**武汉大学计算机学院**

**本科生课程设计报告**

**实验一：RISC-V引导与裸机启动**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：操作系统实践A

指 导 教 师 ：李祖超 职称

学 生 学 号 ：2023302111416

学 生 姓 名 ：肖茹琪

二○二五年九月

**郑 重 声 明**

本人呈交的设计报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计报告不包含他人享有著作权的内容。对本设计报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计报告的知识产权归属于培养单位。

本人签名： 日期：

摘 要

（黑体小2）

本次实验的核心目标是深入理解并亲手实现一个最小化操作系统的引导过程。实验参考MIT的xv6-riscv项目并进行大幅简化，聚焦于最核心的启动步骤。实验设计遵循从底层硬件初始化到上层应用逻辑的构建原则，内容主要包括：分析xv6启动流程、设计链接脚本与内存布局、编写汇编启动代码以设置运行环境、实现串口驱动作为输出通道，并在C语言环境中完成BSS段清零及主循环逻辑。最终，在QEMU模拟的RISC-V硬件平台上，通过自行编写的汇编和C代码，成功引导系统并由串口打印出启动信息，并实现了交互式回显功能。

实验结论为成功构建了一个能够正确引导、具备基本输入输出能力的最小化RISC-V操作系统内核，为后续操作系统核心功能的开发奠定了坚实基础。

**关键词：**RISC-V；操作系统；引导启动；裸机编程；QEMU

目录示例:

**目 录**（黑体小2）

**1** **实验目的和意义**

1.1 实验目的 ………………………………………………………………………1

1.2 实验意义 ………………………………………………………………………1

(各章的名称黑体4号，其余宋体小4)

**……**

**……**

**……**

**2** **实验设计**

2.1 概述……………………………………………………………………………… 35

2.2 前端设计………………………………………………………………………… 37

2.3 中端设计………………………………………………………………………… 39

2.4 后端设计………………………………………………………………………… 39

**3** **实验结论**………………………………………………………………………… 57

**4** **小组成员和分工**……………………………………………………………… 58

**……**

**……**

**……**

**参考文献 ………………………………………………………………………………** 59

**附录 ……………**………………………………………………………………………… 72

(结论、参考文献、致谢及附录黑体4号)

**1 实验目的和意义**（黑体小2）

（章标题段前为0.8行、段后为0.5行）

**1.1** **实验目的**（黑体4号）

本实验旨在通过动手实现一个RISC-V最小内核的引导过程，深入理解操作系统从硬件上电到内核初始化的完整流程。具体目标是分析xv6系统的启动设计，亲自编写汇编与C代码完成栈设置、BSS段清零和串口初始化等核心步骤，最终在QEMU模拟器上成功引导内核并实现串口输出，验证裸机启动的正确性。

**1.2** **实验意义**（黑体4号）

本实验是理解计算机系统底层工作原理的关键实践。通过裸机编程，能够打通软硬件界限，揭示硬件上电后协同工作直至软件获得控制权的完整过程。这一过程为后续学习进程、内存等复杂操作系统机制奠定了坚实基础，同时培养了从系统全局视角分析问题与实现解决方案的综合能力。

**2 实验内容**（黑体小2）

（章标题段前为0.8行、段后为0.5行）

**2.1** **任务1：理解xv6启动流程**（黑体4号）

**2．1．1** **阅读 kernel/entry.S，回答以下问题：**（黑体小4号加粗）

* 为什么第一条指令是设置栈指针？

因为栈是C语言函数调用能够正常工作的前提。entry.S 的最后一条指令是 call start，这将跳转到C语言编写的 start() 函数。在执行这条指令之前，必须已经设置好一个有效的栈空间，否则 start 函数及其内部调用的所有其他函数都将无法正确执行（无法保存返回地址、参数和局部变量），导致立即崩溃或行为不可预测。

* la sp, stack0 中的stack0在哪里定义？

在 kernel/start.c中的如下位置定义，这表示为每个CPU核心分配了 4KB（4096字节）的栈空间：



* 为什么要清零BSS段？

为了确保所有未初始化的全局变量和静态变量具有确定的初始值（零）。根据C语言标准，BSS段的变量在程序开始时必须被初始化为零。然而，为了节省磁盘空间，编译器不会在内核镜像文件中为这些零值分配空间。因此，内核自己必须在跳转到C代码之前，显式地将这一整块BSS内存清零。如果不做这一步，这些全局变量的值将是随机的，导致程序行为不可预测。

* 如何从汇编跳转到C函数？

通过RISC-V的 call 指令。汇编代码中的 call start 首先会将下一条指令的地址存入返回地址寄存器 ra，接着将pc设置为函数 start 的地址。

**2．1．2** **分析 kernel/kernel.ld，思考：**

* ENTRY(entry) 的作用是什么？

指定程序执行的入口点。当QEMU使用 -kernel 选项加载内核时，CPU会直接从 \_entry 标签处的指令开始执行。

* 为什么代码段要放在0x80000000？

1. 硬件约定：0x80000000 是RISC-V架构中操作系统内核的标准加载地址，QEMU的 -kernel 选项默认将内核镜像加载到这个地址。
2. 技术原因：
3. 地址空间布局：RISC-V将低地址空间（如0x0-0x80000000）保留 给用户程序或特定用途，内核放在高地址。
4. 内存映射：0x80000000 通常是DRAM的起始地址或内核镜像的固定 加载点。
5. 符号扩展：这个地址在符号扩展后仍然是正数，避免地址计算问题。

* etext、edata、end 符号有什么用途？

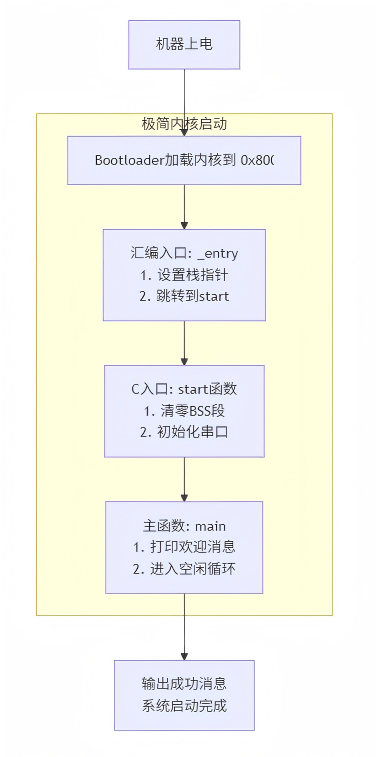
1. etext：代码段结束地址。
2. edata：已初始化数据段结束地址。
3. end：所有已分配内存的结束地址。

**2.2** **任务2~6：设计并实现最小启动流程**（黑体4号）

**2．2．1** **实验原理与设计**

* **RISC-V启动流程图**

参照xv6的启动设计，绘制的简化启动流程图如下：



* **内存布局设计**



* **关键硬件初始化**

1. 栈指针设置：为C代码执行准备环境。
2. BSS段清零：确保未初始化变量值为0。
3. 串口初始化：建立调试输出通道。

**2．2．2** **实验步骤与实现**