

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

Морева Е. С.

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 функции 09h осуществляется вывод сообщения, заданного установлено в сегменте данных. При считывании скан-кода, соответствующего символу 'esc' – работа программы переходит в блок print, который и выполняет 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 с помощью 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 | esc | GOOD | Верно |
| 2 | esc | GOOD | Верно |
| 3 | esc | GOOD | Верно |

Выводы.

Были изучены прерывания на языке Ассемблера и создано собственное прерывание.

ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММ

```
AStack SEGMENT STACK
```

```
DW 1024 DUP(0)
```

```
AStack ENDS
```

```
DATA SEGMENT
```

```
    string DB 'GOOD$'
```

```
    ; хранение сегмента прерывания
```

```
    KEEP_CS DW 0
```

```
    ; хранение смещения прерывания
```

```
    KEEP_IP DW 0
```

```
DATA ENDS
```

```
CODE SEGMENT
```

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

```
userInterrupt PROC FAR
```

```
    jmp userData
```

KEEP_SS DW 0

KEEP_SP DW 0

procStack DW 100 dup(0)

userData:

mov KEEP_SP, SP

mov KEEP_SS, SS

mov AX, SEG userInterrupt

mov SS, SP

mov SP, offset userData

push ax

push dx

input:

in al, 60h

cmp al, 01h

jne input

;ВЫВОД строки

mov ah, 09h

mov dx, offset string

int 21h

pop ax

pop dx

mov SS, KEEP_SS

mov SP, KEEP_SP

mov AL, 20h

out 20h,AL

iret

userInterrupt ENDP

Main PROC FAR

push DS

mov AX, DATA

mov DS, AX

xor ax,ax

MOV AH,35h ; функция получения вектора

MOV AL,16h ; номер вектора

INT 21h

; запоминание смещения

MOV KEEP_IP, BX

; и сегмента

MOV KEEP_CS, ES

PUSH DS

MOV DX, offset userInterrupt ; смещение для процедуры в DX

MOV AX, seg userInterrupt ; 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

;очистка флага IF (CLEAR IF)

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

;установка флага IF(SET IF)
STI

MOV AH, 4CH

INT 21H

RET

MAIN ENDP

CODE ENDS

END MAIN

```

ПРИЛОЖЕНИЕ В ФАЙЛ ЛИСТНГА

Microsoft (R) Macro Assembler Version 5.10

12/17/21 24:06:0

Page 1-1

```

0000          AStack SEGMENT STACK
0000 0400[          DW 1024 DUP(0)
0000
          ]

```



```

0800                                AStack ENDS

0000                                DATA SEGMENT

0000 47 4F 4F 44 24                string DB 'GOOD$'
                                ; C...CᵀP°PSPμPSPëPμ CÍPμPiPjPμPSC,P°
PïC

                                ᵀPμCᵀCᵀPIP°PSPëCᵀ

0005 0000                        KEEP_CS DW 0
                                ; C...CᵀP°PSPμPSPëPμ
CÍPjPμC%oPμPSPëCᵀ PïC

                                ᵀPμCᵀCᵀPIP°PSPëCᵀ

0007 0000                        KEEP_IP DW 0

0009                                DATA ENDS

0000                                CODE SEGMENT

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

0000                                userInterrupt PROC FAR

0000 E9 00CF R                    jmp userData

0003 0000                        KEEP_SS DW 0

0005 0000                        KEEP_SP DW 0

0007 0064[                        procStack DW 100 dup(0)

                                0000

```

]

```
00CF          userData:
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 userInterrupt
00DC 8E D4              mov SS, SP
00DE BC 00CF R          mov SP, offset userData
00E1 50                push ax
00E2 52                push dx

00E3          input:
00E3 E4 60            in al, 60h
00E5 3C 01            cmp al, 01h
00E7 75 FA            jne input

;PIC\PIPsPr CÍC,CṪPsPcPë

00E9 B4 09            mov ah, 09h
00EB BA 0000 R        mov dx, offset string
00EE CD 21            int 21h

