Dokumentace k semestrální práci: PseudoFAT souborový systém

Jan Čácha

10. prosince 2024

Úvod

Tato dokumentace popisuje implementaci zjednodušeného souborového systému založeného na pseudoFAT. Cílem práce je vytvořit funkční program, který umožňuje manipulaci se soubory a adresáři prostřednictvím příkazů zadávaných uživatelem. Program podporuje základní operace jako vytvoření adresáře, odstranění souboru nebo adresáře, zobrazení obsahu adresáře, přesun mezi adresáři a další. Vstupy a výstupy programu musí splňovat stanovený formát.

Specifikace příkazů

Program podporuje následující příkazy:

- **mkdir** *cesta*
  Vytvoří nový adresář na zadané cestě. Cesta může být absolutní nebo relativní.
- **rmdir** *cesta*
  Odstraní prázdný adresář na zadané cestě.
- **rm** *cesta*
  Odstraní soubor na zadané cestě.
- **ls**
  Zobrazí obsah aktuálního adresáře.
- **pwd**
  Zobrazí absolutní cestu k aktuálnímu adresáři.
- **incp** *zdroj cíl*
  Zkopíruje soubor ze systému do pseudoFAT souborového systému.

- **outcp** *zdroj cíl*
  Zkopíruje soubor z pseudoFAT systému do systému.
- **load** *soubor*
  Načte stav souborového systému z daného souboru.
- **bug**
  Testovací příkaz pro odhalení potenciálních chyb v implementaci.
- **check**
  Zkontroluje integritu souborového systému.
- **cp** *zdroj cíl*
  Zkopíruje soubor nebo adresář v rámci pseudoFAT souborového systému.
- **mv** *zdroj cíl*
  Přesune soubor nebo adresář na novou cestu.
- **cd** *cesta*
  Změní aktuální adresář na zadanou cestu.
- **cat** *cesta*
  Zobrazí obsah souboru na zadané cestě.
- **info** *cesta*
  Zobrazí informace o souboru nebo adresáři na zadané cestě.

Popis implementace

Definice datových struktur

V implementaci jsou použity základní datové struktury. Z nich nejdůležitější jsou:

Struktura FSDescription:

Listing 1: Struktura popisující souborový systém

```
typedef struct FSDescription {
    char signature[9];
    int32_t disk_size;
    int32_t cluster_size;
    int32_t cluster_count;
    int32_t fat_count;
    int32_t *fat1_start_address;
    int32_t *fat2_start_address;
    int32_t data_start_address;
} FSDescription;
```

Struktura DirectoryItem:

Listing 2: Struktura adresáře a souboru

```
typedef struct DirectoryItem {
    char item_name[MAX_ITEM_NAME_SIZE];
    bool isFile;
    int32_t size;
    int32_t start_cluster;
    struct DirectoryItem *parent;
    struct DirectoryItem *children[MAX_CHILDREN];
    int child_count;
} DirectoryItem;
```

Globální proměnné

Hlavní globální proměnné, které řídí stav systému, zahrnují:

- `fs_description` : Popis souborového systému.
- `fat_table1` a `fat_table2`: Odkazy na obě FAT tabulky.
- `root_directory`: Kořenový adresář souborového systému.
- `current_directory`: Ukazatel na aktuální pracovní adresář.

Popis implementace hlavní funkce ‘main‘

Funkce `main()` tvoří interaktivní rozhraní mezi uživatelem a pseudoFAT souborovým systémem. Je navržena tak, aby umožnila uživatelům komunikaci se souborovým systémem a vykonávala příkazy pro manipulaci se soubory a adresáři. Funkce obsahuje několik klíčových částí, jejichž detailní vysvětlení je uvedeno níže:

1. Kontrola argumentů

Na začátku funkce se provádí kontrola vstupních argumentů. Pokud uživatel nezadá správný počet argumentů, funkce vypíše chybovou zprávu a program se ukončí. Toto je klíčové pro zajištění, že uživatel specifikuje správný název souborového systému, na kterém bude program pracovat.

