**计算机系统结构实验实验报告 Lab03**

姓名：叶增渝

学号：519030910168

日期：2021年04月06日

**目录**

[1.实验概述 3](#_Toc70973132)

[1.1 实验名称 3](#_Toc70973133)

[1.2 实验目的 3](#_Toc70973134)

[2.实验描述 3](#_Toc70973135)

[2.1 主控制器单元模块 3](#_Toc70973136)

[2.1.1Ctr模块描述 3](#_Toc70973137)

[2.1.2 Ctr模块代码 5](#_Toc70973138)

[2.1.3仿真激励代码 9](#_Toc70973139)

[2.1.4仿真波形 11](#_Toc70973140)

[2.1.5实验结论 11](#_Toc70973141)

[2.2算术逻辑单元（ALU）控制器模块 11](#_Toc70973142)

[2.2.1ALUCtr模块描述 11](#_Toc70973143)

[2.2.2ALUCtr模块代码 13](#_Toc70973144)

[2.2.5实验结论 15](#_Toc70973145)

[2.3ALU模块 15](#_Toc70973146)

[2.3.1ALU模块描述 15](#_Toc70973147)

[2.3.2ALU模块代码 16](#_Toc70973148)

[2.3.3ALU仿真激励代码 18](#_Toc70973149)

[2.3.4仿真波形 19](#_Toc70973150)

[2.3.5实验结论 19](#_Toc70973151)

[3.实验心得与总结 19](#_Toc70973152)

[4.参考资料 20](#_Toc70973153)

1.实验概述

1.1 实验名称  
简单的类 MIPS 单周期处理器部件实现 – 控制器，ALU

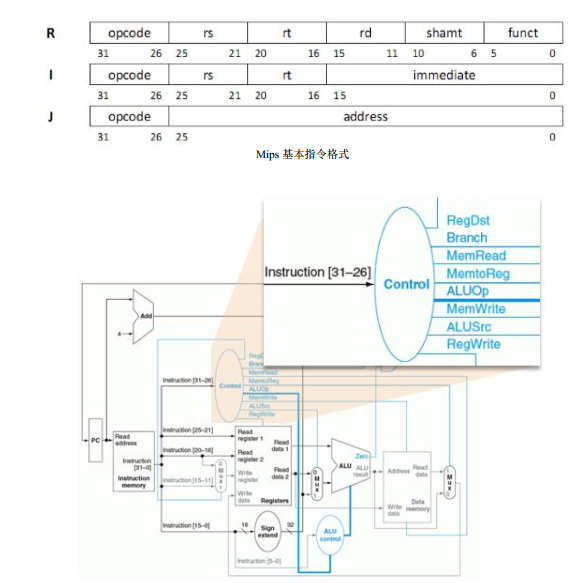
1.2 实验目的  
1.理解CPU控制器，ALU的原理  
2.主控制器 Ctr 的实现  
3.运算单元控制器 ALUCtr 的实现  
4.ALU的实现  
5.使用功能仿真

2.实验描述

# 2.1 主控制器单元模块

## 2.1.1Ctr模块描述

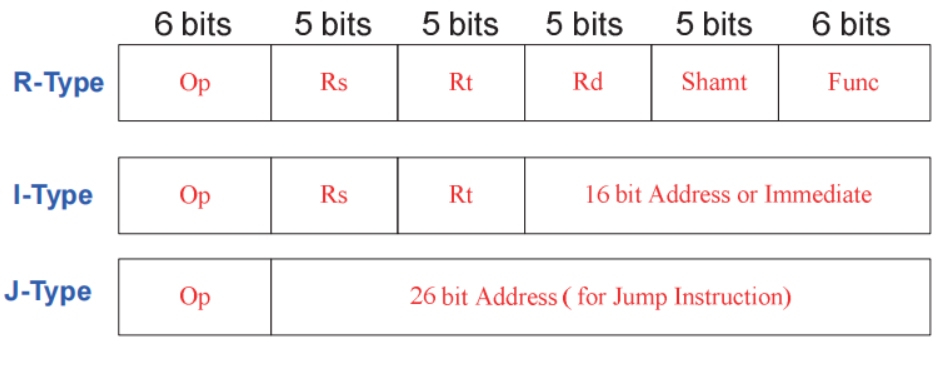
主控制器单元（Ctr）的输入为指令的 opCode字段，操作码经过 Ctr 的译码，给ALUCtr，Data Memory，Registers，Muxs等部件输出正确的控制信号



