ОРГАНІЗАЦІЯ КЕШ-ПАМ'ЯТІ

Кеш-пам'ять персональних комп'ютерів є високошвидкісним буфером, побудованим на мікросхемах SRAM (Static RAM – статична оперативна пам'ять), який безпосередньо обмінюється даними з процесором. Така пам'ять наявна у всіх 32-розрядних сучасних процесорах.

Як уже наголошувалося, як елементна база основної пам'яті в більшості ОМ використовуються мікросхеми динамічних ОЗП, швидкодія яких на порядок нижче швидкодії центрального процесора. В результаті процесор вимушений простоювати декілька тактових періодів, поки інформація з ІМС пам'яті встановиться на шині даних ОМ. Якщо ОП виконати на швидких мікросхемах статичної пам'яті, вартість ОМ істотно зросте. Економічно прийнятне вирішення цієї проблеми було запропоноване М. Уїлксом у 1965 році в процесі розробки ОМ Atlas і полягає воно у використанні дворівневої пам'яті, коли між ОП і процесором розміщується невелика, але швидкодіюча буферна пам'ять. У процесі роботи такої системи в буферну пам'ять копіюються ті ділянки ОП, до яких проводиться звернення з боку процесора. В загальноприйнятій термінології – проводиться відображення ділянок ОП на буферну пам'ять. Виграш досягається за рахунок раніше розглянутої властивості локальності – якщо відобразити ділянку ОП в більш швидкодіючу буферну пам'ять і переадресувати на неї всі звернення в межах скопійованої ділянки, можна добитися істотного підвищення продуктивності ОМ.

Уїлкс називав дану буферну пам'ять підпорядкованою (slave memory). Пізніше поширення набув термін кеш-пам'ять (від англійського слова cache – притулок, тайник), оскільки така пам'ять зазвичай прихована від програміста в тому сенсі, що він не може її адресувати і може навіть взагалі не знати про її існування. Вперше кеш-системи з'явилися в машинах моделі 85 сімейств IBM 360.

1 Загальні питання кешування пам'яті

У загальному вигляді використання кеш-пам'яті пояснимо таким чином. Коли ЦП намагається прочитати слово з основної пам'яті, спочатку здійснюється пошук копії цього слова в кеші. Якщо така копія існує, звернення до ОП не проводиться, а в ЦП передається слово, що є вибраним з кеш-пам'яті. Дану ситуацію прийнято називати успішним зверненням або попаданням (hit). За відсутності слова в кеші, тобто у разі неуспішного звернення – промаху (miss), – необхідне слово передається в ЦП з основної пам'яті, але одночасно з ОП в кеш-пам'ять пересилається блок даних, що містить це слово.

На рис. 1 приведена структура системи з основною і кеш-пам'яттю.

ОП складається з 2n слів, що адресуються, де кожне слово має унікальну n-розрядну адресу. Під час взаємодії з кешем ця пам'ять розглядається як M блоків фіксованої довжини по K слів у кожному (M = 2n/K).

Кеш-пам'ять складається із С блоків аналогічного розміру (блоки в кеш-пам'яті прийнято називати рядками), причому їх число значно менше числа блоків у основній пам'яті (С << М). У разі зчитування слова з якого-небудь блока ОП цей блок копіюється в один з рядків кеша. Оскільки число блоків ОП більше числа рядків, окремий рядок не може бути постійно виділений одному і тому ж блоку ОП. З цієї причини кожному рядку кеш-пам'яті відповідає тег (ознака), що містить відомості про те, копія якого блока ОП у даний момент зберігається в даному рядку.

