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 по лабораторной работе №1 по дисциплине ”Операционные системы” по теме ”Дизассемблирование INT 8h”

Студент Миронов Г.А.

Группа ИУ7-53Б

Оценка (баллы) _____

Преподаватели Рязанова Н.Ю.

Москва
2021 г.

1 Полученный дизассемблированный код

1.1 Листинг INT8h

```
; The following equates show data references outside the range of the program.

    = 0070          data_1e           equ 70h          ; (0000:0070=0ADh)
    = 003F          dsk_motor_stat    equ 3Fh          ; (0040:003F=0)
    = 0040          dsk_motor_tmr     equ 40h          ; (0040:0040=0D7h)
    = 006C          timer_low         equ 6Ch          ; (0040:006C=0C5EFh)
    = 006E          timer_hi          equ 6Eh          ; (0040:006E=9)
    = 0070          timer_rolled      equ 70h          ; (0040:0070=0)
    = 0314          data_2e           equ 314h         ;*(0040:0314=3200h)

                    ;————————————————————————————————— seg_a ———

                    seg_a       segment byte public
                                assume cs:seg_a , ds:seg_a


                            org 746h

                                                    ;* No entry point to code
; —— sub_1 call
020A:0746  E8 0070        ;*     call    sub_1          ;*(07B9)
020A:0746  E8 70 00              db   0E8h, 70h, 00h
; —— save es, ds, ax, dx
020A:0749  06                    push    es
020A:074A  1E                    push    ds
020A:074B  50                    push    ax
020A:074C  52                    push    dx
; —— Load 0040H to DS
020A:074D  B8 0040               mov ax,40h
020A:0750  8E D8                 mov ds,ax
; —— AX = ES = 0
020A:0752  33 C0                 xor ax,ax    ; Zero register
020A:0754  8E C0                 mov es,ax
```

```
; -- increment timer counter, addr = 0040:006C
020A:0756  FF 06 006C          inc word ptr ds:timer_low    ; (0040:006C=0C5EFh)
020A:075A  75 04               jnz loc_1    ; Jump if not zero
; -- increment high timer part (hours)
020A:075C  FF 06 006E          inc word ptr ds:timer_hi     ; (0040:006E=9)
; -- check if it's 24 hours from start already:
;    0040H:006EH == 18H   0040H:006CH == 00B0H
;    24 * 60 * 60 * t == 18H << 16 + B0H, t -
020A:0760                loc_1:                 ;  xref 020A:075A
020A:0760  83 3E 006E 18       cmp word ptr ds:timer_hi,18h     ; (0040:006E=9)
020A:0765  75 15               jne loc_2    ; Jump if not equal
020A:0767  81 3E 006C 00B0     cmp word ptr ds:timer_low,0B0h   ; (0040:006C=0C5EFh)
020A:076D  75 0D               jne loc_2    ; Jump if not equal
; -- Set timer counter to zero. Move 1 to 0040H:0070H (as it is 24 already)
020A:076F  A3 006E             mov ds:timer_hi,ax       ; (0040:006E=9)
020A:0772  A3 006C             mov ds:timer_low,ax      ; (0040:006C=0C5EFh)
020A:0775  C6 06 0070 01       mov byte ptr ds:timer_rolled,1   ; (0040:0070=0)
; -- AL = 8 as AL was equal to 0.
020A:077A  0C 08               or  al,8

020A:077C                loc_2:                 ;  xref 020A:0765, 076D
020A:077C  50                  push    ax
; -- decrement disk motor timer
020A:077D  FE 0E 0040          dec byte ptr ds:dsk_motor_tmr    ; (0040:0040=0D7h)
020A:0781  75 0B               jnz loc_3    ; Jump if not zero
; -- set flags to stop motor
020A:0783  80 26 003F F0       and byte ptr ds:dsk_motor_stat,0F0h ; (0040:003F=0)
; -- send command to stop motor
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
020A:078E                loc_3:                 ;  xref 020A:0781
020A:078E  58                  pop ax
; -- check Parity Flag
020A:078F  F7 06 0314 0004     test    word ptr ds:data_2e,4    ; (0040:0314=3200h)
020A:0795  75 0C               jnz loc_4    ; Jump if not zero
; -- move lower FLAGS to AH
020A:0797  9F                  lahf         ; Load ah from flags
; -- exchange ah   al. Now: AX = 08[AH], [AH] - lower FALGS
```

```
020A:0798  86 E0                xchg    ah,al
020A:079A  50                   push    ax
; -- Call 1CH via vector table
;    Need that because otherwise we would push FLAGS to stack
;    With this way we store AX instead
:    So AX well be used and returned as FLAGS after iret
020A:079B  26: FF 1E 0070       call    dword ptr es:data_1e    ; (0000:0070=6ADh)
020A:07A0  EB 03                jmp short loc_5       ; (07A5)
020A:07A2  90                   db  90h
020A:07A3            loc_4:                           ;  xref 020A:0795
020A:07A3  CD 1C                int 1Ch               ; Timer break (call each 18.2ms)
020A:07A5            loc_5:                           ;  xref 020A:07A0
020A:07A5  E8 0011        ;*    call    sub_1               ;*(07B9)
; -- Reset interrupt controller
020A:07A5  E8 11 00             db  0E8h, 11h, 00h
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
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:07B0h - 164h = 020A:064Ch

               seg_a       ends


; ...........

               ; The following equates show data references outside the range of the program.

    = 0314          data_1e     equ 314h    ;*(0040:0314=3200h)
    = 0314          data_2e     equ 314h    ;*(020A:0314=3231h)

               ;
                    seg_a   ----

               seg_a       segment byte public
                           assume cs:seg_a  , ds:seg_a


                      org 64Ch

                                            ;* No entry point to code
```

```
020A:064C   1E                      push     ds
020A:064D   50                      push     ax
;  . . . . . . . . . . .
020A:06AA   58                      pop ax
020A:06AB   1F                      pop ds
; —— Return from int 8h
020A:06AC   CF                      iret          ; Interrupt return

                   seg_a        ends
```

