Сегмент состояния задачи TSS (Task State Segment) - это структура данных, которая определяет состояние (т.е. контекст) задачи. В ней хранится содержимое всех регистров общего назначения, сегментных и некоторых системных регистров а также некоторая дополнительная информация.

Для каждой задачи в системе нужно определить один сегмент TSS и далее процессор будет сам использовать эти сегменты задач, т.е. структура TSS поддерживается аппаратно.

TSS бывают двух типов - 16- и 32-разрядные. 16-разрядные появились в процессоре i286 и они могут использоваться в 32-разрядных процессорах, но мы их рассматривать не будем, потому что программирование Intel286 здесь не рассматривается.

32-разрядные TSS появились в процессоре Intel386, с тех пор они не претерпели изменений и на них строится механизм мультизадачности любой ОС защищённого режима. Везде далее, где мы будем говорить о TSS, именно такие, 32-разрядные, и будут подразумеваться.

На рис. 3-1 приведена структура 32-разрядного TSS:

Биты 31 15 0	Поле	Смещение
0	Link	00h
ESP0		04h
0	SS0	08h
ESP1		0Ch
0	SS1	10h
ESP2		14h
0	SS2	18h
CR3		1Ch
EIP		20h
EFLAGS		24h
EAX		28h
ECX		2Ch
EDX		30h
EBX		34h
ESP		38h
EBP		3Ch
ESI		40h
EDI		44h
0	ES	48h
0	CS	4Ch
0	SS	50h
0	DS	54h
0	FS	58h
0	GS	5Ch
0	LDTR	60h
I/O Map	0 T	64h

Рисунок 3-1. Структура TSS

Как видите, большая часть полей предназначена для хранения содержимого регистров общего назначения и сегментных регистров. Остальные поля:

- **Link** (смещение 0) - это поле обратной связи. Когда процессор переключается с одной задачи на другую, он записывает в это поле информацию о предыдущей задаче, чтобы "знать", куда ему возвращать управление после того, как эта задача отработает. Подробнее об этом поле см. далее.
- **SSi:ESPi** (i = 0,1,2) - это пары значений указателя на стек для разных уровней привилегий - 0, 1 и 2. Задачу можно вызывать с разных уровней привилегий и для корректной работы нужно разделение стека. Работа с этими полями - отдельная тема и пока мы использовать их не будем.
- **CR3** (смещение 1Ch) - это содержимое регистра CR3 для данной задачи. Как вы

помните, CR3, он же PDBR, определяет параметры каталога страниц. То, что каждая задача имеет в своём TSS поле CR3 означает, что каждая задача может иметь свой "личный" набор страниц и работать по своей собственной схеме страничного преобразования. Однако, на практике, удобнее использовать одно значение для всех CR3 - это повышает надёжность и защищённость системы.

- **LDTR** (смещение 60h) - это селектор дескриптора LDT, который используется данной задачей. Если задаче не нужна LDT, то в этом поле хранится 0 - вот здесь проявляется смысл условия, по которому нулевой дескриптор в GDT не используется - нулевой селектор означает отсутствие этого селектора. Если LDT используется, то задача может иметь свой собственный набор дескрипторов.
- **T** (0-й бит по смещению 64h) - флаг трассировки, применяется для отладки задач, процессор его только считывает. Если этот флаг установлен, то при переключении на данную задачу процессор генерирует исключение отладки (прерывание 1). Этот флаг также может быть полезен в системах, использующих FPU (сопроцессор), MMX и XMM. Как вы заметили, в TSS нет полей для значений регистров этих модулей, поэтому для сохранения и загрузки контекста этих устройств требуется дополнительные усилия со стороны операционной системы.
- **I/O Map** (смещение 66h) - это адрес карты ввода вывода, точнее - смещение её начала от начала TSS. Каждая задача может иметь такую карту, в которой каждый бит определяет возможность доступа к конкретному порту - установлен - нет доступа, сброшен - есть. Таким образом реализуется защита на уровне доступа к портам ввода/вывода. Подробнее об этом - в главе 5.
- **Дополнительная часть TSS** (начиная со смещения 68h) - это та часть TSS, которая предусматривается программистом при создании данной задачи. Здесь должна храниться карта ввода/вывода (если она есть), здесь также может быть контекст FPU, MMX и XMM и другая информация, в зависимости от конкретной реализации ОС. Процессор непосредственно обращается только к карте ввода/вывода, к остальной информации он явно не обращается и на размер этой части не накладывается особых ограничений (т.е. не более 1Мб).

