VLSI Lab Report
Assignment 2 Annexure 1

BCSE 4th Year 2nd Semester

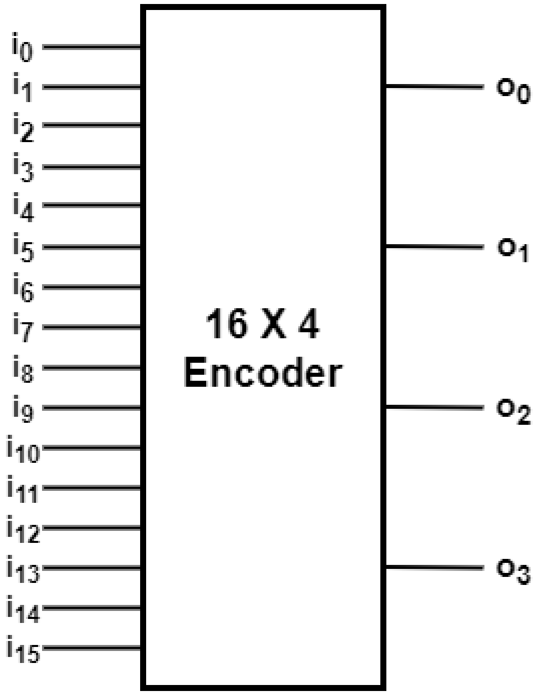
Name: ***Priti Shaw***
Roll No. : ***001710501076***
Batch: ***A3***
Submission Date: ***06th April 2021***

1. Description

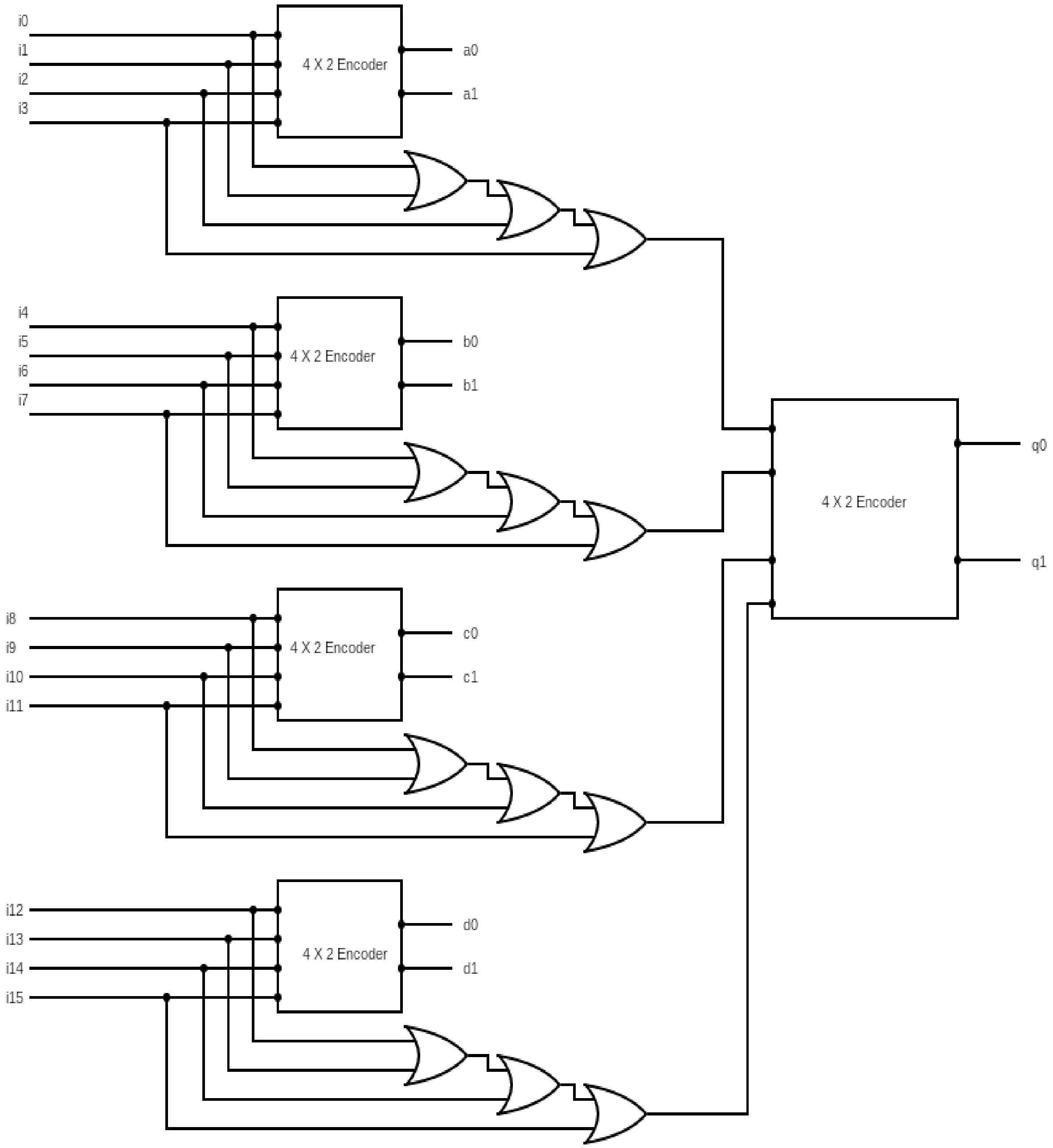
Design 16 X 4 encoder using 4 X 2 encoders using generate statement. Write test bench using loop statements(for and while).

