Lab4 简单的类 MIPS 单周期处理器存储部件的设计与实现（二）

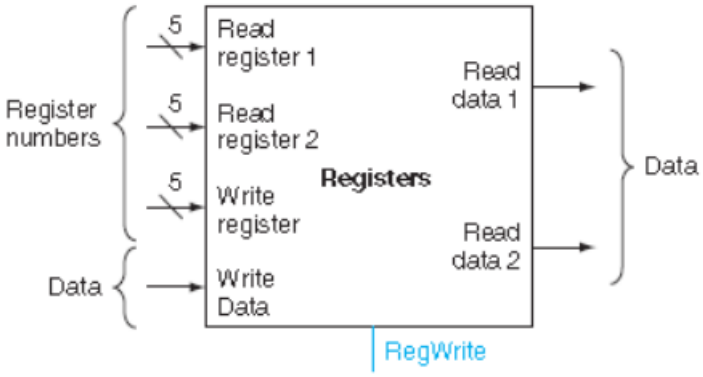
实验目的

- 理解寄存器、数据存储器、有符号扩展单元的 IO 定义
- Registers 的设计实现
- Data Memory 的设计实现
- 有符号扩展部件的实现
- 对功能模块进行仿真

实验原理

寄存器组模块

- 寄存器是指令操作的主要对象，MIPS 中一共有 32 个 32 位的寄存器，用作数据的缓存。
- 因为有32个寄存器，所以需要一个5位的二进制数来进行寄存器的定位。



寄存器模块的 IO 定义

- 可以让寄存器的读操作一直进行，而写操作仅在时钟下降沿进行。这样可以防止在一个周期开始时读取到错误的数据。这个机制在后续解决流水线中的冒险时也有用处。

数据存储器模块

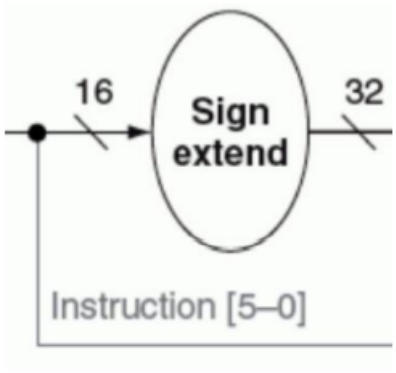
- Data memory 是用来存储运行完成的数据，或者初始化的数据。内存模块的编写与 register 类似，由于写数据也要考虑信号同步，因此也需要时钟。

内存模块的 IO 定义

- 此次实验设计的存储器只存储64个数据，所以理论上只需要6位二进制数即可定位。不过在这里仍然采用了32位地址编码以模拟现代计算机设计，可以理解为虽然有2^32个存储单元，但是只允许使用其中64个。
- 因为存储单元的数目限制，需要在存储器内部加入检查地址是否越界的逻辑，如果输入的地址越界，则让读操作返回0或让写操作无效。
- 可以让存储器的读操作一直进行，而写操作仅在时钟下降沿进行。这样可以防止在一个周期开始时读取到错误的数据。这个机制在后续解决流水线中的冒险时也有用处。

带符号扩展模块

- 将 16 位有符号数扩展为 32 位有符号数。
- 带符号扩展只需要在前面补足符号即可。



带符合扩展单元

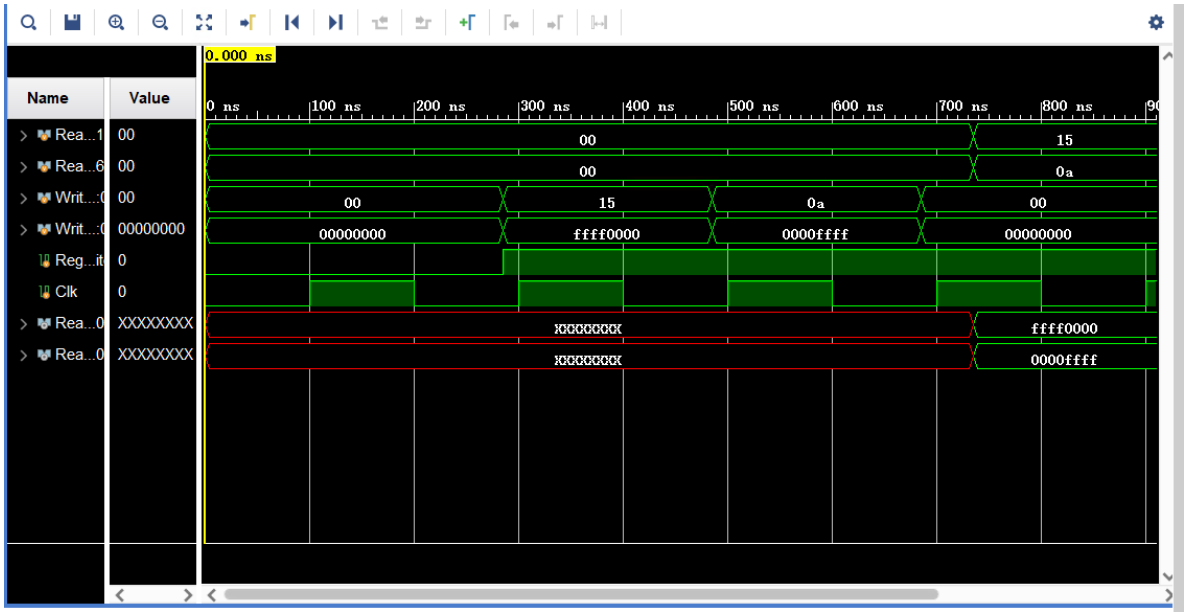
实现细节

- Registers.v
  - 根据输入的读操作地址输出目标寄存器里的数据。
  - 如果写寄存器使能信号为1，则在时钟下降沿，根据输入的要写入的数据和写操作地址向目标寄存器写数据。
- Registers_tb.v
  - 设置时钟输入，时钟周期为200ns。
  - 使用多组不同的输入，对寄存器组进行写、读操作，通过仿真波形检查结果。
- dataMemory.v
  - 如果读寄存器使能信号为1，检查输入的读操作地址，若未越界，则根据地址输出目标地址里的数据，若越界，则将输出数据设为0。
  - 如果写寄存器使能信号为1且写操作地址未越界，则在时钟下降沿，根据输入的要写入的数据和地址向目标地址写数据。
- dataMemory_tb.v

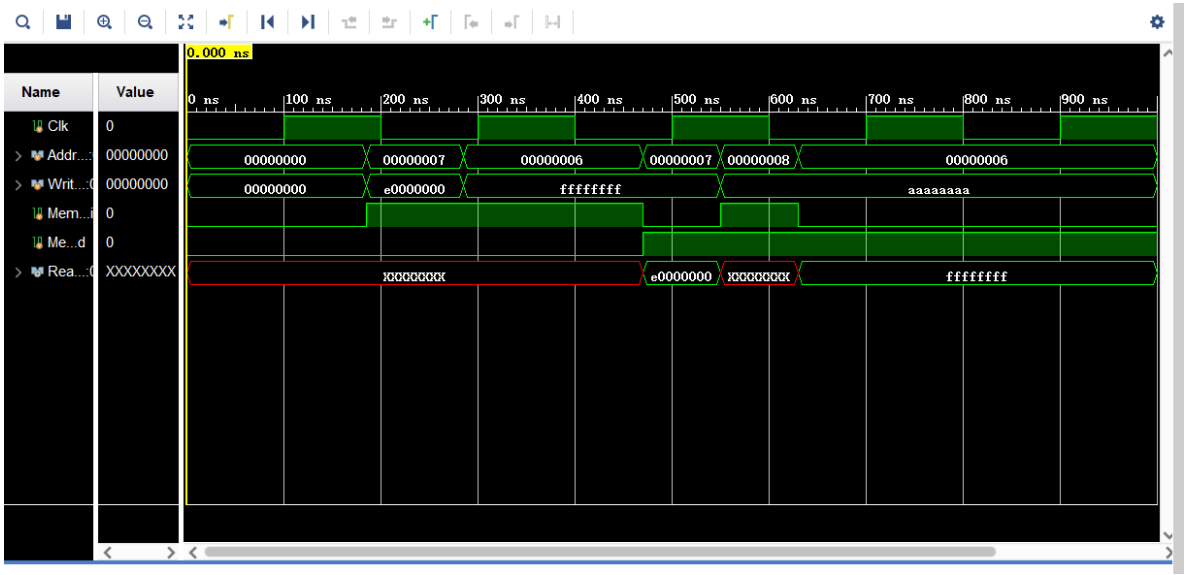
    - 设置时钟输入，时钟周期为200ns。
    - 使用多组不同的输入，对存储器进行写、读操作，通过仿真波形检查结果。
- signext.v
    - 用Verilog的位拼接运算符{}，将输入数据的第一位复制16个，作为扩展的前16位，输入的原始数据作为后16位。
- signext_tb.v
    - 使用不同的输入，通过仿真输出检查结果。

