17300180070 马逸君

ALU实验报告

一、ALU代码

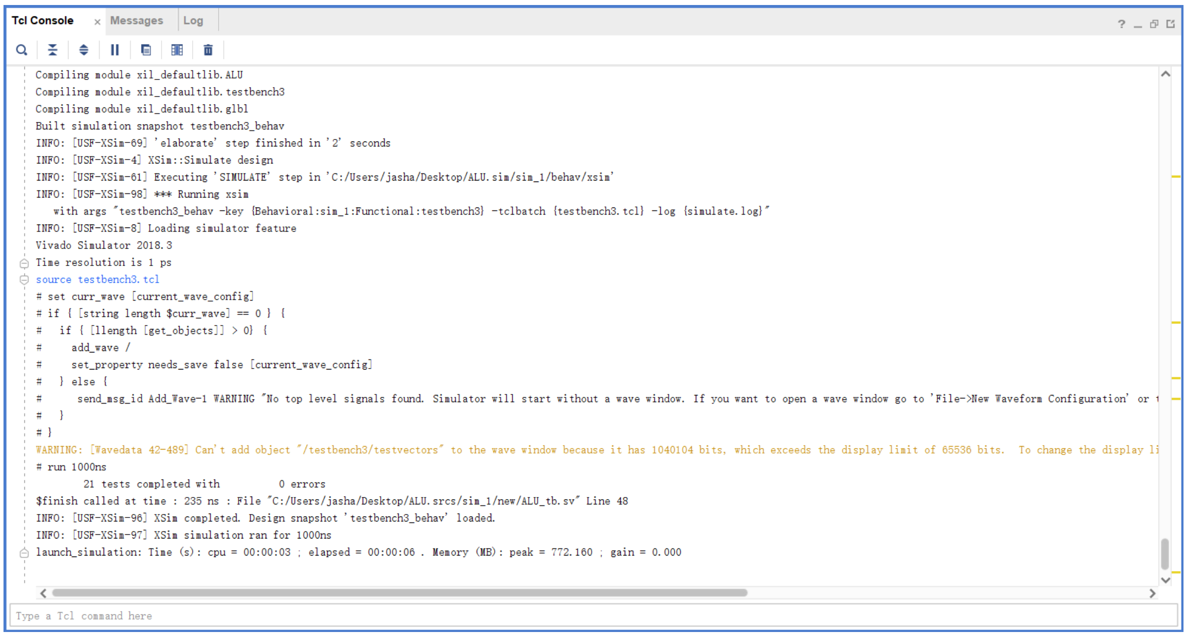
```
module ALU(
    input logic signed [31:0] op1,  // 1st operand
    input logic signed [31:0] op2,  // 2nd operand
    input logic[2:0] con,  // control signal

    output logic signed [31:0] res,  // calculation result
    output logic zf  // zero flag
);
    assign zf = res == '0 ? 1'b1 : 1'b0;
    always_comb
    begin
        case (con)
            3'b000: res = op1 & op2;
            3'b001: res = op1 | op2;
            3'b010: res = op1 + op2;
            3'b011: res = 32'bx;
            3'b100: res = op1 & (~op2);
            3'b101: res = op1 | (~op2);
            3'b110: res = op1 - op2;
            3'b111: res = op1 < op2 ? 1'b1 : 1'b0;  // 若无signed，则此处为无符号比较，是错的
        endcase
    end
endmodule
```

