

**Name: Deep Patel**

**Roll No.: CS21B1008**

## **Introduction to HDL Assignment**

Q.2 Design a system which receives 4-bit 8 data samples sequentially and output odd sequenced data from the fourth data point onwards. Verify the design functionally by writing a test-bench at least for two sets of 4-bit 8 data samples. You need to simulate the entire design using the test bench.

### **Design Code:**

#### **File No. 2: prob2.vhd**

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

entity prob_2 is
port (d: in std_logic_vector(3 downto 0);
      clk,nreset,nready: in std_logic;
      q:out std_logic_vector(3 downto 0));
end prob2;

architecture prob2arch of prob2 is

component d_latch
port (d: in std_logic_vector (3 downto 0);
      clk,nreset,enable: in std_logic;
      q: out std_logic_vector(3 downto 0));
end component;

component counter8 is
port(clk : in std_logic;
      nreset : in std_logic;
      nready : in std_logic;
      count : out std_logic_vector (2 downto 0));
end component;

component en_gen2 is
port(clk : in std_logic;
      nreset: in std_logic;
      count_in: in std_logic_vector(2 downto 0);
      enable : out std_logic);
```

```

end component;

signal int_clk : std_logic;
signal counter : std_logic_vector(2 downto 0);
signal enable_out : std_logic;
begin
DFF4 : d_latch port map(clk => clk,nreset => nreset,enable => enable_out,d => d,q => q);
unit_enable: en_gen2 port map(clk => clk,nreset => nreset,count_in => counter,enable =>
enable_out);
unit_counter: counter8 port map(clk => clk,nreset => nreset,nready => nready,count =>
counter);
end prob2arch;

```

### **Sub Files:**

#### **1. d\_latch.vhd**

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;

entity d_latch is
port(
clk: in std_logic;
nreset: in std_logic;
enable: in std_logic;
d : in std_logic_vector (3 downto 0);
q: out std_logic_vector(3 downto 0)
);
end entity d_latch;

architecture d_arch of d_latch is
begin
process (nreset, clk) is
begin
if (nreset = '0') then
q <= (others=>'0');
elsif (rising_edge(clk)) then
if (enable = '1') then
q<= d;
end if;
end if;
end process;
end d_arch;

```

## **2. engen2.vhd**

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

entity en_gen2 is
port( clk : in std_logic;
nreset: in std_logic;
count_in: in std_logic_vector(2 downto 0);
enable : out std_logic
);
end entity en_gen2;
architecture en_gen_arch2 of en_gen2 is
begin
process (nreset, clk) is
begin
    if (nreset = '0') then
        enable <= '0';
    elsif (rising_edge(clk)) then
        if (count_in > "010") and (count_in(0) = '0') then
            enable <= '1';
        else
            enable <= '0';
        end if;
    end if;
end process;
end en_gen_arch2;
```

## **3. count\_8.vhd**

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;
entity counter8 is
--generic ( n : natural := 4);
port(
clk : in std_logic;
nreset : in std_logic;
nready : in std_logic;
count : out std_logic_vector (2 downto 0)
```

```

);
end counter8;
architecture counter8_arch of counter8 is
signal temp : unsigned (2 downto 0);
constant MAXCOUNT : unsigned (2 downto 0) := "111";
begin
p0: process (nreset, clk) is
--signal temp : unsigned (3 downto 0);
begin
if (nreset = '0') then
temp <= (others => '0');
--count <= (others => '0');
elsif (rising_edge(clk)) then
if (nready = '1') then
temp <= (others => '0');
elsif (temp = MAXCOUNT) then
temp <= (others => '0');
else
temp <= temp + 1;
end if;
end if;
--count <= std_logic_vector(temp);
end process p0;
count <= std_logic_vector(temp);
end counter8_arch;

```

### **Test Bench - 1:**

```

library ieee;
use ieee.std_logic_1164.all;

entity tb_prob2 is end tb_prob2;

architecture test2_1 of tb_prob2 is
component prob2
port(d: in std_logic_vector(3 downto 0);
clk,nreset,nready: in std_logic;
q:out std_logic_vector(3 downto 0));
end component;

signal clk : std_logic := '1';
signal nreset : std_logic := '1';
signal nready : std_logic;

