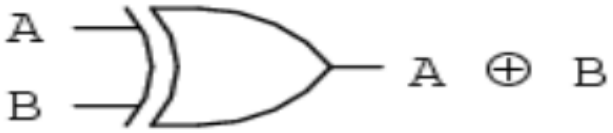
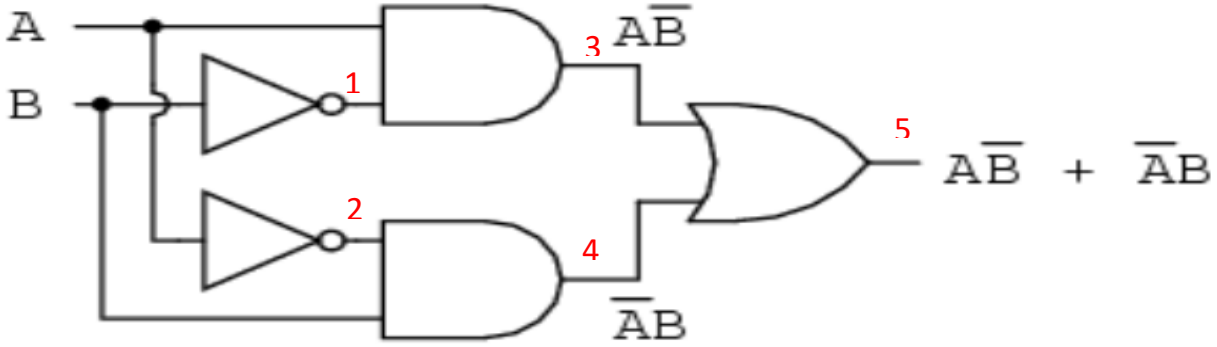
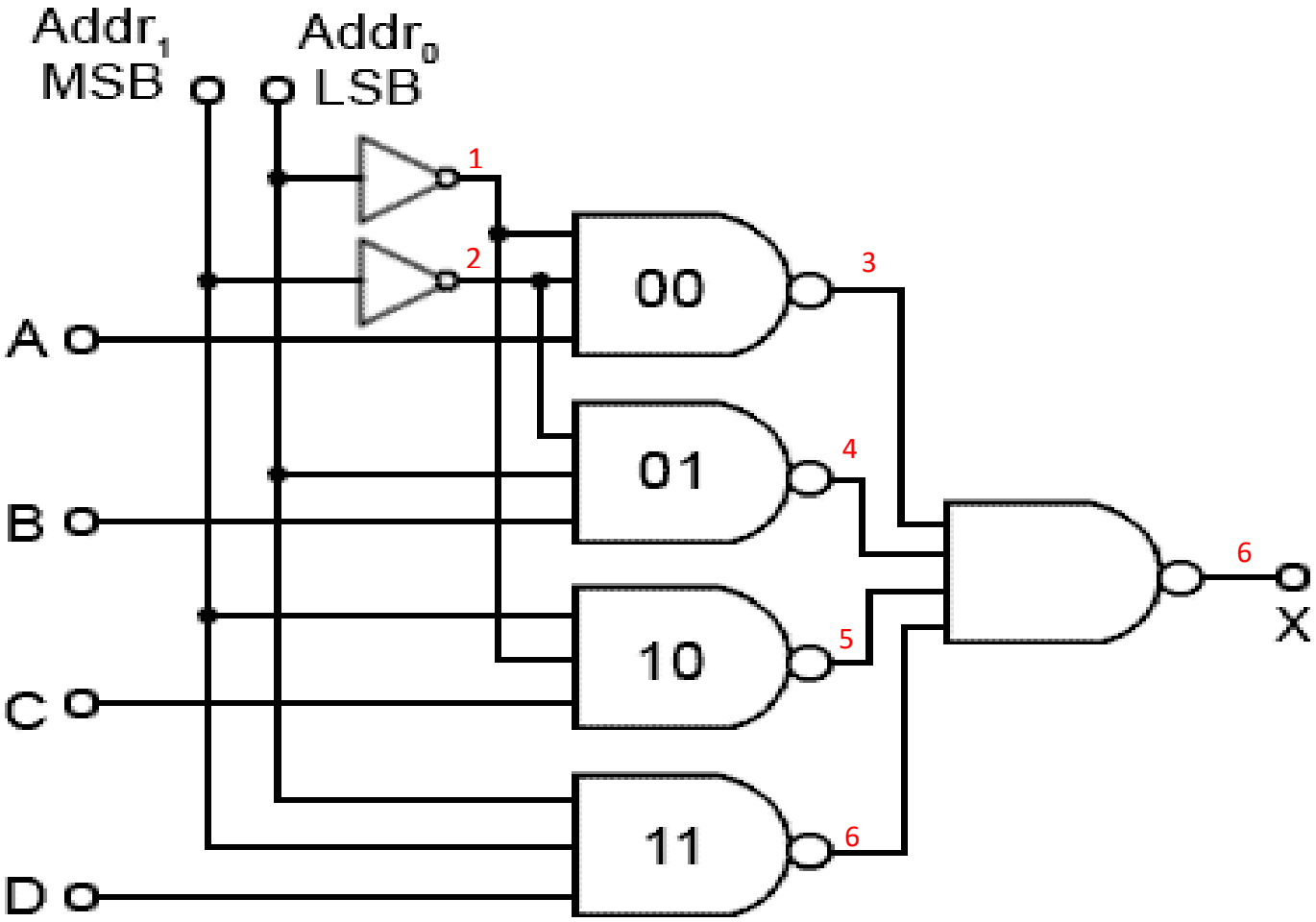
_XOR



