**五级流水线cpu设计及优化**

1. **测试结果**

**功能点测试**

**实现了MIPS全部指令，通过了全部的91条测试。**

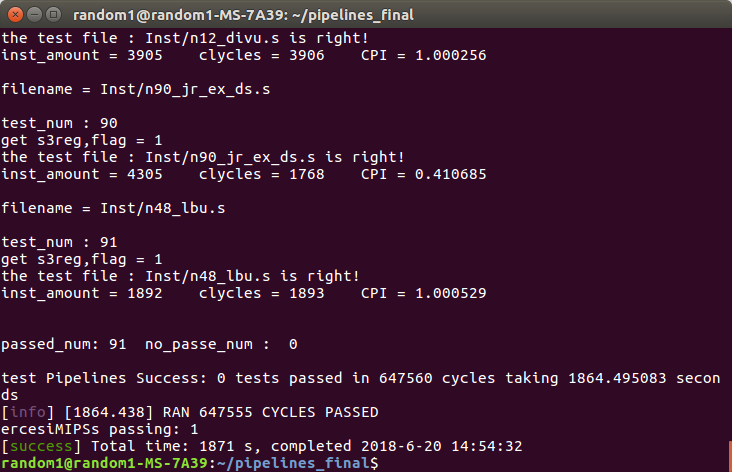
****

图1-1 C仿真通过全部测试

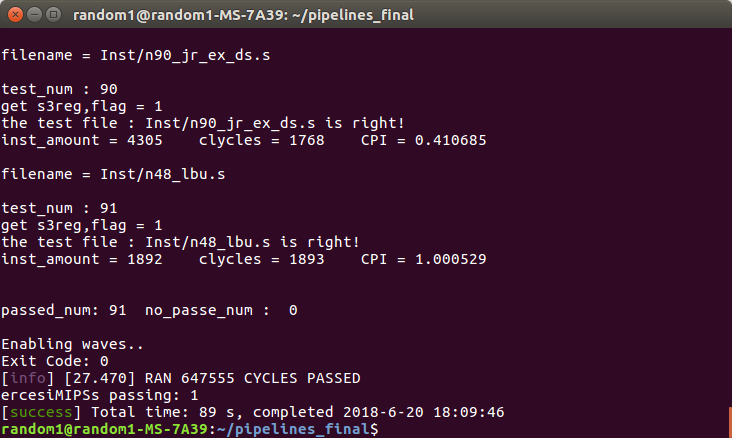


图1-2 V仿真通过全部测试

**分支预测测试**

**实现了本地以及全局相结合的分支预测器，平均预测正确率约94.9826%。**

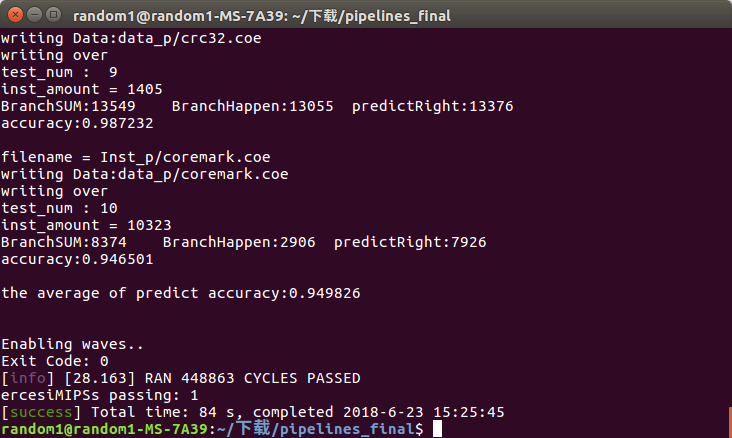
****

图1-3 分支预测正确率测试

1. **测试方法**

src/main/scala/Pipeline/Pipelines.scala文件中每个阶段后面都跟有输出，现已注释，如果需要看到具体的执行效果，将注释取消即可。

