Michał Przychodzeń

PROJEKT SOI
SYSTEM PLIKÓW

Zadanie polega na zaprojektowaniu systemu plików o wielkości 128-1024 bajtów, który byłby odporny na awarie.

Zakładam wielkość systemu plików "pastylki" 256 bajtów.
Zakładam również, że dane w takim systemie będą przede wszystkim odczytywane, także będzie on optymalizowany głównie pod względem odczytu danych.
Z powodu małej ilości miejsca należy zminimalizować wielkość struktur opisujących system i poczynić założenia odnośnie przechowywania danych i ograniczenia co do wielkości, ilości plików.
Pliki będą zaalokowane jako ciąg kolejnych bajtów.
Plik będzie opisany desktyptorem zawierającym
- nazwę(1 B – może to być liczba lub litera – 256 różnych wartości)
- bajt początkowy – baza (1 B – wystarczy do opisania wszystkich plików w systemie – maksymalna wartość = ilość miejsca)
- długość pliku w bajtach (1B – z nadmiarem bo nie będzie pliku o wielkości 256B, także można poczynić dodatkowe założenia co do wielkości plików i zyskując kilka bitów poświęcić je na flagi)

Zatem wielkość desktyptora pliku to 3 bajty, przyjmuję maksymalną ilość plików 8.
Wielkość deskryptorów wynosi 24 bajty.
Dodatkowo potrzebuję 4 bajtów jako superblok do kontroli spójności i poprawności danych w systemie plików. Opiszę go poniżej.

Zajęte i wolne miejsce będzie odwzorowane w postacji mapy bitowej o wielkości: 32 bajty (zapamiętujemy cały system plików)

budowa systemu plików:
[superblok 4 bajty][mapa bitowa 32 bajty][deskryptory plików 24 bajtów][miejsce na dane 196 bajtów]

Usuwanie będzie równoznaczne z wyzerowaniem – usunięciem poprzednich danych z systemu plików – mogą to być np. poufne informacje, które bezwzględnie należy trwale wymazać z pamięci, aby nikt nie mógł ich już odczytać.

Dodatkowo kosztem wolniejszego zapisu plików i usuwania zmieniam za każdym zapisaniem bajtu informacji wartość w superbloku. Jest to czynność, której mogłoby nie być – można by po prostu usuwać cały plik i po usunięciu zmienić 1 bajt na usuwanie deskryptora, ale wtedy np. tworząc/usuwając plik i w przypadku częstego występowania awarii nie zapiszemy/usuniemy nic. W zaprojektowanym systemie istnieje możliwość zapisywania/usuwania nawet w przypadku przerw w transmisji/awarii.
Z tym że ceną jaką za to płacimy jest szybkość tych operacji.

Podstawowe operacje na systemie plików:
- montowanie
- odczyt pliku
- usuwanie pliku
- rozszerzanie wielkości pliku
- zapis pliku
- odmontowanie

superblok:
nad poprawnością będzie czuwał superblok i zawarte w nim informacje:
1 bajt będzie wskazywać na wykonywaną operację:
REMOVING_FILE
CREATING_NEW_FILE
REMOVING_DESCRIPTOR
CREATING_DESCRIPTOR
EVERYTHING_OK
READING_FILE
EXTENDING_FILE
MODIFING_DESCRIPTOR
ADDING_SPACE
REMOVING_SPACE

przy nagłym odłączeniu urządzenia błędy w systemie mogą wystąpić podczas usuwania, tworzenia – te operacje mogą pozostawić system niespójny.
Usunieta może być część pliku a zostanie np część pliku lub zajęty deskryptor.
Tworzony plik może nie zostać zapisany do końca w wyniku czego będziemy mieli tylko jego część co nie jest prawidłowe.

- Tworzenie pliku

superblok:
1 bajt ustawiony na **CREATING_NEW_FILE**
2 bajt zawiera początek miejsca w które kopiujemy dane pliku
3 bajt zawiera ilość bajtów do przekopiowania
4 bajt zawiera ilość przekopiowanych bajtów

po każdym przekopiowaniu bajtu zwiększana jest wartość 4 bajtu superbloku
jeśli jest ona równa 3 bajtowi następuje ustawienie wartości 1 bajtu na **CREATING_DESCRIPTOR** następnie nowo utworzonemu miejscu przypisywany jest deskryptor. W którym ustawiane są dane tak jak 2,3 bajt superbloku.
Po utworzeniu deskryptora ustawiamy 1 bajt superbloku na **REMOVING_SPACE** i korzystając z zapamiętanych danych odnośnie nowo utworzonego pliku ustawiamy odpowiednie bity w mapie bitowej na 0, a po wykonaniu całej operacji zmieniamy bajt 1 superbloku na **EVERYTHING_OK**

przydzielanie miejsca dla nowego pliku jest możliwe tylko gdy dostępna jest odpowiednia ilość bajtów, jeśli nie nie tworzymy nic.