Листинг 1.1: Листинг INT 8h

1.2 Листинг процедуры sub_1

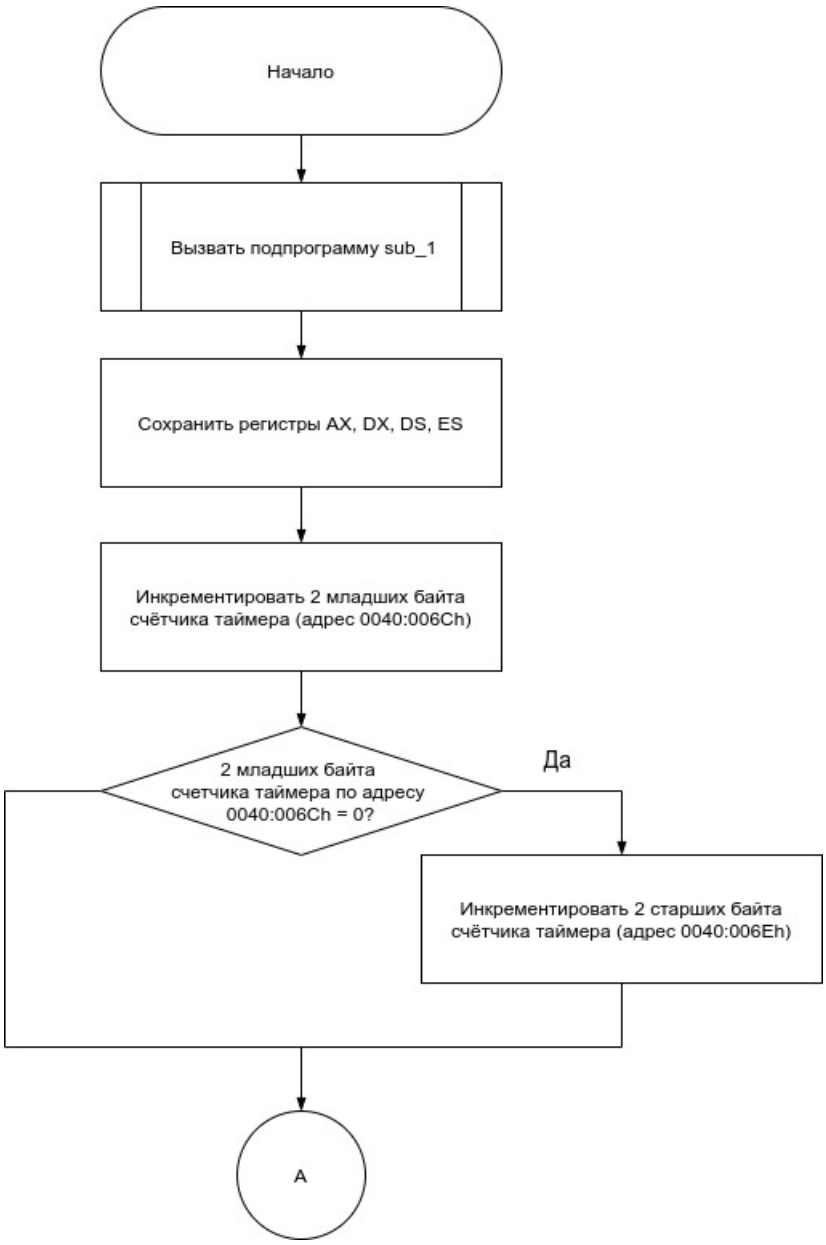
```
; The following equates show data references outside the range of the
    program.

     = 0314           data_1e      equ 314h     ;*(0040:0314=3200h)

                   ;
                        seg_a   ———

                   seg_a        segment byte public
                                assume cs:seg_a   , ds:seg_a


                           org 7B9h

                                                    ;* No entry point to code
;             ds,  ax
020A:07B9   1E                      push     ds
020A:07BA   50                      push     ax
; —— AX = DS = 0040H
020A:07BB   B8 0040                 mov ax,40h
020A:07BE   8E D8                   mov ds,ax
; ——               FLAGS   AH
020A:07C0   9F                      lahf          ; Load ah from flags
; ——         DF              IOPL.
;                       ,    IF            cli
020A:07C1   F7 06 0314 2400         test     word ptr ds:data_1e,2400h     ;
    (0040:0314=3200h)
020A:07C7   75 0C                   jnz loc_2     ; Jump if not zero
; ——      Interrput Enable Flag (9    —      ).
;      lock                                  .
020A:07C9   F0> 81 26 0314 FDFF    lock and word ptr ds:data_1e,0FDFFh  ;
    (0040:0314=3200h)
020A:07D0              loc_1:                   ;   xref 020A:07D6
```

