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 печать строки, после ввода заданного символа.

Выполнение работы.

Процедура myINT:

В сегменте кода реализован обработчик прерывания: при помощи функций 09h осуществляется вывод сообщения, заданного ранее (установлено в сегменте данных). При считывании скан-кода, соответствующего символу ‘s’ – работа программы переходит в блок print, который и выполнят вывод строки. Считывание реализовано при помощи взаимодействия с портом 60h внутри цикла.

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, 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. 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	DONE	Верно
2	S	DONE	Верно
3	S	DONE	Верно

Выводы.

Были изучены прерывания на языке Ассемблера и создано собственное прерывание: назначение вектора прерывания с номером 16h было заменено на следующее - выполнять печать заданной строки после введения соответствующего символа.

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
      str DB '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 dx

input:
      in al, 60h
      cmp al, 1fh
      jne input

print:
      mov ah, 09h ;вывод строки
      mov dx, offset str
      int 21h
final:
      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,16h ; 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, 16h ; номер вектора
      INT 21h ; меняем прерывание
      POP DS

      int 16h

      CLI
      PUSH DS
      MOV DX, KEEP_IP
      MOV AX, KEEP_CS
      MOV DS, AX
      MOV AH, 25H
      MOV AL, 16h
      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                  12/12/21
13:14:2
                                                             Page     1-1


 0000                   AStack SEGMENT STACK
 0000  0200[                   DW 512 DUP(0)
         0000
                ]

 0400                   AStack ENDS

 0000                   DATA SEGMENT
 0000  0000                        KEEP_CS DW 0 ; C…CЂP°PSPµPSPёPµ CЃPµPiP
                        jPµPSC,P° PїCЂPµCЂC‹PIP°PSPёCЏ
 0002  0000                        KEEP_IP DW 0 ; C…CЂP°PSPµPSPёPµ CЃPjPµC
                        ‰PµPSPёCЏ PїCЂPµCЂC‹PIP°PSPёCЏ
 0004  44 4F 4E 45 24              str DB 'DONE$'
 0009                   DATA ENDS

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

 0000                   myINT PROC FAR
 0000  E9 00CF R              jmp next
 0003  0000                        KEEP_SS DW 0
 0005  0000                        KEEP_SP DW 0
 0007  0064[                       MyStack DW 100 dup(0)
         0000
                ]

 00CF                   next:
 00CF  2E: 89 26 0005 R                  mov KEEP_SP, SP
 00D4  2E: 8C 16 0003 R                  mov KEEP_SS, SS
 00D9  B8 ---- R              mov AX, SEG MyStack
 00DC  8E D4                       mov SS, SP
 00DE  BC 00CF R              mov SP, offset next
 00E1  50                     push ax
 00E2  52                     push dx

 00E3                   input:
 00E3  E4 60                       in al, 60h
 00E5  3C 1F                       cmp al, 1fh
 00E7  75 FA                       jne input

 00E9                   print:
 00E9  B4 09                       mov ah, 09h ;PIC‹PIPsPґ CЃC,CЂPsPєPё
 00EB  BA 0004 R              mov dx, offset str
 00EE  CD 21                       int 21h
 00F0                   final:
```

```
 00F0  58                  pop ax
 00F1  5A                  pop dx
 00F2  59                  pop cx
 00F3  2E: 8E 16 0003 R              mov SS, KEEP_SS
 00F8  2E: 8B 26 0005 R              mov SP, KEEP_SP
 00FD  B0 20                       mov AL, 20h
 00FF  E6 20                       out 20h,AL
 0101  CF                    iret


Microsoft (R) Macro Assembler Version 5.10                  12/12/21
13:14:2
                                                             Page     1-2


 0102                 myINT ENDP

 0102                 Main PROC FAR
 0102  1E                  push DS
 0103  2B C0                    sub AX,AX
 0105  50                  push AX
 0106  B8 ---- R           mov AX, DATA
 0109  8E D8                    mov DS, AX

