**实验报告**

2019 年 4 月 9 日 成绩：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 於文卓 | 学号 | 17061833 | 班级 | 17052317 |
| 专业 | 计算机科学与技术 | | 课程名称 | 计算机组成原理课程设计 | |
| 任课老师 | 冯建文 | 指导老师 | 冯建文 | 机位号 |  |
| 实验序号 | 3 | 实验名称 | 多功能ALU设计实验 | | |
| 实验时间 | 4月9日 | 实验地点 | 1-225 | 实验设备号 | #9 |
| **一、实验目的与要求** | | | | | |
| 1. 实验目的： 2. 学习多功能ALU的工作原理，掌握运算器的设计方法 3. 掌握运用Verilog HDL进行行为描述与建模的技巧和方法 4. 实验要求：   （1）使用行为描述方式（case语句），编程实现多功能ALU，供后续CPU的设计使用  （2）为了测试该ALU模块功能的正确性，还需要编写一个实验验证模块，调用该ALU模块，并使用开关控制测试数据的选择和结果的显示 | | | | | |
| **二、实验设计与程序代码** | | | | | |
| 1. 模块设计说明 2. 设计ALU模块，输入为A，B，和操作符OP，输出零标志ZF，溢出标志OF和运算结果F 3. 构建数码管模块，使得F结果显示在数码管上 4. 构建选择模块，由于板卡开关数量的限制，所以使用三位A\_SW来选择八个A数据，同理用三位B\_SW来选择八个B数据 5. 构建顶层模块，调用选择模块、ALU模块和数码管模块 6. 实验程序源代码及注释等 7. ALU模块代码  |  | | --- | | `timescale 1ns / 1ps  //////////////////////////////////////////////////////////////////////////////////  // Company:  // Engineer:  //  // Create Date: 23:37:17 04/08/2019  // Design Name:  // Module Name: AUL  // Project Name:  // Target Devices:  // Tool versions:  // Description:  //  // Dependencies:  //  // Revision:  // Revision 0.01 - File Created  // Additional Comments:  //  //////////////////////////////////////////////////////////////////////////////////  module AUL(  input [31:0]A,  input [31:0]B,  input [2:0] ALU\_OP,  output reg ZF,  output reg OF,  output reg [31:0]F  );  reg C32;  always @(ALU\_OP or A or B)  begin  C32 = 0;  case(ALU\_OP)  3'b001:F = A&B;  3'b010:F = A|B;  3'b011:F = A^B;  3'b100:F = ~(A|B);  3'b101:{C32,F} = A+B;  3'b110:{C32,F} = A-B;  3'b111:F = (A<B);  3'b000:F = B<<A;  endcase  if(F==0) ZF = 1;  else ZF = 0;  OF = A[31]^B[31]^F[31]^C32;  end  endmodule |  1. Choice模块代码  |  | | --- | | `timescale 1ns / 1ps  //////////////////////////////////////////////////////////////////////////////////  // Company:  // Engineer:  //  // Create Date: 23:02:51 04/08/2019  // Design Name:  // Module Name: choice  // Project Name:  // Target Devices:  // Tool versions:  // Description:  //  // Dependencies:  //  // Revision:  // Revision 0.01 - File Created  // Additional Comments:  //  //////////////////////////////////////////////////////////////////////////////////  module choice(  input [2:0]A\_SW,  input [2:0]B\_SW,  output reg[31:0] A,B  );  always @(\*)  begin  case(A\_SW)  3'b000:begin A=32'h0000\_0000;end  3'b001:begin A=32'h0000\_0003;end  3'b010:begin A=32'h8000\_0000;end  3'b011:begin A=32'h7FFF\_FFFF;end  3'b100:begin A=32'hFFFF\_FFFF;end  3'b101:begin A=32'h8000\_0000;end  3'b110:begin A=32'h1234\_5678;end  3'b111:begin A=32'h9ABC\_DEF0;end  default:begin