**计算机系统结构试验**

**Lab03: 简单的类MIPS单周期处理器功能部件的设计与实现（一）**

姓名：N/A

摘要

在Lab03中，我进行了MIPS处理器中Control、ALU Control和ALU的设计与仿真，包括如何解码指令并生成相应的控制信号、如何根据指令类型生成适当的ALU操作码、如何实现MIPS处理器中的算术逻辑单元。通过本次实验，我进一步加深了对Verilog语言的理解和运用，掌握MIPS处理器的关键组成部分的设计和实现方法，并且能够使用仿真工具进行验证和调试，给我带来宝贵的经验和收获。

目录

[摘要 1](#_Toc166093931)

[**1.** 实验目的 2](#_Toc166093932)

[**2.** 原理分析 2](#_Toc166093933)

[2.1 Vivado工程的基本组成 2](#_Toc166093934)

[2.2 Crt模块的原理 2](#_Toc166093935)

[2.3 ALUCrt模块的原理 2](#_Toc166093936)

[2.4 ALU模块的原理 3](#_Toc166093937)

[**3.** 功能实现 3](#_Toc166093938)

[3.1 Crt模块的实现 3](#_Toc166093939)

[3.2 ALUCrt模块的实现 4](#_Toc166093940)

[3.3 ALU模块的实现 4](#_Toc166093941)

[**4.** 结果验证 4](#_Toc166093942)

[4.1 Crt模块的测试 4](#_Toc166093943)

[4.2 ALUCrt模块的测试 5](#_Toc166093944)

[4.3 ALU模块的测试 6](#_Toc166093945)

[**5.** 总结与反思 6](#_Toc166093946)

