ЛЕКЦІЯ №7. НАПИСАННЯ ДРАЙВЕРІВ ПРИСТРОЇВ У РЕАЛЬНОМУ РЕЖИМІ РОБОТИ ПРОЦЕСОРА

Написання драйверів пристроїв.

Драйвери це відокремлені частини BIOS, що виконують функції керування й обслуговування окремих пристроїв. У BIOS є драйвери для клавіатури, екрана, принтера, послідовного інтерфейсу, касетного магнітофона, дискетних пристроїв і системного годиника. Звертання до кожного драйвера здійснюється програмним перериванням.

Деякі з пристроїв потребують асинхронного обслуговування. Наприклад, кожне натиснення клавіші на клавіатурі потрібно реєструвати негайно. Це означає, що роботу поточної програми потрібно перервати і передати керування клавіатурному драйверу.

Він реєструє натиснуту клавішу і повертає керування до перерваної програми. Клавіатура під'єднана до комп'ютера таким чином, що кожне натиснення клавіші викликає апаратне переривання і керування передається за адресою, зазначеною у векторі переривання (вектор 9).

BIOS ініціалізувала під час POST вектор 9 так, щоб він вказував адресу її клавіатурного драйвера. Клавіатурний драйвер обробляє переривання, запам'ятовує необхідну інформацію про натиснуту клавішу і повертає керування перерваній програмі інструкцією IRET. Подібним чином працюють і інші драйвери BIOS.

Драйвер дискетних пристроїв обробляє апаратне переривання 0Eh і виконує обслуговуючі функції за допомогою програмного переривання 13h, драйвер системного годинника обробляється апаратним перериванням 08h і виконує обслуговуючі функції за допомогою програмного переривання 1Ah.

Розбіжності між функціями.

Обслуговуючі функції BIOS працюють з пристроями дуже примітивно - вони можуть виконувати різноманітні операції читання-запису, але не можуть організувати легкокеровані структури даних.

Система використовує програми BIOS для керування пристроями. Функції системи – це проміжний прошарок між прикладною програмою і функціями BIOS.

Важливим аргументом на користь використання функцій системи є несумісність BIOS деяких ПК. Але потрібно зауважити, що функції системи працюють повільніше, ніж потрібно для великої кількості задач. Наприклад, гранична швидкість передачі даних через послідовний порт досяжна тільки при прямій роботі програми з послідовним портом.

В основному система використовує функції BIOS у стандартних драйверах пристроїв.

Вимоги перетворення викликів функцій системи на команди драйверів.

Драйвери пристроїв можуть виконувати тільки прості команди системи. Найчастіше при роботі з пристроями застосовуються функції системи читання і запису. Цим функціям можуть відповідати декілька команд драйвера. Розглянемо приклад використання системної функції 40h (запис даних на диск). Система перетворить цей запит у декілька команд драйвера диска. Перша з них - команда пошуку вільного місця на диску. При виконанні цієї команди драйвер повинен прочитати FAT даного диска.

При наявності вільного місця система дає драйверу команду запису даних у файл. Потім система повинна буде змінити на диску час останнього доступу до файла, знову давши команду драйверу.

Як приклад - гнучкий диск: 1 – прочитати FAT; 2 – записати в файл на диск; 3 – обновити каталог; 4 – обновити FAT.

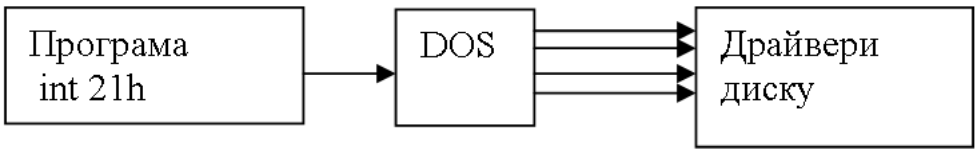

Рис. 2.1 Приклад роботи з драйвером диску

Старі та нові драйвери пристроїв.

Система працює з загальним зв′язаним списком драйверів. Першим в списку розташований драйвер пристрою NUL:. Він містить вказівник на наступний драйвер,

той, в свою чергу, на наступний. Вказівник останнього драйвера містить значення – 1, яке означає кінець списку.

