ECEN 651 Lab Exercise 4 Report

Arithmetic Logic Unit in MIPS

Name: Pranav Anantharam

UIN: 734003886

Date: 10/03/2023

1. Design the ALU module.

Block Diagram:

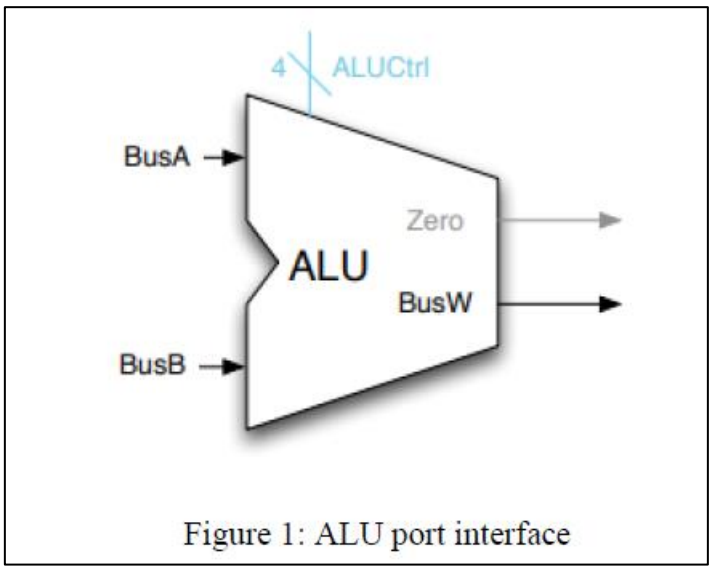


Figure 1: ALU port interface

(1) Write a Verilog module to implement the ALU module.

Design Source code filename: ALU.v

```
`timescale 1ns / 1ps

// MACROs
`define AND 4'b0000
`define OR 4'b0001
`define ADD 4'b0010
`define SLL 4'b0011
`define SRL 4'b0100
`define SUB 4'b0110
`define SLT 4'b0111
```

```
`define ADDU 4'b1000
`define SUBU 4'b1001
`define XOR 4'b1010
`define SLTU 4'b1011
`define NOR 4'b1100
`define SRA 4'b1101
`define LUI 4'b1110

module ALU(BusW, Zero, BusA, BusB, ALUCtrl);
// Inputs
input wire [31:0] BusA, BusB;
input wire [3:0] ALUCtrl;

// Outputs
output reg [31:0] BusW;
output wire Zero ;

wire less;
wire [63:0] Bus64;
// Zero signal is HIGH when BusW is zero
assign Zero = ( BusW == 32'b0 ? 32'b1 : 32'b0);
// less is HIGH when BusA < BusB (unsigned comparison)
assign less = ({1'b0,BusA} < {1'b0,BusB}  ? 1'b1 : 1'b0);
assign Bus64 = 0;
// Switch case - performing arithmetic operations based on ALU Control Line
always@(*)begin
      case (ALUCtrl)
      `AND:   BusW <= BusA & BusB;
      `OR:    BusW <= BusA | BusB;
      `ADD:   BusW <= BusA + BusB;
      `ADDU:  BusW <= BusA + BusB;   // Unsigned Addition
      `SLL:   BusW <= BusA << BusB;  // Shift Left Logical
      `SRL:   BusW <= BusA >> BusB;  // Shift Right Logical
      `SUB:   BusW <= BusA - BusB;
      `SUBU:  BusW <= BusA - BusB;   // Unsigned Subtraction
      `XOR:   BusW <= BusA ^ BusB;
      `NOR:   BusW <= ~(BusA|BusB);
```

```
	// Set if less than (SLT)
	`SLT:	BusW <= $signed(BusA) < $signed(BusB) ? 32'b1 : 32'b0;
	`SLTU:	BusW <= less;				// SLT unsigned
	`SRA:	BusW <= $signed(BusA) >>> BusB;	// Shift Right Arithmetic
	`LUI:	BusW <= BusB << 16;			// Load Upper Immediate
	default:BusW <= 32'bx;
	endcase
end
endmodule
```

Testbench Source code filename: ALUTest.v

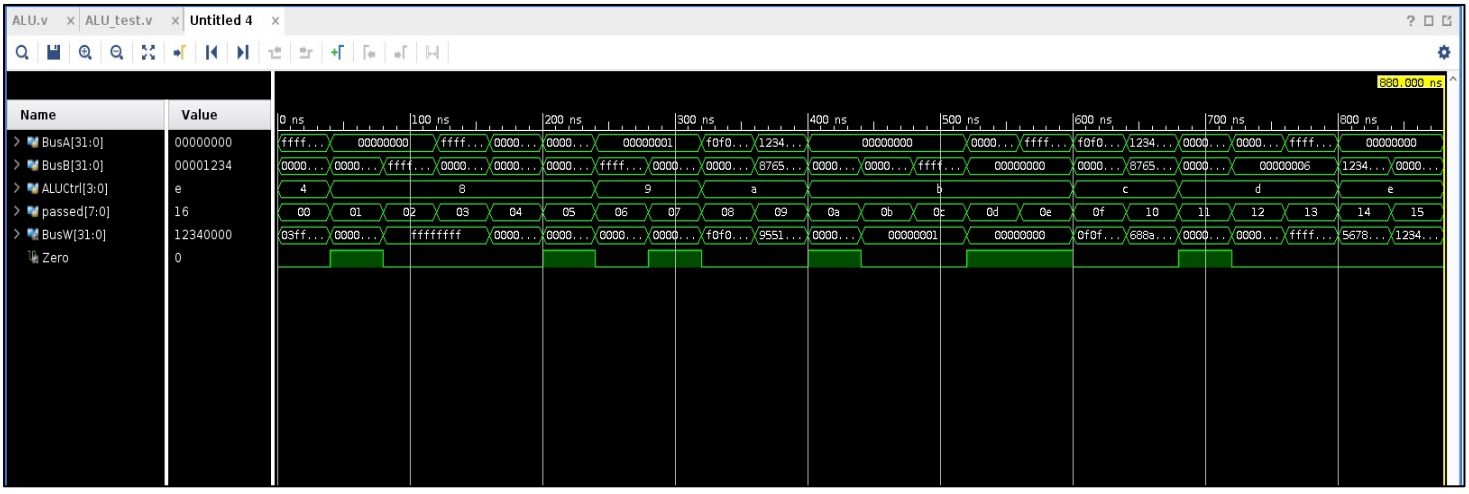
```
`timescale 1ns / 1ps

`define STRLEN 32
module ALUTest_v;
	task passTest;
		input [32:0] actualOut, expectedOut;
		input [`STRLEN*8:0] testType;
		inout [7:0] passed;
		if(actualOut == expectedOut) begin $display ("%s passed", testType); passed =
passed + 1; end
		else $display ("%s failed: %x should be %x", testType, actualOut,
expectedOut);
	endtask
	task allPassed;
		input [7:0] passed;
		input [7:0] numTests;
		if(passed == numTests) $display ("All tests passed");
		else $display("Some tests failed");
	endtask
	// Inputs
	reg [31:0] BusA;
	reg [31:0] BusB;
	reg [3:0] ALUCtrl;
	reg [7:0] passed;
```

```
	// Outputs
	wire [31:0] BusW;
	wire Zero;

	// Instantiate the Unit Under Test (UUT)
	ALU uut (
		.BusW(BusW),
		.Zero(Zero),
		.BusA(BusA),
		.BusB(BusB),
		.ALUCtrl(ALUCtrl)
	);
	initial begin
		// Initialize Inputs
		BusA = 0;
		BusB = 0;
		ALUCtrl = 0;
		passed = 0;
		// Add stimulus here
		//ADD YOUR TEST VECTORS FROM THE PRELAB HERE
		{BusA, BusB, ALUCtrl} = {32'hFFFF1234, 32'd6 , 4'd4}; #40; passTest({Zero,
BusW}, 33'h003FFFC48, "SRL 0xFFFF1234,6", passed);
		{BusA, BusB, ALUCtrl} = {32'h00000000, 32'h00000000, 4'd8}; #40;
passTest({Zero, BusW}, 33'h100000000, "ADDU 0,0", passed);
		{BusA, BusB, ALUCtrl} = {32'h00000000, 32'hFFFFFFFF, 4'd8}; #40;
passTest({Zero, BusW}, 33'h0FFFFFFFF, "ADDU 0,-1", passed);
		{BusA, BusB, ALUCtrl} = {32'hFFFFFFFF, 32'h00000000, 4'd8}; #40;
passTest({Zero, BusW}, 33'h0FFFFFFFF, "ADDU -1,0", passed);
		{BusA, BusB, ALUCtrl} = {32'h000000FF, 32'h00000001, 4'd8}; #40;
passTest({Zero, BusW}, 33'h000000100, "ADDU FF,1", passed);
		{BusA, BusB, ALUCtrl} = {32'h00000000, 32'h00000000, 4'd8}; #40;
passTest({Zero, BusW}, 33'h100000000, "SUBU 0,0", passed);
		{BusA, BusB, ALUCtrl} = {32'h00000001, 32'hFFFFFFFF, 4'd9}; #40;
passTest({Zero, BusW}, 33'h000000002, "SUBU 1,-1", passed);
		{BusA, BusB, ALUCtrl} = {32'h00000001, 32'h00000001, 4'd9}; #40;
passTest({Zero, BusW}, 33'h100000000, "SUBU 1,1", passed);
		{BusA, BusB, ALUCtrl} = {32'hF0F0F0F0, 32'h0000FFFF, 4'd10}; #40;
passTest({Zero, BusW}, 33'h0F0F00F0F, "XOR 0xF0F0F0F0,0x0000FFFF", passed);
		{BusA, BusB, ALUCtrl} = {32'h12345678, 32'h87654321, 4'd10}; #40;
passTest({Zero, BusW}, 33'h095511559, "XOR 0x12345678,0x87654321", passed);
```

```
            {BusA, BusB, ALUCtrl} = {32'h00000000, 32'h00000000, 4'd11}; #40;
passTest({Zero, BusW}, 33'h100000000, "SLTU 0,0", passed);

            {BusA, BusB, ALUCtrl} = {32'h00000000, 32'h00000001, 4'd11}; #40;
passTest({Zero, BusW}, 33'h000000001, "SLTU 0,1", passed);

            {BusA, BusB, ALUCtrl} = {32'h00000000, 32'hFFFFFFFF, 4'd11}; #40;
passTest({Zero, BusW}, 33'h000000001, "SLTU 0,-1", passed);

            {BusA, BusB, ALUCtrl} = {32'h00000001, 32'h00000000, 4'd11}; #40;
passTest({Zero, BusW}, 33'h100000000, "SLTU 1,0", passed);

            {BusA, BusB, ALUCtrl} = {32'hFFFFFFFF, 32'h00000000, 4'd11}; #40;
passTest({Zero, BusW}, 33'h100000000, "SLTU -1,0", passed);

            {BusA, BusB, ALUCtrl} = {32'hF0F0F0F0, 32'h0000FFFF, 4'd12}; #40;
passTest({Zero, BusW}, 33'h00F0F0000, "NOR 0xF0F0F0F0,0x0000FFFF", passed);

            {BusA, BusB, ALUCtrl} = {32'h12345678, 32'h87654321, 4'd12}; #40;
passTest({Zero, BusW}, 33'h0688aa886, "NOR 0x12345678,0x87654321", passed);

            {BusA, BusB, ALUCtrl} = {32'h00000001, 32'd3, 4'd13}; #40; passTest({Zero,
BusW}, 33'h100000000, "SRA 0x00000001,3", passed);

            {BusA, BusB, ALUCtrl} = {32'h00001234, 32'd6, 4'd13}; #40; passTest({Zero,
BusW}, 33'h000000048, "SRA 0x00001234,6", passed);

            {BusA, BusB, ALUCtrl} = {32'hFFFF1234, 32'd6, 4'd13}; #40; passTest({Zero,
BusW}, 33'h0FFFFFC48, "SRA 0xFFFF1234,6", passed);

            {BusA, BusB, ALUCtrl} = {32'h0, 32'h12345678, 4'd14}; #40; passTest({Zero,
BusW}, 33'h056780000, "LUI 0x12345678", passed);

            {BusA, BusB, ALUCtrl} = {32'h0, 32'h00001234, 4'd14}; #40; passTest({Zero,
BusW}, 33'h012340000, "LUI 0x00001234", passed);

            allPassed(passed, 22);

      end

endmodule
```

