Practical 5

Name: Pratiksha Madhav Basawade

Roll no: E43017

Division: BE-III

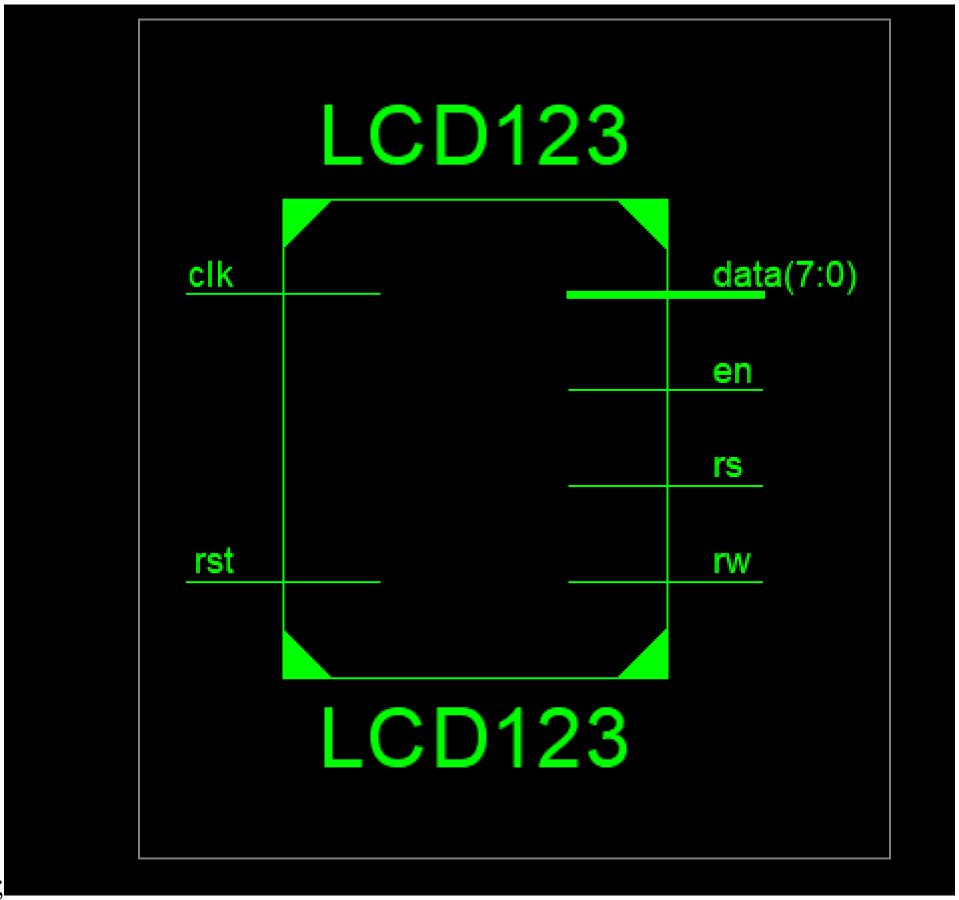
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;

entity LCD123 is


    Port ( rst : in std_logic;
            clk : in std_logic;
            rs : out std_logic;
            en : out std_logic;
                    rw:out        std_logic;
            data : out std_logic_vector(7 downto 0));
end LCD123;
architecture Behavioral of LCD123 is
signal clk1 : std_logic;
type state_type is (s0,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19);
signal state     : state_type;
signal count:std_logic_vector(22 downto 0);
                begin
```

```
                     rw<='0'          ;
process(rst,clk)
begin
if( clk'event and clk ='1')then
          count<=count+1;


    end if;
          clk1<=count(20);
end process;
----- state and output   decoding process


process(clk1, rst)
begin
if( rst='1')then
data<="00000000";
          state<=s0;
elsif(clk1'event and clk1='1' )then


                     case state is
when s0         =>rs <='0';
                                                       en<='1';
                                        data     <= "00110000";                      -- function set
                                                       state<=s1;


