|  |  |
| --- | --- |
| 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 «Программное обеспечение ЭВМ и информационные технологии»

**Лабораторная работа № 1 (часть 1)**

**По курсу «Операционные Системы»**

|  |  |
| --- | --- |
| **Тема**  Дизассемблирование INT 8h  **Студент**  Зайцева А. А.  **Группа** ИУ7-52Б  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель** Рязанова Н. Ю. |  |

Москва

2021 г.

# Цель 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, который эмулирует реальный режим работы вычислительной системы на базе процессоров Intel.

# Задание

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

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

# Листинги кода

Листинг 1 - Обработчик прерывания INT 8h

; Вызвать подпрограмму sub\_2 (запрет прерываний)

020A:0746 E8 0070 call sub\_2 ; (07B9)

; Сохранить значения регистров ES, DS, AX, DX в стеке

; (Сохранить аппаратный контекст)

020A:0749 06 push es

020A:074A 1E push ds

020A:074B 50 push ax

020A:074C 52 push dx

; Загрузить в DS адрес 0040h (начало области данных BIOS)

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

020A:0750 8E D8 mov ds,ax

; Обнулить AX, ES (по адресу 0 располагается таблица векторов прерываний IDT)

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

020A:0754 8E C0 mov es,ax

;

; ПЕРВОЕ ДЕЙСТВИЕ, выполняемое стандартным обработчиком прерывания таймера.

; Увеличение на единицу текущего значения 4-байтовой переменной, располагающейся

; по адресу 0000:046Ch – счётчика таймера. Если счетчик переполнится из-за того,

; что прошло более 24 часов с момента запуска таймера, в ячейку 0000:0470h

; заносится значение 1.

;

; Инкрементировать первые 2 байта (младшие) счетчика таймера

020A:0756 FF 06 006C inc word ptr ds:[6Ch] ; (0040:006C=8E8Bh)

; Если ZF==0 (счетчик не переполнился), то перейти на loc\_1,

; пропустив инкрементирование часов

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

; Если же ZF==1 (счетчик переполнится), значит прошел очередной час с момента

; запуска таймера:

; максимальное значение, которое могут закодировать 2 байта: 2^16-1=65535

; 65535/18,2 (примерное количество прерываний в секунду)≈3600 секунд = 1 час

; Поэтому инкрементировать часы 0040:006Eh (старшие 2 байта счетчика таймера)

020A:075C FF 06 006E inc word ptr ds:[6Eh] ; (0040:006E=14h)

020A:0760 loc\_1:

; Проверить, прошли ли очередные сутки с момента запуска таймера:

; (количество часов (0040:006Eh)==24=18h) и ((0040:006Сh)==0B0h=176).

; Если хотя бы одно из условий не выполняется, перейти на loc\_2, пропустив

; обработку ситуации, когда прошли очередные сутки с момента запуска таймера

020A:0760 83 3E 006E 18 cmp word ptr ds:[6Eh],18h ; (0040:006E=14h)

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=8E8Bh)

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

; Прошли очередные сутки с момента запуска таймера, следовательно, нужно:

; 1) обнулить счетчик таймера

020A:076F A3 006E mov word ptr ds:[6Eh],ax ; (0040:006E=14h)

020A:0772 A3 006C mov word ptr ds:[6Ch],ax ; (0040:006C=8E8Bh)

; 2) занести единицу в ячейку 0040:0070h

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

; 3) Загрузить 8 в AX (AX до этого был равен нулю)

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

;

; ВТОРОЕ ДЕЙСТВИЕ, выполняемое стандартным обработчиком прерывания таймера.

; Контроль за работой двигателей НГМД (накопителей на гибких магнитных дисках):

; если после последнего обращения к НГМД прошло > 2 секунд, выключить двигатель

;

020A:077C loc\_2:

; Сохранить регистр AX в стеке

020A:077C 50 push ax

; Декрементировать время, оставшееся до выключения двигателя

; (расположено в ячейке с адресом 0040:0040h)

020A:077D FE 0E 0040 dec byte ptr ds:[40h] ; (0040:0040=78h)

; Если ZF==1 (результат декрементирования не равен нулю), значит еще не прошло 2

; секунды после последнего обращения к НГМД.

; Перейти на loc\_3, пропустив обработку обратной ситуации

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

; Если же ZF==0, значит после последнего обращения к НГМД прошло 2 секунды.

; Следовательно, нужно отправить команду выключения моторчика дисковода:

; 1) Сбросить соответствующие флаги отключения моторчика (младшие 4 бита)

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

; 2) Поместить в AL (данные для вывода в последующей команде out) значение

; 0Ch=00001100b: 2 бит=1 (разрешение работы контроллера), 3 бит=1 (разрешение

; прерываний и прямого доступа к памяти), 4−7 биты сброшены (значения 1 в каждом

; разряде вызывали бы включение соответствующего двигателя НГМД)

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

; 3) Поместить в DX (примемник в последующей команде out) номер порта 3F2

; (порт цифрового управления)

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

; 4) Вывести данные в порт

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

;

; ТРЕТЬЕ ДЕЙСТВИЕ, выполняемое стандартным обработчиком прерывания таймера.

; Вызов прерывания INT 1Ch (вызывается до сброса контроллера прерывания, поэтому

; во время его выполнения все аппаратные прерывания запрещены (в том числе –

; прерывания от клавиатуры). После инициализации системы вектор INT 1Ch указывает ; на команду IRET, то есть обработчик прерывания INT 1Ch ничего не делает.

;

020A:078E loc\_3:

; Восстановить AX из стека

020A:078E 58 pop ax

; Проверить флаг четноси PF(0100 − 2 бит в области BIOS по адресу

; 0040:0314h, где находится копия флагов, отвечает за флаг PF)

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

; Если он установлен, перейти на loc\_4 для стандартного вызова прерывания 1CH

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

; Иначе будет осуществлен косвенный вызов прерывания 1CH с другими флагами

; Загрузить младший байт FLAGS в регистр AH

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

; Обменять AH (младший байт FLAGS) и AL (8) -> AX=[08][младший байт FLAGS]

020A:0798 86 E0 xchg ah,al

; Сохранить AX в стеке

020A:079A 50 push ax

; Косвенный вызов прерывания 1Ch с помощью адреса в таблице векторов прерываний

; (1Ch\*4=28\*4=112=70H)

; При вызове командой int регистр FLAGS был бы загружен в стек, а в данном случае

; на его месте лежит AX. Тогда по выходу из прерывания 1CH именно AX будет

; установлен в FLAGS командой iret. Отличие AX и FLAGS – в старшем байте

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

; Перейти на loc\_5, чтобы повторно не вызывать прерывание 1CH

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

020A:07A2 90 nop

; Вызвать прерывание 1CH

020A:07A3 loc\_4:

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

; Вызвать подпрограмму sub\_2 (запрет прерываний)

020A:07A5 loc\_5:

020A:07A5 E8 0011 call sub\_2 ; (07B9)

; Сбросить контроллер прерываний

; (Чтобы позволить прерываниям меньшего приоритета обрабатываться)

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

; Восстановить регистры DX, AX, DS, ES

020A:07AC 5A pop dx

020A:07AD 58 pop ax

020A:07AE 1F pop ds

020A:07AF 07 pop es

; Перейти в сторону выхода (020A:07B0 - 164h = 020A:064C)

020A:07B0 E9 FE99 jmp $-164h

;…

; Сохраненить DS, AX

020A:064C 1E push ds

020A:064D 50 push ax

;…

; Восстановить DS, AX

020A:06AA 58 pop ax

020A:06AB 1F pop ds

; Выйти из прерывания

020A:06AC CF iret ; Interrupt return

Листинг 2 – Подпрограмма sub\_2

sub\_2 proc near

; Сохранить значения регистров DS, AX

020A:07B9 1E push ds

020A:07BA 50 push ax

; Загрузить в DS адрес 0040h (начало области данных BIOS)

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

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

; Загрузить младший байт FLAGS в регистр AH

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

; Проверить: поднят ли хотя бы один из флагов 10 или 13

; (2400h = 0010 0100 0000 0000b) в области BIOS по адресу 0040:0314h, где

; находится копия флагов?

; 10 − DF − Флаг направления, контролирует поведение команд обработки строк: 1 - ; в сторону уменьшения адресов, 0 – наоборот

; 12 и 13 − IOPL − Уровень приоритета ввода/вывода.

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

; Если поднят хотя бы один, то перейтит на loc\_22, чтобы командой cli сбросить

; флаг разрешения прерываний IF.

; Процессор перестанет обрабатывать прерывания от внешних устройств

; (только маскируемые, так как они вызываются по маске. Немаскируемые запретить

; нельзя, например, различные ошибки)

020A:07C7 75 0C jnz loc\_22 ; Jump if not zero

; Если оба сброшены, то сбросить IF (9 бит) командой and.

; Операция and объёмная, 2 раза обращается к памяти: считывает значение по адресу

; 0040:0314, затем изменяет его и еще раз обращаяется к памяти на запись.