(2) Simulate the operation of your hardware using ISE and the ALU test bench provided.

Simulation Output Waveform: (ALU module)



Simulation Output Logs: (ALU module)

```
INFO: [USF-XSim-4] XSim::Simulate design

INFO: [USF-XSim-61] Executing 'SIMULATE' step in
'/home/grads/p/pranav_anantharam/lab4_control/project_2/project_2.sim/sim_1/behav/xsim'

INFO: [USF-XSim-98] *** Running xsim

   with args "ALUControlTest_v_behav -key {Behavioral:sim_1:Functional:ALUControlTest_v} -
tclbatch {ALUControlTest_v.tcl} -log {simulate.log}"

INFO: [USF-XSim-8] Loading simulator feature

Vivado Simulator 2017.4

Time resolution is 1 ps

source ALUControlTest_v.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
type 'create_wave_config' in the TCL console."

#   }

# }

# run 1000ns

                  SLL Instruction passed

                  SRL Instruction passed

                  SRA Instruction passed

                  ADD Instruction passed

                 ADDU Instruction passed

                  SUB Instruction passed

                 SUBU Instruction passed

                  AND Instruction passed

                   OR Instruction passed

                  XOR Instruction passed

                  NOR Instruction passed

                  SLT Instruction passed

                 SLTU Instruction passed

                 ANDI Instruction passed

                  ORI Instruction passed

                 ADDI Instruction passed

                 SUBI Instruction passed
```

```
                 SLTI Instruction passed
                ADDIU Instruction passed
                SUBIU Instruction passed
                 XORI Instruction passed
                 SLTU Instruction passed
                 NORI Instruction passed
                  LUI Instruction passed
All tests passed
```

(3) Synthesize your design. Provide the summary results of the synthesis process.

Synthesis Report: (ALU module)

```
---------------------------------------------------------------------------------
Start Writing Synthesis Report
---------------------------------------------------------------------------------

Report BlackBoxes:
+-+--------------+----------+
| |BlackBox name |Instances |
+-+--------------+----------+
+-+--------------+----------+

Report Cell Usage:
+------+-------+------+
|      |Cell   |Count |
+------+-------+------+
|1     |CARRY4 |    24|
|2     |LUT1   |     1|
|3     |LUT2   |    83|
|4     |LUT3   |    38|
|5     |LUT4   |   105|
|6     |LUT5   |   131|
|7     |LUT6   |   280|
|8     |MUXF7  |    15|
|9     |IBUF   |    68|
|10    |OBUF   |    33|
+------+-------+------+
```

Report Instance Areas:

```
+------+---------+-------+------+
|      |Instance |Module |Cells |
+------+---------+-------+------+
|1     |top      |       |   778|
+------+---------+-------+------+
```

Finished Writing Synthesis Report : Time (s): cpu = 00:00:38 ; elapsed = 00:02:38 . Memory (MB): peak = 1782.027 ; gain = 449.836 ; free physical = 332 ; free virtual = 3213

Synthesis finished with 0 errors, 0 critical warnings and 0 warnings.

Synthesis Optimization Runtime : Time (s): cpu = 00:00:38 ; elapsed = 00:02:38 . Memory (MB): peak = 1782.027 ; gain = 449.836 ; free physical = 331 ; free virtual = 3216

Synthesis Optimization Complete : Time (s): cpu = 00:00:38 ; elapsed = 00:02:38 . Memory (MB): peak = 1782.031 ; gain = 449.836 ; free physical = 330 ; free virtual = 3222

INFO: [Project 1-571] Translating synthesized netlist

INFO: [Netlist 29-17] Analyzing 107 Unisim elements for replacement

INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds

INFO: [Project 1-570] Preparing netlist for logic optimization

INFO: [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).

INFO: [Project 1-111] Unisim Transformation Summary:

No Unisim elements were transformed.

INFO: [Common 17-83] Releasing license: Synthesis

14 Infos, 0 Warnings, 0 Critical Warnings and 0 Errors encountered.

synth_design completed successfully

synth_design: Time (s): cpu = 00:00:41 ; elapsed = 00:02:42 . Memory (MB): peak = 1910.051 ; gain = 632.613 ; free physical = 233 ; free virtual = 3190

