

# 计算机组成实验四

## 简单的类MIPS单周期处理器实现—寄存器与内存

### 5140219355

### 朱耀明

[计算机组成实验四 1](#_Toc451097447)

[简单的类MIPS单周期处理器实现—寄存器与内存 1](#_Toc451097448)

[5140219355 1](#_Toc451097449)

[朱耀明 1](#_Toc451097450)

[1.总述 3](#_Toc451097451)

[1.1 实验名称 3](#_Toc451097452)

[1.2 实验目的 3](#_Toc451097453)

[1.3 实验范围 3](#_Toc451097454)

[2.寄存器、内存实现 4](#_Toc451097455)

[2.1 工程文件配置 4](#_Toc451097456)

[2.2寄存器模块实现 4](#_Toc451097457)

[2.3 内存单元模块 5](#_Toc451097458)

[2.4 符号扩展 6](#_Toc451097459)

[3.仿真 8](#_Toc451097460)

[3.1寄存器模块仿真 8](#_Toc451097461)

[3.2内存单元仿真 9](#_Toc451097462)

[3.3符号扩展仿真 9](#_Toc451097463)

[4.小结 11](#_Toc451097464)

[4.1心得 11](#_Toc451097465)

[4.2实验细节 11](#_Toc451097466)

# 1.总述

## 1.1 实验名称

简单的类MIPS单周期处理器实验—寄存器与内存

## 1.2 实验目的

1. 理解CPU的寄存器与内存

2. 使用verilog语言设计存储器件

3. 使用ISim进行行为仿真

## 1.3 实验范围

本次实验将覆盖以下范围

1. ISE的使用

2. Register的实现

3. Data memory的实现

4. 有符号扩展的实现

# 2.寄存器、内存实现

## 2.1 工程文件配置

与之前的实验类似，Device Properties中各属性填写如下

Product Category: ALL

Family: Spartan3E

Device: XC3S500E

Package: FG320

Speed: -4

Synthesis Tool: XST(VHDL/Verilog)

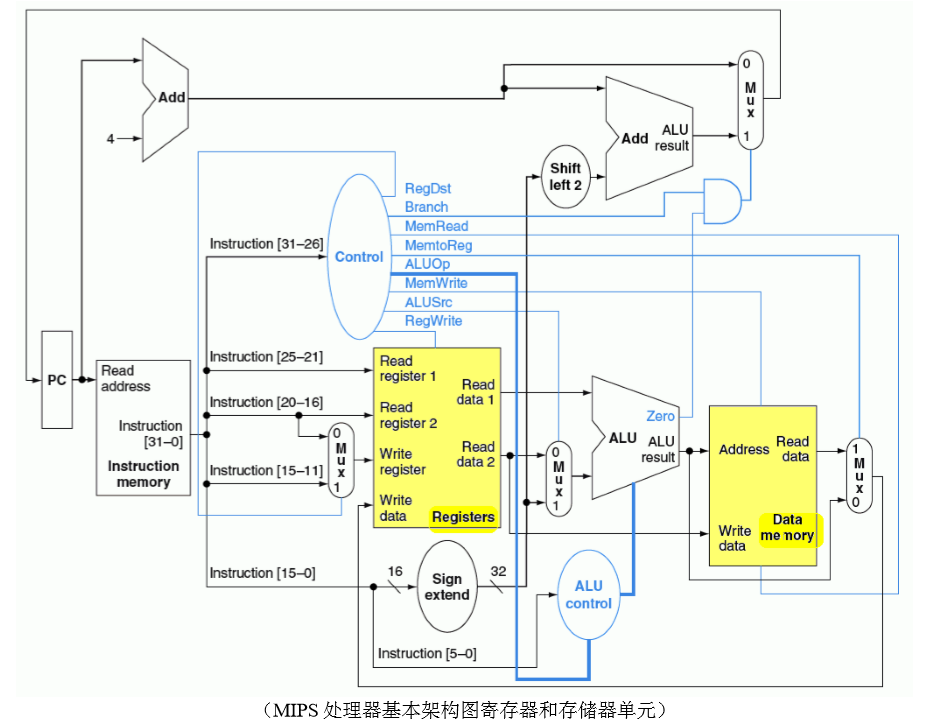
Simulator: ISim(VHDL/Verilog)

Preferred Language: Verilog

## 2.2寄存器模块实现

2.2.1模块描述

本实验需实现的寄存器在MIPS指令计算机中体系结构如Fig.1所示：



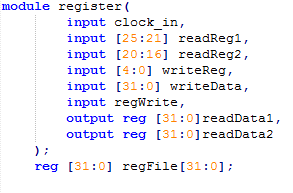
Figure

在MIPS处理器的结构中，通用寄存器为32位，共32个，故需5位的地址线实现访问。实验需要实现的操作为读、写。写操作是在时钟下降沿实现的，需要地址输入信号、时钟信号、写使能信号和写入数据的信号；读操作是随时随地都能进行的，由组合逻辑实现，只需要地址输入信号。

在MIPS指令中，R型指令中包含了Rt,Rd,Rs三个寄存器，因此需要三个地址输入信号。输出信号一共两个。

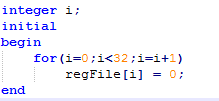
2.2.2 输入输出模块定义

寄存器单元模块的输入输出与寄存器定义如Fig.2



Figure

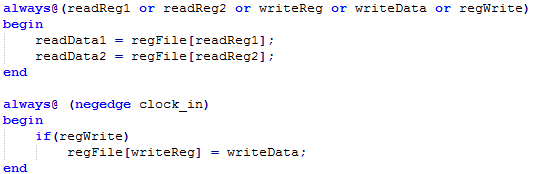
其中clock\_in为系统时钟，regWrite为写使能信号，readReg1,readReg2,writeReg为地址输入信号，writeData为写入数据的信号，readData1,readData2为数据输出。regFile为寄存器组。使用for语句对regFile进行初始化置零，如Fig.3



Figure

2.2.3编写功能代码

如前所述，实验使用时钟下降沿作为写操作的同步信号。寄存器实现见Fig.4



Figure

其中读操作因为是任意时进行的，always模块中的敏感参数实际上是所有输入。也可以使用always @(\*)。写操作需同时满足写使能和时钟下降沿两个条件，这里将时钟下降沿作为了敏感参数。

## 2.3 内存单元模块

2.3.1模块描述

内存单元模块在MIPS处理器中的大致结构见Fig.1，在实验中我们实现的MIPS处理器是哈佛架构的，此处实现的内存单元模块严格意义上说只是数据内存单元。指令内存单元会在后续的实验中实现。

实验需要实现的内存操作与寄存器操作类似，为读、写操作。MIPS指令集作为典型的RISC指令集，对内存的访问只能通过lw, sw两条指令完成。其中写操作非常相似，需要写使能信号、时钟信号和地址输入信号，写操作也是在时钟下降沿完成的。与寄存器操作不同的是，内存读操作也需要使能信号。

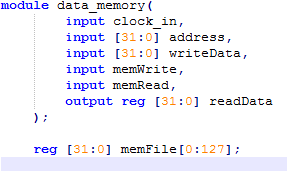
目前常见的内存是以字节(8bit)作为内存等的单位。考虑到本实验中实现的MIPS指令、数据都是32bit的，为方便起见，实验实现的内存编制为32bit(4B)。后续的实验沿用了这一设定。这样本实验的处理器仅支持字对齐的内存访问。

