2023 年秋季计算机体系结构——大作业

Speculative Tomasulo 算法实现

1. Speculative Tomasulo 算法介绍

Tomasulo 算法是一种用于指令动态调度的计算机架构硬件算法，能够有效地利用多个执行单元乱序执行指令。Tomasulo 算法因为控制指令、程序异常和外部中断会截断指令流而无法顺序提交指令，而乱序提交无法满足处理器按照程序顺序来执行指令的需求。

Speculative Tomasulo 算法是带有重新排序缓存的 Tomasulo。重排序缓存可以让乱序执行的指令被顺序地提交，其核心思想是记录下指令在程序中的顺序，一条指令在执行完毕之后不会立马提交，而是先在 Buffer 中等待，等到前面的所有指令都提交完毕，才可以提交结果到逻辑寄存器堆。在大作业中，您将使用熟悉的编程语言（例如，c/c++/java/python 等）实现 Speculative Tomasulo 算法。然后，您还需要回答第 3 节的的附加问题。

当每个周期的指令执行完毕后，您的程序应逐周期输出保留站状态（reservation status）、重新排序缓存（reorder buffer）状态和寄存器结果状态（register result status）。示例输出应该像“Chap03-ILP-Part4-Speculation”里一样清晰易读。具体的输入和输出格式请参见附录 1。

