|  |  |
| --- | --- |
| 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 (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | **1** |

**Название:**

Дизассемблирование int 8h

**Дисциплина:** Операционные системы

|  |  |  |  |
| --- | --- | --- | --- |
| Студент | ИУ7-56Б |  | А.А. Петрова |
|  | (Группа) |  | (И.О. Фамилия) |
|  |  |  |  | |  |
| Преподаватель |  |  | Н.Ю. Рязанова |
|  |  |  | (И.О. Фамилия) |

Москва, 2021

**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 sourcer (sr.exe) получить дизассемблерный код обработчика аппаратного прерывания от системного таймера int 8h.

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

**Листинг кода**

; вызов подпрограммы

020A**:**0746 E8 0070 **call** sub\_0004 ; (07B9)

; сохранение аппаратного контекста

020A**:**0749 06 **push** **es**

020A**:**074A 1E **push** **ds**

020A**:**074B 50 **push** **ax**

020A**:**074C 52 **push** **dx**

; ds = 0040h

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

020A**:**0750 8E D8 **mov** **ds,ax**

; es = 0

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

020A**:**0754 8E C0 **mov** **es,ax**

; 0040:006C - младшие 2 байта счётчика реального времени

020A**:**0756 FF 06 006C **inc** **word** ptr **ds:[**6Ch**]** ; (0040:006C == 4CB4h)

020A**:**075A 75 04 **jnz** loc\_0003 ; Jump if not zero

; 0040:006E - старшие 2 байта счётчика реального времени

020A**:**075C FF 06 006E **inc** **word** ptr **ds:[**6Eh**]** ; (0040:006E == 15h)

020A**:**0760 loc\_0003**:**

; проверка на то, прошло ли 24 часа с момента запуска таймера (18h == 24)

; 0B0h == 176 - кол-во вызовов таймера

020A**:**0760 83 3E 006E 18 **cmp** **word** ptr **ds:[**6Eh**],**18h ; (0040:006E == 15h)

020A**:**0765 75 15 **jne** loc\_0004 ; Jump if not equal

020A**:**0767 81 3E 006C 00B0 **cmp** **word** ptr **ds:[**6Ch**],**0B0h ; (0040:006C == 4CB4h)

020A**:**076D 75 0D **jne** loc\_0004 ; Jump if not equal

; обнуление счётчика реального времени и занесение 1 в 0040:0070, если прошло 24 часа

020A**:**076F A3 006E **mov** **word** ptr **ds:[**6Eh**],ax** ; (0040:006E == 15h)

020A**:**0772 A3 006C **mov** **word** ptr **ds:[**6Ch**],ax** ; (0040:006C == 4CB4h)

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

; al = 8 (т. к. до этого al == 0)

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

020A**:**077C loc\_0004**:**

020A**:**077C 50 **push** **ax**

; 0040:0040 - счётчик времени до выключения моторчика дисковода

020A**:**077D FE 0E 0040 **dec** **byte** ptr **ds:[**40h**]** ; (0040:0040=69h)

020A**:**0781 75 0B **jnz** loc\_0005 ; Jump if not zero

; установка флага отключения моторчика дисковода

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

; посылка команды отключения 0Ch в порт дисковода 3F2h

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

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

020A**:**078D EE **out** **dx,al**

020A**:**078E loc\_0005**:**

020A**:**078E 58 **pop** **ax**

; проверка, установлен ли флаг чётности (PF) в 0040:0314 (4 == 0100b; PF - во 2 бите)

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

020A**:**0795 75 0C **jnz** loc\_0006 ; Jump if not zero

; загузка младшего байта FLAGS в ah

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

; ah = 8, в al - младший байт FLAGS

020A**:**0798 86 E0 **xchg** **ah,al**

020A**:**079A 50 **push** **ax**

; 0000:0070 - адрес вектора прерывания 1Сh (1Ch \* 4 == 70h)

; при вызове через call не будет пушиться регистр флагов и при возврате через iret запушенный ax установится во FLAGS

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

020A**:**07A0 EB 03 **jmp** short loc\_0007 ; (07A5)

020A**:**07A2 90 **nop**

020A**:**07A3 loc\_0006**:**

020A**:**07A3 CD 1C **int** 1Ch

020A**:**07A5 loc\_0007**:**

020A**:**07A5 E8 0011 **call** sub\_0004 ; (07B9)

; сброс контроллера прерываний

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

020A**:**07AA E6 20 **out** 20h**,al** ; ' '

; восстановление аппаратного контекста

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

; ...

020A**:**064C 1E **push** **ds**

020A**:**064D 50 **push** **ax**

; ...

020A**:**06AA 58 **pop** **ax**

020A**:**06AB 1F **pop** **ds**

; возврат из прерывания

020A**:**06AC CF **iret**

sub\_0004 proc **near**

; сохранение регистров

020A**:**07B9 1E **push** **ds**

020A**:**07BA 50 **push** **ax**

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

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

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

; проверка, установлен ли хотя бы один из флагов IOPL (I/O Privilege Level) (12-13 биты) или DF (Direction Flag) (10 бит) (2400h == 0010 0100 0000 0000b)

; если что-то из этого установлено, то флаг разрешения прерываний (IF) сбрасывается через cli

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

020A**:**07C7 75 0C **jnz** loc\_0009 ; Jump if not zero

; сброс флага разрешения прерываний (IF) (9-й бит) (запрет внешних маскируемых прерываний, чтобы гарантировать выполнение участка программы)

; lock запрещает доступ к шине другим процессам на время выполнения этой команды

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

020A**:**07D0 loc\_0008**:**

; загрузка ah в младший байт регистра флагов

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

; восстановление регистров

020A**:**07D1 58 **pop** **ax**

020A**:**07D2 1F **pop** **ds**

020A**:**07D3 EB 03 **jmp** short loc\_0010 ; (07D8)

020A**:**07D5 loc\_0009**:**

; сброс флага разрешения прерываний через cli

020A**:**07D5 FA **cli** ; Disable interrupts

020A**:**07D6 EB F8 **jmp** short loc\_0008 ; (07D0)

