Name: Siddhartha Dhar

VHDL Program for 4 bit Asynchronus Up/Down Counter with PreSet and Clear

Git-hub: Siddhartha1999

Problem Statement:

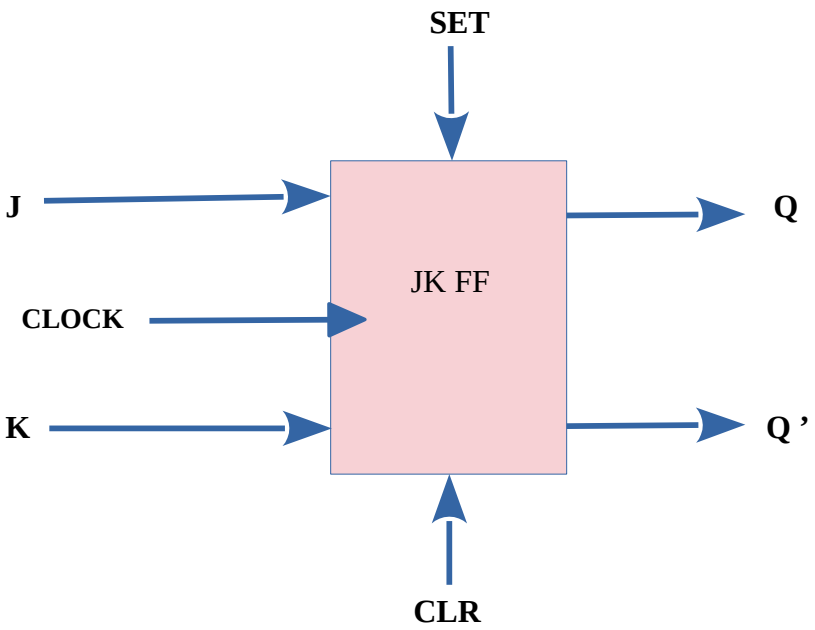
VHDL Structural code for 4 bit Up Down Asynchronous Counter with Asynchronous SET and CLR

Environment Used : *Xilinx 14.7 ISE Design Suite*

Design of Circuit Diagram :

Let make the Design using 4 JK Flip Flops

Properties of the JK Flip Flop that had been made as an Component in the Counter Circuit

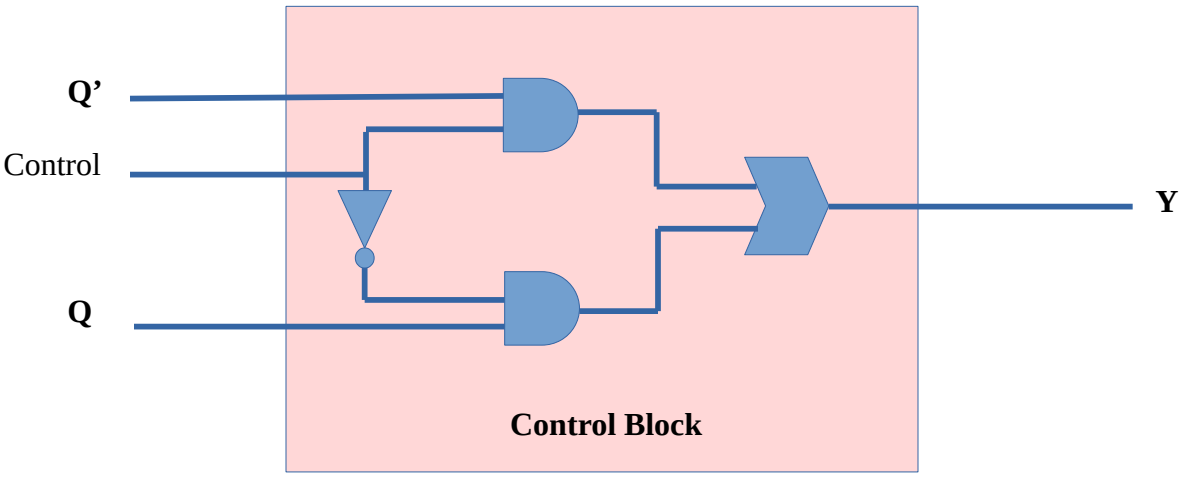


(J = 0) and (K = 0) then --- No Change
(J = 0) and (K = 1) then --- RESET
(J = 1) and (K = 0) then --- SET
(J = 1) and (K = 1) then --- Toogle

