

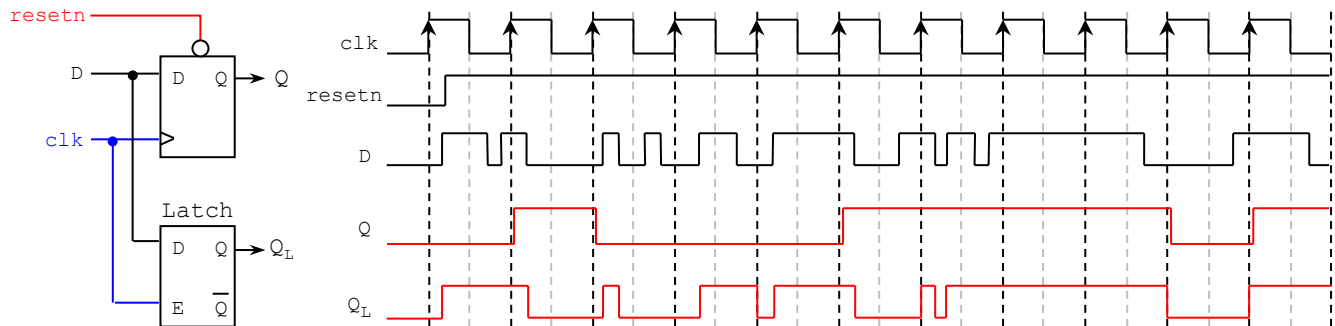
# Solutions - Homework 3

(Due date: October 31<sup>st</sup> @ 11:59 pm)

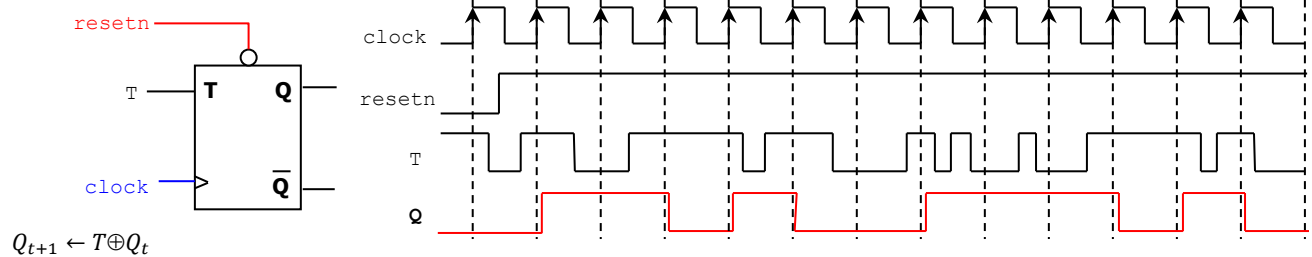
Presentation and clarity are very important! Show your procedure!

## PROBLEM 1 (11 PTS)

a) Complete the timing diagram of the circuits shown below. (5 pts)

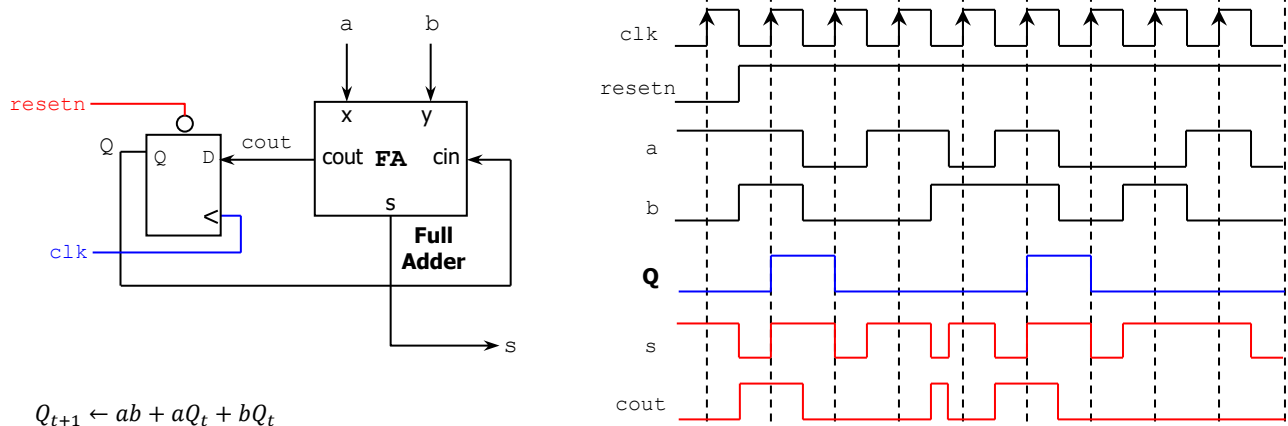


b) Complete the timing diagram of the circuit shown below: (6 pts)

