UNIVERSITY OF NEVADA LAS VEGAS. DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING LABORATORIES.

Class:	CPE300L - Digital System Architecture and Design Lab	Semester:	Fall 2025

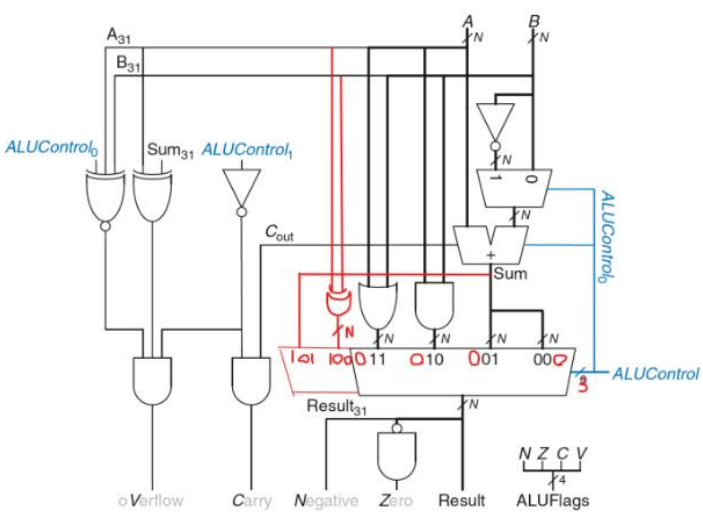
Points		Document author:	Darryll Mckoy
		Author's email:	Mckoyd1@unlv.nevada.edu
		Document topic:	Postlab 1

Instructor's comments:

1. **Hours spent on this lab**
   Around 8 hours

2. **ALU**
   a) Updated table and schematic with added XOR function

ALUControl$_{2:0}$	Function
000	Add
001	Subtract
010	And
011	Or
100	Xor
101	SLT



b) Table of ALU test vector operations with added XOR and SLT:

Test	ALUControl	A	B	Result	ALUFlags
ADD 0+0	0	00000000	00000000	00000000	4
ADD 0+(-1)	0	00000000	FFFFFFFF	FFFFFFFF	8
ADD 1+(-1)	0	00000001	FFFFFFFF	00000000	6
ADD FF+1	0	000000FF	00000001	00000100	2
SUB 0-0	1	00000000	00000000	00000000	4
SUB 0-(-1)	1	00000000	FFFFFFFF	00000001	0
SUB 1-1	1	00000001	00000001	00000000	4
SUB 100-1	1	00000100	00000001	000000FF	0
AND FFFFFFFF, FFFFFFFF	0	FFFFFFFF	FFFFFFFFF	FFFFFFFF	8
AND FFFFFFFF, 12345678	0	FFFFFFFF	12345678	12345678	0
AND 12345678, 87654321	0	12345678	87654321	2244220	0
AND 00000000, FFFFFFFF	0	00000000	FFFFFFFF	00000000	4
OR FFFFFFFF, FFFFFFFF	1	FFFFFFFF	FFFFFFFF	FFFFFFFF	0
OR 12345678, 87654321	1	12345678	87654321	97755779	0
OR 00000000, FFFFFFFF	1	00000000	FFFFFFFF	FFFFFFFF	8
OR 00000000, 00000000	1	00000000	00000000	00000000	4

ADD 80000000, 80000000	0	80000000	80000000	00000000	7
ADD 7FFFFFFF, 00000001	0	7FFFFFFF	00000001	80000000	9
SUB 80000000, 00000010	1	80000000	00000010	7FFFFFF0	1
SUB 7FFFFFFF, FFFFFFFF	1	7FFFFFFF	FFFFFFFF	80000000	B
SLT FFFFFFFF, 1	1	FFFFFFFF	00000001	00000001	0
SLT 0, 0	1	00000000	00000000	00000000	4
SLT 12345678, 87654321	1	12345678	87654321	00000000	4
SLT FFFFFFFF, 00000000	1	FFFFFFFF	000000000	00000001	0
XOR 1, 1	0	00000001	00000001	00000000	4
XOR	0	FFFFFFFF	00000001	FFFFFFFE	9
XOR 12345678, 00040000	0	12345678	00040000	12305678	0
XOR	0	7FFFFFFFF	00000020	7FFFFFFDF	0