. . . is equivalent to . . .

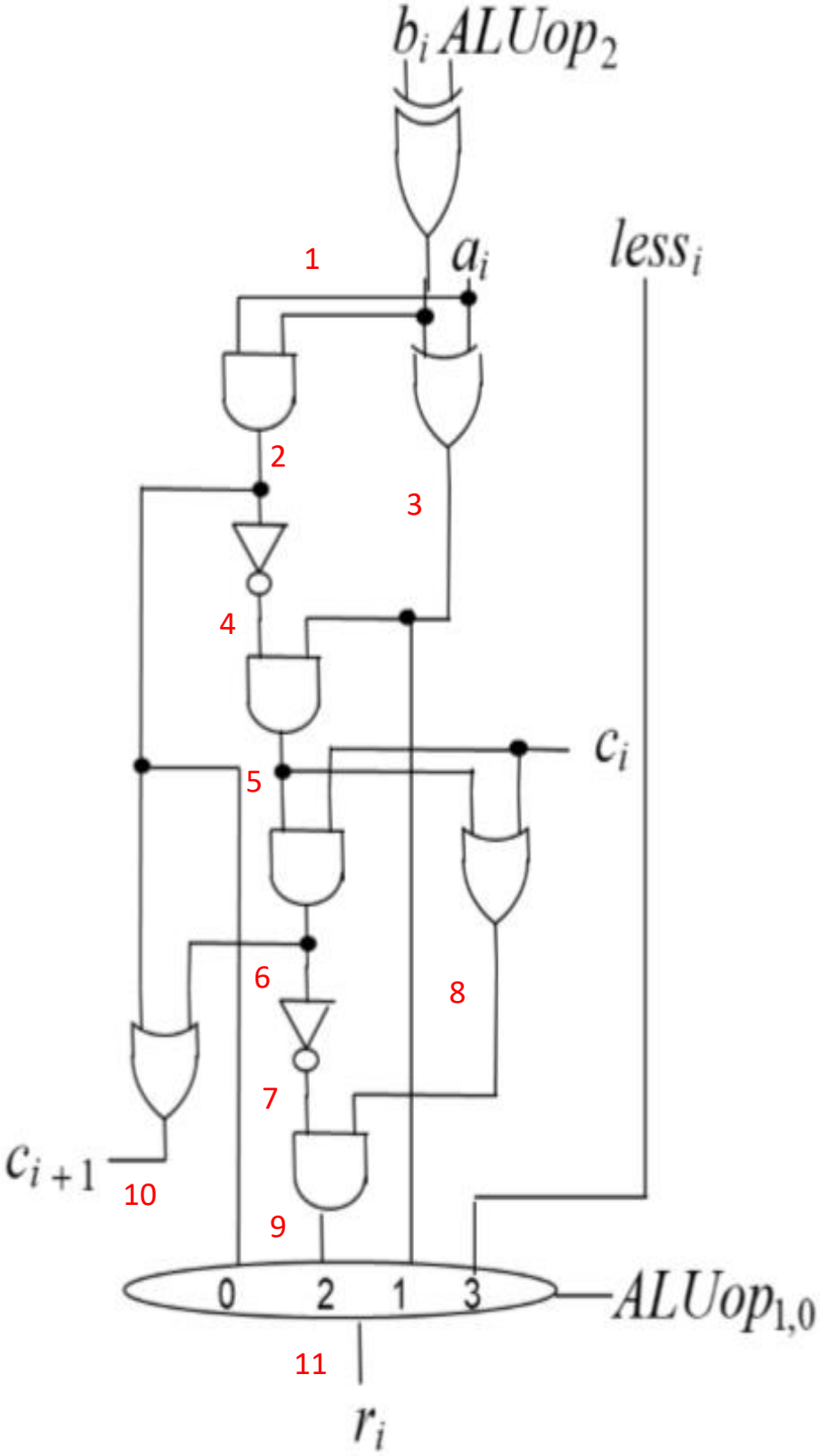


mux4_1

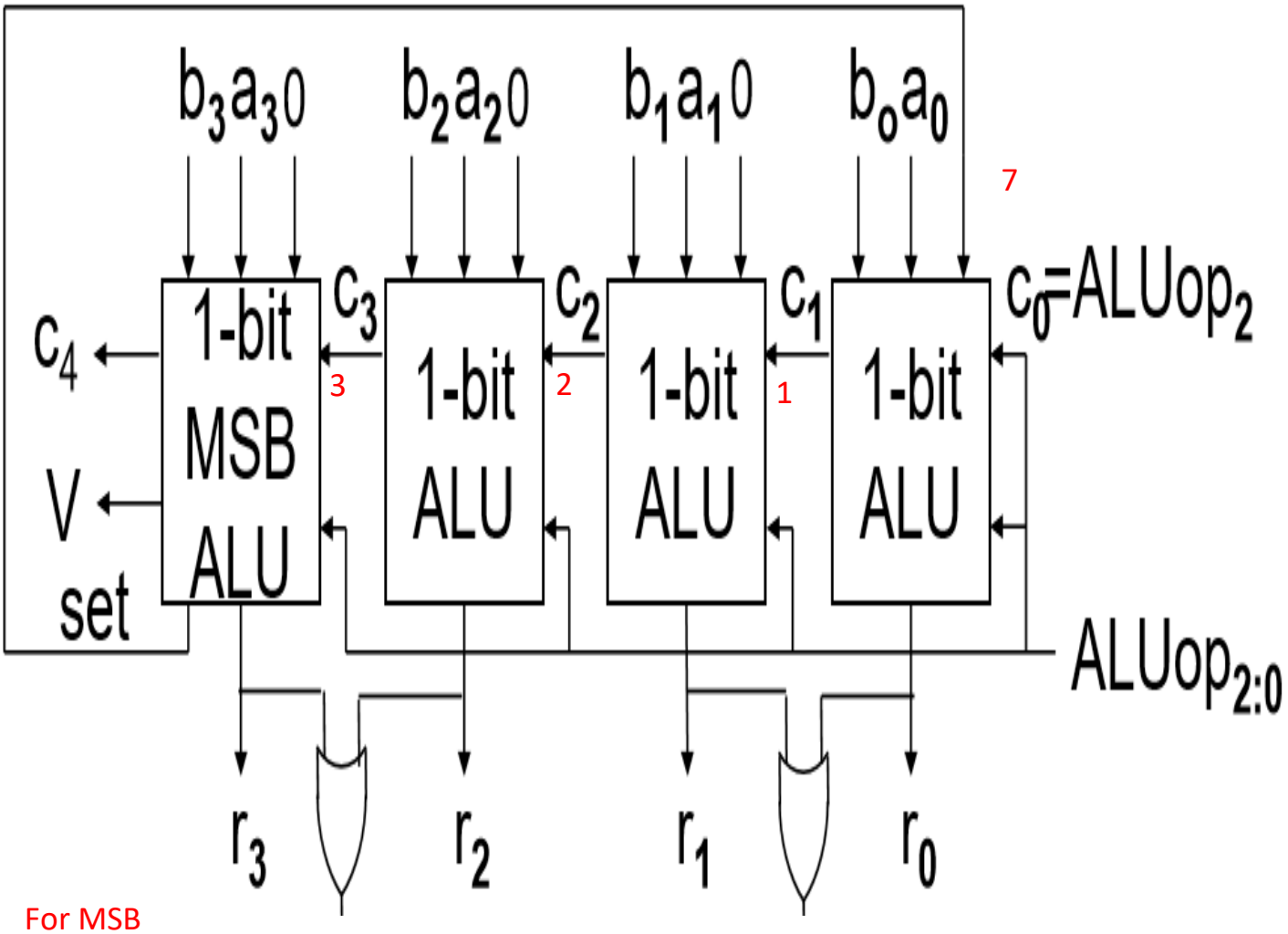


ALU1bit

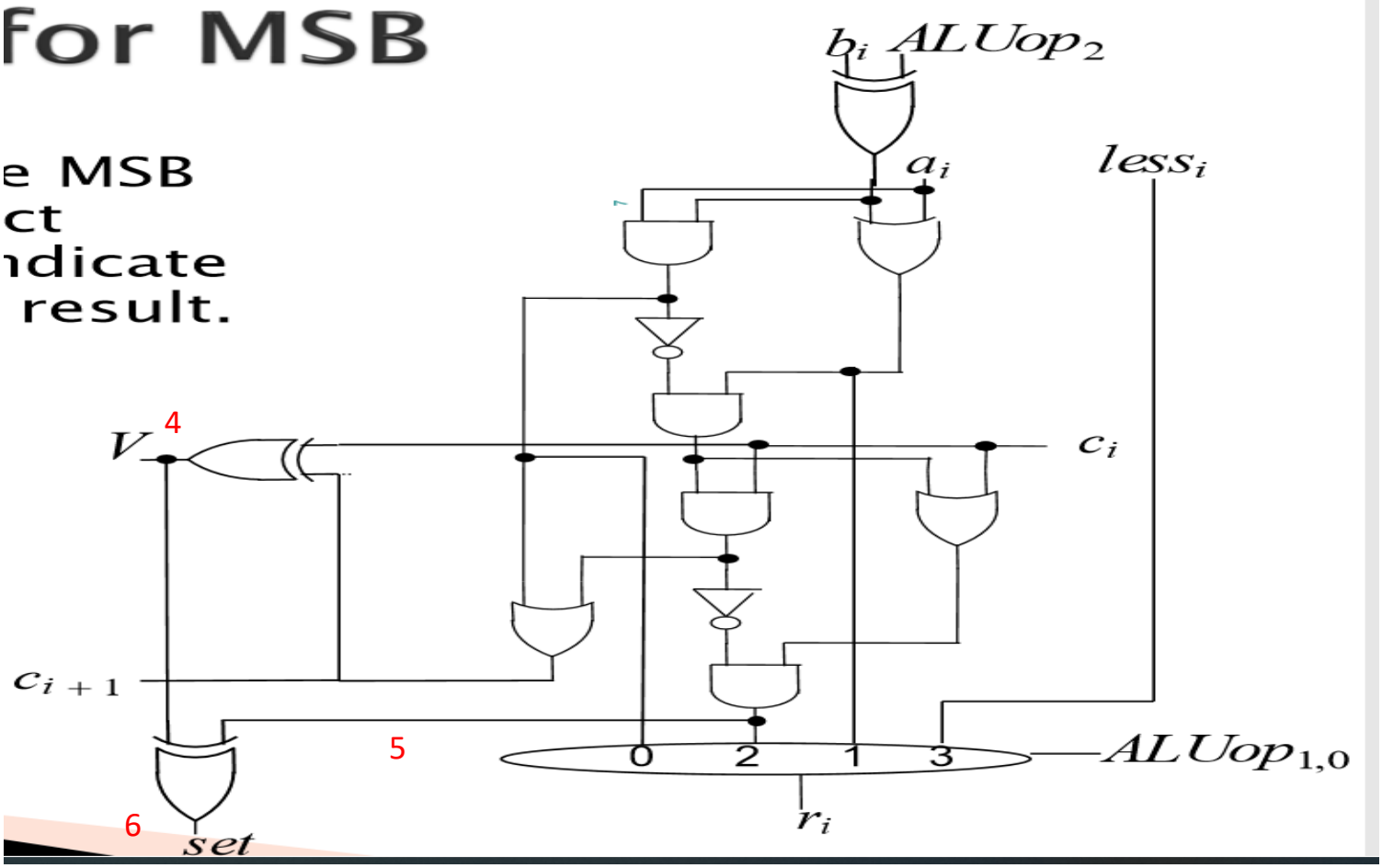
1-bit ALU module will be designed same as shown below:



ALU32bit



For MSB



Each number represent a comment in verilog code, you can see which number operation happens which line .

For example :

```
module _xor( result, input1, input2);

input input1, input2 ;
output result ;
wire notInput1 , notInput2, result_input1, result_input2 ;

not not_input1( notInput1, input1 ) ; // 1
not not_input2( notInput2, input2 ) ; // 2

and before_result1( result_input1, notInput1, input2 ) ; // 4
and before_result2( result_input2, notInput2, input1 ) ; // 3

or now_result( result, result_input1, result_input2);  // 5

endmodule
```

```
module mux4_1( result, S1, S0, I0, I1, I2, I3);

input I0, I1, I2, I3;
input S0, S1 ;

output result ;

wire not_S1, not_S0 ;
wire and_input0, and_input1, and_input2, and_input3 ;

not notS1( not_S1, S1 ) ; // 2
not notS0( not_S0, S0 ) ; // 1

and andInput0( and_input0, not_S1, not_S0, I3 ) ; // 3

and andInput1( and_input1, not_S1, S0, I2 ) ;  // 4

and andInput2( and_input2, S1, not_S0, I1 ) ; // 5

and andInput3( and_input3, S1, S0, I0 ) ;  // 6

or create_result( result, and_input0, and_input1, and_input2, and_input3 ); // 7

endmodule
```