

**Verilog \_HDL\_Report**

**Faculty of Engineering and Technology  
 Electrical and Computer Engineering Department**

**Digital Systems ENCS2340**

|  |  |
| --- | --- |
| **Name: shaimaa dar taha** | **ID number:1222440** |
| **Section :1** | **Date:25/1/2024** |

**Instructor:** **Ismail Khater**

**اللهم ارحم شهداءنا وانصرنا على القوم الظالمين**

**Project Description:**

**In this project, I will design a simple Arithmetic Logic Unit (ALU) using Verilog Hardware Description Language (HDL). The ALU capable of performing four basic arithmetic and logic operations: addition, subtraction, bitwise AND, and bitwise OR.**

**The Alu take two 4-bit inputs (A and B).**

**Include a 3-bit control input (OpCode) to select the operation:**

**• 3'b000: Addition**

**• 3'b001: Subtraction**

**• 3'b010: Bitwise AND**

**• 3'b011: Bitwise OR**

**Contents:**

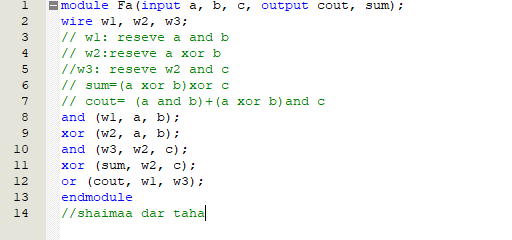
1. Full adder
2. 4\_bit\_Adder
3. 4\_bit \_Subtractor
4. Bitwise AND
5. Bitwise OR
6. Data flow ALU
7. Behavioral\_ALU

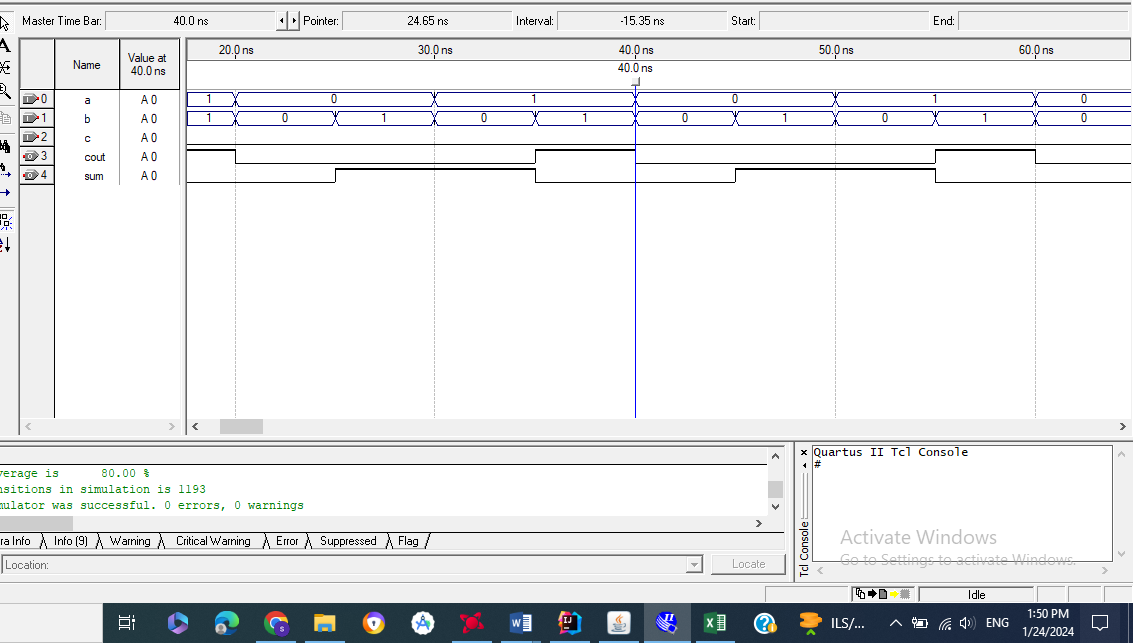
**Full adder :(Fa)**



A full adder is a fundamental building block in digital electronics used to perform binary addition of three input bits: two single-bit binary numbers (A and B) and a carry-in (C) from a previous stage of addition. It produces two outputs: the sum (Sum) of the three input bits and a carry-out (Cout) that represents whether there is a carry

