

一、概述

1.1 初赛/决赛指标

功能测试:

	得分
一级评测	100
二级测评	100
三级测评	100
性能测试	100

性能测试:

	运行时间
STREAM	0.074 s
MATRIX	0.109 s
CRYPTONIGHT	0.260 s

1.2 设计概述

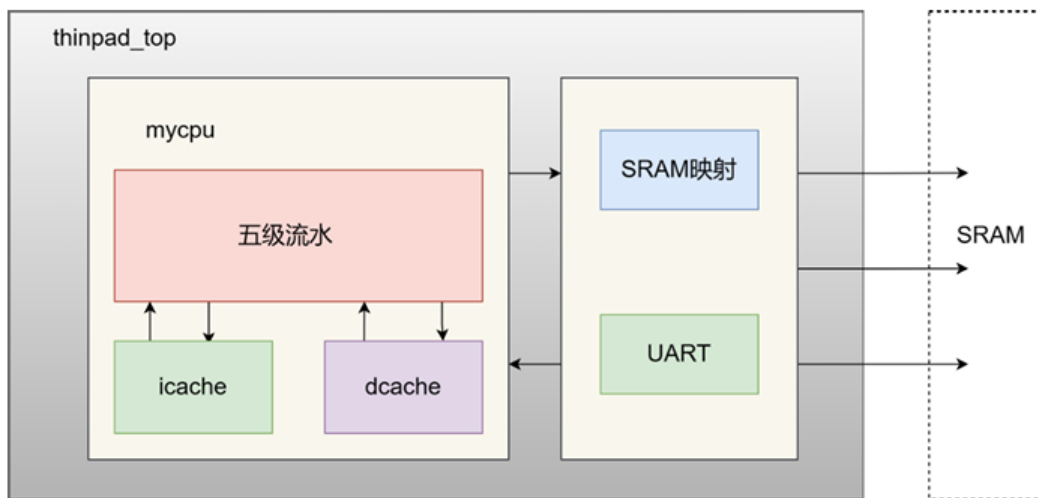
该作品是基于 MIPS32-R2 的一款添加了 cache 的单发射五级流水线 cpu，实现了比赛要求的 MIPS-C1、MIPS-C2、MIPS-C3 指令集中的所有指令，完成了要求的 SUB, SRAV, BGEZ 三条随机指令测试。数据相关问题采用了数据前递的方法解决，Load-Use 冒险采用流水线暂停的方式解决，BaseRAM 结构冒险采用 SRAM 控制器仲裁解决，该 cpu 支持延迟槽，解决了分支跳转的问题。该作品目前可以通过三级功能测试和性能测试，时钟频率最高可达到 107MHz，性能测试的时间为 0.074s、0.109s、0.260s。目前的 dcache 功能还需要继续完善，性能有望再提升。

前期没有刻意准备这个比赛，跟着学校计组课程正常学，到计组期末考结束（6 月底）开始对着《自己动手学 CPU》写 cpu 核，所有测试过完大概是 7 月中旬，后期就是在添加 cache。

二、系统设计

2.1 系统整体架构设计

该作品在经典五级流水线的基础上增加了 icache 和 dcache 模块，使得该处理器能够突破 SRAM 的性能瓶颈。另外采用了 RAM_Serial_ctrl 模块控制串口和 SRAM 的映射。以下主要针对处理器、cache、RAM_Serial_ctrl 控制器 3 个模块进行介绍。



2.2 处理器模块设计

MIPS (Microprocessor without Interlocked Pipeline Stages) 是一种经典的 RISC (Reduced Instruction Set Computer) 架构，其五级流水线结构是 MIPS 处理器的核心设计。五级流水线可以将指令的执行分为五个阶段，从而实现指令的并行执行，提高处理器的性能。该项目就是对 MIPS 经典五级流水线的复现和优化。

1. 取指阶段 (Instruction Fetch): 取指阶段负责从指令存储器中读取指令，并将其送入指令寄存器。取指阶段的主要任务是根据程序计数器 (Program Counter, PC) 中存储的下一条指令的地址，从指令存储器中读取指令，并将其存储在 IR 中。同时，PC 也需要更新为下一条指令的地址。

2. 译码阶段 (Instruction Decode): 译码阶段负责解析指令，并将其转化为相应的操作。在译码阶段，指令的操作码 (Opcode) 被提取出来，并根据操作码进行相应的操作，如读取寄存器中的数据、计算跳转地址等。此外，译码阶段还负责将指令的源操作数和目的操作数从寄存器文件中读取出来，并将其传递给执行阶段。

3. 执行阶段 (Execution): 执行阶段是五级流水线中最复杂的阶段，它负责执行指令的实际操作。执行阶段根据指令的操作码和源操作数，进行相应的运算或逻辑操作。例如，

对于算术指令，执行阶段负责 进行加法、减法、乘法等操作；对于逻辑指令，执行阶段负责进行与、或、非等操作。执行 阶段还负责计算内存地址、进行操作数的读取和写入等操作。

