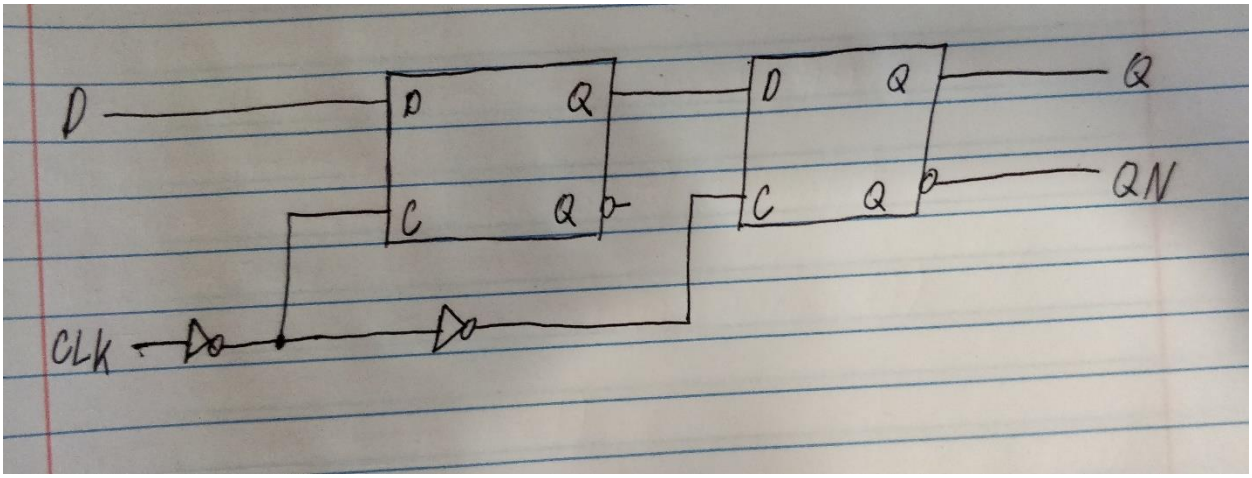
Task 1: Positive Edge Triggered D Flip-Flop

A positive edge triggered D flip-flop is a clock synchronous sequential circuit that holds onto the previous input and reads for new inputs only on positive edges on the clock. It uses two D latches and 2 inverters to achieve this.

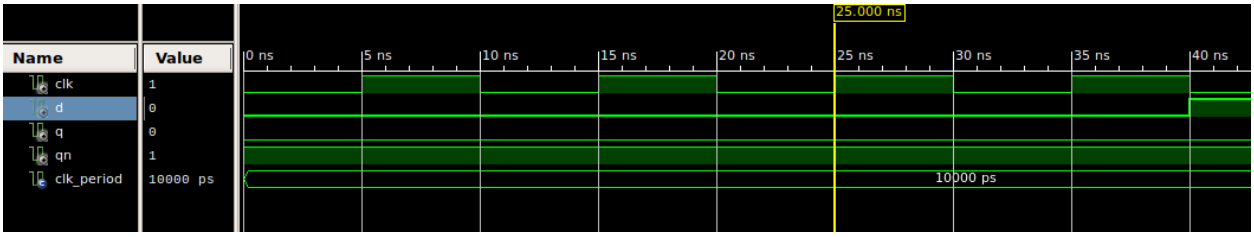
D	CLK	Q	QN
0	↑	0	1
1	↑	1	0
X	0	Last Q	Last QN
X	1	Last Q	Last QN



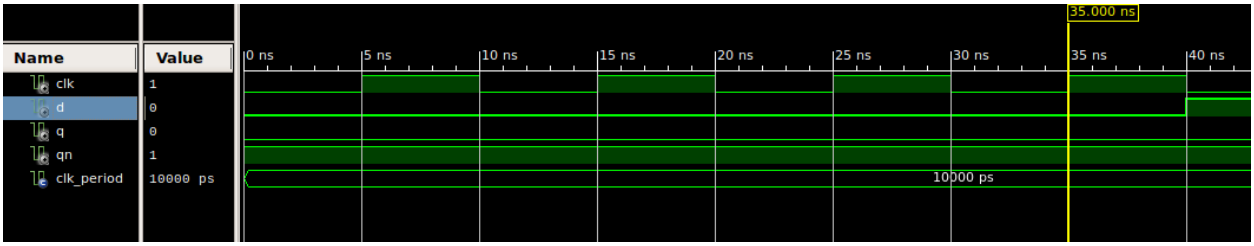
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity pet_d_ff is
    Port ( CLK : in  STD_LOGIC;
           D : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           QN : out  STD_LOGIC);
end pet_d_ff;

architecture Behavioral of pet_d_ff is
   signal CLK_L,Q1,Q2: std_logic;
   component LD
      generic(INIT: bit := '0');
      port(D,G: in std_logic;
           Q: out std_logic);
   end component;
   component INV port(I: in std_logic; O: out std_logic); end component;
begin
   U0: INV port map(CLK,CLK_L);
   U1: LD port map(D,CLK_L,Q1);
   U2: LD port map(Q1,CLK,Q2);
   U3: Q <= Q2;
   U4: INV port map(Q2,QN);
end Behavioral;
```

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TB_pet_d_ff IS
END TB_pet_d_ff;

ARCHITECTURE behavior OF TB_pet_d_ff IS
    COMPONENT pet_d_ff
    PORT(
         CLK : IN  std_logic;
         D : IN  std_logic;
         Q : OUT  std_logic;
         QN : OUT  std_logic
        );
    END COMPONENT;

   --Inputs
   signal CLK : std_logic := '0';
   signal D : std_logic := '0';
   --Outputs
   signal Q : std_logic;
   signal QN : std_logic;

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

```

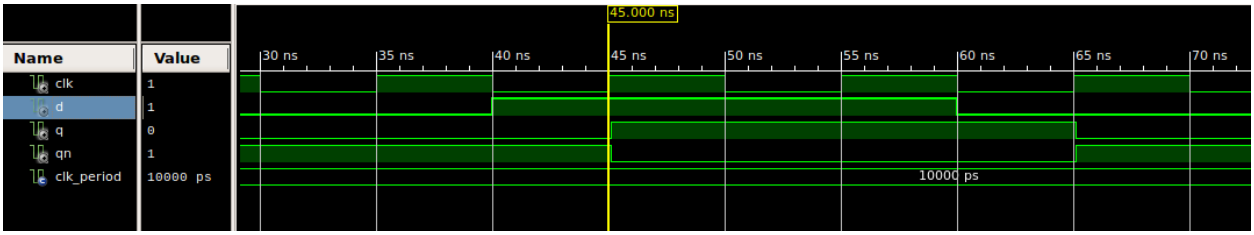
```
26
27  BEGIN
28     -- Instantiate the Unit Under Test (UUT)
29     uut: pet_d_ff PORT MAP (
30             CLK => CLK,
31             D => D,
32             Q => Q,
33             QN => QN
34           );
35
36     -- Clock process definitions
37     CLK_process :process
38     begin
39        CLK <= '0';
40        wait for CLK_period/2;
41        CLK <= '1';
42        wait for CLK_period/2;
43     end process;
44
45     -- Stimulus process
46     stim_proc: process
47     begin

48        wait for 20 ns;
49        d <= '0';
50        wait for CLK_period*2;
51        d <= '1';
52        wait for CLK_period*2;
53        d <= '0';
54        wait for CLK_period*2;
55        d <= '1';
56        wait for CLK_period*2;
57        wait;
58     end process;
59  END;
```



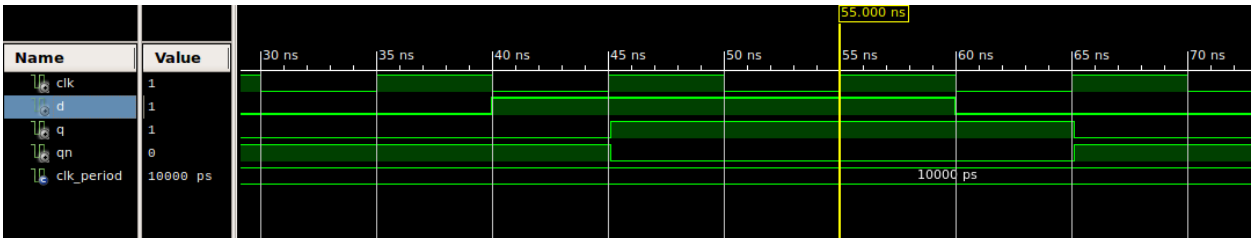
This outcome is correct because D is 0 right before the clock tick therefore Q should be 0 and QN should be 1.



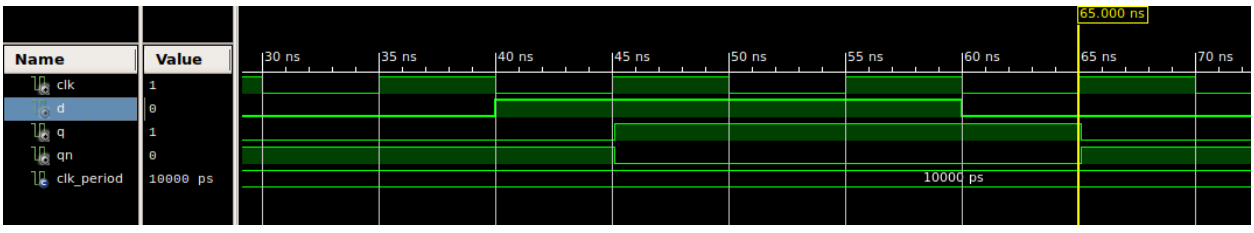
This outcome is correct because D is 0 right before the clock tick therefore Q should be 0 and QN should be 1.



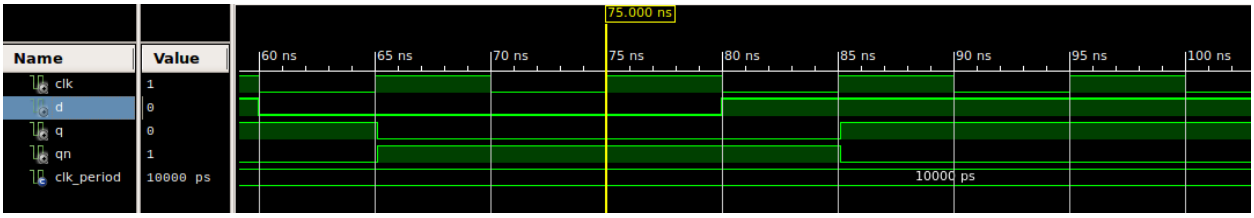
This outcome is correct because D is 0 right before the clock tick therefore Q should be 0 and QN should be 1.



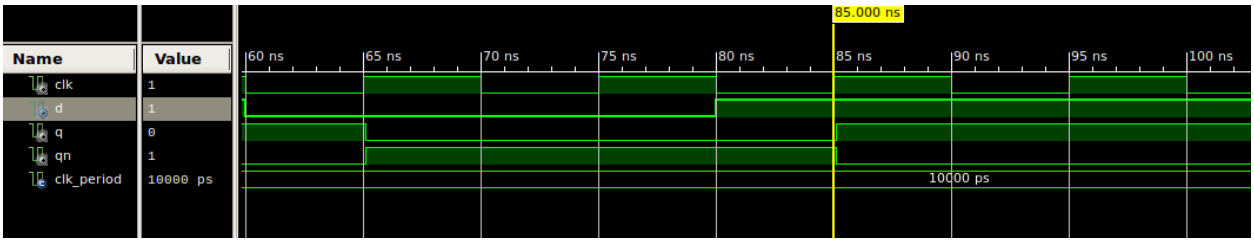
This outcome is correct because D is 1 right before the clock tick therefore Q should be 1 and QN should be 0.



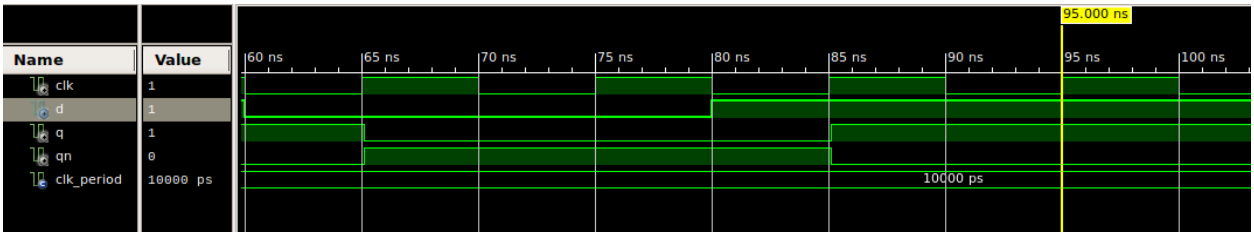
This outcome is correct because D is 1 right before the clock tick therefore Q should be 1 and QN should be 0.



