体系结构LAB5实验报告

JL19110004 徐语林

一.实验目标

1.熟悉Tomasulo模拟器和cache一致性模拟器(监听法和目录法)的使用

2.加深对Tomasulo算法的理解，从而理解指令级并行的一种方式-动态指令调度

3.掌握Tomasulo算法在指令流出，执行，写结果各阶段对浮点操作指令以及load和store指令进行什么处理；给定被执行代码片段，对于具体某个时钟周期，能够写出保留站，指令状态表以及浮点寄存器状态表内容的变化情况

4.理解监听法和目录法的基本思想，加深对多cache一致性的理解

5.做到给出指定的额读写序列，可以模拟出读写过程中发出的替换，换出等操作，同时模拟出cache块的无效，共享和独占态的相互切换

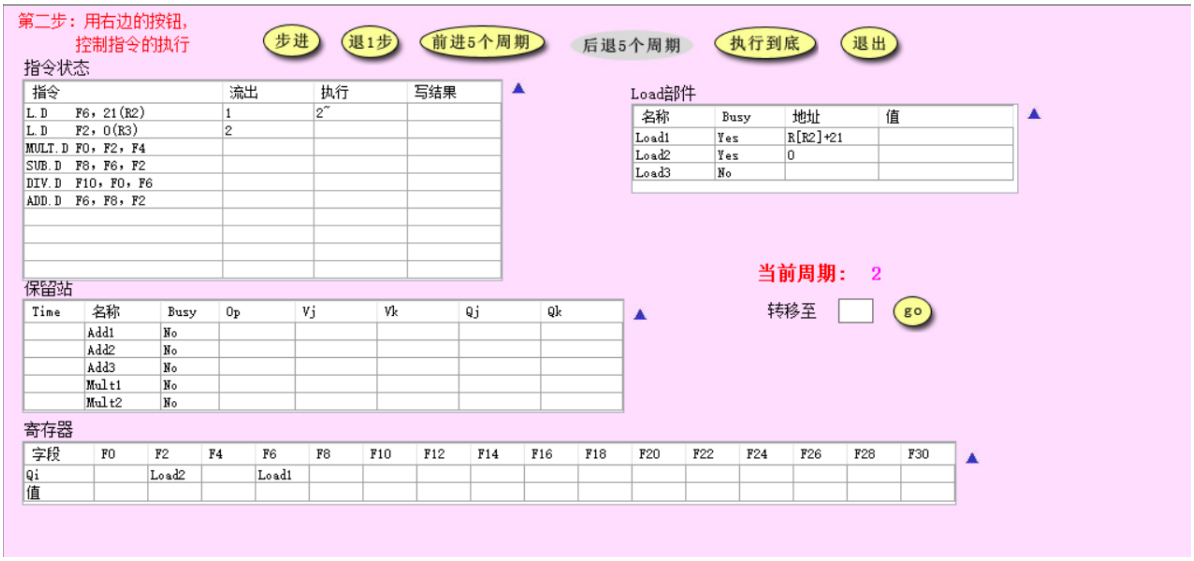
二.实验环境和工具

Lab5-模拟器

三.实验过程及问题的回答

1.Tomasulo算法模拟器

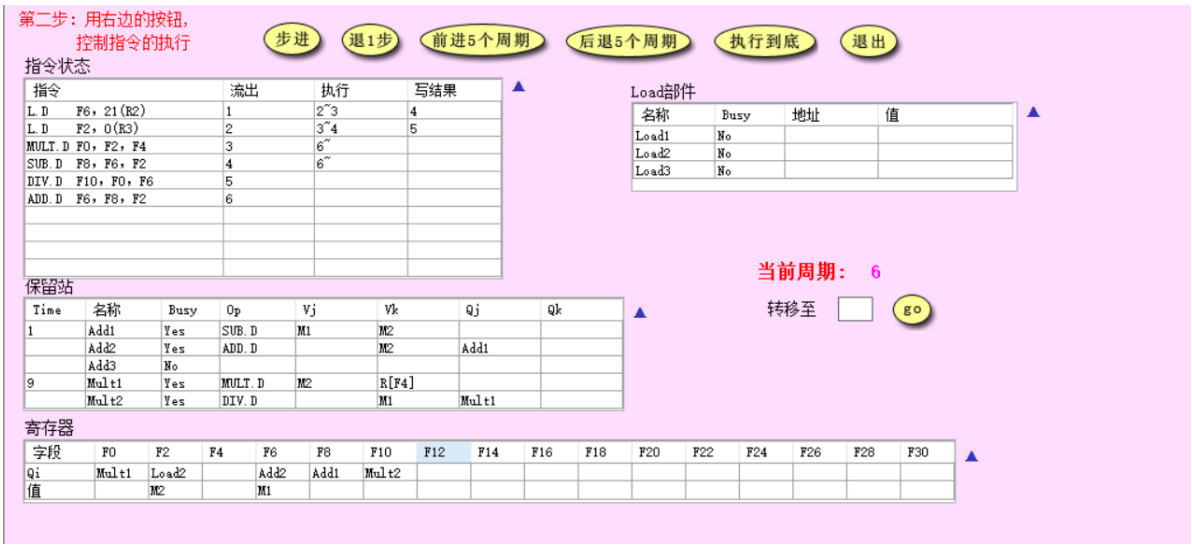
(1).分别截图（当前周期2和当前周期3），请简要说明load部件做了什么改动



load部件的改动：因为第二个周期第二条ld指令流出，会占用一个load部件，因此部件2状态为busy,同时偏移量为0写入地址字段；对于第一条ld指令，将计算出的有效地址写入地址字段

load部件的改动：对于第一条ld指令，将存储器中地址为R[R2]+21的值写入值字段；对于第二条指令，将计算出的有效地址写入地址字段

(2).请截图（MUL.D刚开始执行时系统状态），并说明该周期相比上一周期整个系统发生了哪些改动（指令状态、保留站、寄存器和Load部件）



Load部件：无任何变化

寄存器：F6的Qi字段由Load1变为Add2，这是由于5周期之后L.D指令已经执行结束，ADD.D指令流出所导致的