2. Načítání aktuálního stavu souborového systému

Po kontrole argumentů je zavolána funkce `load_system_state()`, která načítá aktuální stav souborového systému ze souboru. Tato část zajišťuje obnovení stavu systému po předchozím ukončení nebo při restartu.

3. Interaktivní smyčka pro zpracování příkazů

Pomocí nekonečné smyčky `while (1)` je program připraven neustále přijímat uživatelské příkazy. Smyčka umožňuje uživatelům zadávat příkazy pro různé operace, například vytvoření adresářů, mazání souborů, přesun mezi adresáři nebo kopírování souborů.

4. Zpracování uživatelských příkazů

Po zadání příkazu smyčka zpracovává vstup od uživatele. Nový řádek je odstraněn pomocí funkce `strcspn()`, aby bylo možné vstup správně analyzovat. Příkaz je následně předán funkci `process_command()`, která analyzuje a provede odpovídající akci.

5. Uložení aktuálního stavu systému při ukončení

Po ukončení interaktivní smyčky příkazem `exit` je volána funkce `save_system_state()`. Tato funkce zajišťuje, že všechny změny provedené během relace jsou uloženy do souboru, čímž je zabráněno jejich ztrátě.

Shrnutí funkcionality

Celý tok v rámci funkce `main()` umožňuje:

- Načíst stav systému ze souboru.
- Zpracovávat uživatelské příkazy interaktivním způsobem.
- Uložit aktuální stav systému při ukončení.

Popis implementace souboru ‘filesystem.c‘

Funkce: `initialize_filesystem`

`initialize_filesystem` inicializuje souborový systém s danými parametry velikosti disku a velikosti clusterů.

- **Validace parametrů:** Funkce nejprve ověřuje, zda jsou parametry platné.
- **Nastavení základních informací souborového systému:** Nastavují se základní atributy, jako je podpis systému, velikost clusterů, počet clusterů a velikost FAT tabulek.
- **Alokace paměti:** Alokují se paměťové bloky pro FAT tabulky a samotný datový prostor systému.
- **Inicializace tabulek:** FAT tabulky jsou inicializovány hodnotou “FAT_UNUSED”, aby značení clusterů začalo v očekávaném stavu.
- **Vytvoření root adresáře:** Root adresář je nastaven jako výchozí adresář souborového systému.

Funkce: `format_filesystem`

`format_filesystem` formátuje souborový systém a ukládá jeho počáteční stav do souboru.

- **Otevření souboru:** Funkce otevírá soubor pro zápis.
- **Inicializace systému:** Funkce volá `initialize_filesystem`.
- **Uložení stavu:** Po inicializaci je systém uložen do souboru pomocí funkce `save_system_state`.

Funkce: `save_directory`

Funkce `save_directory` rekurzivně ukládá stav adresářů a jejich dětí do souboru.

- Uloží aktuální adresář.
- Pro každý podadresář volá funkci rekurzivně, aby se uložila jeho struktura.

Funkce: `save_system_state`

`save_system_state` ukládá aktuální stav souborového systému do souboru.

- **Uložení FSDescription:** Uloží základní systémové informace.
- **Uložení FAT tabulek:** FAT tabulky jsou zapsány do souboru.
- **Uložení adresářové struktury:** Rekurzivně se ukládají adresáře pomocí funkce `save_directory`.
- **Uložení dat:** Uloží se také obsah datového prostoru.

Funkce: `load_directory`

`load_directory` rekurzivně načítá adresářovou strukturu a její podadresáře ze souboru.

- Načítá aktuální adresář ze souboru.
- Rekurzivně načítá podadresáře a nastavuje vztahy mezi nimi.

Funkce: `load_system_state`

`load_system_state` načítá souborový systém ze souboru.

