Специальность 090203 «Программное обеспечение вычислительной техники и автоматизированных систем»

Отчет
По практическому занятию №1

Дисциплина: МДК 01.01 Системное программирование

Выполнил студент **3** **курса** **__** **группы**

(Ф.И.О)

Принял _____

Дата сдачи

Оценка

2017 – 2018

Практическое занятие 1. Формирование физического адреса памяти в реальном и защищенном режимах процессора

Формирование физического адреса в реальном режиме

Ниже перечислены характеристики механизма адресации физической памяти в реальном режиме.

- Диапазон изменения физического адреса — от 0 до 1 Мбайт. Эта величина определяется тем, что шина адреса i8086 имела 20 линий.
- Максимальный размер сегмента — **64** Кбайт. Причина ограничения в 16-разрядной архитектуре i8086. Нетрудно подсчитать, что максимальное значение, которое могут содержать 16-разрядные регистры, составляет $2^{16} – 1$, что применительно к памяти и определяет величину **64** Кбайт.
- Для обращения к конкретному физическому адресу оперативной памяти необходимо определить адрес начала сегмента (сегментную составляющую) и смещение внутри сегмента.

Понятие адреса начала сегмента ввиду принципиальной важности требует дополнительного пояснения. Отталкиваясь от разрядности сегментных регистров, можно утверждать, что сегментная составляющая адреса (или база сегмента) представляет собой всего лишь 16-разрядное значение, помещенное в один из сегментных регистров. Максимальное значение, которое при этом получается, соответствует $2^{16} – 1$. Если так рассуждать, то получается, что адрес начала сегмента может быть только в диапазоне 0–64 Кбайт от начала оперативной памяти. Возникает вопрос, как адресовать остальную часть оперативной памяти вплоть до 1 Мбайт с учетом того, что размер самого сегмента не превышает 64 Кбайт. Дело в том, что в сегментном регистре содержатся только старшие 16 бит физического адреса начала сегмента. Недостающие младшие четыре бита 20-разрядного адреса получаются сдвигом значения в сегментном регистре влево на 4 разряда. Эта операция сдвига выполняется аппаратно и для программного обеспечения абсолютно прозрачна. Получившееся 20-разрядное значение и является настоящим физическим адресом, соответствующим началу сегмента. Что касается второго компонента (смещения), участвующего в образовании

физического адреса некоторого объекта в памяти, то он представляет собой 16-разрядное значение. Это значение может содержаться явно в команде либо косвенно в одном из регистров общего назначения. Процессор самостоятельно складывает обе составляющие, в результате получается физический адрес памяти размерностью 20 бит. Данный механизм образования физического адреса позволяет сделать программное обеспечение перемещаемым, то есть независящим от конкретных адресов загрузки его в оперативной памяти (рис. 1.4).