c) System Verilog ALU file

C: > altera > 14.1 > alu.sv

```
////////////////////////////////////////////////////
//32-bit ALU
//Darryll Mckoy
//CPE300L 1 ALU and testbenches


module alu(input logic [31:0] A, B,
input logic [2:0] ALUControl,
output logic [31:0] Result,
output logic [3:0] ALUFlags);

logic [32:0] Carry;
//initial begin

//ALUFlags = 0;
//end

////////////////////////////////////////////////////
always @(*) //include all assignments as combinational logic


case (ALUControl)
3'b000: begin
    Carry = A + B;          //Carry is 1 bit larger than A and B so MSB will change to 1 if overflow.
    Result = Carry[31:0];       //We only want the first 31 bits.
    ALUFlags[1] = Carry[32];    //sets overflow flag
    ALUFlags[3] = Result[31];   //Will set negative flag if MSB is 1.

if ( A[31] == B[31] && A[31] != Carry[31]) begin
    ALUFlags[0] = 1'b1;     //sets carry flag to 1.
end

else begin
    ALUFlags[0] = 1'b0; //sets carry flag to zero.
end

end
```

```
////////////////////////////////////////////////
3'b001: begin
    Carry = A - B;          //Carry is 1 bit larger than A and B so MSB will change to 1 if overflow.
    Result = Carry[31:0];       //We only want the first 31 bits.
    //Result = A - B;
    ALUFlags[3] = Result[31];   //Will set negative flag if MSB is 1.

if ( A[31] == B[31] && A[31] != Carry[31]) begin
    ALUFlags[1] = 1'b1;     //sets carry flag to 1.
end

else begin
    ALUFlags[1] = 1'b0; //sets carry flag to zero.
end

end

////////////////////////////////////////////
3'b010: Result = A & B;         //AND
3'b011: Result = A | B;         //OR
3'b100: Result = A ^ B;         //XOR
3'b101: Result = (A < B) ? A : B;   //SLT

default: Result = 32'b0;

endcase

////////////////////////////////////////////
always_comb begin

if (Result == 32'b0) begin   //Zero flag
    ALUFlags[2] = 1'b1;     //sets zero flag to 1 if result is zero.
end

else begin
    ALUFlags[2] = 1'b0; //sets zero flag to zero if > zero.
end


end
endmodule
```

d) ALU text test vector file

C: > altera > 14.1 > alu.txt

```
0_00000000_00000000_00000000_4
0_00000000_FFFFFFFF_FFFFFFFF_8
0_00000001_FFFFFFFF_00000000_6
0 000000FF_00000001_00000100_2
1_00000000_00000000_00000000_4
1_00000000_FFFFFFFF_00000001_0
1_00000001_00000001_00000000_4
1_00000100_00000001_000000FF_0
2_FFFFFFFF_FFFFFFFF_FFFFFFFF_8
2_FFFFFFFF_12345678_12345678_0
2_12345678_87654321_02244220_0
2_00000000_FFFFFFFF_00000000_4
3_FFFFFFFF_FFFFFFFF_FFFFFFFF_0
3_12345678_87654321_97755779_0
3_00000000_FFFFFFFF_FFFFFFFF_8
3_00000000_00000000_00000000_4
0_80000000_80000000_00000000_7
0_7FFFFFFF_00000001_80000000_9
1_80000000_00000010_7FFFFFF0_1
1_7FFFFFFF_FFFFFFFF_80000000_B
5_FFFFFFFF_00000001_00000001_0
5_00000000_00000000_00000000_4
5_12345678_87654321_00000000_4
5_FFFFFFFF_00000000_00000001_0
4_00000001_00000001_00000000_4
4_FFFFFFFF_00000001_FFFFFFFE_0
4_12345678_00040000_12305678_0
4_7FFFFFFF_00000020_7FFFFFDF_0
```

