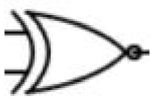
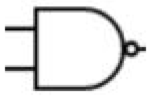
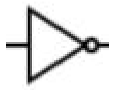
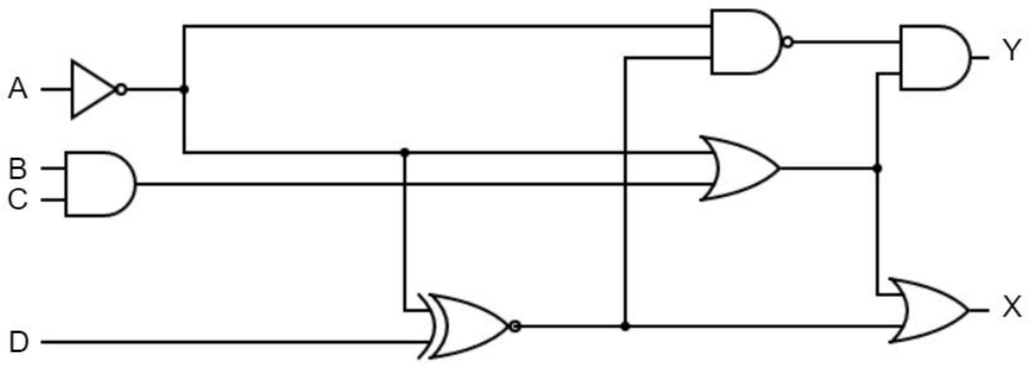
Lab_assignment1

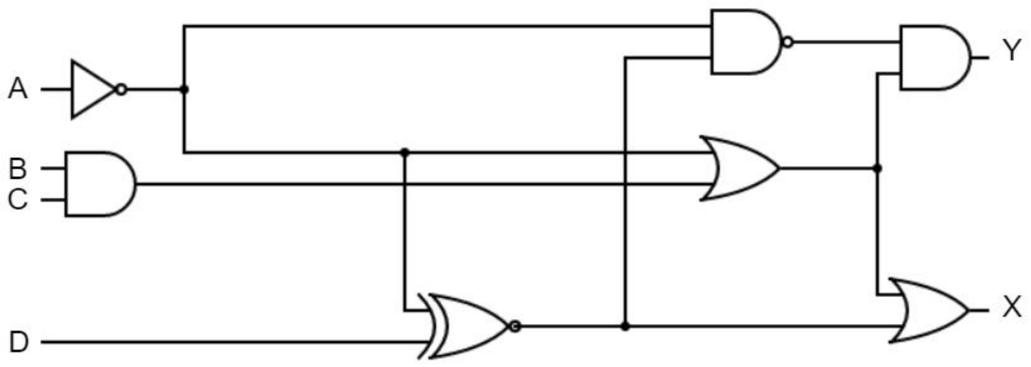
Digital_Design_2023_fallterm@cse.sustech

P1. Use **the logic gates in Verilog** to describe the following circuit in a **structured manner**: the bit width of the four input ports **A**, **B**, **C**, and **D**, as well as the two output ports **X** and **Y**, are all 1 bit. The module name is **lab_a1_p1**. (Please note that **keywords such as “assign” and “always” cannot appear in the submitted code**) (10points)

TIPS：  is a xnor gate.  is a nand gate.  is a not gate.



P2. Use **the data flow method in Verilog** to describe the following circuit. The bit widths of the four input ports A, B, C, and D, as well as the two output ports X and Y, are all 1 bit. The module name is **lab_a1_p2**. (Please note that logic gates **not**, **and**, **or**, **nor**, **nand**, **xnor**, etc. cannot appear in the submitted code) (10points)



