Course Name:	**Hardware Description Language Lab (2UXL401)**	**Semester:**	**IV**
Date of Performance:	**27 / 04 / 2021**	**Batch No:**	**B2**
Faculty Name:	**Prof. Bhargavi Kaslikar**	**Roll No:**	**1912060**
Faculty Sign & Date:		**Grade/Marks:**	

Experiment No: 7

Title: FSM Implementation : Word Problem

Aim and Objective of the Experiment:

Word problem for development of state diagram and design using VHDL.

To study FSM implementation in VHDL and to understand use of test bench for simulation.

COs to be achieved:

CO 1: Use basic Concurrent and Sequential statements in VHDL and write codes for simple applications

CO 2: Test a VHDL code and verify the circuit model.

Work to be done

Draw FSM for given problem. Upload VHDL codes for FSM developed from word problem. Also upload test bench and simulation for the same.

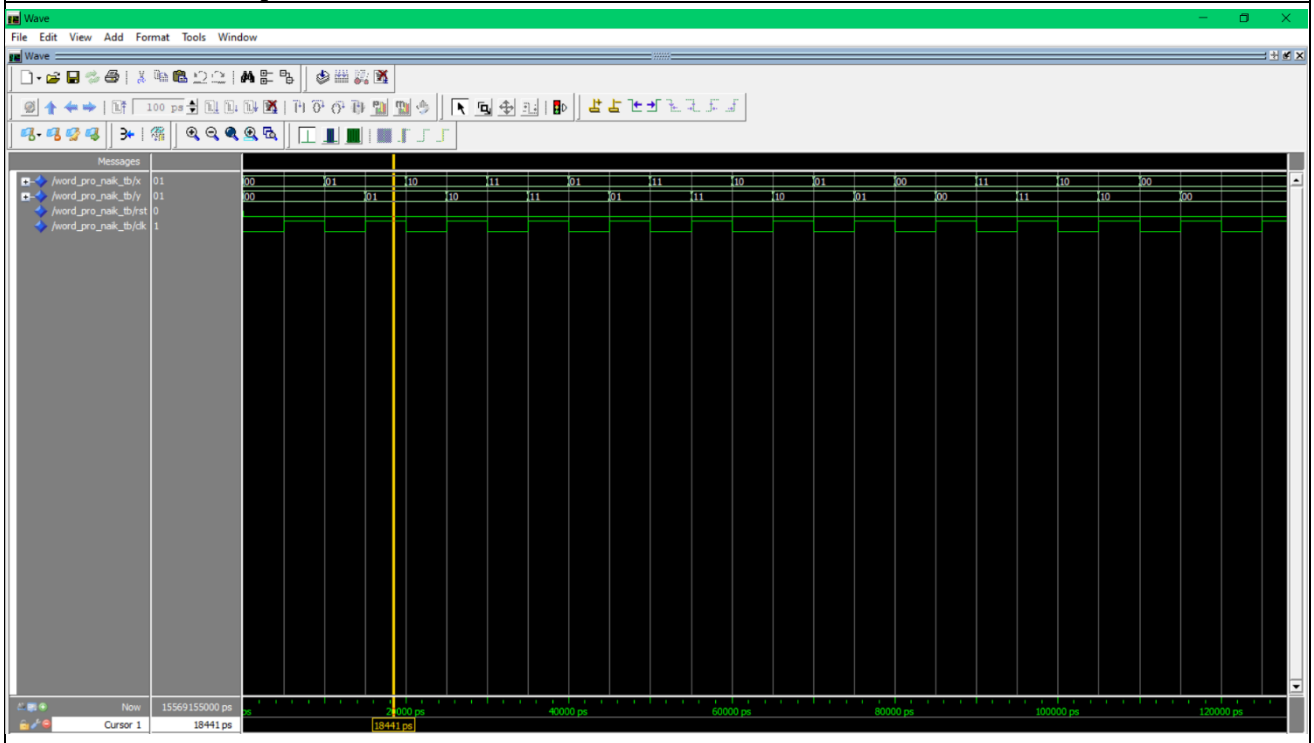
```
library ieee;
use ieee.std_logic_1164.all;

entity word_pro_naik is
        port(
                x: in std_logic_vector(1 downto 0);
                clk, rst : std_logic;
                y : out std_logic_vector(1 downto 0)
                );
end word_pro_naik;

architecture word_pro_naik_arch of word_pro_naik is
        type state is (reset, go_left, go_right, emergency);
        signal ps, ns : state := reset;

        begin
                process(clk, rst)
```

```
                begin
                if(rst = '1')then
                        ps <= reset;
                        elsif(clk'event and clk = '1' )then
                        ps <= ns;
                        end if;
        end process;

        process(ps, x)
        begin
                        if(x = "00") then
                                ns <= reset;
                        elsif (x = "01") then
                                ns <= go_right;
                        elsif (x = "10") then
                                ns <= go_left;
                        else
                                ns <= emergency;
                        end if;
        end process;

        process (ps) begin
                case ps is
                        when reset => y <= "00";
                        when go_left => y <= "10";
                        when go_right => y <= "01";
                        when emergency => y <= "11";
                end case;
        end process;
end word_pro_naik_arch;
```

```
        library ieee;
        use ieee.std_logic_1164.all;

        entity word_pro_naik_tb is
        end word_pro_naik_tb;

        architecture word_pro_naik_tb_arch of word_pro_naik_tb is
        component word_pro_naik is
                port(
                        x: in std_logic_vector(1 downto 0);
                        clk, rst : std_logic;
                        y : out std_logic_vector(1 downto 0)
                        );
        end component;
```

```
signal x,y:std_logic_vector(1 downto 0);
signal rst:std_logic;
signal clk:std_logic:='0';
begin
uut: word_pro_naik port map(x,clk,rst,y);
clk<= not clk after 5ns;

process
begin

rst <= '0';
x<="00";
wait for 10ns;

x<="01";
wait for 10ns;

x<="10";
wait for 10ns;

x<="11";
wait for 10ns;

x<="01";
wait for 10ns;

x<="11";
wait for 10ns;

x<="10";
wait for 10ns;

x<="01";
wait for 10ns;

x<="00";
wait for 10ns;

x<="11";
wait for 10ns;

x<="10";
wait for 10ns;

x<="00";
```

```
wait for 10ns;

end process;
end word_pro_naik_tb_arch;
```

