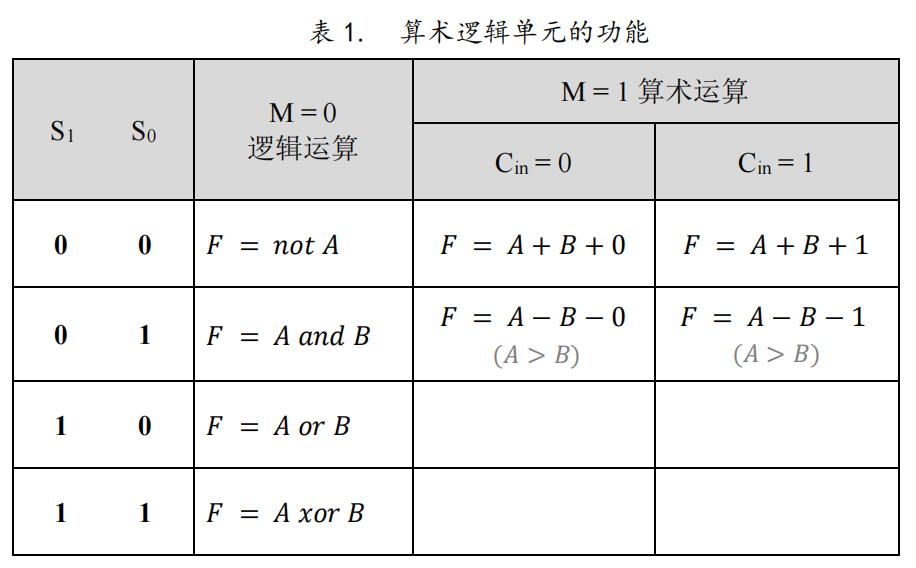
**数字逻辑与部件设计实验报告**

**实验02：【算术逻辑单元（ALU）】**

**姓名：刘丹桐 学号：21307090080 日期：2023.9.20**

**一：实验内容：**

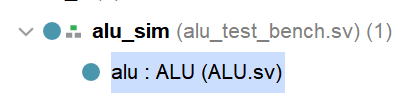
1．设计一个4位算术逻辑单元，输入为两组4位数据（a[3:0]，b[3:0]）和一个进位输入信号cin；输出为4位运算结果（y[3:0]）和一个进位输出信号cout。（以上数据均为无符号正整数。）功能控制信号有：S1、S0、M，当M=0时ALU执行逻辑运算，M=1时ALU执行算术运算，S1、S0 的功能如下表所示。



2. 在 NEXYS4 开发板上实现上述设计，利用十六个拨码开关输入参与运算的两个数（a、b）以及功能控制信号（S1、S0、M），利用 8 个七段数码管显示十六进制的输入数据和输出数据。

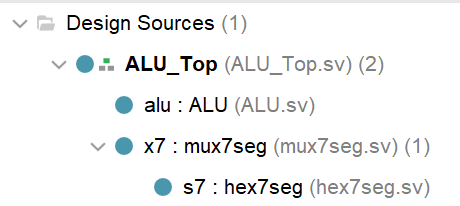
**二：实验方案：**

1.依据课件上给出的ALU设计方案，删除不必要的输出端口（nf、zf、cf、ovf），增加要求的输入和输出端口（cin与cout），令alusel[1]、alusel[1]、slusel[0]分别表示S1、S0和M，并根据上表重写alusel在不同取值下的逻辑，写出ALU模块并编写仿真文件进行仿真测试。



2.在1和实验二【七段数码管】的代码基础上，为ALU编写顶层文件（ALU\_Top），ALU\_Top的输入有CLK100MHZ（时钟）、SW[15:0]（十六个拨码开关）、BTNC（用于复位），输出有DP（小数点）、AN[7:0]（七段数码管的使能）以及A2G[6:0]（对应七段数码管）。ALU\_Top调用ALU模块和mux7seg模块，将SW传入的数据（a、b、cin）与控制信号（alusel）输入ALU模块，ALU模块计算后输出结果（y、cout）。根据运算的类型，写出控制各个数码管显示对应数值或符号的组合逻辑，并将处理后的输入输出数据整理为一个变量x，将x传入Mux7Seg模块进行显示。

整体层次结构和RTL原理图如下：

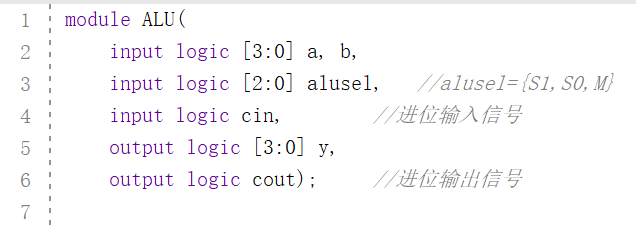




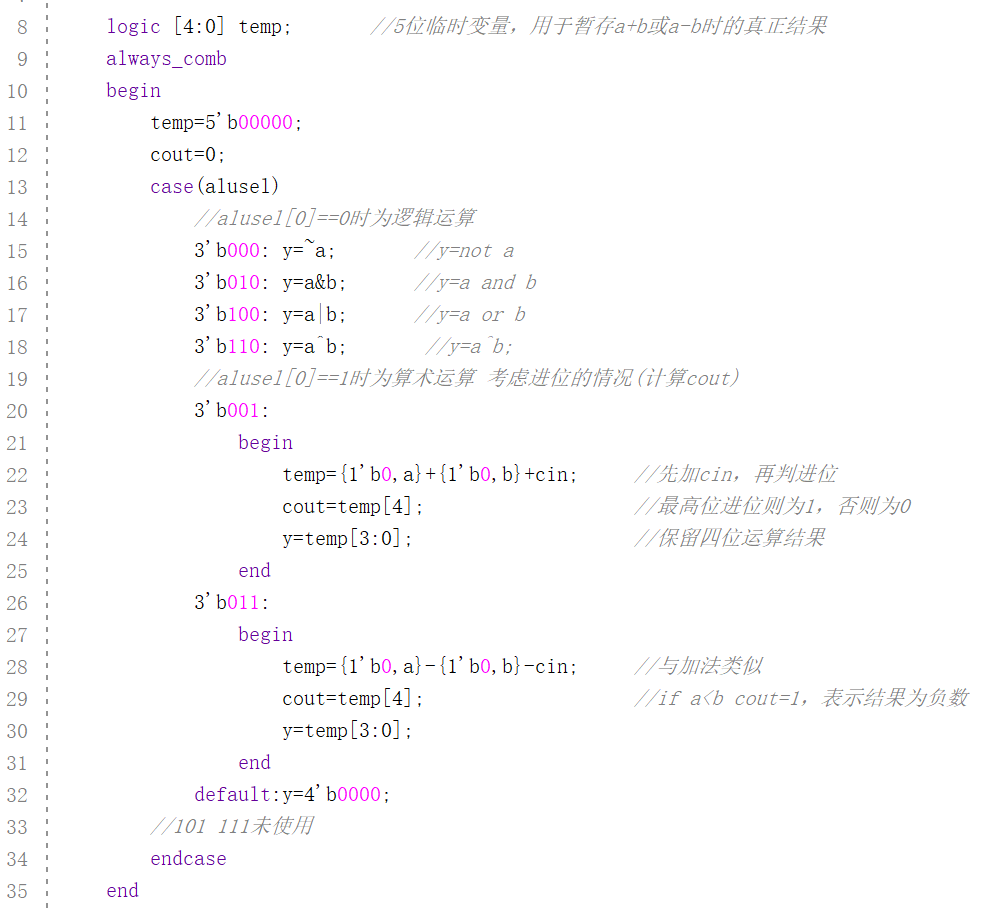
**三、代码实现：**

**1.ALU模块（ALU.sv）**

输入输出端口：

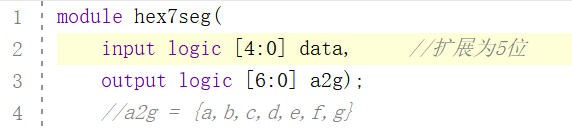


组合逻辑：

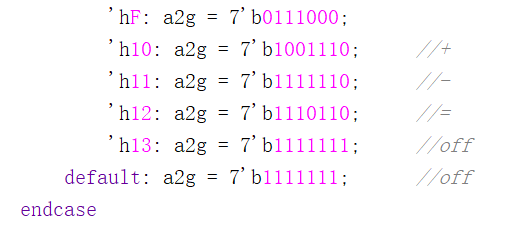


**2.七段数码管的十六进制译码器（hex7seg.sv）**

在本次实验中，七段数码管除了需要显示0-E十六位数值，还需要显示+、-和=或不显示任何东西（LED不亮），因此需要将输入数据由4位扩充至5位，使其能够表示多于16种图案。

