|  |  |
| --- | --- |
|  | **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

***Отчет к лабораторной работе №1 (часть 1)***

*по курсу «Операционные системы»*

*по теме «Дизассемблирование int 8h»*

Студент: Варламова Екатерина

Группа: ИУ7-51Б

Преподаватель: Рязанова Н. Ю.

*2021 г.*

1. **Полученный ассемблерный код прерывания int 8h с адресами команд и комментариями**

**вызов подпрограммы**

**020C:0746 E8 0070 call sub\_4**

**сохранение значений регистров es, ds, ax, dx**

**020C:0749 06 push es**

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

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

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

**загрузка адреса области данных BIOS в сегментный регистр DS**

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

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

**загрузка адреса таблицы векторов прерываний в сегментный регистр ES**

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

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

**инкремент младшей части счётчика времени**

**020C:0756 FF 06 006C inc data\_17 ; (0040:006C=0BEDEh)**

**020C:075A 75 04 jnz loc\_3 ; Jump if not zero**

**если младшая часть счётчика времени нулевая, то увеличивается старшая часть счётчика времени**

**020C:075C FF 06 006E inc data\_18; (0040:006E=0)**

**020C:0760 loc\_3:**

**020C:0760 83 3E 006E 18 cmp data\_18,18h ; (0040:006E=0)**

**020C:0765 75 15 jne loc\_4 ; Jump if not equal**

**020C:0767 81 3E 006C 00B0 cmp data\_17,0B0h; (0040:006C=0BEDEh)**

**020C:076D 75 0D jne loc\_4 ; Jump if not equal**

**если старшая часть счётчика времени равна 24, а младшая часть = 176, то обе части обнуляются, а в ячейку 0000:0470h заносится значение 1 (прерывание вызывается 1193180/65536 раз в секунду, значит 1193180/65536 \* 3600 = 65543 раз в час, а счётчик тиков переполняется каждые 2^16 = 65536 тиков, следовательно, пройдут сутки, когда старшая часть счётчика времени будет равна 24, а младшая часть (1193180/65536 \* 3600 – 65536) \* 24 = 176)**

**020C:076F A3 006E mov data\_18,ax ; (0040:006E=0)**

**020C:0772 A3 006C mov data\_17,ax ; (0040:006C=0BEDEh)**

**020C:0775 C6 06 0070 01 mov data\_19,1 ; (0040:0070=0)**

**сохранение значения ax**

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

**020C:077C loc\_4:**

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

**уменьшение счётчика времени до остановки двигателей НГМД**

**020C:077D FE 0E 0040 dec data\_16 ; (0040:0040=24h)**

**020C:0781 75 0B jnz loc\_5 ; Jump if not zero**

**если счётчик времени нулевой, то:**

**- установка флага отключения моторчика дисковода**

**020C:0783 80 26 003F F0 and data\_15,0F0h; (0040:003F=0)**

**- посылка сигнала на выключение двигателей (назначение битов порта 3F2h: 0Ch = 0 0 0 0 1 1 0 0 =>**

**с помощью 0-1 битов выбирается дисковод;**

**во 2 бите единица => разрешается работа контроллера НГМД;**

**в 3 бите единица => разрешаются прерывания от контроллера;**

**в 4-7 битах нули => выключаются все двигатели дисковода)**

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

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

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

**020C:078E loc\_5:**

**восстановление сохранённого значения ax**

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

**проверка PF**

**020C:078F F7 06 0314 0004 test data\_20,4 ; (0040:0314=3200h)**

**если PF = 0, вызов 1Ch по адресу обработчика**

**020C:0795 75 0C jnz loc\_6 ; Jump if not zero**

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

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

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

**020C:079B 26:FF 1E 0070 call es:data\_5 ; (0000:0070=6ADh)**

**020C:07A0 EB 03 jmp short loc\_7**

**020C:07A2 90 nop**

**020C:07A3 loc\_6:**

**иначе вызов прерывания**

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

**020C:07A5 loc\_7:**

**сброс контроллера прерываний**

**020C:07A5 E8 0011 call sub\_4**

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

**020C:07AA E6 20 out 20h,al; port 20h, 8259-1 int command; al = 20h, end of interrupt**

**восстановление сохранённых значений регистров**

**020C:07AC 5A pop dx**

**020C:07AD 58 pop ax**

**020C:07AE 1F pop ds**

**020C:07AF 07 pop es**

**Выход из обработчика**

**020C:06AC CF iret**

1. **Полученный ассемблерный код подпрограммы с адресами команд и комментариями**