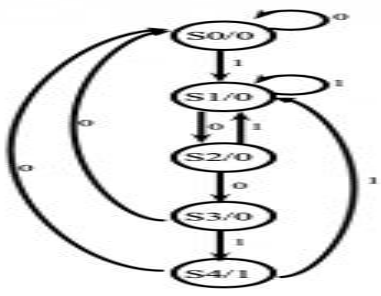


Post Lab Subjective/Objective type Questions:

Upload Answer of following question before coming to next laboratory.

Q1. Write a VHDL code for FSM implementation of a following state diagram.

Write a test bench for the same.



Ans:

Main code:

```
library ieee;

use ieee.std_logic_1164.all;


entity postlab_naik is

        port(

                clk,rst,i:in std_logic;

                y:out std_logic

                );

end postlab_naik;


architecture postlab_naik_arch of postlab_naik is


        type mystate is (s0, s1, s2, s3, s4);

        signal ps,ns:mystate:=s0;

        begin

        process(clk,rst)

        begin

        if(rst='1')then

                ps<= s0;

                elsif(clk'event and clk = '1' )then

                ps<= ns;

                end if;
```

```
end process;


process(ps,i) begin
        case ps is
        when s0 =>
                y <= '0';
                if(i = '0')then
                        ns <= s0;
                else
                        ns <= s1;
                end if;

        when s1 =>
                y <= '0';
                if(i = '0')then
                        ns <= s2;
                else
                        ns <= s1;
                end if;

        when s2 =>
                y <= '0';
                if(i = '0')then
                        ns <= s3;
```

```
                else
                        ns <= s1;
                end if;

        when s3 =>
                y <= '0';
                if(i = '0')then
                        ns <= s0;
                else
                        ns <= s4;
                end if;

        when s4 =>
                y <= '1';
                if(i = '0')then
                        ns <= s0;
                else
                        ns <= s1;
                end if;
        end case;
    end process;
end postlab_naik_arch;
```