MIPS的地址总线为32位，可访问4GB内存，实验模拟的内存仅为128\*4B。

综上所述，数据存储器需要1个32位地址输入，1个32位的数据输入，1个写使能信号输入，1个读使能信号输入以及一个32位的数据输出。

2.3.2输入输出定义

见Fig.5

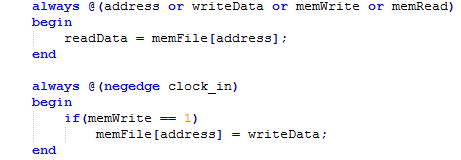


Figure

初始化与寄存器类似，使用for语句，此处不再赘述。

2.3.3编写功能代码

如Fig.6



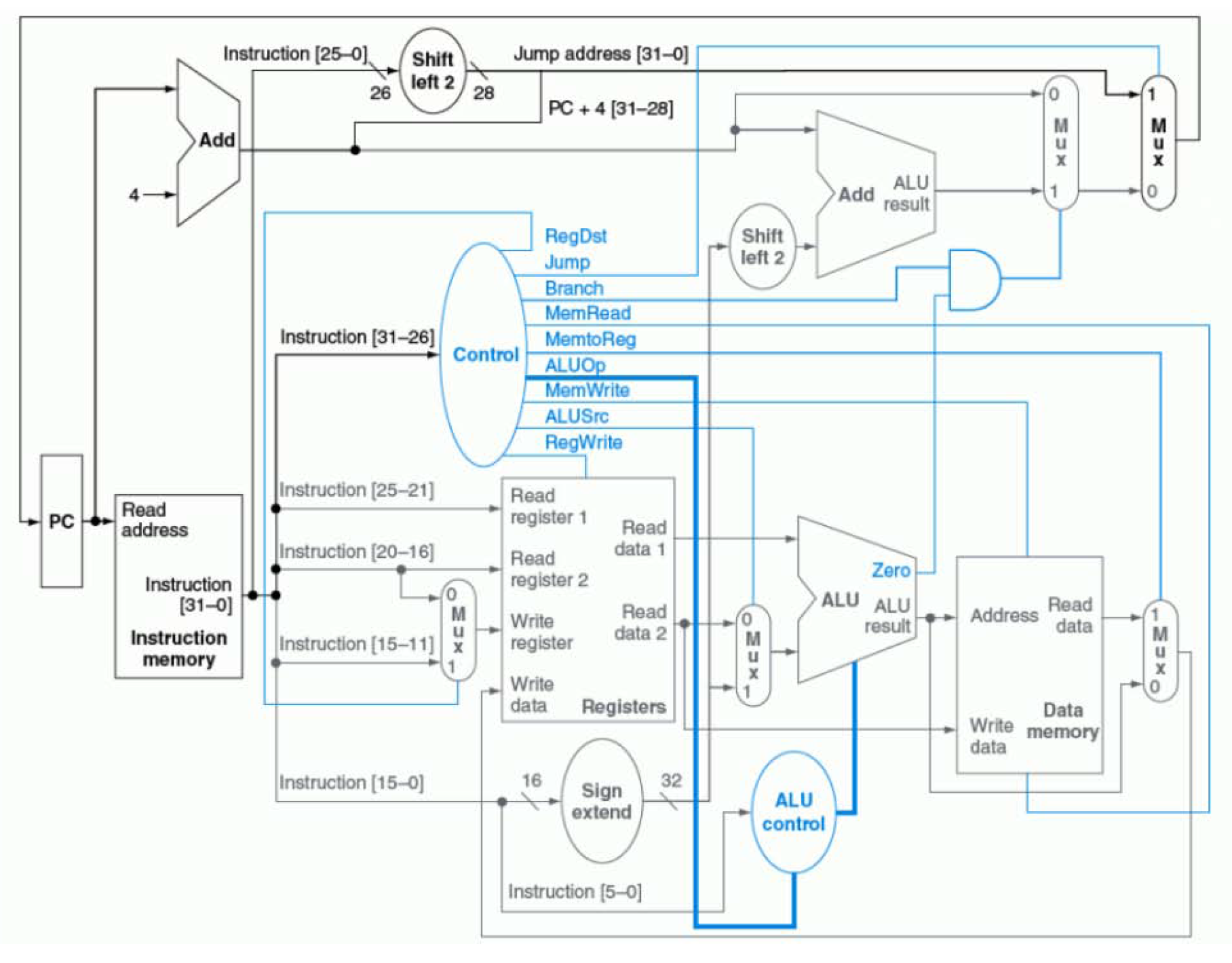
Figure

## 2.4 符号扩展

2.4.1模块描述

该模块时将I型指令中的16位立即数进行拓展。如Fig.7

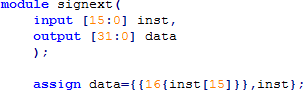
正数和零前补16位0，负数前补16位1.正负数由输入的MSB判定。



Figure

2.4.2模块功能实现

代码见Fig.8



Figure

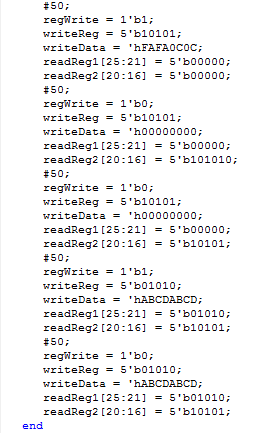
至此完成了寄存器、内存和符号拓展的设计。之后进行仿真。

# 3.仿真

## 3.1寄存器模块仿真

仿真模块的建立等等都和lab3中相同。

文件中添加激励见Fig.9



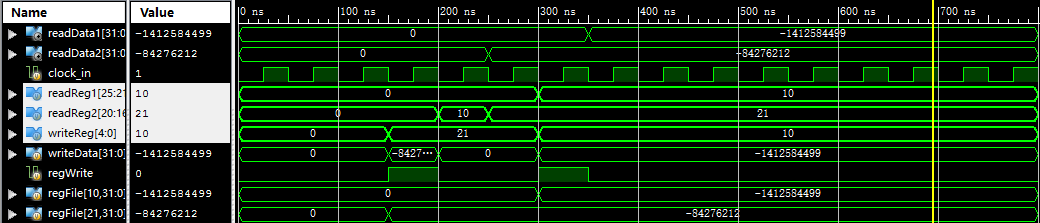
Figure

对于clock\_in信号，作如下处理，使其作为周期为50的固定输入。见Fig.10



Figure

仿真波形见Fig.11

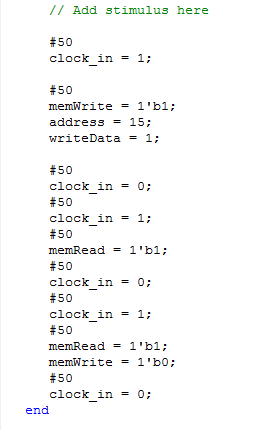


Figure

可见设计满足了实验的所有要求。

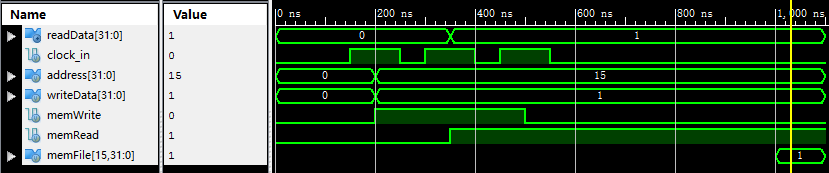
## 3.2内存单元仿真

与之前的仿真类似，激励见Fig.12



Figure

仿真波形见Fig.13