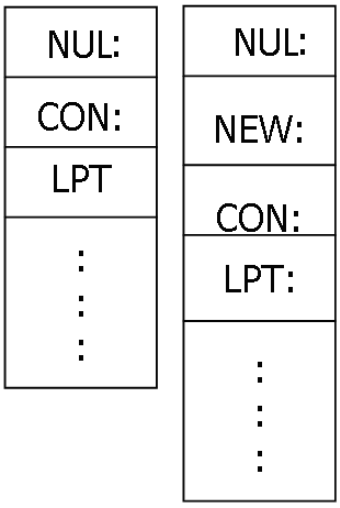

Рис. 2.2 Список драйверів (встановлення нового драйверу)

Ці програми драйверів пристроїв розташовуються в системній таблиці пам′яті ПК. При встановленні нового драйверу, система завжди встановлює його в список після пристрою NUL:. Це дозволяє замінити стандартні драйвери, так як при запиті на доступ до пристрою системи починає шукати драйвер від початку списку. При заміні стандартного драйвера на новий система першим знайде новий драйвер. Аналогічно до списку вносяться драйвери нових пристроїв.

Для доступу до будь-якого драйверу з ланцюжка, система передає тільки адресу початку ланцюжка, тобто драйвера NUL.

Всі стандартні драйвери можуть бути замінені на альтернативні, за винятком драйвера NUL:, що завжди міститься на початку ланцюжка пристроїв.

Структура програми драйвера. Програма драйвера складається з 5 частин.

1. Заголовка пристрою (заголовка драйвера).

2. Області пам'яті для збереження даних і локальних процедур.

3. Процедури стратегії.

4. Процедури переривання.

5. Програм обробки команд.

Початок програми драйвера несхожий на початок звичайних програм. Там розташований заголовок драйвера, який містить інформацію про сам драйвер пристрою.

Ця інформація призначена для системи і включає ім'я пристрою і покажчик на наступний драйвер.

Процедури стратегії і переривання об'єднуються при обробці кожної команди. Вони дозволяють системі передати керування драйверу. Остання частина драйвера містить програми, що виконують ті команди, які система передає драйверу.

Взаємодія системи з драйвером.

При виклику драйвера системи передає йому пакет даних. Цей пакет даних називається заголовком запиту і містить інформацію для драйвера пристрію: номер команди і дані, що записуються на пристрій. При виклику драйвера системи розміщує адресу заголовка в регістри ES і BX.

Рис. 2.3 Заголовок запиту (передача пакету даних)

Не можна плутати заголовок запиту і заголовок драйвера. Заголовок драйвера містить інформацію для системи про програму драйвера, а в заголовку запиту системи розміщує інформацію, необхідну для роботи драйвера.

Складові частини заголовка запиту.

1. Розмір у байтах заголовка запиту (залежить від обсягу даних у запиті)- 1 байт.

2. Номер пристрою - 1б.

3. Код команди - 1б.

4. 16-розрядне слово стану при завершенні - 2б.

5. Зарезервовано.

6.Дані, що залежать від команди. Розмір змінний.

Заголовок запиту має розмір змінної довжини. У першому елементі міститься довжина заголовку запиту (1б). Другий елемент містить номер пристрою (1б). Він зазвичай використовується у випадку під′єднання до контролера декількох пристроїв.

Приклад - контролер дисководу для гнучких дисків, який часто керує двома дисководами. У такому випадку диски A: і B: будуть мати номери 0 і 1. У третьому

елементі міститься код команди, яку повинний виконати драйвер (1б). Четвертий елемент служить індикатором стану (2б). Використання п'ятого елемента не з′ясовано в документації (8б). Довжина шостого елемента залежить від команди в третьому елементі. Ці дані залежать від команди.

Системиа автоматично формує заголовок запиту при запиті програми на виконання дій, пов'язаних із драйвером пристрою. Цей пакет даних розташовується в зарезервованій для системи області пам'яті і будується на основі інформації від викликаючої програми. При передачі керування драйверу системи передає йому адресу заголовку запиту, яка розміщується в локальній пам'яті драйвера. При цьому повинна вказуватися сегментна і відносна адреса заголовка запиту, тому що заголовок запиту може розташовуватися в будь-якому місці пам'яті. Сегментну і відносну адреси системи розміщує відповідно в регістри ES і BX.

Двоступеневий виклик драйвера пристрою.

Щоразу, коли система вимагає від драйвера виконання якої команди, вона викликає драйвер двічі. При цьому спочатку керування передається на процедуру стратегії, потім - на процедуру переривання. Можна вважати процедуру стратегії набором команд для підготування до ініціалізації драйвера. Інформація з процедури стратегії потім використовується процедурою переривання при обробці команди.

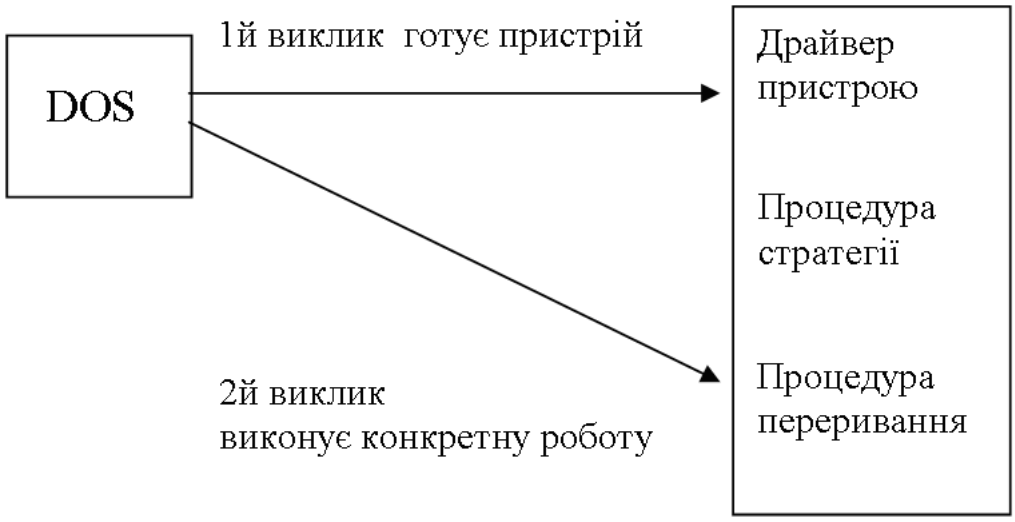

Рис. 2.4 Двоступеневий виклик драйвера пристрою

Такий двоступеневий механізм дозволяє системі рознести в часі перший запит драйвера (його підготування) і безпосередню роботу драйвера.

Всі виклики процедур стратегій зв'язуються в ланцюжок стратегій у порядку їхнього надходження, а виклики процедур переривань зв'язуються в ланцюжок переривань за їхніми пріоритетами.

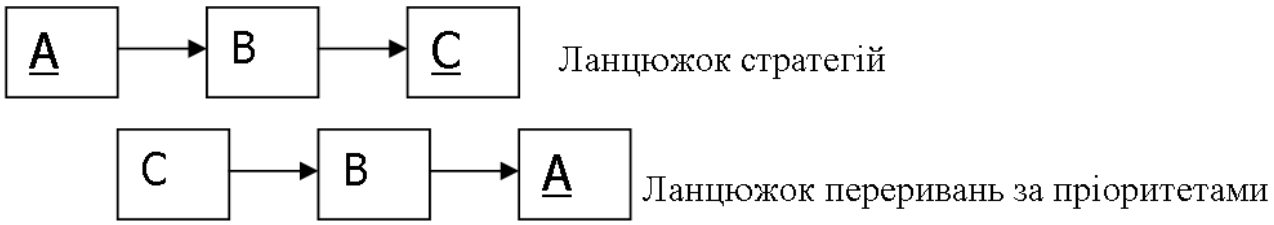

Рис. 2.5 Обробка запитів

Після викликів усіх драйверів через процедури стратегії системи знаходить у ланцюжку переривань драйвер із найбільш високим пріоритетом.

Наприклад, для реалізації багатозадачності в системі. Задача, яка приймає дані через модем, може бути важливіша за інші.

При першому виклику драйвера процедура стратегії зберігає адресу заголовка запиту, яка знаходиться в регістрах ES і BX.

