实验六 CPU综合设计

**一、实验目的**

1. 综合运用Verilog进行复杂系统设计。
2. 深刻理解计算机系统硬件原理。

**二、实验内容**

1. 设计一个基于MIPS/ARM/RISC-V指令集的CPU。
2. CPU需要包含寄存器组、RAM模块、ALU模块、指令译码模块。
3. 该CPU能运行基本的汇编指令。
4. 实现多核，流水线或其他现代CPU的高级功能（加分项）

**三、实验要求**

1. 分析各模块的的程序结构，画出其流程图。
2. 画出模块的电路图。
3. 分析电路的仿真波形，标出关键的数值。
4. 记录设计和调试过程。

CPU

Instruction Memory

Data

Memory

指令内容

指令内容

存取数据

存取地址

单核单周期TOP模块结构图

Main decoder

Alu decoder

Alu

Register file

PC

指令

下一条指令地址

寄存器组输出

单核单周期CPU模块结构图

Imem

CPU1

CPU2

CPU2下一条指令地址 addr2

CPU1下一条指令地址 addr1

[addr1]

[addr2]

多核CPU地址转换关系

**四、实验代码及结果**

**五、调试和心得体会**

1.beq的条件判断细节问题：

Beq指令也需要用到alu，且alucontrol需要设置为减法，才能正确得出zero的值，

3.lw，sw两操作数的alusrc变量细节问题：

4.alu在进行slt操作时碰到的问题：

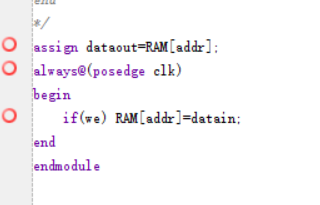
Ain-bin<0

与

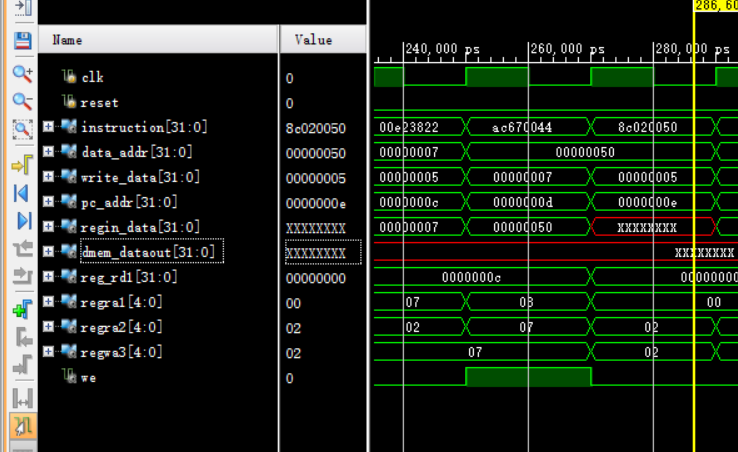
Ain<bin不同。。。。

5.sw往data memory传值时，dmem中传入与读出互斥的问题：

开始时的定义：



Simulation时发现数据并没有成功录入到内存中：



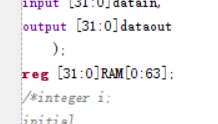
可以看到，ac670044机器指令与8c020050机器指令对应汇编指令为

sw $7,68($3);

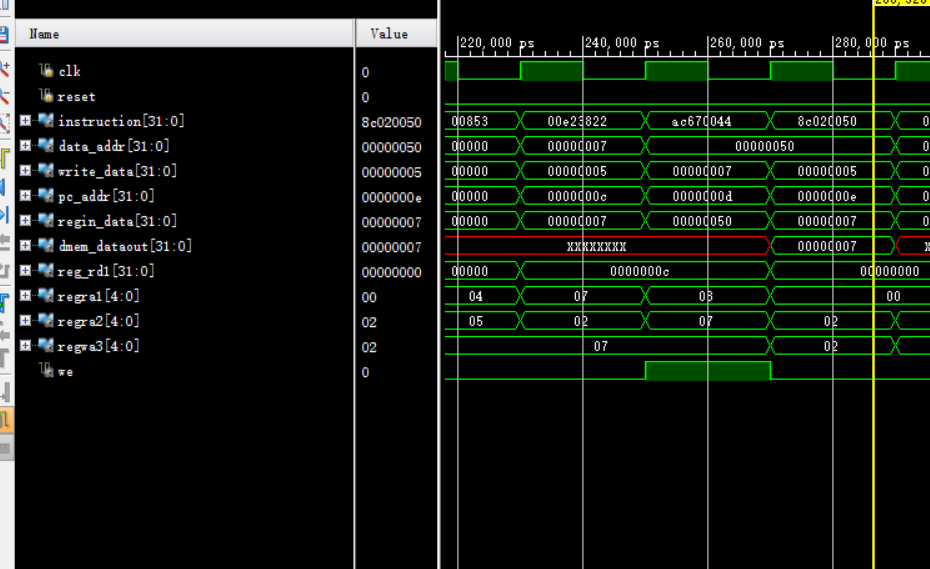
lw $2,80($0);

