ЛЕКЦІЯ №5. АДРЕСНИЙ ПРОСТІР ТА СПОСОБИ АДРЕСАЦІЇ В РЕАЛЬНОМУ І ЗАХИЩЕНОМУ РЕЖИМАХ.

Програмування захищеного режиму. Особливості 32-розрядних процесорів

Процесор містить близько 40 програмно-адресуємих регістрів (не рахуючи регістрів співпроцесорів), з яких 6 є 16-розрядними, а інші – 32-розрядними.

Регістри даних:

eax – акумулятор;

ebx - базовий регістр;

ecx - лічильник;

edx - регістр даних;

Регістри-покажчики:

esi - індекс джерела;

edi - індекс приймача;

ebp - покажчик бази;

esp - покажчик стеку.

Сегментні регістри – 16-розрядні.

- CS, DS, ES, FS, GS, SS.

Регістри загального призначення і регістри–покажчики відрізняються від аналогічних регістрів процесору 8086 тим, що вони є 32-розрядними.

Для збереження сумісності з ранніми моделями процесорів допускається звертання до молодших половин усіх регістрів, що мають ті ж мнемонічні позначення, що й у МП8086 (AX, BX, CX, DX, SI, DI, BP і SP).

Збережено можливість роботи з молодшими і старшими половинками регістрів МП8086. Однак старші половини 32-розрядних регістрів процесора не мають мнемонічних позначень і безпосередньо недоступні.

Для того, щоб одержати вміст старшої половини, наприклад, регістру EAX (біти 31...16) прийдеться зсунути вміст EAX на 16 біт праворуч (у регістр AX) і прочитати потім уміст регістру AX.

До складу сегментних регістрів включено ще 2 регістри: FS і GS, що можуть використовуватися для збереження сегментних адрес двох додаткових сегментів

даних. Тобто при роботі в реальному режимі з програми можна забезпечити доступ до чотирьох сегментів даних, а не до двох, як при використанні МП8086.

Регістр прапорів процесору 486 прийнято називати EFLAGS. Додатково до шести прапорів стану (CF, PF, AF, ZF, SF і OF) і трьох прапорів керування станом процесору (TF, IF, DF), він включає 3 нових прапори:

NT, RF і VM і дворозрядне поле IOPL.

Нові прапори NT, RF, VM і поле IOPL використовуються процесором тільки в захищеному режимі.

Дворозрядне поле привілеїв введення-виведення IOPL (Input/Output Privilege Level) указує на максимальне значення рівня поточного пріоритету (від 0 до 3), при якому команди в/в виконуються без генерації виключної ситуації.

Прапор вкладеної задачі NT (Nested Task) показує, чи є поточна задача вкладеною у виконання іншої задачі. У цьому випадку NT=1. Прапор встановлюється автоматично при перемиканні задач. Значення NT перевіряється командою iret для визначення засобу повернення в процедуру, що викликала.

Керуючий прапор рестарту RF (Restart Flag) використовується разом з налагоджуючими регістрами. Якщо RF=1, то помилки, що виникають під час налагодження при використанні команди, ігноруються до виконання наступної команди.

Керуючий прапор віртуального режиму VM (Virtual Mode) використовується для переведення процесора з захищеного режиму в режим віртуального процесора 8086. У цьому випадку процесор функціонує як швидкодіючий МП 8086, але реалізує механізми захисту пам'яті, сторінкової адресації і ряд інших можливостей.

При роботі з процесором програміст має доступ до чотирьох керуючих регістрів CR0...CR3, у яких міститься інформація про стан комп'ютера. Ці регістри доступні тільки в захищеному режимі для програм, що мають рівень привілеїв 0.

Регістр CR0 - це слово стану системи.

Для керування режимом роботи процесора і визначення його стану використовуються наступні шість бітів регістру CR0.

Біт 6 – біт сторінкового перетворення PG (Paging Enable). Якщо цей біт установлений, то сторінкове перетворення дозволене; якщо скинутий, то заборонено.