An encoder in digital electronics is a one-hot to binary converter. That is if there are 2^n input lines, and at most only one of them will ever be high, the binary code of this 'hot' line is produced on the n-bit output lines. A 16-to-4 encoder has 16 input lines and 4 output lines.

2. Block Diagram



3. Circuit Diagram



4. Truth table

Input																Output			
i_{15}	i_{14}	i_{13}	i_{12}	i_{11}	i_{10}	i_9	i_8	i_7	i_6	i_5	i_4	i_3	i_2	i_1	i_0	o_3	o_2	o_1	o_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

5. Entity

```
entity Annexure2 is
    Port ( input : in  STD_LOGIC_VECTOR (15 downto 0);
           output : out  STD_LOGIC_VECTOR (3 downto 0));
end Annexure2;
```

6. Architecture

```
architecture Behavioral of Annexure2 is
component QuestionTwo_Part2 is
    Port ( ii : in  STD_LOGIC_VECTOR (3 downto 0);
           oo : out  STD_LOGIC_VECTOR (1 downto 0));
end component;
signal p: std_logic_vector(3 downto 0);

begin
p(0) <= input(0) or input(1) or input(2) or input(3);
p(1) <= input(4) or input(5) or input(6) or input(7);
p(2) <= input(8) or input(9) or input(10) or input(11);
p(3) <= input(12) or input(13) or input(14) or input(15);
c5: QuestionTwo_Part2 port map(p, output(3 downto 2));

gen1: for k in 0 to 3 generate
      cc: QuestionTwo_Part2 port map(input(((4*k)+3) downto (4*k)),
output(1 downto 0));
end generate;
end Behavioral;
```

7. TestBench

a) Using for loop

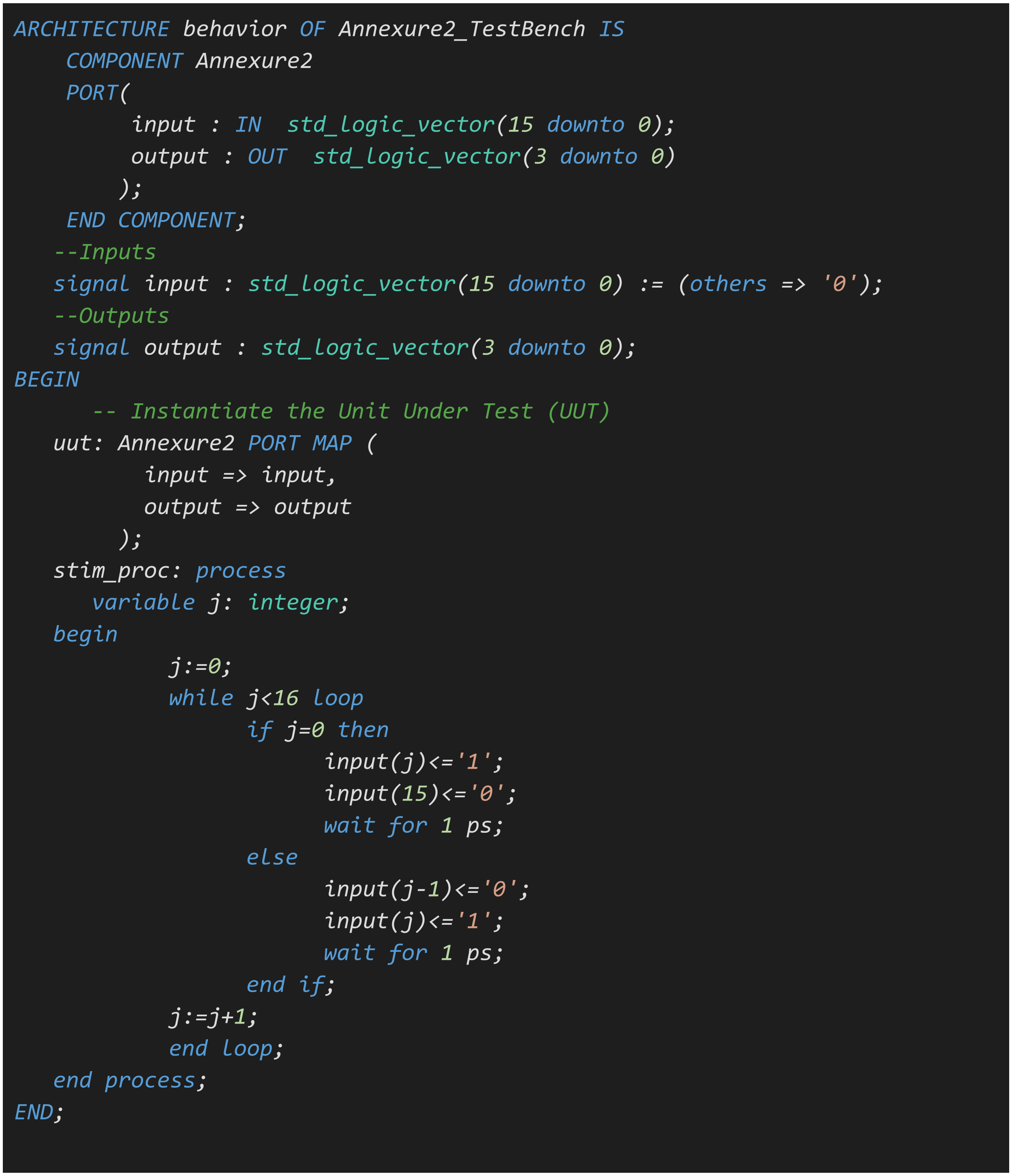
```
ENTITY Annexure2_TestBench IS
END Annexure2_TestBench;

ARCHITECTURE behavior OF Annexure2_TestBench IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT Annexure2
    PORT(
         input : IN  std_logic_vector(15 downto 0);
         output : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;
   --Inputs
   signal input : std_logic_vector(15 downto 0) := (others => '0');
    --Outputs
   signal output : std_logic_vector(3 downto 0);

BEGIN
      -- Instantiate the Unit Under Test (UUT)
   uut: Annexure2 PORT MAP (
          input => input,
          output => output
        );
   stim_proc: process
      variable j: integer;
   begin
            for j in 0 to 15 loop
                  if j=0 then
                        input(j)<='1';
                        input(15)<='0';
                        wait for 1 ps;
                  else
                        input(j-1)<='0';
                        input(j)<='1';
                        wait for 1 ps;
                  end if;
            end loop;
   end process;
END;
```