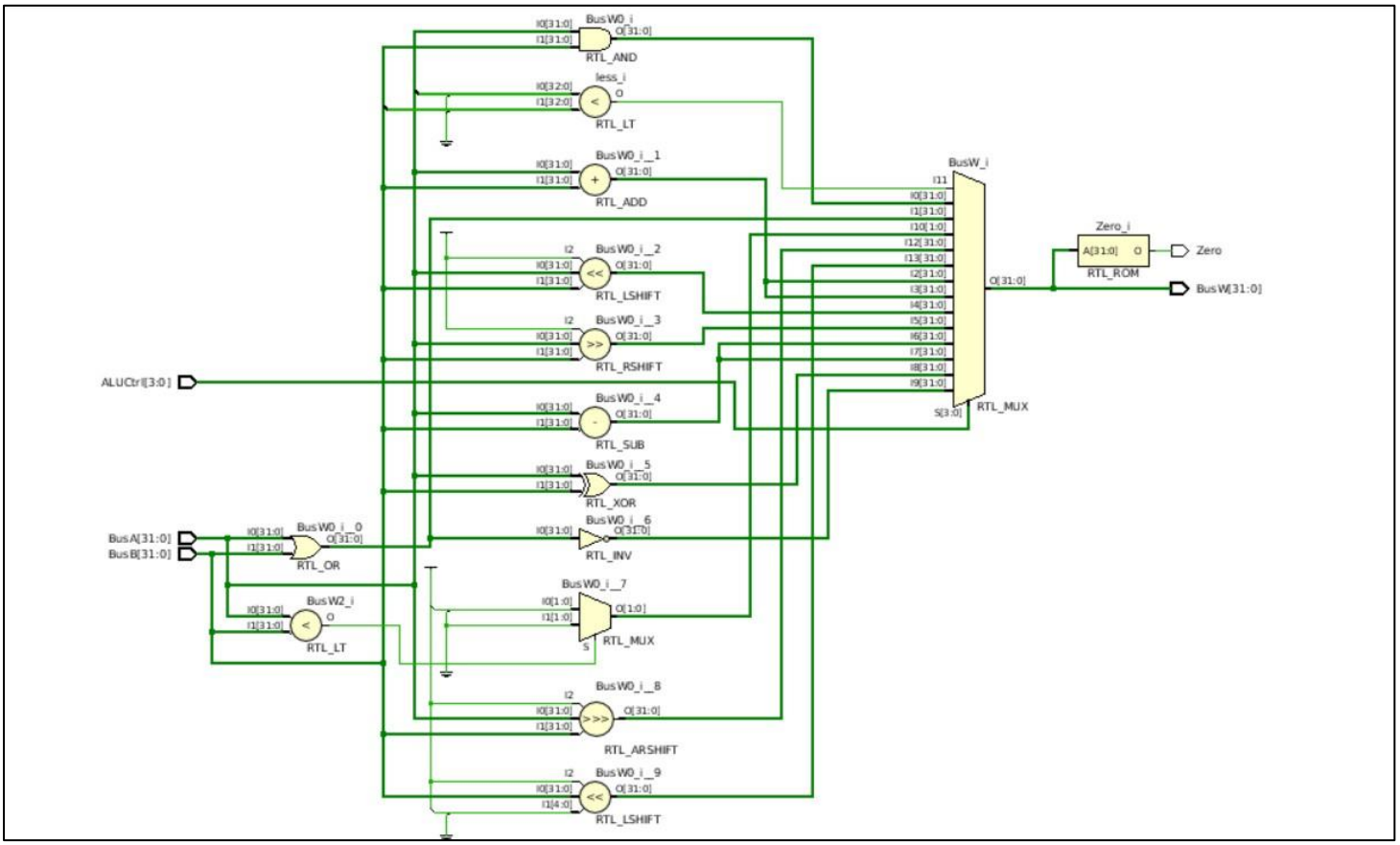
INFO: [Common 17-1381] The checkpoint '/home/grads/p/pranav_anantharam/lab4_alu/project_1/project_1.runs/synth_1/ALU.dcp' has been generated.

INFO: [runtcl-4] Executing : report_utilization -file ALU_utilization_synth.rpt -pb ALU_utilization_synth.pb

report_utilization: Time (s): cpu = 00:00:00.31 ; elapsed = 00:00:00.40 . Memory (MB): peak = 1934.074 ; gain = 0.000 ; free physical = 215 ; free virtual = 3190

INFO: [Common 17-206] Exiting Vivado at Fri Sep 29 15:14:23 2023...

Compiled Hardware Schematic: (ALU module)



---

2. Design the ALU control logic.

(1) Describe the ALU control logic in Verilog.

