**Лекция №7. НАПИСАНИЕ ДРАЙВЕРIВ УСТРОЙСТВ В РЕАЛЬНОМ режим работы процессора**

***Написание драйверiв устройств.***

Драйверыэто отдельные части BIOS, выполняющие функции управления и обслуживания отдельных устройств. В BIOS есть драйверы для клавиатуры, экрана, принтера, последовательного интерфейса, кассетного магнитофона, дисковод гибких устройств и системного Часы. Обращение к каждому драйвера осуществляется программным прерыванием.

Некоторые из устройств требуют асинхронного обслуживания. Например, каждое нажатие клавиши на клавиатуре нужно регистрировать немедленно. Это означает, что работа текущей программы нужно прервать и передать управление клавиатурном драйвера.

Он регистрирует нажатой клавишу и возвращает управление к прерванной программы. клавиатура под"Объединенная к компьютеру таким образом, что каждое нажатие клавиши вызывает аппаратное прерывание и управление передается по адресу, указанному в векторе прерывания (вектор 9).

BIOS инициализирующая во время POST вектор 9 так, чтобы он указывал адрес ее клавиатурного драйвера. Клавиатурный драйвер обрабатывает прерывания, запоминает необходимую информацию о нажатой клавишу и возвращает управление прерванной программе инструкцией IRET. Подобным образом работают и другие драйверы BIOS.

Драйвер дисковод гибких устройств обрабатывает аппаратное прерывание 0Eh и выполняет обслуживающие функции с помощью программного прерывания 13h, драйвер системных часов обрабатывается аппаратным прерыванием 08h и выполняет обслуживающие функции с помощью программного прерывания 1Ah.

***Разногласия между функциями.***

Обслуживающие функции BIOS работают с устройствами очень примитивно - они могут выполнять различные операции чтения-записи, но не могут организовать легкоуправляемые структуры данных.

Система использует программы BIOS для управления устройствами. Функции системы - это промежуточный слой между приложением и функциями BIOS.

Важным аргументом в пользу использования функций системы является несовместимость BIOS некоторых ПК. Но нужно заметить, что функции системы работают медленнее, чем нужно для большого количества задач. Например, предельная скорость передачи данных через последовательный порт достижима только при прямой работе программы с последовательным портом.

* основном система использует функции BIOS в стандартных драйверах устройств.

***Требования преобразования вызовов функций системы на команды драйверов.***

Драйверы устройств могут выполнять только простые команды системы. Чаще всего при работе с устройствами применяются функции системы чтения и записи. Этим функциям могут отвечать несколько команд драйвера. Рассмотрим пример использования системной функции 40h (запись данных на диск). Система преобразует этот запрос в несколько команд драйвера диска. Первая из них - команда поиска свободного места на диске. При выполнении этой команды драйвер должен прочитать FAT данного диска.

При наличии свободного места система дает драйверу команду записи данных в файл. Затем система должна будет изменить на диске время последнего доступа к файлу, снова дав команду драйверу.

Как пример - гибкий диск: 1 - прочитать FAT; 2 - записать в файл на диск; 3 - обновить каталог; 4 - обновить FAT.

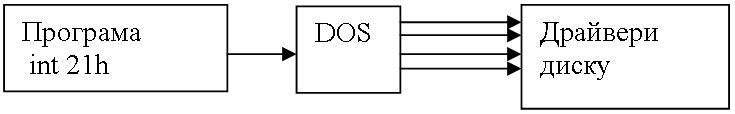


Рис. 2.1 Пример работы с драйвером диска

***Cтарi и новые драйверы устройств.***

Система работает с общим н"связанным списком драйверов. Первым в списке расположен драйвер устройства NUL :. Он содержит указатель на следующий драйвер,

тот, в свою очередь, на следующий. Указатель последнего драйвера содержит значение - 1, означает конец списка.

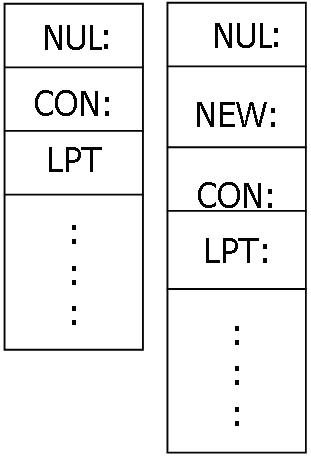


Рис. 2.2 Список драйверов (установление нового драйвера)

Эти программы драйверов устройств располагаются в системной таблице памяти"памяти ПК. При установке нового драйвера, система всегда устанавливает его в список после устройства NUL :. Это позволяет заменить стандартные драйверы, так как при запросе на доступ к устройству системы начинает искать драйвер от начала списка. При замене стандартного драйвера на новый система первым найдет новый драйвер. Аналогично в список вносятся драйверы новых устройств.

Для доступа к любому драйверу из цепочки, система передает только адрес начала цепочки, то есть драйвера NUL.

Все стандартные драйверы могут быть заменены на альтернативные, за исключением драйвера NUL :, всегда содержится в начале цепочки устройств.

Структура программы драйвера. Программа драйвера состоит из 5 частей.

1. Заголовка устройства (заголовка драйвера).
2. Области памяти для хранения данных и локальных процедур.
3. Процедуры стратегии.
4. Процедуры прерывания.
5. Программ обработки команд.

Начало программы драйвера непохожий на начало обычных программ. Там расположен заголовок драйвера, который содержит информацию о самом драйвер устройства.

Эта информация предназначена для системы и включает имя устройства и указатель на следующий драйвер.

Процедуры стратегии и прерывания объединяются при обработке каждой команды. Они позволяют системе передать управление драйвера. Последняя часть драйвера содержит программы, выполняющие те команды, которые система передает драйверу.

Взаимодействие системы с драйвером.

При вызове драйвера системы передает ему пакет данных. Этот пакет данных называется заголовком запроса и содержит информацию для драйвера устройству: номер команды и данные, записываемые на устройство. При вызове драйвера системы размещает адрес заголовка в регистры ES и BX.



Рис. 2.3 Заголовок запроса (передача пакета данных)

Нельзя путать заголовок запроса и заголовок драйвера. Заголовок драйвера содержит информацию для системы о программе драйвера, а в заголовке запроса системы размещает информацию, необходимую для работы драйвера.

Составные части заголовка запроса.

1. Размер в байтах заголовка запроса (зависит от объема данных в запросе) - 1 байт.
2. Номер устройства - 1б.
3. Код команды - 1б.
4. 16-разрядное слово состояния при завершении - 2б.
5. Зарезервировано.

6.Дани, зависящие от команды. Размер переменный.

Заголовок запроса имеет размер переменной длины. В первом элементе содержится длина заголовка запроса (1б). Второй элемент содержит номер устройства (1б). Он обычно используется в случае во"единения к контроллеру нескольких устройств.

Пример - контроллер дисковода гибких дисков, который часто руководит двумя дисководами. В таком случае диски А: и В: будут иметь номера 0 и 1. В третьем

элементе содержится код команды, которую должен выполнить драйвер (1б). Четвертый элемент служит индикатором состояния (2б). Использование пятого элемента не с"выяснено в документации (8б). Длина шестого элемента зависит от команды в третьем элементе. Эти данные зависят от команды.

Системы и автоматически формирует заголовок запроса при запросе программы на выполнение действий, связанных с драйвером устройства. Этот пакет данных располагается

* зарезервированной для системы области памяти и строится на основе информации от вызывающей программы. При передаче управления драйвера системы передает ему адрес заголовке запроса, которая размещается в локальной памяти драйвера. При этом должна указываться сегментная и относительный адрес заголовка запроса, так как заголовок запроса может располагаться в любом месте памяти. Сегментный и относительную адреса системы размещает соответственно в регистры ES и BX.

***Двухступенчатый вызов драйвера устройства.***

Каждый раз, когда система требует от драйвера выполнения которой команды, она вызывает драйвер дважды. При этом сначала управление передается на процедуру стратегии, затем - на процедуру прерывания. Можно считать процедуру стратегии набором команд для подготовки к iнициализации драйвера. Информация по процедуре стратегии затем используется процедурой прерывания при обработке команды.

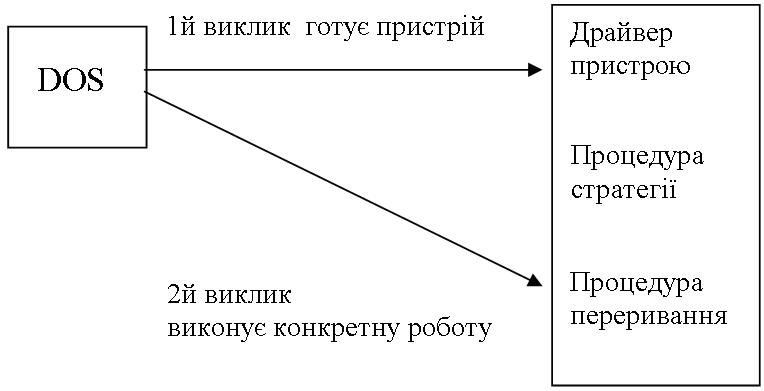


Рис. 2.4 Двухступенчатый вызов драйвера устройства

Такой двухступенчатый механизм позволяет системе разнести во времени первый запрос драйвера (его подготовки) и непосредственную работу драйвера.