- **Načítání FSDescription:** Získávají se základní informace o souborovém systému.
- **Alokace paměti:** Alokují se paměťové prostory pro FAT tabulky a datový prostor.
- **Načítání dat:** Načítají se FAT tabulky, adresářová struktura a datový prostor ze souboru.
- **Nastavení kořenového adresáře:** Po načtení je aktuální adresář nastaven na kořenový adresář.

Popis implementace souboru ‘directory.c‘

Funkce pro správu clusterů

- **Uvolnění clusteru:** Tato funkce označí zadaný cluster jako volný (`FAT_UNUSED`) v tabulce FAT, čímž umožní jeho opětovné použití.

Funkce pro práci s adresáři

- **Uvolnění adresářů:** Funkce zajišťuje správné uvolnění alokované paměti spojené s adresářem.
- **Aktualizace velikosti adresářů:** Funkce iteruje přes soubory a podadresáře v rámci adresáře, vypočítává jejich velikosti a alokuje potřebné clusterové bloky.
- **Alokace clusterů pro adresář:** Funkce provádí alokaci dodatečných clusterů, pokud adresář překročí svou aktuální velikost.

Funkce pro čtení a zápis clusterů

- **Čtení dat z clusteru:** Funkce čte data z konkrétního clusteru do paměti a provádí bezpečnostní kontroly.
- **Zápis dat do clusteru:** Funkce zapisuje data do zadaného clusteru a ověřuje validitu indexů a velikosti dat.

Funkce pro hledání v adresářové struktuře

- **Najít adresář nebo soubor podle názvu:** Funkce hledá soubor nebo adresář v aktuálním adresáři na základě zadaného názvu.
- **Práce s cestami:** Funkce analyzují zadané cesty (např. `/adresar/soubor`) a iterují v adresářové struktuře.

Funkce pro alokaci clusterů

- **Alokace nového clusteru:** Funkce provádí hledání volného clusteru v tabulce FAT, který následně označí jako konec souboru a vrátí jeho index.

Funkce pro rekurzivní operace

- **Odstranění adresářů a souborů:** Funkce provádějí rekurzivní mazání všech souborů a podadresářů uvnitř zadaného adresáře.
- **Kopírování souborů a adresářů:** Funkce provádějí rekurzivní kopírování dat mezi soubory a adresáři, včetně alokace clusterů.

Funkce pro manipulaci s daty

- **Čtení dat z clusterů:** Funkce provádí bezpečné čtení dat z clusteru.
- **Zápis dat do clusterů:** Funkce zapisují data do clusterů a zajišťují správnou manipulaci s tabulkou FAT.

Shrnutí funkcionalit

Celkově implementace těchto funkcí umožňuje:

- Správu volných clusterů a tabulky FAT.
- Práci s adresářovou strukturou, včetně funkcí jako `mkdir` (vytvoření adresáře) a `rmdir` (odstranění adresáře)...
- Manipulaci dat mezi pamětí a diskem prostřednictvím čtení a zápisu do clusterů.
- Rekurzivní mazání a kopírování dat a adresářů.
- Správu adresářů pomocí funkcí, které iterují adresářovou strukturu.

Použití v rámci souborového systému

Implementace těchto funkcí je nezbytnou součástí hlavního souborového systému založeného na modelu FAT. Funkce tvoří základní funkční vrstvu, která umožňuje:

- Vytváření a mazání adresářů.
- Manipulaci se soubory a datovými bloky (clusteru).
- Správu dat a adresářů pomocí rekurzivních operací.
- Přesouvání souboru z pevného disku do PseudoFAT souboru

Pomocné funkce jsou optimalizovány pro bezpečnostní podmínky a kontrolu všech vstupních dat, aby se předešlo chybám v systému.

Závěr

Tento projekt implementuje základní funkce pseudoFAT souborového systému a umožňuje rozhraní pro manipulaci souborů a adresářů. Celkový kód je optimalizován pro výuku a studium datových struktur a souborových systémů.