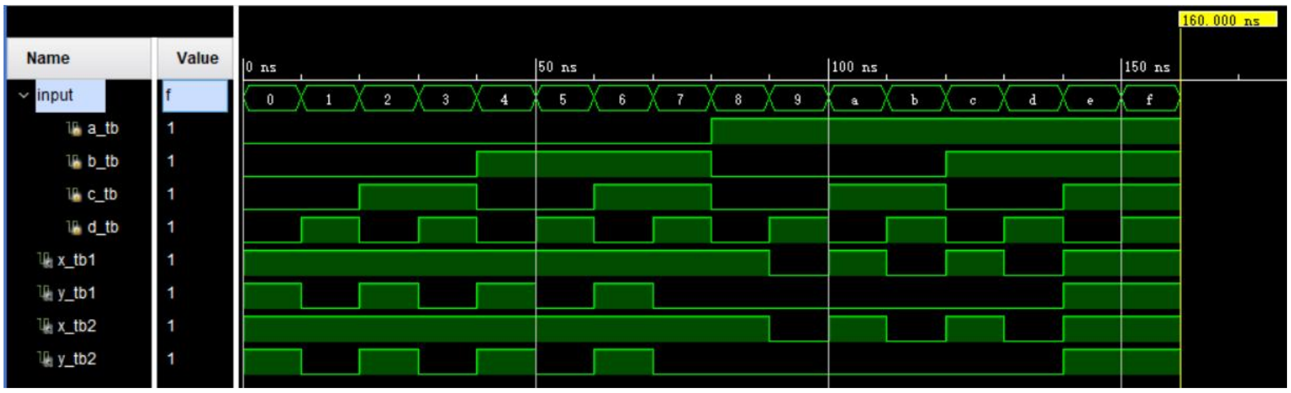
P3. **Build** a testbench to simulate and test the reference circuits, and the modules being tested is named **lab_a1_p1**, **lab_a1_p2**. (10points)

- Both lab_a1_p1 and lab_a1_p2 have four input ports A, B, C, D, and two output ports X and Y. The bit widths of A, B, C, D, X and Y, are all 1 bit.
- **a**_tb in the testbench connects to port **A** of **lab_a1_p1** and **lab_a1_p2;**
- **b**_tb in the testbench connects to port **B** of **lab_a1_p1** and **lab_a1_p2;**
- **c**_tb in the testbench connects to port **C** of **lab_a1_p1** and **lab_a1_p2;**
- **d**_tb in the testbench connects to port **D** of **lab_a1_p1** and **lab_a1_p2.**
- **x_tb1** in the testbench connects to port **X** of **lab_a1_p1**;
- **x_tb2** in the testbench connects to port **X** of **lab_a1_p2;**
- **y_tb1** in the testbench connects to port **Y** of **lab_a1_p1**;
- **y_tb2** in the testbench connects to port **Y** of **lab_a1_p2;**

**NOTE:** The simulation time does not exceed 160ns. While submit your code to Verilog-OJ. Please use the **monitor** statement once in `initial` block (there is only one `initial` block). For example:

```
Initial begin    // just for reference
$monitor ("%d %d %d %d %d %d %d %d", a_tb, b_tb, c_tb, d_tb, x_tb1, x_tb2, y_tb1, y_tb2);
 {a_tb, b_tb, c_tb, d_tb} = 4'b0000;
repeat(15)  #10 {a_tb, b_tb, c_tb, d_tb} = {a_tb, b_tb, c_tb, d_tb} + 1;
#10 $finish;
end
```

**Tips: while run the simulation on the testbench, the expected waveform is like: ( the value of {a_tb,b_tb,c_tb,d_tb} changes from 4'b0000 to 4'b1111 with a step value of 1 for growth)**



P4. Use the **logic gates** in Verilog to describe the following conversion circuit in a structured manner. The module name of this circuit is **xb2yg**, the bit width of both its input port **xb** and its output port **yg** are all 4. The circuit converts **xb** into **yg** according to the following table and method.

(Please note that **keywords such as "assign" and "always" cannot appear in the submitted code**) (10points)

| **Group1 Input: xb** | Groupt1 Output: yg | **Groupt2 Input: xb** | Groupt2 Output: yg | **Groupt3 Input: xb** | Groupt3 Output: yg | **Groupt4 Input: xb** | Groupt4 Output: yg |
|---|---|---|---|---|---|---|---|
| **0000** | 0000 | **0001** | 0001 | **0010** | 0011 | **0011** | 0010 |
| **0100** | 0110 | **0101** | 0111 | **0110** | 0101 | **0111** | 0100 |
| **1000** | 1100 | **1001** | 1101 | **1010** | 1111 | **1011** | 1110 |
| **1100** | 1010 | **1101** | 1011 | **1110** | 1001 | **1111** | 1000 |