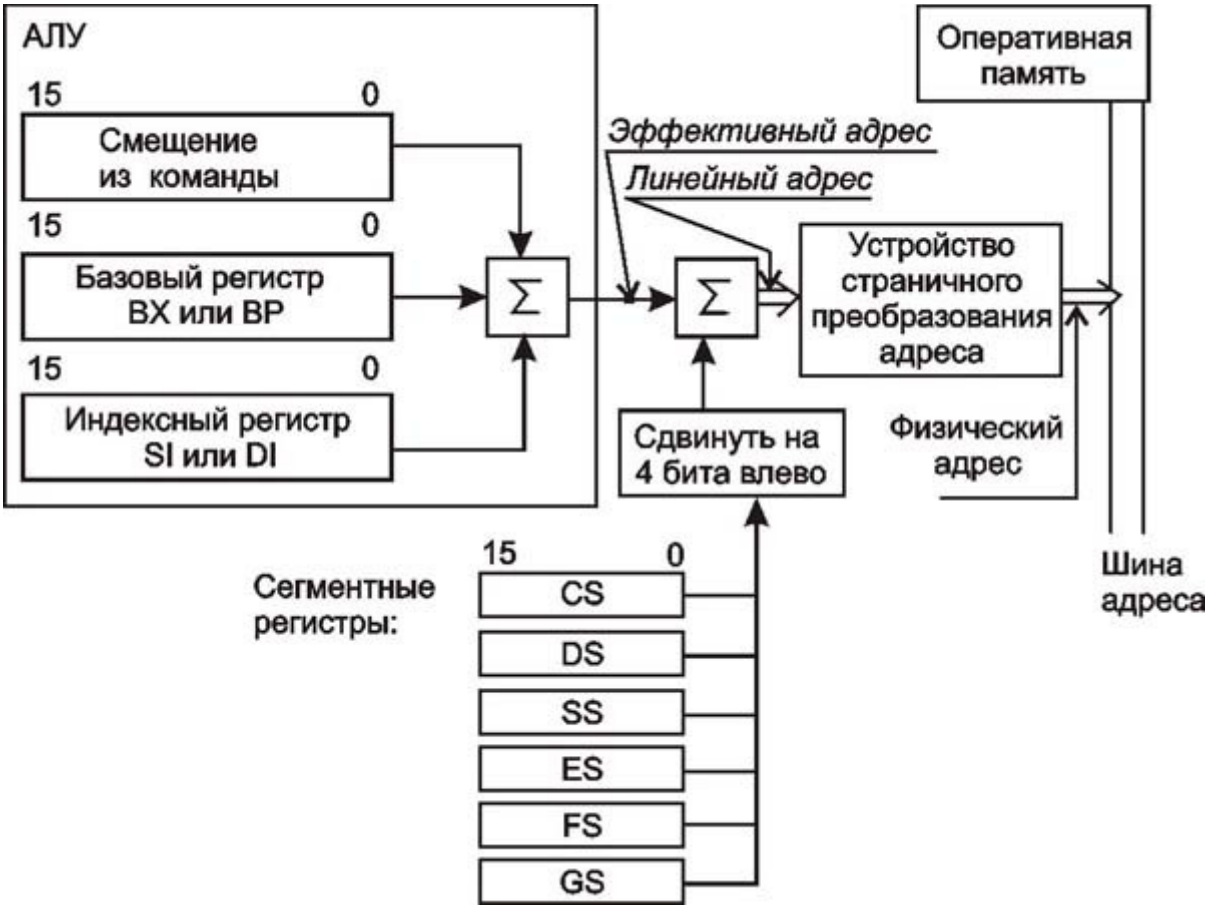

Рис. 1.4. Механизм формирования физического адреса в реальном режиме

На рисунке хорошо видно, как формируется некоторый целевой физический адрес: сегментная часть извлекается из одного из сегментных регистров, сдвигается на четыре разряда влево и суммируется со смещением. В свою очередь, видно, что значение смещения можно получить минимум из одного и максимум из трех источников: из значения смещения в самой машинной

команде и/или из содержимого одного базового и/или одного индексного регистра. Количество источников, участвующих в формировании смещения, определяется кодированием конкретной машинной команды, и если таких источников несколько, то значения в них складываются. В заключение заметим, что не стоит волноваться из-за несоответствия размеров шины адреса процессора i486 или Pentium (32 бита) и 20-разрядного значения физического адреса реального режима. Пока процессор находится в реальном режиме, старшие 12 линий шины адреса попросту недоступны, хотя при определенных условиях и существует возможность работы с первыми 64 Кбайт оперативной памяти, лежащими сразу после первого мегабайта.

Недостатки такой организации памяти:

- сегменты бесконтрольно размещаются с любого адреса, кратного 16 (так как содержимое сегментного регистра аппаратно смещается на 4 разряда), и, как следствие, программа может обращаться по любым адресам, в том числе и реально не существующим;
- сегменты имеют максимальный размер 64 Кбайт;
- сегменты могут перекрываться с другими сегментами.

Желанием ввести в архитектуру средства, позволяющие избавиться от указанных недостатков, и обусловлено, в частности, появление защищенного режима, в котором работают все современные операционные системы, в том числе Windows и Linux.

Например, пусть у нас есть логический адрес **1234h:0123h**. Сегментная компонента равна **1234h**, компонента смещения - **0123h**. Вычислим физический адрес, соответствующий нашему логическому адресу:

- расширяем до 20 бит сегментную компоненту, дописывая справа 4 нулевых бита, получаем число **12340h**;
- расширяем до 20 бит компоненту смещения, дописывая слева 4 нулевых бита, получаем число **00123h**;
- для получения физического адреса складываем полученные числа: **12340h + 00123h = 12463h**.

