Определение конфигурации компьютера программными средствами.

Дополнительный теоретический материал, необходимый для выполнения лабораторной работы №2

Доступ к BIOS.

Прерывания INT 10h – 11Ah – процедуры BIOS.

Пример 1:

```
MOV    AH, 1                ; Функция 1 - установка времени
MOV    CX, HIGH_COUNT  ; CX:DX = новое значение времени MOV
                ; DX,LOW_COUNT
INT 1AH                     ; Точка входа в BIOS, обслуживающая
     ; запросы к таймеру
```

Пример 2:

```
MOV    AH,0     ; Функция 0 - чтение времени
INT 1AH      ; Запрос к BIOS. Результат будет возвращен в регистрах
     ; CX и DX.
```

Области данных BIOS.

Начинается с адреса **0000:0410**

Функции BIOS.

Int 10h - Управление экраном:

00h ***Задание видеорежима***
01h Задание формы курсора
02h Задание позиции курсора
03h Чтение позиции и формы курсора
04h Чтение позиции светового пера
05h Задание активной страницы дисплея
06h Прокрутка окна вверх
07h Прокрутка окна вниз
08h Чтение атрибута и символа в текущей позиции курсора
09h Вывод атрибута и символа в текущую позицию курсора
0Ah Вывод символа в текущую позицию курсора
0Bh Задание цветовой палитры
0Ch Вывод пикселя на экран
0Dh Чтение пикселя
0Eh Вывод символов в режиме телетайпа
0Fh Чтение текущего состояния дисплея
10h Установка регистров палитр (PCjr, EGA, VGA, MCGA)
11h Управление знакогенератором (EGA, VGA, MCGA)
12h Выбор альтернативы (EGA, VGA, MCGA)
13h Вывод строки на экран (многие ПЭВМ)
14h Управление плазменным дисплеем (Convertible)
15h Чтение параметров активного дисплея (Convertible)
1Ah Чтение/вывод кода комбинации дисплеев (PS/2)
1Bh Чтение функциональной информации (PS/2)
1Ch Сохранение/восстановление состояния дисплея (VGA)

Выбор режима работы - функция 00h INT 10h

На входе: **AH** **00h**

AL Номер устанавливаемого режима работы видеоадаптера, если бит D7 = 1, то при установке режима видеопамять не очищается

На выходе: Не используются

Пример:

```
mov ah, 0
mov al, 3 ; Выбираем режим номер 3 (16 цветной, текстовый,
int 10h   ; разрешение 25x80 символов)
```

Информацию о видеоадаптерах и его режимах можно найти, например:

1. «Библиотека системного программиста» - Александр Фролов, Григорий Фролов Том 21, Программирование видеоадаптеров, М.: Диалог-МИФИ, 1995, 271 стр.
2. http://www.codenet.ru/cat/Applications/Graphics/VGA-VESA-Standarts/
3. http://www.codenet.ru/progr/video/vbe-svga.php

и т.д.

Определение текущего режима работы видеоадаптера - функция 0Fh

На входе: **AH** **0Fh**

На выходе: **AH** Количество символов в строке

AL Номер текущего режима

BH Номер активной страницы видеопамяти

Список стандартных режимов работы видеоадаптеров

Режим работы	Тип информации	Количество цветов	Разрешение, пиксел х пиксел	Размер символов, пиксел х пиксел
0, 1	Текстовый цветной	16	40x25	8x8
0*, 1*	Текстовый цветной	16	40x25	8x14
0+, 1+	Текстовый цветной	16	40x25	9x16
2, 3	Текстовый цветной	16	80x25	8x8
2*, 3*	Текстовый цветной	16	80x25	8x14
2+, 3+	Текстовый цветной	16	80x25	9x16
4, 5	Графический цветной	4	320x200	
6	Графический цветной	2	640x200	
7	Текстовый монохромный	2	80x25	9x14
7+	Текстовый монохромный	2	80x25	9x16
8, 9, 0Ah	Используются видеоадаптерами компьютера PC jr, и в настоящее время интереса не представляют			
….	….			

Список режимов работы видеоадаптеров, соответствующих стандарту VESA

Режим работы	Тип информации	Количество цветов	Разрешение, пиксел х пиксел	Размер символов, пиксел х пиксел
100h	Графический цветной	256	640x400	
101h	Графический цветной	256	640x480	8x16
102h	Графический цветной	16	800x600	8x8
103h	Графический цветной	256	800x600	8x8
104h	Графический цветной	16	1024x768	
105h	Графический цветной	256	1024x768	8x16
106h	Графический цветной	16	1280x1024	
107h	Графический цветной	256	1280x1024	
108h	Текстовый цветной	16	80x60	
109h	Текстовый цветной	16	132x25	9x16
10Ah	Текстовый цветной	16	132x43	9x9
10Bh	Текстовый цветной	16	132x50	
10Ch	Текстовый цветной	16	132x60	
10Dh	Графический цветной	32768	320x200	
10Eh	Графический цветной	65536	320x200	
10Fh	Графический цветной	16777216	320x200	
110h	Графический цветной	32768	640x480	

Int 11h - Конфигурация оборудования

Вход: нет

Выход: **AX** = конфигурация оборудования

Описание:

Возвращает в **AX** конфигурацию оборудования ПЭВМ.

Это слово хранится в области данных **BIOS** по адресу **00410h.**

Биты	Значение
0	**1 - система содержит НМД;** **0 - система не содержит НМД.** *Для варианта №1*
1	1 - установлен арифметический сопроцессор; 0 - арифметический сопроцессор не установлен.
2-3	**Объем основной памяти, установленной на материнской плате:** **Биты: 3 2** **0 1 - 16К;** **1 0 - 32К;** **1 1 - 64К и более.** *Для варианта №2*
4-5	**Тип дисплейного контроллера и его режим:** **Биты: 5 4** **0 0 - не используется или EGA;** **0 1 - CGA, EGA, VGA в режиме 40x25;** **1 0 - CGA, EGA, VGA в режиме 80x25;** **1 1 - монохромный контроллер.** *Для варианта №12*

6-7	**Количество установленных НГМД:** **Биты: 7 6** **0 0 - установлен 1 НГМД;** **0 1 - установлено 2 НГМД;** **1 0 - установлено 3 НГМД;** **1 1 - установлено 4 НГМД.** *Для варианта №8*
8	1 - используется контроллер прямого доступа к памяти; 0 - контроллер прямого доступа к памяти не используется.
9-11	**Количество установленных портов последовательной передачи данных RS232S:** **000 - нет портов;** **001 - используется один порт;** **111 - используется 7 портов.** *Для варианта №4*
12	1 - используется игровой адаптер (джойстик); 0 - игровой адаптер не используется.
13	1 - установлен последовательный принтер (только для PC Jr).
14-15	**Количество установленных принтеров:** **00 - нет принтеров;** **01 - используется 1 принтер;** **10 - используется 2 принтера;** **11 - используется 3 принтера.** *Для варианта №4*

```
// ====================================================
// Получение информации о конфигурации компьютера при помощи BIOS
// ====================================================

#include <stdio.h>
#include <conio.h>
#include <memory.h>
#include <dos.h>

// Битовые поля слова конфигурации
typedef struct _HDWCFG
{
        unsigned  HddPresent:  1;             // 0
        unsigned  NpuPresent:  1;             // 1
        unsigned  AmountOfRAM:  2;                    // 2-3
        unsigned  VideoMode:  2;              // 4-5
        unsigned  NumberOfFdd: 2;             // 6-7
        unsigned  DmaPresent:  1;             // 8
        unsigned  NumberOfCom:  3;                    // 9-11
        unsigned  GamePresent:  1;            // 12
        unsigned  JrComPresent: 1;            // 13
        unsigned  NumberOfLpt:  2;            // 14-15
} HDWCFG;

int main(void)
{
        union REGS  rg;
        HDWCFG HdwCfg;
        unsigned uword;

// Вызываем прерывание INT 11h для получения слова конфигурации компьютера
        rg.h.ah = 0x0;
        int86(0x11, &rg, &rg);
// Получаем слово конфигурации и сохраняем его в структуре HdwCfg
        uword = (unsigned int)rg.x.ax;
        memcpy(&HdwCfg, &uword, 2);
```

```
// Выводим на экран конфигурацию компьютера printf("\n\nConfiguration word: %04.4X", HdwCfg);
        if(HdwCfg.HddPresent)
          printf("\nHDD present");
        if(HdwCfg.NpuPresent)
          printf("\nNPU present");
        printf("\nRAM banks: %d", HdwCfg.AmountOfRAM);
        printf("\nVideo Mode: %d", HdwCfg.VideoMode);
        printf("\nNubber of FDD: %d", HdwCfg.NumberOfFdd + 1);
        if(HdwCfg.DmaPresent)
          printf("\nDMA present");
        printf("\nNubber of COM ports: %d", HdwCfg.NumberOfCom);
        if(HdwCfg.GamePresent)
          printf("\nGame adapter present");
        if(HdwCfg.JrComPresent)
          printf("\nPCjr Com present");
        printf("\nNumber of LPT ports: %d", HdwCfg.NumberOfLpt);

// Вызываем прерывание INT 12h для определения объема основной оперативной памяти компьютера
        rg.h.ah = 0x0;
        int86(0x12, &rg, &rg);

// Выводим объем оперативной памяти
        printf("\nRAM istalled: %d Kbytes",
                (unsigned int)rg.x.ax);

// Получаем объем расширенной оперативной памяти,
// доступной через прерывание INT 15h
        rg.h.ah = 0x88;
        int86(0x15, &rg, &rg);

// Выводим объем расширенной оперативной памяти
        printf("\nExtended RAM istalled: %ld Kbytes",
                 (unsigned int)rg.x.ax);
        getch();
        return 0;
}
```