主控制模块的I/O定义

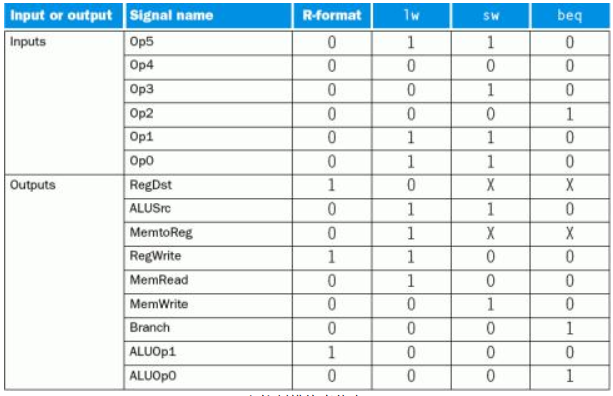
MIPS的指令是32位的，相当于一条指令的含义与操作细节完全由32个二进制数完全决定。对32位二进制数的不同划分使用，构成了不同的指令格式。

MIPS指令集有三种指令格式：R型指令，I型指令，J型指令，以下为各类型指令的具体编码方式

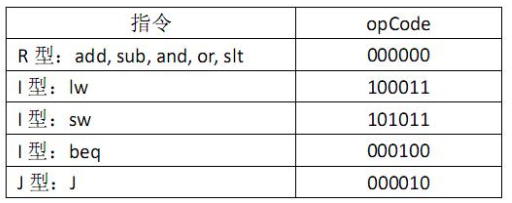


Mips指令基本格式

根据主模块的真值表（下图），可通过输入的opCode输出对应的控制信号



p.s.其中Jump指令编码是000010，Jump信号的输出当且仅当在Jump指令中输出为1，其余指令输出为0

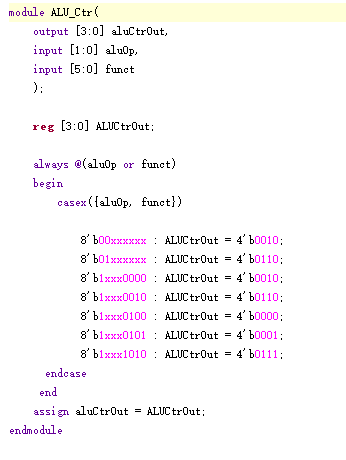
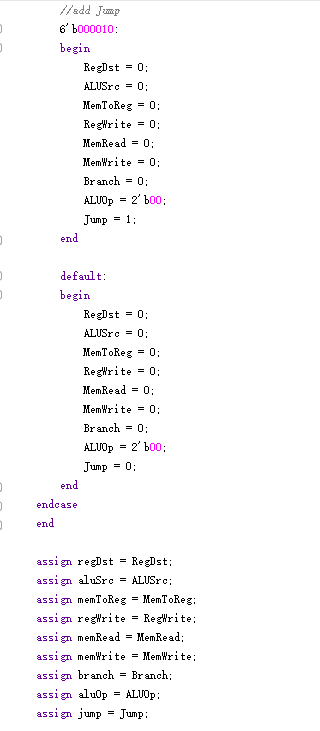
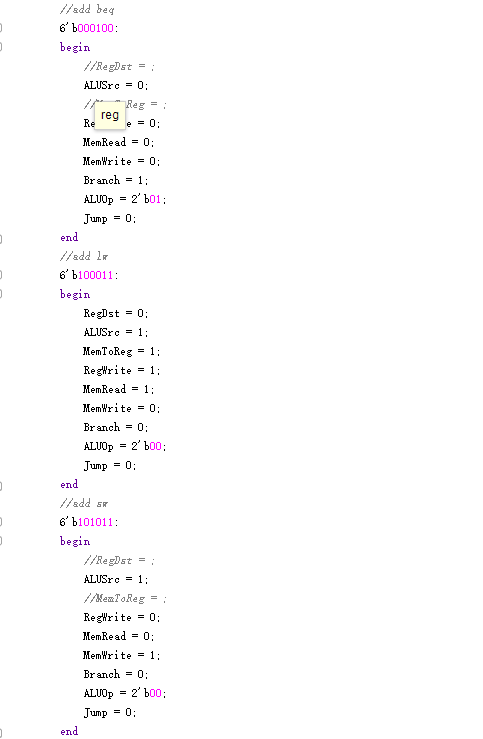