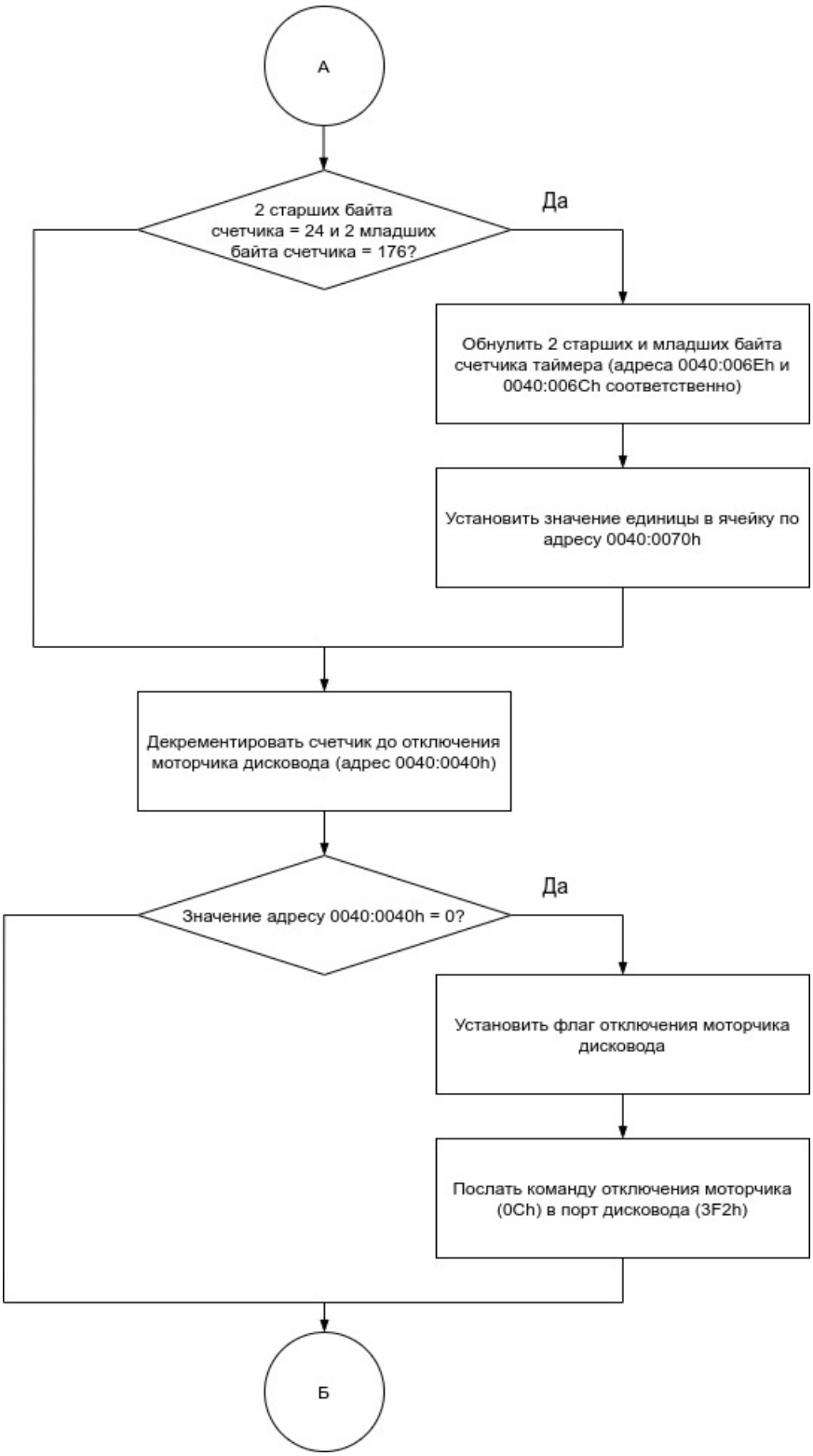
```
30 ; ——       AH            FLAGS.
31 020A:07D0  9E                    sahf        ; Store ah into flags
32 020A:07D1  58                    pop ax
33 020A:07D2  1F                    pop ds
34 020A:07D3  EB 03                 jmp short loc_ret_3     ; (07D8)
35 ; ——     Interrput Enable Flag       cli.
36 020A:07D5            loc_2:                  ;  xref 020A:07C7
37 020A:07D5  FA                    cli         ; Disable interrupts
38 020A:07D6  EB F8                 jmp short loc_1     ; (07D0)
39 ; ——                .
40 020A:07D8            loc_ret_3:              ;  xref 020A:07D3
41 020A:07D8  C3                    retn
42
43                 seg_a       ends
```

