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：201900130133	姓名：施政良	班级： 四班
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： 2022-	

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：

Windows 7

实验步骤与内容：

实验内容

本次实验主要涉及 xxxx,具体的实验步骤可以划分为如下几个步骤

（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） 跟踪程序的运行状态

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。

具体实验过程

1. 汇编代码分析

本次实验以求两个数的最大公约数为例，从汇编代码的角度分析程序的运行过程，并观察指令流水。需要对汇编代码进行分析。

在汇编代码的 12-22 行首先对一些常量进行了定义，例如$promtp1,promp2$等。

```
1.          .data
2.
3.          ;*** Prompts for input
4.  Prompt1:     .asciiz     "First Number:"
5.  Prompt2:     .asciiz     "Second Number: "
6.
7.          ;*** Data for printf-Trap
8.  PrintfFormat:    .asciiz     "gcM=%d\n\n"
9.          .align      2
10.PrintfPar:   .word       PrintfFormat
11.PrintfValue:     .space      4
```

之后是 main 函数对应的汇编代码。在本次实验中 main 函数的主要作用是像显示器输出信息，提示用户输入数据，并且负责将用户输入的数据保存在相应的寄存器中。具体代码如下所示：

```
1.  main:
2.          ;*** Read two positive integer numbers into R1 and R2
3.          addi        r1,r0,Prompt1
4.          jal     InputUnsigned   ;read uns.-integer into R1
5.          add     r2,r1,r0    ;R2 <- R1
6.          addi        r1,r0,Prompt2
7.          jal     InputUnsigned   ;read uns.-integer into R1
```

通过代码可以看到，在 main 函数中调用了 input.s 文件中的 read 函数，实现了数据的读入。

读入数据之后，通过使用 for 循环和 if 判断来求解两个数的最大公约数。

```
1.  Loop:       ;*** Compare R1 and R2
2.          seq     r3,r1,r2    ;R1 == R2 ?
3.          bnez        r3,Result
4.          sgt     r3,r1,r2    ;R1 > R2 ?
5.          bnez        r3,r1Greater
```

在 Loop 循环中主要调用了$r1Greate$汇编代码段。

```
1.  r1Greater:  ;*** subtract r2 from r1
2.          sub     r1,r1,r2
3.          j       Loop
```

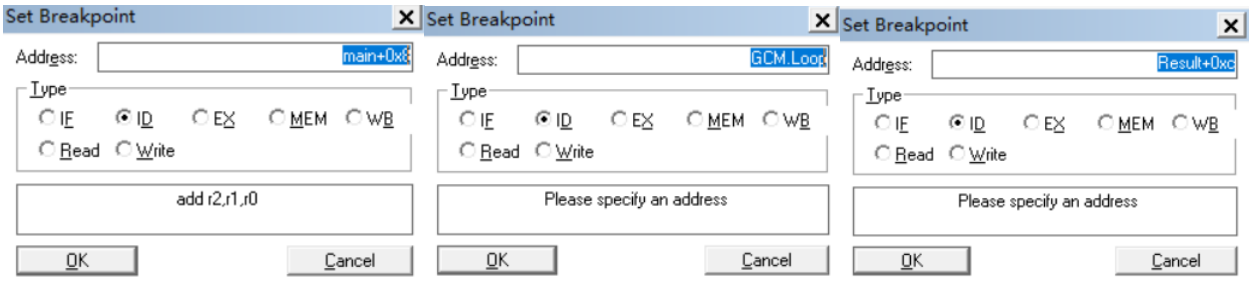
当求解出最大公约数之后，需要结果输出在显示器上，可以通过 Write 代码段实现。

```
1. Result:     ;*** Write the result (R1)
2.          sw      PrintfValue,r1
3.          addi        r14,r0,PrintfPar
4.          trap        5
5.
6.          ;*** end
7.          trap        0
```

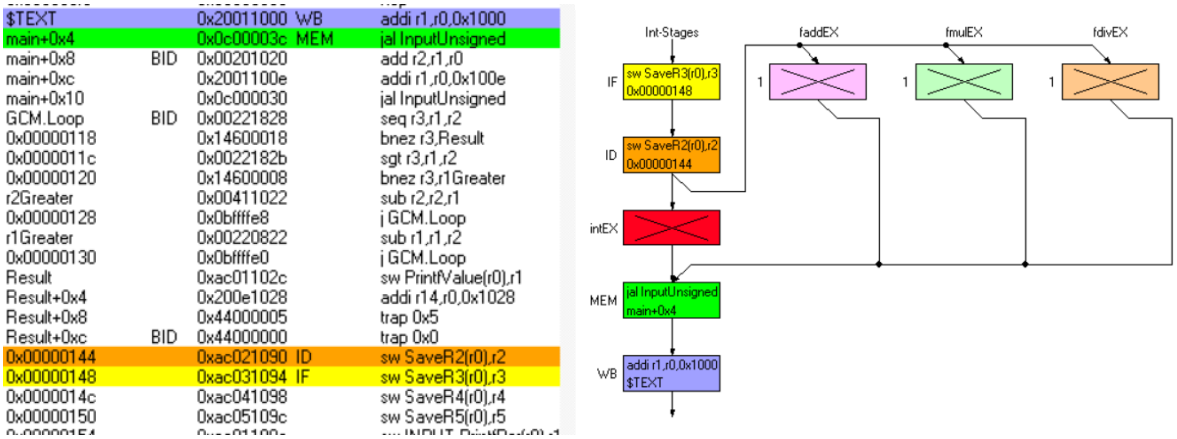
最终实现了最大公约数的求解。

2. 具体实验过程展示

根据实验指导书的提示，在运行程序的指令之前需要首先在 main+0x8(add r2,r1,r0)、gcm.loop(seg r3,r1,r2)和 result+0xc(trap 0x0)设断点

