**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. 6383 |  | Яковлев Е.А. |
| Преподаватель |  | Губкин А.Ф. |

Санкт-Петербург

2018

**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.

**Постановка задачи**

Шаг 1. Необходимо написать и отладить программный модуль типа .COM, выбирает и распечатывает следующую информацию:

1. Количество доступной памяти.

2. Размер расширенной памяти.

3. Выводит цепочку блоков управления памятью.

Адреса при выводе представляются шестнадцатеричными числами. Объем памяти функциями управления памятью выводится в параграфах. Необходимо преобразовать его в байты и выводить в виде десятичных чисел. Последние восемь байт МСВ выводятся как символы, не следует преобразовывать их в шестнадцатеричные числа.

Шаг 2. Далее необходимо изменить программу таким образом, чтобы она освобождала память, которую она не занимает. Для этого используйте функцию 4Ah прерывания 21h (пример в разделе «Использование функции 4АН»).

Шаг 3. Затем необходимо изменить программу еще раз таким образом, чтобы после освобождения памяти, программа запрашивала 64Кб памяти функцией 48Н прерывания 21Н.

Шаг 4. Далее нужно изменить первоначальный вариант программы, запросив 64Кб памяти функцией 48Н прерывания 21Н до освобождения памяти.

Оформить отчёт и ответить на контрольные вопросы.

Процедуры, которые используются в программе.

|  |  |
| --- | --- |
| AVAILABLE\_MEMORY | Получение количества доступной памяти. |
| EXTENDED\_MEMORY | Получение размера расширенной памяти. |
| SHOW\_MCB | Вывод цепочки блоков управления памятью. |
| ENV\_CONTENTS | Получение содержимого области среды в символьном виде. |
| TETR\_TO\_HEX | Перевод из четверичной системы счисления в шестнадцатеричную. |
| BYTE\_TO\_HEX | Перевод из двоичной системы счисления в шестнадцатеричную. |
| WRD\_TO\_HEX | Перевод слова (2 байта) в шестнадцатеричную систему счисления. |
| PRINT | Вывод информации на экран. |
| END\_LINE | Переход на новую строку. |
| FREE\_MEM | Вспомогательная процедура для поиска доступной памяти. |

Переменные, которые используются в программе.

|  |  |  |
| --- | --- | --- |
| availableMemory | db | Применяется для вывода информации о доступной памяти. |
| extendedMemory | db | Применяется для вывода информации о расширенной памяти. |
| controlMemBlocks | db | Применяется для указание на то, что начался вывод цепочки блоков управления памятью. |
| typeMCB | db | Применяется для вывода информации о типе блока управления памятью. |
| owner | db | Применяется для вывода информации о сегментном адресе PSP. |
| areaSize | db | Применяется для вывода информации о размере участка. |
| eightBYTE | db | Применяется для вывода информации о последний восьми байтах блока управления памятью. |

**Ход работы**

*Шаг 1:* Результат запуска \*.COM модуля представлен на рис. 1.

