**Logo

Description automatically generated San Francisco Bay University**

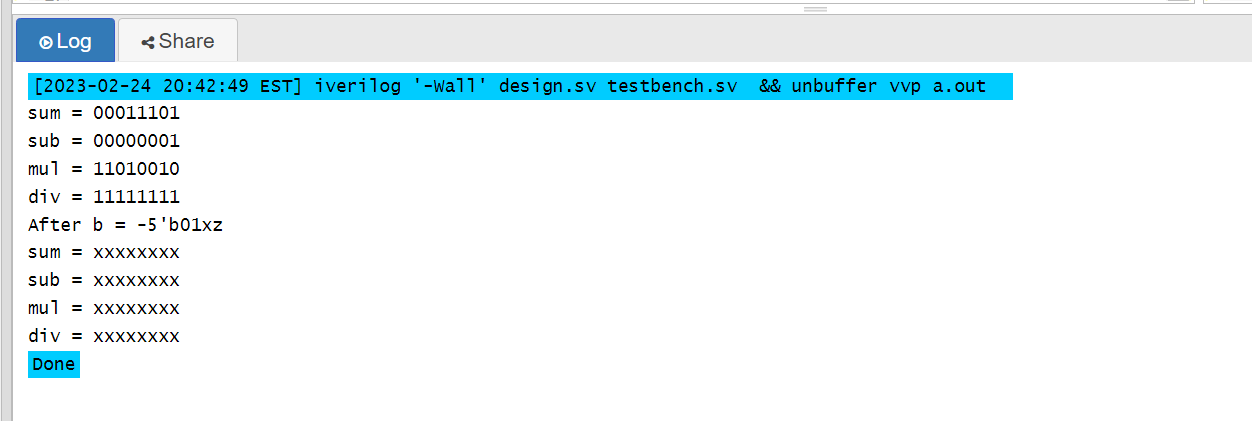
**EE461 Verilog-HDL**

**Homework #2**

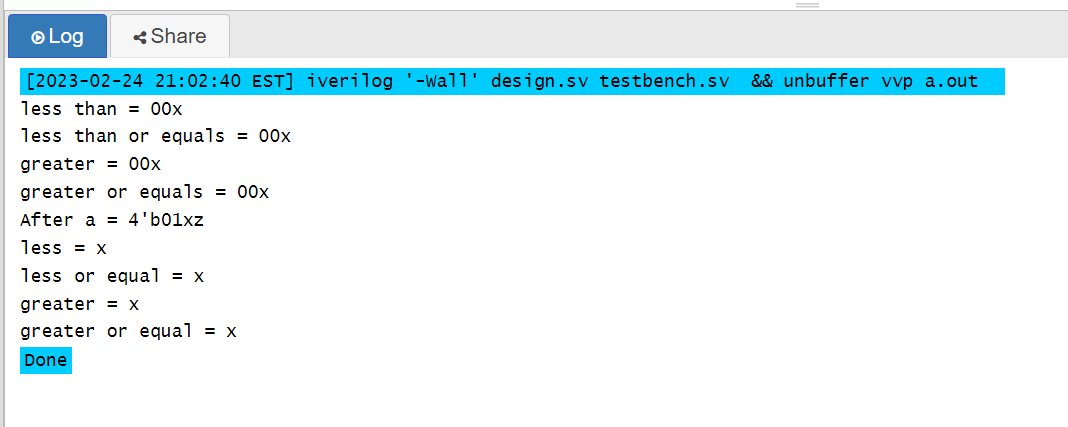
**Due day: 2/25/2023**

**Instruction:**

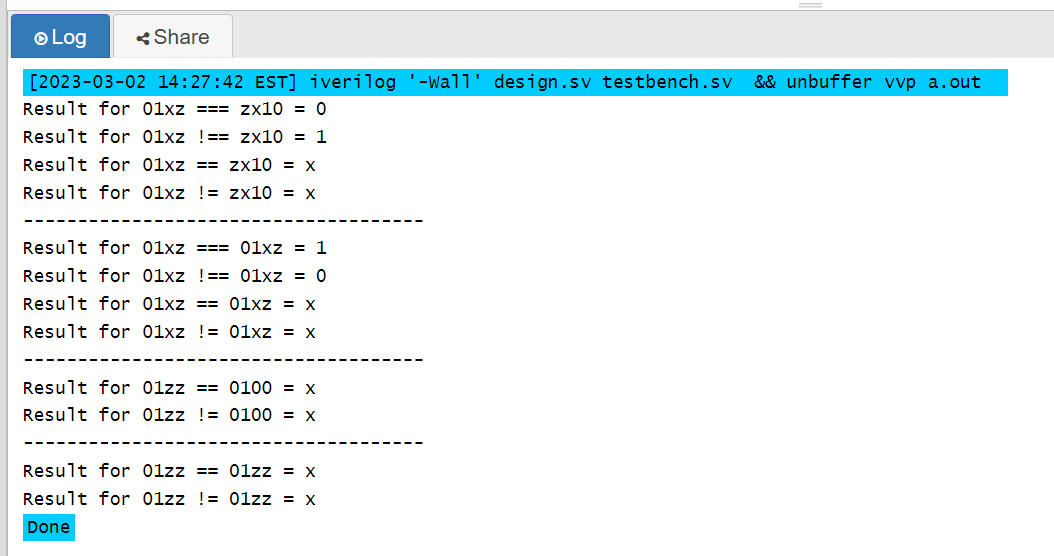
1. **Push answer sheets/source code to Github**
2. **Please follow the code style rule like programs on handout.**
3. **Overdue homework submission could not be accepted.**
4. **Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**
