**题目2：实现基于模块的文件系统**

1. **设计目的与要求**

本实验旨在通过修改Linux文件系统的源代码（ext3或ext4），实现一个基于模块的文件系统。具体目标包括：复制并修改ext3或ext4文件系统代码，调整Makefile文件以支持模块编译；实现动态加载和卸载文件系统模块的功能；在文件写操作中加入后台日志打印功能，用于验证功能实现；

1. **设计内容**

在实现基于模块的文件系统时，首先复制现有的 ext3 或 ext4 文件系统的源代码，重命名为自定义的文件系统名称（如 ext4\_mod），并修改 Makefile 文件，使其支持模块化编译。在文件系统的核心代码中，添加钩子函数以拦截文件写操作，利用 printk 输出写操作相关信息到系统日志后台（如文件名、写入时间、写入大小等）。通过修改 superblock 注册和文件系统挂载函数，确保新文件系统能够动态加载（insmod）和卸载（rmmod）。最后重新编译内核，测试新文件系统的功能及动态加载卸载效果。

1. **设备与环境**

操作系统：Ubuntu 20.04

虚拟化软件：VMware Workstation 17 Pro

开发环境：Visual Studio Code

内核版本：5.6.8

1. **设计思想**

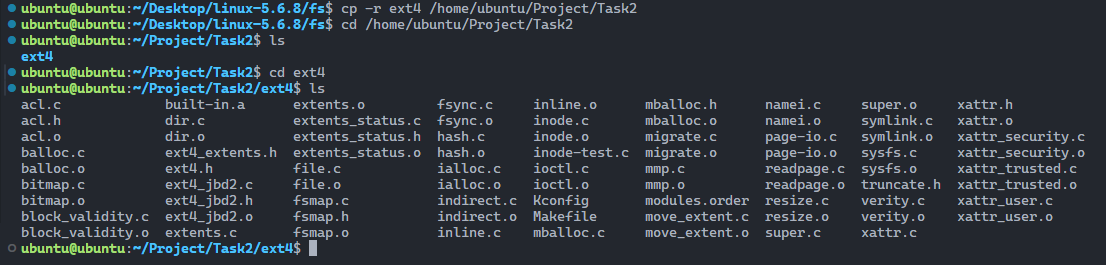
本实验采用模块化设计思想，基于ext3或ext4文件系统的现有代码，通过复制和修改的方式实现一个新的文件系统模块。模块化设计使得文件系统能够动态加载和卸载，不仅便于开发与调试，还提高了系统灵活性和扩展性。

在功能实现上，通过修改挂载逻辑和文件操作函数，实现了文件系统的基本功能。同时，在文件写操作中加入日志输出，能够通过内核日志实时监控模块运行情况。

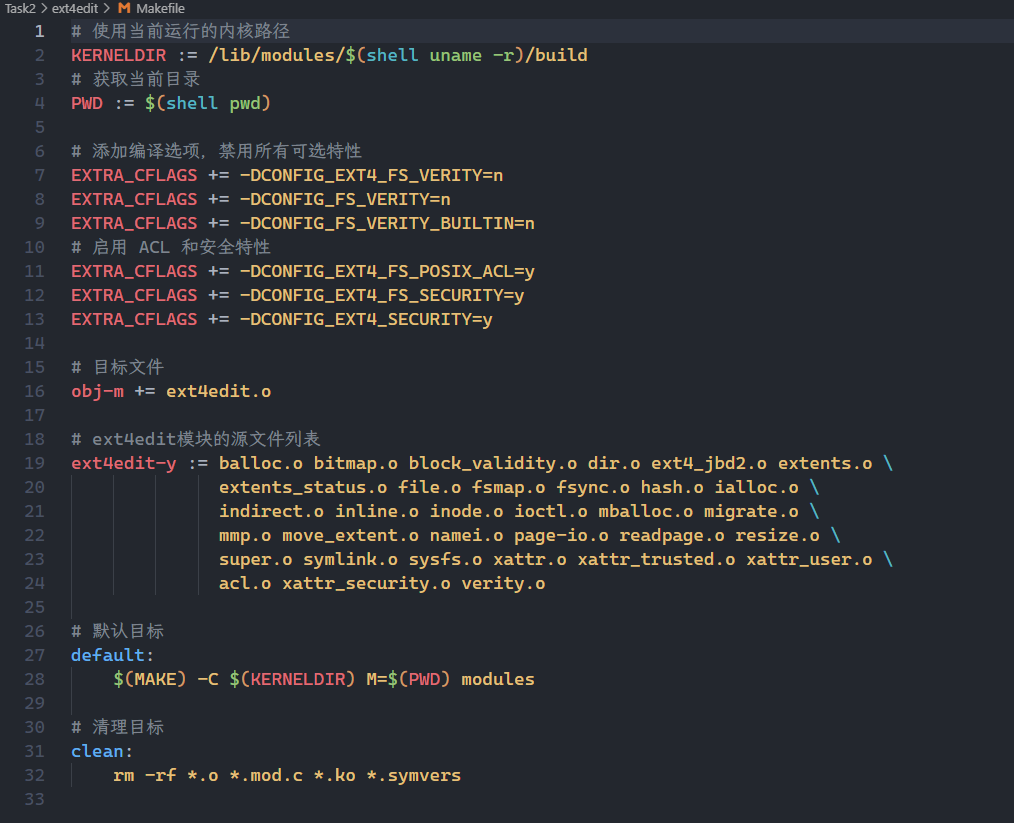
实验过程中充分利用Linux内核的模块化结构，将文件系统核心功能与用户态操作分离，从而提高实验的可操作性与功能验证的效率。

