实 验 报 告

学号：2014K8009929022 姓名：孔静 专业：计算机科学与技术

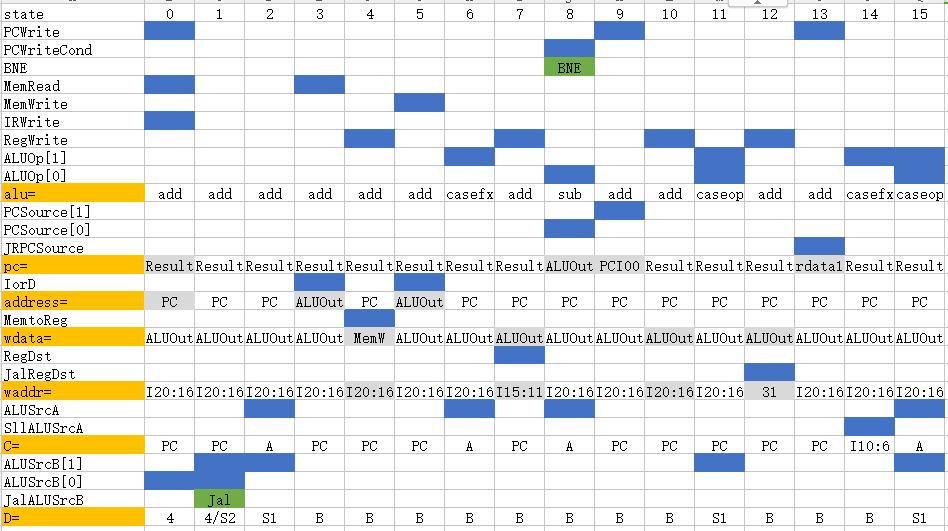
实验序号：4 实验名称：多周期处理器设计（下）

1. **代码以及波形图**

此次代码是在实验3的基础上增加指令，所以该部分就按增加的指令分写相应代码以及波形图。

图1addiu/li图2addu/move图3beq/beqz图4bnez图5j图6jal图7jr图8lui图9sll(nop)图10slti图11sltiu图12subu。

先上真值表部分。 （其中casefx表示是对instruction末6位的case，caseop是对前6位的case，S1/S2是instruction[15:0]的两种方式拓展，绿色的表示只有在那个指令下才是1，蓝色均表示1）



其次是状态机部分。（紫色表示有该状态，其上数字表示按顺序，例如addiu是state0→state1→state2→state10。）

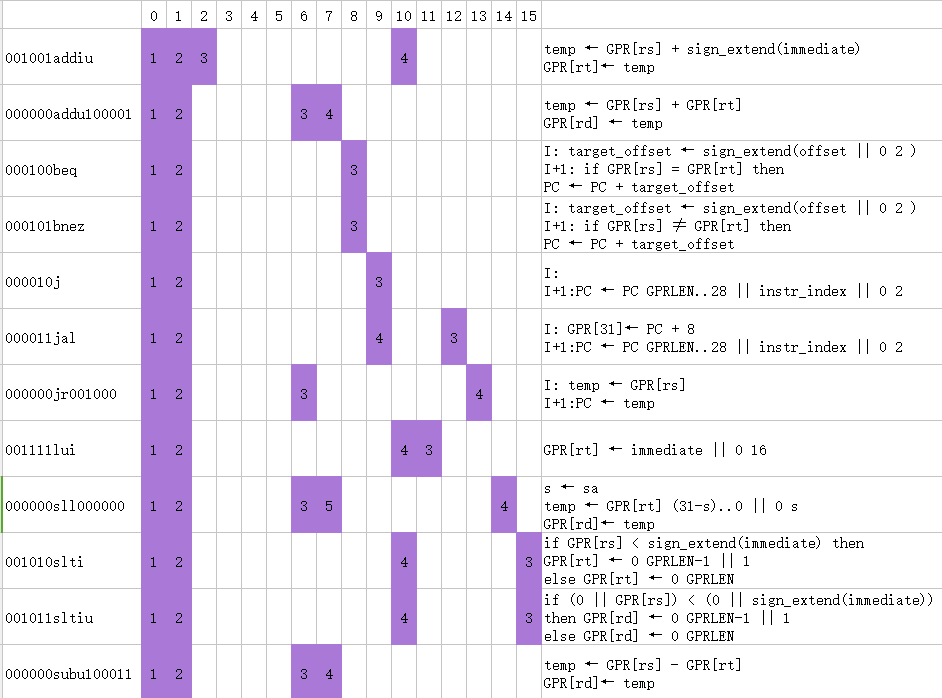
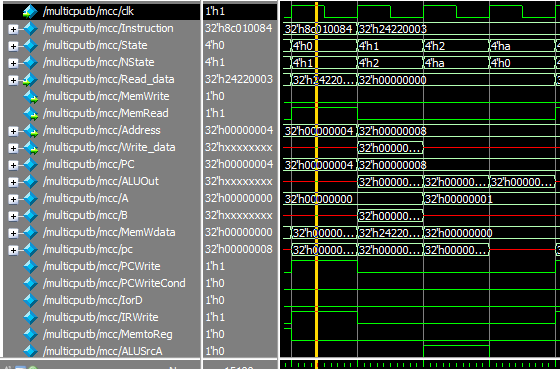


图1-addiu



state0→state1→state2→state10

相应代码只涉及了state10，而state10并没有新增其他变量，略去相应代码。

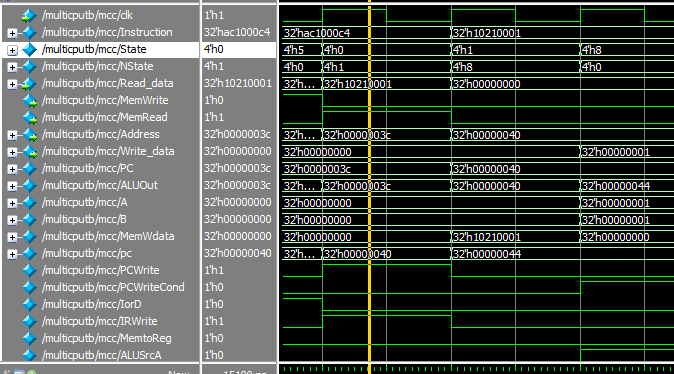
图2-addu

state0→state1→state6→state7

addu指令只涉及了实验3部分的内容，所以测试都没测试，无仿真波形，略去相应代码。

图3-beq

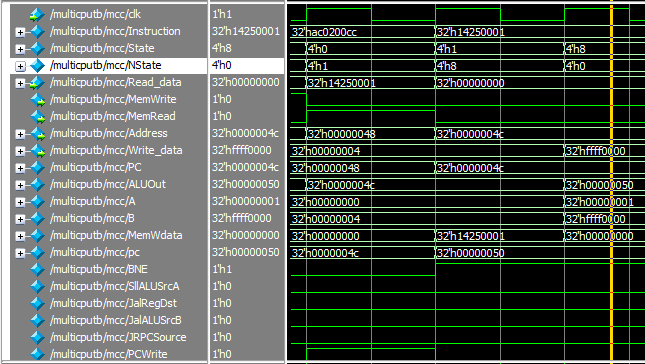
state0→state1→state8



代码部分写到beq的时候，仍然只涉及了实验3部分，虽然相应代码有所改动，但是因为bne的出现，故代码部分放置于bne部分中。

图4-bne

state0→state1→state8



图中BNE在状态8的时候为1，即我真值表状态8中的绿色部分。相应代码如下。



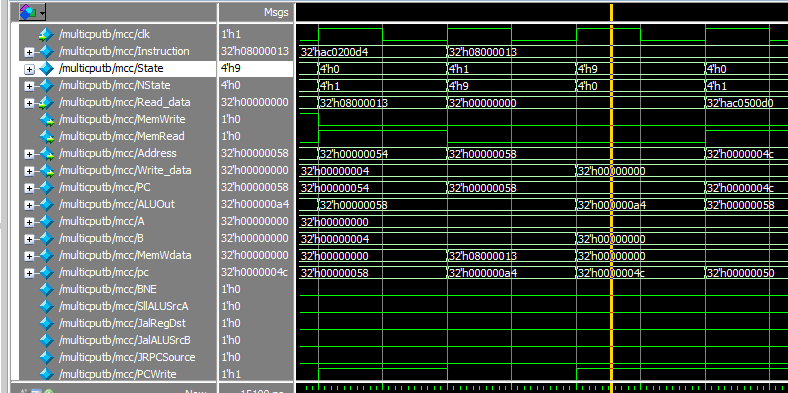
新增BNE变量，用以判断区分是bne还是beq，因为只有这2个指令才涉及到state8



