HDL code for Flipflop

D-FF D flip-flop with negative edge-triggering: Source Code

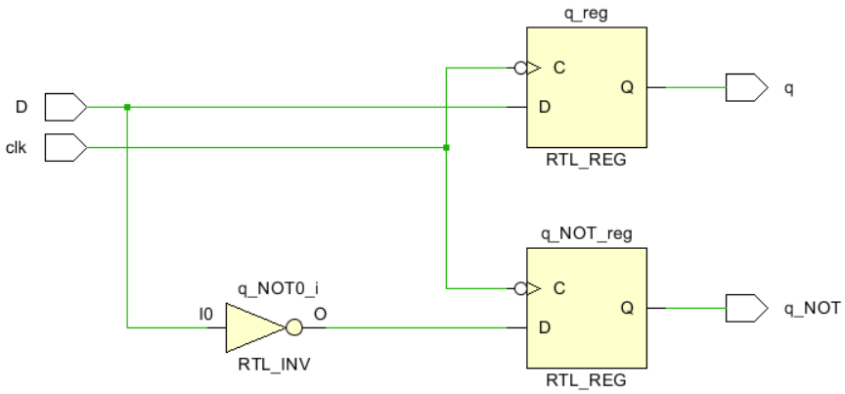
```
library ieee ;
use ieee.std_logic_1164.all;
use work.all;
---------------------------------------------
entity dff is
port (D:      in std_logic;
       CLK:     in std_logic;
       Q,Qb  : out std_logic
);
end dff;
---------------------------------------------
architecture behv of dff is
begin
process (D, CLK)
begin
if (CLK'EVENT AND CLK='0') then      ---this is for data flip-flop, for delay flip-flop use negative edge
    Q <= D;
   Qb <= not D;
end if;
end process;
end behv;
```

D-FF with negative edge-triggering: Testbench Code

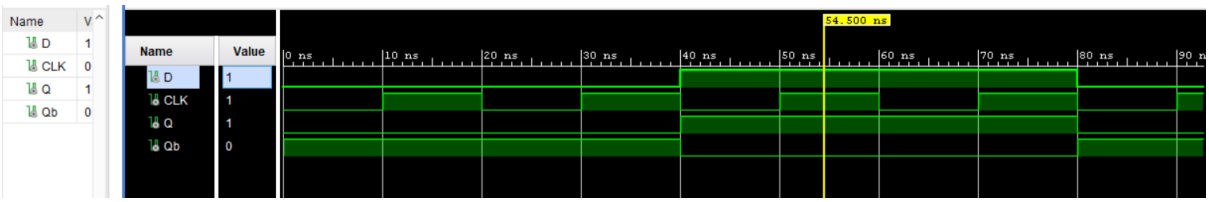
```
library ieee;
use ieee.std_logic_1164.all;

entity dff_TB is              -- entity declaration
end dff_TB;
-----------------------------------------------------------------
architecture TB of dff_TB is

  component dff
   Port ( D, CLK: in STD_LOGIC;
   Q, Qb : out STD_LOGIC);
   end component ;
    signal D, CLK, Q, Qb : STD_LOGIC;
    begin
   uut: dff port map(D => D,CLK => CLK,Q => Q,Qb => Qb);

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

```
    CLK <= '1';
    wait for 10 ns;
    end process;

    stim : process
    begin

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

    end process;
end TB;
```

RTL SCHEMATIC OF D-FF:



TESTBENCH WAVEFORM FOR D-FF:



JK flip-flop with active low asynchronous reset and positive edge-triggering:

source code

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

entity JKFF_RST is
    port( RST,CLK,J, K: in std_logic;
                    Q, Qbar : OUT std_logic);
end JKFF_RST;
architecture behavioral of JKFF_RST is
signal qn : std_logic;
```

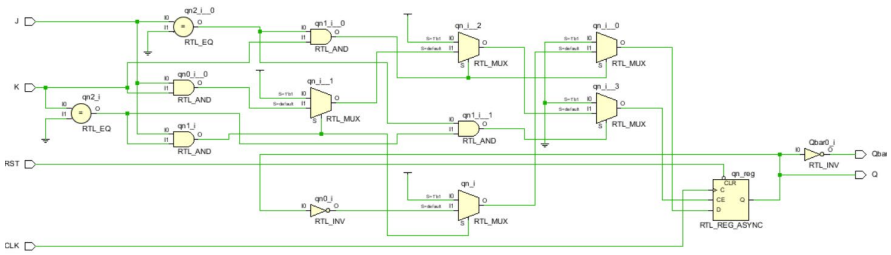
```
begin
process(clk, rst,J,K)

begin
if(rst = '0')then
  qn <= '0';
elsif(clk'event and clk = '1')then
    if(J='0' and K='0')then
       qn <= qn;
    elsif(J='0' and K='1')then
       qn <= '0';
    elsif(J='1' and K='0')then
       qn <= '1';
    elsif(J='1' and K='1')then
       qn <= not qn;
    end if;
end if;

Q <= qn;
Qbar <= not qn;

end process;

end Behavioral;
```