+ w przypadku występienia błędu możemy naprawić system plików bo wiemy że miejsce zostało już przydzielone do końca należy naprawiając system utworzyć albo skończyć tworzenie deskryptora

- Usuwanie:

superblok:
1 bajt ustawiony na **REMOVING_FILE**
2 bajt pokazuje początek miejsca od którego zaczynamy usuwanie danych
3 bajt ilość bajtów do usunięcia
4 bajt ilość bajtów usuniętych

po wyzerowaniu każdego bajtu począwszy od miejsca wskazywanego przez bajt 2, gdy 3 i 4 bajt są równe ustawiany jest bajt 1 na **REMOVING_DESCRIPTOR**
i następnie usuwany jest deskryptor wskazujący na plik zaczynający się od miejsca wskazywanego przez 2 bajt
po usunięciu deskryptora 1 bajt superbloku zostaje ustawiony na **ADDING_SPACE** i korzystając z pozostałych danych superbloku ustawiamy odpowiednie bity w mapie bitowej na 1 a po wykonaniu całej operacji zmieniamy wartość 1 bajtu na **EVERYTHING_OK**

Po usunięciu pliku odnotowujemy w mapie bitowej wolne miejsce – nic nie przesuwamy! - które

zostanie lub nie przydzielone następnemu plikowi. Powoduje to fragmentację.

+ błędy w każdej sytuacji są możliwe do usunięcia – pamiętamy skąd i ile usuwamy, ile usuneliśmy oraz czy usunięty już został deskryptor, jeżeli nastąpi awaria podczas usuwania przy następnym włączeniu usuwamy pozostałą część plików.
Gdy usuniemy wszystkie dane ale nie usuniemy deskryptora – sytuacja jest niedopuszczalna – ale pamiętamy o tym i usuniemy go przy następnym podłączeniu.

- Czytanie pliku

superblok:
1 bajt ustawiony na **READING_FILE**
2 bajt zawiera początek miejsca skąd czytamy
3 bajt zawiera ilość bajtów do odczytania
4 bajt nie używany(ew. zawiera ilość odczytanych bajtów)

Czytanie pliku nie powoduje żadnego rozspójnienia gdyż nie zmienia zawartości systemu, jeśli nastąpi błąd odczytu. Ew. Możliwa reazlizacja odczytu – wolniejsza – ale ponowienie czytania od przerwanego miejsca.
po odczytaniu całego pliku ustawiamy 1 bajt superbloku na **EVERYTHING_OK**

- Rozszerzanie pliku

Rozszerzanie pliku następuje podobnie do tworzenia nowego – gdy dostępna jest pamięć za plikiem i jest ona wystarczająca zostaje ustawiony deskryptor na plik który rozszerzamy i na wartość o jaką rozszerzamy, następuje zliczanie bajtów przy kopiowaniu z pamięci na system plików(pastylkę), w czasie tej realizacji 1 bajt ustawiony jest na **EXTENDING_FILE** natomiast po skopiowaniu bajtów zmieniamy wartość 3 bajtu na ilość dodanych bajtów + stara długość, po czym zmieniamy 1 bajt na **MODIFING_DESCRIPTOR** oraz ustawiamy długość pliku wskazywanego przez superblok – bajt 2 - tak jak 3 bajt superbloku. Po czym odpowiednio modyfikujemy mapę bitową tak jak w poprzednich operacjiach – ustawiając 1 bajt na **REMOVING_SPACE**. Po poprawnie wykonanych operacjach ustawiamy wracamy do stanu **EVERYTHING_OK**.

- Montowanie systemu

jest to najważniejsza część systemu, która sprawdza jego poprawność i jeżeli trzeba przywraca jego spójność.
Gdy 1 bajt jest ustawiony na EVERYTHING_OK i 2,3,4 mają 0 (jeśli nie to je zerujemy), kontynuujemy uruchamianie systemu plików.
Jeżeli 1 bajt jest różny od EVERYTHING_OK to znaczy że nastąpiła awaria podczas operacji zapamiętanej w nim.
Jeśli był to odczyt to spowrotem ustawiamy flagę na EVERYTHING_OK i kontynuujemy pracę – gdyż ta czynność nie powodowała żadnych zmian.
Jeśli usuwaliśmy plik to kontynuujemy usuwanie danych, bądź jeśli 1b==REMOVING_DESCRIPTOR to usuwamy deskryptor usuniętego pliku i wracamy do sytuacji spójnej.
Jeśli tworzyliśmy nowy plik to analogicznie jak w poprzedniej sytuacji przywracamy system plików do stanu spójności. - jeśli błąd był przy kopiowaniu bajtów to albo kontynuujemy kopiowanie od przerwanego miejsca, albo od nowa(może się zdarzyć że skopiowaliśmy jakiś bajt ale błąd wystąpił zanim zmodyfikowaliśmy dane w superbloku, ale wtedy nadpiszemy ten jeden bajt, co nie przeszkadza)
podobnie rozpatrujemy inne możliwe sytuacje i przywracamy system do stanu EVERYTHING_OK.
(Podczas ustawionego stanu EVERYTHING_OK pozostałe bajty powinny mieć wartość 0, i przy ustawianiu wartości pliku zaczynamy od 3 bajtu(długości), potem 2(baza)) – dzięki temu możemy również wykryć błędy powstałe podczas modyfikacji deskryptora.