5. If a = 4’b1111 and b = -5’b00010, write the program to check what the values with 4 bits are when you want to calculate “a (+/-/\*/%) b”. How about b = -5’b01xz?



1. When a = 2’b1z and b = 3’b11z, verify the values for (a>b), (a>=b), (a<b) and (a<=b) by a program. If a = 4’b01xz, check again.



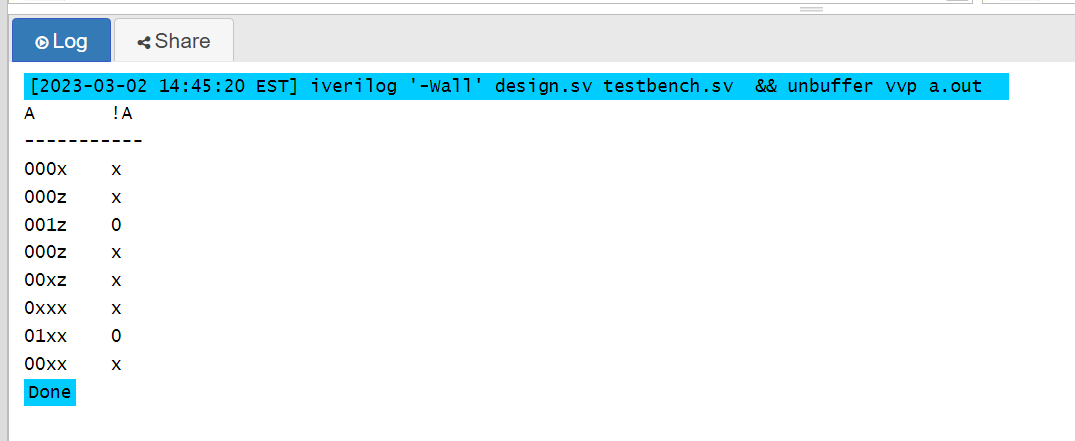
1. Write a program to see results for 4 questions on “Equality Operators” page in the handout.



Comment: in the hand out the result for 01zz == 0100 is zero but from the eda compiler the result is x.

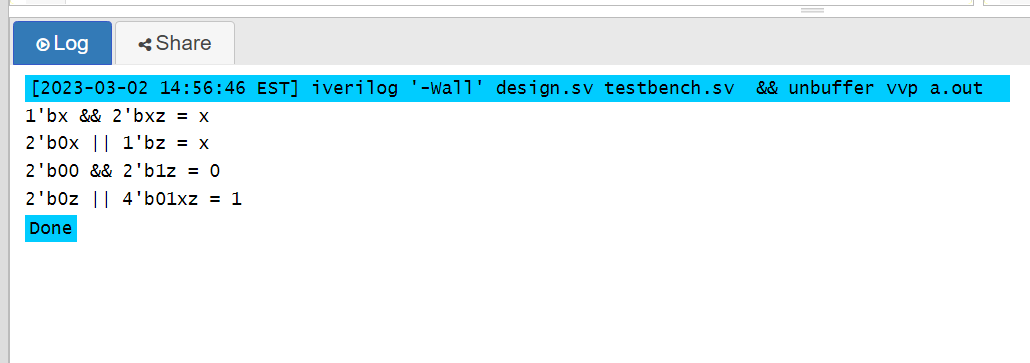
1. Verify the results by a program for the following “A’s values”.

