ЛЕКЦІЯ №7. БАГАТОЗАДАЧНИЙ РЕЖИМ ТА СТОРІНКОВА ОРГАНІЗАЦІЯ ПАМ'ЯТІ

Організація багатозадачного режиму.

Перемикання задач.

Використання таблиць локальних дескрипторів дозволяє рознести фізичні адресні простори окремих задач без права їх звертання до "чужої" пам'яті. Система привілеїв, яка входить в архітектуру процесору, запобігає несанкціонованим викликам задачами захищених процедур.

В простих випадках паралельне виконання декількох задач здійснюється на одному рівні привілеїв.

Захист адресних просторів чи процедур здійснюється шляхом ретельного написання програм комплексу.

Організація багатозадачного режиму засновується на наступних апаратних і програмних засобах:

- сегмент стану задачі (TSS-Task State Segment);
- дескриптор сегменту стану задачі;
- регістр задачі (Task Register,TR);
- дескриптор шлюзу задачі (Task gate).

Сегмент стану задачі (TSS) - це поле даних. Воно може бути включене до складу сегменту даних, або утворювати окремий сегмент невеликого розміру. Кожна задача, яка бере участь в процесі перемикання, повинна мати свій TSS.

Оскільки TSS представляється процесору окремим сегментом, йому повинен відповідати дескриптор. TSS описується системним дескриптором, який може знаходитися тільки в GDT.

Формат системного дескриптору (рис. 1) майже однаковий з форматом дескриптору пам'яті. Відсутній біт замовчання D і код дескриптора – 0, а не 1.

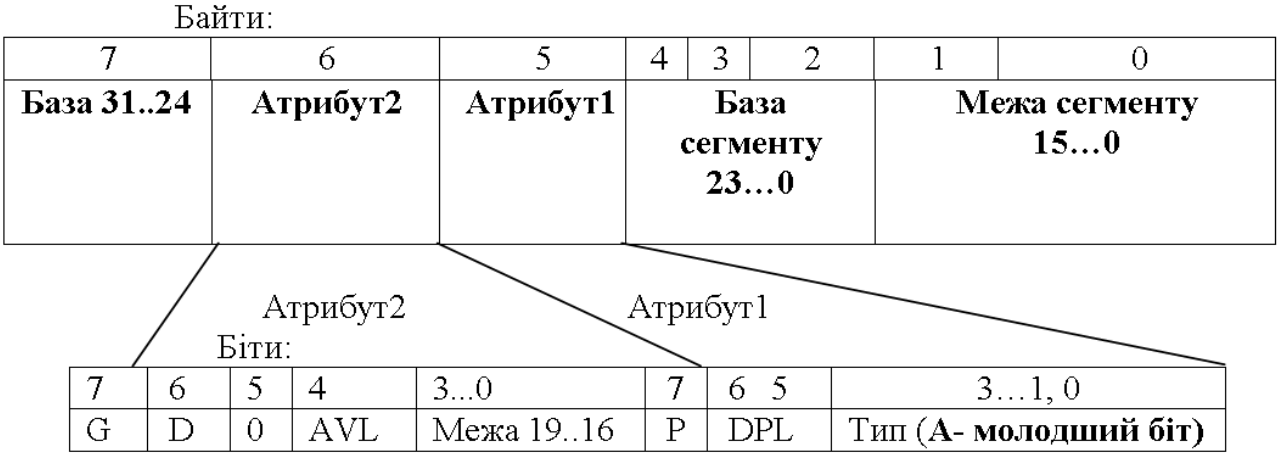

Рис. 1 Формат системного дескриптору

Табл. 1 Тип системного дескриптору може приймати низку значень

Тип	Призначення дескриптору
0	Не визначений
1	Вільний сегмент стану задачі (TSS) 80286
2	LDT
3	Зайнятий сегмент стану задачі (TSS) 80286
8	Не визначений
9	Вільний сегмент стану задачі (TSS) 80386…
Ah	Не визначений
Bh	Зайнятий сегмент стану задачі (TSS) 80386…
Dh	Не визначений

В залежності від порядкового номеру дескриптору TSS в таблиці дескрипторів, йому відповідає той чи інший селектор. Селектор TSS активної задачі повинен бути завантажений в регістр задачі TR. Для головної задачі це завантаження здійснюється програмно за допомогою команди ltr (load task register). При перемиканні на нову задачу програма передає процесору селектор нового TSS, і перевантаження регістру TR процесор здійснює в ході перемикання задач.