**1.** 实验目的

（1）理解主控制部件或单元、ALU控制器单元、ALU单元的原理；

（2）熟悉所需的MIPS指令集；

（3）使用Verilog HD设计与实现主控制器部件(Ctr)；

（4）使用Verilog设计与实现ALU控制器部件(ALUCtr)；

（5）ALU功能部件的实现；

（6）使用Vivado进行功能模块的行为仿真。

**2.** 原理分析

2.1 Vivado工程的基本组成

（1）Crt.v文件

（2）ALUCtr.v文件

（3）ALU.v文件

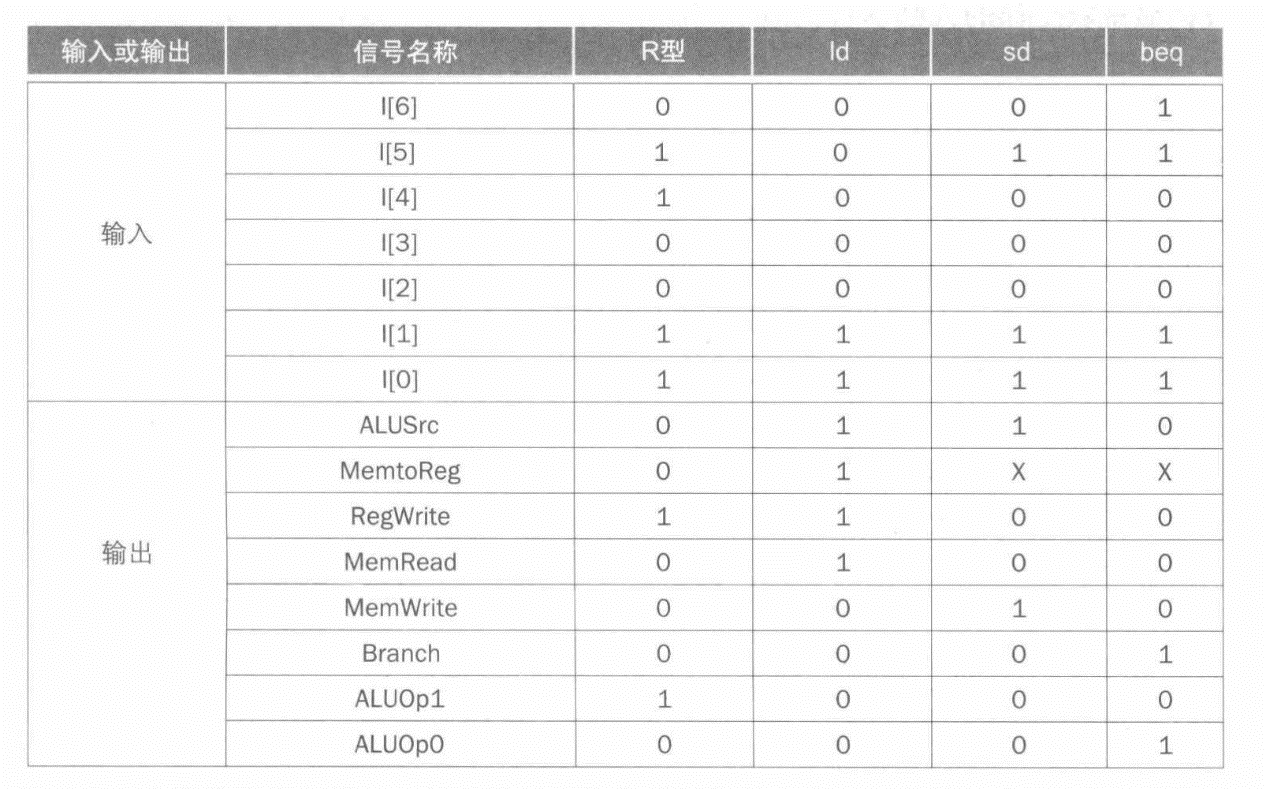
（4）Ctr\_tb.v激励文件

（5）ALUCtr\_tb.v激励文件

（6）ALU\_tb.v激励文件

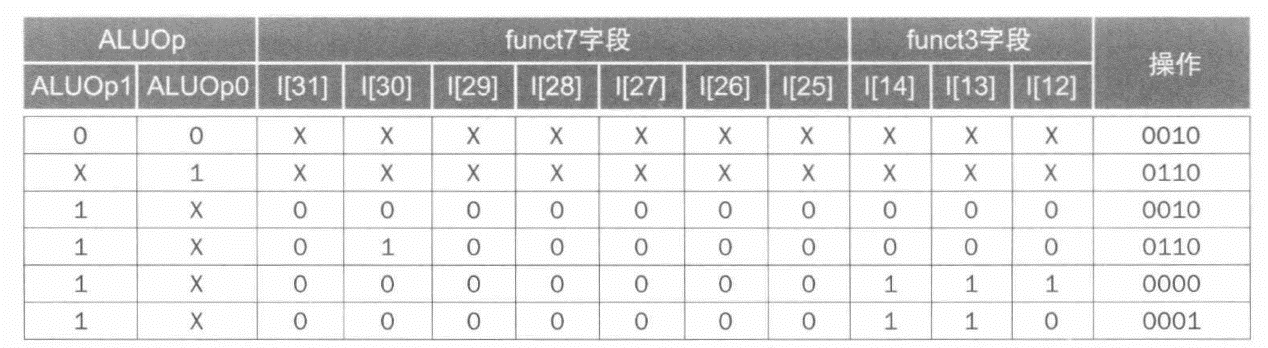
2.2 Crt模块的原理

主控制单元（Ctr）的输入为指令的 opCode 字段，操作码经过Ctr模块的译码，输出ALUOp，RegDst，ALUSrc，MemToReg，RegWrite，MemRead，MemWrite，Branch，Jump等功能单元输出正确的控制信号。译码逻辑如下：



