PS: 由于板卡上的开关不够用，所以这里设计4位ALU，更复杂的ALU原理和步骤都是差不多的

1. 程序文件源码+注释分析

module ALU4(

input [3:0] ina, //定义输入/输出变量

input [3:0] inb,

output [3:0] out,

input [2:0] opcode,

input clk

);

reg [3:0] t; //此处为临时变量，由于输出变量应为连线型，而always块内的被赋值变量必须为寄存器型变量，当遇到想本实验需要在always语句块内对输出变量赋值的情况，就可以考虑中间的临时变量来调节

always @(posedge clk) begin

case(opcode)

3'b000:t=ina+inb;

3'b001:t=ina-inb;

3'b010:t=ina+1;

3'b011:t=ina-1;

3'b100:t=ina&&inb;

3'b101:t=ina||inb;

3'b110:t=~ina;

3'b111:t=ina^inb;

default:t=0;

endcase

end

assign out=t; //将中间变量的结果送至输出

endmodule

1. 测试文件源码+注释分析

module ALU4\_sim();

reg [3:0] ina\_tb; //定义输入/输出文件类型

reg [3:0] inb\_tb;

wire [3:0] out\_tb;

reg [2:0] opcode\_tb;

reg clk;

ALU4 t1(ina\_tb,inb\_tb,out\_tb,opcode\_tb,clk); //调用程序进行测试

initial begin

ina\_tb={$random}%16; //产生0~15的随机数作为初始数据

inb\_tb={$random}%16;

opcode\_tb=0;

clk=0;

#100 opcode\_tb=000;

#100 opcode\_tb=001;

#100 opcode\_tb=010;

#100 opcode\_tb=011;

#100 opcode\_tb=100;

#100 opcode\_tb=101;

#100 opcode\_tb=110;

#100 opcode\_tb=111;

#100 $stop;

end

always #50 clk=~clk;

endmodule

约束文件源码（下板操作需要）

## Clock signal

set\_property PACKAGE\_PIN E3 [get\_ports clk]

set\_property IOSTANDARD LVCMOS33 [get\_ports clk]

##Switches

set\_property PACKAGE\_PIN J15 [get\_ports {ina[0]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {ina[0]}]

set\_property PACKAGE\_PIN L16 [get\_ports {ina[1]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {ina[1]}]

set\_property PACKAGE\_PIN M13 [get\_ports {ina[2]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {ina[2]}]

set\_property PACKAGE\_PIN R15 [get\_ports {ina[3]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {ina[3]}]

set\_property PACKAGE\_PIN R17 [get\_ports {inb[0]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {inb[0]}]

set\_property PACKAGE\_PIN T18 [get\_ports {inb[1]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {inb[1]}]

set\_property PACKAGE\_PIN U18 [get\_ports {inb[2]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {inb[2]}]

set\_property PACKAGE\_PIN R13 [get\_ports {inb[3]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {inb[3]}]

set\_property PACKAGE\_PIN U12 [get\_ports {opcode[0]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {opcode[0]}]

set\_property PACKAGE\_PIN U11 [get\_ports {opcode[1]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {opcode[1]}]

set\_property PACKAGE\_PIN V10 [get\_ports {opcode[2]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {opcode[2]}]

## LEDs

set\_property PACKAGE\_PIN V15 [get\_ports {out[0]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {out[0]}]

set\_property PACKAGE\_PIN V14 [get\_ports {out[1]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {out[1]}]

set\_property PACKAGE\_PIN V12 [get\_ports {out[2]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {out[2]}]

set\_property PACKAGE\_PIN V11 [get\_ports {out[3]}]

set\_property IOSTANDARD LVCMOS33 [get\_ports {out[3]}]

1. 仿真结果如下图

可以看出当opcode的值依次递增时，ina和inb依次做加，减，加1，减1，逻辑与，逻辑或，逻辑非，逻辑异或等操作，结果存放在out里

