Операционные системы

Лабораторная №1

Дизассемблирование INT 8h

Цель лабораторной работы: знакомство со средством дизассемблирования – **sourcer** и с получением дизассемблерного кода ядра операционной системы Windows на примере обработчика прерывания **Int 8h** в **virtual mode** – специальном режиме защищенного режима, который эмулирует реальный режим работы вычислительной системы на базе процессоров Intel.

Задание:

Используя sourser **(sr.exe**) получить дизассемблерный код обработчика аппаратного прерывания от системного таймера Int 8h.

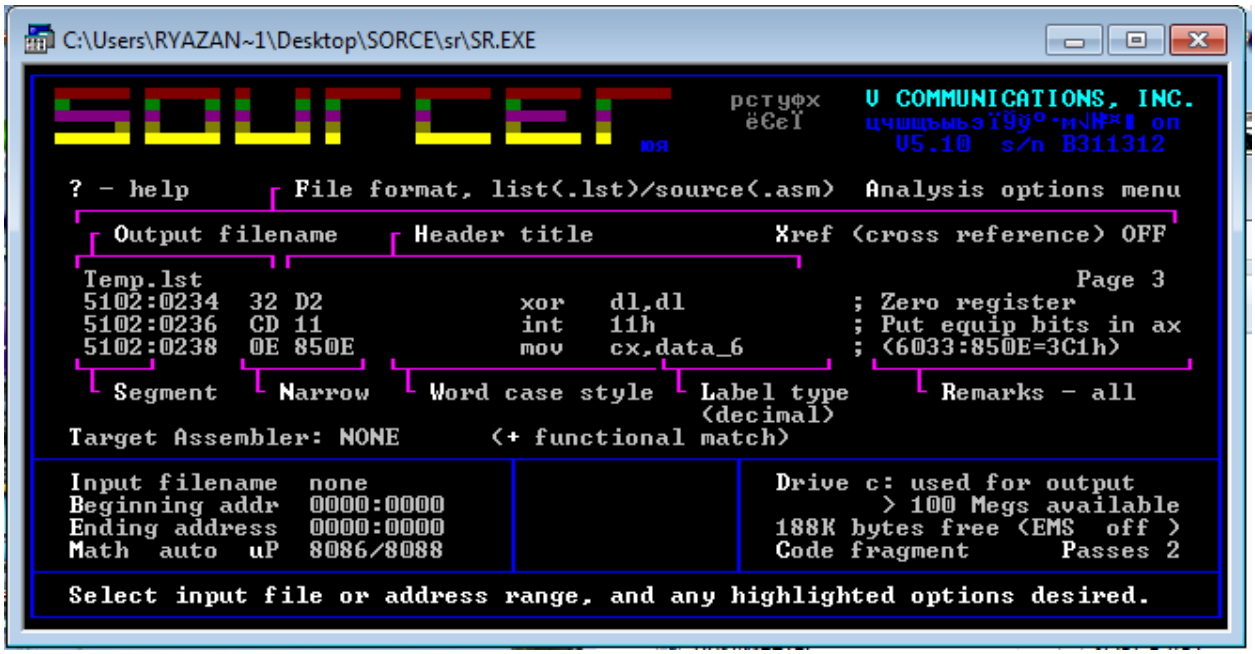
На основе полученного кода составить алгоритм работы обработчика Int 8h.

По данной лабораторной работе составляется отчет в письменном виде.

- Отчет должен содержать: полученный ассемблерный код с адресами команд и коментариями;
- Графический алгоритм работы обработчика прерывания Int 8h, структурированный и выполненный в соответствии с ГОСТ 19.701-90 ЕСПД – «Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения».
  Источник:http://www.znaytovar.ru/gost/2/GOST_1970190_ESPD_Sxemy_algori.html
  Алгоритм должен быть структурирован – теорема Бема, Якопини.

Методические указания:

Сорсер не является интерактивной программой, а имеет с буквенно-цифровой интерфейс,



в которой для начала дизассемблирования надо указать начальный адрес кода - **Beginning addr** и конечный адрес - **Ending address**. Начальный адрес Вы получаете из таблицы векторов прерываний реального режима по номеру прерывания или используя 35 функцию DOS.

Конечный адрес выбирется из соображений примерного размера обработчика и затем уточняется в процессе работы. Результат работы sourcer записывается в файл **Temp.lst**. Имя файла можно изменить, нажав клавишу «o» (латинское) и в поле **Enter output file**

ввести свой идентификатор. Аналогично можно изменить уровень коментированности выходного файла в окне, которое откроется при нажатии клавиши “a” (латинское) – **Analysis options menu**.

Таблица векторов прерываеий располагается в памяти начиная с нулевого адреса. Каждый вектор занимает в таблице 4 байта: 2 байта сегмент и 2 байта смещение (far адрес).

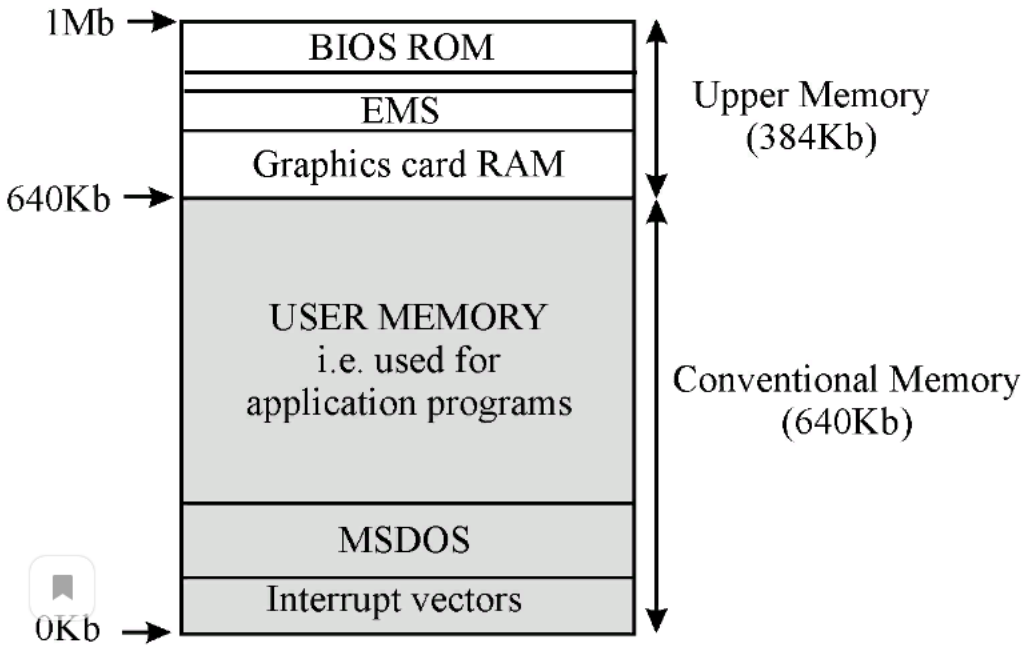


Рис. 15 базовая карта памяти системы XT под управлением MS-DOS.

Для получения адреса вектора из таблицы векторов прерываний необходимо:

- Вычислить смещение нужного вектора прерывания умножив номер вектора на 4 (длину far адреса в DOS). Например, вектор для 5-го прерывания (Print Screen) имеет смещение 20. Переведя 20 в шестнадцетиричное число, получим 14h, т.е. шестнадцетиричное смещение
- Используя отладчик DEBUG с помощью команд:

  DEBUG
  -D 0000:0014 L 4

  Получим содержимое четырех байтов в шестнадцетиричном виде:
  54FF:00F0

  Следует учитывать обратный порядок следования байтов:

  54FF : 00F0

  смещение : сегмент
  младшая:старшая младшая:старшая части

  В результате получим адрес – F000 : FF54

Для получения адреса обработчика можно также использовать 35 функцию ДОСа.