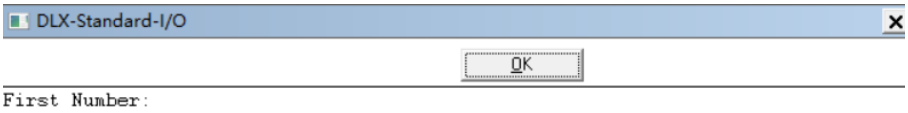


之后单步运行指令，同时观察寄存器和各个执行部件的变化。

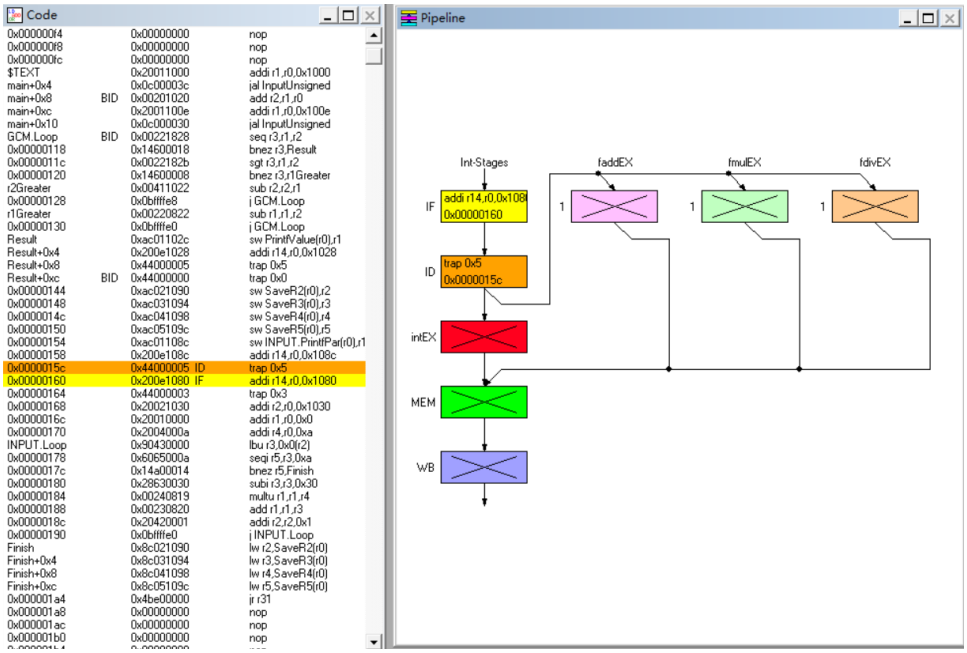


可以看到，当前程序*jal inputUnsigned*指令跳转到地址0*x*0000148位置，说明正在调用 input.s 中的数据读入函数。

继续单步运行，直到显示器输出*First number*，此时说明需要输入第一个数据。

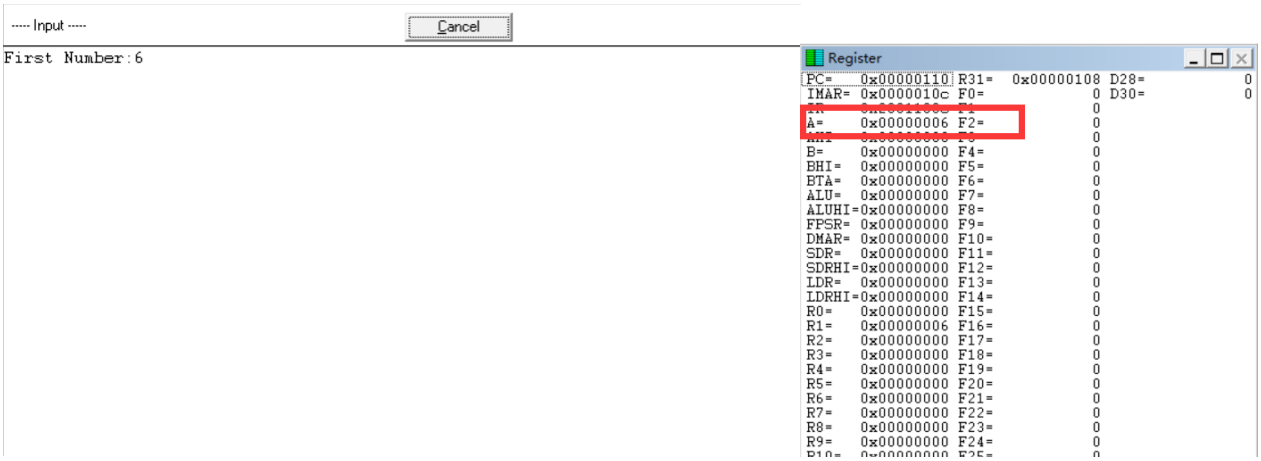


此时 PC 所在的位置和流水线的执行情况如下所示：

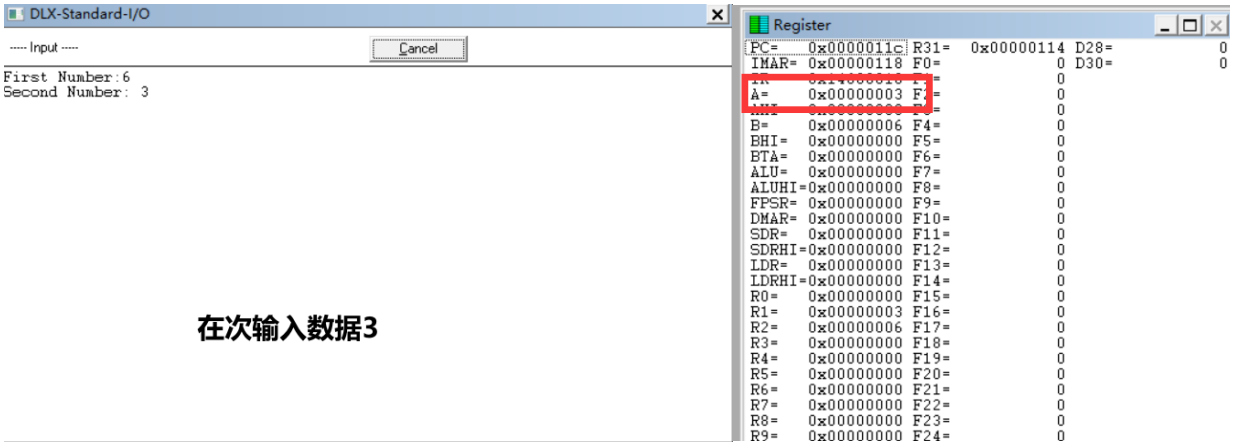


