МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5
по дисциплине «Организация систем и ЭВМ»
Тема «Написание собственного прерывания.»

Студент гр. 1303 _____ Кропотов Н.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Изучить особенности прерываний на языке Ассемблера, написать собственное прерывание.

Задание.

Вариант 2С.

Требуется заменить обработчик прерывания 60h на вывод времени с датой в формате YYYY:MM:DD hh:mm:ss

Выполнение работы.

Объявляются два двухбайтовых сегмента памяти SAVE_IP и SAVE_CS. Они используются для сохранения смещения до оригинального прерывания и позволяют в дальнейшем восстановить исходные вектора прерывания.

Описывается процедура SUBR_INT, которая является написанным пользовательским прерыванием. В данной процедуре в начале все регистры, которые будут изменены, для сохранения кладутся в стек, затем осуществляется взаимодействие с динамиком компьютера – выставляется частота звука, время звучания, сохраняется состояние порта, биты, отвечающие за доступ к динамику и его включение выставляются в 1.

После проигрывания звука порт возвращается в исходное состояние. Все сохраненные регистры вынимаются из стека, а также обеспечивается разрешение прерываний более низкого уровня во время действия данного.

В главной процедуре смещение и сегмент прерывания, которое требуется заменить сохраняются в объявленные сегменты памяти. Функция 35H прерывания 21H дает вектор прерывания, записанного в нижний байт регистра AX. Смещение и сегмент данного регистра записываются в регистры BX и ES, соответственно, они сохраняются в SAVE_IP и SAVE_CS. Далее записывается новое прерывание.

Функция 25H считывает смещение до него из DX и сегмент из DS и устанавливает его в вектор прерывания. Так как прерывание 08H вызывается

18 раз в секунду, то для того, чтобы наблюдать результат выполнения пользовательского прерывания, используется зацикливание, которое можно прервать нажатием клавиши Esc.

После выхода из цикла, исходный вектор прерывания восстанавливается, и программа завершается. Исходный код программы см. в приложении А.

Вывод.

В ходе выполнения лабораторной работы были изучены виды прерываний и работа с ними. В соответствии с задание было разработано собственное прерывание.

ПРИЛОЖЕНИЕ А

Тексты исходных файлов программ lab5.asm.

```
DATA SEGMENT
    KEEP_CS dw 0
    KEEP_IP dw 0
     MyString db 10, 13, 'TLOU$'
     OutNum dw 3
     EndMessage db 10, 13, 'End int$'
DATA ENDS


AStack SEGMENT STACK
    db 1024 DUP (?)
AStack ENDS


CODE SEGMENT

ASSUME CS:CODE, DS:DATA, SS:AStack

SUBR_INT PROC FAR
     push ax
     push cx
     push bx
     push dx

     mov        al,0Bh
    out         70h,al
    in          al,71h
    and         al,11111011b
    out         71h,al
    mov         al,32h
    call        PRINT_FUNC
    mov         al,9
    call        PRINT_FUNC
    mov         al,'-'
```

```
    int         29h
    mov         al,8
    call        PRINT_FUNC
    mov         al,'-'
    int         29h
    mov         al,7
    call        PRINT_FUNC
    mov         al,' '
    int         29h
    mov         al,4
    call        PRINT_FUNC
    mov         al,':'
    int         29h
    mov         al,2
    call        PRINT_FUNC
    mov         al,':'
    int         29h
    mov         al,0h
    call        PRINT_FUNC
     mov            al, 0dh
     int      29h
     mov      al, 0ah
     int      29h

     pop dx
     pop bx
     pop cx
     pop ax
     mov al, 20h
     out 20h, al
     iret
SUBR_INT ENDP

PRINT_FUNC proc near
        out         70h,al
        in          al,71h
```

```
        push       ax
           mov        cl, 4
        shr        al,cl
        add        al,'0'
        int        29h
        pop        ax
        and        al,0Fh
        add        al,30h
        int        29h
        ret
PRINT_FUNC endp

Main  PROC  FAR
      push DS
      sub AX, AX
      push AX
      mov AX, DATA
      mov DS, AX

      MOV AH, 35H
      MOV AL, 60H
      INT 21H
      MOV KEEP_IP, BX
      MOV KEEP_CS, ES
      PUSH DS
      MOV DX, OFFSET SUBR_INT
      MOV AX, SEG SUBR_INT
      MOV DS, AX
      MOV AH, 25H
      MOV AL, 60H
      INT 21H
      POP DS

      int 60H

      CLI
```

```
        PUSH DS
        MOV DX, KEEP_IP
        MOV AX, KEEP_CS
        MOV DS, AX
        MOV AH, 25H
        MOV AL, 60H
        INT 21H
        POP DS
        STI

        mov ah, 4ch
        int 21h
Main      ENDP
CODE ENDS
        END Main
```