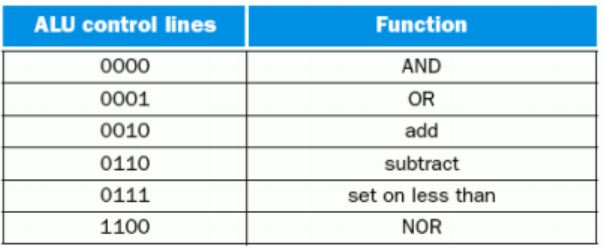
2.3 ALUCrt模块的原理

算数逻辑单元ALU的控制单元（ALUCtr）根据主控制器的ALUOp控制信号来判断指令类型，并依据指令的后6位区分R型指令。综合这两种输入，以控制ALU做正确操作。译码逻辑如下：



2.4 ALU模块的原理

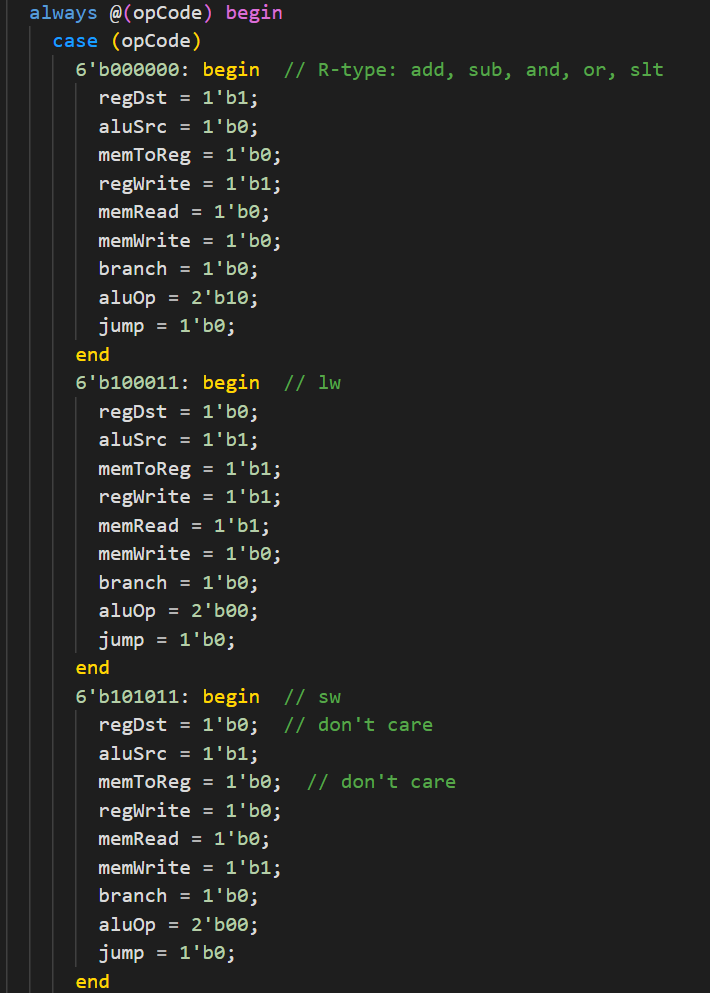
算术逻辑单元ALU根据ALUCtr的控制信号将两个输入执行与之对应的操作。ALURes为输出结果。若减法操作ALURes的结果为 0 时，则 Zero 输出置为1。计算逻辑如下：



**3.** 功能实现

3.1 Crt模块的实现

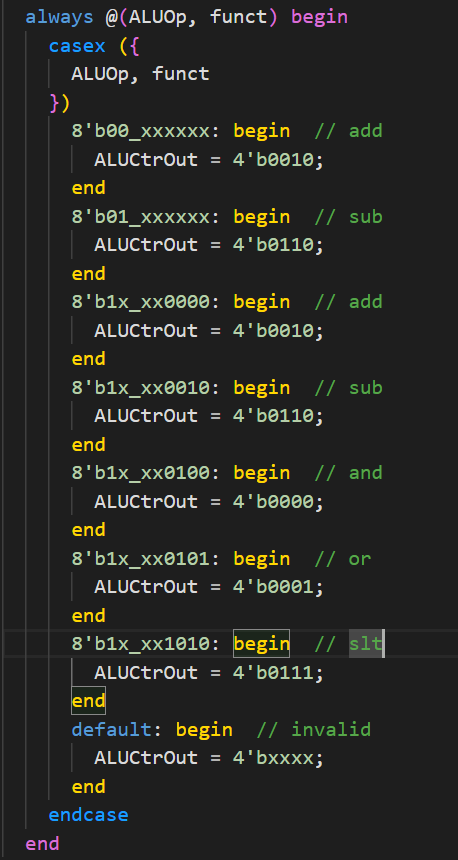
为实现译码逻辑，使用always块来响应opCode的变化，根据opCode的不同值，使用case语句生成相应的控制信号。每个case语句对应一个操作码，包括R-type、lw、sw、beq、jump和默认操作。根据操作码的不同，分别给输出信号赋予相应的值。代码如下：