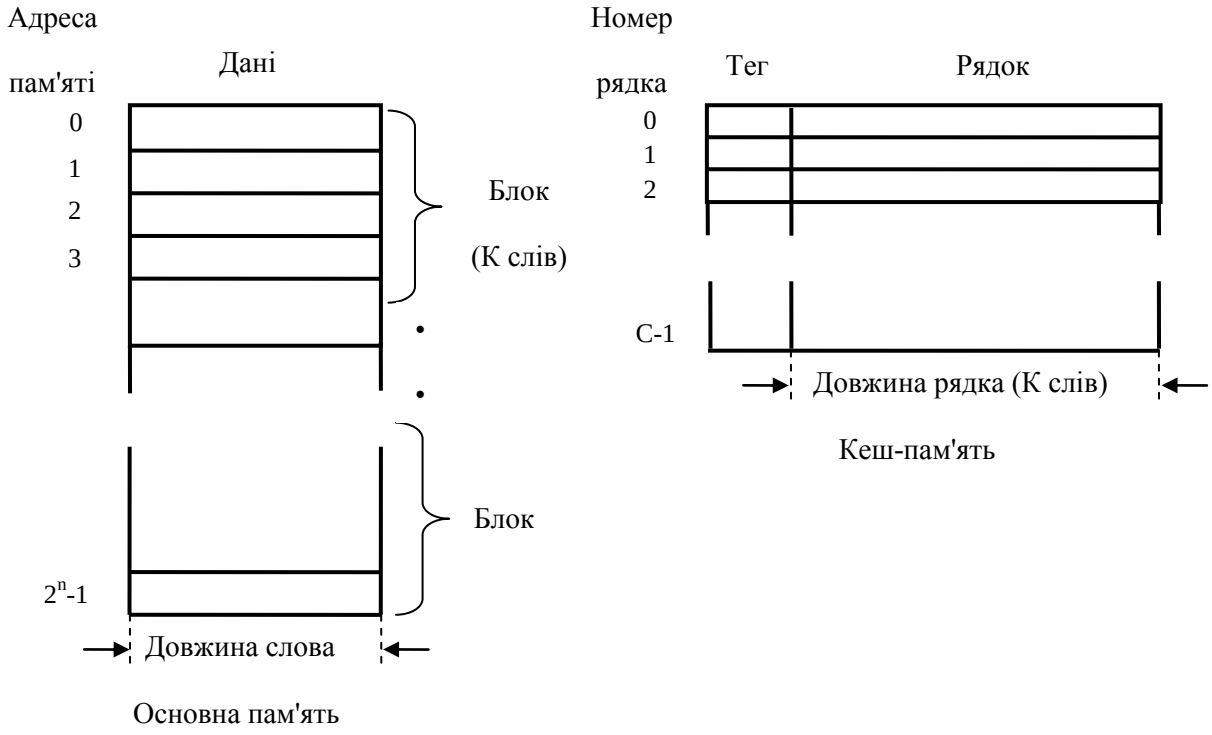

Рисунок 1 - Структура системи з основною і кеш-пам'яттю

Інформація про те, який саме блок займає даний рядок, і про його стан зберігається у пов'язаній з даним рядком комірці спеціальної пам'яті тегів (tag RAM). Для процесорів i486 і старших процесорів P5 довжина рядка збігається з об'ємом даних, що передаються за один пакетний цикл (для i486 це – 4 x 4 = 16 байт, для Pentium – 4 x 8 = 32 байт). Можливий також варіант секторованого (sectored) кеша, в якому один рядок містить кілька суміжних комірок – секторів. Їх розміри відповідають мінімальній порції обміну даних кеша з оперативною пам'яттю. При цьому в записі каталогу, що відповідає кожному рядку, повинні зберігатися біти дійсності для кожного сектора даного рядка. Секторування економить пам'ять, яка необхідна для зберігання каталогу у разі збільшення ємності кеша, оскільки більша кількість бітів каталогу відводиться під тег.

На ефективність застосування кеш-пам'яті в ієрархічній системі пам'яті впливає цілий ряд моментів. До найбільш істотних з них можна віднести:

- ємність кеш-пам'яті;
- розмір рядка;
- спосіб відображення основної пам'яті на кеш-пам'ять;
- алгоритм заміщення інформації в заповненій кеш-пам'яті;
- алгоритм узгодження вмісту основної і кеш-пам'яті;
- число рівнів кеш-пам'яті.

Вибір ємності кеш-пам'яті − це завжди певний компроміс. З одного боку, кеш-пам'ять повинна бути достатньо мала, щоб її вартісні показники були близькі до величини, характерної для ОП. З іншої – вона повинна бути достатньо великою, щоб середній час доступу в системі, що складається з основної і кеш-пам'яті, визначався часом доступу до кеш-пам'яті. Чим більша ємність кеш-пам'яті, тим більше логічних схем повинні брати участь в її адресації. Як наслідок, ІМС кеш-пам'яті підвищеної ємності працюють повільніше в порівнянні з мікросхемами меншої ємності, навіть якщо вони виконані за однією і тією ж технологією.

Реальна ефективність використання кеш-пам'яті залежить від характеру вирішуваних завдань, і неможливо заздалегідь визначити, яка її ємність буде дійсно оптимальною. Незважаючи на очевидні протиріччя, є видимою і загальна тенденція: у міру збільшення ємності кеш-пам'яті вірогідність промахів спочатку істотно знижується, але досягши певного значення ефект згладжується і стає неістотним [25]. Встановлено, що для більшості завдань близькою до оптимальної є кеш-пам'ять ємністю від 1 до 512 Кбайт.

Ще одним важливим чинником, що впливає на ефективність використання кеш-пам'яті, служить розмір рядка. Коли в кеш-пам'ять поміщається рядок, разом з необхідним словом туди потрапляють і сусідні слова. У міру збільшення розміру рядка вірогідність промахів спочатку падає, оскільки в кеш, згідно принципу локальності, потрапляє все більше даних, які знадобляться найближчим часом. Проте вірогідність промахів починає рости, коли розмір рядка стає надмірно великим.

Пояснюється це тим, що:

- великі розміри рядка зменшують загальну кількість рядків, які можна завантажити в кеш-пам'ять, а мале число рядків приводить до необхідності частої їх зміни;
- у міру збільшення розміру рядка кожне додаткове слово виявляється далі від запитаного, тому таке додаткове слово менш імовірно знадобиться в найближчому майбутньому.

Залежність між розміром рядка і вірогідністю промахів багато в чому визначається характеристиками конкретної програми, через що важко рекомендувати певне значення величини рядка. Практика показує, що найбільш близьким до оптимального можна визнати розмір рядка, рівний 4-8 одиницям, що адресуються (словам або байтам). На практиці розмір рядка зазвичай вибирають рівним ширині шини даних, що пов'язує кеш-пам'ять з основною пам'яттю.

У табл. 1 наведено динамічні параметри кешів різних рівнів у сучасних комп'ютерах фірм Intel та AMD.

