|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学部/院 | 智能与计算学部 | 年级 | 2020 | 班级 | 软工3班 |
| 姓名 | 李培新 | 学号 | 3020244295 | 实验日期 | 2022.4.25 |

实验项目名称 算数逻辑单元

**一. 实验目的**

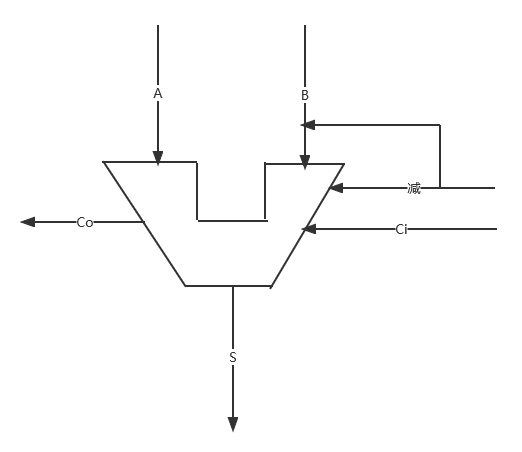
1. 掌握全加器和行波进位加法器的结构；  
 2. 熟悉加减法运算及溢出的判断方法；  
 3. 掌握算术逻辑单元（ALU）的结构；  
 4. 熟练使用 SystemVerilog HDL 的行为建模和结构化建模方法对 ALU 进行描述实现；  
 5. 为“单周期 MIPS 处理器的设计与实现”奠定基础。

**二. 实验内容**

基于 SystemVerilog HDL 设计并实现一个 4 位 ALU 单元。整个工程的顶层模块如图 2-4 所示，输入/输出端口如表 2-2 所示。注意，顶层模块由两个子模块组成，其中，一个是 ALU 单元，另一个是 7 段数码管动态显示扫描单元。同学们只需要实现 ALU 单元即可，动态显示扫描单元在工程中直接提供。

**三．实验原理与步骤（注：步骤不用写工具的操作步骤，而是设计步骤）**

1. 画出实现加/减法运算的逻辑电路原理图，并说明为什么加/减法可以只使用一个加法器进行实现？

可以由信号改变B的补码。  
 2. 给出有符号数加/减法溢出的判断规则？

AB同号且和值异号/AB异号且和值异号。  
 3. 给出 ALU 单元的 SystemVerilog HDL 代码。

module alu(

input [3 : 0] A,

input [3 : 0] B,

input [3 : 0] aluop,

output logic[7:0] alures,

output logic ZF,

output logic OF

);

logic Cout;

logic [3:0] S;

logic [3:0]Cin;

assign Cin=aluop[2]==0 ? 4'b0000 : 4'b1111;

rca R(.A(A),.B((~B&Cin)|(B&(~Cin))),.Ci(Cin[0]),.S(S),.Co(Cout));

always\_comb begin

OF=0;

alures[7:4]=4'b0000;

case(aluop)

4'b0000:alures[3:0]=A&B;

4'b0001:alures[3:0]=A|B;

4'b0010:alures[3:0]=A^B;

4'b0011:alures[3:0]=~(A&B);

4'b0100:alures[3:0]=~A;

4'b0101:alures[3:0]=A<<B[2:0];

4'b0110:alures[3:0]=A>>B[2:0];

4'b0111:alures[3:0]=A>>>B[2:0];

4'b1000:alures=A\*B;

4'b1001:alures={{4{A[3]}},A}\*{{4{B[3]}},B};

4'b1010:begin

if((A[3]==B[3])&&(A[3]!=S[3]))begin

alures[3:0]=S;

OF=1;

end

else begin

alures[3:0]=S;

end

end

4'b1011:alures[3:0]=S;

4'b1100:begin

if((A[3]!=B[3])&&(S[3]!=A[3]))begin

alures[3:0]=S;

OF=1;

end

else begin

alures[3:0]=S;

end

end

4'b1101:alures[3:0]=S;

4'b1110:alures[3:0]=$signed(A)<$signed(B)?1:0;