A=32'h9ABC\_DEF0;end  endcase  end    always @(\*)  begin  case(B\_SW)  3'b000:begin B=32'h0000\_0000;end  3'b001:begin B=32'h0000\_0607;end  3'b010:begin B=32'h8000\_0000;end  3'b011:begin B=32'h7FFF\_FFFF;end  3'b100:begin B=32'hFFFF\_FFFF;end  3'b101:begin B=32'hFFFF\_FFFF;end  3'b110:begin B=32'h3333\_2222;end  3'b111:begin B=32'h1111\_2222;end  default:begin B=32'h1111\_2222;end  endcase  end      endmodule |  1. 数码管模块  |  | | --- | | `timescale 1ns / 1ps  //////////////////////////////////////////////////////////////////////////////////  // Company:  // Engineer:  //  // Create Date: 10:42:47 12/11/2018  // Design Name:  // Module Name: scan  // Project Name:  // Target Devices:  // Tool versions:  // Description:  //  // Dependencies:  //  // Revision:  // Revision 0.01 - File Created  // Additional Comments:  //  //////////////////////////////////////////////////////////////////////////////////  module scan(    input clk,//100MHZ  input rst\_,//reset  input [31:0]key,//通过拨动按钮控制段选  output reg [7:0]wei,//在always里赋值 需要用到reg  output reg [7:0]duan  );  wire clk\_4ms;//分频过后2ms一个方波(改过了 是2ms了)  reg [2:0]wei\_count;//位码计数器，控制4个数码管  reg [7:0]segcode[0:15] = {//用数组的方式定义 左表示位数 右表示个数  8'b0000\_0011,//0  8'b1001\_1111,//1  8'b0010\_0101,//2  8'b0000\_1101,//3  8'b1001\_1001,//4  8'b0100\_1001,//5  8'b0100\_0001,//6  8'b0001\_1111,//7  8'b0000\_0001,//8  8'b0000\_1001,//9  ~8'b11101110,//A  ~8'b00111110,//b  ~8'b10011100,//C  ~8'b01111010,//d  ~8'b10011110,//E  8'b01110001//F  };  div f1(rst\_,clk,clk\_4ms);//分频器模块  wire clk\_slow;//一秒钟两次  div\_slow f2(rst\_,clk,clk\_slow);  always @(posedge clk\_4ms or negedge rst\_)//每一个上升沿都使得计数器+1 做到连续扫描数码管  begin  if(!rst\_)  begin  wei\_count<=3'b00;  end  else  begin  wei\_count<=wei\_count+1'b1;  if(wei\_count ==3'b111)  begin  wei\_count<=3'b000;  end  end    end  reg [7:0] pos\_1,pos\_2,pos\_3,pos\_4,pos\_5,pos\_6,pos\_7,pos\_8;  //用开关控制数码管显示  always @(posedge clk\_4ms or negedge rst\_)  begin  if(!rst\_)  begin  pos\_1 <= 4'b0000;  pos\_2 <= 4'b0000;  pos\_3 <= 4'b0000;  pos\_4 <= 4'b0000;  pos\_5 <= 4'b0000;  pos\_6 <= 4'b0000;  pos\_7 <= 4'b0000;  pos\_8 <= 4'b0000;  end  else  begin  pos\_1 <= key[3:0];//if(pos\_1 > 4'd9) pos\_1<=4'b0000;  pos\_2 <= key[7:4];//if(pos\_2 > 4'd9) pos\_2<=4'b0000;  pos\_3 <= key[11:8];//if(pos\_3 > 4'd9) pos\_3<=4'b0000;  pos\_4 <= key[15:12];//if(pos\_4 > 4'd9) pos\_4<=4'b0000;  pos\_5 <= key[19:16];  pos\_6 <= key[23:20];  pos\_7 <= key[27:24];  pos\_8 <= key[31:28];  end    end  always @(\*)//数码管扫描进程 想想为什么是\* ？？？  