

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

ОТЧЕТ
по лабораторной работе №7
по дисциплине «Операционные системы»
ТЕМА: Построение модуля оверлейной структуры

Студент гр. 7381

Дорох С.В.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2019

Цель работы.

Исследование возможности построение загрузочного модуля оверлейной структуры. Исследуется структура оверлейного сегмента и способ загрузки и выполнения оверлейных сегментов. Для запуска вызываемого оверлейного модуля используется функция 4B03h прерывания int 21h. Все загруженные и оверлейные модули находятся в одном каталоге.

В этой работе также рассматривается приложение, состоящее из нескольких модулей, поэтому все модули помещаются в один каталог и вызываются с использованием полного пути.

Необходимые сведения для составления программы.

Для организации программы, имеющей оверлейную структуру, используется функция 4B03h прерывания 21h. Эта функция позволяет в отведённую область памяти, начинающуюся с адреса сегмента, загрузить программу, находящуюся в файле на диске. Передача управления загруженной программе этой функцией не осуществляется и префикс сегмента программы (PSP) не создаётся.

Если флаг переноса $CF = 1$ после выполнения функции, то произошли ошибки и регистр AX содержит код ошибки. Значение регистра AX характеризует следующие ситуации, представленные в табл. 1.

Таблица 1 – Возможные ошибки при выполнении функции 4B03h.

Код ошибки	Описание
1	Несуществующая функция
2	Файл не найден
3	Маршрут не найден
4	Слишком много открытых файлов
5	Нет доступа
8	Мало памяти
10	Неправильная среда

Если флаг переноса $CF = 0$, то оверлей загружен в память.

Перед загрузкой оверлея вызывающая программа должна освободить память по функции 4Ah прерывания 21h. Затем определить размер оверлея. Это можно сделать с помощью функции 4Eh прерывания 21h. Перед обращением к функции необходимо определить область памяти размером в 43 байта под буфер DTA, которую функция заполнит, если файл будет найден.

Функция использует следующие параметры: *CX* – значение байта атрибутов, которое для файла имеет значение 0; *DS:DX* – указатель на путь к файлу, который записывается в формате строки ASCIIZ.

Если флаг переноса *CF* = 1 после выполнения функции, то произошли ошибки и регистр *AX* содержит код ошибки. Значение регистра *AX* характеризует ситуации, представленные в табл. 2.

Таблица 2 – Возможные ошибки при выполнении функции 4Eh.

Код ошибки	Описание
2	Файл не найден
3	Маршрут не найден

Если *CF* = 0, то в области памяти буфера DTA со смещением 1Ah будет находиться младшее слово размера файла, а в слове со смещением 1Ch – старшее слово размера памяти в байтах.

Полученный размер файла следует перевести в параграфы, причём следует взять большое целое числа параграфов. Затем необходимо отвести память с помощью функции 48h прерывания 21h. После этого необходимо сформировать параметры для функции 4B03h и выполнить её.

После отработки оверлея необходимо освободить память с помощью функции 49h прерывания 21h.

Оверлейный сегмент не является загрузочным модулем типов .COM или .EXE. Он представляет собой кодовый сегмент, который оформляется в ассемблере как функция с точкой входа по адресу 0 и возврат осуществляется командой *retf*. Это необходимо сделать, потому что возврат управления должен быть осуществлён в программу, выполняющую оверлейный сегмент.

Если использовать функции выхода 4Ch прерывания 21h, то программа закончит свою работу.

Описание функций и структур данных

Название функции	Назначение
PRINT_STR	макрос, печатающий строку
READ_SIZE_OF_OVL	считывание размера файла оверлея и запрос нужного для загрузки объема памяти
LOAD_OVL	запуск и выполнение оверлейного сегмента
BEGIN	головная функция
FREE_MEM	Освобождение лишней памяти
CLEAN_MEM	освобождение отведенной под оверлейный сегмент памяти
PREPARE_PATH	макрос, подготавливающий строку, содержащую путь к первому оверлейному сегменту
PREPARE_PATH_2	макрос, подготавливающий строку, содержащую путь ко второму оверлейному сегменту

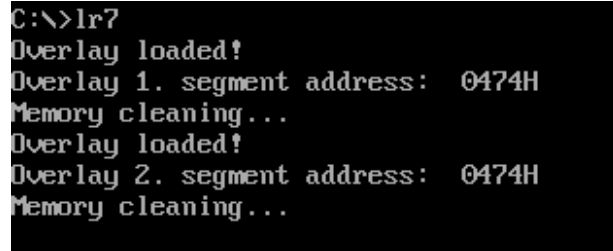
Последовательность действий, выполняемых утилитой

1. Освобождение памяти для загрузки оверлеев.
2. Поиск пути к оверлею.
3. Чтение размера файла оверлея и выделение памяти, достаточной для его загрузки.
4. Загрузка и выполнение оверлейного сегмента.
5. Освобождение памяти, отведённой для оверлейного сегмента.
6. Повторение пунктов 1-5 для второго оверлейного сегмента.

7. Выход в DOS.

Результаты работы программы

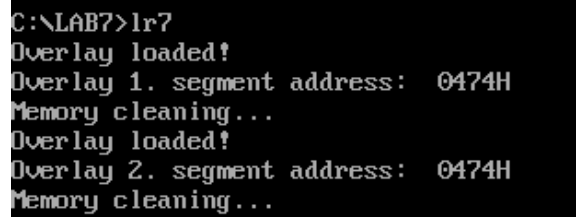
1. Запуск программы lr7.exe(оверлейные сегменты загружены с одного адреса)(см Рис.1):



```
C:\>lr7
Overlay loaded!
Overlay 1. segment address: 0474H
Memory cleaning...
Overlay loaded!
Overlay 2. segment address: 0474H
Memory cleaning...
```

Рисунок 1 – Запуск программы lr7.exe

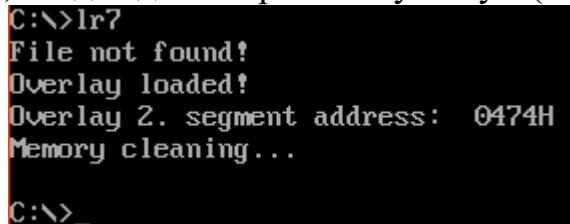
2. Запуск программы с другого каталога(см. Рис.2):



```
C:\LAB7>lr7
Overlay loaded!
Overlay 1. segment address: 0474H
Memory cleaning...
Overlay loaded!
Overlay 2. segment address: 0474H
Memory cleaning...
```

Рисунок 2 – Запуск программы lr7.exe из другого каталога

3. Запуск программы, когда один оверлей отсутствует(см. Рис.3):



```
C:\>lr7
File not found!
Overlay loaded!
Overlay 2. segment address: 0474H
Memory cleaning...
C:\>_
```

Рисунок 3 – Запуск программы lr7.exe при отсутствии одного оверлея

Вывод: В процессе выполнения данной лабораторной работы была исследована возможность построения загрузочного модуля динамической структуры.

Ответы на контрольные вопросы.

Как должна быть устроена программа, если в качестве оверлейного сегмента использовать .COM модули?

Ответ: Для выполнения такого сегмента необходимо в начале выделенной памяти сформировать блок PSP размером в 100h. При обращении к оверлейному сегменту необходимо обращаться к сегменту, смещённому на 100h, так как com-сегмент com-модуля-оверлея загружается без этого смещения(без смещения 100h).