为了加快程序跟踪的速度，采用连续跟踪的方式，直接运行到断点 1 所在的位置。此时程序通过系统调用，陷入内核并执行相应的 I/O 操作，负责数据的读入。

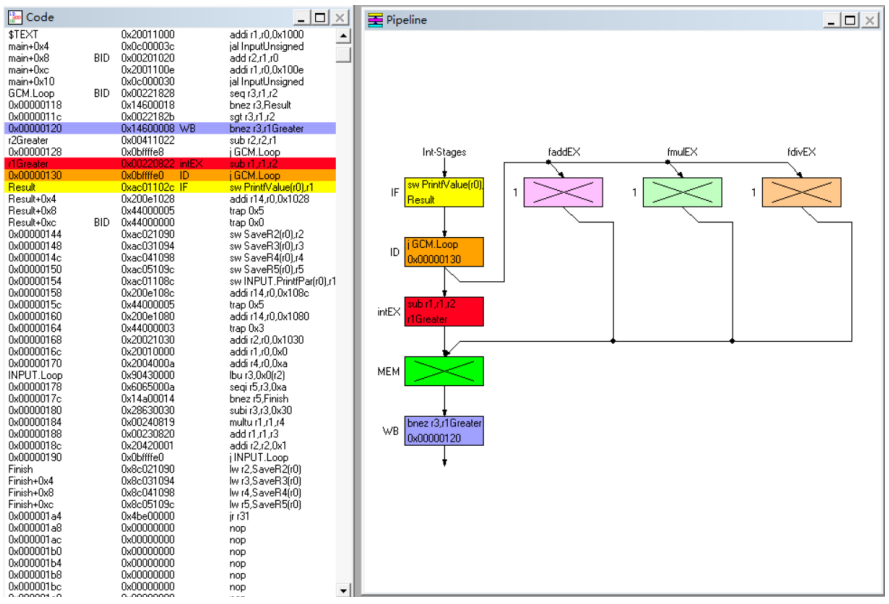
以求解 6 和 3 的最大公约数为例，键盘键入数字 6，并保存在相应的寄存器中。



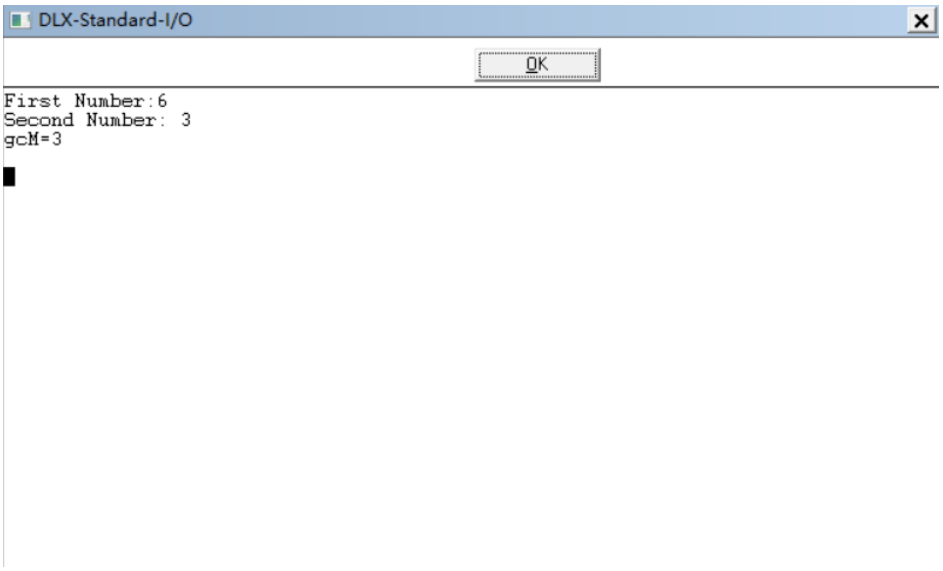
之后，根据上述对汇编代码的分析，需要再次输入第二个数字，且指令执行的逻辑同上。



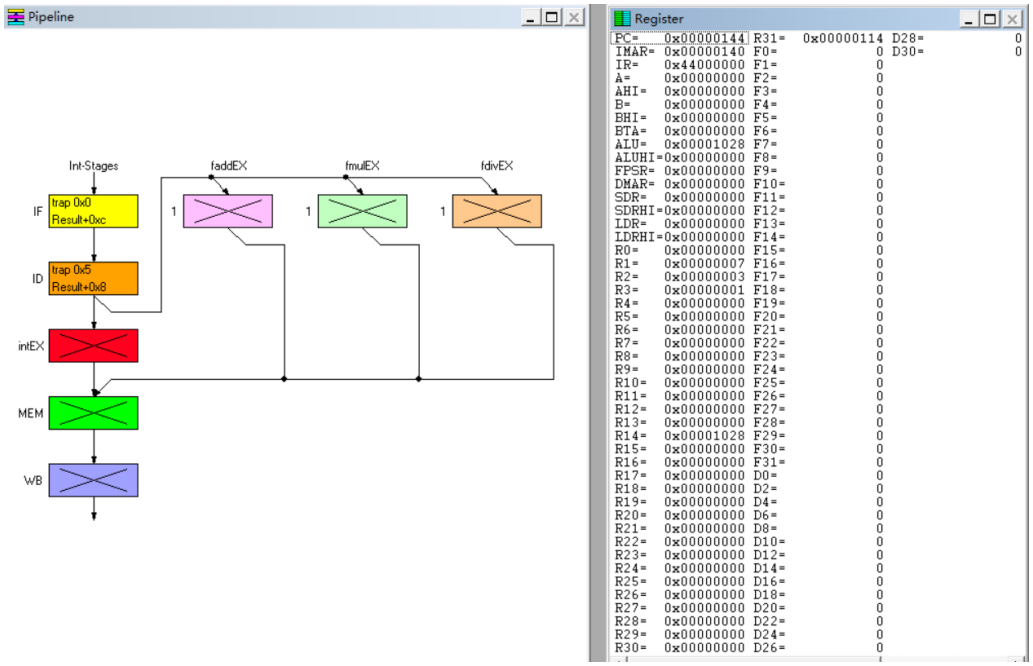
此时，完成数据的读入，进入 Loop 循环求解最大公约数。执行过程中流水线和 PC 指向的变化如下所示：

