Name: Laiba Fatima Khan

CA Lab 11: Single Cycle RISC Processor

RISCV Processor:

```
module  RISC_V_Processor
(
	input clk, reset
);

   wire[63:0] PC_In_wire;
	wire[63:0] PC_Out_wire;
	wire[31:0] Instruction_Out_wire;
	wire [63:0] Adder1_Out_Wire;
	wire [63:0] Adder2_Out_Wire;
	wire [3:0] Operation_Wire;
	wire [63:0] MUX_Data_Out_Wire;
	wire [63:0] ALU_Result_Wire;
	wire [63:0] Immediate_Data_Wire;
	wire [6:0] opcode_wire, funct7_wire;
	wire [4:0] rd_wire, rs1_wire, rs2_wire;
	wire [2:0] funct3_wire;
	wire [63:0]ReadData1_wire, ReadData2_wire;
	wire Zero_Wire;
	wire [63:0] Register_WriteData_Wire;
	wire [63:0] DataMemory_Read_Data;

	wire Branch_wire, MemRead_wire, MemtoReg_wire, MemWrite_wire, ALUSrc_wire, RegWrite_wire;
	wire [1:0]ALUOp_wire;

Program_Counter PC
(
	.clk(clk),
	.reset(reset),
	.PC_In(PC_In_wire),
	.PC_Out(PC_Out_wire)
);

Adder Adder1
(
	.a(PC_Out_wire),
	.b(64'd4),
	.out(Adder1_Out_Wire)

);

Adder Adder2
(
```

```
        .a(PC_Out_wire),
        .b(Immediate_Data_Wire<<1),
        .out(Adder2_Out_Wire)
);
multiplexer Mux1
(
        .sel(Zero_Wire & Branch_wire),
        .b(Adder1_Out_Wire),
        .a(Adder2_Out_Wire),
        .data_out(PC_In_wire)
);

multiplexer Mux2
(
        .sel(ALUSrc_wire),
        .b(ReadData2_wire),
        .a(Immediate_Data_Wire),
        .data_out(MUX_Data_Out_Wire)
);

multiplexer Mux3
(
        .sel(MemtoReg_wire),
        .b(ALU_Result_Wire),
        .a(DataMemory_Read_Data),
        .data_out(Register_WriteData_Wire)
);

Instruction_Memory InsMem
(
        .Inst_Address(PC_Out_wire),
        .Instruction(Instruction_Out_wire)
);

instruction Ins
(
        .instruction(Instruction_Out_wire),
        .rs1(rs1_wire),
        .rs2(rs2_wire),
        .rd(rd_wire),
        .funct3(funct3_wire),
        .funct7(funct7_wire),
        .opcode(opcode_wire)
);

Control_Unit CtrlUnit
(
        .Opcode(opcode_wire),
        .Branch(Branch_wire),
```

```
        .MemRead(MemRead_wire),
        .MemtoReg(MemtoReg_wire),
        .MemWrite(MemWrite_wire),
        .ALUSrc(ALUSrc_wire),
        .RegWrite(RegWrite_wire),
        .ALUOp(ALUOp_wire)
);

registerFile RegFile
(
        .RS1(rs1_wire),
        .RS2(rs2_wire),
        .RD(rd_wire),
        .ReadData1(ReadData1_wire),
        .ReadData2(ReadData2_wire),
        .WriteData(Register_WriteData_Wire),
        .RegWrite(RegWrite_wire),
        .clk(clk),
        .reset(reset)
);

immediate_data_extractor DataExtract
(
        .instruction(Instruction_Out_wire),
        .imm_data(Immediate_Data_Wire)
);

ALU_Control ALUCtrl
(
        .ALUOp(ALUOp_wire),
        .Funct({Instruction_Out_wire[30], Instruction_Out_wire[14:12]}),
        .Operation(Operation_Wire)
);

ALU_64_bit ALU
(
        .a(ReadData1_wire),
        .b(MUX_Data_Out_Wire),
        .ALUop(Operation_Wire),
        .Result(ALU_Result_Wire),
        .ZERO(Zero_Wire)
);
Data_Memory DataMem
(
        .Mem_Addr(ALU_Result_Wire),
        .Write_Data(ReadData2_wire),
        .clk(clk),
        .Mem_Write(MemWrite_wire),
        .Mem_Read(MemRead_wire),
```

```
        .Read_Data(DataMemory_Read_Data)
);
endmodule
```