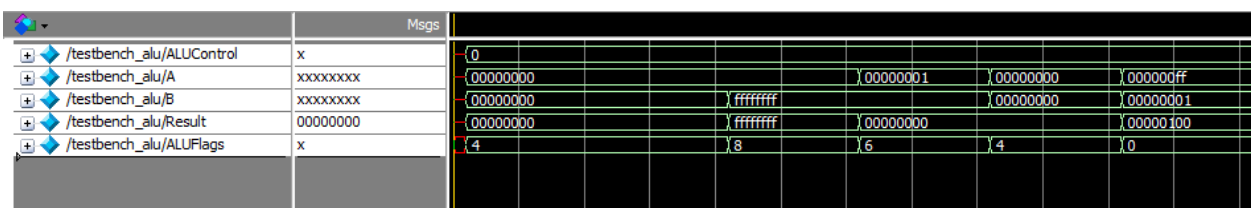
e) ALU testbench

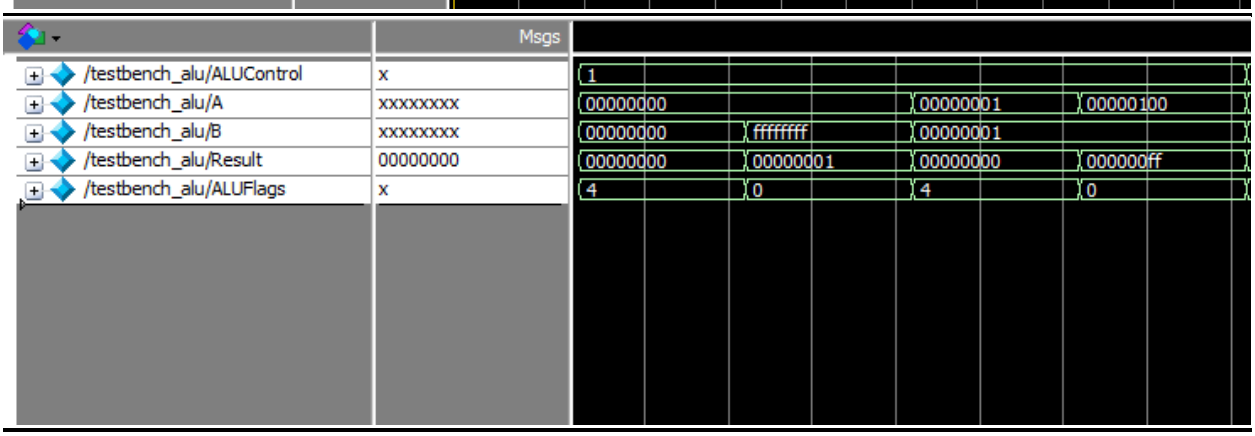
C: > altera > 14.1 > testbench_alu.sv

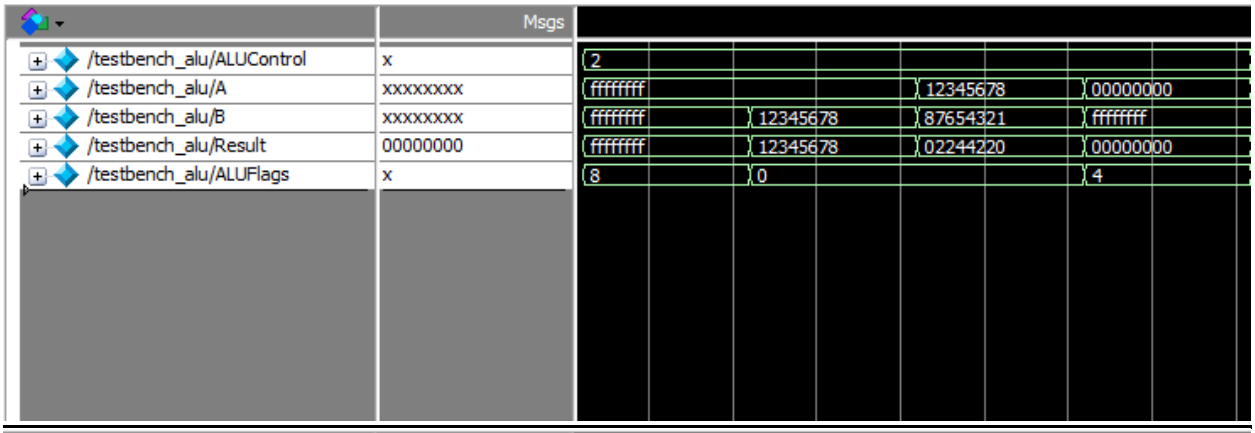
```
//ALU testbench

module testbench_alu();
  logic        clk, reset;
  logic [31:0] A, B, Result, ResultExpected;
  logic [2:0]  ALUControl;
  logic [3:0]  ALUFlags;
  logic [31:0] vectornum, errors;
  logic [250:0]  testvectors[10000:0];

  // instantiate device under test
  alu dut(.A(A), .B(B), .ALUControl(ALUControl), .Result(Result), .ALUFlags(ALUFlags));

  // generate clock
  always
    begin
      clk = 1; #5; clk = 0; #5;
    end

  // at start of test, load vectors
  // and pulse reset
  initial
    begin
      $readmemh("alu.txt", testvectors);
      vectornum = 0; errors = 0;
      reset = 1; #15; reset = 0; //#30;
    end
```

