Lab 3 Report

Wentao Wang

CMPEN 331 Section 001

Code:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 2022/03/20 19:04:18

// Design Name:

// Module Name: Datapath

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module Datapath(

input clk,

output [31:0] pc,

output ewreg,

output em2reg,

output ewmem,

output [3:0] ealuc,

output ealuimm,

output [4:0] edestReg,

output [31:0] eqa,

output [31:0] eqb,

output [31:0] eimm32,

output [31:0] dinstOut

);

IF iF(clk,dinstOut, pc);

ID iD(clk,dinstOut,ewreg, em2reg,ewmem,ealuc, ealuimm,edestReg, eqa,eqb,eimm32);

Endmodule

Fetch:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 2022/03/20 19:31:02

// Design Name:

// Module Name: IF

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module pc(input [31:0] nextPc, input clk, output reg [31:0] opc);

initial opc=100;

always @(posedge clk) begin

opc<=nextPc ;

end

endmodule

module InsMem(input [31:0] opc, output reg [31:0] instOut);

reg [31:0] memory [0:63];

initial begin

memory [25]={

6'b100011,

5'b00001,

5'b00010,

5'b00000,

5'b00000,

6'b000000

};

memory [26]={

6'b100011,

5'b00001,

5'b00011,

5'b00000,

5'b00000,

6'b000100};

end

always @(\*) begin

instOut <=memory [opc[7:2]];

end

endmodule

module pcAdder(input [31:0] opc, output reg [31:0] nextPc);

always @(\*) begin

nextPc <=opc+4;

end

endmodule

module IFID(input [31:0] instOut, input clk, output reg [31:0] dinstOut);

always @(posedge clk) begin

dinstOut<= instOut;

end

endmodule

module IF(

input clk,

output [31:0] odinstOut,

output [31:0] opc

);

wire [31:0] wnPC;

wire [31:0] winstOut;

pc pci(wnPC , clk,opc );

pcAdder pcadd(opc,wnPC );

InsMem imem(opc, winstOut );

IFID ifid(winstOut,clk,odinstOut );

Endmodule

Decode:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 2022/03/20 19:31:02

// Design Name:

// Module Name: ID

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module control(input [5:0] op, input [5:0] func,

output reg wreg, output reg m2reg, output reg wmem,output reg [3:0] aluc,

output reg aluimm, output reg regrt);

always @(\*) begin

case (op)

6'b000000: // r-­-types

begin

case (func)

6'b100000: // ADD instruction

begin

wreg<=1;

m2reg<=0;

wmem<=0;

aluc<=4'b0010;

aluimm<=0;

regrt<=0;

end

6'b100010: // SUB instruction

begin

wreg<=1;

m2reg<=0;

wmem<=0;

aluc<=4'b0110;

aluimm<=0;

regrt<=0;

end

endcase

end

6'b100011: // LW

begin

wreg<=1;

m2reg<=1;

wmem<=0;

aluc<=4'b0010;

aluimm<=1;

regrt<=1;

end

endcase

end

endmodule

module regMux(input [4:0] rt, input [4:0] rd, input regrt, output reg [4:0]destReg);

always @(\*) begin

if(regrt) destReg <=rt;

else if (!regrt ) destReg <=rd;

end

endmodule

module regFile(input [4:0] rs, input [4:0] rt,

output reg [31:0] qa,output reg [31:0] qb);

reg[31:0] regs [0:31];

initial begin

regs[0] = 32'd0;

regs[1] = 32'd0;

regs[2] = 32'd0;

regs[3] = 32'd0;

regs[4] = 32'd0;

regs[5] = 32'd0;

regs[6] = 32'd0;

regs[7] = 32'd0;

regs[8] = 32'd0;

regs[9] = 32'd0;

regs[10] = 32'd0;

regs[11] = 32'd0;

regs[12] = 32'd0;

regs[13] = 32'd0;

regs[14] = 32'd0;

regs[15] = 32'd0;