Int 12h - Размер ОЗУ

Вход: нет

Выход: **AX=**размер ОЗУ в килобайтах

Описание:

Возвращает в **AX** размер оперативной памяти в Кб (не более 640), определенный в процессе **POST** анализом DIP-переключателей на PC/XT или содержимого CMOS на AT и PS/2.

Это слово хранится в области данных **BIOS** по адресу **00413h**.

Для определения размера расширенной памяти – **Int 15h, функцию 88h – Получить размер расширенной памяти (AT, XT-286, PS/2)**.

Выход: **AX =** число непрерывных 1K блоков сверх 1M (1024K)

Для определения размера дополнительной памяти – **Int 67h, функцию 42h**.

Дополнительная память (EMS) начинается с адреса **0D000h.**

Вызывать функции EMS можно, только есть драйвер EMMXXXX0.

Для проверки ее существования можно, например, вызвать функцию 3Dh.

INT 67h, AH = 42h — Получить объем памяти.

Ввод: **AH** = 42h

Вывод: **AH** = 0
DX = объем EMS-памяти в 16-килобайтных страницах
BX = объем свободной EMS-памяти в 16-килобайтных страницах

Int 13h - Управление дисками

Функции прерывания INT 13h:

00h	***Сброс дисковой подсистемы***
01h	***Получить состояние дисковой подсистемы***
02h	Чтение сектора
03h	Запись сектора
04h	Проверка сектора
05h	Форматирование дорожки
06h	Форматирование дорожки (НМД)
07h	Форматирование диска (НМД)
08h	***Получить текущие параметры дисковода (НМД)***
09h	Инициализация таблиц параметров жесткого диска
0Ah	Чтение длинное (НМД)
0Bh	Запись длинная (НМД)
0Ch	Поиск цилиндра (НМД)
0Dh	Альтернативный сброс дисковода (НМД)
0Eh	Чтение буфера сектора (НМД)

0Fh	Запись буфера сектора (НМД)
10h	Проверка готовности дисковода (НМД)
11h	Рекалибровка дисковода (НМД)
12h	Проверка памяти контроллера (НМД)
13h	Проверка дисковода (НМД)
14h	Проверка контроллера (НМД)
15h	***Получить тип дисковода***
16h	Проверка замены диска
17h	Установка типа дискеты
18h	Установка среды носителя данных для форматирования
19h	Парковка головок (НМД)
1Ah	Форматирование диска (ESDI НМД)

Некоторые функции прерывания INT 13h

00h – сброс дисковой подсистемы.

На входе:	**AH = 00h**
	DL = Адрес дисковода (0, 1, ...,80h, 81h, ...)
На выходе:	-
Примечание:	PC, XT, AT, PS/2

01h - получить состояние дисковой подсистемы.

На входе:	**AH = 01h**
	DL = Адрес дисковода (0, 1, ...,80h, 81h, ...)
На выходе:	**AL** = Состояние дисковода после завершения последней операции
Примечание:	PC, XT, AT, PS/2

Код ошибки прерывания INT13h функции 01h.

00h	Успешное завершение операции
01h	Неправильная команда
02h	Не найдена адресная метка
03h	Попытка записи на диск, защищенный от записи
04h	Сектор не найден
05h	Ошибка при сбросе (НМД)
06h	Произошла замена дискеты
07h	Неправильные параметры дисковода (НМД)
08h	Переполнение канала ПДП (НГМД)
09h	Переход за границу 64К при работе с ПДП
0Ah	Обнаружен плохой сектор (НМД)
0Bh	Обнаружена плохая дорожка (НМД)
0Ch	Неправильный номер дорожки

0Dh	Неправильный номер сектора при форматировании (НМД)
0Eh	Обнаружена адресная метка управляющих данных (НМД)
0Fh	Ошибка ПДП (НМД)
10h	Обнаружена ошибка в CRC/ECC
11h	Данные скорректированы с использованием ECC (НМД)
20h	Сбой контроллера
40h	Сбой при поиске дорожки
80h	Таймаут - программа не успевает обрабатывать данные
AAh	Дисковод не готов (НМД)
BBh	Неизвестная ошибка (НМД)
CCh	Сбой при записи (НМД)
E0h	Ошибка регистра состояния (НМД)
FFh	Ошибка операции считывания (НМД)

08h - получить текущие параметры дисковода (НМД).

На входе:	**AH = 08h**
	DL = Адрес дисковода (0, 1, ..., 80h, 81h, ...)
На выходе:	**AH = Состояние дисковода после завершения последней операции**
	CF = 1, если произошла ошибка, 0, если ошибки нет
	BL = тип дисковода (только для AT и PS2)
	DL = количество НМД, обслуживаемых первым контроллером
	DH = максимальный номер головки
	CL = максимальный номер сектора
	CH = максимальный номер цилиндра
	ES:DI = адрес таблицы параметров дисковода
Примечание:	**PC, XT, AT, PS/2**

Тип дисковода в регистре BL:

0	не используется;
1	360К, 40 дорожек, 5,25 дюймов;
2	1,2М, 80 дорожек, 5,25 дюймов;
3	720 К, 80 дорожек, 3,5 дюйма;
4	1,44М, 80 дорожек, 3,5 дюйма.

15h – Получить тип дисковода

На входе:	**AH = 15h**
	DL = Адрес дисковода (0, 1, ..., 80h, 81h, ...)
На выходе:	**AH** = Тип дисковода
	CX:DX = количество секторов размером 512 байтов
Примечание:	AT, PS/2

Тип дисковода:

0	диск отсутствует;
1	НГМД без аппаратных средств обнаружения замены дискеты;
2	НГМД оснащенный средствами обнаружения замены дискеты;
3	НМД.

Организация доступа к CMOS-памяти:

```
mov al,12h
out  70h,al  ; задаем адрес в CMOS-памяти
jmp  $+2     ; небольшая задержка
in al,71h    ; записываем в AL считанное значение
```

Ячейки CMOS-памяти, отвечающие за конфигурацию дисковой подсистемы:

- *14h - байт конфигурации*
- *10h - тип НГМД*
- *12h - тип НМД C: и D*

В памяти CMOS хранится:

Адрес ячейки	Содержимое	
00h - 0Dh	**Используются часами реального времени**	*Вариант №6.*
0Eh	**Байт состояния диагностики при включении питания**	*Вариант №7.*
0Fh	Байт состояния отключения	
10h	**Тип НГМД**	*Вариант №8.*
11h	Зарезервировано	
12h	**Тип НМД (если меньше 15)**	*Вариант №1.*
13h	Зарезервировано	
14h	**Конфигурация оборудования**	*Вариант №11.*
15h - 16h	**Объем основной памяти**	*Вариант №2.*
17h - 18h	**Объем расширенной памяти**	*Вариант №2.*
19h	**Тип первого НМД (если он больше 15)**	*Вариант №1*
1Ah	**Тип второго НМД (если он больше 15)**	*Вариант №1*
1Bh - 20h	Зарезервировано	
21h - 2Dh	Зарезервировано	
2Eh - 2Fh	Контрольная сумма ячеек 10h - 20h	
30h - 31h	**Объем расширенной памяти**	*Вариант №2.*
32h	Текущее столетие в двоично-десятичном коде (19h для 19-го столетия)	
33h	Различная информация	
34h - 3Fh	Зарезервировано	

