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: Разработка собственного прерывания

Студент(ка) гр. 1381 _____ Денисова О.К.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы

Разработать собственное прерывание

Общая формулировка задачи

Вариант 6, задание 2C

2 - 60h - прерывание пользователя - должно генерироваться в программе

C - Приостановить вывод на экран (вставить цикл задержки).

Выполнение работы

В ходе выполнения лабораторной работы была написана программа, вызывающая созданное в рамках задания прерывание.

На стеке, согласно заданию, выделяется 1Кб памяти.

В процедуре main устанавливаем вектор прерывания с помощью команды mov ah, 25h, которая устанавливает значение элемента таблицы векторов прерываний для прерывания с номером al, равным ds:dx. На момент вызова прерывания 21h в ds лежит SEG SUBR_INT, в dx OFFSET SUBR_INT, а в al - 60h. С помощью int 21h меняем прерывание.

Далее в процедуре main устанавливаем в dx смещение строки, которую будем выводить, в ah устанавливаем 9, чтобы выводить строку. Выводим один раз, вызываем прерывание, выводим второй раз.

В процедуре SUBR_INT устанавливаем ax в 0, затем уменьшаем значение ax на 1 с помощью dec. На первом шаге ax становится 65535, и каждый последующий шаг уменьшается на 1, пока не станет снова равным 0. Далее уменьшаем cl на 1 (изначально cl устанавливается в 100). Повторяем всё вышеописанное, пока cl не станет равным 0. Таким образом мы выполняем 65535*100 проходов.

Далее происходит выход из прерывания, восстановление старого прерывания и выход из программы.

Выводы

В ходе выполнения лабораторной работы были приобретены знания о прерываниях, и приобретен опыт написания собственного прерывания на языке Ассемблера.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ
ФАЙЛ LR4.ASM

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

DATA SEGMENT
     keep_cs dw 0
     keep_ip dw 0
Str_1  LABEL  BYTE
     DB '******************',13,10,'$'
DATA ENDS

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

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

     mov ah, 35h
     mov al, 60h
     int 21h
     mov keep_ip, bx
     mov keep_cs, es

     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  ax, SEG DATA              ; Загрузка в DS адреса начала
     mov  ds, ax                    ; сегмента данных
     mov   dx, OFFSET Str_1             ; Загрузка в dx смещения
текста
Display:
     mov  ah, 9                     ; # функции ДОС печати строки
     int  21h

     mov ah, 100
     int 60h

     mov  ah, 9                     ; # функции ДОС печати строки
     int  21h


     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

      mov ah, 4ch
      int 21h

Main ENDP

SUBR_INT PROC FAR
      push ax
      push cx
      mov cl, ah
l_2:
      mov ax, 0
l_1:
      dec ax
      jne l_1
      dec cl
      jne l_2

      mov al, 20h
      out 20h, al
      pop cx
      pop ax
      iret
SUBR_INT ENDP

CODE ENDS
 END Main
```

ПРИЛОЖЕНИЕ Б
ФАЙЛ ЛИСТИНГА
ФАЙЛ LR4.LST

```
Microsoft (R) Macro Assembler Version 5.10                  11/15/22
00:45:3
                                                             Page
1-1


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

0400                    AStack ENDS

0000                    DATA SEGMENT
0000  0000                      keep_cs dw 0
0002  0000                      keep_ip dw 0
0004                    Str_1  LABEL  BYTE
0004  2A 2A 2A 2A 2A 2A         DB '******************',13,10,'$'
      2A 2A 2A 2A 2A 2A
      2A 2A 2A 2A 2A 2A
      2A 0D 0A 24
001A                    DATA ENDS

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

0000                    Main PROC FAR
0000  1E                      push DS
0001  2B C0                        sub AX,AX
0003  50                        push AX
0004  B8 ---- R                 mov AX,DATA
0007  8E D8                        mov DS,AX

0009  B4 35                        mov ah, 35h
000B  B0 60                        mov al, 60h
000D  CD 21                        int 21h
000F  89 1E 0002 R                 mov keep_ip, bx
0013  8C 06 0000 R                 mov keep_cs, es