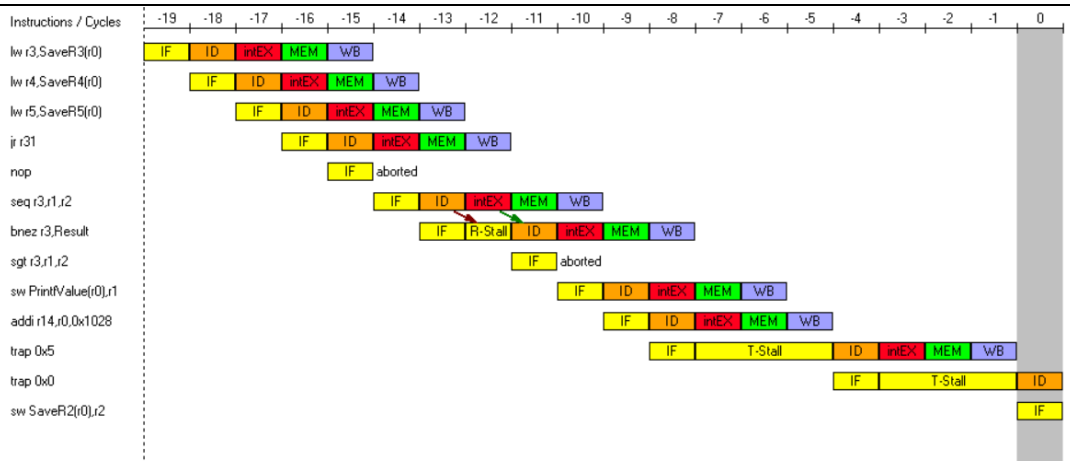


待执行到地址$Result + 0xc$对应的指令时，运行结束，此时程序将最大公约数的结果写入显示器。



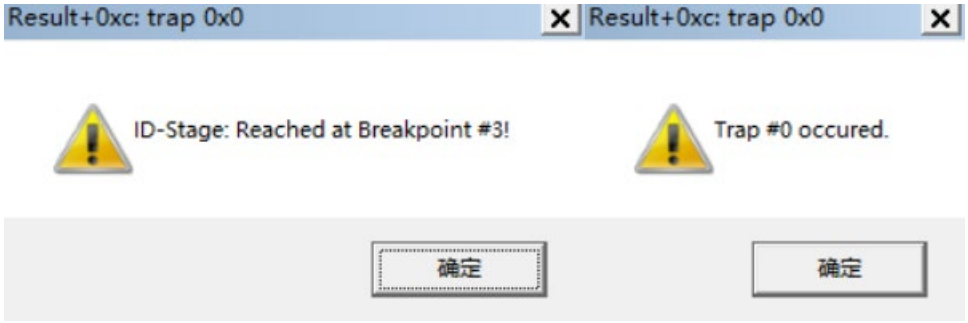
且此时寄存器和流水线的执行状态如下所示，可以发现，由于程序此时正在进行 I/O 操作，因此通过$trap$指令陷入中断。