Все вызовы процедур стратегий связываются в цепочку стратегий в порядке их поступления, а вызовы процедур прерываний связываются в цепочку прерываний по их приоритетами.



Рис. 2.5 Обработка запросов

После вызовов всех драйверов через процедуры стратегии системы находит в цепочке прерываний драйвер с наиболее высоким приоритетом.

Например, для реализации многозадачности в системе. Задача, которая принимает данные через модем, может быть важнее других.

При первом вызове драйвера процедура стратегии сохраняет адрес заголовка запроса, которая находится в регистрах ES и BX.



Рис. 2.6 Подготовка к первому вызова драйвера

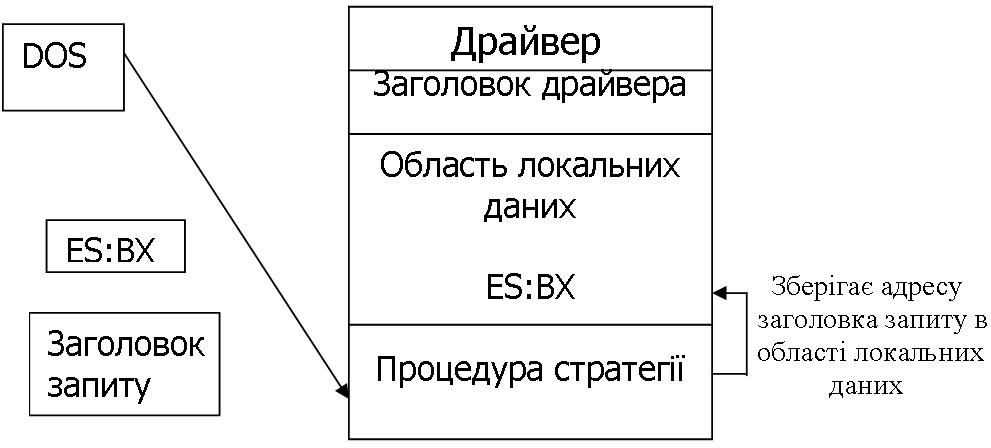


Рис. 2.7 Хранение адреса заголовка запроса в области

локальных данных (процедура стратегии)

Во второй раз система вызывает драйвер устройства через процедуру прерывания. Тогда и начинается работа драйвера. Процедура прерывания обрабатывает заголовок запроса, содержащий информацию для драйвера и передает управление программам обработки команд.

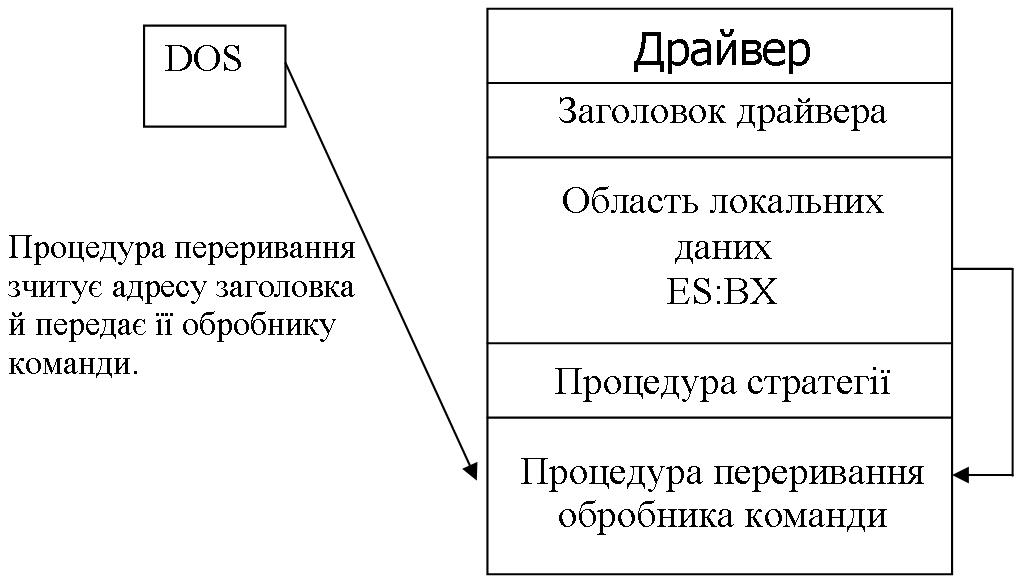


Рис. 2.8 Изъятие заголовке запроса процедурой прерывания

Команды драйвера устройства.

