**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.**
**Вариант 15.**

Студент гр. 0382 _____ Санников В.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

**Цель работы.**

Изучить работу прерываний на языке Ассемблера и написать собственное.

**Задание.**

3 — 23h — прерывание, генерируемое при нажатии клавиш Control + C;

A — Выполнить вывод сообщения на экран заданное число раз, после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

**Ход работы:**

В сегменте данных DATA хранятся следующие переменные: KEEP_CS, KEEP_IP – для хранения сегмента и смещения старого прерывания, COUNTER – для количества выводимых строк, MESSAGE – сообщение, которое надо вывести несколько раз, FINALLY – сообщение о завершении обработчика.

Процедура пользовательского прерывания называется FUNC. В начале данной процедуры мы сохраняем все изменяемые регистры в стеке с помощью push. Далее запускаемся цикл по метке start для вывода сообщения MESSAGE на экран несколько раз. Как только COUNTER = 0, цикл прекращается. После этого мы кладем временной промежуток в cx и dx, в ah кладем 86h, следовательно вызываем прерывание паузы. После данной паузы печатается сообщение о завершении. Для вывода строк на экран написана процедура WriteMsg. В конце процедуры прерывания восстанавливаем регистры из стека и выходим из пользовательского прерывания.

В главной процедуре программы Main запоминаем смещение и сегмент прерывания 23h в KEEP_IP, KEEP_CS с помощью 35h и 21h. С помощью 25h прерывания 21, устанавливаем вектор прерывания 23h на пользовательское прерывание FUNC и производим его вызов. По завершении прерывания восстанавливаем его старый вектор.

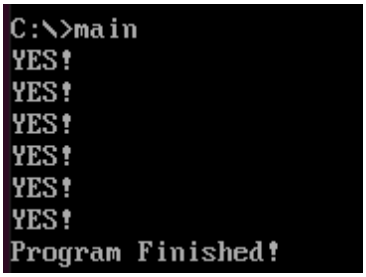
Исходный код программы см. в приложении А.

Файл листинга см. в приложении Б.

**Тестирование:**

Для проверки работоспособности программы были проведены тесты, см. Таблицу 1.

Таблица 1 — Результаты тестирования.

| № теста | Входные данные | Выходные данные | Оценка результата |
|---|---|---|---|
| 1 | (нажато Ctrl + C) |  | Выводится фиксированное количество сообщений(6 штук), после с задержкой в 3с выводится сообщение о завершении обработчика |

**Выводы.**

В данной лабораторной работе были изучены прерывания языка Ассемблера. Написано собственное прерывание, которое выводит строки на экран и сообщение о завершении с задержкой.

# ПРИЛОЖЕНИЕ А
# ИСХОДНЫЙ КОД ПРОГРАММЫ

## Название файла: main.asm

```
AStack  SEGMENT STACK
    DB 1024 DUP(?)
AStack  ENDS

DATA    SEGMENT
     MESSAGE DB 'YES!', 0dh, 0ah, '$'
     FINALLY DB 'Program Finished!$'
     COUNTER DW 6
    KEEP_CS DW 0    ; для хранения сегмента вектора прерывания
    KEEP_IP DW 0    ; для смещения вектора прерывания
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
     mov dx, OFFSET MESSAGE
     push ax
     push bx
     push cx
     push dx
     push ds
     start:
          call WriteMsg
          sub COUNTER, 1
          cmp COUNTER, 0
          jnz start
     mov cx, 0033h
     mov dx, 00FFh
     mov ah, 86h
     int 15h
     mov dx, OFFSET FINALLY
     call WriteMsg
     pop ax
     pop bx
     pop cx
     pop dx
     pop ds
     mov al, 20h
     out 20h, al
    iret
```

```
FUNC ENDP

MAIN PROC FAR
    push ds
    mov ax, DATA
    mov ds, ax

    mov ah, 35h ; функция получения вектора
    mov al, 23h ; номер вектора
    int 21h
    mov KEEP_IP, bx ; запоминание смещения
    mov KEEP_CS, es ; и сегмента вектора прерывания

    push ds
    mov dx, OFFSET FUNC ; смещение для процедуры в DX
    mov ax, SEG FUNC ; сегмент процедуры
    mov ds, ax ; помещаем в DS
    mov ah, 25h ; функция установки вектора
    mov al, 23h ; номер вектора
    int 21h ; меняем прерывание
    pop ds

    begin:
     mov ah, 0
          int 16h
          cmp al, 3
          jnz begin
          int 23h
     quit:
     cli
     push ds
     mov dx, KEEP_IP
     mov ax, KEEP_CS
     mov ds, ax
     mov ah, 25h
     mov al, 23h
     int 21h ; восстанавливаем старый вектор прерывания
     pop ds
     sti
          mov ah, 4ch
          int 21h
MAIN ENDP
CODE ENDS
     END MAIN
```