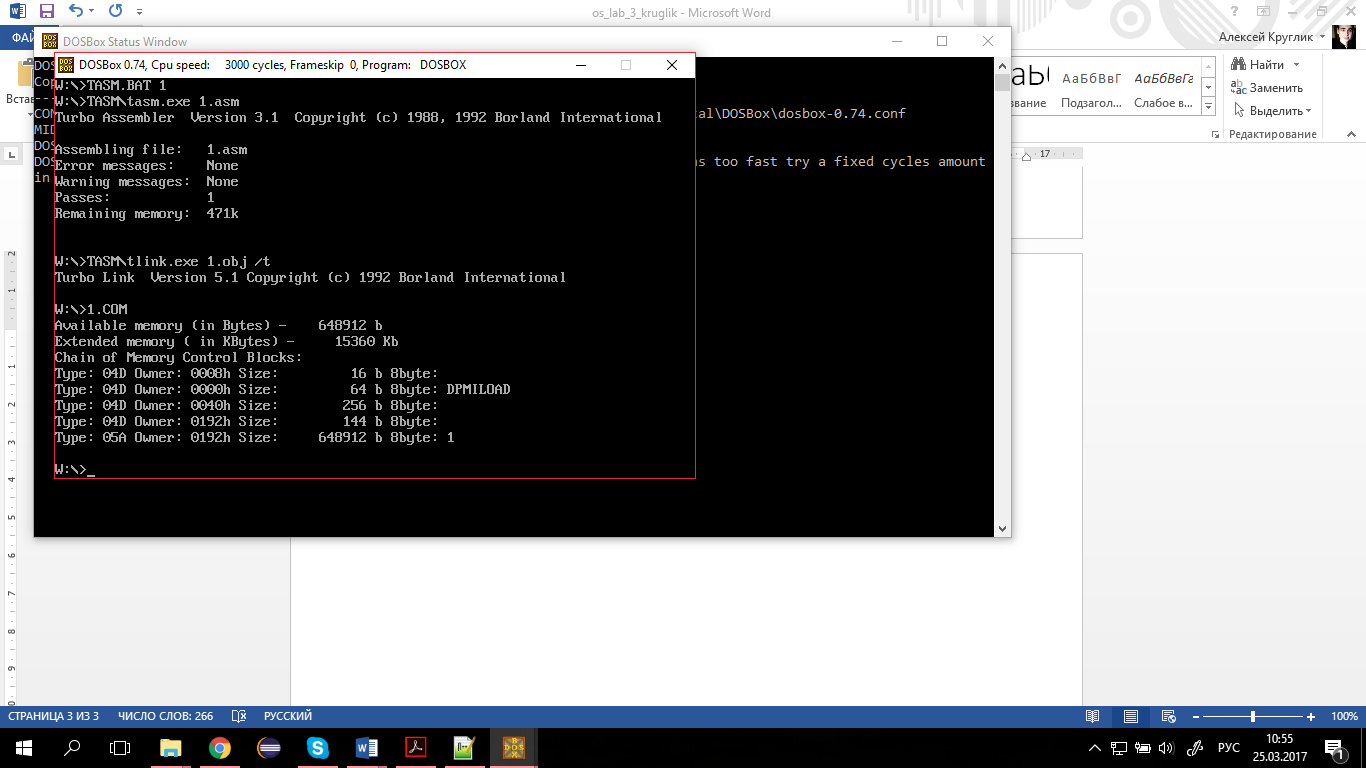


Рис. 1

*Шаг 2:* Результат запуска \*.COM модуля представлен на рис. 2.

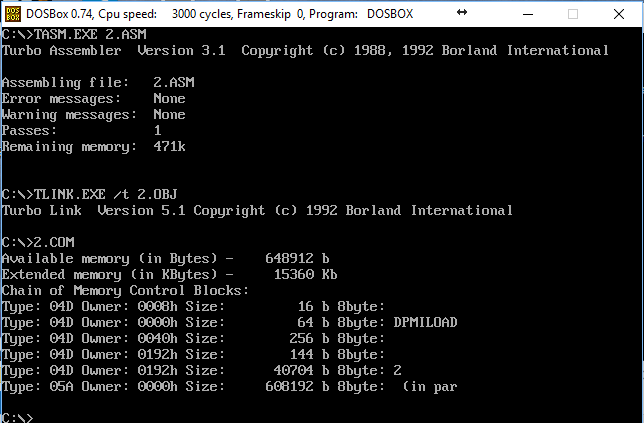


Рис. 2

*Шаг 3:* Результат запуска \*.COM модуля представлен на рис. 3.

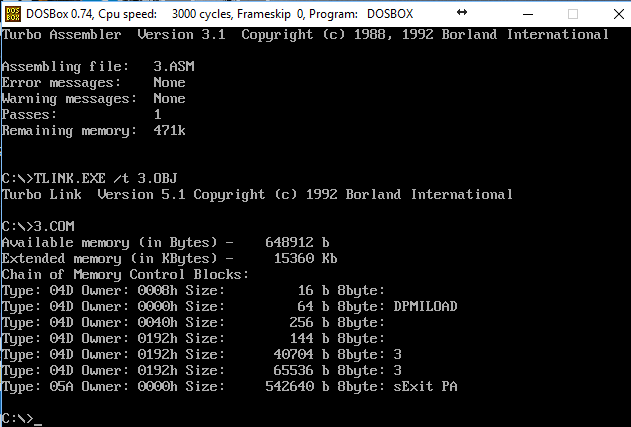


Рис. 3

*Шаг 4:* Результат запуска \*.COM модуля представлен на рис. 4.

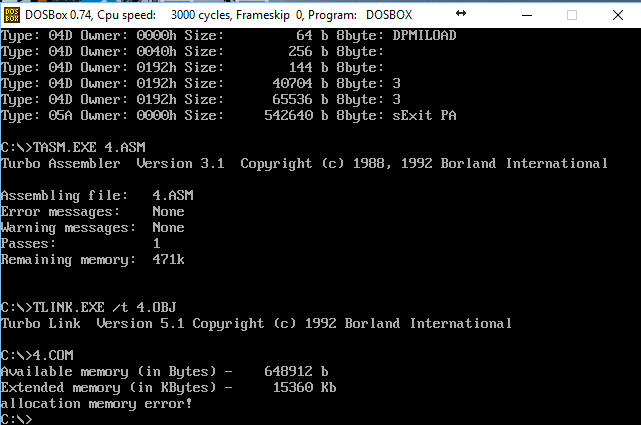


Рис. 4

**Ответы на контрольные вопросы**

1. *Что означает "доступный объем памяти"?*

Доступный объём памяти - это тот максимальный размер оперативной памяти, который может использовать программа.