(SET = 1) then Set the Counter Irrespective of Clock
(CLR = 1) then Clear the Counter Irrespective of Clock

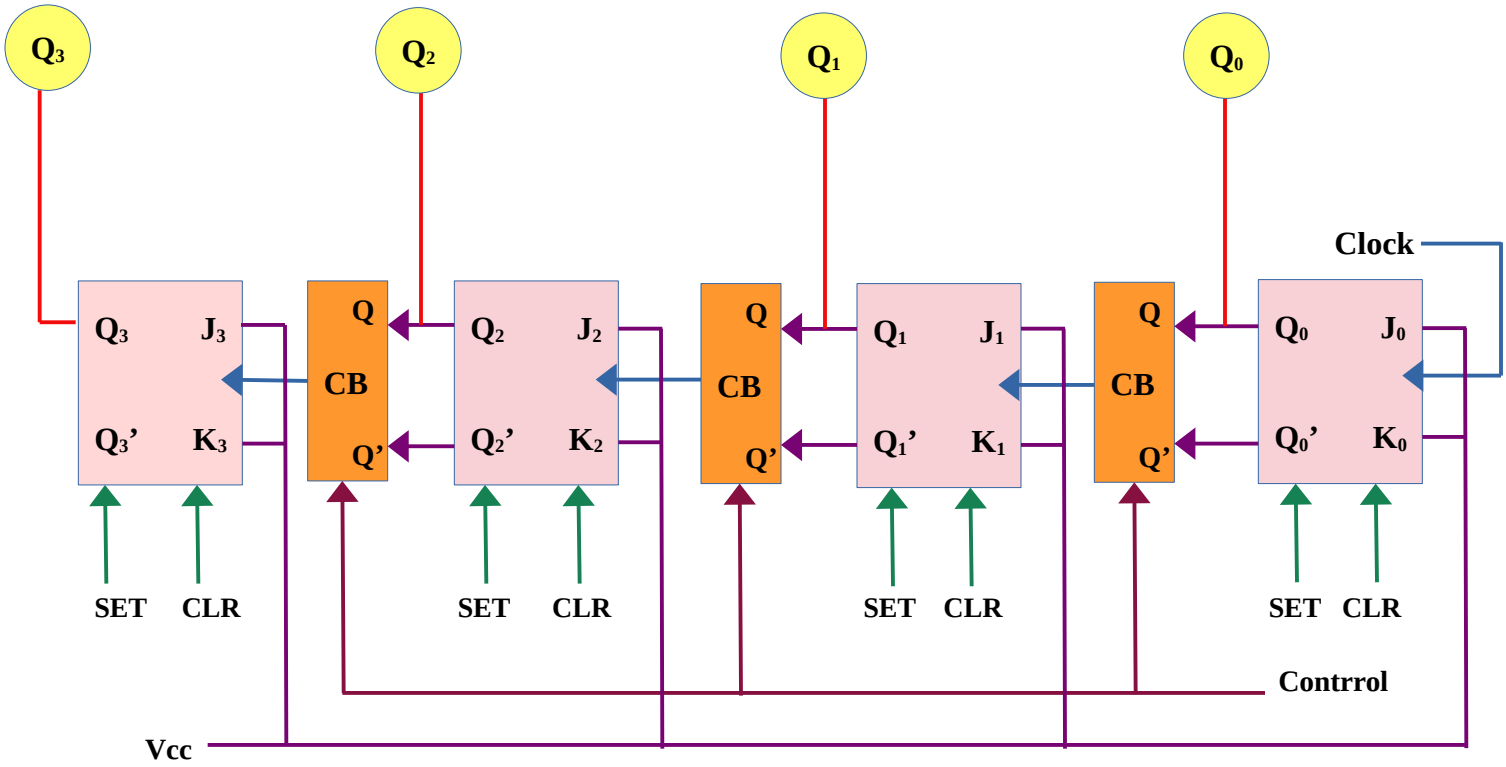
The JK Flip Flop is Edge Triggered

Control Block Component



If the Control = '1' then Q will propagated to Y
else Q' will be propagated to Y

Control => UP'/DOWN (*i.e. when Control is 0 then Count UP else Count Down* ***)***
Now let us have a look at the Asynchronous Design of the 4 bit up-down counter with Preset and Clear