Кеш-пам'ять в сучасних комп'ютерах будується за дворівневою схемою, але може бути і трирівневою.

Первинний (внутрішній) кеш (L1 Cache), або кеш першого рівня, вбудований у процесори, починаючи з i486. Ємність такого кеша порівняно з основною оперативною пам'яттю невелика і становить 8-64 Кбайт, швидкодія – 10-12 нс.

Таблиця 1 - Динамічні параметри кешів різних рівнів

Тип процесора	Pentium III	Athlon	Pentium 4	Athlon 64 X2	Core2 Duo/Quad
Швидкодія кеша першого рівня, нс (МГц)	0,71(1400)	0,71(1400)	0,26(3800)	0,33(3000)	0,34(2930)
Об’єм кеша першого рівня, (МБ)	32	128	20	256	64/128
Швидкодія кеша другого рівня, нс (МГц)	0,71(1400)	0,71(1400)	0,26(3800)	0,33(3000)	0,34(2930)
Об’єм кеша другого рівня, (МБ)	512	512	2048	2048	4096/8192
Швидкодія шини пам’яті, нс (МГц)	7,5(133)	3,8(266)	1,25(800)	2,5(400)	0,94(1066)

Він може бути побудований з використанням двох архітектур: принстонської та гарвардської. Принстонська архітектура передбачає використання спільної пам'яті L1 для зберігання даних і команд. За гарвардської архітектури пам'ять L1 розподіляється на дві рівні частини, одна з яких використовується для зберігання команд, інша – для даних. Кеш L1 безпосередньо вбудований в мікросхему ядра процесора.

Вторинний (зовнішній) кеш (L2 Cache), або кеш другого рівня, встановлюється на системній платі, а в процесорах P6 – на картриджі в одній упаковці з ядром і підключається до спеціальної внутрішньої шини процесора. Виготовляється у вигляді окремої мікросхеми. Якщо він встановлюється на системній платі, то працює на її частоті й знаходиться поруч з мікросхемою процесора.

Кеш третього рівня (L3 Cache) застосовується не часто, тому що його реалізація багато коштує (ємність повинна бути більша, ніж у вторинного). Вбудований кеш L3 є у процесорів Xeon MP, його розмір досягає 8 Мбайт.

Спочатку кеші проектувались як асинхронні і не були синхронізовані із системною шиною. Згодом, у 1995 році, було розроблено кеш синхронного типу, який працює синхронно із системною шиною процесора. Це підвищує його швидкодію та ефективність. Крім того, в цей же час функціонування процесорів було доповнено конвеєрним монопольним режимом (Pipeline Burst mode), що дало можливість скоротити період очікування за рахунок зменшення кількості станів очікування після першої передачі даних. Ці режими дали змогу підвищити ефективність комп'ютерів приблизно на 20%.

Ефективність і можливості кеша залежать від його контролера. Більшість контролерів мають обмеження на ємність кешованої пам'яті. Для комплекту мікросхем системної логіки 430TX, який застосовується в комп'ютерах на основі процесора Pentium, вона становить 64 Мбайт. У такому випадку кешуються дані тільки в межах перших 64 Мбайт основної оперативної пам'яті, а всі дані після перших 64 Мбайт ніколи не потрапляють в кеш, і при зверненні до них завжди необхідні всі стани очікування, що визначаються повільнішою логікою DRAM [23].

До значного уповільнення роботи комп'ютера загалом можуть призвести тип програмного забезпечення та адреси, за якими зберігаються дані оперативної пам'яті. Так, наприклад, операційні системи Windows 95/98 і NT завантажуються в оперативну пам'ять; і якщо встановлено оперативну пам'ять ємністю 96 Мбайт, то операційна система і прикладні програми завантажуватимуться безпосередньо у верхні 32 Мбайт, які не кешуються. В цьому випадку можна зменшити ємність оперативної пам'яті до 64 Мбайт, тобто немає необхідності встановлювати ємність оперативної пам'яті більшу, ніж дає змогу її кешувати комплект мікросхем системної логіки.

Контролер кеша повинен забезпечувати когерентність (coherency) – узгодженість даних кеш-пам'яті обох рівнів з даними основної пам'яті при тому, що звернення до цих даних може здійснюватись не тільки процесором, а й іншими активними пристроями (busmaster), під'єднаними до системних шин PCI, VLB, ISA та ін. У складі обчислювальної системи (мережі) може бути кілька процесорів зі своїм внутрішнім кешем, що слід враховувати.

Контролер кеша сучасних процесорів розміщений або в мікросхемі North Bridge комплекту мікросхем системної логіки на основі процесора Pentium, або на платі процесорів Pentium Pro, Pentium II, Pentium III і Pentium IV.

У процесі обчислень ЦП може не тільки зчитувати існуючу інформацію, але і записувати нову, що приводить до оновлення вмісту кеш-пам'яті.

Поведінка кеш-контролера під час здійснення операції запису в пам'ять, коли копія ділянки, що вимагається, знаходиться в певному рядку кеша, визначається його алгоритмом роботи чи стратегією запису (Write Policy).

Існують *дві основні стратегії запису даних* з кеша в основну пам'ять: *наскрізний запис* – WT (*Write Through*) і *зворотний запис* – WB (*Write Back*).

