Week 7

1. Write a SV program for Instruction Decode Stage

module instr\_decode (

input logic [31:0] inst,

output logic [31:0] rs1\_data,

output logic [31:0] rs2\_data,

output logic [31:0] imm32,

output logic [6:0] func7,

output logic [2:0] func3,

input logic [31:0] wr\_data,

input logic reset,

output logic [6:0]opcode);

wire [4:0] rs1\_addr;

wire [4:0] rs2\_addr;

wire [4:0] rd\_addr;

wire [11:0] imm12;

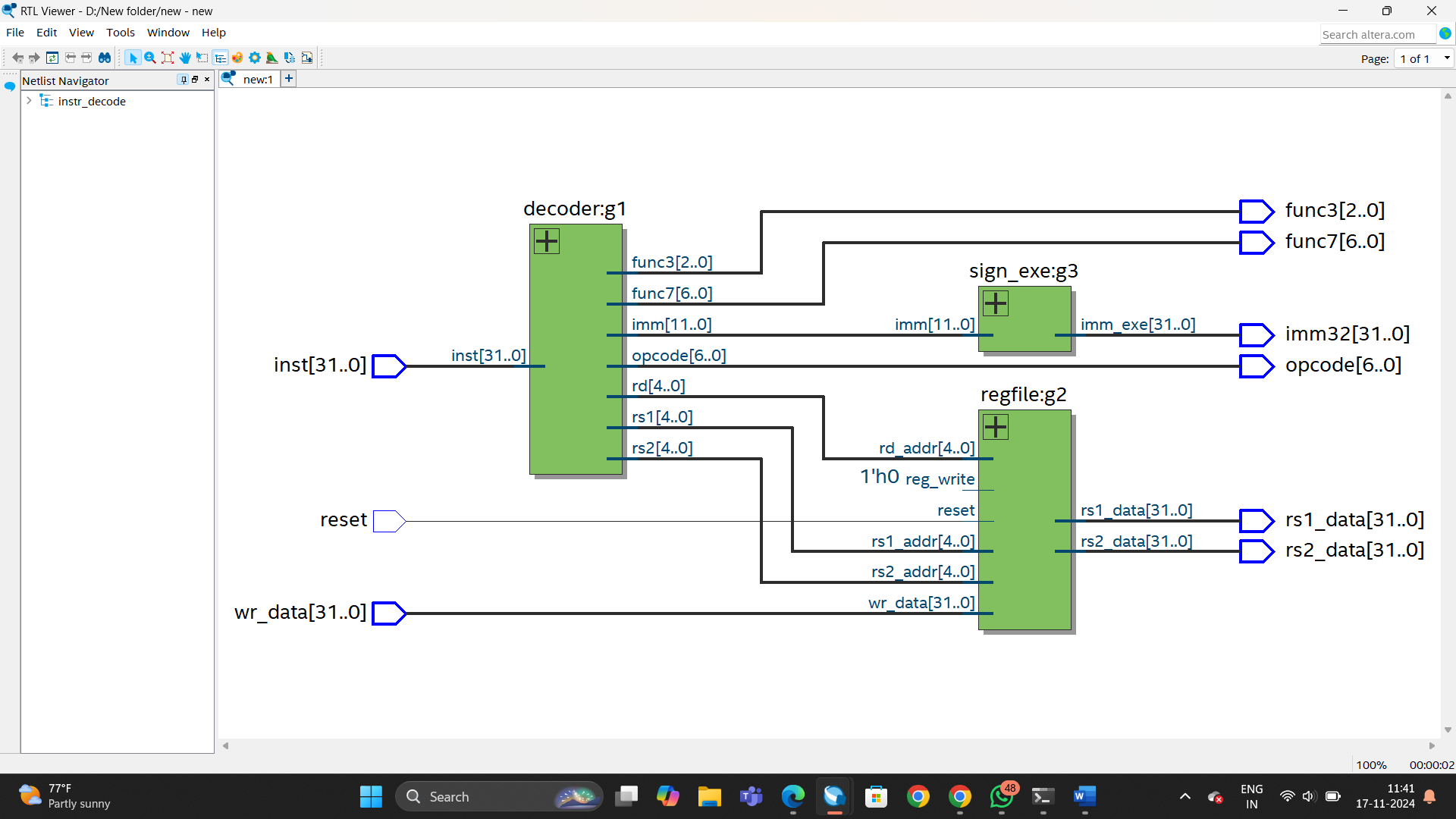
wire regwrite;

decoder g1(inst,rd\_addr,rs1\_addr,rs2\_addr,func3,func7,imm12,opcode);

regfile g2(rs1\_addr,rs2\_addr,rd\_addr,regwrite,wr\_data,reset,rs1\_data,rs2\_data);

sign\_exe g3(imm12,imm32);

endmodule



1. Write a SV program for the Decoder

module decoder (input logic [31:0]inst, output logic [4:0]rd,output logic [4:0]rs1, output logic [4:0]rs2,output logic [2:0]func3, output logic [6:0]func7,output logic [11:0]imm,output logic [6:0]opcode);

always\_comb

begin

rd = 5'b0;

rs1 = 5'b0;

rs2 = 5'b0;

func3 = 3'b0;

func7 = 7'b0;

imm = 12'b0;

opcode=inst[6:0];

case(inst[6:0])

7'b0110011:

begin

rd=inst[11:7];

rs1=inst[19:15];

rs2=inst[24:20];

func3=inst[14:12];

func7=inst[31:25];

end

7'b0010011:

begin

rd=inst[11:7];

rs1=inst[19:15];

func3=inst[14:7];

imm=inst[11:0];

end

7'b1100011:

begin

rs1=inst[19:15];

rs2=inst[24:20];

func3=inst[14:12];

imm={inst[31],inst[30:25],inst[11:8],inst[7]};

end

7'b0100011:

begin

rs1=inst[19:15];

rs2=inst[24:20];

func3=inst[14:7];

imm={inst[31:25],inst[11:7]};

end

7'b0000011:

begin

rd=inst[11:7];

rs1=inst[19:15];

func3=inst[14:7];

imm=inst[11:0];

end

endcase

end

endmodule

1. Write a SV program for the Register File

module regfile(input logic [4:0]rs1\_addr,rs2\_addr,rd\_addr,input logic reg\_write,input logic [31:0]wr\_data, input logic reset,output logic [31:0]rs1\_data,rs2\_data);

logic [31:0]rfile[0:31];

always\_comb

begin

if(~reset)

begin

rs1\_data=rfile[rs1\_addr];

rs2\_data=rfile[rs2\_addr];

end

else

begin

rs1\_data=32'b0;

rs2\_data=32'b0;

end

end

always\_ff @(posedge reg\_write or posedge reset) begin

if (reset) begin

for (int i = 0; i < 32; i++) begin

rfile[i] = 32'b0; // Clear all registers

end

end

else if (reg\_write) begin

rfile[rd\_addr] = wr\_data; // Write data to the specified register

end

end

endmodule

1. Write a SV program for the Immediate Unit

module sign\_exe (input logic [11:0]imm,output logic [31:0]imm\_exe);

always\_comb

begin

if(imm[11]==0)

begin

imm\_exe[31:12]=20'b0;

imm\_exe[11:0]=imm[11:0];

end

else

begin

imm\_exe[31:12]=20'b1;

imm\_exe[11:0]=imm[11:0];

end

end

endmodule