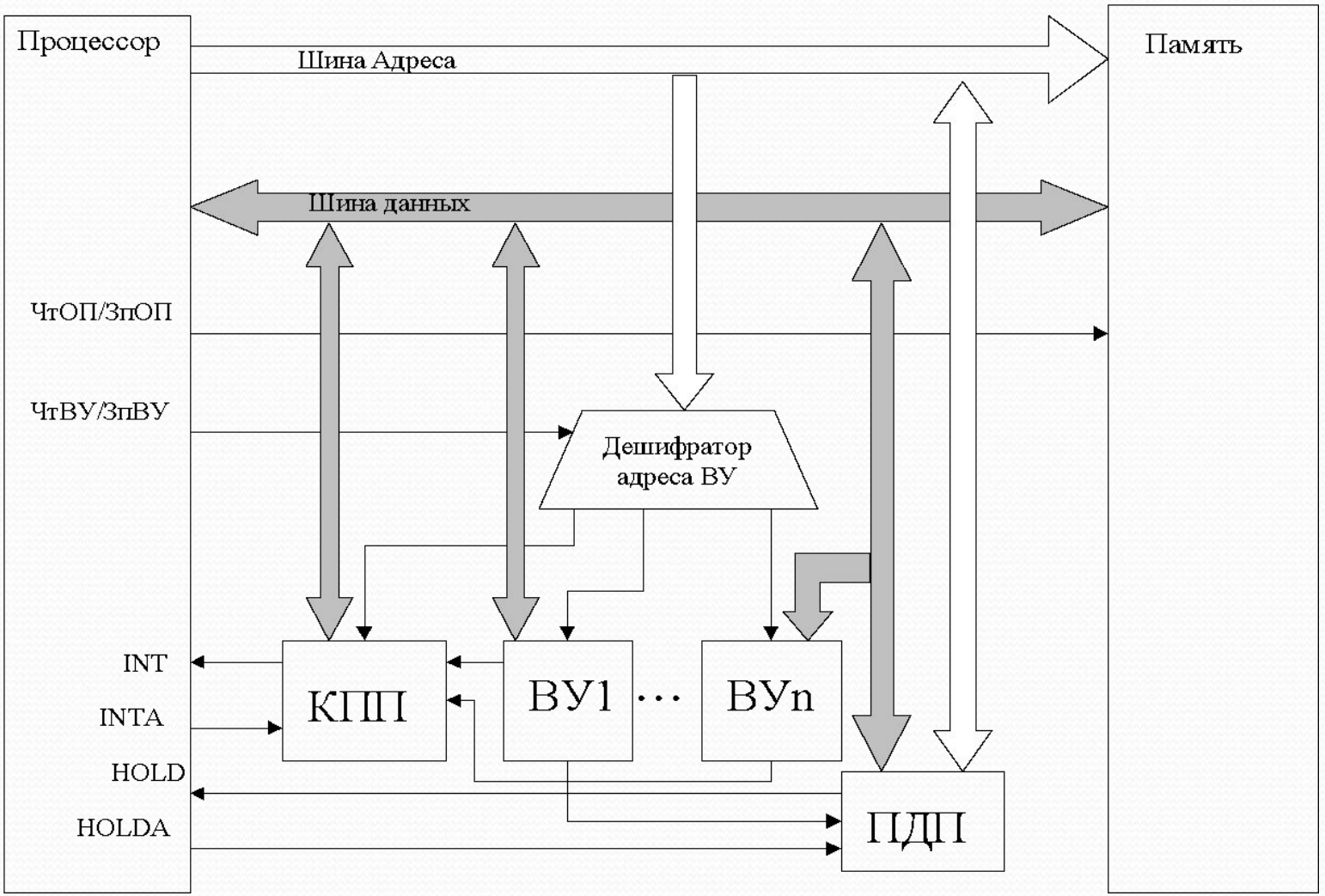
Лекция 12. Система ввода-вывода

Модель аппаратного обеспечения современных ПЭВМ с точки зрения ОС

современных ПЭВМ с точки зрения ОС



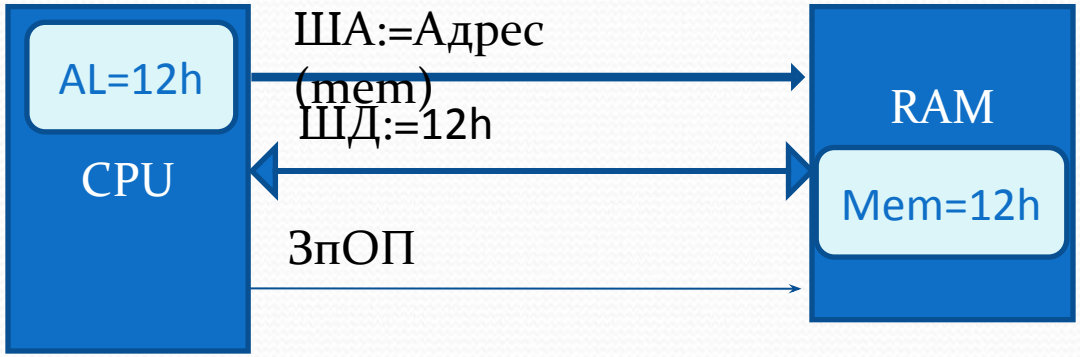
Система ввода-вывода

Система ввода-вывода, т. е. комплекс средств обмена информацией с внешними устройствами, является важнейшей частью архитектуры процессора и машины в целом. К системе ввода-вывода можно отнести и способы подключения к системной шине различного оборудования, и процедуры взаимодействия процессора с этим оборудованием, и команды процессора, предназначенные для обмена данными с внешними устройствами.

Системная шина представляет собой набор интерфейсов, к которым единообразно подключаются все устройства компьютера:

- интерфейс адреса;
- интерфейс данных;
- интерфейс сигналов управления.

mov mem,AL



Процедуры записи и чтения справедливы не только по отношении к памяти. За каждым устройством закреплена определенная группа адресов, на которые оно должно отзываться. Обнаружив свой адрес на магистрали, устройство либо считывает с магистрали поступившие данные, либо устанавливает имеющиеся в нем данные на магистраль.

Система ввода-вывода

Все устройства компьютера можно условно разбить на две категории.

1) Устройства, отображаемые в память. Пример - видеобуфер, входящий в видеосистему компьютера. Устройство управления видеобуфером настроено на две группы адресов, которые как бы продолжают адреса, относящиеся к оперативной памяти (00000-9FFFF – ОП, A0000-B1000H - видеобуфер).
2) Устройства, подключенные к портам ввода/вывода. Адресное пространство портов перекрывается с адресами памяти. Например, за контроллером клавиатуры закреплены два адреса: 60h и 61h. По адресу 60h выполняется чтение кода нажатой клавиши, а адрес 61h используется для управления работой контроллера. И тот, и другой адрес имеются в оперативной памяти и, таким образом, возникает проблема распознавания устройства, к которому происходит обращение.

Аппаратное решение:

идентификация устройств на системной шине осуществляется с помощью сигнала ЧтВУ/ЗпВУ, который генерируется процессором в любой операции записи или чтения. При обращении к памяти или видеобуферу процессор устанавливает значение сигнала ЧтОП/ЗпОП. Таким образом осуществляется аппаратное разделение устройств «типа памяти» и устройств «ввода-вывода».