This outcome is correct because D is 0 right before the clock tick therefore Q should be 0 and QN should be 1.



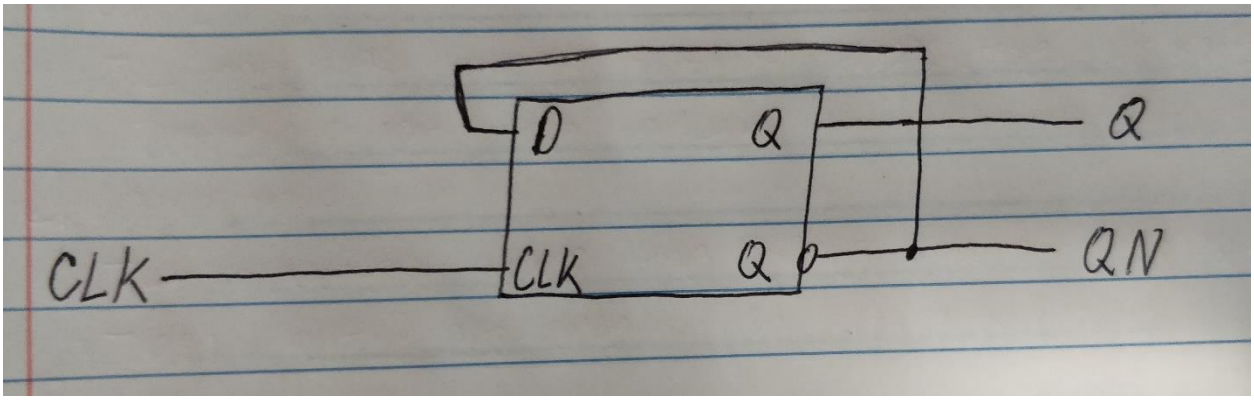
This outcome is correct because D is 0 right before the clock tick therefore Q should be 0 and QN should be 1.



This outcome is correct because D is 1 right before the clock tick therefore Q should be 1 and QN should be 0.

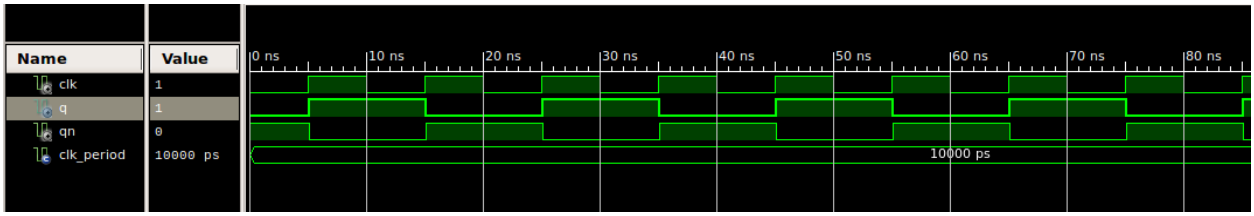
Task 2: T Flip-Flop

A T flip-flop is a clock synchronous sequential circuit that flips the value of Q and QN at each clock tick. It uses a positive edge triggered D flip-flop to achieve this.



```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library UNISIM;
use UNISIM.VComponents.all;

entity t_ff is
    Port ( CLK : in  STD_LOGIC;
           Q : out  STD_LOGIC;
           QN : out  STD_LOGIC);
end t_ff;

architecture Behavioral of t_ff is
   component pet_d_ff port(CLK: in std_logic;
                           D: in std_logic;
                           Q: out std_logic;
                           QN: out std_logic);
   end component;
   signal Q1,Q1_L: std_logic;
begin
   U0: pet_d_ff port map(CLK,Q1_L,Q1,Q1_L);
   U1: Q <= Q1;
   U2: QN <= Q1_L;
end Behavioral;
```