020A**:**07D8 loc\_0010**:**

; возврат из подпрограммы

020A**:**07D8 C3 **retn**

sub\_0004 endp

**Схема алгоритма**

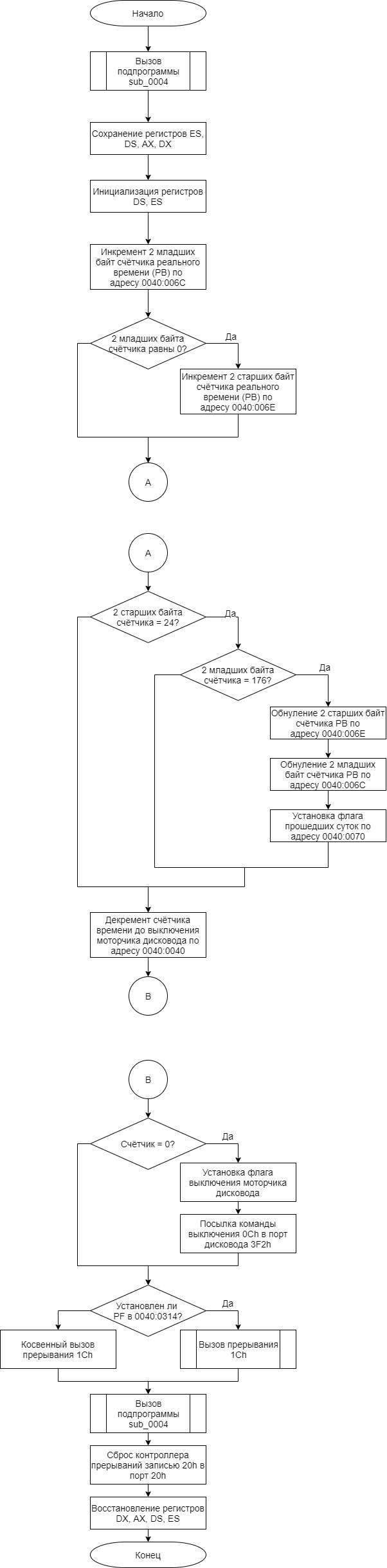


Рисунок 1 – схема алгоритма работы обработчика прерываний int 8h

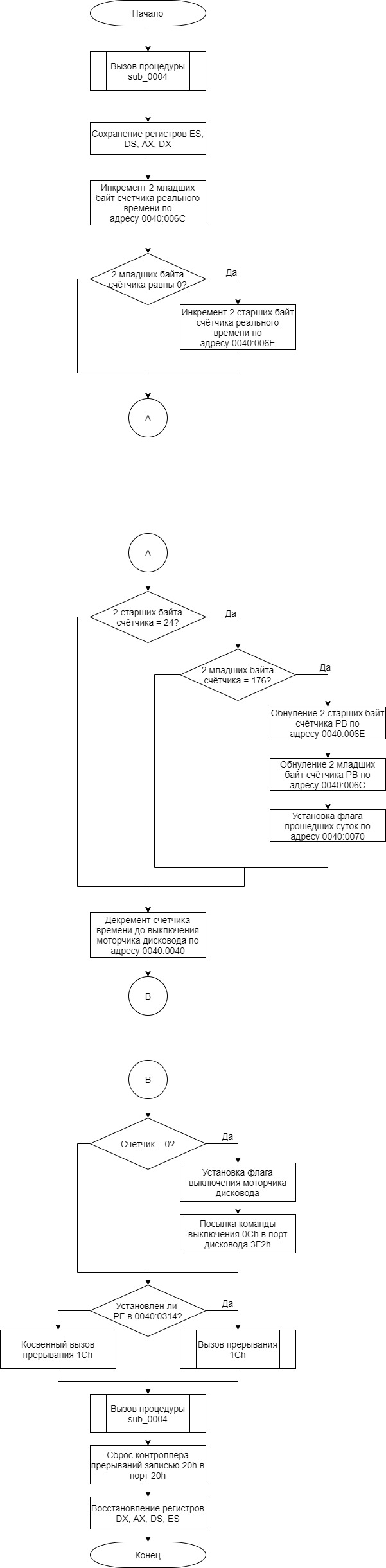


Рисунок 2 – схема алгоритма работы обработчика прерываний int 8h

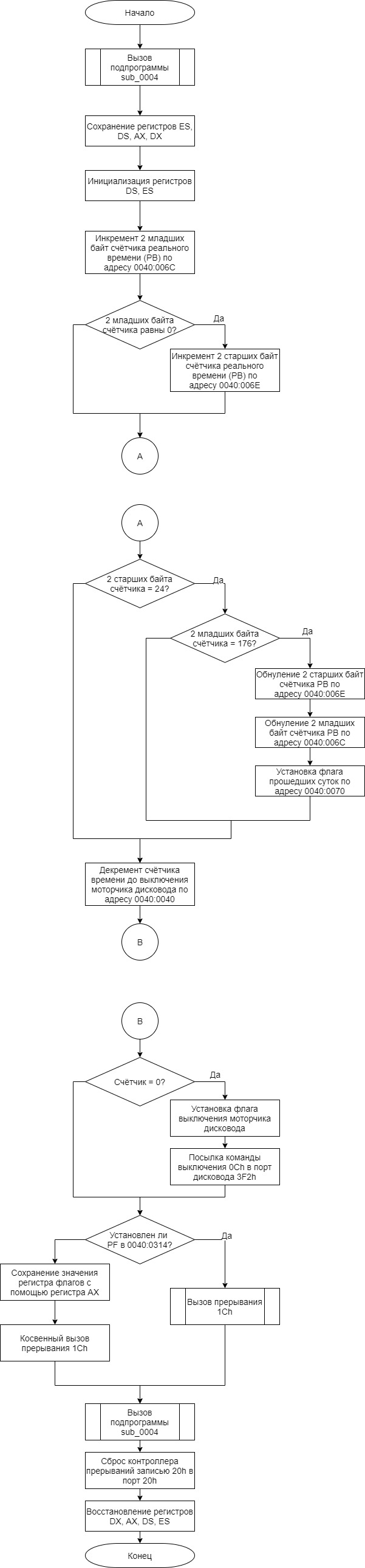


Рисунок 3 – схема алгоритма работы обработчика прерываний int 8h

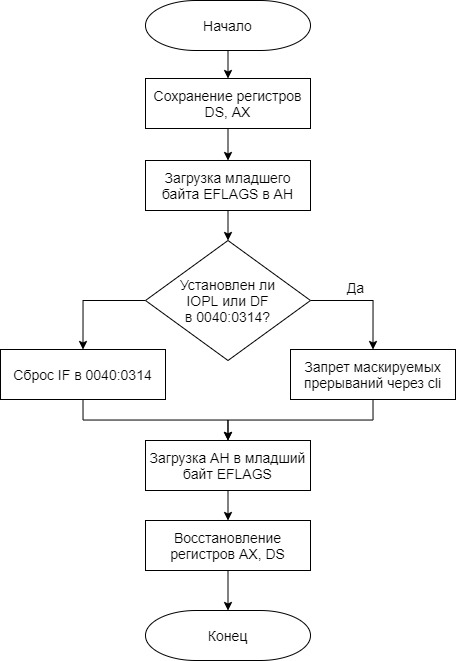


Рисунок 4 – схема алгоритма работы подпрограммы sub\_0004