Flip Flops and Counters

VLSI Systems

Assignment-7

PREPARED BY

Priyank Lohariwal

BCSE-IV

001710501055

Jadavpur University

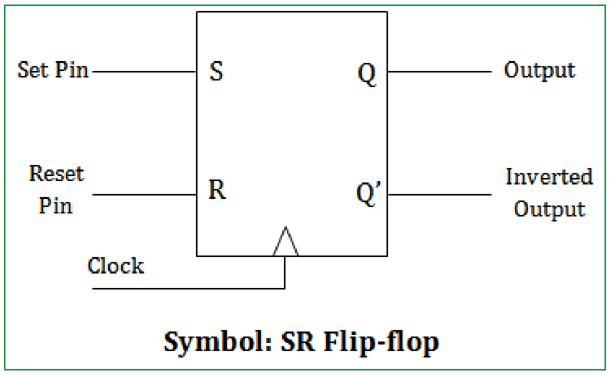
Description

Designing flip flops and counters

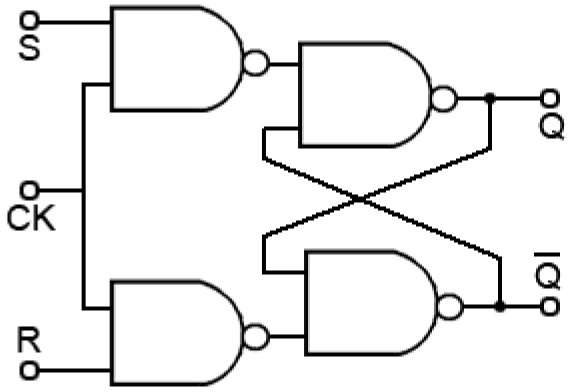
1. SR Flip Flop
2. 3 bit up counter using SR Flip Flop
    a. Using component instantiation
    b. Using behavioral modelling
3. JK Flip Flop
4. 3 bit up counter using JK Flip Flop
    a. Using component instantiation
    b. Using behavioural modelling
5. T Flip Flop
6. 3 bit up counter using T Flip FLop
7. 3 bit down counter using T Flip Flop

SR Flip Flop

Block Diagram



Circuit Diagram



Truth Table

S	R	Q_{n+1}
0	0	**Q_n**
0	1	**0**
1	0	**1**
1	1	**x**

Code

Implementation

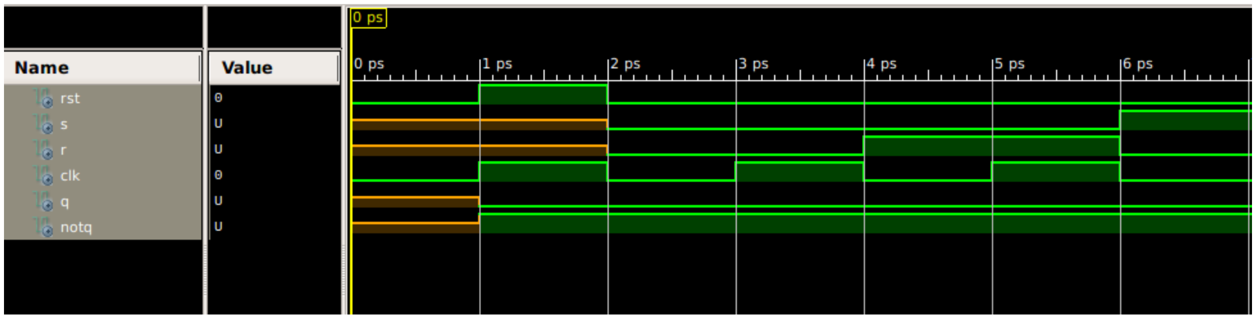
```
architecture Behavioral of srff is
   shared variable q1, notq1: std_logic;
begin
   p1: process(clk, rst)
   begin
       if rst = '1' then
           q <= '0';
           notq <= '1';
           q1 := '0';
           notq1 := '1';
       elsif (clk'event and clk = '1') then
           if (s = '0' and r = '0') then
               q <= q1;
               notq <= notq1;
           elsif (s = '0' and r = '1') then
               q <= '0';
               notq <= '1';
               q1 := '0';
               notq1 := '1';
           elsif (s = '1' and r = '0') then
               q <= '1';
               notq <= '0';
               q1 := '1';
               notq1 := '0';
           else
               q  <= 'Z';
               notq <= 'Z';
               q1 := 'Z';
               notq1 := 'Z';
           end if;
       end if;
   end process;
end Behavioral;
```

Test Bench

```
      ARCHITECTURE behavior OF testbench IS

      COMPONENT srff
              Port ( s : in  STD_LOGIC;
                     r : in  STD_LOGIC;
                     clk : in  STD_LOGIC;
                     rst: in  STD_LOGIC;
                     q : out  STD_LOGIC;
                     notq : out  STD_LOGIC);
      END COMPONENT;

      SIGNAL rst : std_logic;
      SIGNAL s : std_logic;
      SIGNAL r : std_logic;
      SIGNAL clk : std_logic;
      SIGNAL q : std_logic;
      SIGNAL notq : std_logic;
BEGIN
-- Component Instantiation
       uut: srff PORT MAP(
               s => s,
                   r => r,
                   clk => clk,
                   rst => rst,
                   q => q,
                   notq => notq
       );
--  Test Bench Statements
   tb : PROCESS
   BEGIN
       clk <= '0';
       rst <= '0';
       wait for 1ps;

       rst <= '1';
       clk <= '1';
       wait for 1ps;

       rst <= '0';

       loop1: loop
           s <= '0';
```

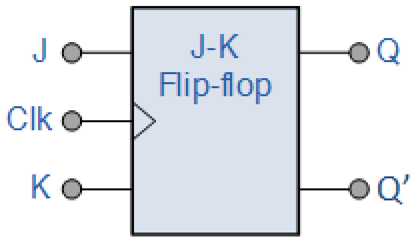
```
            r <= '0';
            clk <= '0';
            wait for 1ps;
            clk <= '1';
            wait for 1ps;

            s <= '0';
            r <= '1';
            clk <= '0';
            wait for 1ps;
            clk <= '1';
            wait for 1ps;

            s <= '1';
            r <= '0';
            clk <= '0';
            wait for 1ps;
            clk <= '1';
            wait for 1ps;

            s <= '1';
            r <= '1';
            clk <= '0';
            wait for 1ps;
            clk <= '1';
            wait for 1ps;

        end loop;
    END PROCESS tb;
END;
```

Timing Diagram

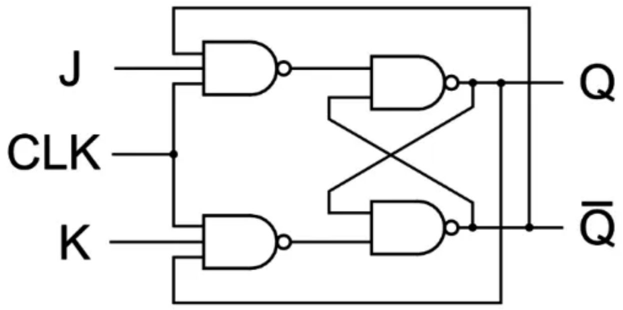


JK Flip Flop

Block Diagram



Circuit Diagram



Truth Table

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	Q_n'

Code

Implementation

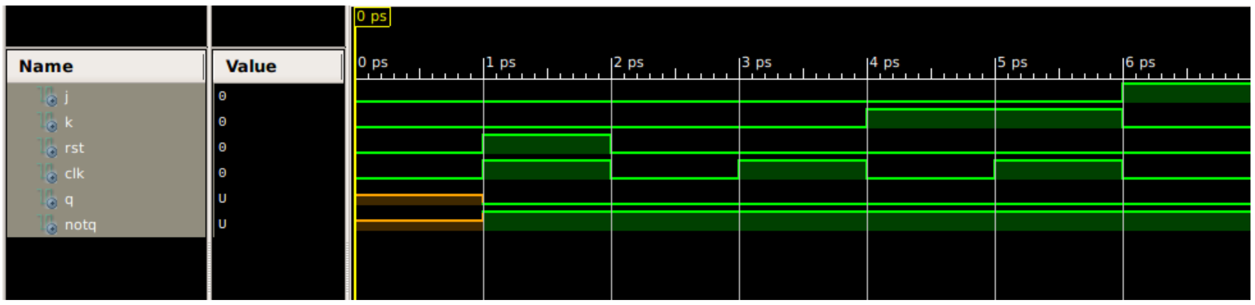
```
architecture Behavioral of jkff is
    shared variable q1, notq1: std_logic;
begin
   p1: process(clk, rst)
        variable temp: std_logic;
   begin
        if rst = '1' then
            q <= '0';
            notq <= '1';
            q1 := '0';
            notq1 := '1';
        elsif (clk'event and clk = '1') then
            if (j = '0' and k = '0') then
                q <= q1;
                notq <= notq1;
            elsif (j = '0' and k = '1') then
                q <= '0';
                notq <= '1';
                q1 := '0';
                notq1 := '1';
            elsif (j = '1' and k = '0') then
                q <= '1';
                notq <= '0';
                q1 := '1';
                notq1 := '0';
            else
                q  <= notq;
                notq <= q1;
                temp := q1;
                q1 := notq1;
                notq1 := temp;
            end if;
        end if;
   end process;
end Behavioral;
```

Test Bench

```
ARCHITECTURE behavior OF jkff_test_bench IS
   COMPONENT jkff
   PORT(
        j : IN  std_logic;
        k : IN  std_logic;
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : INOUT  std_logic;
        notq : INOUT  std_logic
       );
   END COMPONENT;
  --Inputs
  signal j : std_logic := '0';
  signal k : std_logic := '0';
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';
  signal q : std_logic;
  signal notq : std_logic;

BEGIN
   -- Instantiate the Unit Under Test (UUT)
  uut: jkff PORT MAP (
         j => j,
         k => k,
         rst => rst,
         clk => clk,
         q => q,
         notq => notq
       );

  -- Stimulus process
  stim_proc: process
  begin
     clk <= '0';
           rst <= '0';
           wait for 1ps;

           rst <= '1';
           clk <= '1';
           wait for 1ps;

           rst <= '0';
```

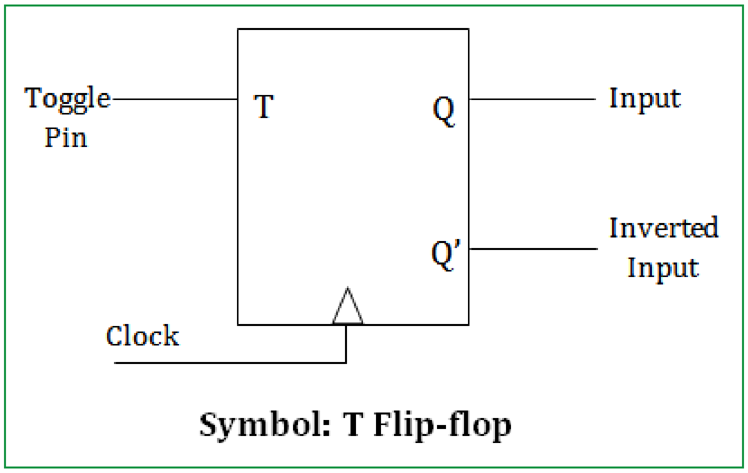
```
            loop1: loop
                j <= '0';
                k <= '0';
                clk <= '0';
                wait for 1ps;
                clk <= '1';
                wait for 1ps;

                j <= '0';
                k <= '1';
                clk <= '0';
                wait for 1ps;
                clk <= '1';
                wait for 1ps;

                j <= '1';
                k <= '0';
                clk <= '0';
                wait for 1ps;
                clk <= '1';
                wait for 1ps;

                j <= '1';
                k <= '1';
                clk <= '0';
                wait for 1ps;
                clk <= '1';
                wait for 1ps;
            end loop;
    end process;
  END;
```