0 Инициализация.

* Проверка носителя (блочные устройства).

2 Получение BPB (блочные устройства).

3 IOCTL - введение.

4 Введение.

5 неразрушающего введения (символьные устройства).

6 Состояние ввода (символьные устройства).

* Очистка ввода (символьные устройства).

8 Вывод.

9 Вывод с проверкой.

10 Состояние вывода (символьные устройства).

11 Очистка вывода (символьные устройства).

12 IOCTL - вывод.

13 Открытие устройства.

14 Закрытие устройства.

15 Сменный носитель (блочные устройства).

16 Вывод, пока не занято (символьные устройства).

17-18 не определены.

19 Обобщенный IOCTL (блочные устройства).

20-22 не определены.

23 Получение логического устройства.

24 Установление логического устройства.

25 IOCTL запрос.

Обзор драйверов устройств.

Костяк, на основе которого разрабатывается драйвер это: 1. Заголовок комментария.

2. Директивы ассемблера.

3. Основная процедура.

4. Заголовок драйвера.

5. Рабочая область драйвера.

6. Процедура стратегии.

7. Процедура прерывания.

8. Локальные процедуры.

9. Обработка команд системы.

10. Выход при ошибке.

11. Общий выход.

12. Конец программы.

***Обзор частей драйвера устройства***

заголовок комментария. Это краткое описание назначения программы, дата создания, имя автора и т.д.

директивы ассемблера. В этом разделе содержатся директивы ассемблера, определяющие конкретные требования к структурам данных драйвера. Здесь приводятся определения сегмента и основной процедуры и устанавливаются адреса в регистрах CS, ES, DS. Эти директивы подходят для большинства создаваемых драйверов, поэтому почти во всех драйверах можно обращаться к фрагментам программы и к данным одинаково.

Важной особенностью при написании этого раздела драйвера является использование структур при описании данных. Структуры не обеспечивают выделение пространства памяти для данных, а лишь определяют средство рассмотрения этих данных.

**Составные части любого драйвера.**

***Заголовок драйвера.***

К заголовок драйвера включаются данные, передаваемые системой сразу после начала работы с устройством. По ним операционная система определяет, как работать с данным устройством.

Пять основных частей заголовка драйвера:

|  |  |  |
| --- | --- | --- |
| 1. | Указатель на следующий драйвер. | next\_dev dd -1 |
| 2. | Атрибуты драйвера. | attribute dw 8000h |
| 3. | Указатель на процедуру стратегии. | strategy dw dev\_strategy |
| 4. | Указатель на процедуру прерывания. | interrupt dw dev\_int |
| 5. | Имя устройства. | dev\_name db 'SIMPLE $' |

Три из пяти разделов заголовка драйвера содержат указатели адресов.

первыйсодержит двосливний указатель на следующий драйвер в файле. Когда система загружает драйвер из файла в память, в этом файле могут содержаться и другие драйверы.

Например, стандартные драйверы для консоли, накопителя на гибких дисках, принтера, коммуникационных портов и часов содержатся в едином файле IBMBIO.COM.

Система использует указатель, расположенный в текущем драйвере для поиска следующего драйвера. Для того, чтобы сообщить системе о том, что

следующего драйвера нет, то нужно поместить - 1 в оба слова этого первого поля.

второйи третий указатели используются системой для определения местоположения процедур стратегии и прерывания. Эти поля содержат сдвиги, по которым можно найти расположение этих процедур в памяти.

***Поле атрибутов устройства***

* + этом поле содержится информация о типе устройства, управляемого драйвером
* в нем определены типы команд, которые должны быть реализованы в драйвере устройства. В ранних версиях системы биты этого поля определяли только тип устройства. В следующих версиях некоторые биты стали использоваться для указания команд, которые обрабатываются драйвером. Рассмотрим назначение каждого бита:

бит 15позволяет системе определить тип устройства, управляемого драйвером: 0 - символьный устройство, 1 - блочный. От значения этого бита зависит интерпретация содержания большинства других битов. Поле имени заголовка устройству тоже определяется типом устройства.

