Tema 4 - Driver de sistem de fişiere

Termen de predare: **joi, 5 Mai 2011, ora 23:00**

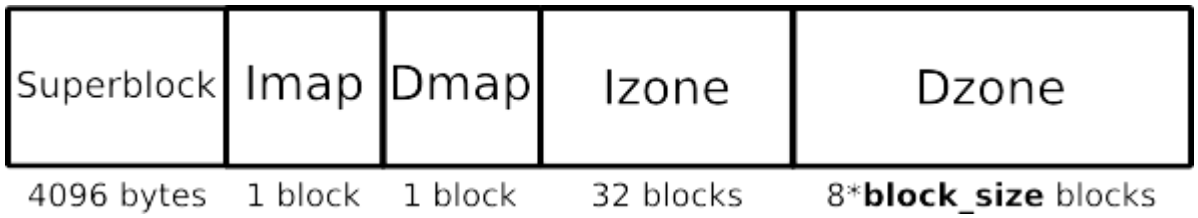
Enunţ

Să se scrie un modul de kernel care să implementeze sistemul de fişiere `PITIX`. Acest sistem de fişiere va oferi suport doar pentru fişiere şi directoare. Nu se vor implementa operaţiile de suport pentru link-uri hard sau simbolice. De asemenea, nu se vor implementa operaţiile de suport pentru fişiere speciale (pipe-uri, dispozitive caracter sau bloc). Practic va trebui să implementaţi următoarele operaţii:

- pentru directoare: `lookup`, `unlink`, `mkdir`, `rmdir`, `readdir`
- pentru fişiere: `create`, `truncate`, funcţia de bitmap (vezi minix_get_block)

Restul funcţiilor fie au implementări generice în kernel, fie nu trebuie să le implementaţi.

Structura pe disk a sistemului de fişiere este prezentată mai jos:



unde:

- `Superblock` conţine superblocul (4096 octeţi)
- `Imap` conţine harta de biţi (bitmap) a blocurilor ocupate de inode-uri (1 bloc)
- `Dmap` conţine harta de biţi (bitmap) a blocurilor ocupate de date (1 bloc)
- `Izone` conţine inode-urile (32 blocuri)
- `Dzone` conţine datele (conţinutul efectiv al fişierelor) (8*block_size blocuri)

Structura superblocului (pe disc) este descrisă de următoarea structură:

```
struct pitix_super_block {
        unsigned long magic;
        __u8 version;
        __u8 inode_data_blocks;
        __u8 block_size_bits;
        __u8 imap_block;
        __u8 dmap_block;
        __u8 izone_block;
        __u8 dzone_block;
        __u16 bfree;
        __u16 ffree;
//...
}
```

unde

- `magic` trebuie să fie iniţializat cu `PITIX_MAGIC`
- `version` trebuie să fie iniţializat cu 1
- `inode_data_blocks` este numărul de blocuri de date din inode (vezi mai jos la descrierea inode-ului)
- `block_size_bits` este dimensiunea blocului în puteri ale lui doi; dimensiunea blocului trebuie să fie 512, 1024, 2048, sau 4096.
- `imap_block` este numărul blocului (relativ la dispozitiv) pentru vectorul de biţi folosit pentru alocarea/eliberarea inode-urilor
- `dmap_block` este numărul blocului (relativ la dispozitiv) pentru vectorul de biţi folosit pentru alocarea/eliberarea blocurilor de date
- `izone_block` este numărul primului bloc (relativ la dispozitiv) al zonei de inode-uri
- `dzone_block` este numărul primului bloc (relativ la dispozitiv) al zonei de date
- `bfree` este numărul de blocuri libere (nealocate)

- `ffree` este numărul de inode-uri libere (nealocate)

Inode-urile vor fi stocate în zona de inode-uri şi sunt descrise de următoarea structură:

```
struct pitix_inode {
        __u32 mode;
        __u32 uid;
        __u32 gid;
        __u32 size;
        __u32 time;
        __u16 data_blocks[0];
};
```

unde

- `mode` reprezintă drepturile de acces şi tipul inode-ului (fişier sau director) aşa cum sunt reprezentate în kernel
- `uid` reprezintă UID-ul aşa cum este el reprezentat în kernel
- `gid` reprezintă GID-ul aşa cum este el reprezentat în kernel
- `size` reprezintă dimensiunea fişierului/directorului
- `time` reprezintă timpul de modificare aşa cum este el reprezentat în kernel
- `data_blocks` este un vector (cu dimensiunea precizată in superbloc) care indică numerele blocurilor de date relativ la zona de date; intrările nefolosite trebuie setate pe 0 (adică dacă avem un fişier cu dimensiune 2050 şi dimensiunea blocului de 1024 atunci vom avea primele trei valori din vector strict pozitive şi restul 0)

Directoarele au asociate un singur bloc de date în care vor fi stocate intrările de director. Acestea sunt descrise de următoarea structură:

```
struct pitix_dir_entry {
        __u32 ino;
        char name[PITIX_NAME_LEN];
};
```

unde

- `ino` reprezintă numărul inode-ului fişierului sau directorului; acest număr este un index în zona de inode-uri
- `name` reprezintă numele fişierului sau al directorului; lungimea maximă a numelui este 16 octeţi (`PITIX_NAME_LEN`); dacă lungimea numelui e mai mică decât 16 octeţi, atunci numele va fi terminat cu caracterul ASCII ce are codul 0 (la fel ca la şirurile de caractere)

Toate valorile numerice sunt stocate pe disc în CPU byte-order.

Directorul rădăcină va avea alocat inode-ul 0 şi blocul de date 0.

Pentru simplificare, la `mkdir` nu este necesară crearea intrărilor `"."` şi `".."` în noul director; tester-ul se foloseşte de această presupunere.

Precizări Linux

- un exemplu de imagine, ce conţine aceste fişiere
- structurile descrise le puteţi găsi şi în acest header
- sursele sistemului de fişiere minix vă pot fi de folos

Testare

Pentru simplificarea procesului de corectare al temelor, dar şi pentru a reduce greşelile temelor trimise, corectarea temelor se va face automat cu ajutorul unor teste publice. Testele presupun că numele modulului de kernel este `pitix`.

Depunctări

În corectarea temei se va ţine cont de următoarele criterii de depunctare:

- -1.0 implementare incorectă a cerinţelor temei
- -0.2 omiterea eliberării resurselor
- -0.0 observaţii

În cazuri excepţionale (tema trece testele prin nerespectarea cerinţelor) şi în cazul în care tema nu trece toate testele se poate scădea mai mult decât este menţionat mai sus.

De asemenea, consultaţi sfaturile şi depunctările generale.

Notare

Datorită dificultăţii ridicate, tema valorează 20 de puncte.

Resurse utile

1. Laboratorul 9. Drivere sisteme de fişiere (Linux) partea 1
2. Laboratorul 10. Drivere sisteme de fişiere (Linux) partea 2
3. Sursele sistemului de fişiere minix

Întrebări

Pentru întrebări legate de tema puteţi consulta arhivele listei de discuţii sau puteţi trimite un e-mail (trebuie să fiţi înregistraţi).

Înainte să puneţi o întrebare verificaţi că:

1. aţi citit bine enunţul temei
2. nu este deja prezentată întrebarea pe pagina de FAQ
3. nu se poate găsi răspunsul în arhivele listei de discuţii