HACETTEPE UNIVERSITY
ELECTRICAL AND ELECTRONICS ENGINEERING
ELE227 FUNDAMENTALS OF DIGITAL SYSTEMS LABORATORY
FALL 2019

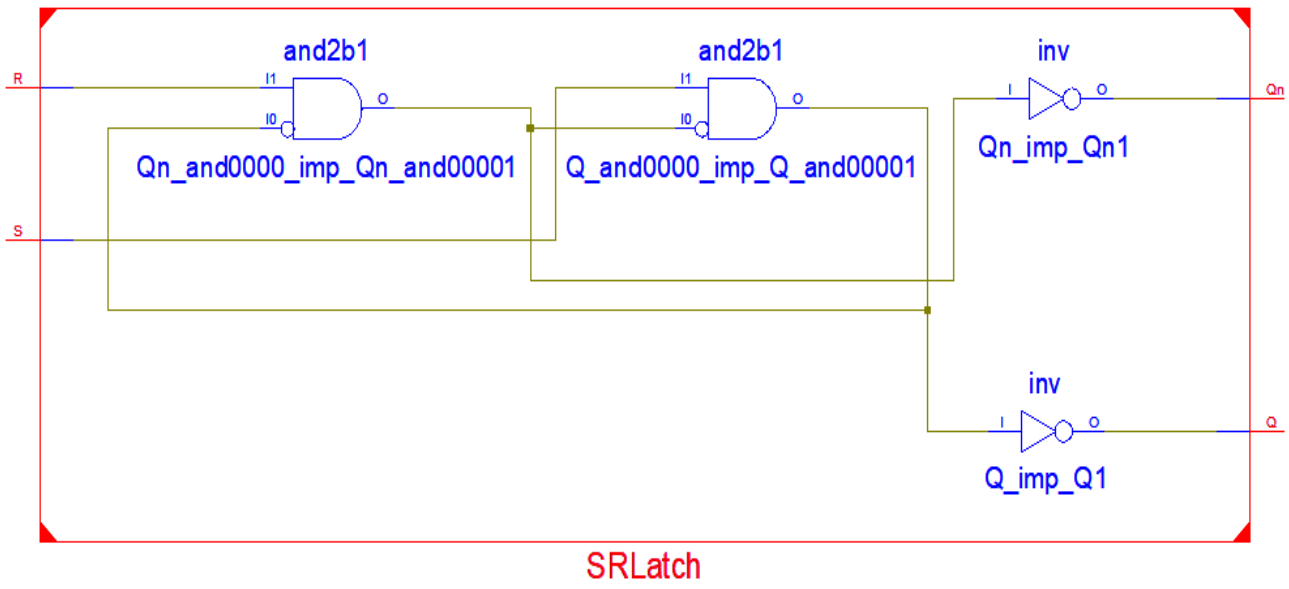
Egemen Can AYDUĞAN - 21728036

1-A) S-R Latch Circuit Diagram

VHDL Implementation:

```
entity SRLatch is
    Port ( S : in  STD_LOGIC;
           R : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end SRLatch;

architecture Behavioral of SRLatch is

signal T : STD_LOGIC_VECTOR(1 downto 0);

begin
Q <= S nand T(1);
Qn <= R nand T(0);
T(0) <= S nand T(1);
T(1) <= R nand T(0);
end Behavioral;
```

RTL Schematic:



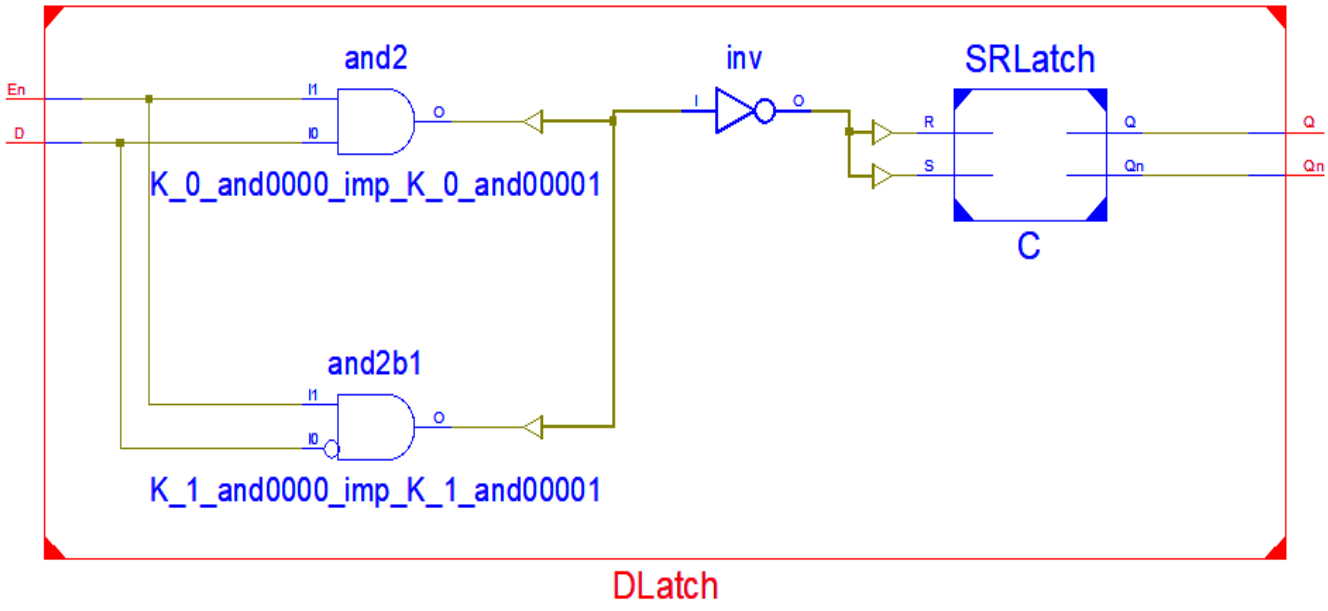
1-B) D Latch

Circuit Diagram

VHDL Implementation:

```
entity DLatch is
    Port ( D : in  STD_LOGIC;
           En : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end DLatch;

architecture Behavioral of DLatch is

component SRLatch
    Port ( S : in  STD_LOGIC;
           R : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;
signal K : STD_LOGIC_VECTOR(1 downto 0);

begin
K(0) <= D nand En;
K(1) <= (not D) nand En;
C : SRLatch port map (K(0),K(1),Q,Qn);
end Behavioral;
```

RTL Schematic:



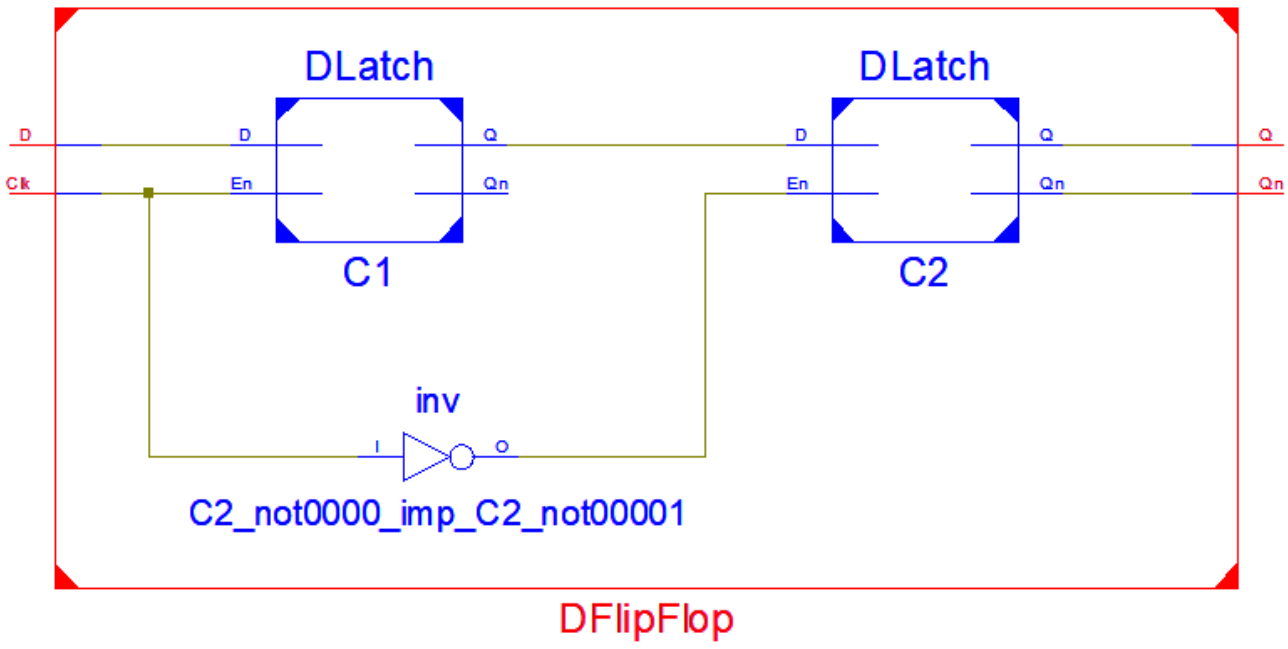
1-C) Positive Edge D Flip-Flop

Circuit Diagram

VHDL Implementation:

```
entity DFlipFlop is
    Port ( D : in  STD_LOGIC;
           Clk : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end DFlipFlop;

architecture Behavioral of DFlipFlop is

component DLatch
    Port ( D : in  STD_LOGIC;
           En : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;
signal T : STD_LOGIC_VECTOR(1 downto 0);

begin
C1 : DLatch port map(D,Clk,T(0),T(1));
C2 : DLatch port map(T(0),(not Clk),Q,Qn);
end Behavioral;
```

RTL Schematic:



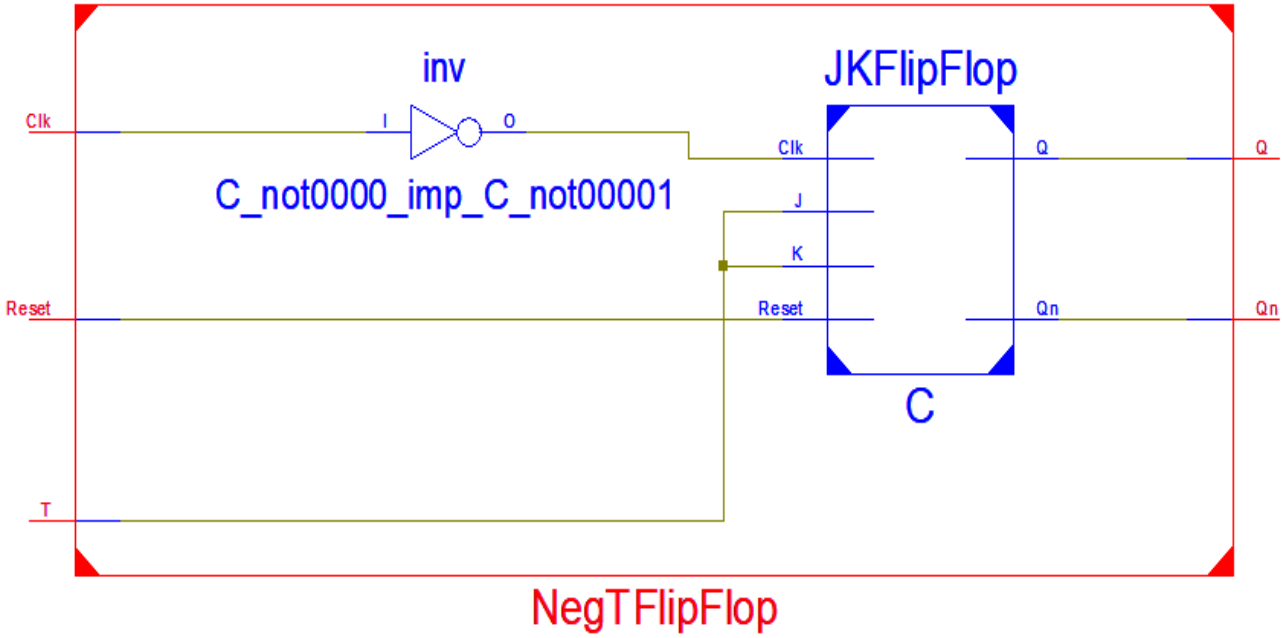
1-D) Negative Edge T Flip-Flop

Circuit Diagram

VHDL Implementation:

```
entity NegTFlipFlop is
    Port ( T : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clk : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end NegTFlipFlop;
architecture Behavioral of NegTFlipFlop is

component JKFlipFlop
    Port ( J : in  STD_LOGIC;
           K : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clk : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;

begin
C : JKFlipFlop port map(T,T,Reset,(not Clk),Q,Qn);
end Behavioral;
```

