Comp E 475

Microprocessors

Lab 5

Tornike Paitchadze

Contents

[Task Description 2](#_Toc22460835)

[Solution 2](#_Toc22460836)

[Simulation & Verification 2](#_Toc22460837)

[Comparison 3](#_Toc22460838)

[Conclusion 3](#_Toc22460839)

# Task Description

We had to modify our previous assignment include add memory instruction type and jump instruction type registers.

* **mem\_instr\_type**, 2 bits, values from 0 to 2
  + 1 if given Memory instruction type is of Immediate
  + 2 if it's "Register shifted by value" type
  + 0 if it's not identifiable
* **jmp\_instr\_type**, 2 bits , values from 0 to 2
  + 1 if given Jump instruction type is of Branch only
  + 2 if it's Branch and Link
  + 0 if it's not identifiable

# Solution

We can continue with the same logic that we did in the previous assignments. I added the following registers m\_type and J\_type. In order to finish the task I assigned jump type to existing data type.

The code :

`timescale 1ns / 1ps

module decoder(

input [31:0] ins,

output reg [1:0] ins\_type,

output reg [1:0] m\_type,

output reg [1:0] j\_type

output reg [2:0] d\_type,

);

always @(\*) begin

case(ins[27:26])

00: begin

ins\_type = 2'b01;

d\_type = ins[25] ? 3'b001 :

!ins[25] && !ins[4] ? 3'b010 :

ins[25] && ins[7] && ins [4] ? 3'b011 :

!ins[25] && !ins[24] && !ins[7] &&

ins[6] && ins[5] && !ins[4] ? 3'b100 : 3'b000;

end

01: ins\_type=2'b10;

10: begin

ins\_type=2'b11; //branch

j\_type = ins[25] && !ins[24] ? 2'b01 :

ins[25] && ins[24] ? 2'b10 : 2'b00;

end

default: ins\_type=2'b00;

endcase

m\_type = d\_type == 1 ? 1 :

d\_type == 2 ? 2 : 0;

end

endmodule

# Simulation & Verification

# Comparison

Was a bit harder than previous labs because we got introduced to git.

# Conclusion