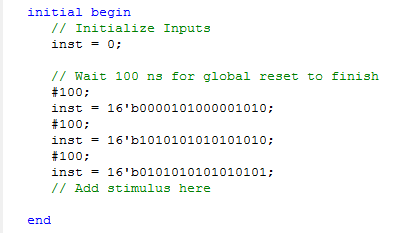


Figure

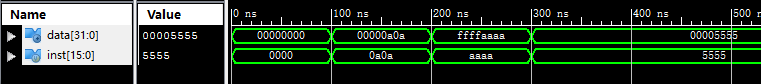
注：内存只显示了1000ns后部分，原因会在后文的实验细节处解释。

## 3.3符号扩展仿真

激励见Fig.14，仿真波形见Fig.15



Figure



Figure

符号扩展的波形是十六进制显示的。

由此可见，lab4的实验要求都已完成。

# 4.小结

## 4.1心得

lab4的难度不算大，是后面几个实验中完成得最快的一个了。不过做完全部实验后回看lab4，发现实验指导书上的要求还没有满足实际的处理器设计要求，譬如没有reset功能等等。对于寄存器和内存单元的进一步改进会在之后的实验报告中详细叙述。

另外借阅的书籍也对代码优化带来了很大的帮助。例如在符号拓展单元，我就按照书上的实例使用了*{{16{inst[15]}},inst}*这种形式的扩展，相比使用if语句判定MSB再进行扩展节省了不少时间。

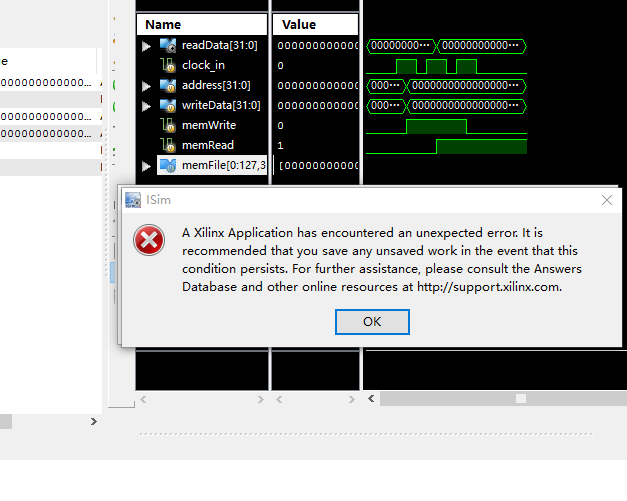
## 4.2实验细节

在波形仿真环节，默认生成的波形仅显示了输入输出信号的波形，对于regFile和memFile则需要自己从左侧的模块中调出。此时需要重新进行仿真，需要对下方的控制台输入类似Fig.16的命令。



Figure

但如此操作可能会导致强制退出(见Fig.17).我询问了一起做实验，也遇到强制退出问题的同学，发现仿真环节的强制退出都出现在win10系统，ISE版本为14.7的情况，实验室的电脑则不会出现强制退出。个人猜测可能是因为win10系统安装ISE时为了解决兼容问题，需要修改一些软件文件造成的。



Figure

解决办法是尽量少添加波形。如本次实验中的regFile我只添加了被修改的两个而不是全体寄存器，就不再出现强制退出。但这种方法不一定都生效，memFile的仿真只要添加被修改的内存就会出现强制退出。解决方法是多次模拟，每次run \*ns使用不同的时间来观察。