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：	姓名：	班级：
实验题目：熟悉 WinDLX 的使用		
实验学时：2	实验日期：2024.4.26	

实验目的：

（1）通过本实验，熟悉 WinDLX 模拟器的操作和使用。

（2）了解 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 模拟器执行求阶乘程序 fact.s 。执行步骤详见“WinDLX 教程”。 这个程序说明浮点指令的使用。该程序从标准输入读入一个整数，求其阶乘，然后将结果输出。该程序中调用了 input.s 中的输入子程序，这个子程序用于读入正整数。

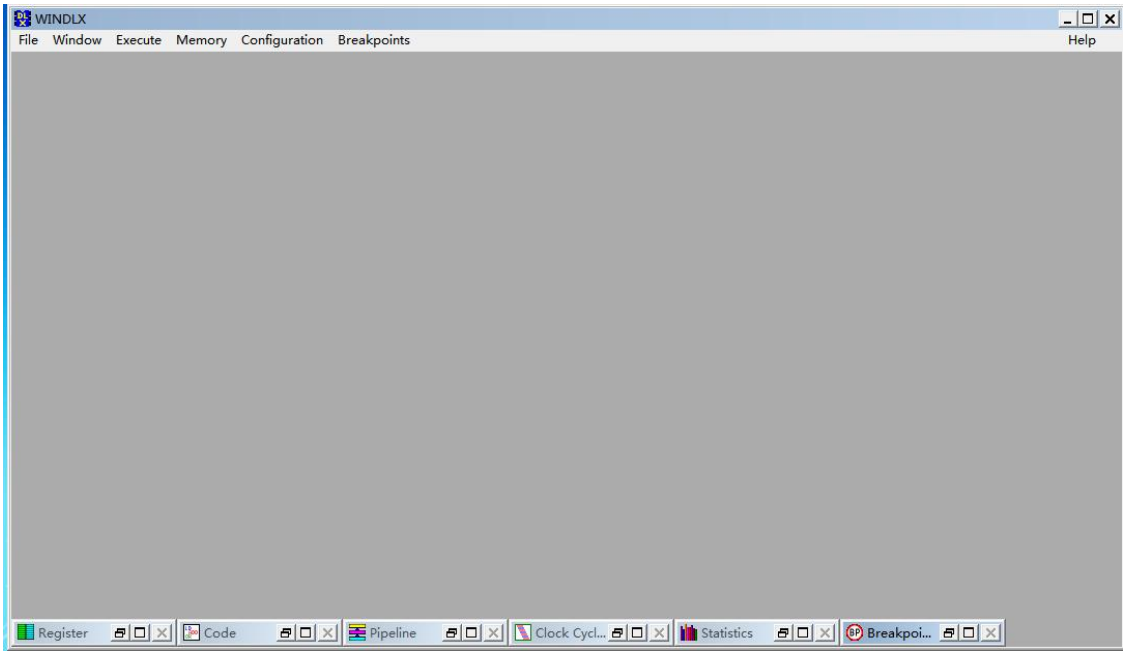
（2）输入数据“3”采用单步执行方法，完成程序并通过上述使用 WinDLX，总结 WinDLX 的特点。

