ТЕМА

ЕЛЕМЕНТИ ПАРАЛЕЛЬНИХ КС.

ПЛАН:

1 Процесори для побудови систем високої продуктивності.
2 Пам’ять систем високої продуктивності.
Висновки

1 ПРОЦЕСОРИ ДЛЯ ПОБУДОВИ СИСТЕМ ВИСОКОЇ ПРОДУКТИВНОСТІ.

Процесор є основним елементом обчислювальної системи, що визначає рівень його швидкодії.

Підвищення продуктивності обчислювальної техніки пов'язано в великій мірі з розробкою нових технологій і залученням найоригінальніших і несподіваних рішень побудови процесорів.

Основні архітектури процесорів:

- Традиційна архітектура фон Неймана.
- Асоціативні процесори.
- Конвеєрні процесори.
- Суперскалярні процесори.
- Процесори з RISC і CISC архітектурою.
- Процесори зі наддовгим командним словом.
- Векторна обробка даних.
- Багатопотокова архітектура і комунікаційні процесори.

1. **Процесори для побудови систем високої продуктивності**

Традиційна архітектура фон Неймана:

- програма зберігається в комп'ютері;
- програма під час її виконання і необхідні для її роботи дані ***знаходяться в оперативній пам'яті***;
- **є арифметико-логічний пристрій**, що виконує арифметичні і логічні операції з даними;
- **є пристрій керування** (ПК), яке інтерпретує команди, які обираються з пам'яті, і виконує їх;
- **пристрої введення і виведення** (ВВ) працюють під управлінням ПК, використовуються для введення програм і даних, а також для виведення результатів розрахунків.

1. **Процесори для побудови систем високої продуктивності**

Особливості архітектури фон Неймана:

- центральний процесор і оперативна пам'ять утворюють **ядро системи**;
- вторинна («зовнішня») пам'ять і пристрої введення-виведення, що утворюють «периферію»;
- комунікації між компонентами системи, що здійснюються за допомогою шин.

У традиційному послідовному комп'ютері, заснованому на ідеях фон Неймана, швидкодія центрального процесора може бути збільшено за рахунок збільшення тактової частоти, величина якої залежить від щільності елементів в інтегральній схемі і швидкодії мікросхем оперативної пам'яті.

1. **Процесори для побудови систем високої продуктивності**

Для підвищення продуктивності необхідні:

- конвеєрна обробка даних і команд;
- використання процесорів зі скороченим набором команд (RISC-процесорів). В RISC-процесорах велика частина команд виконується за 1-2 такти;
- використання суперскалярних процесорів;
- векторну обробку даних;
- використання процесорів з наддовгим командним словом;
- використання багатопроцесорних конфігурацій.

Асоціативні процесори: Існуючі в даний час алгоритми прикладних задач, системне програмне забезпечення і апаратні засоби переважно орієнтована на традиційну адресну обробку даних.

Дані повинні бути представлені у вигляді обмеженої кількості форматів (наприклад, масиви, списки, записи)

Недоліки традиційної обробки:

Такий підхід зумовлює громіздкість операційних систем і систем програмування, а також служить перешкодою до створення обчислювальних засобів з архітектурою, орієнтованою на більш ефективне використання паралелізму обробки даних.

Особливості асоціативної обробки:

Асоціативний спосіб обробки даних дозволяє подолати багато обмежень, властиві адресному доступу до пам'яті, за рахунок завдання деякого критерію відбору і проведення необхідних перетворень, тільки над тими даними, які задовольняють цьому критерію.

Критерієм відбору може бути збіг з будь-яким елементом даних, достатній для виділення шуканих даних з усіх даних.

1. **Процесори для побудови систем високої продуктивності**

Способи асоціативної обробки

Пошук даних може відбуватися за фрагментом, що має більшу або меншу кореляцію з заданим елементом даних.

Якщо реалізується тільки асоціативна вибірка даних з подальшим послідовним використанням знайдених даних, то говорять про асоціативну пам'ять або пам'ять, що адресується за вмістом.

