

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 计算机组成 |
| 姓 名： | 胡亮泽 |
| 学 院： | 计算机科学与技术学院 |
| 系： | 计算机科学与技术系 |
| 专 业： | 计算机科学与技术 |
| 学 号： | 3120102116 |
| 指导教师： | 姜晓红 |

2014年 3 月 14 日

**浙江大学实验报告**

课程名称： Computer Organization 实验类型： 综合

实验项目名称： Lab3： multi function ALU

学生姓名： 胡亮泽 专业： 计算机科学与技术

学号： 3120102116

同组学生姓名： 王艺 指导老师： 姜晓红

实验地点： 东4-509 实验日期： 2014 年 3月 11 日

1. 实验目的和要求

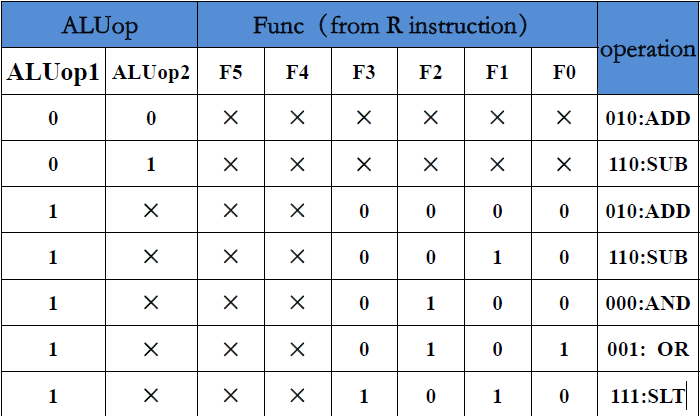
1. Master the principle and design method of ALU.

2. Master the principle and design method of ALU Controller.

1. 实验内容和原理

ALU由两部分组成，分别是ALU 和 ALU control, 其中ALU负责运算，而ALU control 负责解码的功能，即将相应的输入码转换成对应的指令码传输给ALU，从而实现对应的运算功能。

ALU control的解码对应真值表如下：



其中ALUop1和ALUop2由两个开关实现，F0~F3由4个开关实现。

其对应代码如下：

|  |
| --- |
| module aluc(input wire[1:0] button,  input wire[3:0] switch,  output wire[2:0]control  );  wire ALUop1,ALUop2;  wire result1,result2,result3,result4,result5,result6;  wire or\_result1,or\_result2;  assign ALUop1 = button[1];  assign ALUop2 = button[0];  and  and1(result1,~switch[3],~switch[2],~switch[1],~switch[0],ALUop1),  and2(result2,~switch[3],~switch[2],switch[1],~switch[0],ALUop1),  and3(result3,~switch[3],switch[2],~switch[1],~switch[0],ALUop1),  and4(result4,~switch[3],switch[2],~switch[1],switch[0],ALUop1),  and5(result5,switch[3],~switch[2],switch[1],~switch[0],ALUop1),  and6(result6,~ALUop1,ALUop2);  or  or1(or\_result1,result4,result5),  or2(control[1],result1,result2,result5,result6),  or3(control[2],result6,result2,result5);  and  and7(control[0],ALUop1,or\_result1);  endmodule |

如代码所示，本次实验采用结构化描述的方法实现ALU control的功能。以输出相应的指令码。具体对应关系不做赘述。

ALU的代码如下所示：

|  |
| --- |
| module alu(input wire[15:0] op1,  input wire[15:0] op2,  input wire[2:0] control,  output wire o\_zf,  output reg[15:0] disp\_code  );  wire switch;  wire co;  wire[15:0] add\_sub\_result;  assign switch = (control==3'b110);  assign o\_zf = (disp\_code==0);  adder\_16bits m0(op1, op2, switch, add\_sub\_result, co);  always @\* begin  case(control)  3'b010: disp\_code= add\_sub\_result;  3'b110: disp\_code= add\_sub\_result;  3'b010: disp\_code= add\_sub\_result;  3'b110: disp\_code= add\_sub\_result;  3'b000: disp\_code= op1 & op2;  3'b001: disp\_code= op1 | op2;  3'b111: disp\_code= (op1<op2?1:0);  endcase  end  endmodule |

