简单的类 MIPS 单周期处理器部件实现——寄存器，存储器，符号扩展

目的

设计模块实现 CPU 中寄存器、存储器、符号扩展的逻辑。

设计思路

寄存器、存储器主要由存储数组、读写控制信号、数据输入输出、时钟信号和 reset 信号组成。在相应时钟边沿和读写信号中进行读写操作。在寄存器中，由于即使读出的数据没有用，也不会造成效率问题，故寄存器不需要控制是否读取数据；而实际存储器每次读取会造成下一次读取前的的等待，因此需要控制读取。符号扩展判断输入数据最高位，并将高 16 位全部设置为与输入数据最高位相同。

模块描述

寄存器所有操作在时钟下降沿进行，先读后写，若有重置信号则将所有寄存器内容置 0。代码中，将读/写寄存器编号作为索引，以读出或写入相应的寄存器。

存储器在时钟上升沿读，下降沿可读写，将读写地址作为索引，以读出或写入相应的内存单元。

符号扩展接收输入一个 16 位数据，并输出对应的符号扩展结果。代码中，将输入数据与 16 进制数 8000 进行按位与操作以判断符号，然后通过与 16 进制数 0000ffff 进行按位与或与 16 进制数 ffff0000 进行按位或，将最高位设为 0 或 1。

仿真描述

寄存器仿真进行写入、读出模拟，先向 31 号、10 号寄存器写入值，然后读出。测试是否能正常写入、读出。

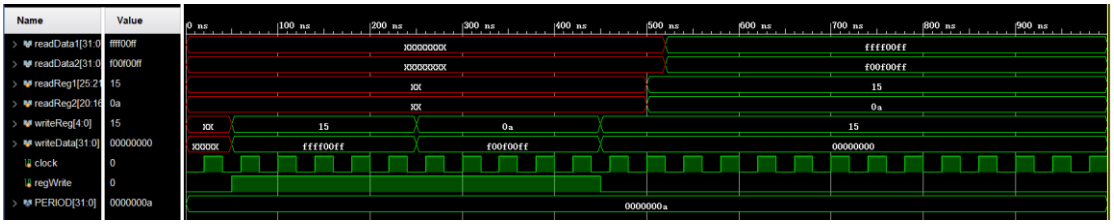
存储器仿真进行写入、读出模拟，先向 1 号、2 号地址写入值，然后读出 1 号地址的值，测试是否能正常写入、读出。

符号扩展仿真考虑 2 种情况，即输入为正、负的情况。本次仿真选取的值为 0000 和 ffff（16 进制）。

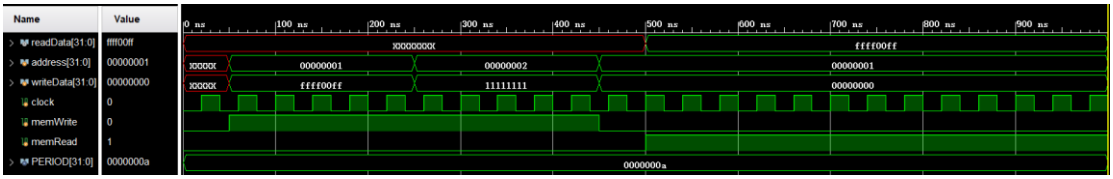
实验结果

仿真波形如下所示（由于软件显示问题，显示的值可能并非完整值）。

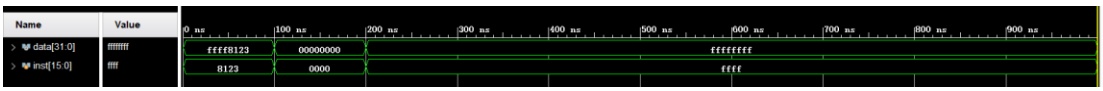
寄存器波形:



存储器波形:



符号扩展波形：



通过分析相应仿真波形，结果正确。

总结

本次实验较为简单。寄存器和内存是 CPU 设计中主要考虑的模块。通过设计、模拟寄存器与存储器的运行，可以更生动地理解这些部件的设计。值得思考的是，现实中的内存并非此实验模拟中的理想内存。由于内存访问速度远远慢于 CPU 的运行，在当今的实际计算机中大多有 Cache；出于内存管理的需要，还会加入 Page Table 及相应的 TLB 等等提高内存效率的办法。因此在存储器设计中，还有许多可以改进的地方。本次实验完成的只是理想化的存储器模型。且在后续实验中，也是以这样的模型为基础的。