

**计算机与信息 学院实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 实验课程： | 计算机组成原理实验 | | | | | | |
| 实验编号： | Project 3 | | | | | | |
| 实验名称： | Verilog HDL结构化建模 | | | | | | |
| 实验人员： | 学号 | 18111303044 | | |  | |  |
| 姓名 | 邵一波 | | |  | |  |
| 班级 | 18计算机类一班 | | |  | |  |
| 实验日期： | 2019-9-23 | | | | | | |
| 实验室： | 学苑南楼1幢305 | | | | | | |
|  |  | | | | | | |
| 实验评价： |  | | | | | | |
| 实验成绩： | |  | 评价日期： | |  | |
|  | 指导教师： | |  | | | | |

# Verilog HDL结构化建模

# 一、实验目的

## 1.掌握Verilog HDL的结构化建模方法。

## 2.掌握Verilog HDL的结构化验证技术。

# 二、实验工具

## 1.Xilinx Vivado 2014.2软件。

## 2.Windows系统PC机。

# 三、实验要求

## 通过“三人表决器”实例演示学习Verilog HDL的结构化建模方法和验证技术。利用Verilog HDL的结构化建模方法对“并行加法器/减法器”建模和验证。

# 四、实验内容

## 1. “三人表决器”的结构化建模与验证。

设计一个3人判决电路，若3个人有2人或超过2人同意，则表决结果通过，否则表决结果为不通过。

### (1)列真值表

设a、b、c分别代表3个人 ，同意用1表示，不同意用0表示，y代表表决结果，1表示通过，0表示不通过。根据题意，当a、b、c三个中2个为1，或者3个全为1时，y为1，否则y为0。

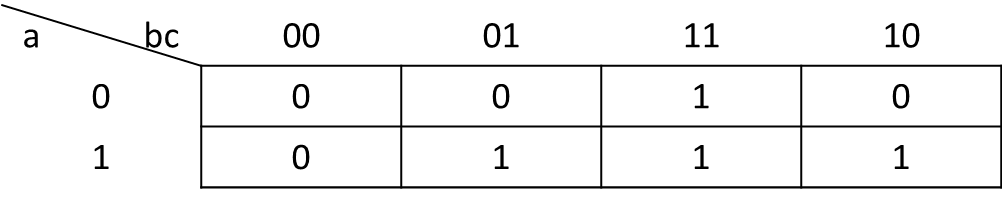
表1 “三人表决器”真值表

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | | | 输出 |
| **a** | **b** | **c** | **y** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

### (2)列输出方程



### (3)化简输出方程



**y**

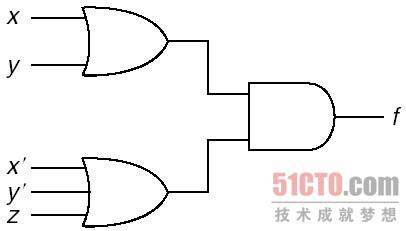
图1 “三人表决器”卡诺图

### (4)根据化简后的方程画电路图

**a**

**b**

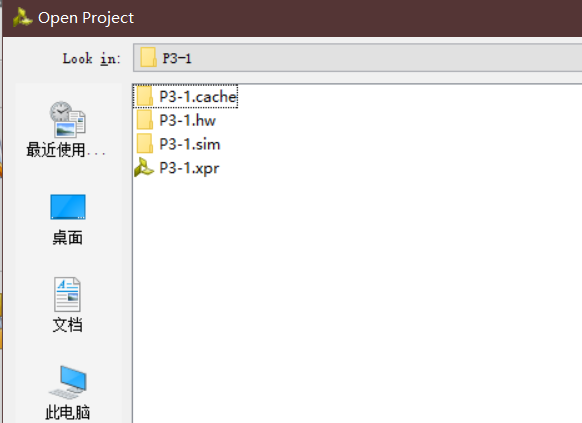
**c**



**y**

图2 “三人表决器”简化逻辑电路图

### (5)建立工程P3-1



### (6)结构化门级建模

//decision\_structure.v文件

module decision\_structure (a,b,c,y);