在bne的情况下要Zero=0才对PC改写，beq则相反，所以利用BNE变量与Zero进行异或。

图5-j

state0→state1→state9

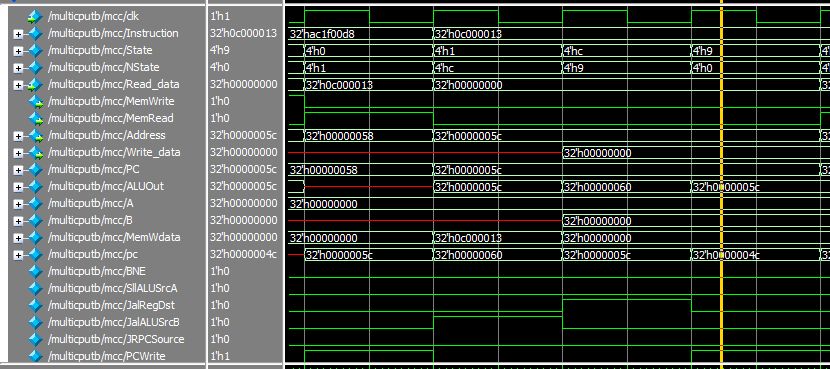


如图所示，在状态9的时候，pc（我代码中，若对PC进行改写，PC<=pc）已经变成了要跳转的地址32’h0000004c，在下一阶段PC也变成了这个值。

代码部分，在实验3中按照几个pdf中的真值表状态转换表已经完成了，故略去代码。

图6-jal

state0→state1→state12→state9



如图所示，在state9的时候，因为state12的操作，r[31]已经写入了PC+4的数据。

相应代码只涉及了state12，新增代码如下。

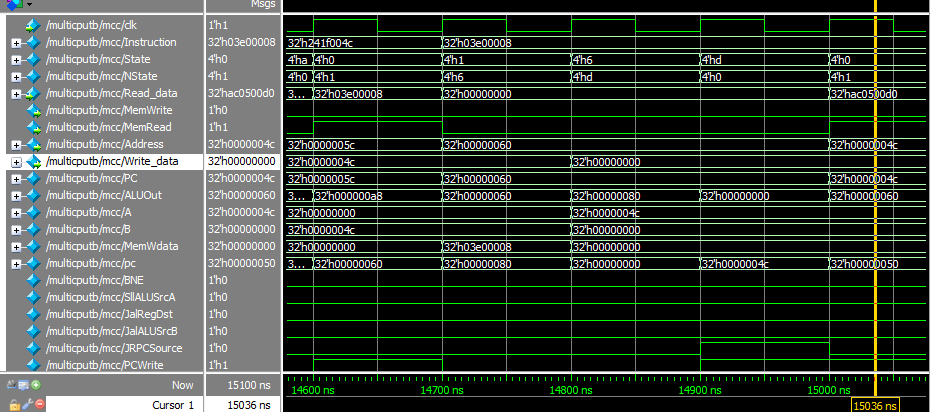






图7-jr

state0→state1→state6→state13



