**ECEN 323 – Winter 2020**

Lab 6: RISC-V Control

Johnson, Ryan

Section 1

Preliminary

| **Type** | **Instruction** |
| --- | --- |
| Arithmetic | add, sub, slt |
| Arithmetic Immediate | addi, slti |
| Logical | and, or, xor |
| Logical Immediate | andi, ori, xori |
| Branch | beq |
| Memory | lw, sw |

**ALUCtrl**

Complete the ALU Control Table

| **Instruction** | **ALU Control** | **Decoding** |
| --- | --- | --- |
| and | 4’b0000 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b111) |
| or | 4’b0001 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b110) |
| xor | 4’b1100 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b100) |
| add | 4’b0010 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b000) && (i[30]==1’b0) |
| sub | 4’b0110 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b000) && (i[30]==1’b1) |
| slt | 4’b0111 | (i[6:0] == 7'b0110011) && (i[14:12] == 3'b010) |
| addi | 4’b0010 | (i[6:0] == 7’b0010011) && (i[14:12] == 3’b000) |
| slti | 4’b0111 | (i[6:0] == 7’b0010011) && (i[14:12] == 3’b010) |
| andi | 4’b0000 | (i[6:0] == 7’b0010011) && (i[14:12] == 3’b111) |
| ori | 4’b0001 | (i[6:0] == 7’b0010011) && (i[14:12] == 3’b110) |
| xori | 4’b1100 | (i[6:0] == 7’b0010011) && (i[14:12] == 3’b100) |
| beq | 4’b0110 | (i[6:0] == 7’b1100011) && (i[14:12] == 3’b000) |
| lw | 4’b0010 | (i[6:0] == 7’b0000011) && (i[14:12] == 3’b010) |
| sw | 4’b0010 | (i[6:0] == 7’b0100011) && (i[14:12] == 3’b011) |

**ALUSrc**

Determine those instructions in which the ALUSrc signal must be set to 1

addi, stli, andi, ori, xori, lw, sw

**RegWrite**

Determine the instructions in which you should not write to the register file (i.e., RegWrite=0)

beq, sw

**PCSrc**

Provide the dataflow Verilog statements needed to generate the PCSrc signal

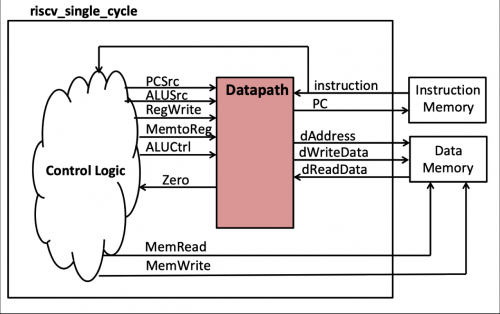
assign PCSrc = ((instruction[6:0] == 7'b1100011) && Zero) ? 1’b1 : 1’b0;

//DOUBLE CHECK THIS BEFORE SUBMISSION

**Memory Control**

Complete the following Memory control signal table

|  |  |
| --- | --- |
| **Signal** | **Instructions that need this signal set to 1** |
| MemtoReg | lw |
| MemRead | lw |
| MemWrite | sw |



Exercise #1

Include the Verilog code for your single-cycle RISC-V controller in your laboratory report.

`timescale 1ns / 1ps

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Module: riscv\_simple\_datapath

\*

\* Author: Ryan Johnson

\* Class: ECEN 323, Section 01, Winter 2020

\* Date: 11 Feb 2020

\*

\* Description: acts as a control that implements one instruction at a time

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

module riscv\_single\_cycle #(parameter INITIAL\_PC = 32'h00400000) (

input wire logic clk,

input wire logic rst,

input wire logic [31:0] instruction,

input wire logic [31:0] dReadData,

output logic [31:0] PC,

output logic [31:0] dAddress,

output logic [31:0] dWriteData,

output logic MemRead,

output logic MemWrite

);

logic PCSrc;

logic ALUSrc;

logic MemtoReg;

logic RegWrite;

logic Zero;

logic [3:0] ALUCtrl;

