




Lab 10 – Designing a Control Unit of a RISC Processor

Objectives

In this lab, we will develop an instruction-fetch datapath and verify its functionality.

Section	
a) <u>Introduction</u>  A brief overview of this lab.	10
b) <u>Implementation</u>  In this section, you will implement a control unit of a RISC-V processor.	90





a. Introduction

The last module we are left with is the control unit, which we have to design before we proceed further to integrate the already-designed processor components. In this lab, we will develop a module for generating control signals for specific instructions. These control signals are used to control the data flow and enabling or disabling the modules which are not needed for specific instructions.

b. Implementation

Besides control unit, we also have to develop ALU Control unit to set the control signals of ALU.

i. Lab Task 01

As shown in Fig. 10.1, Write a module, named `Control_Unit`, which takes a 7-bit wide input, named `Opcode`, and generate 7 output signals. Out of these seven outputs, one is `ALUOp` which is 2-bits wide, and the remaining six are 1-bit wide, which are `Branch`, `MemRead`, `MemtoReg`, `MemWrite`, `ALUSrc`, and `RegWrite`.

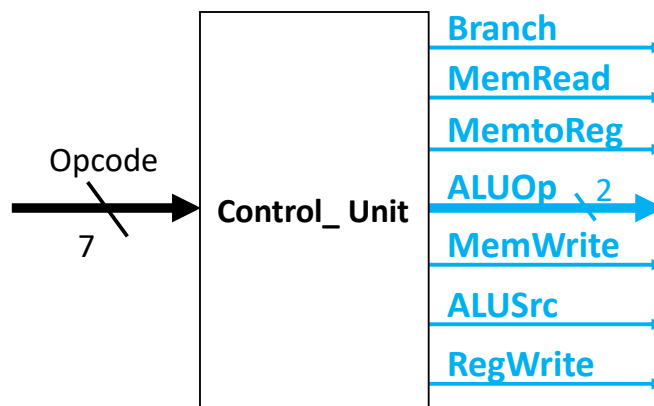


Fig. 10.1. I/O diagram of Control Unit.

The behavior of `Control_Unit` module should be designed according to the Table 10.1. The outputs depend only on the input, `Opcode`.

Table. 10.1. This table shows how the control signals are set based on the input values of `Opcode`.

Instruction Type	Opcode	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	ALUOp [1:0]
R-Type	0110011	0	0	1	0	0	0	10
I-Type (ld)	0000011	1	1	1	1	0	0	00
I-Type (sd)	0100011	1	X	0	0	1	0	00
SB-Type (Beq)	1100011	0	X	0	0	0	1	01





ii. Lab Task 02

Write a module, named `ALU_Control`, which takes a 2-bit input, named `ALUOp` and a 4-bit input, named `Funct`, and produces a 4-bit output, named `Operation`, as shown in Figure. 10.2.

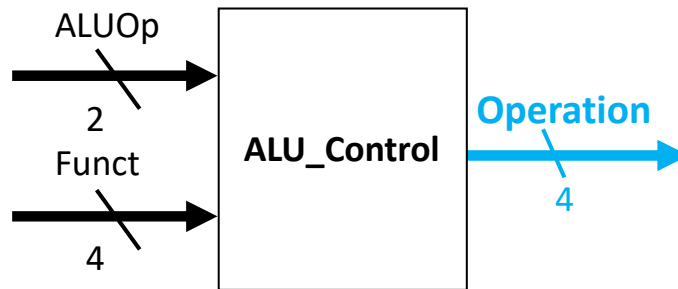


Fig. 10.2. I/O diagram of `ALU_Control` module.

The values of the output, `Operation`, should be set based on the input signals, `ALUOp`, as shown in Table 10.2.

Table. 10.2. The values of the output, `Operation`, based on the corresponding input, `ALUOp`, signals.

Instruction Type	ALUOp [1:0]	Funct	Operation
I/S-Type (ld, sd)	00	xxxx	0010
SB-Type (Beq)	01	xxxx	0110
R-Type	10	0000	0010
		1000	0110
		0111	0000
		0110	0001

iii. Lab Task 03

Integrate above two modules in a top module, named `top_control`. Write a testbench and perform verification.

Computer Architecture

Spring 20

Lab 10

Marks Distribution:

Task	LR2: Design/Code	LR5: Results
Task 1 (Control Unit)	10 Points	-
Task 2 (ALU Control)	10 Points	-
Task 3 (Top Control)	20 points	10 Points
Total		50 Points

Marks Obtained:

Task	LR2: Design/Code	LR5: Results
Task 1 (Control Unit)		-
Task 2 (ALU Control)		-
Task 3 (Top Control)		
Total		