其中adder\_16bits m0(op1, op2, switch, add\_sub\_result, co) 模块用于进行加减法的运算，switch用于表示加法或减法，在switch为0时进行加法运算，否则进行减法运算。根据指令真值表的关系，采用assign赋值语句：

assign switch = (control==3'b110);

进行赋值，而control即为ALU control传输的指令序号。

之后便是根据不同的指令进行不同的运算了，通过case语句实现。

3'b010: disp\_code= add\_sub\_result;

3'b110: disp\_code= add\_sub\_result;

3'b010: disp\_code= add\_sub\_result;

3'b110: disp\_code= add\_sub\_result;

3'b000: disp\_code= op1 & op2;

3'b001: disp\_code= op1 | op2;

3'b111: disp\_code= (op1<op2?1:0);

另外，16位加法器的功能代码如下：

|  |
| --- |
| module adder\_16bits(A, B, Ctr, S, Co);  parameter size=16;  input [size:1] A;  input [size:1] B;  input Ctr;  output [size:1] S;  output Co;  wire[size-1:1] Ctemp;  wire[size:1] Bo;    assign Bo={16{Ctr}}^B;  adder\_1bit A1(.a(A[1]),.b(Bo[1]),.ci(Ctr),.s(S[1]),.co(Ctemp[1])),  A2(A[2],Bo[2],Ctemp[1],S[2],Ctemp[2]),  A3(A[3],Bo[3],Ctemp[2],S[3],Ctemp[3]),  A4(A[4],Bo[4],Ctemp[3],S[4],Ctemp[4]),  A5(A[5],Bo[5],Ctemp[4],S[5],Ctemp[5]),  A6(A[6],Bo[6],Ctemp[5],S[6],Ctemp[6]),  A7(A[7],Bo[7],Ctemp[6],S[7],Ctemp[7]),  A8(A[8],Bo[8],Ctemp[7],S[8],Ctemp[8]),  A9(A[9],Bo[9],Ctemp[8],S[9],Ctemp[9]),  A10(A[10],Bo[10],Ctemp[9],S[10],Ctemp[10]),  A11(A[11],Bo[11],Ctemp[10],S[11],Ctemp[11]),  A12(A[12],Bo[12],Ctemp[11],S[12],Ctemp[12]),  A13(A[13],Bo[13],Ctemp[12],S[13],Ctemp[13]),  A14(A[14],Bo[14],Ctemp[13],S[14],Ctemp[14]),  A15(A[15],Bo[15],Ctemp[14],S[15],Ctemp[15]),  A16(A[16],Bo[16],Ctemp[15],S[16],Co);  endmodule  module adder\_1bit(a, b, ci, s, co);  input wire a,b,ci;  output wire s,co;  and (c1,a,b), (c2,b,ci), (c3,a,ci);  xor (s1,a,b), (s,s1,ci);  or (co,c1,c2,c3);  endmodule |

加法器原理为串行加法进位原理，这里不多做赘述。

另外，还通过assign赋值语句，使得在运算结果为0时，LED变亮，具体方法简单，不做赘述。

1. 实验过程和数据记录

将代码下载到开发板上之后，将所有开关置于低电平，发现初始数码管显示“1122”，即op1的值。将最右端的开关上拨，显示“3344”，即op2的值。将switch[1]开关上拨后，显示4466，即加法（初始状态下两个按钮均未按下）运算得到的结果。按下ALUop2后，显示ddde（减法）的运算结果。接着，在按下ALUop1的情况下，分别将四个控制开关拨到0000,0010,0100，0101,1010的状态，分别显示4466（加），ddde（减）,1100（与）,3366（或）,0001（比较判断）的结果，与结果相符。

另外，在输入两个完全相同的数字时，利用减法得到0的运算结果后LED灯会变亮。

1. 实验结果

实验过程中的结果完全与预期符合。但过程中通过门级机构化描述的方法实现功能需要极大的耐心与细心，并且在发现错误的时候较难发现并纠正。通常可能出现的情况为线路的错误逻辑关系，遗漏部分逻辑关系，以及同一点高低电平出现矛盾的情况。

1. 讨论与心得

本次实验在纠错后，主要让我掌握了ALU的功能实现方法，以及进一步加深了结构化描述语言的实现方法。另外，为了防止错误的产生，一定要在描述之前清晰的列出逻辑关系，并且代码书写时一定要清晰，有条理，否则会对后期的调试造成致命性的难度！