**Verilog code**



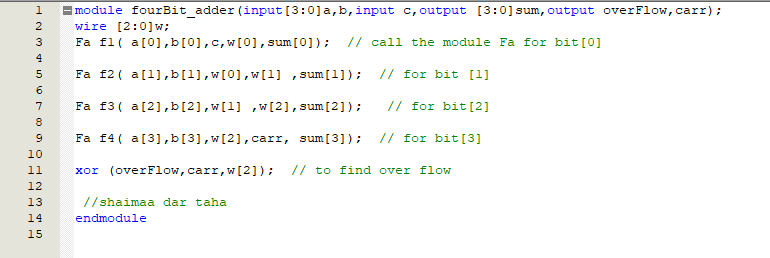
Simulation report:(a=0,b=1 ,sum =1,cout =0) so its true 

**4\_bit\_Adder :**

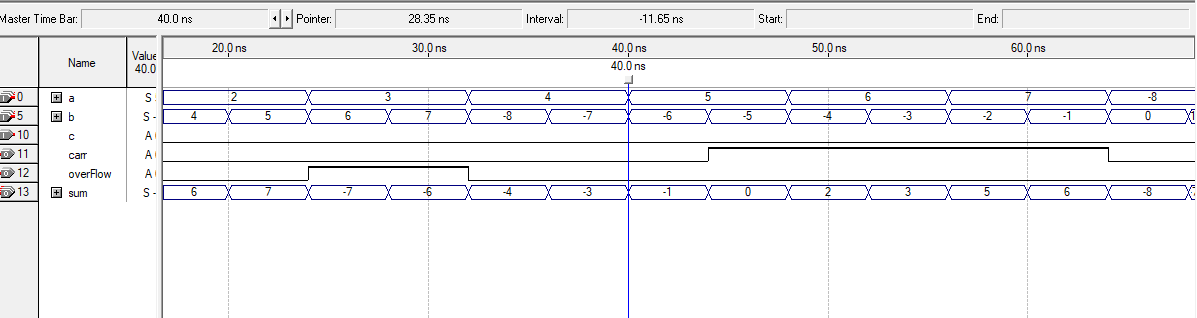


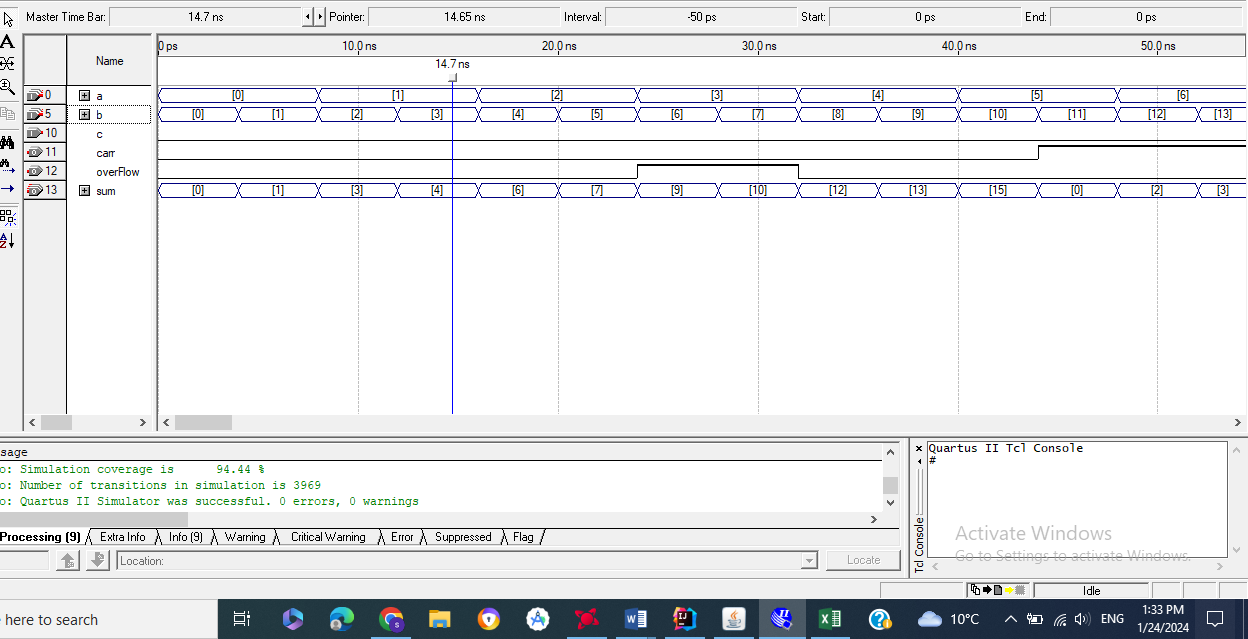
**can add two four-bit binary numbers together. It's constructed using four full adders and some additional logic to handle the carry between each stage.**

