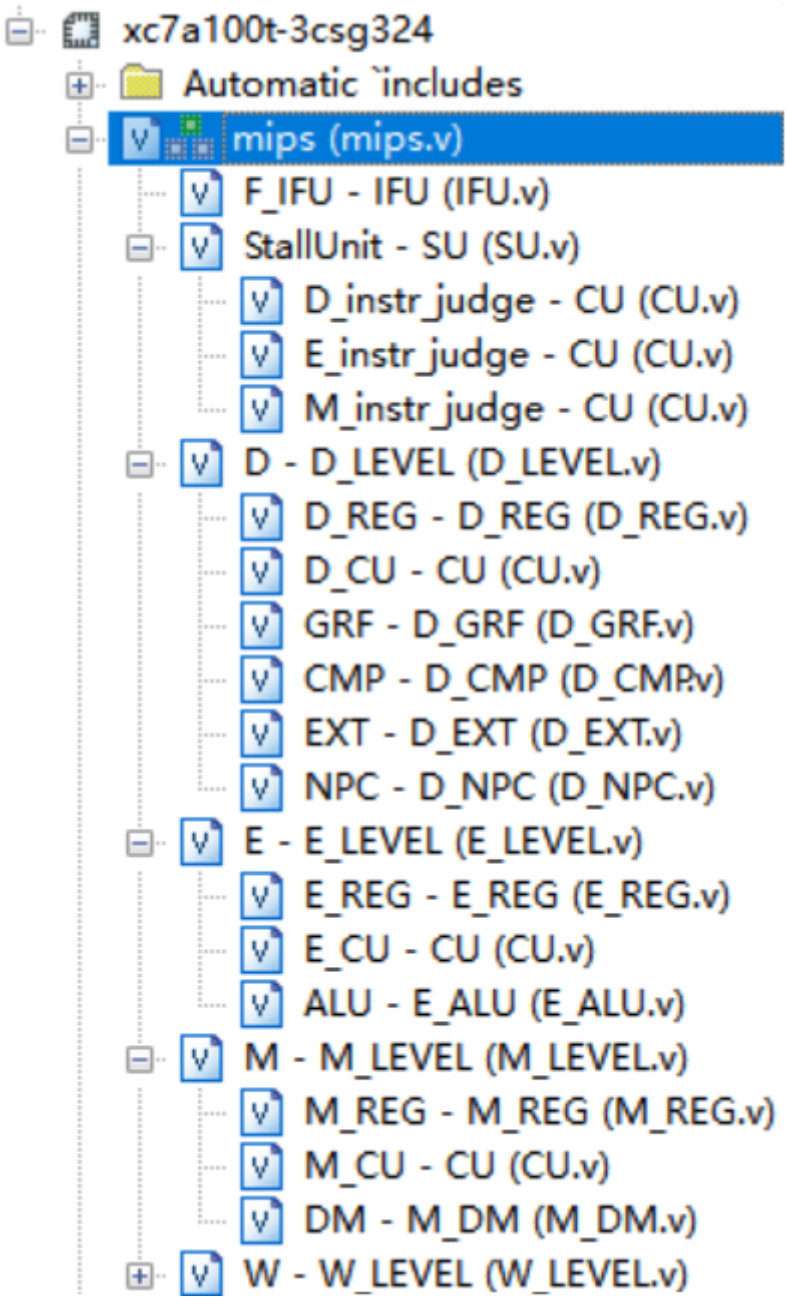
Verilog流水线CPU设计

一、CPU设计方案综述

（一）总体设计概述

本CPU为Logisim实现的32位5级流水线MIPS-CPU，支持指令集包含`{addu, subu, ori, lw, sw, beq, j, jal, jr, lui, nop, j, lh, lb, sh, sb}`。为实现相关功能，CPU主要包含IFC、NPC、NPC、IM、GRF、EXT、ALU、DM、CU、SU等模块。遵循形式化建模综合方法完成设计与实现。

（二）关键模块定义



（三） 重要机制实现方法

1.跳转

这里确实容易出错， 因为寄存器的原因导致了PC和NPC的分离，若使用D级PC信号会导致NPC的滞后而产生类似指令回流错觉的错误。因而得用F级PC的信号做PC正常累加。其他的同单周期CPU设置NPCOp的方法。