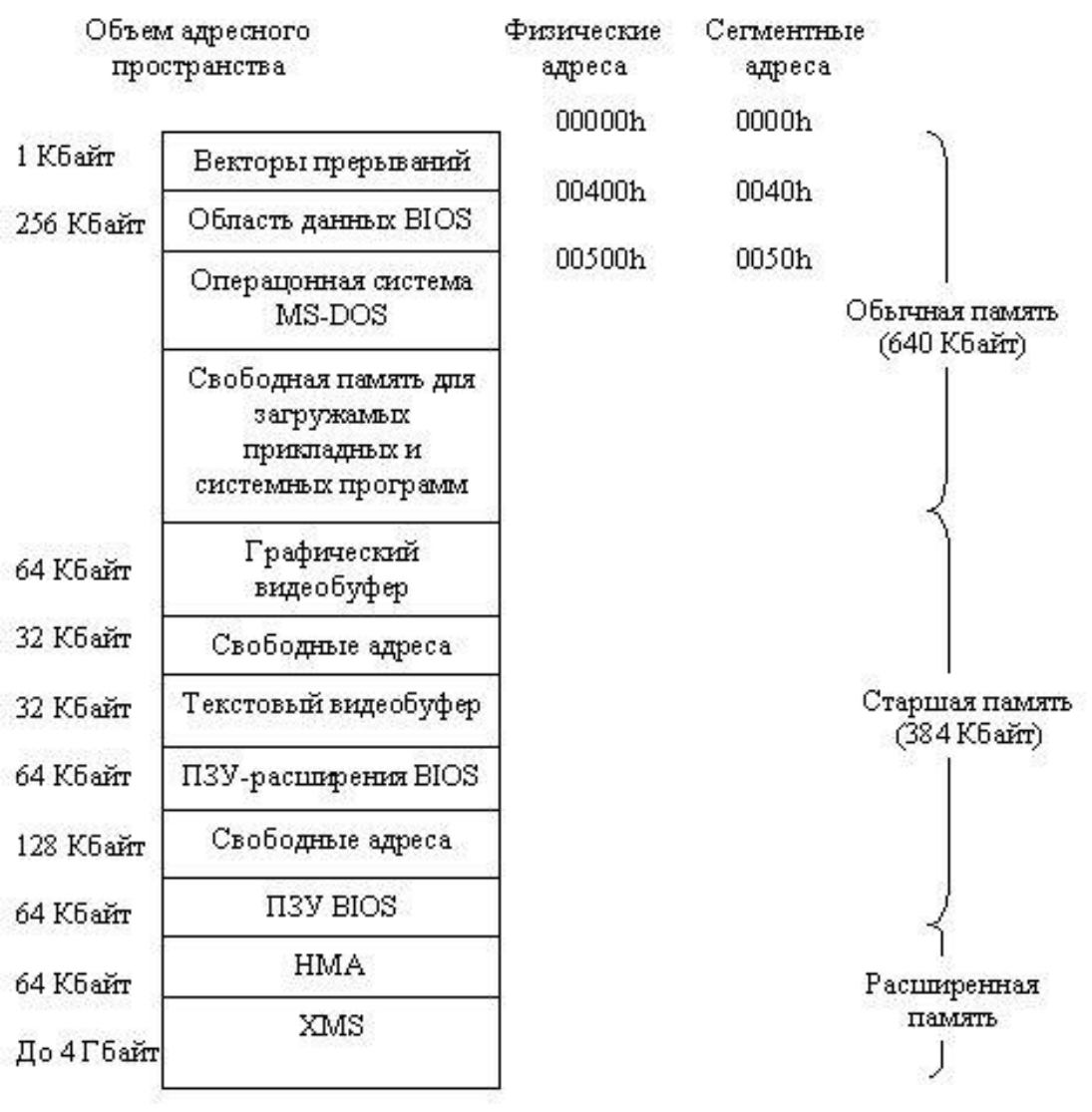
Система ввода-вывода

Программное разделение устройств:
реализуется с помощью двух наборов команд процессора — для памяти и для устройств ввода-вывода. В первую группу команд входят практически все команды процессора, с помощью которых можно обратиться по тому или иному адресу — команды пересылки mov и movs, арифметических действий add, imul и div, сдвигов rol, ror, sal и sar, анализа содержимого байта или слова test и многие другие, фактически в эту группу команды входит большинство команд процессора. Вторую группу команд образуют специфические команды ввода-вывода — команда ввода in и команда вывода out.