00h - 0Dh - область часов реального времени

0Eh - байт диагностики

Бит	Описание
0-1	Не используется, равно 0
2	0 - неправильная установка часов реального времени; 1 - часы реального времени установлены правильно
3	0 - НМД исправен; 1 - неисправность НМД, невозможно загрузить операционную систему с жесткого диска
4	0 - размер оперативной памяти указан правильно; 1 - фактический размер оперативной памяти не соответствует указанному в памяти CMOS
5	0 - конфигурация указана правильно; 1 – ошибка в конфигурации системы, фактическая конфигурация не соответствует указанной в байте конфигурации оборудования (ячейка 14h)
6	0 - контрольная сумма памяти CMOS правильная; 1 - ошибка в контрольной сумме памяти CMOS
7	0 – аккумулятор, питающий память CMOS, исправен и заряжен; 1 - разрядка аккумулятора выше нормы

10h – ячейка CMOS-памяти:

Значение	Емкость, Кбайт	Диаметр	Количество секторов на одну дорожку	Количество дорожек
0000	НГМД не установлен	-	-	-
0001	360	5,25"	9	40
0010	1200	5,25"	15	80
0011	720	3,5"	9	40
0100	1440	3,5"	18	80

12h - Тип НМД (если тип меньше 15)
19h - Тип первого НМД (если тип > 15)
1Ah - Тип второго НМД (если тип > 15)

Сокращенная таблица параметров для стандартных типов НМД:

Тип	Количество цилиндров	Количество головок	Емкость диска в байтах
1	306	4	10.653.696
2	615	4	21.411.840
3	615	6	32.117.760
4	940	8	65.454.080
5	940	6	49.090.560
6	615	4	21.411.840
7	462	8	32.169.984
8	733	5	31.900.160
9	900	15	117.504.000
10	820	3	21.411.840
11	855	5	37.209.600
12	855	7	52.093.440
13	306	8	21.307.392
14	733	7	44.660.224
15	0	0	0
16	612	4	21.307.392
17	977	5	42.519.040
18	977	7	59.526.656
19	1024	7	62.390.272
20	733	5	31.900.160
21	733	7	44.660.224
22	733	5	31.900.160
23	306	4	10.653.696
24	977	5	42.519.040
25	1024	9	80.216.064
26	1224	7	74.575.872
27	1224	11	117.190.656
28	1224	15	159.805.440
29	1024	8	71.303.168
30	1024	11	98.041.856
31	918	11	87.892.992
32	925	9	72.460.800
33	1024	10	89.128.960
34	1024	12	106.954.752
35	1024	13	115.867.648
36	1024	14	124.780.544
37	1024	2	17.825.792
38	1024	16	142.606.336
39	918	15	119.854.080
40	820	6	42.823.680

Таблица параметров дискеты DPT (Diskette Parameter Table) – INT 1Eh

Смещение, байт	Размер, байт	Имя поля	Описание
0	1	srt_hut	Биты 0...3:SRT (Step Rate Time) - задержка для переключения головок, лежит в пределах 1 - 16 мс и задается с интервалом 1 мс (0Fh - 1 мс, 0Eh - 2 мс, 0Dh - 3 мс, ...).Биты 4...7:Задержка разгрузки головки , лежит в пределах 16 - 240 мс и задается с интервалом 16 мс (1 - 16 мс, 2 - 32 мс, ..., 0Fh - 240 mc)
1	1	dma_hlt	Бит 0: Значение этого бита, равное 1, говорит о том, что используется прямой доступ к памяти; Биты 2...7: Время загрузки головок HLT - интервал между сигналом загрузки головок и началом операции чтения или записи, лежит в пределах 2 - 254 мс и задается с интервалом 2 мс (1 - 2 мс, 2 - 4 мс, ..., 0FFh - 254 мс)
2	1	motor_w	Задержка перед выключением двигателя
3	1	sec_size	Код размера сектора в байтах:0 - 128;1 - 256;2 - 512;3 - 1024
4	1	eot	Номер последнего сектора на дорожке
5	1	gap_rw	Длина межсекторного промежутка для чтения или записи
6	1	dtl	Максимальная длина передаваемых данных. Используется, когда не задана длина сектора
7	1	gap_f	Длина межсекторного промежутка для операции форматирования
8	1	fill_char	Байт-заполнитель для форматирования, обычно используется F6h
9	1	hst	Время установки головки в мс
10	1	mot_start	Время запуска двигателя в 1/8 долях секунды

Таблица параметров жесткого диска HDPT (Hard Disk Parameter Table) – INT 41h и INT 46h

Смещение, байт	Размер, байт	Имя поля	Описание
0	2	max_cyl	Максимальное количество дорожек на диске
2	1	max_head	Максимальное количество магнитных головок
3	2	srwcc	Начальная дорожка для предварительной записи (Starting reduced-write current cylinder)
5	2	swpc	Начальная дорожка для предварительной компенсации при записи (Starting write precompensation cylinder)
7	1	max_ecc	Максимальная длина блока коррекции ошибок ECC (Maximum ECC data burst length)
8	1	dstopt	Параметры устройства: бит 7 - запрет восстановления; бит 6 - запрет восстановления по блоку коррекции ошибок ECC (Error Correction Code); биты 2-0 - дополнительные параметры устройства
9	1	st_del	Стандартная величина задержки
10	1	fm_del	Величина задержки для форматирования диска
11	1	chk_del	Величина задержки для проверки диска
12	4	reserve	Зарезервировано

14h - конфигурация оборудования

Бит	**Описание**
0	1 - в системе установлены НГМД; 0 - НГМД не используются
1	1 - установлен арифметический сопроцессор; 0 - арифметический сопроцессор не установлен
2-3	не используются, равны 0
4-5	Тип видеоадаптера и видеорежим: 00 - не используется или EGA; 01 - CGA, EGA, VGA в режиме 40x25; 10 - CGA, EGA, VGA в режиме 80x25; 11 – монохромный видеоадаптер
6-7	Количество установленных НГМД, уменьшенное на единицу; 00 – один НГМД; 01 – два НГМД; 10 – три НГМД; 11 – четыре НГМД

Ячейки CMOS-памяти, отвечающие за оперативную память

Номера ячеек	Назначение	Описание
15h - 16h	Объем основной памяти	Ячейка 15h содержит младший байт, а ячейка 16h - старший байт объема основной памяти. *Например*: 0100h - 256K 0200h - 512K 0280h - 640K
17h - 18h	Объем расширенной (extended) памяти	Ячейки 17h и 18h содержат, соответственно, младший и старший байты размера дополнительной памяти (расположенной выше границы 1 М) в килобайтах.
30h - 31h	Объем расширенной (extended) памяти	Ячейки 30h и 31h содержат, соответственно, младший и старший байты размера дополнительной памяти (расположенной выше границы 1 М) в килобайтах. Эта информация дублирует аналогичную информацию, расположенную в ячейках с адресами 17h-18h.

```
// ====================================================
// Чтение и отображение ячеек памяти CMOS
// ====================================================
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
        unsigned char cmos[64];
        int i;
        unsigned long nExtRam;
// Читаем 64 ячейки CMOS-памяти в массив cmos
        for(i=0; i<64; i++)
        {
          outp(0x70,i);
          cmos[i]=inp(0x71);
        }
// Отображаем ячейки часов реального времени
        printf("\nRTC:              ");
        for(i=0; i<0xd; i++)
        {
          printf("%02.2x ",(unsigned)cmos[i]);
        }
// Отображаем состояние байта диагностики после включения питания
        printf("\nDiagnostics byte: %02.2x",cmos[0xe]);
// Отображаем содержимое байта отключения
        printf("\nShutdown byte:    %02.2x\n",cmos[0xf]);
// Отображаем содержимое зарезервированных ячеек
        printf("Reserved:         ");
        for(i=0x34; i<0x40; i++)
        {
          printf("%02.2x ",(unsigned)cmos[i]);
        }
// Вычисляем объем расширенной памяти и отображаем его на консоли
        nExtRam = ((unsigned long)cmos[0x18] << 8) + cmos[0x17];
        printf("\nExtended RAM:     %ld Kbyte\n", nExtRam);
        getch();
        return 0;
}
```

Int 14h - Управление стыком RS-232C

Функции прерывания:

Функция	Назначение	Примечания
00h	Инициализация порта	Поддерживаются BIOS всех ПЭВМ
01h	Вывод одного символа в порт	
02h	Ввод одного символа из порта	
03h	Опрос состояния порта	
04h	Расширенная инициализация порта (PS/2)	Поддерживаются только ПЭВМ семейства PS/2
05h	Расширенное управление портом (PS/2)	

INT 14h AH = 03 — Получить текущее состояние порта

Ввод:	**AH** = 03h **DX** = номер порта (00h – 03h)
Вывод:	**AH** = состояние линии **AL** = состояние модема