Очевидно, что одному физическому адресу может соответствовать несколько логических. Например, физическому адресу 12463h соответствует логический адрес 1246h:0003h.

Задание 1. Пусть у Вас есть физический адрес. Вычислите сегментный адрес и смещение.

Для своего варианта (номер по журналу) вычислите физические адреса для указанных трех логических адресов.			
Вариант	Физический адрес	Физический адрес	Физический адрес
2	**2F3EH**	**5EF3H**	**c3F3H**
	2+17+3+16H	**5+16+17+3H**	**14+3+17+3H**
	19+19H	**21+20H**	**17+20H**
	38H	**41H**	**37H**
		38H+41H	**37H+79H**
		79H	**116H**
ИТОГО			**116H**

Задание 2. Пусть у Вас есть логический адрес **1F34H:012BH.** Вычислите физический адрес, соответствующий этому логическому адресу.

Для своего варианта (номер по журналу) вычислите физические адреса для указанных трех логических адресов.			
Вариант	Логический адрес	Логический адрес	Логический адрес
1	**2F3EH:01ABH**	**5EF3H:5DB1H**	**c3F3H:5DB1H**
	2+17+3+16H **1+12+13H**	**5+16+17+3H** **5+15+13+1H**	**14+3+17+3H** **5+15+13+1H**
	19+19H **13+13H**	**21+20H** **20+14H**	**17+20H** **20+14H**
	38H **26H**	**41H** **34H**	**37H** **34H**
	1.4H	**1.2H**	**1.08H**

Формирование физического адреса в защищенном режиме

Основная идея защищенного режима — оградить исполняемые процессором программы от взаимного влияния. В защищенном режиме процессор реализует различные механизмы безопасности. В контексте нашего изложения интерес представляет защита по доступу к памяти. Для введения любого охранного механизма нужно иметь как можно больше информации об охраняемых объектах. Для процессора такими объектами являются исполняемые им программы. Организуя защиту программ по доступу к памяти, фирма Intel не стала нарушать принцип сегментации, свойственный ее процессорам. Так как каждая программа занимает один или

несколько сегментов в памяти, то логично иметь больше информации об этих сегментах, как об объектах, реально существующих в данный момент в вычислительной системе. Если каждому из сегментов присвоить определенные атрибуты, то часть функций по контролю за доступом к ним можно переложить на процессор. Что и было сделано. Любой сегмент памяти в защищенном режиме имеет следующие основные атрибуты:

- расположение сегмента в памяти;
- размер;
- уровень привилегий (определяет права данного сегмента относительно других сегментов);
- тип доступа (определяет назначение сегмента).

В отличие от реального режима в защищенном режиме программа уже не может запросто обратиться по любому линейному адресу памяти. Для этого она должна иметь определенные полномочия и удовлетворять ряду требований. Ключевым объектом защищенного режима является специальная структура — дескриптор сегмента, который представляет собой 8-байтовый дескриптор (краткое описание) непрерывной области памяти, содержащий перечисленные выше атрибуты. На рис. 1.5 представлена структура дескриптора сегмента.

Охарактеризуем назначение некоторых полей дескриптора сегмента:

- **LIMIT_1** и **LIMIT_2** — **20-**разрядное поле, определяющее размер сегмента;
- **BASE_1** И **BASE_2** — **32-**разрядное поле, содержит значение линейного адреса начала сегмента в памяти;
- **AR** — байт, поля которого определяют права доступа к сегменту;
- **D** — бит разрядности операндов и адресов;
- **G** — бит гранулярности.