VHDL Code Implementing 4 bit Asynchronous Up Down Counter using Structural Modeling :

```
----------------------------------------------------------------------------------
-- University:     University of Calcutta
-- Engineer:       Siddhartha Dhar
--
-- Module Name:    Counter_UD - Behavioral
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Counter_UD is
    Port ( CLOCK : in  STD_LOGIC; -- System Clock
                       SET : in STD_LOGIC;    -- Asynchronous Preset
                       CLR : in STD_LOGIC;    -- Asynchronous Clear
                       CONTROL : in STD_LOGIC; -- Control is ( UP'/DOWN )
           Q : out  STD_LOGIC_VECTOR (3 downto 0);
                       Qb : out  STD_LOGIC_VECTOR (3 downto 0));
end Counter_UD;

architecture COUNTER_UD_ARCH of Counter_UD is
-------------------------------------------------JK Flip Flop component
component JK_FF
    Port ( CLOCK : in STD_LOGIC;
                       J : in  STD_LOGIC;
           K : in  STD_LOGIC;
           SET : in  STD_LOGIC;
           CLR : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           Qb : out  STD_LOGIC);
end component;
-------------------------------------------------Control Block Component to select the
counting order (UP/DOWN)
component Control_Block
    Port ( Control : in  STD_LOGIC;
           Q : in  STD_LOGIC;
           Qb : in  STD_LOGIC;
           Y : out  STD_LOGIC);
end component;
-------------------------------------------------Signal Definitions
signal F: STD_LOGIC_VECTOR(3 downto 0);
signal C: STD_LOGIC_VECTOR(3 downto 0);
signal CLK: STD_LOGIC_VECTOR(2 downto 0);
```

```
begin

      T0: JK_FF PORT MAP(CLOCK=>CLOCK, J=>'1', K=>'1', SET=>SET, CLR=>CLR, Q=>F(0),
            Qb=>C(0));   -- Q0(JK Flip Flop for the LSB of the Counter)
      T1: Control_Block PORT MAP(Control=>Control, Q=>F(0), Qb=>C(0), Y=>CLK(0)); --
            Control Block Q0

      T2: JK_FF PORT MAP(CLOCK=>CLK(0), J=>'1', K=>'1', SET=>SET, CLR=>CLR, Q=>F(1),
            Qb=>C(1));  -- Q1(JK Flip Flop)
      T3: Control_Block PORT MAP(Control=>Control, Q=>F(1), Qb=>C(1), Y=>CLK(1)); --
            Control Block Q1

      T4: JK_FF PORT MAP(CLOCK=>CLK(1), J=>'1', K=>'1', SET=>SET, CLR=>CLR, Q=>F(2),
            Qb=>C(2));  -- Q2(JK Flip Flop)
      T5: Control_Block PORT MAP(Control=>Control, Q=>F(2), Qb=>C(2), Y=>CLK(2)); --
            Control Block Q2

      T6: JK_FF PORT MAP(CLOCK=>CLK(2), J=>'1', K=>'1', SET=>SET, CLR=>CLR, Q=>F(3),
            Qb=>C(3));  -- Q3(JK FlipFlop for the MSB of the Counter)

      Q <= F; -- Sorting the Signal element to the Actual Qouput Q
      Qb <= C;
end COUNTER_UD_ARCH;
```