指令操作码

## 2.1.2 Ctr模块代码

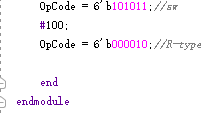
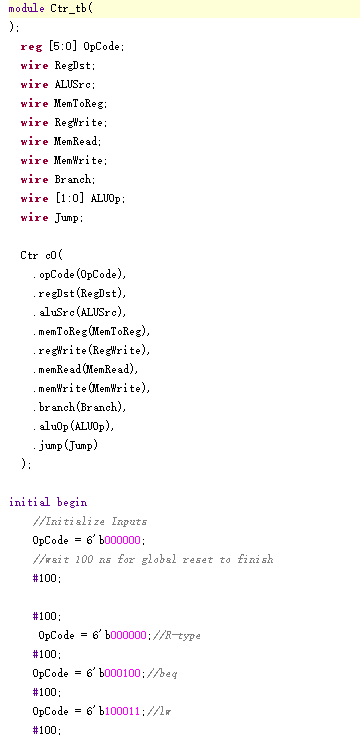
design source file主要利用case语句将opCode的输入分解成上述5种形式，对应真值表中的各值对应输出信号



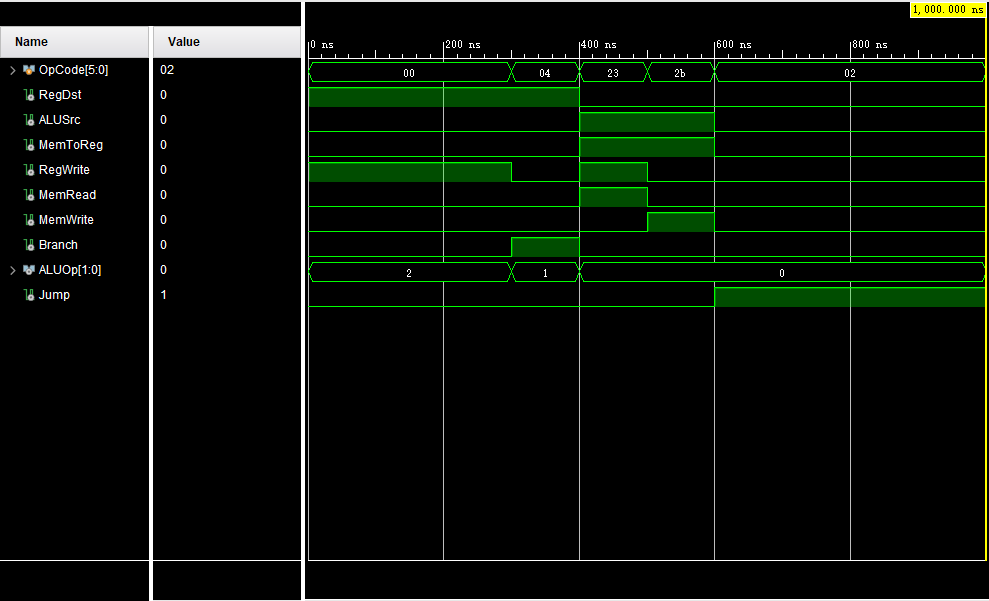


## 2.1.3仿真激励代码

激励文件主要通过每隔100ns更换输入来模拟总共5种指令操作码



## 2.1.4仿真波形



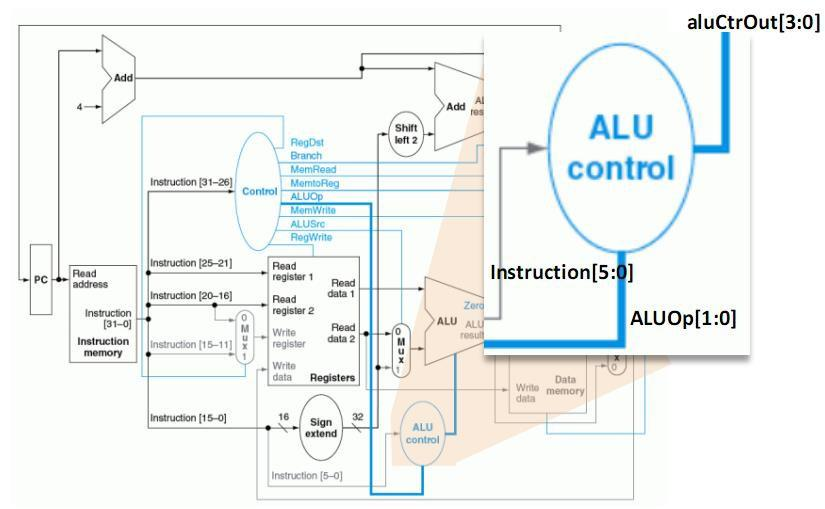
