华中科技大学 计算机学院

[硬件综合训练]

[课程设计任务书 V5.0]

课程设计任务书

计算机组成原理是计算机专业的核心基础课。本课程力图以“培养学生现代计算机系统设计能力”为目标，贯彻“强调软/硬件关联与协同、以 CPU 设计为核心/层次化系统设计的组织思路，有效地增强对学生的计算机系统设计与实现能力的培养”。课程设计是完成该课程并进行了多个单元实验后，综合利用所学的理论知识，并结合在单元实验中所积累的计算机部件设计和调试方法，设计出一台具有一定规模指令系统的简单计算机系统。所设计的系统能在 LOGISIM 仿真平台和 FPGA 实验平台上正确运行，通过检查程序结果的正确性来判断所设计计算机系统正确性。

课程设计属于设计型实验，不仅锻炼学生简单计算机系统的设计能力，而且通过进行中央处理器底层电路的实现、故障分析与定位、系统调试等环节的锻炼，进一步提高学生分析和解决问题的能力。

1．课程设计题目

5 段流水线 CPU 的设计

2、简单计算机系统的设计目标

本课程设计的总体目标是利用 FPGA 以及相关外围器件，设计五段流水 CPU，要求所设计的流水 CPU 系统能支持自动和单步运行方式，能正确地执行存放在主存中的程序的功能，对主要的数据流和控制流通过 LED、数码管等适时的进行显示，方便监控和调试。尽可能利用 EDA 软件或仿真软件对模型机系统中各部件进行仿真分析和功能验证。在学有余力的前提下，可进一步扩展相关功能。

3、主要技术指标

1) 支持规定的 32 位 mips/risc-v 指令集（指令集任选），具体见表 1；
2) 在扩展指令集中支持 2 条 C 类运算指令，1 条 M 类存储指令，1 条 B 类分支指令，具体任务每位同学不一样，任务要求由指导教师制定；
3) 支持多级嵌套中断，利用中断触发扩展指令集测试程序；
4) 支持 5 段流水机制，可处理数据冒险、结构冒险、分支冒险；
5) 能运行由自己所设计的指令系统构成的一段测试程序，测试程序应能涵盖

所有指令，程序执行功能正确。

6) 能运行教师提供的标准测试程序，并**自动统计执行周期数**

7) **能自动统计各类无条件分支指令数目，条件分支成功次数、插入气泡数目、load-use 冲突次数、动态分支预测流水线能自动统计预测成功与失败次数。**

表 1 基本指令集

#	MIPS	RISC-V	简单功能描述	备注
1	ADD	ADD	加法	指令格式参考 MIPS32/RISC-V32 指令集，最终功能以 MARS/RARS 模拟器为准。 注意标准 MIPS 指令集中采用了延迟槽技术，分支指令在译码段执行，所以 JAL 指令返回地址是 PC+8，课程设计没有延迟槽技术，所以遇到 PC+8 的功能全部修改为 PC+4
2	ADDI	ADDI	立即数加	
3	ADDIU		无符号立即数加	
4	ADDU		无符号数加	
5	AND	AND	与	
6	ANDI	ANDI	立即数与	
7	SLL	SLLI	逻辑左移	
8	SRA	SRAI	算数右移	
9	SRL	SRLI	逻辑右移	
10	SUB	SUB	减	
11	OR	OR	或	
12	ORI	ORI	立即数或	
13	NOR	XORI	或非/立即数异或	
14	LW	LW	加载字	
15	SW	SW	存字	
16	BEQ	BEQ	相等跳转	
17	BNE	BNE	不相等跳转	
18	SLT	SLT	小于置数	
19	SLTI	SLTI	小于立即数置数	
20	SLTU	SLTU	小于无符号数置数	
21	J		无条件转移	
22	JAL	JAL	转移并链接	
23	JR	JALR	转移到指定寄存器	
24	SYSCALL	ECALL	系统调用	If $v0/a7==50 暂停 等待按键 else 数码管显示$a0 值
25	MFC0	CSRRSI	访问 CP0/CSR 寄存器	中断相关，可简化为开中断
26	MTC0	CSRRCI	访问 CP0/CSR 寄存器	中断相关，可简化为关中断
27	ERET	URET	中断返回	异常返回，选做

表 2 扩展指令集

指令分类	编号	MIPS	RISC-V	备注
运算 (C)	1	SLLV	SLL	指令格式参考 MIPS32/RISC-V32 指令集，最终功能以 MARS/RARS 模拟器为准。
	2	SRLV	SRL	
	3	SRAV	SRA	
	4	XOR	XOR	
	5	XORI	AUIPC	
	6	LUI	LUI	
	7	SLTIU	SLTIU	
	8	MULTU	MUL	
	9	DIVU	DIVU	
	A	MFLO	REM	
存储访问 (A)	1	LB	LB	
	2	LBU	LBU	
	3	LH	LH	
	4	LHU	LHU	
	5	SB	SB	
	6	SH	SH	
跳转 (B)	1	BLEZ	BLT	
	2	BGTZ	BGE	
	3	BLTZ	BLTU	
	4	BGEZ	BGEU	

4、系统设计要求

1) 根据课程设计指导书的要求，制定出设计方案；
2) 分析指令系统格式，指令系统功能。
3) 根据指令系统构建基本功能部件，主要数据通路。
4) 根据功能部件及数据通路连接，分析所需要的控制信号以及这些控制信号的有效形式；
5) 设计出实现指令功能的硬布线控制器；
6) 调试、数据分析、验收检查；
7) 课程设计报告和总结。

5．课程设计成绩的评定

1) 评定成绩根据考核、课程设计的过程、课程设计的效果、课程设计报告的质量等几部分组成；
2) 评分标准为设计过程占 65%，团队作业 15%，报告和图纸部分占 20%；
3) 课程设计的成绩评定等级为不及格、及格、中、良好、优秀五级，具体的评定标准见评分规则；
4) 对基本功能进行扩展或设计具有非常鲜明的特征和一定程度的创新，可根据实际情况加分。
5) 鼓励团队协作，小组杰出贡献奖和优秀奖有加分，各班按团队进行天梯赛，各班天梯赛前两名有加分。

6、对课程设计报告的要求

1) 课程设计报告是体现和总结课程设计成果的载体，主要内容包括：设计题目、设计目的、设备器材、设计原理及内容、设计步骤、遇到的问题及解决方法、设计总结、参考文献等。
2) 在适当位置配合相应的实验原理图、数据通路图等图表进行说明。应做到文理通顺，内容正确完整，书写工整，装订整齐。
3) 设计总结部分主要写本人工作简介以及设计体会，包括通过课程设计学到了什么，哪里遇到了困难，解决的办法以及今后的目标。
4) 为统一格式和要求，课程设计报告要求采用《计算机组成原理》专用设计报告模板，采用 A4 纸双面打印（教师不要求提交纸质版除外）。为减轻报告撰写工作量，关注报告书写质量，报告总页码不允许超过 40 页。

7．课程设计时间安排

课程设计的总体时间为 2 周，总体安排如下:

1) 第 1 天：到实验室布置任务和集中讲解，领取开发设备。
2) 第 1~3 天：学生查阅资料，开始方案设计。
3) 第 4 天：中期进度检查，单周期 CPU 验收检查，文档检查。
4) 第 6~10 天：阶段性成果随时检查。
5) 第 10 天：最终结果验收。

验收采用分步检查验收方法，以适合各种层次的学生，不断提高学生动手能

力。验收辅以答辩的方式，**抄袭被抄袭均按零分处理**。

参考文献：

[1] DAVID A.PATTERSON(美).计算机组成与设计硬件/软件接口(原书第 4 版).北京：机械工业出版社.

[2] David Money Harris（美）. 数字设计和计算机体系结构（第二版）. 机械工业出版社

[3] 谭志虎，秦磊华，吴非，肖亮.计算机组成原理. 北京：人民邮电出版社，2021 年.

[4] 张晨曦，王志英. 计算机系统结构. 高等教育出版社，2008 年.

[5] 张志刚，FPGA 与 SOPC 设计教程-DE2 实践. 西安：电子科技大学出版社，2007

课程设计成绩评分标准

课程设计评定成绩根据平时成绩、课程设计的过程、结果、课程设计报告的质量等几部分组成；设计过程和结果占65%，团队作业15，课程设计报告20%；对基本功能进行扩展或设计具有非常鲜明的特征和一定程度的创新，可根据实际情况加分。旷课达4天、或不交设计报告者，课程设计按0分计。

1. 课程设计过程与结果评分标准

- 完成支持单级中断的单周期CPU设计与实现（LOGISIM平台）： 50分
- 实现单级中断（LOGISIM平台）： +5分
- 实现多级嵌套中断（LOGISIM平台）： +5分
- 实现单周期上开发板（FPGA平台）： +5分
- 完成理想流水线的多周期CPU（LOGISIM平台）： 55分
- 完成气泡流水线（LOGISIM平台）： 65分
- 完成重定向流水线（LOGISIM平台）： 75分
- 完成流水中断机制（LOGISIM/FPGA平台）： +5分
- 实现流水线上开发板（FPGA平台）： +5分
- 扩展完成动态预测方式的分支冒险处理（FPGA平台）： +10分

2. 团队任务（团队互评）

团队开发一个具有展示度的演示系统，有输入输出设备，完整的软硬件系统

3. 课程设计报告评分标准

- 报告规范，清楚表述设计思想、设计思路、设计过程、设计结果，文档资料完整，书写和画图规范： 20分
- 满足课程设计报告格式要求，能较清楚的表述设计思想、设计思路、设计过程、设计结果，文档资料较完整，书写和画图规范性较好： 16分
- 满足课程设计报告格式要求，基本能表述设计思想、设计思路、设计过程、设计结果，文档资料完整一般，书写和画图规范性一般： 14分
- 报告不规范，内容不完整，不能体现设计原理、方法和自己的工作： 10分

- 严重不规范、内容空洞，完全不能体现课程设计的内容：　　5 分
- 课程设计报告抄袭　　0 分

课程设计指南

在本次课程设计，需要使用 LOGISIM 来创建一个 32-位 5 段流水 CPU， 该 CPU 运行的是标准 MIPS 指令集的子集，请仔细阅读本文档并查阅 mips 手册。

1. 单周期 CPU 设计流程

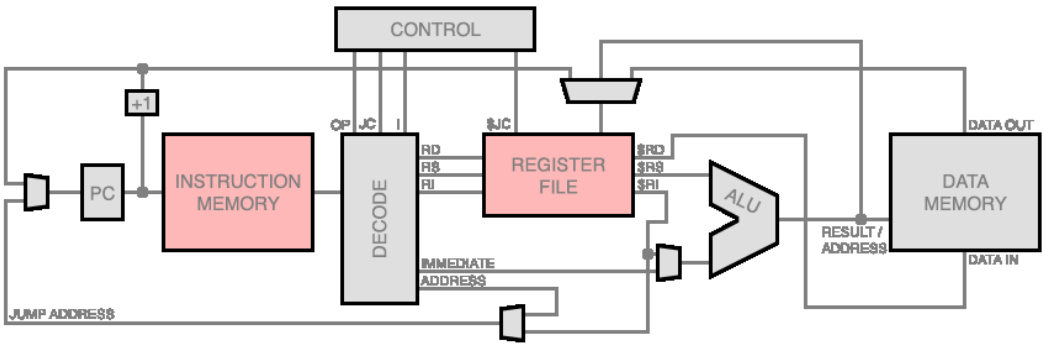

图 1 单周期 CPU 示意图

1) 做一下相关的实验,了解 LOGISIM 的一些模块和组件,如 RAM 和显示单元,尽量深入地理解相关模块的用法及原理。
2) 制作一个 ALU,该 ALU 应支持课程设计所要求的指令系统中所有的算术逻辑运算功能（此部分工作在课程实验中已经完成）。**考虑后续设计日益复杂，封装应该尽可能小，可以自己调整实验中的 ALU 封装模块）**
3) 制作一个寄存器组(也称寄存器文件)模块。具体参见实验二文档。
4) 可以制作取下一条指令的逻辑,即 PC(程序计数器)的逻辑,最初你可以简单的实现,只用让该寄存器的值,在每次时钟信号到达时加一即可，以后再考虑更复杂的情形。
5) 也许此时你可以用上面构建的模块，搭建一个最简单的 CPU 了，该 CPU 只能进行加法运算。你可以先做一个大致的模型，实现加法。而我们希望是，你的 CPU 应该包括以下器件：

- ALU、寄存器文件
- PC 及 PC+4 的逻辑。
- 指令内存（为了简单，建议你使用系统提供的 ROM，而不是 RAM，ROM 可以方便的加载镜像，重启后也不需要单独加载，RAM 重启程序，或者 CTRL+R 后数据清零），由于 mips 32 位地址总线是按字节编址，而系统提供的 ROM 是 32 位的，且 ROM 地址总线也无法达到 32 位，所以连接时可以将高位地址屏蔽，低位字节偏移地址也进行屏蔽，使得取指令工作能正常进行。

试着将以下汇编指令翻译成机器码：

nor $s0, $0, $1

然后将机器码存入指令 ROM 中。如果你成功地完成了此最简单的 CPU，则时钟每跳一次，将执行一条指令。

6) 你可以进一步编写一些新的 CPU 指令，如 SUB，AND，OR 等，来对你前面做的工作进行一些进一步的测试。当然，对于 SUB，AND，OR，你可能需要修改 ALU 的相应控制，否则，还是在做 nor。
7) 你可以编写更多的模块，如零扩展和符号扩展，然后将其组合成一个通用的扩展器，这样就可以实现立即数的运算了。
8) 你可以加入 LOAD 及 STORE 逻辑。当然，这里需要加入数据内存了。
9) 不同 mips 指令的数据通路，大家可以参考 mips 仿真器 MARS 中的 x ray 功能观看，该仿真器可以动态演示指令的数据通路以及对应的控制信号生成。

