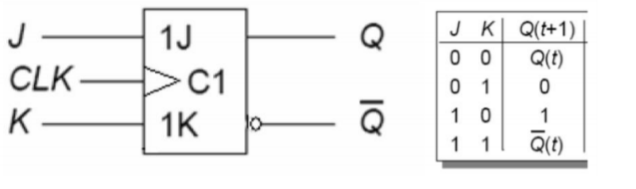
Lab 7 – VHDL

Name: Nhut Quang Tran

Group: NTIVIS14

TURKU UNIVERSITY OF APPLIED SCIENCES

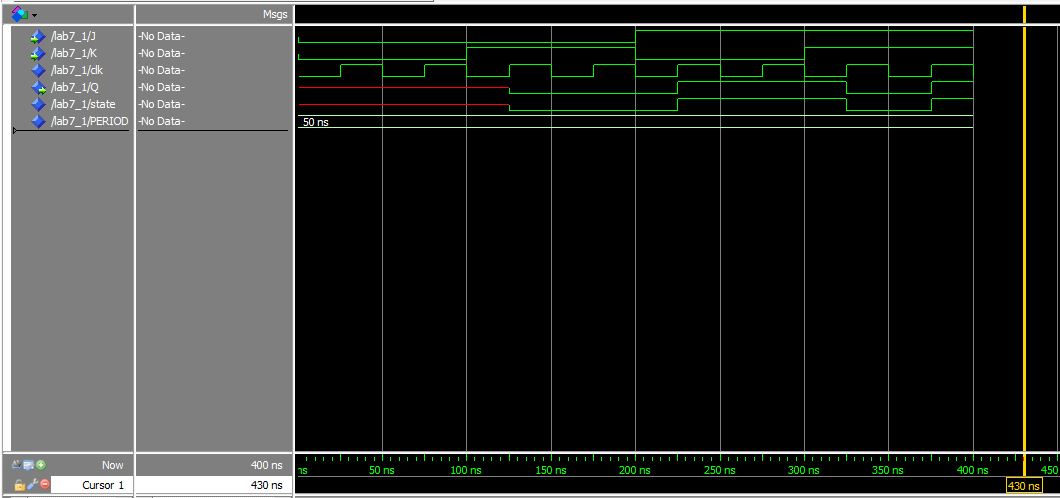
**Exercise 1:**

1. **JK flip-flop works according to given table. Make VHDL-model of it.**

**The code:**

|  |
| --- |
| LIBRARY ieee ;  USE ieee.std\_logic\_1164.all ;  use IEEE.NUMERIC\_STD.all;  use IEEE.std\_logic\_unsigned.all;  ENTITY Lab7\_1 IS  PORT (J,K: IN STD\_LOGIC;  --clk: STD\_LOGIC;  Q: OUT STD\_LOGIC );  END Lab7\_1;  ARCHITECTURE JKFlipFlop OF Lab7\_1 IS  signal state : std\_logic;  constant PERIOD: time := 50ns;  signal clk : std\_logic := '0';  BEGIN  P0:PROCESS(J,K,clk)  BEGIN  clk <= not clk after PERIOD/2;  IF rising\_edge(clk) THEN  IF( (J OR K) = '0' ) THEN  state <= state;    ELSIF(J = '1' AND K = '1') THEN  state <= not state;    ELSE  IF(K = '1') THEN  state <= '0';  ElSIF(J = '1') THEN  state <= '1';  END IF;  END IF;  END IF;  END PROCESS P0;  Q <= state;  END JKFlipFlop; |