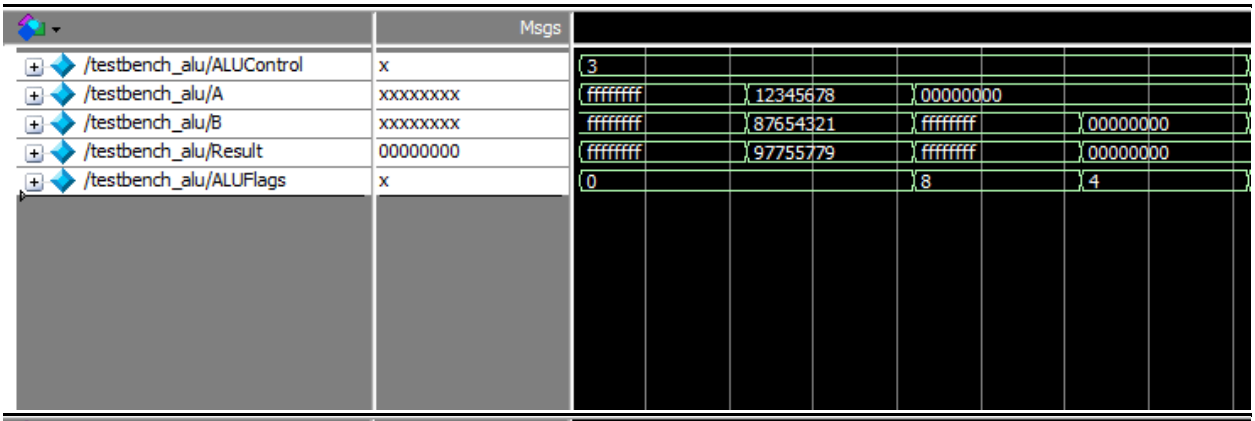
```
// apply test vectors on rising edge of clk
always @(posedge clk)
  begin
    #1; {ALUControl, A, B, ResultExpected, ALUFlags} = testvectors[vectornum];
  end

// check results on falling edge of clk
always @(negedge clk)
  if (~reset) begin // skip during reset
    if (Result !== ResultExpected) begin  // check result
      $display("Error: inputs = %h", {ALUControl, A, B});
      $display("  outputs = %h (%h expected)",Result, ResultExpected);
      errors = errors + 1;
    end


    vectornum = vectornum + 1;

    if (testvectors[vectornum] === 27'hx) begin
      $display("%h tests completed with %h errors",
              vectornum, errors);
      $finish;
    end
end
endmodule
```