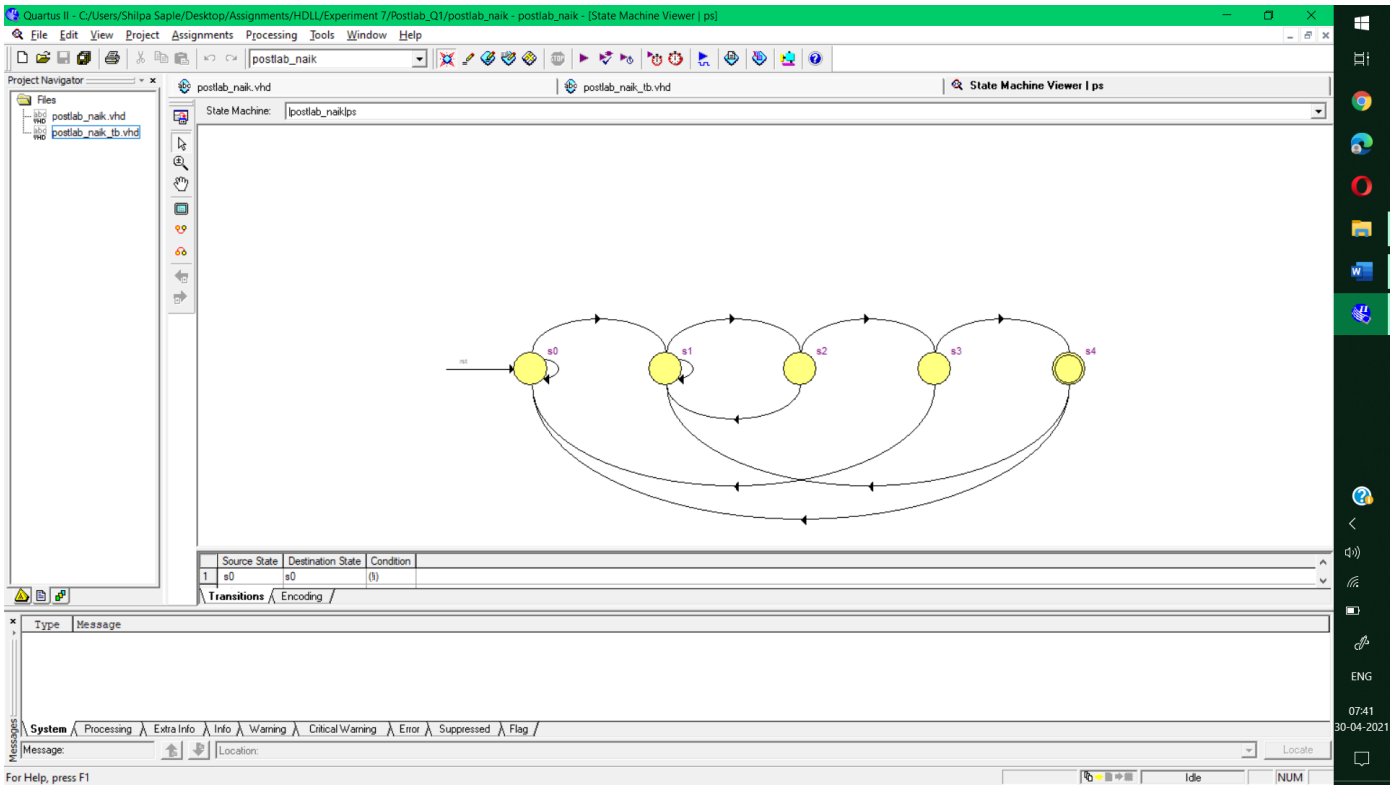
TestBench:

```
library ieee;

use ieee.std_logic_1164.all;


entity postlab_naik_tb is

end postlab_naik_tb;


architecture postlab_naik_tb_arch of postlab_naik_tb is


component postlab_naik is
        port(
                clk,rst,i:in std_logic;
                y:out std_logic
                );
end component;


signal rst,i,y:std_logic;
signal clk:std_logic:='0';


begin
t1: postlab_naik port map(clk,rst,i,y);


        process begin
                clk <= '0';
                wait for 5ns;
```