## ПРИЛОЖЕНИЕ Б

## ФАЙЛЫ ЛИСТИНГА ПРОГРАММЫ

### Название файла: main.lst

```
# Microsoft (R) Macro Assembler Version 5.10
12/7/21 01:00:29

Page     1-1


 0000                   AStack  SEGMENT STACK
 0000  0400[                     DB 1024 DUP(?)
       ??
                ]

 0400                   AStack  ENDS

 0000                   DATA    SEGMENT
 0000  59 45 53 21 0D 0A          MESSAGE DB 'YES!', 0dh, 0ah, '$'
       24
 0007  50 72 6F 67 72 61          FINALLY DB 'Program Finished!$'
       6D 20 46 69 6E 69
       73 68 65 64 21 24
 0019  0006                       COUNTER DW 6
 001B  0000                      KEEP_CS DW 0
 001D  0000                      KEEP_IP DW 0
 001F                   DATA    ENDS

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

 0000                   WriteMsg  PROC  NEAR
 0000  B4 09                          mov AH, 9
 0002  CD 21                          int 21h
 0004  C3                        ret
 0005                   WriteMsg  ENDP

 0005                   FUNC PROC FAR
 0005  BA 0000 R              mov dx, OFFSET MESSAGE
 0008  50                     push ax
 0009  53                     push bx
 000A  51                     push cx
 000B  52                     push dx
 000C  1E                     push ds
 000D                         start:
 000D  E8 0000 R                   call WriteMsg
 0010  83 2E 0019 R 01                        sub COUNTER, 1
 0015  83 3E 0019 R 00                        cmp COUNTER, 0
 001A  75 F1                           jnz start
 001C  B9 0033                    mov cx, 0033h
 001F  BA 00FF                    mov dx, 00FFh
 0022  B4 86                      mov ah, 86h
 0024  CD 15                      int 15h
```

```
 0026  BA 0007 R           mov dx, OFFSET FINALLY
 0029  E8 0000 R           call WriteMsg
 002C  58                  pop ax
 002D  5B                  pop bx
 002E  59                  pop cx
 002F  5A                  pop dx
 0030  1F                  pop ds
 0031  B0 20                    mov al, 20h
# Microsoft (R) Macro Assembler Version 5.10
12/7/21 01:00:29

Page     1-2


 0033  E6 20                    out 20h, al
 0035  CF                iret
 0036                FUNC ENDP

 0036                MAIN PROC FAR
 0036  1E                 push ds
 0037  B8 ---- R          mov ax, DATA
 003A  8E D8                   mov ds, ax

 003C  B4 35                   mov ah, 35h
 003E  B0 23                   mov al, 23h
 0040  CD 21                   int 21h
 0042  89 1E 001D R            mov KEEP_IP, bx
 0046  8C 06 001B R            mov KEEP_CS, es

 004A  1E                 push ds
 004B  BA 0005 R          mov dx, OFFSET FUNC
 004E  B8 ---- R          mov ax, SEG FUNC
 0051  8E D8                   mov ds, ax
 0053  B4 25                   mov ah, 25h
 0055  B0 23                   mov al, 23h
 0057  CD 21                   int 21h
 0059  1F                 pop ds

 005A                     begin:
 005A  B4 00                    mov ah, 0
 005C  CD 16                         int 16h
 005E  3C 03                         cmp al, 3
 0060  75 F8                         jnz begin
 0062  CD 23                         int 23h
 0064                      quit:
 0064  FA                  cli
 0065  1E                  push ds
 0066  8B 16 001D R             mov dx, KEEP_IP
 006A  A1 001B R           mov ax, KEEP_CS
 006D  8E D8                   mov ds, ax
 006F  B4 25                   mov ah, 25h
 0071  B0 23                   mov al, 23h
 0073  CD 21                   int 21h
 0075  1F                  pop ds
 0076  FB                  sti
```

```
 0077  B4 4C                          mov ah, 4ch
 0079  CD 21                          int 21h
 007B                  MAIN ENDP
 007B                  CODE ENDS
                            END MAIN
# Microsoft (R) Macro Assembler Version 5.10
12/7/21 01:00:29

Symbols-1


Segments and Groups:

                N a m e         Length     Align     Combine
Class

ASTACK . . . . . .              0400  PARA  STACK
CODE . . . . . .                007B  PARA  NONE
DATA . . . . . .                001F  PARA  NONE

Symbols:

                N a m e         Type  Value      Attr

BEGIN  . . . . . .              L NEAR     005A  CODE

COUNTER  . . . . . .            L WORD     0019  DATA

FINALLY  . . . . . .            L BYTE     0007  DATA
FUNC . . . . . .                F PROC     0005  CODE
     Length = 0031

KEEP_CS  . . . . . .            L WORD     001B  DATA
KEEP_IP  . . . . . .            L WORD     001D  DATA

MAIN . . . . . .                F PROC     0036  CODE
     Length = 0045
MESSAGE  . . . . . .            L BYTE     0000  DATA

QUIT . . . . . .                L NEAR     0064  CODE

START  . . . . . .              L NEAR     000D  CODE

WRITEMSG . . . . . .            N PROC     0000  CODE
     Length = 0005

@CPU . . . . . .                TEXT  0101h
@FILENAME  . . . . . .          TEXT  main
@VERSION . . . . . .            TEXT  510


     91 Source  Lines
     91 Total   Lines
     19 Symbols
```

```
48004 + 459256 Bytes symbol space free

    0 Warning Errors
    0 Severe  Errors
```