Біт 4 – біт типу співпроцесора ET (Extension Type) у ПМ 80286 і 80386 вказував на тип підключеного співпроцесора. Якщо ET=1, то 80387, якщо ET=0, то 80287. У більш нових процесорах біт ET завжди встановлений.

Біт 3 – біт перемикання задачі TS (Task Switched). Цей біт автоматично встановлюється процесором при кожному перемиканні задачі. Біт може бути очищений командою *clts*, яку можна використовувати тільки на нульовому рівні привілеїв.

Біт 2 – біт емуляції співпроцесору EM (Emulate). Якщо EM=1, то обробка команд співпроцесору виконується програмно.

Біт 1 – біт присутності арифметичного співпроцесору MP (Math Present). Операційна система встановлює MP=1, якщо співпроцесор присутній. Цей біт керує роботою команди wait, яка використовується для синхронізації роботи програми і співпроцесору.

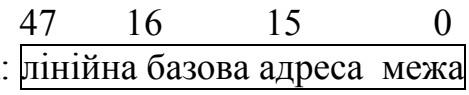
Біт 0 – біт дозволу захисту PE (Protection Enable). При PE=1 процесор працює в захищеному режимі; при PE=0 – у реальному. PE може бути встановлений при завантаженні регістру CR0, командами *lmsw* чи *mov cr0*, а скинутий тільки командою *mov cr0*.

Регістр CR1 зарезервований фірмою INTEL для наступних моделей процесорів. Регістри CR2 і CR3 служать для підтримки сторінкового перетворення адреси. Ці два регістри використовуються разом. CR2 містить повну лінійну адресу команди, що викликала останню виключну ситуацію на сторінці, а CR3 – адресу, що вказує базу каталогу сторінки.

Регістри системних адрес.

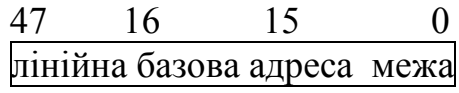
Регістри системних адрес входять до складу процесору і використовуються в захищеному режимі роботи процесору. Вони задають розташування системних таблиць, що служать для організації системної адресації в захищеному режимі.

Регістр таблиці глобальних дескрипторів.



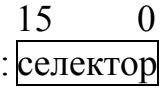
GDTR (Global Descriptor Table Register) – служить для збереження лінійної базової адреси і межі таблиці глобальних дескрипторів.

Регістр таблиці дескрипторів переривань.



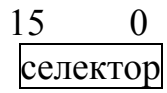
IDTR (Interrupt Descriptor Table Register) – для збереження лінійної базової адреси і межі таблиці дескрипторів переривань.

Регістр таблиці локальних дескрипторів.



LDTR (Local Descriptor Table Register) – для збереження селектору сегменту таблиці дескрипторів.

Регістр стану задачі.



TR (Task Register) – для збереження селектору сегменту стану задачі.

У систему команд сучасних процесорів включений ряд нових команд. Команди загального призначення.

bound – перевірка індексу масиву щодо границь масиву.

bsf/bsr – команди сканування бітів.

bt/btc/btr/bts – команди виконання бітових операцій.

bswap – зміна порядку байтів операнду.

cdq – перетворення подвійного слова на четверне.

cmpsd – порівняння рядків за подвійними словами.

cmpxchg – порівняння й обмін операндів.

cmpxchg86 – порівняння й обмін 8-бітових операндів.

cpuid – ідентифікація процесору.

cwde – перетворення слова на подвійне слово з розширенням

enter – створення кадру стеку для параметрів процедур.

imul reg, imm – множення операнду зі знаком на безпосереднє значення.

ins/outs – введення/виведення з порту в рядок.

iretd – повернення з переривання в 32-розрядному режимі.

j(cc) – команди умовного переходу, що допускають 32-бітовий зсув.

leave – вихід із процедури з видаленням кадру стеку, створеного командою enter.

lss/lfs/lgs – команди завантаження сегментних регістрів.