**功能点测试**

**①verilog仿真**

进入文件夹运行其中的**run-pipe-v.sh**文件可以进行verilog仿真，耗时约90s（有可能运行不了，因为对版本有一定要求）。

**②c仿真**

运行其中的**run-pipe-c.sh**文件可以进行c仿真，耗时约半个小时。

**分支预测测试**

运行其中的**run-pipe-pc.sh或run-pipe-pv.sh**文件可以进行测试，对十个测试程序进行分支预测测试。

由于该测试程序消耗资源过大，故限定了运行周期，不影响分支预测测试。

1. **设计思路**

**电路图位于根目录中，其中红线与tunnel同理。**

在上一次作业中实现了除了异常的所有指令，故不再赘述，现对本设计中关于异常处理以及分支预测作详细阐述。

**1.异常处理**

对应于CP0协处理器模块，实现了ERET,MFC0,MTC0,BREAK,SYSCALL五条指令，并可以处理溢出、地址错例外、保留指令例外等异常情况。

具体实现：

①当检测到异常时，根据具体情况将CP0中的BadVAddr，Status，Cause，EPC置为合适的值，如下图所示：

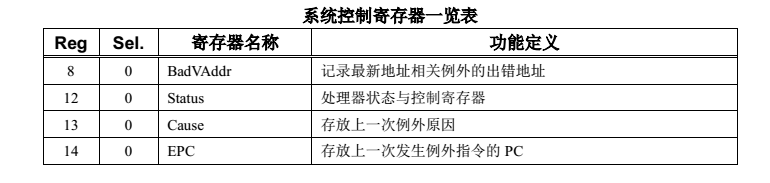


图3-1 系统控制寄存器一览表

②将当前出现异常的指令以及已经进入流水线的之前的指令全部刷为空。

③pc跳到具体的异常处理入口，在本设计中为0x000040b0。

④异常处理完成后，通过ERET指令回到EPC中记录的PC值继续执行。

**2.分支预测**

**①静态分支预测：**

即cpu默认选择预测选择分支跳转或者不跳转的一种静态预测方式。

**②本地分支预测：**

一个分支历史寄存器（Branch History Register, BHR），用于记录所有相关分支最近的执行历史。一个双模态预测机，用两位状态来记录该指令当前状态，并根据跳转情况实现一个四个状态的状态机，如下图所示：

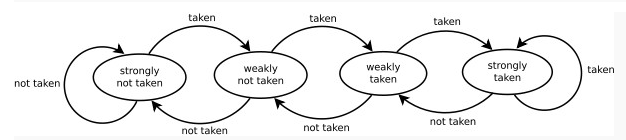


图3-2 双模态预测机

其中00,01,10,11分别表示强不选择，弱不选择，弱选择，强选择。如一段循环程序需要循环100次后跳转，那么该动态预测的命中率就在98%左右，随着循环次数的增加，命中率不断提高。

**③全局分支预测：**

全局分支预测器（global branch predictor）并不为每条条件跳转指令保持专用的历史记录。相反，它保持一份所有条件跳转指令的共用的历史记录。优点是能识别出不同的跳转指令之间的相关性。缺点是历史记录被不相关的不同的条件跳转指令的执行情况稀释了（diluted）；

**④融合分支预测：**

本设计结合了这三种分支预测器，即CPU可以自行在合适的情况下选择某种分支预测器。当在BHR分支表中有当前指令的历史记录时，则选择本地分支预测器，当没有当前指令的历史记录时，则向GBP中的全局信息中搜寻，使用全局分支预测。而当无法找到任何信息时，则使用静态分支预测的方法。

仅仅使用本地分支预测的命中率约为91%，而使用融合分支预测达到了约95%的预测命中率。能达到这么高的命中率也有部分是因为测试程序中循环非常之多。

1. **布线结果**

使用Quartus软件仿真后布线结果在根目录中的仿真.pdf。

1. **具体代码**

**模块代码在src/main/scala/Pipeline文件夹中，而测试代码在src/test/scala/Pipeline中。**

下对代码结构作简要阐述：

**①模块代码**

**Pipeline.scala**:流水线cpu的主体代码，包括所有五级以及Top层，代码中分段注释使结构已经较为清晰。

**ALU.scala,CMP.scala**:本质为ALU模块，为了清晰分开写。

**CP0.scala**:协处理器模块，用于所有的异常处理。

**Datachek.scala**:数据冒险单元，只提供了两部分数据冒险处理，剩余的数据冒险处理在Pipeline.scala中实现。

**Define.scala**:将代码中出现的指令以及控制信号作了自定义，便于代码阅读。

**①测试代码**

**PipeTests.scala**：功能测试文件，默认将Inst文件夹中的所有测试文件测试。

**Launcher.scala**：功能测试仿真启动文件。

**PipeTests\_p.scala**：分支预测测试文件，默认将Inst\_p文件夹中的所有测试文件测试。

**Launcher\_p.scala**：分支预测测试仿真启动文件。