|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Hardware Description Language Lab (2UXL401)** | **Semester:** | **IV** |
| **Date of Performance:** | **12/04/2021** | **Batch No:** | **B2** |
| **Faculty Name:** | **BGK** | **Roll No:** | **1912052** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

**Experiment No: 5**

**Title:** Use of sequential statements:

1. Shift Registers using Flip flops
2. Shift Registers using IC 74194

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| 1. Write a VHDL code for implementing a 4-bit universal shift register with synchronous reset. It has 2 select inputs which control the operation as follows   00: pause  01: shift left  10: shift right  11: Rotate right   1. Write a VHDL code to implement IC 74194 2. Write a test bench to verify your results.   Also, generate a programming file and download the code on CPLD kit and verify the results.  Write a VHDL code  To study basic sequential statements of 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.  **CO 3**: Synthesize and Implement the designed circuits on CPLD/ FPGA. |

|  |
| --- |
| **Work to be done** |
| Upload VHDL codes for 4 bit universal shift register with synchronous reset. Also upload test bench and simulation for the same.  JK FF  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_arith.all;  use ieee.std\_logic\_unsigned.all;    entity jk\_Vedant is  port (  JK: IN STD\_LOGIC\_VECTOR(1 downto 0);  clock: IN STD\_LOGIC;  reset: IN STD\_LOGIC;  q: out STD\_LOGIC  );  end jk\_Vedant;  architecture jk\_Vedant\_arch of jk\_Vedant is    signal q\_s : std\_logic := '0';    begin  process(reset,clock)  begin  if (reset = '1')then  q\_s <='0';  elsif (clock'event and clock = '1')then  case (JK) is  when "00" => q\_s <= q\_s;  when "01" => q\_s <= '0';  when "10" => q\_s <= '1';  when others => q\_s <= not q\_s;  end case;  end if;  q <= q\_s;  end process;  end jk\_Vedant\_arch;  Shift Register Entity  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.std\_logic\_arith.all;  entity Shift\_Vedant is  port(  rst,clk : in std\_logic;  output : out std\_logic\_vector(3 downto 0)  );  end Shift\_Vedant;  architecture Shift\_Vedant\_arch of Shift\_Vedant is    component jk\_Vedant is  port (  JK: IN STD\_LOGIC\_VECTOR(1 downto 0);  clock: IN STD\_LOGIC;  reset: IN STD\_LOGIC;  q: out STD\_LOGIC  );  end component;  signal temp\_output:std\_logic\_vector(3 downto 0);  signal temp1,temp2,temp3: std\_logic\_vector(1 downto 0);  begin  FF1 : jk\_Vedant port map("11" , clk , rst , temp\_output(0));    temp1 <= temp\_output(0) & (not temp\_output(0));  FF2 : jk\_Vedant port map(temp1 , clk , rst , temp\_output(1));    temp2 <= temp\_output(1) & (not temp\_output(1));  FF3 : jk\_Vedant port map(temp2 , clk , rst , temp\_output(2));    temp3 <= temp\_output(2) & (not temp\_output(2));  FF4 : jk\_Vedant port map(temp3 , clk , rst , temp\_output(3));    output <= temp\_output;  end Shift\_Vedant\_arch;  Shift Register TB  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.std\_logic\_arith.all;  entity Shift\_Vedant\_tb is  end entity;  architecture Shift\_Vedant\_tb\_arch of Shift\_Vedant\_tb is    component Shift\_Vedant is  port(  rst,clk : in std\_logic;  output : out std\_logic\_vector(3 downto 0)  );  end component;  signal rst,clk : std\_logic;  signal output : std\_logic\_vector(3 downto 0);  begin  SR : Shift\_Vedant port map(rst , clk , output);    process begin  clk <= '1';  wait for 10ns;  clk <= '0';  wait for 10ns;  end process;    process  begin  rst <= '0';  wait for 2000ns;  rst <= '1';  wait for 20ns;  end process;  end Shift\_Vedant\_tb\_arch;    Upload scanned copy of post lab questions |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| Upload Answer of following question before coming to next laboratory.  **Q1** **Differentiate the properties of signal and variable. Give suitable example**  **Signals vs. Variables:**   * Variables can only be used inside processes, signals can be used inside or outside processes. * Any variable that is created in one process cannot be used in another process, signals can be used in multiple processes *though they can only be assigned in a single process*. * Variables need to be defined after the keyword *process* but before the keyword *begin*. Signals are defined in the architecture before the *begin* statement. * Variables are assigned using the **:=** assignment symbol. Signals are assigned using the **<=** assignment symbol. * Variables that are assigned immediately take the value of the assignment. Signals depend on if it's combinational or sequential code to know when the signal takes the value of the assignment. |

|  |
| --- |
| **Conclusion:**  Wrote a VHDL code for implementing a 4-bit universal shift register with synchronous reset |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |