

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

**ОТЧЕТ**  
**по практической работе №5**  
**по дисциплине «Операционные системы»**  
**Тема: Сопряжение стандартного и пользовательского обработчиков**  
**прерываний**

Студент гр. 8382

Торосян Т.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

## **Цель 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 (int 09h) при нажатии клавиши на клавиатуре. Он обрабатывает) 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.

## **Необходимые сведения для составления программы.**

Клавиатура содержит микропроцессор, который воспринимает каждое нажатие на клавишу и посылает скан-код в порт микросхемы интерфейса с периферией. Когда скан-код поступает в порт, то вызывается аппаратное прерывание клавиатуры (int 09h) при нажатии клавиши на клавиатуре. Он обрабатывает). Процедура обработки этого прерывания считывает номер клавиши из порта 60h, преобразует номер клавиши в соответствующий код, выполняет установку флагов в байтах состояния, загружает номер клавиши и полученный код в буфер клавиатуры.

В прерывании клавиатуры можно выделить три основных шага:

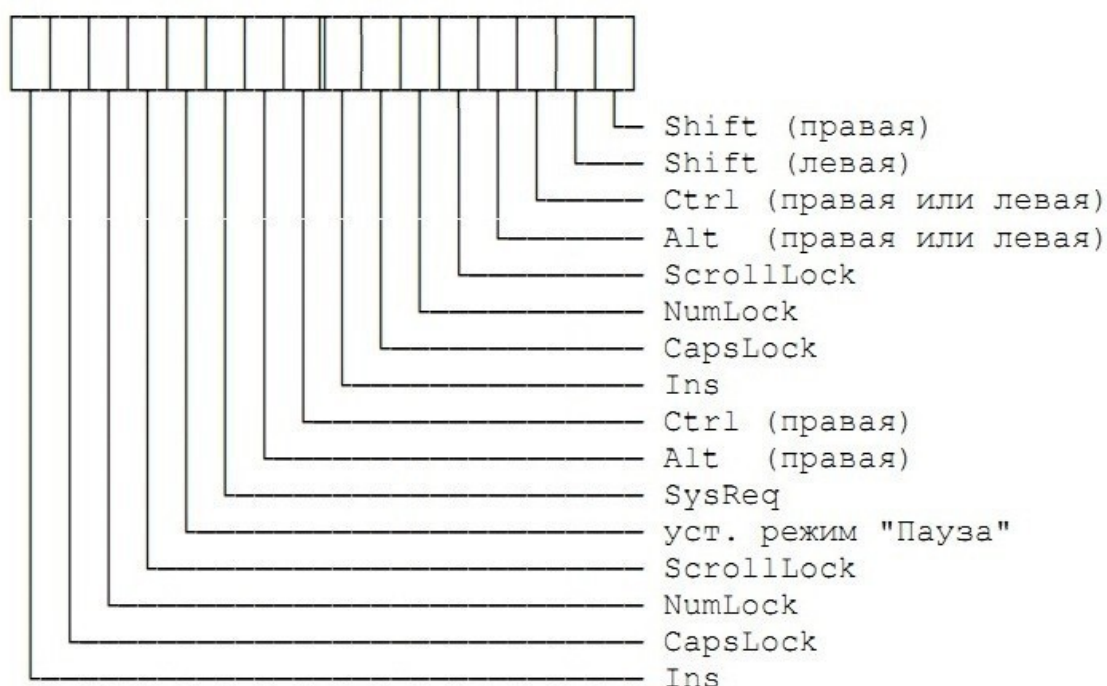
- 1) Прочитать скан-код и послать клавиатуре подтверждающий сигнал.
- 2) Преобразовать скан-код в номер кода или в установку регистра статуса клавиш- переключателей.
- 3) Поместить код клавиши в буфер клавиатуры.

Текущее содержимое буфера клавиатуры определяется указателями на начало и

Адрес в памяти	Размер в байтах	Содержимое
----------------	-----------------	------------

0040:001A	2	Адрес начала буфера
0040:001C	2	Адрес конца буфера
0040:001E	32	Буфер клавиатуры
0040:0017	2	Байты состояния

Флаги в байтах состояния устанавливаются в 1, если нажата соответствующая клавиша или установлен режим. Соответствие флагов и клавиш показано ниже.



