

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

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

Студент гр. 8382

Преподаватель

Торосян Т.А.

Ефремов М.А.

Санкт-Петербург

2020

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.

Необходимые сведения для составления программы.

Учёт занятой и свободной памяти ведется при помощи списка блоков управления памятью MCB (Memory Control Block). MCB занимает 16 байт (параграф) и располагается всегда с адреса кратного 16 (адрес сегмента ОП) и находится в адресном пространстве непосредственно перед тем участком памяти, которым он управляет.

MCB имеет следующую структуру:

Смещение	Длина поля (байт)	Содержимое поля
00h	1	тип MCB: 5Ah, если последний в списке, 4Dh, если не последний
01h	2	Сегментный адрес PSP владельца участка памяти, либо 0000h - свободный участок, 0006h - участок принадлежит драйверу OS XMS UMB 0007h - участок является исключенной верхней памятью драйверов

		0008h - участок принадлежит MS DOS FFFAh - участок занят управляющим блоком 386MAX UMB FFFDh - участок заблокирован 386MAX FFFEh - участок принадлежит 386MAX UMB
03h	2	Размер участка в параграфах
05h	3	Зарезервирован
08h	8	"SC" - если участок принадлежит MS DOS, то в нем системный код "SD" - если участок принадлежит MS DOS, то в нем системные данные

По сегментному адресу и размеру участка памяти, контролируемого этим MCB можно определить местоположение следующего MCB в списке.

Адрес первого MCB хранится во внутренней структуре MS DOS, называемой "List of Lists" (список списков). Доступ к указателю на эту структуру можно получить, используя функцию f52h "Get List of Lists" int 21h. В результате выполнения этой функции ES:BX будет указывать на список списков. Слово по адресу ES:[BX-2] и есть адрес самого первого MCB.

Размер расширенной памяти находится в ячейках 30h, 31h CMOS. CMOS это энергонезависимая память, в которой хранится информация о конфигурации ПЭВМ. Объем памяти составляет 64 байта. Размер расширенной памяти в Кбайтах можно определить обращаясь к ячейкам CMOS следующим образом:

```

mov AL,30h ; запись адреса ячейки CMOS
out 70h,AL
in AL,71h ; чтение младшего байта

mov BL,AL ; размера расширенной памяти

mov AL,31h ; запись адреса ячейки CMOS
out 70h,AL

```

in AL,71h ; чтение старшего байта размера
расширенной памяти

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 MCB выводятся как символы, не следует преобразовывать их в шестнадцатеричные числа.

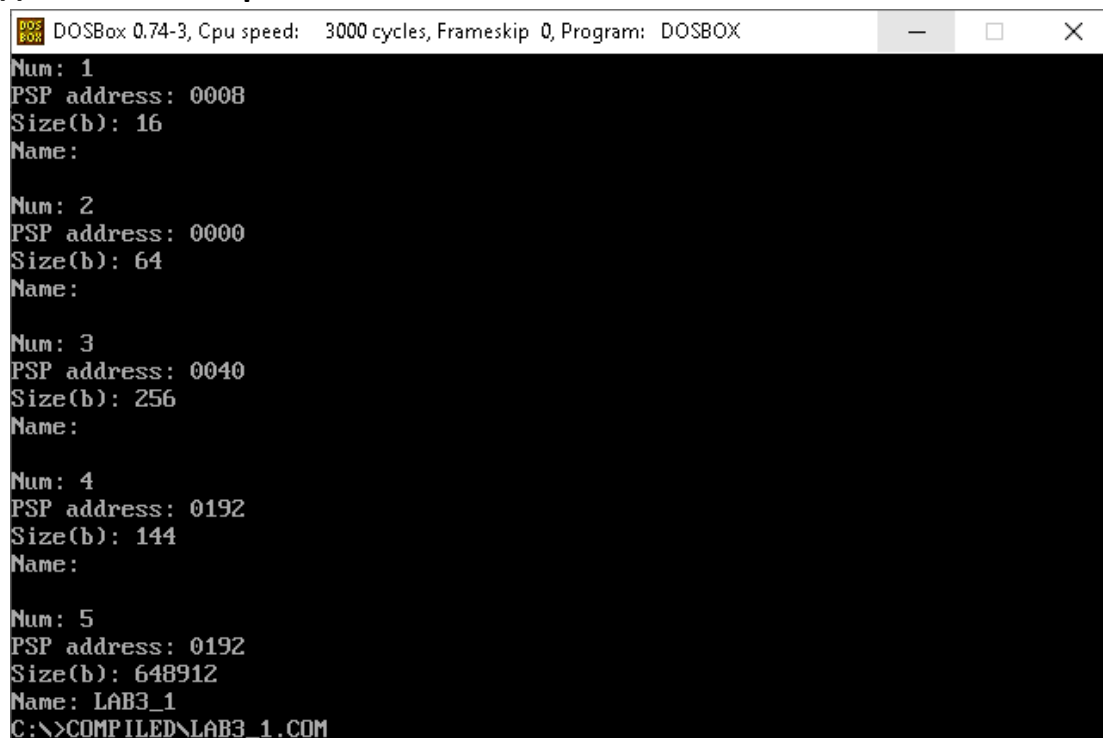
Шаг 2. Далее необходимо изменить программу таким образом, чтобы она освобождала память, которую она не занимает. Для этого используйте функцию 4Ah прерывания 21h (пример в разделе «Использование функции 4AH»).).