0017  1E                        PUSH ds
0018  BA 0052 R                 mov dx, OFFSET SUBR_INT ; CЃPjPµC‰PµPSP
                        ёPµ PrP»CЏ PїCЂPsC†PµPrCѓCЂC‹ PI DX
001B  B8 ---- R                 mov ax, SEG SUBR_INT ; CЃPµPiPjPµPSC, P
                        їCЂPsC†PµPrCѓCЂC‹
001E  8E D8                        mov ds, AX ; PїPsPjPµC‰P°PµPj PI DS
0020  B4 25                        mov   ah,   25h   ;   C„CѓPSPєC†PёCЏ
CѓCЃC,P°PS
                        PsPIPєPё PIPµPєC,PsCЂP°
0022  B0 60                        mov    al,    60h    ;    PSPsPjPµCЂ
PIPµPєC,PsCЂP°
```

```
      0024  CD 21                     int      21h      ;      PjPµPSCЏPµPj
PïCЂPµCЂC‹ PIP°PS
                                PëPµ
      0026  1F                  pop ds

      0027  B8 ---- R           mov  ax, SEG DATA                ; P—P°P
                                iCЂCґP·PєP° PI DS P°PґCЂPµCЃP° PSP°C‡P°P»P°
      002A  8E D8                       mov  ds, ax                         ;
CЃPµP
                                iPjPµPSC,P° PґP°PSPSC‹C…
      002C  BA 0004 R           mov  dx, OFFSET Str_1            ; P—P°P
                                iCЂCґP·PєP°      PI      dx      CЃPjPµC‰PµPSPëCЏ
C,PµPєCЃC,P°
      002F                      Display:
      002F  B4 09                       mov  ah, 9                          ; #
C„C



Microsoft (R) Macro Assembler Version 5.10                      11/15/22
00:45:3
                                                                     Page
1-2


                                ґPSPєC†PëPë P”PћPЎ PїPµC‡P°C,Pë CЃC,CЂPsPєPë
      0031  CD 21                       int  21h

      0033  B4 64                       mov ah, 100
      0035  CD 60                       int 60h

      0037  B4 09                       mov  ah, 9                          ; #
C„C
                                ґPSPєC†PëPë P”PћPЎ PїPµC‡P°C,Pë CЃC,CЂPsPєPë
      0039  CD 21                       int  21h


      003B  FA                  CLI
      003C  1E                  push ds
      003D  8B 16 0002 R                mov dx, KEEP_IP
      0041  A1 0000 R           mov ax, KEEP_CS
      0044  8E D8                       mov ds, ax
      0046  B4 25                       mov ah, 25h
      0048  B0 60                       mov al, 60h
      004A  CD 21                       int              21h                ;
PIPsCЃCЃC,P°PSP°PIP»PëPIP°PµP
                                j           CЃC,P°CЂC‹ P№           PIPµPєC,PsCЂ
PïCЂPµCЂC‹ PIP°PSPëC
                                Џ
      004C  1F                  pop ds
      004D  FB                  STI

      004E  B4 4C                       mov ah, 4ch
      0050  CD 21                       int 21h

      0052                      Main ENDP
```

```
0052                    SUBR_INT PROC FAR
0052  50                      push ax
0053  51                      push cx
0054  8A CC                         mov cl, ah
0056                    l_2:
0056  B8 0000                     mov ax, 0
0059                    l_1:
0059  48                      dec ax
005A  75 FD                         jne l_1
005C  FE C9                         dec cl
005E  75 F6                         jne l_2

0060  B0 20                         mov al, 20h
0062  E6 20                         out 20h, al
0064  59                      pop cx
0065  58                      pop ax
0066  CF                      iret
0067                    SUBR_INT ENDP

0067                    CODE ENDS
                         END Main
```

Microsoft (R) Macro Assembler Version 5.10 11/15/22 00:45:3

Symbols-1

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0400	PARA	STACK	
CODE	0067	PARA	NONE	
DATA	001A	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
DISPLAY	L NEAR	002F	CODE
KEEP_CS	L WORD	0000	DATA
KEEP_IP	L WORD	0002	DATA
L_1	L NEAR	0059	CODE
L_2	L NEAR	0056	CODE
MAIN	F PROC	0000	CODE Length = 0052
STR_1	L BYTE	0004	DATA
SUBR_INT	F PROC	0052	CODE Length = 0015

```
@CPU . . . . . .     TEXT  0101h
@FILENAME  . . . . . .     TEXT  lr4
@VERSION . . . . . .     TEXT  510


     87 Source  Lines
     87 Total   Lines
     16 Symbols

  48008 + 446675 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```