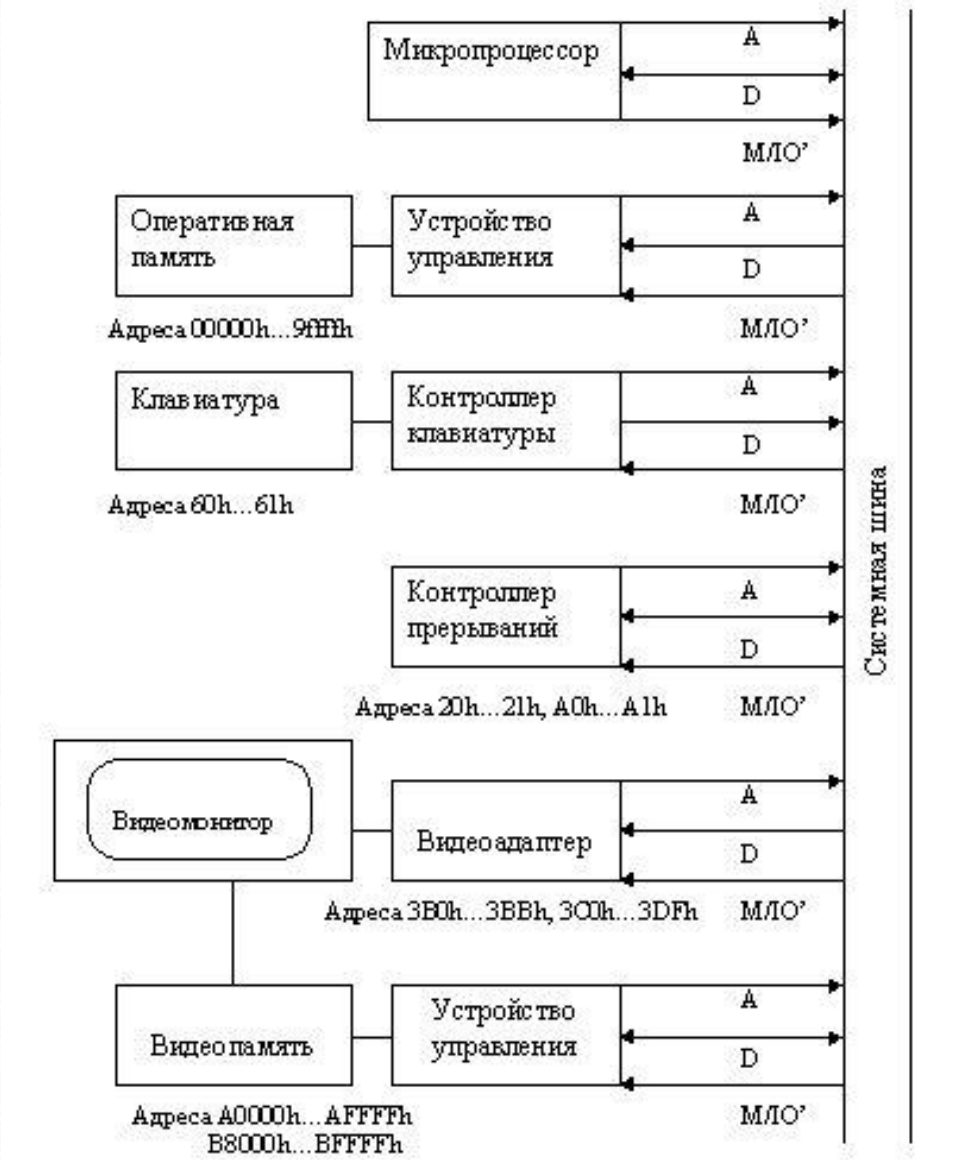
Наличие двух категорий адресов устройств дает основание говорить о существовании двух адресных пространств — пространства памяти, куда входит сама память, а также видеобуферы и ПЗУ, и пространства вводавывода (пространства портов), куда входят адреса остальной аппаратуры компьютера.

Объем адресного пространства памяти составляет 1 Мбайт (в защищенном режиме 4 Гбайт), адресное пространство портов - 64 Кбайт.

