) = '0')   ELSE '1';
109
110
111
112
113
114   end architecture syn;
115
```