Timing Diagram

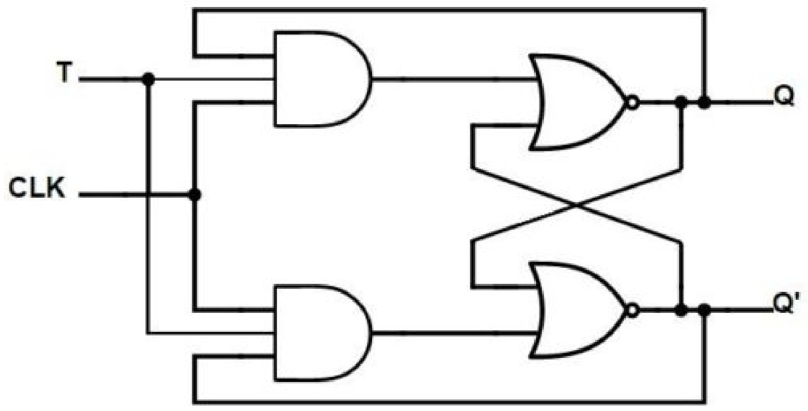


T Flip Flop

Block Diagram



Circuit Diagram



Truth Table

Qn	T	Qn+1
0	0	0
0	1	1
1	0	1
1	1	0

Code

Implementation

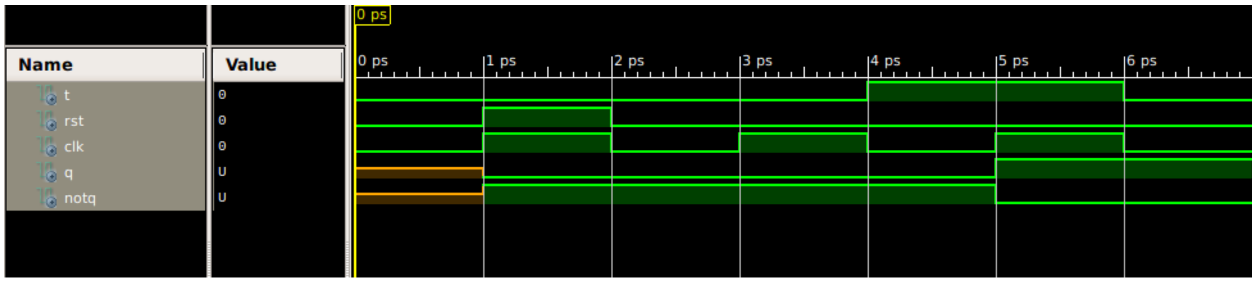
```
architecture Behavioral of tff is
   shared variable q1, notq1: std_logic;
begin
   p1: process(clk, rst)
   begin
       if rst = '1' then
           q <= '0';
           notq <= '1';
           q1 := '0';
           notq1 := '1';
       elsif (clk'event and clk = '1') then
           if t = '0' then
               q <= q1;
               notq <= notq1;
           else
               q <= notq1;
               notq <= q1;
               q1 := notq1;
               notq1 := not q1;
           end if;
       end if;
   end process;
end Behavioral;
```

Test Bench

```
ARCHITECTURE behavior OF tff_test_bench IS
   COMPONENT tff
   PORT(
        t : IN  std_logic;
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : OUT  std_logic;
        notq : OUT  std_logic
       );
   END COMPONENT;
  --Inputs
  signal t : std_logic := '0';
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';
 --Outputs
```

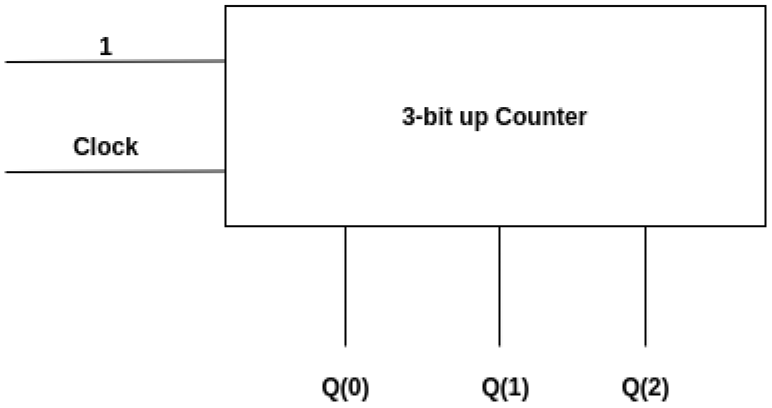
```
    signal q : std_logic;
    signal notq : std_logic;
  BEGIN
   -- Instantiate the Unit Under Test (UUT)
    uut: tff PORT MAP (
           t => t,
           rst => rst,
           clk => clk,
           q => q,
           notq => notq
         );
    -- Stimulus process
    stim_proc: process
    begin
     clk <= '0';
     rst <= '0';
     wait for 1ps;

     rst <= '1';
     clk <= '1';
     wait for 1ps;

     rst <= '0';

     loop1: loop
       t <= '0';
       clk <= '0';
       wait for 1ps;
       clk <= '1';
       wait for 1ps;

       t <= '1';
       clk <= '0';
       wait for 1ps;
       clk <= '1';
       wait for 1ps;
     end loop;
   end process;
  END;
```

