## 实验二 Tomasulo算法分析

### 一、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。

（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。

### 二、实验内容

（1）首先要掌握Tomasulo算法模拟器的使用方法。

（2）写出实验过程中保留站、指令状态表以及浮点寄存器状态表内容的变化情况。

### 三、实验原理及方案

#### 1、Tomasulo算法模拟器的使用方法

**（1）设置指令和参数**

本模拟器最多可以模拟10条指令。可以在“指令”区选择和设置所要的指令。“指令”区如图3.1所示。

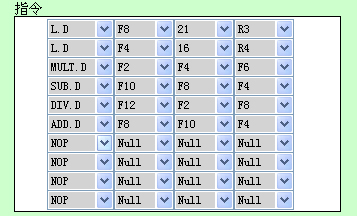


图3.1 “指令”区

你可以从下拉框中选择指令，供选择的指令有以下5种：

1）L.D指令：从主存读取一个双精度浮点数；

2）ADD.D：双精度浮点加法指令；

3）SUB.D：双精度浮点减法指令；

4）MULT.D：双精度浮点乘法指令；

5）DIV.D：双精度浮点除法指令。

指令的各参数也可以从各自的下拉框中选择。

你还可以在窗口的右上区域设置各部件的执行时间（时钟周期数），如图3.2所示。

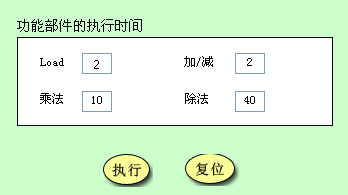


图3.2 设置功能部件时间

其中“复位”的作用是使所有设置恢复为默认值。

**（2）执行**

点击“执行”按钮，就进入执行状态。你可以用中间的按钮来控制指令的执行，包括“步进”、“退1步”、“前进5个周期”、“后退5个周期”、“执行到底”、“退出”等。还可以用“go”按钮直接跳转到你所指定的时钟周期。如果想修改被执行的代码，按“退出”按钮，即可回到设置指令和参数页面。

向前执行后，状态表中抹色的字段表示其内容发生了变化。

**（3）对比状态表**

每一个状态表的右上角外侧都有一个小三角，用鼠标左键点击它，会弹出该表在上一个时钟周期的内容。这是为了让你通过对比来了解哪些内容发生了变化。在弹出表以外的区域再次点击鼠标，就可以将其收回。

**（4）各个表的内容**

1）指令状态表

指令状态表如图3.3所示。它列出了各指令什么时候执行到了哪一步。其中的数字表示时钟周期，“~”表示时钟周期期间。例如，图3.3中的2~3表示在第2到第3个时钟周期（含第3个），第一条L.D指令是在“执行”这一步。



图3.3 指令状态表

其中抹色的区域表示最近一个时钟周期其内容发生了变化。下同。

2）保留站

保留站的内容如图3.4所示。

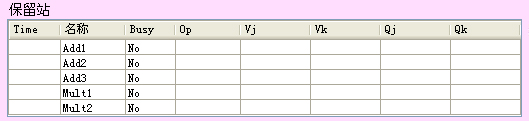


图3.4 保留站

其中各字段的名称和含义如下：

Time：表示相应的保留站还有要执行多少个时钟周期；

名称：保留站的名称。用于唯一地标识相应的保留站；

Op：要对源操作数进行的操作；

Qj，Qk：将产生源操作数的保留站名称。其值等于0表示操作数已经就绪且在Vj或Vk中，或者不需要操作数。

Vj，Vk：源操作数的值。对于每一个操作数来说，V和Q字段只有一个有效。

Busy：为“Yes”表示该保留站 “忙”。

3）Load部件

Load部件的内容如图3.5所示。它按队列方式工作，每次处理新的访存都是从队列头部取走一条。



图3.5 Load部件

其中各字段的含义如下：

名称：相应单元的名称（标识）；

Busy：“忙”标志，为“Yes”表示已被占用；

地址：访存的有效地址；

值：存放从存储器读来的数据。

4）寄存器

寄存器的内容如图3.6所示。

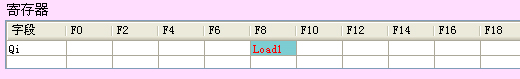


图3.6 寄存器的内容

各字段的含义如下：

Qi：寄存器状态，用于存放将把结果写入该寄存器的保留站的站号。为0表示当前没有正在执行的指令要写入该寄存器，也即该寄存器中的内容就绪。

值：寄存器的值。

当上述表中的内容写不下时，模拟器会采用缩写的方法。这时，在上面中间的区域中会显示缩写及其值。

#### 2、CASE 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即将确认时，保留站、load缓冲器以及寄存器状态表中的内容。

L.D F6, 24（R2）

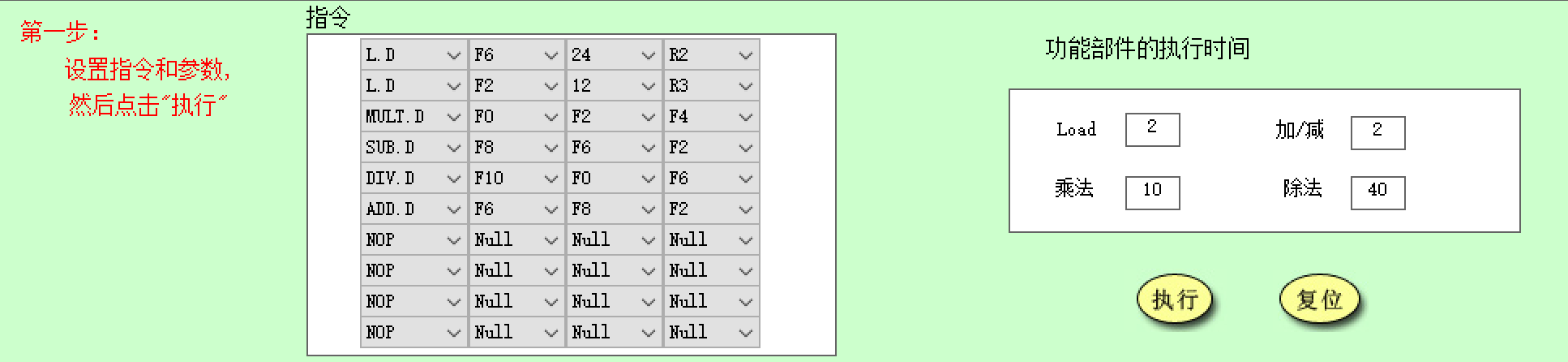
L.D F2, 12 （R3）

MUL.D F0, F2, F4

SUB.D F8, F6, F2

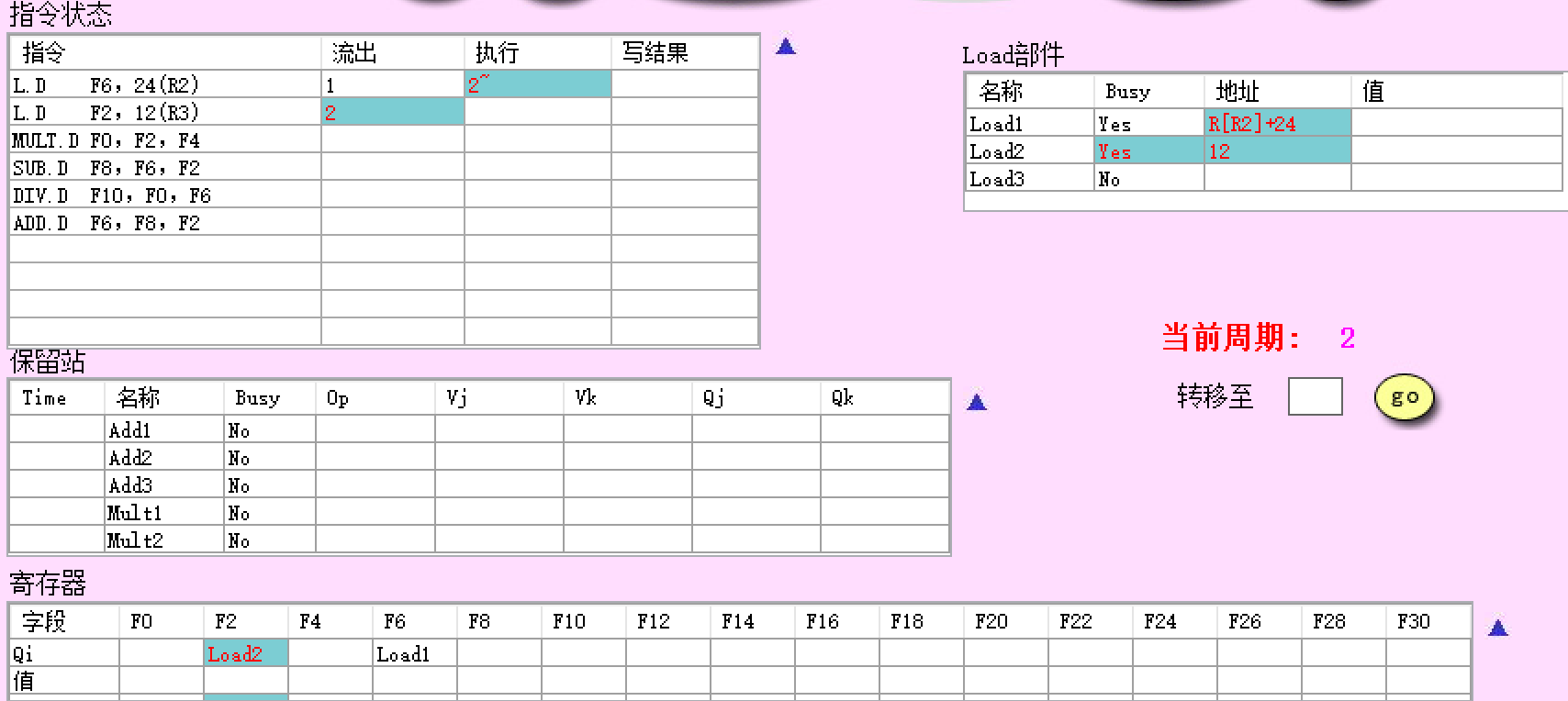
DIV.D F10, F0, F6

ADD.D F6, F8, F2

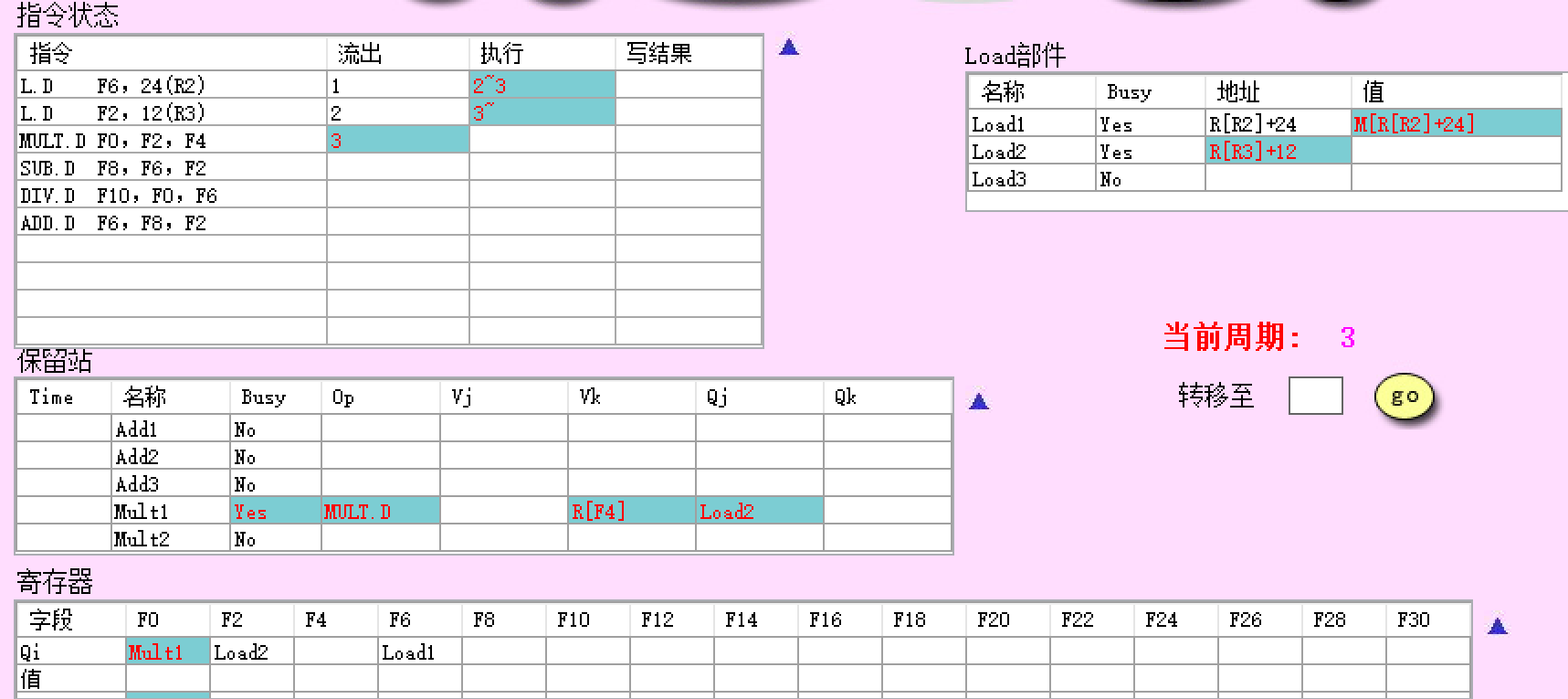




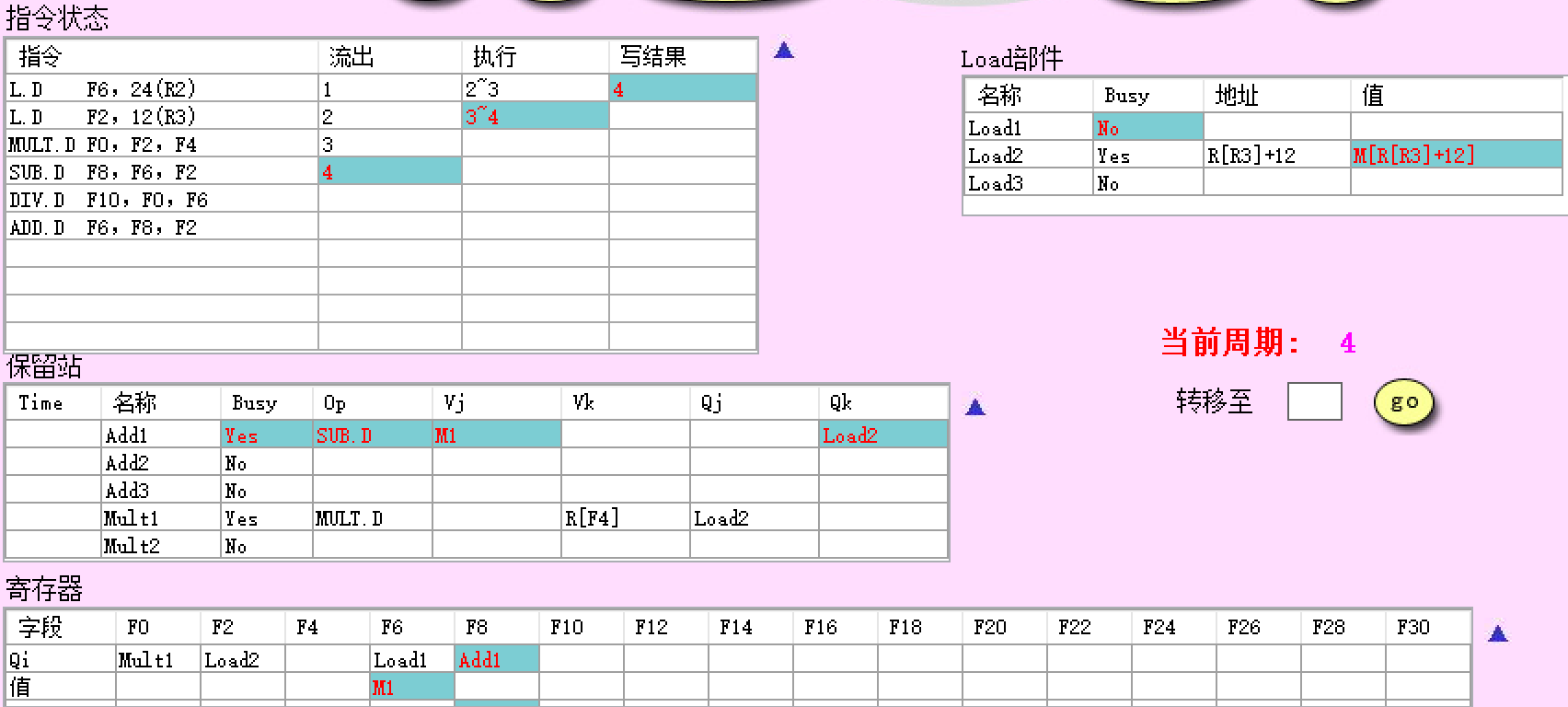
周期1：取出第一条指令L.D F6, 24(R2)，地址偏移量24写入LOAD1，LOAD1名存入寄F6。



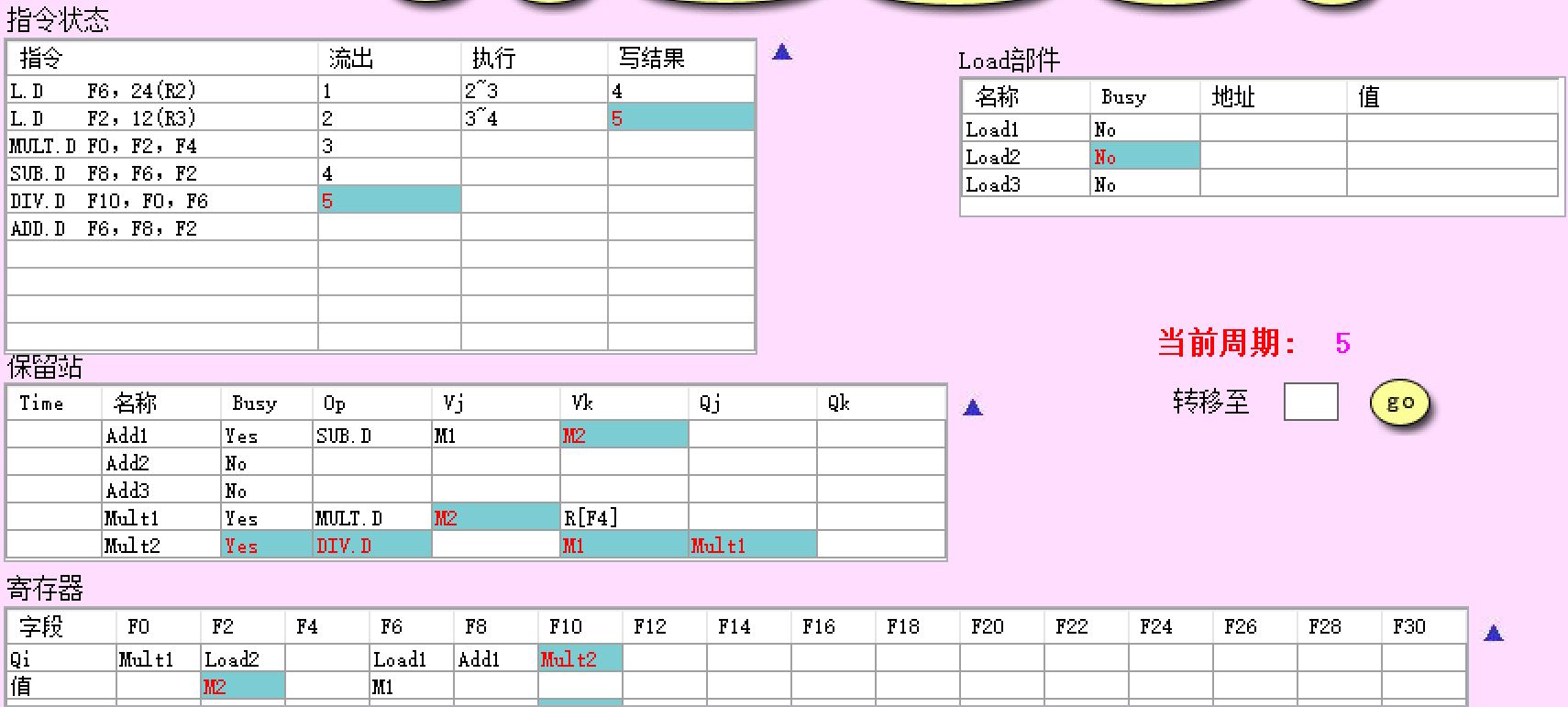
周期2：取出第二条指令L.D F2, 12(R3)，地址偏移量12写入LOAD2，LOAD2名存入寄器F2，同时第一条指令开始执行，LOAD1上写入绝对地址。



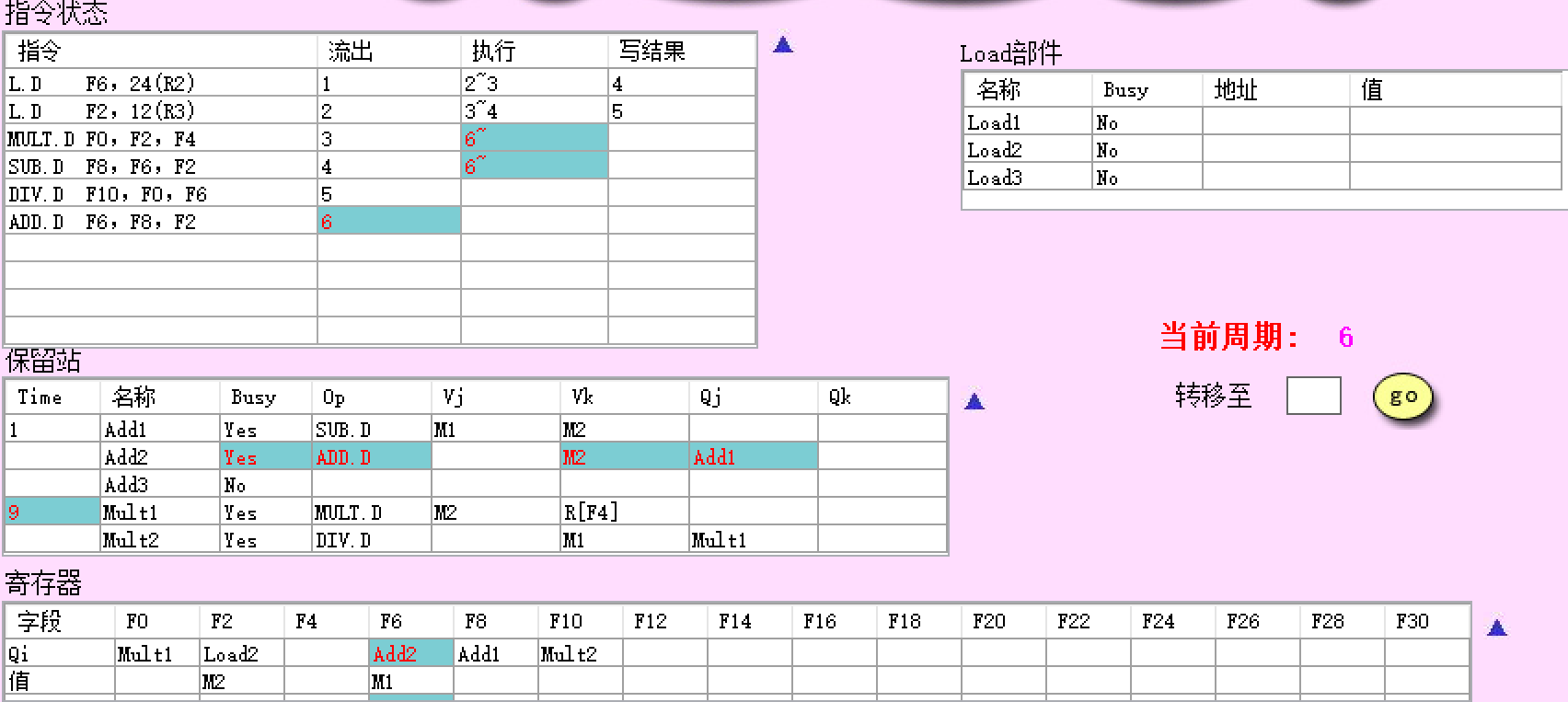
周期3：取出第三条指令MUL.D F0, F2,F4，第一条指令完成，第二条指令开始执行,LOAD2上写入绝对地址。保留站中存入待运算的操作数和操作。寄存器F0上QI写入保留站中待运算命令的名称。



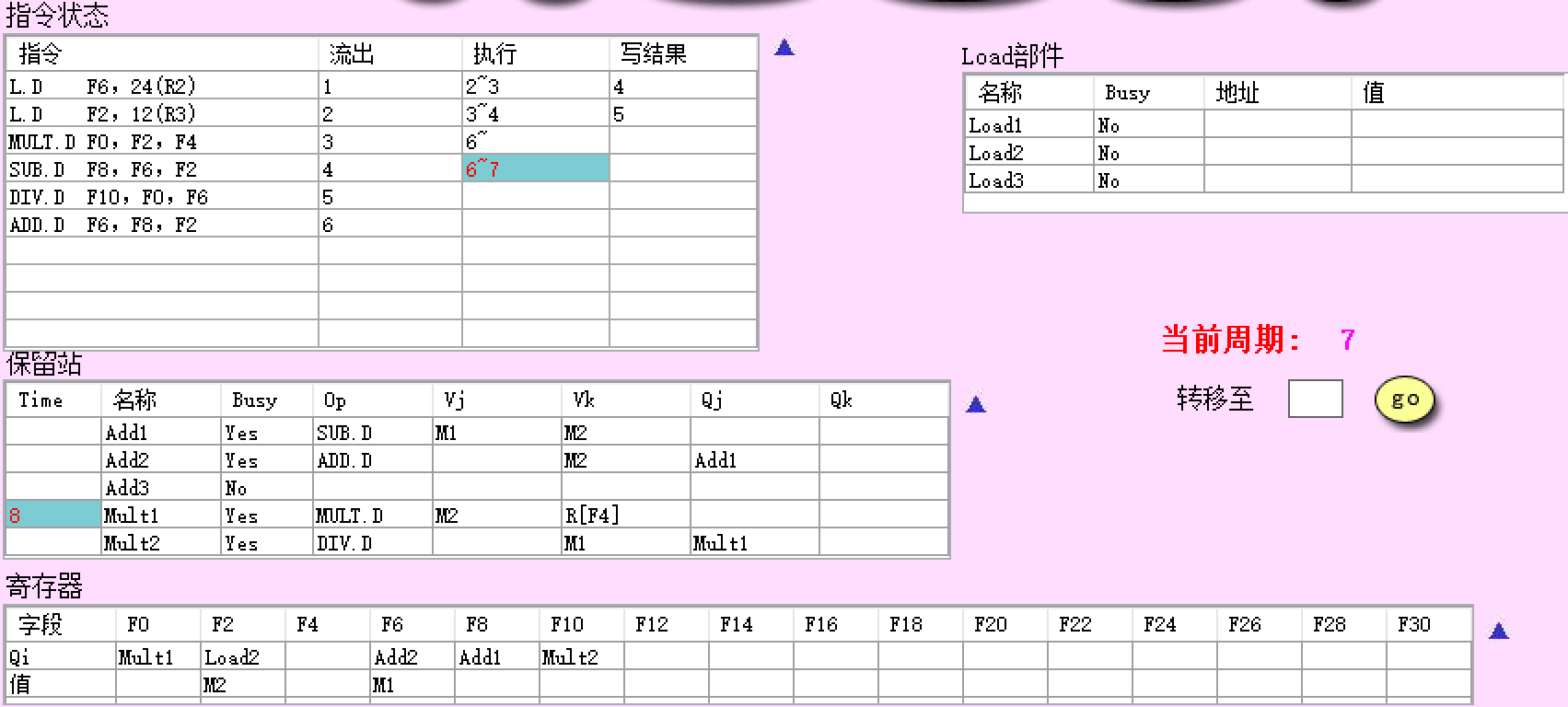
周期4：取出第四条指令SUB.D F8,F6,F2，第二条指令执行完成，第一条指令写入结果M1到寄存器F6，保留站中存入第四条指令的待运算操作数和操作。LOAD1清空。



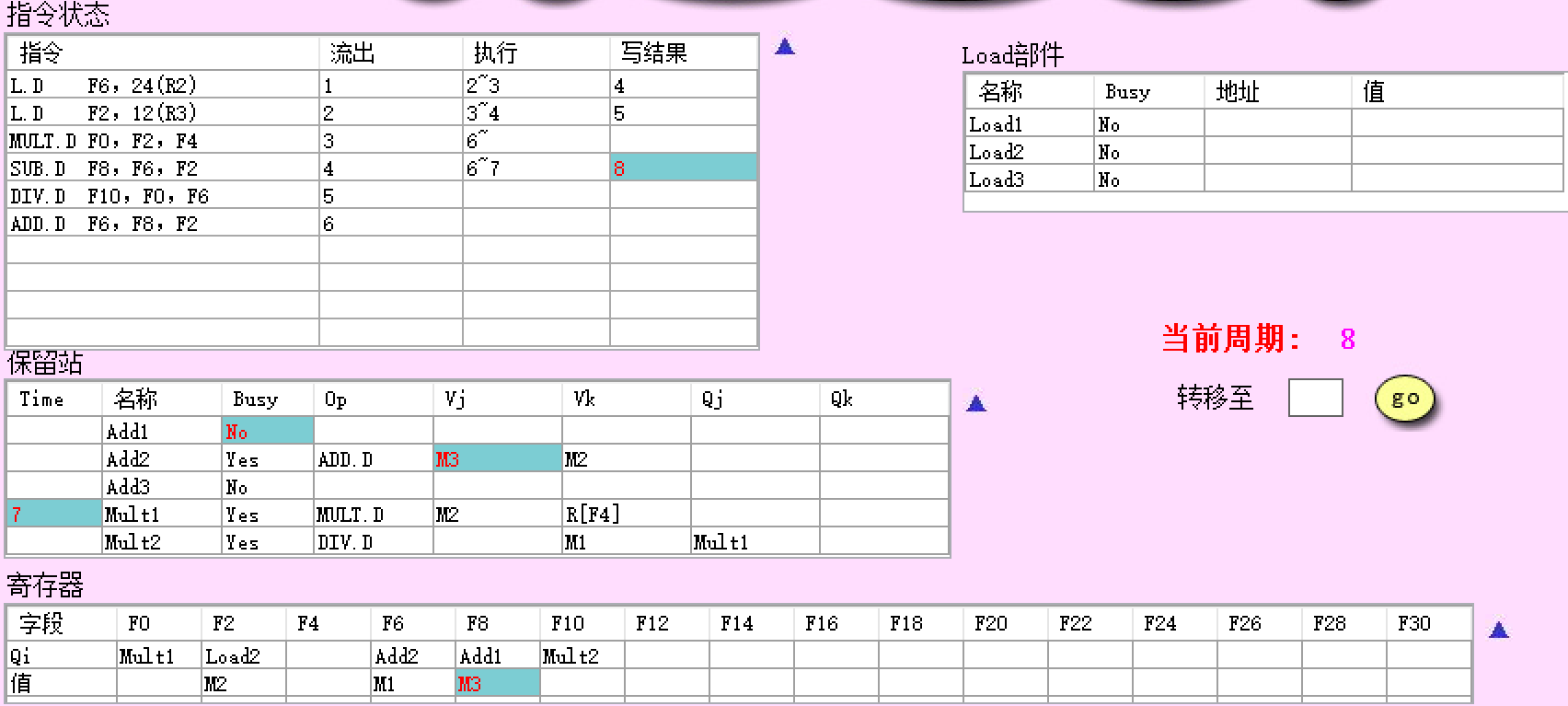
周期5：取出第五条指令DIV.D F10,F0,F6，第二条指令写结果M2到寄存器F2，LOAD2 清空。保留站中存入第五条指令的待运算操作数和操作



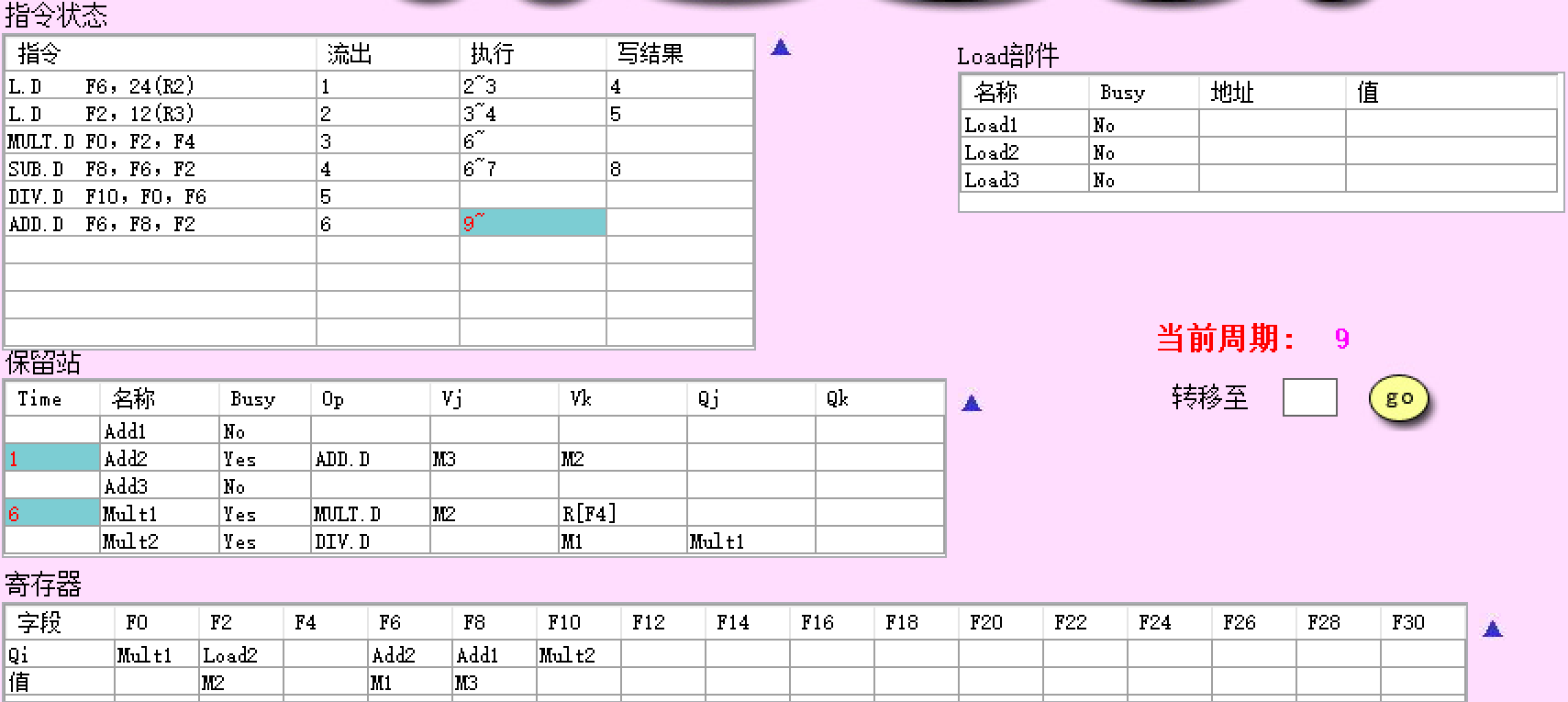
周期6：取出第六条指令ADD.D F6,F8,F2，第三条和第四条指令开始执行，相关的操作数和操作符被存入保留站



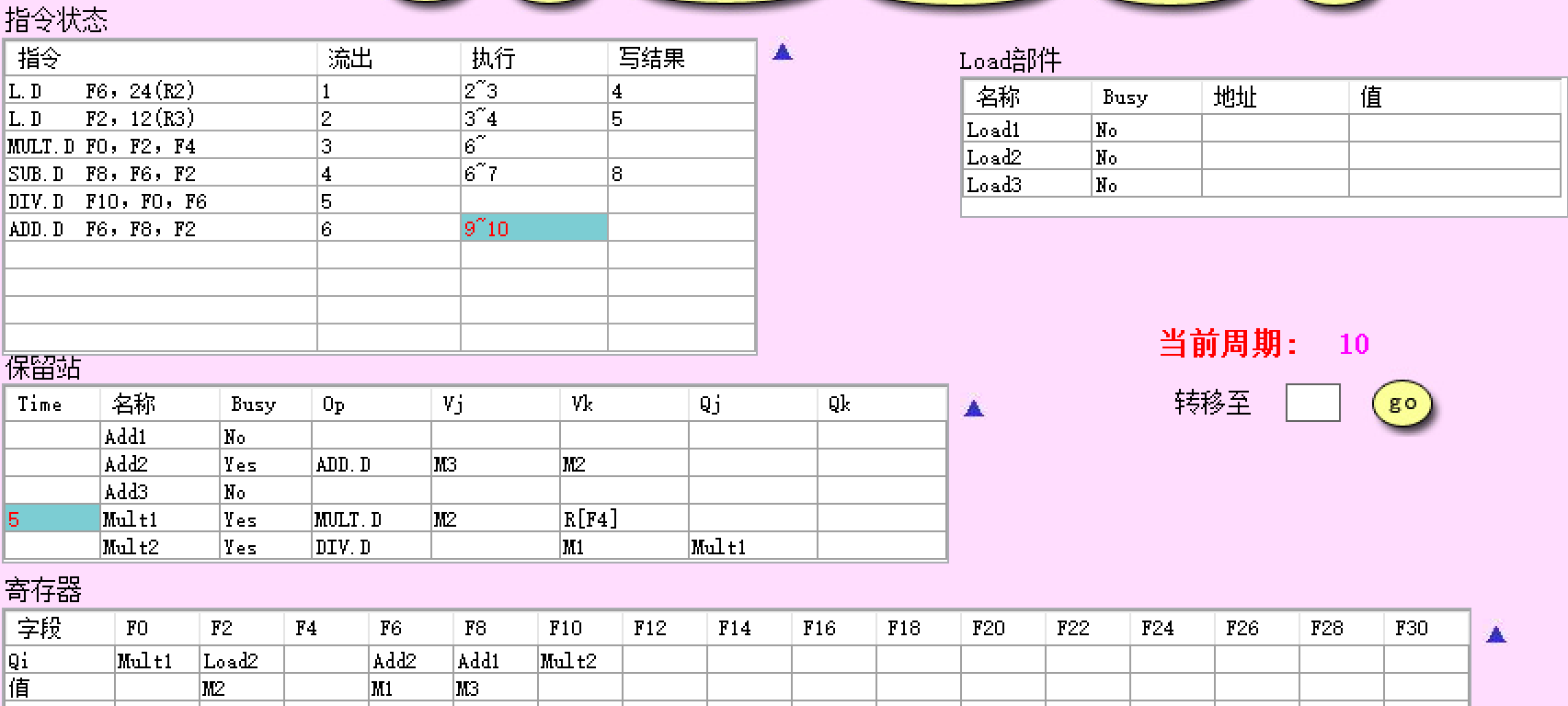
周期7：第四条指令执行完成，保留站中的第三条指令继续执行。



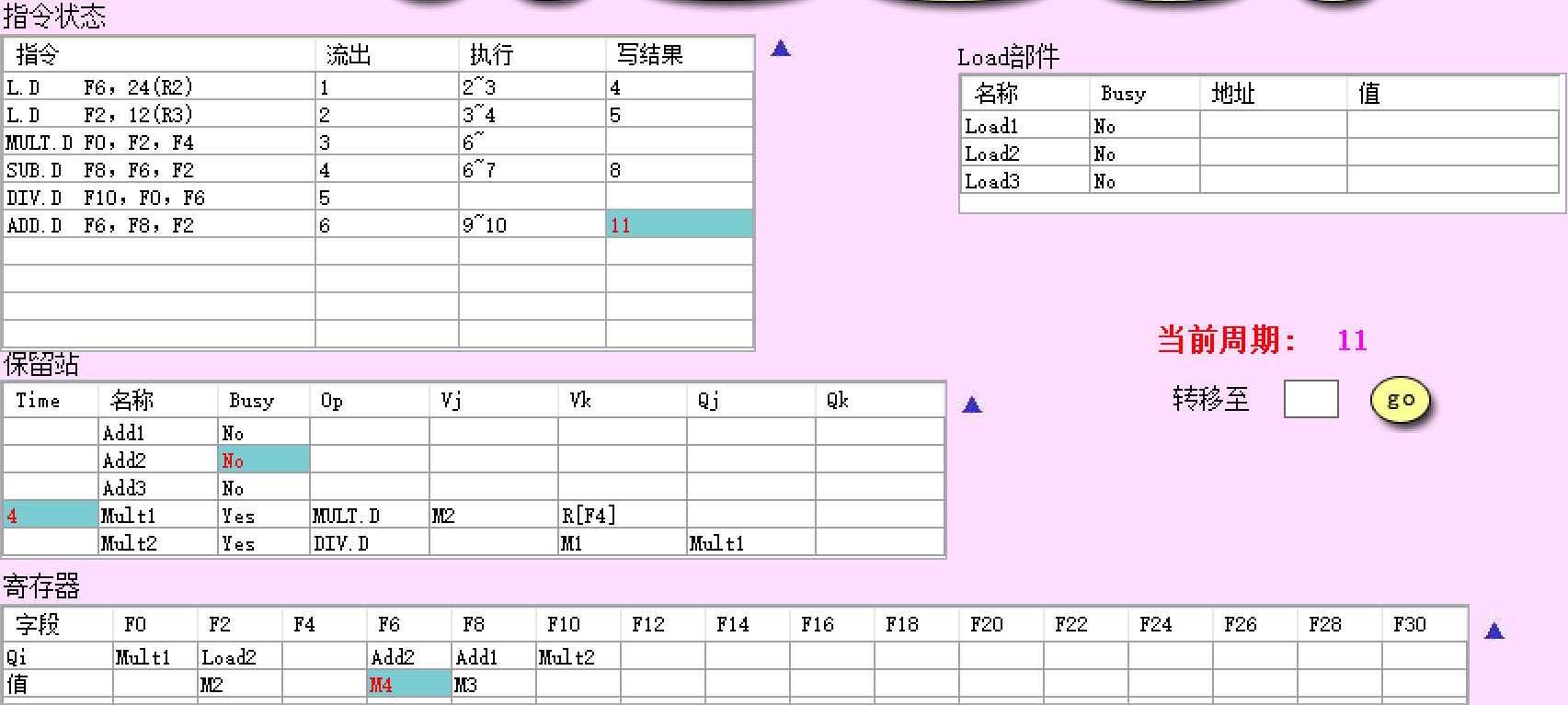
周期8：第四条指令写结果M3到寄存器F8，保留站中存放第四条指令的位置清空，第三条指令继续执行。



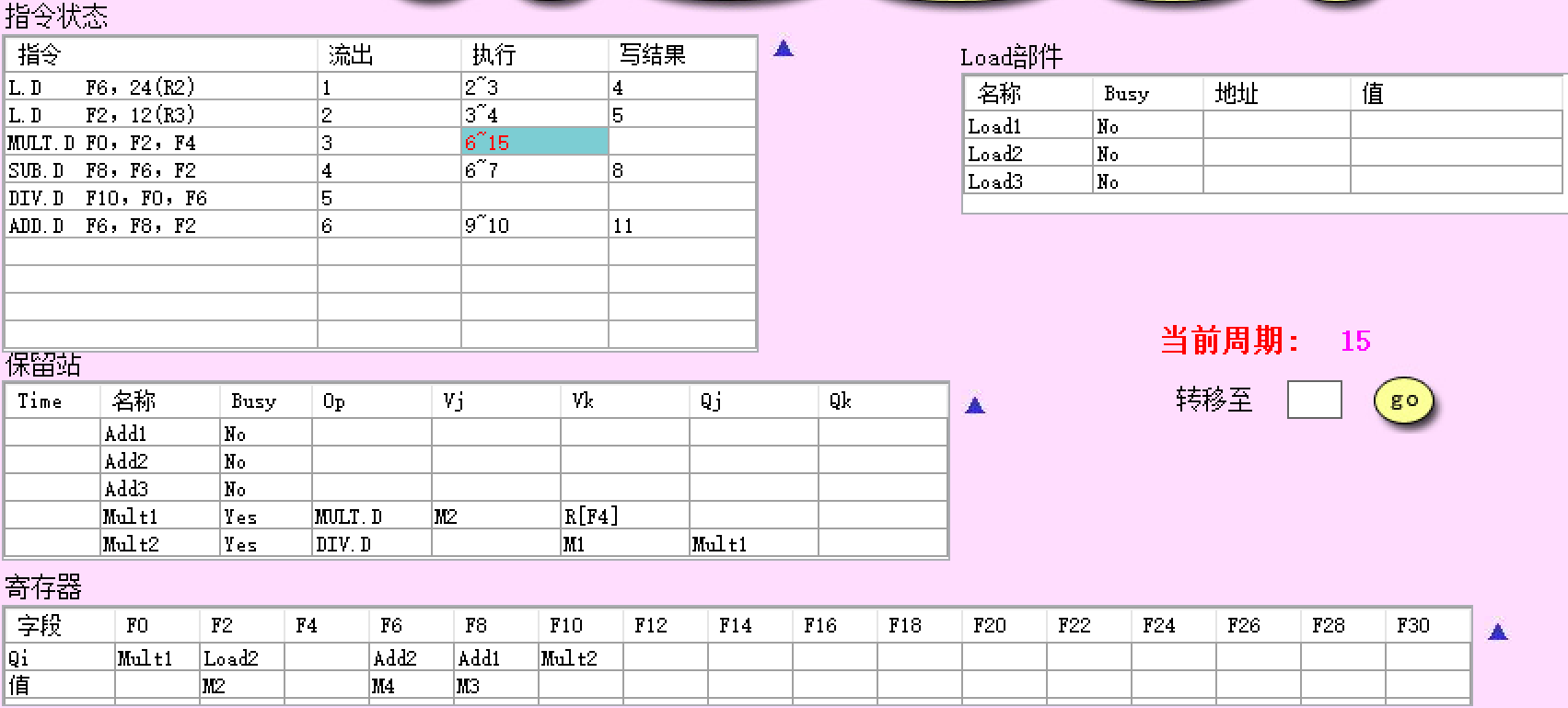
周期9：第六条指令开始执行。第三条指令继续执行。



周期10：第六条指令执行完成，第三条指令继续执行。



周期11：第六条指令写结果M4到寄存器F6中，清空保留站中原来存放第六条指令的位置。第三条指令继续执行。



周期12~15：第三条指令继续执行，直到完成。



周期16：第三条指令写结果M5到寄存器F0，保留站中原来存放第三条指令的位置清空。





周期17~56：第五条指令开始执行，直到结束



周期57：第五条指令执行结束后写结果M6到F10,保留站中原来存放第五条指令的位置清空。执行完毕。

（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。

#### 3、CASE 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。

#### 4、CASE 3

假设浮点功能部件的延迟时间为加减法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。

### 四、实验要求

（1）做好实验预习。了解Tomasulo算法及相关指令，了解采用Tomasulo算法的浮点处理部件结构和保留站的结构。

（2）根据实验原理及方案中的内容，完成实验。根据实验要求记录信息。

（3）撰写实验报告。

————————————————————————————————————

为什么Tomasulo算法消除了WAR和WAW冒险。

Tomasulo的FU称做保留站，保留站可以完成寄存器重命名，避免WAR、WAW。它采用寄存器重命名的方法来消除寄存器数据流之间的假相关，即用虚拟寄存器集代替真实的FPR，虚拟寄存器集在tomasulo算法中由每一功能单元所带的保留站、取数缓冲区和存数缓冲区组成。保留站可以多于寄存器，实现更多编译器无法完成的优化。