RTL Schematic:

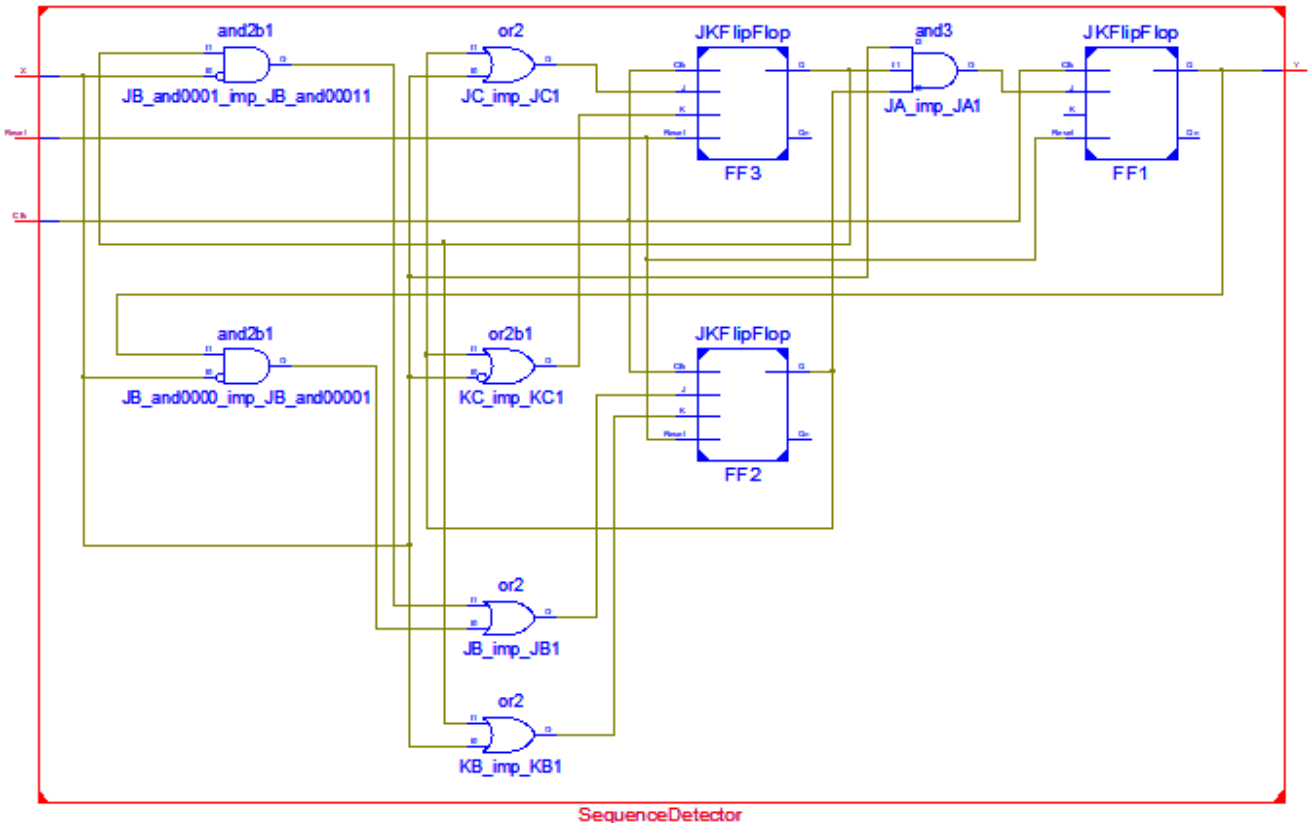


2) Circuit Diagram

VHDL Implementation:

```
entity SequenceDetector is
    Port ( X : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clk : in  STD_LOGIC;
           Y : out  STD_LOGIC);
end SequenceDetector;

architecture Behavioral of SequenceDetector is

component JKFlipFlop
    Port ( J : in  STD_LOGIC;
           K : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clk : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;
signal JA,JB,JC,KA,KB,KC : STD_LOGIC;
signal A,B,C : STD_LOGIC;
begin

JA <= B and C and x;
KA <= '1';
JB <= (A and (not x)) or (C and (not x));
KB <= x or C;
JC <= x or B;
KC <= (not x) or B;

FF1 : JKFlipFlop port map(JA,KA,Reset,Clk,A);
FF2 : JKFlipFlop port map(JB,KB,Reset,Clk,B);
FF3 : JKFlipFlop port map(JC,KC,Reset,Clk,C);

Y <= A ;

end Behavioral;
```