**sub\_4 proc near**

**сохранение значений регистров ax и ds**

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

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

**Настройка сегментного регистра**

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

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

**Загрузка младшего байта flags в ah**

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

**Проверка DF и старшего бита IOPL регистра флагов**

**020C:07C1 F7 06 0314 2400 test data\_20,2400h; (0040:0314=3200h)**

**020C:07C7 75 0C jnz loc\_9 ; Jump if not zero**

**Если Df и IOPL нулевые, то блокировка системной шины и обнуление IF**

**020C:07C9 F0 lock ; Lock the bus**

**020C:07CA 81 26 0314 FDFF and data\_20,0FDFFh ; (0040:0314=3200h)**

**020C:07D0 loc\_8:**

**Восстановление flags из ah**

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

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

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

**020C:07D3 EB 03 jmp short loc\_10**

**020C:07D5 loc\_9:**

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

**020C:07D6 EB F8 jmp short loc\_8**

**020C:07D8 loc\_10:**

**020C:07D8 C3 ret**

**sub\_4 endp**

1. **Схема алгоритма прерывания int 8h**

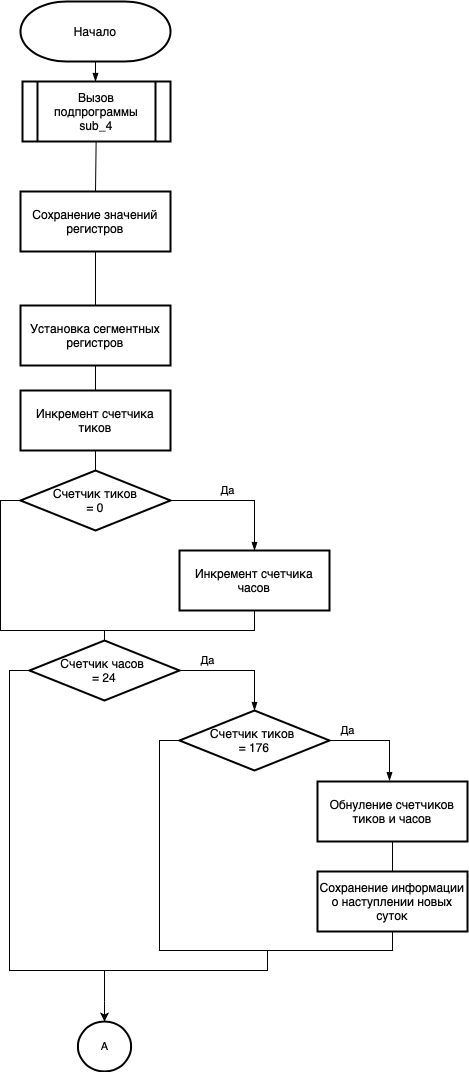
****

Рисунок 1. Схема алгоритма прерывания int 8h (часть 1)

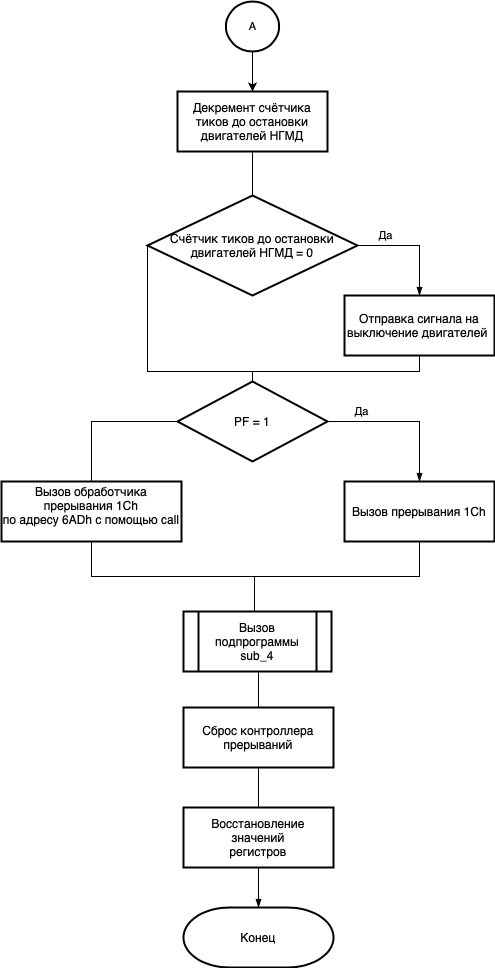
****

Рисунок 2 Схема алгоритма прерывания int 8h (часть 2)

1. **Схема алгоритма подпрограммы**

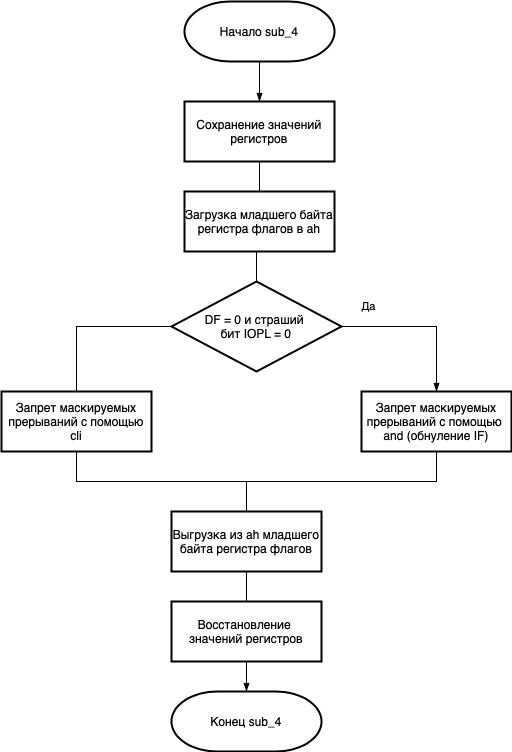
****

Рисунок 3 Схема алгоритма подпрограммы sub\_4