|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **ПРОГРАМНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.03 ПРОГРАММНАЯ ИНЖЕНЕРИЯ**

**Отчет**

|  |  |
| --- | --- |
| **По лабораторной работе №** | 1 |

**Название:**

Дизассемблирование INT 8h

**Дисциплина:** Операционные системы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-52Б |  |  | Н.А. Гарасев |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Н.Ю. Рязанова |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

Оглавление

[Введение 3](#_Toc51234700)

[Листинг 4](#_Toc51234703)

[Схема – алгоритма 7](#_Toc51234724)

[Заключение: 11](#_Toc51234725)

# Введение

Цель лабораторной работы: знакомство со средством дизассемблирования – **sourcer** и с получением дизассемблерного кода ядра операционной системы Windows на примере обработчика прерывания **Int 8h** в **virtual mode** – специальном режиме защищенного режима (32-разрядный режим работы), который эмулирует реальный режим работы вычислительной системы на базе процессоров Intel.

## Задание:

Используя sourser (sr.exe) получить дизассемблированный код обработчика аппаратного прерывания от системного таймера Int 8h.

На основе полученного кода составить алгоритм работы обработчика Int 8h.

По данной лабораторной работе составляется отчет в письменном виде.

• Отчет должен содержать: полученный ассемблерный код с адресами команд и комментариями;

• Графический алгоритм работы обработчика прерывания Int 8h, структурированный и выполненный в соответствии с ГОСТ 19.701-90 ЕСПД – «Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения».

# **Листинг**

Листинг 1. прерывания INT 08h

020A:0746 E8 0070 ;\* call sub\_1 ; (07B9)

**сохранение значений регистров ES, DS, AX, DX**

020A:0746 E8 70 00 db 0E8h, 70h, 00h

020A:0749 06 push es

020A:074A 1E push ds

020A:074B 50 push ax

020A:074C 52 push dx

**в DS помещаем начало области данных BIOS**

020A:074D B8 0040 mov ax,40h

020A:0750 8E D8 mov ds,ax

**в ES помещаем адрес начала таблицы векторов прерывания**

020A:0752 33 C0 xor ax,ax ; Zero register

020A:0754 8E C0 mov es,ax

**инкремент счётчика таймера**

**инкремент младшей части счётчика таймера**

020A:0756 FF 06 006C inc word ptr ds:[6Ch] ; (0040:006C=0AB71h)

020A:075A 75 04 jnz loc\_1 ; Jump if not zero

**инкремент старшей части счётчика таймера**

020A:075C FF 06 006E inc word ptr ds:[6Eh] ; (0040:006E=0Dh)

**сброс счётчика таймера и выставление флага окончания суток, если счётчик таймера превысил 24 часа**

020A:0760 loc\_1:

020A:0760 83 3E 006E 18 cmp word ptr ds:[6Eh],18h ; (0040:006E=0Dh)

020A:0765 75 15 jne loc\_2 ; Jump if not equal

020A:0767 81 3E 006C 00B0 cmp word ptr ds:[6Ch],0B0h ; (0040:006C=0AB71h)

значит в счётчике записано число 1573040 тиков == 24 часа

020A:076D 75 0D jne loc\_2 ; Jump if not equal

020A:076F A3 006E mov word ptr ds:[6Eh],ax ; (0040:006E=0Dh)

020A:0772 A3 006C mov word ptr ds:[6Ch],ax ; (0040:006C=0AB71h)

**В ячейку 0040:0070 записываем единицу (наступил новый день)**

020A:0775 C6 06 0070 01 mov byte ptr ds:[70h],1 ; (0040:0070=0)

020A:077A 0C 08 or al,8

**декремент счетчика до отключения моторчика дисковода**

020A:077C loc\_2:

020A:077C 50 push ax

020A:077D FE 0E 0040 dec byte ptr ds:[40h] ; (0040:0040=0FFh)

020A:0781 75 0B jnz loc\_3 ; Jump if not zero

020A:0783 80 26 003F F0 and byte ptr ds:[3Fh],0F0h ; (0040:003F=0)