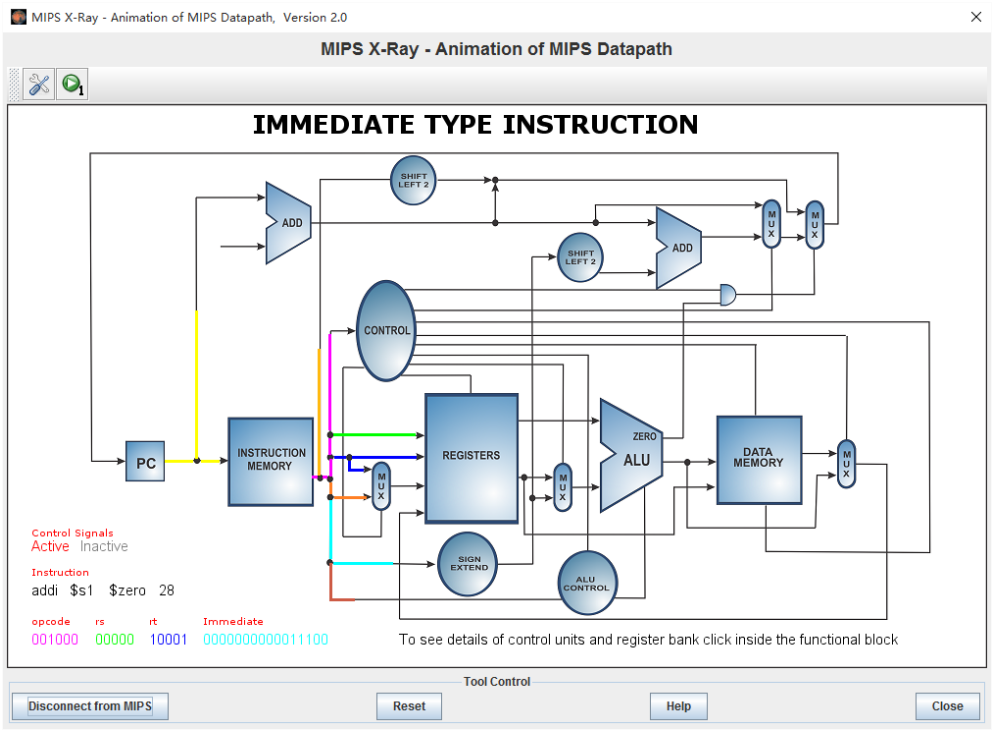

图 2 显示部件示意图

10) 也许你可以把所有的工作集成在一起，完成 DATAPATH（数据通道）了，注意在进行数据通路集成时会发现很多输入有多个来源，此时需要加入多路选择器，选择器的控制端成为新的控点或微命令。
11) 根据前述步骤完成的数据通路以及对应的控点，设计控制器，控制器应该进行封装，控制器封装完毕你的单周期 CPU 就完成了（最好将每个控点的逻辑表达式写出来，可以给出最小项之和，LOGISIM 可以帮你化简）。

2. 五段流水 CPU 设计流程

1) 首先完成单周期 CPU，具体参考单周期 CPU 设计流程。
2) 完成支持理想流水线的多周期 CPU。将指令过程分成 5 个阶段 IF ID EX MEM WB(不得简化成 4 段流水线)，**建议分支指令在 EX 段完成，**不同阶段之间设置缓冲接口部件，构建各阶段之间的接口部件，接口定义尽可能简化，流水线应向后续段传递数据信息，控制信息，向前段传递反馈信息，后续部件对数据的加工处理依赖于前阶段传递过来的信息。ID 段译码生成该指令的所有控制信号，控制信号将逐段向后传递（越到最后阶段，信号越少），后续部件控制信号不再单独生成。**注意单周期 CPU 中的控制器是可以在 ID 段复用的。**
3) 增加流水冲突检测器。要思考流水冲突检测器在那个阶段？不同类型的指令数据相关性的区别，Load-Use 相关如何检测？相关检测器如何封装，输入输出是什么？**（问题很多，很多内容课程中并未有讲述，请大家大量阅读参考文献），设计时请注意增加计数器正确统计冲突，气泡等关键指标。**
4) 增加流水冲突处理机制
   - 首先实现气泡机制的流水线。如何插入气泡，接口是否需要修改，如果需要修改，应该在步骤 2 中就考虑清楚，分支相关和数据相关时 CPU 如何插入气泡，如何保证流水线功能的正确性？
   - 然后实现采用数据重定向机制的流水线。需要思考在那个阶段完成数据重定向，那个阶段进行检测，答案可能不是唯一的。数据重定向的数据来源来自哪里？采用数据重定向机制后是否还需要插入气泡？Load-Use 冲突如何解决？

3. 中断机制实现

- **实现要求：**引入若干中断源，LOGISIM 中的按键，编号 1,2,3。。。每个按键对应一个中断源，比如按键 1 按下后进入数字 1 的走马灯，按键 2 按下后进入数字 2 的走马灯，按键 3 按下后进入数字 3 的走马灯。（中断演示程序待定,或者学生自己编写),3 个中断源对应不同的中断优先级,其中 3>2>1，支持多级中断嵌套。
- **实现原理：**MIPS 中断控制一般通过协处理器 CP0 完成，有两条指令负责 CP0 通用寄存器的读写（MFC0，MTC0），中断使能位 IE，中断屏蔽位均可以利用 CP0 完成,可以按需要设计实现中断硬件,能完成中断即可,不一定参考 MIPS CP0 具体实现。需要考虑的问题: 中断识别问题，可参考书上的中断仲裁电路。中断隐指令的实现（PC 压栈，中断识别，中断服务程序入口地址送 PC，参考书上的流程图），压栈涉及的访存操作，中断识别寻找入口地址涉及访存操作，至少需要两个周期，这个过程中 PC，以及其他功能部件如 ALU，

REGFILE 等如何动作？堆栈寄存器需要在主程序初始化，放在哪个地址合适，中断服务程序放在那里合适，现场包括哪些内容、流水线中断如何处理？

4. 指令集结构 Instruction Set Architecture (ISA)

需要实现一个 MIPS32-位处理器，32 位指令字，具体指令除 syscall 外全部参考 mips 指令集规范，请大家按附件包中的参考文献查询指令功能和指令格式. 该处理器具有独立的数据和指令内存(即有两个内存,一个指令内存,一个数据内存)。

重要注意事项： 由于 LOGISIM 的限制，也为了让事情更简单一些，我们以 32 位为单位对内存编址，这和 MIPS 不同，MIPS 指令是字长是 32 位,而内存是以字节为单位编址，具体进行地址连接时要考虑屏蔽掉字节地址。

注意:全零的指令(0x0000000)为 NOP，其指令含义是 sll $0,$0,0

显示指令:每个计算机都由五个部分组成：控制，数据通道，内存，输入和输出. 相应的，在本项目中指令系统中，增加了一条 syscall 指令，该指令在$v0 寄存器不等于 10 时，将寄存器$a0 的值在特定显示窗口显示。设计时需要包含 8 个 16 进制发光二级管作为显示输出,如图 3 所示，具体显示时调用如下指令

```
addi $v0,$0,1
syscall      #v0!=10 display $a0
```

设计显示指令逻辑时需要考虑**如何锁存过去的数据，否则数据一闪而过**。

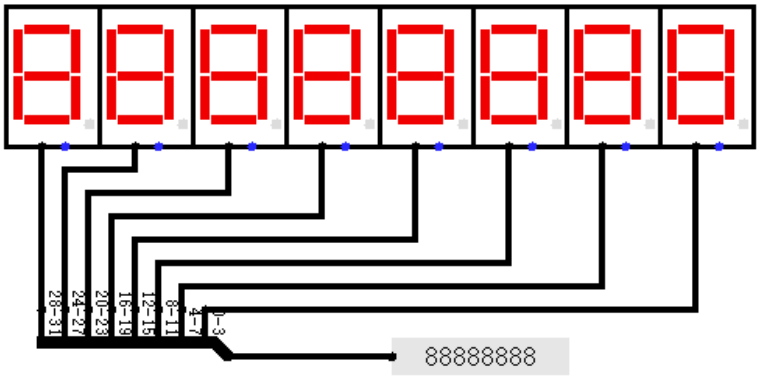

图 3 显示部件示意图

另当$v0 等于 10 时，系统自动停机。

```
addi $v0,$0,10
syscall      #v0==10  halt/stop clk
```