**Verilog code :**



**Simulation report: (signed number)**:(3+6= -7 but over flow=1 ) ,(2+5=7)so the adder work correct



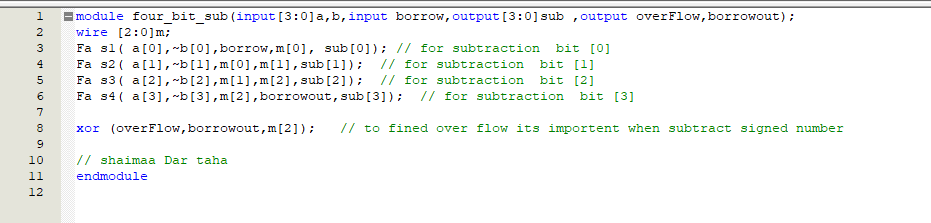
**ASCII code =the same result at unsigned number :(2+4=6,carry=0)**

**4\_bit \_Subtractor**

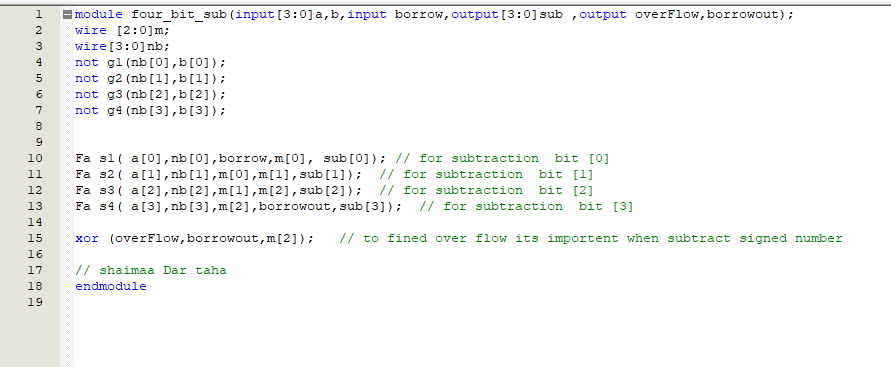


**four-bit subtractor is a digital circuit that computes the subtraction of two four-bit binary numbers. It's essentially a combination of adders I call the module (Fa)and inverters to perform the subtraction operation. One common method to implement a four-bit subtractor is by using a two's complement approach.**