input wire a,b,c;

output wire y;

wire ab,bc,ac;

and g1(ab,a,b);

and g2(bc,b,c);

and g3(ac,a,c);

or g4(y,ab,bc,ac);

endmodule

### (7)设计Test Bench

// test\_decision.v文件

`timescale 1ns/100ps

module test\_decision();

reg clk\_1Hz,a,b,c;

wire y;

decision\_structure u0(a,b,c,y);

always #10 clk\_1Hz = ~clk\_1Hz ;

initial begin

clk\_1Hz=0;

#20;

a=0;b=0;c=0;

#20;

a=0;b=0;c=1;

#20;

a=0;b=1;c=0;

#100;

a=0;b=1;c=1;

#20;

a=1;b=0;c=0;

#20;

a=1;b=0;c=1;

#20;

a=1;b=1;c=0;

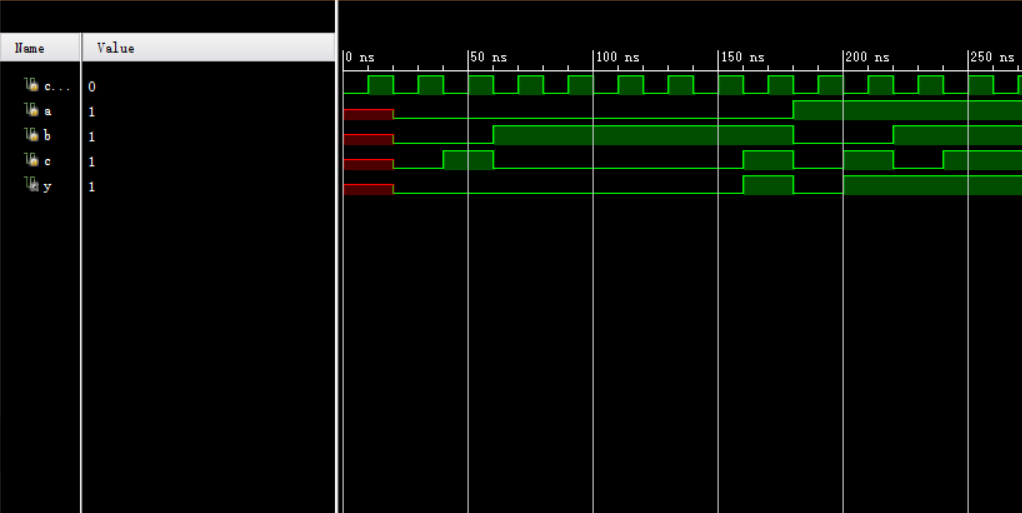
#20;

a=1;b=1;c=1;