regs[16] = 32'd0;

regs[17] = 32'd0;

regs[18] = 32'd0;

regs[19] = 32'd0;

regs[20] = 32'd0;

regs[21] = 32'd0;

regs[22] = 32'd0;

regs[23] = 32'd0;

regs[24] = 32'd0;

regs[25] = 32'd0;

regs[26] = 32'd0;

regs[27] = 32'd0;

regs[28] = 32'd0;

regs[29] = 32'd0;

regs[30] = 32'd0;

regs[31] = 32'd0;

end

always @(\*) begin

qa=regs[rs];

qb=regs[rt];

end

endmodule

module immEx(input [15:0] imm,output reg [31:0] imm32);

always @(\*) begin

imm32 <= {{17{imm[15]}},imm[14:0]};

end

endmodule

module IDPileReg(

input ewreg,

input em2reg,

input ewmem,

input [3:0] ealuc,

input ealuimm,

input [4:0] edestReg,

input [31:0] eqa,

input [31:0] eqb,

input [31:0] eimm32,

input clk,

output reg owreg,

output reg om2reg,

output reg owmem,

output reg [3:0] oaluc,

output reg oaluimm,

output reg [4:0] odestReg,

output reg[31:0] oqa,

output reg[31:0] oqb,

output reg [31:0] oimm32

);

always @(posedge clk) begin

owreg <= ewreg;

om2reg <= em2reg;

owmem <= ewmem;

oaluc <= ealuc;

oaluimm <= ealuimm;

odestReg <= edestReg;

oqa <= eqa;

oqb <= eqb;

oimm32 <= eimm32;

end

endmodule

module ID(

input clk,

input [31:0] dinstOut,

output owreg,

output om2reg,

output owmem,

output [3:0] oaluc,

output oaluimm,

output [4:0] odestReg,

output [31:0] oqa,

output [31:0] oqb,

output [31:0] oimm32

);

wire wwreg;

wire wm2reg;

wire wwmem;

wire [3:0] waluc;

wire waluimm;

wire [4:0] wdestReg;

wire [31:0] wqa;

wire [31:0] wqb;

wire [31:0] wimm32;

wire wregrt;

control cUnit(dinstOut[31:26],dinstOut[5:0],wwreg,

wm2reg,wwmem,waluc,waluimm ,wregrt );

regMux rmux(dinstOut[20:16],dinstOut[15:11],wregrt,wdestReg );

regFile regf(dinstOut[25:21],dinstOut[20:16],wqa,wqb);

immEx ImmEx(dinstOut[15:0],wimm32 );

IDPileReg idp(wwreg, wm2reg, wwmem, waluc, waluimm, wdestReg, wqa, wqb, wimm32, clk,

owreg, om2reg, owmem, oaluc, oaluimm, odestReg, oqa, oqb, oimm32);

endmodule

testbrench:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 2022/03/20 22:24:20

// Design Name:

// Module Name: Testbanch

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module Testbanch();

reg clk;

wire [31:0] TBpc;

wire [31:0] TBdinstOut;

wire TBewreg;

wire TBem2reg;

wire TBewmem;

wire [3:0] TBealuc;

wire TBealuimm;

wire [4:0] TBedestReg;

wire [31:0] TBeqa;

wire [31:0] TBeqb;

wire [31:0] TBeimm32;

Datapath df(clk, TBpc, TBewreg, TBem2reg ,TBewmem,

TBealuc ,TBealuimm ,TBedestReg ,TBeqa ,TBeqb,TBeimm32 ,TBdinstOut );

initial begin

clk=0;

#100 $finish;

end

always begin

#5;

clk=~clk;

end

endmodule

I/O planning

电脑萤幕画面

描述已自动生成

Floor planning

电脑萤幕画面

描述已自动生成

Schematic

图形用户界面, 应用程序, Word

描述已自动生成

Waveform with IF and ID

图片包含 日程表

描述已自动生成

图片包含 图形用户界面

描述已自动生成