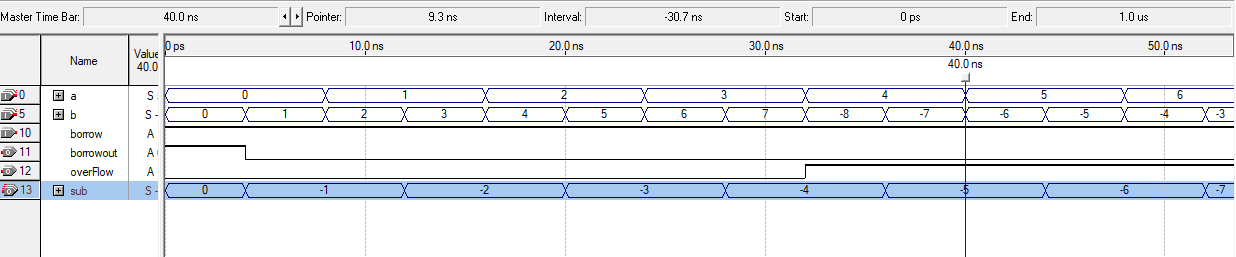
**Verilog code :**

****

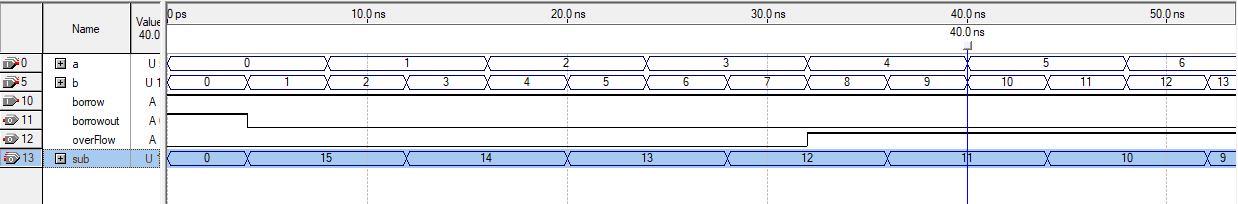
**Another code :**

****

**Simulation report: (signed number): (3-6=-3 ,over flow =0)so the module is correct**

****

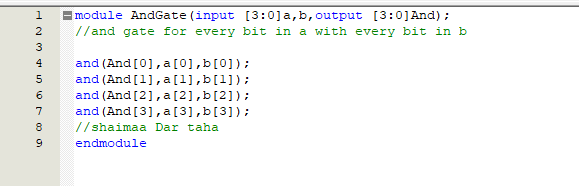
**Simulation report: (unsigned number):(5-10=11 but over flow is 1)**