Timing Diagram



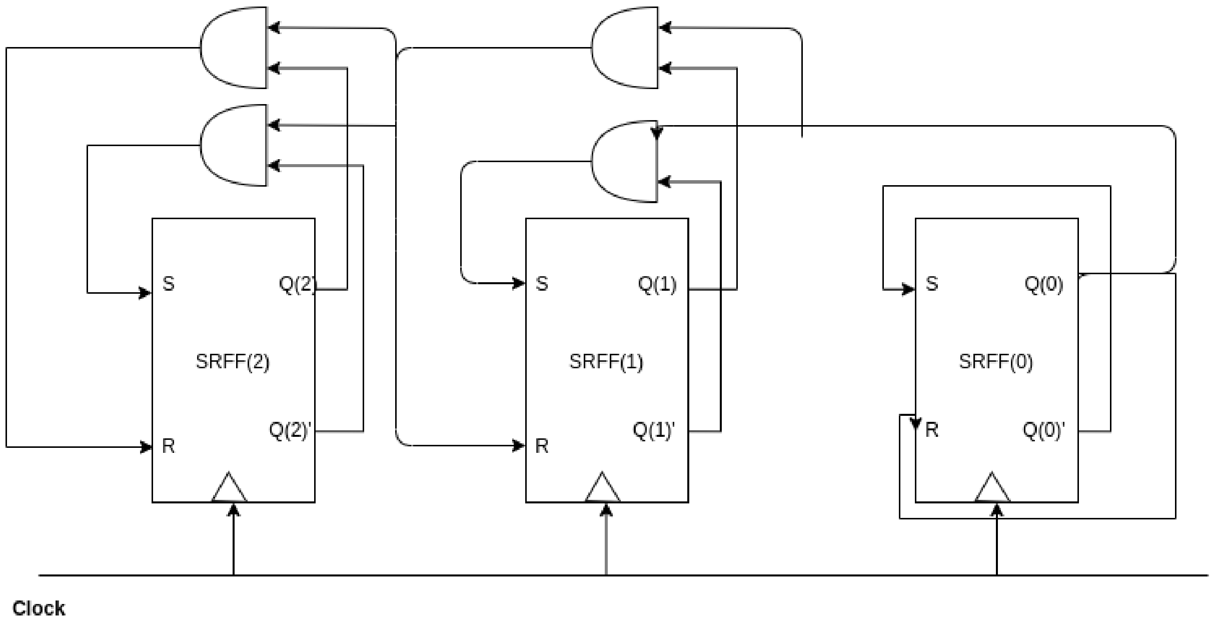
3-bit up counter

Block Diagram



Using SR Flip Flops

Circuit Diagram



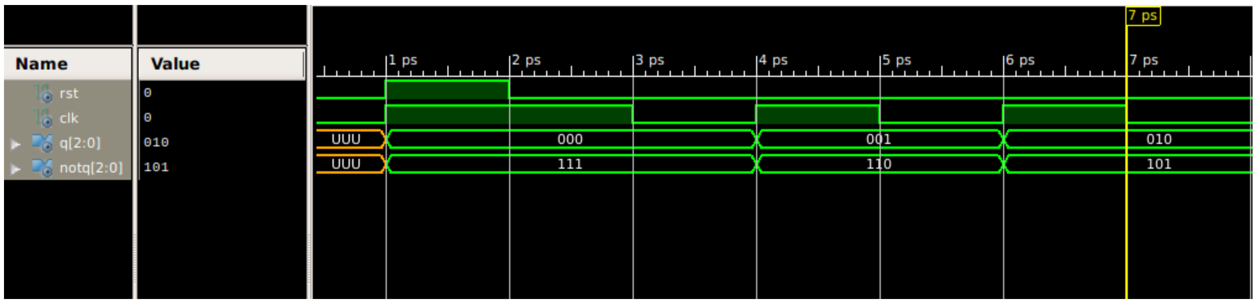
Code

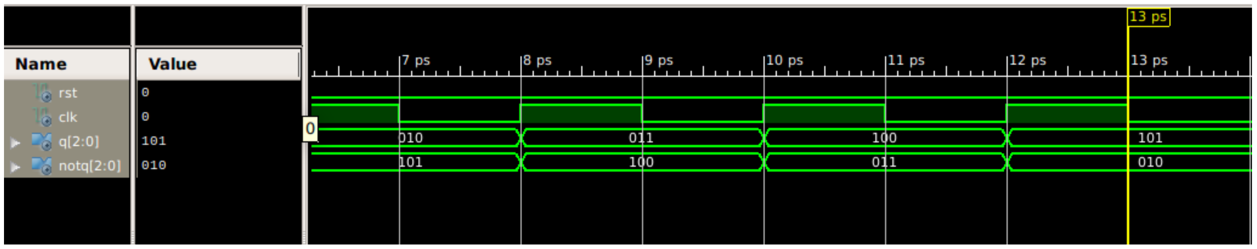
```
architecture Behavioral of counter3bit_a is
   COMPONENT srff
   Port ( s : in  STD_LOGIC;
         r : in  STD_LOGIC;
         clk : in  STD_LOGIC;
         rst: in  STD_LOGIC;
         q : out  STD_LOGIC;
         notq : out  STD_LOGIC);
   END COMPONENT;
   signal s,r: std_logic_vector(2 downto 0);
begin
   s(2) <= notq(2) and q(1) and q(0);
   r(2) <= q(2) and q(1) and q(0);
   s(1) <= notq(1) and q(0);
   r(1) <= q(1) and q(0);
   s(0) <= notq(0);
   r(0) <= q(0);

   gen1: for k in 0 to 2 generate
       proc: srff port map(s(k), r(k), clk, rst, q(k), notq(k));
   end generate;
end Behavioral;
```

