Name: Laiba Fatima Khan

CA Lab 6 Report

TASK 1:

Multiplexer:

```
module multiplexer
(
	input sel,
	input [63:0] a,
	input [63:0] b,
	output reg [63:0] data_out
);
	always @(*)
	begin
		case ({sel})
			1'b0 : data_out <= b;
			1'b1 : data_out <= a;
		endcase
	end
endmodule
```

Testbench:

```
module tb
(

);
	reg [63:0] a;
	reg [63:0] b;
	reg sel;
	wire[63:0] q;

	multiplexer mux
	(
		.a(a),
		.b(b),
		.sel(sel),
		.data_out(q)

	);
	initial
	begin
```

```
                a = 64'd49;
                b = 64'd92;
                sel = 1'b0;
                #20
                sel = 1'b1;
                #20
                a = 64'd1;
        end

        initial
        $monitor ("Time = ", $time, "  a= ", a, "  b= ", b, "  select= ", sel,  " --------> DataOut= %d",
data_out);
endmodule
```

TASK 2:

Instruction Parser:

```
module instruction
(
        input [31:0] instruction,

        output reg [6:0] opcode, funct7, [4:0] rd, rs1, rs2, [2:0] funct3
);
        always @(instruction)
        begin
                opcode = instruction[6:0];
                rd = instruction[11:7];
                funct3 = instruction[14:12];
                rs1 = instruction[19:15];
                rs2 = instruction[24:16];
                funct7 = instruction[31:25];
        end
endmodule
```

Testbench:

```
module tb
(
);
        reg [31:0] instruction;
        wire[6:0] opcode, funct7;
        wire[4:0] rd, rs1, rs2;
        wire[2:0] funct3
        instruction ins
        (
```

```
                .instruction(instruction),
                .opcode(opcode),
                .funct7(funct7),
                .rd(rd),
                .rs1(rs1),
                .rs2(rs2),
                .funct3(funct3)
        );
        initial
        instruction = 32'b00000000100001001000010100110011; //add with rs2:16 rs1:17 rd:18
endmodule
```

TASK 3:

Immediate Data Generator:

```
module immediate_data_extractor
(
        input [31:0] instruction,

        output reg [63:0] imm_data
);
        always @(instruction)
        begin
                case(instruction[6:0])

                        7'b0000011 : imm_data <= {{52{instruction[31]}},instruction[31:20]}; //I type
Load
                        7'b0100011 : imm_data <= {{52{instruction[31]}}, instruction[31:25],
instruction[11:7]}; //S type Store
                        7'b1100111 : imm_data <=
{{51{instruction[31]}},instruction[31],instruction[7],instruction[30:25],instruction[11:8],1'b0};//R type
                endcase
        end

endmodule
```

Testbench:

```
module tb
(
);
        reg [31:0] instruction;
        wire[63:0] imm_data;
```

```
        immediate_data_extractor dExt
        (
                .instruction(instruction),
                .imm_data(imm_data)
        );
        initial
        begin
        instruction = 32'b00001111000001010011010010000011;
        end

endmodule
```