; Необходимо, чтобы в промежуток, когда выполняется сама логическая операция,

; никто не обращался к этому участку памяти, для чего используется префиксная

; команда lock. Будет заблокирована шина данных, и, если в системе присутствует

; другой процессор, он не сможет обращаться к памяти, пока не закончится

; выполнение and

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

020A:07D0 loc\_21:

; Восстановить значения флагов SF, ZF, AF, PF и CF регистра FLAGS из AH

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

; Восстановить значения регистров AX, DS

020A:07D1 58 pop ax

020A:07D2 1F pop ds

; Пропустить сброс IF командой cli

020A:07D3 EB 03 jmp short loc\_23 ; (07D8)

020A:07D5 loc\_22:

; Сбросить IF с помощью cli

020A:07D5 FA cli ; Disable interrupts

020A:07D6 EB F8 jmp short loc\_21 ; (07D0)

; Выйти из подпрограммы

020A:07D8 loc\_23:

020A:07D8 C3 retn

sub\_2 endp

# Схемы алгоритмов

# 

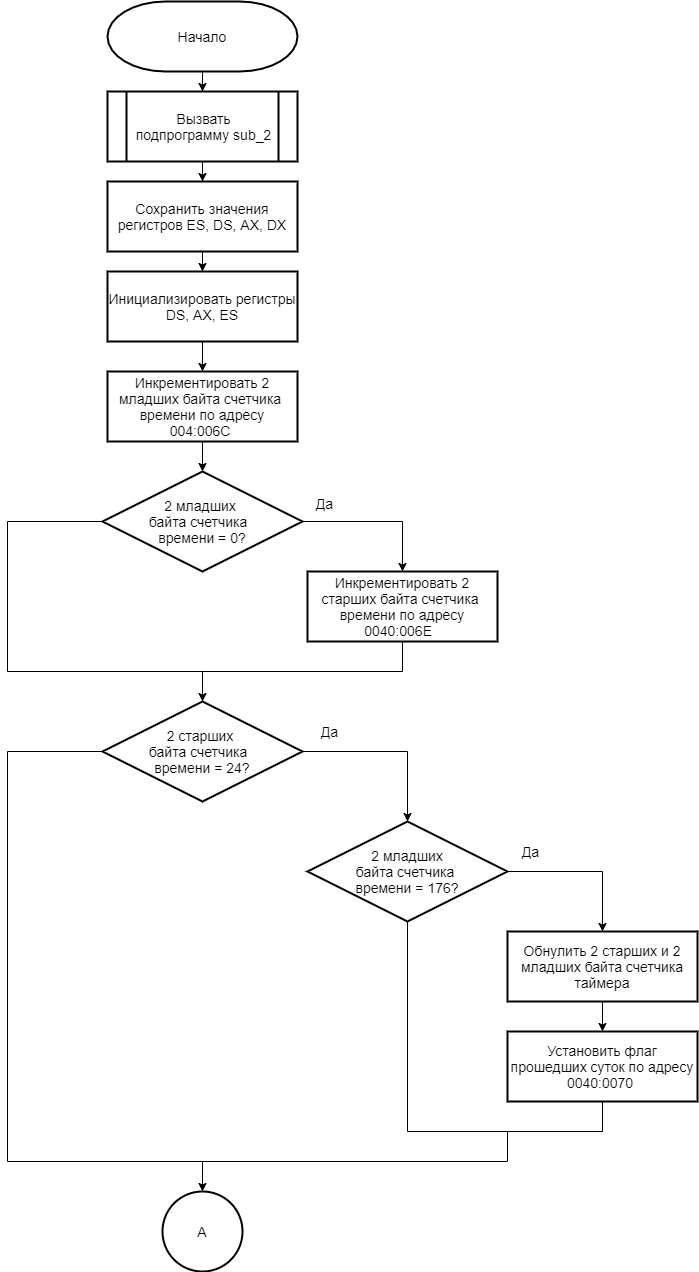


Рисунок 1 – Схема обработчика прерывания INT 8h (начало)

