|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Hardware Description Language Lab (2UXL401)** | **Semester:** | **IV** |
| **Date of Performance:** | **23/03/2021** | **Batch No:** | **B2** |
| **Faculty Name:** | **BGK** | **Roll No:** | **1912052** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

**Experiment No: 4**

**Title:** Use of sequential statements:

1. Counter using IC 74163
2. Synchronous Counter using Flip flops(3 Bit)

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| Write a VHDL code for implementing a JK flip flop.  Write a VHDL code for implementing a 3bit asynchronous counter with use of JK flip flop component. ( Structural)  Write a testbench to verify your results.  Write a VHDL code for implementing a 4 bit up down synchronous counter using IC 74163  Write a testbench to verify your results.  Also, generate a programming file and download the code on CPLD kit and verify the results for synchronous counter    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 codes for JK flip-flop, synchronous counter and synchronous counter using 74163. Also upload test benches and simulation for the same.  ENTITY 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;  ENTITY Counter  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.std\_logic\_arith.all;  entity async\_Vedant is  port(  rst,clk:In std\_logic;  o:Out std\_logic\_vector(2 downto 0)  );  end async\_Vedant;  architecture async\_Vedant\_arch of async\_Vedant is  component jk\_Vedant is  port(  JK:In std\_logic\_vector(1 downto 0);  reset:In std\_logic;  clock:In std\_logic;  q:Out std\_logic  );  end component;  signal s:std\_logic\_vector(2 downto 0);  begin  asc0:jk\_Vedant port map("11",rst,clk,s(0));  asc1:jk\_Vedant port map("11",rst,s(0),s(1));  asc2:jk\_Vedant port map("11",rst,s(1),s(2));  o<=s;  end async\_Vedant\_arch;  Testbench Counter  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.std\_logic\_arith.all;  entity async\_tb is  end async\_tb;  architecture async\_Vedant\_arch of async\_tb is  component async\_Vedant is  port(  rst,clk:In std\_logic;  o:Out std\_logic\_vector(2 downto 0)  );  end component;  signal rst,clk:std\_logic;  signal o:std\_logic\_vector(2 downto 0);  begin  uut:async\_Vedant port map(rst,clk,o);  process  begin  clk<='1';  wait for 10ns;  clk<=not clk;  wait for 10ns;  end process;  process  begin  rst<='1';  wait for 20ns;  rst<='0';  wait for 200ns;  end process;  end async\_Vedant\_arch ;  IC 74163  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_arith.all;  use ieee.std\_logic\_unsigned.all;  entity IC74163\_Vedant is  port  (  clk : in std\_logic;  sel : in std\_logic\_vector(3 downto 0);  l : in std\_logic\_vector(3 downto 0);  o : out std\_logic\_vector(3 downto 0)  );  end IC74163\_Vedant;  architecture IC74163\_Vedant\_arch of IC74163\_Vedant is  signal s :std\_logic\_vector(3 downto 0);  begin  process(clk,sel,l)  begin  if (clk'event and clk='0') then  if(sel(0) = '0') then  s <= "0000";  elsif(sel(1) = '0') then  s <= l;  else  if(sel = "1111") then  s <= s+1;  end if;  end if;  end if;  end process;  o <= s;  end IC74163\_Vedant\_arch;  Test Bench  library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_arith.all;  use ieee.std\_logic\_unsigned.all;  entity IC74163\_Vedant\_tb is  end IC74163\_Vedant\_tb;  architecture IC74163\_Vedant\_tb\_arch of IC74163\_Vedant\_tb is    component IC74163\_Vedant is  port  (  clk : in std\_logic;  sel : in std\_logic\_vector(3 downto 0);  l : in std\_logic\_vector(3 downto 0);  o : out std\_logic\_vector(3 downto 0)  );  end component;    signal clk: std\_logic;  signal sel, l, o: std\_logic\_vector(3 downto 0);    begin  IC74163: IC74163\_Vedant port map(clk, sel, l, o);    process begin  clk <= '0';  wait for 20ns;    clk <= '1';  wait for 20ns;  end process;      process begin    sel <= "0000";  wait for 20ns;      l <= "0000";  sel <= "1000";  wait for 20ns;      sel <= "1111";  wait for 2000ns;      end process;    end IC74163\_Vedant\_tb\_arch; |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| **Q1. If synchronous preset and asynchronous clear inputs are to be added in the above flip-flops then how will you modify the code?**    **Q.2. What are the ways in which clock can be generated in test bench?** |

|  |
| --- |
| **Conclusion:**  We wrote a VHDL code for implementing a 3bit asynchronous counter with use of JK flip flop component. |

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