**深 圳 大 学 实 验 报 告**

**课程名称： 计算机系统(3)**

**实验项目名称： 实验一：MIPS指令集实验**

**学院： 计算机与软件学院**

**专业： 软件工程（腾班）**

**指导教师： 李琰**

**报告人： 叶茂林 学号： 2021155015 班级： 腾班**

**实验时间： 2023.10.6**

**实验报告提交时间： 2023.10.8**

**教务部制**

|  |
| --- |
| 实验目的与要求： 一、实验目的 了解WinMIPS64的基本功能和作用；  熟悉MIPS指令、初步建立指令流水执行的感性认识；  掌握该工具的基本命令和操作，为流水线实验作准备。 |
| 方法、步骤： 二、实验步骤及相关说明 按照下面的实验步骤及说明，完成相关操作**记录实验过程的截图**：  1）下载WinMIPS64；运行样例代码并观察软件各个观察窗口的内容和作用，掌握软件的使用方法。（80分）  2）学会正确使用WinMIPS64的IO方法；（10分）  3）编写完整的排序程序；（10分） |
| 实验过程及内容： 三、实验内容 **（1）下载WinMIPS64；运行样例代码并观察软件各个观察窗口的内容和作用，掌握软件的使用方法。（80分）**  打开WinMIPS64模拟器，如图1所示。    图1  打开Configure菜单点击Architecture可以修改配置，如图2所示。    图2  新建一个文件sum.s，内容如图3所示。    图3  用asm.exe检验该输入程序的语法正确性，如图4所示，无语法错误。    图4  将sum.s文件装入主存，如图5所示。    图5  在Pipeline窗口中，如图6所示，展示了MIPS64处理器的内部结构，其中包括了MIPS64的五级流水线和浮点操作（加法/减法，乘法和除法）的单元，展示了处于不同流水段的指令。    图6  在Code窗口，如图7所示，可以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，初始时，第一行为黄色，表示该行指令处于“取指”阶段。    图7  点击Execution开始模拟，点击Single Cycle或按 F7键，如图8所示，第一行变成了了蓝色，第二行变成了黄色，这表示第一行指令处于“译码”阶段，而第二行指令处于“取指”阶段。    图8  这些不同的颜色代表指令分别处于不同的流水线阶段。黄色代表“取指”，蓝色代表“译码”，红色代表“执行”，绿色代表“内存数据读或写”，紫色代表“写回”。  接着按F7，第五个时钟周期的时候，“dadd r3，r4，r5”指令没有从“译码”跳到其下一个流水阶段“执行”，并且“sd r3，C(r0)”指令，仍然停留在“取指”阶段，同时在terminal窗口显示一行信息“RAW Stall in ID (R5)”，如图9所示。    图9  在Cycls窗口上显示着流水线的时空图，如图10所示，模拟正在第五时钟周期，第一条指令正在WB段，第二条命令在MeM段，第四条命令在处于暂停状态（installed），第五条指令也因此停滞不前，因为第四条指令的dadd命令需要用到寄存器r5的值，但是r5的值并不可用。    图10  接着点击F7，直至第十三个时钟周期全部指令执行结束，如图11所示，Cycls窗口左边的子窗口是一系列的指令，右边的窗口是图示的指令执行过程。其中，左边窗口的命令是动态出现的，当一条指令在进行“取指”时，该指令才出现，而且，当出现了数据相关的时候，所涉及到的指令会变色，暂停的指令会变成蓝色，而被其影响的后续指令会变成灰色。    图11  在Data窗口中可以看到内存中的数据，如图12所示，左边一行即为用64位表示的内存地址，中间行为数据的内容，右边的一行为相关的代码，如果想改变一个整型的数据的值，左键双击该值所在的行，如果是想改变一个浮点类型的数据的值，那么右键双击该值所在的行。    图12  在Register窗口中可以观察到寄存器中的值，如图13所示，如果该寄存器为灰色，那么它正处于被一条指令写入的过程，如果它用一种颜色表示，那么就代表，该颜色所代表的的流水线阶段的值可以用来进行前递（forwarding）。同时，这个窗口允许你交互式的该变寄存器的值，但是前提是该寄存器不能处于被写入或者前递的阶段。如果想改变一个整型的数据的值，左键双击该值所在的行，如果是想改变一个浮点类型的数据的值，那么请右键双击该值所在的行，然后按OK来进行确定。    图13  在Statics窗口中可以观察到模拟周期的统计数据，如图14所示，Execution用来显示模拟周期中指令数，执行周期数和CPI（每条指令所用周期数），Stalls用来表示暂停的周期数，并且分门别类的进行了统计，其中包括RAW Stalls，WAW Stalls，WAR Stalls, Structural Stalls, Branch Taken Stalls和Branch misprediction Stalls。Code Size表示了代码的大小，用byte表示。    图14  点击File/Reset MIPS64（ctrl + R）可以进行重置而不删除内存中的数据，如果想要删除内存中的数据可以点击File/Full Reset。  可以一次推进多个时钟周期，方法是点击Execute/Multi cycle（F8），而多个时钟周期数是在Configure/Multi-step中设置的。也可以通过按F4一次完成整个程序的模拟。同时也可以设置断点，方法是，在Code窗口中左键双击想要设置断点的指令，该指令会变成蓝色，然后点击F4，程序就会停在这条指令执行“ 取指”的阶段，如果想要清除断点，再次左键双击改行指令。  **（2）学会正确使用WinMIPS64的IO方法；（10分）**  编写一个简单的终端输出“Hello World！！”的小程序，首先写好一些数据包括CONTROL和DATA的地址以及字符串Hello World，然后将CONTROL和DATA的地址存储在寄存器中以之作为基址，将字符串首地址写入DATA，将4写入CONTROL，如图15所示。    图15  用asm.exe检验该程序的语法，如图16所示，语法无误。    图16  用WinMIPS64打开该程序，如图17所示。    图17  运行该程序，如图18所示。    图18  其中terminal窗口如图19所示，说明我们的程序成功打印出来Hello World。    图19  **（3）编写完整的排序程序；（10分）**  课本上的swap函数和sort函数对于排序这64位的整数是不适合的，而且也有些许的bug，因此我选择自己重写这两个函数。  首先先初始化一些数据，如图20所示，包括需要排序的数据和数据的长度以及需要输出的字符串和CONTROL和DATA的地址。    图20  先写一个swap函数，大体上与课本上的相同，不过我加上了所对使用寄存器的栈保存，如图21所示。    图21  然后写sort函数，同样是要注意使用栈来保存所使用到的寄存器的值，特别是寄存器r31，即$ra的值，其中保存着函数调用的返回地址，然后便是两层循环加一个判断跳转swap函数，如图22所示。    图22  然后再写一个输出数据的函数，因为要输出两次数据，所以写成一个函数比较方便，如图23所示。    图23  最后是主函数的编写，先初始化栈指针寄存器的值为内存最高地址，然后在开始和结束都输出一次字符串和调用一次输出函数，中间调用一次排序函数，如图24所示。    图24  使用asm.exe检验所写的程序语法正确性，如图25所示，无语法错误。    图25  运行程序，结果如图26所示，证明程序成功对数组排序。    图26 |

|  |
| --- |
| 实验结论：  在本次实验中，我在下载WinMIPS64和运行样例代码的过程中，了解了该软件的各个观察窗口的内容和作用，通过学习这些窗口的内容和作用，我能够更加全面地理解和掌握软件的使用方法。  然后学会了正确使用WinMIPS64的IO方法。在汇编语言中，IO是一种非常重要的操作，可以通过简单的输入输出语句实现与用户交互或者与外部设备通信。通过参照内存映射，我了解了如何使用WinMIPS64实现标准输入输出，包括读取字符、读取字符串和打印字符串等操作。这为我设计编写程序提供了很大的便利。  最后，我成功地编写了一个完整的排序程序。在编写过程中，我深入理解了冒泡排序算法的逻辑和思路，并将其转化为汇编语言的形式。在编写过程中，我发现了很多逻辑错误和语法错误，并通过不断地调试和修改最终得到了正确的结果。通过这一过程，我深刻体会到了汇编语言的调试过程和优势，即每一条指令都可以单独执行和检查，使得程序的调试效率更加高效。  通过本次实验，我掌握了MIPS汇编语言的基本知识和使用方法，深入理解了算法的逻辑和思路，在实践中积累了丰富的经验，对我的计算机科学学习和研究将有着长远的影响。 |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。