CPU设计文档思考题

**一、模块规格**

Q1.若PC（程序计数器）位数为30位，试分析其与32位PC的优劣。

A1:30位PC在连接IM时无需处理，但在对NPC进行计算时不如32位方便；32位PC计算NPC时无需处理，但在连接IM时需要去掉最后两位位。

Q2.现在我们的模块中 IM使用ROM， DM使用RAM， GRF使用寄存器，这种做法合理吗？ 请给出分析，若有改进意见也请一并给出。

A2：合理。

（1）IM在运行过程中无需写入，故使用ROM即可满足要求，还可以防止运行过程中由于失误而导致指令改变。

（2）DM使用RAM可以同时满足读和写的要求，且要求简单，使用RAM即可简单实现，无需使用寄存器自行搭建。

（3）GRF功能较复杂，无法直接使用RAM与ROM实现，故需使用寄存器自行搭建来实现其功能。

**二、控制器设计**

Q1.结合上文给出的样例真值表，给出RegDst， ALUSrc， MemtoReg，RegWrite, nPC\_Sel, ExtOp与op和func有关的布尔表达式（表达式中只能使用“与、或、非”3 种基本逻辑运算。）

A1：

RegDst=~op5 ~op4 ~op3 ~op2 ~op1 ~op0 (func5 ~func4 ~func3 ~func2 ~func1 ~func0 + func5 ~func4 ~func3 ~func2 func1 ~func0)

ALUSrc=~op5 ~op4 op3 op2 ~op1 op0 + op5 ~op4 ~op3 ~op2 op1 op0 + op5 ~op4 op3 ~op2 op1 op0

MemtoReg= op5 ~op4 op3 ~op2 op1 op0

RegWrite=(~op5 ~op4 ~op3 ~op2 ~op1 ~op0 (func5 ~func4 ~func3 ~func2 ~func1 ~func0 + func5 ~func4 ~func3 ~func2 func1 ~func0)) + ~op5 ~op4 op3 op2 ~op1 op0 + op5 ~op4 ~op3 ~op2 op1 op0

nPC\_Sel= ~op5 ~op4 ~op3 op2 ~op1 ~op0

ExtOp= op5 ~op4 ~op3 ~op2 op1 op0 + op5 ~op4 op3 ~op2 op1 op0

Q2.充分利用真值表中的 X 可以将以上控制信号化简为最简单的表达式， 请给出化简后的形式。

A2：同上？

Q3.事实上，实现nop空指令，我们并不需要将它加入控制信号真值表，为什么？请给出你的理由。

A3：因为nop的机器码为0x00000000，其并不执行任何操作，因此对应的各路控制信号值均为0，因此无需加入真值表。

**三、测试CPU**

Q1.前文提到，“可能需要手工修改指令码中的数据偏移”，但实际上只需再增加一个 DM片选信号,就可以解决这个问题。请阅读相关资料并设计一个DM 改造方案使得无需手工修改数据偏移。

A1：

Q2.除了编写程序进行测试外，还有一种验证CPU设计正确性的办法——形式验证。 形式验证的含义是根据某个或某些形式规范或属性，使用数学的方法证明其正确性或非正确性。请搜索“形式验证（Formal Verification)"了解相关内容后，简要阐述相比与测试，形式验证的优劣。

A2：

优点：对指定描述的所有可能的情况进行验证，覆盖率达到了100%；时间短，可以很快发现和改正电路设计中的错误，缩短设计周期。

缺点：在实现时序逻辑验证上存在缺陷；无法测试出设计上的错误。