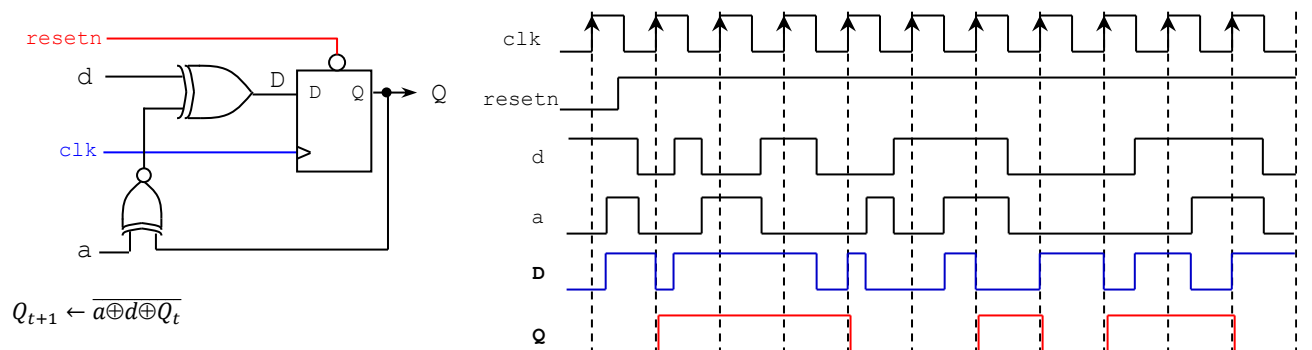


## PROBLEM 2 (17 PTS)

Complete the timing diagram of the circuit shown below. Get the excitation equation for Q. (10 pts)



Complete the timing diagram of the circuit shown below. Get the excitation equation for Q. (7 pts)



### PROBLEM 3 (10 PTS)

a) Complete the timing diagram of the circuit whose VHDL description is shown below. Also, get the excitation equation for  $q$ .

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

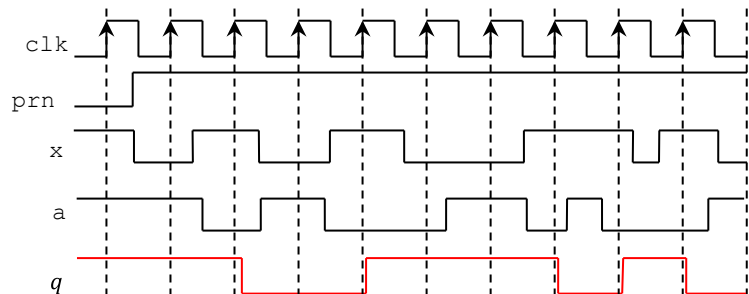
entity circ is
  port (prn, clk, a, x: in std_logic;
        q: out std_logic);
end circ;
```

architecture a of circ is

signal qt: std\_logic;