Рис. 2.6 Підготування до першого виклику драйвера

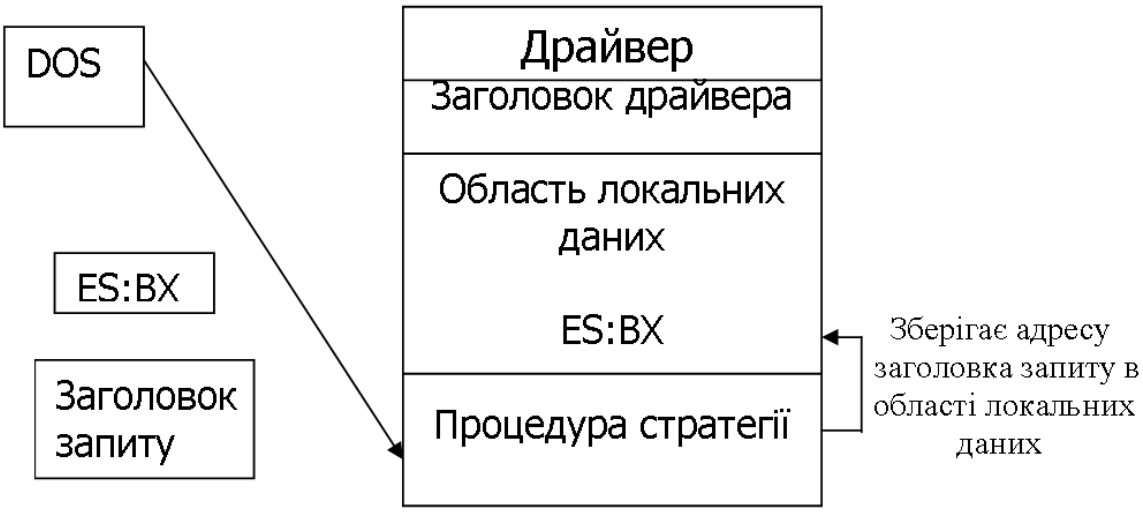

Рис. 2.7 Зберігання адреси заголовка запиту в області локальних даних (процедура стратегії)

Другого разу система викликає драйвер пристрою через процедуру переривання. Тоді і починається робота драйвера. Процедура переривання обробляє заголовок запиту, який містить інформацію для драйвера і передає керування програмам обробки команд.

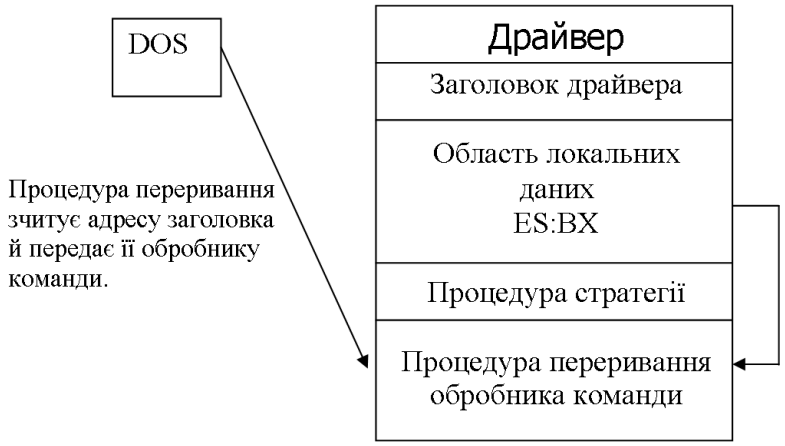

Рис. 2.8 Вилучення заголовку запиту процедурою переривання

Команди драйвера пристрою.

0 Ініціалізація.

1 Перевірка носія (блочні пристрої).

2 Одержання BPB (блочні пристрої).

3 IOCTL - введення.

4 Введення.

5 Неруйнуюче введення (символьні пристрої).

6 Стан введення (символьні пристрої).

7 Очищення введення (символьні пристрої).

8 Виведення.

9 Виведення з перевіркою.

10 Стан виведення (символьні пристрої).

11 Очищення виведення (символьні пристрої).

12 IOCTL - виведення.

13 Відкриття пристрою.

14 Закриття пристрою.

15 Змінний носій (блочні пристрої).

16 Виведення, доки не зайнято (символьні пристрої).

17-18 Не визначені.

