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. 0382 _____ Чегодаева Е.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучить прерывания на языке Ассемблера, создать собственное прерывание.

Задание.

Вариант 28: Написать собственное прерывание с номером 16h, которое будет выполнять ввод и печать заданного количества символов, после чего выедет сообщение о завершении обработчика.

Выполнение работы.

Процедура myINT:

В сегменте кода реализован обработчик прерывания: при помощи функций 01h и 09h осуществляется ввод и вывод сообщения пользователя, заданной длины (3 – установлено в сегменте данных). Переменные KEEP_CS и KEEP_IP хранят адрес сегмента и смещения собственного прерывания. Вместе с этим инициализирован стек – MyStack, используемый внутри данного блока программы. Процедура обработки прерывания оканчивается командами для возможности обработки прерываний с более низким уровнями, чем данное.

Основная процедура:

Включает в себя изменение назначения заданного вектора прерывания, ввод символа, вызывающего прерывание (по заданному скан-коду клавиши “s”), а также восстановление старого вектора прерывания в конце программы.

В KEEP_CS и KEEP_IP записываются соответствующие данные полученного (при помощи функции 35h и прерывания 21h) вектора. Далее указываются адрес сегмента и смещения процедуры myINT; затем (посредством функции 25h) устанавливается изменённое прерывания. Считывание скан-кода заданной клавиши реализовано в блоке run, после выхода из которого (что означает то, что необходимый символ был введён) осуществляется вызов собственного прерывания. С помощью функции 25h и прерывания 21h реализовано восстановление вектора прерывания до первоначального состояния.

Исходный код программы см. в приложении А.

Файл листинга см. в приложении В.

Тестирование.

Результаты тестирования представлены в таблице 1.

Таблица 1 – результаты тестирования.

№	Входные данные	Выходные данные	Комментарии
1	S abc	abc DONE	Верно
2	S fff	fff DONE	Верно
3	S lol	lol DONE	Верно

Выводы.

Были изучены прерывания на языке Ассемблера и создано собственное прерывание: назначение вектора прерывания с номером 16h было заменено на следующее - выполнять ввод и печать заданного количества символов и затем вывод сообщения о завершении обработчика.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

Название файла: lb5.asm

```
AStack SEGMENT STACK
 DW 512 DUP(0)
AStack ENDS

DATA SEGMENT
      KEEP_CS DW 0 ; хранение сегмента прерывания
      KEEP_IP DW 0 ; хранение смещения прерывания
      n equ 3
      str DB 0Dh, 0Ah, n dup(0), '$'
      str_end DB 0Dh, 0Ah,'DONE', '$'
DATA ENDS

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

myINT PROC FAR
      jmp next
      KEEP_SS DW 0
      KEEP_SP DW 0
      MyStack DW 100 dup(0)
next:
      mov KEEP_SP, SP
      mov KEEP_SS, SS
      mov AX, SEG MyStack
      mov SS, SP
      mov SP, offset next

      push AX
      push CX
      push DX
      mov cx, n
      mov di, offset str
      add di, 2
input:
      mov ah, 01h
      int 21h
      mov [di], al
      add di, 1
      loop input

      mov ah, 09h ;вывод строки
      mov dx, offset str
      int 21h
      mov dx, offset str_end
      int 21h
      pop AX
      pop DX
      pop CX
      mov SS, KEEP_SS
      mov SP, KEEP_SP
```

```
        mov AL, 20h
        out 20h,AL
        iret
myINT ENDP

Main PROC FAR
      push DS
      sub AX,AX
      push AX
      mov AX, DATA
      mov DS, AX

      MOV AH,35h ; функция получения вектора
      MOV AL,17h ; номер вектора
      INT 21h
      MOV KEEP_IP, BX ; запоминание смещения
      MOV KEEP_CS, ES ; и сегмента

      PUSH DS
      MOV DX, offset myINT ; смещение для процедуры в DX
      MOV AX, seg myINT ; сегмент процедуры
      MOV DS, AX ; помещаем в DS
      MOV AH, 25h ; функция установки вектора
      MOV AL, 17h ; номер вектора
      INT 21h ; меняем прерывание
      POP DS
run:
      mov ah, 0h
      int 16h
      cmp al,'s'
      jne run

      int 17h ;!!!!

      CLI
      PUSH DS
      MOV DX, KEEP_IP
      MOV AX, KEEP_CS
      MOV DS, AX
      MOV AH, 25H
      MOV AL, 17h
      INT 21H ;восстанавливаем вектор
      POP DS
      STI
      MOV AH, 4CH
      INT 21H
      RET
MAIN ENDP
CODE ENDS
      END MAIN
```

ПРИЛОЖЕНИЕ В

ФАЙЛ ЛИСТНГА

Название файла: lb5.lst

```
Microsoft (R) Macro Assembler Version 5.10                  12/1/21
01:00:46
                                                             Page     1-1


 0000                   DATA SEGMENT
 0000  0000                     old_seg dw 0
 0002  0000                     old_ip dw 0
 0004  41 20 76 65 72 79        out_msg DB 'A very informative message...
$'
       20 69 6E 66 6F 72
       6D 61 74 69 76 65
       20 6D 65 73 73 61
       67 65 2E 2E 2E 24
 0022  57 6F 72 6B 20 69        end_msg DB 'Work is done!$'
       73 20 64 6F 6E 65
       21 24
 0030                   DATA ENDS

 0000                   AStack SEGMENT STACK
 0000  0200[                    DW 512 DUP(?)
         ????
                    ]

 0400                   AStack ENDS

 0000                   CODE SEGMENT
                            ASSUME CS:CODE, DS:DATA, SS:AStack
                        ; ds:dx must contain message adress ends with '
                        $'
                        ; cx must contain number of prints
                        ; bx must contain time (in cpu ticks)
                        ; data segment must contain 'end_msg' string
 0000                   CUSTOM_INT PROC FAR
                            ;storing registers
 0000  50                   push ax
 0001  53                   push bx
 0002  51                   push cx
 0003  52                   push dx

                            ;print cx times
 0004  B4 09                     mov ah, 9h
 0006                   print_loop:
 0006  CD 21                     int 21h
 0008  E2 FC                     loop print_loop

                            ;pause
 000A  B4 00                     mov ah, 0
 000C  CD 1A                     int 1Ah
 000E  03 DA                     add bx, dx
```

```
0010                    pause:
0010  B4 00                      mov ah, 0
0012  CD 1A                      int 1Ah
0014  3B DA                      cmp bx, dx
0016  7F F8                      jg pause

                            ;printing end message
0018  BA 0022 R             mov dx, offset end_msg
001B  B4 09                      mov ah, 9h
001D  CD 21                      int 21h
```



```
                            ;restoring registers
001F  5A                    pop dx
0020  59                    pop cx
0021  5B                    pop bx
0022  58                    pop ax

                            ;return
0023  B0 20                      mov al, 20h
0025  E6 20                      out 20h, al
0027  CF                    iret
0028                    CUSTOM_INT ENDP

0028                    Main PROC FAR
0028  1E                    push DS
0029  2B C0                      sub ax, ax
002B  50                    push ax
002C  B8 ---- R             mov ax, DATA
002F  8E D8                      mov ds, ax

                            ;storing old int
0031  B8 3560                    mov ax, 3560h
0034  CD 21                      int 21h
0036  8C 06 0000 R               mov old_seg, es
003A  89 1E 0002 R               mov old_ip, bx

                            ;setting custom int
003E  1E                    push ds
003F  BA 0000 R             mov dx, offset CUSTOM_INT
0042  B8 ---- R             mov ax, seg CUSTOM_INT
0045  8E D8                      mov ds, ax
0047  B8 0023                    mov ax, 23h
004A  CD 21                      int 21h
004C  1F                    pop ds

                            ;setting registers according to custom int
                        manual
004D  BA 0004 R             mov dx, offset out_msg
0050  B9 0010                    mov cx, 10h
0053  BB 0036                    mov bx, 36h
0056  CD 60                      int 60h
```

```
                        ;restoring old int
 0058  FA               CLI
 0059  1E               push ds
 005A  8B 16 0002 R            mov dx, old_ip
 005E  A1 0000 R         mov ax, old_seg
 0061  8E D8                   mov ds, ax
 0063  B8 0023                 mov ax, 23h
 0066  CD 21                   int 21h
 0068  1F               pop ds
 0069  FB               STI

 006A  CB               ret

Microsoft (R) Macro Assembler Version 5.10                  12/1/21
01:00:46
                                                             Page     1-3


 006B                Main ENDP

 006B                CODE ENDS
                          END Main


Microsoft (R) Macro Assembler Version 5.10                  12/1/21
01:00:46
                                                             Symbols-1


Segments and Groups:

                N a m e         Length      Align     Combine Class

ASTACK . . . . . .              0400  PARA  STACK
CODE . . . . . .                006B  PARA  NONE
DATA . . . . . .                0030  PARA  NONE

Symbols:

                N a m e         Type   Value      Attr

CUSTOM_INT . . . . . .          F PROC     0000  CODE  Length = 0028

END_MSG  . . . . . .            L BYTE     0022  DATA

MAIN . . . . . .                F PROC     0028  CODE  Length = 0043

OLD_IP . . . . . .              L WORD     0002  DATA
OLD_SEG  . . . . . .            L WORD     0000  DATA
OUT_MSG  . . . . . .            L BYTE     0004  DATA

PAUSE  . . . . . .              L NEAR     0010  CODE
PRINT_LOOP . . . . . .          L NEAR     0006  CODE

@CPU . . . . . .                TEXT  0101h
@FILENAME  . . . . . .          TEXT  lb
```

```
@VERSION . . . . . .     TEXT  510


    101 Source  Lines
    101 Total   Lines
     16 Symbols

  48028 + 461279 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```