**RV32I指令集流水线CPU设计报告**

PB16111245王立峰

**一、主要模块的设计思想**

1、RV32Core.v

此模块为RV32I 指令集CPU的顶层模块，声明了各个主要模块单元，以及总线排布情况；

2、ALU.v

算术逻辑单元模块；

3、BranchDecisionMaking.v

分支预测模块；

4、ControlUnit.v

控制单元模块，根据输入的Op, fn3, fn7信号来出所有控制信号，分别对每个输出信号分析是在那些指令会用到，对应的值是多少；

5、DataExt.v

该模块是用来处理非字对齐load的情形，同时根据load的不同模式对Data Mem中load的数进行符号或者无符号拓展，组合逻辑电路，根据输入的RegWrite确定相应的是符号拓展还是零拓展；

6、HazardUnit.v

该模块用来处理流水线冲突，通过插入气泡，forward以及冲刷流水段解决数据相关和控制相关，组合逻辑电路；主要是数据转发，stall和flush;

7、ImmOperandUnit.v

立即数模块，利用正在被译码的指令的部分编码值，生成不同类型的32bit立即数；

8、NPC\_Generator.v

PC计数器模块，用来生成Next PC值得模块，根据不同的跳转信号选择不同的新PC值；

9、Parameters.v

定义常量值模块；

10、RegisterFile.v

寄存器文件模块；

11、IFSegReg.v

IF-ID段寄存器模块；

12、IDSegReg.v

IF-ID段寄存器模块；

13、EXSegReg.v

EX段寄存器模块，本模块是支持同步清零的段寄存器，当EN==0时寄存器状态保持不变（也不会执行清零）；

14、MEMSegReg.v

EX-MEM段寄存器模块；

15、WBSegReg.v

WB段寄存器模块；

16、DataRam.v

数据存储器模块；

17、InstructionRam.v

指令存储器模块；

**二、回答问题**

1、为什么将DataMemory和InstructionMemory嵌入在段寄存器中？

直接进入下一个流水阶段，不用等待下一个周期

2、DataMemory和InstructionMemory输入地址是字（32bit）地址，如何将访存地址转化为字地址输入进去？

地址使用A[31,2]

3、如何实现DataMemory的非字对齐的Load？

进行选位和拓展

4、如何实现DataMemory的非字对齐的Store？

WE写使能信号控制

5、为什么RegFile 的时钟要取反？

不同步读，便于五级流水

6、NPC\_Generator中对于不同跳转target 的选择有没有优先级？

有优先级

7、ALU模块中，默认wire变量是有符号数还是无符号数？

无符号数

8、AluSrc1E执行哪些指令时等于1’b1？

AUIPC

9、AluSrc2E执行哪些指令时等于2‘b01？

SLLI, SRAI,SRLI

10、哪条指令执行过程中会使得LoadNpcD==1？

JALR, JAL

11、DataExt模块中，LoadedBytesSelect的意义是什么？

选位

12、Harzard模块中，有哪几类冲突需要插入气泡？

LOAD相关

13、Harzard 模块中采用默认不跳转的策略，遇到branch 指令时，如何控制ﬂush 和stall信号？

FlushD=1, FlushE=1

14、Harzard模块中，RegReadE 信号有什么用？

判断是否需要转发

15、0号寄存器值始终为 0，是否会对forward的处理产生影响？

可能会

**三、总结**

还没有开始写Verilog代码，目前只是各个模块看了一下，把设计图细细地观察了一下，向同学了解了一下大致流程，总的感觉想要完成整个项目十分困难，有点慌……