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.ALL;
3  USE ieee.numeric_std.ALL;
4
5  ENTITY TB_t_ff IS
6  END TB_t_ff;
7
8  ARCHITECTURE behavior OF TB_t_ff IS
9      COMPONENT t_ff
10     PORT(
11          CLK : IN  std_logic;
12          Q : OUT  std_logic;
13          QN : OUT  std_logic
14         );
15     END COMPONENT;
16
17    --Inputs
18    signal CLK : std_logic := '0';
19    --Outputs
20    signal Q : std_logic;
21    signal QN : std_logic;
22
23    -- Clock period definitions
24    constant CLK_period : time := 10 ns;
25
26 BEGIN
27    uut: t_ff PORT MAP (
28           CLK => CLK,
29           Q => Q,
30           QN => QN
31         );
32
33    -- Clock process definitions
34    CLK_process :process
35    begin
36       CLK <= '0';
37       wait for CLK_period/2;
38       CLK <= '1';
39       wait for CLK_period/2;
40    end process;
41 END;
42
```



This outcome is correct because it can be seen that at each clock tick, the values of Q and QN flip.

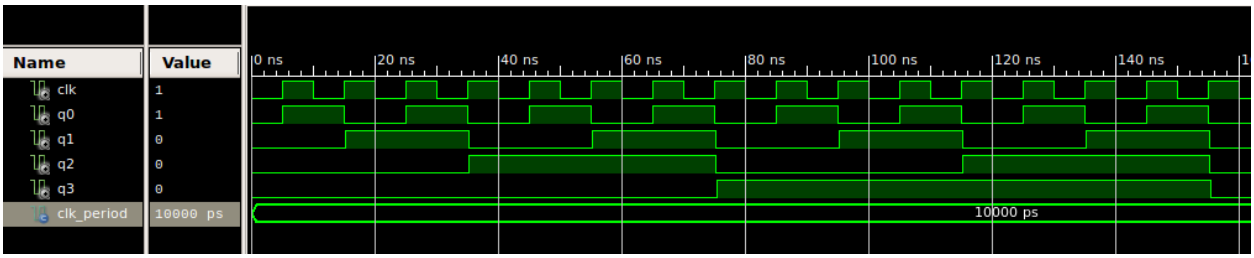
Task 3: 4-bit Binary Ripple Counter

A 4-bit ripple counter is a series of 4 T flip-flops that count the number of clock tick that have passed and then stays asserted for that many clock ticks. The n-th T flip-flop will stay negated for 2^n clock ticks then switch to being asserted for 2^n clock ticks. For example, the first T flip-flop will flip every clock tick, the second T flip-flop will switch every two clock tick, and so on.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  library UNISIM;
4  use UNISIM.VComponents.all;
5
6  entity ripple_counter is
7      Port ( CLK : in  STD_LOGIC;
8             Q0 : out  STD_LOGIC;
9             Q1 : out  STD_LOGIC;
10            Q2 : out  STD_LOGIC;
11            Q3 : out  STD_LOGIC);
12 end ripple_counter;
13
14 architecture Behavioral of ripple_counter is
15    component t_ff port(CLK: in std_logic;
16                        Q: out std_logic;
17                        QN: out std_logic);
18    end component;
19    signal Q0_L,Q1_L,Q2_L,Q3_L: std_logic;
20 begin
21    U0: t_ff port map(CLK,Q0,Q0_L);
22    U1: t_ff port map(Q0_L,Q1,Q1_L);
23    U2: t_ff port map(Q1_L,Q2,Q2_L);
24    U3: t_ff port map(Q2_L,Q3,Q3_L);
25 end Behavioral;
26
```

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TB_ripple_counter IS
END TB_ripple_counter;

ARCHITECTURE behavior OF TB_ripple_counter IS
    COMPONENT ripple_counter
    PORT(
         CLK : IN  std_logic;
         Q0 : OUT  std_logic;
         Q1 : OUT  std_logic;
         Q2 : OUT  std_logic;
         Q3 : OUT  std_logic
        );
    END COMPONENT;

   --Inputs
   signal CLK : std_logic := '0';
   --Outputs
   signal Q0 : std_logic;
   signal Q1 : std_logic;
   signal Q2 : std_logic;
   signal Q3 : std_logic;

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

```

```
28
29  BEGIN
30     uut: ripple_counter PORT MAP (
31             CLK => CLK,
32             Q0 => Q0,
33             Q1 => Q1,
34             Q2 => Q2,
35             Q3 => Q3
36           );
37
38     -- Clock process definitions
39     CLK_process :process
40     begin
41        CLK <= '0';
42        wait for CLK_period/2;
43        CLK <= '1';
44        wait for CLK_period/2;
45     end process;
46  END;
47
```



This outcome is correct because Q0 flips every clock tick, Q1 flips every two clock ticks, Q2 flips every four clock ticks, and Q4 flips every eight clock ticks.