4. 访存阶段（Memory Access）： 访存阶段负责处理与内存的交互。在该阶段，执行阶段计算得到的内存地址将被用于读 取或写入内存中的数据。对于读数据的指令，访存阶段将从内存中读取数据，并将其传递给 下一个阶段；对于写数据的指令，访存阶段将把数据写入内存。

5. 写回阶段（Write Back）： 写回阶段负责将执行阶段计算得到的结果写回到寄存器文件中。在该阶段，执行阶段计 算得到的结果将被写入目的操作数所对应的寄存器中。这样，下一条指令就可以使用这个结 果作为源操作数来执行相应的操作。

2.3 Cache 模块设计

在本项目中，iCache（指令缓存）和 dCache（数据缓存）是两个重要的模块，用于提高指令和数据的访问效率。iCache 用于存储指令，而 dCache 用于存储数据。下面将对本项目的 iCache 和 dCache 模块的设计进行详细的介绍。

● iCache（指令缓存）的设计：

iCache 是用于存储指令的缓存模块，其设计目标是提高指令的访问效率，减少指令的访问延迟。比赛提供的工程模板 SRAM 数据线宽度只有 32 位，因此访存一次最多也只能读取 一个字，基于这个设定，本项目的 icache 设计如下：

- a) Cache 块大小：一个字。
- b) 缓存映射方式：直接映射，将指令地址直接映射到缓存行。
- c) 缓存结构：根据观察，比赛性能测试需要跑的代码长度一般在 32 个字左右，因此设计 icache 大小 $32 \times 32 = 1\text{kb}$ 。采用直接映射，每块只一个字，块内寻址 2 位，cache 地址 5 位，而 basesram 大小 4MB，地址 22 位，tag $22 - 5 - 2 = 15$ 位，另外有一位 valid 位。

● dCache（数据缓存）的设计：

dCache 是用于存储数据的缓存模块，其设计目标是提高数据的访问效率，减少数据的访问延迟。仿照 iCache 设计，dCache 设计如下：

- a) Cache 块大小：一个字。
- b) 缓存映射方式：直接映射，将数据地址直接映射到缓存行。
- c) 写策略：写直达，当 cpu 要求写，则在 cache 和 sram 中同时写，以保持 cache 和 sram 的一致性。
- d) 缓存结构：dCache 仿照 iCache 设计，大小 $32 \times 32 = 1\text{kb}$ 。采用直接映射，每块只一 个字，块内寻址 2 位，cache 地址 5 位，而 basesram 和 extsram 大小 8MB，地址 23 位，

tag 23-5-2=16 位，另外有一位 valid 位。

- 后续优化： 为了进一步提高数据的访问效率，dCache 可以采取 writebuffer 设计，目前还未实现。

2.4 RAM_Serial_ctrl 控制器模块设计

RAM_Serial_ctrl 模块：整个模块采用组合逻辑设计，主要功能为对来自 icache 和 dcache 的信息进行处理，包括内存地址映射、处理串口收发数据、以及连接 BaseRAM 和 ExtRAM 从而获取数据，对应 RAM_Serial_ctrl.v 文件

三、关键技术及亮点

Icache+双周期，频率到 107mhz，突破 sram 访存限制

四、测试及优化

4.1 测试方法

主要是用 vivado 仿真，其他方法也都尝试过，但效果不太好。遇到仿真正常，上板出错的情况，请教老师之后知道了可以先把确定正确的框架留着，其他注释掉，再一步步还原，感觉效果很好。

4.2 优化过程

第一版没加 cache 的五级单发流水只能跑到 57mhz，sram 访存是最大限制，于是想到了可以改成双周期访存，改过之后就能跑到 107mhz 了。双周期访存之后就很方便加 icache 了，加 icache 性价比很高，代码很简单，效果很不错。Dcache 因为涉及到要写入，因此逻辑上要难很多，到后面也没时间了，就没加。

五、遇到的问题及解决方案

刚开始写完五级流水线只能跑 40mhz 左右，后面参考往届作品，发现是因为时序问题，访存不稳定，了解到可以加延时，就到了 57mhz。在写串口的时候发现发送正常，接收不到，请教同学老师之后，尝试在执行串口命令的时候暂停，问题解决了，但现在还没懂是什么原因。

六、总结与展望

6.1 竞赛总结

比赛最大的遗憾是自己汇编能力太差，初赛成绩不错的情况下决赛题目都没写出来，还是应该听江老师的话早点练汇编。

6.2 未来展望

这个比赛让我对微系统架构入了门，收获还是很多的，以后有机会的话可能还会参加硬件方面的比赛。

6.3 备赛经验及建议

刚开始做这个比赛的时候，眼高手低，想做成顺序双发，花费很多时间去了解这些，后面真的自己开始写代码才发现难度很高。

总的来说这个比赛想进决赛的话，在计组学好的基础上不需要花费太多时间，但是要搞一些跟大家不一样的设计还是很有挑战性的。

最后，一定一定一定要早点练汇编！决赛尽量不要改硬件！（网站真的很慢）也是我的一个遗憾，希望之后参赛的学弟学妹们不要重蹈覆辙。