```
begin
  process (prn, clk, x, a)
  begin
    if prn = '0' then
      qt <= '1';
```

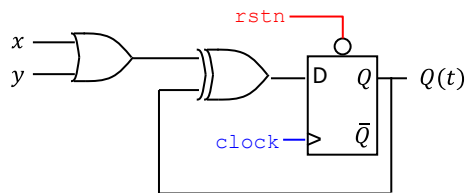
```
    elsif (clk'event and clk = '1') then
      if x = '1' then
        qt <= a xnor qt;
      end if;
    end if;
  end process;
  q <= qt;
end a;
```



$$q(t+1) \leftarrow \bar{x}q(t) + x(a \oplus q(t))$$

b) With a flip flop and logic gates, sketch the circuit whose excitation equations is given by (4 pts):

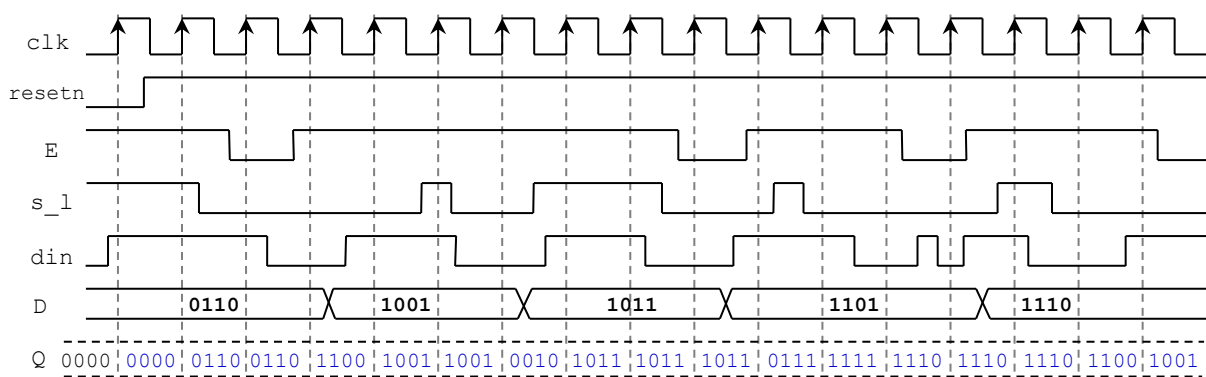
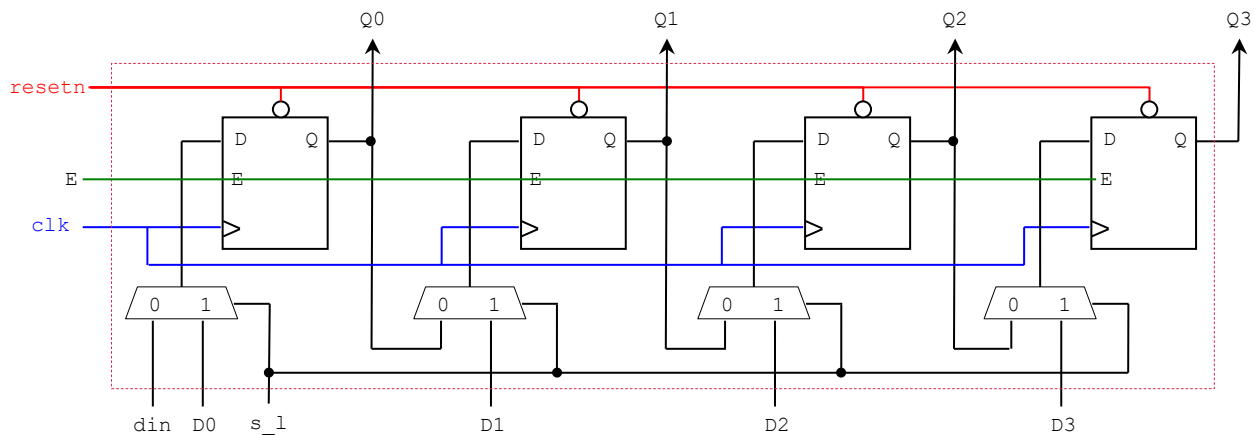
$$Q(t+1) \leftarrow (x + y) \oplus Q(t)$$



### PROBLEM 4 (10 PTS)

Complete the timing diagram of the following 4-bit parallel access shift register with enable input.

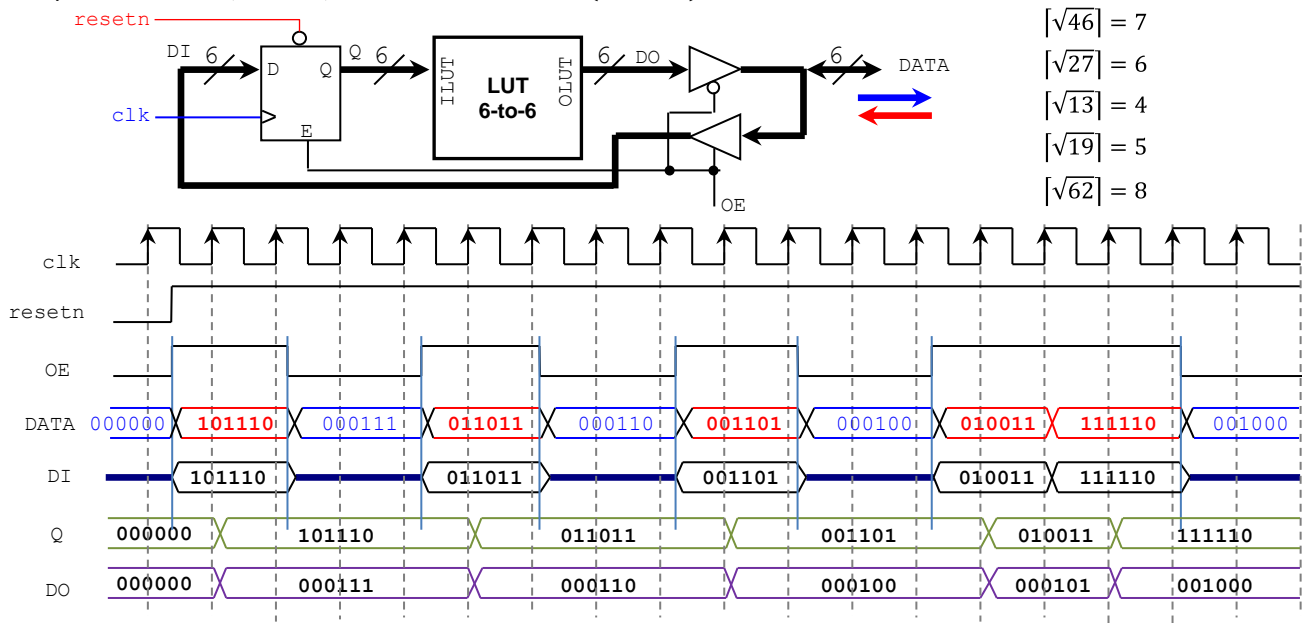
When  $E=1$ : If  $s\_l=0$  (shifting operation). If  $s\_l=1$  (parallel load) Note that  $Q = Q_3Q_2Q_1Q_0$ .  $D = D_3D_2D_1D_0$



### PROBLEM 5 (12 PTS)

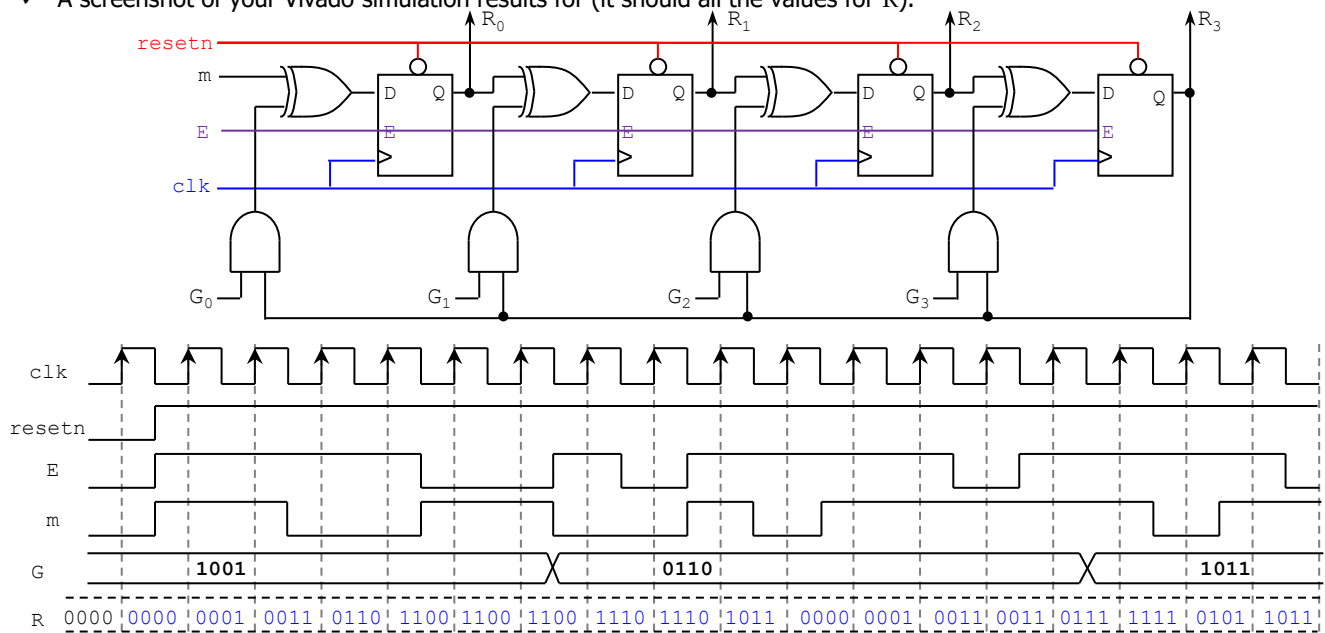
- Given the following circuit, complete the timing diagram (signals  $DO$ ,  $DI$ ,  $Q$ , and  $DATA$ ). The LUT 6-to-6 implements the following function:  $OLUT = [ILUT^{0.5}]$ , where  $ILUT$  is an unsigned number.

Example:  $ILUT = 53 (110101_2) \rightarrow OLUT = [53^{0.5}] = 8 (001000_2)$



### PROBLEM 6 (22 PTS)

- For the following circuit, complete the timing diagram and get the excitation equations of the flip flop outputs.  $R = R_3R_2R_1R_0$ .  
 $R_3(t+1) \leftarrow \bar{E}R_3(t) + E \cdot ((G_3(t)R_3(t)) \oplus R_2(t))$   
 $R_2(t+1) \leftarrow \bar{E}R_2(t) + E \cdot ((G_2(t)R_3(t)) \oplus R_1(t))$   
 $R_1(t+1) \leftarrow \bar{E}R_1(t) + E \cdot ((G_1(t)R_3(t)) \oplus R_0(t))$   
 $R_0(t+1) \leftarrow \bar{E}R_0(t) + E \cdot ((G_0(t)R_3(t)) \oplus m)$
- Write the VHDL for the given circuit and simulate your circuit.
  - Write structural VHDL code. Create two files: i) flip flop, ii) top file (where you will interconnect the flip flops and the logic gates). (10 pts)
  - Write a VHDL testbench according to the timing diagram shown below (100 MHz clock with 50% duty cycle). Run the simulation (Behavioral Simulation). Verify the results: compare them with the manually completed timing diagram (8 pts)
- Upload (as a .zip file) the following files to Moodle (an assignment will be created):
  - VHDL code files and testbench.
  - A screenshot of your Vivado simulation results for (it should all the values for R).



✓ **VHDL Code: Top File**

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

entity lfsr_crc is
    generic (N: INTEGER:= 4);
    port ( m_in, E: in std_logic;
          resetn, clock: in std_logic;
          G: in std_logic_vector (N-1 downto 0);
          R: out std_logic_vector (N-1 downto 0));
end lfsr_crc;

architecture structural of lfsr_crc is
    component dffe
        port ( d : in  STD_LOGIC;
              clrn: in std_logic:= '1';
              prn: in std_logic:= '1';
              clk : in  STD_LOGIC;
              ena: in std_logic;
              q : out  STD_LOGIC);
    end component;

    signal B, D, Q: std_logic_vector (N-1 downto 0);

begin

    D(0) <= m_in xor B(0);
    g0: for i in 1 to N-1 generate
        D(i) <= B(i) xor Q(i-1);
    end generate;

    g1: for i in 0 to N-1 generate
        di: dffe port map (d => D(i), clrn => resetn, prn => '1', clk => clock, ena => E, q => Q(i));
        R(i) <= Q(i);
        B(i) <= G(i) and Q(N-1);
    end generate;

end structural;
```

✓ **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 Tesbench:

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY tb_crc IS
    generic (N: integer:= 4);
END tb_crc;

ARCHITECTURE behavior OF tb_crc IS
    component lfsr_crc
        port ( m_in, E : IN  std_logic;
              resetn, clock : IN  std_logic;
              G: in std_logic_vector (N-1 downto 0);
              R : OUT  std_logic_vector(N-1 downto 0));
    end component;

    --Inputs
    signal m_in, E : std_logic := '0';
    signal resetn, clock: std_logic := '0';
    signal G: std_logic_vector (N-1 downto 0);

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

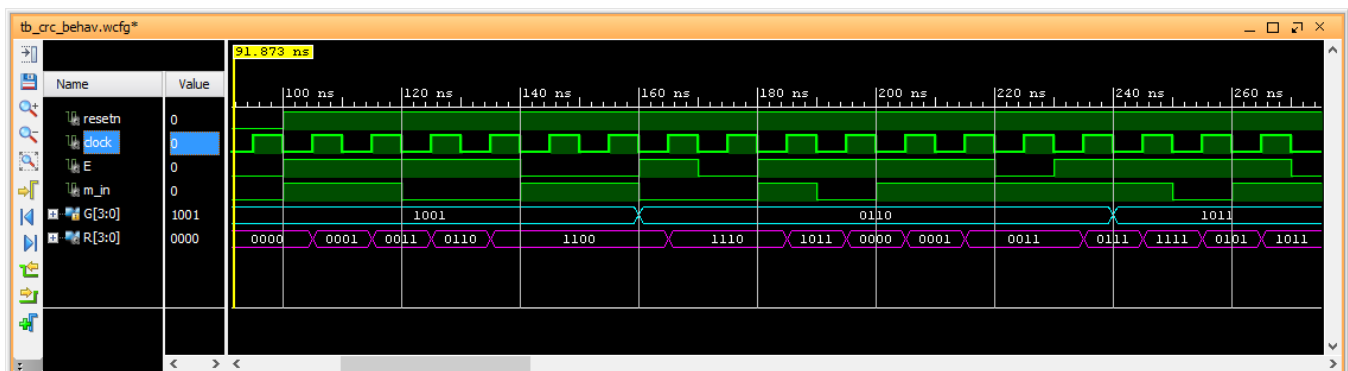
    constant T : time := 10 ns; -- clock period definition

BEGIN
    -- Instantiate the Unit Under Test (UUT)
    uut: lfsr_crc PORT MAP (m_in => m_in, E => E, resetn => resetn, clock => clock, G => G, R => R);

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

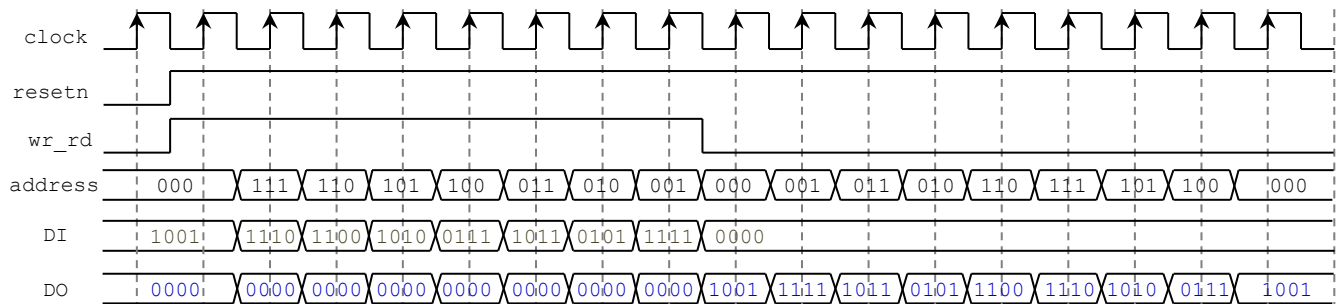
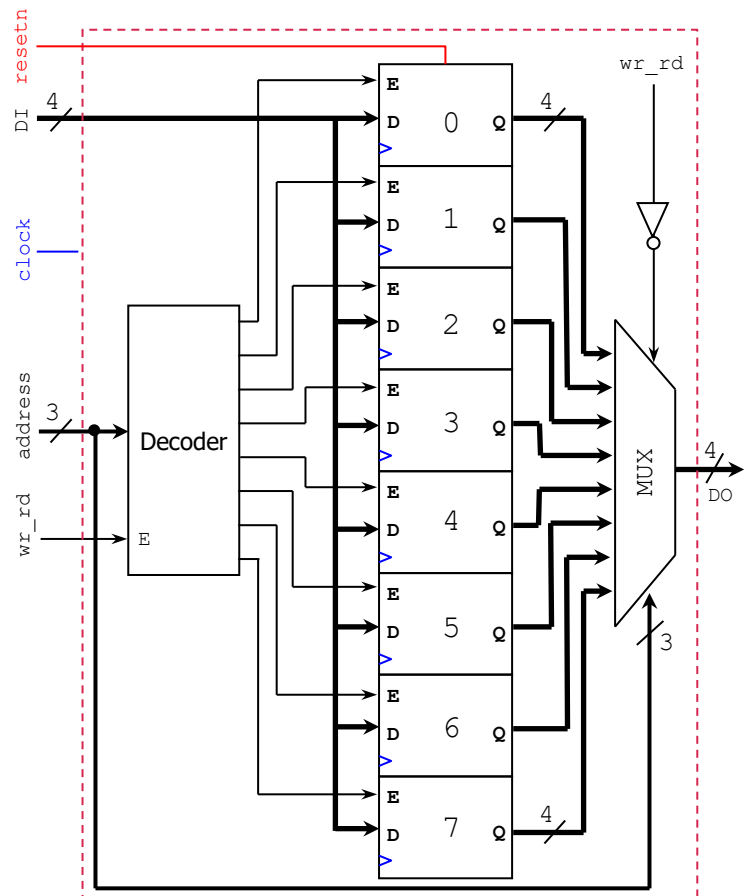
    -- Stimulus process
    stim_proc: process
    begin
        resetn <= '0'; m_in <= '0'; G <= "1001";
        wait for 100 ns; resetn <= '1';
        G <= "1001"; E <= '1'; m_in <= '1'; wait for 2*T;
        G <= "1001"; E <= '1'; m_in <= '0'; wait for 2*T;
        G <= "1001"; E <= '0'; m_in <= '1'; wait for 2*T;
        G <= "0110"; E <= '1'; m_in <= '0'; wait for T;
        G <= "0110"; E <= '0'; m_in <= '0'; wait for T;
        G <= "0110"; E <= '1'; m_in <= '1'; wait for T;
        G <= "0110"; E <= '1'; m_in <= '0'; wait for T;
        G <= "0110"; E <= '1'; m_in <= '1'; wait for 2*T;
        G <= "0110"; E <= '0'; m_in <= '1'; wait for T;
        G <= "0110"; E <= '1'; m_in <= '1'; wait for T;
        G <= "1011"; E <= '1'; m_in <= '1'; wait for T;
        G <= "1011"; E <= '1'; m_in <= '0'; wait for T;
        G <= "1011"; E <= '1'; m_in <= '1'; wait for T;
        G <= "1011"; E <= '0'; m_in <= '1';
        wait;
    end process;

END;
```



### PROBLEM 7 (8 PTS)

- Complete the timing diagram (output DO) of the following Random Memory Access (RAM) Emulator.
- RAM Emulator: It has 8 addresses, where each address holds a 4-bit data. The memory positions are implemented by 4-bit registers. The *resetn* and *clock* signals are shared by all the registers. Data is written or read onto/from one of the registers (selected by the signal address).
- Operations:
  - Writing onto memory (*wr\_rd*=1): The 4-bit input data (DI) is written into one of the 8 registers. The address signal selects which register is to be written.
    - For example: if address = "101", then the value of DI is written into register 5.
    - Note that because the BusMUX 8-to-1 includes an enable input, if *wr\_rd*=1, then the BusMUX outputs are 0's.
  - Reading from memory (*wr\_rd*=0): The address signal selects the register from which data is read. This data appears on the BusMUX output.
    - For example: If address = "010", then data from register 2 appears on BusMUX output.



### PROBLEM 8 (10 PTS)

- Attach your Project Status Report (no more than 1 page, single-spaced, 2 columns, only one submission per group). This report should contain the initial status of your project. For formatting, use the provided template (Final Project - Report Template.docx). The sections included in the template are the ones required in your Final Report. At this stage, you are only required to:
  - Include a (draft) project description and title.
  - Include a draft Block Diagram of your hardware architecture.
- As a guideline, the figure shows a simple Block Diagram. There are input and output signals, as well as internal components along with their interconnection.
  - At this stage, only a rough draft is required. There is no need to go into details: it is enough to show the tentative top-level components that would constitute your system as well as the tentative inputs and outputs.
- Only student is needed to attach the report (make sure to indicate all the team members).

