实验三 加法器和ALU

实验报告

181860085 汤昊

2019.9.22

一．思考题

* 思考题一

答：应比较A1,B1和运算结果的符号位，因为加法器实际只进行加法操作，减法操作是将减数变成补码再相加，因此A1，B1是实际参加运算的数，若使用另一种方法，当A>0,B>0,A<B时，结果符号位与A，B不同，实际没有溢出却被判为溢出

* 思考题二

答：进位位一样，溢出位值不一样

方法一中首先对cin位做n位扩展，再与B异或，这样当cin为0时，t\_no\_cin得到的就是B，当cin为1时得到B按位取反的结果，之后也是判断t\_no\_cin的符号位

方法二中t\_add\_cin相比于t\_no\_cin加上了cin，做加法时即为B，做减法时即为-B的补码，因此差别就在于第一步是否加上cin的值

当A=0x01（真值1），B=0x80（真值-128）时A-B=129溢出，Result为0x81(-127),按照方法一判断出溢出，而方法二判断不出溢出，因此方法一正确。

* 思考题三

答：一元约简操作是单目操作符，对单个操作数进行运算最后返回一位数，运算过程是首先将操作数的第一位和第二位进行与、或、非操作，再将运算结果与第三位进行与、或、非操作，以此类推到最后一位

|Result即为将每一位或，只要有一位不为0结果则不为0

二．简单加减法运算器的设计

（1）aluop设计

共四位，前三位为操作码，指示进行何种运算，最后一位为使能端

|  |  |  |
| --- | --- | --- |
| aluop | 运算 | 表达式 |
| 0001 | 加法 | A+B |
| 0011 | 减法 | A-B |
| Xxx0 | 不做运算 | 无 |

在硬件中，四个按钮即KEY[3:0]代表aluop

（2）加减法设计及符号位设计

加法设计不再赘述，减法将减数取反加一，再进行加法

{LEDR[4],LEDR[3:0]}=(~SW[3:0]+1)+SW[7:4];

设置CF（进位位），OF（溢出位），ZF（零位）

SW[7:4]代表A，SW[3:0]代表B，LEDR[3:0]代表结果，LEDR[4]代表CF，LEDR[5]代表OF，LEDR[6]代表ZF

CF： {LEDR[4],LEDR[3:0]}=(~SW[3:0]+1)+SW[7:4];//LEDR[4]作为进位位

OF： a=~SW[3:0];

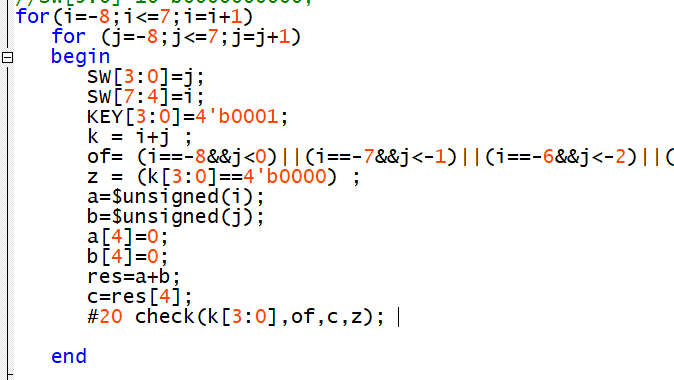
LEDR[5]=(SW[7]==a[3])&&(SW[7]!=LEDR[3]);

//采用思考题二的方法一，a存放的是减数取反之后的结果

ZF: LEDR[6]=(LEDR[3:0]==4'b0000);//判断是否全零

（3）测试设计

测试代码：



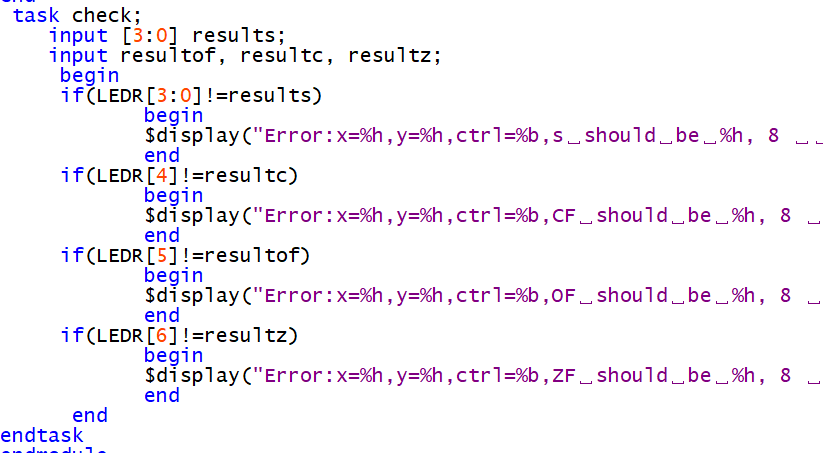
其中a和b是将i和j转化为无符号数的结果，res存放加法结果（包括最高进位），用于CF的测试

