TASK-02

FIG : BLOCK DIAGRAM OF FLIP FLOP

D	Q(Current)	Q(n+1) (Next)
0	0	0
0	1	0
1	0	1
1	1	1

FIG : TRUTH TABLE FOR D FLIP FLOP

CODE FOR D FLIP FLOP

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
entity D_FLIPFLOP_SOURCE is
Port ( D, CLK, RST : in  STD_LOGIC;
Q, Qb : out  STD_LOGIC);
end D_FLIPFLOP_SOURCE;

architecture Behavioral of D_FLIPFLOP_SOURCE is

begin
process (D, CLK, RST)
begin

if (RST = '1') then
Q <= '0';
elsif (rising_edge(CLK)) then ---this is for data flip-flop, for delay flip-flop use negative edge
Q <= D;
Qb <= not D;

end if;
end process;
end Behavioral;
```

TEST BENCH

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity DFF_tb is
end entity;

architecture tb of DFF_tb is
```

```
component D_FLIPFLOP_SOURCE is
Port ( D, CLK, RST : in STD_LOGIC;
Q, Qb : out STD_LOGIC);
end component ;

signal D, CLK, RST, Q, Qb : STD_LOGIC;

begin
uut: D_FLIPFLOP_SOURCE port map(
D => D,
CLK => CLK,
RST => RST,
Q => Q,
Qb => Qb);

Clock : process
begin
CLK <= '0';
wait for 10 ns;
CLK <= '1';
wait for 10 ns;
end process;

stim : process
begin

RST <= '0';
D <= '0';
wait for 40 ns;
D <= '1';
```

wait for 40 ns;

end process;

end tb;