1. **主要数据结构和流程**

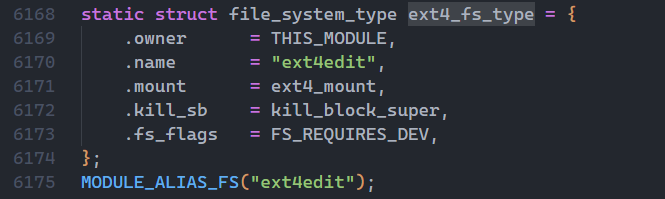
1.本实验基于实验1中下安装的内核进行，拷贝内核源码文件夹下fs/ext4文件夹里的所有内容，在本实验中复制到/home/ubuntu/Project/task2目录下。



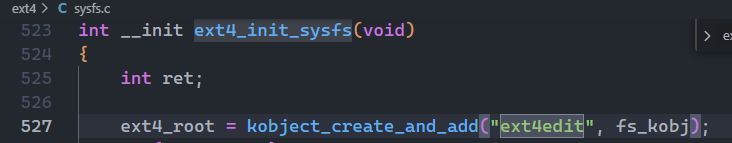
2.打开新目录文件系统根目录下的Makefile文件,并做相应修改，其中KERNELDIR变量为内核源代码位置，PWD变量为当前工作目录的绝对路径。



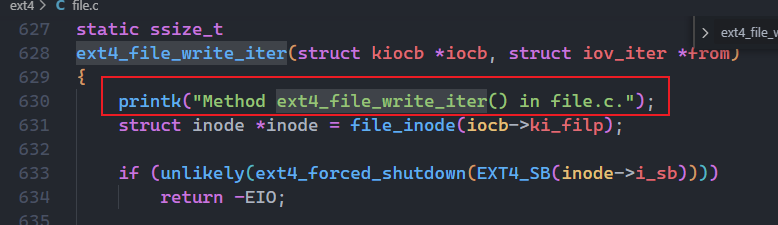
3. 修改文件系统名称，修改super.c 文件中的文件系统注册部分，将原文件系统名称ext4改为 ext4edit。



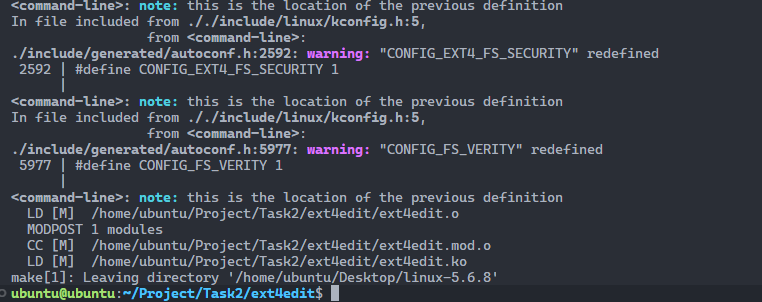
4. 找到与顶层虚拟文件系统相关的sysfs.c文件，。找到初始化函数ext4\_init\_sysfs（），修改kobject\_set\_name()里的字段为“ext4edit”。



5. 添加文件写操作日志：在 file.c 中，找到文件写入操作的实现函数ext4\_file\_write\_iter（），添加一条printk（）语句，打印提示信息。该函数用于写文件时延迟分配磁盘空间时，将数据按字节写入页缓存。

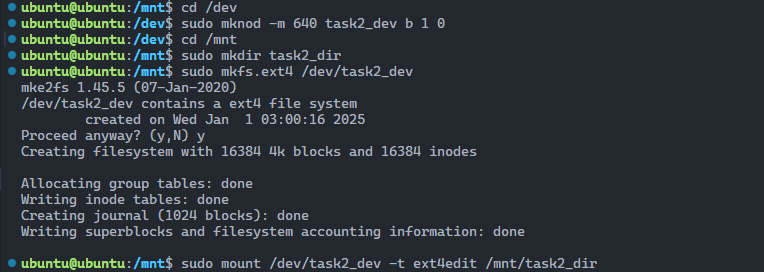


6. 进入模块文件夹,用make命令编译模块

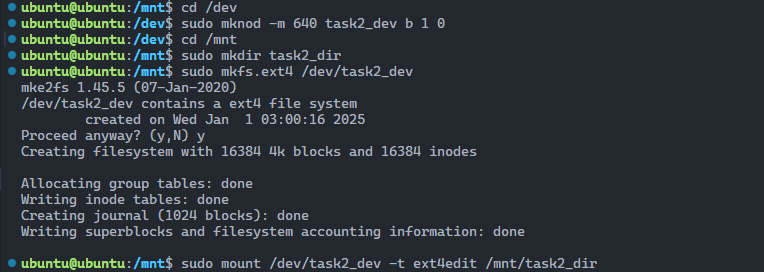


7.输入insmod ext4edit.ko命令加载文件系统

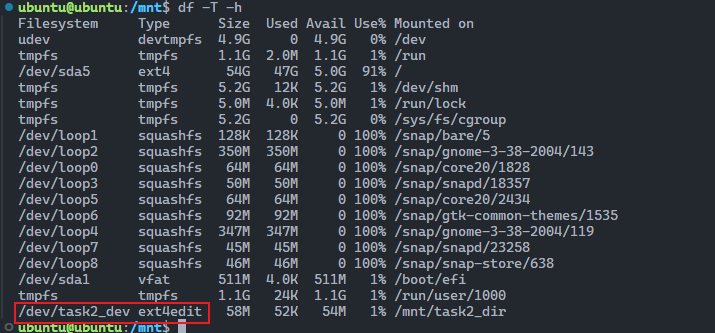
1. **实验测试结果及结果分析**
2. 进入/dev目录，后创建块设备文件，并格式化块设备文件



1. 进入/mnt目录，创建yoching文件夹，并挂载文件系统：



1. 使用df -T -h命令查看挂载信息：



1. 卸载文件系统

