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.

Студентка гр. 1381 _____ Манцева Т.К.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Изучить прерывания, их написание.

Задание.

1. Краткие сведения.

Прерывание - это процесс вызова процедур для выполнения некоторой задачи, обычно связанной с обслуживанием некоторых устройств (обработка сигнала таймера, нажатия клавиши и т.д.). Когда возникает прерывание, процессор прекращает выполнение текущей программы (если ее приоритет ниже) и запоминает в стеке вместе с регистром флагов адрес возврата(CS:IP) - места, с которого будет продолжена прерванная программа. Затем в CS:IP загружается адрес программы обработки прерывания и ей передается управление. Адреса 256 программ обработки прерываний, так называемые векторы прерывания, имеют длину по 4 байта (в первых двух хранится значение IP , во вторых - CS) и хранятся в младших 1024 байтах памяти. Программа обработки прерывания должна заканчиваться инструкцией IRET (возврат из прерывания), по которой из стека восстанавливается адрес возврата и регистр флагов.

Программа обработки прерывания - это отдельная процедура, имеющая структуру:

```
SUBR_INT PROC FAR
PUSH AX ; сохранение изменяемых регистров
<действия по обработке прерывания>
POP AX ; восстановление регистров
MOV AL, 20H
OUT 20H,AL
IRET
SUBR_INT ENDP
```

Две последние строки перед IRET необходимы для разрешения обработки прерываний с более низкими уровнями, чем только что

обработанное. Замечание: в лабораторной работе действиями по обработке прерывания может быть вывод на экран некоторого текста, вставка задержки в вывод сообщений, включение звукового сигнала и т.п.

Программа, использующая новые программы обработки прерываний, при своем завершении должна восстанавливать оригинальные векторы прерываний. Функция 35 прерывания 21H возвращает текущее значение вектора прерывания, помещая значение сегмента в ES, а смещение в BX.

В этом случае программа должна содержать следующие инструкции:

```
; -- в сегменте данных
KEEP_CS DW 0 ; для хранения сегмента
KEEP_IP DW 0 ; и смещения прерывания
; -- в начале программы
MOV AH, 35H ; функция получения вектора
MOV AL, 1CH ; номер вектора
INT 21H
MOV KEEP_IP, BX ; запоминание смещения
MOV KEEP_CS, ES ; и сегмента
```

Для задания адреса собственного прерывания с заданным номером в таблицу векторов прерываний используется функция 25H прерывания 21H, которая устанавливает вектор прерывания на указанный адрес нового обработчика.

```
PUSH DS
MOV DX, OFFSET ROUT ; смещение для процедуры в DX
MOV AX, SEG ROUT ; сегмент процедуры
MOV DS, AX; помещаем в DS
MOV AH, 25H; функция установки вектора
MOV AL, 60H; номер вектора
INT 21H; меняем прерывание
POP DS
```

В конце программы восстанавливается старый вектор прерывания

CLI

PUSH DS

MOV DX, KEEP_IP

MOV AX, KEEP_CS

MOV DS, AX

MOV AH, 25H

MOV AL, 1CH

INT 21H; восстанавливаем вектор

POP DS

STI

Вариант 7 (1g)

1 - 08h - прерывание от системного таймера — генерируется автоматически операционной системой 18 раз в сек. G - Выполнить ввод и печать заданного количества символов, после чего вывести сообщение о завершении обработчика.

Выполнение работы.

Для работы программы созданы сегмент стека AStack размером 1024 байта, сегмент данных Data, в котором хранятся переменные symbol, msg, KEEP_CS, KEEP_IP, сегмент кода Code с прерыванием и главной процедурой. Реализовано собственное прерывание interruption, в котором сначала в цикле print_loop 10 раз(ind) выводится символ из переменной symbol, затем выводится сообщение msg о завершении прерывания. В главной процедуре Main в регистр ds перемещается смещение на сегмент данных, cx присваивается значение ind, с клавиатуры считывается символ, сохраняется оригинальное прерывание 08h. Затем это прерывание меняется на собственное, вызывается, его старый вектор восстанавливается. Разработанный код представлен в приложении А. Файл листинга представлен в приложении В.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	a	aaaaaaaaaa Interruption has been completed	Программа работает правильно
2.	.	……. Interruption has been completed	Программа работает правильно

Выводы.