when s1         =>en<='0';
                                                       state<=s2;
```

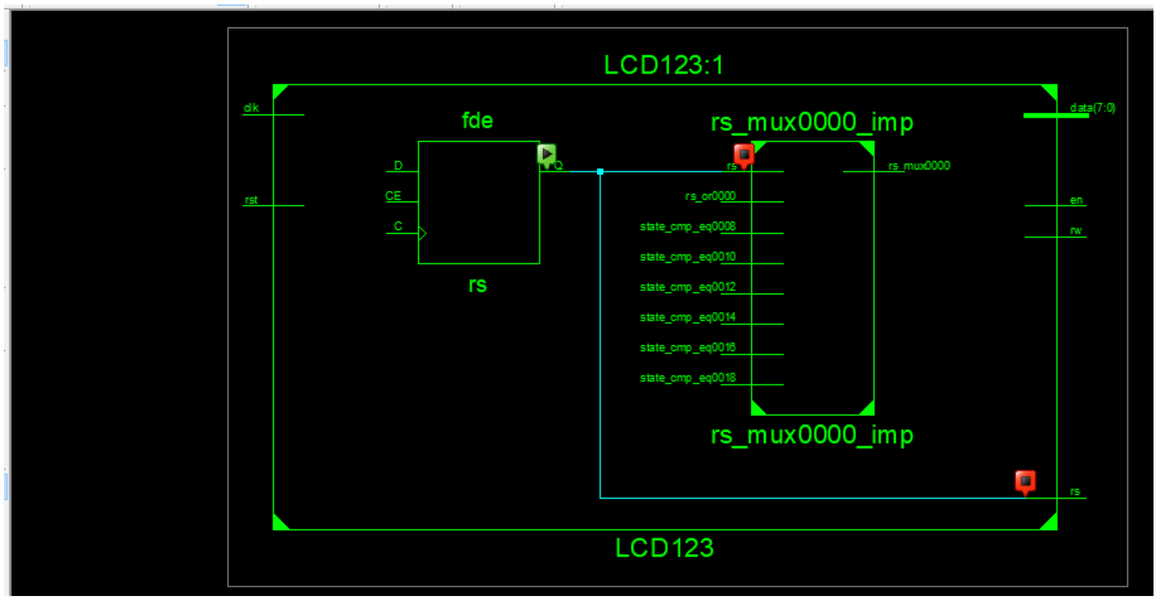
```
when s2         =>rs <='0';
                                    en<='1';
                        data    <= "00001111";              -- cursor blinking
                                    state<=s3;

when s3         =>en<='0';
                                    state<=s4;



when s4         =>rs <='0';
                                    en<='1';
                        data    <= "00000001";              -- clear LCD
                                    state<=s5;

when s5         =>en<='0';
                                    state<=s6;
when s6         =>rs <='0';
                                    en<='1';
                        data    <= "10000100";              -- Cursor position
                                    state<=s7;

when s7         =>en<='0';
                                    state<=s8;
when s8         =>rs <='1';
                                    en<='1';
```

```
                                data    <= "01010011";              -- Binary value of S
                                          state<=s9;


when s9         =>en<='0';
                                          state<=s10;
when s10        =>rs<='1';
                                          en<='1';
                                data    <= "01001011";              -- Binary value of K
                                          state<=s11;


when s11        =>en<='0';
                                          state<=s12;
when s12        =>rs<='1';
                                          en<='1';
                                data    <= "01001110";              -- Binary value of N
                                          state<=s13;


when s13        =>en<='0';
                                          state<=s14;
when s14        =>rs<='1';
                                          en<='1';
                                data    <= "01000011";              -- Binary value of C



                                          state<=s15;