****

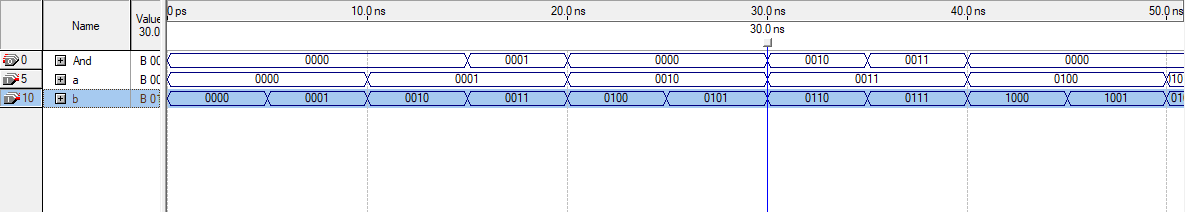
**Bitwise AND:**

****

**Verilog code : this module all 4 bit( a )anding with 4 bit (b)**

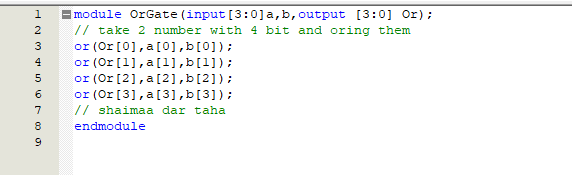
****

**Simulation report: (0000 and 0000 give 0000)so the simulation is true**

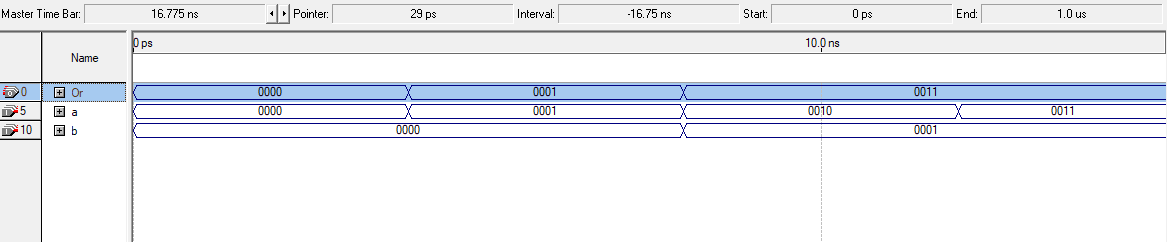
****

**Bitwise OR**: **Verilog code : this module all 4 bit( a ) oring with 4 bit (b)**

****



**Simulation report**:( 0000 or0001=0001) so the simulation is true

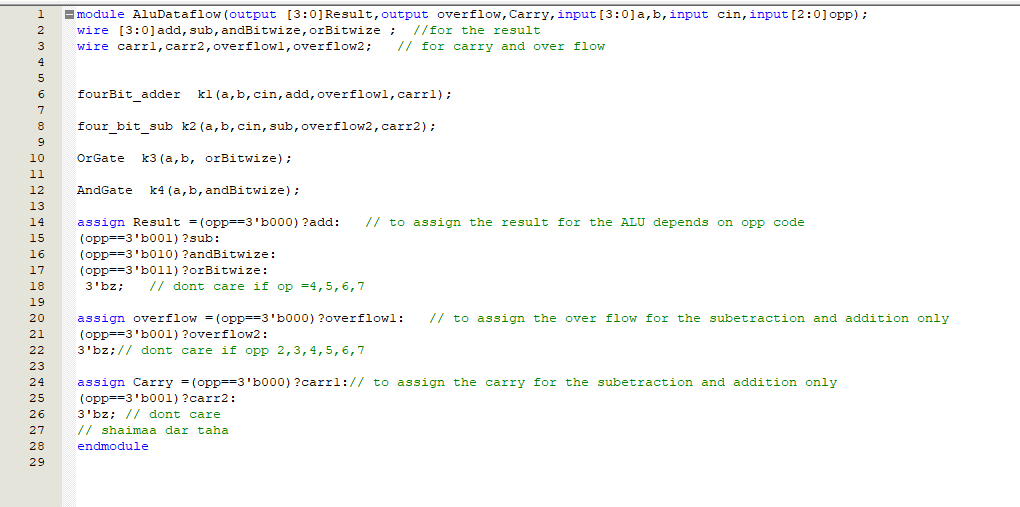


**Data flow ALU:**



**Module for ALU that capable of performing four basic arithmetic and logic operations: addition, subtraction, bitwise AND, and bitwise OR.in data flow**

