实验名称：实验三 加法器与 ALU

姓名：张涵之

学号：191220154

班级：周一5-6

邮箱：[191220154@smail.nju.edu.cn](mailto:191220154@smail.nju.edu.cn)

实验时间：2020/9/22

3.3.1 简单加减法运算器的设计

实验目的：完成一个进行补码加减运算4位加减运算器，能够根据控制端完成加、减运算，能判断结果是否为0，是否溢出，是否有进位等。输入的操作数A和B都是补码。

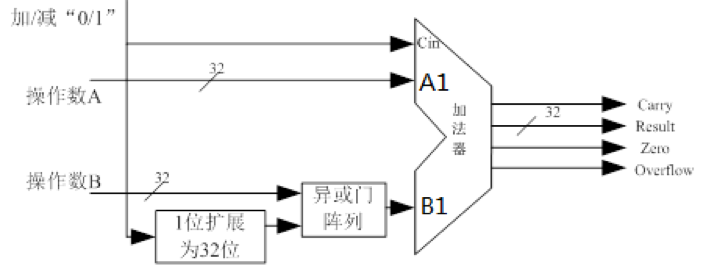
输入信号有：两个4位的参与运算的数据的补码操作数A和操作数B，

一个控制做加法还是做减法的加/减控制端Cin。

输出信号有：一个4位的结果 Result、一位进位位Carry，一位溢出位Overflow，

一位判断结果是否为零的输出位Zero。

实验原理：此加减运算器的核心部件是一个4位加法器。



进行减法运算时，对操作数B取负，已知一个4位二进制数的表示范围是-8~7，4位二进制数和它的相反数的补码代数和为二进制数10000（0除外，0000的补码仍为0000，即结果仅保留低4位），则取B1=(16–B)%16得到-B的补码表示。

计算结果：运算结果为补码相加和的低4位。

判断进位：最高位产生进位，即{out\_c,out\_s} = in\_x + in\_y；

判断溢出：负数相加得到正数/非负数相加得到负数即溢出，即

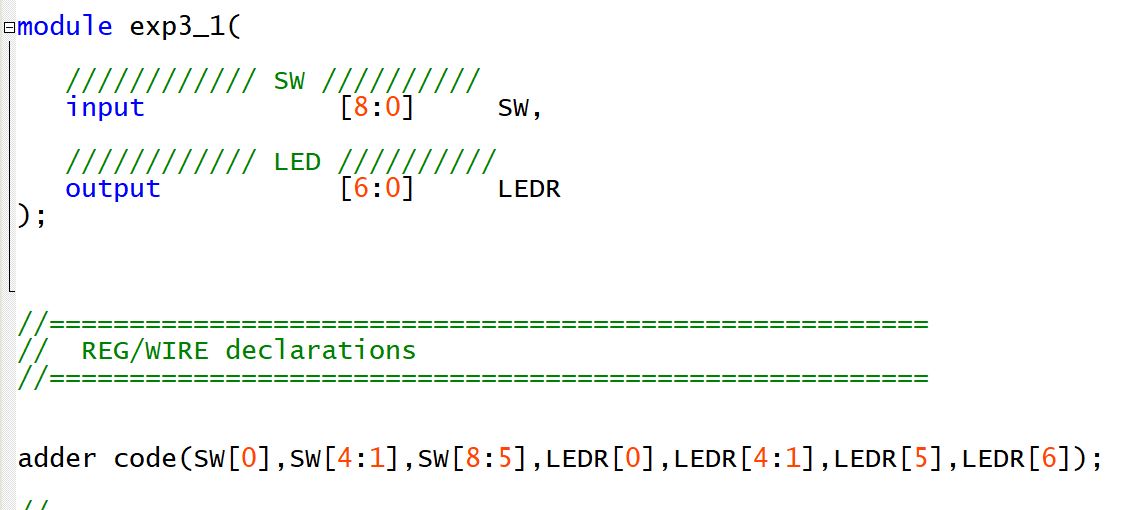
Overflow = (in\_xn 1 == in\_yn 1)&&(out\_sn 1! = in\_xn 1)；

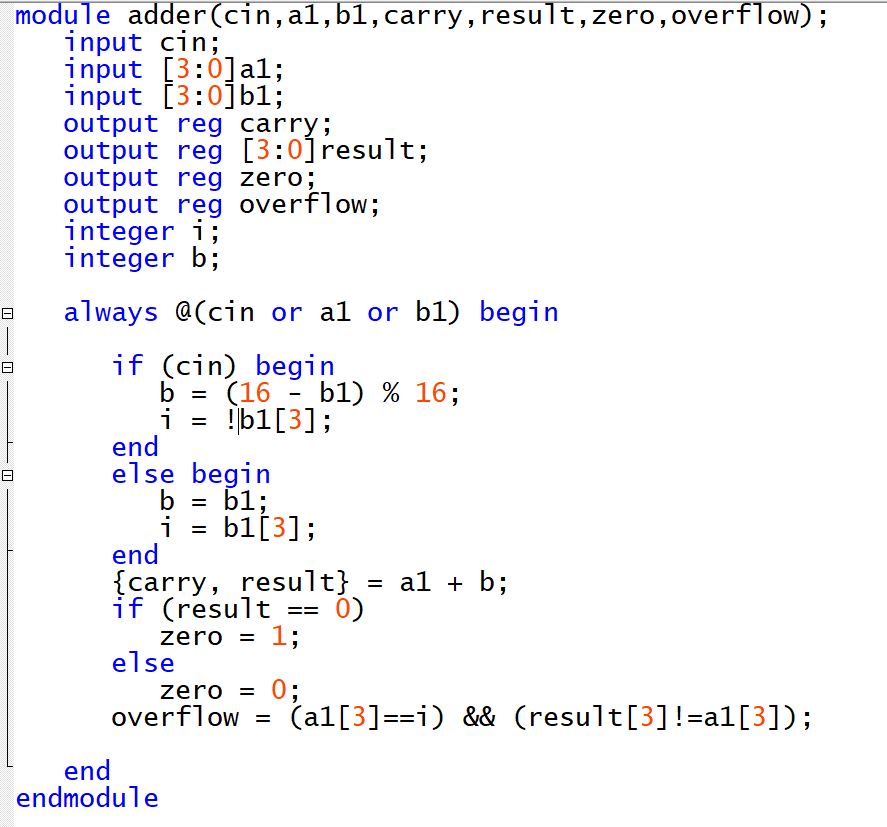
判断是否为零：out\_s == 0则结果为零。

边界条件问题：受到二进制补码表示的局限，减数为-8时，对B1=16-B=1000，取值仍为-8，-(-8)转化为+(-8)而不是+8，取B1最高位与A1比较无法得到正确的溢出位，故Cin为1时应该直接对B的最高位取反与A1进行比较，从而得到正确的溢出位。

实验环境/器材：实验箱一个，笔记本电脑一台。

程序代码或流程图：





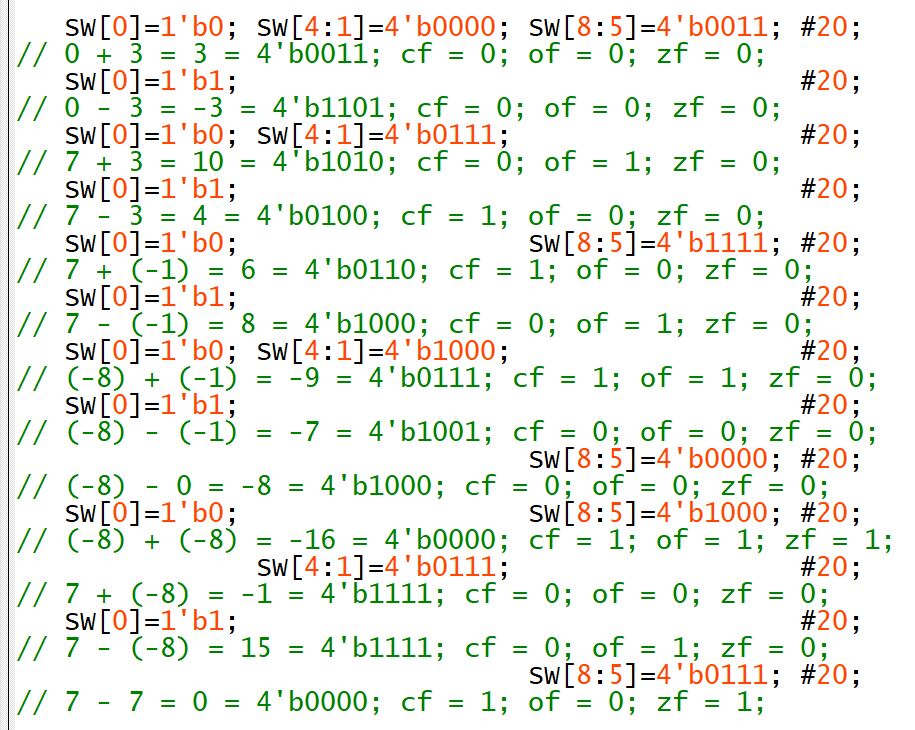
A1=A，B1根据Cin判断是否取负 —> 运算和设置标志位。

实验步骤/过程：

根据Cin判断操作数B是否需要取负，若为减法则求B1为-B的补码。

运算A+B1，得出结果并设置Carry，Overflow和Zero标志位。

测试方法：



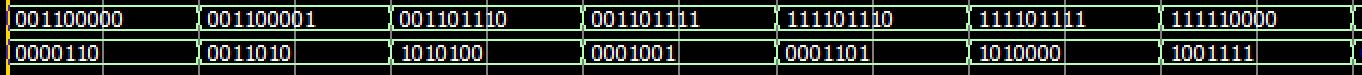
其中SW[0]为Cin，SW[4:1]和SW[8:5]分别为操作数A和B的补码。

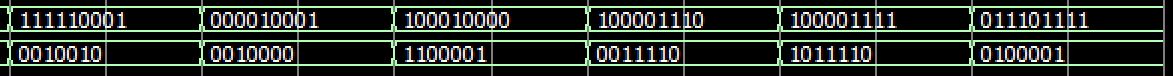
LEDR[0]为Carry，LEDR[4:1]为Result，LEDR[5]为Zero，LEDR[6]为Overflow。

根据输入的测试数据手动计算结果和判断标志位，用于仿真时对照判断。

样例设置进位/不进位，溢出/不溢出，零/非零，并对0和(-8)的加减进行额外测试。

实验结果：





通过观察对比，运算结果与符号位均与手动计算所得相符。

经接入实验箱检验，显示也符合预期。

实验中遇到的问题及解决办法：

对减0和减-8时边界条件的处理：

1）0的补码仍为0，用二进制数10000做减法需舍去最高位，模16可以解决；

2）对-8取补码仍为-8而非8，造成溢出位的判断发生错误，结合计算机系统基础相关知识得出运算时不应先取负再取相反数的最高位，而应直接对原操作数最高位取反。

实验得到的启示：应该关注特殊值和边界条件的处理，从具体例子发现和解决问题。

意见和建议：无。

思考题1：应该比较A、B和运算结果的符号位，其中若Cin为1则B的符号位取反。若比较A1、B1和运算结果的符号位，则当Cin为1，B1为-8时出现问题。

思考题2：两种方法的产生的运算结果一样，进位位和溢出位不完全一样。

方法一在做减法且B为0时出现问题，0000的相反数补码为0000，计算B1为10000，则进位位为1，而任何数减0都不应该有借位，是第二步中相反数补码计算错误造成的。