## 2.1.5实验结论

依次测试R型、beq、lw、sw、Jump指令，得到的仿真波形能完全对应Ctr模块真值表的要求，因此，我们能用使用Verilog编写程序，根据不同的输入opCode来输出对应的控制信号

# 2.2算术逻辑单元（ALU）控制器模块

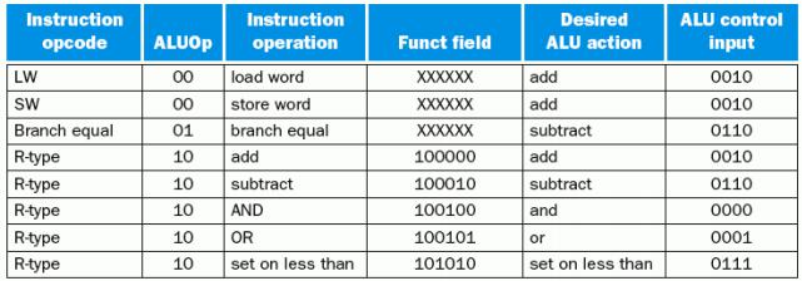
## 2.2.1ALUCtr模块描述

ALU 的控制器模块 （ALUCtr） 是根据主控制器的 ALUOp 来判断指令类型。 根据指令的后6位Funct[5:0]区分R型指令。综合这两种输入， 控制ALU做正确的操作。