b) Using while loop

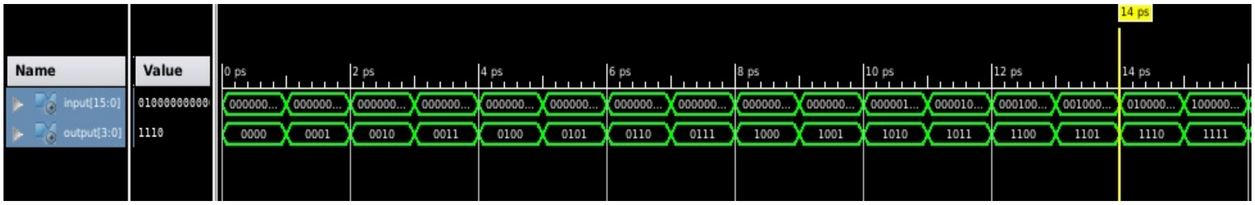
```
ENTITY Annexure2_TestBench IS
END Annexure2_TestBench;
```



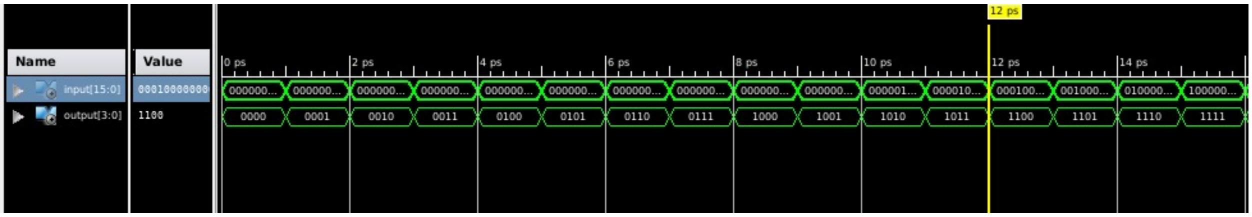
```
ARCHITECTURE behavior OF Annexure2_TestBench IS
    COMPONENT Annexure2
    PORT(
         input : IN  std_logic_vector(15 downto 0);
         output : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;
   --Inputs
   signal input : std_logic_vector(15 downto 0) := (others => '0');
   --Outputs
   signal output : std_logic_vector(3 downto 0);
BEGIN
      -- Instantiate the Unit Under Test (UUT)
   uut: Annexure2 PORT MAP (
          input => input,
          output => output
        );
   stim_proc: process
      variable j: integer;
   begin
            j:=0;
            while j<16 loop
                  if j=0 then
                        input(j)<='1';
                        input(15)<='0';
                        wait for 1 ps;
                  else
                        input(j-1)<='0';
                        input(j)<='1';
                        wait for 1 ps;
                  end if;
            j:=j+1;
            end loop;
   end process;
END;
```

8. Timing Diagram

a) Using *For* loop



b) Using *While* loop


12 ps
Name
Value
input[15:0]
0001000000000
output[3:0]
1100
0 ps
2 ps
4 ps
6 ps
8 ps
10 ps
12 ps
14 ps
000000...
000000...
000000...
000000...
000000...
000000...
000000...
000000...
000000...
000000...
000001...
000010...
000100...
001000...
010000...
100000...
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111