Перемикання на нову задачу здійснюється командою далекого виклику call dword ptr або далекого переходу jmp dword ptr. В якості аргументу цих команд вказується двослівне поле, в першому слові якого записується селектор потрібного TSS. Існує і інший засіб перемикання – не через селектор TSS, а через шлюз задачі (дескриптор шлюзу з кодом типу 5).

В цьому випадку селектор потрібного TSS вказується в полі для селектору в шлюзі задачі. Перемикання через шлюз задачі має ту перевагу, що його можна

здійснити за апаратним перериванням (шлюз задачі можна розмістити в таблиці дескрипторів переривань).

В процесі перемикання задач процесор зберігає контекст поточної задачі в TSS і завантажує новий контекст із TSS нової задачі (включаючи селектор і відносну адресу точки входу в задачу). Контекст задачі: EAX, EBX,…, SS:ESP, CS:EIP (адреса наступної команди).

Задача, на яку здійснюється перемикання, повинна завершуватися командою iret, яка обробляється процесором не так, як iret звичайного обробника переривання. В випадку перемикання задач процесор за командою iret виконує зворотнє переміщення контекстів. Контекст завершеної задачі зберігається в її TSS, а в регістри процесору завантажується з TSS вхідної задачі збережений там контекст, відповідний моменту перемикання на іншу задачу. Таким чином, TSS можна розглядати, як функціональний аналог стеку. Але зберігання в стеку та відновлення зі стеку здійснюється за допомогою послідовних команд процесора, а зберігання і відновлення контекстів за допомогою TSS здійснюється процесором апаратно в процесі перемикання задач.

Поля TSS вхідної задачі заповнюються процесором при першому перемиканні на нову задачу. Програміст може не піклуватися про їхній вміст. Поля TSS задачі, на яку відбувається перемикання, містять адресу точки входу і вхідний вміст сегментних регістрів і регістрів загального призначення. Ці поля повинні бути заповнені у вхідній задачі перед перемиканням на нову.

Вміст TSS. (Розмір TSS мінімум 104 байти). На початку TSS розташоване 16-бітове поле зв'язку, яке використовується при перемиканні задач. Для вхідної задачі його вміст не має значення.

При перемиканні на нову задачу процесор заносить у поле зв'язку TSS нової задачі селектор TSS вхідної задачі. У результаті створюється зв'язний список вкладених задач (рис. 2).

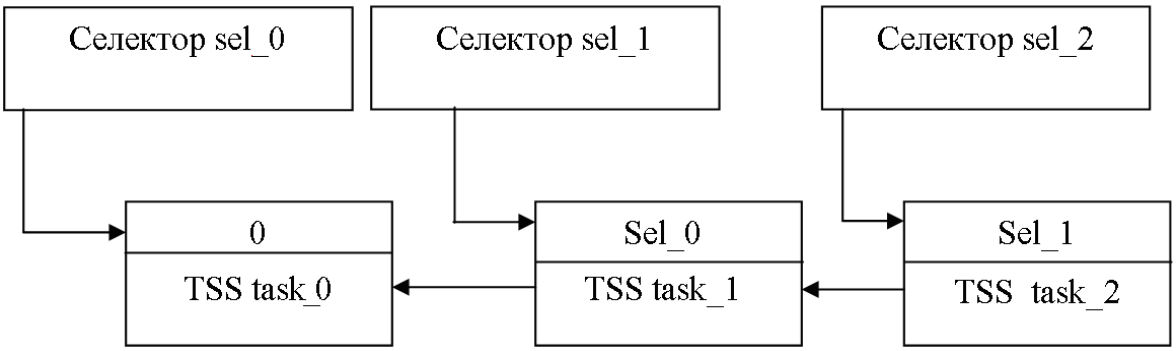

Рис. 2 Зв'язний список вкладених задач

Табл. 4.2 Формат сегменту стану задачі (TSS)

Зсув від бази TSS			
0	Зв’язок		0h
ESP0			04h – кадр стеку рівня привілеїв 0
0	SS0		08h
ESP1			0ch - кадр стеку рівня привілеїв 1
0	SS1		10h
ESP2			14h - кадр стеку рівня привілеїв 2
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	LDT		60h
Адреса карти в/в	000…000	T	64h
Карта в/в (якщо вона є)			68h

Команда iret, якою завершується кожна вкладена задача, виконує зворотнє перемикання задач. В ході цієї операції процесор витягає з поля зв’язку TSS поточної задачі селектор TSS попередньої (по порядку вкладеності), відновлює з TSS її контекст і передає їй керування.

За адресами 04h, 0Ch, 14h відносно бази TSS розташовуються кадри стеків рівнів привілеїв 0,1,2. Вміст цих полів завантажується в регістри SS і ESP, якщо при перемиканні задачі відбувається зміна рівнів привілеїв.

За адресою 1Ch міститься поле регістру керування CR3. Він містить базову адресу каталогу сторінки та використовується, якщо ввімкнене сторінкове перетворення. Наявність у TSS цього поля дозволяє мати для кожної задачі свій каталог сторінок і свої таблиці відображення віртуальних сторінок програми на фізичні адреси пам'яті.

Двослівне поле TSS за адресою 20h служить для збереження значення покажчика команд EIP. У TSS вхідної задачі це поле процесор заповнює адресою повернення. У TSS задачі, на яку відбувається перемикання, поле для EIP повинно бути заповнене програмно зсувом точки входу в задачу. При поверненні в стару задачу командою iret процесор записує в поле для EIP задачі, що завершилася, адресу команди, яка йде за iret, як адресу повернення. Тобто адресу вже за межами задачі. Якщо планується повторне перемикання на цю задачу, то перед кожним перемиканням необхідно відновлювати в її TSS адресу точки входу.

За адресою 24h у TSS зберігається поточний вміст регістру прапорів EFLAGS. Це дозволяє здійснювати перемикання задач у будь-якій точці задачі без втрати її працездатності.

Частина TSS з адресами 28h –47h відведена для збереження регістрів загального призначення. При перемиканні з вхідної задачі на нову в цих полях TSS вхідної задачі зберігається поточний вміст регістрів, при зворотному перемиканні командою iret відновлюється їхній уміст із цих полів. Заповнивши раніше поля регістрів у TSS нової задачі, можна передати їй вхідні параметри.

Молодші половини шести двослівних полів, починаючи з адреси 48h відведені під уміст сегментних регістрів.

Якщо нова задача використовує LDT, її селектор потрібно занести в TSS за адресою 60h .

Біт 0 слова за адресою 64h використовується для налагодження задач, які перемикаються. Якщо в TSS нової задачі встановлений цей біт, то відразу ж після перемикання генерується виключення налагодження з номером 1. Інші біти слова за адресою 64h у TSS повинні дорівнювати 0.

Слово за адресою 66h містить зсув бітової карти в\в, яка, якщо вона є, розташовується за наступними адресами і використовується для захисту портів комп'ютеру від НСД. Кожен біт цієї карти відповідає одному порту (64 Кбітів або 8 Кбайтів).

Якщо біт, закріплений за деяким портом, дорівнює 1, то при звертанні до порту задачею з недостатньо високим рівнем привілеїв генерується виключення загального захисту. Якщо він =0, то задача з будь-яким рівнем привілеїв може звертатися до порту.

У біті 14 регістру прапорів NT (Nested Task, вкладена задача) встановлюється режим виконання команди iret. Команда iret аналізує стан прапору NT і якщо він скинутий, здійснює звичайне повернення з програми обробки переривання (через стек). Якщо прапор NT встановлений, то iret ініціює зворотнє перемикання задач через селектор у TSS.

Після завантаження комп'ютера прапор NT встановлений. Але будь-яке програмне, апаратне переривання чи виключення скидає цей прапор. Після завершення обробника NT знову встановлений (iret відновлює вхідний вміст регістру прапорів).

При перемиканні задач через шлюз задачі чи через TSS, процесор зберігає в TSS поточної задачі слово прапорів і встановлює в регістрі прапорів біт NT.

Дескриптори-псевдоніми.

У багатьох випадках, працюючи в захищеному режимі, потрібно розширити права програми при звертанні до сегментів. У тих випадках, коли потрібні характеристики, не передбачені серед стандартних типів дескрипторів, розширення прав доступу до сегменту здійснюється за допомогою опису для того ж самого сегменту нового дескриптору з іншими правами доступу. Наприклад, якщо потрібно не тільки виконувати рядки сегменту команд, але й читати їх, то дескриптору цього сегменту потрібно привласнити тип 5. Якщо ж потрібно сегмент ще й модифікувати, то його потрібно описати двома дескрипторами: з типом 5 і типом 1.

Такий додатковий дескриптор, що розширює права доступу до сегменту, називається дескриптором-псевдонімом.

Процесор забороняє звертання до сегментів стану задач з метою їх читання чи модифікації. Селектор дескрипторів TSS можна використовувати тільки для перемикання задач. Завантажити такий селектор у сегментний регістр не можна. Для того, щоб у захищеному режимі виконати налаштування сегменту стану задачі, для нього необхідно створити дескриптор-псевдонім з атрибутом дозволу читання-запису. Завантаживши в який-небудь сегментний регістр селектор цього дескриптору, можна звертатися до цього сегменту як до звичайного сегменту даних.