end

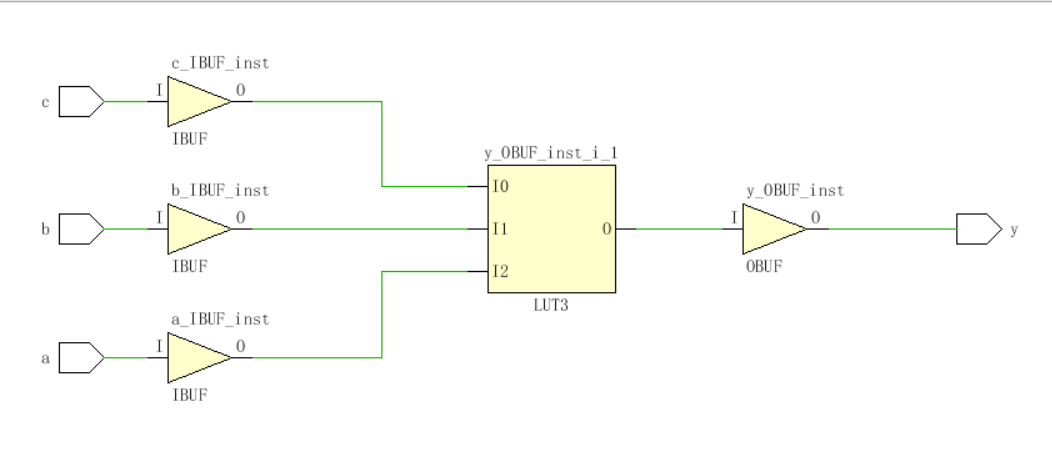
endmodule

### (8)仿真，查看波形



### (9)分析，查看RTL原理图

### (9)综合，查看综合后原理图



## 2.“ 二进制并行加法/减法器”建模与验证

### (1)列“4位超前进位加法器”输出方程

超前进位加法器的逻辑函数如下：



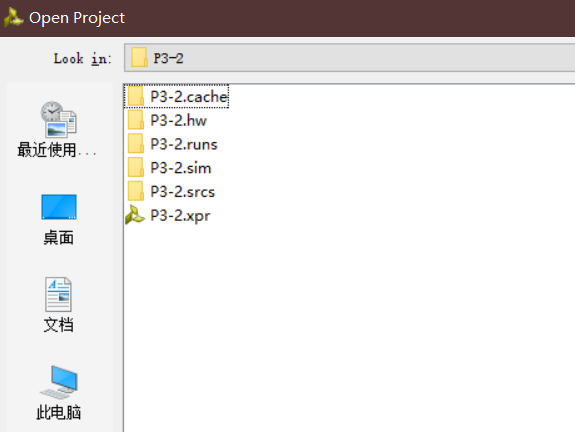
### (2)画“4位超前进位加法器”逻辑电路图

“4位超前进位加法器”对应逻辑电路图如图3所示。



图3 “超前进位加法器”逻辑电路图

### (3)建立工程P3-2



### (4)“4位超前进位加法器”结构化门级建模

//四位全加器的描述

module parallel\_adder(a,b,cin,s,cout);

parameter N=4;//二进制位数

input wire [N-1:0]a;//加数

input wire [N-1:0]b;//被加数

input wire cin;//进位输入

output wire [N-1:0]s;//和

output wire cout;//进位输出

wire c0,c1,c2;

fulladder u0\_fa(a[0],b[0],cin,s[0],c0);

fulladder u1\_fa(a[1],b[1],c0,s[1],c1);

fulladder u2\_fa(a[2],b[2],c1,s[2],c2);

fulladder u3\_fa(a[3],b[3],c2,s[3],cout);

endmodule

//一位全加器的描述

module fulladder(input a,b,cin,output sum,cout);

wire s,d1,d2;

halfadder ha0(.b(b),.a(a),.s(s),.c(d1));

halfadder ha1(.b(cin),.a(s),.s(sum),.c(d2));

or g1(cout,d2,d1);

endmodule

//一位半加器的描述

module halfadder(input a,b,output s,c);

xor(s,a,b);

and(c,a,b);

endmodule

### (5)设计“4位超前进位加法器”Test Bench

`timescale 1ns/100ps

module test\_adder();

reg clk\_1Hz;

reg [3:0]a,b;

reg cin;

wire [3:0]s;

wire cout;

parallel\_adder u0(a,b,cin,s,cout);

always #10 clk\_1Hz = ~clk\_1Hz ;

initial

begin

clk\_1Hz=0;

#20;

a=4'b0000;b=4'b0000;cin=1'b0;

#20;

a=4'b0000;b=4'b0001;cin=1'b0;

#20;

b=4'b1111;cin=1'b1;

#20;

a=4'b1101;b=4'b0100;

#20;

a=4'b0000;b=4'b1001;cin=1'b0;

#20;

a=4'b1000;b=4'b0110;