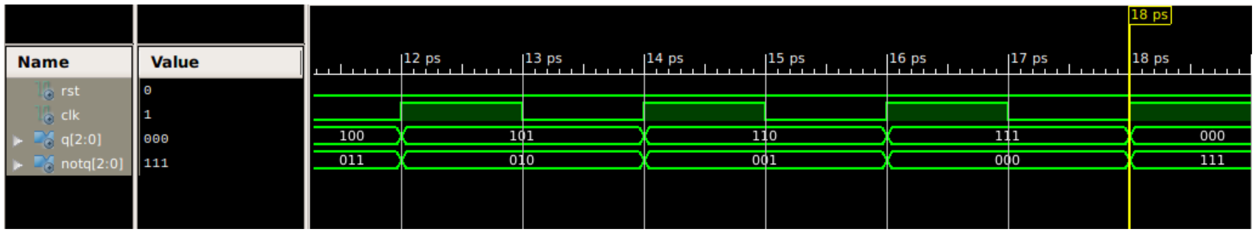
Test Bench

```
ARCHITECTURE behavior OF counter3bit_a_test_bench IS
   COMPONENT counter3bit_a
   PORT(
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : INOUT  std_logic_vector(2 downto 0);
        notq : INOUT  std_logic_vector(2 downto 0)
       );
   END COMPONENT;
  --Inputs
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';
  signal q : std_logic_vector(2 downto 0);
  signal notq : std_logic_vector(2 downto 0);
BEGIN
  -- Instantiate the Unit Under Test (UUT)
  uut: counter3bit_a PORT MAP (
         rst => rst,
         clk => clk,
         q => q,
         notq => notq
       );
  -- Stimulus process
  stim_proc: process
  begin
     clk <= '0';
     rst <= '0';
     wait for 1ps;
     rst <= '1';
     clk <= '1';
     wait for 1ps;
     rst <= '0';
     loop1: loop
        clk <= '1';
        wait for 1ps;
        clk <= '0';
        wait for 1ps;
     end loop;
  end process;
END;
```

Timing Diagram

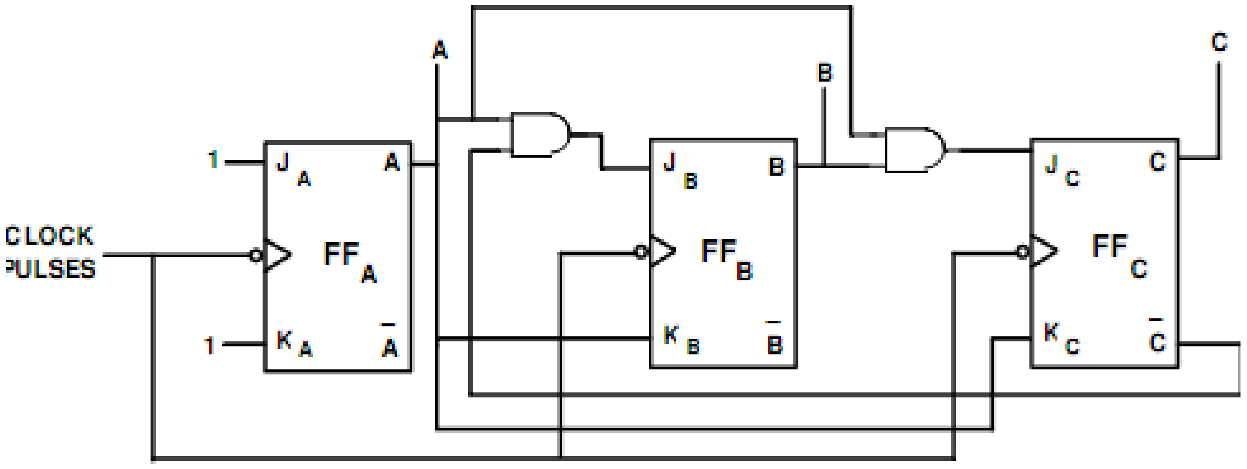






Using JK Flip Flops

Circuit Diagram



Code

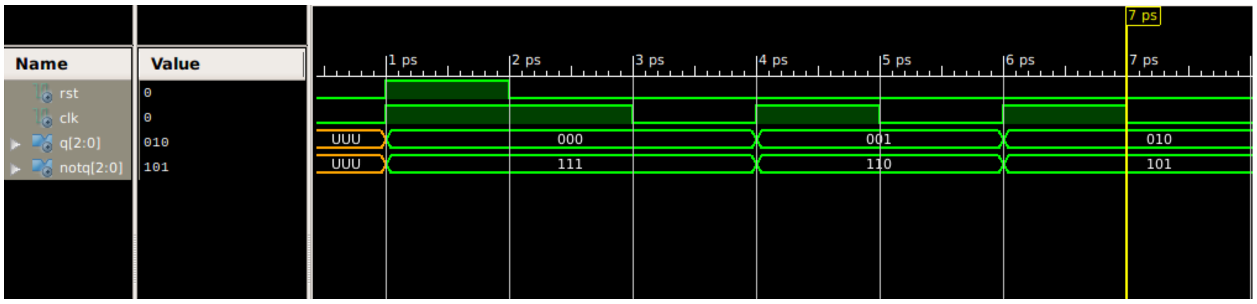
```
architecture Behavioral of counter3bitupjkff is
   COMPONENT jkff
   PORT(
        j : IN  std_logic;
        k : IN  std_logic;
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : INOUT  std_logic;
        notq : INOUT  std_logic
       );
   END COMPONENT;
    signal j, k: std_logic_vector(2 downto 0);
begin

   j(2) <= q(1) and q(0);
   k(2) <= q(1) and q(0);
   j(1) <= q(0);
   k(1) <= q(0);
   j(0) <= '1';
   k(0) <= '1';

   gen: for i in 0 to 2 generate
       proc: jkff port map(j(i), k(i), rst, clk, q(i), notq(i));
   end generate;

end Behavioral;
```

