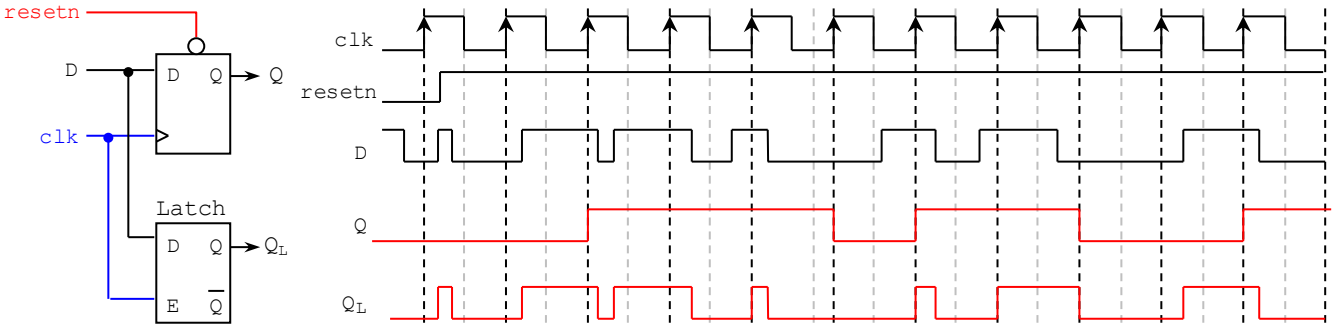
# Solutions - Homework 3

(Due date: November 1st @ 5:30 pm)

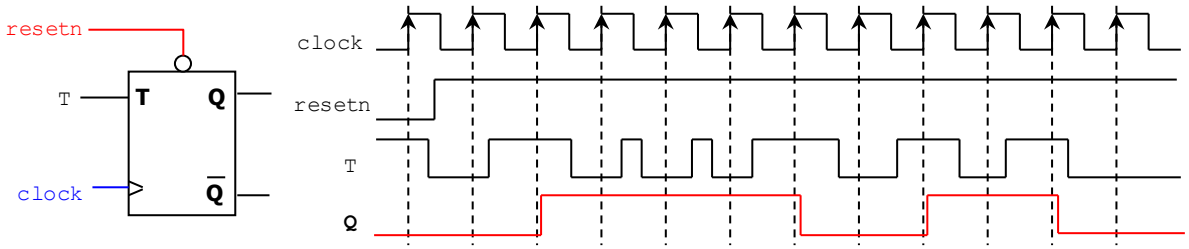
Presentation and clarity are very important! Show your procedure!

## Problem 1 (12 pts)

- Complete the timing diagram for the flip flop and the latch shown below: (7 pts)

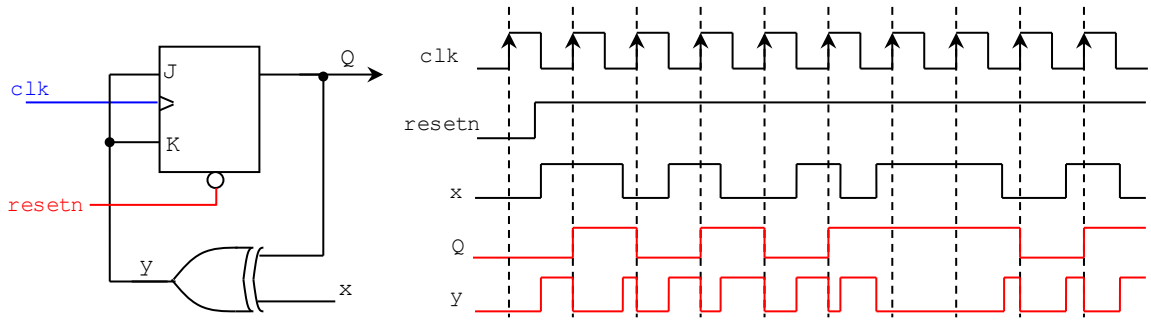



- Complete the timing diagram of the circuit shown below. (5 pts)