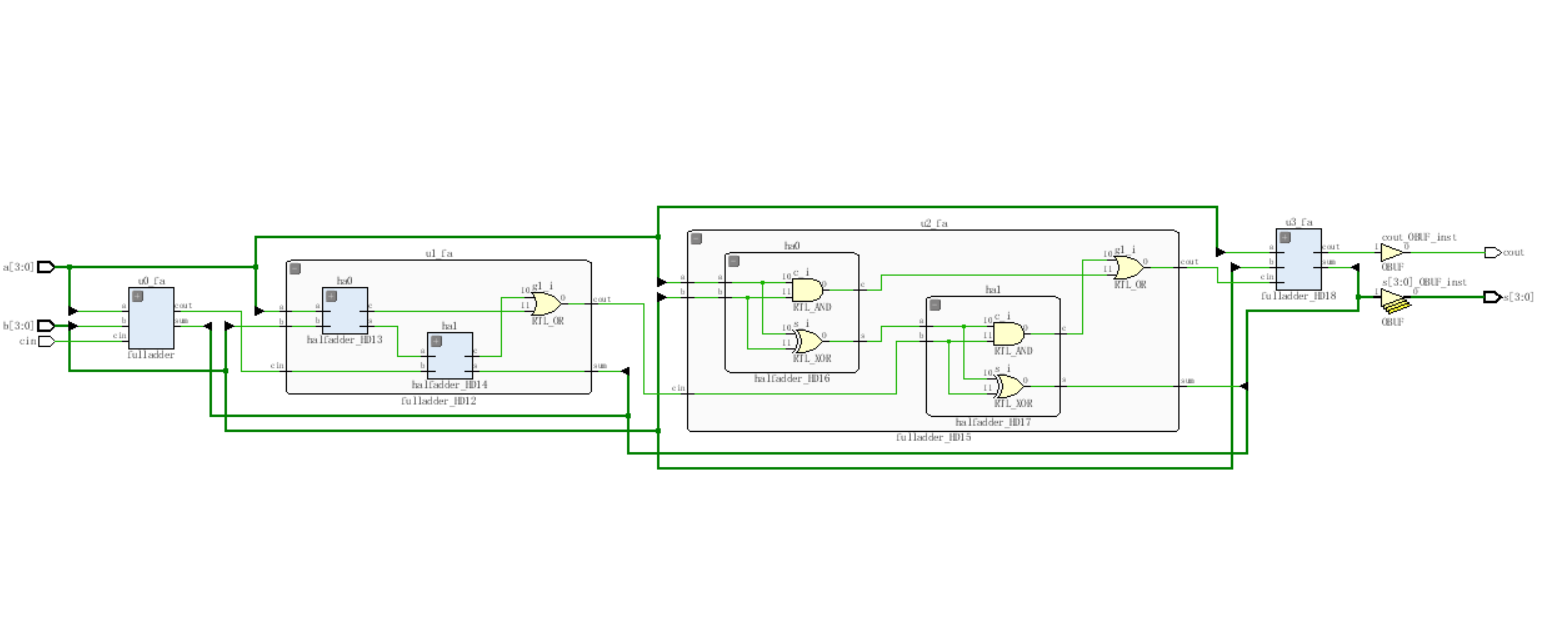
end

endmodule

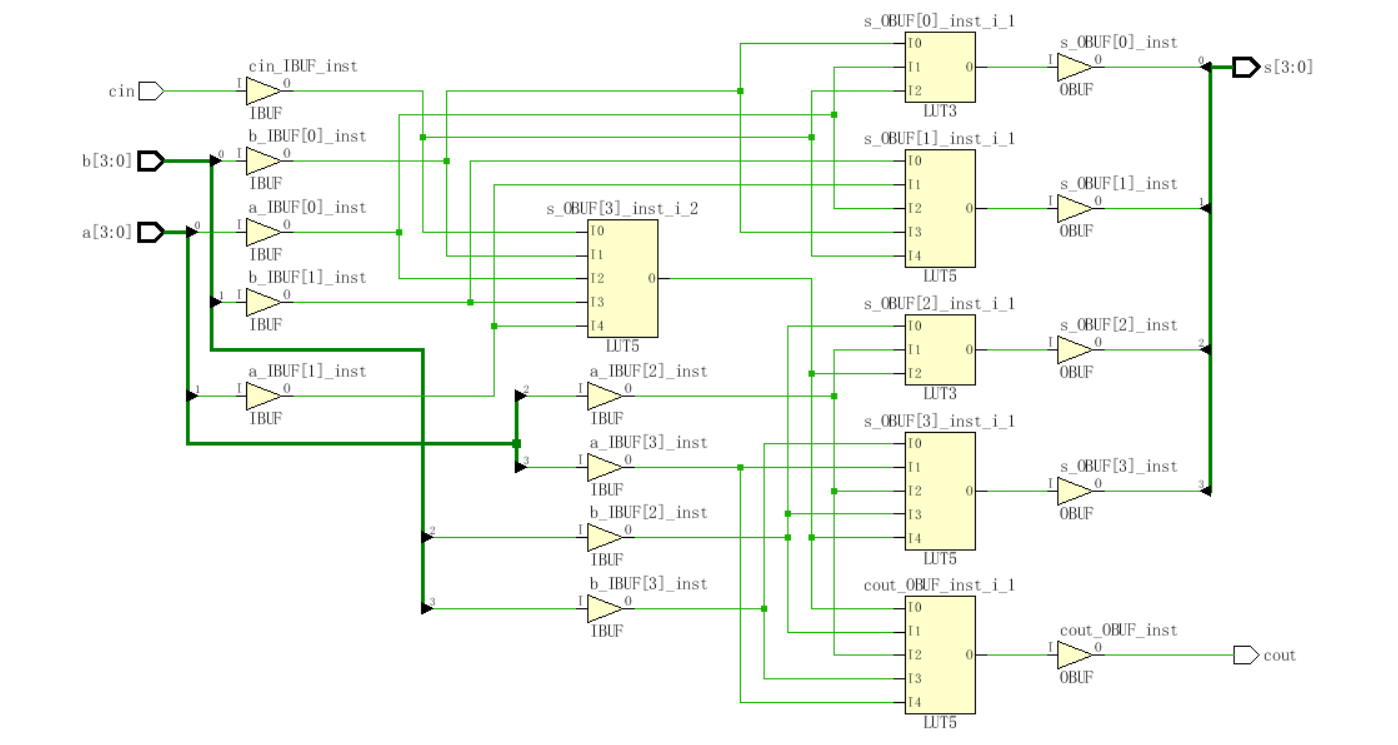
### (6)仿真“4位超前进位加法器”，查看波形



### (7)分析 “4位超前进位加法器”，查看RTL原理图



### (8)综合“4位超前进位加法器”，查看原理图



### (9)“二进制并行加法/减法器”逻辑电路图

时，，电路执行运算；当时，，电路执行运算。

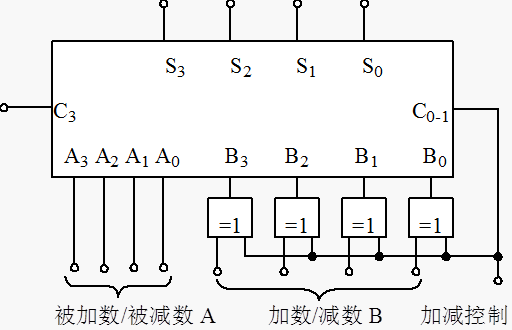


图4 “二进制并行加法/减法器”逻辑电路图

### (10) “二进制并行加法/减法器”结构化建模

module adder(a,b,op,s,cout);