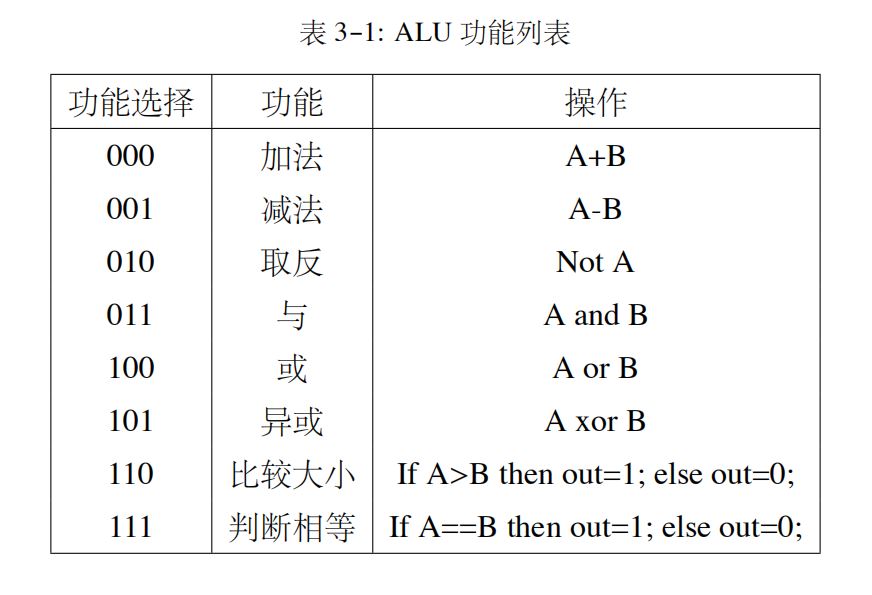
方法二在做减法且B为-8时出现问题，1000的相反数补码为1000，计算B1为1000，该补码表示的是-8而非8，符号位为1而非0，则根据B1符号位判断的溢出位错误，是第四步中使用相反数符号位而非对原操作数符号位取反造成的。

思考题3：assign zero = ~(| Result)，其中一元约简运算的操作过程为对操作数各位依次执行同一操作直到最后一位，此处Result为4位二进制数，相当于zero = ~(Result[0] | Result[1] | Result[2] | Result3)，即Result各位全部为0时zero = 1，否则zero = 0。