00F0 58                pop ax
```

00F1 5A pop dx

00F2 2E: 8E 16 0003 R mov SS, KEEP_SS

00F7 2E: 8B 26 0005 R mov SP, KEEP_SP

Microsoft (R) Macro Assembler Version 5.10

12/17/21 24:06:0

Page 1-2

00FC B0 20 mov AL, 20h

00FE E6 20 out 20h,AL

0100 CF iret

0101 userInterrupt ENDP

0101 Main PROC FAR

0101 1E push DS

0102 B8 ---- R mov AX, DATA

0105 8E D8 mov DS, AX

0107 33 C0 xor ax,ax

0109 B4 35 MOV AH,35h ; C,,CfPSPeC†PëCŮ
PĩPsP»CŕC‡P

μPSPëCİİ PIPμPeC,PsCṪP°

010B B0 16 MOV AL,16h ; PSPsPjPμCṪ
PIPμPeC,PsCṪP°

010D CD 21 INT 21h

; P·P°PİPsPjPëPSP°PSPëPμ
CÍPjPμC%oPμPSPë

Cİİ

010F 89 1E 0007 R MOV KEEP_IP, BX

; Pë CÍPμPiPjPμPSC,P°

0113 8C 06 0005 R MOV KEEP_CS, ES

0117 1E PUSH DS

0118 BA 0000 R MOV DX, offset userInterrupt ; CÍPjPμC%o

PμPSPëPμ PrP»Cİİ PİCṪPsC†PμPrCíCṪC< PI DX

011B B8 ---- R MOV AX, seg userInterrupt ; CÍPμPiPjPμP

SC, PİCṪPsC†PμPrCíCṪC<

011E 8E D8 MOV DS, AX ; PİPsPjPμC%oP°PμPj PI
DS

0120 B4 25 MOV AH, 25h ; C,,CíPSPeC†PëCİİ
CíCÍC,P°PS

PsPIPePë PIPμPeC,PsCṪP°

0122 B0 16 MOV AL, 16h ; PSPsPjPμCṪ
PIPμPeC,PsCṪP°

| | | |
|------|--------------|--|
| 0124 | CD 21 | INT 21h ; PjPμPSClPμPj |
| | | PiCτPμCτC<PIP°PS |
| | | PëPμ |
| 0126 | 1F | POP DS |
| 0127 | CD 16 | int 16h |
| | | ;PsC‡PëCíC,PëP° C,,P»P°PiP° IF (CLEAR IF |
| | |) |
| 0129 | FA | CLI |
| 012A | 1E | PUSH DS |
| 012B | 8B 16 0007 R | MOV DX, KEEP_IP |
| 012F | A1 0005 R | MOV AX, KEEP_CS |
| 0132 | 8E D8 | MOV DS, AX |
| 0134 | B4 25 | MOV AH, 25H |
| 0136 | B0 16 | MOV AL, 16h |
| 0138 | CD 21 | INT 21H |
| | | ;PIPsCíCíC,P°PSP°PIP»PëPIP°PμP |
| | | j PIPμPëC,PsCτ |
| 013A | 1F | POP DS |
| | | ;CíCíC,P°PSPsPIPeP° C,,P»P°PiP° IF(SET I |
| | | F) |

```

013B FB          STI
013C B4 4C          MOV AH, 4CH
013E CD 21          INT 21H
0140 CB          RET
0141          MAIN ENDP
0141          CODE ENDS

          END MAIN

```

Microsoft (R) Macro Assembler Version 5.10

12/17/21 24:06:0

Symbols-1

Segments and Groups:

| N a m e | Length | Align | Combine | Class |
|--------------|--------|-------|---------|-------|
| ASTACK | 0800 | PARA | | STACK |
| CODE | 0141 | PARA | | NONE |
| DATA | 0009 | PARA | | NONE |

Symbols:

| N a m e | Type | Value | Attr | |
|---------------------|--------|-------|------|---------------|
| INPUT | L NEAR | 00E3 | CODE | |
| KEEP_CS | L WORD | 0005 | DATA | |
| KEEP_IP | L WORD | 0007 | DATA | |
| KEEP_SP | L WORD | 0005 | CODE | |
| KEEP_SS | L WORD | 0003 | CODE | |
| MAIN | F PROC | 0101 | CODE | Length = 0040 |
| PROCSTACK | L WORD | 0007 | CODE | Length = 0064 |
| STRING | L BYTE | 0000 | DATA | |
| USERDATA | L NEAR | 00CF | CODE | |
| USERINTERRUPT | F PROC | 0000 | CODE | Length = 0101 |
| @CPU | TEXT | 0101h | | |
| @FILENAME | TEXT | lab5 | | |
| @VERSION | TEXT | 510 | | |

98 Source Lines

98 Total Lines

18 Symbols

48016 + 461291 Bytes symbol space free

0 Warning Errors

0 Severe Errors