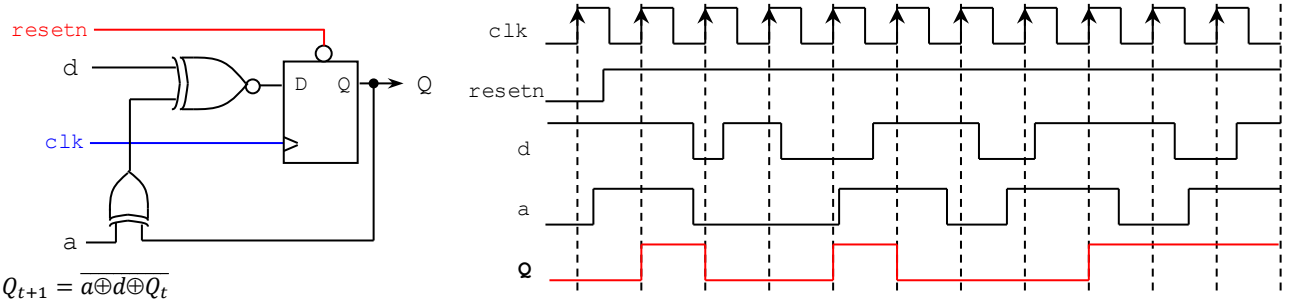



## Problem 2 (40 pts)

- Complete the timing diagram of the circuit shown below: (7 pts)

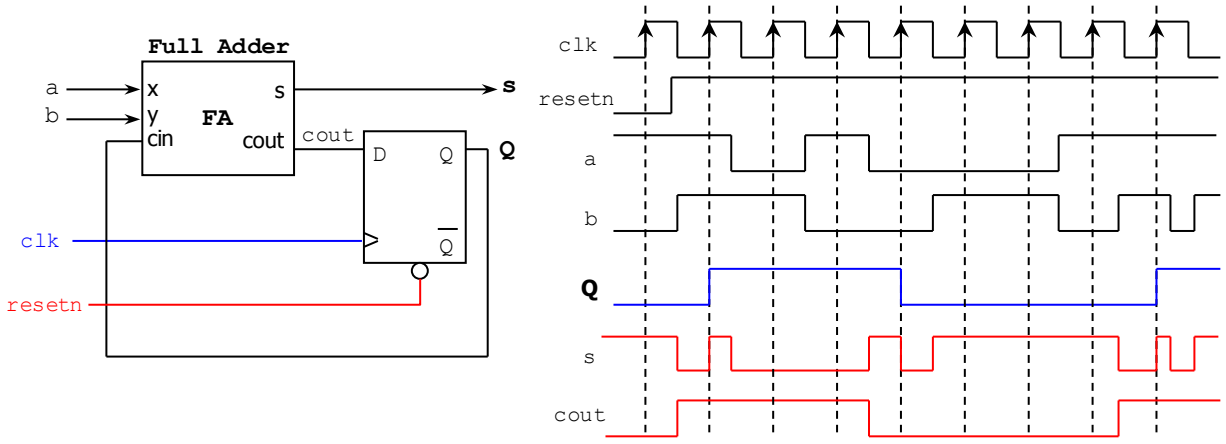



- Complete the timing diagram of the circuit shown below: (8 pts)

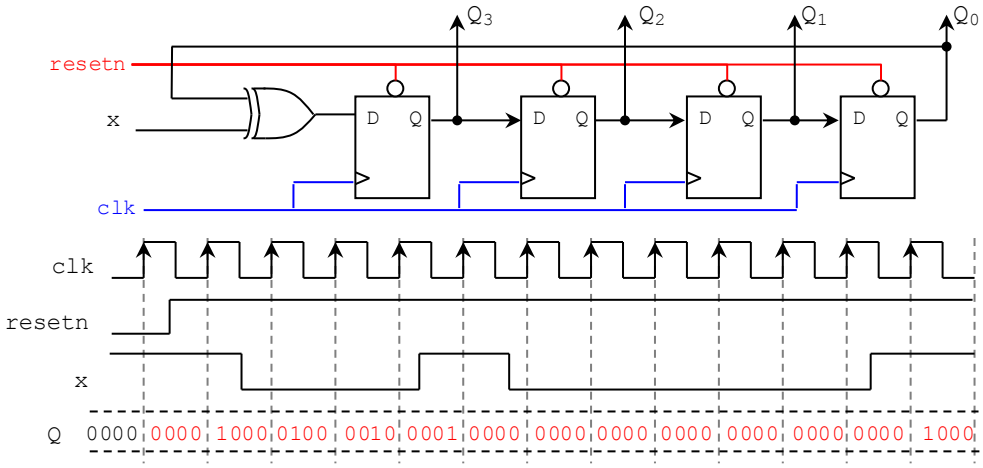



$Q_{t+1} = \overline{a \oplus d \oplus Q_t}$

- Complete the timing diagram of the circuit shown below: (10 pts)

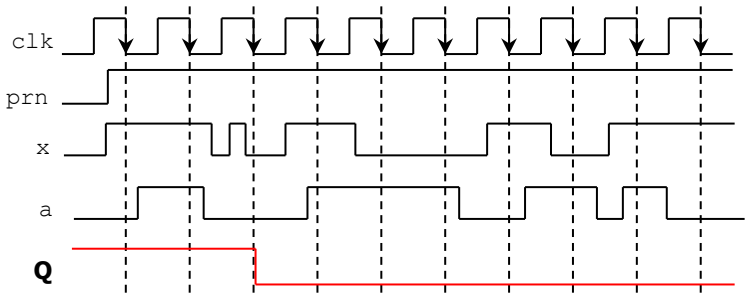



- Complete the timing diagram of the circuit shown below. $Q = Q_3Q_2Q_1Q_0$ (9 pts)




- Complete the timing diagram of the circuit whose VHDL description is shown below: (6 pts)

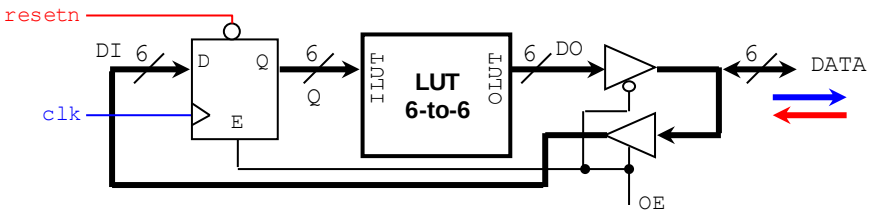
```
library ieee;
use ieee.std_logic_1164.all;

entity circ is
  port ( prn, a,x,clk: in std_logic;
         q: out std_logic);
end circ;

architecture t of circ is

   signal qt: std_logic;

begin
   process (prn, clk, a, x)
   begin
     if prn = '0' then
         qt <= '1';
      elsif (clk'event and clk = '0') then
         if x = '0' then
             qt <= qt xor not(a);
         end if;
      end if;
   end process;
   q <= qt;
end t;
```

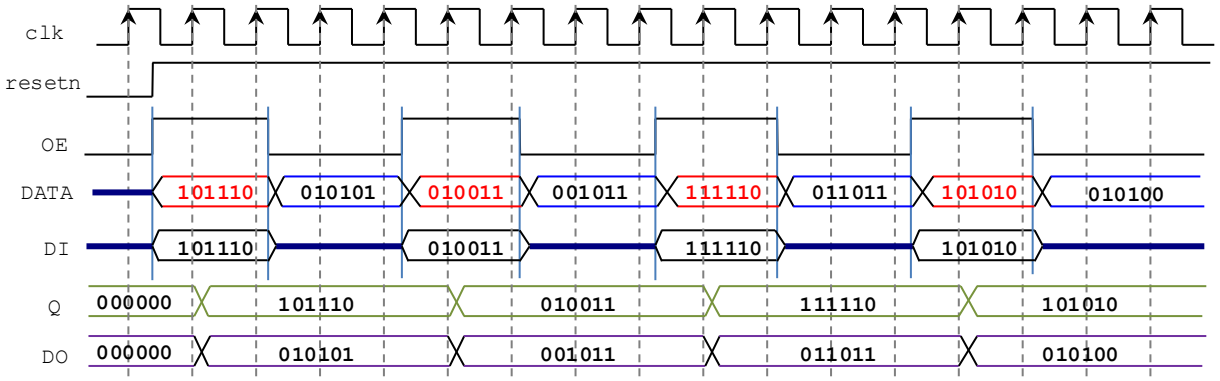


# PROBLEM 3 (18 PTS)

- Given the following circuit, complete the timing diagram (signals $DO$ and $DATA$).
  The LUT 6-to-6 implements the following function: $OLUT = \lceil ILUT^{0.79} \rceil$, where $ILUT$ is a 6-bit unsigned number.
  For example $ILUT = 35\ (100011_2) \rightarrow OLUT = \lceil 35^{0.79} \rceil = 17\ (010001_2)$

| | |
|---|---|
| $ILUT = 46\ (101110_2) \rightarrow OLUT = \lceil 46^{0.79} \rceil = 21\ (010101_2)$ | $ILUT = 19\ (010011_2) \rightarrow OLUT = \lceil 19^{0.79} \rceil = 11\ (001011_2)$ |
| $ILUT = 62\ (111110_2) \rightarrow OLUT = \lceil 62^{0.79} \rceil = 27\ (011011_2)$ | $ILUT = 42\ (101010_2) \rightarrow OLUT = \lceil 42^{0.79} \rceil = 20\ (010100_2)$ |




# PROBLEM 4 (30 PTS)

- The following circuit is a 4-bit parallel/serial load shift register with enable input.
  Shifting operation: `s_l=0`. Parallel load: `s_l=1`. Note that $Q = Q_3Q_2Q_1Q_0$. $D = D_3D_2D_1D_0$
  - ✓ Write a structural VHDL code. You MUST create a file for: i) flip flop, ii) MUX 2-to-1, and iii) top file (where you will interconnect the flip flops and MUXes). Provide a printout. (10 pts)

**VHDL Code**: Top File

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

entity my_st_shiftreg is
     generic (N: INTEGER:= 4;
               DIR: STRING:= "RIGHT"); -- RIGHT/LEFT
     port (D: in std_logic_vector (N-1 downto 0);
           resetn, clock, din, E, s_l: in std_logic;
           Q: out std_logic_vector (N-1 downto 0));
end my_st_shiftreg;

architecture structure of my_st_shiftreg is
  component dffe
     port ( d : in  STD_LOGIC;
            clrn, prn, clk, ena: in std_logic;
            q : out  STD_LOGIC);
  end component;

  component mux2to1
     port (a,b, sel : in std_logic;
           y : out std_logic);
  end component;

  signal ds, md, Qt: std_logic_vector (N-1 downto 0);
begin

a0: assert (DIR = "LEFT" or DIR = "RIGHT")
    report "DIR can only be LEFT or RIGHT"
    severity error;
```

```
rr: if DIR = "RIGHT" generate
        ds(N-1) <= din;
        ds(N-2 downto 0) <= Qt(N-1 downto 1);
    end generate;

rl: if DIR = "LEFT" generate
        ds(0) <= din;
        ds(N-1 downto 1) <= Qt(N-2 downto 0);
    end generate;

ti: for i in N-1 downto 0 generate
       fi: dffe port map (d => md(i), clrn => resetn, prn =>'1', clk => clock, ena => E, q => Qt(i));
       mi: mux2to1 port map (a => ds(i), b => D(i), sel => s_l, y => md(i));
    end generate;

    Q <= Qt;
end structure;
```

**VHDL Code**: D-Type flip flop

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

entity dffe is
    port ( d : in  STD_LOGIC;
           clrn, prn, clk, ena: in std_logic;
           q : out  STD_LOGIC);
end dffe;

architecture behaviour of dffe is

begin
     process (clk, ena, prn, clrn)
     begin
        if clrn = '0' then q <= '0';
        elsif prn = '0' then q <= '1';
        elsif (clk'event and clk='1') then
           if ena = '1' then q <= d; end if;
        end if;
     end process;
end behaviour;
```

**VHDL Code**: MUX 2-to-1

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

entity mux2to1 is
    port (a,b, sel: in std_logic;
          y : out std_logic);
end mux2to1;

architecture structure of mux2to1 is

begin

    with sel select
            y <= a when '0',
                 b when others;

end structure;
```

✓ Write a VHDL testbench according to the timing diagram shown below. Complete the timing diagram by simulating your circuit (Behavioral Simulation). The clock frequency must be 50 MHz with 50% duty cycle. Provide a printout. (20 pts)

**VHDL Tesbench**:

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

ENTITY tb_my_st_shiftreg IS
    generic (N: INTEGER:= 4);
END tb_my_st_shiftreg;

ARCHITECTURE behavior OF tb_my_st_shiftreg IS
    COMPONENT my_st_shiftreg
    PORT(
         D : IN  std_logic_vector(N-1 downto 0);
         resetn, clock, din : IN  std_logic;
         E, s_l : IN  std_logic;
         Q : OUT  std_logic_vector(N-1 downto 0));
    END COMPONENT;

   --Inputs
   signal D : std_logic_vector(N-1 downto 0) := (others => '0');
   signal resetn : std_logic := '0';
   signal clock : std_logic := '0';
   signal din : std_logic := '0';
   signal E : std_logic := '0';
   signal s_l : std_logic := '0';

    --Outputs
   signal Q : std_logic_vector(N-1 downto 0);
```

```
    -- Clock period definitions
    constant T : time := 20 ns;

BEGIN
     -- Instantiate the Unit Under Test (UUT)
    uut: my_st_shiftreg PORT MAP (D => D, resetn => resetn, clock => clock, din => din,
                                  E => E, s_l => s_l, Q => Q);

    -- Clock process definitions
    clock_process: process
    begin
       clock <= '0'; wait for T/2;
       clock <= '1'; wait for T/2;
    end process;

    -- Stimulus process
    stim_proc: process
    begin
      resetn <= '0'; wait for 100 ns;
      D <= "1000"; din <= '1'; E <= '0'; s_l <= '1'; wait for T;
      resetn <= '1';
      D <= "1000"; din <= '1'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1000"; din <= '0'; E <= '1'; s_l <= '0'; wait for T;
      D <= "1010"; din <= '0'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1010"; din <= '1'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1010"; din <= '1'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1010"; din <= '0'; E <= '0'; s_l <= '1'; wait for T;
      D <= "1101"; din <= '1'; E <= '0'; s_l <= '0'; wait for T;
      D <= "1101"; din <= '0'; E <= '1'; s_l <= '0'; wait for T;
      D <= "1101"; din <= '0'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1110"; din <= '0'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1110"; din <= '1'; E <= '1'; s_l <= '0'; wait for T;
      D <= "1110"; din <= '1'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1110"; din <= '0'; E <= '0'; s_l <= '1'; wait for T;
      D <= "1011"; din <= '0'; E <= '1'; s_l <= '1'; wait for T;
      D <= "1011"; din <= '1'; E <= '1'; s_l <= '0'; wait for T;
      D <= "1011"; din <= '0'; E <= '1'; s_l <= '1'; wait for T;
      wait;
    end process;
END;
```