**Verilog code:**

****

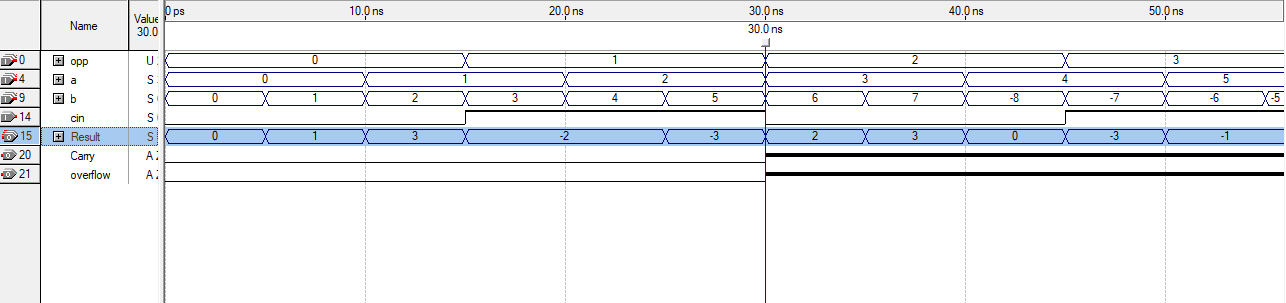
**Simulation report: (signed number )**

when opp=000 (add )(0+1=1) carry is 0 ,over flow is 0Its true

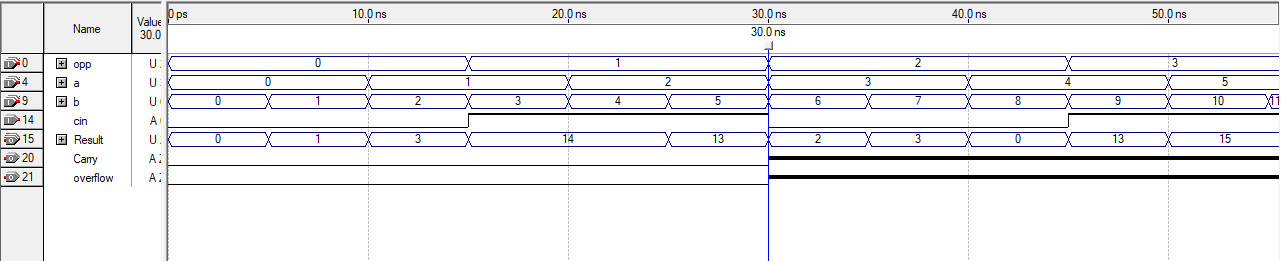
when opp=001(sub)(2-4= -2)so its true

When opp=010(and)(0011&0110=0010) its true

When opp =011(or )(0100|1001=1101=-3)so its true



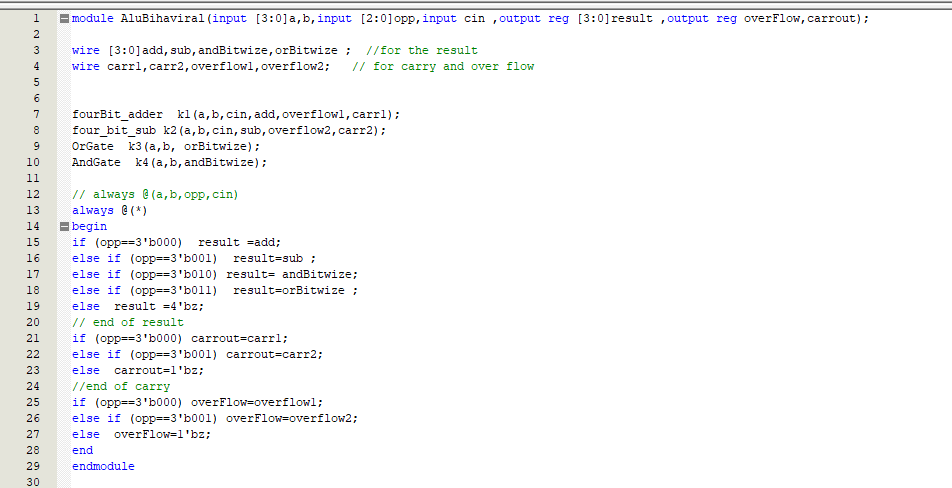
**Simulation report: (unsigned number ):**