19 Узагальнений IOCTL (блочні пристрої).

20-22 Не визначені.

23 Одержання логічного пристрою.

24 Встановлення логічного пристрою.

25 IOCTL запит.

Огляд драйверів пристроїв.

Кістяк, на основі якого розробляється драйвер це:

1. Заголовок коментаря.

2. Директиви асемблера.

3. Основна процедура.

4. Заголовок драйвера.

5. Робоча область драйвера.

6. Процедура стратегії.

7. Процедура переривання.

8. Локальні процедури.

9. Обробка команд системи.

10. Вихід при помилці.

11. Загальний вихід.

12. Кінець програми.

Огляд частин драйвера пристрою

Заголовок коментаря. Це стислий опис призначення програми, дата створення, ім'я автора і т.д.

Директиви асемблера. У цьому розділі утримуються директиви асемблера, що визначають конкретні вимоги до структур даних драйвера. Тут наводяться визначення сегмента й основної процедури і встановлюються адреси в регістрах CS, ES, DS. Ці директиви підходять для більшості створюваних драйверів, тому майже в усіх драйверах можна звертатися до фрагментів програми і до даних однаково.

Важливою особливістю при написанні цього розділу драйвера є використання структур при описанні даних. Структури не забезпечують виділення простору пам'яті для даних, а лише визначають засіб розгляду цих даних.

Складові частини будь-якого драйвера.

Заголовок драйвера.

До заголовоку драйвера включаються дані, передані системою відразу після початку роботи з пристроєм. По ним операційна система визначає, як працювати з даним пристроєм.

П'ять основних частин заголовка драйвера:

1. Покажчик на наступний драйвер. next_dev dd -1
2. Атрибути драйвера. attribute dw 8000h
3. Покажчик на процедуру стратегії. strategy dw dev_strategy
4. Покажчик на процедуру переривання. interrupt dw dev_int
5. Ім'я пристрою. dev_name db ‘SIMPLE$’

Три з п'ятьох розділів заголовка драйвера містять покажчики адрес.

Перший містить двослівний покажчик на наступний драйвер у файлі. Коли система завантажує драйвер із файла в пам'ять, у цьому файлі можуть утримуватися й інші драйвери.

Наприклад, стандартні драйвери для консолі, накопичувача на гнучких дисках, принтера, комунікаційних портів і годинника утримуються в єдиному файлі IBMBIO.COM.

Система використовує покажчик, розташований у поточному драйвері для пошуку наступного драйвера. Для того, щоб повідомити систему про те, що

наступного драйвера немає, потрібно помістити - 1 в обидва слова цього першого поля.

Другий і третій покажчики використовуються системою для визначення місця розташування процедур стратегії і переривання. Ці поля містять зсуви, по яких можна знайти розташування цих процедур у пам'яті.

Поле атрибутів пристрою

У цьому полі утримується інформація про тип пристрою, керованого драйвером і в ньому визначені типи команд, які повинні бути реалізовані в драйвері пристрою. У ранніх версіях системи біти цього поля визначали тільки тип пристрою. У наступних версіях деякі біти стали використовуватися для вказання команд, які обробляються драйвером. Розглянемо призначення кожного біту:

Біт 15 дозволяє системі визначити тип пристрою, керованого драйвером: 0 - символьний пристрій, 1 - блочний. Від значення цього біту залежить інтепретація вмісту більшості інших бітів. Поле імені заголовка пристрію теж визначається типом пристрою.

Біт 14 використовується для того, щоб повідомити систему, чи підтримуються драйвером команди керування введенням/виведенням. Керування в/в використовується для передачі в драйвер і отримання з нього керуючої інформації. Якщо цей біт установлений, то потрібно реалізувати процедури обробки двох IOCTL команд.

Біт 13. Інтепретація системи умісту біту 13 визначається типом пристрою. Для блочно-орієнтованих пристроїв наявність одиниці в біті 13 свідчить про те, що дисковий накопичувач не є стандартним пристроєм IBM. Нуль у біті 13 означає, що системиа працює зі стандартним пристроєм. Якщо пристрій символьний встановлення біту означає, що драйвер може обробляти команду “Виведення доки не зайнято”. Якщо біт 13 не встановлений, то для визначення типу носія драйвер пристрою використовує дескриптор носія з FAT. Якщо біт 13 установлений, то використовується BPB. Але так як диски можуть мати будь-який дескриптор носія, то щоб врахувати всі можливі варіанти, можна встановити біт 13, дозволивши системі використовувати BPB для визначення розташування всіх компонентів диска.