WT – запис передбачає виконання кожної операції запису, яка потрапляє в кешований блок, одночасно в рядок кеша і в основну пам'ять. При цьому процесор вимушений щоразу здійснювати тривалу операцію запису в основну пам'ять. За такого запису достатньо тільки інформації тега. Але така простота запису не дає високої ефективності. Існують варіанти алгоритму WT із застосуванням *відкладеного буферизованого запису*, за якого дані в основну пам'ять переписуються через FIFO – буфер (First-In First-Out – "першим прийшов – першим і вийшов") під час вільних тактів шини. Під час використання буферизації процесор повністю звільняється від роботи з основною пам'яттю.

WB – запис дає змогу зменшити кількість операцій запису на системній шині основної пам'яті. Коли блок основної пам'яті, в який повинен здійснюватися запис, відображений в кеші, то фізичний запис спочатку відбуватиметься в цей дійсний рядок кеша і буде позначений як нечистий (dirty) або модифікований, тобто такий, що вимагає вивантаження в основну пам'ять. Тільки після цього вивантаження рядок стане чистим (clean), і його можна буде використати для кешування інших блоків без втрати цілісності даних. В основну пам'ять дані переписуються тільки цілим рядком.

WB-алгоритм складніший в реалізації, ніж WT-алгоритм, але істотно ефективнішій. Підтримка кешування із зворотним записом потребує додаткових інтерфейсних сигналів, якщо здійснюється звернення з боку інших контролерів системної шини.

2 Основні архітектури кеш-пам'яті

Залежно від способу визначення взаємної відповідальності рядка кеша і ділянки основної пам'яті розрізняють три архітектури кеш-пам'яті: кеш прямого відображення (direct-mapped cache), повністю асоціативний кеш (fully associative cache) та їх комбінація – набірно-асоціативний кеш (set-associative cache), який має дві модифікації – множинно-асоціативне відображення і відображення секторів.

Кеш прямого відображення. Адреса пам'яті, за якою відбувається звернення до кеша прямого відображення, однозначно визначає рядок кеша, де може знаходитись необхідний блок.

Сутність архітектури прямого відображення полягає в тому, що кожен рядок кеша може відображати з будь-якої сторінки кешованої пам'яті тільки відповідний йому рядок. При цьому на кожний рядок кеша може претендувати багато сторінок пам'яті з однаковою молодшою частиною адреси, що є зміщенням всередині сторінки. Один рядок у певний момент містить копію тільки однієї з цих сторінок. Адресу (номер) рядка в кеш-пам'яті називають *індексом* (index).

Інформацію про те, яку саме сторінку основної пам'яті займає даний рядок, тобто старшу частину адреси чи номер сторінки, містить тег. Пам'ять тегів має кількість комірок, яка дорівнює кількості рядків кеша, а її розрядність повинна бути достатньою, щоб вмістити старші біти адреси кешованої пам'яті, які не потрапили на шину адреси кеш-пам'яті. Крім адресної частини тега кожний рядок кеша відображає біти ознак дійсності та модифікування даних. Організація кеш-пам'яті з прямим відображенням показана на рис. 2.

На цьому рисунку показана основна пам'ять з 14-розрядною адресою блоків, тобто адреса блоків може змінюватись від 0 до 16383. Логіка кеш-пам'яті інтерпретує ці 14 біт як 7-розрядний тег і 7-розрядне поле рядка.