4'b1111:alures[3:0]=A<B?1:0;

endcase

ZF=!alures;

end

endmodule

4. 给出具有自动化测试功能的仿真程序和对应的波形图截图，并说明为什么选取这些测试向量？

module ALU\_4bits\_tb();

logic [3:0] A;

logic [3:0] B;

logic [3:0] aluop;

logic [7:0] alures;

logic ZF,OF;

logic [7:0] alures\_y;

logic ZF\_y,OF\_y;

logic [21:0] stim[3855:0];

int i;

alu DUT(.A(A),.B(B),.aluop(aluop),.alures(alures),.ZF(ZF),.OF(OF));

initial begin

$readmemb("testcase.txt",stim);

for(i=0;i<3856;i=i+1) begin

{A,B,aluop,alures\_y,ZF\_y,OF\_y}=stim[i];

#0.1;

if(alures== alures\_y&&ZF==ZF\_y&&OF==OF\_y)

$display($time,"test pass!");

else

$display($time,"Error:A=%b,B=%b,aluop=%b,alures=%b,ZF=%b,OF=%b,alures\_y=%b,ZF\_y=%b,OF\_y=%b",A,B,aluop,alures,ZF,OF,alures\_y,ZF\_y,OF\_y);

end

end

endmodule

网上找到一个全部的测试用例，

自测只用了有符号数1010\*0010，0100+0100，0001-0010这三个有溢出的：

$readmemb("test.txt",stim);

for(i=0;i<3;i=i+1) begin

{A,B,aluop,alures\_y,ZF\_y,OF\_y}=stim[i];

#0.1;

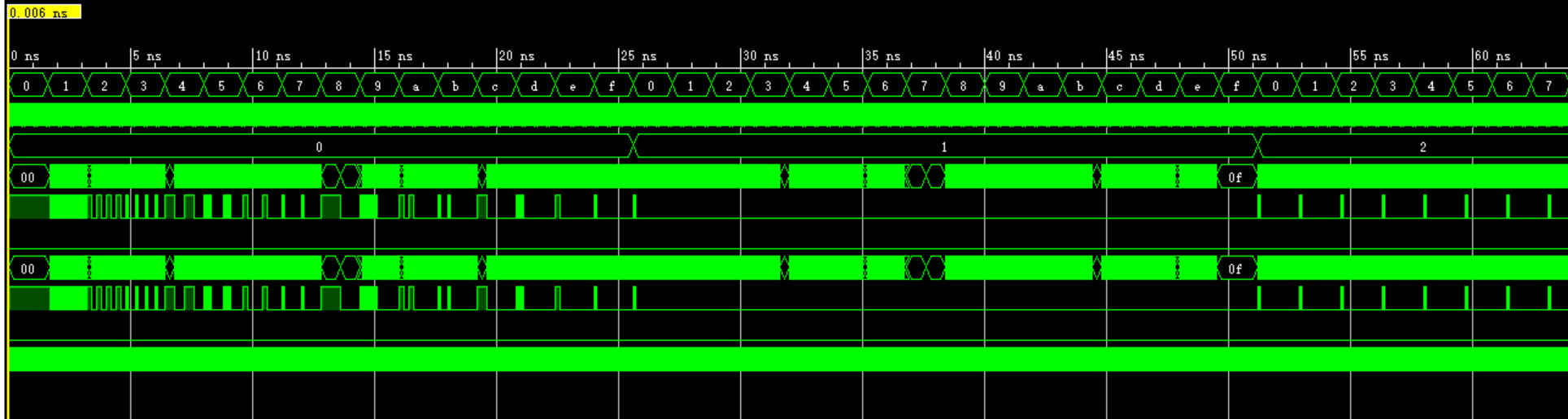
if(alures== alures\_y&&ZF==ZF\_y&&OF==OF\_y)

$display($time,"test pass!");

else

$display($time,"Error:A=%b,B=%b,aluop=%b,alures=%b,ZF=%b,OF=%b,alures\_y=%b,ZF\_y=%b,OF\_y=%b",A,B,aluop,alures,ZF,OF,alures\_y,ZF\_y,OF\_y);

end



**四．问题及解决方法**

有符号数乘法，最后采取溢出解决。

自动化测试有点难写。

**教师签字：**

**年 月 日**