Lab3 简单的类MIPS单周期处理器功能部件的设计与实现（一）

实验目的

- 理解主控制部件或单元、ALU 控制器单元、ALU 单元的原理
- 熟悉所需的 Mips 指令集
- 使用 Verilog HD 设计与实现主控制器部件（Ctr）
- 使用 Verilog 设计与实现 ALU 控制器部件（ALUCtr）
- ALU 功能部件的实现
- 使用 Vivado 进行功能模块的行为仿真

实验原理

主控制单元模块

- 主控制单元（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，Mux 等功能单元输出正确的控制信号。

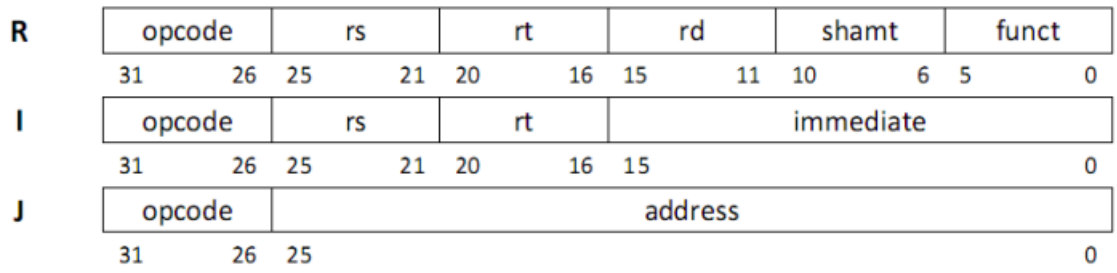


图 1. Mips 基本指令格式

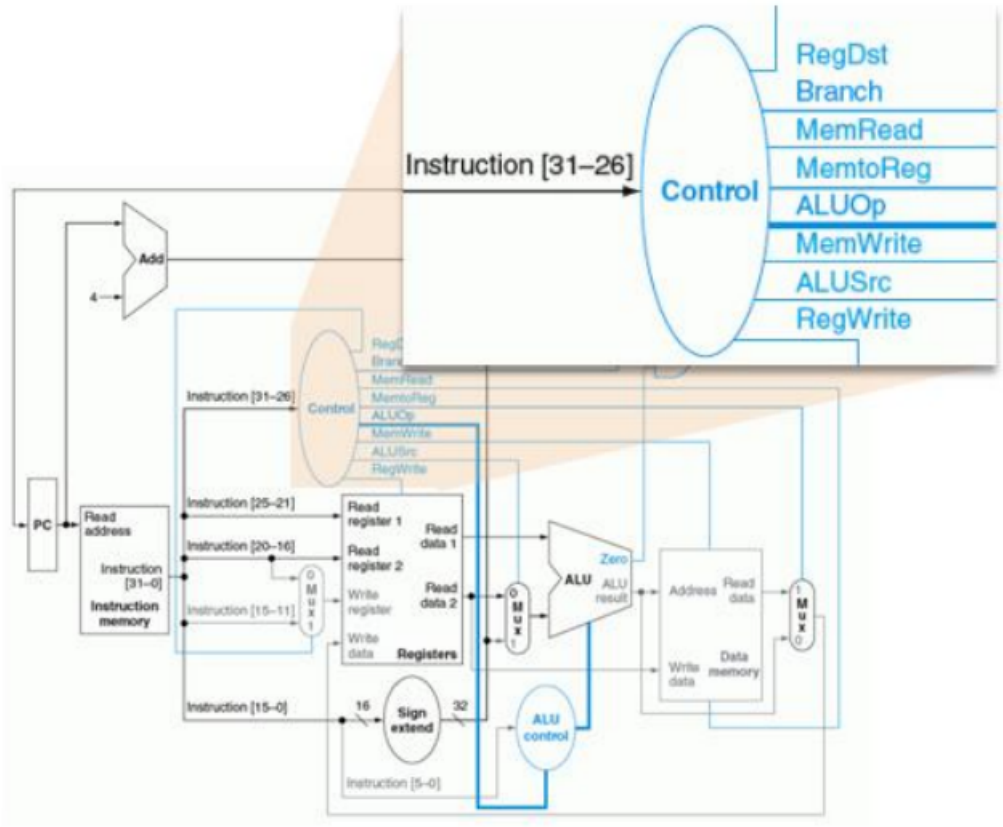


图 2. 简单的 Mips 处理器主控制器单元模块的 IO 定义

- 本次实验用到的控制信号有：
  - RegDst：目标寄存器的选择信号。（0：写入rt代表的寄存器，1：写入rd代表的寄存器）
  - ALUSrc：ALU的第二个操作数的来源。（0：使用rt寄存器中的数，1：使用立即数）
  - MemToReg：写寄存器的数据来源。（0：使用ALU的结果，1：使用从内存读取的数据）
  - RegWrite：写寄存器使能信号。（高电平表示当前指令需要写寄存器）
  - MemRead：读内存使能信号。（高电平表示当前指令需要读内存，比如load）
  - MemWrite：写内存使能信号。（高电平表示当前指令需要写内存，比如store）
  - Branch：条件跳转信号。（高电平表示当前指令是条件跳转指令，比如branch）
  - ALUOp：发送给ALU控制器，用来进一步解析运算类型的控制信号。
  - Jump：无条件跳转信号。（高电平表示当前指令是无条件跳转指令，比如jump）
- 下面是一些指令对应的控制信号：
  - jump 指令编码是 000010，Jump 信号输出 1，其余输出 0。

Input or output	Signal name	R-format	lw	sw	beq
Inputs	Op5	0	1	1	0
	Op4	0	0	0	0
	Op3	0	0	1	0
	Op2	0	0	0	1
	Op1	0	1	1	0
	Op0	0	1	1	0
Outputs	RegDst	1	0	X	X
	ALUSrc	0	1	1	0
	MemtoReg	0	1	X	X
	RegWrite	1	1	0	0
	MemRead	0	1	0	0
	MemWrite	0	0	1	0
	Branch	0	0	0	1
	ALUOp1	1	0	0	0
	ALUOp0	0	0	0	1

图 3. 主控制模块的真值表

- 本次实验译码的指令的操作码如下：

指令	opCode
R 型：add, sub, and, or, slt	000000
I 型：lw	100011
I 型：sw	101011
I 型：beq	000100
J 型：J	000010

图 4. 指令操作码

ALU控制单元模块

- 算数逻辑单元 ALU 的控制单元（ALUCtr）是根据主控制器的 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。