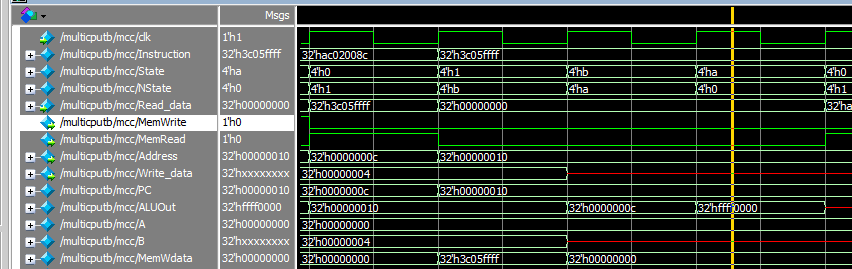
如图所示，在下一个state0的时候，因为jr指令已经跳到r[31]中的数32’h0000004c

相应代码只涉及了state13，新增代码如下。



图8-lui

state0→state1→state10→state11



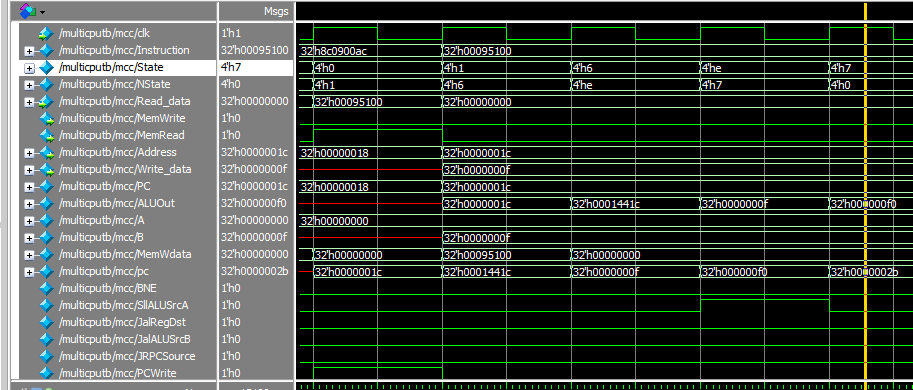


如图所示，该指令向r[5]中存了{Instruction[15:0],16’b0}

代码部分略去，涉及了state10，和state11。在state11中有关alu，alu部分代码放置于最后。

图9-sll

state0→state1→state6→state14→state7





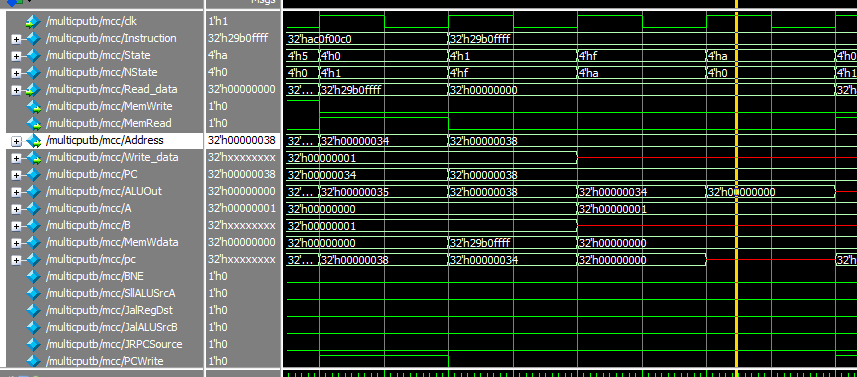
如图所示，该指令向r[10]中存了r[9]左移Instruction[10:6]位