二、测试用例

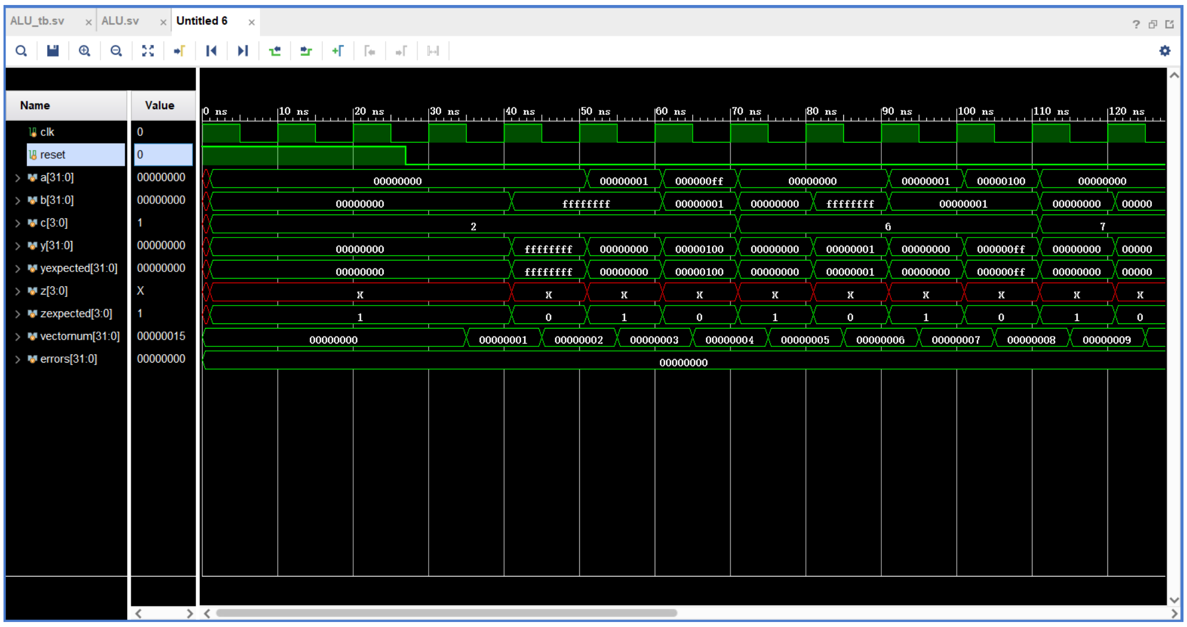
Test	a	b	alucont	result	zero
ADD 0+0	0000_0000	0000_0000	2	0000_0000	1
ADD 0+(-1)	0000_0000	FFFF_FFFF	2	FFFF_FFFF	0
ADD 1+(-1)	0000_0001	FFFF_FFFF	2	0000_0000	1
ADD FF+1	0000_00FF	0000_0001	2	0000_0100	0
SUB 0-0	0000_0000	0000_0000	6	0000_0000	1
SUB 0-(-1)	0000_0000	FFFF_FFFF	6	0000_0001	0
SUB 1-1	0000_0001	0000_0001	6	0000_0000	1
SUB 100-1	0000_0100	0000_0001	6	0000_00FF	0
SLT 0,0	0000_0000	0000_0000	7	0000_0000	1
SLT 0,1	0000_0000	0000_0001	7	0000_0001	0
SLT 0,-1	0000_0000	FFFF_FFFF	7	0000_0000	1
SLT 1,0	0000_0001	0000_0000	7	0000_0000	1
SLT –1,0	FFFF_FFFF	0000_0000	7	0000_0001	0
AND FFFFFFFF, FFFFFFFF	FFFF_FFFF	FFFF_FFFF	0	FFFF_FFFF	0
AND FFFFFFFF, 12345678	FFFF_FFFF	1234_5678	0	1234_5678	0
AND 12345678, 87654321	1234_5678	8765_4321	0	0224_4220	0
AND 00000000, FFFFFFFF	0000_0000	FFFF_FFFF	0	0000_0000	1
OR FFFFFFFF, FFFFFFFF	FFFF_FFFF	FFFF_FFFF	1	FFFF_FFFF	0
OR 12345678, 87654321	1234_5678	8765_4321	1	9775_5779	0
OR 00000000, FFFFFFFF	0000_0000	FFFF_FFFF	1	FFFF_FFFF	0
OR 00000000, 00000000	0000_0000	0000_0000	1	0000_0000	1