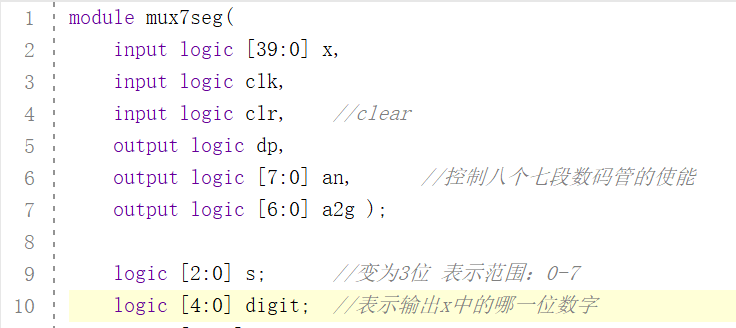


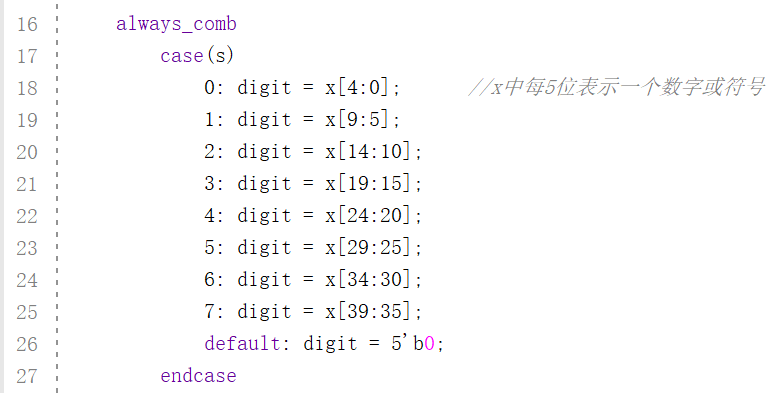
同时在组合逻辑中增加data的可能取值（'h10、'h11、'h12、'h13），并写出对应的a2g。default设置为不亮。



**3.七段数码管的分时复用模块（mux7seg.sv）**

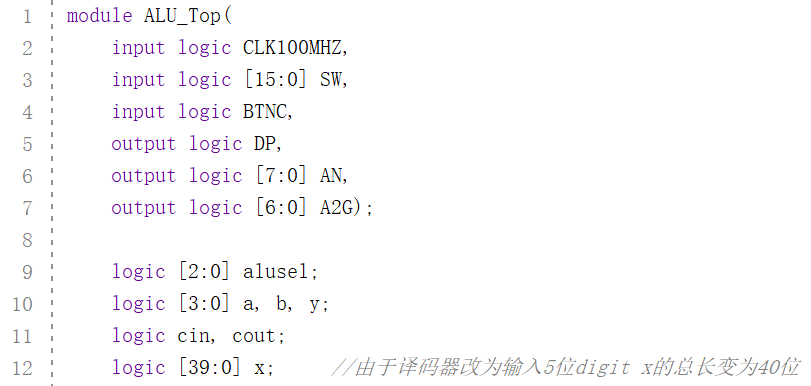
此外，对作出相应的，将x从32（8\*4）位扩充至40（8\*5）位。每个七段数码管所显示的digit也为5位。其他部分沿用上次实验的代码不变。



