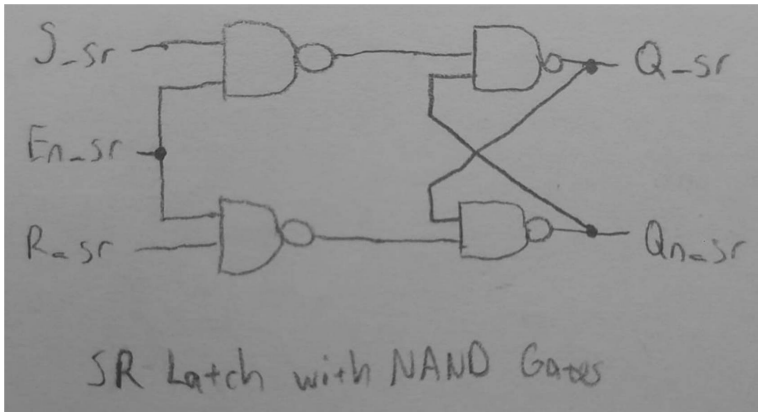
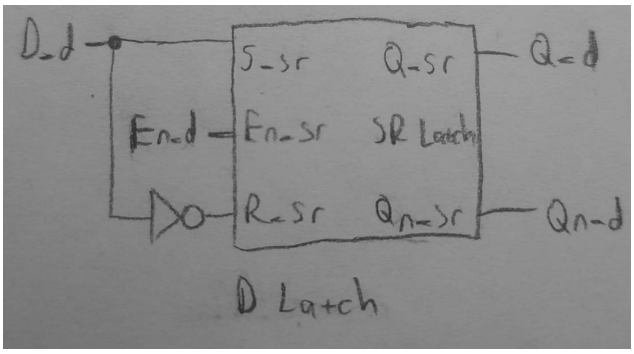
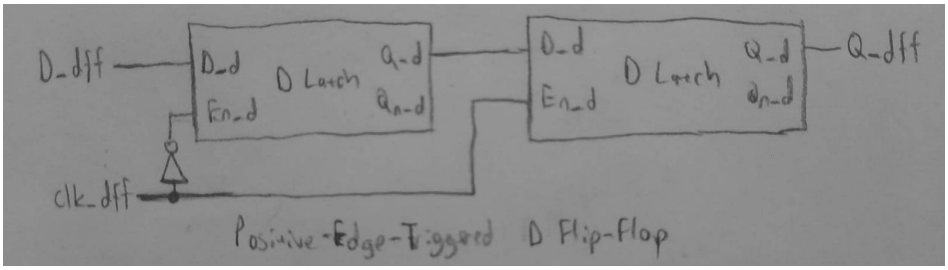
EXPERIMENT 4 PRELIMINARY WORK

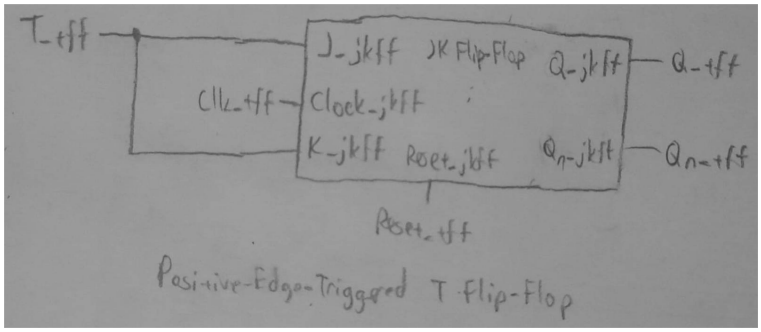
Q1: 1. Analytical Solution

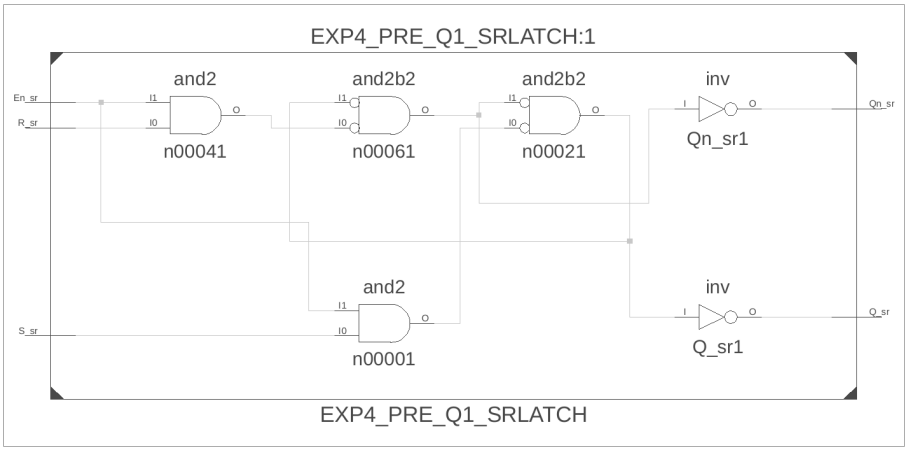
Hand-drawn circuits and RTL schematics

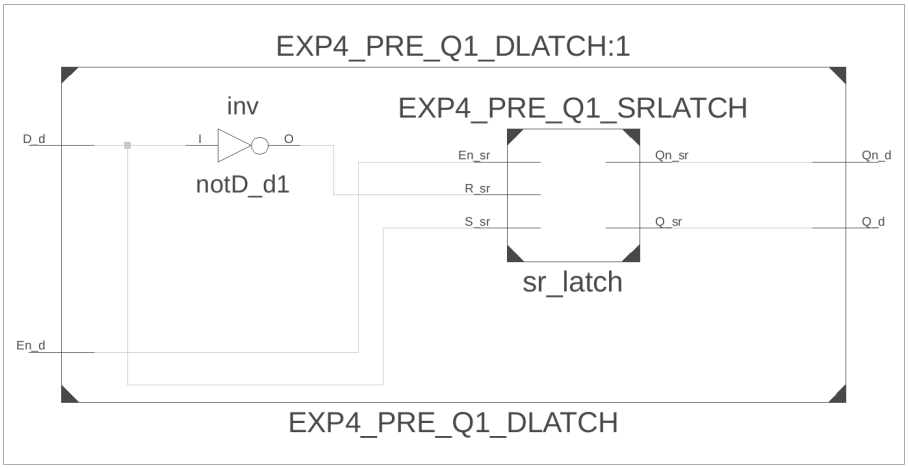


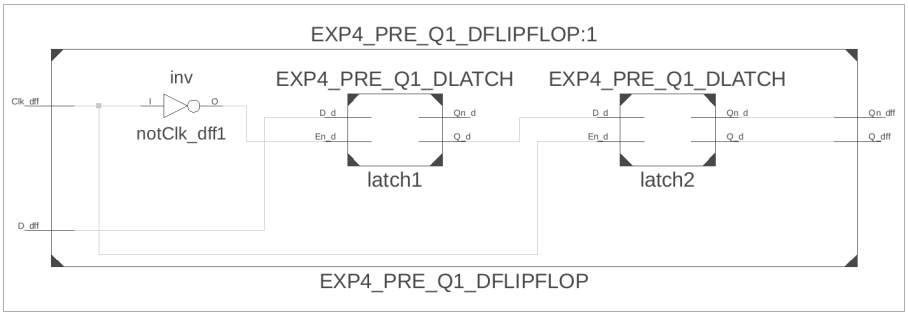


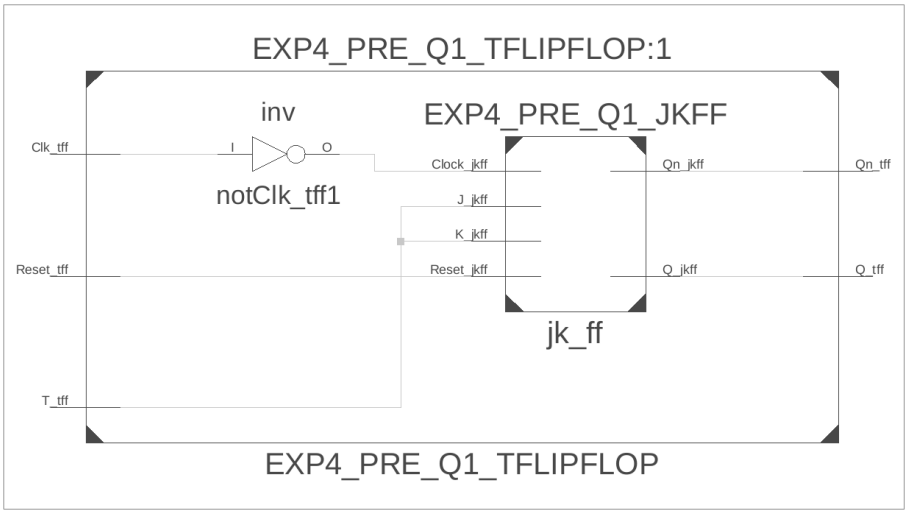





EXP4_PRE_Q1_SRLATCH:1
and2
and2b2
and2b2
inv
En_sr
R_sr
I1
I0
O
n00041
n00061
n00021
Qn_sr1
Qn_sr
and2
inv
S_sr
n00001
Q_sr1
Q_sr
EXP4_PRE_Q1_SRLATCH