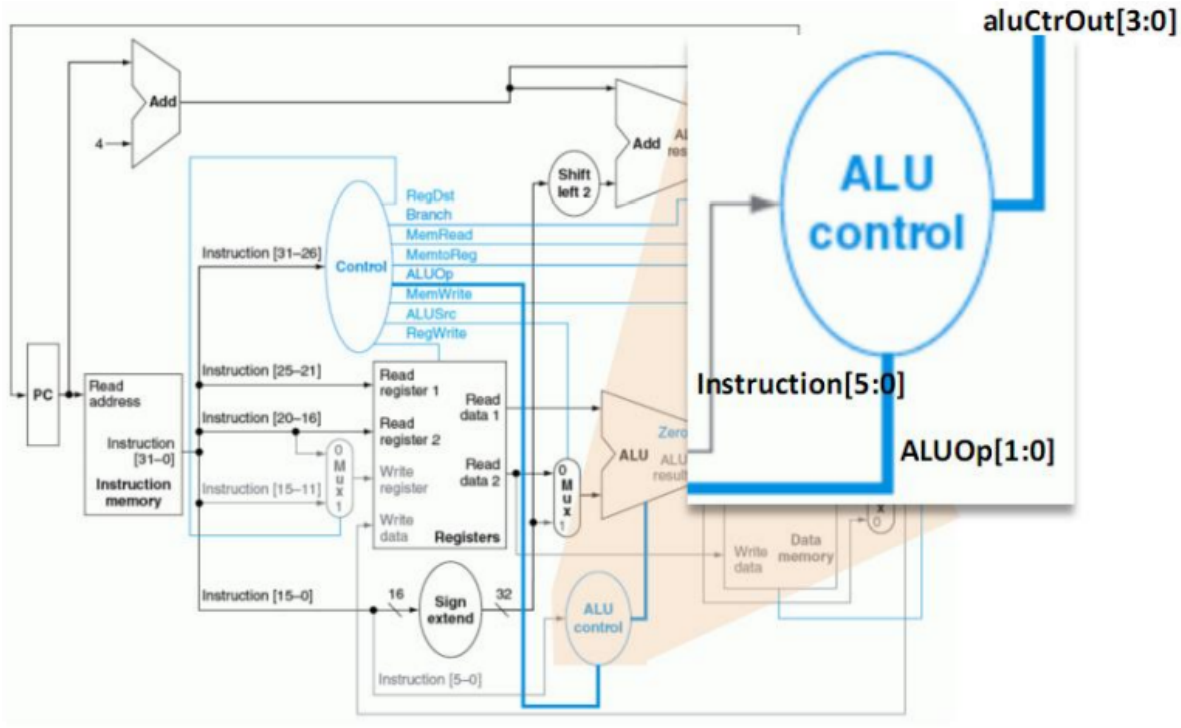


图 5　ALU 控制器模块的 IO 定义

- ALU Control的输出与输入的对应关系如下：

ALUOp		Funct field						Operation
ALUOp1	ALUOp0	F5	F4	F3	F2	F1	F0	
0	0	X	X	X	X	X	X	0010
X	1	X	X	X	X	X	X	0110
1	X	X	X	0	0	0	0	0010
1	X	X	X	0	0	1	0	0110
1	X	X	X	0	1	0	0	0000
1	X	X	X	0	1	0	1	0001
1	X	X	X	1	0	1	0	0111