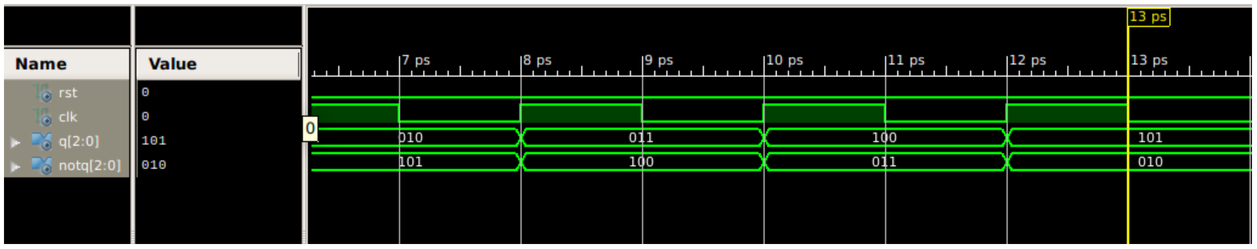
Test Bench

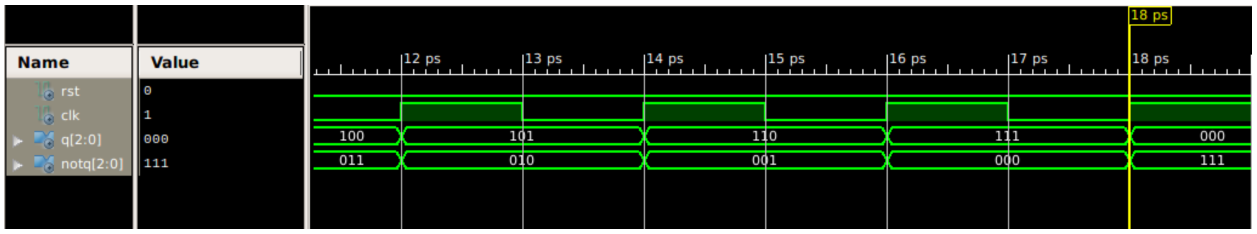
```
ARCHITECTURE behavior OF counter3bitjkff_test_bench IS
   COMPONENT counter3bitjkff
   PORT(
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : INOUT  std_logic_vector(2 downto 0);
        notq : INOUT  std_logic_vector(2 downto 0)
       );
   END COMPONENT;

  --Inputs
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';
```

```
      signal q : std_logic_vector(2 downto 0);
      signal notq : std_logic_vector(2 downto 0);

    BEGIN
      -- Instantiate the Unit Under Test (UUT)
      uut: counter3bitjkff PORT MAP (
             rst => rst,
             clk => clk,
             q => q,
             notq => notq
           );

      -- Stimulus process
      stim_proc: process
      begin

         clk <= '0';
         rst <= '0';
         wait for 1ps;

         rst <= '1';
         clk <= '1';
         wait for 1ps;

         rst <= '0';

         loop1: loop
            clk <= '1';
            wait for 1ps;
            clk <= '0';
            wait for 1ps;
         end loop;
    end process;
 END;
```

