TRABALHO PRÁTICO 3 – SISTEMA DE ARQUIVOS

João Vitor Ferreira – 2021039654

Lucas Roberto S. Avelar – 2021039743

Luiza de Melo Gomes – 2021040075

Este documento descreve a organização inicial do sistema de arquivos e a organização das páginas de blocos livres. Além disso, discute e justifica as escolhas realizadas na implementação.

Organização Inicial do Sistema de Arquivos

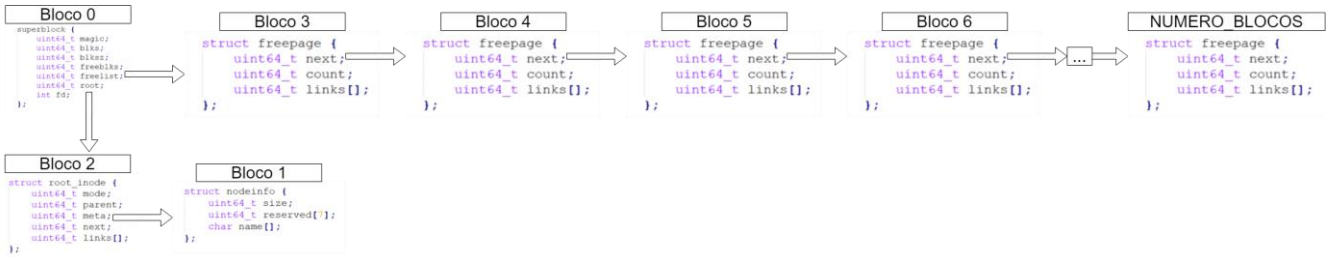
O sistema de arquivos é organizado em blocos, cada um com um tamanho definido na criação do sistema de arquivo. O primeiro bloco é reservado para o superbloco, que contém informações sobre o sistema de arquivos como um todo, incluindo o número total de blocos, o número de blocos livres e a localização do diretório raiz e da lista de blocos livres.

O segundo bloco é reservado para o nodeinfo do diretório raiz, que contém informações sobre o diretório, como seu nome e tamanho. O terceiro bloco é o inode do diretório raiz, que contém informações sobre os arquivos e subdiretórios dentro do diretório raiz.

O quarto bloco é a lista de blocos livres, que mantém o controle dos blocos que não estão sendo usados atualmente.

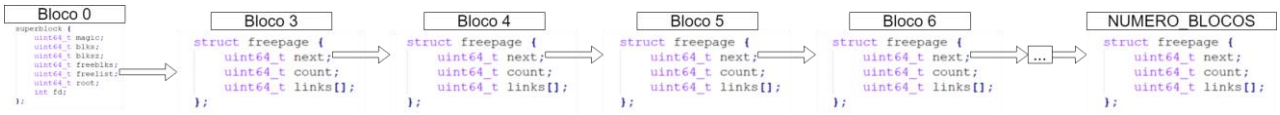
Os blocos restantes são usados para armazenar os dados dos arquivos e diretórios.

Segue imagem da organização inicial do sistema de arquivos:

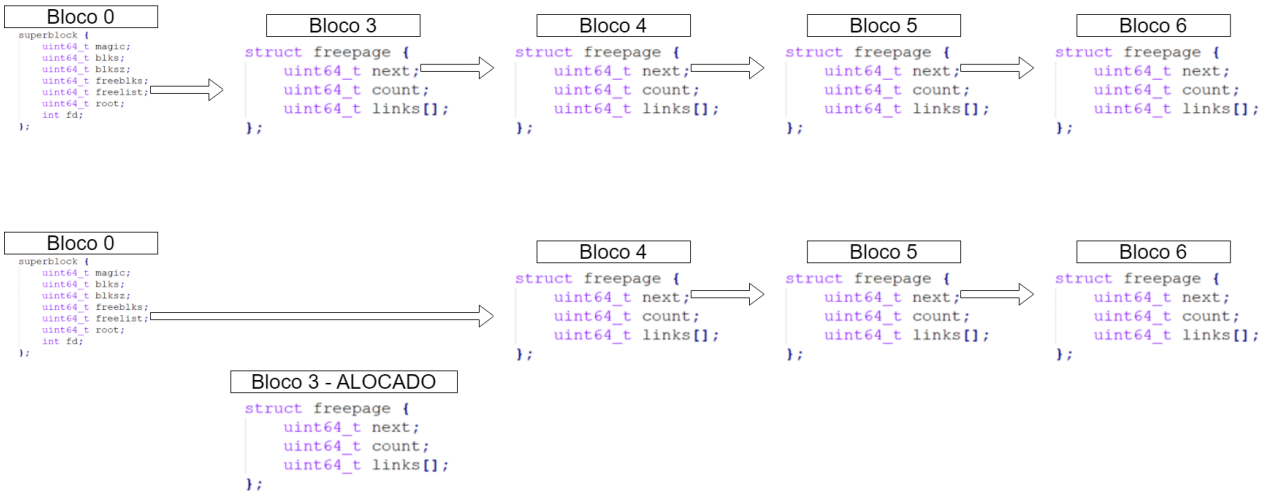


Organização das Páginas de Blocos Livres

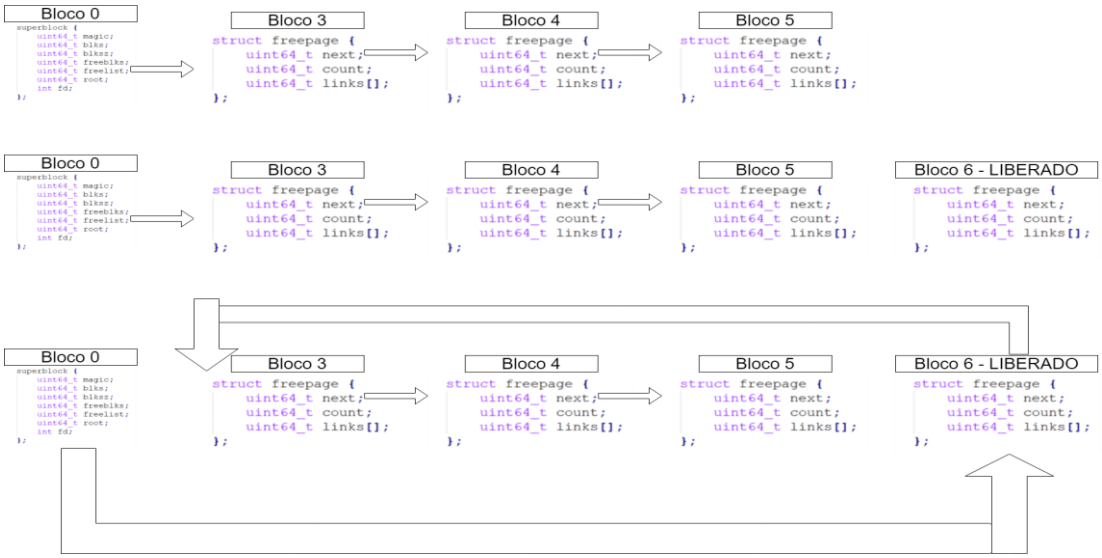
A lista de blocos livres é uma lista encadeada de blocos, onde cada bloco contém o número do próximo bloco livre na lista. Isso permite que o sistema de arquivos saiba rapidamente onde encontrar um bloco livre quando precisar armazenar novos dados. Inicialmente, as páginas de blocos livres encontram-se da seguinte forma:



Quando um bloco é alocado, ele é removido da lista de blocos livres. O primeiro bloco livre é retornado e a página de livres passa a apontar para o bloco seguinte:



Quando um bloco é liberado, ele é adicionado de volta à lista de blocos livres. O bloco recém-livre passa a apontar para o primeiro da lista de livres, e o superbloco aponta a lista de livres para o bloco devolvido:



Escolhas de Implementação

A escolha de organizar o sistema de arquivos em blocos foi feita para permitir uma alocação eficiente do espaço em disco. Cada bloco pode ser alocado independentemente, permitindo que o sistema de arquivos use o espaço em disco de maneira granular.

A lista de blocos livres foi implementada como uma lista encadeada para permitir uma alocação e liberação eficientes de blocos. A alternativa seria usar um bitmap para rastrear os blocos livres, mas isso exigiria a varredura de todo o bitmap sempre que um bloco fosse alocado ou liberado.

A escolha de armazenar o nodeinfo e o inode para cada diretório e arquivo permite que o sistema de arquivos mantenha informações detalhadas sobre cada item no sistema de arquivos. Isso inclui não apenas o tamanho e a localização dos dados do item, mas também metadados como o nome do item.

Finalmente, a escolha de reservar o primeiro bloco para o superbloco permite que o sistema de arquivos mantenha informações globais sobre o estado do sistema de arquivos em um local conhecido. Isso simplifica a implementação, pois o código que precisa acessar essas informações sabe exatamente onde encontrá-las.

Estrutura de Dados

Superbloco

O superbloco é a estrutura de dados central do sistema de arquivos. Ele contém informações globais sobre o sistema de arquivos, incluindo:

- O número mágico que identifica o sistema de arquivos.
- O número total de blocos no sistema de arquivos.
- O tamanho de cada bloco.
- O número de blocos livres restantes.
- A localização do diretório raiz.
- A localização da lista de blocos livres.

Nodeinfo

Cada arquivo e diretório no sistema de arquivos tem uma estrutura nodeinfo associada a ele. Esta estrutura contém informações sobre o item, incluindo:

- O nome do item.
- O tamanho do item (em bytes para arquivos, em número de itens para diretórios).
- Informações reservadas para uso futuro.

Inode

Cada arquivo e diretório também tem um ou mais inodes associados a ele. Cada inode contém:

- O modo do item (arquivo regular, diretório etc.).
- A localização do inode pai (para inodes filho).
- A localização do nodeinfo do item.
- A localização do próximo inode na lista de inodes do item (para arquivos com mais de um bloco de dados).
- Para diretórios, uma lista de links para os inodes dos itens no diretório.

Operações do Sistema de Arquivos

Formatação

A operação de formatação inicializa um novo sistema de arquivos em um arquivo existente. Ela configura o superbloco, cria o diretório raiz e inicializa a lista de blocos livres.

Abertura e Fechamento

As operações de abertura e fechamento permitem que o sistema de arquivos seja usado por um programa. A abertura verifica se o arquivo contém um sistema de arquivos válido e retorna o superbloco. O fechamento libera todos os recursos usados pelo sistema de arquivos.

Alocação e Liberação de Blocos

As operações de alocação e liberação de blocos permitem que o sistema de arquivos gerencie o espaço em disco. A alocação remove um bloco da lista de blocos livres e o retorna para uso. A liberação devolve um bloco à lista de blocos livres.

Leitura e Escrita de Arquivos

As operações de leitura e escrita de arquivos permitem que os programas acessem os dados armazenados no sistema de arquivos. A leitura recupera os dados de um arquivo existente, enquanto a escrita armazena novos dados em um arquivo.

Criação e Remoção de Diretórios

As operações de criação e remoção de diretórios permitem que os programas organizem os arquivos no sistema de arquivos. A criação cria um diretório vazio, enquanto a remoção exclui um diretório existente (se estiver vazio).

Listagem de Diretórios

A operação de listagem de diretórios permite que os programas vejam o conteúdo de um diretório. Ela retorna uma lista de nomes de arquivos e diretórios no diretório especificado.

Justificativas para as Escolhas de Implementação

A implementação deste sistema de arquivos foi projetada para ser simples, mas eficiente. A organização em blocos permite uma alocação granular do espaço em disco, enquanto a lista encadeada de blocos livres permite uma alocação e liberação eficientes de blocos. As estruturas nodeinfo e inode fornecem uma representação flexível e detalhada de arquivos e diretórios. Finalmente, a reserva do primeiro bloco para o superbloco simplifica a implementação, garantindo que as informações globais sobre o sistema de arquivos estejam sempre em um local conhecido.