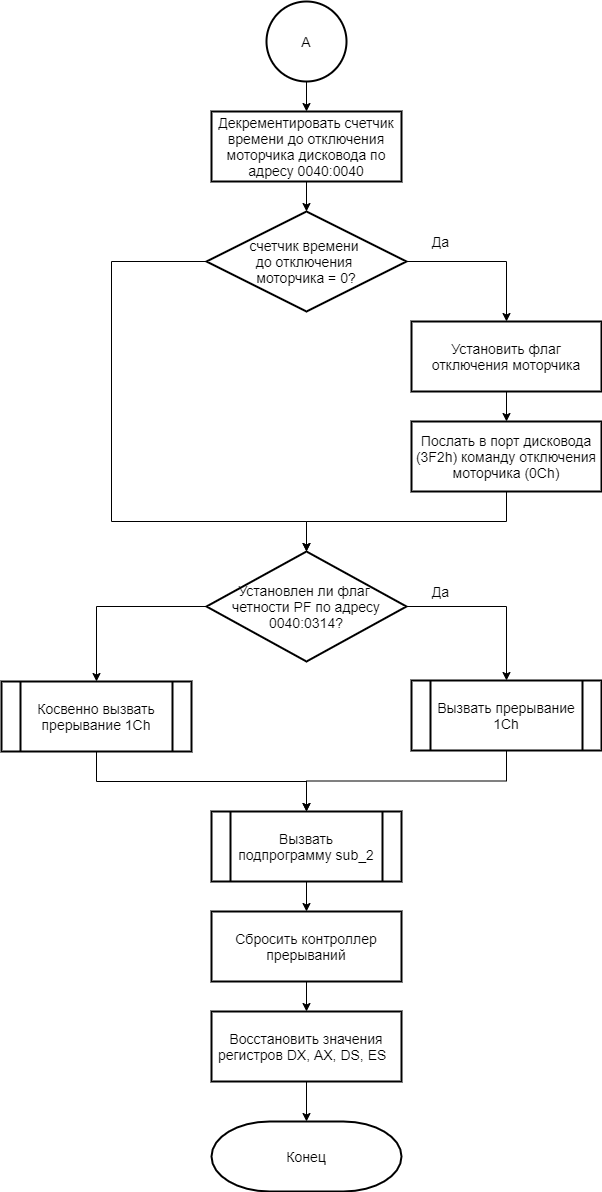


Рисунок 2 – Схема обработчика прерывания INT 8h (конец)

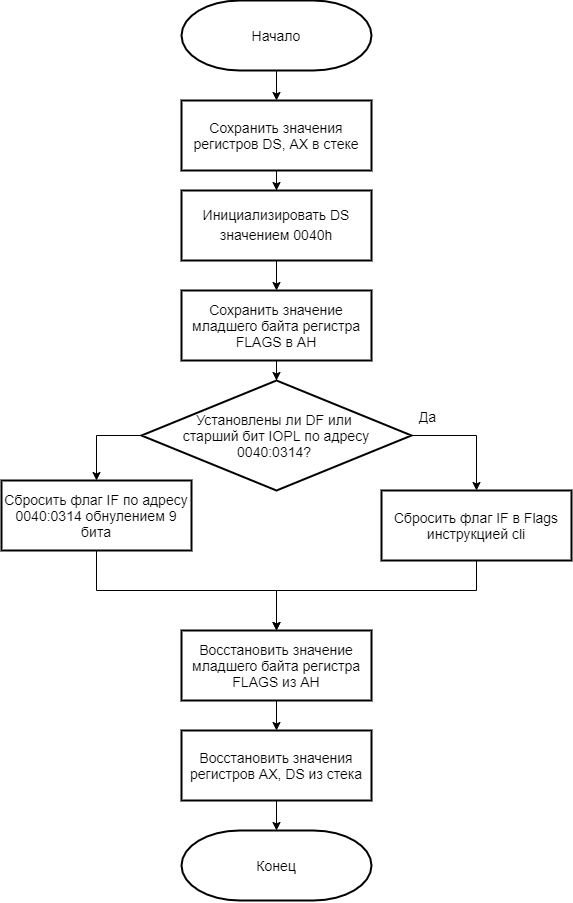


Рисунок 3 – Схема подпрограммы sub\_2

# Вывод

В ходе выполнения лабораторной работы с помощью средства дизассемблирования sourcer (sr.exe) был получен код обработчика аппаратного прерывания от системного таймера Int 8h в virtual mode.

На основе полученного кода был составлен алгоритм работы обработчика Int 8h и изучены его функции:

* увеличение на единицу текущего значения 4-байтовой переменной, располагающейся по адресу 0000:046Ch – счётчика таймера; при его переполнении (прошло более 24 часов) - занесение значения 1 по адресу 0000:0470h;
* уменьшение времени, оставшегося до выключения двигателя накопителя на гибких магнитных дисках (НГМД); если после последнего обращения к НГМД прошло более 2 секунд - выключение моторчика дисковода;
* вызов прерывания 1Ch.