3.3.2 实现一个带有逻辑运算的简单ALU

实验目的：实现对4位有符号数操作的ALU，SW作为数据输入，button为选择端。

其中选择端共可以实现八种功能，输入的操作数均为有符号数的补码表示。



实验原理：用case语句选择需要实现的功能：

加法与减法：原理同3-3-1，根据加减法确定操作数B是否需要取负，对A1和B1进行补

码的加法运算得到计算结果，并设置相应的进位和溢出标志位；

取反、与、或、异或：对操作数进行简单逻辑运算即可。

比较大小：分类讨论，若A和B一正一负可直接得出大小关系；

若A和B均为正数，则补码和原码相同，可以通过A>B直接判断；

若A和B均为负数，则它们与各自相反数的补码之和为二进制数10000，补码的无符

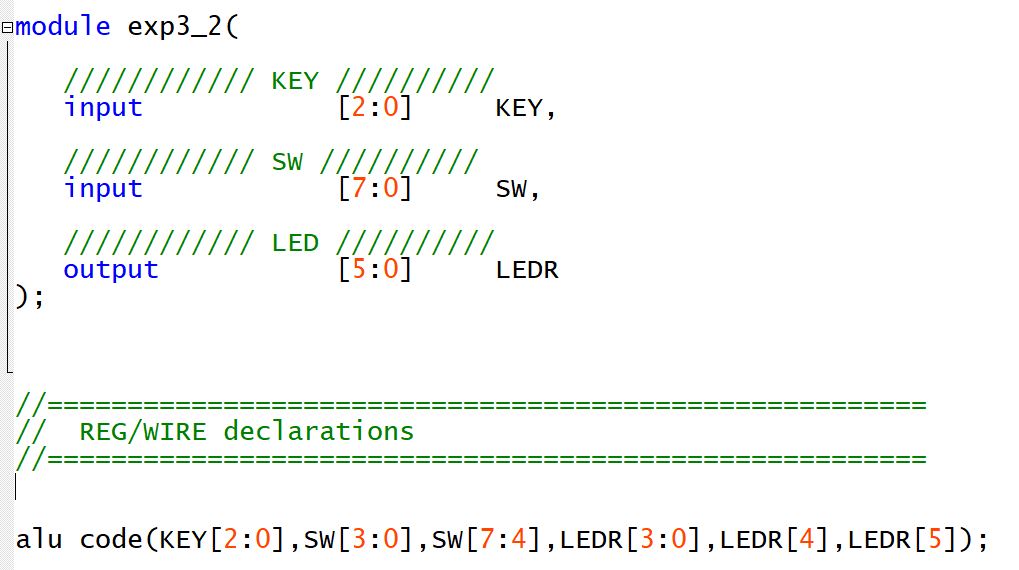
号值越大，其绝对值越小，真值越大，也可以通过A>B直接判断；

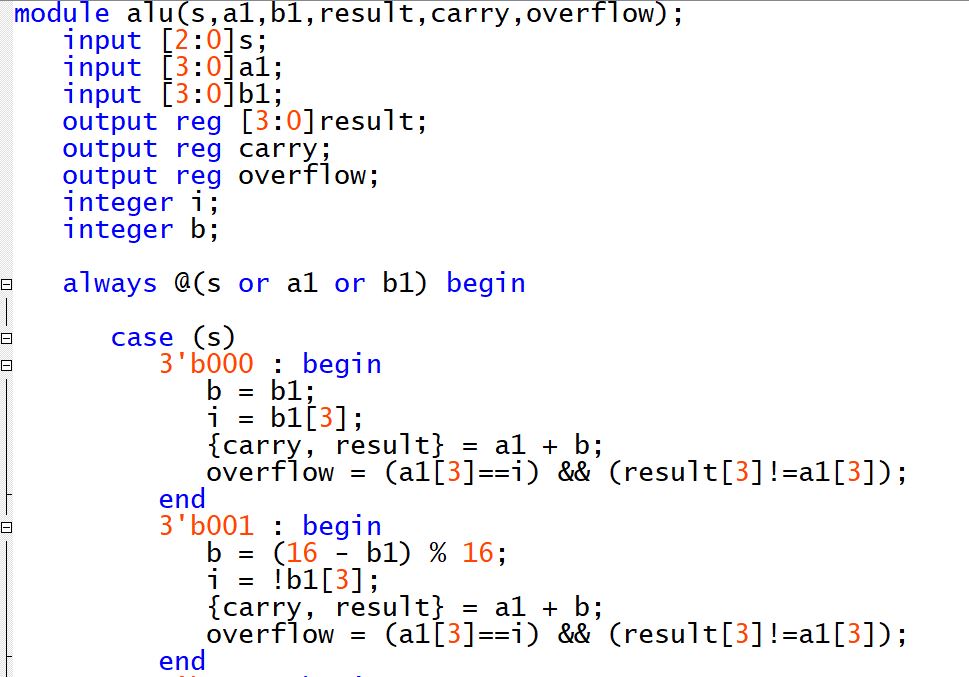
判断相等：直接使用==判断相等。

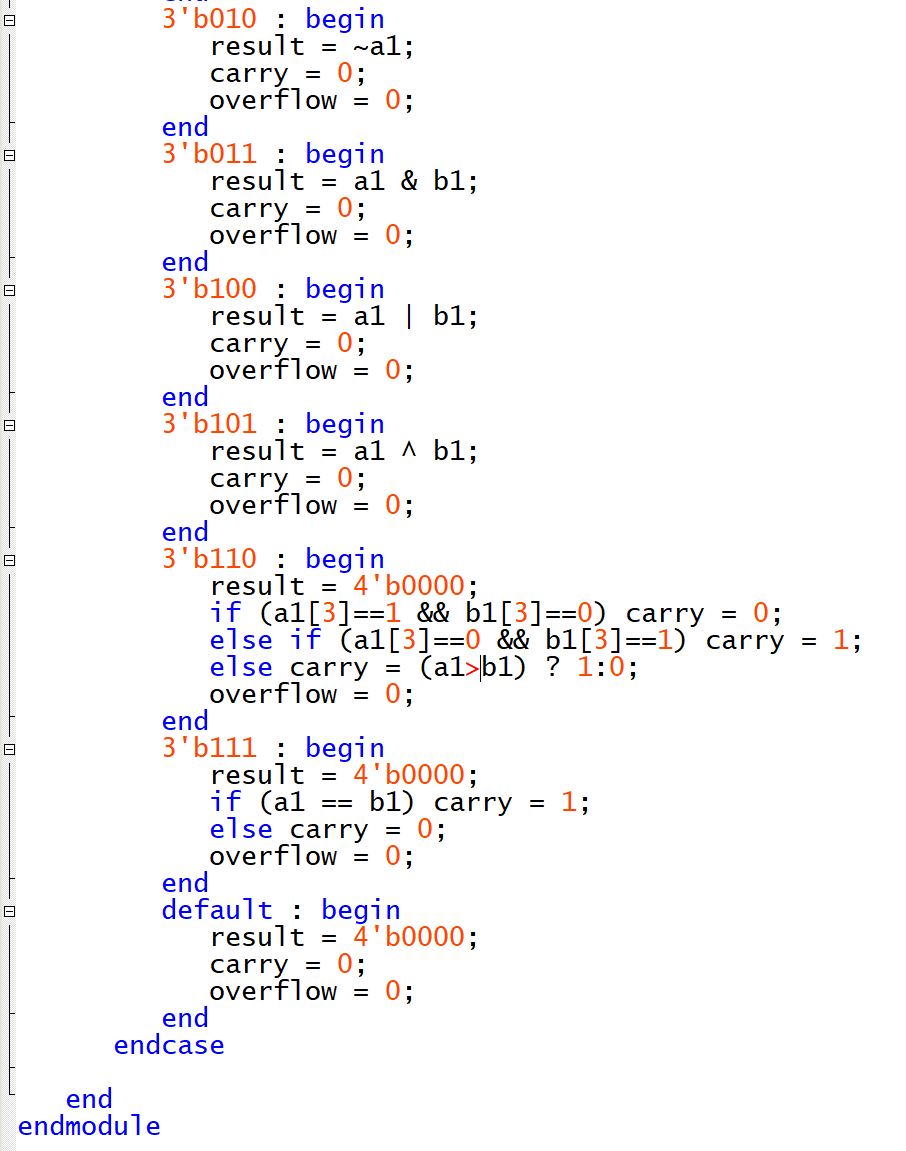
实验环境/器材：实验箱一个，笔记本电脑一台。

程序代码或流程图：