1. *Где MCB блок Вашей программы?*

Это можно определить по байтам 01h – 02h, где хранится идентификатор владельца процесса (PID - process identifier). PID для процесса получается от адреса сегмента, взятого из сегмента программного префикса (PSP - program segment prefix) данного процесса. Экспериментально было установлено, что это два последних блока MCB являются блоками MCB моей программы.

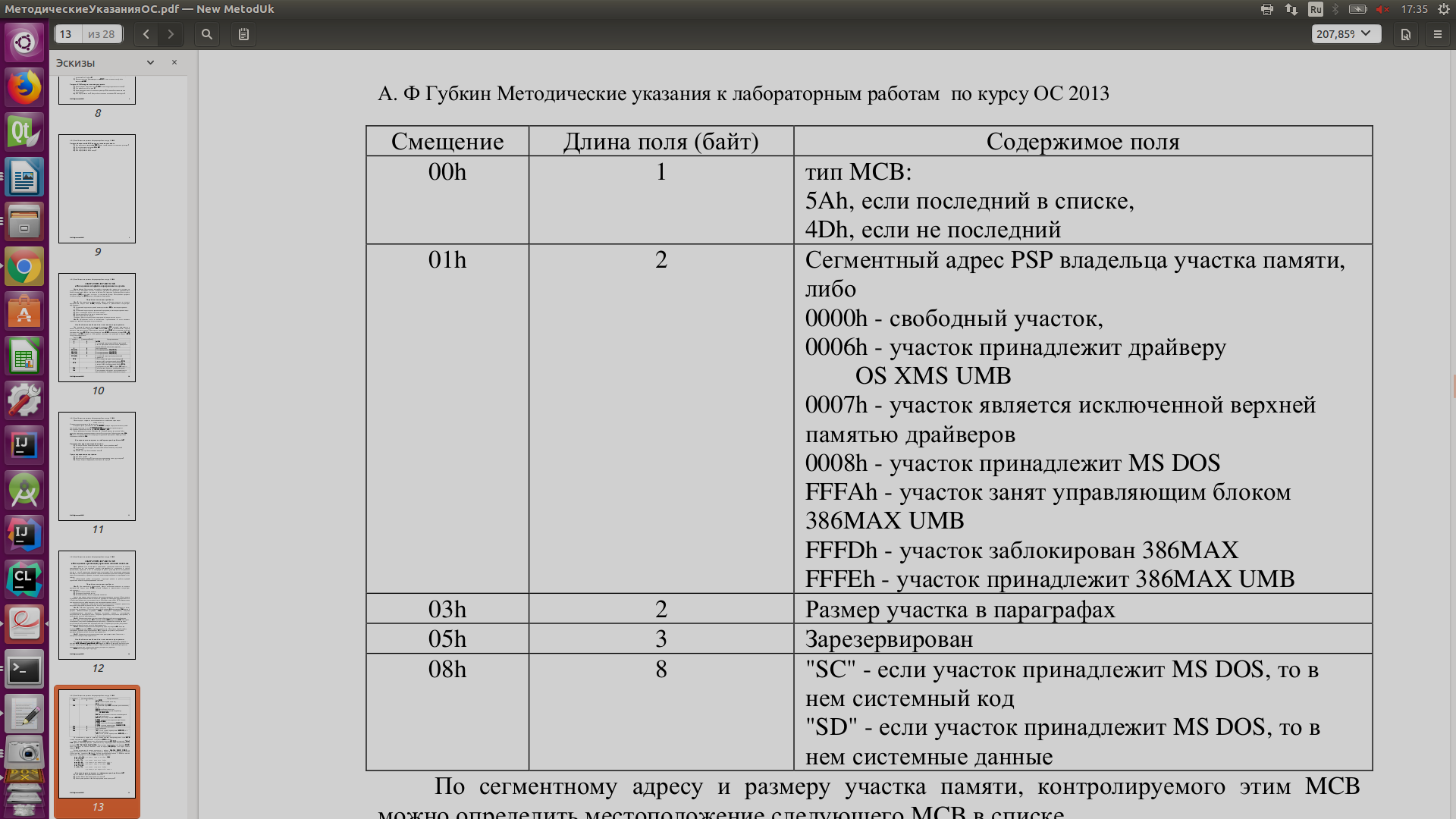
1) Для программы 1 её MCB-блок – последний в списке.

2) Для программы 2 её MCB-блок – предпоследний в списке.