图 6　ALU 控制单元输入输出真值表

ALU模块

- 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。

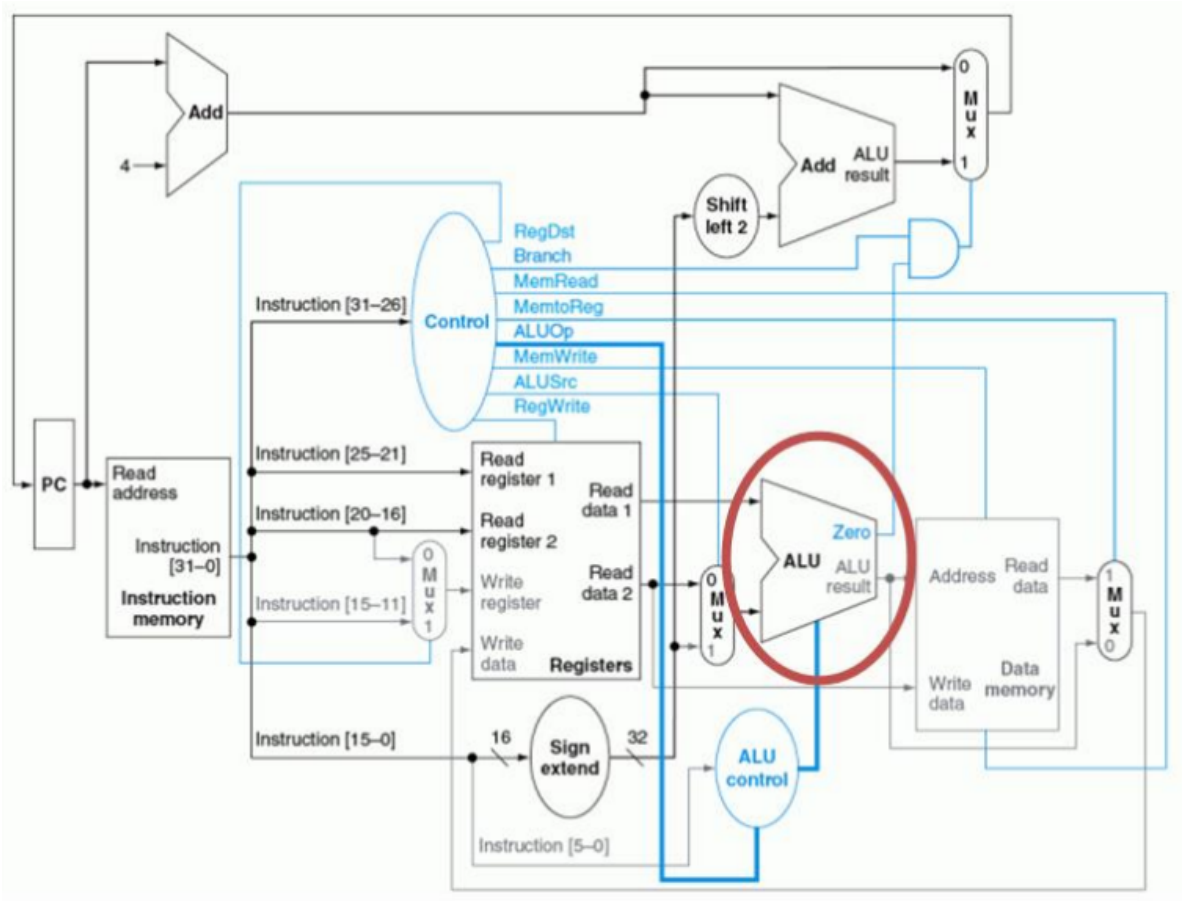


图7　ALU 模块的 IO 定义

- aluCtrOut[3:0]的值与 ALU 操作的对应关系如下：

ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR

图8　aluCtrOut 和 alu 操作的对应关系

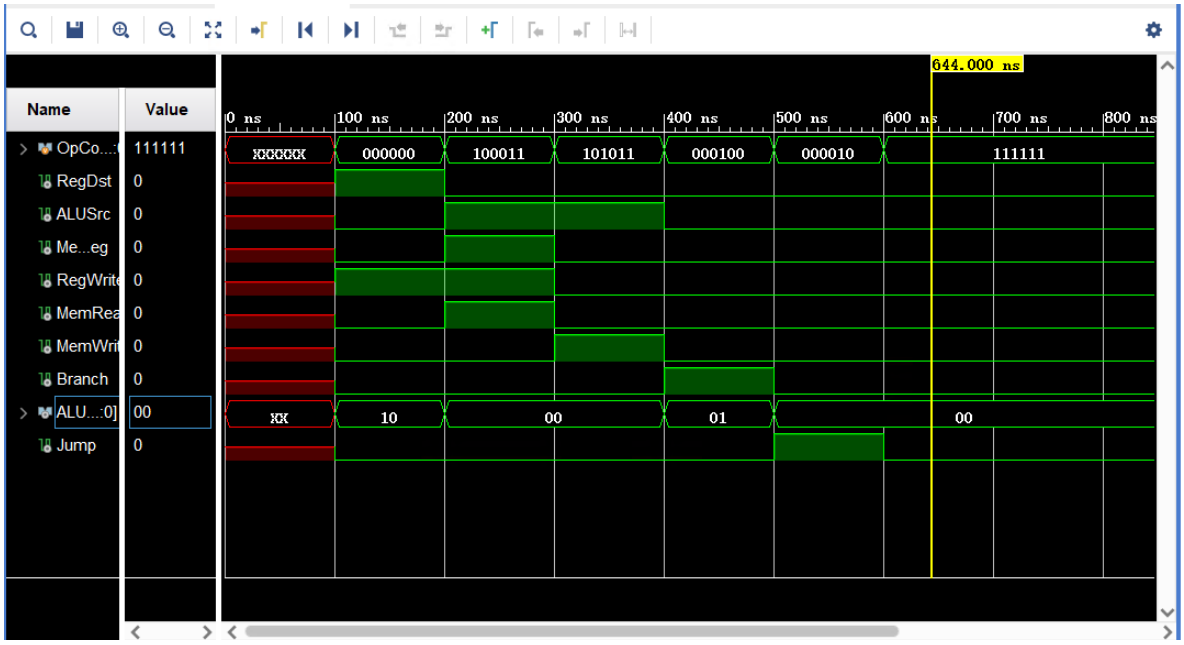
实现细节

- Ctr.v
  - 使用case语句，根据不同的操作码，对各个信号进行赋值。