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

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

Шаг 5. Оформить отчёт и ответить на контрольные вопросы.

Ход выполнения работы.



```
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:

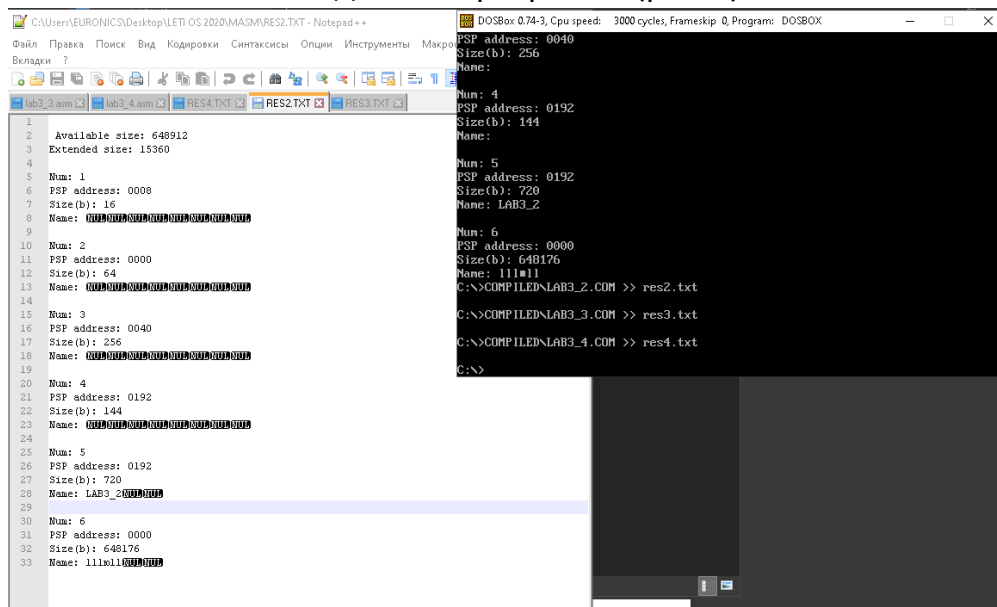
Num: 3
PSP address: 0040
Size(b): 256
Name:

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

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

Рисунок 1 – Результат работы lab3_1.com

Все доступные 648912 байт отдаются программе(рис.1).



```
C:\Users\EUROMICS\Desktop\LEET OS 2020\MASM\RES2.TXT - Notepad++
Файл Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы
Вкладки ?
lab3_3.asm lab3_4.asm RES4.TXT RES2.TXT RES3.TXT
1
2 Available size: 648912
3 Extended size: 15360
4
5 Num: 1
6 PSP address: 0008
7 Size(b): 16
8 Name: 00000000000000000000000000000000
9
10 Num: 2
11 PSP address: 0000
12 Size(b): 64
13 Name: 00000000000000000000000000000000
14
15 Num: 3
16 PSP address: 0040
17 Size(b): 256
18 Name: 00000000000000000000000000000000
19
20 Num: 4
21 PSP address: 0192
22 Size(b): 144
23 Name: 00000000000000000000000000000000
24
25 Num: 5
26 PSP address: 0192
27 Size(b): 720
28 Name: LAB3_20000000000000000000000000000000
29
30 Num: 6
31 PSP address: 0000
32 Size(b): 648176
33 Name: 11111100000000000000000000000000