Descrição Detalhada das Funções e do Código

search_inode(struct superblock *sb, struct inode *in, struct inode *in2, struct nodeinfo *info2, int k)

Esta função é usada para buscar um inode de um arquivo. Ela lê o inode no índice k do array de links do inode fornecido. Se o inode lido é um inode filho (ou seja, não é o primeiro inode de um arquivo ou diretório), ela pula para o primeiro inode.

copy_inode(struct inode *a, struct inode *b, struct nodeinfo *c) e copy_nodeinfo(struct nodeinfo *a, struct nodeinfo *b)

Essas funções são usadas para copiar os dados de um inode ou nodeinfo para outro. Elas copiam todos os campos de b para a.

create_superblock(const char *fname, uint64_t blocksize, uint64_t numero_blocos)

Esta função cria um superbloco para um sistema de arquivos. Ela aloca memória para o superbloco, define seus campos e retorna um ponteiro para ele.

create_root_directory(uint64_t blocksize)

Esta função cria um diretório raiz para um sistema de arquivos. Ela aloca memória para o inode do diretório raiz, define seus campos e retorna um ponteiro para ele.

fs_format(const char *fname, uint64_t blocksize)

Esta função cria uma imagem de sistema de arquivos no arquivo especificado por fname. Ela verifica se o tamanho do bloco é menor que o tamanho mínimo do bloco e se há espaço suficiente para armazenar o número mínimo de blocos em fname. Se essas condições forem atendidas, ela cria o superbloco e o diretório raiz e inicializa a lista de páginas vazias.

fs_open(const char *fname)

Esta função abre o sistema de arquivos no arquivo especificado por fname e retorna seu superbloco. Ela verifica se o sistema de arquivos já está aberto e se o valor mágico no superbloco corresponde ao esperado.

fs_close(struct superblock *sb)

Esta função fecha o sistema de arquivos apontado por sb. Ela libera a memória alocada para o superbloco e fecha o descritor de arquivo associado ao sistema de arquivos.

fs_get_block(struct superblock *sb)

Esta função obtém um bloco livre no sistema de arquivos. Ela lê a primeira página livre na lista de páginas livres, remove-a da lista e retorna seu número de bloco.

fs_put_block(struct superblock *sb, uint64_t block)

Esta função coloca um bloco de volta no sistema de arquivos como um bloco livre. Ela adiciona o bloco à frente da lista de páginas livres e atualiza o superbloco.

fs_write_file(struct superblock *sb, const char *fname, char *buf, size_t cnt)

Esta função escreve cnt bytes de buf no sistema de arquivos apontado por sb. Os dados serão escritos em um arquivo chamado fname. Se o arquivo já existir, ele será sobrescrito pelos novos dados.

fs_unlink(struct superblock *sb, const char *fname)

Esta função remove o arquivo chamado fname do sistema de arquivos apontado por sb. Ela libera todos os blocos associados ao arquivo e remove sua entrada do diretório pai.

fs_mkdir(struct superblock *sb, const char *dname)

Esta função cria um diretório no caminho especificado por dname. Ela aloca um novo inode e nodeinfo para o diretório, adiciona uma entrada para o diretório no diretório pai e atualiza o superbloco.

fs_rmdir(struct superblock *sb, const char *dname)

Esta função remove o diretório no caminho especificado por dname. Ela verifica se o diretório está vazio, libera o inode e o nodeinfo do diretório e remove sua entrada do diretório pai.

fs_list_dir(struct superblock *sb, const char *dname)

Esta função retorna uma string com o nome de todos os elementos (arquivos e diretórios) no diretório especificado por dname. Os nomes dos elementos são separados por espaços e os diretórios são indicados com uma barra no final do nome.