INT 14h AH = 00 — Инициализация последовательного порта

Ввод:	**AH = 00h** **AL** = параметры инициализации: биты **7 – 5**: **000** — 19 200 бод (110 бод без FOSSIL) **001** — 38 400 бод (150 бод без FOSSIL) **010** — 300 бод **011** — 600 бод **100** — 1200 бод **101** — 2400 бод **110** — 4800 бод **111** — 9600 бод **биты 4 – 3**: четность (01 — четная, 11 — нечетная, 00 или 10 — нет) **бит 2**: число стоп-бит (0 — один, 1 — два) **биты 1 – 0**: длина слова (00 — 5, 01 — 6, 10 — 7, 11 — 8) **DX** = номер порта (00h – 03h)
Вывод:	**AH = состояние порта** **бит 7**: тайм-аут **бит 6:** буфер вывода пуст (без FOSSIL: регистр сдвига передатчика пуст) **бит 5**: в буфере вывода есть место (без FOSSIL: регистр хранения передатчика пуст) **бит 4**: обнаружено состояние BREAK **бит 3**: ошибка синхронизации **бит 2**: ошибка четности **бит 1**: ошибка переполнения — данные потеряны **бит 0**: в буфере ввода есть данные **AL = состояние модема** **бит 7**: обнаружена несущая (состояние линии DCD) **бит 6**: обнаружен звонок (состояние линии RI) **бит 5**: запрос для передачи (состояние линии DSR) **бит 4**: сброс для передачи (состояние линии CTS) **бит 3**: линия DCD изменила состояние **бит 2**: линия RI изменила состояние **бит 1**: линия DSR изменила состояние **бит 0**: линия CTS изменила состояние

BIOS Data Area

Адрес	Размер, байт	Назначение
040:000	4x2	Базовые адреса портов COM 1— COM4 ***Вариант №4***
040:008	3x2	Базовые адреса портов LPT1— LPT3 ***Вариант №4***
040:010 **040:017**	2 39	Установленное оборудование ***Вариант №11*** Область флагов и буфер клавиатуры
040:049	1	Текущий видеорежим ***Вариант №12***
040:04A	2	Ширина экрана (число колонок символов)
040:050	16	Позиция курсора (младшая половина — колонка, старшая — ряд)
040:060	2	Размер курсора (в младшем байте — последняя строка, в старшем — первая)
040:067	5	Область данных POST

Int 15h - Системные утилиты

Функция	Назначение
00h	Включить мотор кассетного магнитофона (PC, PCjr)
01h	Выключить мотор кассетного магнитофона (PC, PCjr)
02h	Чтение с кассетного магнитофона (PC, PCjr)
03h	Запись на кассетный магнитофон (PC, PCjr)
0Fh	Ловушка форматирования дисков (PS/2)
21h	**Регистрация ошибок POST (PS/2)** ***Вариант №7.***
40h	Операции с профилем системы (Convertible)
41h	Ожидание внешнего события (Convertible)
42h	Отключение ПЭВМ (Convertible)
43h	Чтение состояния системы (Convertible)
44h	Управление встроенным модемом (Convertible)
4Fh	Ловушка клавиатуры (многие)
80h	Открыть устройство (многие)
81h	Закрыть устройство (многие)
82h	Освободить устройство (многие)
83h	Ожидание события (многие)
84h	Работа с джойстиком (многие)
85h	Обработка SysReq (многие)
86h	Задержка (AT, XT-286, PS/2)
87h	Обмен с расширенной памятью (AT, XT-286, PS/2)
88h	**Получить размер расширенной памяти (AT, XT-286, PS/2)** ***Вариант №2.***
89h	Переключение в защищенный режим (AT, XT-286, PS/2)
90h	Устройство занято (многие)
91h	Прерывание завершено (многие)
C0h	**Получить конфигурацию системы (многие)** ***Вариант №5.***
C1h	Взять адрес дополнительной области данных BIOS (PS/2)
C2h	**Управление мышью (PS/2)** ***Вариант №3.***
C3h	Управление сторожем (PS/2)
C4h	Выбор программируемых опций (PS/2)

Байт BIOS для идентификации типа компьютера:

FF	оригинальный IBM PC;
FE	XT, Portable PC;
FD	PCjr;
FC	AT;
FB	XT с памятью 640 К на материнской плате;
FA	PS/2 модель 25 или 30;
F9	Convertible PC;
F8	PS/2 модели 55SX, 70, 80;
9A	Compaq XT, Compaq Plus;
30	Sperry PC;
2D	Compaq PC, Compaq Deskpro

Пример:

```
#include <stdio.h>
#include <dos.h>
#include "sysp.h"

char unsigned pc_model(void) {

    char unsigned _far *modptr;

    modptr = FP_MAKE(0xf000,0xfffe);

    return *modptr;
}
```

Функция C0h INT 15h

На входе: **AH = C0h**

На выходе: **ES:BX** = адрес таблицы конфигурации, таблица находится в ПЗУBIOS;

CF = 0 при успешном вызове прерывания;

CF = 1 если данная версия BIOS не поддерживает функцию C0h.

Формат таблицы по адресу в ES:BX – результат C0h INT 15h

Смещение и размер	Описание
(+0) 2	размер таблицы в байтах
(+2) 1	код модели
(+3) 1	дополнительный код модели
(+4) 1	версия BIOS revision: **0 –** для первой реализации; **2 –** для второй и т.д.
(+5) 1	байт конфигурации оборудования: **бит 7** = канал 3 контроллера прямого доступа к памяти используется дисковой системой базового ввода-вывода (дисковой BIOS) **бит 6** = установлен второй контроллер прерываний 8259 **бит 5** = установлены часы реального времени **бит 4** = каждый раз после вызова прерывания от клавиатуры INT 9h вызывается функция 4Fh прерывания INT 15h **бит 3** = BIOS поддерживает ожидание внешнего события **бит 2** = используется расширенная область данных BIOS **бит 1** = если этот бит установлен в 1, то используется шина Micro Channel, в противном случае – ISA **бит 0** = зарезервирован
(+6) 2	зарезервировано и равно 0
(+8) 2	зарезервировано и равно 0

Коды моделей, дополнительные коды моделей и версии BIOS для некоторых широко распространенных типов компьютеров:

Код модели	Доп.код модели	Версия BIOS	Тип компьютера
FFh	-	04/24/81	оригинальная версия IBM PC
FFh	-	10/19/81	IBM PC, в этой версии BIOS исправлены некоторые ошибки
FFh	-	10/27/82	IBM PC, используется накопитель на магнитном диске (НМД), оперативная память 640 К, поддерживается адаптер дисплея EGA
FEh	-	08/16/82	IBM PC XT
FEh	-	11/08/82	IBM PC XT, Portable
FDh	-	06/01/83	PCjr
FCh	-	01/10/84	IBM AT, модели 068, 099, частота тактового генератора 6 MHz, емкость НМД - 20MB
FCh	00h 01h	06/10/85	IBM AT, модель 239, частота тактового генератора 6 MHz, емкость НМД - 30MB
FCh	01h 00h	11/15/85	IBM AT, модели 319, 339, частота тактового генератора 8 MHz, используются расширенная клавиатура, BIOS может работать с накопителями на гибких магнитных дисках формата 3,5 дюйма
FCh	01h	-	Compaq 286/386
FCh	02h 00h	04/21/86	IBM PC XT-286
…..	…..	…..	…..

