

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МО ЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №5**  
**по дисциплине «Операционные системы»**  
**ТЕМА: Сопряжение стандартного и пользовательского**  
**обработчиков прерывания**

Студентка гр. 7383

\_\_\_\_\_

Рудоман В.А.

Преподаватель

\_\_\_\_\_

Ефремов М.А.

Санкт-Петербург

2019

## 1. Цель лабораторной работы

Исследование возможности встраивания пользовательского обработчика прерываний в стандартный обработчик от клавиатуры. Пользовательский обработчик прерывания получает управление по прерыванию (int 09h) при нажатии клавиши на клавиатуре. Он обрабатывает скан-код и осуществляет определенные действия, если скан-код совпадает с определенными кодами, которые он должен обрабатывать. Если скан-код не совпадает с этими кодами, то управление передается стандартному прерыванию.

## 2. Постановка задачи

Шаг 1. Для выполнения лабораторной работы необходимо написать и отладить программный модуль типа .EXE, который выполняет такие же функции, как в программе ЛР 4, а именно:

Проверяет, установлено ли пользовательское прерывание с вектором 09h.

Если прерывание не установлено то, устанавливает резидентную функцию для обработки прерывания и настраивает вектор прерываний. Адрес точки входа в стандартный обработчик прерывания находится в теле пользовательского обработчика. Осуществляется выход по функции 4Ch прерывания int 21h.

Если прерывание установлено, то выводится соответствующее сообщение и осуществляется выход по функции 4Ch прерывания int 21h.

Выгрузка прерывания по соответствующему значению параметра в командной строке /un. Выгрузка прерывания состоит в восстановлении стандартного вектора прерываний и освобождении памяти, занимаемой резидентом. Затем осуществляется выход по функции 4Ch прерывания int 21h.

Для того чтобы проверить установку прерывания, можно поступить следующим образом. Прочитать адрес, записанный в векторе прерывания. Предположим, что этот адрес указывает на точку входа в установленный резидент. На определенном, известном смещении в теле резидента располагается сигнатура, некоторый код, который идентифицирует резидент. Сравнив известное значение сигнатуры с реальным кодом, находящимся в резиденте, можно определить, установлен ли резидент. Если значения совпадают, то резидент установлен. Длину кода сигнатуры должна быть достаточной, чтобы сделать случайное совпадение маловероятным.

Программа должна содержать код устанавливаемого прерывания в виде удаленной процедуры. Этот код будет работать после установки при возникновении прерывания. Он должен выполнять следующие функции:

Сохранить значения регистров в стеке при входе и восстановить их при выходе.

При выполнении тела процедуры анализируется скан-код.

Если этот код совпадает с одним из заданных, то требуемый код записывается в буфер клавиатуры.

Если этот код не совпадает ни с одним из заданных, то осуществляется передача управления стандартному обработчику прерывания.

Шаг 2. Запустите отлаженную программу и убедитесь, что резидентный обработчик прерывания 09h установлен. Работа прерывания проверяется введением различных символов, обрабатываемых установленным обработчиком и стандартным обработчиком.

Шаг 3. Также необходимо проверить размещение прерывания в памяти. Для этого запустите программу ЛР 3, которая отображает карту памяти в виде с писка блоков МСВ. Полученные результаты поместите в отчет.

Шаг 4. Запустите отлаженную программу еще раз и убедитесь, что программа определяет установленный обработчик прерываний. Полученные результаты поместите в отчет.

Шаг 5. Запустите отлаженную программу с ключом выгрузки и убедитесь, что резидентный обработчик прерывания выгружен, то есть сообщения на экран не выводятся, а память, занятая резидентом освобождена. Для этого также следует запустить программу ЛР 3. Полученные результаты поместите в отчет.

Оформить отчёт и ответить на контрольные вопросы.

Процедуры, которые используются в программе.

ROUT	Функция обработчика прерывания.
outputAL	Функция вывода символа из AL.
outputBP	Функция вывода строки по адресу ES:BP на экран
comprasionCode	Функция сравнения скан-кода с заданными.
conversionCodeInSymbol	Функция преобразования скан-кода в символ.
PRINT	Функция печати на экран.