文本

描述已自动生成

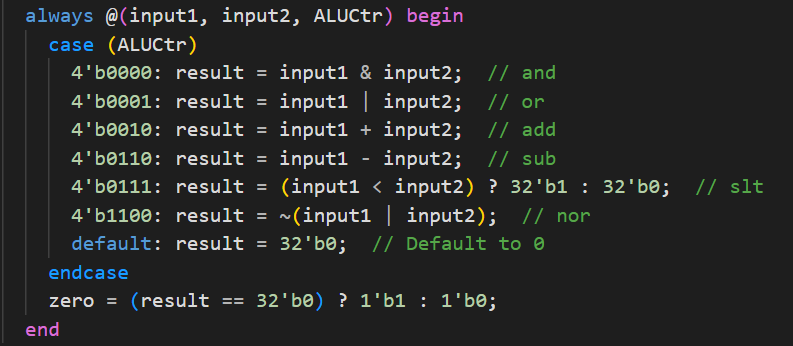
3.2 ALUCrt模块的实现

ALUCtr同样使用always块来响应ALUOp和funct的变化，根据它们的不同值，使用casex语句生成相应的ALU控制信号。casex语句根据ALUOp和funct的组合值进行匹配，每个casex语句对应一种组合情况，包括add、sub、and、or、slt操作和默认（invalid）情况。根据组合值的不同，分别给ALUCtrOut赋予相应的4位值。代码如下：



3.3 ALU模块的实现

算数逻辑单元ALU使用always块来响应input1、input2和ALUCtr的变化，根据ALUCtr的不同值，使用case语句执行相应的算术逻辑运算并给出结果。case语句根据ALUCtr的值进行匹配，每个case语句对应一种ALU操作，包括and、or、add、sub、slt、nor和默认为0情况。根据ALUCtr的不同，分别执行相应的算术逻辑运算，并将结果赋给result。在case语句之后，根据计算得到的结果判断是否为零，并将结果赋给zero。如果result等于32位的0，则zero为1；否则，zero为0。代码如下：