Біт 12 не визначений і повинний містити 0.

Біт 11 використовується для того, щоб показати, чи підтримує драйвер команди відкриття і закриття пристрою і зміни носія

Біти з 10 по 8 не визначені і повинні містити 0.

Біт 7 використовується драйвером при роботі з системою починаючи з версії 5.0. Встановлення біту дає можливість програмі користувача запитати, чи доступні для використання визначені функції IOCTL

Біт 6 визначає, чи використовує драйвер команди одержання логічного пристрою й встановлення логічного пристрою. Він також показує, що драйвер підтримує команди узагальненого IOCTL як для символьних, так і для блочних пристроїв.

Біт 5 не визначений і повинний містити 0.

Встановлення біта 4 визначає, що драйвер підтримує швидке введення-виведення через консоль, реалізоване за допомогою переривання 29h.

Біт 3 установлений, якщо драйвер обслуговує годинник. У цьому випадку система заміняє стандартний драйвер годинника поточним драйвером годинника.

Біт 2 установлений, якщо драйвер обслуговує стандартний пристрій NUL. Не можна замінити драйвер пристрою NUL, тому даний біт використовувати не можна. Біт 2 встановлюється тільки в драйвері пристрою NUL, що дозволяє системі його ідентифікувати.

Біт 1 установлений, якщо драйвер заміняє стандартний консольний пристрій виведення. Встановлення цього біта означає, що драйвер блочного пристрою реалізує 32-розрядну адресацію сектора. Тобто, таким чином підтримуються диски з обсягом більш 32 Мбайт.

Біт 0. Для символьних пристроїв встановлення цього біта означає, що даний драйвер заміняє драйвер стандартного пристрою виведення.

Від встановлення визначених бітів поля атрибутів залежить можливість посилки драйверу пристрою різних типів команд. Тобто, ці біти використовуються для визначення команд, не характеристик пристрою. Отже, слово атрибутів являє собою потужний засіб, який дозволяє кожному драйверу ідентифікувати себе.

Табл. 2.1 Значення слова атрибутів для різних пристроїв

Ім'я пристрою	Слово атрибутів	Встановлені біти
1. Пристрій NUL:	8004h	15 символьний пристрій 2 пристрій NUL:
2. CON:	8013h	15 символьний пристрій 4 швидке введення / виведення 0 стандартне виведення
3. AUX:	8000h	15 символьний пристрій
4. LPTx:	A0C0h	15 символьний пристрій 13 виведення, доки не зайнято 7 IOCTL –запит 6 узагальнений IOCTL
5. COMx:	8000h	15 символьний пристрій
6. CLOCK$	8008h	15 символьний пристрій 3 годинник
7.Disk	08C2h	блочний пристрій 11 відкриття / закриття змінний носій 7 IOCTL –запит 6 одержання/встановлення логічного пристрою 1 32 розрядна адресація сектора

Поле імені заголовку драйвера

Поле імені займає 8 байтів. Для символьних пристроїв це поле містить текст - ім'я пристрою. Якщо ви замінюєте будь-який стандартний пристрій системи, туди необхідно записати ім'я заміненого пристрою: CON:, PRN: і т.д. Якщо стандартний пристрій не замінюється, у цьому полі записується ім'я, яке буде використовуватися для ідентифікації пристрою. Воно не повинно збігатися з жодним із використовуваних імен. Ім'я пристрою повинно бути написане великими літерами. Якщо воно займає менше 8 байтів, у позиції, що залишилися, необхідно записати прогалини. При використанні блочних пристроїв це поле призначається не для збереження імені пристрою. У першому байті поля записується кількість пристроїв, керованих драйвером. Так як блочні пристрої є дисковими накопичувачами, кількість пристроїв, уже встановлених системах, визначає літеру, що буде ідентифікувати конкретний драйвер.