TSS определяется специальным дескриптором. Такой дескриптор является системным объектом и может находиться только в GDT (в LDT - нельзя), а это значит, что задача самостоятельно не может определять для себя другие задачи в LDT и для операционной системы есть условия для контроля задач.

Формат дескриптора TSS следующий:

```
dw      limit_low       ; Младшая часть предела
dw      address_low     ; Младшая часть адреса
db      address_mid     ; 3-й (из 4-х) байт адреса
db      access_rights   ; Права доступа
db      limit_hi        ; Старшая часть предела
db      address_hi      ; Старший байт адреса
```

Примечание 1:

Байт прав доступа access_rights имеет следующий формат:

```
бит: описание
 0: = 1
 1: Бит B (Busy) - занятость задачи (см. следующую главу)
 2: = 1
```

```
 3: = 0
 4: = 0
5,6: = DPL - Уровень привилегий сегмента TSS (см. следующую
главу)
 7: P - бит присутствия сегмента, обычно установлен в 1.
```

Примечание 2:

Старшая часть предела limit_hi имеет такой формат:

```
 бит: описание
0..3: Старшие 4 бита предела
  4: Бит U (User) - этот бит ОС может использовать в своих
целях
5..7: = 0
```

Для наглядности формат дескриптора TSS приведен на рис. 3-1:

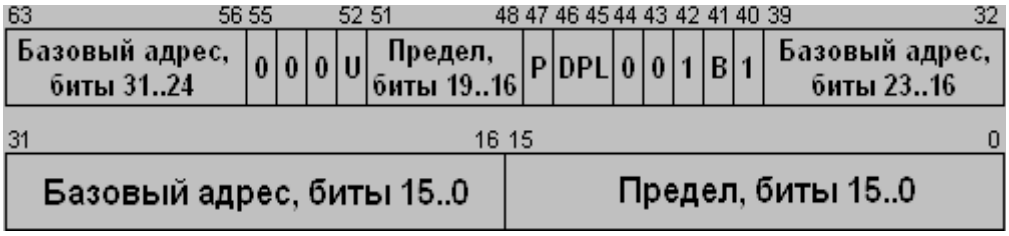

Рисунок 3-2. Формат дескриптора TSS

Дескриптор TSS определяет сегмент состояния задачи. Этот дескриптор описывает системный объект, т.к. бит S в нём равен 0 (это 44-й бит дескриптора или 4-й бит прав доступа). То, что дескриптор описывает системный объект означает, что непосредственно пользоваться этим объектом может только процессор, но не программа - любая попытка загрузки селектора этого дескриптора в сегментный регистр вызовет генерацию исключения общей защиты (#GP - прерывание 0Dh).

Как уже говорилось в предыдущей главе, в локальной дескрипторной таблице LDT нельзя использовать некоторые системные объекты. Т.к. дескрипторы LDT и TSS являются такими объектами, то их нельзя применять в LDT, иначе при обращении к ним будет генерироваться исключения. Смысл такого ограничения состоит в том, что задаче запрещено определять задачи внутри себя и устанавливать дополнительные дескрипторные таблицы и таким образом, эти действия может выполнять только ядро ОС.

Обратите внимание на то, что в сегменте состояния задачи нет полей для регистров модулей FPU, MMX и XMM. Дело в том, что не всякая задача работает с каким-либо из этих модулей и наличие дополнительных полей замедлило бы переключение задачи. Сегмент TSS и так достаточно большой, к тому же поля из него загружаются по одному - процессор выполняет проверку параметров значений каждого сегментного и системного регистра.

Если в системе несколько задач использует FPU, MMX и/или XMM, то при смене задач необходимо менять контекст соответствующего модуля. Процессор автоматически этого не делает, зато он создаёт условия для операционной системы, позволяя ей дополнить контекст задачи, например, сохранить состояние FPU, используемое старой задачей и загрузить в FPU значения, используемые новой задачей.