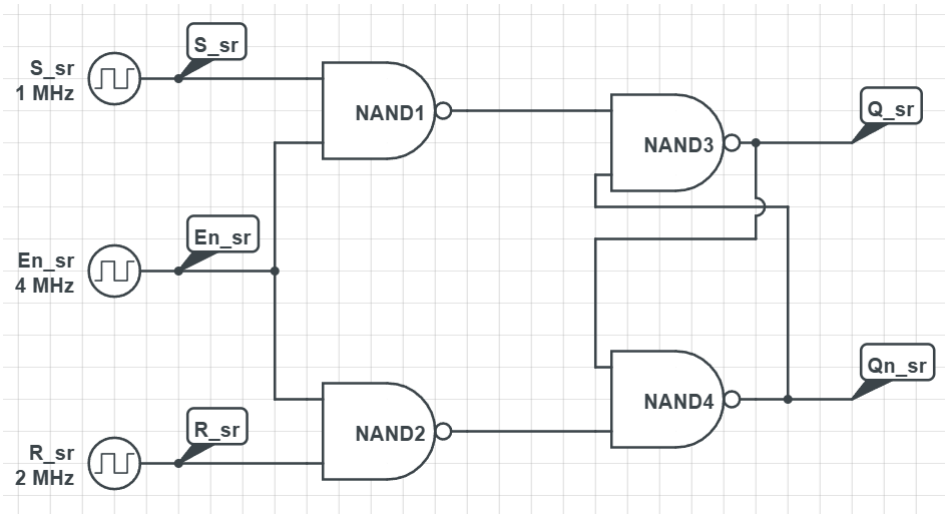

EXP4_PRE_Q1_DLATCH:1
inv
EXP4_PRE_Q1_SRLATCH
D_d
I
O
notD_d1
En_sr
R_sr
S_sr
Qn_sr
Q_sr
Qn_d
Q_d
sr_latch
En_d
EXP4_PRE_Q1_DLATCH


EXP4_PRE_Q1_DFLIPFLOP:1
inv
EXP4_PRE_Q1_DLATCH
EXP4_PRE_Q1_DLATCH
Clk_dff
I
O
notClk_dff1
D_d
En_d
Qn_d
Q_d
latch1
latch2
Qn_dff
Q_dff
D_dff
EXP4_PRE_Q1_DFLIPFLOP

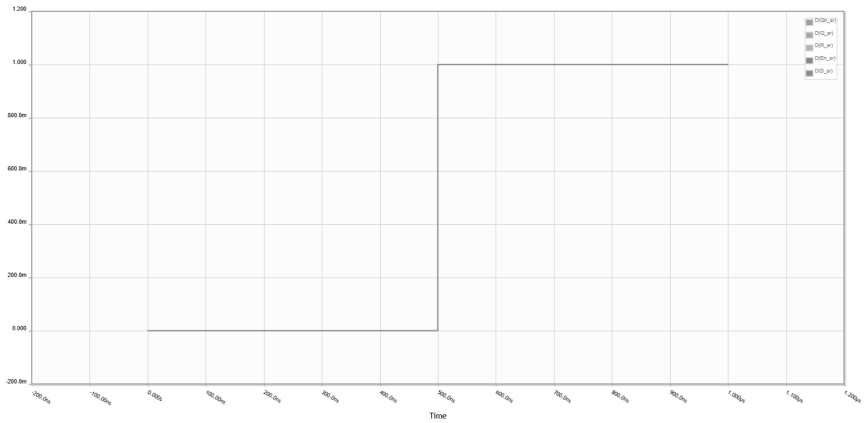

EXP4_PRE_Q1_TFLIPFLOP:1
inv
EXP4_PRE_Q1_JKFF
Clk_tff
I
O
notClk_tff1
Clock_jkff
J_jkff
K_jkff
Reset_jkff
Qn_jkff
Q_jkff
Qn_tff
Q_tff
Reset_tff
jk_ff
T_tff
EXP4_PRE_Q1_TFLIPFLOP