Якщо за поточним драйвером йде інший драйвер пристрою дискового типу, то літера що відповідає йому визначається сумою числа уже встановлених у систему пристроїв і числа пристроїв, керованих поточним драйвером.

В робочій області драйвера визначаються локальні змінні для процедур.

```
Процедура стратегії для роботи в середовищі
dev_strategy: mov cs:rh_seg,es ;зберегти сегмент
mov cs:rh_ofs,bx ; зберегти зсув
ret; повернутися в систему.
```

```
Процедура переривання - другий виклик з системи
dev_interrupt: cld ;зберегти стан при виході
push ds
push es
push ax
push bx
push cx
push dx
push di
push si
;Встановлюємо ES:BX на заголовок
;запиту
mov ax,cs:rh_seg ;відновити в ES,BX
mov es,ax ;значення, збережені

mov bx,cs:rh_ofs ;при виклику процедури страт.
                                ;Перехід на відповідну процедуру виконання команди.
mov al, es:[bx].rh_cmp ;одержання коду команди з
                                            ;заголовка запиту
rol al,1; команда зсуву ліворуч:помножити x2 для отримання
           ; індексу в таблиці CMDTAB.
lea di,cmdtab ;адреса таблиці команд.
mov ah,0 ;очистити старшу половину.
add di,ax ;додати індекс до адреси табл.
jmp word ptr[di] ;перехід на адресу з таблиці.
```

Cmdtab - це таблиця, яка містить адреси кожної з команд (таблиця переходів). Команда, яку потрібно виконати, буде отримана з заголовка запиту. Процедура переривання забезпечить перехід за адресою, що відповідає отриманій драйвером команді на програму, яка забезпечує виконання цієї команди. Таблиця складається з 16-байтових змінних. Якщо використовувати код команди для вибору адреси, індексування таблиці буде виконано по байтам. В результаті отримаємо тільки половину 2х-байтової адреси. Для правильного індексування треба перетворити байтове значення коду команди в 2х-байтове.

```
CMDTAB label byte ;тільки для символьних пристроїв.
dw initialization ;ініціалізація
dw media_check ;перевірка носія (блочні пристрої)
dw get_BPB ;одержання BPB
dw IOCTL_INPUT ;IOCTL - Введення
dw INPUT ;Введення (читання)
dw ND_INPUT ; Введення, що неруйнує
dw INPUT_STATUS ;стан введення
dw INPUT_FLUSH ;очищення введення
dw OUTPUT ;виведення(запис)
```

```
dw OUTPUT_VERIFY ;виведення з перевіркою
dw OUTPUT_STATUS ;стан виведення
dw OUTPUT_FLUSH ;очищення виведення
dw IOCTL-OUTPUT ;IOCTL - виведення
dw OPEN ;відкриття пристрою
dw CLOSE ;закриття пристрою
dw REMOVABLE ;змінний носій
dw OUTPUT_BUSY ;виведення доки не зайнято
dw COMMAND17 ;не визначено
dw COMMAND18 ;не визначено
dw GENERIC_IOCTL ;узагальнений IOCTL
dw COMMAND20 ;не визначено
dw COMMAND21 ;не визначено
dw COMMAND22 ;не визначено
dw GET_DEVICE ;одержання логічного пристрою
dw SET_DEVICE ;встановлення логічного пристрою
dw IOCTL_QUERY ;IOCTL - запит.
```

Ця процедура переривання дозволяє створювати драйвери пристроїв для системи всіх версій. Таблиця адрес процедур виконання команд містить адреси для команд з номерами від 0 до 25.

Виконання команд.

При звертанні до драйвера системи посилає номер команди в заголовок запиту. При цьому системва очікує, що драйвер виконає всі дії, які вимагає команда. Жоден драйвер не призначається для всіх 26 команд. Кількість команд, виконання яких повинно бути реалізовано драйвером, визначається чотирма чинниками: 1 - набором припустимих для драйвера операцій; 2 - типом керованого пристрою; 3 - встановленими бітами слова атрибутів; 4 – версією системи. У драйверах для пристроїв, які забезпечують тільки виведення інформації необхідно реалізувати виконання тільки команд виведення (виведення, виведення з перевіркою, стан виводу). Для того, щоб уникнути використання яких-небудь команд, необхідно скинути відповідні біти в слові атрибутів.