|  |  |
| --- | --- |
| A | !A |
| 1’bx |  |
| 1’bz |  |
| 2’b1z |  |
| 2’b0z |  |
| 2’bxz |  |
| 3’bxxx |  |
| 3’b1xx |  |
| 3’b0xx |  |



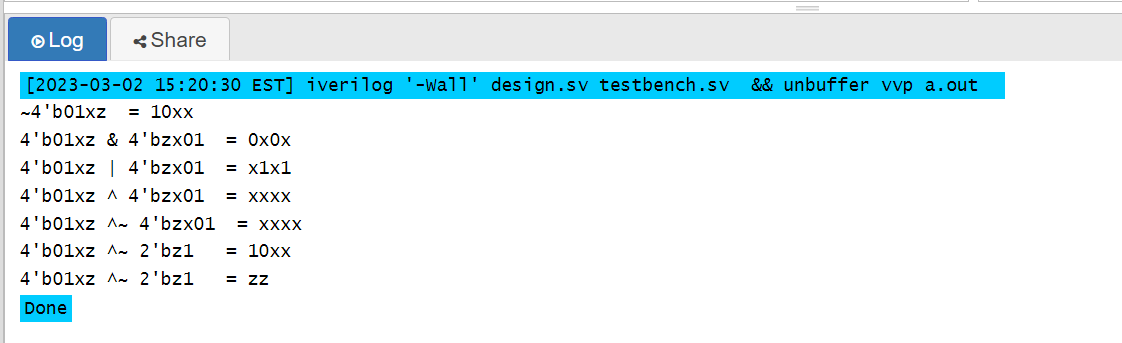
1. Write a program to see what you will get for 1’bx && 2’bxz , 2’b0x || 1’bz ,

2’b00 && 2’b1z and 2’b0z || 4’b01xz.



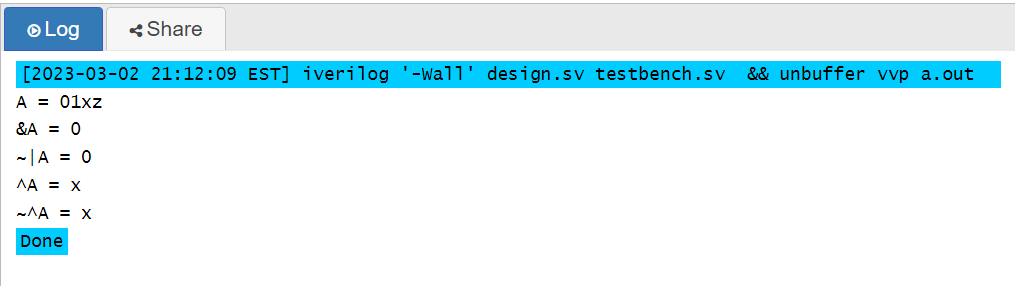
1. What are the results in the following operations and verify them by Verilog code?

|  |
| --- |
| ~4’b01xz = ? |
| 4'b01xz & 4'bzx01 = ? |
| 4'b01xz | 4'bzx01 = ? |
| 4'b01xz ^ 4'bzx01 = ? |
| 4'b01xz ^~ 4'bzx01 = ? |
| 4'b01xz ^~ 2'bz1 = 4’b? |
| 4'b01xz ^~ 2'bz1 = 2’b? |