保留站：由于ADD.D指令的流出会占据一个Add2部件，因此Busy表示改为YES,写入Op以及Vk,Qj；同时Add1的执行周期变为1，Mult1的执行周期变为9

指令状态：ADD.D指令在第六个周期流出，SUB.D和MULT.D指令开始执行

(3)简要说明是什么相关导致MUL.D流出后没有立即执行

因为在MULT流出之后，需要读取F2的值，但是在此之前的上一条指令的LD并没有写回F2

(4)请分别截图（15周期和16周期的系统状态），并分析系统发生了哪些变化

15周期：

第二步：用右边的按钮，
控制指令的执行

步进 退1步 前进5个周期 后退5个周期 执行到底 退出

指令状态

指令	流出	执行	写结果
L.D F6，21(R2)	1	2~3	4
L.D F2，0(R3)	2	3~4	5
MULT.D F0，F2，F4	3	6~15	
SUB.D F8，F6，F2	4	6~7	8
DIV.D F10，F0，F6	5		
ADD.D F6，F8，F2	6	9~10	11

Load部件

名称	Busy	地址	值
Load1	No		
Load2	No		
Load3	No		

当前周期： 15

转移至 go

保留站

Time	名称	Busy	Op	Vj	Vk	Qj	Qk
	Add1	No					
	Add2	No					
	Add3	No					
	Mult1	Yes	MULT.D	M2	R[F4]		
	Mult2	Yes	DIV.D		M1	Mult1	

寄存器

字段	F0	F2	F4	F6	F8	F10	F12	F14	F16	F18	F20	F22	F24	F26	F28	F30
Qi	Mult1	Load2		Add2	Add1	Mult2										
值		M2		M4	M3											