Design Source code filename: ALUControl.v

```
`timescale 1ns / 1ps

// MACROs
`define AND         4'b0000
`define OR          4'b0001
`define ADD         4'b0010
`define SLL         4'b0011
`define SRL         4'b0100
```

```
`define SUB        4'b0110
`define SLT        4'b0111
`define ADDU       4'b1000
`define SUBU       4'b1001
`define XOR        4'b1010
`define SLTU       4'b1011
`define NOR        4'b1100
`define SRA        4'b1101
`define LUI        4'b1110

`define SLLFunc  6'b000000
`define SRLFunc  6'b000010
`define SRAFunc  6'b000011
`define ADDFunc  6'b100000
`define ADDUFunc 6'b100001
`define SUBFunc  6'b100010
`define SUBUFunc 6'b100011
`define ANDFunc  6'b100100
`define ORFunc   6'b100101
`define XORFunc  6'b100110
`define NORFunc  6'b100111
`define SLTFunc  6'b101010
`define SLTUFunc 6'b101011
`define MULAFunc 6'b111000

module ALUControl(ALUCtrl, ALUop, FuncCode);

// Inputs
input [3:0] ALUop;
input [5:0] FuncCode;
// Outputs
output reg [3:0] ALUCtrl;

always@(*) begin
      // If ALUop is not equal to 4'b1111, ALUop is passed directly to the ALU
      if( ALUop != 4'b1111 )
            ALUCtrl <= ALUop;
```

```
        else
                // Function code is used to determine the ALU control code
                case(FuncCode)
                        `SLLFunc: ALUCtrl <= SLL;
                        `SRLFunc: ALUCtrl <= SRL;
                        `SRAFunc: ALUCtrl <= SRA;
                        `ADDFunc: ALUCtrl <= ADD;
                        `ADDUFunc: ALUCtrl <= ADDU;
                        `SUBFunc: ALUCtrl <= SUB;
                        `SUBUFunc: ALUCtrl <= SUBU;
                        `ANDFunc: ALUCtrl <= AND;
                        `ORFunc: ALUCtrl <= OR;
                        `XORFunc: ALUCtrl <= XOR;
                        `NORFunc: ALUCtrl <= NOR;
                        `SLTFunc: ALUCtrl <= SLT;
                        `SLTUFunc: ALUCtrl <= SLTU;
                default: ALUCtrl <= 4'bx;
                endcase
end
endmodule
```

Testbench code filename: ALUControlTest.v

```
`timescale 1ns / 1ps

`define AND         4'b0000
`define OR          4'b0001
`define ADD         4'b0010
`define SLL         4'b0011
`define SRL         4'b0100
`define MULA        4'b0101
`define SUB         4'b0110
`define SLT         4'b0111
`define ADDU        4'b1000
`define SUBU        4'b1001
`define XOR         4'b1010
```