mov DRx, reg; reg, DRx

mov CRx, reg; reg, CRx

mov TRx, reg; reg, TRx – команди обміну даними зі спеціальними регістрами. Як джерело приймача можуть бути використані регістри CR0...CR3, DR0 – DR7, TR3 – TR5.

movsx/movzx – знакове/беззнакове розширення до розміру приймача і передатчика.

popa – витяг зі стеку всіх 16-розрядних регістрів загального призначення (ax, bx, cx, dx, sp, bp, si, di).

popad – витяг зі стеку всіх 32-розрядних регістрів загального призначення (eax, ebx, ecx, edx, esp, ebp, esi, edi).

push imm – запис у стек безпосереднього операнду розміром байт, слово, подвійне слово (наприклад, push 0FFFFFFFFh).

pusha – запис у стек усіх 16-розрядних регістрів загального призначення.

pushad – запис у стек усіх 32-розрядних регістрів загального призначення.

rcr/rcl/ror/rol reg/mem, imm – циклічний зсув на безпосереднє значення.

sar/sal/shr/shl reg/mem, imm – арифметичний зсув на безпосереднє значення.

scads – сканування рядка подвійних слів з метою порівняння.

set(cc) – встановлення біту за умовою.

shrd/shld – логічний зсув з подвійною точністю.

stosd – запис подвійного слова в рядок.

xadd – обмін і додавання.

xlatb – таблична трансляція.

Команди захищеного режиму.

arpl – коригування поля RP2 селектору.

clts – скид прапору перемикання задач у регістрі CR0.

lar – завантаження байту дозволу доступу.

lgdt – завантаження регістру таблиці глобальних дескрипторів.

lidt – завантаження регістру таблиці переривань.

lldt – завантаження регістру таблиці локальних дескрипторів.

lmsw – завантаження слова стану машини.

lsl – завантаження межі сегменту.

ltr – завантаження регістру задачі.

rdmsr – читання особливого регістру моделі.

sgdt – збереження регістру таблиці глобальних дескрипторів.

sidt – збереження регістру таблиці переривань.

sedt – збереження регістру таблиці локальних дескрипторів.

smsw – збереження слова стану.

ssl – збереження межі сегменту.

str – збереження регістру задачі.

verr – перевірка доступності сегменту для читання.

verm – перевірка доступності сегменту для запису.

Знайомство з захищеним режимом.

Переведенням мікропроцесору в захищений режим можна цілком реалізувати всі можливості, закладені в його архітектуру і недоступні в реальному режимі. Сюди можна віднести:

- збільшення адресного простору до 4 Гбайт;

- можливість працювати у віртуальному адресному просторі, що перевищує максимально можливий обсяг фізичної пам'яті і сягає величини 64 Тбайти.

Для реалізації такого режиму потрібна відповідна операційна система, що зберігає всі сегменти виконуваних програм у великому дисковому просторі, автоматично завантажуючи в оперативну пам'ять ті чи інші сегменти за потребою.

Організація багатозадачного режиму з паралельним виконанням декількох програм (процесів). Такий режим організує багатозадачна операційна система, але

мікропроцесор надає для цього режиму механізм захисту задач одна від одної за допомогою чотирьохрівневої системи привілеїв.

Сторінкова організація пам'яті, що підвищує рівень захисту задач одна від одної, та ефективність їх виконання.

Програми, написані для захищеного режиму дуже складні. Але їх можна спростити, використавши те, що окремі архітектурні особливості захищеного режиму виявляються достатньо замкнутими і незалежними. Так, при роботі в однозадачному режимі відпадає необхідність у вивченні різноманітних методів захисту задач одна від одної. У багатьох випадках можна не включати механізм сторінкової організації пам'яті. Часто немає необхідності використовувати рівні привілеїв.

На відміну від реального режиму, у якому сегменти визначаються їхніми базовими адресами, що задаються програмістом у явній формі, у захищеному режимі для кожного сегменту програми повинен бути визначений дескриптор – 8-байтове поле, у якому записується базова адреса сегменту, його довжина й інші характеристики.