实验结果

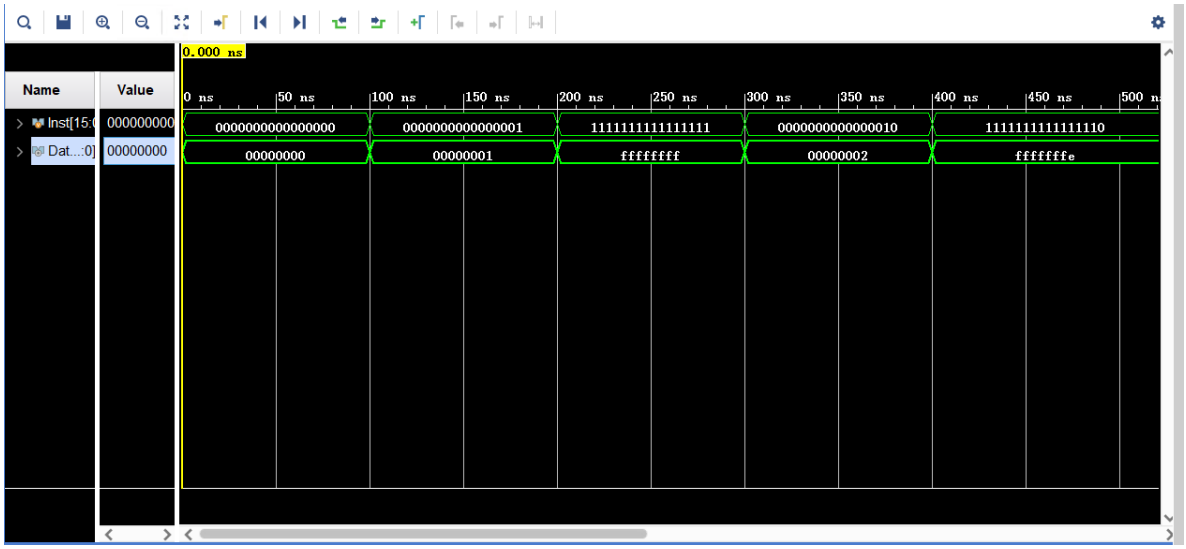
- 寄存器组模块



- 数据存储器模块



- 带符号扩展模块



总结与反思

- 又复习了MIPS单周期处理器中一些部件的工作原理。
- 继续练习自主进行Verilog程序的设计和仿真。
- 再次练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。
- 根据原理，自行对各部件逻辑进行了一些优化。
- 新学到了Verilog的位拼接运算符{}。