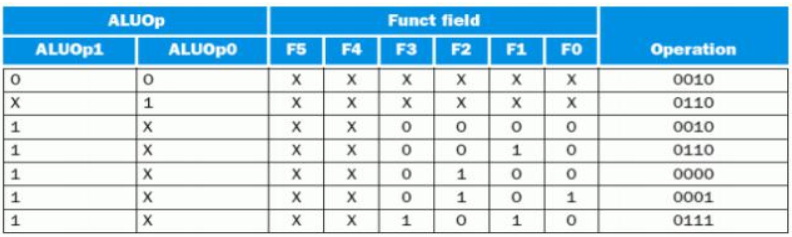


ALU控制模块

输入Funct、ALUOp与输出ALUctr的编码关系如下：



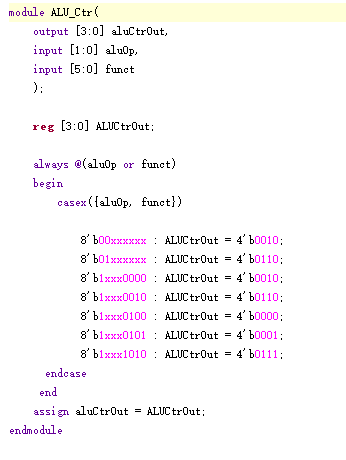
上表我们可以进一步转化为3个量之间的输入输出真值表



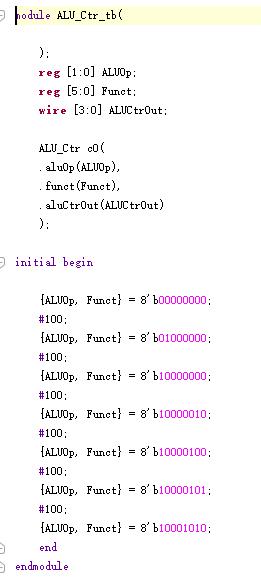
p.s.表中的x指代0或1均可，最后得到的输入不变，其中第二行ALUOp1中应为0，否则第二行输入与第三行输入有交叉，不符合定义

## 2.2.2ALUCtr模块代码

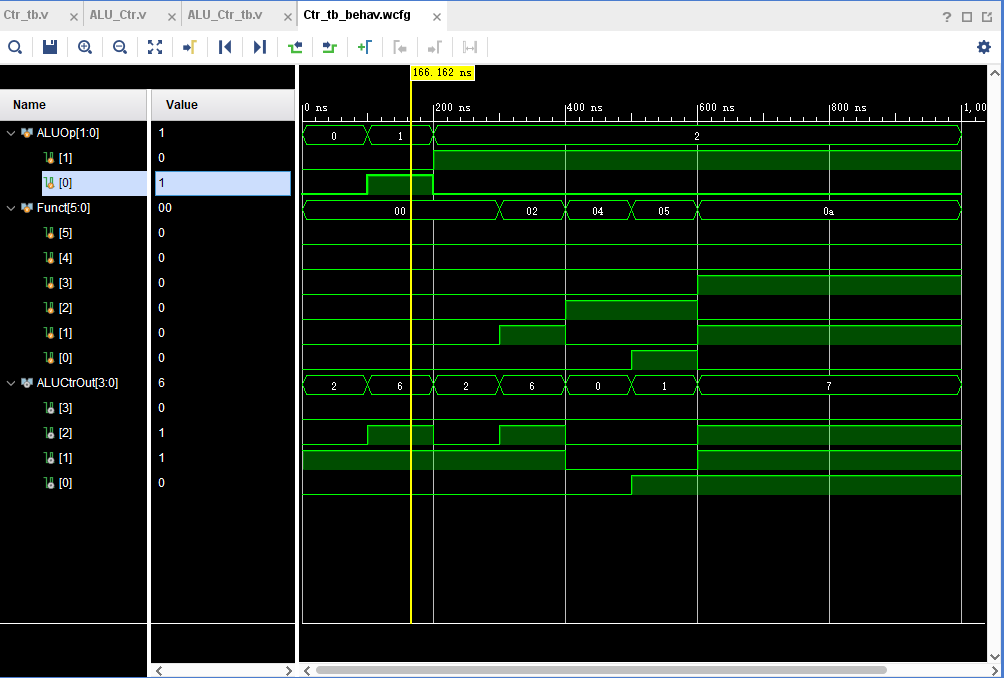
利用casex语句与{}位拼接运算符同时对ALUOp与Funct的各类情况进行分类，再按照真值表赋值