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
PSP address: 0040
Size(b): 256
Name:

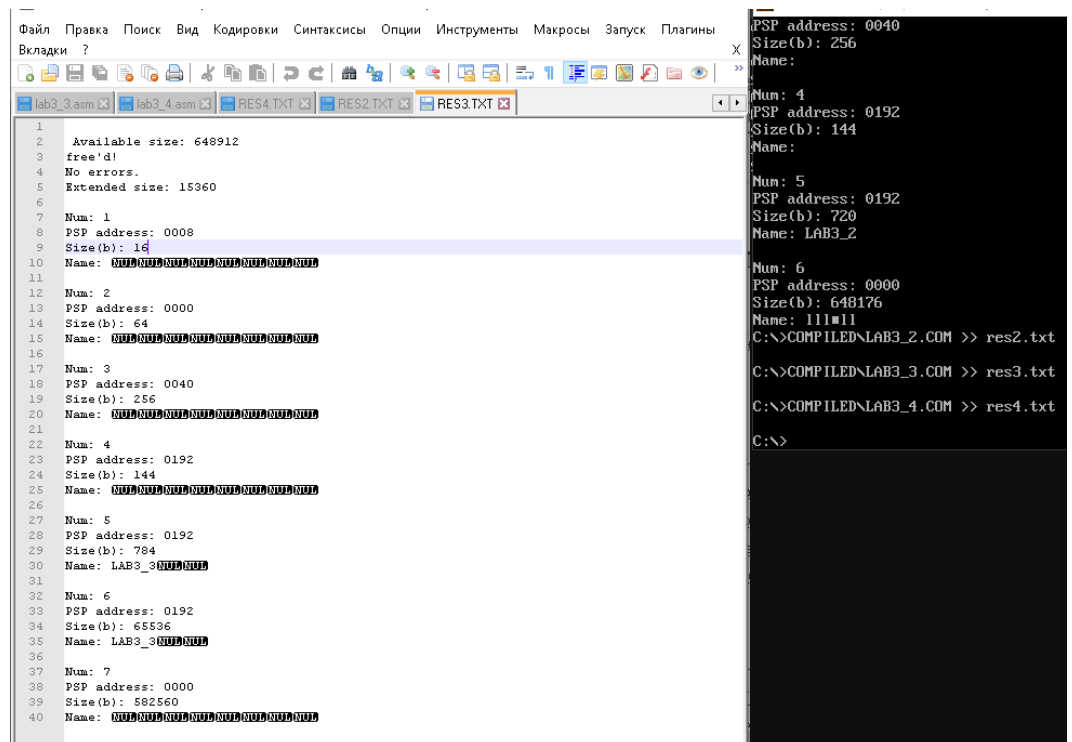
Num: 4
PSP address: 0192
Size(b): 144
Name:

Num: 5
PSP address: 0192
Size(b): 720
Name: LAB3_2

Num: 6
PSP address: 0000
Size(b): 648176
Name: 111111
C:\>COMPILED\LAB3_2.COM >> res2.txt
C:\>COMPILED\LAB3_3.COM >> res3.txt
C:\>COMPILED\LAB3_4.COM >> res4.txt
C:\>
```

Рисунок 2 – Результат работы lab3_2.com

Исходный код программы был изменён: теперь программа освобождает не занимаемую ею память. Создается новый блок, который обозначен как свободный участок, размером 648160 байт. Результат предоставлен на рис.2.



The screenshot shows a debugger window with a memory dump on the left and a command prompt on the right. The memory dump lists several memory blocks with their addresses, sizes, and names. The command prompt shows the execution of a program and the resulting memory dump.

