

《 计算机体系结构 》

实验报告

|  |  |
| --- | --- |
| 班 级： | **计213** |
| 学 号： | **21013134** |
| 姓 名： | **徐昊博** |
| 指导教师： | **梁建宁** |

信息科学与工程学院

2023年11月

**实验名称Tomasulo算法** **实验地点信息楼418实验日期11/29**

1. **实验目的**

（1）加深对指令级并行性及其开发的理解。

（2）加深对Tomasulo算法的理解。

（3）掌握Tomasulo算法在指令流出、执行、写回各阶段对浮点指令和Load/ Store指令进行的操作。

（4）掌握采用了Tomasulo算法的浮点处理部件的结构。

（5）掌握保留站的结构

（6）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。

1. **实验设备**

实验平台采用Tomasulo算法模拟器。

1. **实验原理**

1、指令集并行性及其开发

（1）由于指令可以并行执行，所以指令之间可能实现的重叠执行称为指令级并行（ILP）。

（2）基础块是一段顺序执行的代码，除了入口外没有其他的转入分支，除了出口外没有其他的转出分支。一个基础块一般包含3-6条指令，基础块类的指令一般互相依赖，挖掘并行性很受限制，所以一般需要在不同的基础块之间寻找指令并行性。

（3）指令级并行大体有两种不同的开发方法：一是依靠硬件来帮助动态发现和开发并行，二是依靠软件技术在编译时静态发现并行。

（4）挖掘指令级并行性的目的：最大化IPC，或者最小化CPI。

2、Tomasulo算法基本步骤

1、指令译码（Instruction Issue）：

将指令从指令流中取出并译码，确定指令的操作类型、源操作数和目标操作数等信息。

2、发射（Issue）：

将指令发送到可用的执行单元，并分配对应的功能单元和寄存器等资源。如果所需的资源可用，则指令被发射到执行单元中。

3、操作数读取（Operand Read）：

当所有指令所需的操作数都可用时，执行单元从寄存器或保留站（Reservation Station）中读取这些操作数。

4、执行（Execution）：

执行单元执行指令操作，可能需要多个时钟周期。

结果写回（Write Result）：

当指令完成执行时，将结果写回寄存器文件（Register File）或更新保留站中的结果。

1. **实验操作及运行结果**

（1）假设浮点功能部件的延迟时间为：加减法2个时钟周期，乘法10个时钟周期，除法40个时钟周期，Load部件2个时钟周期。

① 对于下面的代码段，给出当指令MUL.D即将写回时，保留站、Load缓冲器以及寄存器状态表中的内容。

L.D F6, 24(R2)

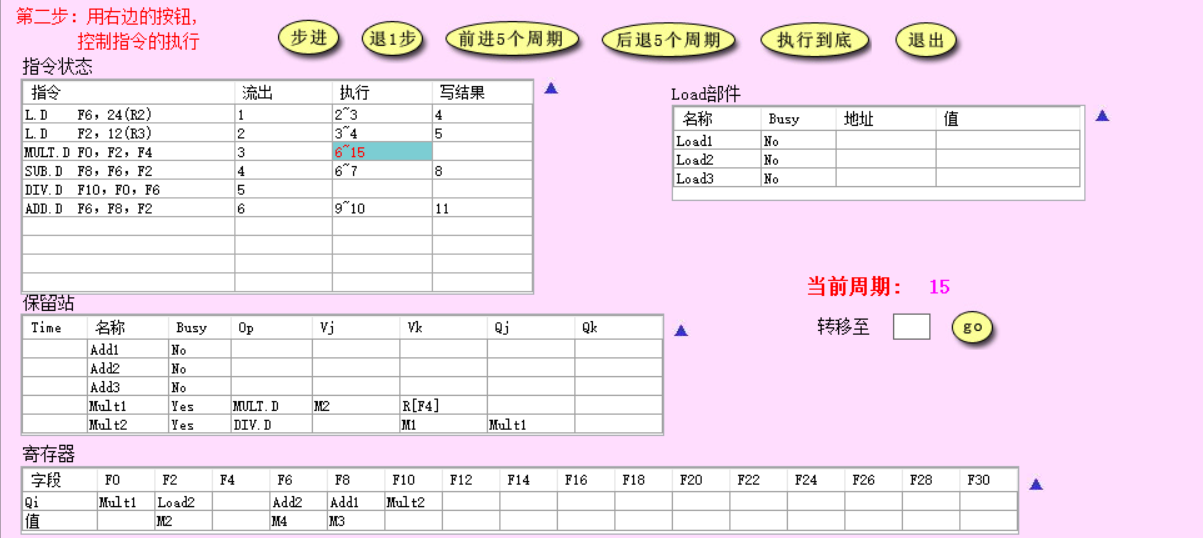
L.D F2, 12(R3)

MUL.D F0, F2, F4

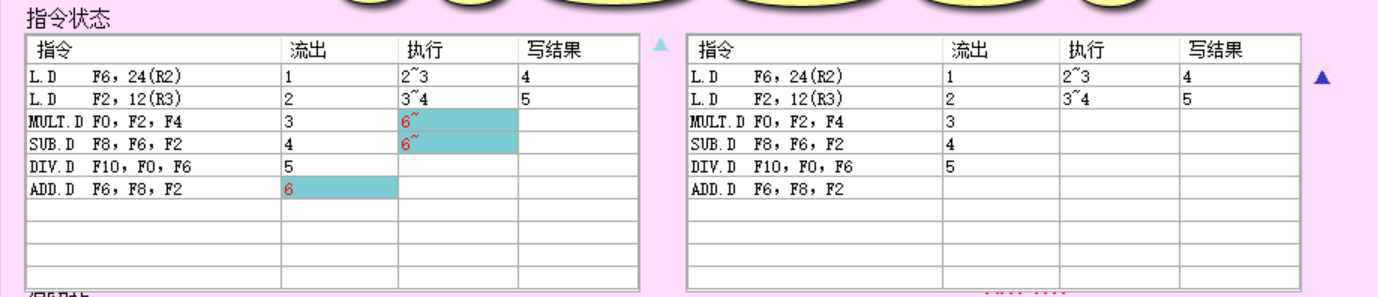
SUB.D F8, F6, F2

DIV.D F10, F0, F6

ADD.D F6, F8, F2



②按步进方式执行上述代码，利用模拟器的“小三角按钮”的对比显示功能，观察每一个时钟周期前后各信息表中内容的变化情况。



（2）对于与上面相同的延迟时间和代码段：

①给出在第3个时钟周期时保留站的内容。



②步进5个时钟周期，给出这时保留站、Load缓冲器以及寄存器状态表中的内容。



③再步进10个时钟周期，给出这时保留站、Load缓冲器以及寄存器状态表中的内容。



1. 假设浮点功能部件的延迟为：加减法3个时钟周期，乘法8个时钟周期，除法40个时钟周期。自己编写一段程序（要在实验报告中给出），重复上述步骤（2）的工作。

编写的代码如下：

L.D F6, 24(R2)

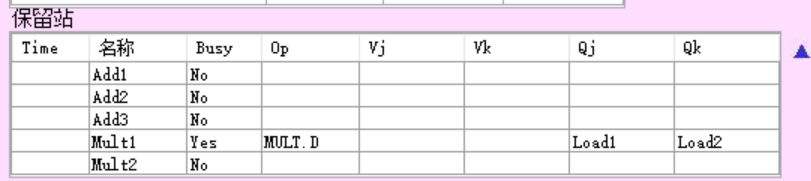
L.D F2, 12(R3)

MUL.D F0, F6, F2

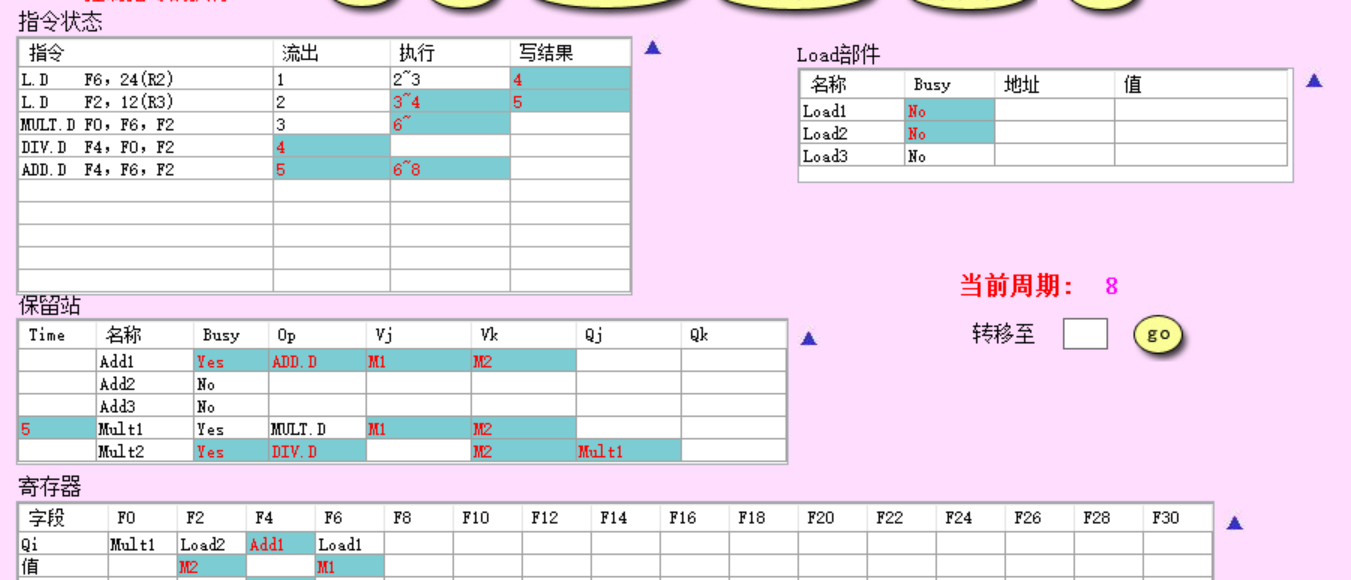
DIV.D F4, F0, F2

ADD.D F4, F6, F2

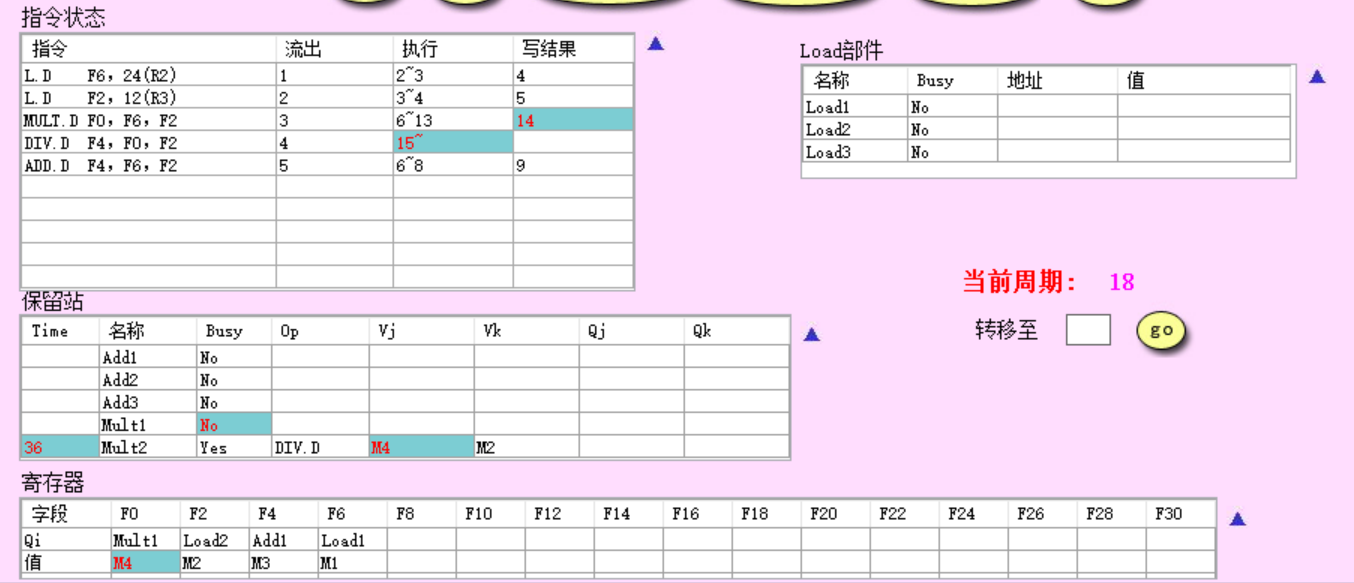
①给出在第3个时钟周期时保留站的内容。



②步进5个时钟周期，给出这时保留站、Load缓冲器以及寄存器状态表中的内容。



③再步进10个时钟周期，给出这时保留站、Load缓冲器以及寄存器状态表中的内容。

**五、实验中出现的问题和解决方法**

问题：最后一个实验需要修改相关操作的时钟数，但缺少相关代码对修改时钟数后相关周期变化进行实验。

解决方案：在设计相关代码时尽量做到全部覆盖即可。