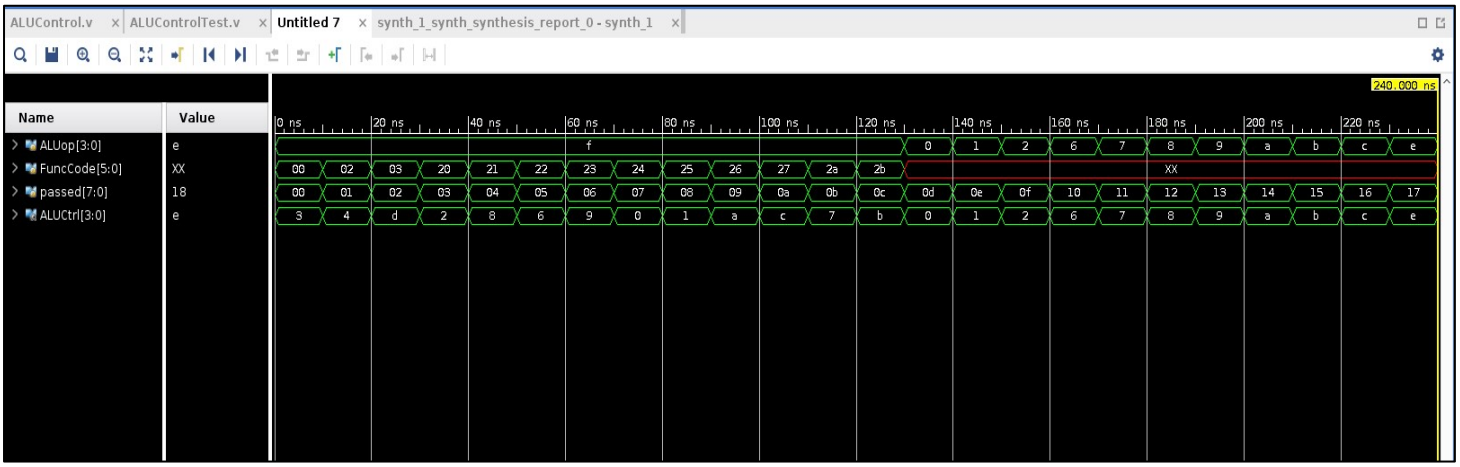
```
`define SLTU       4'b1011
`define NOR        4'b1100
`define SRA        4'b1101
`define LUI        4'b1110

`define SLLFunc  6'b000000
`define SRLFunc  6'b000010
`define SRAFunc  6'b000011
`define ADDFunc  6'b100000
`define ADDUFunc 6'b100001
`define SUBFunc  6'b100010
`define SUBUFunc 6'b100011
`define ANDFunc  6'b100100
`define ORFunc   6'b100101
`define XORFunc  6'b100110
`define NORFunc  6'b100111
`define SLTFunc  6'b101010
`define SLTUFunc 6'b101011
`define MULAFunc 6'b111000

`define STRLEN 32
module ALUControlTest_v;

      task passTest;
            input [5:0] actualOut, expectedOut;
            input [`STRLEN*8:0] testType;
            inout [7:0] passed;

            if(actualOut == expectedOut) begin $display ("%s passed", testType); passed =
passed + 1; end
            else $display ("%s failed: %d should be %d", testType, actualOut,
expectedOut);
      endtask

      task allPassed;
            input [7:0] passed;
            input [7:0] numTests;
```

```
        if(passed == numTests) $display ("All tests passed");
        else $display("Some tests failed");
endtask

// Inputs
reg [3:0] ALUop;
reg [5:0] FuncCode;
reg [7:0] passed;

// Outputs
wire [3:0] ALUCtrl;

// Instantiate the Unit Under Test (UUT)
ALUControl uut (
        .ALUCtrl(ALUCtrl),
        .ALUop(ALUop),
        .FuncCode(FuncCode)
);

initial begin
        // Initialize Inputs
        passed = 0;

        {ALUop, FuncCode} = {4'b1111, `SLLFunc};#10
        passTest(ALUCtrl, `SLL, "SLL Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `SRLFunc};#10
        passTest(ALUCtrl, `SRL, "SRL Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `SRAFunc};#10
        passTest(ALUCtrl, `SRA, "SRA Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `ADDFunc};#10
        passTest(ALUCtrl, `ADD, "ADD Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `ADDUFunc};#10
        passTest(ALUCtrl, `ADDU, "ADDU Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `SUBFunc};#10
        passTest(ALUCtrl, `SUB, "SUB Instruction", passed);
        {ALUop, FuncCode} = {4'b1111, `SUBUFunc};#10
        passTest(ALUCtrl, `SUBU, "SUBU Instruction", passed);
```

```
            {ALUop, FuncCode} = {4'b1111, `ANDFunc};#10
            passTest(ALUCtrl, `AND, "AND Instruction", passed);
            {ALUop, FuncCode} = {4'b1111, `ORFunc};#10
            passTest(ALUCtrl, `OR, "OR Instruction", passed);
            {ALUop, FuncCode} = {4'b1111, `XORFunc};#10
            passTest(ALUCtrl, `XOR, "XOR Instruction", passed);
            {ALUop, FuncCode} = {4'b1111, `NORFunc};#10
            passTest(ALUCtrl, `NOR, "NOR Instruction", passed);
            {ALUop, FuncCode} = {4'b1111, `SLTFunc};#10
            passTest(ALUCtrl, `SLT, "SLT Instruction", passed);
            {ALUop, FuncCode} = {4'b1111, `SLTUFunc};#10
            passTest(ALUCtrl, `SLTU, "SLTU Instruction", passed);
            {ALUop, FuncCode} = {`AND, 6'bXXXXXX};#10
            passTest(ALUCtrl, `AND, "ANDI Instruction", passed);
            {ALUop, FuncCode} = {`OR, 6'bXXXXXX};#10
            passTest(ALUCtrl, `OR, "ORI Instruction", passed);
            {ALUop, FuncCode} = {`ADD, 6'bXXXXXX};#10
            passTest(ALUCtrl, `ADD, "ADDI Instruction", passed);
            {ALUop, FuncCode} = {`SUB, 6'bXXXXXX};#10
            passTest(ALUCtrl, `SUB, "SUBI Instruction", passed);
            {ALUop, FuncCode} = {`SLT, 6'bXXXXXX};#10
            passTest(ALUCtrl, `SLT, "SLTI Instruction", passed);
            {ALUop, FuncCode} = {`ADDU, 6'bXXXXXX};#10
            passTest(ALUCtrl, `ADDU, "ADDIU Instruction", passed);
            {ALUop, FuncCode} = {`SUBU, 6'bXXXXXX};#10
            passTest(ALUCtrl, `SUBU, "SUBIU Instruction", passed);
            {ALUop, FuncCode} = {`XOR, 6'bXXXXXX};#10
            passTest(ALUCtrl, `XOR, "XORI Instruction", passed);
            {ALUop, FuncCode} = {`SLTU, 6'bXXXXXX};#10
            passTest(ALUCtrl, `SLTU, "SLTU Instruction", passed);
            {ALUop, FuncCode} = {`NOR, 6'bXXXXXX};#10
            passTest(ALUCtrl, `NOR, "NORI Instruction", passed);
            {ALUop, FuncCode} = {`LUI, 6'bXXXXXX};#10
            passTest(ALUCtrl, `LUI, "LUI Instruction", passed);
            allPassed(passed, 24);
        end
endmodule
```

(2) Simulate the operation of your hardware using ISE and the ALU Control test bench.

