

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ  
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ»  
ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)»  
Кафедра МО ЭВМ**

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

Студент гр. 1303

Преподаватель

Мусатов Д.Е.

Ефремов М.А.

Санкт-Петербург

2022

## **Цель работы.**

Изучить особенности прерываний на языке Ассемблера, написать собственное прерывание.

## **Задание.**

Вариант 1.

Заменить прерывание 08h от системного таймера, который генерируется автоматически операционной системой 18 раз в секунду, на собственную процедуру: выполнить вывод сообщения на экран заданное число раз, после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

## **Выполнение работы.**

В сегменте данных хранятся следующие данные:

- `int_segment dw` — место для хранения сегмента заменённого прерывания;
- `int_offset dw` — место для хранения смещения заменённого прерывания;
- `msg db` — строка сообщения, которая будет выводиться в написанной процедуре прерывания;
- `int_msg db` — строка сообщения, которая будет выводиться при завершении процедуры прерывания;
- `lag db` — флаг для того, чтобы прерывание выводило сообщения лишь 1 раз.

Написана процедура `display_message`, которая выводит на экран сообщение, лежащее в регистре `dx`.

Написана процедура прерывания, которая работает по следующему алгоритму:

Проверяется, работала ли она уже до этого? Если да, то процедура завершается. После этого с помощью инструкции `loop` на экран 5 раз

выводится сообщение "Some message". Далее, с помощью цикла и инструкции `por` происходит задержка на некоторое количество времени. Затем выводится сообщение об окончании прерывания.

В "главном теле" программы вначале с помощью функции `35h` прерывания `int 21h` узнаётся вектор прерывания под номером `08h`, информация о котором записывается в соответствующие области памяти, указанные в сегменте данных.

Далее, с помощью прерывания функции `25h` прерывания `int 21h` в прерывание под номером `08h` устанавливается вектор написанной выше процедуры прерывания.

После этого программа ожидает 1 секунду, чтобы пронаблюдать за корректностью выполнения процедуры прерывания.

В конце концов, в прерывание под номером `08h` устанавливается вектор стандартного (ранее заменённого) прерывания.

### **Выводы**

В ходе выполнения лабораторной работы были изучены виды прерываний и работа с ними. В соответствии с заданием было разработано собственное прерывание, которым было заменено стандартное прерывание.

# ПРИЛОЖЕНИЕ

## АКОД

## ПРОГРАММ

Название файла: lab5.asm

```
assume ss:my_stack, cs:my_code, ds:my_data

my_stack segment stack
    db 1024 dup(0)
my_stack ends

my_data segment
    int_segment dw 0
    int_offset dw 0

    msg db 'Some message', 0ah, 0dh, '$'
    int_msg db 'Interruption is done', 0ah, 0dh, '$'
    flag db 0
my_data ends

my_code segment

display_message proc near ; print message from `dx` register
    push ax

    mov  ah, 09h
    int  21h

    pop  ax
    ret
display_message endp

my_int_proc proc far
; To run procedure only once
    cmp  flag, 0
    jne  my_int_proc_end
    mov  flag, 1

    push cx
    push dx
    push ax

; Print messages
    xor  cx, cx
    mov  cx, 5
    mov  dx, offset msg

print_msg_loop:
    call display_message
```

```

        loop print_msg_loop

; Wait...
        xor    cx, cx
        mov    cx, 20
update_dx:
        mov    dx, 0ffffh
wait_loop:
        nop
        dec    dx
        cmp    dx, 0
        jne    wait_loop
        loop   update_dx

; Print `end` message
        mov    dx, offset int_msg
        call   display_message

        pop    ax
        pop    dx
        pop    cx

my_int_proc_end:
        mov    al, 20h
        out    20h, al

        iret
my_int_proc endp

main proc far
        push   ds
        xor    ax, ax
        push   ax

        mov    ax, my_data
        mov    ds, ax

; Save previous interruption info
        mov    ah, 35h
        mov    al, 08h
        int    21h
        mov    int_offset, bx
        mov    int_segment, es

; Change interruption to custom
        push   ds

        mov    dx, offset my_int_proc
        mov    ax, seg my_int_proc
        mov    ds, ax

        mov    ah, 25h
        mov    al, 08h

```

```

    int 21h

    pop ds

; Wait 1 second
    xor  ax, ax
    mov  ah, 86h
    mov  cx, 0fh
    mov  dx, 4240h
    int  15h

; Restore interruption
    cli
    push ds

    mov  dx, int_offset
    mov  ax, int_segment
    mov  ds, ax

    mov  ah, 25h
    mov  al, 08h
    int  21h

    pop  ds
    sti

    ret
main endp

my_code ends

end main

```