

Project2 Mars 实验

Mars 是 MIPS 汇编语言的模拟机，由密苏里州立大学开发。在后续的实验
中，我们将使用 Mars 来模拟 MIPS CPU 的执行过程。通过本实验，达到熟练使用
Mars 来进行 MIPS 汇编代码的编写、调试以及 CPU 执行过程的分析。

1. 工具使用视频

请访问 <http://mooc.buaa.edu.cn>，选择统一认证入口登陆。

注册课程《M_G06B2830 数字系统设计工具集》。

访问课程页面中 Mars 工具介绍相关内容。

2. Mars 下载及环境设置

请参看视频——《Mars》。

2.1. 下载 Mars

从 <http://courses.missouristate.edu/KenVollmar/MARS/download.htm> 下载：

注意：Mars 为 “.jar” 文件，可以在 Java 环境中直接运行，如果你的计算机
上没有安装 Java 环境请继续下面的步骤。

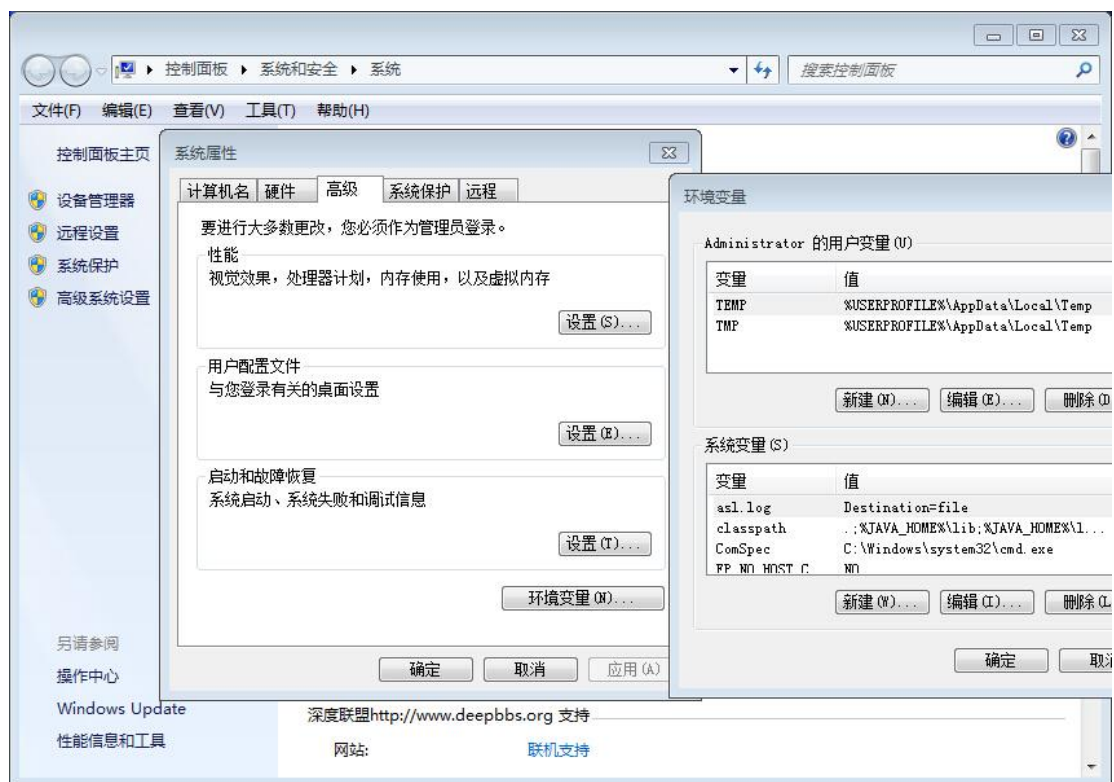
2.2. 下载 JAVA J2SE SDK

从 <http://www.oracle.com/technetwork/java/javase/downloads/index.html> 下载：

下载后在本地进行安装，安装过程中可将安装目录拷贝下来，后面在设置环
境变量的时候要用到。

2.3. 在 Win7 中设置环境变量

右键点击【计算机】选择【属性】，再点击左边的【高级系统设置】，然后点
击【环境变量】，如下图所示。



1. 在【系统变量】栏中新建一个系统变量，变量名“JAVA_HOME”，变量值为“C:\Program Files\Java\jdk1.8.0_11\”。注意，这里的变量值取决于你的 JDK 安装路径，安装在哪儿就填哪儿。
2. 修改【系统变量】中的变量“path”的值，在最前面加上“%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin”；
3. 在【系统变量】中新建变量“classpath”，变量值为“.;%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar”。

下面，我们可以测试一下 Java 环境是否搭建成功。在 DOS 中输入“java -version”，如果出现 java 版本信息，说明 JAVA 环境搭建成功。



3. 运行 Mars

这里我们介绍两种运行 Mars 的方式：

第一种就是在 DOS 环境下，用 DOS 命令来运行 Mars。这里可以将下载的 Mars4_4.jar 文件拷贝到 C 盘根目录下，然后运行 DOS，如下所示即可打开 Mars。



第二种就是通过直接双击 Mars 来运行程序。这里需要做一些设置。

安装好 Java 虚拟机后，双击需要运行的 jar 文件或在 jar 文件上点击鼠标右键，在弹出的菜单中选择“打开方式”→“选择默认程序”。

在“打开方式”设置窗口中点击选中选项“始终使用选择的程序打开这种文件”前的复选框，再点击旁边的“浏览”按钮。

选择 JAVA 虚拟机的安装文件夹，如果安装的是 Java 7，安装文件夹一般为“C:\Program Files\Java\jre7\bin”（具体文件夹请自行查找），找到 javaw.exe 文件，点击“打开”按钮。

在 Windows 开始菜单的搜索框中输入“regedit”，在上方搜索出的文件 regedit 上点击鼠标右键，在弹出的菜单中选择“以管理员身份运行”。

在注册表编辑器中，找到“HKEY_CLASSES_ROOT\Applications\javaw.exe\shell\open\command”，在其中文件打开命令中加入参数“-jar”（无引号），修改后的数值类似：“C:\Program Files\Java\jre7\bin\javaw.exe" -jar "%1"”（只需要添加-jar 参数，无需修改其他信息），退出注册表编辑器。

4. Mars 使用实例：斐波那契数列

下面，我们通过示例程序 Fibonacci.asm（用来计算斐波那契数列）来演示 Mars 的操作步骤。

1. 双击“Mars4_4.jar”，启动 Mars 程序。
2. 点击菜单 File...Open，打开 Fibonacci.asm 程序。



3. 双击工具栏按钮 ，汇编该程序。

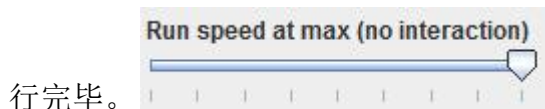
4. 熟悉程序初始数据在内存中的位置和具体的数值。单击复选框

☒ **Hexadecimal Values**，在十进制显示和十六进制显示之间进行切换。

5. 使用 Setting 菜单配置 Mars 的显示内容。

6. 观察寄存器显示窗口，这里显示了 32 个通用 MIPS 寄存器的内容。寄存器显示窗口里的另外两个标签页显示了浮点运算寄存器 (Coproc1) 和 (Coproc0)，暂时可以不看。


7. 使用滑动条来改变运行速度，便于观察汇编程序的执行过程，而不是瞬间运

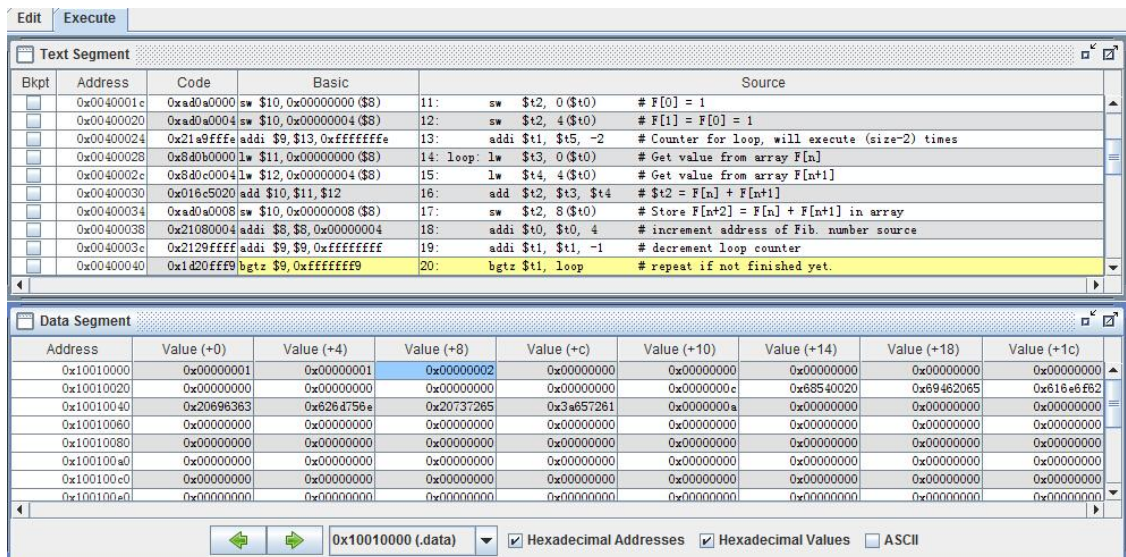



行完毕。

8. 选择程序运行方式：



- ：直接运行程序，通过黄色高亮部分来观察程序的运行，并在数据段显示窗口 Data Segment display 中观察斐波那契数列值得变化，如下图中黄色和蓝色部分所示。

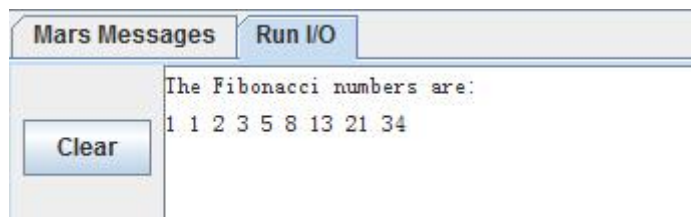


- ：单步执行程序。



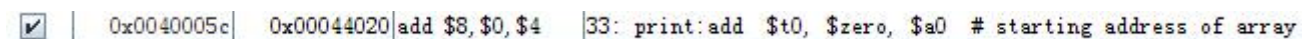
- ：重置程序为初始值。

9. 在输入输出窗口中观察程序的输出，如下图所示。



10. 修改某个内存地址中的内容。

- 在输出结果子程序的第一条指令前添加断点，即勾中该指令的复选框。



- 重新运行程序后，将在断点处停止运行。
- 双击存储斐波那契数列的某个内存位置，该单元将高度显示，并接受键盘输入。输入一个数值，按回车键结束输入，如下图所示，内存地址 0X10010008 中现在存放的数据是 0X00000024。
- 单击运行，从断点处继续执行程序。程序的输出将会包含刚才输入的数值，而不是原来的斐波那契数列。

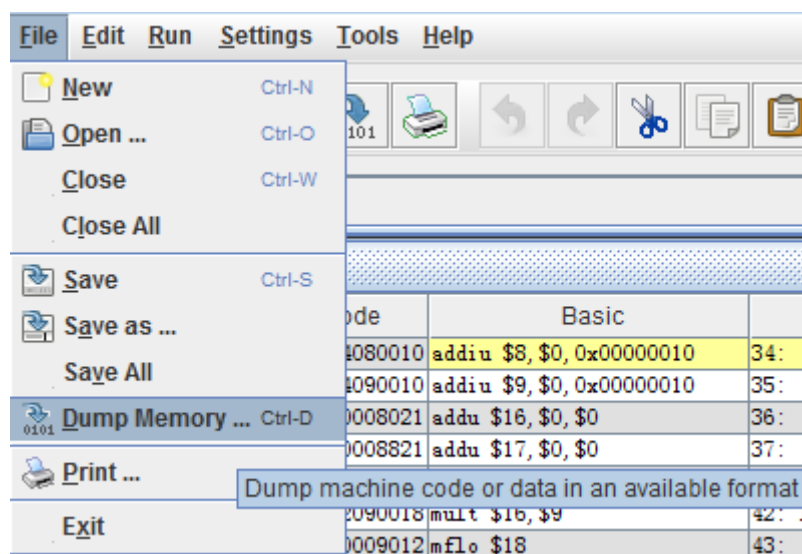
5. 导出机器码和运行结果

最后，我们可以将 MARS 中的工作成果保存下来，为后续的硬件设计和分析提供输入。这里，我们需要保存两个文件，第一个就是要将编写的汇编代码保存成机器码文件，在后续硬件设计过程中，我们可以将该文件直接加载到存储器中运行。第二个就是要将 MIPS 模拟器的运行结果，也就是数据段的最终的值保留下来，这个数据可以与我们的 MIPS 系统运行结果进行比较，从而帮助我们发现设计中的问题。具体的操作步骤为：

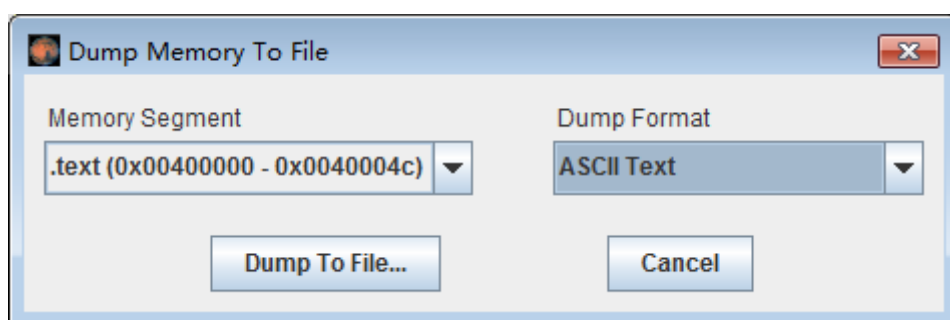
1. 在程序已经汇编完，并运行结束后，才可以进行程序机器码和运行结果数据的导出。这里有两种方法。第一种就是通过菜单栏中的“Dump

Memory”，第二种就是点击工具栏中的快捷键





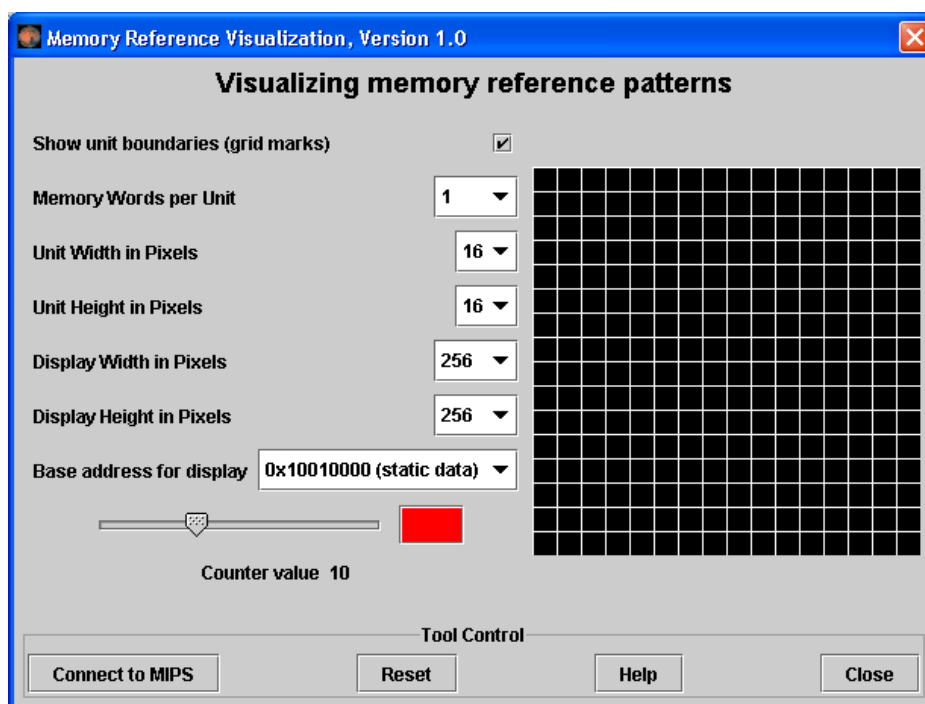
2. 在弹出的窗口中，根据需要进行设置，可以选择要导出的文件类型（Data/Text），选择要导出的文件位置，以及导出的文件格式。



