Program Code:

SR Flip Flop:

```
library ieee;
use ieee.std_logic_1164.all;

entity srff is
        port(S, R, CLK: in std_logic;
             Q, Qbar: out std_logic);
end srff;

architecture arch_srff of srff is
        begin

        process(CLK)
          begin
            if (S /= R) and rising_edge(CLK) then
              Q <= S;
              Qbar <= R;

            elsif (S = '1') and rising_edge(CLK) then
              Q <= 'Z';
              Qbar <= 'Z';
            end if;

        end process;
end arch_srff;
```

D Flip Flop:

```
library ieee;
use ieee.std_logic_1164.all;

entity dff is
        port(D, CLK: in std_logic;
             Q, Qbar: out std_logic);
end dff;

architecture arch_dff of dff is
        begin

        process(CLK)
          begin
            if rising_edge(CLK) then
              Q <= D;
              Qbar <= not D;

            end if;
        end process;
end arch_dff;
```

JK Flip Flop:

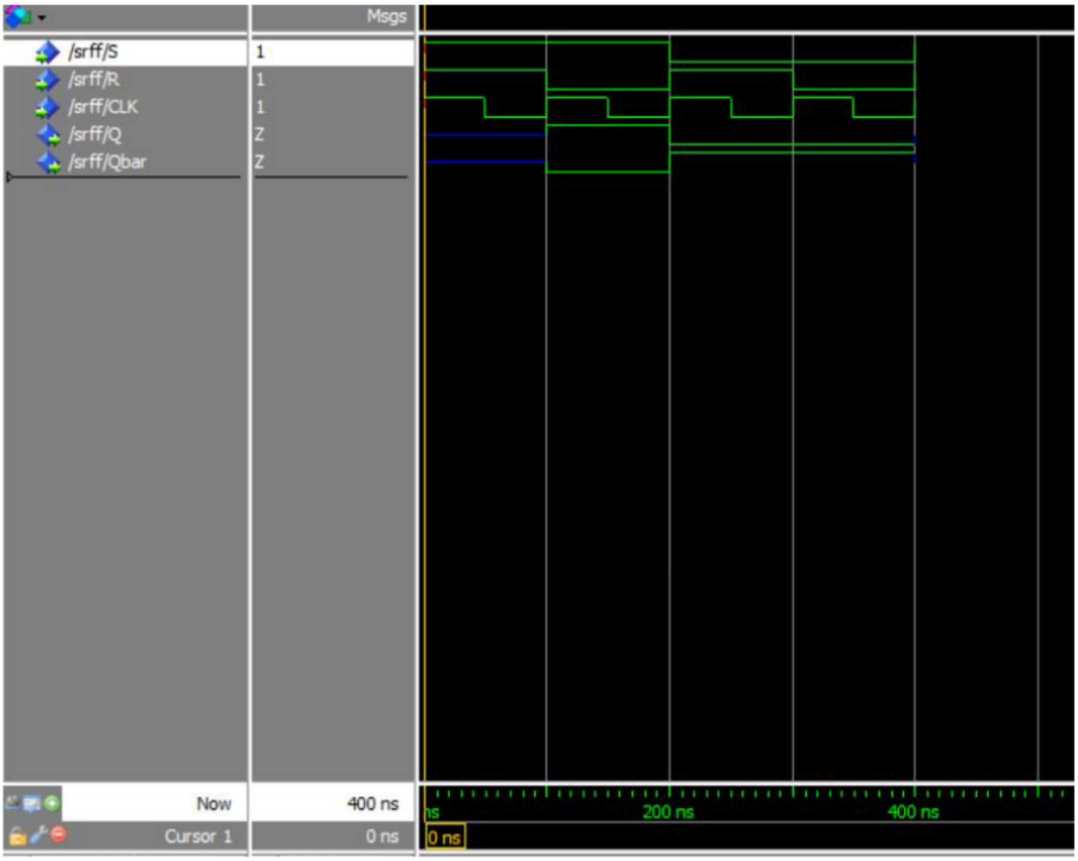
```
library ieee;
use ieee.std_logic_1164.all;

entity jkff is
    port(J, K, CLK: in std_logic;
          Q, Qbar: inout std_logic);
end jkff;

architecture arch_jkff of jkff is
    begin

    process(CLK)
     begin
      if (J /= K) and rising_edge(CLK) then
       Q <= J;
       Qbar <= K;

      elsif (J = '1') and rising_edge(CLK) then
       Q <= not Q;
       Qbar <= not Qbar;
      end if;

    end process;
end arch_jkff;
```

T Flip Flop:

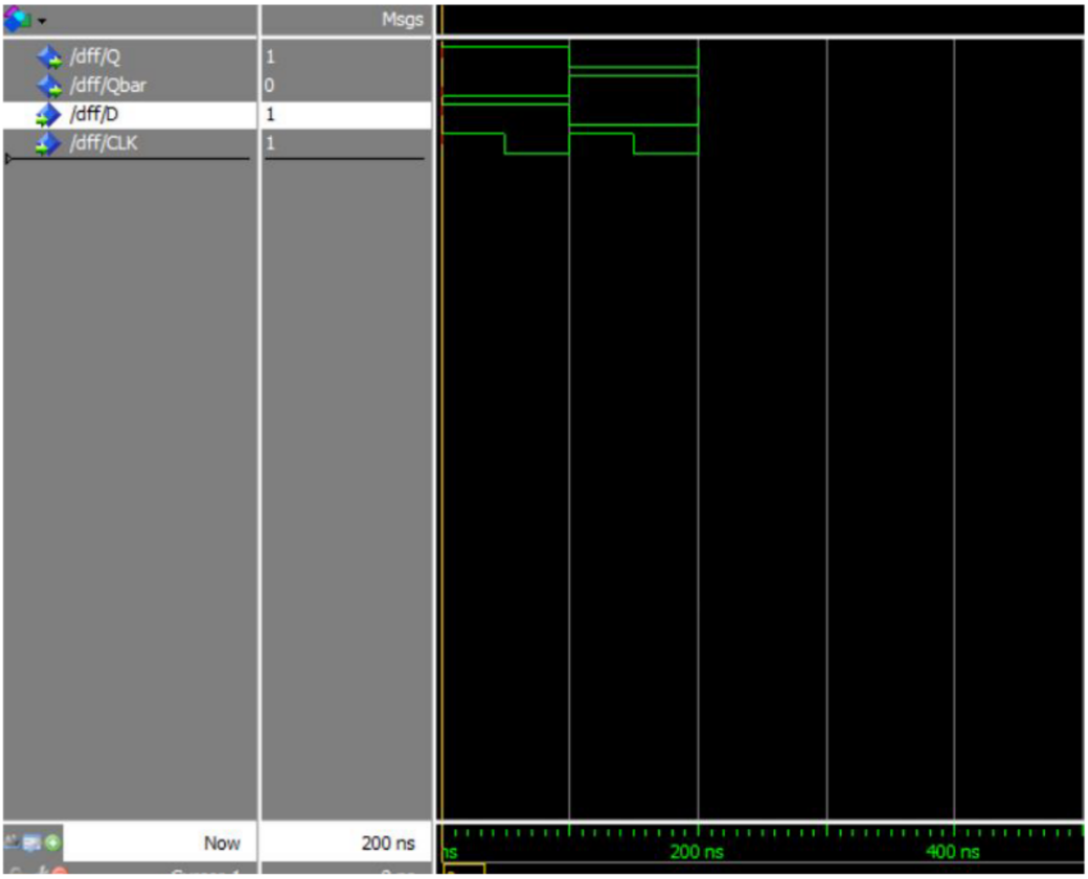
```
library ieee;
use ieee.std_logic_1164.all;

entity tff is
    port(T, CLK: in std_logic;
          Q, Qbar: inout bit);
end tff;

architecture arch_tff of tff is
    begin

    process(CLK)
     begin
      if (T = '1') and rising_edge(CLK) then
       Q <= not Q;
       Qbar <= not Qbar;
      end if;

    end process;
end arch_tff;
```

OUTPUT:

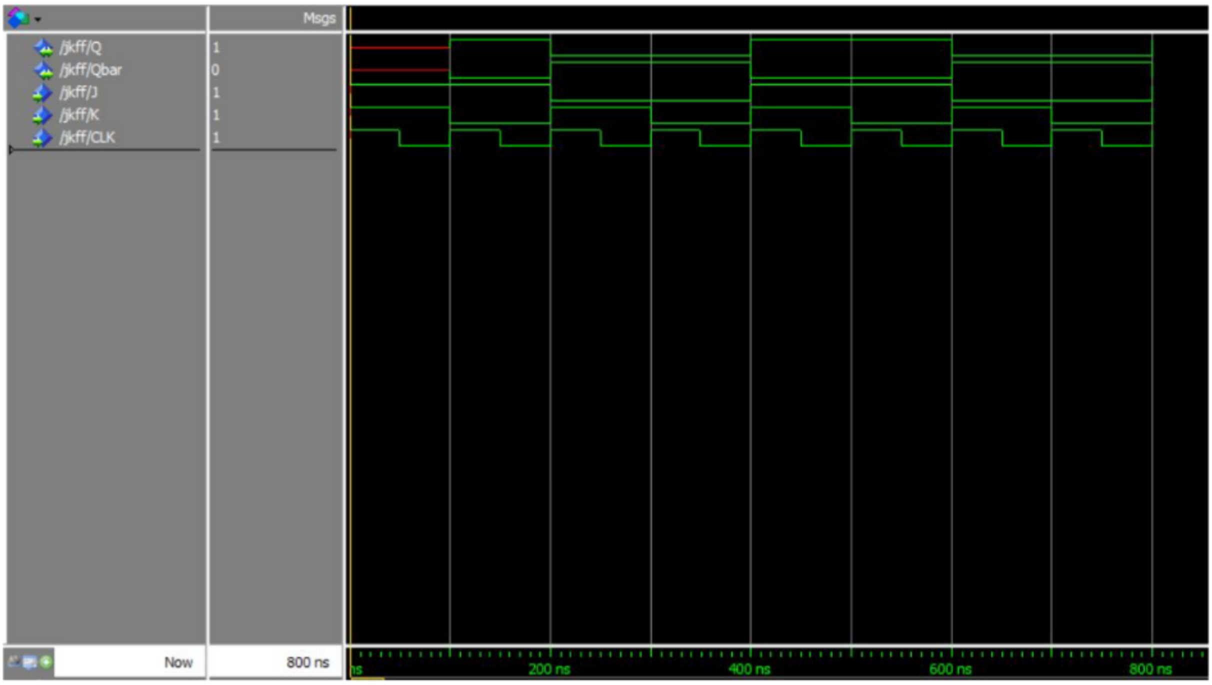
SR Flip Flop



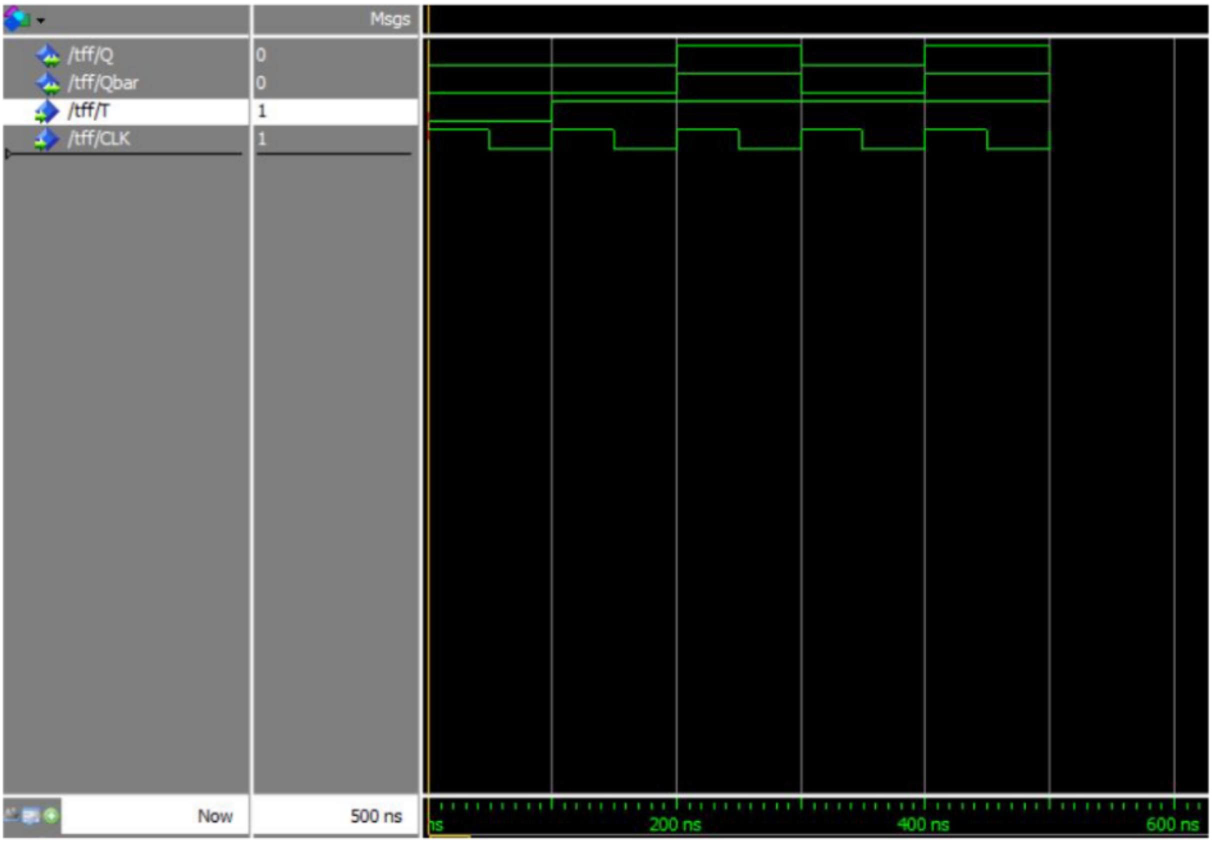
D Flip Flop



JK Flip Flop



T Flip Flop



Result: Hence we have studied and programmed the following flip flops in vhdl.