TESTBENCH CODE:

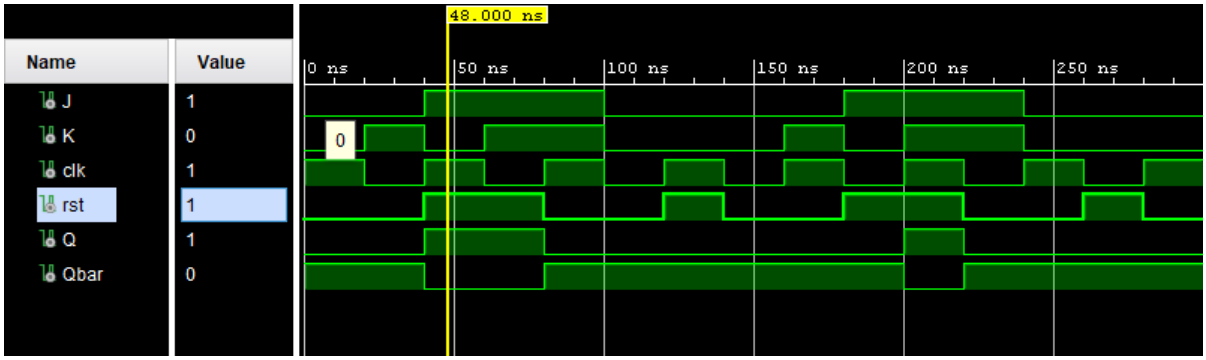
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity JK_FF_tb is
--  Port ( );
end JK_FF_tb;

architecture tB of JK_FF_tb is

component JKFF_RST is
port (RST,CLK, J, K: in std_logic;
        Q, Qbar : out std_logic);
end component;

signal J, K, clk, rst : std_logic;
signal Q, Qbar : std_logic;

begin
uut: JKFF_RST port map(J => J, K => K, clk => clk, rst => rst, Q => Q, Qbar => Qbar);

clock: process
begin
```

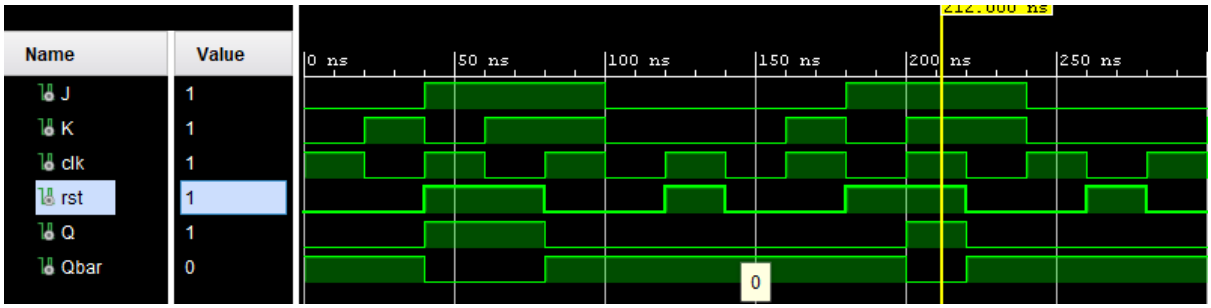
```
clk <= '1';
wait for 10 ns;
clk <= '0';
wait for 10 ns;
end process;

Force: process
begin
J <= '0';
K <= '0';
rst <= '0';
wait for 20 ns;

J <= '0';
K <= '1';
rst <= '0';
wait for 20 ns;
J <= '1';
K <= '0';
rst <= '1';
wait for 20 ns;

J <= '1';
K <= '1';
rst <= '1';
wait for 20 ns;

J <= '1';
K <= '1';
rst <= '0';
wait for 20 ns;

J <= '0';
K <= '0';
rst <= '0';
wait for 20 ns;

J <= '0';
K <= '0';
rst <= '1';
wait for 20 ns;
end process;
END tb;
```

RTL SCHEMATIC OF JK flip-flop with active low asynchronous reset and positive edge-triggering



TESTBENCH WAVEFORM





Obj-3: T-FF with active low asynchronous reset and positive edge-triggering:
SOURCE CODE:

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

entity T_FF_RST is
    Port ( RES,CLK, T: in  STD_LOGIC;
           Q,Q_NOT : BUFFER STD_LOGIC);
end T_FF_RST;

architecture Behavioral of T_FF_RST  is
begin
Q_NOT<= NOT Q;
PROCESS(T,CLK,RES)
```