begin  case(wei\_count)  3'b000: begin wei<=8'b11111110; duan<=segcode[pos\_1];end  3'b001: begin wei<=8'b11111101; duan<=segcode[pos\_2];end  3'b010: begin wei<=8'b11111011; duan<=segcode[pos\_3];end  3'b011: begin wei<=8'b11110111; duan<=segcode[pos\_4];end  3'b100: begin wei<=8'b11101111; duan<=segcode[pos\_5];end  3'b101: begin wei<=8'b11011111; duan<=segcode[pos\_6];end  3'b110: begin wei<=8'b10111111; duan<=segcode[pos\_7];end  3'b111: begin wei<=8'b01111111; duan<=segcode[pos\_8];end  default:begin wei<=8'b11111111;end  endcase  end    endmodule |  1. 顶层模块  |  | | --- | | `timescale 1ns / 1ps  //////////////////////////////////////////////////////////////////////////////////  // Company:  // Engineer:  //  // Create Date: 00:26:45 04/09/2019  // Design Name:  // Module Name: top  // Project Name:  // Target Devices:  // Tool versions:  // Description:  //  // Dependencies:  //  // Revision:  // Revision 0.01 - File Created  // Additional Comments:  //  //////////////////////////////////////////////////////////////////////////////////  module top(  input [2:0]A\_SW,  input [2:0]B\_SW,  input [2:0]ALU\_OP,  input clk,  input rst\_,  output [7:0]wei,  output [7:0]duan,  output wire ZF,  output wire OF  );  wire [31:0]A,B;  wire [31:0]F;  choice m1(A\_SW,B\_SW,A,B);  AUL m2(A,B,ALU\_OP,ZF,OF,F);  scan m3(clk,rst\_,F,wei,duan);  endmodule | | | | | | |
| **三、实验仿真** | | | | | |
| 1. 仿真代码  |  | | --- | | `timescale 1ns / 1ps  ////////////////////////////////////////////////////////////////////////////////  // Company:  // Engineer:  //  // Create Date: 13:49:21 04/09/2019  // Design Name: AUL  // Module Name: D:/fpga/alu/test.v  // Project Name: alu  // Target Device:  // Tool versions:  // Description:  //  // Verilog Test Fixture created by ISE for module: AUL  //  // Dependencies:  //  // Revision:  // Revision 0.01 - File Created  // Additional Comments:  //  ////////////////////////////////////////////////////////////////////////////////  module test;  // Inputs  reg [31:0] A;  reg [31:0] B;  reg [2:0] ALU\_OP;  // Outputs  wire ZF;  wire OF;  wire [31:0] F;  // Instantiate the Unit Under Test (UUT)  AUL uut (  .A(A),  .B(B),  .ALU\_OP(ALU\_OP),  .ZF(ZF),  .OF(OF),  .F(F)  );  initial begin  // Initialize Inputs  A = 0;  B = 0;  ALU\_OP = 0;  // Wait 100 ns for global reset to finish  #100;    // Add stimulus here  A=32'h0000\_0000;B=32'h0000\_0000;ALU\_OP = 3'b101;//加法  #100;    A=32'h0000\_0003;B=32'h0000\_0607;ALU\_OP = 3'b101;//加法 60A  #100  A=32'h8000\_0000;B=32'h8000\_0000;ALU\_OP = 3'b110;//减法 0  #100  A=32'h7FFF\_FFFF;B=32'h7FFF\_FFFF;ALU\_OP = 3'b101;//加法 FFFFFFFE 溢出  #100  A=32'hFFFF\_FFFF;B=32'hFFFF\_FFFF;ALU\_OP = 3'b001;//与运算  #100  A=32'h8000\_0000;B=32'hFFFF\_FFFF;ALU\_OP = 3'b010;//或运算  #100  A=32'h1234\_5678;B=32'h3333\_2222;ALU\_OP = 3'b011;//异或  #100  A=32'h9ABC\_DEF0;B=32'h1111\_2222;ALU\_OP = 3'b100;//或非  #100  A=32'h9ABC\_DEF0;B=32'h1111\_2222;ALU\_OP = 3'b001;//比大小  end    endmodule |  1. 