VHDL JK Flip Flop Component Behavioral Modeling :

```
----------------------------------------------------------------------------------
-- University:     University of Calcutta
-- Engineer:       Siddhartha Dhar
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-----------------------------------Entity Definition
entity JK_FF is
    Port ( CLOCK : in STD_LOGIC;
                      J : in  STD_LOGIC;
           K : in  STD_LOGIC;
           SET : in  STD_LOGIC;
           CLR : in  STD_LOGIC;
           Q : out  STD_LOGIC := '0';
           Qb : out  STD_LOGIC := '1');
end JK_FF;
---------------------------------------Architecture Definition
architecture JK_FF_ARCH of JK_FF is
begin
p1: process(CLOCK, SET, CLR)
variable tmp: STD_LOGIC := '0';
begin
      if(SET='1')then
            tmp := '1';
            Q <= '1';
            Qb <= '0';
      elsif(CLR='1')then
            tmp := '0';
            Q <= '0';
            Qb <= '1';
      elsif(rising_edge(CLOCK))then --If Clock's Poisitive Edge Trigerred
            if(J='0' and K='0')then    -- No Change
                  tmp := tmp;
            elsif(J='1' and K='1')then -- Toogle
                  tmp := not tmp;
            elsif(J='0' and K='1')then -- Reset
                  tmp := '0';
            else                       -- Set
                  tmp := '1';
            end if;
      end if;
      Q <= tmp;
      Qb <= not tmp;
end process p1;
end JK_FF_ARCH;
```

VHDL Control-Block Component Data Flow Modeling :

```
----------------------------------------------------------------------------------
-- University:     University of Calcutta
-- Engineer:       Siddhartha Dhar
--
-- Module Name:    Control_Block - Behavioral
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

---------------------------------------Entity Definition
entity Control_Block is
    Port ( Control : in  STD_LOGIC;
           Q : in  STD_LOGIC;
           Qb : in  STD_LOGIC;
           Y : out  STD_LOGIC);
end Control_Block;
-------------------------------------------------Architecture Definition
architecture CB_DFLOW of Control_Block is
signal a, b, cinv: STD_LOGIC;
begin
      cinv <= not Control;
      a <= (Q and Control);
      b <= (Qb and cinv);
      Y <= (a or b);
end CB_DFLOW;
```

Test Bench to test 4 bit Up/Down Asynchronous Counter :

```
--------------------------------------------------------------------------------
-- University:    University of Calcutta
-- Engineer:      Siddhartha Dhar
--
-- Project Name:  Counter_UP_DWN_4b_STRCTL
--------------------------------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY COUNTER_UD_TB IS
END COUNTER_UD_TB;

ARCHITECTURE behavior OF COUNTER_UD_TB IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT Counter_UD
    PORT(
         CLOCK : IN  std_logic;
         SET : IN  std_logic;
         CLR : IN  std_logic;
         CONTROL : IN  std_logic;
         Q : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;


   --Inputs
   signal CLOCK : std_logic := '0';
   signal SET : std_logic := '0';
   signal CLR : std_logic := '0';
   signal CONTROL : std_logic := '0';

      --Outputs
   signal Q : std_logic_vector(3 downto 0);

   -- Clock period definitions
   constant CLOCK_period : time := 10 ns;

BEGIN

      -- Instantiate the Unit Under Test (UUT)
   uut: Counter_UD PORT MAP (
          CLOCK => CLOCK,
          SET => SET,
          CLR => CLR,
          CONTROL => CONTROL,
          Q => Q
        );
   -- Clock process definitions
   CLOCK_process :process
   begin
             CLOCK <= '0';
             wait for CLOCK_period/2;
             CLOCK <= '1';
             wait for CLOCK_period/2;
   end process;
```

```
   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
      wait for 100 ns;

      wait for CLOCK_period*10;

      -- insert stimulus here
             Control <= '1';
             wait for CLOCK_period*10;
             Control <= '0';
             wait for CLOCK_period*10;
             wait for CLOCK_period*10;
             SET <= '1';
             CLR <= '0';
             wait for CLOCK_period/2;
             SET <= '0';
             CLR <= '0';
             wait for CLOCK_period*10;
             SET <= '0';
             CLR <= '1';
             wait for CLOCK_period/2;
             SET <= '0';
             CLR <= '0';
             wait for CLOCK_period*10;
   end process;

END;
```