C0h INT 15h
F000h:FFFEh
F000h:FFF5h.

```
#include <stdio.h>
#include <dos.h>
#include "sysp.h"
void main(void);
void main(void) {
      union REGS  rg;
      struct SREGS srg;
      int i;
      BIOSINFO far *biosinf_ptr;
// Конструируем указатель на дату изготовления BIOS.
// Эта дата записана в ПЗУ по адресу F000h:FFF5h.
      biosinf_ptr = FP_MAKE(0xf000, 0xfff5);
// Выводим дату на экран
      printf("\n\nДата изготовления BIOS:     ");
      for(i=0; i<8; i++)
        putch(*((char far *)biosinf_ptr + i));
// Вызываем функцию C0h для получения адреса таблицы конфигурации компьютера.
      rg.h.ah = 0xc0;
      int86x(0x15, &rg, &rg, &srg);
// Если данная функция не поддерживается BIOS, читаем код модели компьютера из ПЗУ
// по адресу F000h:FFFEh.
      if(rg.x.cflag == 1) {
          printf("\nФункция C0h прерывания INT 15h данной версией BIOS не поддерживается\n");
// Конструируем указатель на код модели
          biosinf_ptr = FP_MAKE(0xf000, 0xfffe);
// Выводим код модели компьютера на экран
          printf("\nКод модели: %02.2X", (unsigned char)(*(char far *)biosinf_ptr));
          exit(-1);
      }
// Конструируем указатель на таблицу конфигурации
      biosinf_ptr = FP_MAKE(srg.es, rg.x.bx);
```

```
// Выводим на экран содержимое таблицы
        printf("\nАдрес таблицы конфигурации: %Fp"
                "\nРазмер таблицы в байтах:    %d"
                "\nКод модели:                 %02.2X"
                "\nДополнительный код модели:  %d"
                "\nВерсия BIOS:                %d"
                "\nКонфигурация оборудования:  %02.2X",
                biosinf_ptr,
                biosinf_ptr->size,
                biosinf_ptr->model,
                biosinf_ptr->submodel,
                biosinf_ptr->version,
                biosinf_ptr->hardcfg);
// Определяем конфигурацию компьютера
        printf("\n\nКонфигурация оборудования компьютера"
                "\n-----------------------------------");
// Запоминаем байт конфигурации
        i = biosinf_ptr->hardcfg;
// Расшифровываем байт конфигурации
        if(i & 0x80)
           printf("\nКанал 3 контроллера DMA используется дисковой BIOS");
        if(i & 0x40)
           printf("\nУстановлен второй контроллер прерываний 8259");
        if(i & 0x20)
           printf("\nУстановлены часы реального времени");
        if(i & 0x10)
           printf("\nПосле INT 9h вызывается функция 4Fh прерывания INT 15h");
        if(i & 0x8)
           printf("\nBIOS поддерживает функцию ожидания внешнего события");
        if(i & 0x4)
           printf("\nИспользуется расширенная область данных BIOS");
        if(i & 0x2)
           printf("\nИспользуется шина Micro Channel");
        if(!(i & 0x2))
           printf("\nИспользуется шина ISA");
        exit(0); }
```

Int 16h - Обслуживание клавиатуры

Функция	Назначение
00h	Чтение клавиатуры
01h	Опрос клавиатуры
02h	Состояние клавиатуры
03h	Задание скорости клавиатуры (PCjr, AT, PS/2)
04h	Управление щелчком клавиш (PCjr, Convertible)
05h	Запись в буфер клавиатуры (многие)
10h	Расширенное чтение клавиатуры (многие)
11h	Расширенный опрос клавиатуры (многие)
12h	Расширенное состояние клавиатуры (многие)

Int 17h - Управление принтером

Функция	Назначение
00h	Вывод символа на печать
01h	Инициализация принтера
02h	Опрос состояния принтера

Int 18h - Интерпретатор Бейсика

F600h:0

GW-BASIC

Int 19h - Загрузка системы

0:7C00h

F000h:FFF0h

00472h

1234h

Int 1Ah - Обслуживание таймера_F

Функция	Назначение
00h	Чтение системного таймера
01h	Установка системного таймера
02h	Чтение текущего времени RTC (AT)
03h	Установка текущего времени RTC (AT)
04h	Чтение текущего даты RTC (AT)
05h	Установка текущего даты RTC (AT)
06h	Задание времени побудки RTC (AT)
07h	Сброс времени побудки RTC (AT)
08h	Задание времени включения ПЭВМ (Convertible)
09h	Чтение параметров побудки (Convertible, PS/2-30)
0Ah	Чтение количества дней (некоторые XT, PS/2)
0Bh	Установка количества дней (некоторые XT, PS/2)
80h	Управление звукогенератором (PCjr)

Int 1bH – прерывание с клавиатуры

0:006c

IRET

Int 1cH – пользовательское прерывание по таймеру

0:0070

Int 1dH – указатель видеопараметров

0:0074

INT 10H

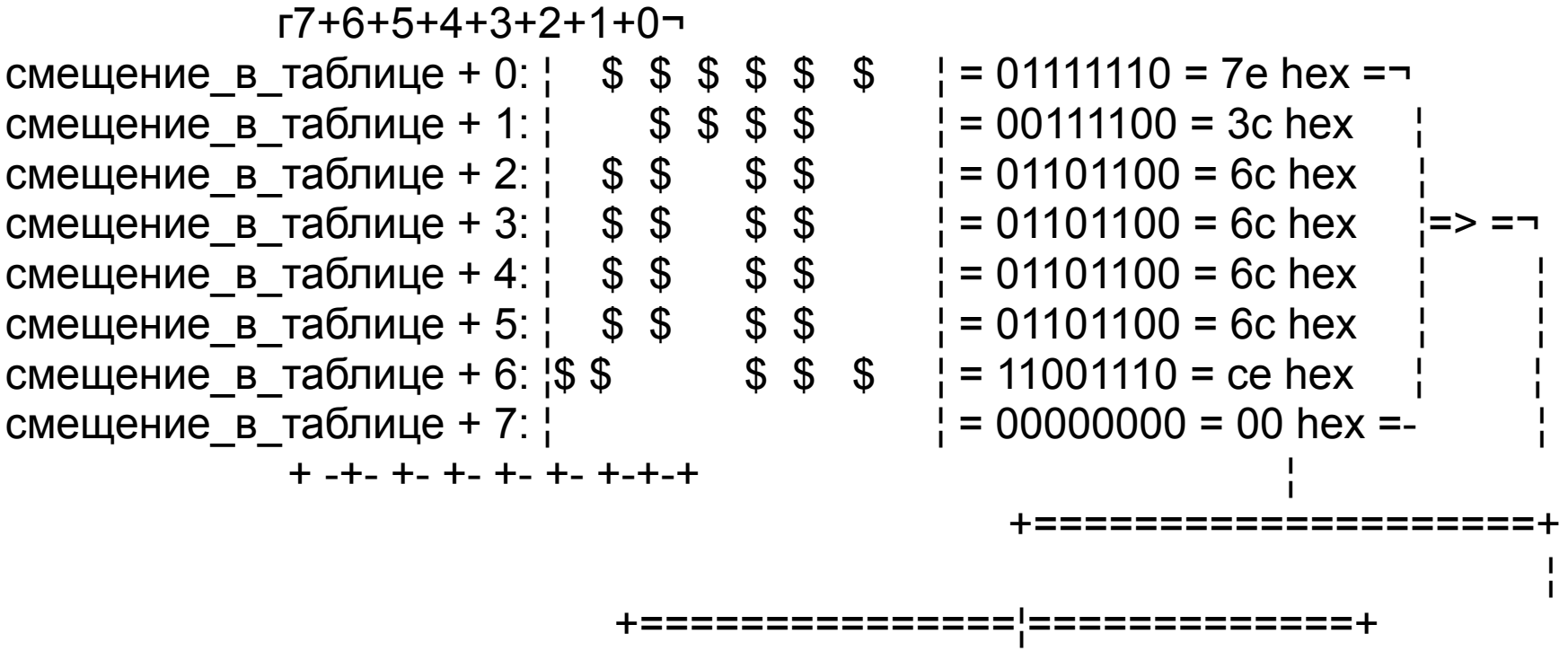
Int 1eH – указатель параметров дискеты

0:0078

INT 13H

Int 1fH – указатель графических символов

0:007
ROM-BIOS
F000:0000



8-байтовая последовательность: 7eH,3cH,6cH,6cH,6cH,6cH,ceH,00H стояла бы в таблице по смещению, соответствующему символу "Л". Так как код ASCII буквы 'Л' равен 139, а таблица начинается для символа с кодом 128, это будет 12-я группа из 8 байт (смещение - 88 байт от начала таблицы).

f000:fa6e

Определение типа центрального процессора

Модели Intel 8086/8088

```
pushf
pop ax

mov cx, ax

and ax, 0fffh
push ax
popf

pushf
pop ax
and ax, 0f000h
cmp ax, 0f000h
je is_8086
```

Модель Intel 80286

```
mov ax, 0f000h
push ax
popf

pushf
pop ax

and ax, 0f000h
jz is_80286
```

Модель Intel 80386

```
pushfd
pop eax
mov ecx, eax

xor eax, 40000h
push eax
popfd

pushfd
pop eax
xor eax, ecx
jz is_80386
```

Модель Intel 80486

```
pushfd
pop eax
mov ecx, eax
xor eax, 200000h
push eax
popfd
pushfd
pop eax
xor eax, ecx
je is_80486
```

Команда CPUID

```
CPU_ID MACRO
    db 0fh
    db 0a2h
ENDM

mov eax, 00h
CPU_ID

_vendor_id_msg db ".....", 0dh, 0ah, "$"
. . .
mov dword ptr _vendor_id_msg,    ebx
mov dword ptr _vendor_id_msg[+4], edx
mov dword ptr _vendor_id_msg[+8], ecx

_cpu_signature dd 0
_features_edx  dd 0
. . .
mov _cpu_signature, eax
mov _features_edx, edx
```

