**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Организация ЭВМ и систем»**

Тема: **Написание собственного прерывания.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Попандопуло А. Г. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2022

## **Цель работы.**

## Практическое изучение прерываний на языке Ассемблера, написание собственного прерывания согласно условию.

**Задание.**

Вариант 22 (шифр 4a):

Написать прерывание 16h - прерывание от клавиатуры, выполняющее вывод сообщения на экран заданное число раз, после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

Замечание: для исключения возможного взаимного влияния системных и пользовательских прерываний рекомендуется отвести в программе под стек не менее 1К байт.

**Выполнение работы**

Для хранения сегмента заменяемого прерывания и для хранения смещения заменяемого прерывания, в сегменте памяти выделяем место под слова keep\_cs и keep\_ip соответственно. Согласно условию, на стек отводим 1 Кб.

С помощью 35h сохраняем адрес прошлого прерывания, с помощью 25h – устанавливаем адрес нового. Инициализируем строки MESSAGE – повторяющееся некоторое количество раз (во время обработки прерывания) сообщение, и END\_MESSAGE – сообщение о завершении обработчика.

Далее следует ожидание ввода символа от пользователя, в соответствии с условием, взят символ «a». На метке check\_key происходит считывание из порта клавиатуры 60h с последующим сравнением на 1Eh – скан, соответствующий клавише «a». Непосредственно прерывание вызывается при нажатии нужной клавиши; в противно случае, вновь переходим на метку check\_key, таким образом, «ожидая» нужного символа.

В сегменте кода, помимо процедуры самого прерывания, была определена процедура WriteMsg – для печати сообщения. В процедуре прерывания, сохраняем в стек изначальные значения регистров, после чего посредством Ip строка из dx выводится заданным в cx числом раз. Задержка после нужного числа выводов строк происходит через прерывания 15h; после нее выводится END\_MESSAGE.

**Тестирование:**

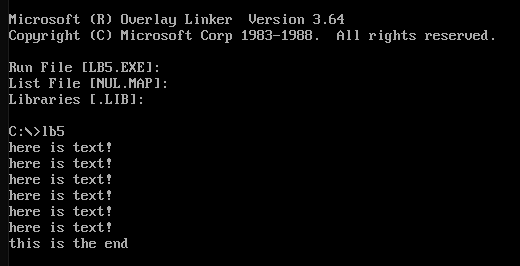


Рис. 1

**Вывод:** в ходе выполнения лабораторной работы, на практике были изучены способы работы с прерываниями на языке Ассемблера; удалось реализовать собственное прерывание, соответствующее заданному условию.

**Приложение А.**

**Исходный код программы.**

AStack segment stack

dw 512 dup(?)

AStack ENDS

DATA segment

keep\_cs dw 0

keep\_ip dw 0

MESSAGE DB 'here is text!', 0dh, 0ah, '$'

END\_MES DB 'this is the end', 0dh, 0ah, '$'

FLAG DB 0

DATA ends

CODE segment

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

WriteMsg PROC NEAR

mov AH, 9

int 21h

ret

WriteMsg ENDP

FUNC proc far

cmp FLAG, 0

jne func\_end

mov FLAG, 1

push ax

push bx

push cx

push dx

push ds

mov dx, OFFSET MESSAGE

mov cx, 6

lp:

call WriteMsg

loop lp

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

mov dx, OFFSET END\_MES

call WriteMsg

pop ds

pop dx

pop cx

pop bx

pop ax

func\_end:

mov al, 20h

out 20h, al

iret

FUNC endp

main proc far

push ds

sub ax, ax

push ax

mov ax, DATA

mov ds, ax

mov ah, 35h

mov al, 16h

int 21h

mov keep\_ip, bx

mov keep\_cs, es

push ds

mov dx, offset FUNC

mov ax, seg FUNC

mov ds, ax

mov ah, 25h

mov al, 16h

int 21h

pop ds

check\_key:

in al, 60h

cmp al, 1Eh

jne check\_key

int 16h

mov al, 0

mov cx, 002Eh

mov dx, 0000h

mov ah, 86h

int 15h

cli

push ds

mov dx, keep\_ip

mov ax, keep\_cs

mov ds, ax

mov ah, 25h

mov al, 16h

int 21h

pop ds

sti

ret

main endp

CODE ends

end main