

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

ОТЧЕТ
по лабораторной работе № 3
по дисциплине «Операционные системы»
Тема: Исследование организации управления основной памятью.

Студент гр.9383

Поплавский И.

Преподаватель

Ефремов М.А.

г. Санкт-Петербург

2021 г.

1. Постановка задачи

1.1. Цель работы:

Для исследования организации управления памятью необходимо ориентироваться на тип основной памяти, реализованный в компьютере и способ организации, принятый в ОС. В лабораторной работе рассматривается нестраничная память и способ управления динамическими разделами. Для реализации управления памятью в этом случае строится список занятых и свободных участков памяти. Функции ядра, обеспечивающие управление основной памятью, просматривают и преобразуют этот список.

В лабораторной работе исследуются структуры данных и работа функций управления памятью ядра операционной системы.

1.2. Сведения о функциях и структурах данных управляющей программы

Функции управляющей программы

Имя функции	Описание функции
TETR_TO_HEX	Функция шаблона, приведенного в методических указаниях. Функция переводит половину байта в шестнадцатеричную систему.
BYTE_TO_HEX	Функция шаблона, приведенного в методических указаниях. Байт в регистре AL переводится в два символа шестнадцатеричного числа в регистре AX.
WRD_TO_HEX	Функция шаблона, приведенного в методических указаниях. Функция переводит в шестнадцатеричную

	систему счисления 16-ти разрядное число.
BYTE_TO_DEC	Функция шаблона, приведенного в методических указаниях. Функция переводит в десятичную систему счисления.
PRINT	Функция выводит сообщение на экран.
GET_AVAILABLE_MEMORY	Функция определяет количество доступной памяти и вызывает функцию для вывода результата на экран.
GET_EXTENDED_MEMORY	Функция определяет размер расширенной памяти и вызывает

	функцию для вывода результата на экран.
GET_MCB_DATA	Функция получает информацию о каждом MCB блоке.
GET_MCB_ADDRESS	Функция определяет адрес расположения MCB блока.
GET_MCB_TYPE	Функция определяет типа MCB блока
GET_PSP_ADDRESS	Функция определяет сегментный адрес PSP
GET_MCB_SIZE	Функция определяет размер участка в параграфах

Структура данных управляющей программы

Имя	Тип	Назначение
AVAILABLE_MEM	db	Вывод строки 'Available memory (B): '
EXTENDED_MEM	db	Вывод строки 'Extended memory (KB): '
TABLE_TITLE	db	Вывод строки ' MCB Adress MCB Type PSP Address Size SC/SD '

1.3. Последовательность действий, выполняемых утилитой

- 1) Определение и вывод количества доступной памяти
- 2) Определение и вывод размера расширенной памяти
- 3) Определение и вывод информации о MCB блоках

2. Ход работы

2.1. Был написан программный модуль типа .COM, который выбирает и распечатывает информацию о количестве доступной памяти, размере расширенной памяти, о блоках управления памятью.

```
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 1 - Программный модуль L3_1.COM

2.2. Написанный на первом шаге программный модуль был изменен таким образом, чтобы программа освобождала память, которую она не занимает. Для этого использовалась функция 4Ah прерывания int 21. В результате был создан новый блок, который обозначен, как пустой участок.

```
C:\>L3_2.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
  4D      0192   13232   L3_2
  5A      0000   635664   66й 6666
```

Рисунок 2 - Программный модуль L3_2.COM

2.3. Программный модуль был изменен таким образом, чтобы после освобождения памяти, программа запрашивала 64Кб памяти. Для этого использовалась функция 48H прерывания int 21h. В результате был создан еще один блок, который занимает 65536 байт (64 Кб).

```
C:\>L3_3.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
4D         0192    13344   L3_3
4D         0192    65536   L3_3
5A         0000   570000
```

Рисунок 3 - Программный модуль L3_3.COM

2.4. Программный модуль был изменен таким образом, чтобы запрос 64 Кб памяти осуществлялся до освобождения памяти. В результате выполнения данной программы на экран было выведено сообщение об ошибке, возникшей из-за того, что вся память уже была выделена под программу и выделение еще 64Кб памяти не возможно. После чего происходит освобождение памяти, которую программа не занимает.

```
C:\>L3_4.COM
Available memory (B): 648912
Extended memory (KB): 15360
Memory allocation error
| MCB Type | PSP Address | Size | SC/SD |
4D         0008      16
4D         0000      64
4D         0040     256
4D         0192     144
4D         0192    13888   L3_4
5A         0000   635008   u u
```

Рисунок 4 - Программный модуль L3_4.COM

3. Ответы на контрольные вопросы.

3.1. Что означает «доступный» объем памяти?

Ответ: Доступный объем памяти – это максимальный объем памяти, выделенный программе операционной системой.

3.2. Где МСВ блок Вашей программы в списке?

Ответ: Для программы, реализованной на первом шаге, блок МСВ расположен в конце списка.

Для программы, реализованной на втором шаге, блок МСВ расположен в предпоследней строке списка. Последнюю строку списка занимает блок, обозначенный, как пустой участок.

Для программы, реализованной на третьем шаге, блок МСВ расположен в пятой строке списка. После него расположены блок памяти, выделенной по запросу и свободный блок памяти.

Для программы, реализованной на четвертом шаге, блок МСВ расположен в предпоследней строке списка. Последнюю строку списка занимает блок, обозначенный, как пустой участок.

3.3. Какой размер памяти занимает программа в каждом случае?

Ответ:

L3_1.COM: 648912 байт.

L3_2.COM: $648912 - 635664 - 16 = 13232$ байт.

L3_3.COM: $648912 - 570000 - 65536 - 2 * 16 = 13344$ байт.

L3_4.COM: $648912 - 635008 - 16 = 13888$ байт.

Заключение

В процессе выполнения данной лабораторной работы были исследованы структуры данных и работа функций управления памятью ядра операционной системы.