CircuitLab

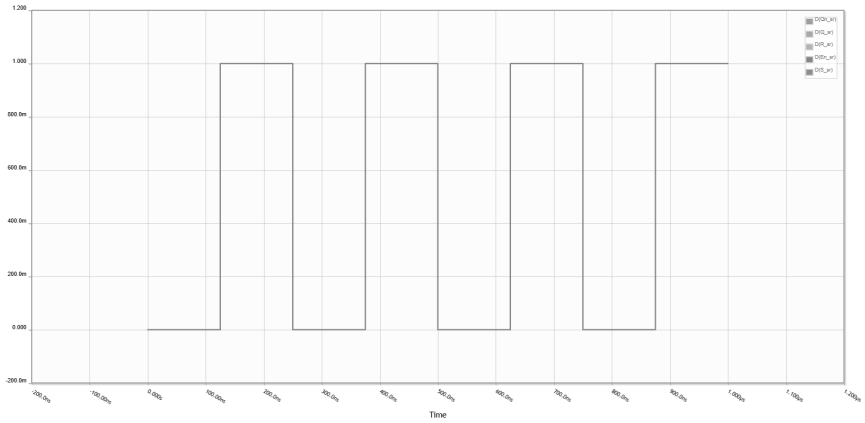
a. Sr Latch with NAND Gates



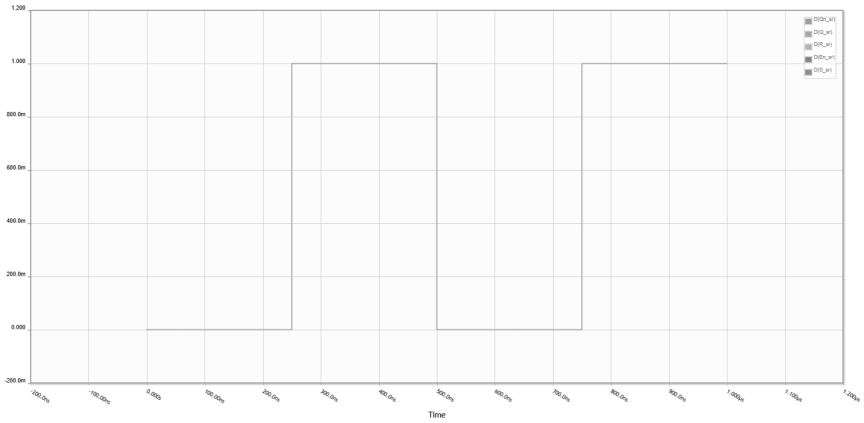
Graph of Input S_sr



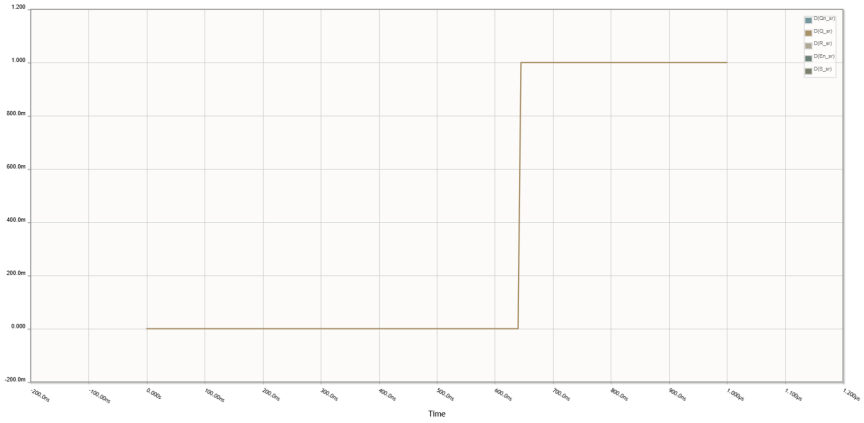
Graph of Input En_sr



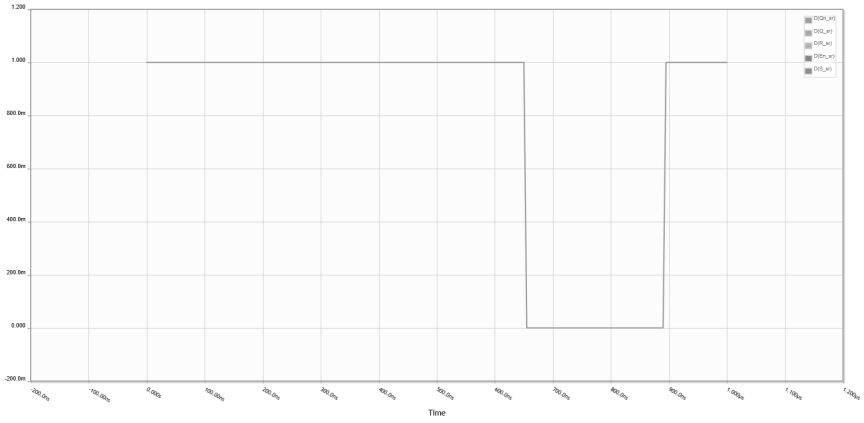
Graph of Input R_sr



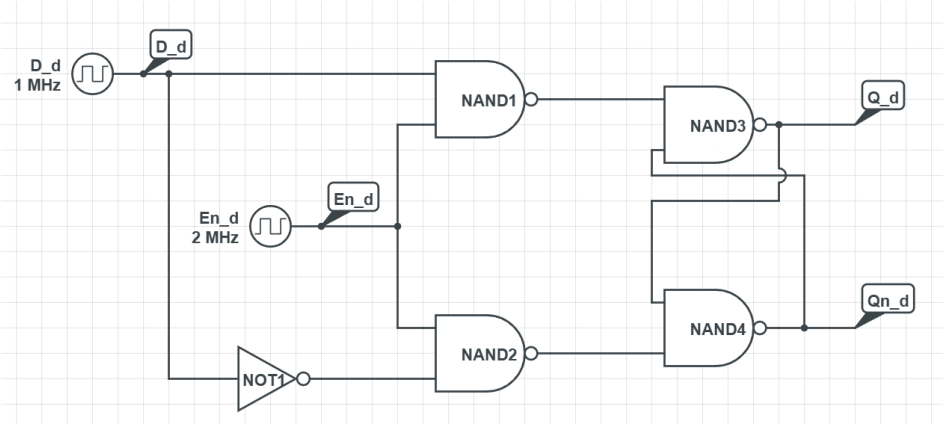
Graph of Output Q_sr



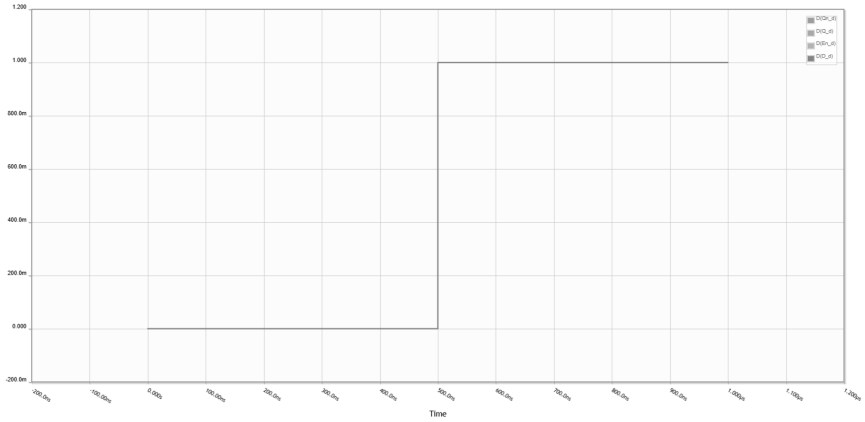
Graph of Output Qn_sr



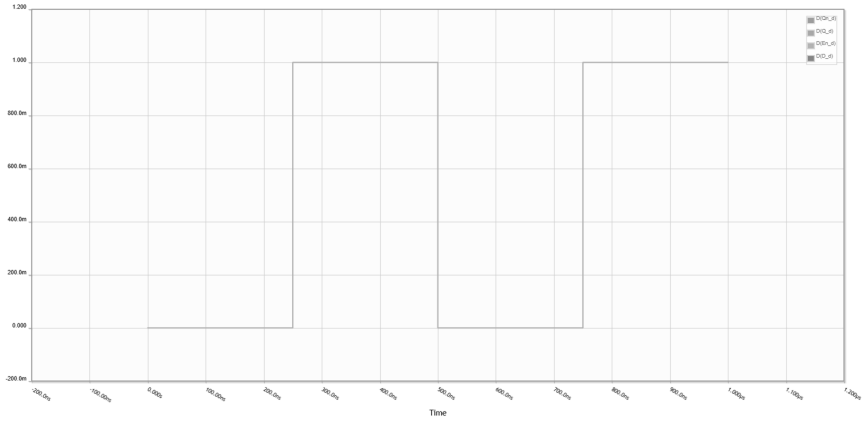
b. D Latch



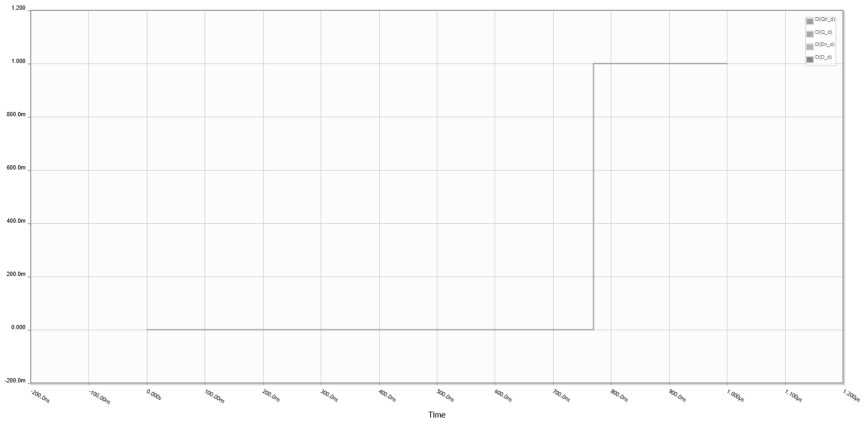
Graph of Input D_d



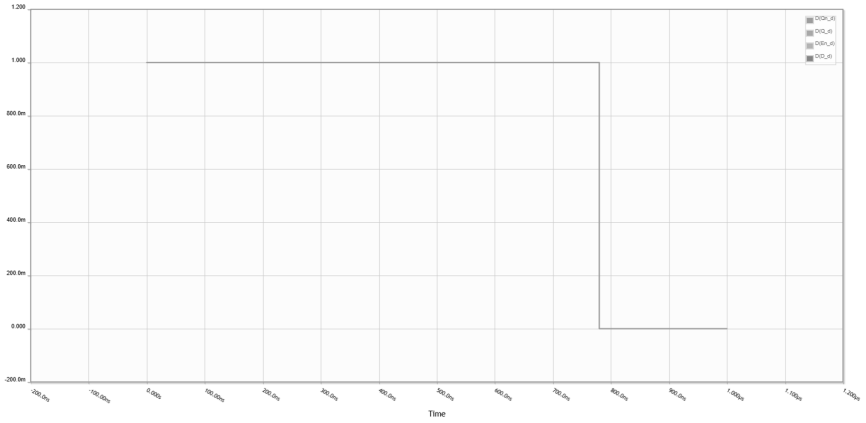
Graph of Input En_d



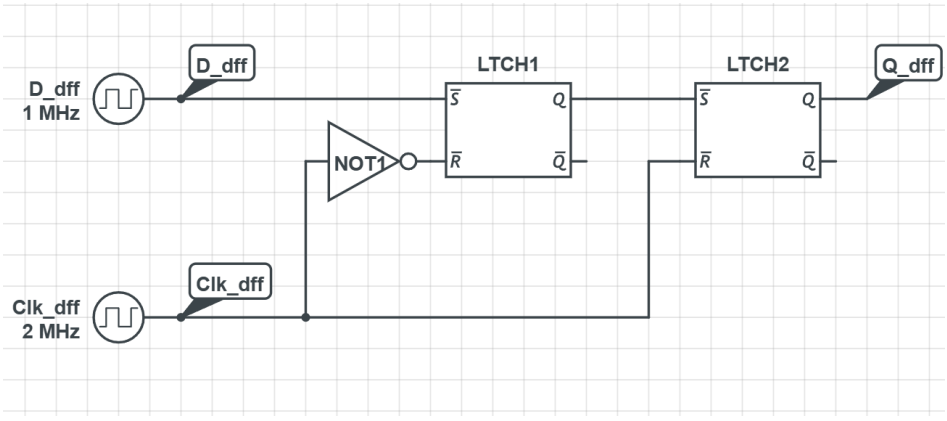
Graph of Output Q_d



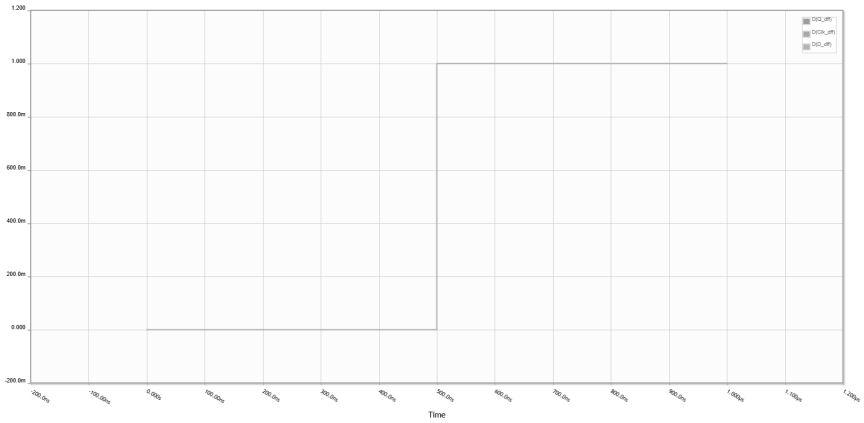
Graph of Output Qn_d



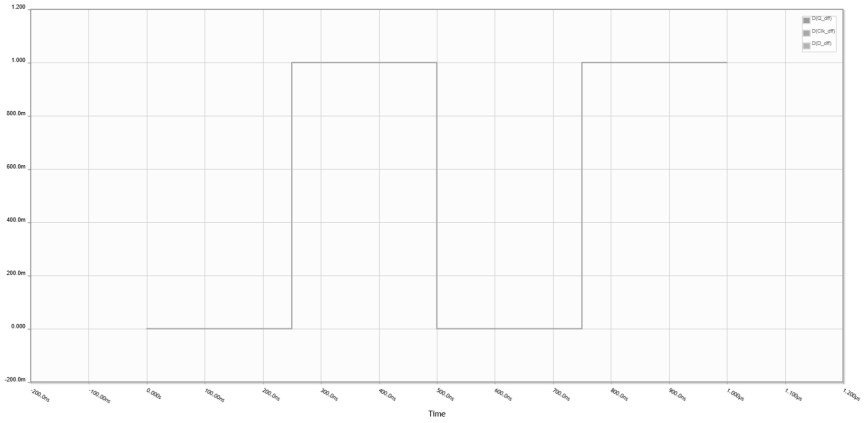
c. Positive-Edge-Triggered D Flip-Flop



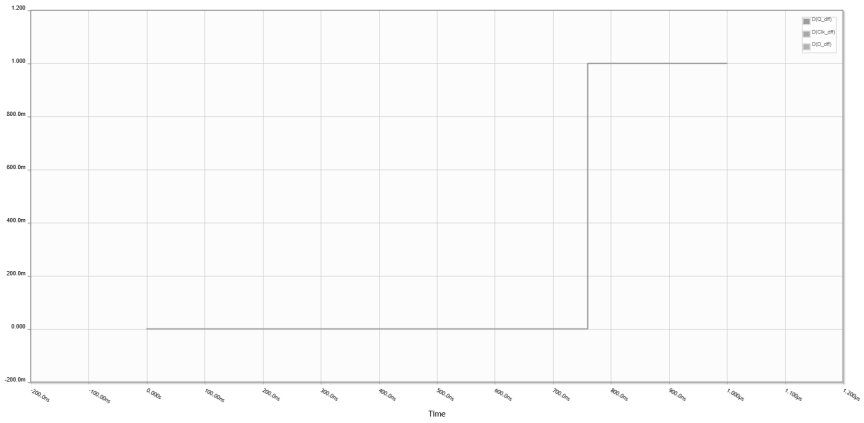
Graph of Input D_dff