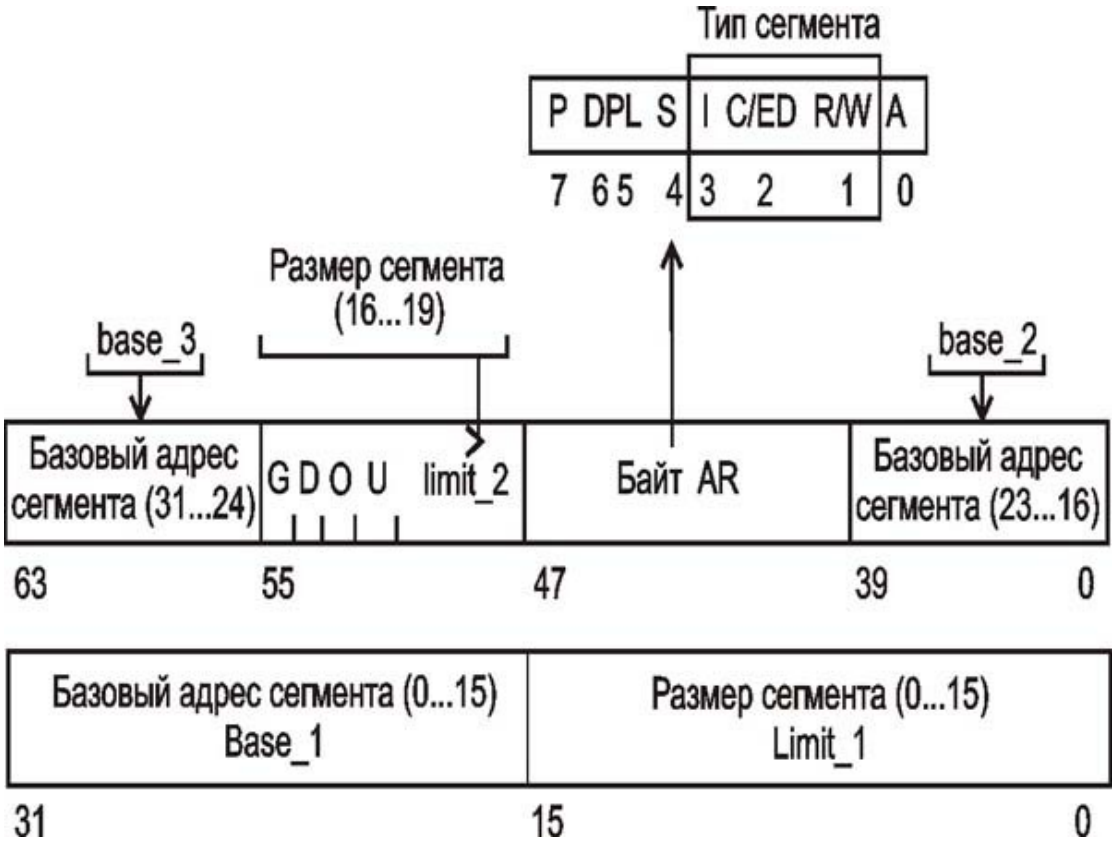

Рис. 1.5. Структура дескриптора сегмента защищенного режима процессора

В защищенном режиме размер сегмента не фиксирован, его расположение можно задать в пределах 4 Гбайт. Если посмотреть на рисунок, то возникнет вопрос: почему разорваны поля, определяющие размер сегмента и его начальный (базовый) адрес? Это результат эволюции процессоров. Защищенный режим впервые появился в процессоре i80286. Он имел 24-разрядную адресную шину и, соответственно, мог адресовать в защищенном режиме до 16 Мбайт оперативной памяти. Для этого ему достаточно было иметь в дескрипторе поле базового адреса в 24 бита и поле размера сегмента длиной 16 бит. После появления процессора i80386 с 32-разрядными шинами команд и данных в целях совместимости программ разработчики не стали менять формат дескриптора, а просто использовали свободные поля. Внутри процессора эти поля объединены. Внешне же они остались разделенными, с чем приходится мириться при программировании. Следующий интересный момент связан с тем, что размер сегмента в защищенном режиме может достигать 4 Гбайт, то есть сегмент способен

занимать все доступное физическое пространство памяти. Как это возможно, если суммарный размер поля размера сегмента составляет всего 20 бит, что соответствует величине 1 Мбайт? Секрет скрыт в поле гранулярности — бит G (см. рис. 1.5). Если G = 0, то значение в поле размера сегмента является размером сегмента в байтах, если G = 1, — в страницах. Размер страницы равен 4 Кбайт. Нетрудно подсчитать, что при максимальном значении поля размера сегмента 0fffffh это соответствует 1 M страниц или величине 1 M x 4 Кбайт = 4 Гбайт.

Выведение информации о базовом адресе сегмента и его размере на уровень процессора позволяет аппаратно контролировать, как программы работают с памятью, и предотвращать обращения к адресам, находящимся вне сегмента.