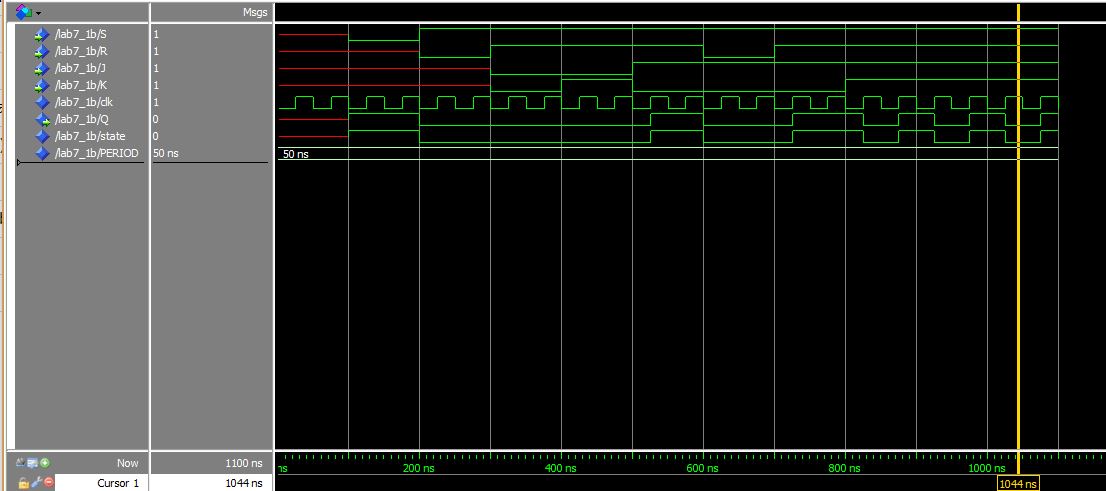
**The simulating picture:**

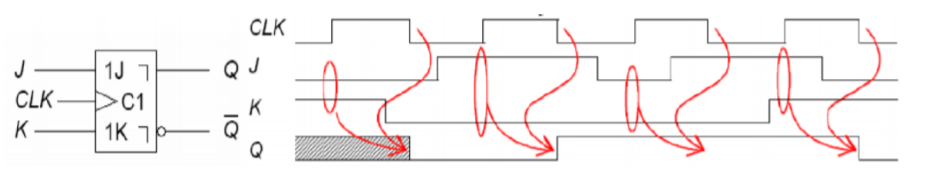


1. **Add zero-active Reset and set to this model.**

**The code:**

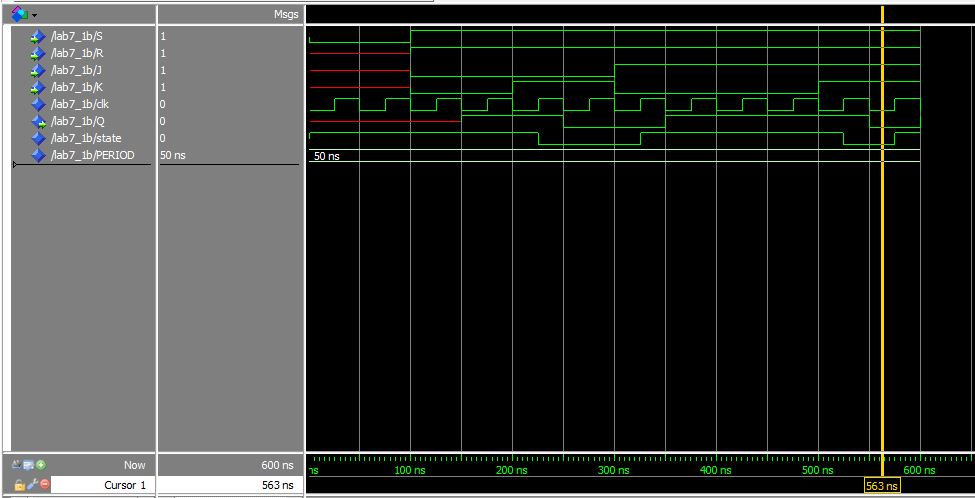
|  |
| --- |
| LIBRARY ieee ;  USE ieee.std\_logic\_1164.all ;  use IEEE.NUMERIC\_STD.all;  use IEEE.std\_logic\_unsigned.all;  ENTITY Lab7\_1b IS  PORT (J,K,R,S: IN STD\_LOGIC;  Q: OUT STD\_LOGIC );  END Lab7\_1b;  ARCHITECTURE JKFlipFlop OF Lab7\_1b IS  signal state : std\_logic;  constant PERIOD: time := 50ns;  signal clk : std\_logic := '0';  BEGIN  P0:PROCESS(J,K,R,S,clk)  BEGIN  clk <= not clk after PERIOD/2;  IF(S = '0') then  state <= '1';  ELSIF(R = '0') then  state <= '0';  ELSIF rising\_edge(clk) THEN  IF( (J OR K) = '0' ) THEN  state <= state;    ELSIF(J = '1' AND K = '1') THEN  state <= not state;    ELSE  IF(K = '1') THEN  state <= '0';  ElSIF(J = '1') THEN  state <= '1';  END IF;  END IF;  END IF;  END PROCESS P0;  Q <= state;  END JKFlipFlop; |