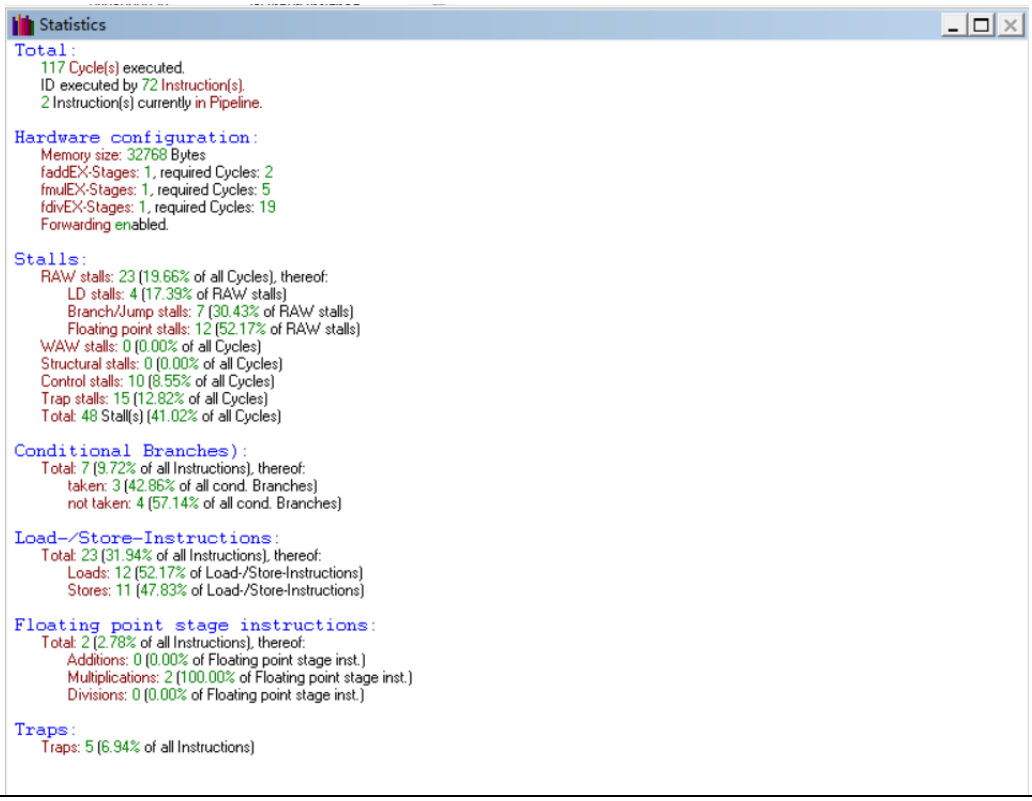


各个执行部件流水线示意图

再次执行连续跟踪，程序在第三个断点处停下，同时提示$Trap\ \#0\ occured$



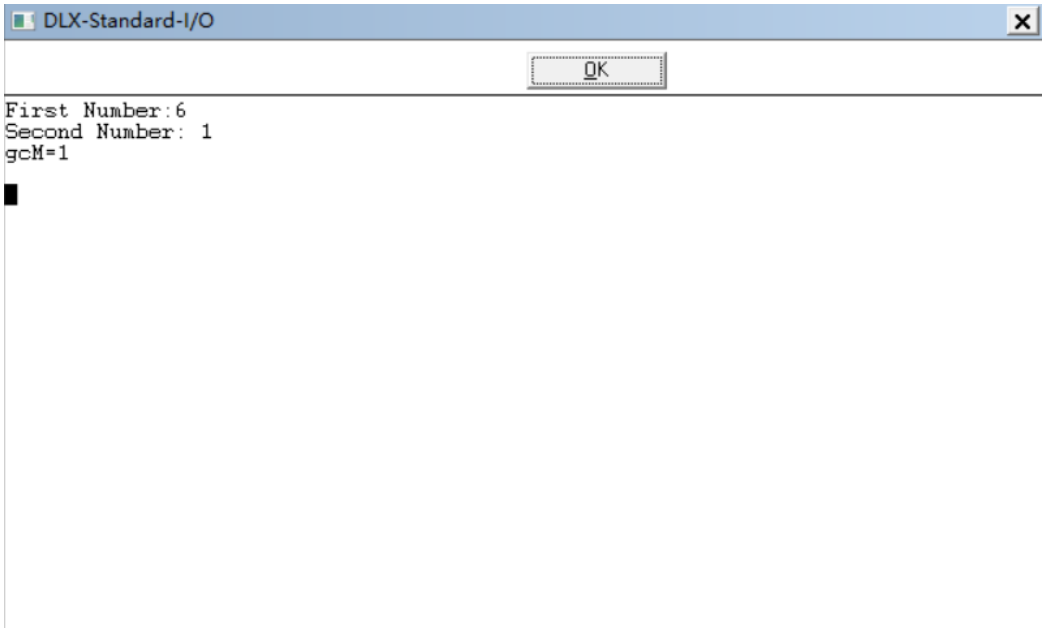
最终程序运行结束。程序执行的统计信息如下：



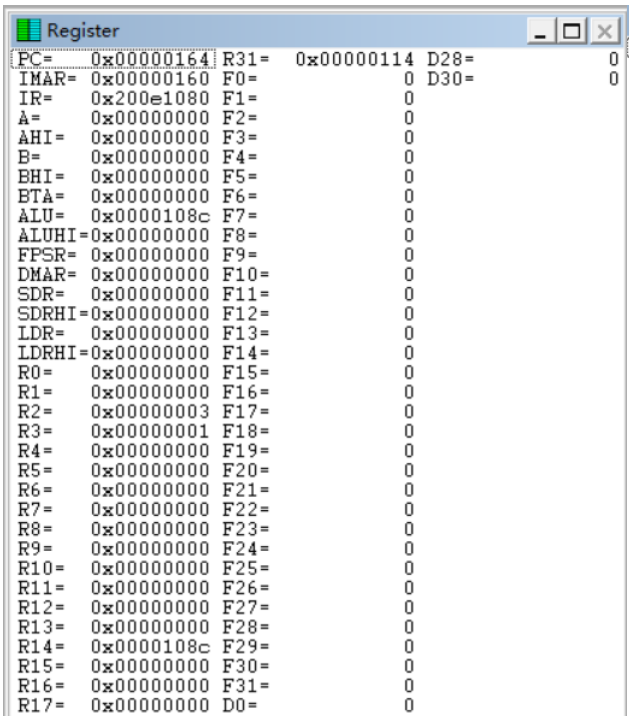
3. 重复性实验

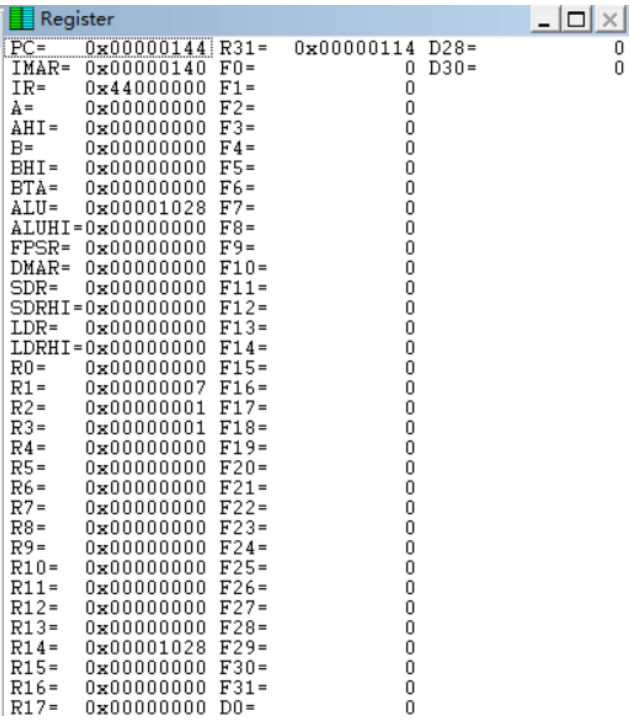
上述实验中以数字 6 和数字 3 为例分析了指令执行的具体过程。为了进一步感受指令执行时寄存器的变化，可以以数字 6 和数字 1 作为程序的输入，重复试验，对比两个实验结果中寄存器的区别。

重复上述的操作，输入数据 6 和 1，最终显示器输出结果如下所示：



对比两次试验结束时寄存器的状态，可以发现，大部分寄存器的都未被使用到，取值为 0，而由于数据输入的不同，最终程序计数器 PC，以及寄存器 IMAR 和 IR 都有所不同。