бит 14используется для того, чтобы сообщить системе, поддерживаются драйвером команды управления вводом / выводом. Управление в / в используется для передачи в драйвер и получения из него управляющей информации. Если этот бит установлен, то нужно реализовать процедуры обработки двух IOCTL команд.

Бит 13.Интерпретация системы содержимого бита 13 определяется типом устройства. Для блочно-ориентированных устройств наличие единицы в бите 13 свидетельствует о том, что дисковый накопитель не является стандартным устройством IBM. Ноль в бите 13 означает, что системы и работает со стандартным устройством. Если устройство символьный установления бита означает, что драйвер может обрабатывать команду "Вывод пока не занято". Если бит 13 не установлен, то для определения типа носителя драйвер устройства использует дескриптор носителя с FAT. Если бит 13 установлен, то используется BPB. Но так как диски могут иметь любой дескриптор носителя, то чтобы учесть все возможные варианты, можно установить бит 13 позволив системе использовать BPB для определения расположения всех компонентов диска.

бит 12 не определен и должен содержать 0.

бит 11 используется для того, чтобы показать, поддерживает драйвер команды открытия и закрытия устройства и изменения носителя

Биты с 10 по 8 не определены и должны содержать 0.

бит 7используется драйвером при работе с системой начиная с версии 5.0. Установка бита дает возможность программе пользователя спросить, доступны для использования определенные функции IOCTL

бит 6определяет, использует драйвер команды получения логического устройства и установления логического устройства. Он также показывает, что драйвер поддерживает команды обобщенного IOCTL как для символьных, так и для блочных устройств.

бит 5 не определен и должен содержать 0.

Установка бита 4 определяет, что драйвер поддерживает быстрый ввод-вывод через консоль, реализовано с помощью прерывания 29h.

бит 3установлен, если драйвер обслуживает часы. В этом случае система заменяет стандартный драйвер часов текущим драйвером часов.

бит 2установлен, если драйвер обслуживает стандартное устройство NUL. Нельзя заменить драйвер устройства NUL, поэтому данный бит использовать нельзя. Бит 2 устанавливается только в драйвере устройства NUL, что позволяет системе его идентифицировать.

бит 1установлен, если драйвер заменяет стандартный консольный устройство вывода. Установка этого бита означает, что драйвер блочного устройства реализует 32-разрядную адресацию сектора. То есть, таким образом поддерживаются диски с объемом более 32 Мбайт.

бит 0. Для символьных устройств установки этого бита означает, что данный драйвер заменяет драйвер стандартного устройства вывода.

От установления определенных битов поля атрибутов зависит возможность посылки драйвера устройства различных типов команд. То есть, эти биты используются для определения команд, не характеристика устройства. Итак, слово атрибутов представляет собой мощное средство, которое позволяет каждому драйверу идентифицировать себя.

Табл. 2.1 Значение слова атрибутов для различных устройств

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| имя устройства | слово атрибутов |  |  | установленные биты |
| 1. Устройство NUL: | 8004h | 15 | | символьный устройство |
|  |  | 2 | устройство NUL: | |
| 2. CON: | 8013h | 15 | | символьный устройство |
|  |  | 4 | быстрое ввода / вывода | |
|  |  | 0 | стандартный вывод | |
| 3. AUX: | 8000h | 15 | | символьный устройство |
| 4. LPTx: | A0C0h | 15 | | символьный устройство |
|  |  | 13 | | вывода, пока не занято |
|  |  | 7 IOCTL-запрос | | |
|  |  | 6 | обобщенный IOCTL | |
| 5. COMx: | 8000h | 15 | | символьный устройство |
| 6. CLOCK $ | 8008h | 15 | | символьный устройство |
|  |  | 3 | часы | |
| 7.Disk | 08C2h | блочный устройство | | |
|  |  | 11 | | открытие / закрытие сменный носитель |
|  |  | 7 IOCTL-запрос | | |
|  |  | 6 | получения / установки логического | |
|  |  | устройства | | |
|  |  | 1 | 32-разрядная адресация | |
|  |  | сектора | | |

***Поле имени заголовке драйвера***

Поле имени занимает 8 байтов. Для символьных устройств это поле содержит текст - имя устройства. Если вы заменяете любой стандартный системы, туда необходимо записать имя замененного устройства: CON :, PRN: и т.д. Если стандартный устройство заменяется, в этом поле записывается имя, которое будет использоваться для идентификации устройства. Оно не должно совпадать с одним из используемых имен. Имя устройства должно быть написано большими буквами. Если оно занимает менее 8 байтов, в позиции, оставшиеся необходимо записать пробелы. При использовании блочных устройств это поле предназначается не для сохранения имени устройства. В первом байте поля записывается количество устройств, управляемых драйвером. Так как блочные устройства является дисковыми накопителями, количество устройств, уже установленных системах, определяет букву, которая будет идентифицировать конкретный драйвер.