**The simulating picture:**

1. **Master-Slave flip-flop works so that inputs are read during the rising edge of the clk, but actual change in the output happens during the falling edge. Build a VHDL model for a master-slave JK flip-flop.**

**The code:**

|  |
| --- |
| LIBRARY ieee ;  USE ieee.std\_logic\_1164.all ;  use IEEE.NUMERIC\_STD.all;  use IEEE.std\_logic\_unsigned.all;  ENTITY Lab7\_1c IS  PORT (J,K,R,S: IN STD\_LOGIC;  Q: OUT STD\_LOGIC );  END Lab7\_1c;  ARCHITECTURE JKFlipFlop OF Lab7\_1c IS  signal state : std\_logic;  constant PERIOD: time := 50ns;  signal clk : std\_logic := '0';  BEGIN  P0:PROCESS(J,K,R,S,clk)  BEGIN  clk <= not clk after PERIOD/2;  IF(S = '0') then  state <= '1';  ELSIF(R = '0') then  state <= '0';  ELSIF rising\_edge(clk) THEN  IF( (J OR K) = '0' ) THEN  state <= state;    ELSIF(J = '1' AND K = '1') THEN  state <= not state;    ELSE  IF(K = '1') THEN  state <= '0';  ElSIF(J = '1') THEN  state <= '1';  END IF;  END IF;  ELSIF falling\_edge(clk) THEN  Q <= state;  END IF;  END PROCESS P0;  END JKFlipFlop; |

**The simulating picture:**

**Exercise 2:**

**T flip-flop works so that if T = 0 output doesn´t change and if T = 1 output changes.**

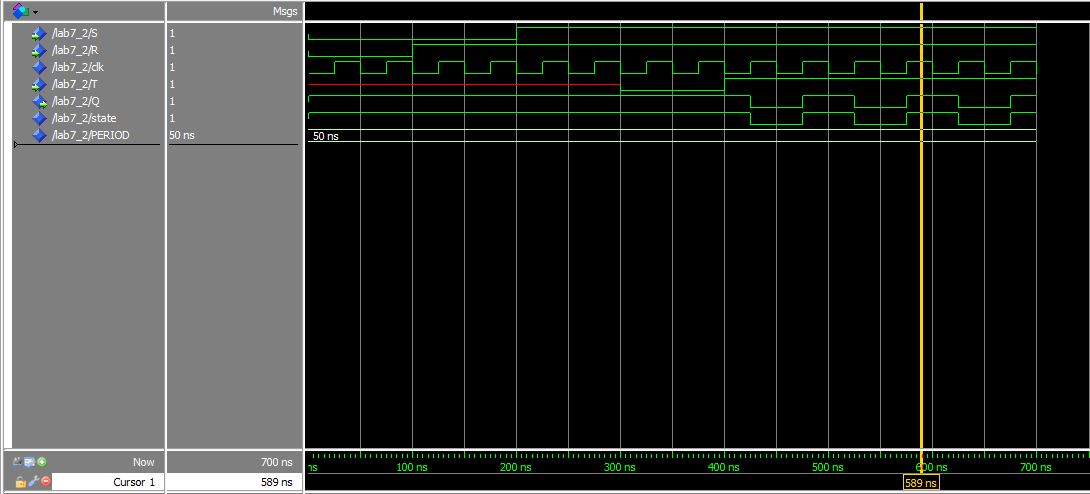
**a) Make a VHDL-model of a T flip-flop.**

**b) Add one-active Reset and set to this model**

**The code:**

|  |
| --- |
| LIBRARY ieee ;  USE ieee.std\_logic\_1164.all ;  use IEEE.NUMERIC\_STD.all;  use IEEE.std\_logic\_unsigned.all;  ENTITY Lab7\_2 IS  PORT (T,R,S: IN STD\_LOGIC;  Q: OUT STD\_LOGIC);  END Lab7\_2;  ARCHITECTURE JKFlipFlop OF Lab7\_2 IS  signal state : std\_logic := '0';  constant PERIOD: time := 50ns;  signal clk : std\_logic := '0';  BEGIN  P0:PROCESS(T,R,S,clk)  BEGIN  clk <= not clk after PERIOD/2;  IF(S = '0') then  state <= '1';  ELSIF(R = '0') then  state <= '0';  ELSIF rising\_edge(clk) THEN  IF( T = '0' ) THEN  state <= state;    ELSIF(T = '1') THEN  state <= not state;    END IF;  END IF;  END PROCESS P0;  Q <= state;  END JKFlipFlop; |

**The simulating picture:**

****