Типичное распределение адресного пространства памяти



Подключение устройств компьютера к системной шине



A - адреса;
D - данные;
M/IO - один из сигналов управления

Система ввода/вывода

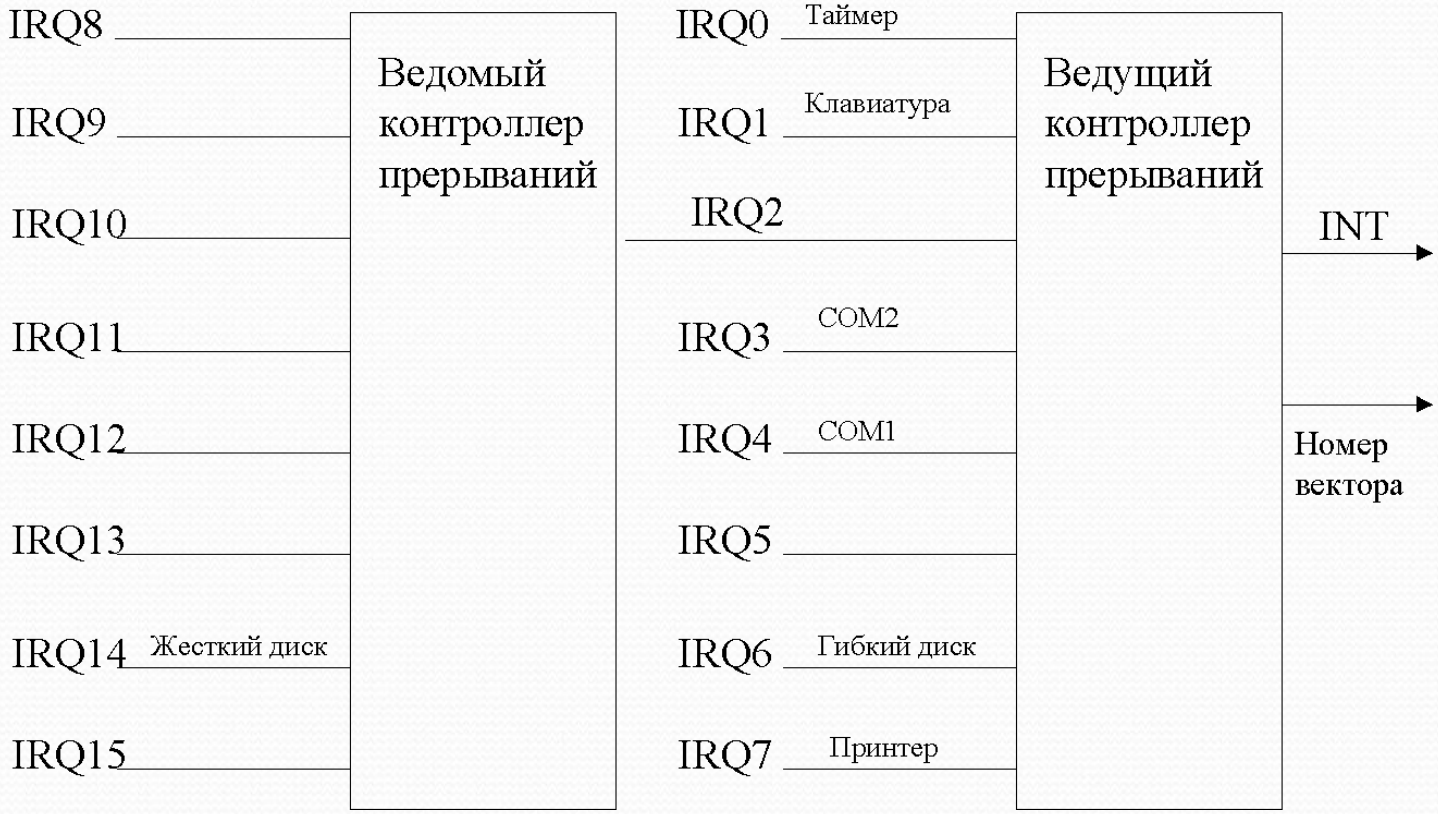
Подсистема прерываний. Реальный режим.

Подсистема прерываний

Система прерываний любого компьютера является его важнейшей частью, позволяющей быстро реагировать на события, обработка которых должна выполнятся немедленно: сигналы от машинных таймеров, нажатия клавиш клавиатуры или мыши, сбои памяти и пр.