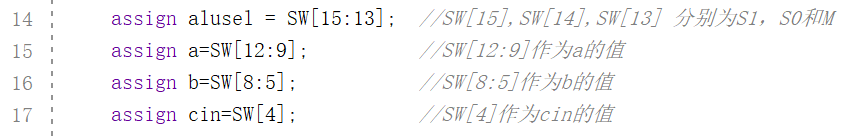


**4.顶层文件（ALU\_Top.sv）**

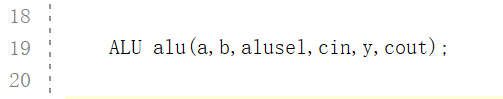
输入输出端口与定义变量：



将SW输入的数据进行赋值：



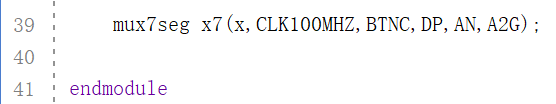
调用ALU：



定义七段数码管应显示的数据：



将构造好的数据传入mux7seg使其在开发板上进行显示：

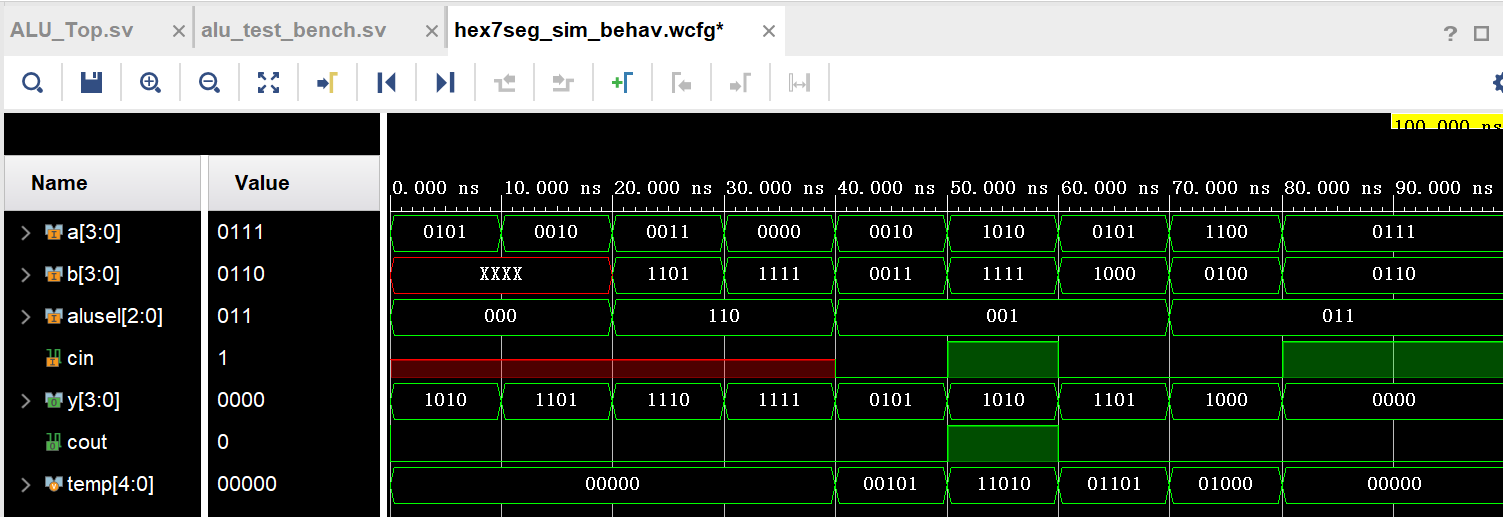


（BTNC用于七段数码管的复位）

**四、实验现象：**

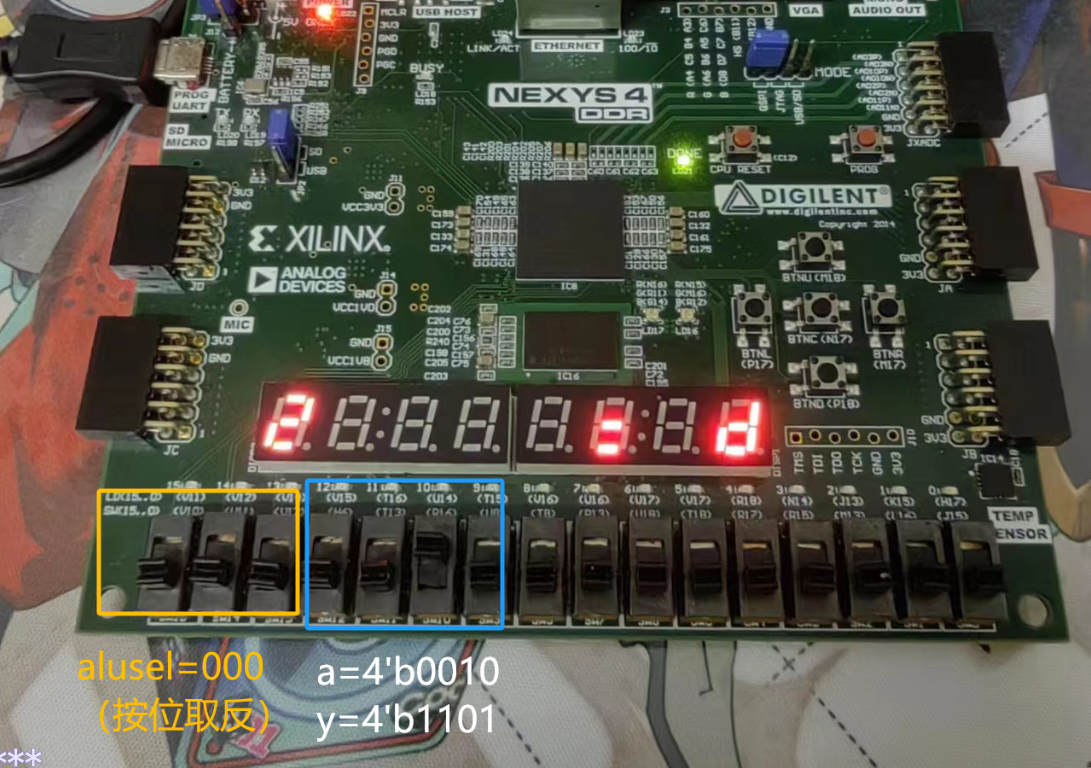