020A:0788 B0 0C mov al,0Ch

020A:078A BA 03F2 mov dx,3F2h

020A:078D EE out dx,al ; port 3F2h, dsk0 contrl output

**проверка, установлен ли PF(parity flag) – т.е. разрешен ли ответ на маскируемые прерывания**

020A:078E loc\_3:

020A:078E 58 pop ax

020A:078F F7 06 0314 0004 test word ptr ds:[314h],4 ; (0040:0314=3200h)

**если вызов маскируемых прерываний разрешен, переход к прямому вызову int 1Ch (в loc\_4)**

020A:0795 75 0C jnz loc\_4 ; Jump if not zero

**LAHF перемещает младший байт регистра флагов EFLAGS в регистр AH**

020A:0797 9F lahf ; Load ah from flags

020A:0798 86 E0 xchg ah,al

020A:079A 50 push ax

**иначе, косвенный вызов 1Сh - как процедуры командой call и переход к loc\_5**

020A:079B 26: FF 1E 0070 call dword ptr es:[70h] ; (0000:0070=6ADh)

020A:07A0 EB 03 jmp short loc\_5 ; (07A5)

020A:07A2 90 nop

**прямой вызов пользовательского прерывания по таймеру**

020A:07A3 loc\_4:

020A:07A3 CD 1C int 1Ch ; Timer break (call each 18.2ms)

**после инициализации системы вектор INT 1Ch указывает на команду IRET**

**сброс контроллера прерываний**

020A:07A5 loc\_5:

020A:07A5 E8 0011 call sub\_1 ; (07B9)

020A:07A8 B0 20 mov al,20h ; ' '

020A:07AA E6 20 out 20h,al ; port 20h, 8259-1 int command

; al = 20h, end of interrupt

**восстановление значений регистров**

020A:07AC 5A pop dx

020A:07AD 58 pop ax

020A:07AE 1F pop ds

020A:07AF 07 pop es

020A:07B0 E9 FE99 jmp $-164h ; прыжок в адрес 020A:064C

**Выход из прерывания**

020A:06AC CF iret ; Interrupt return

;\* No entry point to code

*Листинг 2. Sub\_1 SUBROUTINE*

**сохранение значений регистров**

020A:07B9 1E push ds

020A:07BA 50 push ax

**в DS помещаем начало области данных BIOS**

020A:07BB B8 0040 mov ax,40h

020A:07BE 8E D8 mov ds,ax

**сохранение значений флагов (значение флагового регистра в регистр АН)**

020A:07C0 9F lahf ; Load ah from flags

**проверка флага DF и старшего бита IOPL**

020A:07C1 F7 06 0314 2400 test word ptr ds:[314h],2400h ; (0040:0314=3200h)

020A:07C7 75 0C jnz loc\_7 ; Jump if not zero

**сброс флага IF**

020A:07C9 F0> 81 26 0314 FDFF lock and word ptr ds:[314h],0FDFFh ; (0040:0314=3200h)

**восстановление значений флагов**

020A:07D0 loc\_6:

020A:07D0 9E sahf ; Store ah into flags

**восстановление значений регистров**

020A:07D1 58 pop ax

020A:07D2 1F pop ds

020A:07D3 EB 03 jmp short loc\_8 ; (07D8)

**сброс IF – запрет прерываний с помощью команды cli**

020A:07D5 loc\_7:

**CLI − сбрасывает флаг IF ( процессор перестает принимать прерывания от внешних устройств)**

020A:07D5 FA cli ; Disable interrupts

020A:07D6 EB F8 jmp short loc\_6 ; (07D0)

**выход из подпрограммы**

020A:07D8 loc\_8:

020A:07D8 C3 retn

sub\_1 endp

# Схема – алгоритма

На рис. 1 – 4 представлены схемы алгоритма int 8h и подпрограммы sub\_1

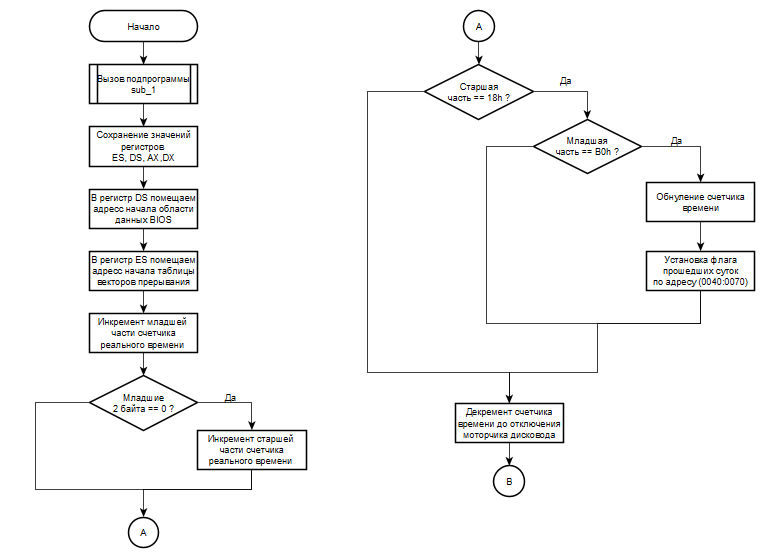
****

Рисунок 1. Int 8h

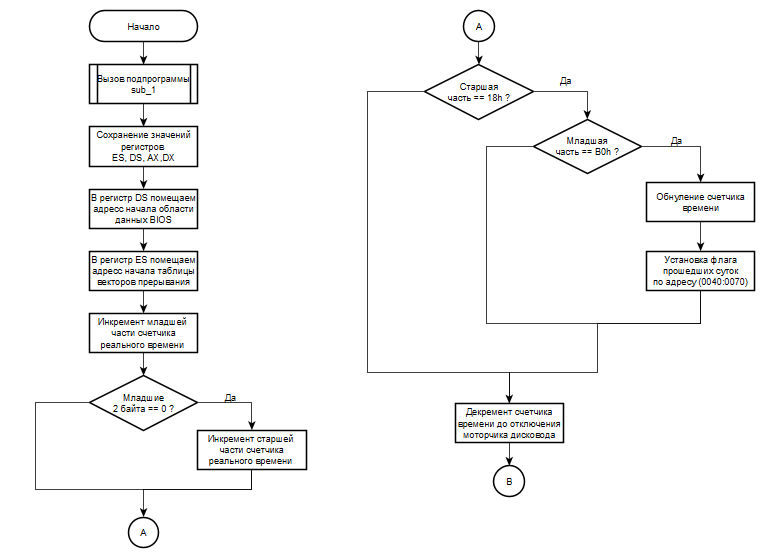
****

Рисунок 2. Int 8h

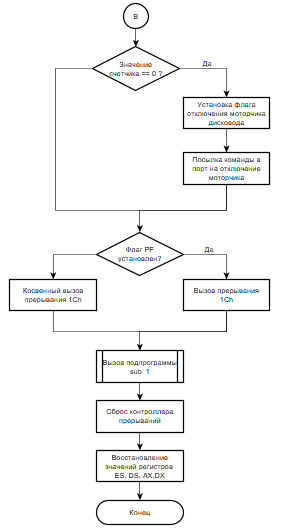
****

Рисунок 3. Int 8h

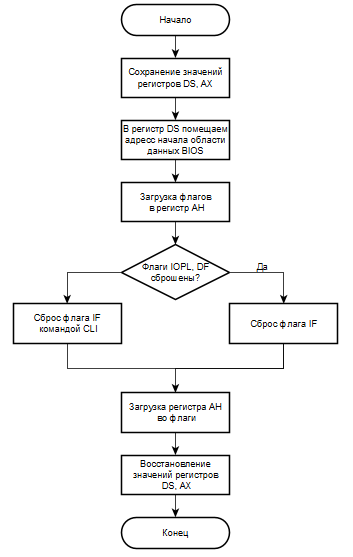
****

Рисунок 4. Подпрограмма sub\_1

**Функции обработчика прерывания int 8h:**

* Инкремент счетчика времени
* Вызов пользовательского прерывания 1Ch
* Декремент счетчика времени до отключения моторчика дисковода

# Заключение:

В ходе данной лабораторной работы были приобретены навыки получения адреса начала прерывания и листинг прерывания с помощью дизассемблера. Были изучены алгоритмы работы прерывания int 8h. Данное прерывание отвечает за изменение счетчика системного времени, управление контроллером дисковода, а также является способом вызова пользовательского прерывания.