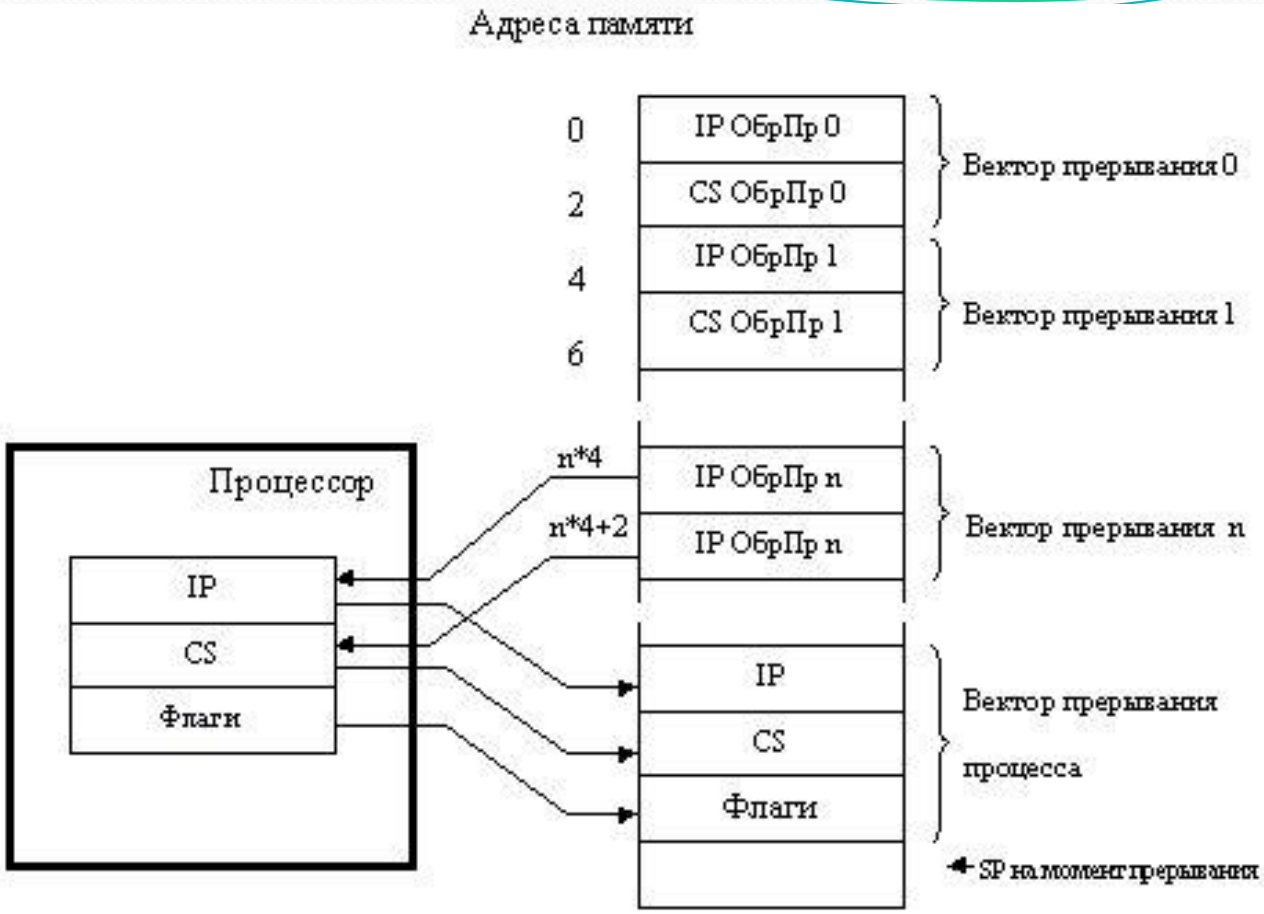
Сигналы аппаратных прерываний, возникающие в устройствах, входящих в состав компьютера или подключенных к нему, поступают в процессор не непосредственно, а через два контроллера прерываний (КПП), один из которых называется ведущим, а второй — ведомым. Два контроллера используются для увеличения допустимого количества внешних устройств. Дело в том, что каждый контроллер прерываний может обслуживать сигналы лишь от 8 устройств. Для обслуживания большего количества устройств контроллеры можно объединять, образуя из них веерообразную структуру.

Подсистема прерываний



Номера базовых векторов заносятся в контроллеры автоматически в процессе начальной загрузки компьютера. Для ведущего контроллера базовый вектор всегда равен 8, для ведомого – 70H. Таким образом, номера векторов, закрепленных за аппаратными прерываниями, лежат в диапазонах 08h...0Fh и 70h...77h. Очевидно, что номера векторов аппаратных прерываний однозначно связаны с номерами линий, или уровнями IRQ, а через них – с конкретными устройствами компьютера.