Если по текущему драйвером идет другой драйвер устройства дискового типа, то буква соответствующей ему определяется суммой числа уже установленных в систему устройств и числа устройств, управляемых текущим драйвером.

В рабочей области драйвера определяются локальные переменные для процедур.

Процедура стратегии для работы в среде

dev\_strategy: mov cs: rh\_seg, es сохранить сегмент mov cs: rh\_ofs, bx; сохранить смещение ret; вернуться в систему.

Процедура прерывания - второй вызов из системы

dev\_interrupt: cld сохранить состояние при выходе

push ds

push es

push ax

push bx

push cx

push dx

push di

push si

; Устанавливаем ES: BX на заголовок

; запроса

mov ax, cs: rh\_seg; восстановить в ES, BX

mov es, ax; значения, сохраненные

mov bx, cs: rh\_ofs, при вызове процедуры казней.

; Переход на соответствующую процедуру выполнения команды. mov al, es: [bx] .rh\_cmp; получения кода команды с

; Заголовка запроса

rol al, 1; команда сдвига слева: умножить x2 для получения

* индекса в таблице CMDTAB.

lea di, cmdtab, адрес таблицы команд. mov ah, 0; очистить старшую половину.

add di, ax; добавить индекс к адресу табл.

jmp word ptr [di], переход на адрес из таблицы.

Cmdtab - это таблица, которая содержит адреса каждой из команд (таблица переходов). Команда, которую нужно выполнить, будет получена из заголовка запроса. Процедура прерывания обеспечит переход по адресу, соответствует полученной драйвером команде на программу, которая обеспечивает выполнение этой команды. Таблица состоит из 16-байт переменных. Если использовать код команды для выбора адреса, индексирование таблицы будет выполнен по байтам. В результате получим только половину 2х-байтовой адреса. Для правильного индексирования надо превратить байтовое значение кода команды в 2х-байтовое.

CMDTAB label byte, только для символьных устройств.

dw initialization; инициализация

dw media\_check, проверка носителя (блочные устройства)

dw get\_BPB; получения BPB

dw IOCTL\_INPUT; IOCTL - Введение

dw INPUT; Введение (чтение)

dw ND\_INPUT; Введение неразрушающего

dw INPUT\_STATUS, состояние ввода

dw INPUT\_FLUSH; очистки введение

dw OUTPUT; вывода (запись)

dw OUTPUT\_VERIFY; вывода с проверкой

dw OUTPUT\_STATUS, состояние вывода

dw OUTPUT\_FLUSH; очистки вывода

dw IOCTL-OUTPUT; IOCTL - вывод

dw OPEN; открытие устройства

dw CLOSE; закрытия устройства

dw REMOVABLE; сменный носитель

dw OUTPUT\_BUSY; вывода пока не занято

dw COMMAND17: не определено

dw COMMAND18: не определено

dw GENERIC\_IOCTL; обобщенный IOCTL

dw COMMAND20: не определено

dw COMMAND21: не определено

dw COMMAND22: не определено

dw GET\_DEVICE; получения логического устройства

dw SET\_DEVICE установление логического устройства dw IOCTL\_QUERY; IOCTL - запрос.

Эта процедура прерывания позволяет создавать драйверы устройств для системы всех версий. Таблица адресов процедур выполнения команд содержит адреса для команд с номерами от 0 до 25.

Выполнение команд.

При обращении к драйверу системы посылает номер команды в заголовок запроса. При этом системва ожидает, что драйвер выполнит все действия, которые требует команда. Ни один драйвер не назначается для всех 26 команд. Количество команд, выполнение которых должно быть реализовано драйвером, определяется четырьмя факторами: 1 - набором допустимых для драйвера операций; 2 - типом управляемого устройства; 3 - установленными битами слова атрибутов 4 - версии системы. В драйверах для устройств, обеспечивающих только вывод информации необходимо реализовать выполнение только команд вывода (вывод, вывод с проверкой, состояние вывода). Для того, чтобы избежать использования каких-либо команд, необходимо сбросить соответствующие биты в слове атрибутов.