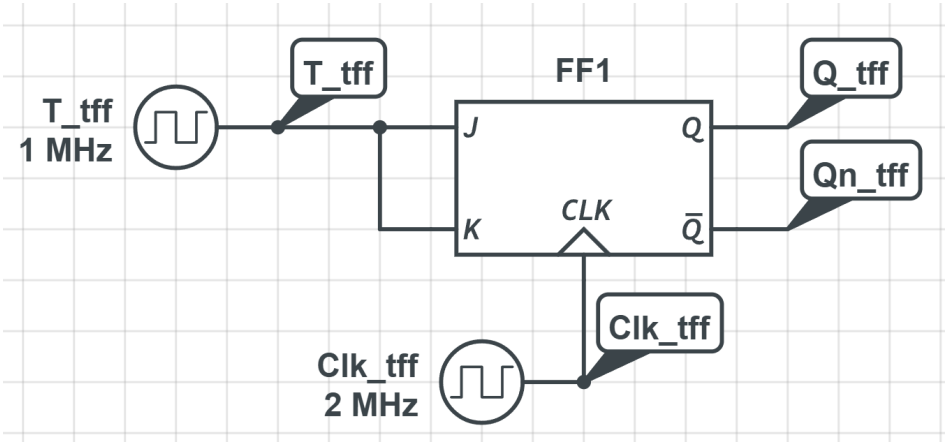
Graph of Input Clk_dff



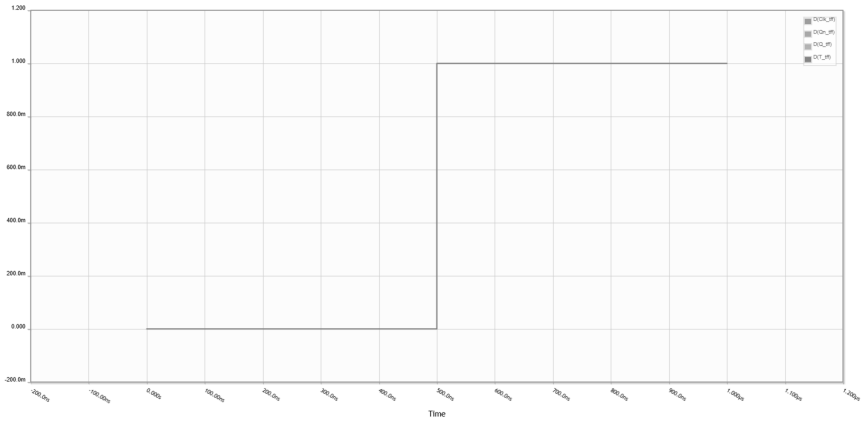
Graph of Output Q_dff



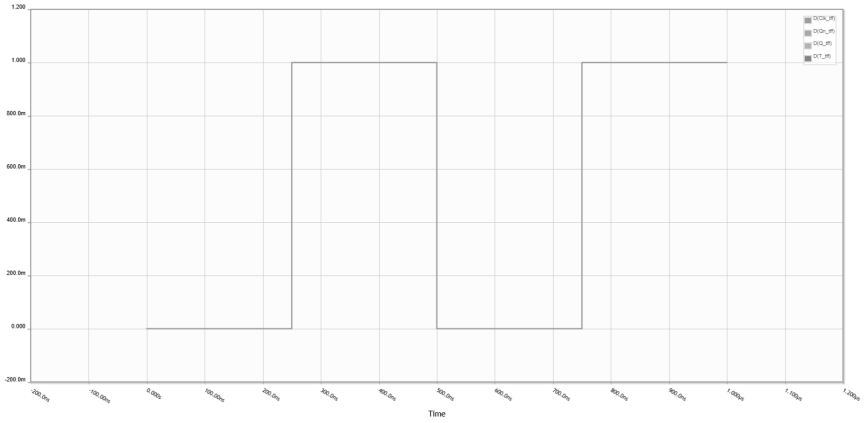
d. Positive-Edge-Triggered T Flip-Flop



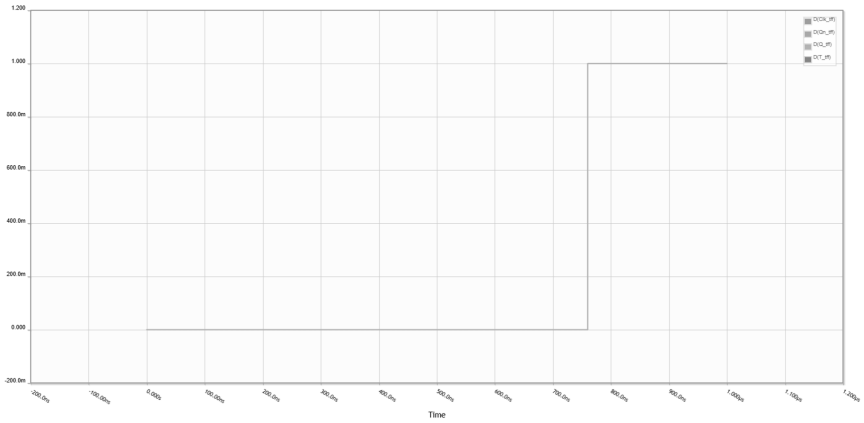
Graph of Input T_tff



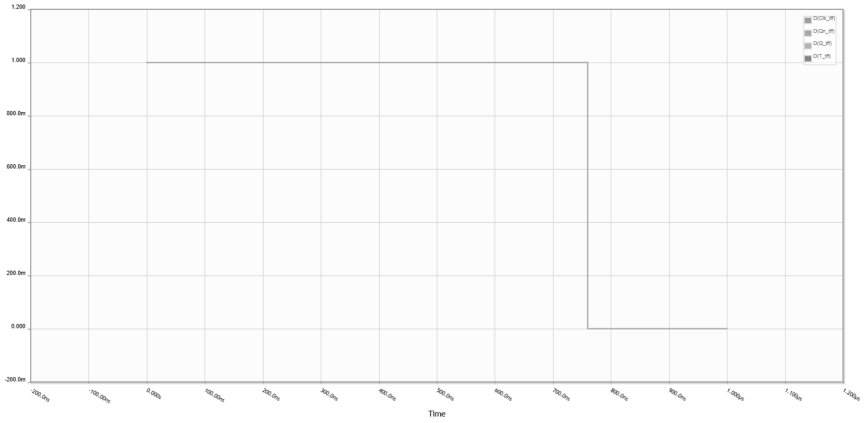
Graph of Input Clk_tff



Graph of Output Q_tff



Graph of Output Qn_tff



2. Codes

*VHDL - **a**. SR Latch with NAND Gates*

```
library ieee;
use ieee.std_logic_1164.all;
entity EXP4_PRE_Q1_SRLATCH is
    port ( S_sr, R_sr, En_sr : in  std_logic; Q_sr, Qn_sr : out std_logic);
end EXP4_PRE_Q1_SRLATCH;
architecture Behavioral of EXP4_PRE_Q1_SRLATCH is
signal temp_Q, temp_Qn : STD_LOGIC;
begin
  temp_Q <= (S_sr nand En_sr) nand temp_Qn;
  temp_Qn <= (R_sr nand En_sr) nand temp_Q;
  Q_sr <= temp_Q; Qn_sr <= temp_Qn;
end Behavioral;
```