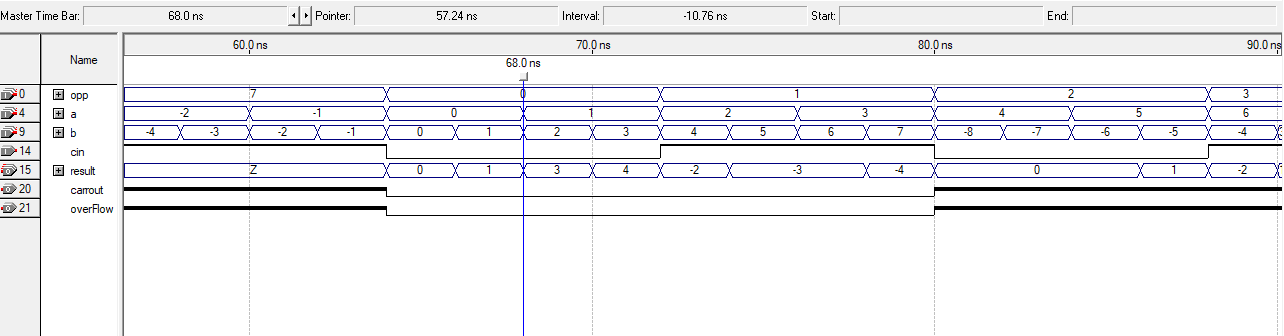


**Behavioral\_ALU:**

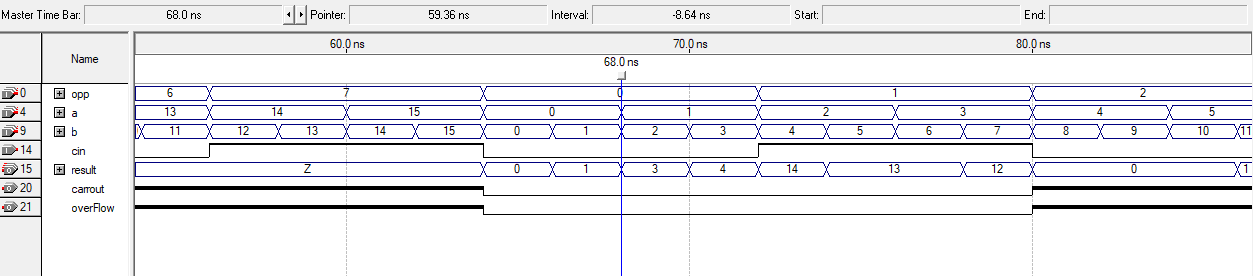
**Module for ALU that capable of performing four basic arithmetic and logic operations: addition, subtraction, bitwise AND, and bitwise OR.in behavioral**

**Verilog code:**

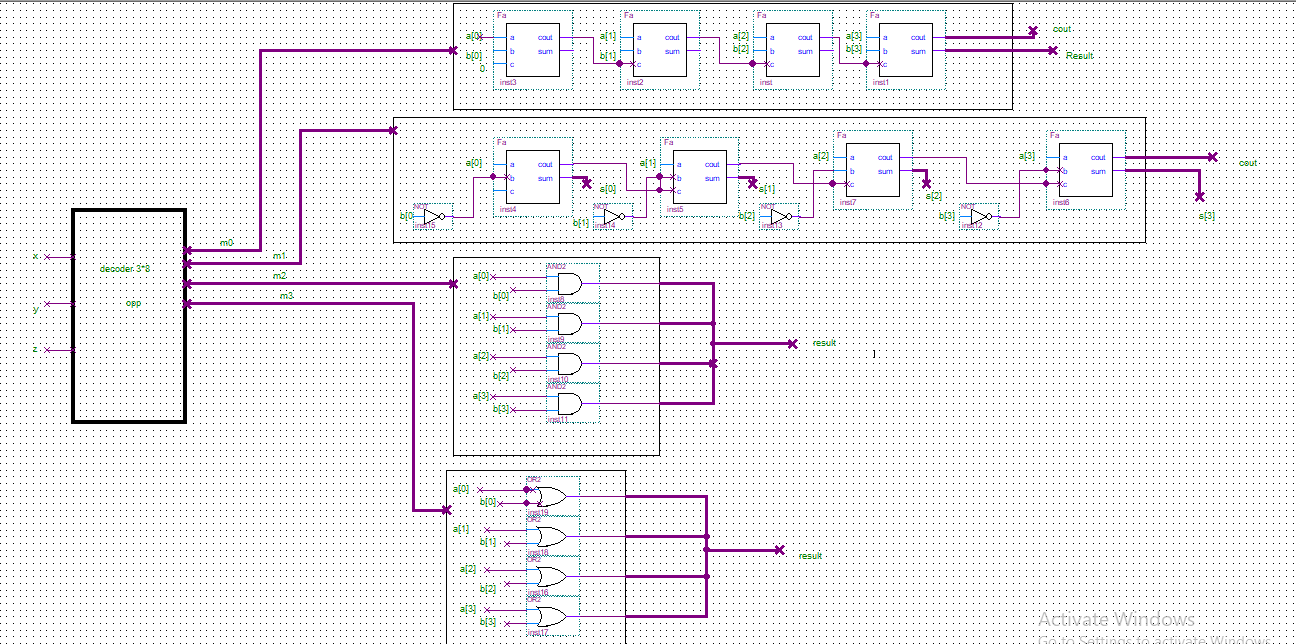
****

**Simulation report: (signe d number ) (true 3-6=-3 , 0+1=1)**

**Simulation report: (unsigned number ):**

****

**Sympole for project:**



**Another sympole:**