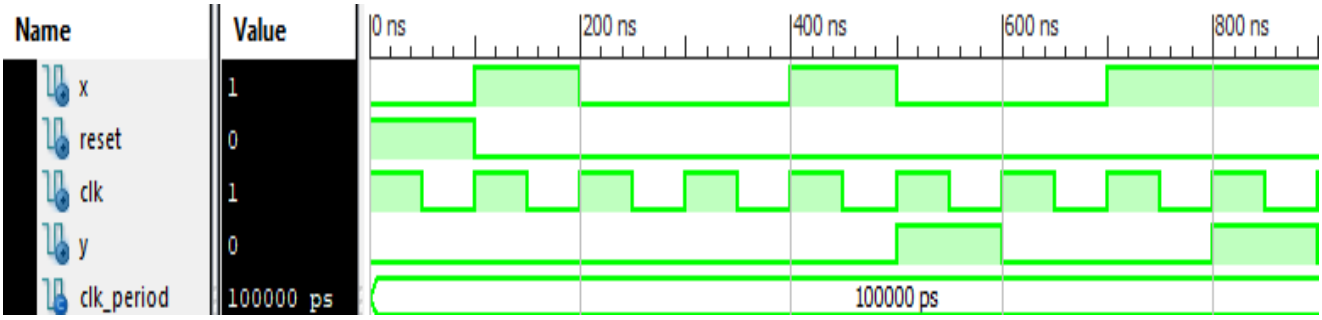
RTL Schematic:



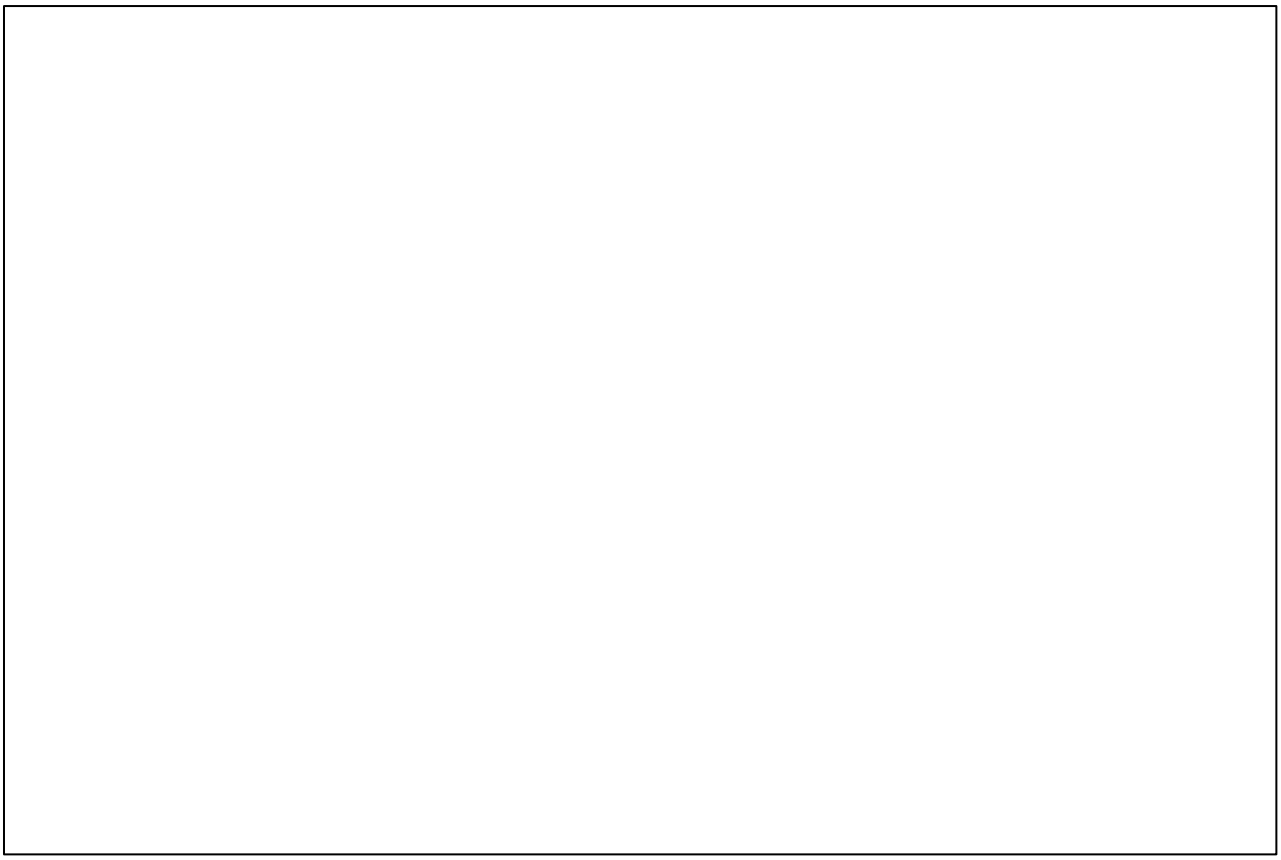
Test Bench Input

```
constant Clk_period : time := 100 ns;

   Clk_process :process
   begin
            Clk <= '1';
            wait for Clk_period/2;
            Clk <= '0';
            wait for Clk_period/2;
   end process;

   stim_proc: process
   begin
      -- hold reset state for 100 ns.
            Reset <= '1';
      wait for 100 ns;
            x <= '1';
            Reset <= '0';
      wait for 100 ns;
            x <= '0';
      wait for 100 ns;
            x <= '0';
      wait for 100 ns;
            x <= '1';
      wait for 100 ns;
            x <= '0';
      wait for 100 ns;
            x <= '0';
      wait for 100 ns;
            x <= '1';
      wait;
   end process;
```

Test Bench Result



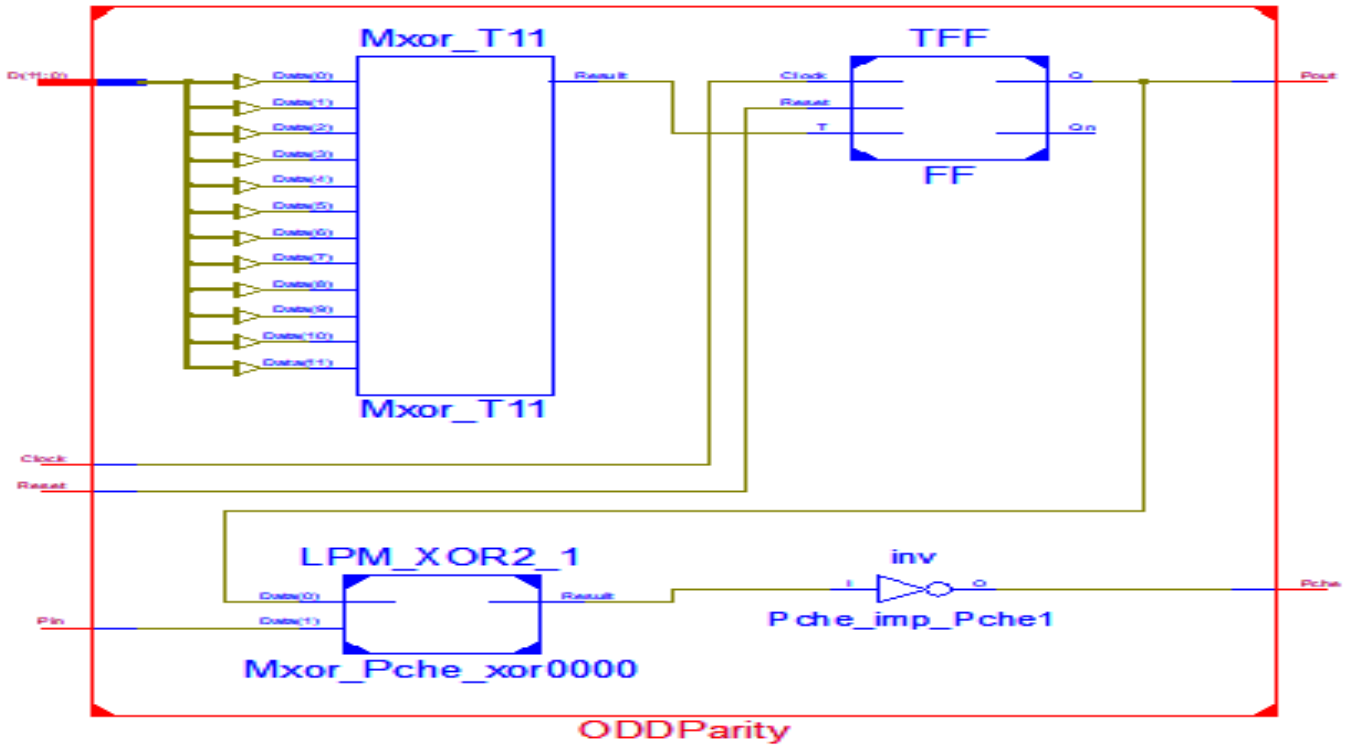
3) Circuit Diagram



VHDL Implementation:

```
entity ODDParity is
    Port ( D : in  STD_LOGIC_VECTOR (11 downto 0);
           Pin : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clock : in  STD_LOGIC;
           Pout : out  STD_LOGIC;
           Pche : out  STD_LOGIC);
end ODDParity;

architecture Behavioral of ODDParity is

component TFF
    Port ( T : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clock : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;

signal T1,T2,T3 : STD_LOGIC;
begin
T1 <= D(0) xor D(1) xor D(2) xor D(3) xor
D(4) xor D(5) xor D(6) xor D(7) xor
D(8) xor D(9) xor D(10) xor D(11) ;

FF : TFF port map(T1,Reset,Clock,T2,T3);
Pout <= T2;
Pche <= Pin xnor T2;
end Behavioral;
```

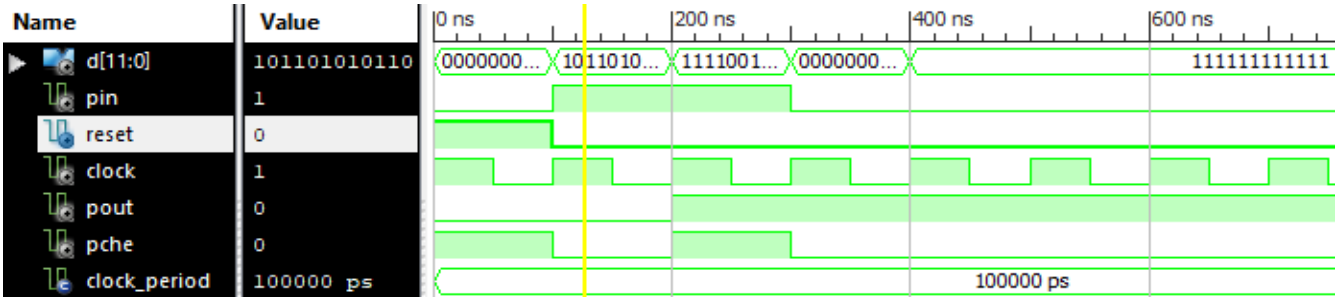
RTL Schematic:



Test Bench Input

```
constant Clock_period : time := 100 ns;

Clock_process :process
   begin
            Clock <= '1';
            wait for Clock_period/2;
            Clock <= '0';
            wait for Clock_period/2;
   end process;

begin

            Reset <= '1';
      wait for 100 ns;
             D <= "101101010110";
            Pin <= '1';
            Reset <='0';
      wait for 100 ns;
             D <= "111100101101";
            Pin <= '1';
      wait for 100 ns;
             D <= "000000001010";
            Pin <= '0';
      wait for 100 ns;
             D <= "111111111111";
            Pin <= '0';

      wait;
   end process;
```

