

**МИНОБРНАУКИ РОССИИ
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: Разработка собственного прерывания.

Студент гр. 1383

Ковалев П. А.

Преподаватель

Ефремов М. А.

Санкт-Петербург
2022

Цель работы

Разработать программу обработки прерываний.

Задание

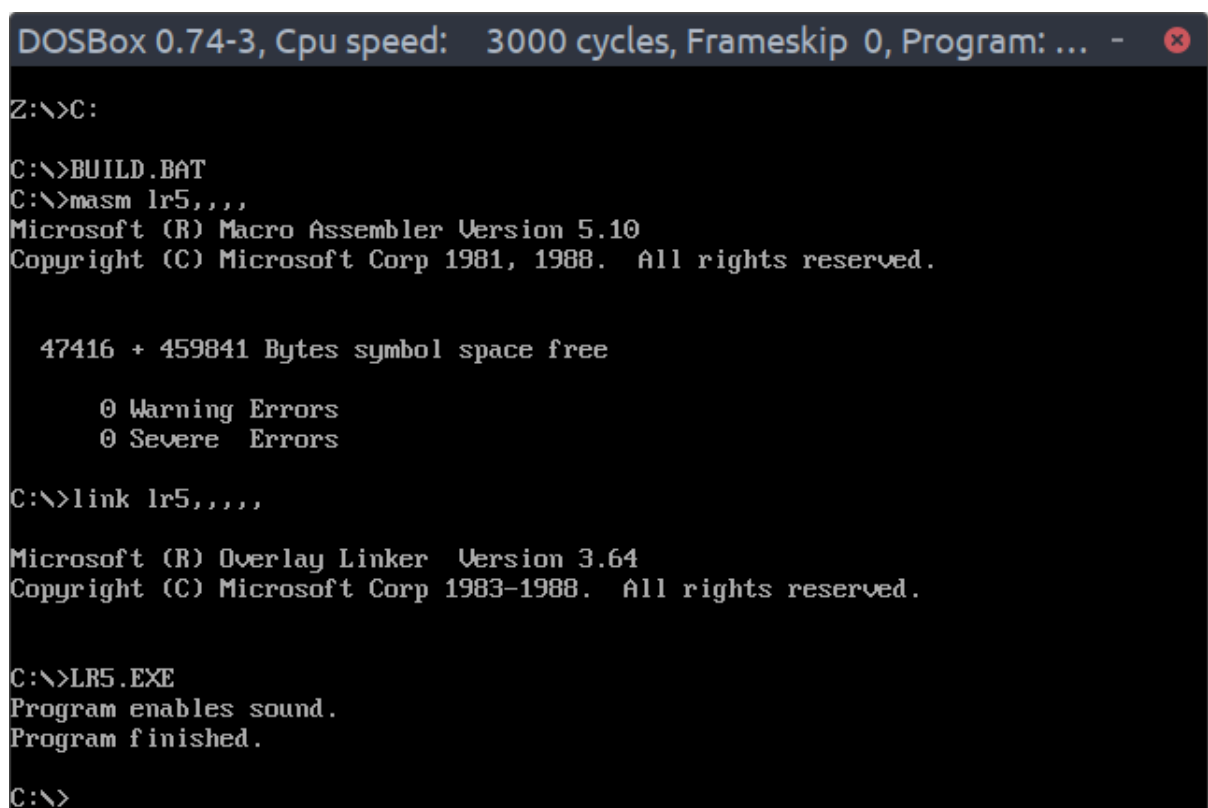
Вариант 2. (1В)

Цифра в шифре задает номер и назначение заменяемого вектора прерывания:

1 - 08h - аппаратное прерывание от часов - генерируется автоматически 18.2 раз в сек;

В - Выдача звукового сигнала заданной частотой.

Выполнение работы



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: ... -
Z:\>C:
C:\>BUILD.BAT
C:\>masm lr5,,,,
Microsoft (R) Macro Assembler Version 5.10
Copyright (C) Microsoft Corp 1981, 1988. All rights reserved.

47416 + 459841 Bytes symbol space free

0 Warning Errors
0 Severe Errors
C:\>link lr5,,,,
Microsoft (R) Overlay Linker Version 3.64
Copyright (C) Microsoft Corp 1983-1988. All rights reserved.