设置以上指令的原因是为了让你的程序能停下来，并能保持和汇编器的兼容。注意统计周期数时停机指令也需要统计，很多同学会漏掉这个，不能简单加 1。

思考一下，mips 中有 I/O 指令吗？ Mips 实际如何实现输入输出设备的访问？

5. 汇编器

汇编器采用附件包中的 MARS 仿真器，该仿真器功能强大，请主动学习之。

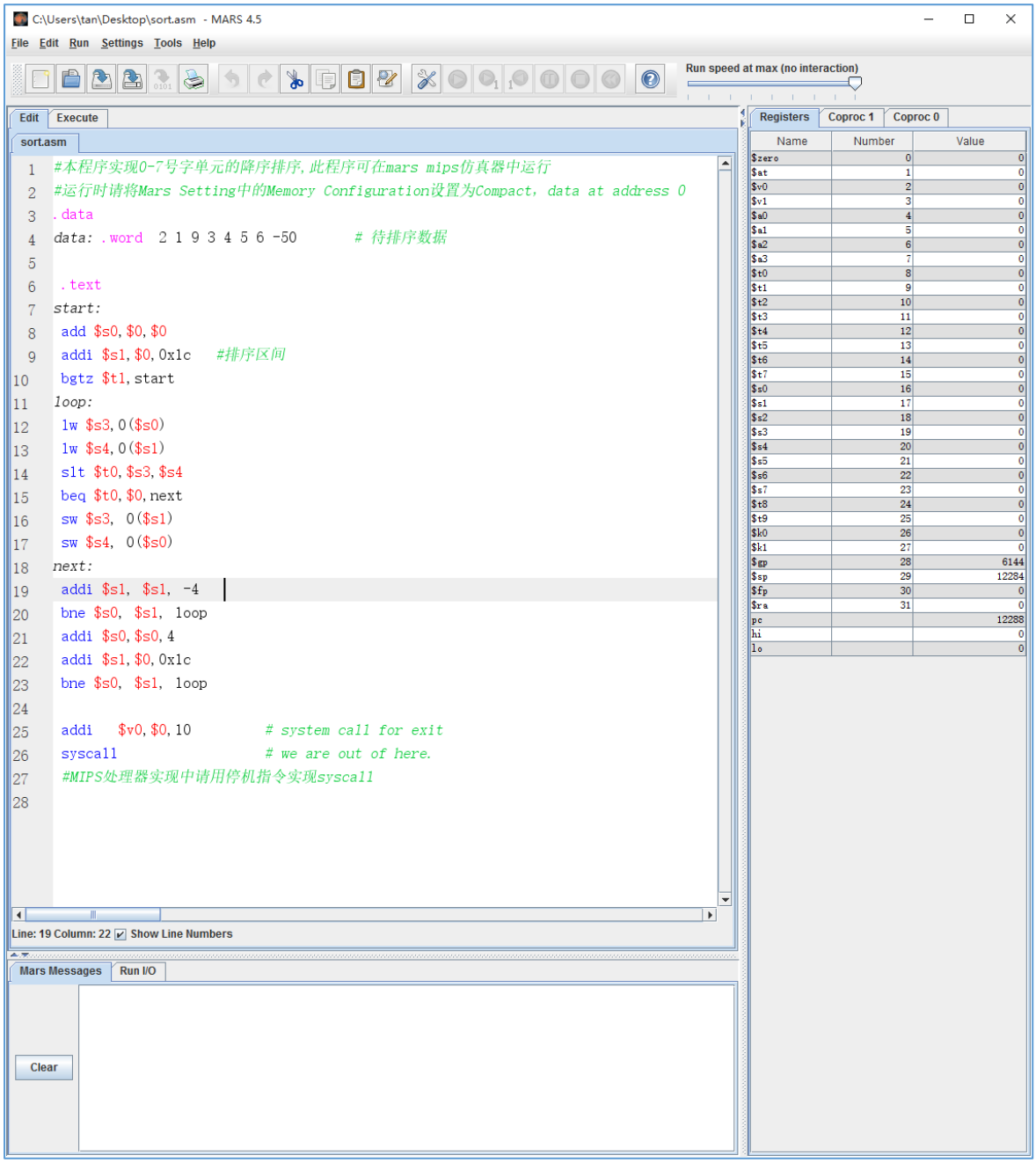

图 4 MARS 仿真器

注意为了能让 MARS 中汇编的机器码能在 LOGISIM 中使用，需要设置 MARS 界面中 setting 的 Memory Configration，将内存模式设置为下图的模式，这样数据段起始位置就是 0 开始的位置。