when s15        =>en<='0';
```

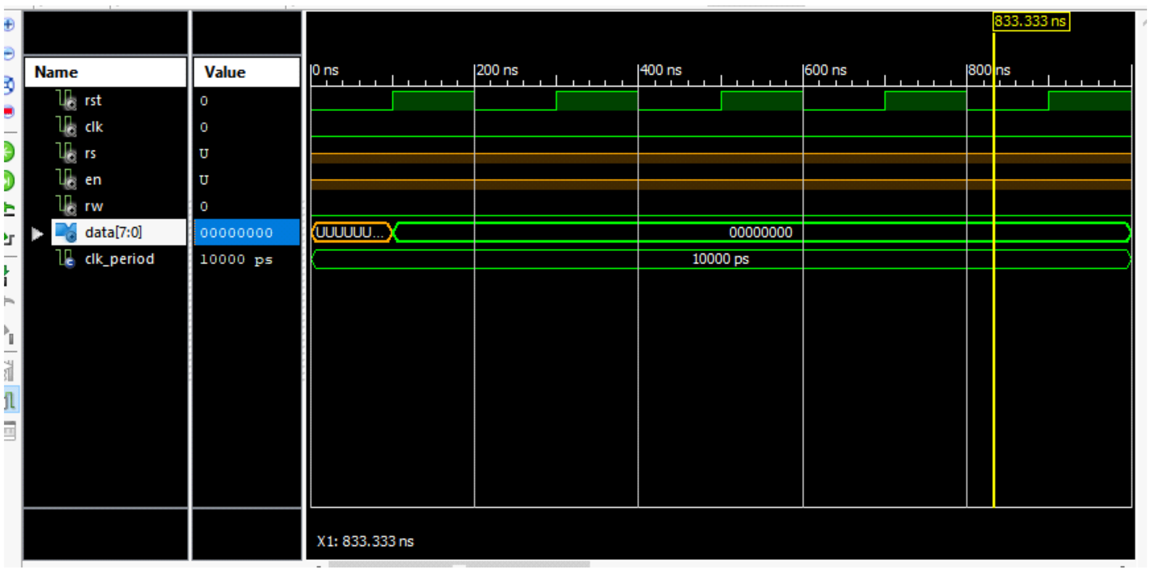
```
                                        state<=s16;



when s16        =>rs<='1';
                                        en<='1';
                              data    <= "01001111";              -- Binary value of O
                                        state<=s17;

when s17        =>en<='0';
                                        state<=s18;
when s18        =>rs<='1';
                                        en<='1';
                              data    <= "01000101";              -- Binary value of E
                                        state<=s19;

when s19        =>en<='0';
                                        state<=s0;
                                        when others=>state<=s0;
                                        end case;
                                        end if;
                                        end Process;
                               end    Behavioral
```


LCD123
clk
rst
data(7:0)
en
rs
rw
LCD123

;


LCD123:1
clk
rst
fde
D
CE
C
Q
rs
rs_mux0000_imp
rs
rs_or0000
state_cmp_eq0008
state_cmp_eq0010
state_cmp_eq0012
state_cmp_eq0014
state_cmp_eq0016
state_cmp_eq0018
rs_mux0000
rs_mux0000_imp
data(7:0)
en
rw
rs
LCD123

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY SSS IS
END SSS;

ARCHITECTURE behavior OF SSS IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT LCD123
    PORT(
         rst : IN   std_logic;
         clk : IN   std_logic;
         rs : OUT   std_logic;
         en : OUT   std_logic;
         rw : OUT   std_logic;
         data : OUT   std_logic_vector(7 downto 0)
        );
    END COMPONENT;



   --Inputs
   signal rst : std_logic := '0';
   signal clk : std_logic := '0';

        --Outputs
    signal rs : std_logic;

    signal en : std_logic;

    signal rw : std_logic;

    signal data : std_logic_vector(7 downto 0);


    -- Clock period definitions

    constant clk_period : time := 10 ns;


BEGIN


        -- Instantiate the Unit Under Test (UUT)

    uut: LCD123 PORT MAP (

            rst => rst,

            clk => clk,

            rs => rs,

            en => en,

            rw => rw,

            data => data

        );


    -- Clock process definitions

    clk_process :process

    begin

                rst <= '1';

                wait for 100 ns;

                rst <= '0';

                wait for 100 ns;
    end process;


    -- Stimulus process
    stim_proc: process
    begin
        -- hold reset state for 100 ns.
        wait for 100 ns;

        wait for clk_period*10;

        -- insert stimulus here

        wait;
    end process;

END;
```



```
rst         0
clk         0
rs          U
en          U
rw          0
data[7:0]   00000000    UUUUUU...  00000000
clk_period  10000 ps    10000 ps
```

UCF FILE

```
net clk loc=p183;
net rst loc=p102;
net data(0) loc=p62;
net data(1) loc=p63;
net data(2) loc=p64;
net data(3) loc=p65;
net data(4) loc=p67;
net data(5) loc=p68;
net data(6) loc=p71;
net data(7) loc=p72;
net en loc=p61;
net rw loc=p58;
net rs loc=p57;
```