- Ctr_tb.v
  - 用不同的操作码作为输入，测试各种情况。
- ALUCtr.v
  - 使用casex语句，根据不同的ALUOp信号和Funct，输出对应的运算控制信号。
- ALUCtr.v
  - 用不同的ALUOp信号和Funct作为输入，测试各种情况。
- ALU.v
  - 使用case语句，根据不同的运算控制信号，进行对应的运算操作。
- ALU_tb.v

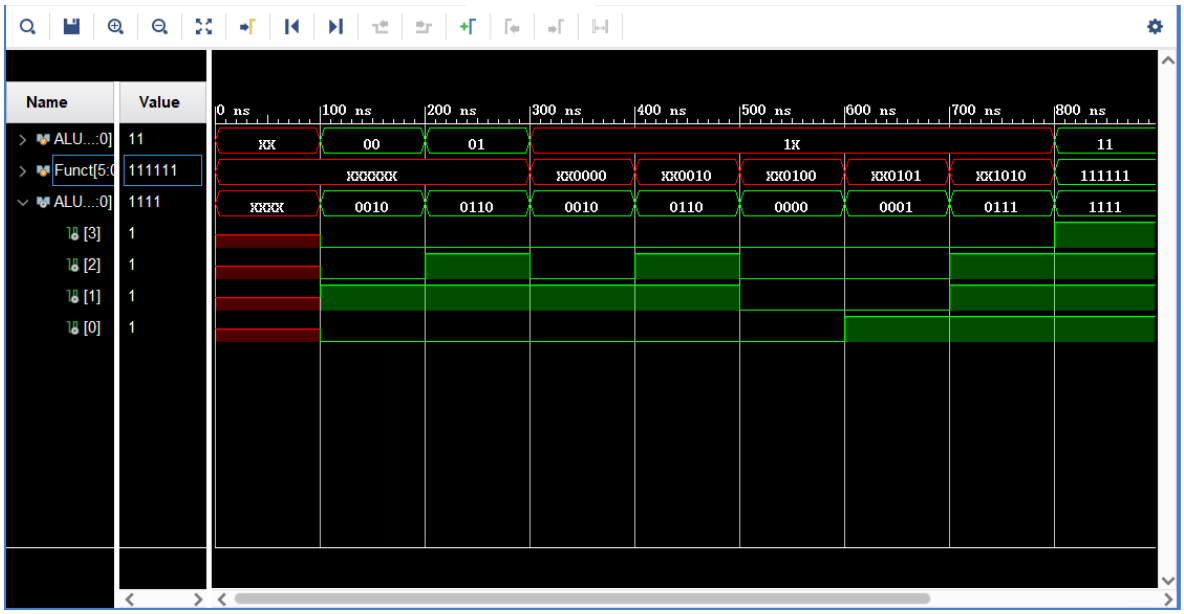
- 用不同的运算控制信号作为输入，测试各种情况。
- 在设计两个操作数输入时，可以通过一些技巧在一次输入测试多种情况，比如测试AND运算时，可将两个操作数设为001和011，这样就测试到了AND的真值表的所有情况。