Timing Diagram

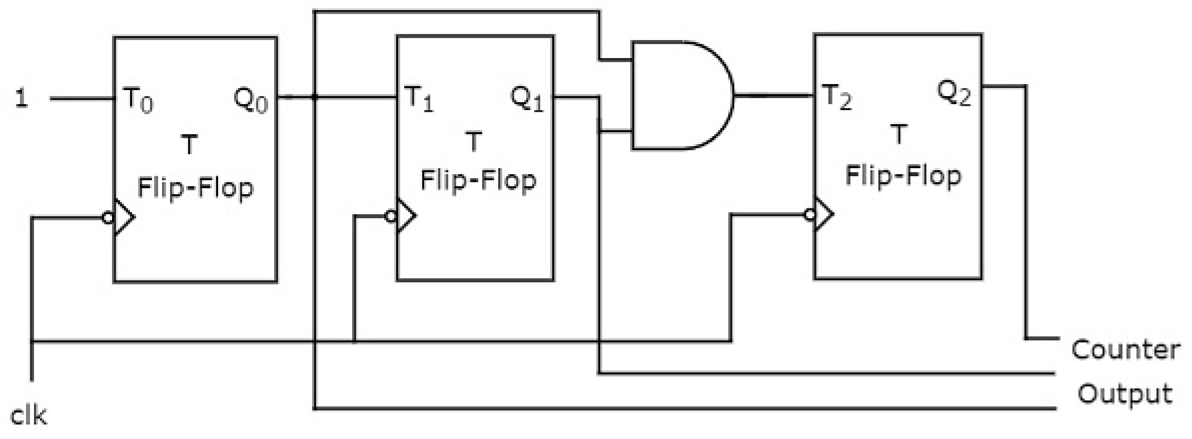






Using T Flip Flops

Circuit Diagram



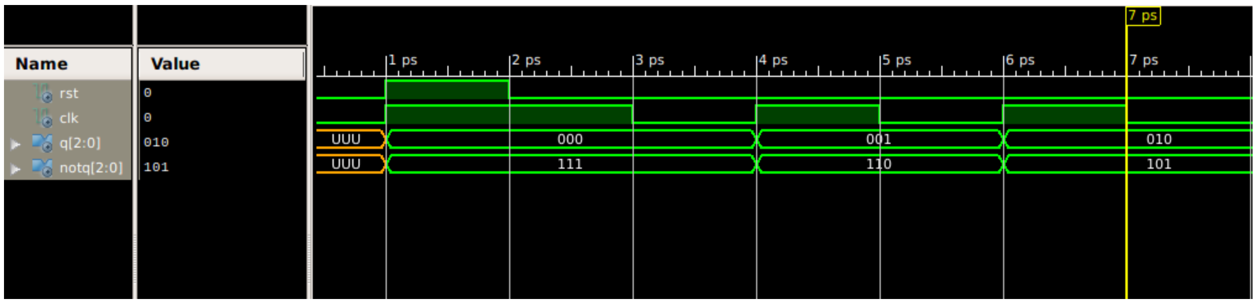
Code

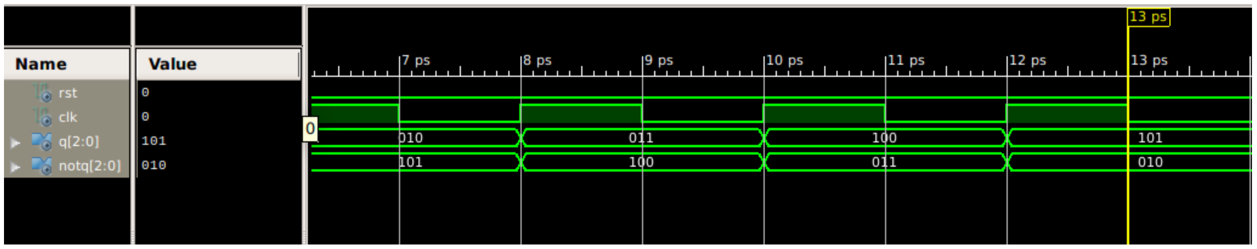
```
architecture Behavioral of counter3bitup_tff is
   COMPONENT tff
   PORT(
        t : IN  std_logic;
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : OUT  std_logic;
        notq : OUT  std_logic
       );
   END COMPONENT;

    signal t: std_logic_vector(2 downto 0);
begin
   t(0) <= '1';
   t(1) <= q(0);
   t(2) <= q(0) and q(1);

   gen: for k in 0 to 2 generate
       proc: tff port map(t(k), rst, clk, q(k), notq(k));
   end generate;

end Behavioral;
```

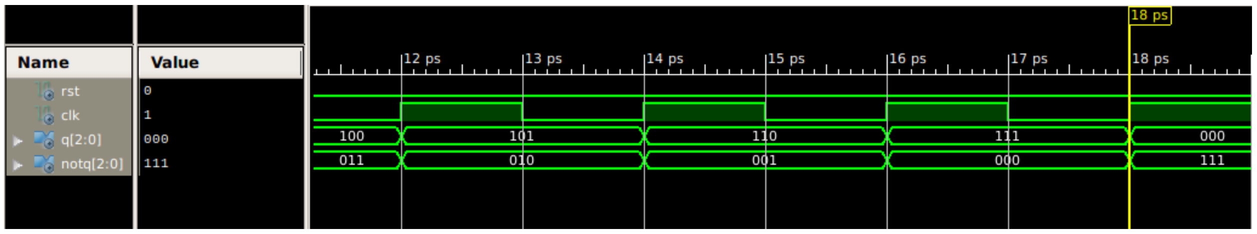
Test Bench

```
ARCHITECTURE behavior OF counter3bitupjkff_test_bench IS

   COMPONENT counter3bitupjkff
   PORT(
        rst : IN  std_logic;
        clk : IN  std_logic;
        q : INOUT  std_logic_vector(2 downto 0);
        notq : INOUT  std_logic_vector(2 downto 0)
       );
   END COMPONENT;


  --Inputs
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';

  --BiDirs
  signal q : std_logic_vector(2 downto 0);
  signal notq : std_logic_vector(2 downto 0);
BEGIN
  -- Instantiate the Unit Under Test (UUT)
  uut: counter3bitupjkff PORT MAP (
         rst => rst,
         clk => clk,
         q => q,
         notq => notq
       );

  -- Stimulus process
  stim_proc: process
  begin
     clk <= '0';
     rst <= '0';
     wait for 1ps;

     rst <= '1';
     clk <= '1';
     wait for 1ps;

     rst <= '0';

     loop1: loop
        clk <= '0';
        wait for 1ps;
        clk <= '1';
        wait for 1ps;
     end loop;
  end process;

END;
```