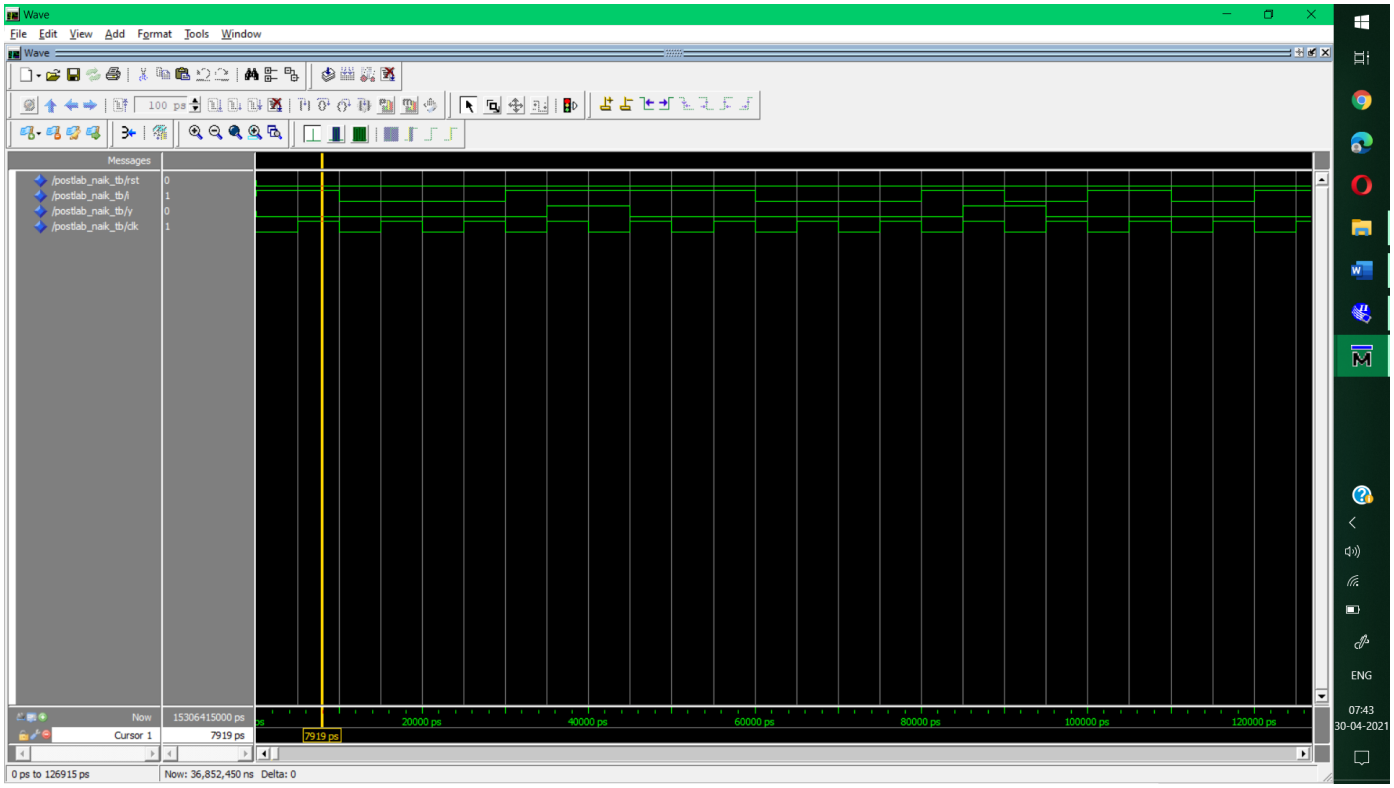
```
        clk <= '1';
        wait for 5ns;
end process;

process
begin
        rst<= '0';
        i<='1';
        wait for 10ns;
        i<='0';
        wait for 10ns;
        i<='0';
        wait for 10ns;
        i<='1';
        wait for 10ns;
        i<='1';
        wait for 10ns;
        i<='1';
        wait for 10ns;
        i<='0';
        wait for 10ns;
        i<='0';
        wait for 10ns;
        i<='1';
```

```
                wait for 10ns;

                i<='0';

                wait for 10ns;

                i<='1';

                wait for 10ns;

                i<='0';

                wait for 10ns;


end process;

end postlab_naik_tb_arch;
```

State Machine:



Output:

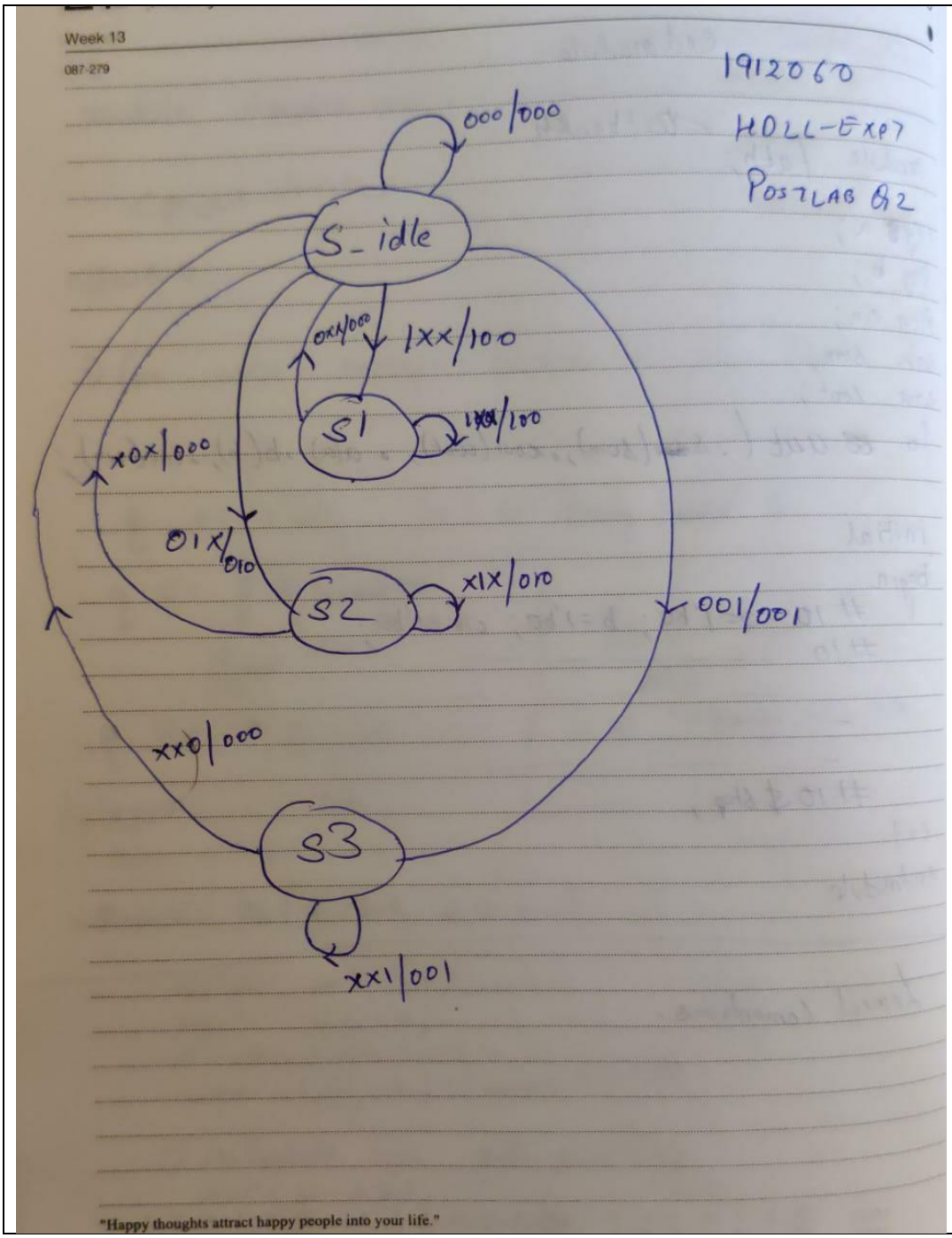


Q2. Analyse the code and draw the state diagram

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity fsm  is
    Port (
          ip1 : in std_logic;
          ip2 : in std_logic;
         ip3 : in std_logic;
         op1 : out std_logic;
         op2 : out std_logic;
         op3 : out std_logic;
         reset : in std_logic;
         clk : in std_logic
         );
```

```
end fsm;

architecture fsm_a of fsm is

  type state_t is (s_idle,s1,s2,s3);
  signal present_state, next_state : state_t;

process(present_state,ip1,ip2,ip3)
    begin
        case present_state is
        when s_idle =>
            if (ip1 ='1') then
           op1<='1'; op2<='0';  op3<='0';
           next_state<= s1; elsif
            (ip2='1') then
                op1<='0'; op2<='1';  op3<='0';
                  next_state<= s2;
            elsif (ip3='1') then
                  op1<='0'; op2<='0';  op3<='1';
                  next_state<= s1;
            else
                  op1<='0'; op2<='0';  op3<='0';
                  next_state<= s_idle;
            end if;
when s1 =>
     if (ip1 ='1') then
       op1<='1'; op2<='0'; op3<='0';
       next_state<= s1;
     else
       op1<='0'; op2<='0'; op3<='0';
       next_state<= s_idle;
   end if;
when s2 =>
     if (ip2 ='1') then
       op1<='0'; op2<='1';op3<='0';
       next_state<= s2;
     else
       op1<='0'; op2<='0'; op3<='0';
       next_state<= s_idle;
   end if;
when s3 =>
        if (ip3 ='1') then
        op1<='0'; op2<='0';  op3<='1';
                next_state<= s3;
        else
```

```
            op1<='0'; op2<='0';  op3<='0';
                    next_state<= s_idle;
            end if;
end case;
end process;
process(clk,reset)
begin
     if reset = '1' then
        present_state <= s_idle;
     elsif clk'event and clk='1' then
        present_state<= next_state;
    end if; end
process; end
fsm_a;
```



Conclusion:

Thus, in this experiment we have implemented a word problem (Vehicle Tail Light) by using finite state machine in VHDL.

Signature of faculty in-charge with Date: