山东大学 计算机科学与技术 学院

计算机体系结构 课程实验报告

学号：202222130195	姓名： 阎发祥	班级： 22.3
实验题目：用 WinDLX 模拟器执行程序求最大公约数		
实验学时：2	实验日期：2025.4.23	

实验目的：

（1）通过本实验，熟练掌握 WinDLX 模拟器的操作和使用，清楚 WinDLX 五段流水线在执行 具体程序时的流水情况。

（2）熟悉 DLX 指令集结构及其特点。

硬件环境：

WinDLX(一个基于 Windows 的 DLX 模拟器)

软件环境：

VMware Workstation 16 Player

Windows 7

实验步骤与内容：

实验内容：

(1) 用 WinDLX 模拟器执行程序 gcm.s。该程序从标准输入读入两个整数，求他们的 greatest common measure，然后将结果写到标准输出。该程序中调用了 input.s 中的输入子程序。

(2) 给出两组数 6、3 和 6、1，分别在 main+0x8(add r2,r1,r0)、gcm.loop(seg r3,r1,r2) 和 result+0xc(trap 0x0)设断点，采用单步和连续混合执行的方法完成程序，注意中间过程 和寄存器的变化情况，然后单击主菜单 execute/display dlx-i/0,观察结果。

实验步骤：

1.汇编代码分析

本实验要求两个数的最大公约数，先从汇编代码的角度分析程序的运行过程，如下所示：

在汇编代码中首先是对一些常量进行了定义，例如 Prompt1, Prompt2 等。

```
        .data

        ;*** Prompts for input
Prompt1:    .asciiz     "First Number:"
Prompt2:    .asciiz     "Second Number: "

        ;*** Data for printf-Trap
PrintfFormat:   .asciiz     "gcM=%d\n\n"
        .align      2
PrintfPar:      .word       PrintfFormat
PrintfValue:    .space      4
```

之后是 main 函数对应的汇编代码。在本次实验中 main 函数的主要作用是提示用户输入数据，计算最大公约数以及向显示器输出信息，具体代码如下图所示：

```
        .text
        .global main
main:
        ;*** Read two positive integer numbers into R1 and R2
        addi    r1,r0,Prompt1
        jal     InputUnsigned   ;read uns.-integer into R1
        add     r2,r1,r0        ;R2 <- R1
        addi    r1,r0,Prompt2
        jal     InputUnsigned   ;read uns.-integer into R1
```

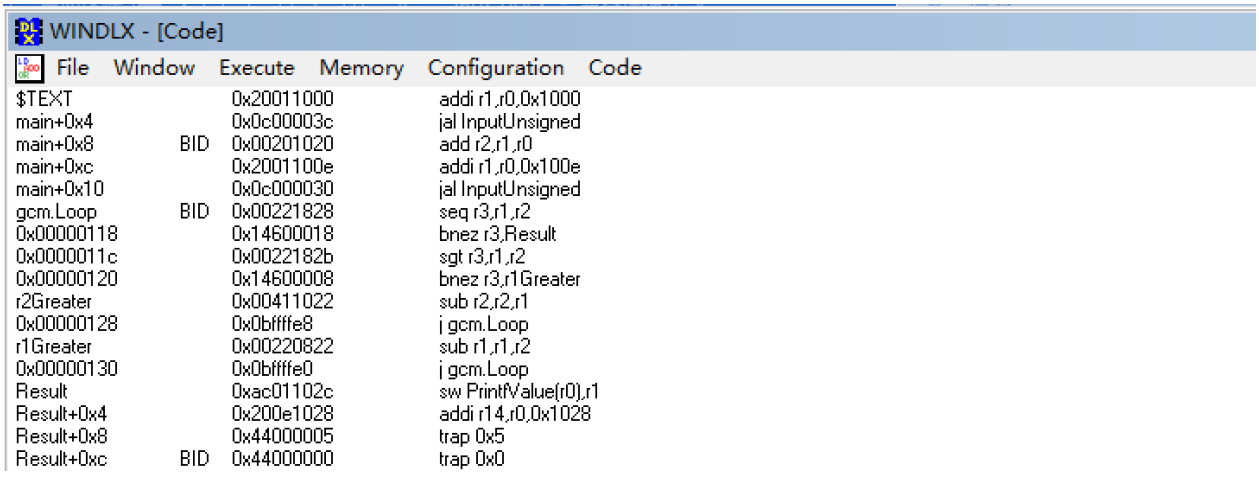
通过上面代码可以看到，在 main 函数中调用了 input.s 文件中的 InputUnsigned 函数，实现了数据的读入。当读入两个数之后，通过使用 Loop 和比较大小来求解两个数的最大公约数。

```
Loop:                           ;*** Compare R1 and R2
        seq     r3,r1,r2        ;R1 == R2 ?
        bnez    r3,Result
        sgt     r3,r1,r2        ;R1 > R2 ?
        bnez    r3,r1Greater

r2Greater:  ;*** subtract r1 from r2
        sub     r2,r2,r1
        j       Loop

r1Greater:  ;*** subtract r2 from r1
        sub     r1,r1,r2
        j       Loop
```

最后计算出最大公约数后，需要将结果输出在显示器上，通过 trap 执行系统调用实现

```
Result:     ;*** Write the result (R1)
        sw      PrintfValue,r1
        addi    r14,r0,PrintfPar
        trap    5

        ;*** end
        trap    0
```

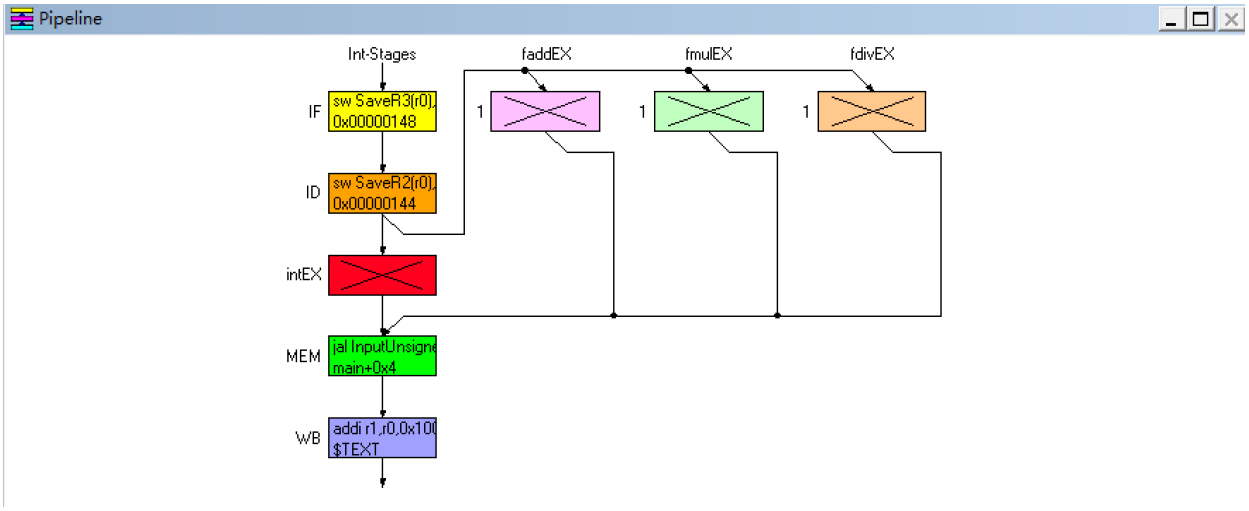
2. 装入程序单步执行



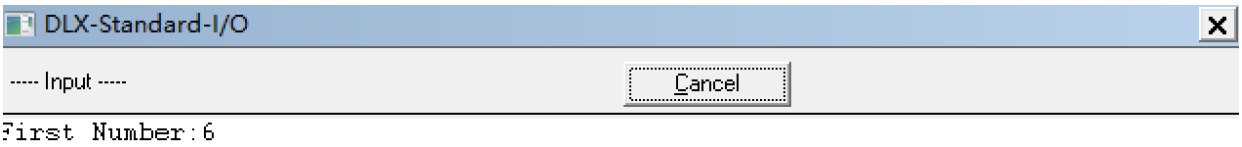
在运行程序之前首先在 main+0x8(add r2，r1，r0)、gcm.loop(seg r3,r1,r2)和result+0xc (trap 0x0)设置断点，之后单步执行指令，同时观察各个部件之间的变化

此时，执行过程中从 jal InputUnsigned 跳转到 0x00000144 的位置，说明正在调用 input.s 中的函数读入数据。

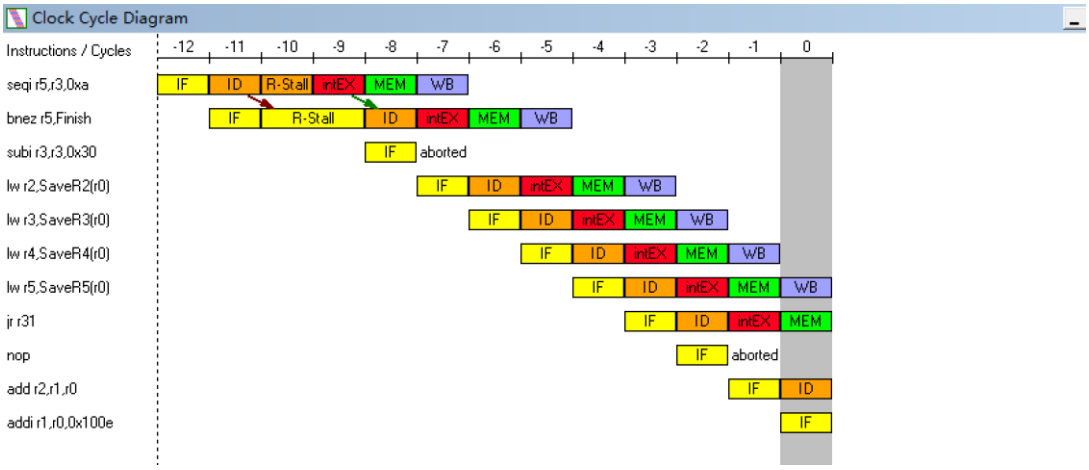
观察到 Pipeline 中 DLX 的五段流水线如下所示：



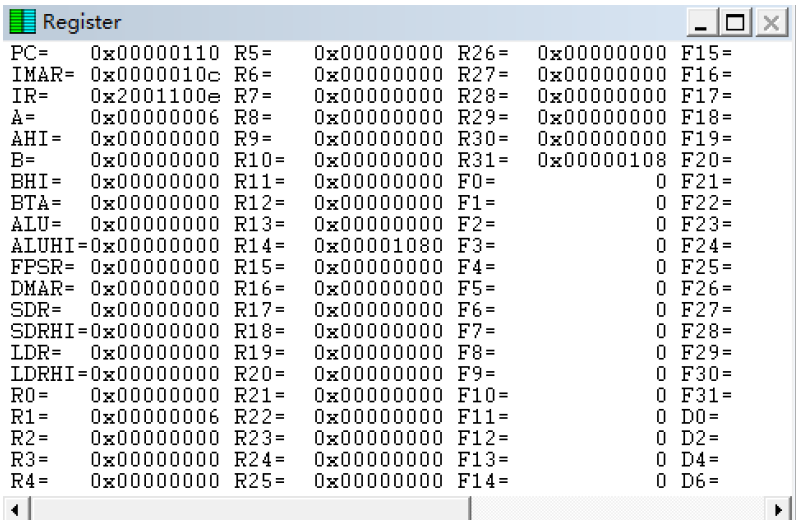
继续单步执行，直到提示输出 First number，此处我们输入 6。



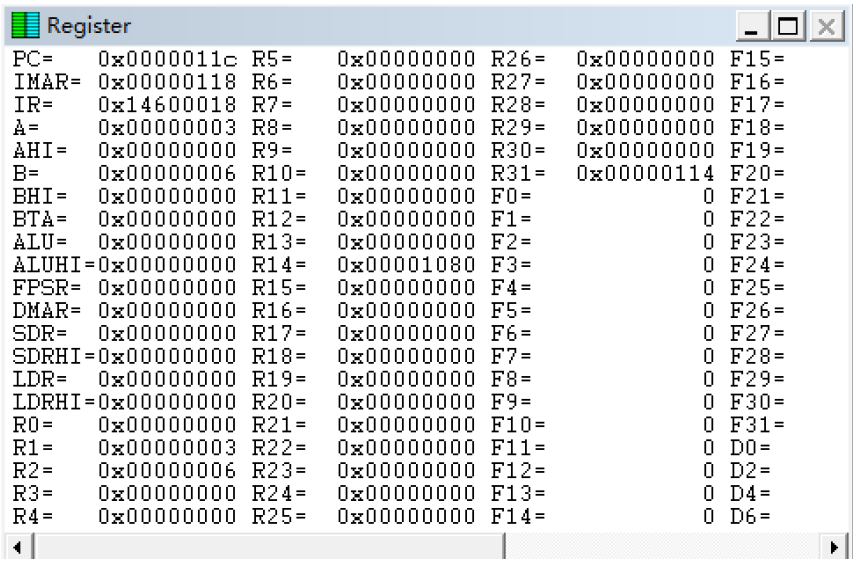
此时查看 Clock Cycle Diagram 模块，发现出现了红和绿两种颜色的箭头，红色箭头是因为指令之间发生了冲突，需要一个暂停，箭头指向处显示了暂停的原因，R-Stall（R-暂停）表示引起暂停的原因是 RAW。此处是因为第二条指令 lbu r3，0x0[r2]和第三条指令 seqi r5，r3，0xa，前者在 EX 执行阶段时要向 r3 中写入数据，而后者要在 ID 阶段取 r3 寄存器中的值作为源操作数，此时发生数据相关。而绿色箭头表示的是定向技术的使用，即在第二条指令访存后，立刻将 r3 寄存器中的值送入到第三条指令 ID 阶段，减少等待时间。



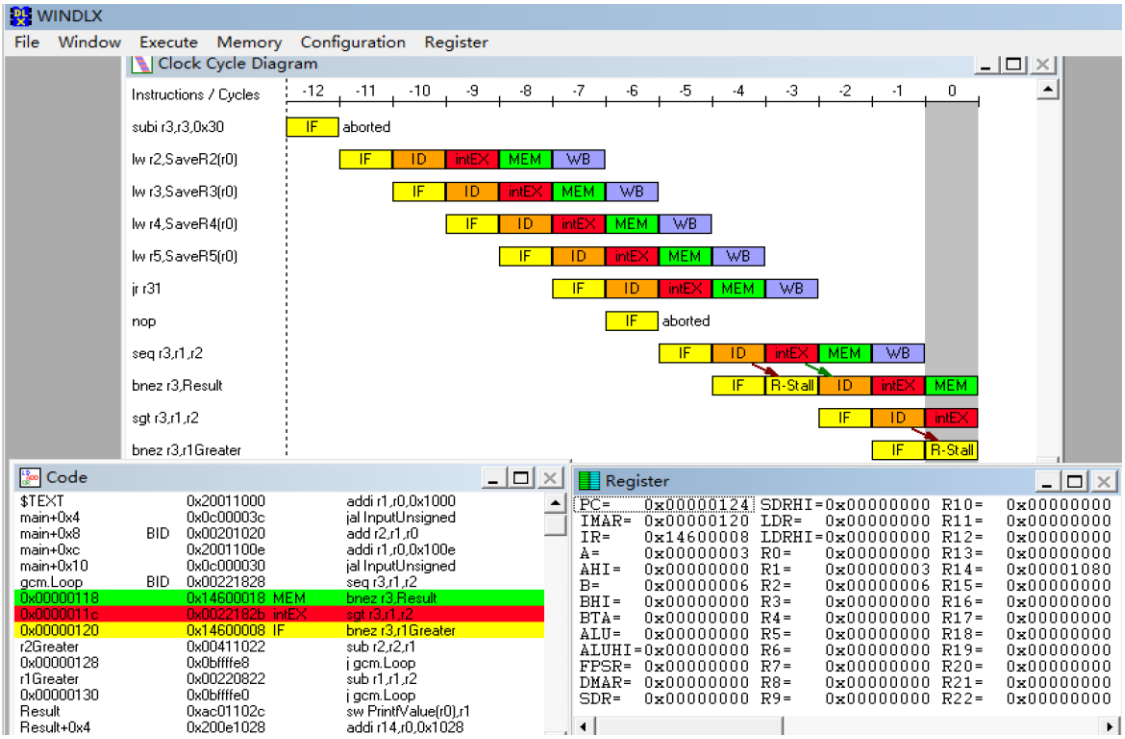
之后查看 Register 模块，可以发现 R1 寄存器中值变为 6，即我们刚才输入的值。



继续按下 F5，输入第二个输 3，此时输入完成。可以看到的是第一个数被保存在 R2，第二个数被保存在 R1， 准确来说调用 InputUnsigned 函数的返回值会被保存到 R1 寄存器中。



接下来进入 Loop 循环中求解最大公约数，执行过程中流水线和 PC 值变化如下：



此处具体分析最大公约数求解过程：

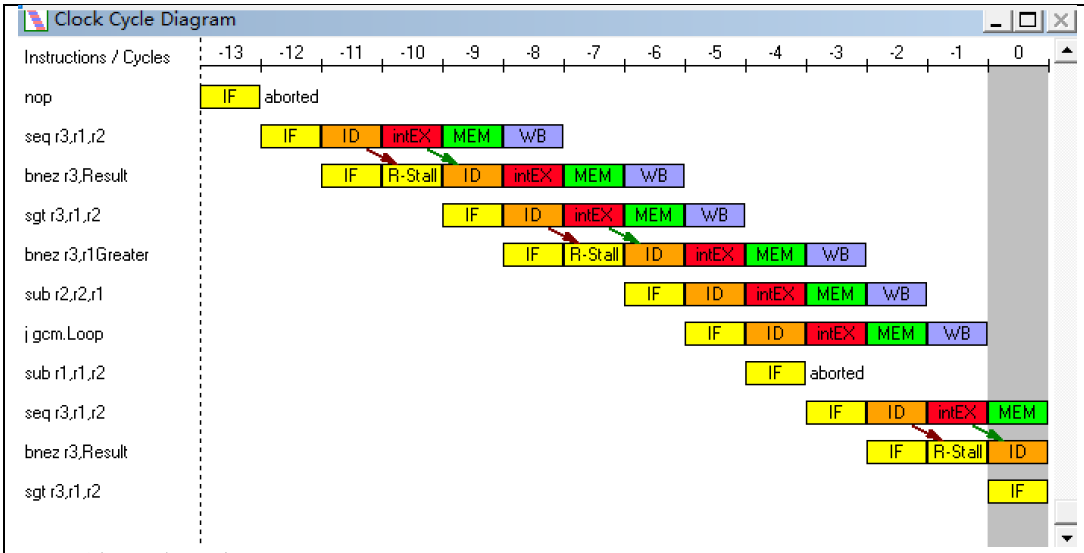
（1）Seq r3，r1，r2：这是比较寄存器 r1 和寄存器 r2 的值，如果相等的话，直接跳转到 Result 处输出结果，这里我们 r2=6，r1=3，并不相等，向下继续执行。

（2）Sgt r3，r1，r2：比较寄存器 r1 和寄存器 r2 的值，如果 r1 大于 r2 的话，就跳转到 r1Greater，此处 r1 小于 r2，继续向下执行。

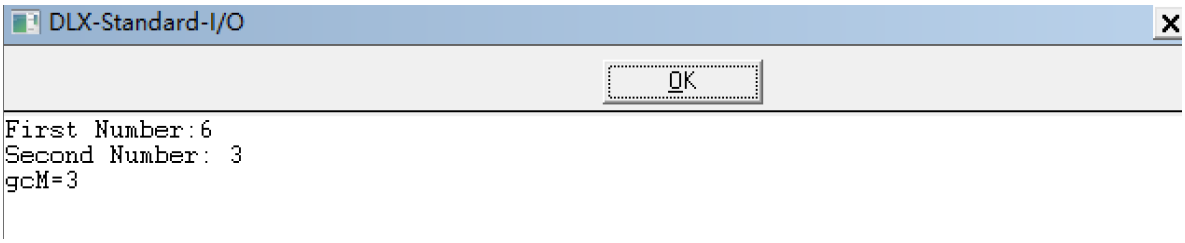
（3）若前两步都没有跳转，则说明 r1 小于 r2，那么就执行指令 r2- r1-> r2，更相减损 法用大数减去小数，得到新的两个数的最大公约数等于原来两个数的最大公约数，此时 r1 = 3，r2 = 3。

（4）接下来执行 j Loop 指令，是一条跳转指令，意味着流水线中其后面那条指令被废弃，如下图所示：

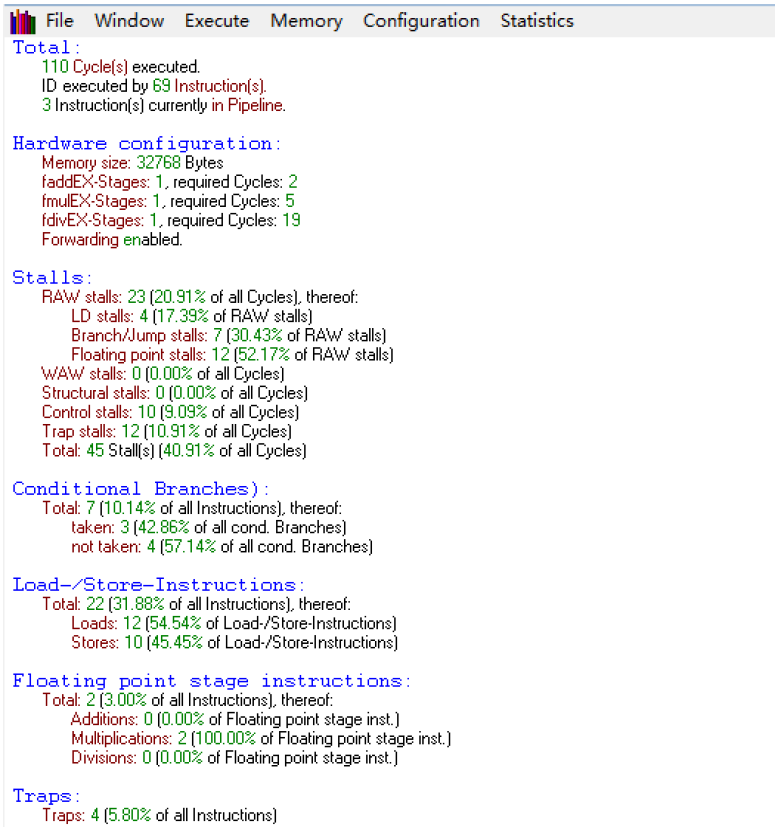
（5）重新回到 Seq，r3，r1，r2：此时两个寄存器中的值相等，直接跳转到 Result 处，将 结果存储到指定位置后，执行系统调用输出。 可以看到红绿箭头一共出现了三次，都是因为前一条指令的结果是下一条指令条件跳转的判断条件。



最后输出结果为 3，如下所示：

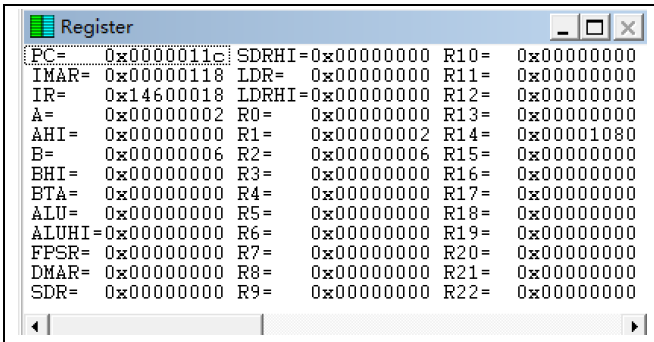


此时程序运行结束，查看 Statistics 窗口，程序执行的统计信息如下：

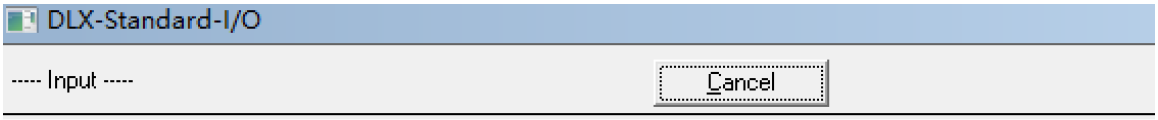


3. 重复性实验

上述实验中以 6 和 3 为例分析了指令执行的具体过程。为了进一步感受指令执行时寄存器中值的变化，以 6 和 2 作为程序的输入，重复试验，对比两个实验结果中寄存器的区别。



可以看到此时两个寄存器的值分别变为了 2 和 6，最终显示器输出结果如下所示：



查看 Statistics 窗口，程序执行的统计信息如下：

结论分析与体会：

结论分析：

1. 从两次实验的 Statistics 窗口中不难发现，第二次实验执行的时钟周期数比第一次多不少，这是跟两次实验输入数据有关的。第一次输入的 6 和 3，只做 1 次减法就结束了，而第二次输入的 6 和 1，做了 5 次减法才结束，因此第二次实验执行的时钟周期数多出不少。

2. 实验中计算最大公约数的办法

实验中使用的是更相减损法实现最大公约数的求解，具体过程如下：设第一个输入的数是 a，第二个输入的数是 b

（1）若 a== b，则 a(或 b)即为最大公约数

（2）若 a>b，则 a=a-b

（3）若 b>a，则 b=b-a

（4）若 a!= b，则回到 1

也可以使用辗转相除法来求解最大公约数。

体会：
本次实验通过单步跟踪最大公约数的求解过程，从汇编层面更好地理解了指令执行的基本步骤。实验中的 GCM.s 代码在逻辑上可以划分为 3 个部分。首先是数据的读入，涉及到 I/O 操作，之后就是通过 Loop 循环进行具体的计算，最终再次进行 I/O 操作，将计算的结果输出在显示器上。通过本次的实验操作，我进一步掌握了 WinDLX 模拟器的基本操作和使用，并且通过读源码基本上进一步熟悉了 DLX 指令集结构和特点