```
1 Available size: 648912
2 free'd!
3 No errors.
4 Extended size: 15360
5
6
7 Num: 1
8 PSP address: 0008
9 Size(b): 16
10 Name: 00000000000000000000000000000000
11
12 Num: 2
13 PSP address: 0000
14 Size(b): 64
15 Name: 00000000000000000000000000000000
16
17 Num: 3
18 PSP address: 0040
19 Size(b): 256
20 Name: 00000000000000000000000000000000
21
22 Num: 4
23 PSP address: 0192
24 Size(b): 144
25 Name: 00000000000000000000000000000000
26
27 Num: 5
28 PSP address: 0192
29 Size(b): 784
30 Name: LAB3_30000000
31
32 Num: 6
33 PSP address: 0192
34 Size(b): 65536
35 Name: LAB3_30000000
36
37 Num: 7
38 PSP address: 0000
39 Size(b): 582560
40 Name: 00000000000000000000000000000000
```

```
PSP address: 0040
Size(b): 256
Name:
>>
Num: 4
PSP address: 0192
Size(b): 144
Name:
Num: 5
PSP address: 0192
Size(b): 720
Name: LAB3_2
Num: 6
PSP address: 0000
Size(b): 648176
Name: 111111
C:\>COMPILED\LAB3_2.COM >> res2.txt
C:\>COMPILED\LAB3_3.COM >> res3.txt
C:\>COMPILED\LAB3_4.COM >> res4.txt
C:\>
```

Рисунок 3 – Результат работы lab3_3.com

Код программы снова был изменён (рис.3). Вначале происходит то же самое, что и во втором случае (освобождение памяти). Затем программа запрашивает 64 Кбайт (65536 байт) памяти. На свободном участке создается новый блок, который следует за основным блоком программы и занимает 65536 байт.

```
Файл Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы Запуск Плагины
Вкладки ?
lab3_3.asm lab3_4.asm RES4.TXT RES2.TXT RES3.TXT
1
2 Available size: 648912
3 Error!
4 free'd!
5 Extended size: 15360
6
7 Num: 1
8 PSP address: 0008
9 Size(b): 16
10 Name: NNNNNNNNNNNNNNNNNNNN
11
12 Num: 2
13 PSP address: 0000
14 Size(b): 64
15 Name: NNNNNNNNNNNNNNNNNNNN
16
17 Num: 3
18 PSP address: 0040
19 Size(b): 256
20 Name: NNNNNNNNNNNNNNNNNNNN
21
22 Num: 4
23 PSP address: 0192
24 Size(b): 144
25 Name: NNNNNNNNNNNNNNNNNNNN
26
27 Num: 5
28 PSP address: 0192
29 Size(b): 784
30 Name: LAB3_4NNNNNN
31
32 Num: 6
33 PSP address: 0000
34 Size(b): 648112
35 Name: LAB3_3NNNNNN
PSP address: 0040
Size(b): 256
Name:
Num: 4
PSP address: 0192
Size(b): 144
Name:
Num: 5
PSP address: 0192
Size(b): 720
Name: LAB3_2
Num: 6
PSP address: 0000
Size(b): 648176
Name: 111=11
C:\>COMPILED\LAB3_2.COM >> res2.txt
C:\>COMPILED\LAB3_3.COM >> res3.txt
C:\>COMPILED\LAB3_4.COM >> res4.txt
C:\>
```

Рисунок 4 – Результат работы lab3_4.com

Код программы снова был изменён (рис.4). Происходит запрос 64 Кбайт до освобождения памяти. Однако выдаётся ошибка(во второй строчке), так как запрос памяти происходит в тот момент, когда вся доступная память занята программой. Затем происходит освобождение памяти, аналогично второму случаю.

Ответы на контрольные вопросы.

1) Что означает «доступный объём памяти»?

Доступный объём памяти – это объём базовой или стандартной памяти (conventional memory), эта память представляет собой "нижние" 640 Кбайт ОЗУ. Для использования базовой памяти не нужны никакие дополнительные драйверы, поскольку операционная система MS DOS изначально создана для работы в адресах 0 - 640 Кбайт

2) Где MCB блок Вашей программы в списке?

На рисунках 1, 2, 4 - это блоки под номерами 4 и 5.

На рисунке 3 - это блоки под номерами 4, 5, 6.

3) Какой размер памяти занимает программа в каждом случае?

1. 648912 байт.
2. $144 + 720 = 864$ байт.
3. $144 + 784 + 65536 = 66464$ байт.
4. $144 + 784 = 928$ байт.

Вывод.

В ходе работы было проведено исследование структуры данных и работы функций управления памятью ядра операционной системы, а также рассмотрены нестраничная память и способы управления динамическими разделами.