parameter N=4;//二进制位数

input wire [N-1:0]a;//被加(减)数

input wire [N-1:0]b;//加(减)数

input wire op;//操作控制

output wire [N-1:0]s;//和

output wire cout;//进位输出

wire c1,c2,c3;

wire b0,b1,b2,b3;

xor(b0,op,b[0]);

xor(b1,op,b[1]);

xor(b2,op,b[2]);

xor(b3,op,b[3]);

fulladder u0\_fa(a[0],b0,op,s[0],c1);

fulladder u1\_fa(a[1],b1,c1,s[1],c2);

fulladder u2\_fa(a[2],b2,c2,s[2],c3);

fulladder u3\_fa(a[3],b3,c3,s[3],cout);

endmodule

//一位全加器的描述

module fulladder(a,b,cin,sum,cout);

input a,b,cin;

output cout,sum;

wire x,y,c;

xor(x,a,b);

xor(sum,x,cin);

and(y,a,b);

and(c,cin,x);

or(cout,y,c);

endmodule

### (11) “二进制并行加法/减法器”结构化仿真

`timescale 1ns/100ps

module test\_adder();

reg clk\_1Hz,op;

reg [3:0]a,b;

wire [3:0]s;

wire cout;

adder u0(a,b,op,s,cout);

always #10 clk\_1Hz = ~clk\_1Hz ;

initial

begin

clk\_1Hz=0;

#20;

a=4'b1110;b=4'b1011;op=1'b0;

#20;

a=4'b1110;b=4'b1011;op=1'b1;

#20;

a=4'b0000;b=4'b0000;op=1'b0;

#20;

a=4'b0000;b=4'b0000;op=1'b1;

#20;

a=4'b0000;b=4'b0001;op=1'b0;

#20;

a=4'b0000;b=4'b0001;op=1'b1;

#20;

a=4'b1101;b=4'b0100;op=1'b0;

#20;

a=4'b1101;b=4'b0100;op=1'b1;

#20;

a=4'b0000;b=4'b1001;op=1'b0;

#20;

a=4'b0000;b=4'b1001;op=1'b1;

#20;

a=4'b1000;b=4'b0110;op=1'b0;

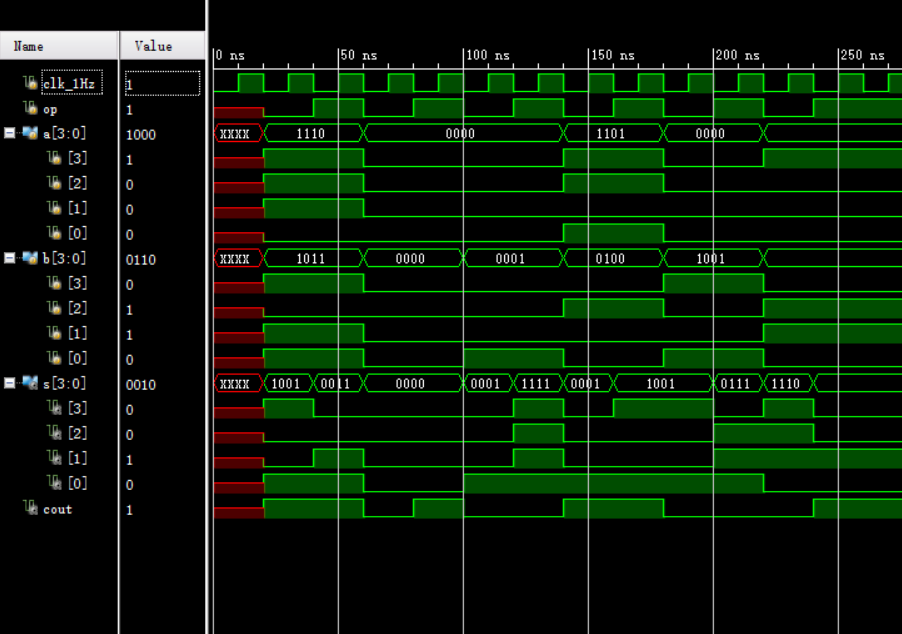
#20;

a=4'b1000;b=4'b0110;op=1'b1;

