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: Создание собственных прерываний

Студент гр. 0383 _____ Бояркин Н.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучить механизм создания собственного прерывания.

Задание.

Вариант 8. (2а)

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.

Прерывание реализовано в процедуре SUBR_INT. В процедуре Main с помощью функции 35h/int 21h запоминается текущий вектор прерывания под номером 60h в переменные KEEP_CS, KEEP_IP. С помощью функции 25h/int 21h устанавливается новый вектор прерывания (реализованная процедура прерывания). Далее это прерывание вызывается в программе, предварительно в CX командой MOV заносится некоторое положительное число, соответствующее количеству вывода строки на экран. В конце программы вектор прерывания под номером 60h восстанавливается с помощью переменных KEEP_CS и KEEP_IP.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	mov cx, 5 int 60h	Hi there! Hi there! Hi there! Hi there! Hi there! End!	Верно
2.	mov cx, 7 int 60h	Hi there! Hi there! Hi there! Hi there! Hi there! Hi there! Hi there! End!	Верно

Выводы.

Был изучен механизм разработки собственных прерываний на языке Ассемблер.

Приложение А

Исходный код программы

Название файла: lab5.asm

```
DATA SEGMENT
        KEEP_CS DW 0 ; для хранения сегмента
        KEEP_IP DW 0 ; и смещения вектора прерывания
      TMP1 DW 0
      TMP2 DW 0
      TMP3 DW 0
        HELLO   DB 'Hi there!',10,13,'$'
      MESEND   DB 'End!',10,13,'$'

DATA ENDS


AStack    SEGMENT  STACK
          DW 12 DUP(?)
AStack    ENDS



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

SUBR_INT PROC FAR
        jmp start_proc

        ST_SS DW 0000
        ST_AX DW 0000
        ST_SP DW 0000
        IStack DW 30 DUP(?)
start_proc:

    mov ST_SP, SP
mov ST_AX, AX
mov AX, SS
mov ST_SS, AX
mov AX, IStack
mov SS, AX
mov AX, ST_AX
push ax
push ds

MOV   DX, OFFSET HELLO
MOV   AH,9
metka:
int   21h  ; Вызов функции DOS по прерыванию
loop metka ; Вывод сообщения заданное число раз

mov di,32
mov ah,0
```

```
int 1Ah
mov bx,dx; счетчик с момента сброса
Delay:
mov ah,0
int 1Ah
sub dx,bx
cmp di,dx
ja Delay;переход,если больше

MOV DX, OFFSET MESEND ;Выводсообщенияозавершении обработчика
MOV AH,9
int 21h
pop dx
    pop ax
mov ST_AX,AX
mov AX,ST_SS
mov SS,AX
mov SP,ST_SP
mov AX,ST_AX
mov al,20h
out 20h,al

iret
SUBR_INT ENDP


Main      PROC  FAR

        push  DS       ;\  Сохранение адреса начала PSP в стеке
        sub   AX,AX    ; > для последующего восстановления по
        push  AX       ;/  команде ret, завершающей процедуру.
        mov   AX,DATA             ; Загрузка сегментного
        mov   DS,AX               ; регистра данных.

        MOV AH, 35H ; функция получения вектора
        MOV AL, 60H ; номер вектора
        INT 21H ; возвращает текущее значение вектора прерывания
        MOV KEEP_IP, BX ; запоминание смещения
        MOV KEEP_CS, ES ; и сегмента вектора прерывания

        PUSH DS
        MOV DX, OFFSET SUBR_INT ; смещение для процедуры в DX
        MOV AX, SEG SUBR_INT ; 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

      mov cx, 10
        int 60H; вызов измененного прерывания

        CLI
```

```
            PUSH DS
            MOV DX, KEEP_IP
            MOV AX, KEEP_CS
            MOV DS, AX
            MOV AH, 25H
            MOV AL, 60H
            INT 21H ; восстанавливаем старый вектор прерывания
            POP DS
            STI

            RET
      Main    ENDP
      CODE    ENDS
              END Main
```

Название файла: lab5.lst

```
      Microsoft      (R)      Macro      Assembler      Version      5.10
12/8/21 22:23:25
                                                                    Page
1-1


       0000                 DATA SEGMENT
       0000  0000                        KEEP_CS DW 0 ; для хранения
                    сегмента
       0002  0000                        KEEP_IP DW 0 ; и смещения ве
                    ктора прерывания
       0004  0000                          TMP1 DW 0
       0006  0000                          TMP2 DW 0
       0008  0000                          TMP3 DW 0
       000A  48 65 6C 6C 6F 20                    HELLO       DB  'Hello
World!',10,13,'$'
             57 6F 72 6C 64 21
             0A 0D 24
       0019  45 6E 64 21 0A 0D              MESEND   DB 'End!',10,13,'$'
             24

       0020                 DATA ENDS


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

       0018                 AStack    ENDS



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

       0000                 SUBR_INT PROC FAR
       0000  EB 43 90                    jmp start_proc
```

```
0003  0000                        ST_SS DW 0000
0005  0000                        ST_AX DW 0000
0007  0000                        ST_SP DW 0000
0009  001E[                       IStack DW 30 DUP(?)
  ????
          ]

0045                start_proc:

0045  2E: 89 26 0007 R              mov ST_SP, SP
004A  2E: A3 0005 R           mov ST_AX, AX
004E  8C D0                   mov AX, SS
0050  2E: A3 0003 R           mov ST_SS, AX
0054  2E: A1 0009 R           mov AX, IStack
0058  8E D0                   mov SS, AX
005A  2E: A1 0005 R           mov AX, ST_AX
005E  50                 push ax
005F  1E                 push ds

0060  BA 000A R          MOV   DX, OFFSET HELLO
0063  B4 09                   MOV   AH,9
0065                     metka:
```



```
0065  CD 21                   int   21h  ; Вызов функции
              DOS по прерыванию
0067  E2 FC                   loop metka ; Вывод сообщен
              ия заданное число раз

0069  BF 0020                 mov di,32
006C  B4 00                   mov ah,0
006E  CD 1A                   int 1Ah
0070  8B DA                   mov bx,dx; счетчик с момен
              та сброса
0072                Delay:
0072  B4 00                   mov ah,0
0074  CD 1A                   int 1Ah
0076  2B D3                   sub dx,bx
0078  3B FA                   cmp di,dx
007A  77 F6                   ja Delay;переход,если бол
              ьше

007C  BA 0019 R          MOV DX, OFFSET MESEND ;Выводсоо
              бщенияозавершении обрабо
              тчика
007F  B4 09                   MOV AH,9
0081  CD 21                   int 21h
0083  5A                 pop dx
```

```
0084  58                pop ax
0085  2E: A3 0005 R            mov ST_AX,AX
0089  2E: A1 0003 R            mov AX,ST_SS
008D  8E D0                    mov SS,AX
008F  2E: 8B 26 0007 R              mov SP,ST_SP
0094  2E: A1 0005 R            mov AX,ST_AX
0098  B0 20                    mov al,20h
009A  E6 20                    out 20h,al

009C  CF                 iret
009D              SUBR_INT ENDP


009D              Main      PROC  FAR

009D  1E                    push  DS       ;\  Сохранение
                адреса начала PSP в стеке
009E  2B C0                      sub   AX,AX    ; > для послед
                ющего восстановления по
00A0  50                    push  AX       ;/  команде ret,
                завершающей процедуру.
00A1  B8 ---- R             mov   AX,DATA             ; Загру
                ка сегментного
00A4  8E D8                       mov    DS,AX                ;
регис
                ра данных.

00A6  B4 35                      MOV AH, 35H ; функция получ
                ения вектора
00A8  B0 60                      MOV AL, 60H ; номер вектора
00AA  CD 21                      INT 21H ; возвращает теку
     Microsoft      (R)      Macro      Assembler      Version      5.10
12/8/21 22:23:25
                                                                Page
1-3


                щее значение вектора прер
                ывания
00AC  89 1E 0002 R               MOV KEEP_IP, BX ; запоминани
                  смещения
00B0  8C 06 0000 R               MOV KEEP_CS, ES ; и сегмента
                ектора прерывания

00B4  1E                    PUSH DS
00B5  BA 0000 R             MOV DX, OFFSET SUBR_INT ; смещен
                е для процедуры в DX
00B8  B8 ---- R             MOV AX, SEG SUBR_INT ; сегмент
                роцедуры
00BB  8E D8                      MOV DS, AX ; помещаем в DS
00BD  B4 25                      MOV AH, 25H ; функция устан
                овки вектора
00BF  B0 60                      MOV AL, 60H ; номер вектора
```

```
 00C1  CD 21                        INT 21H ; меняем прерыван
                ие
 00C3  1F                    POP DS

 00C4  B9 000A                        mov cx, 10
 00C7  CD 60                        int 60H; вызов измененно
                o прерывания

 00C9  FA                    CLI
 00CA  1E                    PUSH DS
 00CB  8B 16 0002 R                 MOV DX, KEEP_IP
 00CF  A1 0000 R             MOV AX, KEEP_CS
 00D2  8E D8                        MOV DS, AX
 00D4  B4 25                        MOV AH, 25H
 00D6  B0 60                        MOV AL, 60H
 00D8  CD 21                        INT 21H ; восстанавливае
                  старый вектор прерывани
                
 00DA  1F                    POP DS
 00DB  FB                    STI

 00DC  CB                    RET
 00DD                Main      ENDP
 00DD                CODE      ENDS
                         END Main
     Microsoft      (R)      Macro      Assembler      Version      5.10
12/8/21 22:23:25

Symbols-1


     Segments and Groups:

                N a m e         Length     Align     Combine
Class

     ASTACK . . . . . .      0018  PARA  STACK
     CODE . . . . . .        00DD  PARA  NONE
     DATA . . . . . .        0020  PARA  NONE

     Symbols:

                N a m e         Type  Value      Attr

     DELAY  . . . . . .      L NEAR     0072  CODE

     HELLO  . . . . . .      L BYTE     000A  DATA

     ISTACK . . . . . .      L WORD     0009  CODE  Length
= 001E

     KEEP_CS  . . . . . .    L WORD     0000  DATA
     KEEP_IP  . . . . . .    L WORD     0002  DATA
```

```
     MAIN . . . . . .      F PROC     009D  CODE  Length
= 0040
     MESEND . . . . . .    L BYTE     0019  DATA
     METKA  . . . . . .    L NEAR     0065  CODE

     START_PROC . . . . .  L NEAR     0045  CODE
     ST_AX  . . . . . .    L WORD     0005  CODE
     ST_SP  . . . . . .    L WORD     0007  CODE
     ST_SS  . . . . . .    L WORD     0003  CODE
     SUBR_INT . . . . . .  F PROC     0000  CODE  Length
= 009D

     TMP1 . . . . . .      L WORD     0004  DATA
     TMP2 . . . . . .      L WORD     0006  DATA
     TMP3 . . . . . .      L WORD     0008  DATA

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


         115 Source  Lines
         115 Total   Lines
          24 Symbols

       48014 + 459246 Bytes symbol space free

           0 Warning Errors
           0 Severe  Errors
```

Приложение Б

Тестирование

Если результаты тестирования велики (больше 1 страницы), то их выносят в приложение.

Процесс тестирования можно представить в виде таблицы, например:

Таблица Б.2 - Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарии
3.			
4.			
5.			
...			

Обратите внимание, что в нумерации таблицы в приложении обязательно должен быть в качестве префикса номер самого приложения: А.