*VHDL - **b.** D Latch*

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q1_DLATCH is
    port ( D_d, En_d : in  std_logic; Q_d, Qn_d : out std_logic);
end EXP4_PRE_Q1_DLATCH;
architecture Behavioral of EXP4_PRE_Q1_DLATCH is
component EXP4_PRE_Q1_SRLATCH
    port ( S_sr, R_sr, En_sr : in  std_logic; Q_sr, Qn_sr : out std_logic);
end component;
signal notD_d : std_logic;
begin
    notD_d <= not D_d;
    sr_latch : EXP4_PRE_Q1_SRLATCH port map (D_d, notD_d, En_d, Q_d, Qn_d);
end Behavioral;
```

*VHDL - **c.** Positive-Edge-Triggered D Flip-Flop*

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q1_DFLIPFLOP is
    port ( D_dff, Clk_dff : in  std_logic;
           Q_dff, Qn_dff : out std_logic);
end EXP4_PRE_Q1_DFLIPFLOP;
architecture Behavioral of EXP4_PRE_Q1_DFLIPFLOP is
component EXP4_PRE_Q1_DLATCH
    port ( D_d, En_d : in  std_logic; Q_d, Qn_d : out std_logic);
end component;
signal notClk_dff, Q1_dff, Q1n_dff : std_logic;
begin
  notClk_dff <= not Clk_dff;
  latch1 : EXP4_PRE_Q1_DLATCH port map (D_dff, notClk_dff, Q1_dff, Q1n_dff);
  latch2 : EXP4_PRE_Q1_DLATCH port map (Q1_dff, Clk_dff, Q_dff, Qn_dff);
end Behavioral;
```

*VHDL - **d.** JK Flip-Flop with Reset State*

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q1_JKFF is
    Port ( J_jkff, K_jkff, Reset_jkff, Clock_jkff : in STD_LOGIC;
           Q_jkff, Qn_jkff : out STD_LOGIC);
end EXP4_PRE_Q1_JKFF;
architecture Behavioral of EXP4_PRE_Q1_JKFF is
signal tempQ: std_logic;
begin
    process (Reset_jkff, Clock_jkff)
```

```
    begin
        if Clock_jkff'event and Clock_jkff = '1' then
            if Reset_jkff = '1' then tempQ <= '0';
                elsif (J_jkff='0' and K_jkff='0') then tempQ <= tempQ;
                elsif (J_jkff='0' and K_jkff='1') then tempQ <= '0';
                elsif (J_jkff='1' and K_jkff='0') then tempQ <= '1';
                elsif (J_jkff='1' and K_jkff='1') then tempQ <= not (tempQ);
            end if;
        end if;
    end process;
    Q_jkff <= tempQ; Qn_jkff <= not tempQ;
end Behavioral;
```

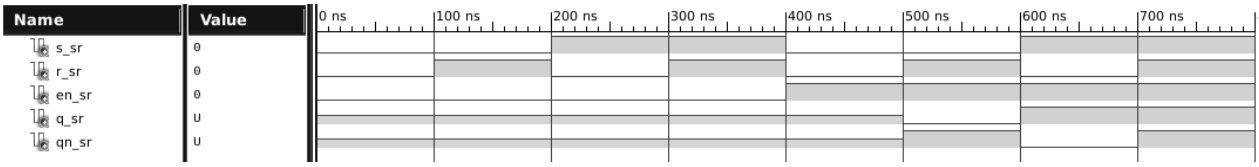
*VHDL - **d.** Positive-Edge-Triggered T Flip-Flop*

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q1_TFLIPFLOP is
    Port ( T_tff, Reset_tff, Clk_tff : in std_logic;
           Q_tff, Qn_tff : out std_logic);
end EXP4_PRE_Q1_TFLIPFLOP;
architecture Behavioral of EXP4_PRE_Q1_TFLIPFLOP is
component EXP4_PRE_Q1_JKFF
    Port ( J_jkff, K_jkff, Reset_jkff, Clock_jkff : in STD_LOGIC;
           Q_jkff, Qn_jkff : out STD_LOGIC);
end component;
signal notClk_tff : std_logic;
begin
    jkff : EXP4_PRE_Q1_JKFF port map(T_tff, T_tff, Reset_tff,
                                     Clk_tff, Q_tff, Qn_tff);
end Behavioral;
```

3. Results

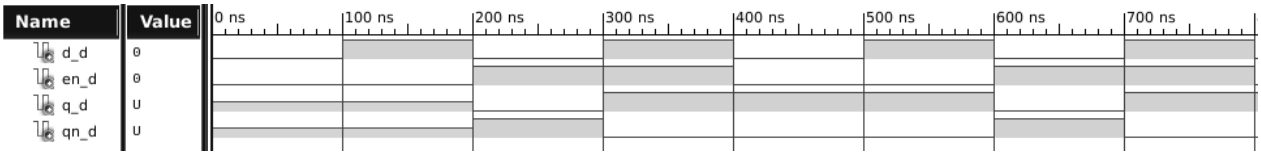
*Test bench - **a.** SR Latch with NAND Gates*

```
stim_proc: process
begin
    En_sr <= '0'; S_sr <= '0'; R_sr <= '0'; wait for 100 ns;
    En_sr <= '0'; S_sr <= '0'; R_sr <= '1'; wait for 100 ns;
    En_sr <= '0'; S_sr <= '1'; R_sr <= '0'; wait for 100 ns;
    En_sr <= '0'; S_sr <= '1'; R_sr <= '1'; wait for 100 ns;
    En_sr <= '1'; S_sr <= '0'; R_sr <= '0'; wait for 100 ns;
    En_sr <= '1'; S_sr <= '0'; R_sr <= '1'; wait for 100 ns;
    En_sr <= '1'; S_sr <= '1'; R_sr <= '0'; wait for 100 ns;
    En_sr <= '1'; S_sr <= '1'; R_sr <= '1'; wait for 100 ns;
end process;
```



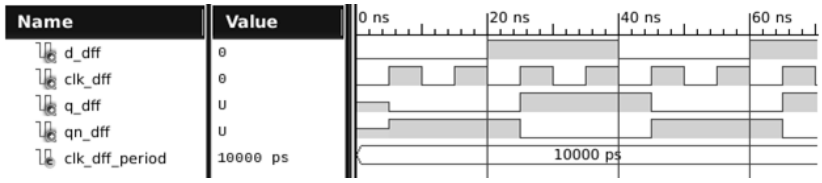
*Test bench - **b.** D Latch*

```
stim_proc: process
begin
    En_d <= '0'; D_d <= '0'; wait for 100 ns;
    En_d <= '0'; D_d <= '1'; wait for 100 ns;
    En_d <= '1'; D_d <= '0'; wait for 100 ns;
    En_d <= '1'; D_d <= '1'; wait for 100 ns;
end process;
```



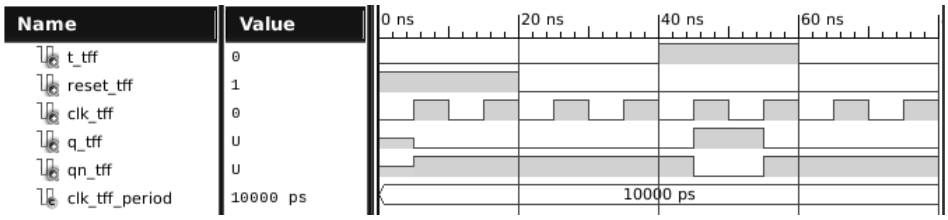
*Test bench - **c.** Positive-Edge-Triggered D Flip-Flop*

```
stim_proc: process
begin
    D_dff <= '0'; wait for Clk_dff_period*2;
    D_dff <= '1'; wait for Clk_dff_period*2;
    D_dff <= '0'; wait for Clk_dff_period*2;
    D_dff <= '1'; wait for Clk_dff_period*2;
end process;
```



*Test bench - **d.** Positive-Edge-Triggered T Flip-Flop*

```
stim_proc: process
begin
    Reset_tff <= '1'; T_tff <= '0'; wait for Clk_tff_period*2;
    Reset_tff <= '0'; T_tff <= '0'; wait for Clk_tff_period*2;
    T_tff <= '1'; wait for Clk_tff_period*2;
    T_tff <= '0'; wait for Clk_tff_period*2;
end process;
```



Q2: 1. Analytical Solution

Present State		Input	Next State		Output
D1(t)	D2(t)	x	D1(t+1)	D2(t+1)	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	1	0	0
1	1	1	0	1	1