**1.仿真文件（alu\_sim.sv）：** 

**2.仿真结果：**

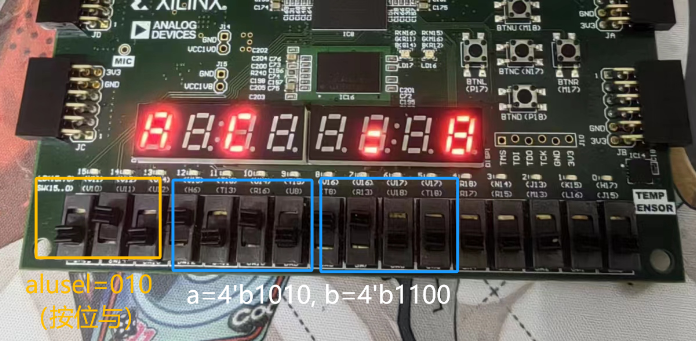


**3.开发板照片：**

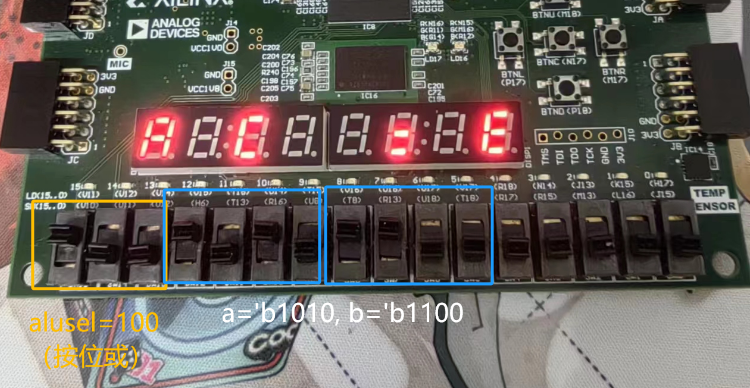
(1)y=~a



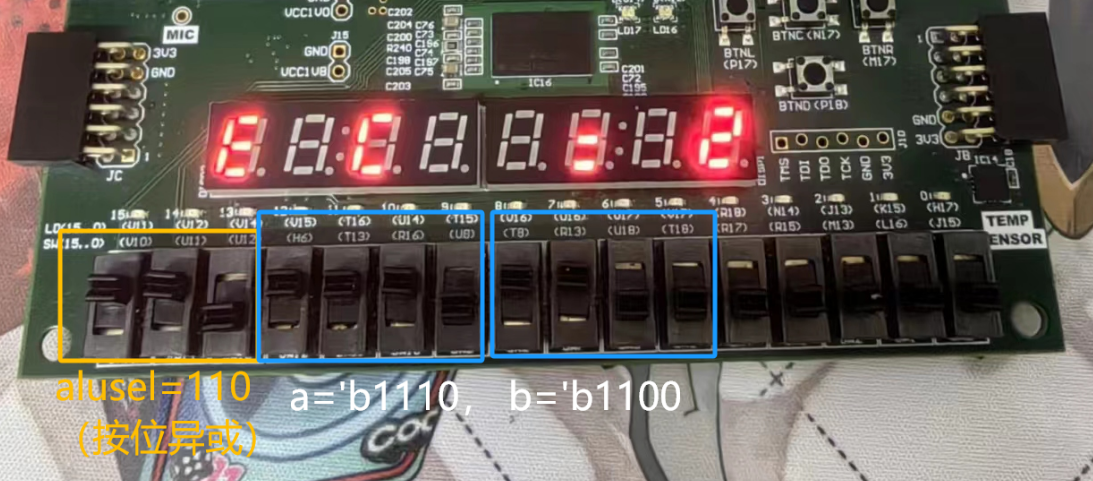
(2)y=a&b