Для звертання до необхідного сегменту програміст заносить у сегментний регістр не сегментну адресу, а селектор, до складу якого входить номер відповідного сегменту дескриптору.

Біти	15 3	2	1	0
Назва	Індекс дескриптору (номер)	TTI	RPL	

Рис. 1 Селектор дескриптору:

Де: RPL – рівень привілеїв; TI=0 дескриптор глобальний; TI=1дескриптор локальний.

Процесор за цим номером знаходить потрібний дескриптор, витягає з нього базову адресу сегменту, і додаючи до неї зазначений в конкретній команді зсув, формує чарунки пам'яті. Індекс дескриптору записується в селектор починаючи з біту 3, що еквівалентно множенню його на 8 (2^3). Таким чином, можна вважати, що селектори послідовних дескрипторів являють собою числа 0, 8, 16, 26....

Дескриптори можуть бути 3-х типів: пам'яті, системні і шлюзи. Формати дескрипторів пам'яті і системних збігаються. Формат дескриптору сегменту пам'яті зображено на рисунку 3.2.

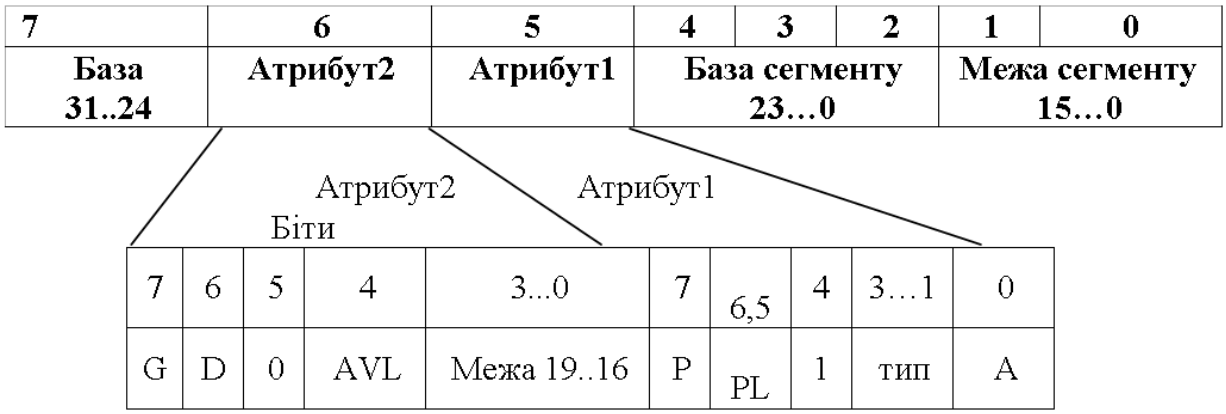


Рис. 2 Формат дескриптору сегменту пам'яті

Дескриптор займає 8 байтів і призначений для використання прикладними програмами. У байтах 2-4 і 7 записується лінійна сегментна адреса бази сегменту. Так як базова адреса має 32 біти, вона може бути призначена у будь-якій точці 4-гігабайтного адресного простору.

У байтах 0,1 записуються молодші 16 бітів межі сегменту, а в молодші 4 біти байту атрибутів 2 – біти 16...19, що залишилися. Межа описується 20-ма бітами і її числове значення не може перевищувати 1М (2^20) .

Але одиниці, в яких задається межа, можна змінювати, що здійснюється за допомогою біту дробовості G (біт 7 байту атрибутів2).

Якщо G=0, межа вказується у байтах, якщо G =1 - у блоках по 4К. База сегменту в обох випадках задається з точністю до байту.

Якщо G =1, дійсна межа сегменту визначається наступним чином «Межа сегменту = Межа в дескрипторі *4К+4095».

Таким чином, сегмент завжди простирається до кінця останнього 4К-байтного блоку.

Нехай, наприклад, базова адреса сегменту дорівнює 0, межа теж дорівнює 0 і біт дробовості встановлений. Тоді дійсна межа сегменту дорівнює: 0*4К+4095=4095.

