

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. 0383

Сергеев Д.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Написать собственное прерывание, согласно заданию.

Вариант 14(2g).

2 - 60h - прерывание пользователя - должно генерироваться в программе;

g - выполнить ввод и печать заданного количества символов, после чего вывести сообщение о завершении обработчика.

Выполнение работы:

В сегменте данных DATA содержатся переменные: KEEP_CS, KEEP_IP для хранения сегмента и смещения старого прерывания. Для записи введенных символов - message, message2 для вывода строки о завершении.

Процедура пользовательского прерывания foo. Сначала мы сохраняем все изменяемые регистры в стеке. Помещаем в регистр AX значение siz, количество вводимых символов, помещаем значение регистра AX в CX, далее следует цикл. Получаем смещение на начало строки message, откуда начинать ввод, в регистр DI. Начинаем цикл lp. В нем мы используем функцию 01h прерывания 21h для получения очередного введенного символа (помещается в AL). Заносим в [DI] этот символ. Увеличиваем DI, чтобы на следующем шаге записывать символ в свободное место. По завершении цикла, в AH помещается значение функции вывода строки (09h). В DX помещается смещение на начало строки message. Вызываем 21h прерывание. В DX помещается смещение на начало строки о завершении. Вызываем 21h прерывание. Строки выведены в консоль.

В процедуре Main запоминаем смещение и сегмент текущего 60h прерывания в KEEP_IP, KEEP_CS с помощью функции 35h прерывания 21h. Используя же функцию 25h прерывания 21, устанавливаем вектор прерывания 60h на наше

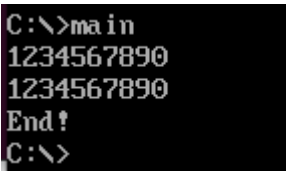
прерывание foo. Затем происходит его вызов. Когда его работа будет завершена – восстанавливаем старый вектор прерывания.

Исходный код программы см. в приложении А.

Тестирование:

Для проверки работоспособности были проведены тесты, результаты представлены в таблице 1.

Таблица 1 – Тестирование и результаты

Номер теста	Входные данные	Выходные данные	Верность результата
1	(ввод 10 символов) 1234567890		верно
2	(ввод 5 символов) qwe!2		верно
3	(ввод 1 символа) 9		верно

Выводы.

В ходе работы были изучены прерывания. Также было написано собственное прерывание по вводу-выводу строки и строки о завершении.

ПРИЛОЖЕНИЕ А

ТЕКСТЫ ИСХОДНЫХ ФАЙЛОВ ПРОГРАММ

Название файла: **main.asm**

```
siz equ 10 ; кол-во символов в строке для ввода
```

```
AStack SEGMENT STACK
```

```
    DW 12 DUP(?)
```

```
AStack ENDS
```

```
DATA SEGMENT
```

```
    KEEP_CS DW 0 ; для хранения сегмента
```

```
    KEEP_IP DW 0 ; и смещения прерывания
```

```
    message DB 0Dh, 0Ah,siz dup("$"), '$'
```

```
    message2 DB 0Dh, 0Ah,'End!','$' ; строка о завершении
```

```
DATA ENDS
```

```
CODE SEGMENT
```

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

```
foo PROC FAR
```

```
    push AX ; сохранение изменяемых регистров
```

```
    push CX
```

```
    push DX
```

```
    mov ax, siz
```

```
    mov cx, ax
```

```
    mov di, offset message ; получаем смещение на начало  
сообщения
```

```
    add di, 2
```

```
lp:
```

```
    mov ah, 01h ; функция ввода с клавиатуры
```

```
    int 21h
```

```

mov [di], al ; помещаем символ в строку
add di, 1
loop lp ;цикл

mov ah, 09h ; функция вывода строки
mov dx, offset message ; указываем смещение строки
int 21h ; вывели
mov dx, offset message2
int 21h ; вывели

pop DX ;восстанавливаем регистры
pop CX
    pop AX
        mov AL, 20h ; для разрешения обработки прерываний
            out 20h,AL ; с более низкими уровнями, чем только что
обработанное
                iret
foo ENDP

```

```

Main PROC FAR

```

```

push DS

```

```

sub AX,AX

```

```

push AX

```

```

mov AX, DATA

```

```

mov DS, AX

```

```

mov AH,35h ; возвращение текущего значения вектора прерывания

```

```

mov AL,60h ; номер вектора

```

```

int 21h

```

```

mov KEEP_IP, BX ; запоминание смещения

```

```

mov KEEP_CS, ES ; запоминание сегмента

```

```

push DS
mov DX, offset foo ; смещение для процедуры
mov AX, seg foo ; сегмент процедуры
mov DS, AX
mov AH, 25h ; функция установки вектора
mov AL, 60h ; номер вектора
int 21h ; устанавливаем вектор прерывания на указанный адрес
нового обработчика
pop DS

int 60h ; вызываем прерывание пользователя

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

ret
Main ENDP
CODE ENDS
END Main

```

ПРИЛОЖЕНИЕ Б

ТЕКСТЫ ФАЙЛОВ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: **main.lst**

Microsoft (R) Macro Assembler Version 5.10
 12/9/21 12:52:37

Page 1-1

```

      = 000A                                siz equ 10 ; 0°0Y0»-0²0Y
Ñ 0ž000²0Y0»0Y0² 0² Ñ
      Ñ,Ñ 0Y0°0µ 0ž0»Ñ 0²0²0Y0ž0°

0000                                AStack SEGMENT STACK
0000 000C[                            DW 12 DUP(?)
      ???
      ]

0018                                AStack ENDS

0000                                DATA SEGMENT
0000 0000                            KEEP_CS DW 0 ; 0ž0»Ñ
Ñ...Ñ 0°0000µ0000žÑ Ñ 0µ0³0
      0000µ0000,0°

0002 0000                            KEEP_IP DW 0 ; 0ž Ñ 000µÑ%0µ000žÑ
0žÑ 0µÑ Ñ<0
      ²0°000žÑ

0004 0D 0A                            message DB 0Dh, 0Ah,siz dup("$"),
'$'

      000A[
      24
      ]

      24

0011 0D 0A 45 6E 64 21  message2 DB 0Dh, 0Ah,'End!','$' ;
Ñ Ñ,Ñ 0Y0°0
      ° 0Y 0·0°0²0µÑ Ñ^0µ000žž

```

```

24
0018                                DATA ENDS

0000                                CODE SEGMENT
                                ASSUME CS:CODE, DS:DATA, SS:AStack

0000                                foo PROC FAR
0000  50                                push AX ; Ń ĐŸŃ...Ń Đ°ĐœĐµĐœĐžĐµ
ĐžĐ·ĐœĐµ
                                ĐœŃ ĐµĐœŃ<Ń... Ń ĐµĐ³ĐžŃ Ń,Ń ĐŸĐ²
0001  51                                push CX
0002  52                                push DX

0003  B8 000A                                mov ax, siz
0006  8B C8                                mov cx, ax
0008  BF 0004 R                                mov di, offset message ;
ĐžĐŸĐ»ŃfŃ‡Đ°Đµ
                                Đœ Ń ĐœĐµŃ%ĐµĐœĐžĐµ ĐœĐ° ĐœĐ°Ń‡Đ°Đ»ĐŸ Ń ĐŸĐŸĐ±Ń
                                %ĐµĐœĐžŃ
000B  83 C7 02                                add di, 2
000E                                lp:
000E  B4 01                                mov ah, 01h ; Ń,ŃfĐœĐ°Ń‡ĐžŃ
Đ²Đ²ĐŸĐžĐ°
                                Ń Đ°Đ»Đ°Đ²ĐžĐ°Ń,ŃfŃ Ń<
0010  CD 21                                int 21h
0012  88 05                                mov [di], al ; ĐžĐŸĐœĐµŃ%Đ°ĐµĐœ
Ń ĐžĐœĐ
                                ²ĐŸĐ» Đ² Ń Ń,Ń ĐŸĐ°Ńf
0014  83 C7 01                                add di, 1
0017  E2 F5                                loop lp ;Ń‡ĐžĐ°Đ»

0019  B4 09                                mov ah, 09h ; Ń,ŃfĐœĐ°Ń‡ĐžŃ
Đ²Ń<Đ²ĐŸĐž

```


Đ° Ñ Ñ,Ñ ĐŸĐ°Đž

001B BA 0004 R mov dx, offset message ;
ÑƒĐ°Đ°Đ·Ñ<Đ²Đ°

ĐµĐ€ Ñ Đ€ĐµÑ%ĐµĐœĐžĐµ Ñ Ñ,Ñ ĐŸĐ°Đž

001E CD 21 int 21h ; Đ²Ñ<Đ²ĐµĐ»Đž

Microsoft (R) Macro Assembler Version 5.10

12/9/21 12:52:37

Page 1-2

0020 BA 0011 R mov dx, offset message2

0023 CD 21 int 21h ; Đ²Ñ<Đ²ĐµĐ»Đž

0025 5A pop DX

;Đ²ĐŸÑ Ñ Ñ,Đ°ĐœĐŸĐ²Đ»ĐµĐ²Đ°ĐµĐ€

Ñ ĐµĐ³ĐžÑ Ñ,Ñ Ñ<

0026 59 pop CX

0027 58 pop AX

0028 B0 20 mov AL, 20h ; ĐžĐ»Ñ

Ñ Đ°Đ·Ñ ĐµÑ~ĐµĐœĐžÑ Đ

ŸĐ±Ñ Đ°Đ±ĐŸĐ°Đž ĐžÑ ĐµÑ Ñ<Đ²Đ°ĐœĐžĐ¹

002A E6 20 out 20h,AL ; Ñ Đ±ĐŸĐ»ĐµĐµ

ĐœĐžĐ·Đ°ĐžĐ€Đž Ñ

ƒÑ ĐŸĐ²ĐœÑ Đ€Đž, ÑžĐµĐ€ Ñ,ĐŸĐ»Ñ€Đ°ĐŸ ÑžÑ,ĐŸ ĐŸĐ

±Ñ Đ°Đ±ĐŸÑ,Đ°ĐœĐœĐŸĐµ

002C CF iret

002D foo ENDP

002D Main PROC FAR

002D 1E push DS

002E 2B C0 sub AX,AX

```

0030  50                                push AX
0031  B8 ---- R                          mov AX, DATA
0034  8E D8                              mov DS, AX

0036  B4 35                              mov AH, 35h ;
Đ²ĐŸĐ ·Đ²Ń Đ°ŃĐμĐœĐžĐμ Ń,Đ
                                μĐ°ŃfŃĐμĐ³ĐŸ Đ·ĐœĐ°ŃĐμĐœĐžŃ Đ²ĐμĐ°Ń,ĐŸŃ Đ° Đ
                                ċŃ ĐμŃ ŃĐ²Đ°ĐœĐžŃ
0038  B0 60                              mov AL, 60h ; ĐœĐŸĐœĐμŃ
Đ²ĐμĐ°Ń,ĐŸŃ Đ°
003A  CD 21                              int 21h
003C  89 1E 0002 R                      mov KEEP_IP, BX ;
Đ·Đ°ĐċĐŸĐœĐžĐœĐ°ĐœĐžĐ
                                μ Ń ĐœĐμŃĐμĐœĐžŃ
0040  8C 06 0000 R                      mov KEEP_CS, ES ;
Đ·Đ°ĐċĐŸĐœĐžĐœĐ°ĐœĐžĐ
                                μ Ń ĐμĐ³ĐœĐμĐœŃ,Đ°

0044  1E                                push DS
0045  BA 0000 R                          mov DX, offset foo ;
Ń ĐœĐμŃĐμĐœĐžĐμ Đ
                                žĐ»Ń ĐċŃ ĐŸŃĐμĐžŃfŃ Ń
0048  B8 ---- R                          mov AX, seg foo ; Ń ĐμĐ³ĐœĐμĐœŃ,
ĐċŃ ĐŸ
                                ŃĐμĐžŃfŃ Ń
004B  8E D8                              mov DS, AX
004D  B4 25                              mov AH, 25h ; Ń,ŃfĐœĐ°ŃĐžŃ
ŃfŃ Ń,Đ°Đœ
                                ĐŸĐ²Đ°Đž Đ²ĐμĐ°Ń,ĐŸŃ Đ°
004F  B0 60                              mov AL, 60h ; ĐœĐŸĐœĐμŃ
Đ²ĐμĐ°Ń,ĐŸŃ Đ°
0051  CD 21                              int 21h ;
ŃfŃ Ń,Đ°ĐœĐ°Đ²Đ»ĐžĐ²Đ°ĐμĐœ Đ²

```


Symbols-1

Segments and Groups:

	N a m e	Length	Align
Combine Class			
ASTACK	0018	PARA STACK
CODE	006A	PARA NONE
DATA	0018	PARA NONE

Symbols:

	N a m e	Type	Value	Attr
FOO	F PROC	0000	CODE Length
= 002D				
KEEP_CS	L WORD	0000	DATA
KEEP_IP	L WORD	0002	DATA
LP	L NEAR	000E	CODE
MAIN	F PROC	002D	CODE Length
= 003D				
MESSAGE	L BYTE	0004	DATA
MESSAGE2	L BYTE	0011	DATA
SIZ	NUMBER	000A	
@CPU	TEXT	0101h	

```
@FILENAME . . . . . TEXT main
@VERSION . . . . . TEXT 510
```

```
86 Source Lines
```

```
86 Total Lines
```

```
16 Symbols
```

```
48016 + 459244 Bytes symbol space free
```

```
0 Warning Errors
```

```
0 Severe Errors
```