16周期：

保留站：由于MULT指令执行完毕，因此释放MULT部件，BUSY段置为NO；MULT2监听到MULT1的数据后，修改Vj字段

Load部件：无任何变化

寄存器：F0的值字段变为M5

指令状态：MULT指令执行完毕

(5)回答所有指令刚刚执行完毕时是第多少周期，同时请截图（最后一条指令写CBD时认为指令流执行结束）

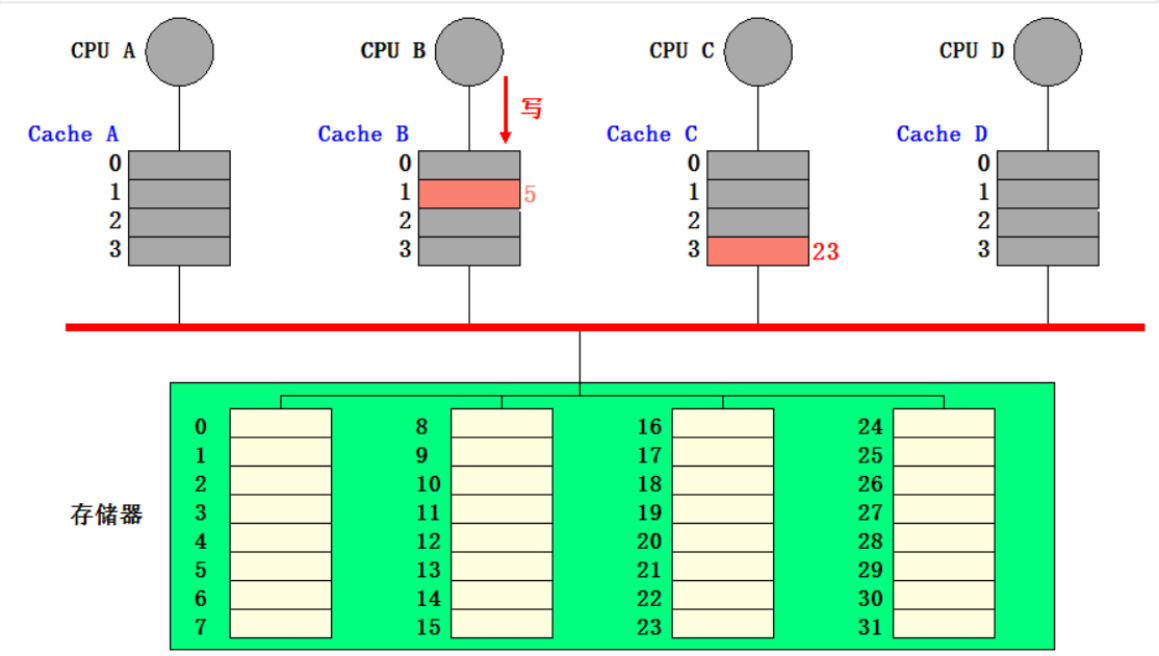
第57周期

2.多cache一致性算法-监听法

(1)利用模拟器进行下述操作，并填写下表

所进行的访问	是否发生了替换	是否发生了写回	监听协议进行的操作与块状态改变
CPU A 读第5块	否	否	Cache A读不命中，读不命中传到总线，从存储器中读取第5块的数据到cache A,cache A 块1状态由无效改为共享
CPU B 读第5块	否	否	Cache B读不命中，读不命中传到总线，从存储器中读取第5块的数据到cache B,cache B块1状态由无效改为共享
CPU C 读第5块	否	否	Cache C读不命中，读不命中传到总线，从存储器中读取第5块的数据到cache C,cache C块1状态由无效改为共享
CPU B 写第5块	否	否	Cache B写命中，作废信号传到总线，Cache A和Cache C块1的状态修改为无效，Cache B写入数据后，状态修改为独占
CPU D 读第5块	否	是	Cache D读不命中，读不命中传到总线，Cache B 写回块1，修改块1状态为共享，从存储器中读取第5块的数据到cache D,cache D块1的状态由无效改为共享
CPU B 写第21块	是	否	Cache B写不命中，写不命中传到总线，替换Cache B的块1，Cache B块1的状态由共享改为独占
CPU A 写第23块	否	否	Cache A写不命中，写不命中传到总线，cache A块3的状态由无效改为独占
CPU C 写第23块	否	是	Cache C写不命中，写不命中传到总线，Cache A写回块3，且状态修改为无效，Cache C写块3，状态为无效
CPU B 读第29块	是	是	Cache B读不命中，写回块1，状态修改为无效；存储器中读取29块的数据到Cache B的块1，状态为共享
CPU B 写第5块	是	否	Cache B写不命中，写不命中传到总线，从存储器读取块5到Cache B的块1，状态修改为独占，Cache D块1状态为无效