Тобто, сегмент буде займати адреси 0...4095 і мати розмір 4-Кбайти. Якщо установити значення межі=1, розмір сегменту буде 8 Кбайт і т.д.

Розглянемо атрибути сегменту:

Біт А - молодший біт типу (Accessed, було звертання) установлюється процесором у той момент, коли в який-небудь сегментний регістр завантажується селектор даного сегменту. Аналізуючи біти звертання різних сегментів, програма може судити про те, чи було звертання до даного сегменту після того, як вона скинула біт А.

Тип сегменту - займає 3 біти і може мати 8 значень:

0 - дозволене тільки читання (сегмент даних);

1 - дозволене читання і запис (сегмент даних);

2 - розширення вниз, дозволене тільки читання(сегмент стеку);

3 - розширення вниз, дозволені читання і запис (сегмент стеку);

4 - дозволене тільки виконання (сегмент команд);

5 - дозволене виконання і читання (сегмент команд);

6 - дозволене тільки виконання (підлеглий сегмент);

7 - дозволене виконання і читання (підлеглий сегмент).

Тип визначає правила доступу до сегменту. Таким чином, в захищеному режимі не передбачене складання програм, які самі модифікуються.

Підлеглі чи узгоджені сегменти (conforming) зазвичай використовуються для зберігання підпрограм загального користування; для них не діють загальні правила захисту програм одна від одної.

Сегменти з розширенням вниз, тобто в бік менших адрес, використовуються для організації стеків, які по ходу виконання програми мусять розширюватися.

Біт 4 байту атрибутів1 є ідентифікатором сегменту. Якщо він дорівнює 1, дескриптор описує сегмент пам'яті, 0 - це дескриптор системного сегменту.

Поле DPL (Descriptor Privilege Level) служить для захисту програм одна від одної. Програмам операційної системи зазвичай призначається рівень 0, прикладним програмам - рівень 3.

Біт P - присутність сегменту в пам'яті.

Біт AVL (від Available - доступний) не використовується та не реалізується процесором.

Біт D (Default, замовчання) визначає діючий за замовчанням розмір для операндів і адрес. Якщо D=0, то використовуються 16-бітові, D=1 - 32-бітові. Він змінює характеристики сегментів 2-х типів: виконуємих і стеку. Атрибут сегменту, що діє за замовчанням, можна змінити на протилежний за допомогою префіксів зміни розміру операнду (66h) і зміни розміру адреси (67h). Таким чином, після сегменту з D=0 префікс 66h перед деякою командою змушує її розглядати свої операнди як 32-бітові, а для сегменту з D=1 той самий префікс 66h, навпаки, зробить операнди 16-бітовими .

Іноді транслятор сам включає в об'єктний модуль необхідні префікси. При використанні команди iret у захищеному режимі префікс 66h необхідно включати в програму в явному вигляді, щоб процесор зняв зі стеку не три слова, як звичайно, а три подвійних слова. Без префіксу команда iret буде виконуватись невірно.

Префікс зміни адреси 67h необхідно використовувати, наприклад, перед командою loop, якщо в якості лічильника циклу використовується не CX, а ECX.

Сегменти стеку, що адресуються через регістр SS, за допомогою біту D зазначають, який регістр використовувати як покажчик стеку в командах push і pop. Якщо D=0 використовується регістр SP, якщо D =1, то ESP.

У байті атрибутів1 задаються характеристики сегменту. Наприклад, для сегменту команд цей байт може мати значення 98h (сегмент, що виконується), для сегменту даних цей байт може мати значення 92h (дозволені запис і читання).

Додаткові характеристики сегменту вказуються в старшому напівбайті байту атрибутів 2 (зокрема, тип дробовості).

Сегмент даних у програмі повинен починатися з опису таблиці глобальних дескрипторів. У таблиці дескрипторів повинно бути описано стільки дескрипторів, скільки сегментів використовує програма. Порядок дескрипторів у таблиці не має значення, крім нульового, який завжди займає перше місце.

Сегменти команд у захищеному режимі не можна модифікувати по ходу виконання програми.