Test Bench Result

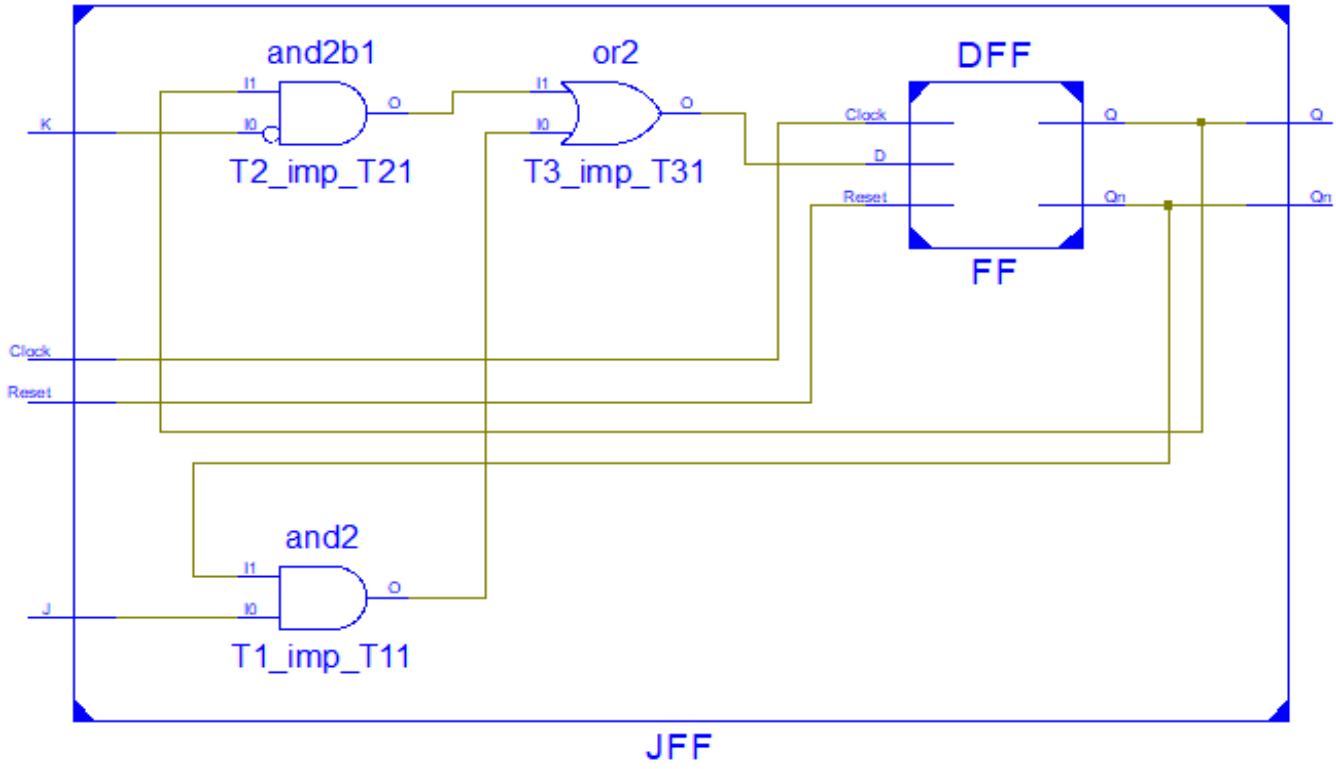


4) Circuit Diagram

VHDL Implementation:

```
entity JFF is
    Port ( J : in  STD_LOGIC;
           K : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clock : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end JFF;

architecture Behavioral of JFF is
component DFF
    Port ( D : in  STD_LOGIC;
           Reset : in  STD_LOGIC;
           Clock : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qn : out  STD_LOGIC);
end component;
signal T1,T2,T3,T4,T5 : STD_LOGIC;
begin
T1 <= J and T5;
T2 <= (not K) and T4;
T3 <= T1 or T2;
FF : DFF port map (T3,Reset,Clock,T4,T5);
Q <= T4;
Qn <= T5;
end Behavioral;
```

RTL Schematic:



Test Bench Input

```
constant Clock_period : time := 100 ns;

-- Clock process definitions
   Clock_process :process
   begin
            Clock <= '0';
            wait for Clock_period/2;
            Clock <= '1';
            wait for Clock_period/2;
   end process;

stim_proc: process
   begin
      Reset <= '1';
      wait for 100 ns;
            J <= '0';
            K <= '0';
            Reset <= '0';
      wait for 100 ns;
            J <= '0';
            K <= '1';
      wait for 100 ns;
            J <= '1';
            K <= '0';
      wait for 100 ns;
            J <= '1';
            K <= '1';
      wait;
   end process;
```

Test Bench Result