 010B  B4 35                    MOV AH,35h ; C„CґPSPєC†PëCЏ PïPsP»CґC‡P
                      µPSPëCЏ PIPµPєC,PsCЋP°
 010D  B0 16                    MOV AL,16h ; PSPsPjPµCЋ PIPµPєC,PsCЋP°
 010F  CD 21                    INT 21h
 0111  89 1E 0002 R             MOV KEEP_IP, BX ; P·P°PïPsPjPëPSP°PSPëP
                      µ CЃPjPµC‰PµPSPëCЏ
 0115  8C 06 0000 R             MOV KEEP_CS, ES ; Pë CЃPµPiPjPµPSC,P°

 0119  1E                  PUSH DS
 011A  BA 0000 R           MOV DX, offset myINT ; CЃPjPµC‰PµPSPëPµ
                       PґP»CЏ PïCЋPsC†PµPґCґCЋC‹ PI DX
 011D  B8 ---- R           MOV AX, seg myINT ; CЃPµPiPjPµPSC, PïCЋ
                      PsC†PµPґCґCЋC‹
 0120  8E D8                    MOV DS, AX ; PïPsPjPµC‰P°PµPj PI DS
 0122  B4 25                    MOV AH, 25h ; C„CґPSPєC†PëCЏ CґCЃC,P°PS
                      PsPIPєPë PIPµPєC,PsCЋP°
 0124  B0 16                    MOV AL, 16h ; PSPsPjPµCЋ PIPµPєC,PsCЋP°
 0126  CD 21                    INT 21h ; PjPµPSCЏPµPj PïCЋPµCЋC‹PIP°PS
                      PëPµ
 0128  1F                  POP DS

 0129  CD 16                    int 16h

 012B  FA                  CLI
 012C  1E                  PUSH DS
 012D  8B 16 0002 R             MOV DX, KEEP_IP
 0131  A1 0000 R           MOV AX, KEEP_CS
 0134  8E D8                    MOV DS, AX
 0136  B4 25                    MOV AH, 25H
 0138  B0 16                    MOV AL, 16h
 013A  CD 21                    INT 21H  ;PIPsCЃCЃC,P°PSP°PIP»PëPIP°PµP
                      j PIPµPєC,PsCЋ
 013C  1F                  POP DS
```

```
 013D  FB                  STI
 013E  B4 4C                    MOV AH, 4CH
 0140  CD 21                    INT 21H
 0142  CB                  RET
 0143                MAIN ENDP
 0143                CODE ENDS
                          END MAIN

Microsoft (R) Macro Assembler Version 5.10                  12/12/21
13:14:2
                                                             Symbols-1


Segments and Groups:

                N a m e         Length     Align    Combine Class

ASTACK . . . . . .              0400  PARA  STACK
CODE . . . . . .                0143  PARA  NONE
DATA . . . . . .                0009  PARA  NONE

Symbols:

                N a m e         Type   Value     Attr

FINAL  . . . . . .              L NEAR    00F0  CODE

INPUT  . . . . . .              L NEAR    00E3  CODE

KEEP_CS  . . . . . .            L WORD    0000  DATA
KEEP_IP  . . . . . .            L WORD    0002  DATA
KEEP_SP  . . . . . .            L WORD    0005  CODE
KEEP_SS  . . . . . .            L WORD    0003  CODE

MAIN . . . . . .                F PROC    0102  CODE  Length = 0041
MYINT  . . . . . .              F PROC    0000  CODE  Length = 0102
MYSTACK  . . . . . .            L WORD    0007  CODE  Length = 0064

NEXT . . . . . .                L NEAR    00CF  CODE

PRINT  . . . . . .              L NEAR    00E9  CODE

STR  . . . . . .                L BYTE    0004  DATA

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


     88 Source  Lines
     88 Total   Lines
     20 Symbols

  48030 + 461277 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```