The conversion method is:

1. The **highest bit** of output **yg** is the same as the highest bit of input **xb**; (Tips: Gate buf could keep output and input the same).
2. The calculation method for **other bits of output yg** is as follows:

**yg [i]=xb [i] ^ xb [i+1]**    here "^" is **XOR** processing.

For example:

**xb** is 4'b 0100: xb[3] = 1'b0,  xb[2] = 1'b1,  xb[1]= 1'b0,  xb[0] = 1'b0;

yg[3] = xb[3] = 1'b0;    yg[2] = xb[2] ^ xb[3] = 1'b1 ^ 1'b0 = 1'b1;

yg[1] = xb[1] ^ xb[2] = 1'b0 ^ 1'b1=1'b1;    yg[0] = xb[0] ^ xb[1] = 1'b0 ^ 1'b0 = 1'b0;

so, while **xb** is 4'b0100, **yg** is 4'b0110.

P5. Use the **data flow** method in Verilog to describe the following conversion circuit. The module name of this circuit is **yg2xb**, the bit width of both its input port **yg** and its output port **xb** are all 4. The circuit converts **yg** into **xb** according to the following table and method.

(Please note that logic gates **not**, **and**, **or**, **nor**, **nand**, **xnor**, etc. cannot appear in the submitted code) (10points)

| **Groupt1 input: yg** | Group1 output: xb | **Groupt2 input: yg** | Groupt2 output: xb | **Groupt3 input: yg** | Groupt3 Output: xb | **Groupt4 input: yg** | Groupt4 Output: xb |
|---|---|---|---|---|---|---|---|
| **0000** | 0000 | **0001** | 0001 | **0011** | 0010 | **0010** | 0011 |
| **0110** | 0100 | **0111** | 0101 | **0101** | 0110 | **0100** | 0111 |
| **1100** | 1000 | **1101** | 1001 | **1111** | 1010 | **1110** | 1011 |
| **1010** | 1100 | **1011** | 1101 | **1001** | 1110 | **1000** | 1111 |

The conversion method is：

1. The highest bit of output **xb** is the same as the highest bit of input **yg**;
2. The calculation method for **other bits of output xb** is as follows:

**xb [i]=yg [i] ^ xb [i+1]**  here “^” is **XOR** processing

For example:

yg is 4’b 0110: yg[3] = 1’b0, yg[2] = 1’b1, yg[1]= 1’b1, yg[0] = 1’b0;

xb[3] = yg[3] = 1’b0; xb[2] = yg[2] ^ xb[3] = 1’b1 ^ 1’b0 = 1’b1;

xb[1] = yg[1] ^ xb[2] = 1’b1 ^ 1’b1=1’b0; xb[0] = yg[0] ^ xb[1] = 1’b0 ^ 1’b0 = 1’b0;

so, while **yg** is 4’b0110, **xb** is 4’b0100.

P6. Build a testbench to simulate and test the reference circuits, and the modules being tested is named **xb2yg**, **yg2xb**. (10points)

- The bitwidth of both **xb2yg**’s input port **xb** and its output port **yg** are all 4.
- The bitwidth of both **yg2xb**’s input port **yg** and its output port **xb** are all 4.
- **in**_tb in the testbench connects to port **xb** of **xb2yg** and port **yg** of **yg2xb;**
- **out_xb2yg** in the testbench connects to port **yg** of **xb2yg**;
- **out_yg2xb** in the testbench connects to port **xb** of **yg2xb;**

NOTE: The simulation time does not exceed 160ns. While submit your code to Verilog-OJ, please use the monitor statement once in `initial` block (there is only one `initial` block). For example