(2)请截图，展示执行完以上操作后整个cache系统的状态

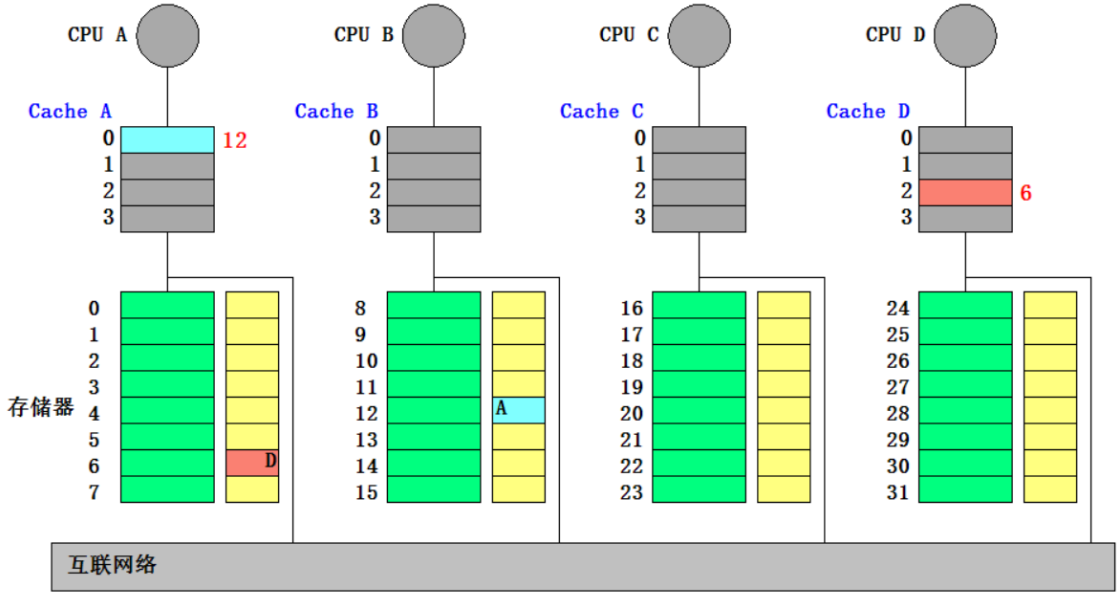


3.多cache一致性算法-目录法

(1)利用模拟器进行下述操作，并填写下表

所进行的访问	监听协议进行的操作与块状态改变
CPU A 读第6块	读不命中，本地向宿主结点发读不命中(A,6)消息，宿主把数据块送给本地结点，共享集合为{A};存储器块6和Cache A中块2的状态变为共享
CPU B 读第6块	读不命中，本地向宿主结点发读不命中(B,6)消息，宿主把数据块送给本地结点，共享集合为{A,B};Cache B的块2状态为共享
CPU D 读第6块	读不命中，本地向宿主结点发读不命中(D,6)消息，宿主把数据块送给本地结点，共享集合为{A,B,D};Cache D的块2状态为共享
CPU B 写第6块	写命中，本地向宿主结点发写命中(B,6)消息，宿主向远程结点A发作废(6)消息，宿主向远程结点D发作废(6)消息，共享集合为{B},状态变为独占，Cache A,D块状态变为无效，Cache B块2状态变为独占
CPU C 读第6块	读不命中，本地向宿主结点发读不命中(C,6)消息，宿主向远程结点发取数据块(6)的消息，远程把数据块送给宿主结点，Cache B块2状态变为共享，宿主把数据块给本地结点，Cache C的块2变为共享状态，共享结合为{B,C}
CPU D 写第20块	写不命中，本地向宿主结点发写不命中(B,20)消息，宿主把数据块送给本地结点，共享集合为{D},Cache C块2状态为共享，Cache D块0状态为独占，存储器块20状态为独占
CPU A 写第20块	写不命中，本地向宿主结点发写不命中(A,20)消息，宿主向远程结点发送取并作废(20)的消息，远程把数据块送给宿主结点，把cache中的该块作废。宿主把数据块给本地结点，共享集合为{A},Cache A块0状态为独占
CPU D 写第6块	写不命中，本地向宿主结点发写不命中(B,6)消息，宿主向远程结点B,C发作废(6)消息，Cache B,C块2状态变为无效，宿主A把数据送给本地结点，Cache D 块2状态变为独占，存储器块6状态变为独占，共享集合变为{D}
CPU A 读第12块	读不命中，本地向被替换块的宿主结点发写回并修改共享集(A,20)消息，存储器块20状态变为未缓冲，本地向宿主结点发读不命中(A,12)消息，宿主把数据块送给本地结点，共享集合为{A},存储块12和Cache A块0状态为共享

(2)请截图，展示执行完以上操作后整个cache系统的状态



四.综合回答

(1)目录法和监听法分别是集中式和基于总线，两者优劣是什么？（言之有理即可）

目录法：优点：缺失时不需要向所有其他的cache进行广播

缺点：存在有目录的开销，因此成本更高

监听法：优点：可以降低成本

缺点：总线上可能会出现一定的瓶颈，以及一定的竞争

(2)Tomasulo算法相比Score Board算法有什么异同？（简要回答两点：1.分别解决了什么相关，2.分别是分布式还是集中式）（参考第五版教材）

同：都检测到了结构相关；都是通过推迟指令的执行，直到操作数可用来避免的RAW相关

异：后者可以检测到WAR和WAW，但是无法消除，而前者可以通过寄存器重命名来消除。

前者由于有很多部件，因此属于分布式，后者利用的是记分牌，因此是集中式

(3)Tomasulo算法是如何解决结构、RAW、WAR和WAW相关的？（参考第五版教材）

WAW和WAR：通过了寄存器的重命名来消除的WAW和WAR相关

RAW:如果操作数不可用，便进行数据的监听，当一个操作数可用那么就放到等待它的保留站中，直到所有操作数准备完毕，则进行下一步，通过了延迟指令的执行阶段来消除RAW

结构相关：如果保留站为空，那么可以执行，这时说明没有结构相关；如果保留站不为空，那么该指令会停顿，说明有结构相关

四.实验总结与建议

本次实验对于Tomosulo算法以及多cache一致性的算法有了更为深入的理解，跟着模拟器一步一步的也让之前上课有疑问的算法变得更为形象，易于理解。由于是本学期最后一个体系结构实验，总体来说实验很符合课程的内容，也体会到了人类对于速度的追求。

建议：无。完结散花~