Сначала скан-код анализируется на предмет того, была ли клавиша нажата (код нажатия) или отпущена (код освобождения). Код освобождения состоит из двух байтов: сначала OF0H, а затем скан-код. Все коды освобождения отбрасываются, кроме случая клавиш- переключателей, для которых делаются соответствующие изменения в байтах их статуса. С другой стороны, все коды нажатия обрабатываются. При этом опять могут измениться байты статуса клавиш- переключателей. В случае же символьных кодов, надо проверять байты статуса, чтобы определить, например, что сканкод 30 соответствует нижнему или верхнему регистру буквы А. После того как введенный символ идентифицирован, процедура ввода с клавиатуры должна найти соответствующий ему код ASCII или расширенный код. Приведенный пример слишком короток, чтобы рассмотреть все случаи.

В общем случае скан-коды со оставляются элементам таблицы данных, которая анализируется инструкцией XLAT. XLAT принимает в AL число от 0 до 255, а возвращает в AL 1-байтное значение из 256-байтной таблицы, на которую указывает DS:BX. Таблица может находиться в сегменте данных. Если в AL находился скан-код 30, то туда будет помещен из таблицы байт номер 30 (31й байт, так как отсчет начинается с нуля). Этот байт в таблице должен быть установлен равным 97, давая код ASCII для "а". Конечно для получения заглавной А нужна другая таблица, к которой обращение будет происходить, если статус сдвига установлен. Или заглавные буквы могут храниться в другой части той же таблицы, но в этом случае к скан-коду надо будет добавлять смещение, определяемое статусом клавиш-переключателей.

Номера кодов должны быть помещены в буфер клавиатуры. Процедура должна сначала проверить, имеется ли в буфере место для следующего символа. Буфер устроен как циклическая очередь. Ячейка памяти 0040:001A содержит указатель на голову буфера, а 0040:001C - указатель на хвост. Эти словные указатели дают смещение в области данных BIOS (которая начинается в сегменте 40H) и находятся в диапазоне от 30 до 60. Новые символы вставляются в ячейки буфера с более старшими адресами, а когда достигнута верхняя граница, то следующий символ переносится в нижний конец буфера. Когда буфер полон, то указатель хвоста на 2 меньше указателя на голову - кроме случая, когда указатель на голову равен 30 (начало области буфера), а в этом случае буфер полон, когда указатель хвоста равен 60. Для вставки символа в буфер, надо поместить его в позицию, на которую указывает хвост буфера и затем увеличить указатель хвоста на 2; если указатель хвоста был равен 60, то надо изменить его значение на 30.

Код для отработки прерывания 09H push ax

```
in al,60H ;читать ключчитать ключ cmp al,REQ KEY
          ;читать ключэто требуемый код? je do-req
          ;читать ключ да, активизировать обработку REQ
KEY
```

```

;читать ключ нет, уйти на исходный
обработчик pop ax
jmp cs:[int9_vect] ;читать ключпереход на первоначальный
обработчик
do_req:
;читать ключследующий код необходим для отработки
аппаратного прерывания in al,61H ;читать ключвзять значение
порта управления клавиатурой
mov,ah,al or ;читать ключсохранить его
al, 8 0h out ;читать ключустановить бит разрешения для
клавиатуры
61H,al xchg ;читать ключи вывести его в управляющий
порт
ah,al ;читать ключизвлечь исходное значение
порта
;читать ключпослать сигнал "конец
prerывания"
mov al,20H
out 20H,al ;читать ключ контроллеру прерываний 8259

;читать ключ дальше - прочие проверки

```

**Записать символ в буфер клавиатуры можно с помощью функции 05h прерывания 16h:**

```

mov ah,05h ;читать ключ Код функции
mov cl,'D' ;читать ключ Пишем символ в буфер
клавиатуры mov ch,00h ;читать ключ int
16h ;читать ключ
or al,al ;читать ключ проверка переполнения буфера
jnz skip ;читать ключ если переполнен идем skip ;читать
ключ работать дальше skip: ;читать ключ очистить буфер и
повторить

```

### **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 /up. Выгрузка прерывания состоит в восстановлении стандартного вектора прерываний и освобождении памяти, занимаемой резидентом. Затем осуществляется выход по функции 4Ch int21h.

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 списка 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 используемые в программе.**

Таблица 1 - Процедуры используемые в программе

Название процедуры	Назначение
ROUT	Функция обработчика прерывания
IS_INT_L	Проверка установки резидента
IS_FLAG_UN	Проверка команды '/un'
INT_LOAD	Загрузка резидента
INT_UNLOAD	Выгрузка резидента
PRINT	Вывод на экран

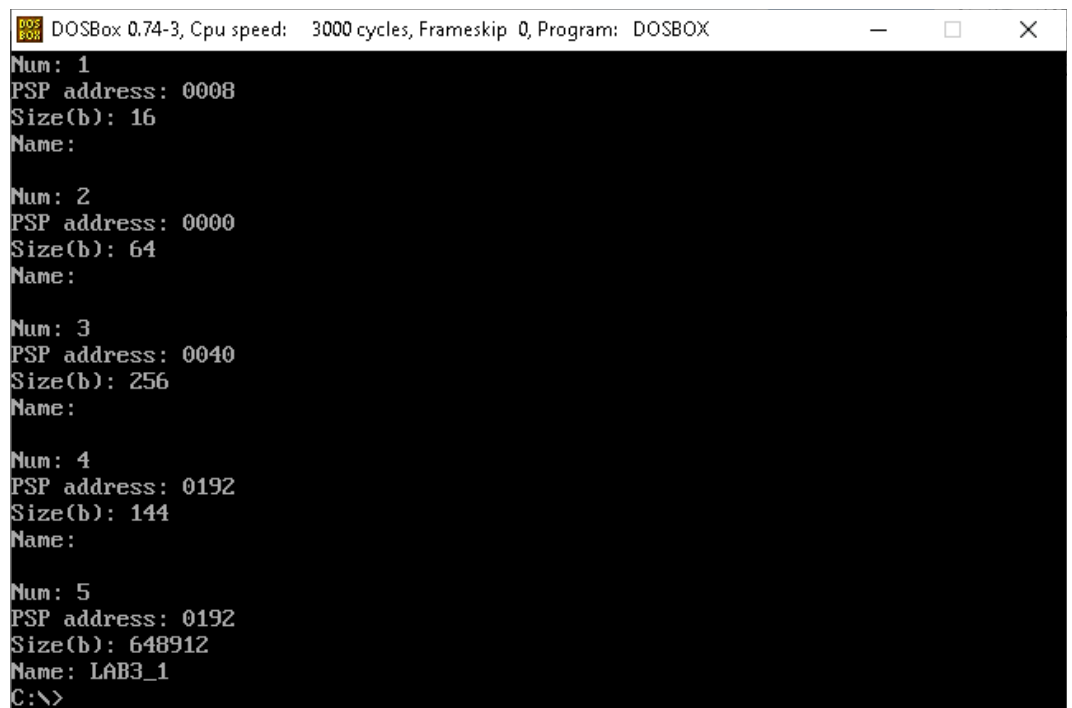
### **Структуры данных.**

Таблица 2 - Структуры данных используемые в программе

Название поля данных	Тип	Назначение
NOT_LOADED	db	Резидент не загружен
LOADED	db	Резидент уже загружен
LOAD	db	Резидент загружен
UNLOAD	db	Резидент был выгружен

### **Результат работы.**

1) Проверим состояние памяти до выполнения разработанного модуля



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
Num: 1
PSP address: 0008
Size(b): 16
Name:

Num: 2
PSP address: 0000
Size(b): 64
Name:

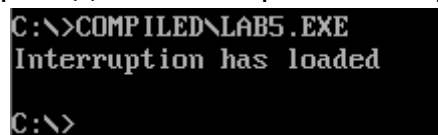
Num: 3
PSP address: 0040
Size(b): 256
Name:

Num: 4
PSP address: 0192
Size(b): 144
Name:

Num: 5
PSP address: 0192
Size(b): 648912
Name: LAB3_1
C:\>
```

Рисунок 1 – Состояние памяти до выполнения программы

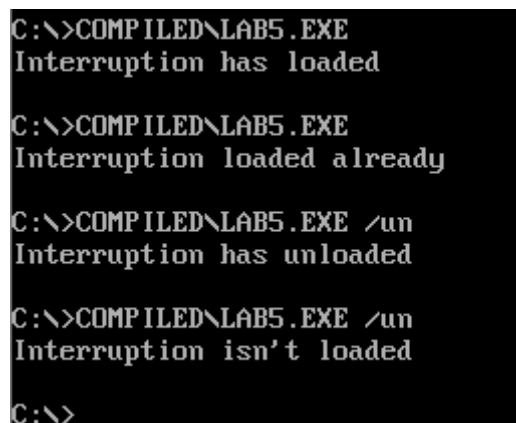
2) Установим резидентный обработчик прерываний