C:\>LR5.EXE
Program enables sound.
Program finished.
C:\>
```

Рис. 1: Сборка и запуск программы в эмуляторе dosbox

Сначала создаются переменные KEEР_CS (для хранения сегмента) и KEEР_IP (для смещения прерывания). Функция 35H прерывания 21H воз-

вращает текущее значение вектора прерывания, его смещение и сегмент записываются в соответствующие переменные для их последующего восстановления. Далее с целью задания адреса собственного прерывания (процедура SUBR_INT по вектору 08h) применяется функция 25h прерывания 21h.

В обработчике прерывания сначала сохраняются все изменяемые регистры (ax, cx). Так как новое прерывание выполняет действия зависящие от времени, отправляется сигнал контролеру прерываний. Затем реализован цикл по счётчику cx, в который записывается длительность звукового сигнала. Перед циклом задержки на порт динамика и таймера записываются значения для включения динамика и частота звука. После цикла задержки динамик выключается. Перед окончанием процедуры восстанавливаются исходные значения регистров. При запуске программа создает прерывистые звуковые сигналы заданное количество времени.

В конце программы восстанавливается старый вектор прерывания.

Вывод

Успешно реализована программа на языке Ассемблера, обрабатывающая прерывания.

Приложение А

Исходный код программы

Название файла: lr5.asm

```
intNum EQU 08h
BEEP_TONE EQU 20
BEEP_DURATION EQU 500

ASTACK SEGMENT STACK
    DB 2048 DUP(?)
ASTACK ENDS

DATA SEGMENT

keep_cs DW 0
keep_ip DW 0
msgHello DB 'Program enables sound.', 10, 13, '$'
msgBye    DB 'Program finished.', 10, 13, '$'
check DW 300

DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, SS:ASTACK

MAIN PROC FAR
    push ds                ; Initialize program
    xor ax, ax             ;
    push ax               ;
    mov ax, DATA          ;
    mov ds, ax             ; ---

    mov dx, OFFSET msgHello
    mov ah, 09h
    int 21h

    call SET_USER_TABLE

endlessness:
    cmp BYTE PTR [check], 0
    jne endlessness
```

```

    call RESTORE_TABLE

    mov dx, OFFSET msgBye
    mov ah, 09h
    int 21h

    ret
MAIN ENDP

SUBR_INT PROC FAR
    push ax
    push cx

    mov al, 20h
    out 20h, al

    dec WORD PTR [check]

    mov al, BEEP_TONE
    out 42h, al
    in al, 61h
    or al, 3
    out 61h, al

    mov cx, BEEP_DURATION
sound:
    loop sound

    xor al, 3
    out 61h, al

    pop cx
    pop ax
    iret
SUBR_INT ENDP

RESTORE_TABLE PROC NEAR
    push dx
    push ax

```

```

        cli                                ; Interrupt protection; Clear
Interrupt Flag
        push ds                            ; Save ds
        mov dx, keep_ip                    ; Restore original ip and cs
values
        mov ax, keep_cs                    ;
        mov ds, ax                        ;
        mov ah, 25h                        ; ---
        mov al, intNum                     ; Restore original handler
        int 21h                            ; ---
        pop ds                            ; Restore ds
        sti                                ; Interrupt protection; -> cli

        pop ax
        pop dx
        ret
RESTORE_TABLE ENDP

SET_USER_TABLE PROC NEAR
        push ax
        push dx

        mov ah, 35h                        ; Save ip and cs values
        mov al, 1ch                        ; before registering a new one
        int 21h                            ;
        mov keep_ip, bx                    ;
        mov keep_cs, es                    ; ---

        push ds                            ; Set interruption handler
        mov dx, OFFSET subr_int            ; proc offset
        mov ax, SEG subr_int               ; ax = proc segment
        mov ds, ax                        ; ds = ax
        mov ah, 25h                        ; Set function
        mov al, intNum                     ; Set int handler number
        int 21h                            ; Change handler
        pop ds                            ; Restore original ds

        pop dx
        pop ax
        ret
SET_USER_TABLE ENDP

```

CODE ENDS
END MAIN