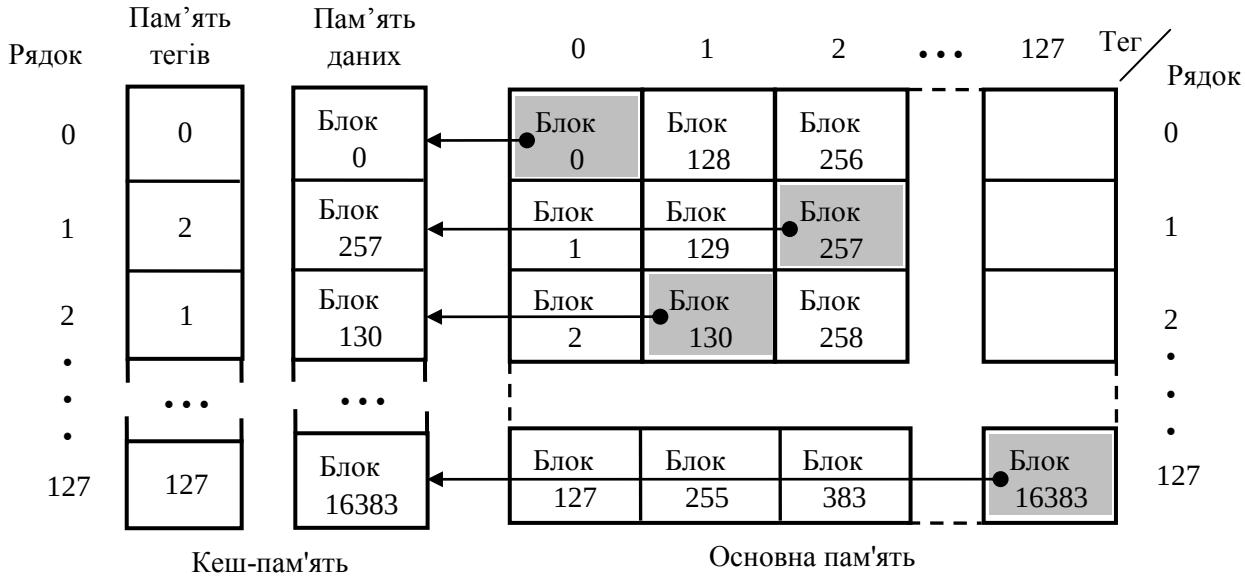

Рисунок 2 - Організація кеш-пам'яті з прямим відображенням

Поле рядка вказує на один із 128 рядків кеш-пам'яті, а саме на той, де може бути відображеним блок з заданою адресою. У свою чергу поле тега визначає, який саме зі списку блоків, що закріплені за даним рядком кеша, буде відображеним. Коли блок фактично заноситься в пам'ять даних кеша, в пам'ять тегів кеш-пам'яті необхідно записати тег цього блока. Тегом служать сім старших розрядів адреси блока.

На початку кожного звернення до кеш-пам'яті контролер спочатку зчитує комірку каталогу із даним індексом, порівнює біти тега зі старшими бітами адреси пам'яті та аналізує ознаку дійсності. Такий аналіз виконується в спеціальному *циклі стеження* (snoop cycle), який ще називають *циклом запиту* (inguire). Якщо в результаті аналізу з'ясується, що потрібний блок не знаходиться в кеші, то генерується або продовжується цикл звернення до основної пам'яті (випадок кеш-промаху). Коли ж є кеш-попадання, то запит обслуговується кеш-пам'яттю. У випадку кеш-промаху після зчитування основної пам'яті нові дані розміщуються в рядку кеша за умови, що він чистий, а в його тегу розташовуються старші біти адреси і встановлюється ознака дійсності даних. З основної пам'яті рядок переписується в кеш повністю, незалежно від обсягу даних, що вимагаються, оскільки ознака дійсності належить до всіх його байтів. Якщо контролер реалізує випереджаюче зчитування (read ahead), то в наступні вільні цикли шини поновиться і наступний рядок, якщо він був чистим. Читання "із запасом" дає змогу за необхідності здійснювати пакетний цикл зчитування з кеша через межу рядка.

Кеш розглянутого типу використовується у вторинному кеші більшості системних плат. Недоліком такої організації кеш-пам'яті є робота "вхолосту" (cache trashing), коли в процесі виконання програми процесора по черзі будуть потрібні блоки (сторінки) пам'яті, зміщені один стосовно іншого на величину, кратну розміру сторінки. Чергове звернення заміщуватиме дані, зчитані у попередньому зверненні, які будуть необхідні в наступному. Таким чином, це буде суцільна низка кеш-промахів. Зменшує кількість кеш-попадань також переключення сторінок у багатозадачних обчислювальних системах. Оскільки різні задачі претендуватимуть на одні й ті самі рядки кеша, то збільшення його розмірів за архітектури прямого відображення не дає суттєвого підвищення ефективності. Підвищити її, не збільшуючи ємність кеша, можна тільки зміною структури кеш-пам'яті.

Повністю асоціативний кеш. У повністю асоціативному кеші, на відміну від попередньої архітектури, будь-який його рядок може відображати будь-який блок пам'яті (рис. 3). Це істотно підвищує ефективність використання його обмеженої ємності.

Такий кеш містить два банки: пам'ять даних і пам'ять тегів. Всі біти адреси кешованого блока, за винятком бітів, які визначають розташування даних у рядку (зміщення), зберігаються в пам'яті тегів. Повна адреса ОП для повністю асоціативної пам'яті ділиться на дві частини. Старші розряди є тегом і визначають номер блока даних в ОП, а молодші – зміщення в рядку кеша.