Подсистема прерываний



Процессор, получив сигнал прерывания, выполняет последовательность стандартных действий, обычно называемых процедурой прерывания (*здесь идет речь лишь о реакции самого процессора на сигналы прерываний, а не об алгоритмах обработки прерываний*)

Подсистема прерываний

Самое начало оперативной памяти от адреса 0000H до 03FFH отводится под векторы прерываний — четырехбайтовые области, в которых хранятся адреса обработчиков прерываний. В два старшие байта каждого вектора записывается сегментный адрес обработчика, в два младшие — смещение. Векторы, как и соответствующие им прерывания, имеют номера, причем вектор с номером 0 располагается, начиная с адреса 0, вектор 1 — с адреса 4, вектор 2 — с адреса 8 и т.д. Вектор с номером n занимает, таким образом, байты памяти от n*4 до n*4+3. Всего в выделенной под векторы области памяти помещается 256 векторов.

Получив сигнал на выполнение процедуры прерывания с определенным номером, процессор сохраняет в стеке выполняемой программы текущее содержимое трех регистров процессора: регистра флагов, CS и IP. Два последних числа образуют полный адрес возврата в прерванную программу. Далее процессор загружает CS и IP из соответствующего вектора прерываний, осуществляя, тем самым, переход на обработчик прерывания, связанный с этим вектором.

Обработчик прерываний всегда заканчивается командой iret (interrupt return, возврат из прерывания), выполняющей обратные действия — извлечение из стека сохраненных там слов и помещение их назад в регистры IP и CS, а также в регистр флагов. Это приводит к возврату в основную программу в ту

Подсистема прерываний

Запросы на обработку прерываний могут иметь различную природу. Имеются еще два типа прерываний: внутренние и программные.

Внутренние прерывания возбуждаются цепями самого процессора при возникновении одной из специально оговоренных ситуаций, например, при выполнении операции деления на ноль или при попытке выполнить несуществующую команду. За каждым из таких прерываний закреплен определенный вектор, номер которого известен процессору. Например, за делением на 0 закреплен вектор 0, а за неправильной командой — вектор 6.

Программные прерывания вызываются командой int с числовым аргументом, который рассматривается процессором, как номер вектора прерывания. Если в программе встречается, например, команда

int 13h

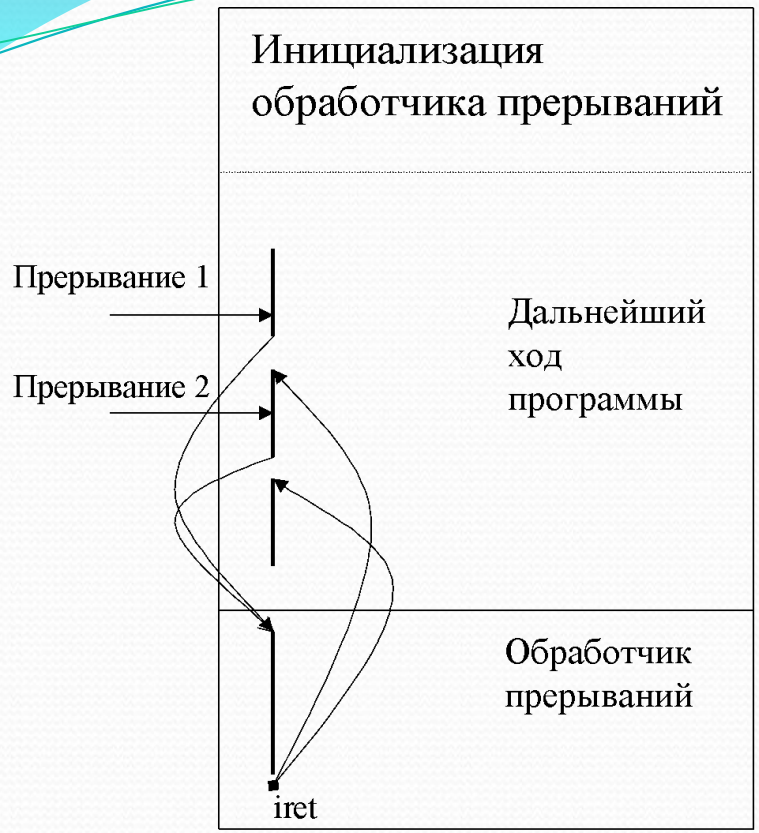
то процессор выполняет ту же процедуру прерывания, используя в качестве номера вектора операнд команды int. *Программные прерывания применяются в первую очередь для вызова системных обслуживающих программ — функций DOS и BIOS.*

