CSc 21100 (Fall 2019)
Project 04 (20 points)

IMPORTANT!

Please follow the **submission guidelines** below or your submission will be rejected.

1. You are expected to submit both a lab report and the source files to Blackboard in a single submission attempt.
2. The source codes must be under a single project.
3. The VHDL project needs to be exported from Xilinx ISE Design Suite. To export VHDL project file, please refer to Blackboard -> Content -> Lab -> **Exporting VHDL Project Files**.
4. Naming convention:
   Report: "FirstName_LastName_Project_**XX**_MM**Y**.pdf"
   Project: "FirstName_LastName_Project_**XX**_MM**Y**.zip"

   Replace "XX" and "Y" with the actual project number and section number, respectively.

In this project, students are expected to use the Xilinx ISE Design Suite (Webpack edition) 14.7 to complete the following tasks.

Please read the instructions carefully. Failing to follow the instructions would lead to significant point deductions.

Task 1: Positive Edge Triggered D Flip-Flop (7 points)

Write a VHDL program to implement a Positive Edge Triggered D Flip-Flop. Please adopt the following entity declaration.

```
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;
```

In this task, use the *LD* component (essentially a D Latch) defined in the *UNISIM* library. In order to use the *LD* component, please make sure you have the following lines in your VHDL program to include the *UNISIM* library.

```
library UNISIM;
use UNISIM.VComponents.all;
```

Then you will need to declare the component in the VDHL architecture definition. For example:

```
architecture Structural of TEST is
   component LD
      generic( INIT : bit :=  '0');
      port ( D : in    std_logic;
             G : in    std_logic;
             Q : out   std_logic);
   end component;
```

Then you can use it like this:

```
D_Latch_1 : LD port map (D=>Signal_1, G=>Signal_2, Q=>Signal_3);
```

In your program, please make sure you name the input/output signals according to the truth table.

D	CLK	Q	QN
0	↑	0	1
1	↑	1	0
x	0	last Q	last QN
x	1	last Q	last QN

Write a test-bench program and run simulations to validate your design. Use the given test cases in your test-bench program. Pay attention to the signal names, signal values, and the time.

```
36    constant CLK_period : time := 10 ns;
37
38  BEGIN
39     UUT: pet_d_ff PORT MAP(
40        D => D,
41        CLK => CLK,
42        Q => Q,
43        QN => QN
44     );
45     CLK_process : process
46     begin
47        CLK <= '0';
48        wait for CLK_period/2;
49        CLK <= '1';
50        wait for CLK_period/2;
51     end process;
52  -- *** Test Bench - User Defined Section ***
53     tb : PROCESS
54     BEGIN
55        wait for 20 ns;
56        d <= '0';
57        wait for CLK_period*2;
58        d <= '1';
59        wait for CLK_period*2;
60        d <= '0';
61        wait for CLK_period*2;
62        d <= '1';
63        wait for CLK_period*2;
64        WAIT; -- will wait forever
65     END PROCESS;
66  -- *** End Test Bench - User Defined Section ***
67
68  END;
```

Requirement(s):
In your VHDL implementation
(1) Please follow the structural design method;
(2) Make use of the *LD* component.

Deliverable(s):

Your report:

1. Use your own language to describe the function of the module to be implemented in VHDL. (1 point)
2. Draw a circuit diagram of the module to show the design. (1 point)
3. Include your VHDL entity declaration(s), architecture definition(s) and the testbench program. (1 point)
4. Show simulation results (e.g. the waveforms). Explain the outcome of **each** testcase with screenshots. Show why the simulation result is correct. (2 points)

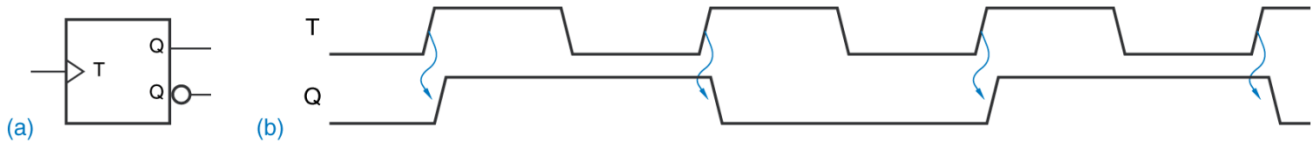
Your project file(s):

1. Can compile without any errors. (1 point)
2. Can run simulations without any errors. (1 point)

Note: no points will be given if requirements are not satisfied.

Task 2: T Flip-Flop (7 points)

A *T* (*toggle*) flip-flop changes state on every tick of the clock. The figures below show the symbol and illustrate the behavior of a positive-edge-triggered T flip-flop. Note that according to the logic symbol, the T flip-flop is edge triggered.



A T flip-flop can be built from a positive-edge-triggered (PET) D flip-flop. In the previous task, you implemented such a PET D flip-flop. In this task, please write a VHDL program to implement a T flip-flop by using the PET D flip-flop. Please adopt the following entity declaration.

```
entity t_ff is
   port ( CLK : in    std_logic;
          Q   : out   std_logic;
          QN  : out   std_logic);
end t_ff;
```

Use the following test-bench program to run simulations and to validate your design.

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.ALL;
3  USE ieee.numeric_std.ALL;
4  LIBRARY UNISIM;
5  USE UNISIM.Vcomponents.ALL;
6
7  ENTITY t_ff_tb IS
8  END t_ff_tb;
9
10 ARCHITECTURE behavioral OF t_ff_tb IS
11    COMPONENT t_ff
12    PORT( QN :  OUT   STD_LOGIC;
13          CLK:  IN    STD_LOGIC;
14          Q  :  OUT   STD_LOGIC);
15    END COMPONENT;
16    SIGNAL QN   :  STD_LOGIC;
17    SIGNAL CLK  :  STD_LOGIC;
18    SIGNAL Q :  STD_LOGIC;
19    constant CLK_period : time := 10 ns;
21 BEGIN
22
23    UUT: t_ff PORT MAP(
24       QN => QN,
25       CLK => CLK,
26       Q => Q
27    );
28    -- Clock process definitions
29    CLK_process :process
30    begin
31       CLK <= '0';
32       wait for CLK_period/2;
33       CLK <= '1';
34       wait for CLK_period/2;
35    end process;
36
37 END;
```

Requirement(s):
In your VHDL implementation
(1) Please follow the structural design method;
(2) Make use of the module(s) you implemented before.

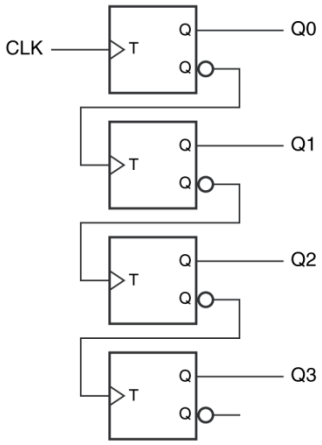
Deliverable(s):

Your report:
1. Draw a circuit diagram of the module to show the design. (1 point)
2. Use your own language to explain how the circuit works. (1 point)
3. Include your VHDL entity declaration(s), architecture definition(s) and the testbench program. (1 point)
4. Show simulation results (e.g. the waveforms). Explain the outcome of **each** testcase with screenshots. Show why the simulation result is correct. (2 points)

Your project file(s):
5. Can compile without any errors. (1 point)
6. Can run simulations without any errors. (1 point)

Note: no points will be given if requirements are not satisfied.

Task 3: 4-bit Binary Ripple Counter (6 points)



An n-bit binary counter can be constructed using n T flip-flops. In this task, please use the given design to implement a 4-bit binary ripple counter in VHDL.

Please use the following entity declaration for the counter.

```
entity RippleCounter4b is
   port ( CLK : in    std_logic;
          Q0  : out   std_logic;
          Q1  : out   std_logic;
          Q2  : out   std_logic;
          Q3  : out   std_logic);
end RippleCounter4b;
```

Use the following test-bench program to run simulations and to validate your design.

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
LIBRARY UNISIM;
USE UNISIM.Vcomponents.ALL;

ENTITY RippleCounter4b_tb IS
END RippleCounter4b_tb;

ARCHITECTURE behavioral OF
RippleCounter4b_tb IS
   COMPONENT RippleCounter4b
   PORT( CLK   :  IN STD_LOGIC;
         Q3    :  OUT   STD_LOGIC;
         Q2    :  OUT   STD_LOGIC;
         Q1    :  OUT   STD_LOGIC;
         Q0    :  OUT   STD_LOGIC);
   END COMPONENT;
   SIGNAL CLK  :  STD_LOGIC;
   SIGNAL Q3   :  STD_LOGIC;
   SIGNAL Q2   :  STD_LOGIC;
   SIGNAL Q1   :  STD_LOGIC;
   SIGNAL Q0   :  STD_LOGIC;
   constant CLK_period : time := 10 ns;
BEGIN
   UUT: counter_4bit PORT MAP(
      CLK => CLK,
      Q3 => Q3,
      Q2 => Q2,
      Q1 => Q1,
      Q0 => Q0
   );
   -- Clock process definitions
   CLK_process :process
   begin
      CLK <= '0';
      wait for CLK_period/2;
      CLK <= '1';
      wait for CLK_period/2;
   end process;
END;
```

Requirement(s):
In your VHDL implementation
(1) Please follow the structural design method;
(2) Make use of the module(s) you implemented before.

Deliverable(s):

Your report:

1. Use your own language to explain how the circuit works. (1 point)
2. Include your VHDL entity declaration(s), architecture definition(s) and the testbench program. (1 point)
3. Show simulation results (e.g. the waveforms). Explain the outcome of **each** testcase with screenshots. Show why the simulation result is correct. (2 points)

Your project file(s):

1. Can compile without any errors. (1 point)
2. Can run simulations without any errors. (1 point)

Note: no points will be given if requirements are not satisfied.