**SOI PROJEKT**

*Maciej Kaczkowski 300660*

Przykładowy system plików

Spis treści

[Opis koncepcji systemu plików 2](#_Toc137763030)

[Położenie w pamięci, rozmiar, opis wszystkich pól w pamięci karty 3](#_Toc137763031)

[Bootsector 3](#_Toc137763032)

[Tablicę FAT 3](#_Toc137763033)

[Kopię tablicy FAT 3](#_Toc137763034)

[Obszar danych 3](#_Toc137763035)

[Wyliczona procentowo efektywność wykorzystania pamięci 5](#_Toc137763036)

[Maksymalne wykorzystanie pamięci 5](#_Toc137763037)

[Minimalne wykorzystanie pamięci 5](#_Toc137763038)

[Lista operacji możliwych do wykonania na systemie plików 6](#_Toc137763039)

[Funkcje biblioteki opisane w pseudokodzie 6](#_Toc137763040)

# Opis koncepcji systemu plików

System plików jest to metoda przechowywania plików, a także zarządzaniami nimi oraz informacjami o nich. Do głównych zadań systemu plików należy umożliwienie łatwego dostępu do plików użytkownikowi systemu.

Przy projektowaniu opisanego systemu plików przyjęto następujące założenia:

* System jest przeznaczony dla kart z pamięcią 2 KB
* Są dostępne atomowe operacje zapisu i odczytu 16-bitowego słowa
* System jest odporny na przerwanie w dowolnym momencie wykonywanej operacji
* Po awarii, przy próbie ponownego włożenia karty do czytnika, są wykonywane proste operacje naprawcze
* System nie umożliwia dostępu współbieżnego

# Położenie w pamięci, rozmiar, opis wszystkich pól w pamięci karty

Do dyspozycji jest 2 KB, czyli 2048 bajtów. Informacje na nośniku umieszczone są w jednostkach zwanych sektorami. Sektory wielkości 4 bajtów. W związku z tym występuje 2048 / 4 = 512 unikatowych adresów, wskazujących na początek sektora, co ogranicza ilość plików do . W tej sytuacji adres zmieści się na jednym bajcie. Dla bezpieczeństwa każdy adres będzie powtórzony 2-krotnie, zatem minimalna długość adresu to 2 bajty, czyli w 1 sektorze można zapisać maksymalnie 2 adresy.

System plików na dysku zawiera:

## Bootsector

Zawiera podstawowe informacje o partycji

* Informacja o wielkości partycji (4 bajty – 1 sektor)
* Wielkość sektora (4 bajty – 1 sektor)
* Typ partycji (4 bajty – 1 sektor)
* Ilość kopii tablicy FAT (4 bajty – 1 sektor)
* Wskaźniki do wszystkich kopii tablicy FAT (4 bajty – 1 sektor)

## Tablicę FAT

Zawiera informacje o położeniu każdego pliku na karcie, tzn. wskazuje na nagłówek pliku, który zawiera dalsze informacje (vide: ‘obszar danych’)

* Każdy adres zajmuje 2 bajty
* Oprócz adresu w danym wpisie w tablicy FAT znajduje się identyfikator pliku – zajmuje on pozostałe 2 bajty, a więc maksymalna ilość plików (na ten moment) to 2^16
* Tablica FAT ma wielkość 32 sektorów, co ogranicza ilość plików to 32

## Kopię tablicy FAT

Zawiera dokładnie te same informacje co tablica FAT.

## Obszar danych

* Zawiera pliki, które są opisane następującymi danymi:
  + Nazwa pliku (8 bajtów – 2 sektory)
  + Rozszerzenie (4 bajty – 1 sektor)
  + Atrybuty (4 bajty – 1 sektor)
    - Aktualna pozycja w plicku (2 sektory)
    - Czy zablokowany (1 bit)
    - Czy tylko do odczytu (1 bit)
    - Czy ukryty (1 bit)
    - Czy plik systemowy (1 bit)
    - Czy jest to katalog (1 bit)
    - Czy plik jest właśnie modyfikowany (1 bit)
  + Numer sektora gdzie rozpoczyna się plik (4 bajty – 1 sektor)
  + Rozmiar pliku (ilość sektorów) (4 bajty – 1 sektor)

Bootsector zajmuje pierwsze adresy, następnie tablica FAT, kopia tablicy FAT i obszar danych. Zatem:

* 0x0 – 0x4 to adresy bootsectora
* 0x5 – 2x5 to adresy tablicy FAT
* 2x5 – 4x5 to adresy kopii tablicy FAT
* 4x6 – FxF to adresy obszaru danych

# Wyliczona procentowo efektywność wykorzystania pamięci

Zatem wykorzystanie pamięci to:

512 sektorów – 5 sektorów (bootsector) – 32 sektory (tablica FAT) – 32 sektory (kopia tablicy FAT) = 443 sektory (vide: ‘położenie w pamięci’)

Każdy plik zawiera, oprócz danych, nagłówek, który składa się z 6 sektorów (vide: ‘obszar danych’).

## Maksymalne wykorzystanie pamięci

W przypadku, gdyby karta zawierała 1 duży plik, wykorzystanie pamięci przedstawia się następująco:

443 sektory – 6 sektorów = 437 sektorów

Dane zajmują 437/512 \* 100 % = 85,35%

## Minimalne wykorzystanie pamięci

W przypadku, gdyby karta zawierała maksymalną możliwą ilość małych plików (zawierających 1 sektor danych), wykorzystanie pamięci przedstawia się następująco:

443 / (6+1) = 63

Dane zajmują 63/512 \* 100 % = 12,3%

# Lista operacji możliwych do wykonania na systemie plików

* Otwarcie pliku
* Zamknięcie pliku
* Odczyt pliku
* Zapis do pliku
* Usunięcie pliku

## Funkcje biblioteki opisane w pseudokodzie

*open\_file(file\_name):*

*if is\_file\_in\_FAT(file\_name):*

*file\_desc = get\_file\_desc();*

*file\_desc.is\_file\_open = True;*

*return file\_desc;*

*close\_file(file\_name):*

*if is\_file\_in\_FAT(file\_name):*

*file\_desc = get\_file\_desc();*

*file\_desc.is\_file\_open = False;*

*return file\_desc;*

*read\_file(file\_handle):*

*current\_position = file\_handle.begin*

*// read file chunks with 16-bit atomic operation*

*// return data to operation memory*

*write\_file(file\_handle, data):*

*current\_position = file\_handle.current\_position*

*while data is not empty:*

*// write data with 16-bit atomic operation*

*file\_desc.current\_position += 16 // bits*

*delete\_file(file\_handle):*

*if not file\_handle.read\_only and not file\_handle.is\_open:*

*// start remove file contents with 16-bit atomic operation*

*remove\_from\_FAT(file\_handle);*