

МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе № 4

по дисциплине «Операционные системы»

Тема: Обработка стандартных прерываний.

Студентка гр.8383

Преподаватель

Поплавский И.

Ефремов М.А,

г. Санкт-Петербург

2021 г.

1. Постановка задачи

1.1. Цель работы:

В архитектуре компьютера существуют стандартные прерывания, за которыми закреплены определенные вектора прерываний. Вектор прерываний хранит адрес подпрограммы обработчика прерываний. При возникновении прерывания, аппаратура компьютера передает управление по соответствующему адресу вектора прерывания. Обработчик прерываний получает управление и выполняет соответствующие действия.

В лабораторной работе №4 предлагается построить обработчик прерываний сигналов таймера. Эти сигналы генерируются аппаратурой через определенные интервалы времени и, при возникновении такого сигнала, возникает прерывание с определенным значением вектора. Таким образом, управление будет передано функции, чья точка входа записана в соответствующий вектор прерывания.

1.2. Сведения о функциях и структурах данных управляющей программы

Функции управляющей программы

Имя функции	Описание функции
setCurs	Функция устанавливает позицию курсора.
getCurs	Функция определяет позицию и размер курсора.
printSTR	Функция вывода строки.
ROUT	Функция - обработчик прерывания. Функция считает общее число прерываний и выводит на экран полученное значение. Если

	значение флага равно 1, то функция восстанавливает стандартное прерывание и выгружается из памяти.
SET_INTERRUPT	Функция устанавливает пользовательское прерывание.
BASE_FUNC	Функция проверяет было ли установлено пользовательское прерывание. Если в командной строке был передан параметр /up, то функция устанавливает значение флага функции обработчика прерывания равное 1.
MAIN PROC	Основная функция программы.

Структура данных управляющей программы

Имя	Тип	Назначение
message_1	db	Вывод строки 'Resident was loaded'
message_2	db	Вывод строки 'Resident was unloaded'
message_3	db	Вывод строки 'Resident has already been loaded'
message_4	db	Вывод строки 'Resident has already been unloaded'

Структура данных собственного прерывания

Имя	Тип	Назначение
KEEP_IP	dw	Переменная для сохранения значения регистра IP.
KEEP_CS	dw	Переменная для сохранения значения регистра CS.
KEEP_PSP	dw	Переменная для сохранения значения регистра PSP.
KEEP_SS	dw	Переменная для сохранения значения регистра SS.
KEEP_AX	dw	Переменная для сохранения значения регистра AX.
KEEP_SP	dw	Переменная для сохранения значения регистра SP.

2. Ход работы.

2.1. Был написан программный модуль типа .EXE, который выполняет следующие функции:

- Проверяет установлено ли пользовательское прерывание с вектором 1Ch.

- Устанавливает резидентную функцию для обработки прерывания и настраивает вектор прерываний, если прерывание не установлено, и осуществляется выход по функции 2Ch 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.

2.2. Был выполнен запуск отлаженной программы. Для проверки размещения прерывания в памяти была запущена программа лабораторной работы №3, которая отображает карту памяти в виде блоков MCB.

```
C:\>L3_1.COM
Available memory (B): 648912
Extended memory (KB): 15360
I MCB Type I PSP Address I Size I SC/SD I
  4D      0008      16
  4D      0000      64
  4D      0040     256
  4D      0192     144
  5A      0192   648912   L3_1
```

Рисунок 1 - Запуск программы лабораторной работы №3 до загрузки резидентной программы.

```
C:\>14.exe
Resident was loaded
C:\>
```

Рисунок 2 - Запуск программы лабораторной работы №4.

```
C:\>14.exe
Resident was loaded
C:\>13_1.com
Available memory (B): 647920
Extended memory (KB): 15360
| MCB Type | PSP Address | Size | SC/SD |
|-----|-----|-----|-----|
| 4D      | 0008      | 16   |      |
| 4D      | 0000      | 64   |      |
| 4D      | 0040      | 256  |      |
| 4D      | 0192      | 144  |      |
| 4D      | 0192      | 816  | L4   |
| 4D      | 01D0      | 144  |      |
| 5A      | 01D0      | 647920 | L3_1 |
```

Рисунок 3 - Запуск программы лабораторной работы №3 после загрузки резидентной программы.

После запуска программы на экран было выведено сообщение о том, что резидентная программа загружена в память. Так же была выведена строчка, в которой отображается подсчет количества вызовов данной резидентной программы (использование int 10h).

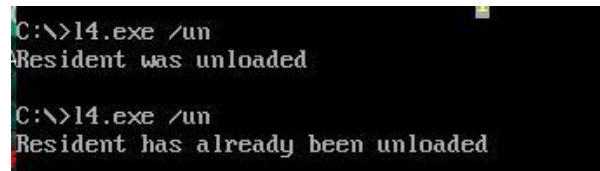
2.3. Для того, чтобы убедиться, что программа определяет установленный обработчик прерываний, был выполнен еще один запуск программы.

```
C:\>14.exe
Resident has already been loaded
```

Рисунок 4 - Повторный запуск программы лабораторной работы №4.

После повторного запуска программы на экран было выведено сообщение о том, что резидентная программа уже загружена в память.

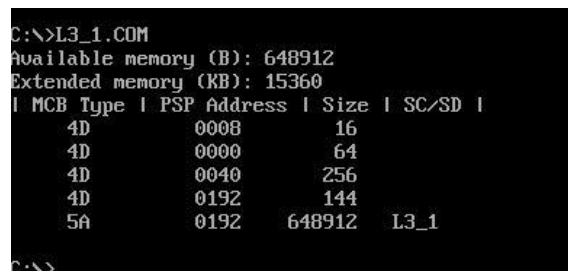
2.4. Был выполнен запуск отлаженной программы с ключом выгрузки. Для того чтобы проверить, что память, занятая резидентом освобождена был выполнен запуск программы лабораторной работы №3.



```
C:\>l4.exe /un
Resident was unloaded

C:\>l4.exe /un
Resident has already been unloaded
```

Рисунок 5 - Запуск программы лабораторной работы №4 с ключом выгрузки.



```
C:\>L3_1.COM
Available memory (B): 648912
Extended memory (KB): 15360
| MCB Type | PSP Address | Size | SC/SD |
|-----|-----|-----|-----|
| 4D       | 0008       | 16   |       |
| 4D       | 0000       | 64   |       |
| 4D       | 0040       | 256  |       |
| 4D       | 0192       | 144  |       |
| 5A       | 0192       | 648912 | L3_1
```

Рисунок 6 - Запуск программы лабораторной работы №3 после выгрузки резидентной программы.

3. Ответы на контрольные вопросы

3.1. Как реализован механизм прерывания от часов?

Ответ: Прерывание по таймеру вызывается автоматически по каждому тикку аппаратных часов (каждые 55мс). После вызова, сохраняется содержимое регистров, затем определяется источник прерывания, по номеру которого определяется смещение в таблице векторов прерываний. Полученный адрес сохраняется в регистры CS:IP, после чего управление передается по этому адресу, т.е. выполняется запуск обработчика прерываний и происходит его выполнение. После выполнения, происходит возврат управления прерванной программе.

3.2. Какого типа прерывания использовались в работе?

Ответ: В данной лабораторной работе использовались пользовательские прерывания `int 10h`, `int 21h` и аппаратное прерывание `int 1Ch`, возникающее каждые 55 мс по системному таймеру.

Заключение

В результате выполнения лабораторной работы был создан обработчик прерываний сигналов таймера, изучены дополнительные функции работы с памятью, такие как установка программы-резидента и его выгрузка из памяти.