UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL REI
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
SISTEMAS OPERACIONAIS
2º semestre de 2019
Professor: Rafael Sachetto Oliveira

Trabalho Prático 3
Data de Entrega: 12/12/2019. Trabalho em Trio

Este trabalho tem por objetivo o melhor entendimento sobre sistemas de arquivos de um sistema operacional.

1 O Trabalho

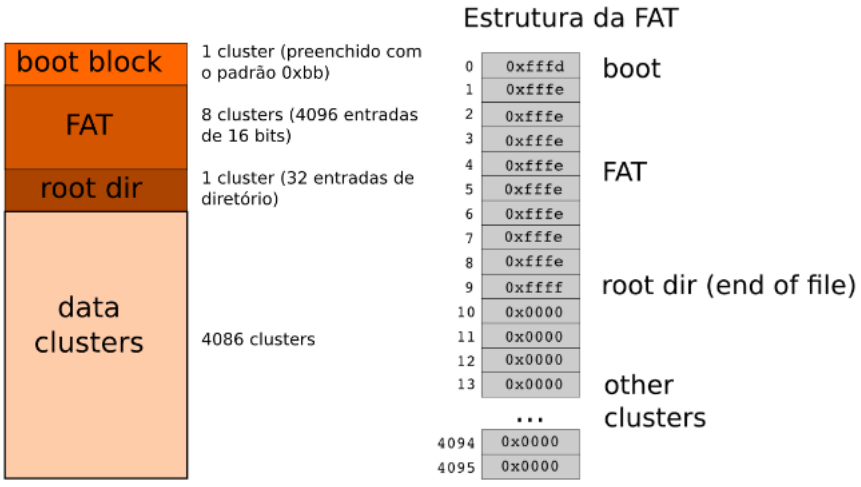
O terceiro trabalho prático da disciplina de Sistemas Operacionais consiste na implementação de um simulador de um sistema de arquivos simples baseado em tabela de alocação de 16 bits (FAT) e um shell usado para realizar operações sobre este sistema de arquivos. O sistema de arquivos virtual deverá ser armazenado em uma partição virtual e suas estruturas de dados mantidas em um único arquivo nomeado fat.part.

A partição virtual teria um tamanho total determinado por:

- 512 bytes por setor;
- cluster de 1024 bytes (2 setores por cluster);
- 4096 clusters;

Dessa forma, seu tamanho pode ser calculado por 512 bytes por setor * 2 setores por cluster * 4096 clusters = 4MB. Os dados devem ser alocados sempre em clusters, ou seja, um arquivo ocupará básicamente no mínimo um cluster (1024 bytes) no sistema de arquivos virtual.

O primeiro cluster é definido como boot block, e conterá informações referentes ao volume (partição). Por motivos de simplificação, o boot block terá o tamanho de 1 cluster (1024 bytes) e não 1 setor (512 bytes) como seria o usual, e deve ser preenchido com o valor 0xbb. A FAT terá um tamanho determinado por 4096 clusters de dados * 2 bytes por entrada (16 bits) = 8192 bytes (8 clusters). Inicialmente a FAT será inicializada com valores definidos adiante. O diretório root estará localizado logo após a FAT e terá um tamanho de 1 cluster (assim como todos os outros diretórios). O diretório root possui um conjunto de entradas de diretório que podem apontar para outros diretórios ou arquivos. Inicialmente, as entradas de diretório devem estar livres, inicializando-se todas as estruturas com 0x00.



Após a FAT e o diretório root, encontra-se a seção de dados contendo o restante dos clusters. Outros diretórios (e sub-diretórios) são definidos como clusters que possuem diversas entradas de diretório (assim como o diretório root), possuindo uma estrutura apresentada adiante.

