ALU 设计实验

（练习实验一）

设计一个 32 位的 ALU。

1、项目名：ALU32

2、代码文件内容，如下：

```
module ALU32(
    input [2:0] ALUopcode,
    input [31:0] rega,
    input [31:0] regb,
    output reg [31:0] result,
    output zero
    );

    assign zero = (result==0)?1:0;
    always @( ALUopcode or rega or regb ) begin
        case (ALUopcode)
          3'b000 : result = rega + regb;
          3'b001 : result = rega - regb;
          3'b010 : result = rega & regb;
          3'b011 : result = rega | regb;
          3'b100 : result = (rega < regb)?1:0;      // 不带符号比较
          3'b101 : begin                            // 带符号比较
                     if (rega<regb &&(( rega[31] == 0 && regb[31]==0)   ||
                         (rega[31] == 1 && regb[31]==1)))   result = 1;
                     else if (rega[31] == 0 && regb[31]==1)   result = 0;
                     else if ( rega[31] == 1 && regb[31]==0)   result = 1;
                     else result = 0;
                   end
          default : begin
                   result = 8'h00000000;
                   $display (" no match");
            end
         endcase
     end
endmodule
```

3、测试文件内容，如下：

```
module ALU32_sim( );

    // Inputs
```

```
reg [2:0] ALUopcode;
reg [31:0] rega;
reg [31:0] regb;

// Outputs
wire [31:0] result;
wire zero;

// Instantiate the Unit Under Test (UUT)
ALU32 uut (
	.ALUopcode(ALUopcode),
	.rega(rega),
	.regb(regb),
	.result(result),
	.zero(zero)
);

initial begin
	// Initialize Inputs
	  ALUopcode = 0;
	  rega = 0;
	  regb = 0;

	// Wait 100 ns for global reset to finish
	#100;
	  ALUopcode = 0;   // rega + regb
	  rega = 1;
	  regb = 1;

	#100;
	  ALUopcode = 1;   // rega - regb
	  rega = 2;
	  regb = 1;

	#100;
	  ALUopcode = 1;   // rega - regb
	  rega = 1;
	  regb = 2;

	#100;
	  ALUopcode = 2;   // rega & regb
	  rega = 5;
	  regb = 1;
```

```
#100;
  ALUopcode = 3;   // rega | regb
  rega = 4;
  regb = 1;

#100;
  ALUopcode = 4;   // rega < regb? 不带符号比较
  rega = 4;
  regb = 5;

#100;
  ALUopcode = 4;   // rega < regb? 不带符号比较
  rega = 5;
  regb = 4;

#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = 5;
  regb = 4;

#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = 4;
  regb = 5;

#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = -1;
  regb = -2;

#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = -2;
  regb = -1;
#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = -1;
  regb = 0;
#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
  rega = 0;
  regb = -2;
#100;
  ALUopcode = 5;   // rega < regb? 带符号比较
```

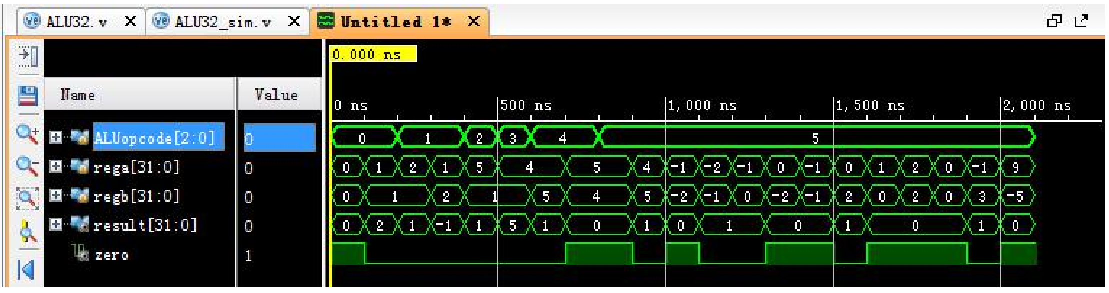
```
        rega = -1;
        regb = -1;
       #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = 0;
        regb = 2;
      #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = 1;
        regb = 0;
      #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = 2;
        regb = 2;
      #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = 0;
        regb = 0;
      #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = -1;
        regb = 3;
      #100;
        ALUopcode = 5;   // rega < regb? 带符号比较
        rega = 9;
        regb = -5;

      #100; $stop;

      // Add stimulus here

    end
endmodule
```

4、相关波形


ALU32.v
ALU32_sim.v
Untitled 1*
0.000 ns
Name
Value
0 ns
500 ns
1,000 ns
1,500 ns
2,000 ns
ALUopcode[2:0] 0
rega[31:0] 0
regb[31:0] 0
result[31:0] 0
zero 1
0 1 2 3 4 5
0 1 2 1 5 4 5 4 -1 -2 -1 0 -1 0 1 2 0 -1 9
0 1 2 1 5 4 5 -2 -1 0 -2 -1 2 0 2 0 3 -5
0 2 1 -1 1 5 1 0 1 0 1 0 1 0 1 0