Simulation Output Waveform: (ALU Control Logic)



Simulation Output Logs: (ALU Control Logic)

```
INFO: [USF-XSim-96] XSim completed. Design snapshot 'ALUControlTest_v_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 = 7931.559
; gain = 7.000 ; free physical = 396 ; free virtual = 3834

restart

INFO: [Simtcl 6-17] Simulation restarted

run all

                  SLL Instruction passed

                  SRL Instruction passed

                  SRA Instruction passed

                  ADD Instruction passed

                 ADDU Instruction passed

                  SUB Instruction passed

                 SUBU Instruction passed

                  AND Instruction passed

                   OR Instruction passed

                  XOR Instruction passed

                  NOR Instruction passed

                  SLT Instruction passed

                 SLTU Instruction passed
```

```
                 ANDI Instruction passed
                  ORI Instruction passed
                 ADDI Instruction passed
                 SUBI Instruction passed
                 SLTI Instruction passed
                ADDIU Instruction passed
                SUBIU Instruction passed
                 XORI Instruction passed
                 SLTU Instruction passed
                 NORI Instruction passed
                  LUI Instruction passed
All tests passed
```

(3) Synthesize your design. Provide the summary results of the synthesis process.

Synthesis Report: (ALU Control Logic)

```
---------------------------------------------------------------------------------
Start Writing Synthesis Report
---------------------------------------------------------------------------------

Report BlackBoxes:
+-+--------------+----------+
| |BlackBox name |Instances |
+-+--------------+----------+
+-+--------------+----------+

Report Cell Usage:
+------+-----+------+
|      |Cell |Count |
+------+-----+------+
|1     |LUT4 |     1|
|2     |LUT5 |     5|
|3     |LUT6 |     1|
|4     |IBUF |     9|
|5     |OBUF |     4|
+------+-----+------+
```

Report Instance Areas:

```
+------+---------+-------+------+
|      |Instance |Module |Cells |
+------+---------+-------+------+
|1     |top      |       |    20|
+------+---------+-------+------+
```

Finished Writing Synthesis Report : Time (s): cpu = 00:00:35 ; elapsed = 00:02:36 . Memory (MB): peak = 1744.027 ; gain = 411.832 ; free physical = 267 ; free virtual = 3436

Synthesis finished with 0 errors, 0 critical warnings and 0 warnings.

Synthesis Optimization Runtime : Time (s): cpu = 00:00:35 ; elapsed = 00:02:36 . Memory (MB): peak = 1744.027 ; gain = 411.832 ; free physical = 264 ; free virtual = 3439

Synthesis Optimization Complete : Time (s): cpu = 00:00:35 ; elapsed = 00:02:36 . Memory (MB): peak = 1744.027 ; gain = 411.832 ; free physical = 263 ; free virtual = 3446

INFO: [Project 1-571] Translating synthesized netlist

INFO: [Netlist 29-17] Analyzing 9 Unisim elements for replacement

INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds

INFO: [Project 1-570] Preparing netlist for logic optimization

INFO: [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).

INFO: [Project 1-111] Unisim Transformation Summary:

No Unisim elements were transformed.

INFO: [Common 17-83] Releasing license: Synthesis

12 Infos, 0 Warnings, 0 Critical Warnings and 0 Errors encountered.

synth_design completed successfully

synth_design: Time (s): cpu = 00:00:38 ; elapsed = 00:02:39 . Memory (MB): peak = 1882.164 ; gain = 604.723 ; free physical = 235 ; free virtual = 3391

INFO: [Common 17-1381] The checkpoint '/home/grads/p/pranav_anantharam/lab4_control/project_2/project_2.runs/synth_1/ALUControl.dcp' has been generated.

INFO: [runtcl-4] Executing : report_utilization -file ALUControl_utilization_synth.rpt -pb ALUControl_utilization_synth.pb

report_utilization: Time (s): cpu = 00:00:00.30 ; elapsed = 00:00:00.38 . Memory (MB): peak = 1906.188 ; gain = 0.000 ; free physical = 217 ; free virtual = 3390

INFO: [Common 17-206] Exiting Vivado at Fri Sep 29 15:36:33 2023...

Compiled Hardware Schematic: (ALU Control Logic)