3) Для программы 3 её MCB-блоки – 3 и 2 с конца (3 создан при запуске программы, 2 – дан при запросе памяти; последний – свободная память).

4) Для программы 4 её MCB-блок – предпоследний в списке.

**На принадлежность MCB процессу указывает поле размером 2 байта по смещению относительно блока MCB 01h.**



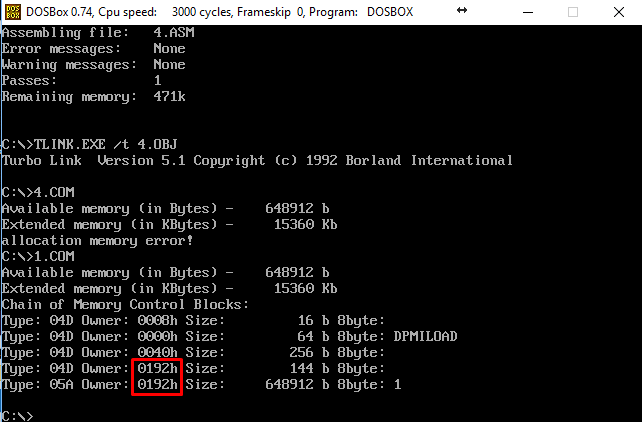


Рис. 5

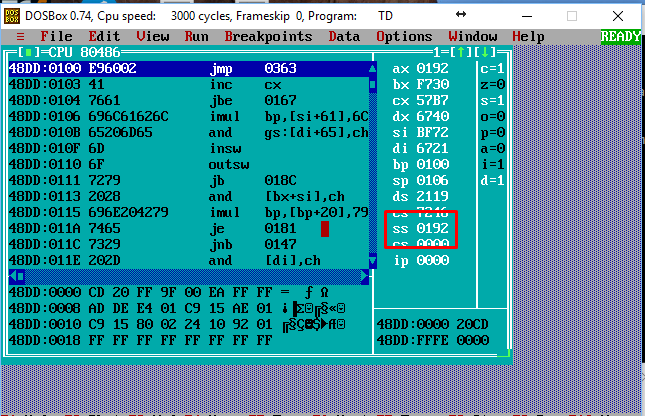


Рис. 6

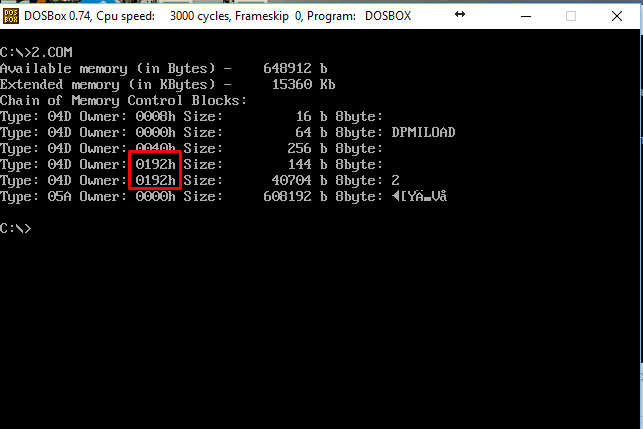


Рис. 7

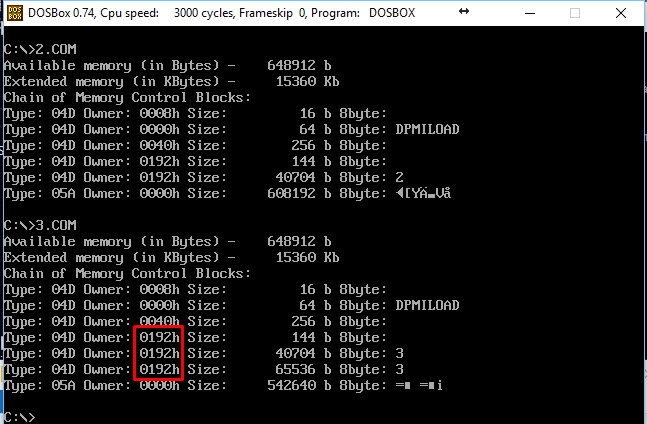


Рис. 8

*3.**Какой размер памяти занимает программа в каждом случае?*

* Для первого COM модуля - программа занимает всю выделенную ей память. МСВ не входит в эту память.
* Для второго COM модуля - только объем, занимаемый самой программой
* Для третьего COM модуля - объем, занимаемый самой программой и 64 Кб, выделенные ей по требованию.
* В четвертом – только объем, занимаемый самой программой

**Вывод**

В результате выполнения данной лабораторной работы была исследована организация управления оперативной памятью. Была написана программа, которая выводит на экран количество доступной памяти, размер расширенной памяти и цепочку блоков управления памятью.