```
C:\>COMPILED\LAB5.EXE
Interruption has loaded
C:\>
```

Рисунок 2 – Установка резидентного обработчика прерываний

3) Проверим, определяет ли программа установленный обработчик прерываний



```
C:\>COMPILED\LAB5.EXE
Interruption has loaded

C:\>COMPILED\LAB5.EXE
Interruption loaded already

C:\>COMPILED\LAB5.EXE /un
Interruption has unloaded

C:\>COMPILED\LAB5.EXE /un
Interruption isn't loaded

C:\>_
```



### Рисунок 3 – Проверка установки обработчика

#### 4) Проверим состояние памяти на наличие загруженного модуля

```
1
2 Available size: 643888
3 Extended size: 15360
4
5 Num: 1
6 PSP address: 0008
7 Size(b): 16
8 Name: NUL NUL NUL NUL NUL NUL NUL NUL
9
10 Num: 2
11 PSP address: 0000
12 Size(b): 64
13 Name: NUL NUL NUL NUL NUL NUL NUL NUL
14
15 Num: 3
16 PSP address: 0040
17 Size(b): 256
18 Name: NUL NUL NUL NUL NUL NUL NUL NUL
19
20 Num: 4
21 PSP address: 0192
22 Size(b): 144
23 Name: NUL NUL NUL NUL NUL NUL NUL NUL
24
25 Num: 5
26 PSP address: 0192
27 Size(b): 4848
28 Name: LAB5 NUL NUL NUL NUL
29
30 Num: 6
31 PSP address: 02CC
32 Size(b): 144
33 Name: NUL NUL NUL NUL NUL NUL NUL NUL
34
35 Num: 7
36 PSP address: 02CC
37 Size(b): 643888
38 Name: LAB3_1 NUL NUL
```

### Рисунок 4 – Проверка состояния памяти

#### 5) Проверим работает ли программа: введем “w@nn@ [pl@y](#) asdfghjk?”

Буква a меняется на V

Буква s меняется на A

Буква d меняется на L

Буква f меняется на O

Буква g меняется на R

Буква h меняется на A

Буква j меняется на N

Буква k меняется на T

```
C:\>COMPILED\LAB5.EXE
Interruption has loaded

C:\>COMPILED\LAB3_1.COM >> r1.txt

C:\>w@nn@ pl@y VALORANT?
Illegal command: w@nn@.

C:\>
```

Рисунок 5 – Проверка работы программы

- 6) Запустим отложенную программу с ключом /un. и проверим состояние памяти после выгрузки резидента

<pre>Available size: 648912 Extended size: 15360  Num: 1 PSP address: 0008 Size(b): 16 Name: NUL NUL NUL NUL NUL NUL NUL NUL  Num: 2 PSP address: 0000 Size(b): 64 Name: NUL NUL NUL NUL NUL NUL NUL NUL  Num: 3 PSP address: 0040 Size(b): 256 Name: NUL NUL NUL NUL NUL NUL NUL NUL  Num: 4 PSP address: 0192 Size(b): 144 Name: NUL NUL NUL NUL NUL NUL NUL NUL  Num: 5 PSP address: 0192 Size(b): 648912 Name: LAB3 1 NUL NUL</pre>	<pre>C:\&gt;w@nn@ pl@y VALORANT? Illegal command: w@nn@.  C:\&gt;COMPILED\LAB5.EXE /un Interruption has unloaded  C:\&gt;COMPILED\LAB3_1.COM &gt;&gt; r1.txt  C:\&gt;_</pre>
---	--

Рисунок 6 – Запуск отложенной программы

### Ответы на контрольные вопросы.

- 1) Какого типа прерывания использовались в работе?

Прерывания функций DOS(21h) прерывания функций BIOS(int 09h).

- 2) Чем отличается скан код от кода ASCII?

Код ASCII – это код символа из таблицы ASCII, а скан-код – код, присвоенный каждой клавише, с помощью которого драйвер клавиатуры распознает, какая клавиша была нажата.

**Вывод.**

В ходе лабораторной работы был построен пользовательский обработчик прерывания, встроенный в стандартный обработчик от клавиатуры. Изучены дополнительные функции работы с памятью, такие как: установка программы-резидента и выгрузка его из памяти, а также организация и управление прерываниями.