Подсистема прерываний

Большая часть векторов прерываний зарезервирована для выполнения определенных действий; часть из них автоматически заполняется адресами системных программ при загрузке системы:

00h -внутреннее прерывание, деление на 0;
0lh -внутреннее прерывание, пошаговое выполнение (при TF=1);
02h -немаскируемое прерывание (вывод NMI процессора);
08h -аппаратное прерывание от системного таймера;
09h -аппаратное прерывание от клавиатуры;
0Eh -аппаратное прерывание от гибкого диска;
10h - программное прерывание, программы BIOS управления видеосистемой;
13h - программное прерывание, программы BIOS управления дисками;
16h - программное прерывание, программы BIOS управления клавиатурой;
21h - программное прерывание, диспетчер функций DOS;
22h - программное прерывание, адрес перехода при завершении процесса, используемый DOS;
23h -программное прерывание, обработчик прерываний по <Ctrl>/C, используемый DOS;
25h - программное прерывание, абсолютное чтение диска (функция DOS);
26h - программное прерывание, абсолютная запись на диск (функция DOS);
60h...66h - зарезервировано для программных прерываний пользователя;
68h...6Fh - программные прерывания, свободные векторы;
70h -аппаратное прерывание от часов реального времени (с питанием от аккумулятора);
76h -аппаратное прерывание от жесткого диска;

прерываний



Обработчик прерываний может входить в состав программы в виде процедуры, или просто являться частью программы, начинающейся с некоторой метки (входной точки обработчика) и завершающейся командой выхода из прерывания iret.

Программа, начиная свою работу, прежде всего должна выполнить инициализирующие действия по установке обработчика прерываний. В простейшем случае эти действия заключаются в занесении в соответствующий вектор полного адреса (сегмента и смещения) обработчика.

В случае прихода прерывания, процессор сохраняет в стеке флаги и текущий адрес программы, извлекает из вектора адрес обработчика и передает управление на его входную точку. Завершающая обработчик команда iret извлекает из стека сохраненные там данные и возвращает управление в прерванную программу, которая может продолжить свою работу.

прерываний

```
;---установка прерывания
PUSH DS ;сохраняем DS
MOV DX,OFFSET ROUT ;смещение для процедуры в DX
MOV AX,SEG ROUT ;сегмент процедуры
MOV DS,AX ;помещаем в DS MOV AH,25H
;функция установки вектора
MOV AL,60H ;номер вектора
INT 21H ;меняем прерывание
POP DS ;восстанавливаем DS

;---процедура прерывания
ROUT PROC FAR
PUSH AX ;сохраняем все изменяемые регистры
. .
POP AX ;восстанавливаем регистры
MOV AL,20H ;эти две строки надо использовать
OUT 20H,AL ;только для аппаратных прерываний
IRET ROUT ENDP
```

прерываний

Стандартной методикой является сохранение в памяти исходного содержимого вектора и восстановление этого содержимого перед завершением программы.

```
;---в сегменте данных:
  KEEP_CS  DW   0      ;хранит сегмент заменяемого прерывания
  KEEP_IP  DW   0      ;хранит смещение прерывания
;---в начале программы
        MOV   AH,25H    ;функция получения вектора
        MOV   AL,1CH    ;номер вектора
        INT   21H       ;теперь сегмент в ES, смещение в BX
        MOV   KEEP_IP,BX ;запоминаем смещение
        MOV   KEEP_CS,ES ;запоминаем сегмент
; ---в конце программы
        CLI
        PUSH  DS        ;DS будет разрушен
        MOV   DX,KEEP_IP ;подготовка к восстановлению
        MOV   AX,KEEP_CS ;
        MOV   DS,AX     ;подготовка к восстановлению
        MOV   AH,25H    ;функция установки вектора
        MOV   AL,1CH    ;номер вектора
        INT   21H       ;восстанавливаем вектор
        POP   DS        ;восстанавливаем DS
        STI
```