# Lab 07 – Worksheet

|  |  |  |
| --- | --- | --- |
| Name: Muhammad Haider, Muhammad Ali Anwar | ID:mh0845 | Section: |

## Task 1.

**Code: Design module & testbench**

*Provide appropriately commented code for designed module & its testbecnch, code should contain meaningful variable naming.*

*\*Add snippet or Copy paste the code written in the Editor window. Make sure the irrelevant area of the snip is cropped.*

|  |
| --- |
| **Module**  `timescale 1ns / 1ps  module registerfile(  input [63:0]WriteData,  input [4:0]RS1,  input [4:0]RS2,  input [4:0]RD,  input RegWrite, clk, reset,  output reg [63:0]ReadData1,  output reg [63:0]ReadData2  );  //making 32 regsiters, each 64 bit wide  reg [63:0] Registers [31:0];  integer i;  initial  //using for loop in the initial part to initialize the values of the registers  begin  for(i=0; i < 32; i=i+1)  begin  Registers[i] = i;  end  end  always@(posedge clk)  begin  //checking if RegWrite=1, so assign the value of WriteData to RD(destination register)  if(RegWrite==1)  begin  Registers[RD] = WriteData;  end    end  always @(\*)  begin  // setting ReadData1 and ReadData2 the value of 0 when reset is high  if(reset)  begin  ReadData1 = 0;  ReadData2 = 0;  end    else  // else setting value of RS1 and RS2 from Registers to ReadData1 and ReadData2, respectively  begin  ReadData1 = Registers[RS1];  ReadData2 = Registers[RS2];  end  end  endmodule  **Testbench**:  module tb\_RegisterFile ( );  reg [63:0]WriteData;  reg [4:0]RS1;  reg [4:0]RS2;  reg [4:0]RD;  reg RegWrite, clk, reset;  wire [63:0]ReadData1;  wire [63:0]ReadData2;    registerfile regFile  (  WriteData,RS1, RS2,  RD,  RegWrite,  clk,  reset,  ReadData1,  ReadData2  );  initial  begin  clk = 0;  RegWrite = 0;  reset = 1;  // add x20,x10,x20  RS1 = 10; //ReadData1 reads value of register number 10 and the value is 11  RS2 = 20; //ReadData2 reads value in register number 20 and the value loaded is 21    WriteData = 64'd32; //This value 32 is given by user for now, the add instruction given above is just to show how does the proces works  RD = 20; //The value of register number 20 was '21' . This 21 is the value or 'data'  #10 reset = 0;  #10 RegWrite = 1;//allow write in register 20  end  always  #10 clk=~clk;  endmodule |

## Results (Waveforms)

*\*Add snip of relevant signals’ waveforms. Make sure the irrelevant area of the snip is cropped.*

|  |
| --- |
|  |

## Comments

*\*Observation/Comments on the obtained results/working of code.*

|  |
| --- |
| The register file module takes two source registers (rs1, rs2) and a destination register (rd). It also takes Writedata, RegWrite, clk and reset from the control unit. According to their values, the ReadData1 and ReadData2 take the required values from the array of 32 registers in the module. |

## Exercise

**Code: Design module & testbench**

*Provide appropriately commented code for designed module & its testbecnch, code should contain meaningful variable naming.*

*\*Add snippet or Copy paste the code written in the Editor window. Make sure the irrelevant area of the snip is cropped.*

|  |
| --- |
| **Module**:  `timescale 1ns / 1ps  module top\_module(  input [31:0]instruction,  output wire [63:0]ReadData1,  output wire [63:0]ReadData2  );    wire [63:0]WriteData;  wire RegWrite, clk, reset;      wire [6:0]opcode;  wire [4:0]rd;  wire [2:0]funct3;  wire [4:0]rs1;  wire [4:0]rs2;  wire [6:0]funct7;    //calling instruction\_parser  instruction\_parser IP (instruction, opcode,rd,funct3,rs1,rs2,funct7);    //calling registerfile with rs1,rs2 and rd from the instruction parser fucntion  registerfile RF(WriteData, rs1,rs2,rd,RegWrite,clk,reset,ReadData1,ReadData2);    endmodule  **Testbench**:  `timescale 1ns / 1ps  module toplevelTB(  );  reg [31:0]instruction;  wire [63:0]ReadData1;  wire [63:0]ReadData2;  top\_module toppTB(  instruction,ReadData1,ReadData2  );    initial  begin  //giving binary of RISC V command : add x5,x6,x0  instruction=32'b00000000000000110000001010110011;  end  endmodule |

## Results (Waveforms)

*\*Add snip of relevant signals’ waveforms. Make sure the irrelevant area of the snip is cropped.*

|  |
| --- |
|  |

## Comments

*\*Observation/Comments on the obtained results/working of code.*

|  |
| --- |
| In the top level module, the instruction parser is given a 32 bit instruction and from that, it extracts the values of rs1, rs2 and rd and passes it to the registerfile module which then gives ReadData1 and ReadData2 as outputs of the top level module with the values of the registers in the RISC V command instruction. |

# Assessment Rubrics

|  |  |  |  |
| --- | --- | --- | --- |
| **Task No.** | **LR 2**    **Code** | **LR 5**    **Results** | **AR 7**    **Report Submission** |
| **Task 1** | /30 | /20 | /20 |
| **Task 2** | /20 | /10 |
| **Total Points** |  | /100 Points |  |
| **CLO Mapped** |  | CLO 1 |  |

*For description of different levels of the mapped rubrics, please refer the provided Lab Evaluation Assessment Rubrics.*