**4.** 结果验证

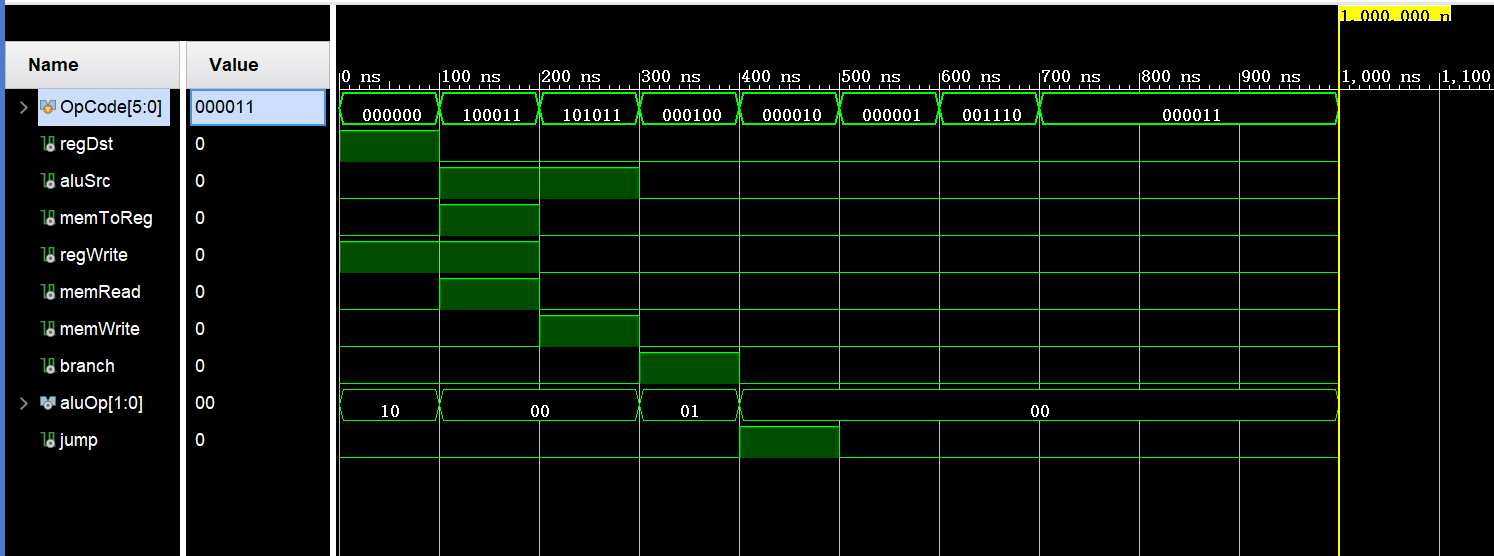
4.1 Crt模块的测试

编写激励文件设置各输入初值，代码如下：

文本

描述已自动生成

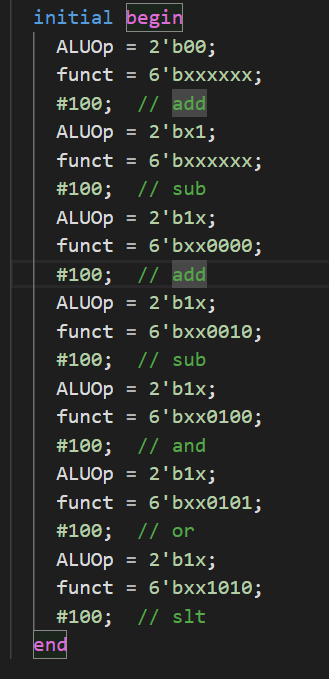
测试结果如图所示：



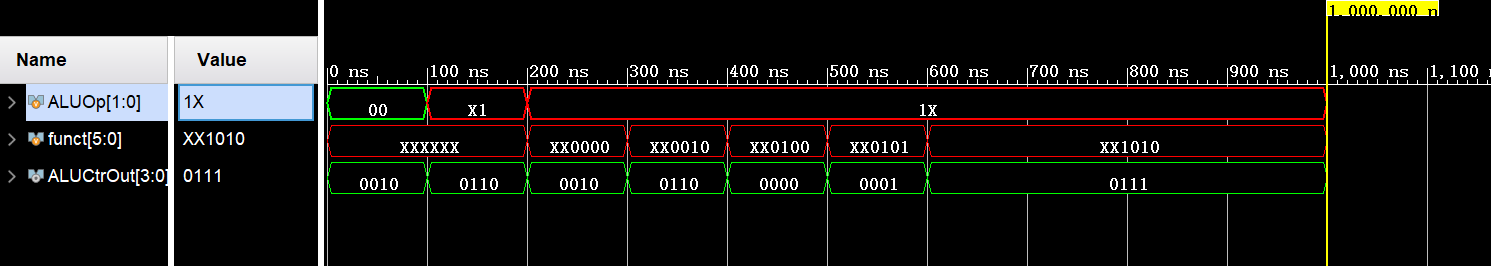
经比较，Crt模块正确地执行了译码功能，在非法输入时输出默认值全0。

4.2 ALUCrt模块的测试

编写激励文件设置各输入初值，代码如下：



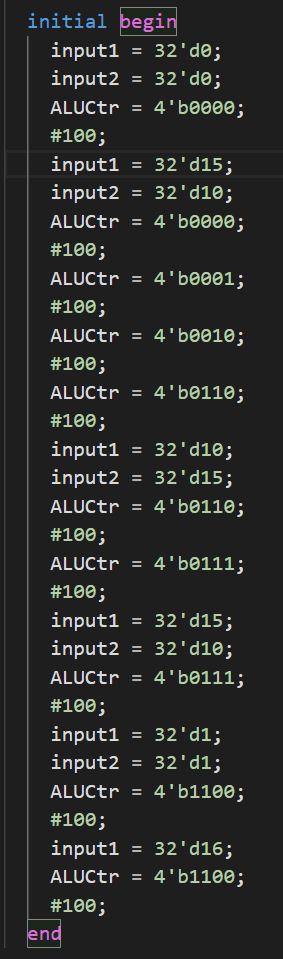
测试结果如图所示：