OF的测试：列举出所有OF为1的情况

CF的测试：由于CF是无符号溢出的标志，将i和j转化为无符号数，相加检测最高位有无产生进位

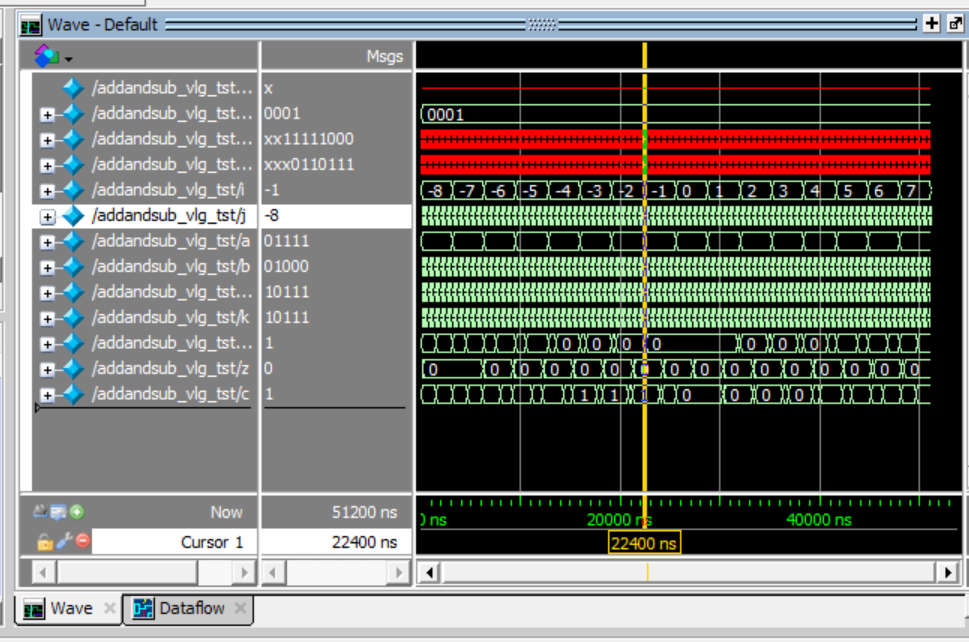
ZF的测试：依然是检测是否全为0

调用的task-check:



依次比较了结果，CF,OF,和ZF

（4）仿真模拟



三．带有逻辑功能的简单alu

（1）加减法设计

由于操作数仍是4位，与上述相同，操作数和符号位的设置也与上述相同

（2）取反，与，或，异或，

LEDR[3:0]=~SW[7:4];//其他逻辑操作替换位相应操作符即可

LEDR[5:4]=2'b00;//包括之后的逻辑操作，CF，OF直接设为0

LEDR[6]=(LEDR[3:0]==4'b0000);

(3)比较大小

使用LEDR[7]作为判断的结果，A>B，out=1,，否则为0

使用减法比较两个数的大小，当OF=SF（结果最高位）且ZF不为0 时，A>B

LEDR[7]=(LEDR[5]==LEDR[3])&&(LEDR[6]!=1);

（4）判断相等

同样使用减法，直接将ZF当作结果，ZF有效时说明相等

（5）测试代码

由于之前的实验对加减已经做了测试，这里主要验证判断大小的正确性

for(i=-8;i<=7;i=i+1)

for (j=-8;j<=7;j=j+1)

begin

SW[3:0]=j;

SW[7:4]=i;

KEY[3:0]=4'b1101;

out=(i>j);//这里out是正确结果

#20 check(out);

end

task check;

input out;

begin

if(LEDR[7]!=out)

begin

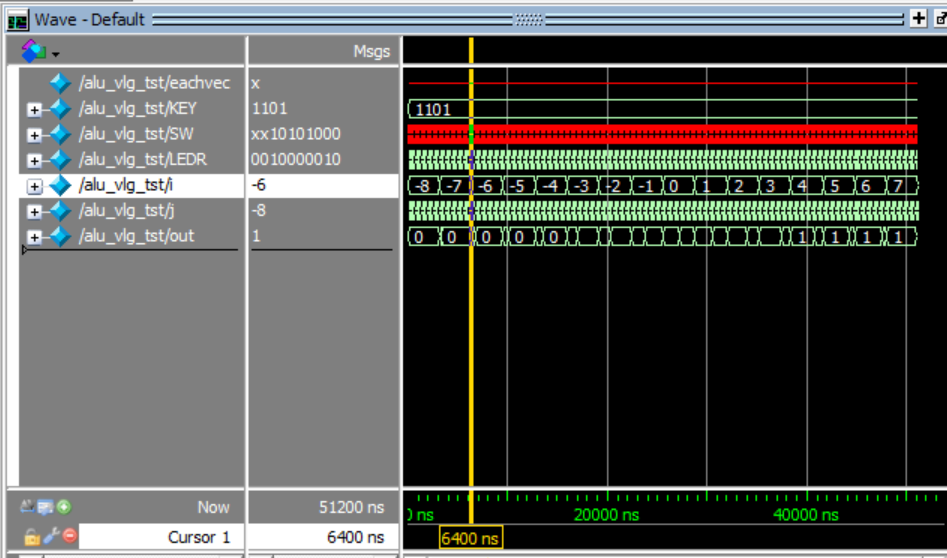
$display("Error:x=%h,y=%h,ctrl=%b,s␣should␣be␣%h, 8 ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣get␣%h", SW[7:4], SW[3:0], KEY[3:0], out, LEDR[7]);

end

end

endtask

（6）仿真模拟



四．实验反思和收获

（1）测试代码中的变量需要在initial外定义，另外测试代码需要注意语法正确，否则modelsim都无法正常运行

（2）熟悉了在测试代码中使用for循环和task进行大量测试

（3）开发板上的按钮也是低电平有效，按下时为0