

## 开发环境要求

- 推荐开发平台: Ubuntu 22.04 LTS (或 WSL2)
- 基础工具: git / make / Python3 / qemu-system-riscv64 / riscv64-unknown-elf-gcc
- 学习方法: 理解原理 → 参考 xv6 → 独立实现 → 测试调试
- 进度要求: 每完成一个阶段运行测试并提交代码

## 核心学习资料

- xv6-riscv 源码: <https://github.com/mit-pdos/xv6-riscv>
- RISC-V 规范: <https://riscv.org/technical/specifications/>
  - 重点: Volume II: Privileged Specification (特权级规范)  
<https://drive.google.com/file/d/17GeetSnT5wW3xNuAHI95-SIIgPGd5sJ/view>
  - 在线版本: [https://riscv.github.io/riscv-isa-manual/snapshot/privileged/#\\_preface](https://riscv.github.io/riscv-isa-manual/snapshot/privileged/#_preface)
- xv6 手册: <https://pdos.csail.mit.edu/6.828/2025/xv6/book-riscv-rev5.pdf>

## 统一提交要求

每个实验需提交以下内容:

1. 源代码仓库 (通过 git 管理)
  - |——完整实现代码 (带注释)
  - |——Makefile
  - |——[README.md](#) (编译运行说明)
  - |——规范的目录结构
2. 综合实验报告 ([report.md](#))
  - |——系统设计部分
  - |——架构设计说明
  - |——关键数据结构
  - |——与 xv6 对比分析
  - |——设计决策理由
  - |——实验过程部分
  - |——实现步骤记录
  - |——问题与解决方案
  - |——源码理解总结
  - |——测试验证部分
  - |——功能测试结果
  - |——性能数据
  - |——异常测试
  - |——运行截图/录屏

---

## 实验 0: 开发环境搭建

### 环境搭建步骤

#### 1. 安装基础依赖

```
# Ubuntu/Debian 系统
sudo apt-get update
sudo apt-get install -y build-essential git python3 qemu-system-misc
expect gdb-multifarch
```

```
# 验证 QEMU 版本 (建议 5.0+)
qemu-system-riscv64 - version
```

#### 2. 安装 RISC-V 工具链

```
# 方案 A: 使用预编译包 (推荐)
wget https://github.com/riscv-collab/riscv-gnu-toolchain/releases/download/2023.07.07/riscv64-elf-ubuntu-20.04-gcc-nightly-2023.07.07-nightly.tar.gz
sudo tar -xzf riscv64-elf-ubuntu-20.04-gcc-nightly-2023.07.07-nightly.tar.gz -C /opt/
echo 'export PATH="/opt/riscv/bin:$PATH"' >> ~/.bashrc
source ~/.bashrc
```

```
# 方案 B: 包含理解安装 (可能版本较旧)
sudo apt-get install gcc-riscv64-unknown-elf
```

```
# 验证安装
riscv64-unknown-elf-gcc - version
```

### 3. 获取参考资料

# 克隆 xv6 源码作为参考

git clone <https://github.com/mit-pdos/xv6-riscv.git>

或者 git clone git://g.csail.mit.edu/xv6-labs-2022

cd xv6-riscv && make qemu # 验证能否正常运行

或者 cd xv6-labs-2022 && make qemu && make qemu

### 4. 创建项目结构

mkdir riscv-os && cd riscv-os

git init

mkdir -p kernel/{boot,mm,trap,proc,fs,net} include scripts

### 5. 验证环境

创建测试文件验证交叉编译:

echo 'int main(){ return 0; }' > test.c

riscv64-unknown-elf-gcc -c test.c -o test.o

file test.o # 应显示 RISC-V 64-bit