代码部分，涉及了state14。在state14中有关alu，alu部分代码放置于最后。



图10-slti

state0→state1→state15→state10

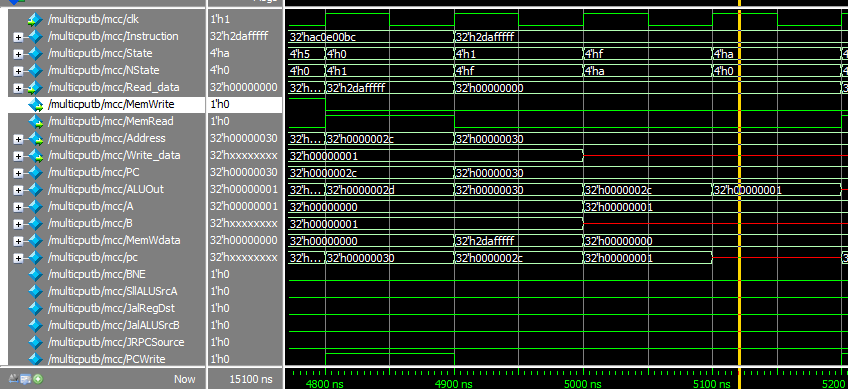


如图所示，该指令向r[16]中存了判断r[13]与-1的比较结果。

代码部分略去，涉及了state15，而state15涉及了alu，该指令中用的是以前写过的alu功能slt。

图11-sltiu

state0→state1→state15→state10



如图所示，该指令向r[15]中存了判断r[13]与ffff的无符号拓展比较结果。

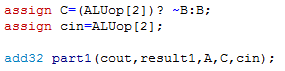
代码部分略去，涉及了state15，而state15涉及了alu，新增了alu功能，alu代码放在最后。

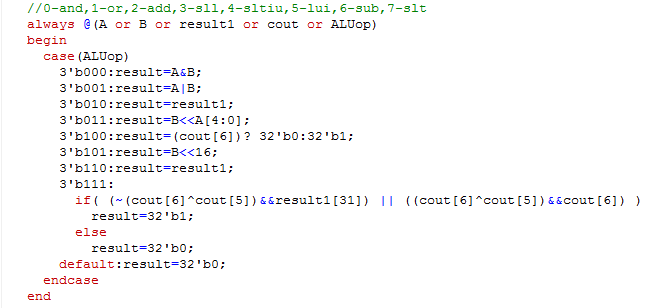
图12-subu

state0→state1→state6→state7

该代码与实验三的sub指令基本一致，且我们忽略了overflow的限制，所以无仿真波形。

Alu



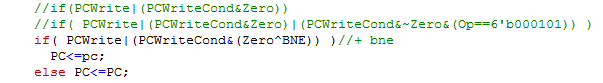


如图所示，在3’b011地方加入了sll，直接对B进行左移A[4:0]；3’b100加入了sltiu，保持了aluop[2]仍为1，使内部加法器做减法，并且由于是不会溢出的减法，所以直接判断符号位进位即可；3’b101加入了lui，直接对B进行左移16位。

1. **问题合集**

这次新增了十多条指令，增加了6个状态，3个alu功能，略去各种粗心造成的写错问题，还有一些思考上的遗漏错误。

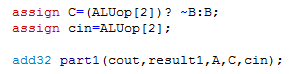
**Problem1**



问题：写bne的时候，忽略了一种情况，在bne 条件下Zero=1时，也会对PC进行改写。

解决：将BNE与Zero进行异或后再并上PCWriteCond，就没有上述错误出现。

**Problem2**





问题：之前在写sltiu的的时候，给他对应的aluop是011，然后就不知道怎么处理减法了，写的比较繁复

解决：将指令位置互换即可，把该指令换到100或101的位置上。

**Problem3**

无图

问题：PC+8的指令没有放入alu，直接在代码部分写了PC+8

解决：改写成进入alu做加法的PC+8。（后来改成了+4，因为借用同学的testbench要求的是+4）

1. **对于此次实验的心得、感受和建议**

增加了几条指令，不能像以前那样无脑按照真值表和状态表写了，每条指令都要自己耐心的去写应该需要哪些信号置1，问题产生也不像之前，耐心看下代码逻辑就可以了，有时候出错了，是自己遗漏的方面，只能对着仿真代码每个信号去看，他理论上应该是什么，但实际上是什么，发现错误，然后去看相应代码。因此此次作业做的比之前的也费时许多，也认真许多。

另外感谢各路大腿借我用的testbench，使我省了不少写testbench的功夫。就这样吧=。=希望下次作业不要死的太惨。