三、测试结果



```
Compiling module xil_defaultlib.ALU
Compiling module xil_defaultlib.testbench3
Compiling module xil_defaultlib.glbl
Built simulation snapshot testbench3_behav
INFO: [USF-XSim-69] 'elaborate' step finished in '2' seconds
INFO: [USF-XSim-4] XSim::Simulate design
INFO: [USF-XSim-61] Executing 'SIMULATE' step in 'C:/Users/jasha/Desktop/ALU.sim/sim_1/behav/xsim'
INFO: [USF-XSim-98] *** Running xsim
   with args "testbench3_behav -key {Behavioral:sim_1:Functional:testbench3} -tclbatch {testbench3.tcl} -log {simulate.log}"
INFO: [USF-XSim-8] Loading simulator feature
Vivado Simulator 2018.3
Time resolution is 1 ps
source testbench3.tcl
# set curr_wave [current_wave_config]
# if { [string length $curr_wave] == 0 } {
#   if { [llength [get_objects]] > 0} {
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window go to 'File->New Waveform Configuration' or t
#   }
# }
WARNING: [Wavedata 42-489] Can't add object "/testbench3/testvectors" to the wave window because it has 1040104 bits, which exceeds the display limit of 65536 bits.  To change the display li
# run 1000ns
        21 tests completed with          0 errors
$finish called at time : 235 ns : File "C:/Users/jasha/Desktop/ALU.srcs/sim_1/new/ALU_tb.sv" Line 48
INFO: [USF-XSim-96] XSim completed. Design snapshot 'testbench3_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:03 ; elapsed = 00:00:06 . Memory (MB): peak = 772.160 ; gain = 0.000
```

TCL日志输出如图所示，可见倒数第5行显示"21 tests completed with 0 errors"。



四、测试代码

```
module testbench3();
  logic         clk, reset;
  logic [31:0]  a, b;
  logic [3:0]   c;
  logic [31:0]  y, yexpected;
  logic [3:0]   z, zexpected;
  logic [31:0]  vectornum, errors;    // bookkeeping variables
  logic [103:0]  testvectors[10000:0]; // array of testvectors

  // instantiate device under test
  ALU alu(a, b, c[2:0], y, z[0]);

  // generate clock
```

```verilog
  always     // no sensitivity list, so it always executes
    begin
      clk = 1; #5; clk = 0; #5;
    end

  // at start of test, load vectors and pulse reset
  initial
    begin
      $readmemh("../Obj/alu_tv.txt", testvectors);
      vectornum = 0;
      errors = 0;
      reset = 1; #27;
      reset = 0;
    end
  // Note: $readmemh reads testvector files written in hexadecimal

  // apply test vectors on rising edge of clk
  always @(posedge clk)
    begin
      #1; {a, b, c, yexpected, zexpected} = testvectors[vectornum];
    end

  // check results on falling edge of clk
  always @(negedge clk)
    if (~reset) begin // skip during reset
      if (y !== yexpected || z[0] !== zexpected[0]) begin
        $display("Error: inputs = %b, %b, %b", a, b, c[2:0]);
        $display("  outputs = %b, %b (%b, %b expected)", y, z[0], yexpected, zexpected[0]);
        errors = errors + 1;
      end
      // increment array index and read next testvector
      vectornum = vectornum + 1;
      if (testvectors[vectornum] === 'x) begin
        $display("%d tests completed with %d errors", vectornum, errors);
        $finish;
      end
    end

endmodule

```

其中ALU_tv.txt是测试数据，内容如下：（格式必须相同，否则$readmemh无法读取）

```
0000_0000_0000_0000_2_0000_0000_1
0000_0000_FFFF_FFFF_2_FFFF_FFFF_0
0000_0001_FFFF_FFFF_2_0000_0000_1
0000_00FF_0000_0001_2_0000_0100_0
0000_0000_0000_0000_6_0000_0000_1
0000_0000_FFFF_FFFF_6_0000_0001_0
0000_0001_0000_0001_6_0000_0000_1
0000_0100_0000_0001_6_0000_00FF_0
0000_0000_0000_0000_7_0000_0000_1
0000_0000_0000_0001_7_0000_0001_0
0000_0000_FFFF_FFFF_7_0000_0000_1
0000_0001_0000_0000_7_0000_0000_1
```

```
FFFF_FFFF_0000_0000_7_0000_0001_0
FFFF_FFFF_FFFF_FFFF_0_FFFF_FFFF_0
FFFF_FFFF_1234_5678_0_1234_5678_0
1234_5678_8765_4321_0_0224_4220_0
0000_0000_FFFF_FFFF_0_0000_0000_1
FFFF_FFFF_FFFF_FFFF_1_FFFF_FFFF_0
1234_5678_8765_4321_1_9775_5779_0
0000_0000_FFFF_FFFF_1_FFFF_FFFF_0
0000_0000_0000_0000_1_0000_0000_1
```