State Table for Mealy Sequence Detector

$$y = D1(t)\, D2(t)\, x$$

x \ D1(t) D2(t)	00	01	11	10
0	0	1	1	0
1	0	0	0	1

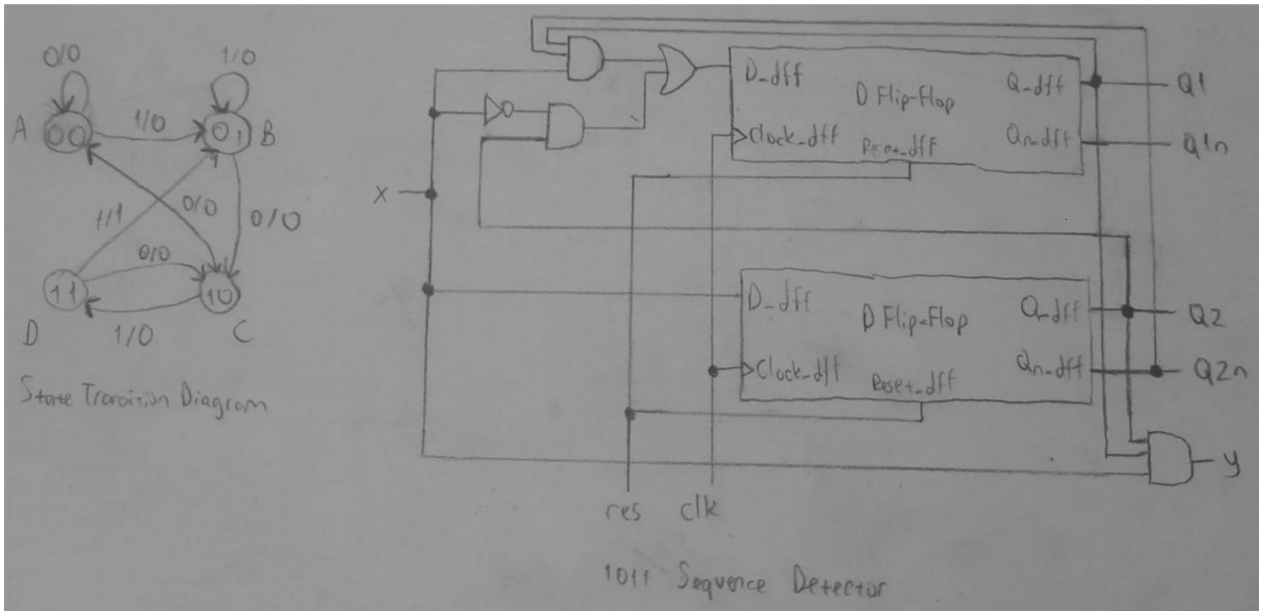
$$D1(t+1) = x'\, D2(t) + x\, D1(t)\, D2'(t)$$

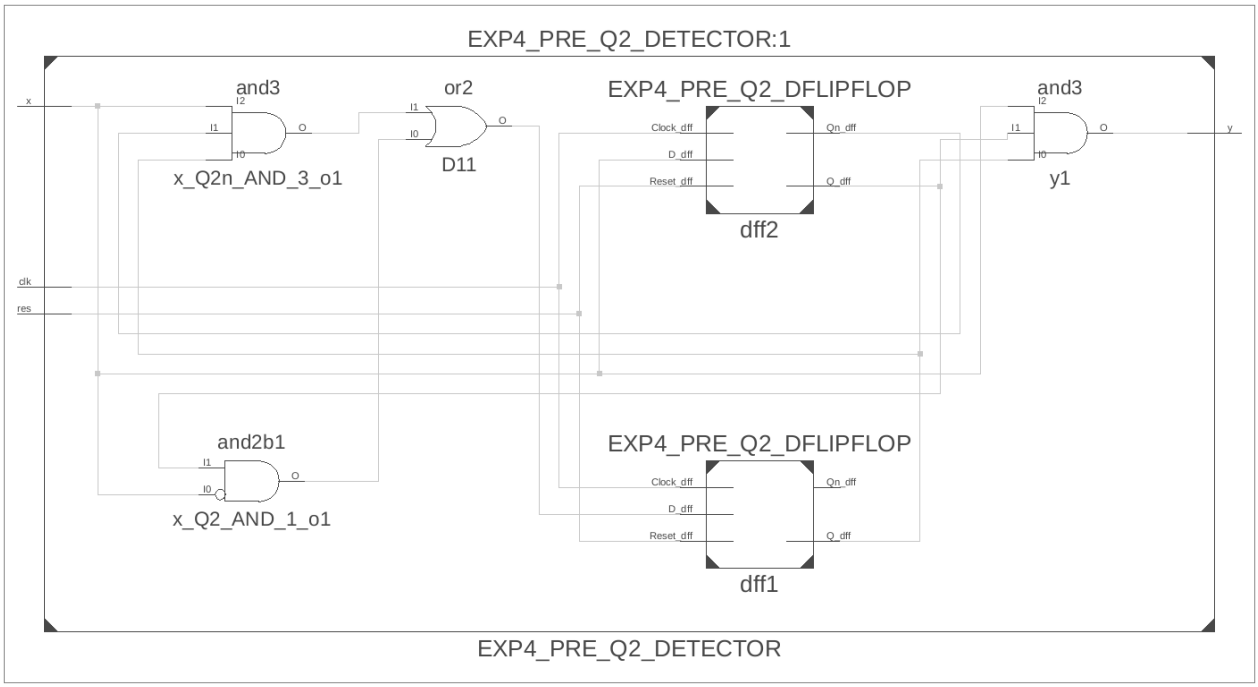
x \ D1(t) D2(t)	00	01	11	10
0	0	0	0	0
1	1	1	1	1

$$D2(t+1) = x$$

The characteristic equation of a D flip-flop is $D = Q(t+1)$. Therefore, we may use 2 D flip-flops for $D1(t+1)$ and $D2(t+1)$ and treat the next states as the inputs of the D flip-flops.

Hand-drawn circuit and RTL schematic





2. Codes

VHDL - D Flip-Flop

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity EXP4_PRE_Q2_DFLIPFLOP is
    Port ( D_dff, Reset_dff, Clock_dff : in STD_LOGIC;
           Q_dff, Qn_dff : out STD_LOGIC);
end EXP4_PRE_Q2_DFLIPFLOP;

architecture Behavioral of EXP4_PRE_Q2_DFLIPFLOP is
signal tempQ : STD_LOGIC;
begin
    process (Reset_dff, Clock_dff)
    begin
        if Clock_dff'event and Clock_dff = '1' then
            if Reset_dff = '1' then tempQ <= '0';
            else tempQ <= D_dff;
            end if;
        end if;
    end process;
    Q_dff <= tempQ;
    Qn_dff <= not tempQ;
end Behavioral;
```

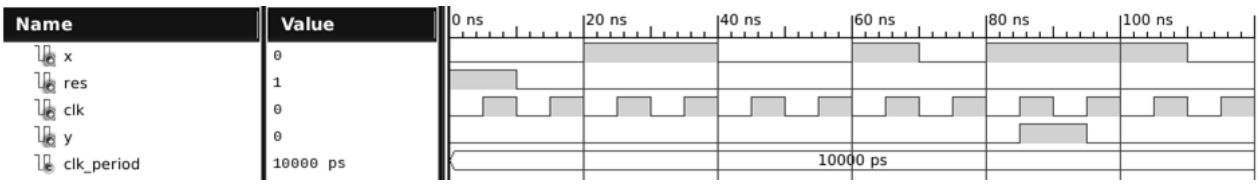
VHDL - Detector

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity EXP4_PRE_Q2_DETECTOR is
    Port ( x, res, clk : in STD_LOGIC;
           y : out STD_LOGIC);
end EXP4_PRE_Q2_DETECTOR;

architecture Behavioral of EXP4_PRE_Q2_DETECTOR is
component EXP4_PRE_Q2_DFLIPFLOP
    Port ( D_dff, Reset_dff, Clock_dff : in STD_LOGIC;
           Q_dff, Qn_dff : out STD_LOGIC);
end component;
signal D1, D2, Q1, Q1n, Q2, Q2n : STD_LOGIC;
begin
    D1 <= (not x and Q2) or (x and Q1 and Q2n) ;
    D2 <= x;
    dff1 : EXP4_PRE_Q2_DFLIPFLOP port map(D1, res, clk, Q1, Q1n);
    dff2 : EXP4_PRE_Q2_DFLIPFLOP port map(D2, res, clk, Q2, Q2n);
    y <= Q1 and Q2 and x;
end Behavioral;
```

3. Results

Test bench

```
stim_proc: process
begin
    res <= '1'; wait for clk_period;
    x <= '0'; res <= '0'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '0'; wait for clk_period;
    x <= '0'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '0'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '1'; wait for clk_period;
    x <= '0'; wait for clk_period
end process;
```



Q3: 1. Analytical Solution

Present State	Input		Next State	Output
T(t)	Data	parityin	T(t + 1)	paritychk
0	0	0	0	1
0	0	1	0	0
0	1	0	1	1
0	1	1	1	0
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