Detalhes sobre o sistema de arquivos

O sistema de arquivos possui uma série de limitações, que foram determinadas com o intuito de simplificar a implementação do trabalho. A primeira limitação refere-se ao tamanho da FAT, onde é possível armazenar apenas 4096 entradas para blocos, o que limita o tamanho da partição virtual em 4MB. Lembre que a FAT é um "mapa"que representa a estrutura dos blocos da partição. Se mais entradas fossem necessárias (para um disco maior), seriam necessários blocos adicionais para a FAT. A segunda limitação refere-se ao número de entradas de diretório em cada nível da árvore. Cada entrada ocupa 32 bytes, o que limita o número de entradas de diretório em 32, tanto no diretório raiz quanto em sub-diretórios (os diretórios possuem tamanho de um bloco e não podem ser aumentados).

Não será permitido o uso de trapaças para a manipulação das estruturas de dados (como ler todo o sistema de arquivos para a memória para manipular as estruturas). Deve-se ler e escrever sempre utilizando a unidade cluster, independente de ser um diretório ou bloco de dados de arquivo. A FAT pode ser lida/gravada completamente no disco (para fim de simplificação). Sugere-se manter dois blocos de dados em memória - FAT (8 blocos) e um bloco para entradas de diretório ou dados. Não esqueça de após manipular a FAT ou dados de atualizar o sistema de arquivos virtual nas entradas de diretório. Lembre-se que o sistema precisa manter-se consistente, ao ponto de poder ser recuperado a qualquer instante.

Informações sobre o valor das entradas na FAT de 16 bits:

```
0x0000                -> cluster livre
0x0001 - 0xfffc       -> arquivo (ponteiro p/ proximo cluster)
0xfffd                -> boot block
0xfffe                -> FAT
0xffff                -> fim do arquivo
```

Informações sobre a estrutura das entradas de diretório:

```
18 bytes         -> nome do arquivo
1 byte           -> atributo do arquivo
7 bytes          -> reservado
2 bytes          -> numero do primeiro cluster ocupado
4 bytes          -> tamanho do arquivo
Byte de atributo do arquivo - valor:  0 - arquivo, 1 - diretório
```

Tipos e estruturas pré-definidas (usadas como referência)

```
/* entrada de diretorio, 32 bytes cada */
typedef struct {
    uint8_t filename[18];
    uint8_t attributes;
    uint8_t reserved[7];
    uint16_t first_block;
    uint32_t size;
} dir_entry_t;
```

```
/* 8 clusters da tabela FAT, 4096 entradas de 16 bits = 8192 bytes*/
uint16_t fat[4096];

/* diretorios (incluindo ROOT), 32 entradas de diretorio
com 32 bytes cada = 1024 bytes ou bloco de dados de 1024 bytes*/
union {
    dir_entry_t dir[CLUSTER_SIZE / sizeof(dir_entry_t)];
    uint8_t data[CLUSTER_SIZE];
} data_cluster;
```

Detalhes do shell

- init - inicializar o sistema de arquivos com as estruturas de dados, semelhante a formatar o sistema de arquivos virtual
- load - carregar o sistema de arquivos do disco
- ls [/caminho/diretorio] - listar diretório
- mkdir [/caminho/diretorio] - criar diretório
- create [/caminho/arquivo] - criar arquivo
- unlink [/caminho/arquivo] - excluir arquivo ou diretório (o diretório precisa estar vazio)
- write "string"[/caminho/arquivo] - escrever dados em um arquivo (sobrescrever dados)
- append "string"[/caminho/arquivo] - anexar dados em um arquivo
- read [/caminho/arquivo] - ler o conteúdo de um arquivo

Avaliação

Deverão ser entregues:

- listagem das rotinas;
- descrição breve dos algoritmos e das estruturas de dados utilizadas;

Distribuição dos pontos:

- execução
  execução correta: 10%
  saída legível: 10%
- estilo de programação
  código bem estruturado: 15%
  código legível: 15%
- documentação
  comentários explicativos: 25%
  análise de resultados: 25%