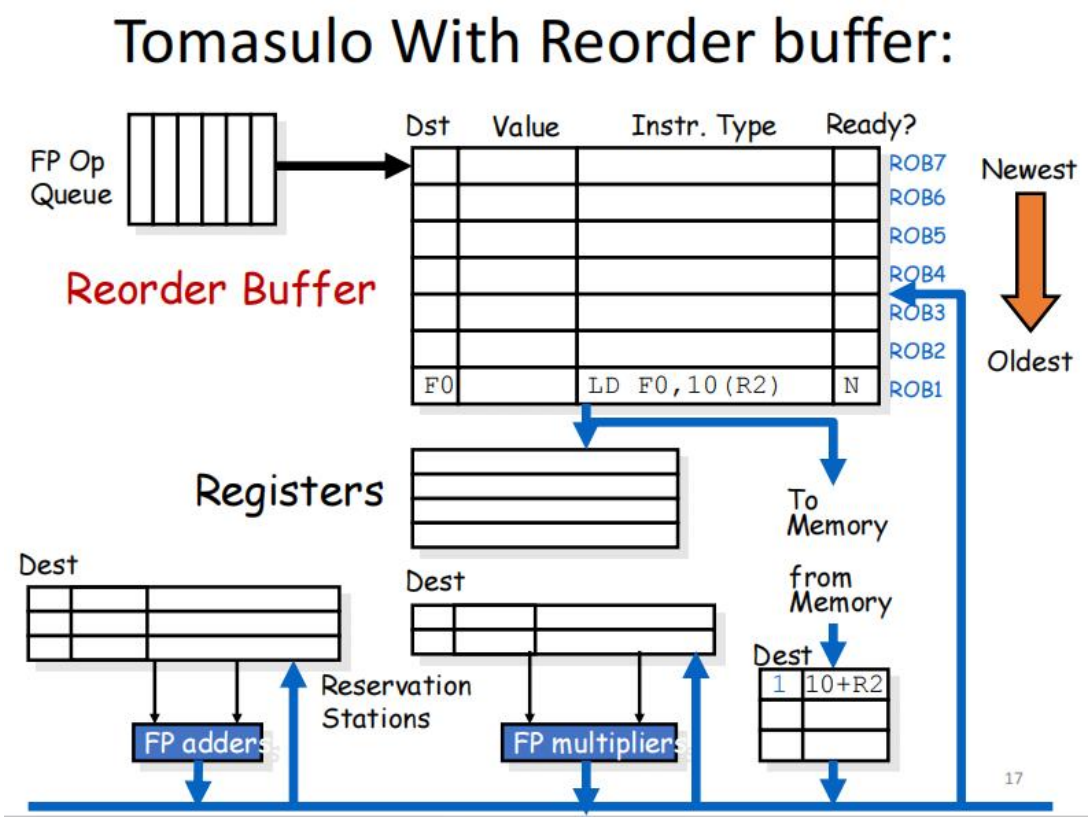

图 1 带有重新排序缓存的 Tomasulo

2. 关于你的仿真器

使用带有重新排序缓存 Tomasulo 算法的 MIPS 浮点单元基本结构，如图 2 所示。

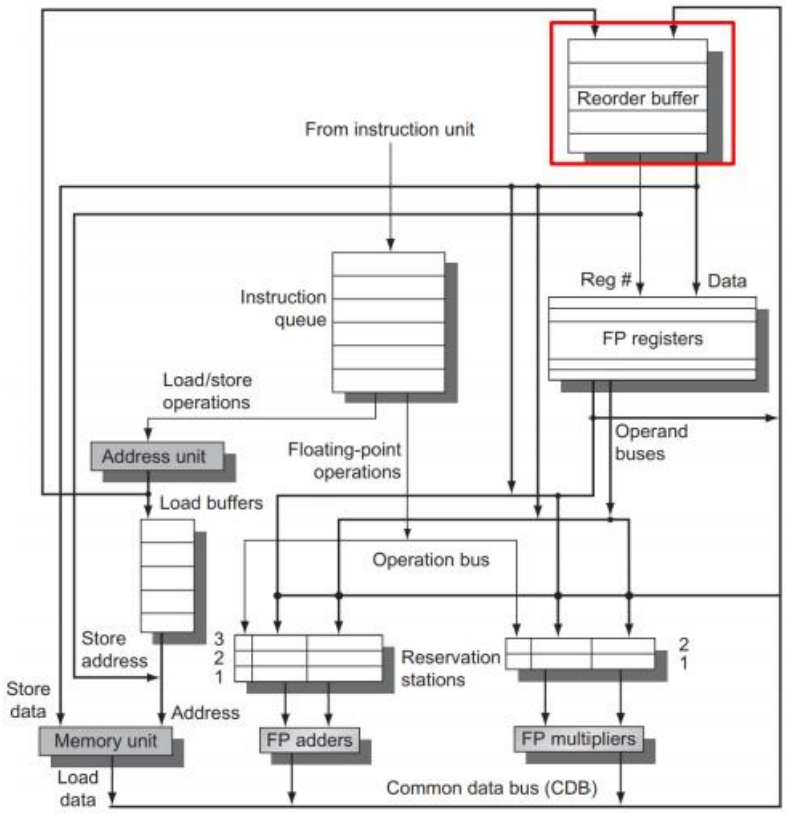

图 2. 仿真器架构

假设如下：

- 功能单元不是流水线式的。
- 功能单元之间没有转发；结果通过公共数据总线（CDB）进行通信。
- 执行阶段（EX）既执行有效地址计算，也执行加载和存储的内存访问。因此，管道是 IF/ID/IS/EX/WB。负载需要两个时钟周期。执行有效地址计算，加载和存储的内存访问各需要一个周期，共需要两个时钟周期。
- 指令发射（IS）和结果写回（WB）阶段各需要一个时钟周期。有三个加载缓冲区插槽和三个存储缓冲区插槽。
- 假设不等于零的分支（BNEZ）指令需要一个时钟周期。

FP 指令的部分执行延迟假设如下表所示。

FP instruction	EX Cycles
fadd	2
fsub	2
fmult	10
fdvi	20

注：报告中应给出上文未提及但在编程中使用的一些假设，以便于批改

3. 附加问题

（1） Tomasulo 算法相对于 Scoreboard 算法的优点？同时简述 Tomasulo 存在的缺点。

（2） 简要介绍引入重排序改进 Tomasulo 的原理。

（3） 请分析重排序缓存的缺点。

4. 任务提交

请您制作一份 pdf 文档，显示您的程序成功完成了“input1.txt”和“input2.txt”中指令下的动态调度，并回答第三节的附加问题。同时，您提交的文件还应提供原始代码和输出文件以供查阅。为了方便我们审阅，您需要适当的注释来提高代码的可读性。

5. 赋分标准

第一部分作业从以下几个方面进行评估：

报告（30 分）、输出正确性（20 分）、代码可读性（20 分）、附加问题（30 分）

附录 1

附输入输出格式规定：

输入格式：

一条指令一行并以 txt 文件格式读入，可能的指令输入格式如下图 3 所示。

```
LD F6 34+ R2
LD F2 45+ R3
MULTD 0 F2 F4
SUBD F8 F6 F2
DIVD F10 F0 F6
ADDD F6 F8 F2
```

