HDL code for Flipflop

<u>D-FF D flip-flop with negative edge-triggering: Source Code</u>

```
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 behavioural of dff is
begin
process (D, CLK)
begin
if (CLK'EVENT AND CLK='0') then ------this is for negative edge triggering of D-FF
Q <= D;
Qb <= not D;
end if;
end process;
end behavioural;
```

<u>JK flip-flop with active low asynchronous reset and positive edge-triggering:</u>

SOURCE CODE

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

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

architecture behavioural 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 Behavioural;
```

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 Behavioural 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 Behavioural;
```