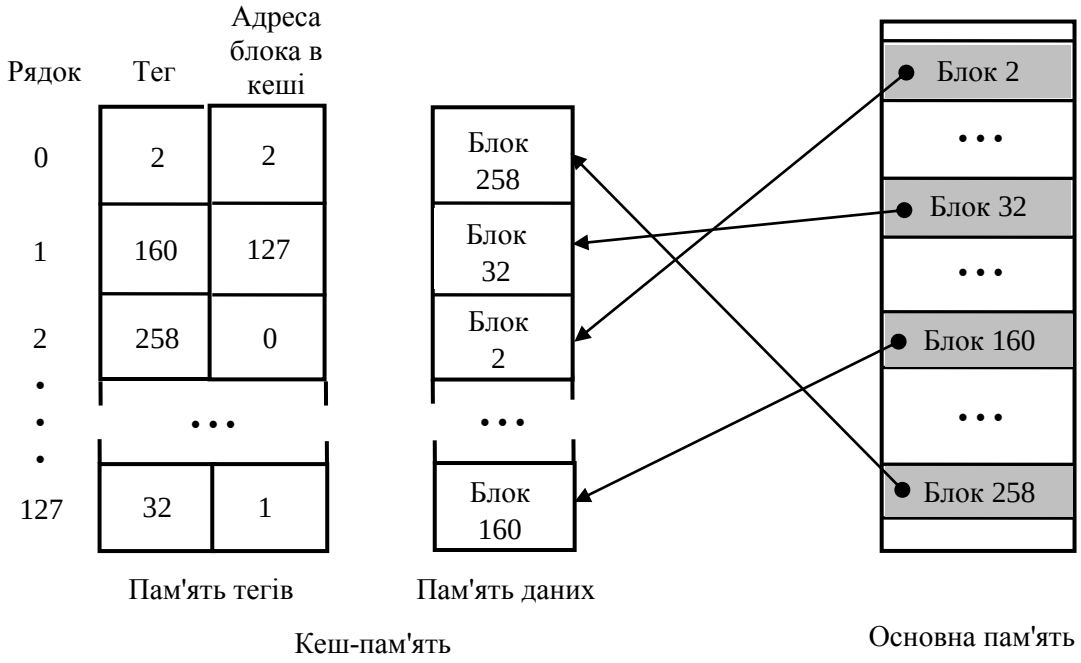

Рисунок 3 - Кеш-пам'ять з асоціативним відображенням

Довжина блока ОП, як і раніше, дорівнює довжині рядка кеш-пам'яті. Як пам'ять тегів у даній архітектурі використовується асоціативна пам'ять. При цьому кількість комірок асоціативної пам'яті тегів дорівнює кількості рядків кеша, тобто кожному рядку кеша віповідає однойменна комірка тега.

Під час запису блока даних з ОП в рядок кеша одночасно в тег записується старша частина адреси (номер рядка). При цьому для визначення наявності даних кеш-пам'яті, які вимагаються, необхідне порівняння тегів усіх рядків зі старшою частиною адреси, а не одного або кількох, як у разі прямого відображення або набірно-асоціативній архітектурі. Таким чином, відпадає необхідність послідовного перебирання комірок пам'яті тегів. Виконується тільки паралельний аналіз усіх комірок.

Читання слова з кеш-пам'яті починається з асоціативного пошуку тега адресованого слова в асоціативній пам'яті тегів по коду, який є старшою частиною адреси ОП. Якщо під час асоціативного пошуку знаходиться тег адресованого слова, то воно зчитується з рядка кеша, номер якого визначається номером комірки тега, що збігся, а положення в рядку – зміщенням. Якщо результат асоціативного пошуку негативний, то адресованого слова немає в кеш-пам'яті і його необхідно зчитувати з ОП. При цьому разом з передачею адресованого слова в процесор в кеш-пам'ять переписується весь блок, в якому знаходиться це слово.

Недоліком цієї архітектури кеш-пам'яті є велика апаратурна складність пошуку інформації, зв'язана з необхідністю виконувати порівняння усіх тегів паралельно по всіх комірках асоціативної пам'яті. З підвищенням ємності кеш-пам'яті, відповідно і асоціативної пам'яті тегів, зростає її складність і вартість. Тому така архітектура використовується тільки для побудови кеш-пам'яті невеликої ємності (первинного кеша в деяких процесорах).

Набірно-асоціативний кеш. Набірно-асоціативна архітектура кеша дає можливість кожній сторінці основної кешованої пам'яті претендувати на один з кількох рядків кеша, об'єднаних в набір (set). В кеші такої архітектури є кілька паралельно і погоджено працюючих каналів прямого відображення, де контролер кеша приймає рішення про те, в який з рядків набору помістити черговий блок даних. Розглянемо організацію кеш-пам'яті з множинно-асоціативним відображенням.

Множинно-асоціативне відображення відноситься до групи методів частково-асоціативного відображення. Воно є одним з можливих компромісів, що поєднує переваги прямого і асоціативного способів відображення і, певною мірою, вільним від їх недоліків. Кеш-пам'ять (як тегів, так і даних) розбивається на ***v*** підмножин (надалі називатимемо такі підмножини модулями), кожна з яких містить ***k*** рядків (прийнято говорити, що модуль має ***k*** входів). Залежність між модулем і блоками ОП така ж, як і при прямому відображенні: на рядки, що входять в модуль ***i***, можуть бути відображені тільки цілком певні блоки основної пам'яті, відповідно до співвідношення ***i = j mod v***, де ***j*** - адреса блока ОП. У той же час розміщення блоків по рядках модуля – довільне, і для пошуку потрібного рядка в межах модуля використовується асоціативний принцип.

На рис. 4 показаний приклад чотиривходової кеш-пам'ятіз множинно-асоціативним відображенням. Пам'ять даних кеш-пам'яті розбита на 32 модулі по 4 рядки в кожному. Пам'ять тегів містить 32 комірки, в кожній з яких може зберігатися 4 значення тегів (поодинці на кожен рядок модуля). 14-розрядна адреса блока ОП подається у вигляді двох полів: 9-розрядного поля тега і 5-розрядного поля номера модуля.