6. Mars 分析工具

在 Mars 中有一个工具菜单，在这个工具菜单中提供了一系列的辅助工具，这些工具提供了 MIPS 程序运行阶段观察 MIPS 存储器和寄存器实时状态。下面，只介绍如何使用 Memory Reference Visualization 工具模拟存储引用过程。

1. 在 Mars 主界面中，打开 Examples 文件夹中的程序 **row-major.asm**。
2. 编译程序。
3. 从菜单栏中选择 “**Tools**” 标签，弹出的下拉框中选择 “**Memory Reference Visualization**”。在屏幕中间会打开一个新的操作窗口，如下图所示。



在工具的右边有一个网格图，它模拟了 MIPS 存储空间分配的情况。基址，即：第一个静态数据块对应着网格中的左上角的第一个网格块。每一个网格块的地址自左到右，自上而下的排列。

网格中每一个单元的颜色会随着它所代表的存储空间被引用的次数而变化。存储单元被引用的次数和对应的颜色的关系如下：黑色代表 0 次，蓝色代表 1 次，绿色代表 2 次，黄色代表 3 次和 4 次，橙色代表 5~9 次，红色代表 10 次及以上。

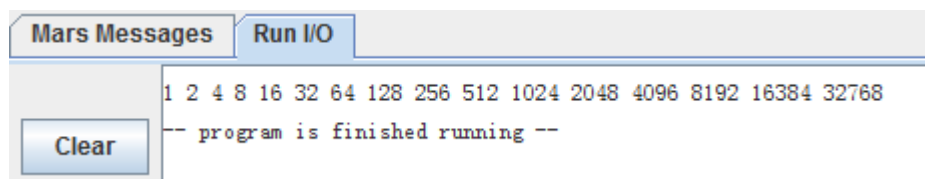
4. 点击“**Connect to MIPS**”按钮，为 MIPS 存储器注册一个观察器，当程序执行的时候，工具将实时捕获存储器状态。
5. 回到 Mars 主界面，调整运行速度滚动条至 30 条指令/秒。
6. 运行程序。观察 Memory Reference Visualization 工具界面，每一次 MIPS 存储器的访问都会有相应的动画显示。在运行过程中的任何时候都可以根据需要中断程序的执行。
7. 载入新的汇编程序 `column-major.asm`，重复 2~7 步的操作。预测汇编程序执行阶段的存储访问次序和次数，观察工具中网格动画的输出是否一致。
8. 载入新的汇编程序 `fibonacci.asm`，重复 2~7 步的操作。预测汇编程序执行阶段的存储访问次序和次数，观察工具中网格动画的输出是否一致。

9. 调整运行速度后，再一次运行，并观察运行现象。
10. 打开一个新的 Data Cache 模拟器，将两个工具同时显示在屏幕中，运行程序，同时观察两个工具的输出。我们应该可以观察到这两个工具可以并行的独立工作。

7. 汇编程序设计

独立设计一个生成“2 的 n 次方 (n=0,1,2,...)”数列的算法，并通过 Run I/O 输出。

要求：将数列的前 16 个元素打印输出；



作业中需要提交的文件：汇编程序文件.asm、汇编程序机器码文件、数据段结果文件。

另外，请独立构思和设计一个汇编程序，提交上述文件以外，还需提交你的程序的功能说明。