实验结果

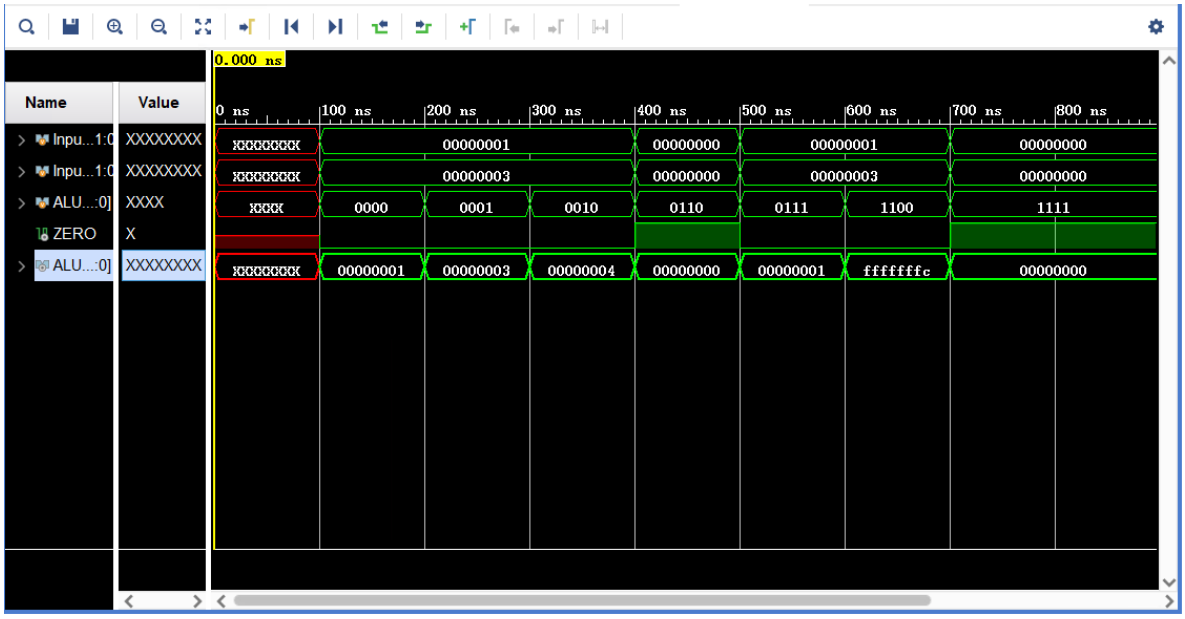
- 主控制单元模块



- ALU控制单元模块



- ALU模块



总结与反思

- 复习了MIPS单周期处理器中一些部件的工作原理。
- 自主进行Verilog程序的设计和仿真。
- 自主学习了如何根据各个条件编写激励程序，以测试每一种可能的情况。
- 自主解决了一些调试过程中遇到的问题：
  - 在调试ALU控制单元模块时，对于x1xxxxxxx -> 0110 的这种情况，没有办法与后面的ALUOp为 1x 的情况区分开。
  - 原因：casex 将 x 视为不必关心的情况。所谓不必关心的情况，即在表达式进行比较时，不将该位的状态考虑在内。所以只要条件表达式和分支表达式的第二位有1个是x，就会陷入这种情况，进行错误的译码。
  - 解决方案：将该情况改为01xxxxxx -> 0110。