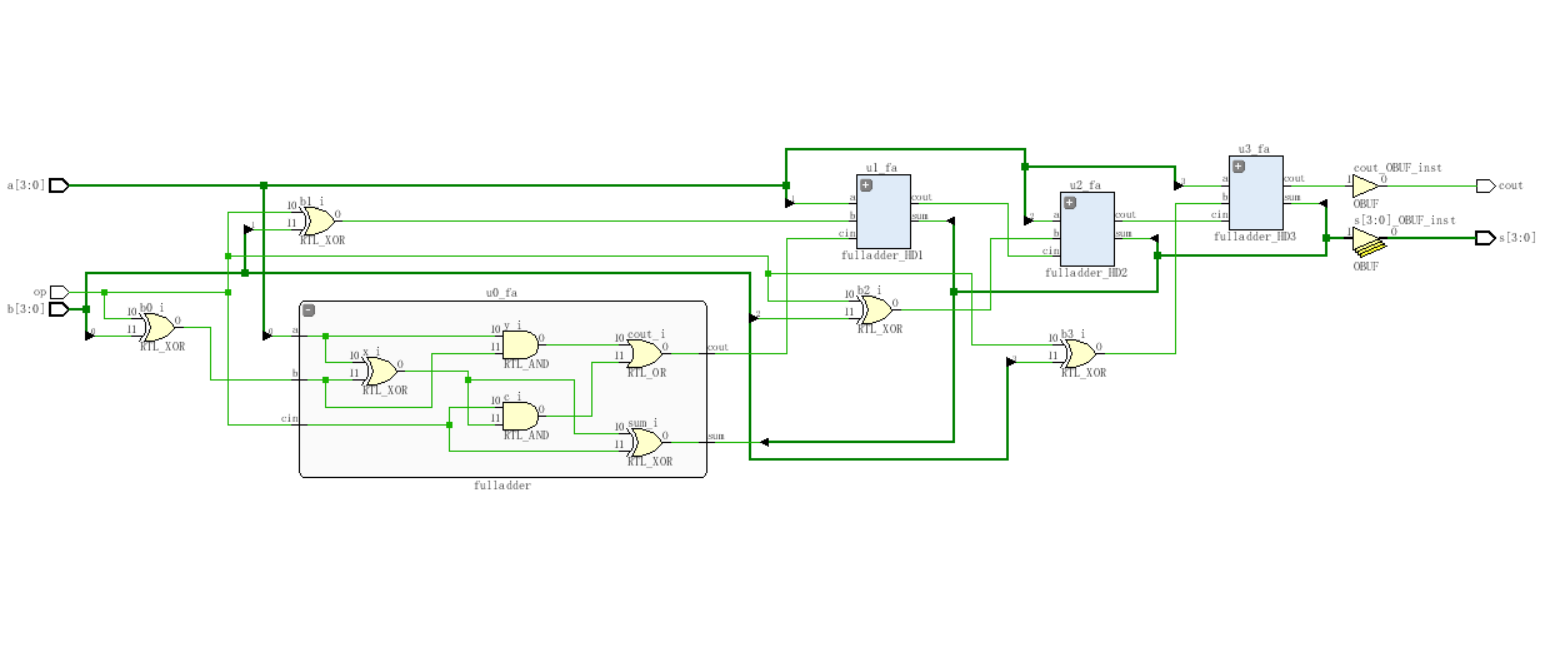
end

endmodule

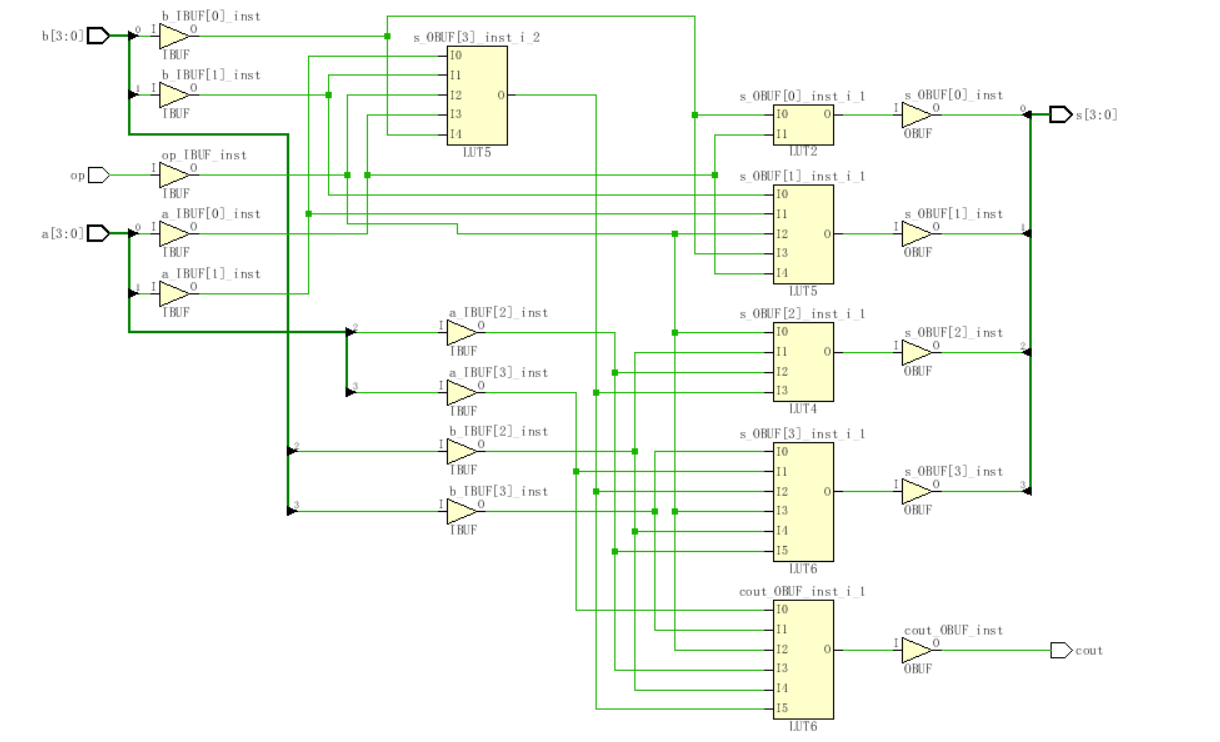
### (12) 仿真“二进制并行加法/减法器”，查看波形



### (13) 分析“二进制并行加法/减法器”，查看RTL原理图



### (14)综合“二进制并行加法/减法器”，查看原理图



# 五、实验思考

## 1.结构化建模的特点是什么？有何优点和缺点？

【答】

特点：用系统工程的思想和工程化的方法，按用户至上的原则，结构化，模块化，自顶向下地对系统进行分析与设计

优点：强调开发过程各阶段的完整性和顺序性，强调严格地区分开发阶段，严格地进行系统分析和设计，及时总结，及时问题反馈和纠正，从而避免了开发过程的混乱状态

缺点：它要求系统开发者在早期调查中就要充分的掌握用户需求、管理状况以及预见能发生的变化，这是不太符合人们循序渐进地认识事务的客观规律性。

## 2.完成模块建模后，在编写对应的Test Bench时，在选择测试数据（向量）时如果用穷举法的话，可能测试数据太多，怎样有选择地测试？

【答】

尽量选择易于判断，与期望值相关的数据。