仿真波形      1. 仿真结果分析   当A与B为0，做加法时，结果F也为0，这时ZF为1；  当A=32'h0000\_0003;B=32'h0000\_0607;ALU\_OP = 3'b101;此时做加法，结果为60A  当A=32'h8000\_0000;B=32'h8000\_0000;ALU\_OP = 3'b110;此时做减法，结果为0  当A=32'h7FFF\_FFFF;B=32'h7FFF\_FFFF;ALU\_OP = 3'b101;此时做加法 结果为FFFFFFFE，OF=1，表示数据溢出  当A=32'hFFFF\_FFFF;B=32'hFFFF\_FFFF;ALU\_OP = 3'b001;此时做与运算，结果为全1  当A=32'h8000\_0000;B=32'hFFFF\_FFFF;ALU\_OP = 3'b010; 此时做或运算，结果为全1  当A=32'h1234\_5678;B=32'h3333\_2222;ALU\_OP = 3'b011;此时为异或运算，结果为2107745a  当A=32'h9ABC\_DEF0;B=32'h1111\_2222;ALU\_OP = 3'b100;此时为或非运算，结果为6442010d  当A=32'h9ABC\_DEF0;B=32'h1111\_2222;ALU\_OP = 3'b001;此时比较A与B的大小，结果为1  仿真结果符合运算结果，仿真结果正确 | | | | | |
| **四、电路图** | | | | | |
|  | | | | | |
| **五、引脚配置（约束文件）** | | | | | |
| NET "clk" LOC = E3 |IOSTANDARD = LVCMOS18;  NET "rst\_" LOC = N17 |IOSTANDARD = LVCMOS18 |CLOCK\_DEDICATED\_ROUTE = FALSE;  NET "A\_SW[2]" LOC = V5 |IOSTANDARD = LVCMOS18;  NET "A\_SW[1]" LOC = T4 |IOSTANDARD = LVCMOS18;  NET "A\_SW[0]" LOC = V6 |IOSTANDARD = LVCMOS18;  NET "B\_SW[2]" LOC = T5 |IOSTANDARD = LVCMOS18;  NET "B\_SW[1]" LOC = T6 |IOSTANDARD = LVCMOS18;  NET "B\_SW[0]" LOC = V7 |IOSTANDARD = LVCMOS18;  NET "ALU\_OP[2]" LOC = R8 |IOSTANDARD = LVCMOS18;  NET "ALU\_OP[1]" LOC = U9 |IOSTANDARD = LVCMOS18;  NET "ALU\_OP[0]" LOC = T9 |IOSTANDARD = LVCMOS18;  NET "ZF" LOC = U6 |IOSTANDARD = LVCMOS18;  NET "OF" LOC = R5 |IOSTANDARD = LVCMOS18;  NET "wei[7]" LOC = C9 |IOSTANDARD = LVCMOS18;  NET "wei[6]" LOC = C10 |IOSTANDARD = LVCMOS18;  NET "wei[5]" LOC = D10 |IOSTANDARD = LVCMOS18;  NET "wei[4]" LOC = C11 |IOSTANDARD = LVCMOS18;  NET "wei[3]" LOC = M17 |IOSTANDARD = LVCMOS18;  NET "wei[2]" LOC = J14 |IOSTANDARD = LVCMOS18;  NET "wei[1]" LOC = K13 |IOSTANDARD = LVCMOS18;  NET "wei[0]" LOC = P14 |IOSTANDARD = LVCMOS18;  NET "duan[7]" LOC = F14 |IOSTANDARD = LVCMOS18;  NET "duan[6]" LOC = N14 |IOSTANDARD = LVCMOS18;  NET "duan[5]" LOC = J13 |IOSTANDARD = LVCMOS18;  NET "duan[4]" LOC = G13 |IOSTANDARD = LVCMOS18;  NET "duan[3]" LOC = F13 |IOSTANDARD = LVCMOS18;  NET "duan[2]" LOC = G14 |IOSTANDARD = LVCMOS18;  NET "duan[1]" LOC = M13 |IOSTANDARD = LVCMOS18;  NET "duan[0]" LOC = H14 |IOSTANDARD = LVCMOS18; | | | | | |
| **六、思考与探索** | | | | | |