Другой аспект защиты заключается в том, что сегменты неравноправны в правах доступа к ним. Информация об этом содержится в специальном байте AR, входящем в состав дескриптора. Тип сегмента определяют следующие элементы байта AR: поле **DPL**, биты **R/W, C/ED** и I. Поле **DPL** — часть механизма защиты по привилегиям. Суть его выражается тем, что конкретный сегмент может находиться на одном из четырех уровней привилегированности с номерами 0, 1, 2 и 3. Максимум поблажек предоставляется уровню 0. Существует ряд ограничений (опять-таки на аппаратном уровне) на взаимодействие сегментов кода, данных и стека с различными уровнями привилегий.
Таким образом, в защищенном режиме перед использованием любой области памяти должна быть проведена определенная работа по инициализации соответствующего дескриптора. Эту работу выполняет операционная система или программа, сегменты которой также описываются подобными дескрипторами. Все дескрипторы собираются вместе в одну из трех таблиц:

- глобальная таблица дескрипторов (Global Descriptor Table, **GDT**), адрес которой хранится в регистре **GDTR**;

- локальная таблица дескрипторов (Local Descriptor Table, **LDT**), ее адрес находится в регистре **LDTR**;
- таблица дескрипторов векторов прерываний (Interrupt Descriptor Table, **IDT**), ее адрес хранится в регистре **IDTR**.

В какую именно таблицу должен быть помещен дескриптор, определяется его назначением. Адрес, по которому размещаются эти дескрипторные таблицы, может быть любым; он хранится в специально предназначенном для этого системном РЕГИСТРЕ. Схемы, показанные на рис. 1.3, б и в, иллюстрируют принцип формирования адреса в защищенном режиме. Важно отметить изменение роли сегментных регистров. В защищенном режиме они содержат не адрес, а селектор, то есть указатель на соответствующую ячейку одной из таблиц дескрипторов (**GDT**или **LDT**).

Задание 3. Ответить письменно (в файле-отчете) на ниже следующие вопросы письменно.

1. Как в защищенном режиме программа может обратиться по любому линейному адресу памяти?

Недостатки такой организации памяти:

- сегменты бесконтрольно размещаются с любого адреса, кратного 16 (так как содержимое сегментного регистра аппаратно смещается на 4 разряда), и, как следствие, программа может обращаться по любым адресам, в том числе и реально не существующим;
- сегменты имеют максимальный размер 64 Кбайт;
- сегменты могут перекрываться с другими сегментами.

2. В каких пределах расположение сегмента можно задать в защищенном режиме?

В защищенном режиме размер сегмента не фиксирован, его расположение можно задать в пределах 4 Гбайт.

3. Каким образом удается сделать так, что сегменты неравноправны в правах доступа к ним?

Другой аспект защиты заключается в том, что сегменты неравноправны в правах доступа к ним. Информация об этом содержится в специальном байте AR, входящем в состав дескриптора. Тип сегмента определяют следующие элементы байта AR: поле **DPL**, биты **R/W, C/ED** и I. Поле **DPL** — часть механизма защиты по привилегиям. Суть его выражается тем, что

конкретный сегмент может находиться на одном из четырех уровней привилегированности с номерами 0, 1, 2 и 3. Максимум поблажек предоставляется уровню 0. Существует ряд ограничений (опять-таки на аппаратном уровне) на взаимодействие сегментов кода, данных и стека с различными уровнями привилегий.
Таким образом, в защищенном режиме перед использованием любой области памяти должна быть проведена определенная работа по инициализации соответствующего дескриптора.

4. Что хранится в регистрах: GDTR, LDTR, IDTR?

Все дескрипторы собираются вместе в одну из трех таблиц:

- глобальная таблица дескрипторов (Global Descriptor Table, **GDT**), адрес которой хранится в регистре **GDTR**;
- локальная таблица дескрипторов (Local Descriptor Table, **LDT**), ее адрес находится в регистре **LDTR**;
- таблица дескрипторов векторов прерываний (Interrupt Descriptor Table, **IDT**), ее адрес хранится в регистре **IDTR**.

Вывод: Во время практической работы, я приобрела навыки формирование физического адреса памяти в реальном и защищенном режимах