State Table for Parity Generator & Checker

T(t) \ Data parityin	00	01	11	10
0	0	0	1	1
1	0	0	1	1

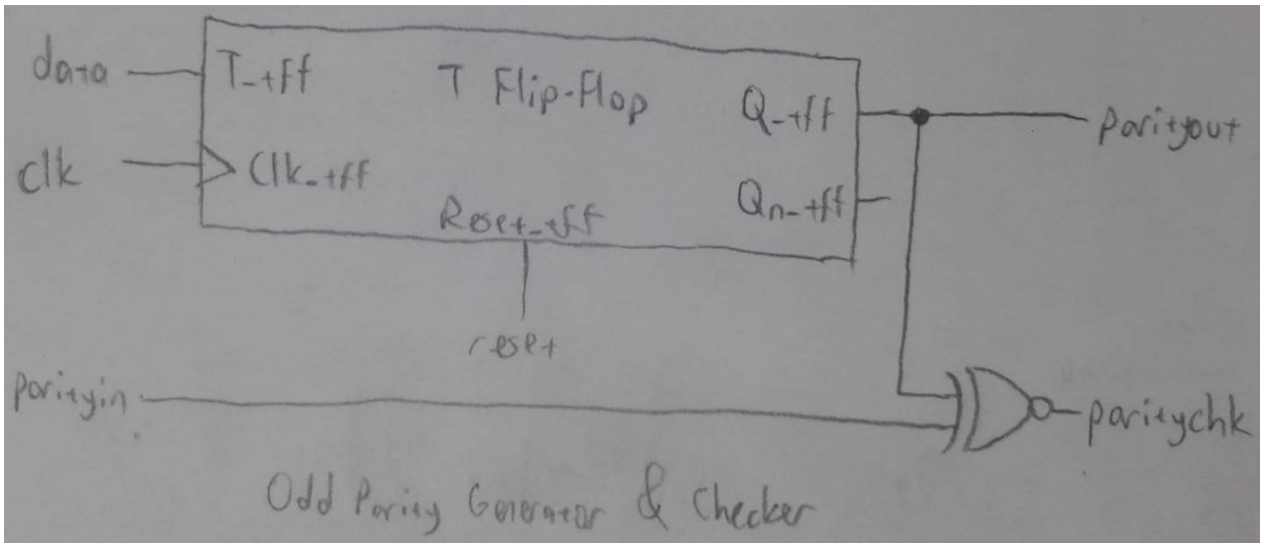
$T(t+1) = \text{Data}$

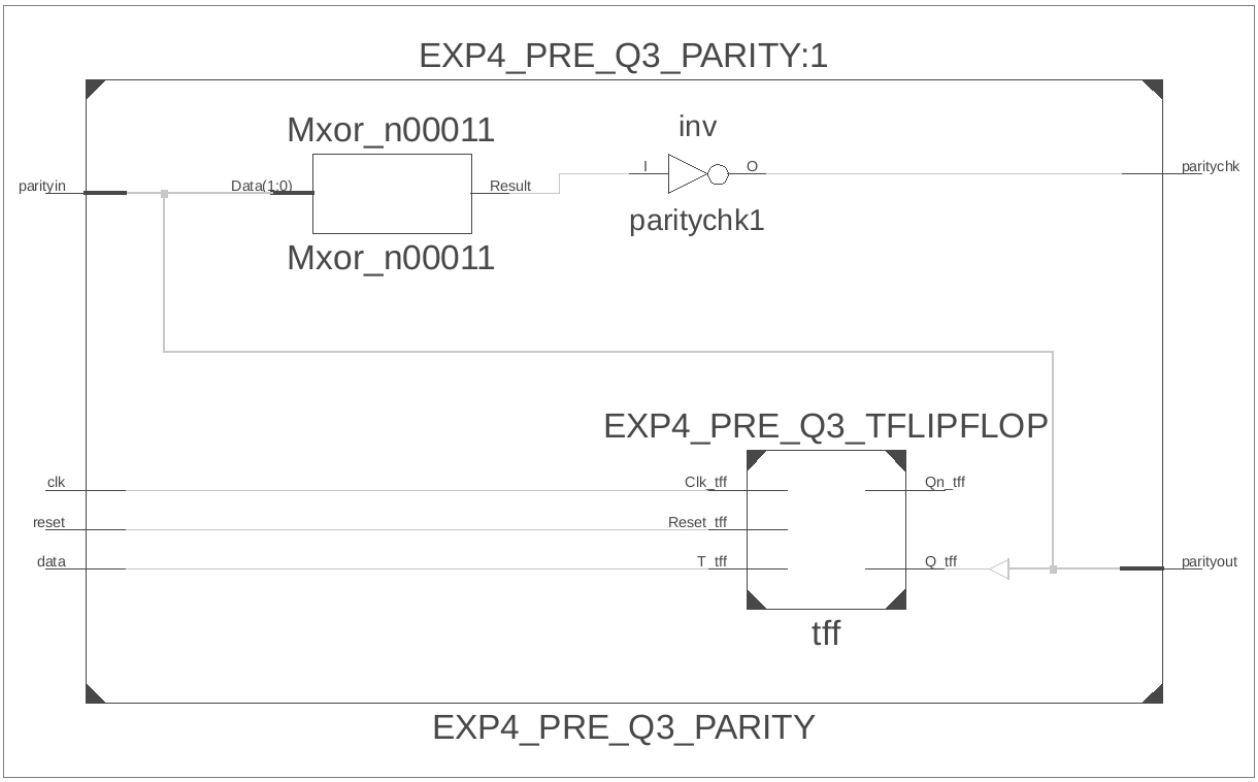
T(t) \ Data parityin	00	01	11	10
0	1	0	0	1
1	0	1	1	0

$\text{paritychk} = T'(t)\,\text{parityin}' + T(t)\,\text{parityin}$

$\text{paritychk} = (T(t) \oplus \text{parityin})'$

Hand-drawn circuit and RTL schematic





2. Codes

VHDL - T Flip-Flop

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q3_TFLIPFLOP is
    port ( T_tff, Reset_tff, Clk_tff : in std_logic;
           Q_tff, Qn_tff : out std_logic);
end EXP4_PRE_Q3_TFLIPFLOP;
architecture Behavioral of EXP4_PRE_Q3_TFLIPFLOP is
signal tempQ : std_logic;
begin
    process (Reset_tff, Clk_tff)
    begin
        if Clk_tff'event and Clk_tff = '1' then
            if Reset_tff = '1' then tempQ <= '0';
            elsif T_tff = '1' then tempQ <= not tempQ;
            end if;
        end if;
    end process;
    Q_tff <= tempQ; Qn_tff <= not tempQ;
end Behavioral;
```

VHDL - Parity Generator & Checker

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity EXP4_PRE_Q3_PARITY is
    Port ( data, parityin, reset, clk : in STD_LOGIC;
         parityout, paritychk : out STD_LOGIC);
end EXP4_PRE_Q3_PARITY;

architecture Behavioral of EXP4_PRE_Q3_PARITY is

component EXP4_PRE_Q3_TFLIPFLOP
    port ( T_tff, Reset_tff, Clk_tff : in std_logic;
           Q_tff, Qn_tff : out std_logic);
end component;

signal tempQ, tempQn: STD_LOGIC;

begin
    tff : EXP4_PRE_Q3_TFLIPFLOP port map (data, reset, clk, tempQ, tempQn);
    parityout <= tempQ;
    paritychk <= tempQ xnor parityin;
end Behavioral;
```

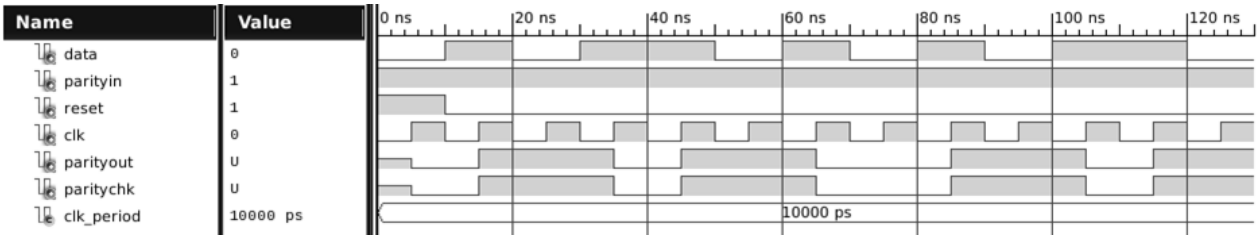
3. Results

Test bench

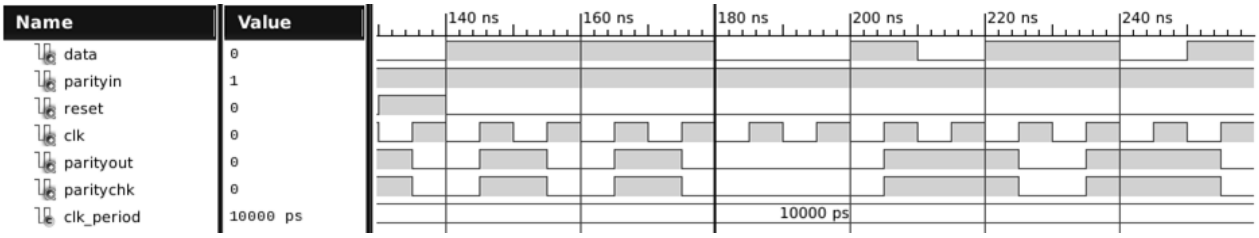
```
stim_proc: process
begin
    -- Data 1
    reset <= '1'; parityin <= '1'; wait for clk_period;
    reset <= '0'; data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
```

```
    -- Data 2
    reset <= '1'; parityin <= '1'; wait for clk_period;
    reset <= '0'; data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;

    -- Data 3
    reset <= '1'; parityin <= '0'; wait for clk_period;
    reset <= '0'; data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '0'; wait for clk_period;

    -- Data 4
    reset <= '1'; parityin <= '0'; wait for clk_period;
    reset <= '0'; data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
    data <= '1'; wait for clk_period;
end process;
```

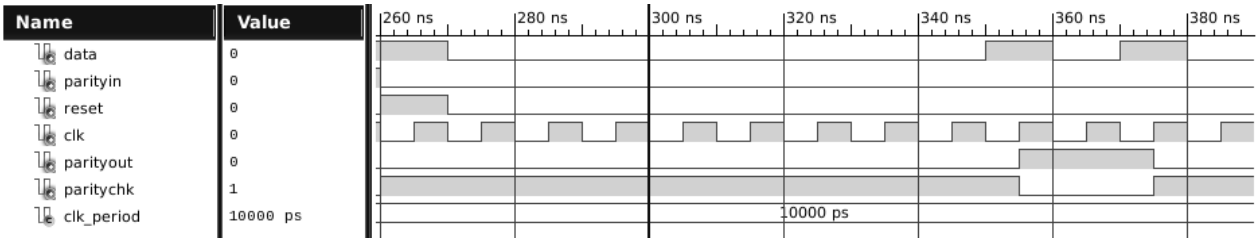
Data 1



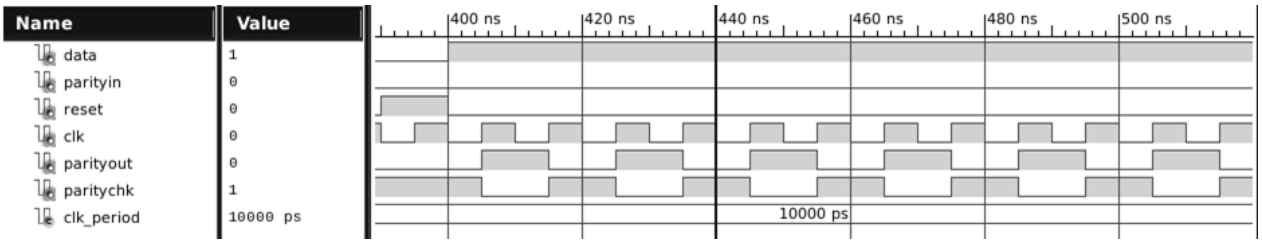
Data 2



Data 3



Data 4



Q4: 1. Analytical Solution

J	K	Q(t)	Q(t+1)	D
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

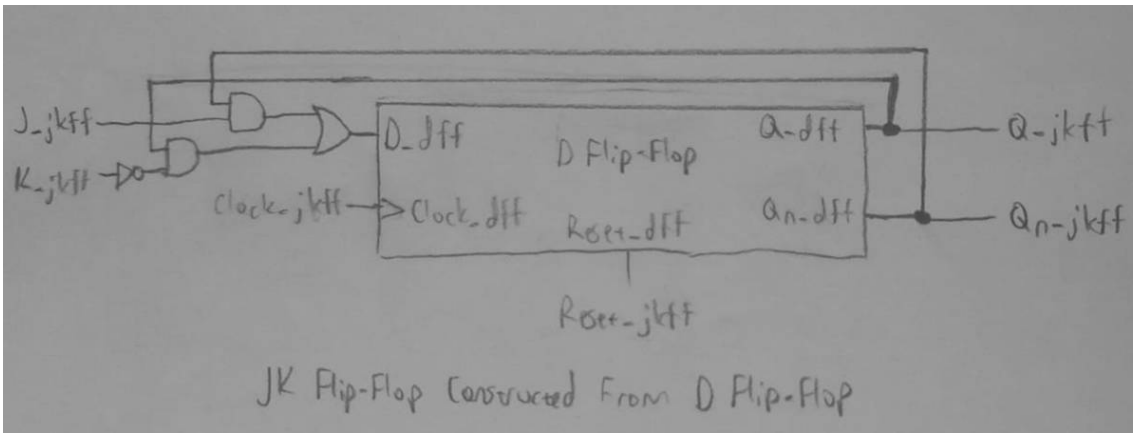
Truth Table for D Flip-Flop

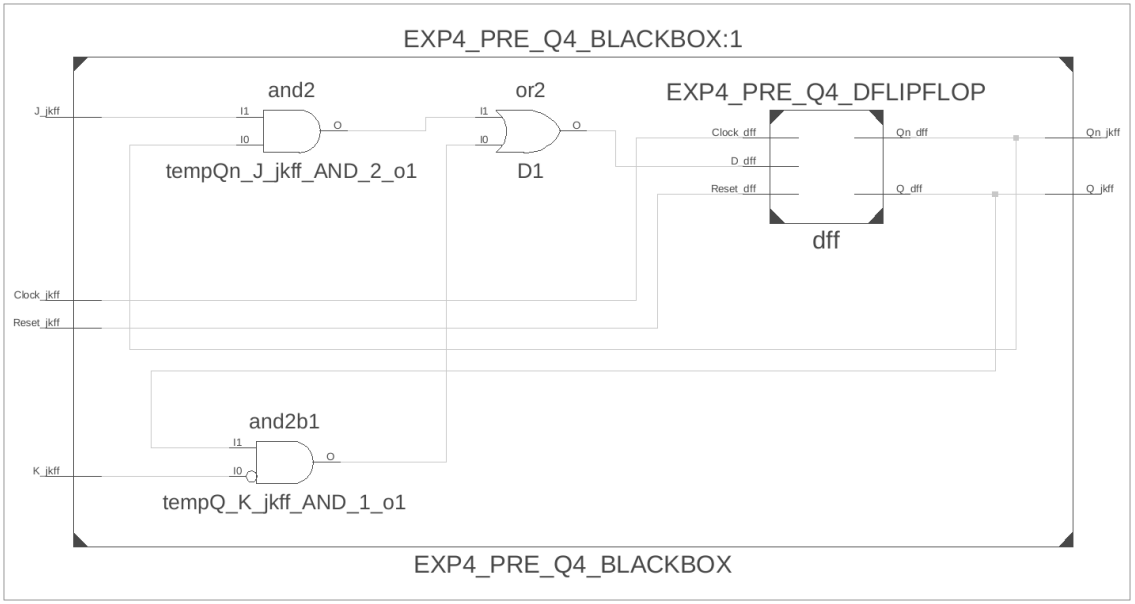
Q \ JK	00	01	11	10
0	0	0	1	1
1	1	0	0	1

$$D = QK' + Q'J$$

Excitation Table for JK Flip-Flop

Hand-drawn circuit and RTL schematic





2. Codes

VHDL - D Flip-Flop

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q4_DFLIPFLOP is
    Port ( D_dff, Reset_dff, Clock_dff : in STD_LOGIC;
           Q_dff, Qn_dff : out STD_LOGIC);
end EXP4_PRE_Q4_DFLIPFLOP;

architecture Behavioral of EXP4_PRE_Q4_DFLIPFLOP is
signal tempQ : STD_LOGIC;
begin
    process (Reset_dff, Clock_dff)
    begin
        if Clock_dff'event and Clock_dff = '1' then
            if Reset_dff = '1' then tempQ <= '0';
            else tempQ <= D_dff;
            end if;
        end if;
    end process;
    Q_dff <= tempQ; Qn_dff <= not tempQ;
end Behavioral;
```

VHDL - Blackbox

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity EXP4_PRE_Q4_BLACKBOX is
    Port ( J_jkff, K_jkff, Reset_jkff, Clock_jkff : in STD_LOGIC;
           Q_jkff, Qn_jkff : out STD_LOGIC);
end EXP4_PRE_Q4_BLACKBOX;

architecture Behavioral of EXP4_PRE_Q4_BLACKBOX is
component EXP4_PRE_Q4_DFLIPFLOP
    Port ( D_dff, Reset_dff, Clock_dff : in STD_LOGIC;
           Q_dff, Qn_dff : out STD_LOGIC);
end component;
signal tempQ, tempQn, D : STD_LOGIC;
begin
    D <= (tempQ and not K_jkff) or (tempQn and J_jkff);
    dff : EXP4_PRE_Q4_DFLIPFLOP port map (D, Reset_jkff, Clock_jkff,
                                          tempQ, tempQn);
    Q_jkff <= tempQ;
    Qn_jkff <= tempQn;
end Behavioral;
```

3. Results

Test bench

```
stim_proc: process
begin
  Reset_jkff <= '1'; wait for Clock_jkff_period;
  J_jkff <= '0'; K_jkff <= '0'; Reset_jkff<='0'; wait for Clock_jkff_period;
  J_jkff <= '0'; K_jkff <= '1'; wait for Clock_jkff_period;
  J_jkff <= '1'; K_jkff <= '0'; wait for Clock_jkff_period;
  J_jkff <= '1'; K_jkff <= '1'; wait for Clock_jkff_period*2;
  J_jkff <= '0'; K_jkff <= '1'; wait for Clock_jkff_period;
  J_jkff <= '0'; K_jkff <= '0'; wait for Clock_jkff_period;
end process;
```