**组成原理实验课程第 6 次实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验名称 | 单周期CPU | | | 班级 | 李涛老师 |
| 学生姓名 | 申宗尚 | 学号 | 2213924 | 指导老师 | 董前琨 |
| 实验地点 | 实验A306 | | 实验时间 | 2024.6.13 | |

1. **实验目的**

结合实验指导手册中的实验六（单周期CPU实验）完成功能改进，在原有CPU基础上，扩充CPU可运行的MIPS指令，注意以下几点：

1、扩充的指令应为一个时钟周期内能够执行完的指令，要求至少一个R型，一个I型，另外一个自选。建议在ALU实验改进基础上补充。

2、实验报告中原理图为指导手册中的display模块图，不用修改，报告中的内容和展示的结果应扩充指令的步骤和实验结果。

3、本次实验报告需要有实验箱上箱验证的照片，同样，针对照片中的数据需要解释说明。若只有仿真波形结果，会适当扣分。

4、实验报告模板参考百度云盘文件，注意提交截至时间为6月14日下午18：00。

1. **实验内容**

单周期CPU能够执行MIPS指令集系统的一个子集，共16条指令，包括存储访问指令、运算指令、跳转指令。根据拥有的字段类型不同，我们将指令分为R型指令、I型指令和J型指令。

其中，R型：



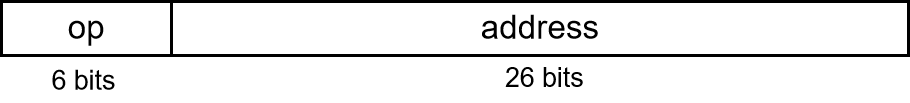
Op恒为000000，rs、rt为两个5位源操作数的寄存器号，rd为5位目的操作数的寄存器号，shamt为5位移位指令的移位数，6位funct决定指令具体功能

I型：



Op6位决定指令具体功能，rs为5位第一个源操作数所在寄存器号，rt位第二个源操作数所在寄存器号或目的操作数所在寄存器号，最后16位为指令中的立即数或地址。

J型：



Op6位决定指令具体功能，26位地址

在实验中我们主要实现R型和I型指令，因此描述其实行过程：

**R 型指令**

1. 从指令存储器中取指令，更新 PC 。

2. ALU 根据 funct 字段确定 ALU 的功能。

3. 从寄存器堆中读出寄存器 rs 和 rt。

4. ALU 根据 2 中确定的功能，对从寄存器堆读出的数据进行操作。

5. 将运算结果写入到 rd 字段对应的目标寄存器。

**I 型指令**

存取指令：

1. 从指令存储器中取指令，更新 PC 。

2. ALU 根据 op 字段确定 ALU 的功能。

3. 从寄存器堆中读出寄存器 rs 的值，并将其与符号扩展后的指令低16位立即数的值相加。

4. 若为存储指令，则将 rt 寄存器中的值存到上步相加得到的存储器地址；

若为取数指令，则将 上步所得存储器地址里所存的数据放到 rt 目标寄存器中。

分支指令：

1. 从指令存储器中取指令，更新 PC 。

2. 从寄存器堆中读出寄存器 rs 和 rt 的值。

3. 将所读寄存器的两值相减。

4. 根据上步的结果是否为0，将 PC+4 的值或 address 字段所对应地址存入PC中。

1. **实验原理图**

**图示, 示意图

描述已自动生成**

1. **实验内容**

**·复现单周期CPU：**

**在这部分，主要按照实验指导手册操作，按照要求进行代码的配置即可，其中单周期 CPU 的 inst\_rom.v 和 data\_ram.v 均为自行搭建的异步存储器。在其基础上加上 regfile.v 和 alu.v。其中 alu.v 还调用了实验一中的 adder.v。**