**2.2.3ALUCtr仿真激励代码**



**2.2.4仿真波形**



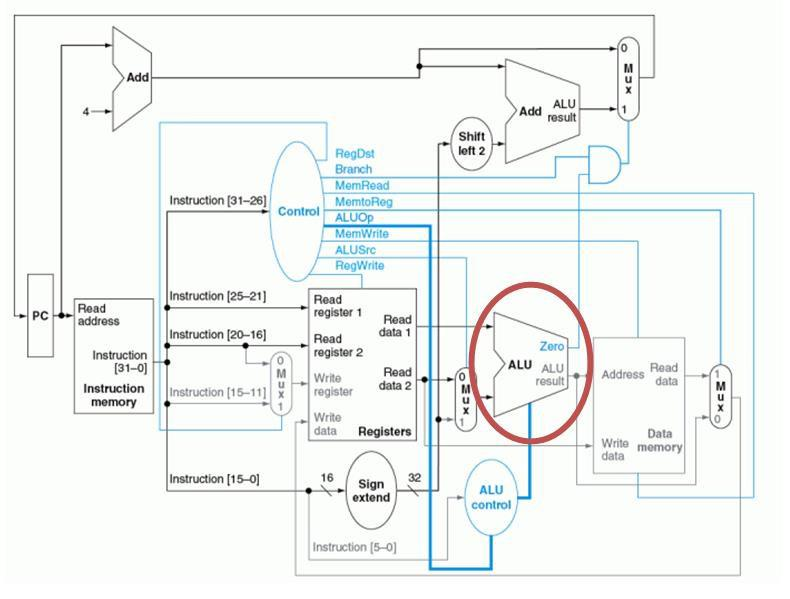
## 2.2.5实验结论

分别测试了LW、R-Type、SW、Beq的测试输入，所得仿真波形与真值表相一致，说明仿真能够根据不同的ALUOp和Funct信号，正确输出ALUCtr信号。

# 2.3ALU模块

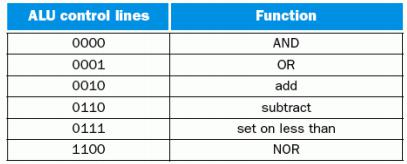
## 2.3.1ALU模块描述

算术逻辑单元 ALU根据ALUCtr信号将两个输入执行对应的操作，ALURes 为输出结果。若做减法操作，当ALURes结果为0时，则Zero输出置为1。



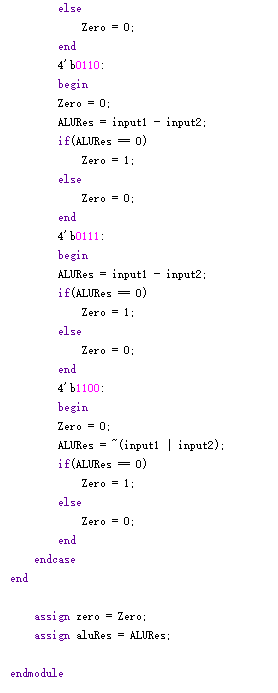
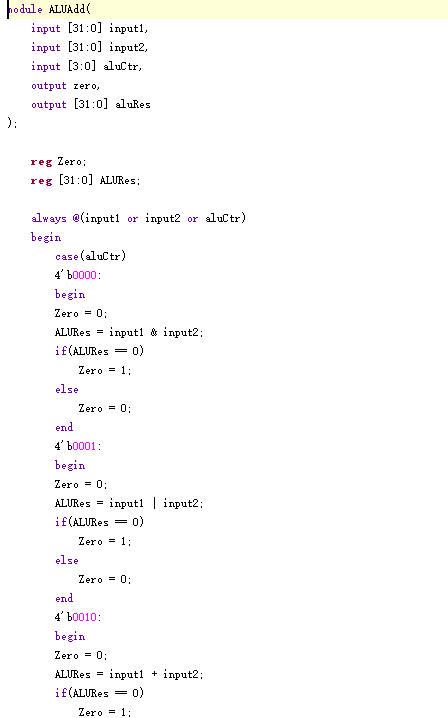
ALU模块示意图

输入ALUCtr与ALU的操作有如下对应关系

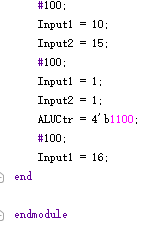
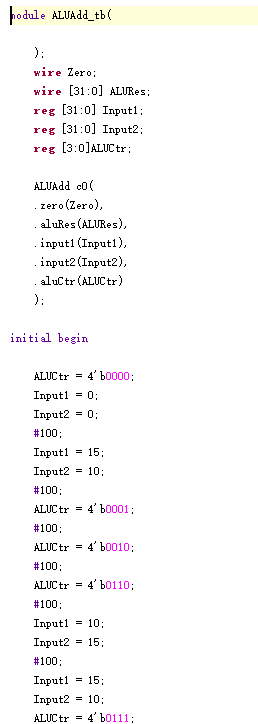