Формат слова сигнатуры

Биты	Описание
0-3	Код модификации модели (stepping)
4-7	Код модели
8-11	Код семейства моделей
12-13	Тип процессора
14-31	Зарезервировано

Биты 12 и 13 определяют тип процессора:

Значение битов 12 и 13	Тип процессора
00	Процессор, изготовленный производителем OEM
01	Процессор OverDrive
10	Процессор типа Dual, который можно использовать в двухпроцессорных системах
11	Зарезервировано

Тип процессора	Код семейства	Код модели	Описание процессора
00	0100	0100	Intel 486 SL
00	0100	0111	Intel DX2
00	0100	1000	Intel DX4
00, 01	0100	1000	Intel DX4 OverDrive
00	0101	0001	Pentium 60, 66; Pentium OverDrive для процессоров Pentium 60, 66
00	0101	0010	Pentium 75, 90, 100, 120, 133, 150, 166, 200
01	0101	0010	Pentium OverDrive для процессоров Pentium 75, 90, 100, 120, 133
01	0101	0011	Pentium OverDrive для систем на базе процессора Intel 486
00	0101	0100	Pentium 166, 200 с командами MMX
01	0101	0100	Зарезервировано. Будет использоваться процессорами Pentium OverDrive для процессоров Pentium 75, 90, 100, 120, 133
00	0110	0001	Pentium Pro
00	0110	0011	Pentium II
00	0110	0101	Зарезервировано для новых процессоров P6
01	0110	0011	Зарезервировано для процессоров Pentium OverDrive для процессоров Pentium Pro

Бит	Описание
0	На кристалле процессора имеется арифметический сопроцессор, совместимый по командам с сопроцессором Intel 387
1	Процессор может работать в режиме виртуального процессора 8086
2	Процессор может работать с прерываниями ввода/вывода, а также с битом DE регистра CR4
3	Возможно использование страниц памяти размером 4 Мбайт
4	В процессоре есть команда RDTSC, которая может работать с битом TSD регистра CR4
5	Набор регистров процессора, специфический для модели, доступен с помощью команд RDMSR, WRMSR
6	Возможна физическая адресация памяти с использованием шины с шириной, большей чем 32 разряда
7	В процессоре реализовано исключение Machine Check (исключение с номером 18). Возможно использование бита MCE регистра CR4
8	В процессоре реализована команда сравнения и обмена 8 байт данных CMPXCHG8
9	В процессоре есть локальный APIC
10	Зарезервировано
11	В процессоре реализованы команды быстрого вызова системы SYSENTER и SYSEXIT
12	В процессоре есть регистры Memory Type Range
13	Доступен глобальный бит в PDE и PTE, а также бит PGE в регистре CR4
14	Применена архитектура Machine Check Architecture
15	В процессоре реализованы команды условного перемещения данных CMOVCC и (при установленном бите 0) FCMOVCC и FCOMI
16-22	Зарезервировано
23	Применена технология MMX
24-31	Зарезервировано

Базовые функции для прерывания DOS INT 21h

00 Завершение программы (аналогично int 20h).

01 Ввод символа с клавиатуры с эхом на экран.

02 Вывод символа на экран.

03 Ввод символа из асинхронного коммуникационного канала.

04 Вывод символа на асинхронный коммуникационный канал.

05 Вывод символа на печать.

06 Прямой ввод с клавиатуры и вывод на экран.

07 Ввод с клавиатуры без эха и без проверки ctrl/break.

08 Ввод с клавиатуры без эха с проверкой ctrl/break.

09 Вывод строки символов на экран.

0A Ввод с клавиатуры с буферизацией.

0B Проверка наличия ввода с клавиатуры.

0C Очистка буфера ввода с клавиатуры и запрос на ввод.

0d Сброс диска.

0E Установка текущего дисковода.

0f Открытие файла через fcb.

10 Закрытие файла через fcb.

11 Начальный поиск файла по шаблону.

12 Поиск следующего файла по шаблону.

13 Удаление файла с диска.

14 Последовательное чтение файла.

15 Последовательная запись файла.

16 Создание файла.

17 Переименование файла.

18 Внутренняя операция dos.

19 Определение текущего дисковода.

1A Установка области передачи данных (dta).

1B Получение таблицы fat для текущего дисковода.

1C Получение fat для любого дисковода.

21 Чтение с диска с прямым доступом.

22 Запись на диск с прямым доступом.

23 Определение размера файла.

24 Установка номера записи для прямого доступа.

25 Установка вектора прерывания.

26 Создание программного сегмента.

27 Чтение блока записей с прямым доступом.

28 Запись блока с прямым доступом .

29 Преобразование имени файла во внутренние параметры.

2A Получение даты (cx-год,dh-месяц,dl-день).

Вариант №6.

2B Установка даты.

2C Получение времени (ch-час,cl-мин,dh-с, dl-1/100c).

Вариант №6.

2d Установка времени.

2E Установка/отмена верификации записи на диск.

Расширенные функции возможны в dos начиная с версии 2.0

2f Получение адреса dta в регистровой паре es:bx.

Для многих вариантов.

30 Получение номера версии dos в регистре AX.

31 Завершение программы, после которого она остается резидентной в памяти.

33 Проверка ctrl/break.

35 Получение вектора прерывания (адреса подпрограммы).

36 Получение размера свободного пространства на диске.

38 Получение государственно зависимых форматов.

Вариант №10.

39 Создание подкаталога (команда mkdir).

3A Удаление подкаталога (команда rmdir).

3B Установка текущего каталога (команда chdir).

3c Создание файла без использования fcb.

3d Открытие файла без использования fcb.

3e Закрытие файла без использования fcb.

3f Чтение из файла или ввод с устройства.

40 Запись в файл или вывод на устройство.

41 Удаление файла из каталога.

42 Установка позиции для последовательного доступа .

43 Изменение атрибутов файла.

44 Управление вводом-выводом для различных устройств.

45 Дублирование файлового номера.

46 "Склеивание" дублированных файловых номеров.

47 Получение текущего каталога.

48 Выделение памяти из свободного пространства.

49 Освобождений выделенной памяти.

4A Изменение длины блока выделенной памяти.

4B Загрузка/выполнение программы (подпроцесса).

4C Завершение подпроцесса с возвратом управления.

4d Получение кода завершения подпроцесса.

4E Начальный поиск файла по шаблону.

4f Поиск следующего файла по шаблону.

54 Получение состояния верификации.

56 Переименование файла.

57 Получение/установка даты и времени изменения файла.

Расширенные функции возможны в dos начиная с версии 3.0

59 Получение расширенного кода ошибки.
5A Создание временного файла.
5B Создание нового файла.
5C Блокирование/разблокирование доступа к файлу.
62 Получение адреса префикса программного сегмента (psp).

DOS INT 21H – 38H – дать/установить информацию о стране

Вход

AH = 38H

DX = 0ffffH чтобы установить код страны равным **AL** (или **BX**)

DS:DX = адрес локального буфера для чтения блока данных страны

AL = 0 чтобы получить данные для текущей страны

= 1 до 0feH чтобы получить данные для указанной страны

= 0ffH чтобы получить данные для кода страны : 255

BX = (если AL=0ffH) 16-битовый код страны

Выход

AX = код ошибки если флаг CF установлен

BX = код страны (если AL=0 при входе)

DX=0ffffH

AL=0ffH

Вход

AL=0

DS:DX = адрес 20-байтового локального буфера

Выход

Пересылает в DS:DX блок данных страны для DOS 2.x

Hex	Dec	страна	Keybrd	дата	время	формат валюты	DS
3dH	061	Австралия		DD-MM-YY	13:00:00	$1,000.00	,
20H	032	Бельгия		DD/MM/YY	13:00:00	1 000,00 F	;
2	002	Французская Канада		YY-MM-DD	13:00:00	1 000,00 $	;
2dH	045	Дания		DD/MM/YY	13.00:00	1.000,00 DKR	;
166H	358	Финляндия		DD-MM-YY	13:00:00	1 000,00 MK	;
21H	033	Франция	keybFR	DD/MM/YY	13:00:00	1 000,00 F	;
31H	049	Германия	keybGR	DD.MM.YY	13.00.00	DM1.000,00	;
27H	039	Италия	keybIT	DD/MM/YY	13:00:00	1.000,Lit.	;
3ccH	972	Израиль		DD/MM/YY	13:00:00	Щ 1,000.00	,
311H	785	Средний Восток		DD/MM/YY	01:00:00 PM	1.000,000 $	;
1fH	031	Нидерланды		DD-MM-YY	13:00:00	Я1.000,00	;
2fH	047	Норвегия		DD/MM/YY	13.00.00	KR 1.000,00	;
15fH	351	Португалия		DD/MM/YY	13:00:00	1.000$00	;
22H	034	Испания	keybSP	DD/MM/YY	13:00:00	1.000,00 Ю	;
2eH	046	Швеция		YY-MM-DD	13.00.00	SEK 1.000,00	;
29H	041	Швейцария		DD.MM.YY	13.00.00	Fr 1,000.00	,
2cH	044	Великобритания	keybUK	DD-MM-YY	13:00:00	Ь1,000.00	,
1	001	Соединенные Штаты		MM-DD-YY	01:00:00 PM	$1,000.00	,

DOS INT 21h – 32H – дать информацию DOS о диске

Вход

AH = 32H

DL = номер диска (0=текущий, 1=A, и т.д.)