Переменные, которые используются в программе.

resUnload	db	Применяется для вывода информации о том, что резидент выгружен из памяти.
resAlrLoad	db	Применяется для вывода информации о том, что резидент уже загружен в память.
resLoad	db	Применяется для вывода информации о том, что резидент загружен в память.
isLoad	db	Флаг указывающий на то, что программа должна быть загружена в память.
isUnload	db	Флаг указывающий на то, что программа

		должна быть выгружена из памяти.
KEEP_IP	dw	Применяется для запоминания смещения вектора прерывания.
KEEP_CS	dw	Применяется для запоминания сегмента вектора прерывания.
ID	dw	Переменная предназначенная для проверки загружен ли резидент в память или нет.
SCAN_CODE	db	Массив предназначенный для хранения скан-кодов клавиш клавиатуры.

### Ход выполнения работы

Происходит считывание номера клавиши, после чего, если нажатая клавиша является цифрой, то заменяем ее на соответствующую букву английского алфавита в лексикографическом порядке.

*Шаг 1:* Пример выполнения работы программы представлен на рисунке №1:

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Assembling file: 1.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 471k

C:\>TASM.EXE 5.ASM
Turbo Assembler Version 3.1 Copyright (c) 1988, 1992 Borland International

Assembling file: 5.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 469k

C:\>TLINK.EXE 5.OBJ
Turbo Link Version 5.1 Copyright (c) 1992 Borland International

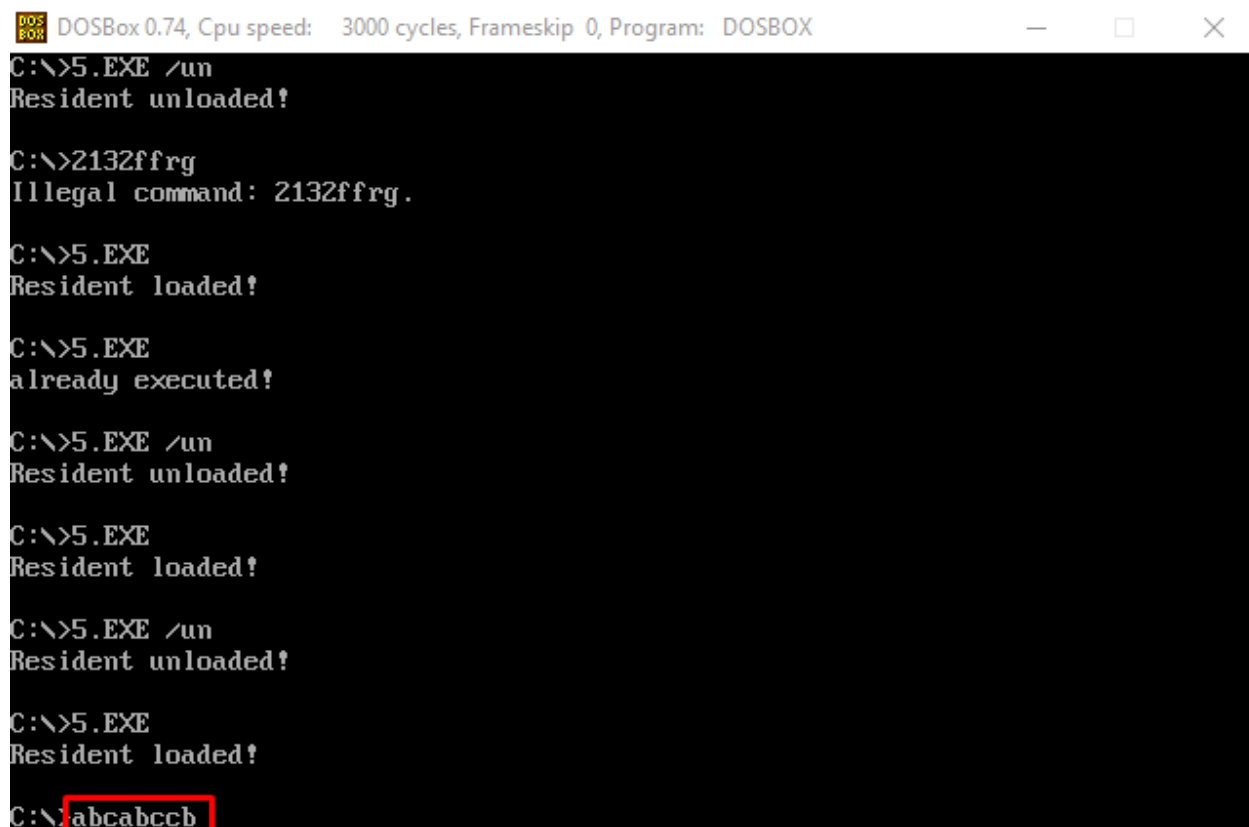
C:\>5.EXE
Resident loaded!

C:\>_

```

Рис. 1 - Состояние памяти до выполнения разработанного модуля

Шаг 2: Пример выполнения программы представлен на рисунках №2-3:



DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

```
C:\>5.EXE /un
Resident unloaded!

C:\>2132ffrg
Illegal command: 2132ffrg.

C:\>5.EXE
Resident loaded!

C:\>5.EXE
already executed!

C:\>5.EXE /un
Resident unloaded!

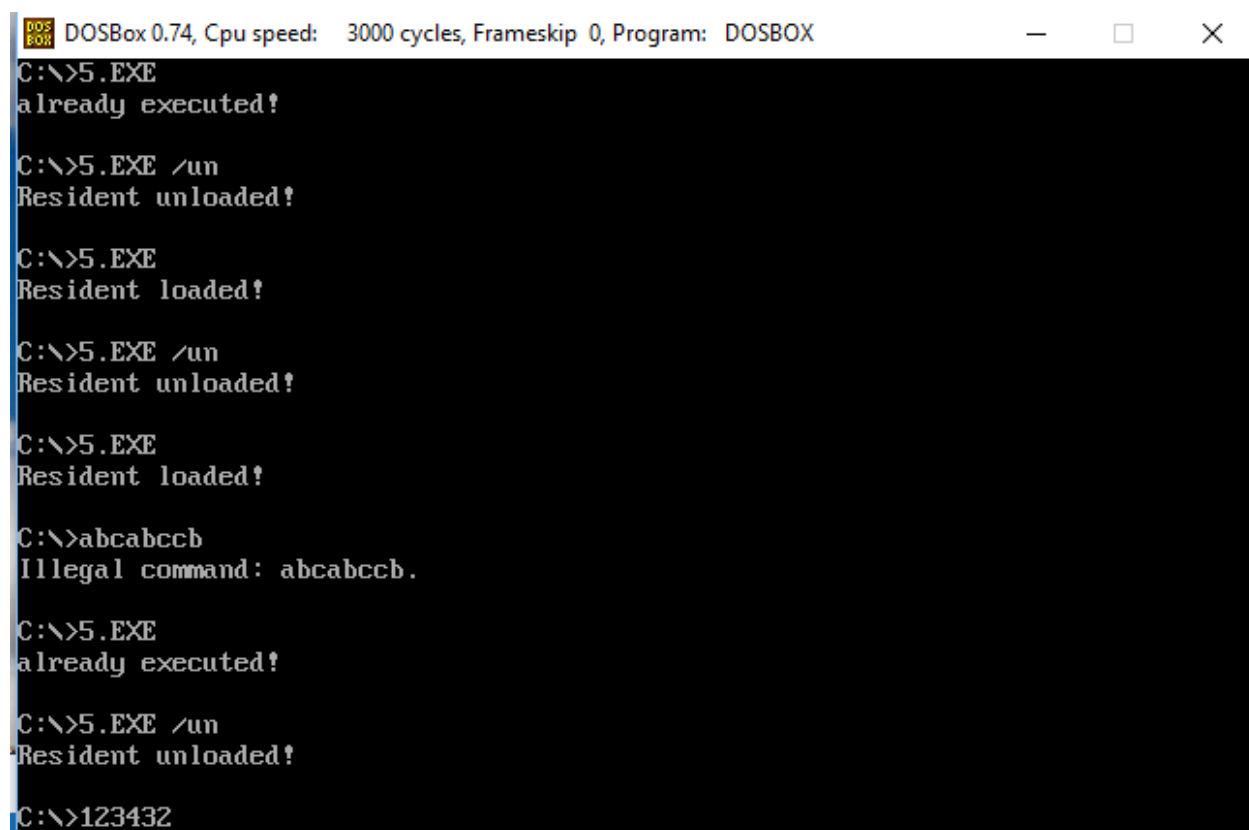
C:\>5.EXE
Resident loaded!

C:\>5.EXE /un
Resident unloaded!

C:\>5.EXE
Resident loaded!

C:\>abcabccb
```

Рис. 2 – Пример работы программы



DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

```
C:\>5.EXE
already executed!

C:\>5.EXE /un
Resident unloaded!

C:\>5.EXE
Resident loaded!

C:\>5.EXE /un
Resident unloaded!

C:\>5.EXE
Resident loaded!

C:\>abcabccb
Illegal command: abcabccb.

C:\>5.EXE
already executed!

C:\>5.EXE /un
Resident unloaded!

C:\>123432
```

Рис.3 – Пример ввода после выгрузки резидента

Шаг 3: Пример выполнения программы представлен на рисунках №4-5:

```
Resident loaded!

C:\>LAB31.COM

Size of available memory: 648224 byte
Freeing mermory...
Suscess!
Size of extended memory: 245760 byte
MCB #01
Addr: 016F      Owner: Area belongs to MS DOS Size: 16 byte Name:
MCB #02
Addr: 0171      Owner: Empty area Size: 64 byte Name:
MCB #03
Addr: 0176      Owner: 0040          Size: 256 byte Name:
MCB #04
Addr: 0187      Owner: 0192          Size: 144 byte Name:
MCB #05
Addr: 0191      Owner: 0192          Size: 512 byte Name: 5
MCB #06
Addr: 01B2      Owner: 01BD          Size: 144 byte Name:
MCB #07
Addr: 01BC      Owner: 01BD          Size: 1840 byte Name: LAB31
MCB #08
Addr: 0230      Owner: Empty area Size: 646368 byte Name:
```

Рис. 4 - Результат работы пользовательского прерывания

```
C:\>5.EXE /un
Resident unloaded!

C:\>LAB31.COM

Size of available memory: 648912 byte
Freeing mermory...
Suscess!
Size of extended memory: 245760 byte
MCB #01
Addr: 016F      Owner: Area belongs to MS DOS Size: 16 byte Name:
MCB #02
Addr: 0171      Owner: Empty area Size: 64 byte Name:
MCB #03
Addr: 0176      Owner: 0040          Size: 256 byte Name:
MCB #04
Addr: 0187      Owner: 0192          Size: 144 byte Name:
MCB #05
Addr: 0191      Owner: 0192          Size: 1840 byte Name: LAB31
MCB #06
Addr: 0205      Owner: Empty area Size: 647056 byte Name: <  ѵ8♦♦0
```

Рис. 5 - Состояние памяти после выгрузки резидента

## **Контрольные вопросы**

*1. Какого типа прерывания использовались в работе?*

Были использованы пользовательские прерывания, такие как `int 10h` и `int 21h` и аппаратные прерывания (`1Ch`).

*2. Чем отличается скан код от кода ASCII?*

Код ASCII – это код символа из таблицы ASCII, а скан-код – код, который определяется нажатием клавиши или комбинации клавиш, который передаётся клавиатурой.

## **Вывод**

В результате выполнения данной лабораторной работы были исследованы организация и управление прерываниями. Была написана программа, в которой построен обработчик прерываний.