

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

ОТЧЕТ
по лабораторной работе №5
по дисциплине «Организация ЭВМ и систем»
Тема: Написание собственного прерывания
Вариант 16

Студент гр.1303

Насонов Я.К.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Изучить, как работают прерывания. Написать собственное прерывание.

Задание.

В соответствии с 16 вариантом шифр задания – 2А, где 2 – 60h прерывание пользователя – должно генерироваться в программе; А – выполнить вывод сообщения на экран заданное число раз, после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

Выполнение работы.

В сегменте данных DATA содержится две переменных для хранения старого прерывания, содержавшегося по смещению 60h, – previous_seg, previous_ip.

Также в этом сегменте содержится output_message – сообщение, которое будет выводиться во время работы прерывания, exit_message – сообщение, которое будет выведено после завершения работы прерывания.

В сегменте стека Astack, как и требуется по заданию, выделяется 1Кбайт памяти, то есть dw 512.

В сегменте кода сначала определяем процедуру пользовательского прерывания INT_CUSTOM. Сначала на стеке сохраняются значения регистров до входа в прерывание. С помощью метки output_loop строка из ds:dx выводится заданное в cx количество раз. Далее реализована задержка после вывода строк с помощью прерывания 1Ah. В регистре bx содержится требуемая задержка в тактах процессора, далее к ней прибавляется текущее время работы программы, которое прерыванием 1Ah записывается в cx, dx. Далее в цикле происходит сравнение bx с текущим временем работы программы, если оно больше, то происходит выход из цикла. И при помощи прерывания 21h происходит вывод строки, сообщающей о завершении работы прерывания. Оно хранится по адресу ds:offset exit_message. Далее перед выходом из прерывания восстанавливаются регистры из стека. Вызов прерывания происходит в процедуре Main. Для этого

сначала с помощью прерывания 21h получается прерывание, хранящееся по смещению 60h. В переменных, указанных в сегменте данных, сохраняется старое прерывание. Новое прерывание INT_CUSTOM записывается по смещению 60h также с помощью прерывания 21h. Далее задаются значения регистров: в ds:dx должна лежать выводимая несколько раз строка, в cx – количество раз сколько нужно вывести строку, в bx – время задержки, в ds:offset – сообщение о завершении.

После вызова нового прерывания происходит восстановление старого прерывания и выход из программы.

Тестирование.

Работа программы с заданными условиями представлена на рисунке 1.

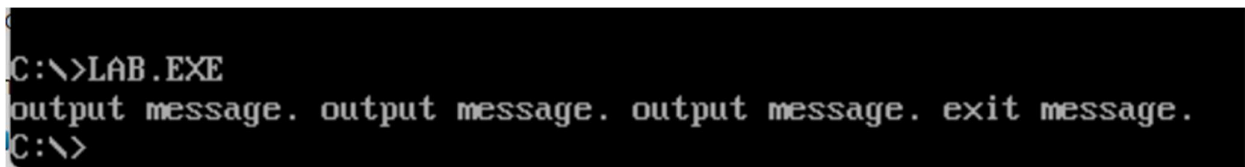
При вызове прерывания заданы следующие регистры:

ds:dx output_message (где output_message – это «output message. »)

cx = 03h (количество повторов выводимых сообщений – 3)

bx = 64h (время задержки в тактах процессора)

ds:offset exit_message (где exit_message – это «exit_message.»)



```
C:\>LAB.EXE
output message. output message. output message. exit message.
C:\>
```

Рисунок 1 – Работа программы

Выводы.

В ходе выполнения лабораторной работы были изучены виды прерываний и работа с ними. В соответствии с заданием было создано собственное прерывание. Была написана программа, выводимая одно сообщение определённое количество раз, а другое – один раз с определённой задержкой.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММ

Название файла: *lab5.asm*

```
DATA SEGMENT
    previous_seg dw 0
    previous_ip dw 0
    output_message db 'output message. $'
    exit_message db 'exit message.$'
DATA ENDS

AStack SEGMENT STACK
    dw 512 dup(?)
AStack ENDS

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

INT_CUSTOM PROC FAR
    push ax    ; storing registers
    push bx
    push cx    ; push numbers of output message prints
    push dx

    mov ah, 9h ; print cx times

output_loop:
    int 21h    ; DOS service
    loop output_loop

    mov ah, 0  ; delay
    int 1Ah    ; I/O for time
    add bx, dx

delay:
    mov ah, 0
    int 1Ah    ; I/O for time
    cmp bx, dx
    jg delay

    mov dx, offset exit_message ; print exit message
    mov ah, 9h
    int 21h    ; DOS service

    pop dx     ; restoring
    pop cx
    pop bx
    pop ax

    mov al, 20h
    out 20h, al
    iret
INT_CUSTOM ENDP

Main PROC FAR
    push ds
    sub ax, ax
```

```

    push ax
    mov ax, data
    mov ds, ax

    mov ax, 3560h ; storing previous interruption
    int 21h ; DOS service
    mov previous_seg, es
    mov previous_ip, bx

    push ds ; setting custom interruption
    mov dx, offset int_custom
    mov ax, seg int_custom
    mov ds, ax
    mov ax, 2560h
    int 21h ; DOS service
    pop ds

    mov dx, offset output_message ; setting registers using
custom interruption manual
    mov cx, 03h ; number of messages
    mov bx, 64h ; delay in ticks of process /seconds/
    int 60h ; user interruption

    CLI ; restoring previous interruption
    push ds
    mov dx, previous_ip
    mov ax, previous_seg
    mov ds, ax
    mov ax, 251ch
    int 21h ; DOS service
    pop ds
    STI

    ret

main ENDP

CODE ENDS
END Main

```