输入S —case语句—> 功能选择 —输入A和B—> 进行运算 —> 输出结果。







实验步骤/过程：

根据Cin判断操作数B是否需要取负，若为减法则求B1为-B的补码。

运算A+B1，得出结果并设置Carry，Overflow和Zero标志位。

测试方法：

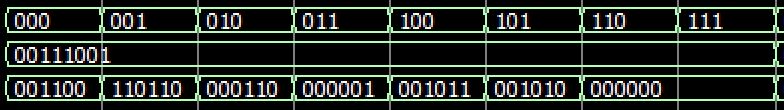


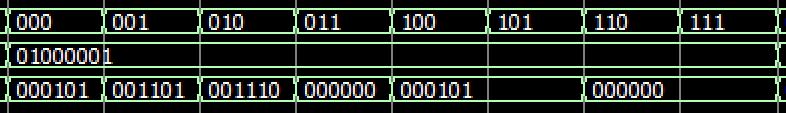
其中KEY[2:0]为功能选择端，SW[3:0]和SW[7:4]分别为操作数A和B的补码。

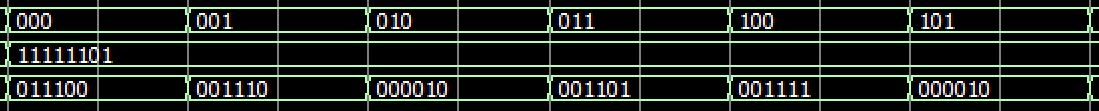
LEDR[3:0]为Result，LEDR[4]为Carry，LEDR[5]为Overflow。

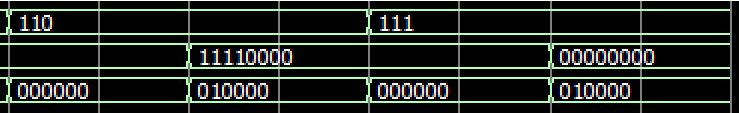
根据输入的测试数据手动计算结果和判断标志位，用于仿真时对照判断。

实验结果：









通过观察对比，运算结果与符号位均与手动计算所得相符。

经接入实验箱检验，显示也符合预期。

实验中遇到的问题及解决办法：问题都在3-3-1解决了，3-3-2没有出现问题。

实验得到的启示：好玩。

意见和建议：无。