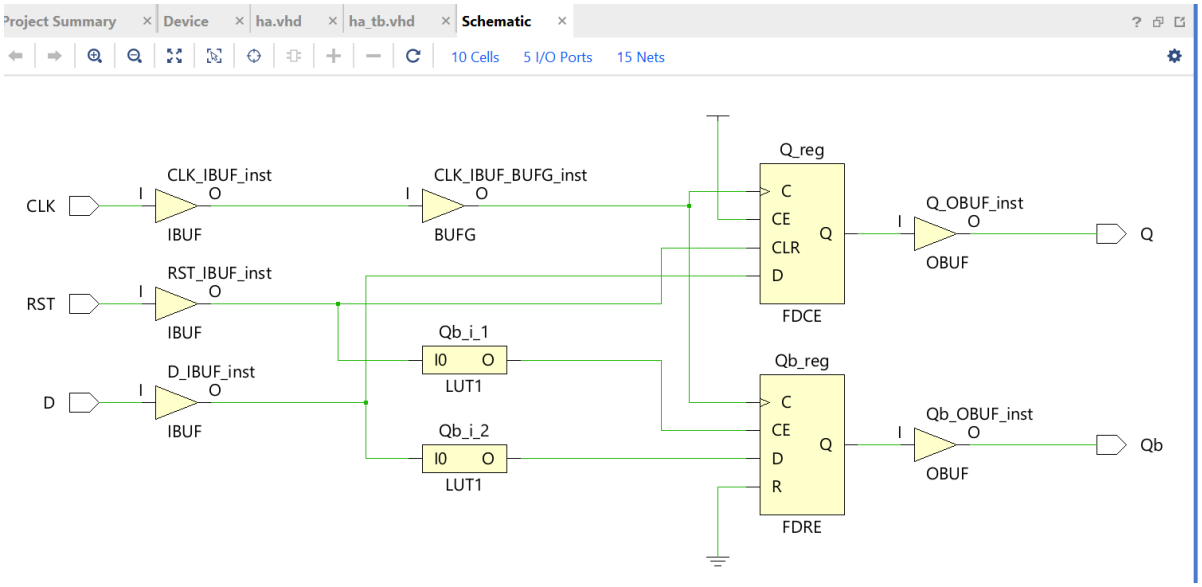

FIG : SCHEMATIC OF D FLIP FLOP

-

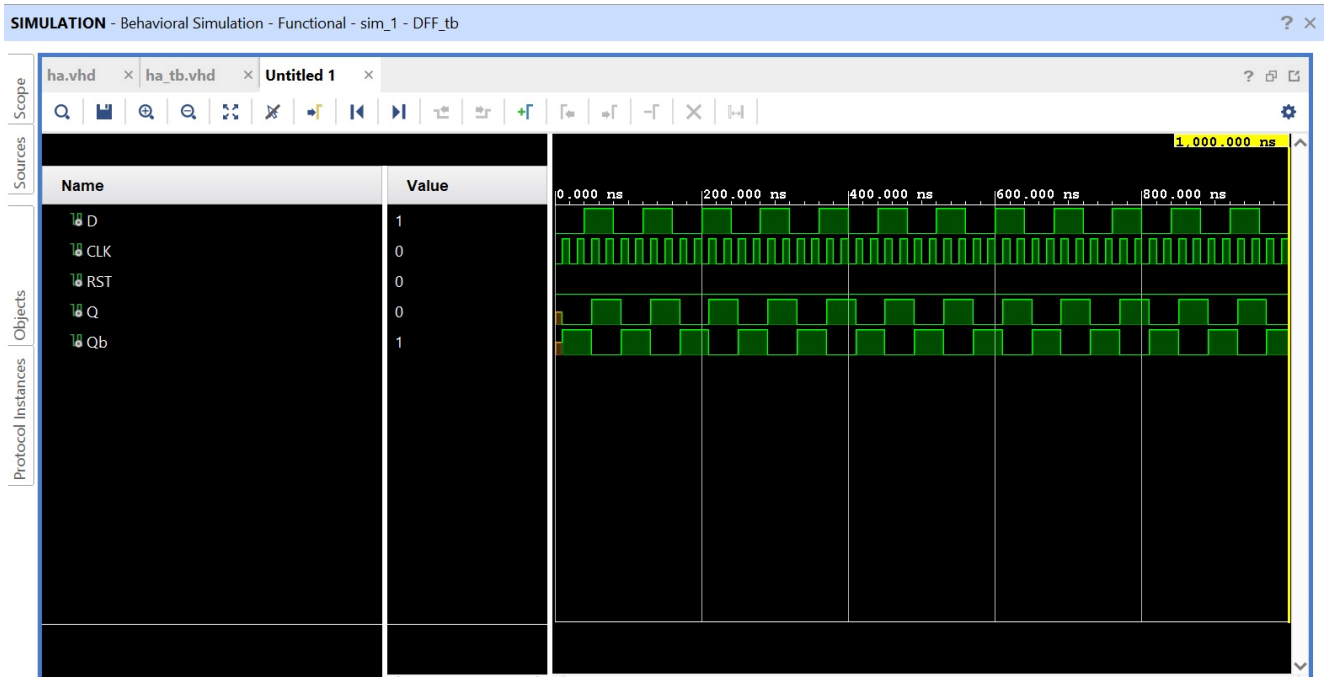

FIG : SIMULATION RESULT OF D FLIP FLOP

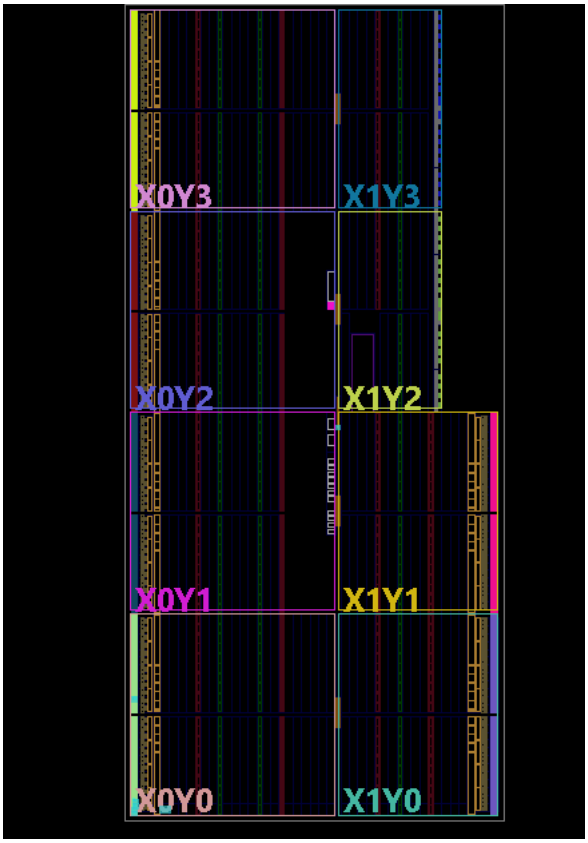

FIG: SYNTHESIS DESIGN