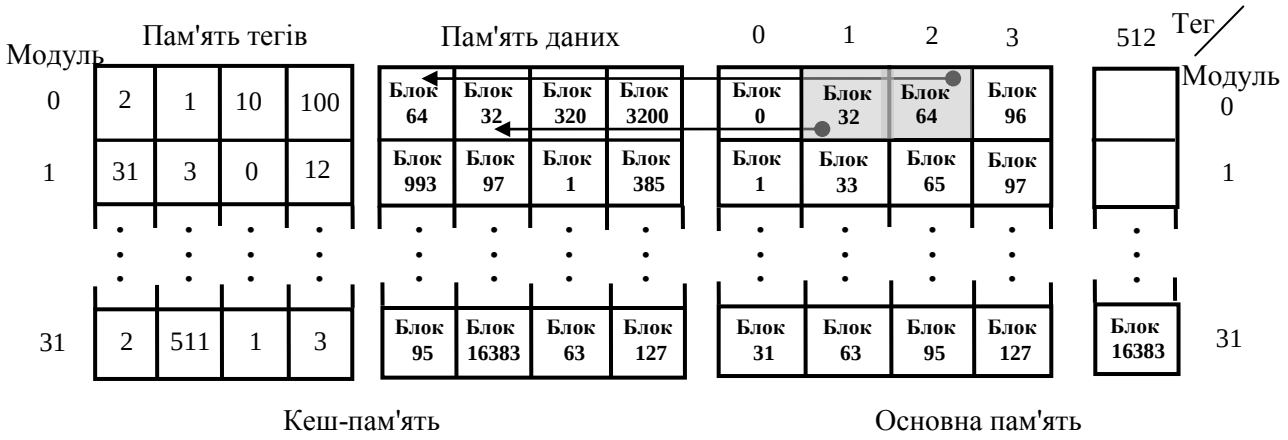

Рис. 4 Кеш-пам'ять з множинно-асоціативним відображенням

Номер модуля однозначно указує на один з модулів кеш-пам'яті. Він також дозволяє визначити номери тих блоків ОП, які можна відображати на цей модуль. Такими є блоки ОП, номери яких під час ділення на 25 дають у залишку число, що збігається з номером даного модуля кеш-пам'яті.

Так, блоки 0, 32, 64, 96 і так далі в основній пам'яті відображаються на модуль з номером 0; блоки 1, 33, 65, 97 і так далі відображаються на модуль 1 і т. д. Будь-який з блоків у послідовності може бути завантажений у будь-який з чотирьох рядків відповідного модуля.

У випадку такої постановки роль тега виконують 9 старших розрядів адреси блока ОП, в яких міститься порядковий номер блока в послідовності блоків, що відображаються на один і той же модуль кеш-пам'яті. Наприклад, блок 65 у послідовності блоків, що відображаються на модуль 1, має порядковий номер 2 (відлік ведеться від 0).

Під час звернення до кеш-пам'яті 5-розрядний номер модуля указує на конкретну комірку пам'яті тегів (це відповідає прямому відображенню). Далі проводиться паралельне порівняння кожного з чотирьох тегів, що зберігаються в цій комірці, з полем тега адреси, що поступила, тобто пошук потрібного тега серед чотирьох можливих здійснюється асоціативно.

У граничних випадках, коли ***v=m, k=1,*** множинно-асоціативне відображення зводиться до прямого, а при ***v=1, к=m*** − до асоціативного.

Найбільш загальний вид організації множинно-асоціативного відображення – використання двох рядків на модуль **(*v=m/2, k=2*).** Чотиривходова множинно-асоціативна кеш-пам'ять **(*v=m/4, k=4*)** дає додаткове поліпшення за порівняно невелику додаткову ціну [25]. Подальше збільшення числа рядків у модулі істотного ефекту не привносить.

Слід зазначити, що саме цей спосіб відображення найширше розповсюджений у сучасних мікропроцесорах.

3 Структура засобів кешування пам'яті

Засоби кешування пам'яті містять два рівні кеш-інструкцій і даних (L1 Cache і L2 Cache), буфери асоціативної трансляції TLB блока сторінкової переадресації і буфери запису. Вони можуть бути подані в різних варіаціях, зокрема розміщені на кристалі або картриджі процесора, або на системній платі, починаючи з процесора i80486. Процесор i80386 містить тільки буфери TLB. Кеш-пам'ять, що встановлювалась на системній платі, не підтримувалась процесором.

Загальну структуру засобів кешування пам'яті для 32-розрядних процесорів фірми Intel, у тому числі для процесорів P6, наведено на рис. 5.