（3）注意观察变量说明语句所建立的数据区，理解 WinDLX 指令系统。

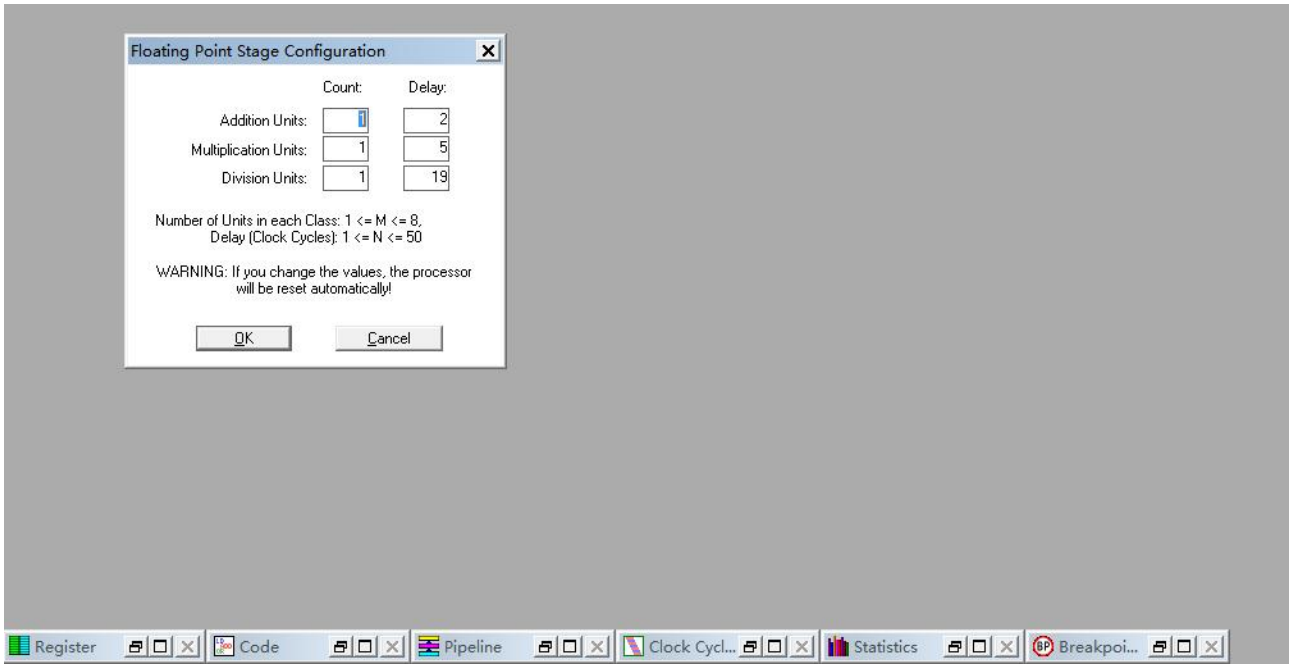
实验步骤：

1. 配置 WinDLX

首先双击 WinDLX 图标，启动 WinDLX，之后将出现一个带有六个图标的主窗口，如下图所示。



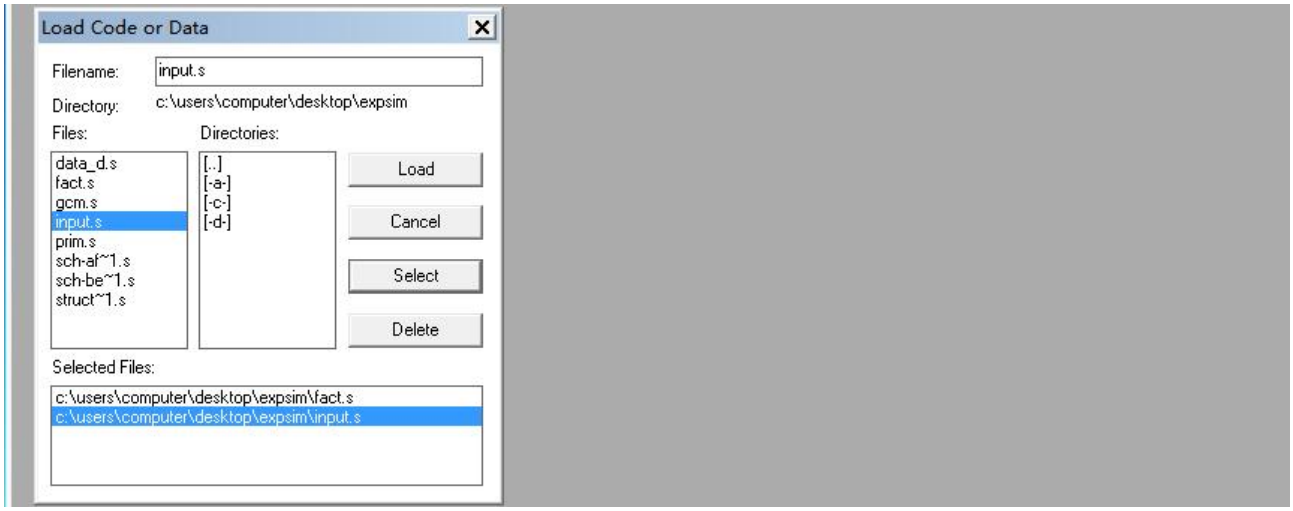
为了初始化模拟器，点击 File 菜单中的 Reset all 菜单项，弹出一个“Reset DLX”的对话框。然后点击窗口中的“确认”按钮即可。点击窗口中的“确认”按钮即可。WinDLX 可以在多种配置下工作。可以改变流水线的结构和时间要求、存储器大小和其他几个控制模拟的参数。点击 Configuration / Floating Point Stages（点击 Configuration 打开菜单，然后点击 Floating Point Stages 菜单项），选择如下标准配置：



点击 Configuration / Memory Size ，可以设置模拟处理器的存储器大小。应设置为 0x8000，然后点击 OK 返回主窗口。

2. 装载测试程序

在开始模拟之前，应装入一个程序到主存。为此，选择 File/Load Code or Data，窗口中会列出目录中所有汇编程序。本实验中要求执行 fact.s 计算一个整型值的阶乘；此处用程序 input.s 中包含一个子程序，它读标准输入（键盘）并将值存入 DLX 处理器的通用寄存器 R1 中。按如下步骤操作，可将这两个文件装入主存

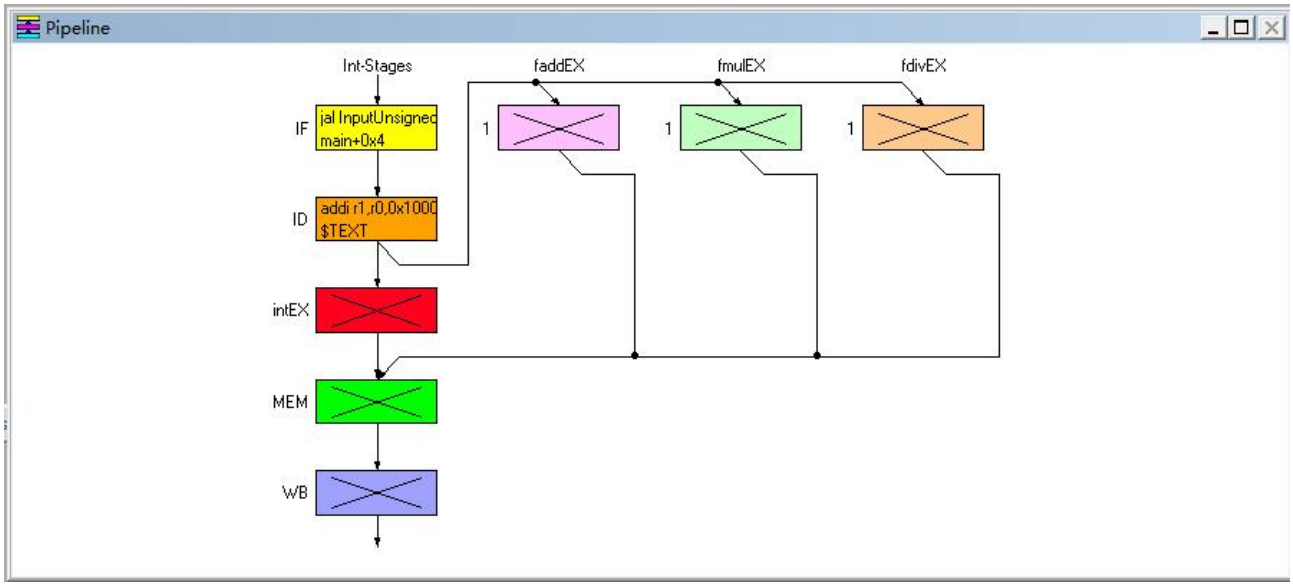


3. 模拟

在主窗口中，存在六个图标，它们分别为“Register”，“Code”，“Pipeline”，“Clock”，“Cycle Diagram”，“Statistics”和“BreakPoints”。

3.1 Pipeline 窗口

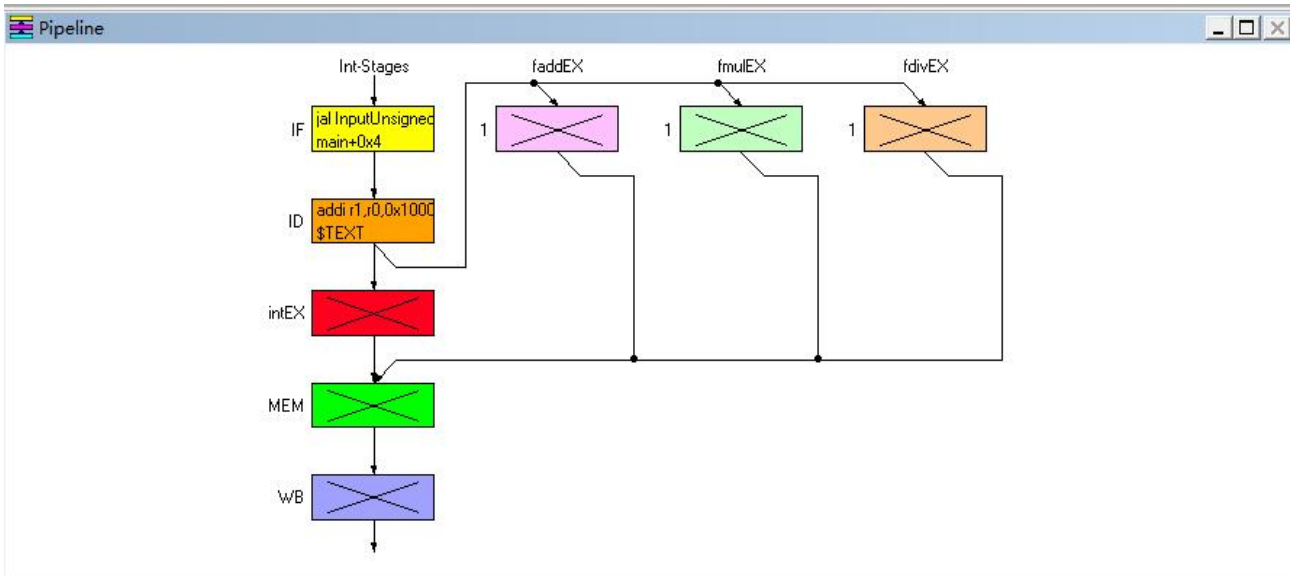
双击 Pipeline，会出现一个子窗口，窗口中用图表形式显示了 DLX 的五段流水线。



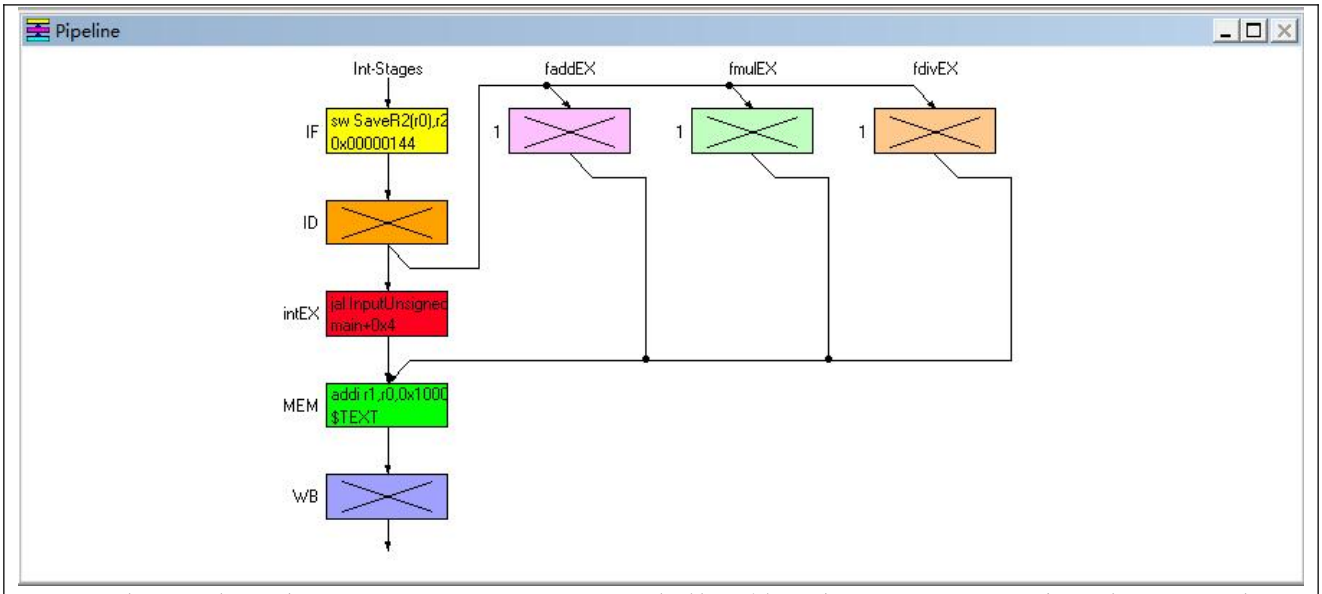
该窗口显示 DLX 处理器的五个流水段和浮点操作（加/减，乘和除）的单元。

3.2 Code 窗口

双击 Code 图标，可看到代表存储器内容的三栏信息，从左到右依次为：地址(符号或数字)、命令的十六进制机器代码和汇编命令。并点击主窗口中的 Execution 开始模拟。按下 F7 键，模拟就向前执行一步，其中不同颜色指明命令处于流水线的哪一段，如下所示：



再次按下 F7 键，代码窗口中的颜色会再改变，红色表明命令处入第三段“intEX”。 再按 F7，图形显示将变为：在代码窗口中，黄色出现在更下面的位置，并且可能是唯一彩色行。查看一下 Pipeline 窗口，你会发现 IF，intEX 和 MEM 段正在使用而 ID 段没有。为什么？

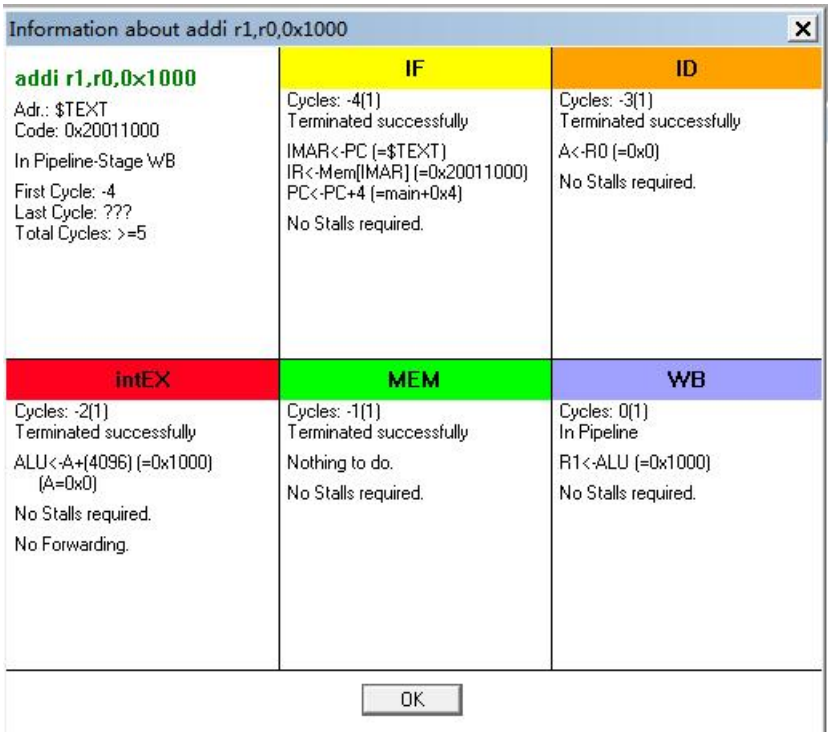


这是因为第二条指令 jal InputUnsigned 是无条件跳转指令，而根据五级流水线的执行过程，只有在第三个时钟周期，jal 指令被译码后才知道需要跳转，而此时第三条指令 movi2fp 已经取出，但需执行的下一条命令在另一个地址处，因此需要取消它，在流水线中留下气泡。

jal 指令的分支地址为 InputUnsigned,在 Memory 和 symbols 中可以找到它的实际值为 0x144，用作地址。

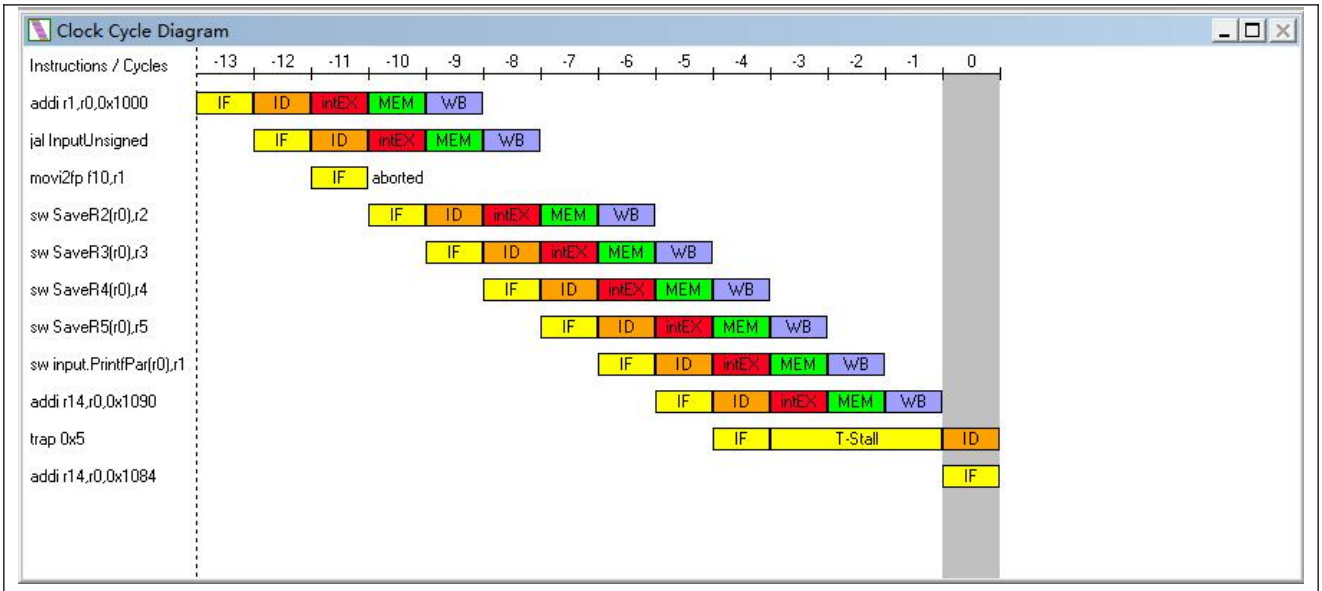
3.3 Clock Cycle Diagram 窗口

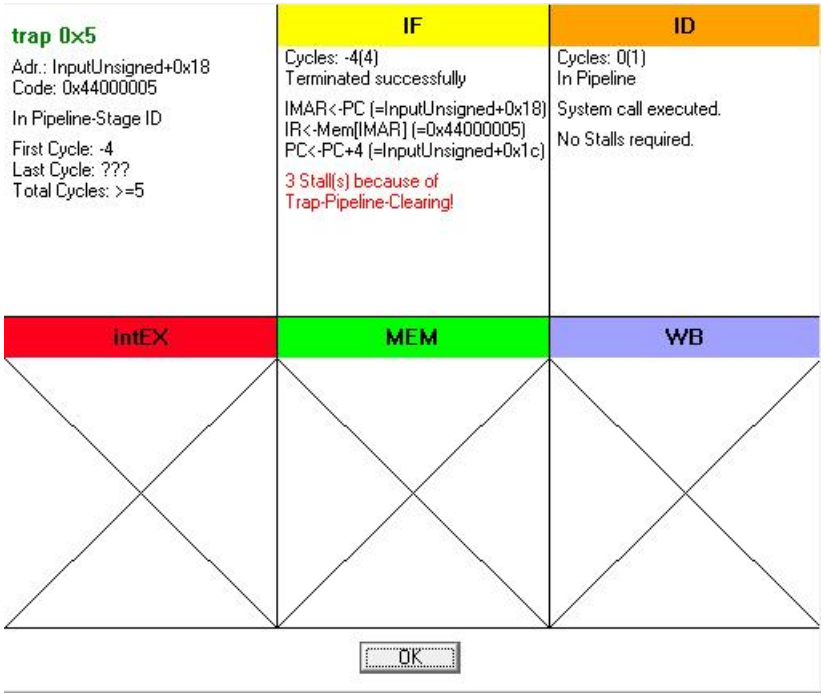
该窗口显示流水线的时空图，再点击一次 F7，此时第一条命令到达流水线的最后一段，Clock Cycle Diagram 窗口中点击指令 addi r1，r0，0x1000 查看指令各个阶段的执行详情。



3.4 Breakpoint 窗口

在 Code 窗口中包含命令 trap 0x5 的 0x0000015c 行处设置断点,该命令是写屏幕的系统调用。此时查看 Clock Cycle Diagram，如下图所示:

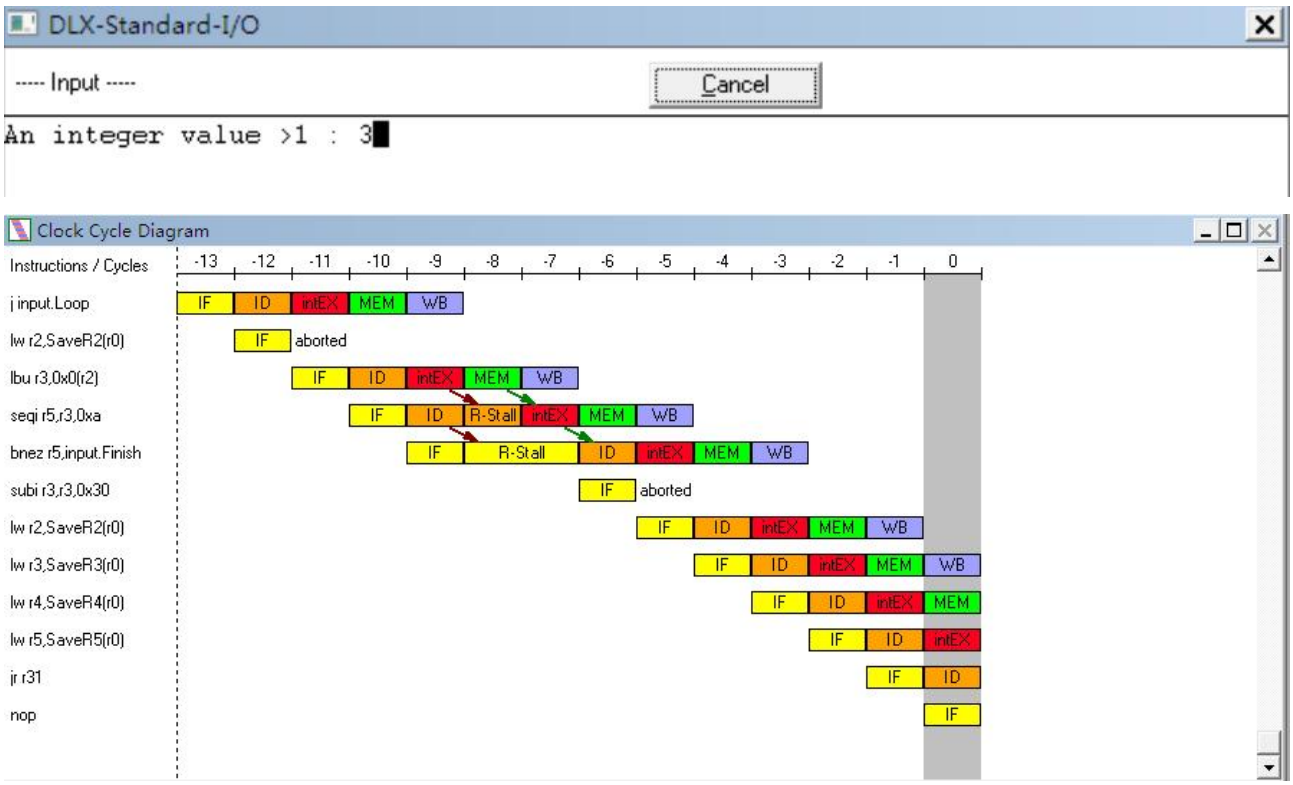




出现上图是因为无论何时遇到一条 trap 指令，DLX 处理器中的流水线将被清空。3stalls because of Trap-Pipeline-Clearing，trap 指令通过阻塞 IF 段将流水段排空。

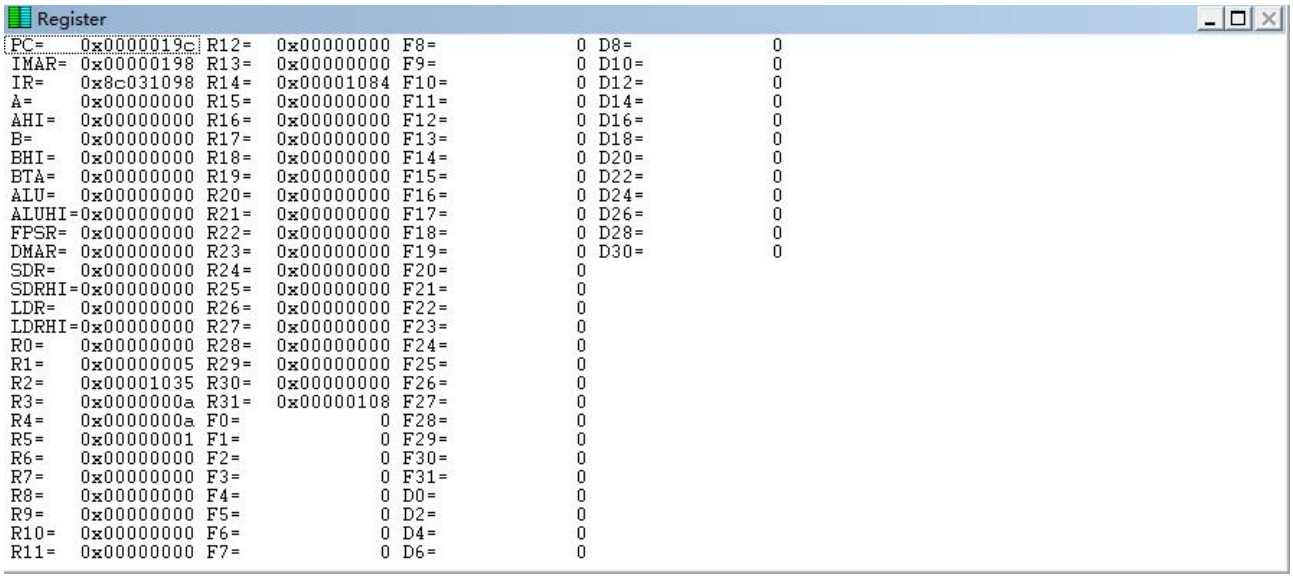
3.5 Register 窗口

运用同样的办法在地址 0x0000001a4 处设置断点，然后按下 F5 继续运行，输入 3 后模拟运行到断点 2 处。

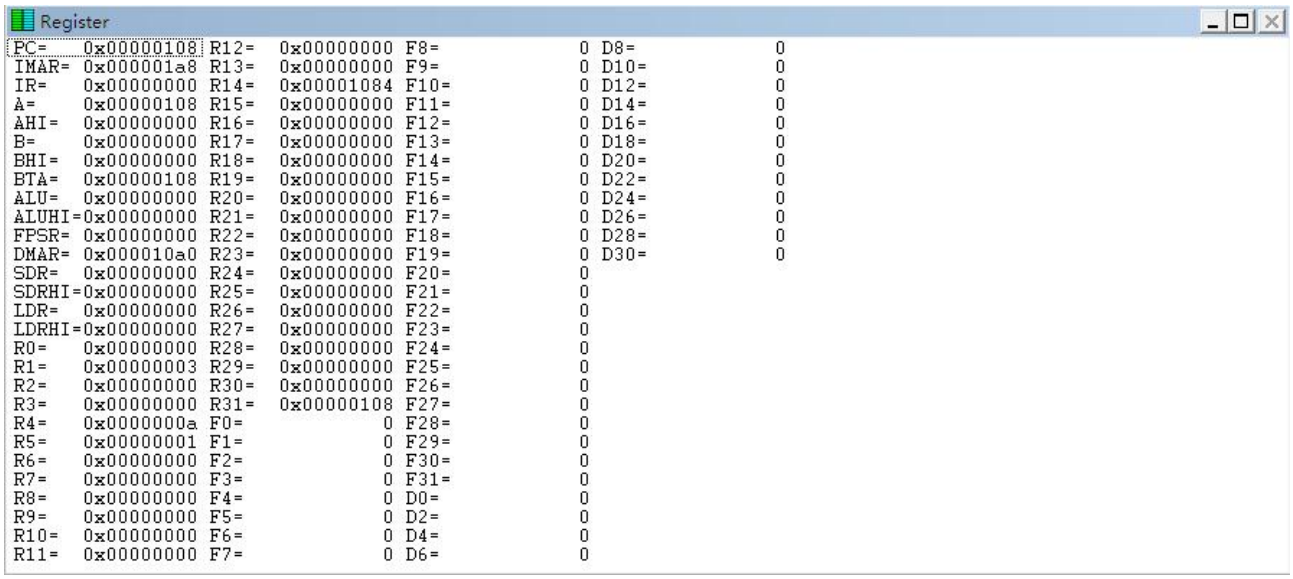


在 Clock cycle diagram 窗口中，在指令之间出现了红和绿的箭头。红色箭头表示需要一个暂停，绿色箭头表示定向技术的使用。

一开始 Register 窗口如下：



在地址 0x000001a4 处设置断点，后按 F5 执行后，此时 Register 窗口如下：



其中寄存器 R1-R3 中的值发生了变化。

3.6 Statistics 窗口

按 F5 使程序完成执行，出现消息“Trap #0 occurred”表明最后一条指令 trap 0 已经执行，Trap 指令中编号 0 没有定义，只是用来终止程序。此时查看 Statistics。Statistics 窗口中提供各个方面的信息：模拟中硬件配置情况、暂停及原因、条件分支、Load/Store 指令、浮点指令和 traps。窗口中给出事件发生的次数和百分比，比如 RAW stalls：10(12.34 % of all Cycles)。

如下图所示：

其中左图是采用了定向技术,右图是关闭了定向技术,RAW 从 10 增加到 26,而 Trap 和 Control stalls 不变，总的周期数从 81 增加到 98，定向技术在这个实验中加速比为 98 /81 =1.2098，快 20.98%

结论分析与体会：

结论分析：

1. 关于指令流水中的相关（流水线冲突）

分析：

流水线中的相关是指相邻或相近的指令因存在某种关联，后面的指令不能在原指定的时钟周期开始执行。一般来说，流水线中的相关主要分为如下三种类型：

(1)结构相关：当硬件资源满足不了指令重叠执行的要求，而发生资源冲突时，就发生了结构相关。

(2)数据相关：当一条指令需要用到前面指令的执行结果，而这些指令均在流水线中重叠执行时，就可能引起数据相关。

(3)控制相关：当流水线遇到分支指令和其它能够改变 PC 值的指令时，就会发生控制相关。

一旦流水线中出现相关，必然会给指令在流水线中的顺利执行带来许多问题，如果不能很好地解决相关问题，轻则影响流水线的性能，甚至导致错误的执行结果。消除相关的基本方法是让流水线暂停执行某些指令，而继续执行其它一些指令。

2.在实际设计时，是否允许存在指令结构的相关

分析：

在实际设计中是允许存在指令结构的相关。主要处于以下两点原因：

（1） 减少功能单元的延迟。

（2） 减少硬件代价，如果为了避免结构相关而将流水线中的 所有功能单元完全流水化，或者设置足够的硬件资源，那么所带来的硬件代价必定很大.

体会：

经过本次实验,从流水线的角度分析了程序的运行。五级流水线分为五个部分,包括取指(IF),译码（ID），执行（EX），访存（MEM）以及写回操作（WB）。并且，通过这种指令流水的方式，充分利用了不同硬件之间的并行性，可以显著的提高程序执行的速度。即在流水线过程中，只有第一个指令需要占用 5 个独立的周期，其余的指令都只需要占用独立的一个周期，剩下的都是和其余的指令周期交叉。 但是指令流水线也存在很多细节问题，例如在程序的运行过程中可能会遇到一些情况使得流水线无法正确执行后续指令而引起流水线阻塞或停顿，这种现象称为流水线冲突或流水线冒险。在实际中可以采用编译器优化指令执行顺序，加入气泡，加入额外的旁路缓解流水线冒险的现象。