| 1. 实验结果记录：  |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | **A=32’h0000\_0003**  **B=32’h0000\_0607** | **A=32’h0000\_0000**  **B=32’h0000\_0000** | **A=32’h8000\_0000**  **B=32’h8000\_0000** | **A=32’h7fff\_ffff**  **B=32’h7fff\_ffff** | | A·B | F=0x00000003  ZF=0 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x80000000  ZF=0 OF=0 | F=0x7fffffff  ZF=0 OF=0 | | A+B | F=0x00000607  ZF=0 OF=0 | F=0x00000000  ZF= 1 OF=0 | F=0x80000000  ZF=0 OF=0 | F=0x7fffffff  ZF=0 OF=0 | | A⊕B | F=0x00000604  ZF=0 OF=0 | F=0x00000000  ZF= 1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | |  | F=0xfffff9f0  ZF=0 OF=0 | F=0xffffffff  ZF=0 OF=0 | F=0x7fffffff  ZF=0 OF=0 | F=0x80000000  ZF=0 OF=0 | | A加B | F=0000060a  ZF=0 OF=0 | F=0x00000000  ZF= 1 OF=1 | F=0x00000000  ZF=1 OF=1 | F=0xfffffffe  ZF=0 OF=0 | | A减B | F=0xfffff9fc  ZF= 0 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | | A<B置1 | F=00000001  ZF=0 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | | B<<A | F=00003038  ZF=0 OF=0 | F=0x00000000  ZF= 1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | |  | **A=32’hffff\_ffff**  **B=32’hffff\_ffff** | **A=32’h8000\_0000**  **B=32’hffff\_ffff** | **A=32’hffff\_ffff**  **B=32’h8000\_0000** | **A=32’h9ABC\_DEF0**  **B=32’h1111\_2222** | | A·B | F=0xffffffff  ZF=0 OF=0 | F=0x80000000  ZF=0 OF=0 | F=0x80000000  ZF=0 OF=0 | F=0x10100220  ZF=0 OF=0 | | A+B | F=0xffffffff  ZF=0 OF=0 | F=0xffffffff  ZF=0 OF=0 | F=0xffffffff  ZF=0 OF=0 | F=0x9bbdfef2  ZF=0 OF=0 | | A⊕B | F=0x00000000  ZF=1 OF=0 | F=0x7fffffff  ZF=0 OF=0 | F=0x7ffffffff  ZF=0 OF=0 | F=0x8badfcd2  ZF=0 OF=0 | |  | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x6442010d  ZF=0 OF=0 | | A加B | F=0xfffffffe  ZF=0 OF=0 | F=0x7fffffff  ZF=0 OF=1 | F=0x7fffffff  ZF=0 OF=1 | F=0xabce0112  ZF=0 OF=0 | | A减B | F=0x00000000  ZF=1 OF=0 | F=0x80000001  ZF=0 OF=0 | F=0x00000000  ZF=1 OF=0 | F=ox89abbcce  ZF=0 OF=0 | | A<B置1 | F=0x00000000  ZF=1 OF=0 | F=0x00000001  ZF=0 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | | B<<A | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 | F=0x00000000  ZF=1 OF=0 |   在板级验证中  当控制A和B的开关拨为001时，表示A的值为00000003，B的值为00000607，控制操作的开关为101时，表示A和B做加法，结果数码管显示0000\_060A  当A和B均输入7FFFFFFF时，控制操作开关为101，表示两者做加法，此时溢出OF灯亮起  当A为0000\_0003，B为0000\_0607时，控制操作开关为000，表示B左移三位，数码管显示3038   1. 实验结论：   实验结果也符合预期，说明实验结果正确   1. 问题与解决方案： 2. 在调用数码管时有一些遗忘，复习了去年数电的知识才得以成功调用 3. 一开始按照书上使用三个开关控制八组AB数据，应该用六个开关，分别控制A的八组数据和B的八组数据 4. 思考题：   (2)这个ALU不能实现MIPS中核心指令集的所有指令，还需要位清除指令、负数比较、位测试指令、相等指令等  (3) A=32'h0;B=32'hFFFF\_FFFF;ALU\_OP = 3'b111; 通过对此数据仿真，结果为F=1，说明A<B,即是对无符号数的比较  (4) if(F<0) SF = 1 else SF = 0;  PF = ~^F;  CF = C; | | | | | |