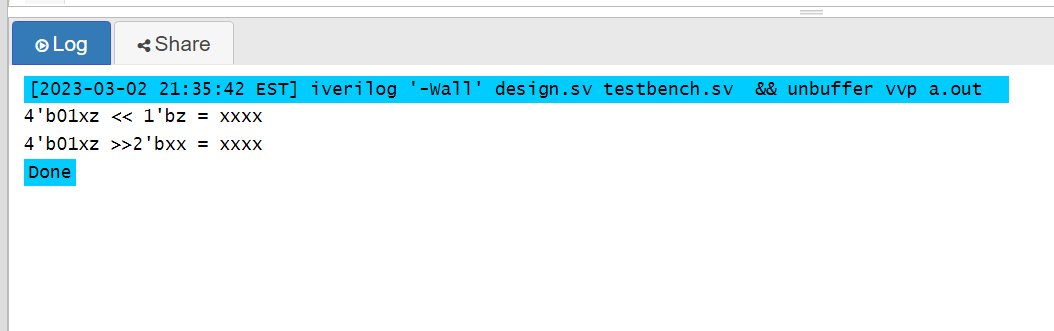


1. What are you going to get for “& 4’b01xz” , “~| 4’b01xz” , “^ 4’b01xz” and

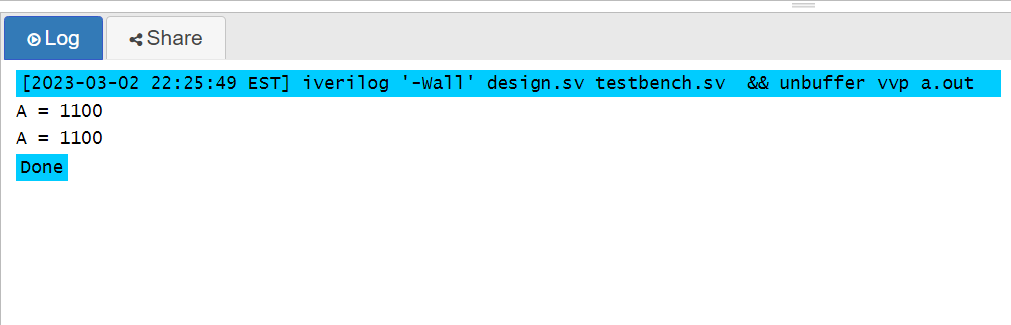
“~^ 4’b01xz”.



1. What are the new values after bit shifting for “4'b01xz << 1'bz” and “4'b01xz >>2'bxx” ?



1. In this expression A = B ? 4’b1100 : 5’b11ZX0 and if B = 2’b1x, What is A(4-bit number)? How about B= 3’b1xz? Write a program to verify your answers.