（注：（左图为数据 6、1 对应的结果，右图为数据 6、1 对应的结果）

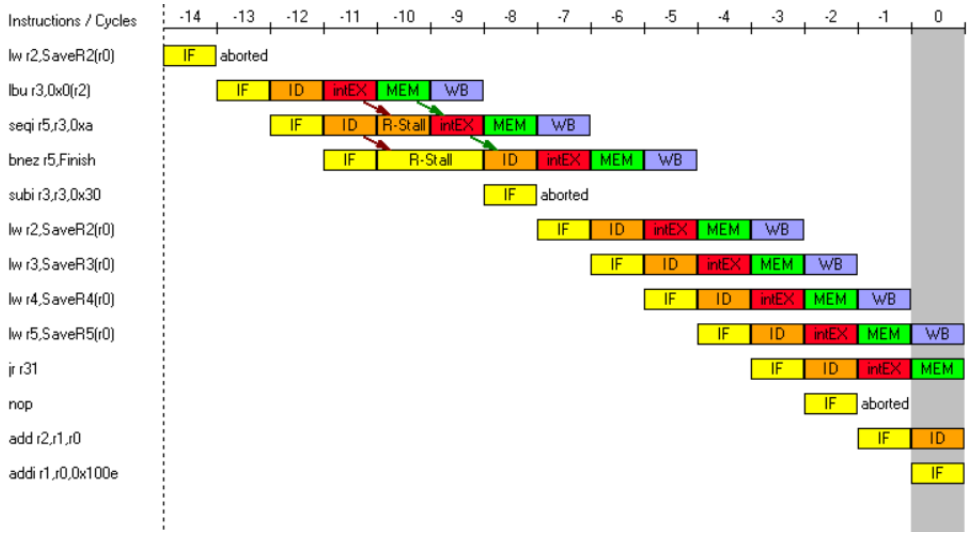
结论分析与体会：

结论分析

1. 在上述程序执行的过程中是否发生了“流水线冒险”，即流水线相关

分析：

根据执行部件的指令流水可知，在程序执行的过程中发生了流水线冒险。例如，当执行第二条和第三条指令时，会发生数据相关，如下图所示：



在指令之间出现了红和绿的箭头。红色箭头表示需要一个暂 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。绿色箭头 表示定向技术的使用。

2. 分析实验中最大公约数的计算方法

分析：

实验中使用更相减损术实现最大公约数的求解。

第一步：任意给定两个正整数；判断它们是否都是偶数。若是，则用 2 约 简；若不是则执行第二步。

第二步：以较大的数减较小的数，接着把所得的差与较小的数 比较，并以大数减小数。继续这个操作，直到所得的减数和差相等为止。则第一步中约掉 的若干个 2 的积与第二步中等数的乘积就是所求的最大公约数。即如下四步：

1 若 a > b，则 a = a - b

2 若 b > a，则 b = b - a

3 若 a == b，则 a(或 b)即为最大公约数

4 若 a != b，则回到 1

体会

本次实验通过单步跟踪最大公约数的求解，从汇编层面了解了指令执行的基本步骤。实验中的 GCM.s 代码在逻辑上可以大致划分为 3 个部分。首先是数据的读入，涉及到 I/O 操作，之后通过 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。

附录

GCM.S 完整代码

```
1   ;*********** WINDLX Ex.1: Greatest common measure *************
2  ;*********** (c) 1991 G藨ther Raidl    *************
3  ;*********** Modified 1992 Maziar Khosravipour   *************
4
5  ;-----------------------------------------------------------------------
6  ; Program begins at symbol main
7  ; requires module INPUT
8  ; Read two positive integer numbers from stdin, calculate the gcm
9  ; and write the result to stdout
10 ;-----------------------------------------------------------------------
11
12   .data
13
14   ;*** Prompts for input
15 Prompt1: .asciiz  "First Number:"
16 Prompt2: .asciiz  "Second Number: "
17
18   ;*** Data for printf-Trap
19 PrintfFormat: .asciiz  "gcM=%d\n\n"
20   .align  2
21 PrintfPar: .word  PrintfFormat
22 PrintfValue: .space  4
23
24
25   .text
26   .global main
27 main:
28   ;*** Read two positive integer numbers into R1 and R2
29   addi  r1,r0,Prompt1
30   jal  InputUnsigned ;read uns.-integer into R1
31   add  r2,r1,r0 ;R2 <- R1
32   addi  r1,r0,Prompt2
```

```
33   jal  InputUnsigned ;read uns.-integer into R1
34
35 Loop:  ;*** Compare R1 and R2
36   seq  r3,r1,r2 ;R1 == R2 ?
37   bnez  r3,Result
38   sgt  r3,r1,r2 ;R1 > R2 ?
39   bnez  r3,r1Greater
40
41 r2Greater: ;*** subtract r1 from r2
42   sub  r2,r2,r1
43   j  Loop
44
45 r1Greater: ;*** subtract r2 from r1
46   sub  r1,r1,r2
47   j  Loop
48
49 Result:  ;*** Write the result (R1)
50   sw  PrintfValue,r1
51   addi  r14,r0,PrintfPar
52   trap  5
53
54   ;*** end
55   trap  0
```