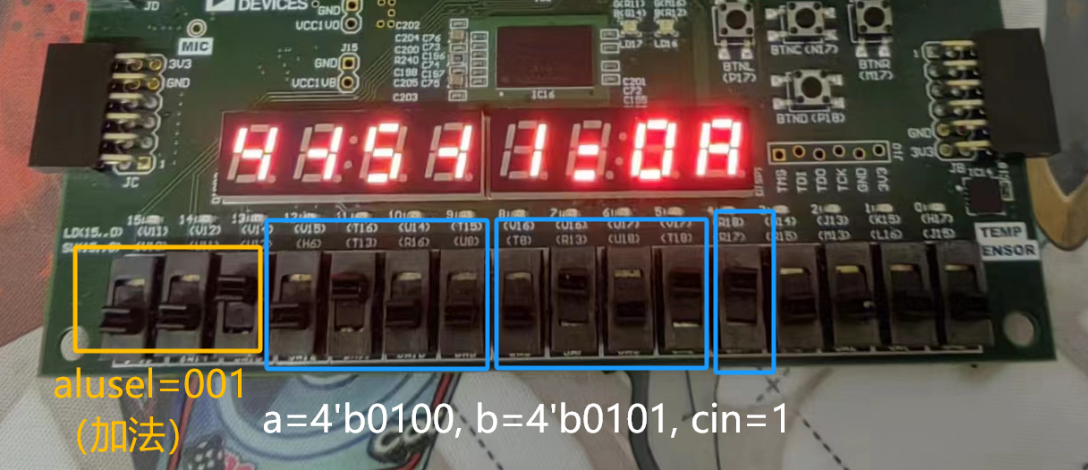
(3)y=a|b



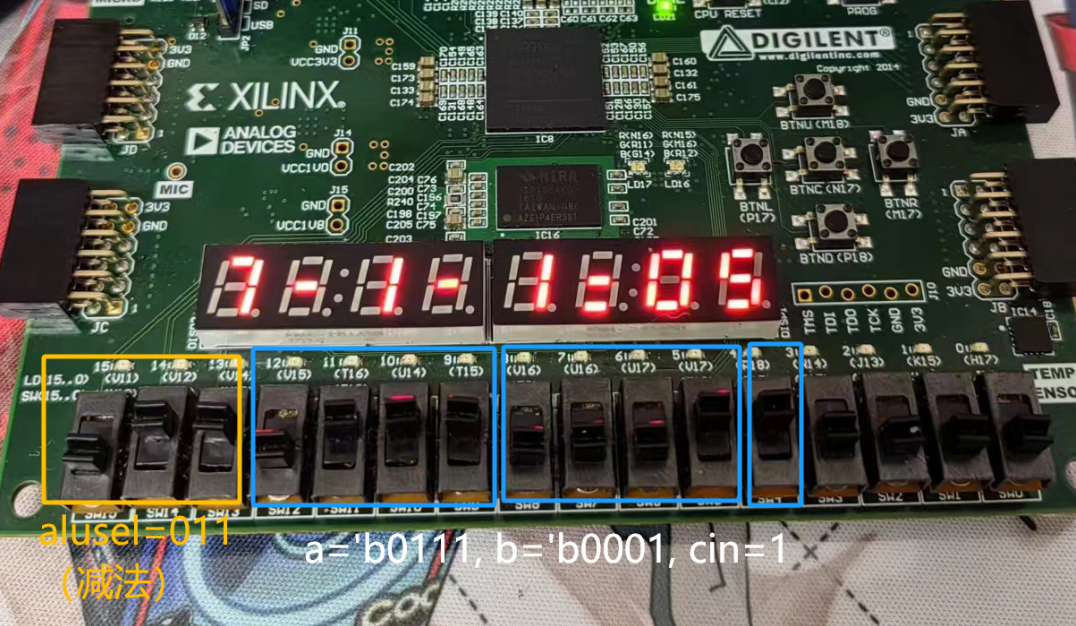
(4)y=a^b



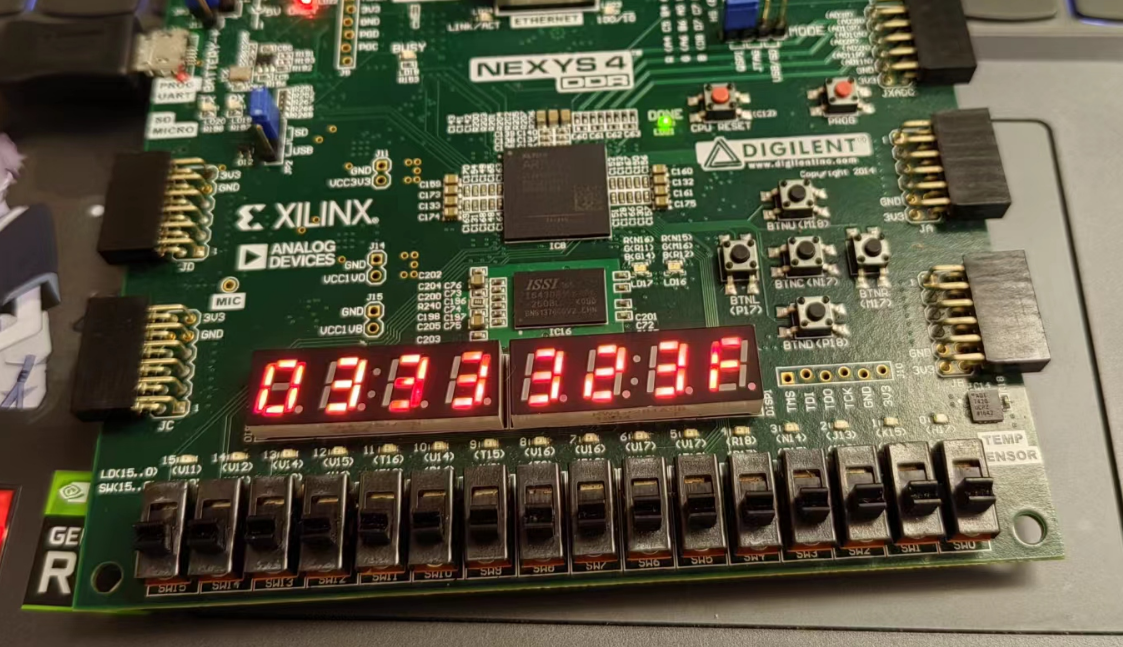
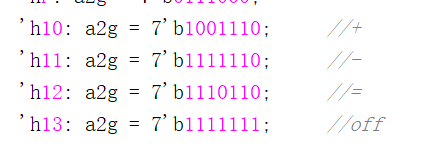
(5)y=a+b+cin