1. Complete the following Verilog modules and display the output strength. Explain why.

*module testStrength1();*

*... ... // Data type declaration for a, b and y*

*buf (strong1, weak0) g1 (y, a)*

*buf (weak1, strong0) g2 (y, b);*

*initial begin*

*a = 1;*

*b = 1;*

*$display(“y = ..., a =... , b = ...”, y, a, b);*

*end*

*endmodule*

*module testStrength2();*

*... ... // Data type declaration for a, b and y*

*bufif0 (strong1, weak0) g1 (y, i1, ctrl);*

*bufif0 (strong1, weak0) g2 (y, i2, ctrl);*

*initial begin*

*ctrl = x;*

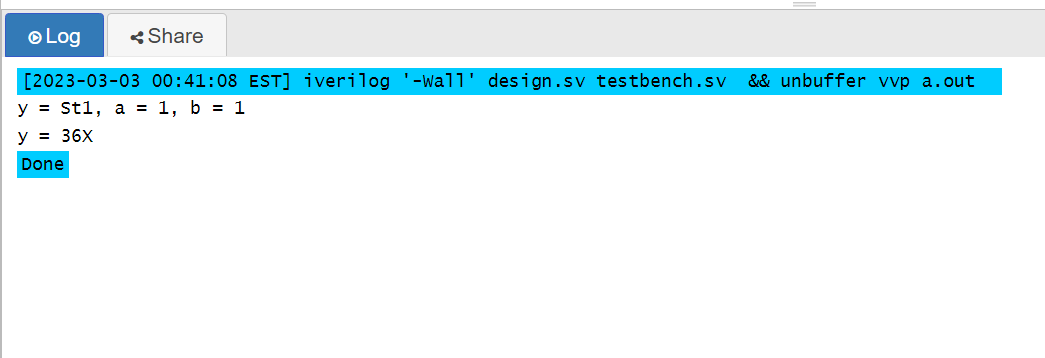
*i1 = 0;*

*i1 = 1;*

*$display(“y = ...”);*

*end*

*endmodule*

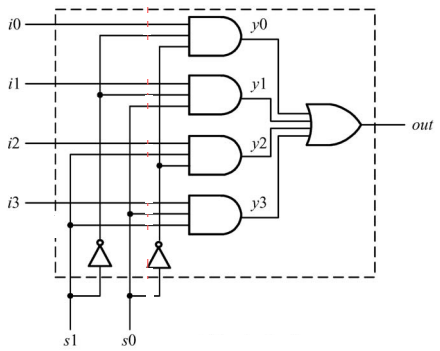
**

*Comment:*

*first module y will be 1 with strong strength because they have the same value 1 and strong has stronger signal strength than weak.*

*The second module: g1 will set y to L with weak strength and g2 will set y to H with strong strength then y will be x with 36X strength.*

1. Design Verilog program for 4 to 1 mux in gate level and write the testbench to verify.



*module fourOneMux(*

*i0\_i,*

*i1\_i,*

*i2\_i,*

*i3\_i,*

*s0\_i,*

*s1\_i,*

*out\_o*

*);*

*... ... ;*

*endmodule*

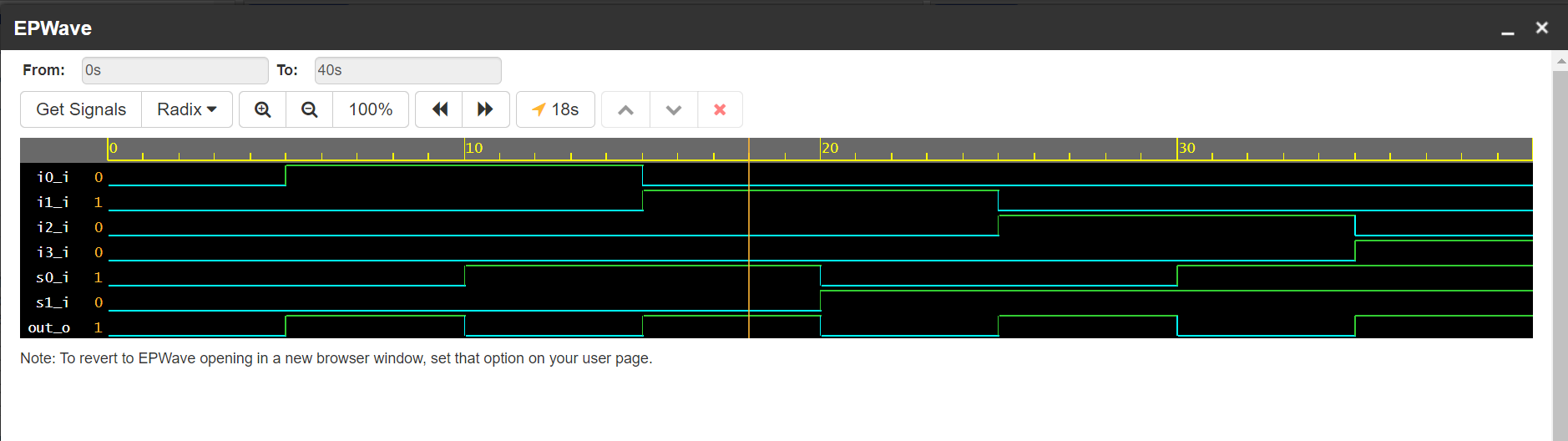
*`include “foutOneMux”*

*module fourOneMuxTB;*

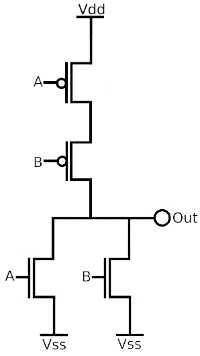
*... ... ;*

*endmodule*

Note: please save two modules in two different files with the same name as module under the one directory.



1. Write nor gate Verilog module using switch devices and testbench to verify it.



Result