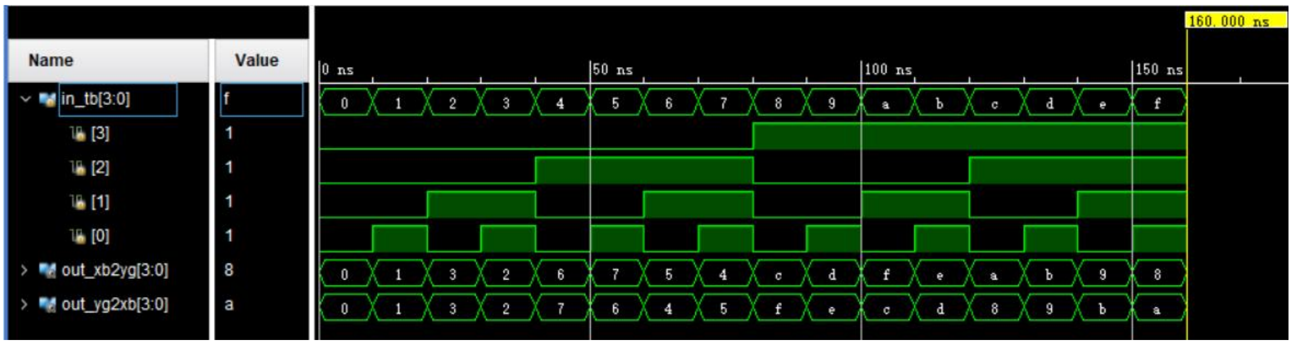
**Initial begin**

**$monitor ("% d % d % d%", in**_tb, **out_xb2yg, out_yg2xb)；**

// your code here

**end**

**Tips: while run the simulation on the testbench, the expected waveform is like: ( the value of in_tb changes from 4’b0000 to 4’b1111 with a step value of 1 for growth)**



P7. Design a circuit to get the addition of two two-bit unsigned numbers. The inputs are a and b which are both 2 bits, the output is sum which is 3 bits. (10points)

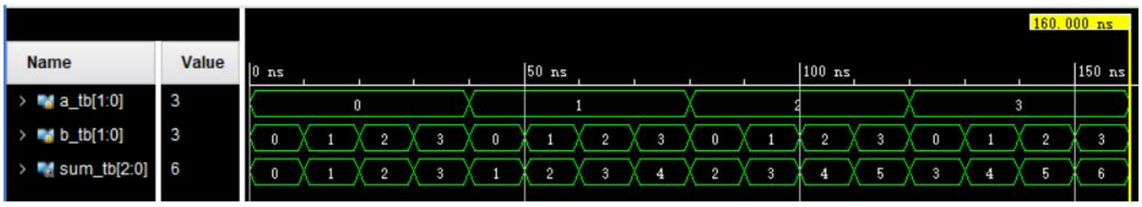
The truth table of the circuit is:

| a[1] | a[0] | b[1] | b[0] | sum[2] | sum[1] | sum[0] |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 |

**NOTE: In the design, the operator + in verilog in NOT allowed here.**

A testbench is built to verify the function of the circuit lab3_practic_add2bit: a_tb is a reg which connects to port a, b_tb is a reg which connects to port b, sum_tb is a wire which connects to port sum.

The expected waveform of the testbench which verify the function of the circuit lab3_practic_add2bit is as following picture:



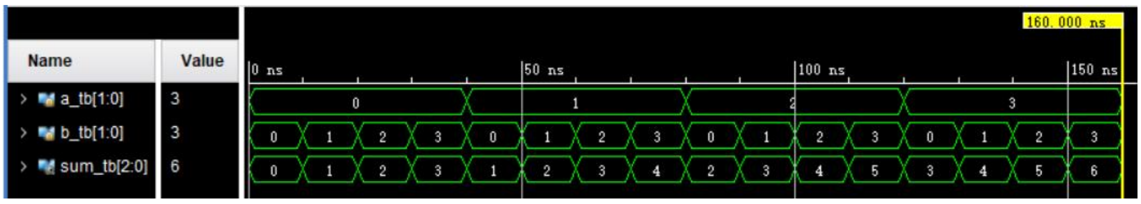
P8. Build a testbench to test the circuit **lab3_practic_add2bit**. (10points)

The function of the circuit **lab3_practic_add2bit** is to get the addition of two two-bit unsigned numbers. It's inputs are a and b which are both 2 bits, it's output is "sum" which is 3 bits:

It is asked to build a **testbench** to verify the function of the circuit **lab3_practic_add2bit**:

- **a_tb** in the testbench connects to port **a** of lab3_practic_add2bit
- **b_tb** in the testbench connects to port **b** of lab3_practic_add2bit
- **sum_tb** in the testbench connects to port **sum** of lab3_practic_add2bit

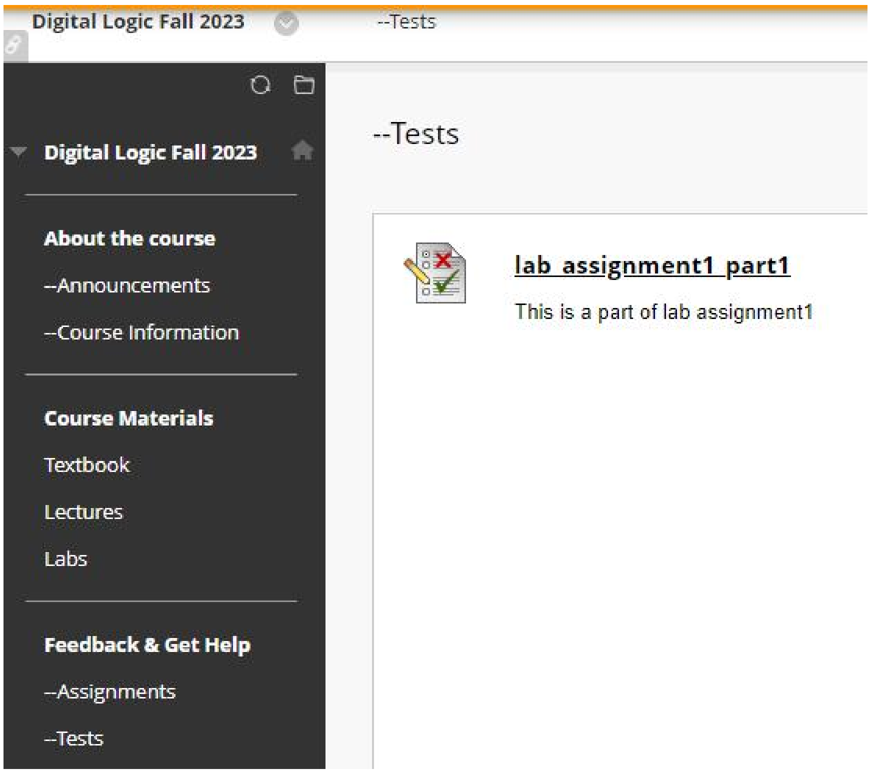
The expected waveform of the testbench which verify the function of the circuit lab3_practic_add2bit is as following picture:



NOTE: The simulation time does not exceed 160ns. While submit your code to Verilog-OJ, please use the monitor statement once in initial block (there is only one initial block). For example

```
Initial begin
$monitor("%d %d %d", a_tb, b_tb, sum_tb);
// your code here
end
```

P9: Please finish the test "lab_assignment1_part1" in the course sit on BlackBoard. (20points)
"Tests" –》"lab_assignment1_part1"

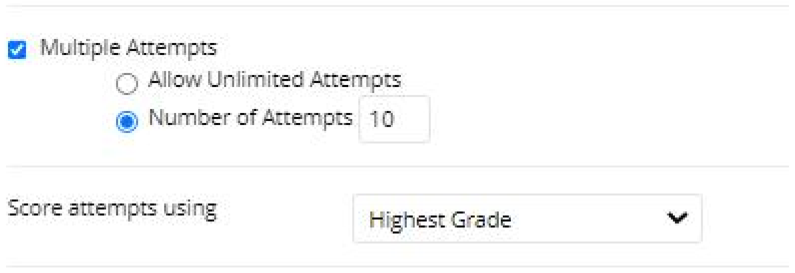


--Tests

lab_assignment1_part1

This is a part of lab assignment1

There are 10 attempts for your submission, the blackboard would record the Highest grade.



[✓] Multiple Attempts
( ) Allow Unlimited Attempts
(●) Number of Attempts: 10

Score attempts using: Highest Grade