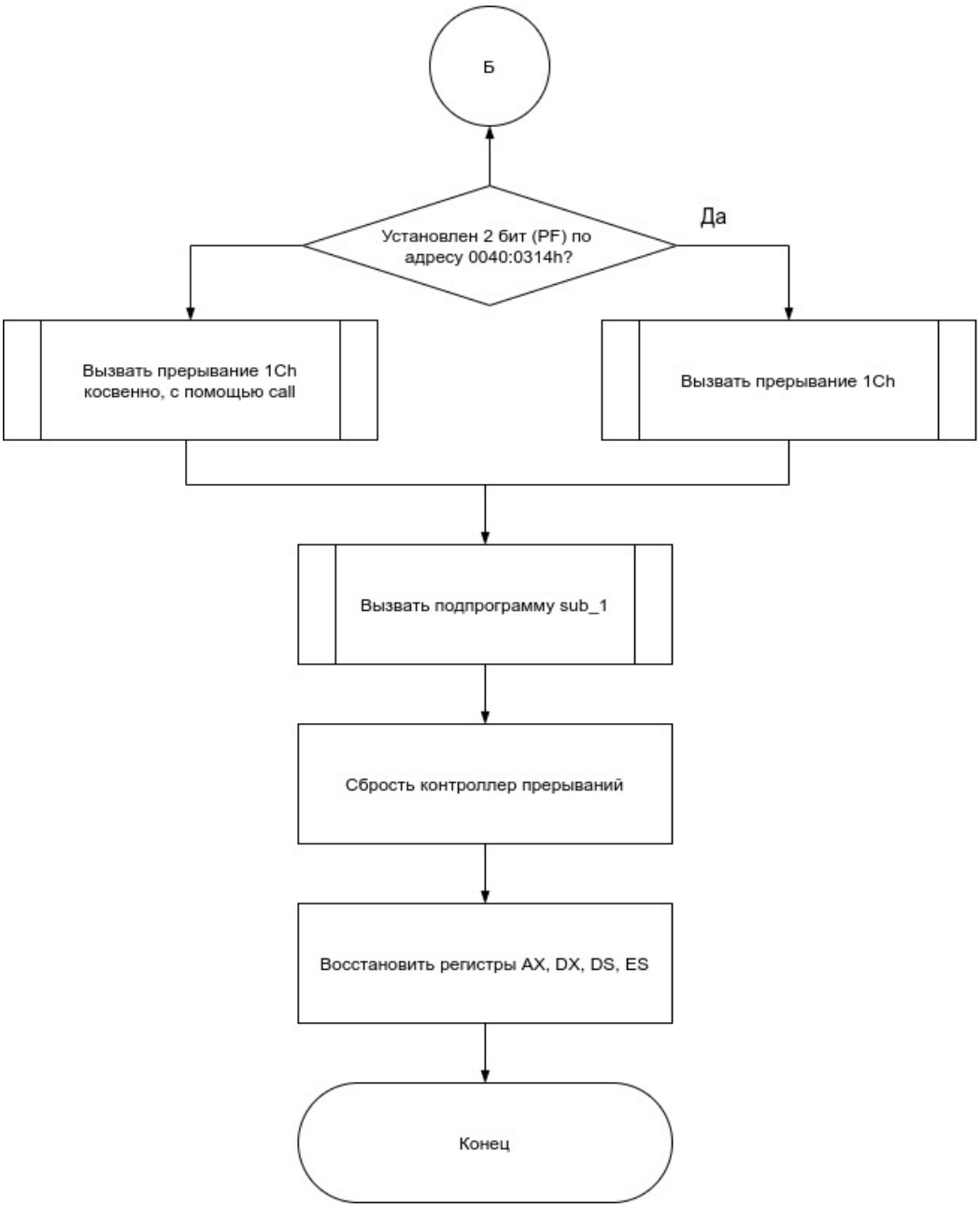
Листинг 1.2: Листинг процедуры sub_1

2 Схема алгоритмов

2.1 Схема алгоритма обработчика INT8h




А
2 старших байта счетчика = 24 и 2 младших байта счетчика = 176?
Да
Обнулить 2 старших и младших байта счетчика таймера (адреса 0040:006Eh и 0040:006Ch соответственно)
Установить значение единицы в ячейку по адресу 0040:0070h
Декрементировать счетчик до отключения моторчика дисковода (адрес 0040:0040h)
Значение адресу 0040:0040h = 0?
Да
Установить флаг отключения моторчика дисковода
Послать команду отключения моторчика (0Ch) в порт дисковода (3F2h)
Б


Б
Установлен 2 бит (PF) по адресу 0040:0314h?
Да
Вызвать прерывание 1Ch косвенно, с помощью call
Вызвать прерывание 1Ch
Вызвать подпрограмму sub_1
Сбрость контроллер прерываний
Восстановить регистры AX, DX, DS, ES
Конец

2.1.1 Схема алгоритма процедуры sub_1