МУЛЬТИЗАДАЧНОСТЬ В ЗАЩИЩЕННОМ РЕЖИМЕ

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Под мультизадачностью подразумевают способность компьютера выполнять несколько задач одновременно. На самом деле процессор некоторое время выполняет один командный поток, затем быстро переключается на второй и выполняет его, переключается на третий и т.д. При этом при каждом переключении сохраняется контекст прерываемого потока, так что потом процессор сможет "безболезненно" продолжить выполнение прерванного потока команд. Благодаря высокому быстродействию создается иллюзия того, что все задачи выполняются одновременно (параллельно).

Для управления мультизадачностью нет специальных команд. Задачи переключаются командами FAR CALL, FAR JMP, INT, IRET. Однако при этом участвуют специальные дескрипторы: дескриптор сегмента состояния задачи (Task State Segment) и дескриптор шлюза задачи. Когда управление передается на один из таких дескрипторов, происходит переключение задачи. При переключении задачи процессор сохраняет (восстанавливает) свой контекст в сегменте состояния задачи (TSS). Селектор TSS выполняемой задачи хранится в регистре задачи (Task Register). При переключении задачи процессор может сменить LDT, что позволяет назначить каждой задаче свое адресное пространство, недоступное для других задач. Можно также перегрузить CR3, что позволяет применить для изолирования задач механизм страничного преобразования.

Дескриптор TSS относится к системным дескрипторам Поле Type дескриптора TSS может содержать код 1001, если это доступный TSS, или 1011, если это занятый TSS, т.е. если задача активна в настоящий момент.

На рис. 1 представлен формат сегмента TSS для процессора i80386. Из рисунка видно, что в TSS предусмотрены поля для хранения сегментных регистров GS, FS, DS, SS, CS, ES. Имеется поле для хранения содержимого регистра LDTR, указывающего на локальную таблицу дескрипторов, распределённую данной задаче. Для хранения содержимого 32-разрядных регистров используются поля TSS, обозначенные на рисунке как EDI, ESI, EBP, ESP, EBX, EDX, ECX, EAX, EFLAGS, EIP.

Поле CR3 хранит содержимое системного регистра CR3. Этот регистр является указателем на каталог таблиц страниц. Таким образом, каждая задача может иметь свой собственный каталог таблиц страниц, что позволяет выполнить изоляцию задач не только на уровне сегментов, но и на уровне страниц.

Битовая карта ввода/вывода (БКВВ)			
Дополнительная информация ОС			
Относительный адрес БККВ	0	T	64
0	LDTR		60
0	GS		5C
0	FS		58
0	DS		54

0	SS	50
0	CS	4C
0	ES	48
EDI		44
ESI		40
EBP		3C
ESP		38
EBX		34
EDX		30
ECX		2C
EAX		28
EFLAGS		24
EIP		20
CR3		1C
0	SS2	18
ESP2		14
0	SS1	10
ESP1		C
0	SS0	8
ESP0		4
0	LINK	0

Рис. 1 – Сегмент TSS процессора i80386

TSS процессора i80386 содержит указатели на стеки для второго, первого и нулевого приоритетных колец. Это поля SS2:ESP2, SS1:ESP1, SS0:ESP0.

Поле *LINK* используется для ссылки на TSS, вызвавшей задачи при вложенном вызове задач, аналогично тому, как это было в процессоре i80286.

Бит *T* используется для отладки. Если он установлен в 1, при переключении на задачу возникает отладочное исключение, которое может быть использовано системным отладчиком.

Для обеспечения безопасной работы системы необходимо ограничить доступ программам пользователя ко всем или по крайней мере к некоторым портам ввода/вывода. Злонамеренная программа, имеющая доступ к портам контроллера прямого доступа к памяти, может выполнить с помощью этого контроллера чтение или запись информации по любым физическим адресам. Процессор i80286 хранит в регистре флагов уровень привилегий I0PL, на котором разрешено выполнять команды ввода/вывода. С помощью этого механизма можно запретить непривилегированным программам выполнять команды ввода/вывода.

Однако такой способ защиты не слишком удобен. Некоторые порты ввода/вывода не только безопасны для использования, но и весьма полезны для обычных программ (например, порт системного динамика или принтера).

Битовая карта ввода/вывода процессора i80386 позволяет для каждой задачи определить порты, которые эта задача может использовать. То есть операционная система имеет возможность санкционировать любую задачу

для использования любого набора адресов портов ввода/вывода. Если задача попытается обратиться к несанкционированному порту ввода/вывода, произойдёт исключение.

Сегмент TSS содержит поле, обозначенное на рис. 1 как база карты ввода/вывода. Оно служит для указания расположения битовой карты ввода/вывода задачи, использующей данный TSS. Поле базы карты ввода/вывода указывает 16-разрядное смещение начала битовой карты ввода/вывода относительно TSS. Предел TSS должен определяться с учётом карты. Каждый бит в карте ввода/вывода соответствует адресу байта порта ввода/вывода (карта состоит из 64 Кбит для описания доступа к 65536 портам). После битовой карты должен располагаться байт 0FFh.

При выполнении 16- или 32-разрядных операций ввода/вывода процессор проверяет все биты (2 или 4 бита), соответствующие адресу порта. Если проверяемый бит установлен в 1, происходит исключение.

Для привилегированных программ, если уровень привилегий меньше или равен уровню IOPL, процессор не выполняет проверку битовой карты ввода/вывода. Чтобы полностью запретить задаче обращаться к портам ввода/вывода, достаточно установить базу карты ввода/вывода большей или равной пределу TSS. В этом случае любая команда ввода/вывода приведёт к генерации исключения.