riscv\_simple\_datapath d212(.clk(clk), .rst(rst), .instruction(instruction),

.PCSrc(PCSrc), .ALUSrc(ALUSrc), .RegWrite(RegWrite),

.ALUCtrl(ALUCtrl), .PC(PC), .Zero(Zero),

.dAddress(dAddress), .dWriteData(dWriteData),

.dReadData(dReadData), .MemtoReg(MemtoReg));

localparam AND = 4'b0000;

localparam OR = 4'b0001;

localparam ADD = 4'b0010;

localparam SUB = 4'b0110;

localparam SLT = 4'b0111;

localparam XOR = 4'b1100;

logic store;

logic load;

logic branch;

logic itype;

logic [6:0] opcode;

logic [2:0] funct3;

assign opcode = instruction[6:0];

assign funct3 = instruction[14:12];

assign store = (opcode == 7'b0100011) ? 1'b1 : 1'b0;

assign load = (opcode == 7'b0000011) ? 1'b1 : 1'b0;

assign branch = (opcode == 7'b1100011) ? 1'b1 : 1'b0;

assign itype = (opcode == 7'b0010011) ? 1'b1 : 1'b0;

assign ALUCtrl = (funct3 == 3'b111) ? AND :

(funct3 == 3'b110) ? OR :

(funct3 == 3'b100) ? XOR :

(store || load) ? ADD :

(funct3 == 3'b010) ? SLT :

(branch) ? SUB :

(itype) ? ADD :

(instruction[30] == 1'b1) ? SUB : ADD;

always\_comb

begin

ALUSrc = itype | load | store;

PCSrc = branch & Zero;

MemtoReg = load;

MemRead = load;

MemWrite = store;

RegWrite = (!branch & !store);

end

endmodule

Exercise #2

Copy the console output for your successful testbench

===== RISCV SINGLE CYCLE TB V1.4 =====

run -all

[0ns] Testing Reset

[100ns] Testing x0

[120ns] Loading Initial Values

INFO: [USF-XSim-96] XSim completed. Design snapshot 'riscv\_single\_cycle\_tb\_behav' loaded.

INFO: [USF-XSim-97] XSim simulation ran for 1000ns

launch\_simulation: Time (s): cpu = 00:00:03 ; elapsed = 00:00:21 . Memory (MB): peak = 1093.238 ; gain = 0.000

run all

[1080ns] Directed Testing

[1000ns] Testing 1000 random commands

You Passed!

$finish called at time : 12830 ns : File "J:/ECEN 323 Labs/project\_7/riscv\_single\_cycle\_tb.sv" Line 694

Exercise #3

Include a copy of the testbench console output in your laboratory report

run all

[1010ns] 0x0040007c: beq x20,x0,8

[1020ns] 0x00400084: lw x20,0(x19)

[1030ns] 0x00400088: lw x20,4(x19)

[1040ns] 0x0040008c: lw x20,8(x19)

[1050ns] 0x00400090: addi x20,x19,16

[1060ns] 0x00400094: lw x21,-8(x20)

[1070ns] 0x00400098: lw x21,-4(x20)

[1080ns] 0x0040009c: lw x21,8(x20)

[1090ns] 0x004000a0: sw x2,-16(x20)

[1100ns] 0x004000a4: sw x6,4(x19)

[1110ns] 0x004000a8: lw x22,-16(x20)

[1120ns] 0x004000ac: beq x22,x2,8

[1130ns] 0x004000b4: lw x23,4(x19)

[1140ns] 0x004000b8: beq x23,x6,8

You Passed!

$finish called at time : 1140 ns : File "J:/ECEN 323 Labs/project\_7/project\_7.srcs/sim\_2/imports/project\_7/single\_cycle\_mem\_tb.sv" Line 740

Exercise #4

Summarize and justify any warnings you had during synthesis. If you did not have any warnings, say such in your laboratory report.

Same as last time, shows a warning for instruction ports that aren’t used riscv\_simple\_datapath, but as I do not need to use them, it is ok.

Summarize the estimated resources for your synthesized logic in the table below.

| **Resource** | **Estimation** |
| --- | --- |
| LUT | 317 |
| FF | 32 |
| IO | 164 |
| BUFG | 1 |

How many hours did you work on the lab?

2 hours

Please provide any suggestions for improving this lab in the future:

None. Great stuff.