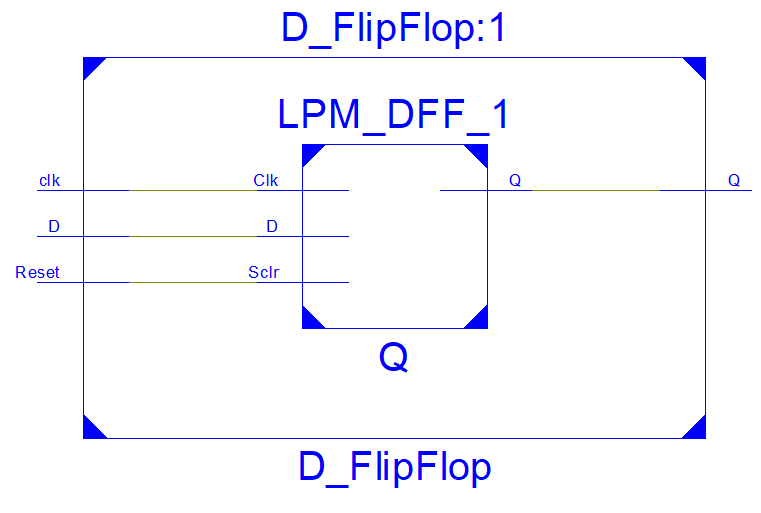
**EXPERIMENT 7**

**Aim:** Design of registers and counters

**Exercise#1**: Design and verify 4-bit serial-in-serial-out (SISO) shift register shown in Figure-4 in structural style of architecture. Use reset control line to reset the register.

**Design Code (Flip Flop):**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity D\_FlipFlop is

Port ( D : in STD\_LOGIC;

Reset : in STD\_LOGIC;

clk : in STD\_LOGIC;

Q : out STD\_LOGIC);

end D\_FlipFlop;

architecture Behavioral of D\_FlipFlop is

begin

process(D, Reset, clk)

begin

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

if(reset = '1' ) then

q <= '0';

else

q <= d;

end if;

end if;

end process;

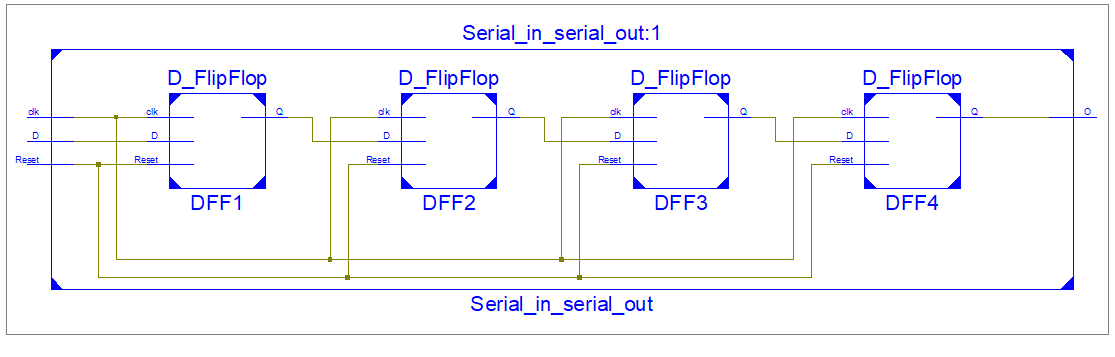
end Behavioral;

**Design Code:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity Serial\_in\_serial\_out is

 Port ( D : in STD\_LOGIC;

Reset : in STD\_LOGIC;

clk : in STD\_LOGIC;

O : out STD\_LOGIC);

end Serial\_in\_serial\_out;

architecture Behavioral of Serial\_in\_serial\_out is

component D\_FlipFlop

Port ( D : in STD\_LOGIC;

Reset : in STD\_LOGIC;

clk : in STD\_LOGIC;

Q : out STD\_LOGIC);

end component D\_FlipFlop;

signal S1, S2, S3 :STD\_LOGIC;

begin

DFF1: D\_FlipFlop port map(D, Reset, clk, S1);

DFF2: D\_FlipFlop port map(S1, Reset, clk, S2);

DFF3: D\_FlipFlop port map(S2, Reset, clk, S3);

DFF4: D\_FlipFlop port map(S3, Reset, clk, O);

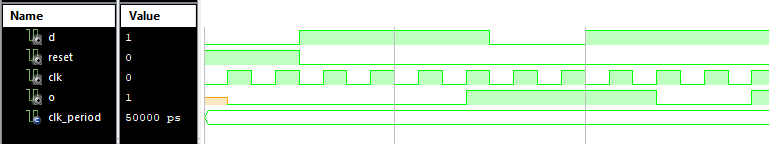
end Behavioral;

**Test Bench:**

D <= '0';

Reset <= '1';

wait for 100 ns;

D <= '1';

Reset <= '0';

wait for 100 ns;

D <= '1';

Reset <= '0';

wait for 100 ns;

D <= '0';

Reset <= '0';

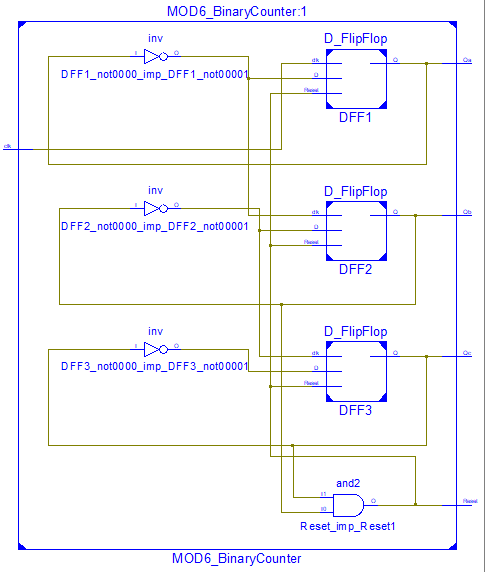
wait for 100 ns;

D <= '1';

Reset <= '0';

wait for 100 ns;

wait for clk\_period\*10;

**Exercise#2**: Design and verify MOD-6 binary counter shown in figure-5 using structural style of architecture.

**Design Code:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity MOD6\_BinaryCounter is

Port (Reset : inout STD\_LOGIC;

clk : in STD\_LOGIC;

Qa,Qb,Qc : inout STD\_LOGIC);

end MOD6\_BinaryCounter;

architecture Behavioral of MOD6\_BinaryCounter is

component D\_FlipFlop

Port ( D : in STD\_LOGIC;

Reset : in STD\_LOGIC;

clk : in STD\_LOGIC;

Q : out STD\_LOGIC);

end component D\_FlipFlop;

begin

Reset <= Qb and Qc;

DFF1: D\_FlipFlop port map((not Qa), Reset, clk,Qa);

DFF2: D\_FlipFlop port map((not Qb), Reset, (not Qa),Qb);

DFF3: D\_FlipFlop port map((not Qc), Reset,(not Qb),Qc);

end Behavioral;

Test Bench:

Qa<='1';

Qb<='0';

Qc<='0';

Reset <='0';

wait for 100 ns;

Qa<='0';

Qb<='1';

Qc<='0';

Reset <='1';

wait for 100 ns;

Qa<='1';

Qb<='1';

Qc<='0';

Reset <='0';

wait for 100 ns;