library ieee;

use ieee.std\_logic\_1164.all;

entity JK\_FF is

port(J,K,Reset: in std\_logic;

Q:out std\_logic);

end JK\_FF;

architecture JK\_a of JK\_FF is

signal clk,intQ: std\_logic:='0';

signal JK: std\_logic\_vector(1 downto 0);

begin

JK <= J & K;

tact: process

begin

clk<=not clk;

wait for 50 ns;

end process;

process(clk,reset)

begin

if reset ='1' then

intQ<='0';

else

case JK is

when "00" => intQ<=intQ;

when "01" => intQ<='0';

when "10" => intQ<='1';

when "11" => intQ<= not intQ;

when others => intQ<='0';

end case;

end if;

end process;

Q<=intQ;

end JK\_a;

library ieee;

use ieee.std\_logic\_1164.all;

entity TB\_JK is

end TB\_JK;

architecture TestBench of TB\_JK is

component JK\_FF is

port(J,K,Reset: in std\_logic;

Q:out std\_logic);

end component;

signal J,K,Reset,Q: std\_logic;

begin

ust: JK\_FF port map(J,K,Reset,Q);

process

begin

Reset<='1';

J<='0';

K<='0';

wait for 50 ns;

Reset<='1';

J<='0';

K<='1';

wait for 50 ns;

Reset<='1';

J<='1';

K<='0';

wait for 50 ns;

Reset<='1';

J<='1';

K<='1';

wait for 50 ns;

Reset<='0';

J<='0';

K<='0';

wait for 50 ns;

Reset<='0';

J<='0';

K<='1';

wait for 50 ns;

Reset<='0';

J<='1';

K<='0';

wait for 50 ns;

Reset<='0';

J<='1';

K<='1';

wait for 50 ns;

end process;

end TestBench;