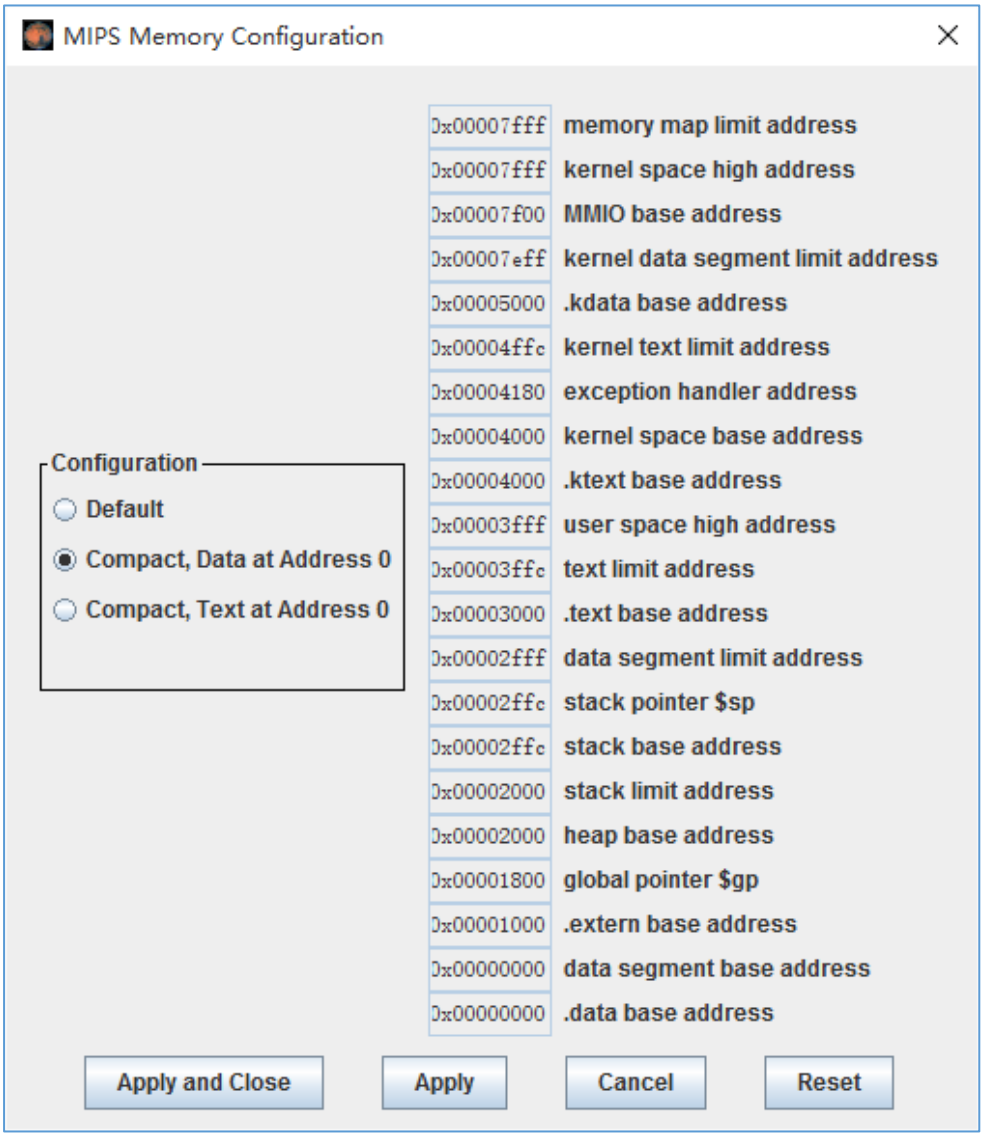

图 5 MARS 仿真器内存模式设置

程序汇编和后可以利用 File 菜单中的 Dump Memory 功能将代码段和数据段导出，采用十六进制文本的方式导出到某个文本文件，然后在文件第一行加入“v2.0 raw”即可在 LOGISIM 中加载到 ROM 或 RAM。

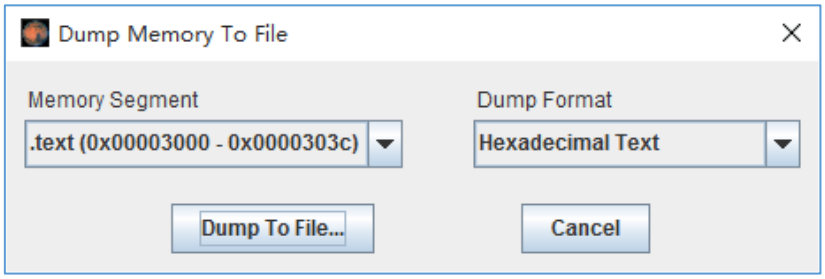

图 6 MARS 内存导出

6. LOGISIM

LOGISIM 使用非常方便，但本身存在 bug，尤其是大面积电路移动是会出问题，**你需要经常按下 ctrl+s 保存你的.circ 文件**，并注意版本管理，我们将使用的 LOGISIM 官方版本是 v2.7。使用 LOGISIM 的过程中如果出现奇怪的问题，可以重启 **LOGISIM!** 不必浪费时间跟踪 bug，这不是你的错。但如果重启后还不能解决问题，则极有可能该 bug 是由你的代码引起的！出现此情况，请注意调试。

