SimpleFS – Technical Documentation

A Simple File System Implementation in C

1. Introduction

SimpleFS is an educational file system designed to demonstrate the internal mechanisms of a Unix-like filesystem. It simulates persistent storage using a disk image and implements core filesystem concepts such as superblocks, inodes, directory entries, block allocation, and file operations.

2. Architecture Overview

The filesystem is implemented as a single contiguous partition in memory and on disk. The partition is composed of three main regions: the superblock, the inode table, and data blocks.
Partition Layout:
- Superblock (1 KB)
- Inode Table (224 inodes, 32 bytes each)
- Data Blocks (4088 blocks of 1024 bytes)

3. Superblock

The superblock stores global metadata about the filesystem, including block size, inode count, free block counters, and the volume name. It is validated using a magic number (SIMPLE_PARTITION).

4. Inodes

Each inode represents a file or directory. It contains metadata such as file size, permissions, timestamps, and pointers to data blocks. SimpleFS supports six direct blocks and one indirect block for larger files.
Supported inode types:
- Regular files
- Directories

5. Data Blocks

Data blocks store raw file contents or directory entries. Each block is 1024 bytes. Directories are implemented as special files containing a sequence of directory entries.

6. Directory Entries

A directory entry maps a filename to an inode number. Each entry stores the inode reference, file type, name length, and record length. This design allows sequential directory traversal.

7. Allocation Management

SimpleFS uses bitmap-based allocation for both inodes and data blocks. Bitmaps are rebuilt at mount time by scanning existing inodes and their associated blocks.

8. File Operations

The filesystem supports the following operations:
- **mount**: Load a disk image into memory
- **format**: Create and initialize a new filesystem
- **create**: Create files or directories
- **read**: Read file contents
- **write**: Write data to files
- **list**: List directory contents
- **save**: Persist filesystem state to disk

9. Path Resolution

Paths are resolved starting from the root inode. Each path component is looked up sequentially in directory entries, mimicking Unix pathname resolution.

10. Interactive Shell

An interactive shell is provided to manipulate the filesystem. It supports commands such as ls, mkdir, touch, cat, write, mount, format, and save, offering a user-friendly interface for testing.

11. Error Handling

Operations return negative error codes on failure (e.g., file not found, no space left, invalid path). This design mirrors traditional POSIX-style error handling.

12. Conclusion

SimpleFS provides a clear and practical demonstration of filesystem internals. It is suitable for operating systems courses and experimentation, and can be extended with features such as caching, permissions, concurrency, or journaling.