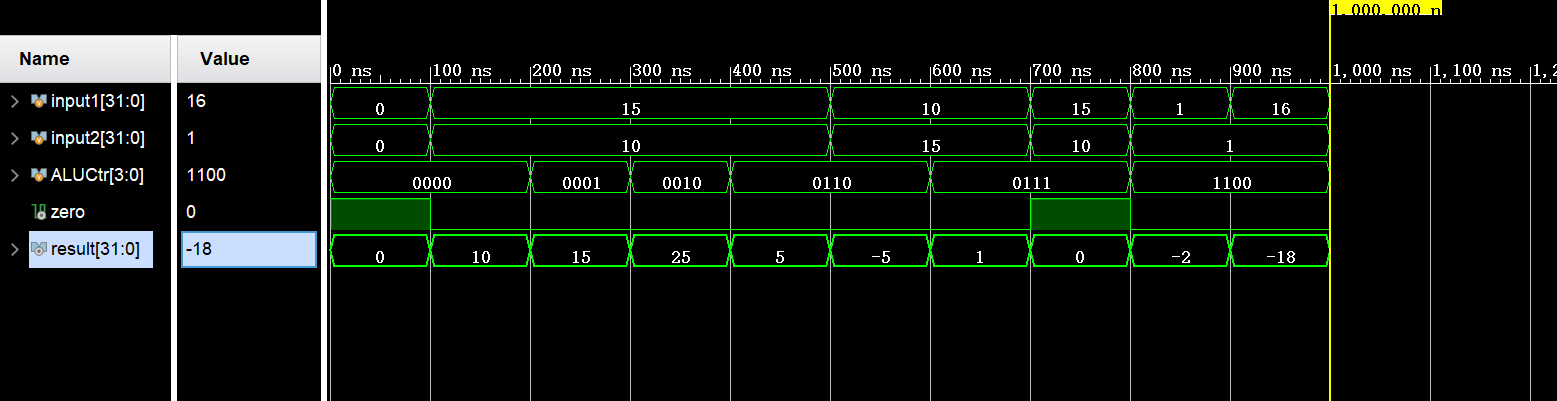
可见输出符合预期。

4.3 ALU模块的测试

编写激励文件设置各输入初值，代码如下：



测试结果如图所示：



输出与正确计算结果一致。

**5.** 总结与反思

在Lab04中，我通过使用Vivado开发环境，进一步熟悉了Verilog HDL的基本语法和编程技巧。通过这次实验，我掌握了使用case块模块化编写分支逻辑的方法，并学会了使用initial begin-end块编写时序激励文件。

我要感谢课程组为我们提供的详细指导书，它为我提供了清晰的实验步骤和示例代码，使我能够更好地理解和实践所学的知识。通过这次实验，我不仅巩固了Verilog的基础知识，还为的学习和设计打下了坚实的基础。