Timing Diagram

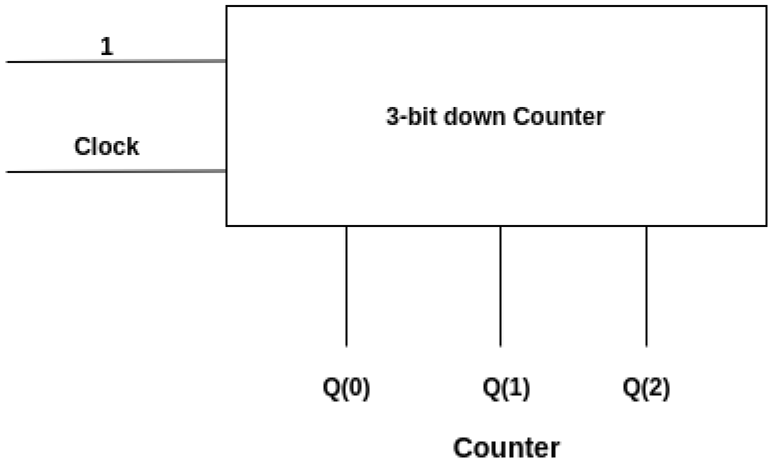




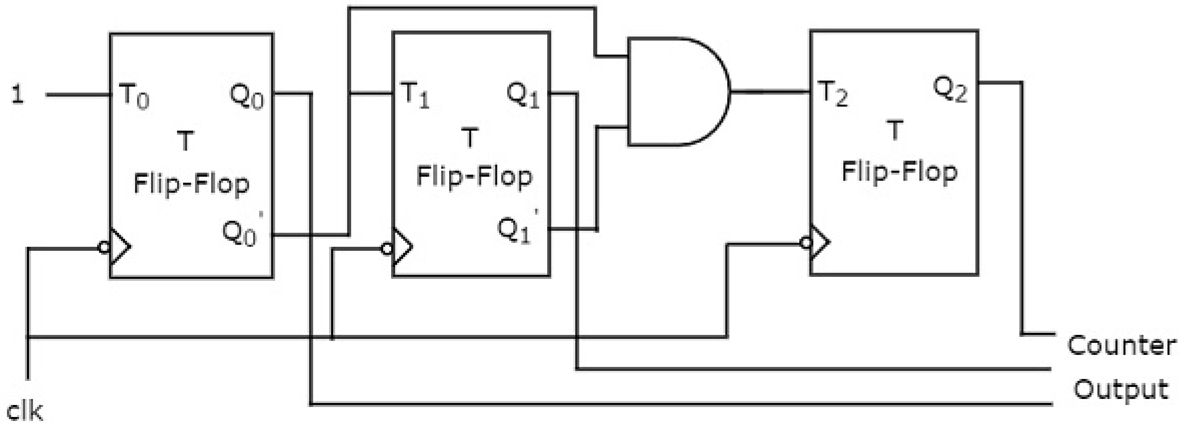


3-bit down counter

Block Diagram



Circuit Diagram



Code

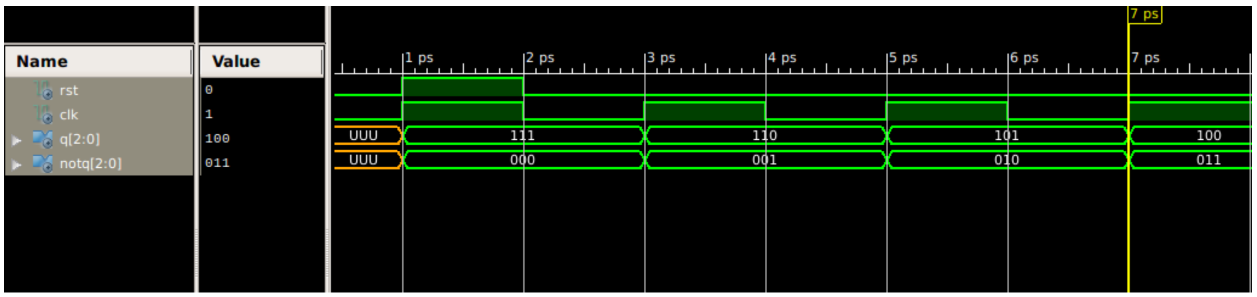
```
architecture Behavioral of counter3bitdown_tff is
   signal t: std_logic_vector(2 downto 0);
   shared variable q1, notq1: std_logic_vector(2 downto 0);
begin
   t(0) <= '1';
   t(1) <= notq(0);
   t(2) <= notq(0) and notq(1);

   gen: for k in 0 to 2 generate
       p1: process(rst, clk)
       begin
```

```
                if rst = '1' then
                    q(k) <= '1';
                    notq(k) <= '0';
                    q1(k) := '1';
                    notq1(k) := '0';
                elsif (clk'event and clk = '1') then
                    if t(k) = '0' then
                        q(k) <= q1(k);
                        notq(k) <= notq1(k);
                    else
                        q(k) <= notq1(k);
                        notq(k) <= q1(k);
                        q1(k) := notq1(k);
                        notq1(k) := not q1(k);
                    end if;
                end if;
            end process;
        end generate;
end Behavioral;
```

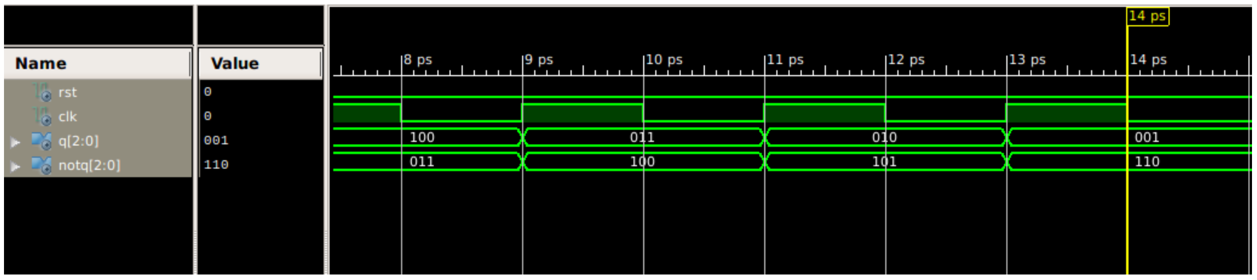
Test Bench

```
ARCHITECTURE behavior OF counter3bitdowntff_test_bench IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT counter3bitdown_tff
    PORT(
         rst : IN  std_logic;
         clk : IN  std_logic;
         q : INOUT  std_logic_vector(2 downto 0);
         notq : INOUT  std_logic_vector(2 downto 0)
        );
    END COMPONENT;
  --Inputs
  signal rst : std_logic := '0';
  signal clk : std_logic := '0';
  --Outputs
  signal q : std_logic_vector(2 downto 0);
  signal notq : std_logic_vector(2 downto 0);
BEGIN
  -- Instantiate the Unit Under Test (UUT)
  uut: counter3bitdown_tff PORT MAP (
         rst => rst,
         clk => clk,
         q => q,
         notq => notq
       );
  -- Stimulus process
  stim_proc: process
  begin
     rst <= '0';
     clk <= '0';
     wait for 1ps;
     rst <= '1';
     clk <= '1';
     wait for 1ps;
     rst <= '0';
     loop1: loop
        clk <= '0';
        wait for 1ps;
        clk <= '1';
        wait for 1ps;
     end loop;
  end process;
END;
```

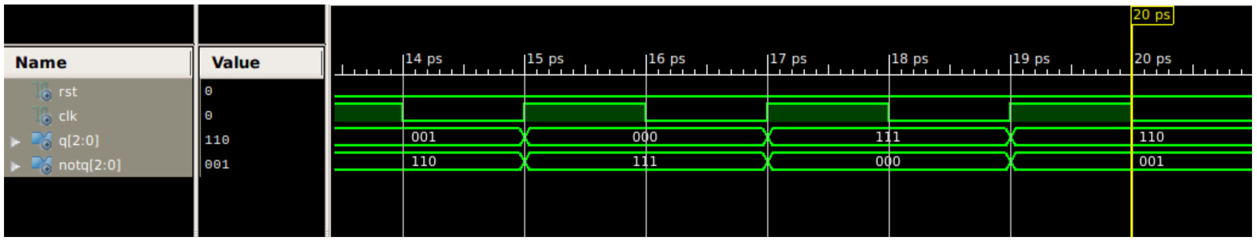
Timing Diagram



Name	Value
rst	0
clk	1
q[2:0]	100
notq[2:0]	011



Name	Value
rst	0
clk	0
q[2:0]	001
notq[2:0]	110



Name	Value
rst	0
clk	0
q[2:0]	110
notq[2:0]	001