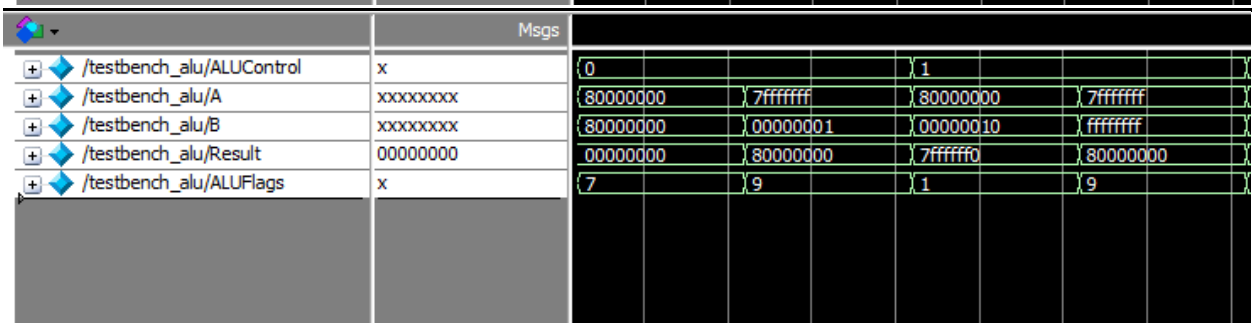
f) Test waveform

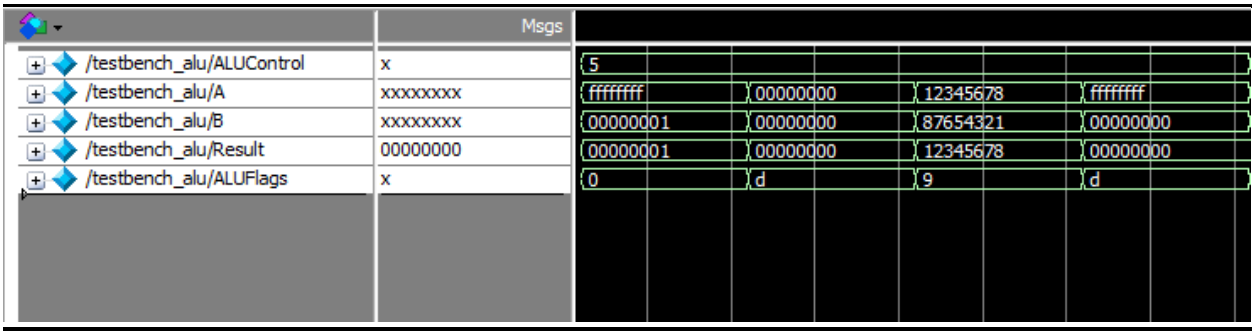


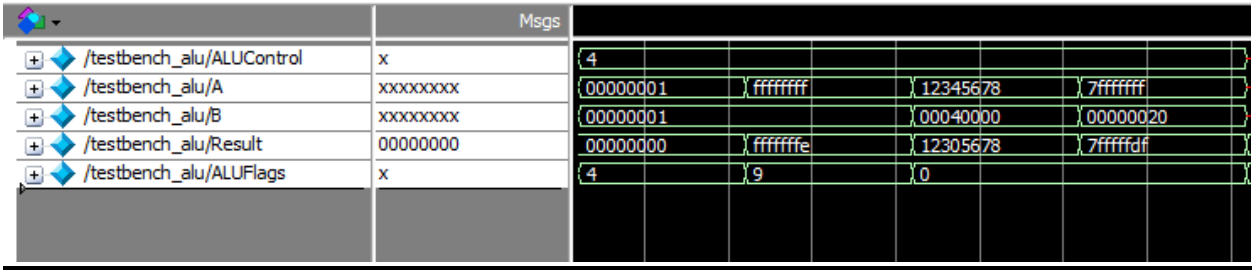



Msgs
/testbench_alu/ALUControl x 2
/testbench_alu/A xxxxxxxx ffffffff 12345678 00000000
/testbench_alu/B xxxxxxxx ffffffff 12345678 87654321 ffffffff
/testbench_alu/Result 00000000 ffffffff 12345678 02244220 00000000
/testbench_alu/ALUFlags x 8 0 4


Msgs
/testbench_alu/ALUControl x 3
/testbench_alu/A xxxxxxxx ffffffff 12345678 00000000
/testbench_alu/B xxxxxxxx ffffffff 87654321 ffffffff 00000000
/testbench_alu/Result 00000000 ffffffff 97755779 ffffffff 00000000
/testbench_alu/ALUFlags x 0 8 4


Msgs
/testbench_alu/ALUControl x 0 1
/testbench_alu/A xxxxxxxx 80000000 7fffffff 80000000 7fffffff
/testbench_alu/B xxxxxxxx 80000000 00000001 00000010 ffffffff
/testbench_alu/Result 00000000 00000000 80000000 7ffffff0 80000000
/testbench_alu/ALUFlags x 7 9 1 9


Msgs
/testbench_alu/ALUControl x 5
/testbench_alu/A xxxxxxxx ffffffff 00000000 12345678 ffffffff
/testbench_alu/B xxxxxxxx 00000001 00000000 87654321 00000000
/testbench_alu/Result 00000000 00000001 00000000 12345678 00000000
/testbench_alu/ALUFlags x 0 d 9 d



3. Feedback

The new format so far has been much better compared to previous years. Labs are much better with the removal of the prelab submissions and change of format.