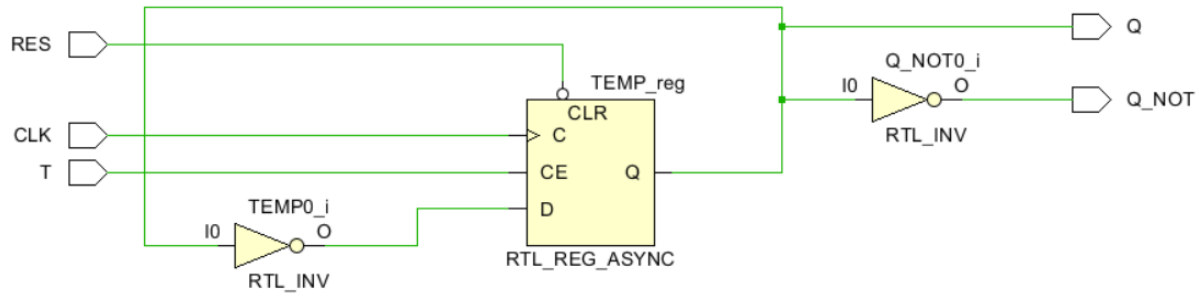
```
BEGIN
IF(RES='0') THEN
    Q <='0';
ELSIF (rising_edge(CLK)) THEN
     IF(T='1')THEN
       Q<= NOT Q;
     END IF;
END IF;

END PROCESS;
END Behavioral;
```

TESTBENCH CODE:

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

entity TFF_tb is
end entity;

architecture tb of TFF_tb is
component T_FF_RST is
Port (RES,CLK, T: in  STD_LOGIC;
        Q,Q_NOT : BUFFER STD_LOGIC);
end component;

signal T,CLK,RES,Q,QB : STD_LOGIC;

begin
uut: T_FF_RST port map(RES, CLK, T,Q,QB);

clock : process
begin

CLK <= '0';
wait for 20 ns;
CLK <= '1';
wait for 20 ns;

end process;

stim: process
begin
RES <= '0';
T <= '0';
wait for 20 ns;
```

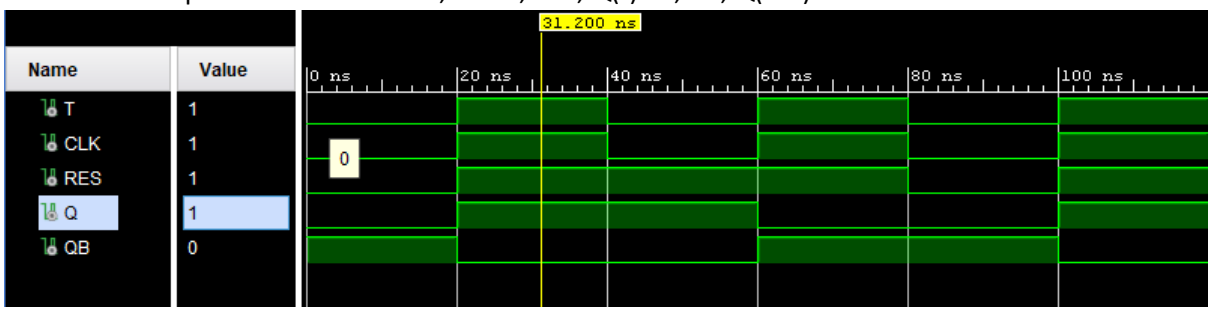
```
T <= '1';
RES <= '1';
wait for 20 ns;

RES <= '1';
T <= '0';
wait for 20 ns;
RES <= '1';
T <= '1';
wait for 20 ns;

end process;
end tb;
```

RTL SCHEMATIC OF T-FF with active low asynchronous reset and positive edge-triggering

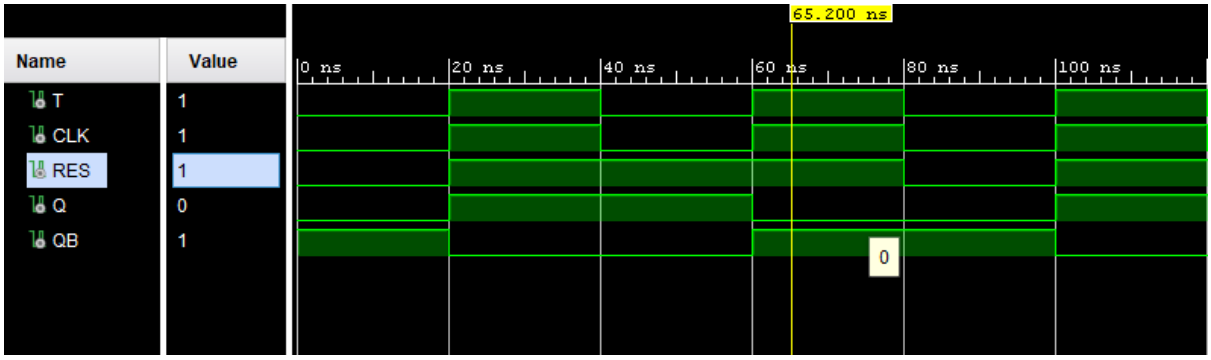


<u>TESTBENCH WAVEFORM</u>

1. Cursor point indicates CLK=1,RST=1, T=1, Q(t) =0, SO, Q(t+1)=1



2. Cursor point indicates CLK=1,RST=1, T=1, Qt =1, So, Q(T+1)=0


Name
Value
T
1
CLK
1
RES
1
Q
0
QB
1
65.200 ns
0 ns
20 ns
40 ns
60 ns
80 ns
100 ns
0