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

Студентка гр. 0383 _____ Александрович В.П.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Разработать собственное прерывание для программы.

Задание.

Вариант 1

1 . 08h - прерывание от часов - 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.

Выполнение работы.

Прерывание было реализовано в процедуре SUBR_INT. В процедуре Main с помощью функции 35h/int 21h запоминается текущий вектор прерывания под номером 08h. С помощью функции 25h/int 21h устанавливается новый вектор прерывания, реализованная в рамках данной лабораторной работы. Далее в регистр cx заносится положительное число – количество раз, которые сообщение будет выведено на экран. Далее прерывание вызывается в программе. Вывод сообщения несколько раз реализовано с помощью инструкции loop. Задержка реализована с помощью функции 86h/int 15h.

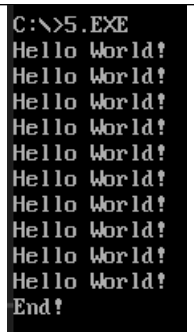
Разработанный программный код см. в приложении А.

Текст файла диагностических сообщений см. в приложении Б.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

Входные данные	Выходные данные	Комментарий
mov cx, 5		ВЕРНО
mov cx, 10		ВЕРНО

Выводы.

В ходе выполнения данной лабораторной работы была изучена работа с прерываниями на языке Ассемблер. Было разработано собственное прерывание.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab5.asm

```
DATA SEGMENT
        KEEP_CS DW 0
        KEEP_IP DW 0
           TMP1 DW 0
           TMP2 DW 0
           TMP3 DW 0
        HELLO   DB 'Hello World!',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, 10
      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
```

ПРИЛОЖЕНИЕ Б

ФАЙЛ ДИАГНОСТИЧЕСКИХ СООБЩЕНИЙ

Название файла: lab5.lst

```
Microsoft (R) Macro Assembler Version 5.10                  12/15/21
19:55:4
                                                              Page
1-1


       0000                 DATA SEGMENT
       0000  0000                          KEEP_CS  DW  0  ;  для  хранения
сегмента
       0002  0000                           KEEP_IP  DW  0  ;  и  смещения
вектора преры
                            вания
       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 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


Microsoft (R) Macro Assembler Version 5.10                  12/15/21
19:55:4
                                                              Page
1-2


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 0019 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


                            ; Запоминание текущего вектора прерыван
                      ия
0076  B4 35                    MOV   AH, 35H    ; функция получения
вект
                      ора
0078  B0 08                    MOV  AL, 08H   ; номер вектора
007A  CD 21                    INT  21H
007C  89 1E 0002 R             MOV   KEEP_IP,  BX    ;  запоминание
смещени
```

```
                        я
0080  8C 06 0000 R             MOV  KEEP_CS, ES  ; и сегмента

                              ; Установка вектора прерывания
0084  1E                      PUSH DS
0085  BA 0000 R               MOV  DX, OFFSET SUBR_INT ; смещение для
                         процедуры в DX
0088  B8 ---- R               MOV  AX, SEG SUBR_INT    ; сегмент проц
                        едуры
008B  8E D8                        MOV  DS, AX            ; помещаем в
DS
008D  B4 25                        MOV   AH, 25H             ; функция
установк
                        и вектора
008F  B0 08                        MOV   AL, 08H              ; номер
вектора
0091  CD 21                        INT   21H                 ; меняем
прерывани
                        е
0093  1F                      POP  DS



Microsoft (R) Macro Assembler Version 5.10                  12/15/21
19:55:4
                                                              Page
1-3


0094  B9 000A                      mov cx, 10
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          ; восстанавливаем
век
                        тор
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


Microsoft (R) Macro Assembler Version 5.10                  12/15/21
19:55:4
                                                              Symbol
s-1
```

```
Segments and Groups:

                N a m e         Length      Align     Combine Class

ASTACK . . . . . .              0018  PARA  STACK
CODE . . . . . .                00B0  PARA  NONE
DATA . . . . . .                0020  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     0019  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  _5
@VERSION . . . . . .            TEXT  510


    110 Source  Lines
    110 Total   Lines
     22 Symbols

  48040 + 461267 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```