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 гр. 1381 _____ Тарасов К.О.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы

Написать программу обработки прерывания

Задание

Вариант 1а:

1 - 08h - прерывание от системного таймера - генерируется автоматически операционной системой 18 раз в сек;

A - Выполнить вывод сообщения на экран заданное число раз,после чего вставить фиксированную задержку и вывести сообщение о завершении обработчика.

Ход работы

Прерывание реализовано в процедуре SUBR_INT. В процедуре Main с помощью функции 35h/int 21h запоминается текущий вектор прерывания под номером 08h 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. Вывод сообщения несколько раз реализован при помощи инструкции loop. Задержка перед выводом сообщения End реализована с помощью int 15h. В конце программы вектор прерывания восстанавливается с помощью переменных KEEP_CS и KEEP_IP

Тестирование

Табл. 1. Результат тестирования.

Вызванные команды	Результат	Комментарий
mov cx, 3	Sample text Sample text Sample text End	Верно
mov cx, 5	Sample text Sample text Sample text Sample text Sample text End	Верно

mov cx, 7	Sample text Sample text Sample text Sample text Sample text Sample text Sample text End	Верно

Выводы:

В ходе выполнения работы было разработано собственное прерывание на языке Ассемблер.

ПРИЛОЖЕНИЕ А

Текст компонентов программы lab5

lab5.asm

```
DATA SEGMENT
	KEEP_CS DW 0
	KEEP_IP DW 0
		TMP1 DW 0
		TMP2 DW 0
		TMP3 DW 0
	HELLO   DB 'Sample text',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

		save_SP DW 0000h
		save_SS DW 0000h
		INT_STACK DB 40 DUP(0)

start_proc:

   MOV save_SP, SP
```

```
        MOV save_SS, SS
        MOV SP, SEG INT_STACK
        MOV SS, SP
        MOV SP, offset start_proc
        PUSH AX
        PUSH DX

        MOV   DX, OFFSET HELLO
        MOV   AH,9
        metka:
        int   21h
        loop metka

        mov     ah,86h
        xor     cx, cx
        mov     dx, 30000
        int     15h

        MOV   DX, OFFSET MESEND
        MOV AH,9
        int 21h

        POP  DX
        POP  AX
        MOV  SS, save_SS
        MOV  SP, save_SP
        MOV  AL, 20H

        OUT  20H,AL

        iret

SUBR_INT ENDP
```

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


       MOV  AH, 35H
       MOV  AL, 08H
       INT  21H
       MOV  KEEP_IP, BX
       MOV  KEEP_CS, ES


       PUSH DS
       MOV  DX, OFFSET SUBR_INT
       MOV  AX, SEG SUBR_INT
       MOV  DS, AX
       MOV  AH, 25H
       MOV  AL, 08H
       INT  21H
       POP  DS

       mov cx, 3
       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

        MOV AH, 4Ch
        INT 21h
Main    ENDP
CODE ENDS
        END Main
```

lab5.lst

```
#Microsoft (R) Macro Assembler Version 5.10              11/22/22 01:41:1
                                                        Page     1-1


 0000                           DATA SEGMENT
 0000  0000                         KEEP_CS DW 0
 0002  0000                         KEEP_IP DW 0
 0004  0000                                 TMP1 DW 0
 0006  0000                                 TMP2 DW 0
 0008  0000                                 TMP3 DW 0
 000A  53 61 6D 70 6C 65            HELLO   DB 'Sample text',10,13,'$'
     20 74 65 78 74 0A
     0D 24
 0018  45 6E 64 0A 0D 24                    MESEND   DB 'End',10,13,'$'

 001E                           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 2D 90                       JMP start_proc

0003  0000                            save_SP DW 0000h
0005  0000                            save_SS DW 0000h
0007  0028[                           INT_STACK DB 40 DUP(0)
        00
                    ]


002F                    start_proc:

002F  2E: 89 26 0003 R                MOV save_SP, SP
0034  2E: 8C 16 0005 R                    MOV save_SS, SS
0039  BC ---- R                   MOV SP, SEG INT_STACK
003C  8E D4                       MOV SS, SP
003E  BC 002F R                   MOV SP, offset start_proc
0041  50                          PUSH AX
0042  52                          PUSH DX

0043  BA 000A R                   MOV   DX, OFFSET HELLO
0046  B4 09                       MOV   AH,9
0048                              metka:
0048  CD 21                       int   21h
004A  E2 FC                       loop metka

004C  B4 86                       mov     ah,86h
004E  33 C9                       xor     cx, cx
```

```
0050  BA 7530                       mov     dx, 30000
0053  CD 15                    int       15h

0055  BA 0018 R                 MOV   DX, OFFSET MESEND
0058  B4 09                    MOV AH,9
005A  CD 21                    int 21h

005C  5A                       POP  DX
005D  58                       POP  AX
005E  2E: 8E 16 0005 R                 MOV  SS, save_SS
0063  2E: 8B 26 0003 R                 MOV  SP, save_SP
0068  B0 20                    MOV  AL, 20H

006A  E6 20                    OUT  20H,AL

006C  CF                       iret

006D                      SUBR_INT ENDP


006D                      Main   PROC  FAR
006D  1E                       push  DS
006E  2B C0                    sub   AX, AX
0070  50                       push  AX
0071  B8 ---- R                mov   AX,DATA
0074  8E D8                    mov   DS,AX


0076  B4 35                    MOV  AH, 35H
0078  B0 08                    MOV  AL, 08H
007A  CD 21                    INT  21H
```

```
007C  89 1E 0002 R            MOV  KEEP_IP, BX
0080  8C 06 0000 R            MOV  KEEP_CS, ES


0084  1E                      PUSH DS
0085  BA 0000 R               MOV  DX, OFFSET SUBR_INT
0088  B8 ---- R               MOV  AX, SEG SUBR_INT
008B  8E D8                   MOV  DS, AX
008D  B4 25                   MOV  AH, 25H
008F  B0 08                   MOV  AL, 08H
0091  CD 21                   INT  21H
0093  1F                     POP  DS

0094  B9 0003                    mov cx, 3
0097  CD 08                  int 08H

0099  FA                     CLI
009A  1E                      PUSH DS
009B  8B 16 0002 R            MOV  DX, KEEP_IP
009F  A1 0000 R               MOV  AX, KEEP_CS
00A2  8E D8                   MOV  DS, AX
00A4  B4 25                   MOV  AH, 25H
00A6  B0 08                   MOV  AL, 08H
```

```
00A8  CD 21                     INT  21H
00AA  1F                        POP  DS
00AB  FB                        STI

00AC  B4 4C                     MOV AH, 4Ch
00AE  CD 21                     INT 21h
00B0                    Main    ENDP
00B0                    CODE ENDS
                                END Main
```

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA	STACK	
CODE	00B0	PARA	NONE	
DATA	001E	PARA	NONE	

Symbols:

N a m e	Type	Value	Attr
HELLO	L BYTE	000A	DATA
INT_STACK	L BYTE	0007	CODE Length = 0028
KEEP_CS	L WORD	0000	DATA
KEEP_IP	L WORD	0002	DATA
MAIN	F PROC	006D	CODE Length = 0043
MESEND	L BYTE	0018	DATA
METKA	L NEAR	0048	CODE
SAVE_SP	L WORD	0003	CODE
SAVE_SS	L WORD	0005	CODE
START_PROC	L NEAR	002F	CODE
SUBR_INT	F PROC	0000	CODE Length = 006D
TMP1	L WORD	0004	DATA
TMP2	L WORD	0006	DATA
TMP3	L WORD	0008	DATA

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


   109 Source  Lines
   109 Total   Lines
    22 Symbols

 48014 + 461293 Bytes symbol space free

     0 Warning Errors
     0 Severe  Errors
```