Лишь значение первых 68h байт сегмента состояния задачи строго определены. Именно это число является минимальным размером TSS. Операционная система может по своему усмотрению устанавливать размер TSS и включать в сегмент TSS дополнительную информацию, необходимую для работы задачи и зависящую от конкретной операционной системы (например, контекст сопроцессора, указатели открытых файлов или указатели на именованные конвейеры сетевого обмена). Включенная в этот сегмент информация автоматически заменяется процессором при выполнении команды CALL или JMP, селектор которой указывает на дескриптор сегмента TSS в таблице GDT (дескрипторы этого типа могут быть расположены только в этой таблице).

При переключении задачи с помощью прерывания или особого случая происходит автоматический возврат к прерванной задаче. Однако, организуя вложение задач, необходимо помнить, что, в отличие от процедур, при переключении задачи в стек ничего не включается. Дескриптор TSS задачи, выполняемой в данный момент, помечается как занятый. При переключении на другую задачу с вложением (по INT или FAR CALL) дескриптор TSS остается помеченным. Переключиться на занятую задачу нельзя (возникает нарушение общей защиты - исключение №13).

Для переключения задач также действуют правила привилегий. По команде JMP или CALL можно переключиться на задачу, TSS которой менее привилегирован:

$$DPL_{TSS} >= \max(CPL,RPL).$$

Для особых случаев и прерываний это правило не действует. Если обработчик прерывания выполнен в виде отдельной задачи, то он может быть вызван независимо от значения CPL.

Не совсем удобно адресовать именно TSS для переключения задачи, т.к., во-первых, TSS могут быть размещены только в GDT (а в IDT или LDT – нет), а во-вторых, если пользоваться только TSS, то каждую задачу мы

"намертво" привязываем к определенному уровню привилегий (DPL_{TSS}), с которого она доступна для переключения. Этих недостатков лишены шлюзы задачи. Формат дескриптора шлюза задачи приведен на рис. 2.

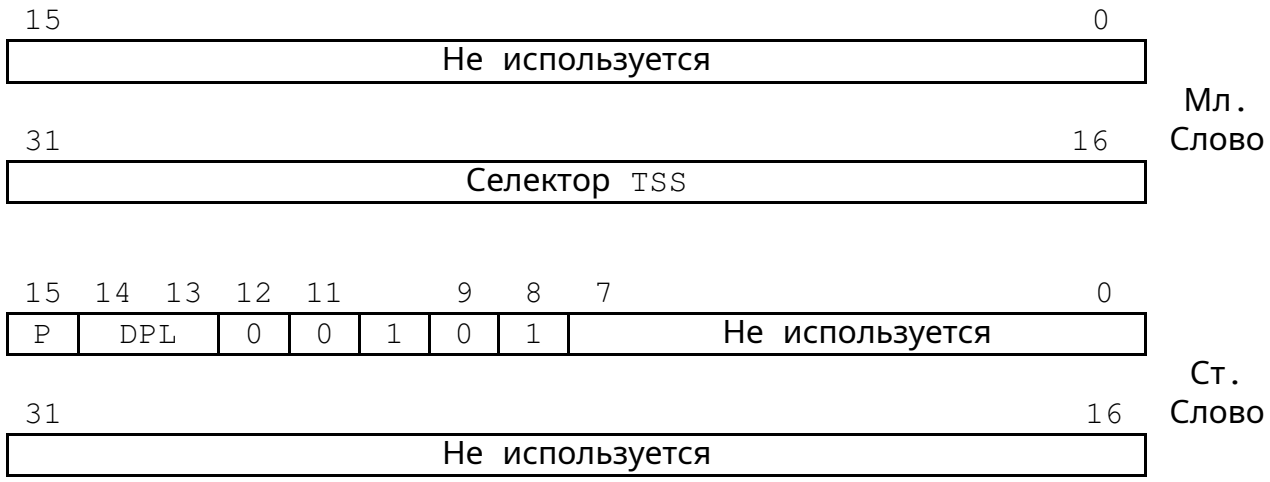

Рис. 2 – Дескриптор шлюза задачи для процессора i80386

Шлюз задачи содержит селектор TSS. Шлюзы задач можно размещать и в IDT, что позволяет выполнять обработчики прерываний в виде отдельных задач, и в LDT, что позволяет более гибко управлять переключением задач: для второй задачи первая может быть видна с одного уровня привилегий, а для третьей – с другого. Последняя возможность обеспечивается особым правилом привилегий: при переключении задачи через шлюз учитывается только $DPL_{шлюза}$, а DPL_{TSS} не играет роли, поэтому одной задаче может соответствовать множество шлюзов с различными DPL.

При переключении задач процессор выполняет следующие действия:

1. Выполняется команда CALL, селектор которой указывает на дескриптор сегмента типа TSS.
2. В TSS текущей задачи сохраняются значения регистров процессора. На текущий сегмент TSS указывает регистр процессора TR, содержащий селектор сегмента.
3. В TR загружается селектор сегмента TSS задачи, на которую переключается процессор.
4. Из нового TSS в регистр LDTR переносится значение селектора таблицы LDT в таблице GDT задачи.
5. Восстанавливаются значения регистров процессора (из соответствующих полей нового сегмента TSS).
6. В поле селектора возврата заносится селектор сегмента TSS снимаемой с выполнения задачи для организации возврата к прерванной задаче в будущем.

Вызов задачи через шлюз происходит аналогично, добавляется только этап поиска дескриптора сегмента TSS по значению селектора дескриптора шлюза вызова.

2. Контрольные вопросы

1. Какая обязательная информация сохраняется в сегменте состояния задачи?
2. Какая дополнительная информация может быть сохранена в сегменте состояния задачи?
3. Для чего используется битовая карта ввод/вывода?
4. Какие существуют в защищенном режиме способы переключения задач? Сравните их.
5. Какие действия выполняет процессор при переключении задач?