尽量**减少对时钟信号的逻辑操作**，否则可能会引发险象和毛刺，导致无法预料的逻辑错误。**所有带状态的部件请尽可能设计统一复位信号，便于系统复位调试。**

7. 测试

1.简单单元测试

在你完成了 CPU 设计后，可以编写程序在 CPU 上运行以测试 CPU 能否工作!建议大家编写多个小程序来分别单独测试其他指令，资料包中也包含了若干测试子程序，大家可以根据需要使用或自己编写程序进行测试。

2.复杂程序测试

- 将课设资料包中提供的标准测试程序 benchmark.asm 利用汇编器汇编成机器指令 benchmark.hex 分别加载到 ROM。该程序的功能将遍历所有指令，请注意数码管显示结果是否正确，改程序最终将 0~15 号内存单元的数据按降序排序。观察内存数据排序是否正确。请注意不同版本的 CPU 均用该程序测试，**另要求使用计数器计数该程序运行完毕所需的时钟周期数（请设置辅助电路完成）。**
- 自己编写测试程序测试 benchmark.asm 不包括的指令，编写你认为更疯狂更有展示度的程序。

8. 注意事项

- 本课程设计的内容基本无法在一本教科书中弄清楚，请尽可能的仔细阅读资料包中的参考文献。
- 在实现 CPU 时，你可以使用任何 LOGISIM 内建的电路组件。
- 指令 ROM 和数据 RAM 必须在 main 电路中可见，不能封装在子电路中。
- 显示模块应该在主电路中可见。
- 控制器必须用逻辑表达式生成，所有信号均应给出对应逻辑表达式，避免使用比较器实现控制器（不允许采用微程序方式实现控制器）。
- 主要部件之间还是需要适当连线，隧道工具不能过度使用，要能看清楚各部件之间的连接关系。

- 尽可能的使用标签工具去注释你的电路，包括控制信号，数据通路，显示模块，总线等，这会让你的电路更加容易调试！
- 注意标签以及注释的命名规范，过长的命名都会对后续的画图连接造成影响。
- LOGISIM 中可以将不同的模块用不同的颜色区分，建议用颜色区分各接口部件和关键模块。
- 接口部件封装尽可能封装的长一点，各接口部件建议等长，否则控制线多了以后可能不方便布线。
- 最终的流水 CPU 图相对较为复杂，请注意画出整齐整洁的原理图，各段之间连接尽量用连线表示，有利于结构的清洗，适量的隧道避免线缆的大量交叉。
- PC，IR 最好一直传递到最后一级，这样方便观测流水线运行的状况。
- 流水线各级是否产生气泡可以用 LED 指示灯显示，方便观察流水线运行状况。
- 先在 LOGISIM 仿真平台上将原理跑通后再上 FPGA 开发板。
- 各里程碑版本经过充分测试后，备份后再开新的分支进行新的开发，以避免新版本无法开发成功，老版本又检查不了的悲剧。（**请勿在公开平台上发布自己电路和设计成果，一经发现，取消成绩**）

9. 项目检查

完成你认为的最终版本后再一次性验收，各里程碑版本需要指导教师查阅给出指导意见后开始新的版本。

10. 成果提交

按班为单位提交电子版本压缩文件，解压后每人一个目录（不是压缩文件）

- **目录命名规范**

  班号_学号_姓名　例子： CS1201_U201214795_李珍帅
- **专业命名规范**

  计算机 CS　卓越班　ZY　本硕博 BSB　ACM 班　ACM　物联网 IT
- **目录内容**

  CS1201_U201214795_姓名_硬件综合训练课程设计报告.docx

  CS1201_U201214795_姓名_硬件综合训练课程设计报告.pdf （两版本均要）

  CS1201_U201214795_姓名_LOGISIM_FPGA.zip/rar （程序文件，代码等）

  CS1201_团队名_团队任务名.zip/rar （程序文件，代码，电路，演示 ppt）

我们将使用链接自动打开你的报告文档，实验报告命名不规范的将寻找不到，会被扣 2 分，请一定注意命名规范！

- 电子版按班为单位集中打包发送至 130757@qq.com 谭志虎老师处归档
- 请学位给出一个报告收缴情况的清单，注明没有上交电子版的同学