Lab1_算数运算单元 ALU

姓名：姜庆彩
学院：计算机科学与技术学院
学号：PB15051087

实验要求：

- 设计一算数运算单元 ALU

采用纯组合逻辑设计

32bit 位宽

完成指定运算功能

- 检查标准：运算结果，仿真结果正确

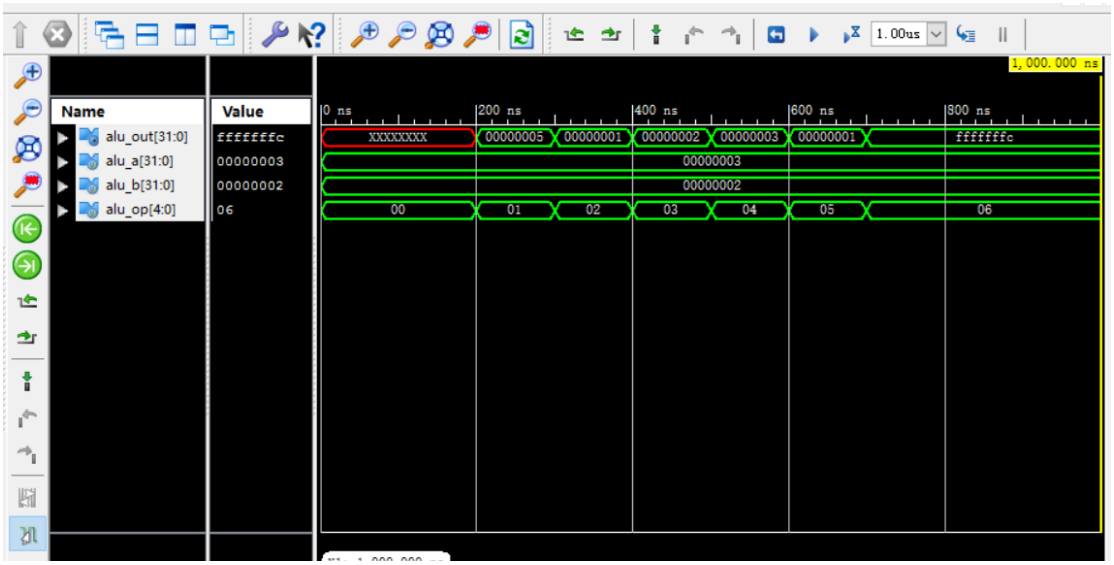
1，对所有的 7 种操作符，设计测试例进行仿真

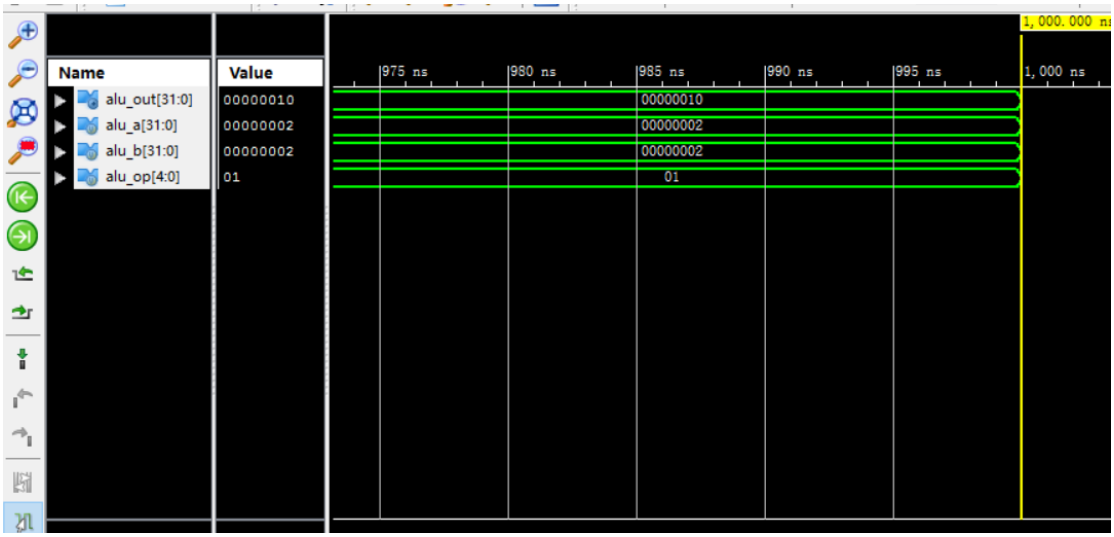
2，对于给定的斐波拉契数列运算，进行仿真

我的实现：这次的实验较为简单，就是数字逻辑电路的一个延伸，第一部分七种操作符运算较为简单，而第二部分斐波那契数列的实验我使用了四次例化加运算叠加去完成需要的操作。