```

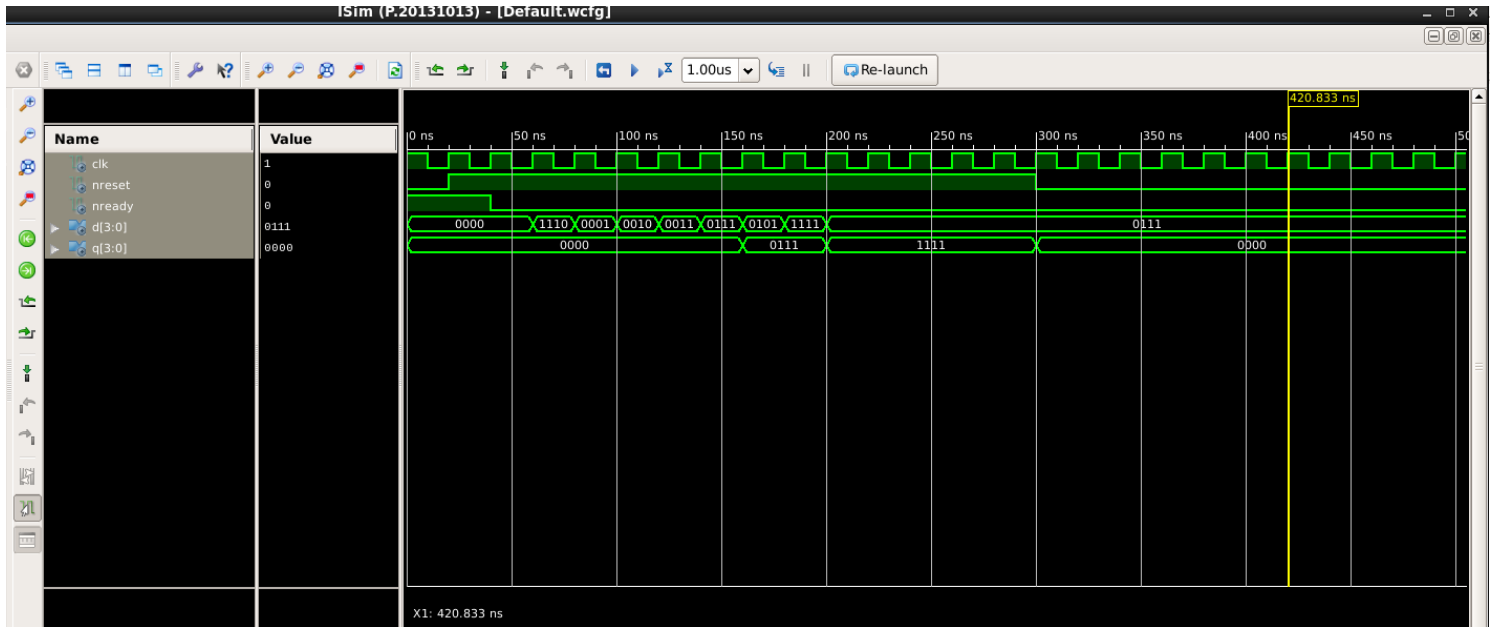
```

signal d: std_logic_vector(3 downto 0) := (others=>'0');
signal q: std_logic_vector(3 downto 0);

begin
dut:prob2 port map(d=>d, nready=>nready, nreset=>nreset, clk=>clk, q=>q);
clock_gen: process(clk)
begin
clk <= not clk after 10 ns;
end process clock_gen;
wavegen_proc: process
begin
nreset <= '0';
nready <= '1';
wait for 20 ns;
nreset <= '1';
wait for 20 ns;
nready <= '0';
wait for 20 ns; d <= "1110";
wait for 20 ns; d <= "0001";
wait for 20 ns; d <= "0010";
wait for 20 ns; d <= "0011";
wait for 20 ns; d <= "0111";
wait for 20 ns; d <= "0101";
wait for 20 ns; d <= "1111";
wait for 20 ns; d <= "0111";
wait for 100 ns;
nreset <= '0';
wait;
end process;
end architecture test2_1;

```

## Simulation of Test Bench - 1:



## Test Bench - 2:

library ieee;

use ieee.std\_logic\_1164.all;

entity tb2\_prob2 is end tb2\_prob2;

architecture test2\_2 of tb2\_prob2 is

component prob2

port(d: in std\_logic\_vector(3 downto 0);

clk,nreset,nready: in std\_logic;

q:out std\_logic\_vector(3 downto 0));

end component;

signal clk : std\_logic := '1';

signal nreset : std\_logic := '1';

signal nready : std\_logic;

signal d: std\_logic\_vector(3 downto 0) := (others=>'0');

signal q: std\_logic\_vector(3 downto 0);

begin

dut:prob2 port map(d=>d, nready=>nready, nreset=>nreset, clk=>clk, q=>q);

clock\_gen: process(clk)

begin

```

clk <= not clk after 10 ns;
end process clock_gen;
wavegen_proc: process
begin
nreset <= '0';
nready <= '1';
wait for 20 ns;
nreset <= '1';
wait for 20 ns;
nready <= '0';
wait for 20 ns; d <= "1111";
wait for 20 ns; d <= "1001";
wait for 20 ns; d <= "1010";
wait for 20 ns; d <= "1011";
wait for 20 ns; d <= "1100";
wait for 20 ns; d <= "1101";
wait for 20 ns; d <= "1110";
wait for 20 ns; d <= "1111";
wait for 100 ns;
nreset <= '0';
wait;
end process;
end architecture test2_2;

```

## Simulation of Test Bench - 2:

