The Chinese University of Hong Kong
Department of Computer Science and Engineering
CENG2010 Digital Logic Design Laboratory

Lab 2: VHDL Concurrent Statements

Submission Instructions:

- You are required to submit **BOTH demo videos** and **VHDL codes** to Blackboard.
- Create each VHDL project with a project name based on the lab and question number, e.g. "**ceng2010_lab1_q2**".
- Zip all the project folders to ONE single zip/rar file named with your student ID number, e.g. "**1155123456.zip**".
- Upload the zip/rar file to Blackboard before the deadline stated in Blackboard
- **No late submission will be accepted**

For each question below, you are required to record a short mp4 video to demonstrate the answers. In the video, the following elements are required:

A. Next to your FPGA board, show your full name and SID on a paper [5 marks]
B. Voice descriptions in English/Cantonese/Mandarin on what you are doing [5 marks]
C. Demonstrate works by presenting all possible input combinations step-by-step clearly [30 marks]

In this lab, both of the switches and LEDs are defined as standard logic **vector**. Therefore, you should double check all **sw** and **led** variables in the constraints file should have a bracket **[ ]** as below:

```
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
```

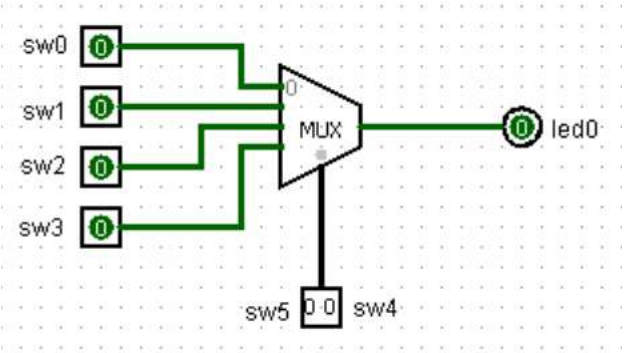
Also, you should access the variables in VHDL with another bracket **( )** as below:

```
led(0) <= sw(0);
```

1. Using VHDL, define both switches and LEDs are vectors as below, implement a 4-to-1 multiplexer (MUX) with the following specifications: [20 marks]

```
entity Lab02_q1 is
    Port ( sw : in STD_LOGIC_VECTOR (5 downto 0);
           led : out STD_LOGIC_VECTOR (0 downto 0));
end Lab02_q1;
```

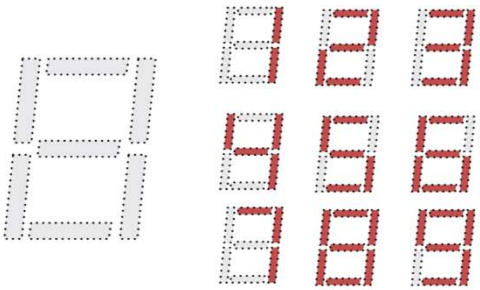
   a. sw3 to sw0 are the data inputs
   b. sw5 to sw4 are the 2-bit selectors
   c. led0 is the output



sw5	sw4	led0
0	0	sw0
0	1	sw1
1	0	sw2
1	1	sw3

2. Using VHDL, implement a BCD to 7-segment decoder for all digits in the following table, **without** using **PROCESS** which will be covered in the coming lessons. Display the decoded digit "**0**" to "**9**" at the right-most digit of the 7-segment display, and display an "**E**" for the rest input patterns. Please noted that a digit can be selected by giving a "0" to an(3:0), and a segment can be lighted up by giving a "0" to seg(6:0). [20 marks]

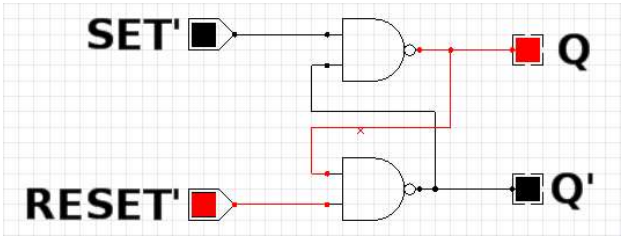
   a. 4-bit BCD code defined as sw(3:0) refers to sw3(MSB), sw2, sw1, and sw0(LSB) respectively
   b. 7-bit seg(6:0) refers to seg6(MSB), seg5, seg4, seg3, seg2, seg1, and seg0(LSB) respectively



3. Using VHDL to implement the RS Flip-flop below using NAND gates only. [20 marks]

   To avoid the combinatorial loops at Q and Q' which will be presented by led1 and led0 respectively, you should double check if the following lines are in the constraints file.

```
set_property ALLOW_COMBINATORIAL_LOOPS TRUE [get_nets led_OBUF[0]]
set_property ALLOW_COMBINATORIAL_LOOPS TRUE [get_nets led_OBUF[1]]
```



SET	RESET	SET'	RESET'	Q	Q'	STATE
1	1	0	0	1	1	Prohibited
1	0	0	1	1	0	Set
0	1	1	0	0	1	Reset
0	0	1	1	Q	Q'	Unchanged

   a. Use sw1 and sw0 for SET' and RESET' respectively.
   b. Use led1 and led0 for Q and Q' respectively.
   c. Please noted that output ports cannot be used as the inputs of SIGNAL assignments.

   Input the SET' and RESET' patterns according to Sequence ID, show the Q and Q' outputs in the video.

Sequence ID	SET'	RESET'	Q	Q'
1	0	1	**1**	**0**
2	1	0	**0**	**1**
3	1	1	**0**	**1**
4	0	1	**1**	**0**
5	1	1	**1**	**0**

THE END