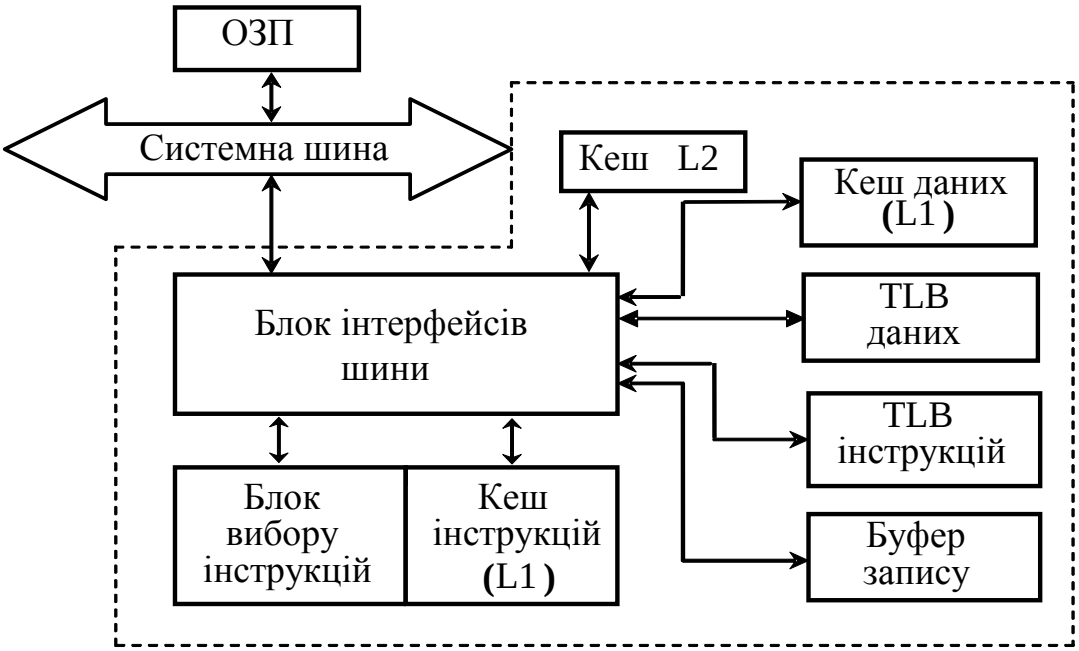

Рисунок 5 - Загальна структура засобів кешування пам'яті

Первинний кеш інструкцій тісно пов'язаний з блоком попередньої вибірки, а первинний кеш даних - з виконавчим блоком процесора. Вторинний кеш є спільним і в процесорах P6 підключений до окремої внутрішньої шини кеш-пам'яті. В процесорах i486 і Pentium вторинний кеш є зовнішнім і підключається до зовнішньої системної шини процесора. В процесорах Celeron 266 і 300 вторинний кеш відсутній.

Довжина рядка кеша в i486 – 16 байт, в процесорах P5 другого покоління і P6 – 32 байт. Рядки заповнюються цілком пакетними циклами зчитування – 4 передачі на рядок з основної пам'яті, вирівняними за 32-байтними межами.

Будь-який внутрішній запит процесора на звернення до пам'яті спрямовується у внутрішній кеш.

Якщо запитувана ділянка пам'яті наявна у рядку внутрішнього кеша, то він обслуговує цей запит. У випадку промаху запит задовольняється, як тільки необхідні дані зчитуються з ОЗП. Заповнення рядка до кінця відбувається паралельно з обробкою одержаних даних. Виділення і заміщення у процесорах i486 і P5 виконуються тільки для кеш-промахів під час зчитування. У випадку промахів запису заповнення рядків здійснюється тільки в процесорах P6. Кешування доступне в будь-якому режимі процесора.

Буфер асоціативної трансляції TLB зберігає входження в каталог і в таблиці сторінок, до яких звертались останнім часом. В i486 для даних та інструкцій використовується єдиний TLB, у процесорах P5 і P6 ці буфери роздільні.

Великі сторінки, зокрема 2 Мбайт у режимі PAE (Physical Address Extension – режим розширення фізичної адреси до 36 біт) і 4 Мбайт в PSE (Page Size Extension – прапорець розширення розміру сторінки) обслуговуються роздільними TLB.

Буфери запису пов'язані з виконавчим блоком процесора. Вони дають змогу на деякий час відкласти фактичний запис у зовнішній кеш і основну пам'ять, пропонуючи шину для інших обмінів, необхідних для виконання наступних інструкцій. Запис буферизується в усіх режимах роботи процесора, але буферизація запису в порти вводу/ виводу не відбувається.

Кеш-пам'ять побудована з урахуванням можливості звернень до неї зовнішніх об'єктів, зокрема інших процесорів та контролерів. Процесори мають механізми зовнішнього стеження за станом свого кеша. Для підтримки узгодження даних кеша та основної пам'яті процесор відпрацьовує цикли стеження (Snoop Cycle чи Inquire Cycle), які ініціюються зовнішньою для нього системою. В цих циклах, які відбуваються у разі звернення до пам'яті з боку зовнішнього абонента, процесор визначає наявність даних, що вимагаються, в своєму кеші. Якщо вони відображаються в кеші, то дії процесора залежать від стану відповідного рядка кеша і типу зовнішнього звернення. Звернення під час запису призведе до анулювання даного рядка. Звернення під час зчитування до ділянки, яка відповідає модифікованому (“брудному”) рядку, призведе до вивантаження його вмісту в основну пам'ять перед тим як зовнішній абонент виконає реальне зчитування. В процесорах P6 звернення до “брудного” рядка з боку інших процесорів може спричинити вивантаження його вмісту безпосередньо в процесор, що звертався. Це відповідно збереже час, а вивантаження цього рядка в основну пам'ять відбудеться пізніше, згідно з алгоритмом оберненого запису.

Починаючи з процесорів Pentium, їх кеш підтримує протокол MESI (Modified-Exclusive-Shared-Invalid – протокол підтримки когерентності пам'яті за наявністю кеша, названий за визначеним станом рядків: Модифікована - Виняткова – Роздільна – Недійсна). Первинний кеш інструкцій реалізує лише частину протоколу – SI, оскільки не допускає запису.

У просторі основної пам'яті комп'ютера є ділянки, для яких кешування принципово недопустиме, зокрема розподільна пам'ять адаптерів. Для таких ділянок непридатний алгоритм оберненого зв'язку. Крім того, кешування інколи відключають у разі виконання однократно виконуваних ділянок програми з тим, щоб з кеша не витісняти корисніші фрагменти програми.