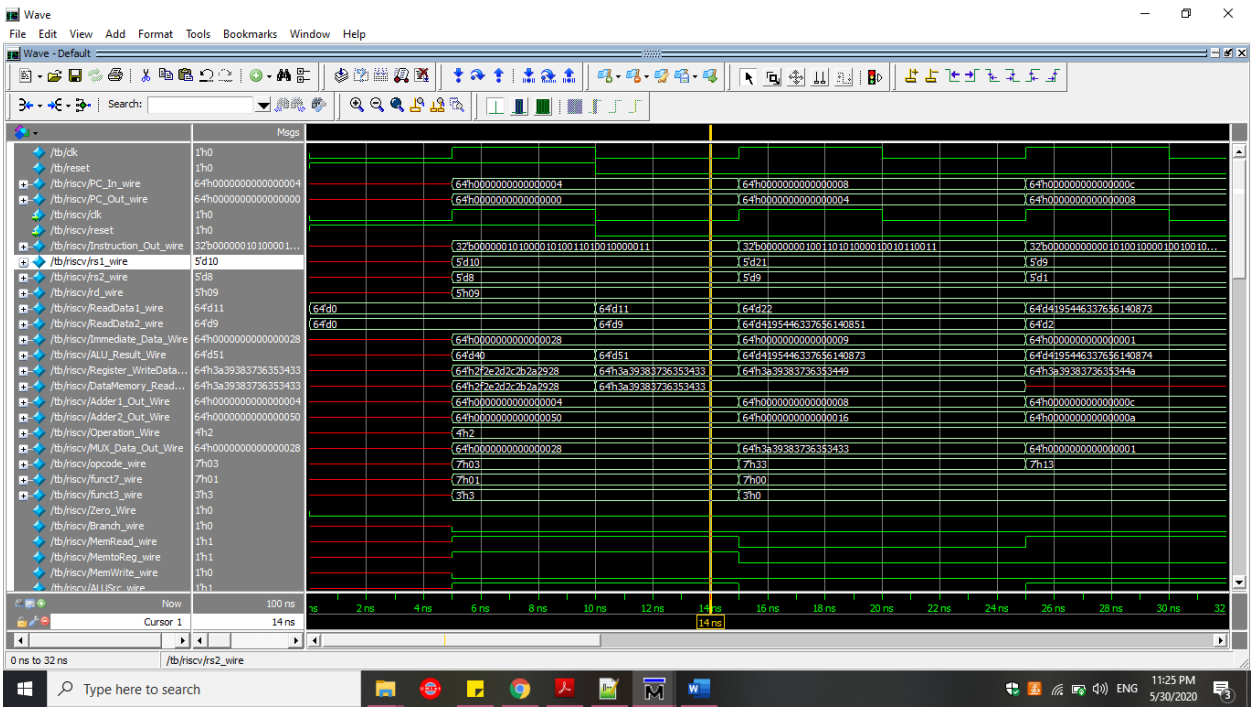
EXERCISE

1: Testbench:

```
module tb
(
);
reg clk;
reg reset;

RISC_V_Processor riscv
(
        .clk(clk),
        .reset(reset)
);

initial
begin
clk = 0;
reset = 1;

#10 reset = ~reset;
End

always
#5 clk = ~clk;

endmodule
```

TEST:

Clk – initial 0 then toggle ever 5ns (works)
Reset - high for first 10ns and low afterwards (works)
PC Out – increases by 4 with every new instruction (works)
Instruction – instruction outputs according to Instruction Memory. On PC_In 0, instruction 0 and so on. (works)
RD – for first three instructions: 9 (x9), for last (sd): imm[4:0]=8 (offset 8) (works)
RS1 – in instructions sequence: 10(x10), 21(x21), 9(x9), 10(x10) (works)
RS2 - in instructions sequence: 8(offset[4:0]) , 9(x9), 1 (imm[4:0], immediate value 1), 9(x9) (works)
ReadData1 – when rs1 = x10 then ReadData1 = 11(screenshot attached), correct acc to RegisterFile (works)
ReadData2 - – when rs2 = x8 then ReadData1 = 9(screenshot attached), correct acc to RegisterFile (works)
ALU Result – 40+11(ReadData1) = 51 (works)
RegFile WriteDate -
DataMemory ReadData -



2: DECODE INSTRUCTIONS (Assembly Code)

1. 00000010100001010011010010000011
   lw x9, 40(x10)

2. 00000000100110101000010010110011
   add x9, x21, x9

3. 00000000000101001000010010010011
   addi x9, x9, 1

4. 00000010100101010011010000100011
   sw x9, 8(x10)

3: DECODE INSTRUCTIONS (C Code)

1. 00000010100001010011010010000011
   lw x9, 40(x10)

   int A[]; //value at memory address stored in x10

   // [00000000000000000000000000000000000000000000000000000000000001011]

   int h=A[10]; // 40/4, because we are considering int array

2. 00000000100110101000010010110011
   add x9, x21, x9

   int h; //value in x9

int temp=22; //x21=22
h+=temp;

3. 00000000000101001000010010010011
addi x9, x9, 1
int h; //value in x9
h+=1;

4. 00000010100101010011010000100011
sw x9, 8(x10)
int h; //value in x9
int A; //array at memory address stored in x10
A[2]=h;