图 3 指令输入格式

输出格式：

由每个周期状态和指令最终执行情况表两部分输出组成，最终输出到 ouput1.txt 或者 ouput2.txt 文件中：

1、 每个时钟周期状态输出格式要求：

- 每个 cycle 输出 15 行状态信息，其中 cycle 标识 1 行，reorder buffer 状态 6 行， 寄存器状态 7 行，功能单元状态 2 行。
- 请用“;” 间隔不同实体，用“:” 间隔实体标识符和实体属性，用“,” 间隔同一实体不同属性。
- 如属性为空或未知，请留空。
- 多个连续周期状态相同仅输出首个相同周期状态，状态标识修改为“cycle_n-m”（其中 n 为第一个周期相同，m 为最后一个相同周期）。

  综上，每个周期输出的状态格式应如图 4 所示：

```
//cycle_n;
//entry1 : (busy) ,(Instruction),(State),(Destination),(Value);
//entry2 : (busy) ,(Instruction),(State),(Destination),(Value);
//entry3 : (busy) ,(Instruction),(State),(Destination),(Value);
//entry4 : (busy) ,(Instruction),(State),(Destination),(Value);
//entry5 : (busy) ,(Instruction),(State),(Destination),(Value);
//entry6 : (busy) ,(Instruction),(State),(Destination),(Value);
//Load1 : (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Load2 : (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Add1 : (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Add2 : (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Add3 : (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Mult1: (busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Mult2:(busy) ,(op),(Vj),(Vk),(Qj),(Qk), (Dest);
//Reorder: f0:(status);F1:(status);.. .;F10: (status);
//Busy:f0:(status);F1:(status);.. .;F10: (status);
```

图 4 单个周期状态输出格式

以“Chap03-ILP-Part4-Speculation”PPT 中个可能的输出如下（仅供格式参考，具体内容应根据实际运行情况获得）：

Reorder Buffer

Reorder buffer

Entry	Busy	Instruction		State	Destination	Value
1	No	fld	f6,32(x2)	Commit	f6	Mem[32 + Regs[x2]]
2	No	fld	f2,44(x3)	Commit	f2	Mem[44 + Regs[x3]]
3	Yes	fmul.d	f0,f2,f4	Write result	f0	#2 × Regs[f4]
4	Yes	fsub.d	f8,f2,f6	Write result	f8	#2 – #1
5	Yes	fdiv.d	f0,f0,f6	Execute	f0	
6	Yes	fadd.d	f6,f8,f2	Write result	f6	#4 + #2

Reservation stations

Name	Busy	Op	Vj	Vk	Qj	Qk	Dest	A
Load1	No							
Load2	No							
Add1	No							
Add2	No							
Add3	No							
Mult1	No	fmul.d	Mem[44 + Regs[x3]]	Regs[f4]			#3	
Mult2	Yes	fdiv.d		Mem[32 + Regs[x2]]	#3		#5	

FP register status

Field	f0	f1	f2	f3	f4	f5	f6	f7	f8	f10
Reorder #	3						6		4	5
Busy	Yes	No	No	No	No	No	Yes	…	Yes	Yes

```
cycle_n;
entry1 : No ,fld f6 32(x2),Commit,f6,Mem[32+Regs[x2]];
entry2 : No ,fld f2 44(x3),Commit,f2,Mem[44+Regs[x3]];
entry3 : Yes,fmul.d f0,f2,f4,Write result,f0,#2 × Regs[f4];
entry4 : Yes,fsub.d f8,f2,f6,Write result,f8,#2-#1;
entry5 : Yes,fdiv.d f0,f0,f6,Execute,f0,;
entry6 : Yes,fadd.d f6,f8,f2,Write result,f6,#4+#2;
Load1 : No,,,,,;
Load2 : No,,,,,;
Add1 : No,,,,,;
Add2 : No,,,,,;
Add3 : No,,,,,;
Mult1: No,fmul.d,Mem[44+Regs[x3]],Regs[f4],,,#3;
Mult2: Yes,fdiv.d,,Mem[32+Regs[x2]],#3,,#5;
Reorder:f0: 3;F1:;.. .;F10: 5;
Busy:f0: Yes;F1:No;.. .;F10: Yes;
```

图 5 一个可能的周期状态及其对应输出

2、 最终指令最终执行情况表输出要求：仅在所有周期状态输出完后输出一次，一条指令对应一行，具体格式为："(Instruction):(Issue cycle),(Exec comp cycle),(Write result cycle),(Commit cycle)；"。