Выход

AL = 0 если DL задавал корректный диск

FF = 0ffH если диск задан неверно

DS:BX = адрес блока информации диска для запрошенного устройства

INT 33h – прерывание для обслуживания мыши

Инициализация мыши

На входе: **AX = 0000h**

На выходе: **AX** = состояние мыши;
BX = количество клавиш у мыши

Содержимое регистра BX	Количество клавиш
0	Больше или меньше, чем две
2	Две клавиши
3	Мышь системы Mouse Systems, имеет три клавиши

Включить курсор мыши

На входе: **AX = 0001h**

На выходе: Регистры не используются

Выключить курсор мыши

На входе: **AX = 0002h**

На выходе: Регистры не используются

Определить положение курсора

На входе: **AX = 0003h**

На выходе: **BX** = состояние клавиш мыши;
CX = координата X курсора;
DX = координата Y курсора

Установленный бит регистра BX	Клавиша, которая была нажата
0	Левая
1	Правая
2	Средняя

Установить курсор

На входе: **AX = 0004h**
CX = новая координата X курсора;
DX = новая координата Y курсора

На выходе: Регистры не используются

Определить положение курсора при нажатии клавиши

На входе: **AX = 0005h**
BX = клавиша, при нажатии которой запоминается состояние мыши:
0 - левая;
1 - правая;
2 - средняя

На выходе: **AX** = состояние клавиш мыши;
BX = количество нажатий на заданную клавишу. Это значение обнуляется после вызова функции;
CX = координата курсора X;
DX = координата курсора Y

Установленный бит регистра AX	Клавиша, которая была нажата
0	Левая
1	Правая
2	Средняя

Определить положение курсора при отпускании клавиши

На входе: **AX = 0006h**
BX = клавиша, при отпускании которой запоминается состояние мыши:
0 - левая;
1 - правая;
2 – средняя

На выходе: **AX** = состояние клавиш мыши;
BX = количество нажатий на заданную клавишу. Это значение обнуляется после вызова функции;
CX = координата курсора X;
DX = координата курсора Y

Задать диапазон движения курсора по горизонтали

На входе: **AX = 0007h**
CX = минимальная координата X;
DX = максимальная координата X

На выходе: Регистры не используются

Задать диапазон движения курсора по вертикали

На входе: **AX = 0008h**
CX = минимальная координата Y;
DX = максимальная координата Y

На выходе: Регистры не используются

Задать форму курсора в графическом режиме

На входе: **AX = 0009h**
BX = номер позиции точки-указателя графического курсора (от -16 до 16);
CX = номер строки точки-указателя (от -16 до 16);
ES:DX = указатель на растровое изображение курсора

На выходе: Регистры не используются

Задать форму курсора в текстовом режиме

На входе: **AX = 000Ah**
BX = тип курсора:
0 - определяемый программно;
1 - определяемый аппаратно;
CX = маска экрана (для BX=0) или начальная строка курсора (для BX=1);
DX = маска курсора (для BX=0) или конечная строка курсора (для BX=1)

На выходе: Регистры не используются

Определить содержимое счетчиков перемещения

На входе: **AX = 000Bh**

На выходе: **CX** = перемещение по горизонтали с момента последнего вызова функции;
DX = перемещение по вертикали с момента последнего вызова функции

Установить драйвер событий

На входе: **AX = 000Ch**
CX = маска вызова;
ES:DX = адрес подключаемого драйвера событий

На выходе: Регистры не используются

Бит маски вызова	Когда выполняется вызов
0	Перемещение мыши
1	Нажатие левой клавиши
2	Отпускание левой клавиши
3	Нажатие правой клавиши
4	Отпускание правой клавиши
5	Нажатие средней клавиши
6	Отпускание средней клавиши

Регистр	Описание
AX	Маска вызова, такая же, как и при вызове функции 0Ch
BX	Состояние клавиш мыши: бит 0 - левая клавиша; бит 1 - правая клавиша; бит 2 - средняя клавиша
CX	Координата X курсора мыши
DX	Координата Y курсора мыши
SI	Относительное перемещение мыши по горизонтали в миках
DI	Относительное перемещение мыши по вертикалив миках
DS	Сегмент данных драйвера мыши

Включить эмуляцию светового пера

На входе: **AX = 000Dh**
На выходе: Регистры не используются

Выключить эмуляцию светового пера

На входе: **AX = 000Eh**
На выходе: Регистры не используются

Задать скорость перемещения курсора мыши

На входе: **AX = 000Fh**
CX = количество миков на 8 точек по горизонтали;
DX = количество миков на 8 точек по вертикали
На выходе: Регистры не используются

Установить область исключения для курсора

На входе: **AX = 0010h**
CX, **DX** = координаты (X, Y) верхнего левого угла области исключения;
SI, **DI** = координаты (X, Y) нижнего правого угла области исключения
На выходе: Регистры не используются

Задать увеличенный графический курсор

На входе: **AX = 0012h**
BH = ширина курсора в словах;
CH = количество строк в изображении курсора;
BL = номер позиции точки-указателя графического курсора (от -16 до 16);
CL = номер строки точки-указателя (от -16 до 16);
ES:DX = указатель на растровое изображение курсора
На выходе: Регистры не используются

Определить порог удвоения скорости

На входе: **AX = 0013h**
На выходе: **DX** = значение порога удвоения, мики в секунду

Заменить драйвер событий

На входе: **AX = 0014h**
CX = маска вызова;
ES:DX = адрес подключаемого драйвера событий
На выходе: **CX** = маска предыдущего драйвера событий;
ES:DX = адрес предыдущего (заменяемого) драйвера событий

Определить размер буфера состояния драйвера

На входе: **AX = 0015h**
На выходе: **BX** = размер буфера, требующийся для хранения состояния драйвера мыши

Сохранить состояние драйвера

На входе: **AX = 0016h**
ES:DX = адрес буфера для записи состояния драйвера
На выходе: Регистры не используются

Восстановить состояние драйвера

На входе: **AX = 0017h**
ES:DX = адрес буфера, содержащего состояние драйвера
На выходе: Регистры не используются

Установить альтернативный драйвер событий

На входе: **AX = 0018h**
CX = маска вызова;
ES:DX = адрес подключаемого драйвера событий

На выходе: **AX** = результат установки:
0018h - драйвер успешно установлен;
FFFFh - ошибка при установке драйвера

Получить адрес альтернативного драйвера событий

На входе: **AX = 0019h**
CX = маска событий, для которой требуется получить адрес драйвера

На выходе: **CX** = маска событий или 0000h, если заданной маске не соответствует ни один установленный драйвер событий;
ES:DX = адрес драйвера событий, использующий заданную маску событий

Установить чувствительность мыши

На входе: **AX = 001Ah**
BX = горизонтальная чувствительность в миках на пиксел;
CX = вертикальная чувствительность в миках на пиксел;
DX = значение порога удвоения, мики в секунду

На выходе: Регистры не используются

Определить чувствительность мыши

На входе: **AX = 001Bh**

На выходе: **BX** = горизонтальная чувствительность в миках на пиксел;
CX = вертикальная чувствительность в миках на пиксел;
DX = значение порога удвоения, мики в секунду

Установить частоту прерываний для Inport Mouse

На входе: **AX = 001Ch**
BX = код скорости прерываний:
1 - нет прерываний;
2 - 30 прерываний в секунду;
4 - 50 прерываний в секунду;
8 - 100 прерываний в секунду;
16 - 200 прерываний в секунду