(6)y=a-b-cin



**五、遇到的问题和解决方案：**

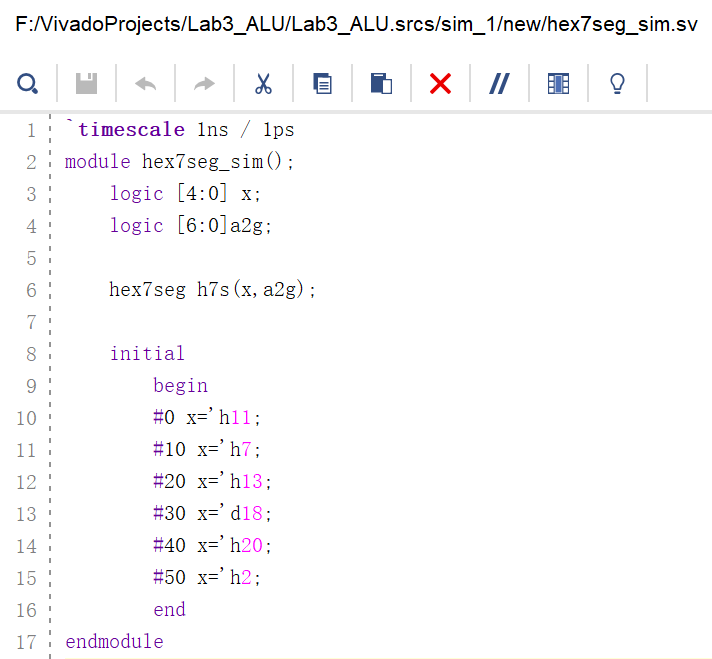
**1.无法正确显示加号、减号、等号和不亮：**  

如图所示，所有拨码开关均为0，alusel=000，即按位取反。七段数码管的第一个和最后一个分别为0和F，证明运算是正常的，只有显示出了问题。检查了一下译码器模块，发现本该显示'h13和'h12对应的a2g分别显示了'h3和'h2对应的a2g，正好是它们模16所得的数。错误原因可能有两处：

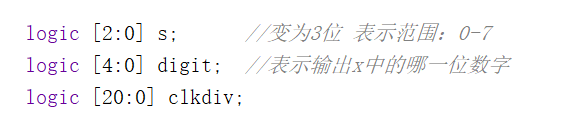
（1）mux7seg将数据传入hex7seg时没有传入最高位；

（2）顶层文件将需要显示的数据传到mux7seg时没有传入最高位；

首先检查mux7seg向hex7seg的输入输出，我写了一个仿真文件测试hex7seg，发现输入'h10~'h13时，输出的a2g是正确的，因此排除（1）。



完整地检查了一遍顶层文件的代码，每一行都没有问题。错误应该在mux7seg模块中。本想再写一个仿真文件，但是不太会模拟时钟信号，遂放弃。转而逐行检查mux7seg文件的源代码，发现原来是沿用上次的代码忘记改digit的位数，也就是下图第二行中，用于分割x的单位少了一位，因此最高位被截掉了。



修改过后果然成功了。

**2.计算加法的逻辑有误。**

最开始的逻辑是先判a+b是否溢出，再加进位cin。结果到板子上测试了9+6+1的数据，得到00而非预期的10。解决方法是调整判溢出和做加法的顺序，将代码改为先做加法后判溢出，从而解决了这一问题。

**六、总结与收获：**

本次实验比上次实验更难，我花了比想象中更久的时间去调试自己的代码，幸好开始的比较早，才能从容不迫地按时完成。在这次实验中，我主要锻炼了写仿真文件调试模块的能力，同时也意识到看报错和警告信息的重要性。有时出了问题自己像无头苍蝇一样不知道该修改哪里，但警告里很明确地指出了哪个位置有哪个问题，一下子就找到了bug所在。

另外和组员的交流讨论也帮我理清了代码逻辑和设计思路，让我逐渐学会以硬件的视角自下而上地去写代码。这次实验的成功使我对下次实验充满信心><