即先把内存中地址为80内容存储为寄存器组中编号为7的寄存器的内容（上面的计算过程可以得到7号寄存器内容为7，占4个字节），再把该内存中的值读入到2号寄存器中；

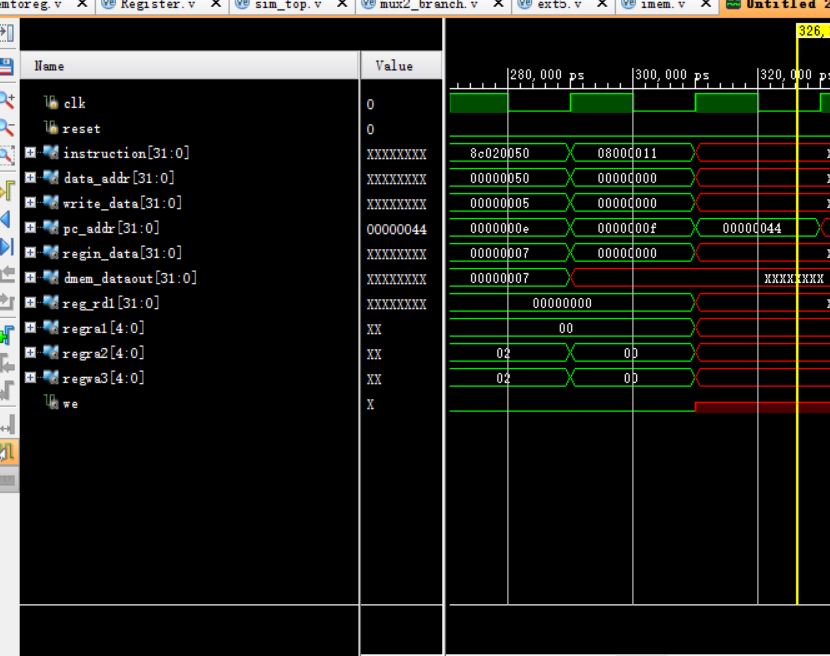
经过检查，发现dmem的内容无法读入竟是dmem的内容容量不足……



重新多设置内存大小后，问题成功解决。



6.jump地址无法正常跳转的问题：

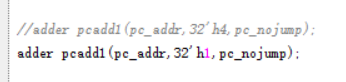


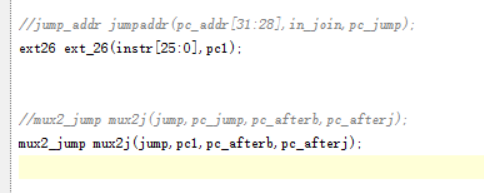
第0000000f语句无法正常跳转到指定位置；

该指令机器指令为08000011，要跳转的位置应该为第17号语句（十进制）；于是更改cpu中的连线，在连线模块有这样一个地址转化：

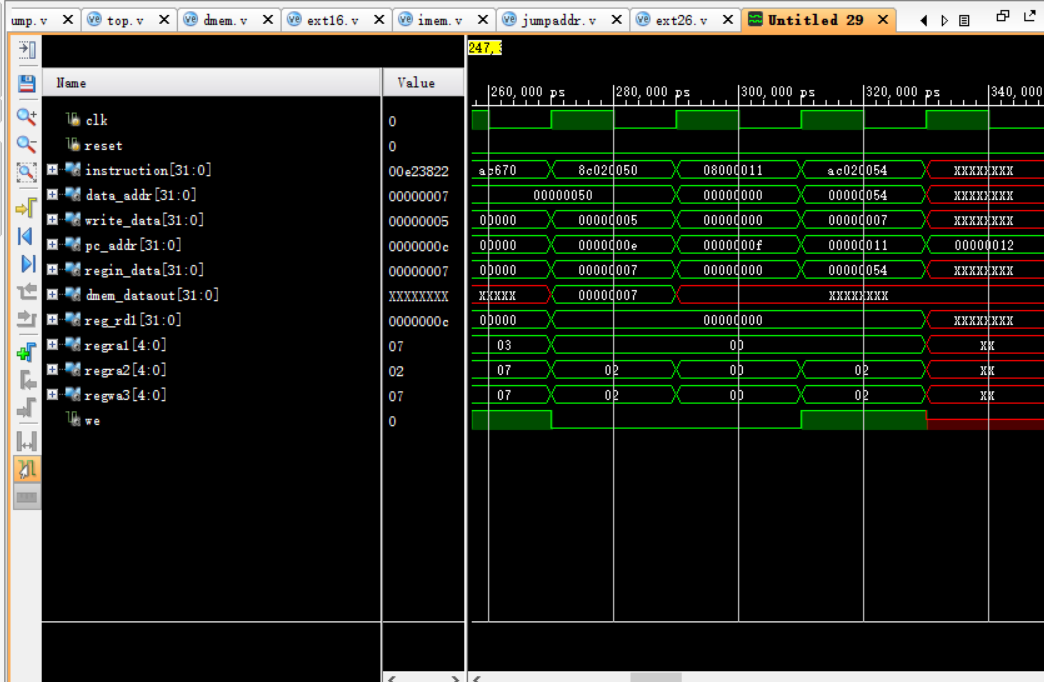
先将指令中的26位立即数左移两位，再把原pc地址的[31:28]的四位接在前4位上。（参考的MIPS 32指令集对jump的定义），但发现之前为了简便考虑，直接把指令的位置对应为imem中的位置。如果按照一个字节为一个单位的话，即每条非转移指令执行后，下一条指令地址为该条指令运行结束后＋4，我的简化版本只需＋1，应该有一个指令顺序与指令寄存器的物理地址转换的过程（其实只需在PC寄存器输出下一条指令右移两位（/4）即可）。

于是，在注释中添加了这两种地址转换方式，使用一种将另一种注释掉即可。





最后结果能够正常输出。



可以看到，程序成功从15号指令跳到17号指令，并成功执行。