2.流水线延迟槽

jal的改存PC+8进入寄存器

3.转发

E、M、W级都具有完整指令，将其解析出所有写寄存器的信息（不管是不是写入指令）一起转发到相应D、E、M共5处转发位点。设置转发判断相关条件已控制是否选择转发的信号

4.暂停

AT法分析Tnew和Tuse并比较，当Tuse < Tnew且当前指令是冲突指令（需要转发）时，需要暂停

T_use < T_new : 暂停

指令	T_use	
	rs([25:21])	rt([20:16]])
cal_r	1	1
cal_i	1	
load	1	
store	1	2
branch	0	0
j_reg	0	0
j_link		
j_addr		

D			E			M
指令	源寄存器	T_use	Tnew_E			Tnew_M
			cal_r (rd)	cal_i (rt)	load (rt)	load (rt)
			1	1	2	1
cal_r	rs/rt	1			stall	
cal_i	rs	1			stall	
load	rs	1			stall	
store	rs	1			stall	
store	rt	2				
branch	rs/rt	0	stall	stall	stall	stall
j_reg	rs	0	stall	stall	stall	stall

二、测试方案

太急了，没时间好好测

三、 思考题

（一）流水线冒险

1.在采用本节所述的控制冒险处理方式下，PC 的值应当如何被更新？请从数据通路和控制信号两方面进行说明。

数据通路： F 级和 D 级相连，由 F 级提供当前 PC 寄存器值，由 D 级提供下一个 PC 值所需的其他值，并由 NPC 输出下一个 PC 值。注意由于寄存器的分割，F和D级的PC不同，应遵从F级PC设置相应NPC

控制信号：D级控制器解码出NPCOp，控制NPC输出

2.对于 jal 等需要将指令地址写入寄存器的指令，为什么需要回写 PC+8？

默认支持延迟槽

（二）数据冒险的分析

1.为什么所有的供给者都是存储了上一级传来的各种数据的流水级寄存器，而不是由 ALU 或者 DM 等部件来提供数据?

组合逻辑的运算延迟各不相同，CPU流水线各级延迟不均衡，将导致流水线性能下降，同时也不利于计算设置流水线时钟频率。

（三）AT 法处理流水线数据冒险

1.“转发（旁路）机制的构造”中的 Thinking 1-4；

Thinking 1：如果不采用已经转发过的数据，而采用上一级中的原始数据，会出现怎样的问题？试列举指令序列说明这个问题。

那么使用的数据是未经前序指令写入的旧值，显然会导致错误。只要后指令要用的数据段是前指令要写的数据段（W-R），则会产生这样的问题 `addu $1 $2 $3` 和 `addu $5 $1 $4`。

Thinking 2：我们为什么要对 GPR 采用内部转发机制？如果不采用内部转发机制，我们要怎样才能解决这种情况下的转发需求呢？

RFA3和A1或A2相同且不为0时，直接选取RFWD的内容作为RD1或RD2的内容输出。不内部转发的话，只好从W级转发数据到D级的RD端，那样子挺蠢。

Thinking 3：为什么 0 号寄存器需要特殊处理？

对0号寄存器的写入无效，若不特殊处理转发将写入0号寄存器的数据则会导致错误

Thinking 4：什么是“最新产生的数据”？

离转发位点最近流水级处的数据。

2.在 AT 方法讨论转发条件的时候，只提到了“供给者需求者的A相同，且不为 0”，但在 CPU 写入 GRF 的时候，是有一个 we 信号来控制是否要写入的。为何在 AT 方法中不需要特判 we 呢？为了用且仅用 A 和 T 完成转发，在翻译出 A 的时候，要结合 we 做什么操作呢?

只有要写寄存器的指令才有 Tnew，需要寄存器值的指令才有 Tuse，因而不必特判 we 信号来确定；判断当前是否需要阻塞。

（四）在线测试相关说明

在本实验中你遇到了哪些不同指令类型组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的?

cal_r/cal_i -- cal_r/cal_i
load -- load
cal_r/cal_i -- branch/jump

如果你是手动构造的样例，请说明构造策略，说明你的测试程序如何保证**覆盖**了所有需要测试的情况；如果你是**完全随机**生成的测试样例，请思考完全随机的测试程序有何不足之处；如果你在生成测试样例时采用了**特殊的策略**，比如构造连续数据冒险序列，请你描述一下你使用的策略如何**结合了随机性**达到强测的效果。

此思考题请同学们结合自己测试 CPU 使用的具体手段，按照自己的实际情况进行回答。