## 2.3.2ALU模块代码

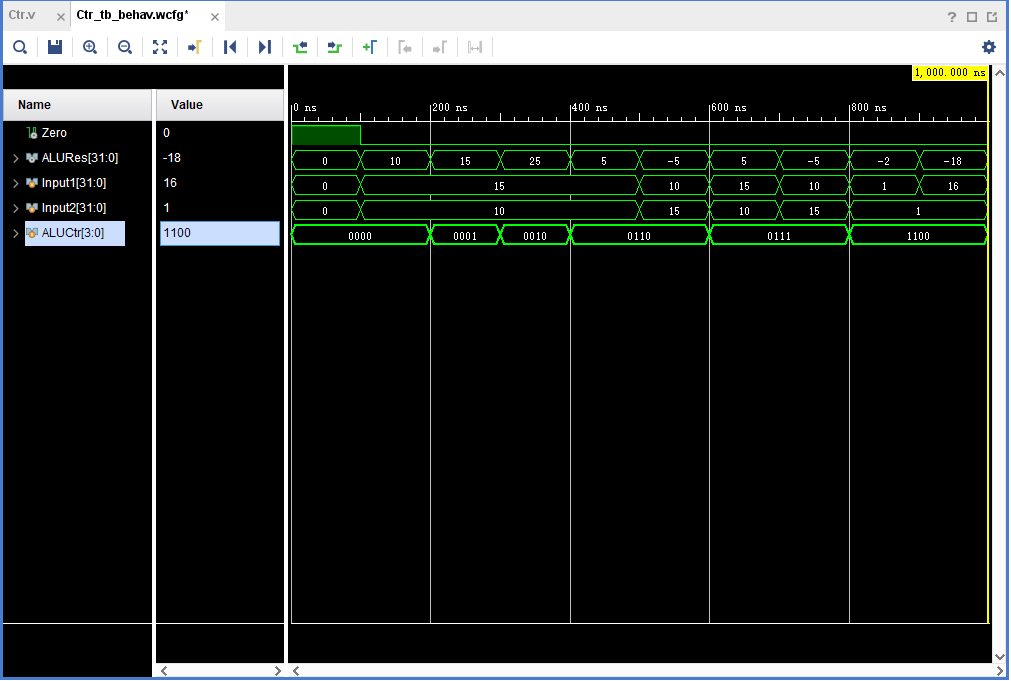
由于情况多种，所以依然使用case语句对不同的输入对应输出，特别注意当计算结果为0时，zero的值应为1



## 2.3.3ALU仿真激励代码



## 2.3.4仿真波形



## 2.3.5实验结论

分别测试了AND、OR、add、substract、slt与NOR对应的ALUCtr与两个输入，波形的输出结果与真值表一一对应，所以仿真能根据ALUCtr正确地对两个输入数进行正确运算并将结果以zero和ALURes输出。

3.实验心得与总结

实验3并没有上板测试过程仅有模拟仿真，而且只需要根据输入分类输出即可，难度并不大。但是由于在此之前并没有接触过Verilog硬件描述语言，lab01与lab02是保姆式教学，我并没有真正了解如何用Verilog编写一个源文件与激励文件，但在这次实验中，我参考之前实验的代码，通过与同学讨论，网上搜索资料学会了基本的语法与各类注意事项。在最初看到实验指导书的示例代码时，我甚至认为module部分与下面reg的标识符大小写不同笑掉大牙了，结果发现是自己的大意，在激励文件中初始化时也有将数组设置为了一个数，导致输出波形有问题的情况也有发生。但不论如何，我顺利完成了这次实验，且在这次实验中不仅学到了Verilog的基本语法，也了解到了模块文件与激励文件编写的注意事项，算是收获良多，对之后的实验必然能起到很大的帮助作用。

当然，我们可以看到，这3个模块ALU、ALUCtr、Ctr当前只支持相应的9条指令，如果想要完成16条甚至31条MIPS指令，那么就必须考虑增加jr指令与有无符号扩展的控制信号，各类情况的case判断都需相应增加，同时，ALUOp需要从2位扩展至3bit才能控制各类不同的指令。

4.参考资料

2020计算机系统结构实验指导书-LAB03\_M