Были изучены прерывания и их написание.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab5.asm

```
ind EQU 10
AStack SEGMENT STACK
          DW 512 DUP(?)
AStack ENDS

DATA SEGMENT
     symbol DB 0
     msg DB 10,13,'Interruption has been completed$'
     KEEP_CS DW 0
     KEEP_IP DW 0

DATA ENDS

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

interruption PROC FAR
print_loop:
         mov dl, symbol
         mov ah, 02h
         int 21h
         loop print_loop

         mov ah, 09h
         mov dx, offset msg
         int 21h
         mov al, 20h
         out 20h, al
         iret
interruption ENDP


Main    PROC FAR
           push ds
           push ax
           mov ax, DATA
           mov ds, ax
           mov cx, ind
           mov ah, 01h
           int 21h
           mov symbol, al
           mov ah, 02h
           mov dl, 10
           int 21h

           mov ah, 35h
           mov al, 08h
           int 21h
           mov KEEP_IP, bx
```

```
            mov KEEP_CS, es
            push ds
            mov dx, OFFSET interruption
            mov ax, SEG interruption
            mov ds, ax
            mov ah, 25h
            mov al, 08h
            int 21h
            pop ds

            int 08h
            cli
            push ds
            mov dx, KEEP_IP
            mov ax, KEEP_CS
            mov ds, ax
            mov ah, 25h
            mov al, 08h
            int 21h
            pop ds
            sti
            ret
Main    ENDP
CODE    ENDS
         END Main
```

ПРИЛОЖЕНИЕ В

Название файла: LAB5.LST

```
#Microsoft (R) Macro Assembler Version 5.10     11/22/22 01:40:5
                                                             Page
1-1


= 000A                  ind EQU 10
0000                 AStack SEGMENT STACK
0000  0200[                          DW 512 DUP(?)
   ????
           ]

0400                 AStack ENDS

0000                 DATA SEGMENT
0000  00                  symbol DB 0
0001  0A 0D 49 6E 74 65       msg  DB  10,13,'Interruption  has
been com
                 pleted$'
      72 72 75 70 74 69
      6F 6E 20 68 61 73
      20 62 65 65 6E 20
      63 6F 6D 70 6C 65
      74 65 64 24
0023  0000               KEEP_CS DW 0
0025  0000              KEEP_IP DW 0

0027                 DATA ENDS

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

0000                 interruption PROC FAR
0000                 print_loop:
0000  8A 16 0000 R                mov dl, symbol
0004  B4 02                      mov ah, 02h
0006  CD 21                      int 21h
0008  E2 F6                       loop print_loop

000A  B4 09                       mov ah, 09h
000C  BA 0001 R              mov dx, offset msg
000F  CD 21                       int 21h
0011  B0 20                        mov al, 20h
0013  E6 20                        out 20h, al
0015  CF                    iret
0016                 interruption ENDP


0016                 Main    PROC FAR
0016  1E                        push ds
0017  50                   push ax
0018  B8 ---- R                 mov ax, DATA
```

```
001B  8E D8                           mov ds, ax
001D  B9 000A                         mov cx, ind
0020  B4 01                           mov ah, 01h
0022  CD 21                         int 21h
0024  A2 0000 R                mov symbol, al
0027  B4 02                           mov ah, 02h
0029  B2 0A                           mov dl, 10
002B  CD 21                         int 21h
```



```
002D  B4 35                           mov ah, 35h
002F  B0 08                           mov al, 08h
0031  CD 21                           int 21h
0033  89 1E 0025 R                    mov KEEP_IP, bx
0037  8C 06 0023 R                    mov KEEP_CS, es
003B  1E                            push ds
003C  BA 0000 R                     mov dx, OFFSET interruption
003F  B8 ---- R                     mov ax, SEG interruption
0042  8E D8                           mov ds, ax
0044  B4 25                           mov ah, 25h
0046  B0 08                           mov al, 08h
0048  CD 21                           int 21h
004A  1F                            pop ds

004B  CD 08                           int 08h
004D  FA                            cli
004E  1E                            push ds
004F  8B 16 0025 R                    mov dx, KEEP_IP
0053  A1 0023 R                     mov ax, KEEP_CS
0056  8E D8                           mov ds, ax
0058  B4 25                           mov ah, 25h
005A  B0 08                           mov al, 08h
005C  CD 21                           int 21h
005E  1F                            pop ds
005F  FB                            sti
0060  CB                            ret
0061                  Main    ENDP
0061                  CODE    ENDS
                          END Main
```


Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0400	PARA	STACK	
CODE	0061	PARA	NONE	

```
    DATA . . . . . .      0027  PARA  NONE

    Symbols:

                    N a m e          Type   Value      Attr

    IND  . . . . . .      NUMBER     000A
    INTERRUPTION . . . . . .      F PROC     0000  CODE  Length
= 0016

    KEEP_CS  . . . . . .      L WORD     0023  DATA
    KEEP_IP  . . . . . .      L WORD     0025  DATA

    MAIN . . . . . .      F PROC     0016  CODE  Length
= 004B
    MSG  . . . . . .      L BYTE     0001  DATA

    PRINT_LOOP . . . . . .      L NEAR     0000  CODE

    SYMBOL . . . . . .      L BYTE     0000  DATA

    @CPU . . . . . .      TEXT  0101h
    @FILENAME  . . . . . .      TEXT  lab5
    @VERSION . . . . . .      TEXT  510


        74 Source  Lines
        74 Total   Lines
        16 Symbols

      48000 + 461307 Bytes symbol space free

         0 Warning Errors
         0 Severe  Errors
```