实验结果：

仿真图：


Name
Value
alu_out[31:0]
alu_a[31:0]
alu_b[31:0]
alu_op[4:0]
ffffffc
00000003
00000002
06
0 ns
200 ns
400 ns
600 ns
800 ns
1, 000. 000 ns
XXXXXXXX
00000005
00000001
00000002
00000003
00000001
ffffffc
00000003
00000002
00
01
02
03
04
05
06


Name
Value
alu_out[31:0]
alu_a[31:0]
alu_b[31:0]
alu_op[4:0]
00000010
00000002
00000002
01
975 ns
980 ns
985 ns
990 ns
995 ns
1, 000 ns
1, 000. 000 ns
00000010
00000002
00000002
01

源代码：

```
module alu(
  input  signed    [31:0] alu_a,
  input  signed    [31:0] alu_b,
  input            [4:0]  alu_op,
  output     reg    [31:0]alu_out
    );
    always@(*)
    begin
    case(alu_op)
      1:   alu_out <= alu_a + alu_b;
      2:   alu_out <= alu_a - alu_b;
      3:   alu_out <= alu_a & alu_b;
      4:   alu_out <= alu_a | alu_b;
      5:   alu_out <= alu_a ^ alu_b;
      6:   alu_out <= ~(alu_a | alu_b);
      default:alu_out <= alu_out;
      endcase
      end

endmodule
```

```
module alu_sim;
   reg [31:0] alu_a;
   reg [31:0] alu_b;
   reg [4:0] alu_op;
   wire [31:0] alu_out;

   alu uut (
      .alu_a(alu_a),
      .alu_b(alu_b),
      .alu_op(alu_op),
      .alu_out(alu_out)
   );
   initial begin
      alu_a = 3;
      alu_b = 2;
      alu_op = 0;
      #100;
      alu_op = 0;
   #100;
      alu_op = 1;
      #100;
      alu_op = 2;
      #100;
      alu_op = 3;
      #100;
      alu_op =4;
         #100;
      alu_op = 5;
         #100;
      alu_op = 6;
   end
endmodule
```

```
module top (
  input  signed    [31:0] alu_a,
  input  signed    [31:0] alu_b,
  input            [4:0]  alu_op,
  output           [31:0] alu_out   );
  wire [31:0] temp1;
  wire [31:0] temp2;
  wire [31:0] temp3;
  alu a(
  .alu_a(alu_a),
  .alu_b(alu_b),
  .alu_op(alu_op),
  .alu_out(temp1)
  );
  alu b(
  .alu_a(temp1),
  .alu_b(alu_b),
  .alu_op(alu_op),
  .alu_out(temp2)
  );
  alu c(
  .alu_a(temp2),
  .alu_b(temp1),
  .alu_op(alu_op),
  .alu_out(temp3)
  );
  alu d(
  .alu_a(temp3),
  .alu_b(temp2),
  .alu_op(alu_op),
  .alu_out(alu_out)
  );

Endmodule
```

```
module top_sim;

   // Inputs
   reg [31:0] alu_a;
   reg [31:0] alu_b;
   reg [4:0] alu_op;

   // Outputs
   wire [31:0] alu_out;

   // Instantiate the Unit Under Test (UUT)
   top f (
      .alu_a(alu_a),
      .alu_b(alu_b),
      .alu_op(alu_op),
      .alu_out(alu_out)
   );

   initial begin
      // Initialize Inputs
      alu_a = 2;
      alu_b = 2;
      alu_op = 1;

      // Wait 100 ns for global reset to finish
      #100;

      // Add stimulus here

   end

endmodule
```