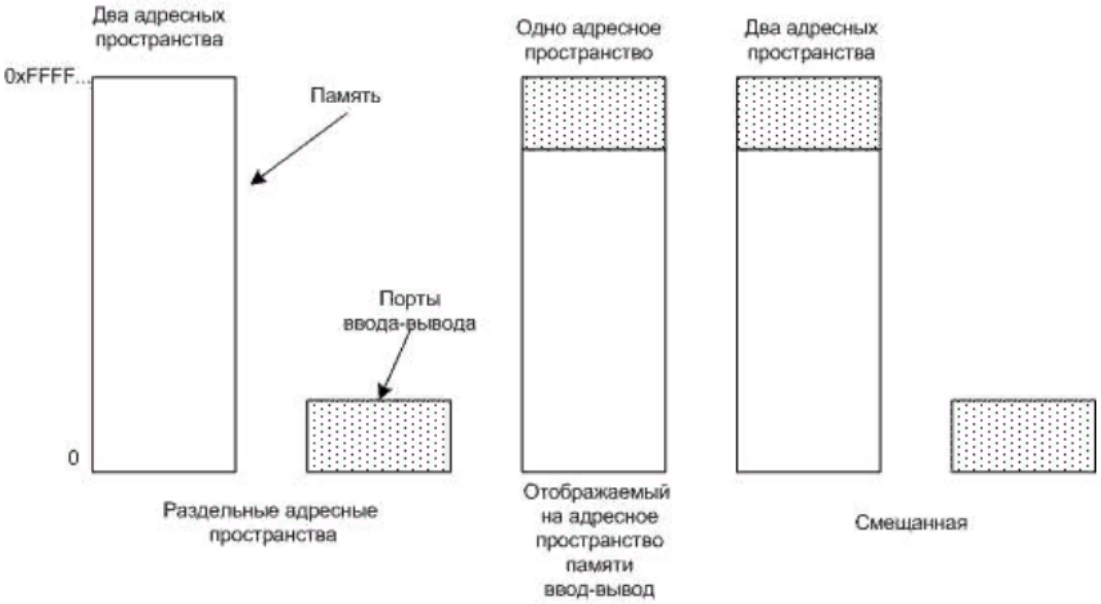
Код обработчика прерывания заканчивается командой iret (interrupt return). В коде эмуляции ДОСа iret надо найти по переходам и включить в листинг кода прерывания.

О функциях восьмого прерывания можно дополнительно прочитать
http://www.frolov-lib.ru/books/bsp.old/v33/ch5.htm

Том 33, М.: Диалог-МИФИ, 1997, 304 стр.

Необходимо обратить внимание на сброс контроллера прерываний.

Дополнительно



Способы реализации доступа к управляющим регистрам и буферам

В конструкции современных персональных компьютеров используется выделенная высокоскоростная шина памяти. Эта шина предназначена для увеличения скорости обмена данными между процессором и памятью. Устройства ввода-вывода не могут «увидеть» адреса памяти, выставляемые процессором на эту шину, следовательно, они не могут реагировать на такие адреса.

Первый способ решения проблемы. Чтобы отображение регистров ввода-вывода могло работать на системах с несколькими шинами необходимо чтобы сначала все обращения к памяти посылались процессором по выделенной быстрой шине напрямую памяти (чтобы не снижать производительности). Если память не может ответить на эти запросы, процессор пытается сделать это еще раз по другим шинам.

Такое решение работоспособно, но требует дополнительного увеличения сложности аппаратуры.

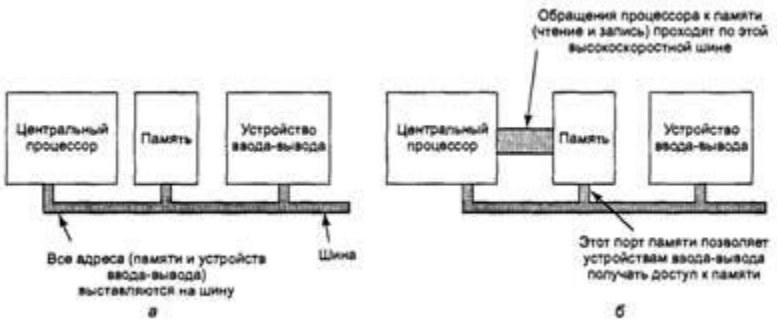


Рис. 5.2. Архитектура с одной шиной (*а*); архитектура памяти с двумя шинами (*б*)

Второй способ решения проблемы. Установка на шину памяти специального следящего устройства, передающего все адреса потенциально заинтересованным устройствам ввода-вывода. Проблема в том, что устройства ввода-вывода могут не успеть обработать эти запросы с той же скоростью, что и память.

Третий способ решения проблемы. Фильтрация адресов микросхемой моста PCI. Эта микросхема содержит регистры диапазона, заполняемые во время загрузки компьютера. Например, диапазон адресов от 640K до 1M может быть помечен как не относящийся к памяти. Все адреса, попадающие в подобный диапазон, передаются не памяти, а на шину PCI. Недостаток этой схемы состоит в необходимости принятия во время загрузки решения о том, какие адреса не являются адресами памяти.