**这部分复现的单周期 CPU 能够执行 MIPS 指令集系统的一个子集，共 16 条指令，包括存储访问指令、运算指令、跳转指令。选取的指令集将会用到各个数据通路部件，并体现控制部分设计的关键原理。**

**图形用户界面, 文本, 应用程序

描述已自动生成**

**表格

描述已自动生成**

**·添加一条R型指令，添加一条I型指令：**

由于之前复现过几条指令，这里需要添加的指令可以选择之前写过的按位同或操作作为R型指令，选择低位加载作为I型指令。下面介绍具体修改之处。

首先进行single\_cycle\_cpu.v文件的修改。

1. 先将需要添加的两条指令放到指令列表中：

文本

描述已自动生成

1. 然后，为添加的两个操作赋予操作数

文本

描述已自动生成

1. 然后定义并实现传递到执行模块的ALU源操作数和操作码

文本

描述已自动生成

1. 修改立即数拓展指令，添加低位加载指令



1. 修改alu\_control的代码，加入我们需添加的指令

表格

描述已自动生成

1. 最后，修改寄存器的写回值。

文本, 信件

描述已自动生成

然后，修改alu.v文件

1. 拓展ALU控制信号

文本, 信件

描述已自动生成

1. 添加alu\_control码

表格

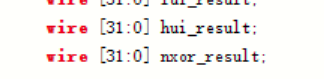
低可信度描述已自动生成

1. 添加控制信号

文本, 表格

中度可信度描述已自动生成

1. 添加结果信号



1. 添加运算逻辑

公司名称

中度可信度描述已自动生成

1. 添加选择结果输出部分

表格

低可信度描述已自动生成

然后，修改inst\_rom.v文件，进行修改，使添加的指令能运行并显示结果。

手机屏幕的截图

中度可信度描述已自动生成

1. **上箱验证**
2. **复现单周期CPU：**

**按照实验手册的inst\_rom设定，运行四条指令时，应该将REG04设置为0x00000004**

**assign inst\_rom[ 3] = 32'h00022082; // 0CH: srl $4 ,$2,#2 | $4 = 0000\_0004H**

**电脑屏幕的照片

描述已自动生成**

**上机验证，REG04 = 0x00000004**

**按照实验手册的设定，运行六条指令时，应该将REG04设置为0xFFFFFFE2**

**assign inst\_rom[ 6] = 32'h00A23027; // 18H: nor $6 ,$5,$2 | $6 = FFFF\_FFE2H**

**电脑屏幕的照片

描述已自动生成**

**REG06 = 0xFFFFFFE2，验证成功**

1. **添加一条R型指令，一条I型指令**

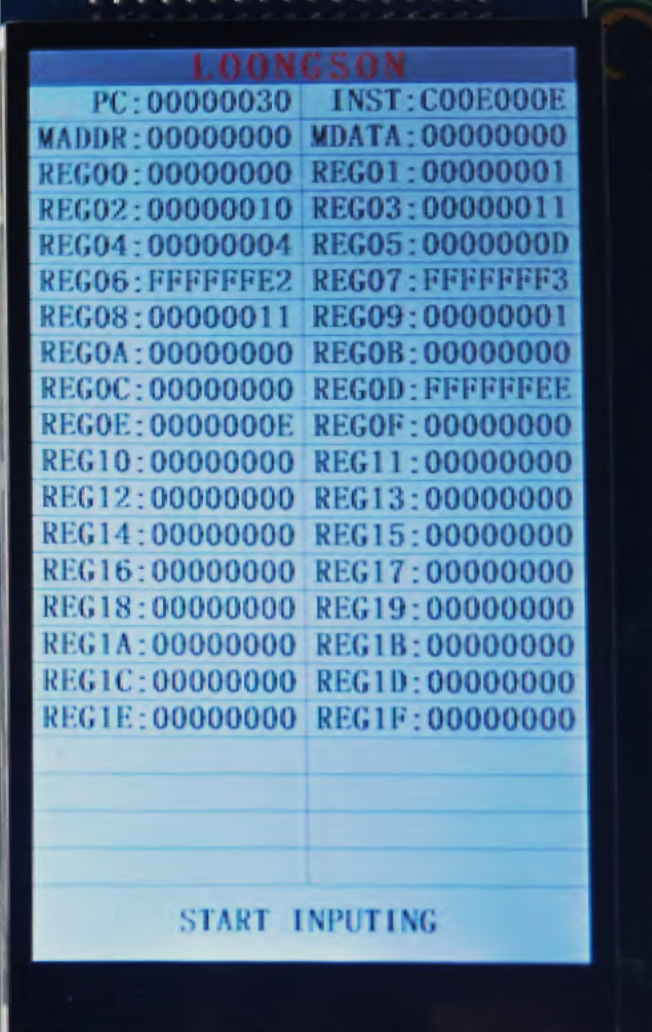
**由于我们加入的设定：**

手机屏幕的截图

中度可信度描述已自动生成

**当第11条指令执行时，对应的应该是**

**Nxor $13, $6, $7**

****

**可以看到，此时REG06 = FFFFFFE2，REG07 = FFFFFFF3**

**REG0D = REG6 nxor REG07 = FFFFFFEE，验证正确**

**类似的，当第十二条指令执行，对应的操作为**

**Hui $14, #14**

**电脑屏幕的照片

描述已自动生成**

**可以看到，此时REG0E的值为0000000E，就是立即数14的低位加载结果，验证成功。**

1. **总结感想**

在本次实验中，通过对单周期CPU原理的学习和实现，系统理解了一个时钟周期完成整个人物的CPU运行方式，同时加入了一条I型指令和一条R型指令，通过对于之前写过的部分的再次调用，对verilog编程的能力有了进一步的加深。

临近期末，本次实验也为我复习计组理论课程中的指令章节提供了一个很好的入口，复习了各种类型指令的组成和执行过程