На выходе: Регистры не используются

Установить номер страницы видеопамяти

На входе: **AX = 001Dh**
BX = номер страницы видеопамяти

На выходе: Регистры не используются

Определить номер страницы видеопамяти

На входе: **AX = 001Eh**

На выходе: **BX** = номер страницы видеопамяти

Отключить драйвер мыши

На входе: **AX = 001Fh**

На выходе: **AX** = результат выполнения:
0001Fh - драйвер отключен;
FFFFh - отключение невозможно;
ES:DX = адрес предыдущего драйвера мыши

Восстановить драйвер мыши

На входе: **AX = 0020h**

На выходе: Регистры не используются

Сбросить драйвер мыши

На входе: **AX = 0021h**

На выходе: **AX** = результат:
0021h - драйвер сброшен успешно;
FFFFh - невозможно сбросить драйвер
(например, из-за того что драйвер
не установлен);
BX = количество клавиш на корпусе мыши

Определить тип мыши

На входе: **AX = 0024h**

На выходе: **BH** = верхний (major)
номер версии драйвера;
BL = нижний (minor)
номер версии драйвера;
CH = тип мыши:
1 - Bus Mouse;
2 - Serial Mouse;
3 - Inport Mouse;
4 - PS/2 Mouse;
5 - HP Mouse;
CL = номер используемого
прерывания (IRQ):
0 - IBM PS/2;
2, 3, 4, 5, 7 - IBM PC

Регистры часов реального времени

Регистры счетчиков

Регистр	Счетчик, который содержится в регистре
0	Секунды
1	Секунды будильника
2	Минуты
3	Минуты будильника
4	Часы
5	Часы будильника
6	День недели (1 - воскресенье)
7	День месяца
8	Номер месяца
9	Последние две цифры текущего года

Регистр состояния A

Биты регистра	Описание
0-3	Переключатель скорости. По умолчанию установлен в 0110
4-6	22-разрядный делитель. По умолчанию установлен в 010
7	Флаг обновления. Значение 0 означает готовность данных для чтения

Регистр состояния B

Биты регистра	Описание
0	Летнее или стандартное время: 1 - летнее время; 0 – стандартное время
1	12 или 24-часовой режим: 0 - 12-часовой режим 1 – 24-часовой режим (установлен по умолчанию)
2	Формат данных: 1 – двоичный; 0 - BCD (установлен по умолчанию)
3	Разрешение прямоугольного сигнала: 1 – включение сигнала, частота которого определяется разрядами 0-3 первого регистра состояния; 0 – сигнал выключен
4	Разрешение прерывания по окончанию изменения данных (по умолчанию сброшен)
5	Разрешение прерывания будильника (по умолчанию сброшен)
6	Разрешение периодических прерываний (по умолчанию сброшен)
7	Запрет счета: 1 – счетчик остановлен; 0 – счетчик запущен

Функции прерывания INT 1Ah

Прочитать показания часов реального времени

На входе: **AH = 02h**

На выходе: **CH** = часы в BCD-формате (например, 13h означает 13 часов);
CL = минуты в BCD-формате;
DH = секунды в BCD-формате;
CF = CY = 1, если часы реального времени не установлены

Установить часы реального времени

На входе: **AH = 03h**
CH = часы в BCD-формате (например, 13h означает 13 часов);
CL = минуты в BCD-формате;
DH = секунды в BCD-формате;
DL = 1, если необходимо использовать летнее время

На выходе: Регистры не используются

Прочитать дату из часов реального времени

На входе: **AH = 04h**

На выходе: **CH** = столетие в BCD-формате;
CL = год в BCD-формате (например, CX=1997h означает 1997 год);
DH = месяц в BCD-формате;
DL = число в BCD-формате;
CF = CY = 1, если часы реального времени не установлены

Установить дату в часах реального времени

На входе: **AH = 05h**
CH = столетие в BCD-формате;
CL = год в BCD-формате (например, **CX**=1997h означает 1997 год);
DH = месяц в BCD-формате;
DL = число в BCD-формате;

На выходе: Регистры не используются

Установить будильник

На входе: **AH = 06h**
CH = часы в BCD-формате;
CL = минуты в BCD-формате;
DH = секунды в BCD-формате

На выходе: **CF** = CY = 1, если часы реального времени не установлены

Сброс будильника

На входе: **AH = 07h**

На выходе: Регистры не используются

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>

typedef struct _SYSTIMER_
{  char hour;
    char min;
    char sec;
    unsigned year;
    char month;
    char day;
    char daylight_savings;
}  SYSTIMER;

#define GET_TIME
#define GET_DATE

int bcd1bin(char *bcd);
int bcd2bin(char *bcd);
int timer(char fn, SYSTIMER *treal);

#pragma check_stack( off )
#pragma check_pointer( off )

union REGS reg;

int main(void)
{
  char *month_to_text[] =
  {  "январь",
     "февраль",
     "март",
     "апрель",
     "май",
     "июнь",
     "июль",
     "август",
     "сентябрь",
     "октябрь",
     "ноябрь",
     "декабрь"  };
  SYSTIMER timereal;
  timer(GET_DATE, &timereal);
  timer(GET_TIME, &timereal);
   printf("\nСейчас %d год, %s, %d число."
    "\n",
       bcd2bin((char*)&(timereal.year)),
       month_to_text[bcd1bin(&(timereal.month)) - 1],
       bcd1bin(&(timereal.day)));
  printf("\nВремя - %02.2d:%02.2d:%02.2d"
    "\n",
        bcd1bin(&(timereal.hour)),
        bcd1bin(&(timereal.min)),
        bcd1bin(&(timereal.sec)));
  getch();
  return 0;
}
int bcd1bin(char *bcd)
{
  return( ((*bcd) & 0x0f) +10 * (((*bcd) & 0xf0) >> 4) );
}
int bcd2bin(char *bcd)
{
  return( bcd1bin(bcd) +100 * bcd1bin(bcd + 1) );
}
int timer(char fn, SYSTIMER *treal)
{  reg.h.ah = fn;
    int86(0x1a,&reg,&reg);
    if(reg.x.cflag == 1)
       return(-1);
    switch (fn)
    {  case GET_TIME:
        {  treal->hour = reg.h.ch;
          treal->min = reg.h.cl;
          treal->sec = reg.h.dh;
            break;  }
       case GET_DATE:
        {  treal->year = reg.x.cx;
          treal->month = reg.h.dh;
          treal->day = reg.h.dl;
            break;  }
    }
  return 0;
}
```

1. **Определить наличие и тип установленного в системе накопителя на жестких магнитных дисках (НЖМД).**

Int 11h (бит 0) Int 13h CMOS (ячейки 12h, 19h, 1Ah)

2. **Определить объем установленной оперативной памяти и ее тип.**

Int 11h (бит 2 – 3) Int 12h Int 15h (функция 88h) Int 67h (функция 42h) CMOS (ячейки 15h – 16h, 17h – 18h)

3. **Определить наличие тип установленного манипулятора «мышь» и написать программу для считывания данных с этого устройства, например, сообщать пользователю о наличии данных, поступающих с мышки.**

Int 15h (функция C2h) Int 33h

4. **Определить количество установленных последовательных и параллельных портов.**

Int 11h (бит 9 – 11, 14 – 15) Data Area BIOS (040:000, 040:008)

5. **Определить модель компьютера (BIOS INT 15h) и версию BIOS.**

Int 15h (функция C0h)

6. **Написать программу, извлекающую текущее время и дату из области CMOS и отображающую эту информацию на экране в реальном времени.**

CMOS (ячейки 00h – 0Dh) Int 21h (функции 2Ah, 2Bh, 2Ch, 2Dh) Int 1Ah

7. **Определить состояние байта диагностики из области CMOS (0Eh) и написать программу, анализирующую возможные неисправности в системе.**

CMOS (ячейка 0Eh) Int 15h (функция 21h)

8. **Определить количество и тип установленных накопителей на гибких магнитных дисках (НГМД).**

Int 11h (бит 6 – 7) Int 13h (функции 08h, 15h) CMOS (ячейка 10h) Int 1Eh

9. **Определить тип центрального процессора.**

CPUID

2. **Определить и вывести на экран информацию о стране (DOS 38h).**

Int 21h (функция 38h)

3. **Определить и вывести на экран список оборудования (BIOS 11h).**

Int 11h CMOS (ячейка 14h) Data Area BIOS (040:010, 040:017)

4. **Определить режим видеоадаптера (текстовый, графический, разрешение) (BIOS 10h).**

Int 10h (функции 00h, 0Fh) Int 11h (биты 4 – 5) Data Area BIOS (040:049) Int 12h

5. **Определить статус последовательного порта связи (BIOS 14h).**

Int 14h

6. **Определить параметры указанного диска (DOS 32h).**

Int 21h (функция 32h)