1. **Процесори для побудови систем високої продуктивності**

Особливості асоціативних систем

Асоціативні системи відносяться до класу: ***один потік команд*** - безліч потоків даних (SIMD).

Ці **системи включають велике число операційних пристроїв, здатних одночасно по командам керуючого пристрою вести обробку декількох потоків даних**. В асоціативних обчислювальних системах інформація на обробку надходить від асоціативних запам'ятовуючих пристроїв (АЗП), що характеризуються тим, що інформація в них вибирається не по певній адресі, а за її змістом.

1. **Процесори для побудови систем високої продуктивності**

Конвеєрні процесори

Ідея конвеєра полягає в тому, щоб складну операцію розбити на кілька простіших, таких, які можуть виконуватися одночасно. Операція підсумовування, наприклад, включає віднімання порядків, вирівнювання порядків, складання мантисс і нормалізацію. Кожна з підопераций може виконуватися на окремому блоці апаратури.

Принципи конвеєрної обробки

При русі об'єктів по конвеєру одночасно на різних його ділянках (сегментах) виконуються різні підопераціі, що дає збільшення продуктивності за рахунок використання паралелізму на рівні команд.

Конвеєри застосовуються як при обробці команд (конвеєри команд), так і в арифметичних операціях (конвеєри даних).

1. **Процесори для побудови систем високої продуктивності**

Для ефективної реалізації конвеєра необхідно:

- система виконує повторювану операцію;
- операція може бути розділена на незалежні частини;
- трудомісткість підопераций приблизно однакова.

Кількість сегментів називають ***глибиною конвеєра***. Важливою умовою нормальної роботи конвеєра є *відсутність конфліктів*, підвідних до простоїв конвеєра.

Простої конвеєра

Об'єкти, що надходять в конвеєр, повинні бути незалежними. Якщо, наприклад, операндом є результат попередньої операції, виникають періоди роботи конвеєра ("конвеєрні бульбашки"), коли він порожній.

"Бульбашка" проходить по конвеєру, займаючи місце, але не виконуючи при цьому ніякої корисної роботи.

Цикл виконання команди складається з декількох кроків:

1. Вибірка команди.
2. Декодування команди, обчислення адреси операнда і його вибірка.
3. Виконання команди.
4. Звернення до пам'яті.
5. Запис результату в пам'ять.

Для виконання кожного етапу команди відводиться один такт і в кожному новому такті починається виконання нової команди.

Для зберігання проміжних результатів кожного етапу використовується швидка пам'ять (регістри).

У результаті, в кожному такті будуть виконуватися кілька команд і, незважаючи на те, що час виконання окремої команди дещо збільшиться, виробництво системи в цілому зросте.

1. **Процесори для побудови систем високої продуктивності**

Простий конвеєр команд

Простий конвеєр команд викликається ситуацією, коли чергову команду з потоку команд не можна завантажити в конвеєр відразу, а з якоїсь причини доводиться очікувати кілька тактів.

Якщо чергова команда у відповідному їй такті не може бути виконана, кажуть про конфлікт. В цьому випадку команда очікує своєї черги, а пропускна здатність конвеєра падає.

Очікувати своєї черги доводиться і всім наступним командам.

Команди, вже завантажені в конвеєр, продовжують виконуватися.

Прийнято виділяти три типи конфліктів:

- структурні конфлікти;
- конфлікти за даними;
- конфлікти з управління;
- причиною структурних конфліктів є одночасний запит на використання одного ресурсу декількома командами;
- недостатня кількість регістрів для запису даних в одному такті.

Конфлікти за даними є наслідком логічної залежності команд між собою і відбуваються, якщо для виконання чергової команди потрібний результат виконання попередньої команди (її результат виступає в якості операнда).

У цьому випадку команда не буде виконуватися до тих пір, поки попередня команда не завершить своє виконання і не передасть їй свій результат.

Конфлікти з управління викликаються наявністю в програмах умовних конструкцій. Виконувана гілка умовного оператора визначається тільки після обчислення умови розгалуження. Якщо врахувати, що в програмах до третини операторів є гылкування, втрати продуктивності внаслідок простоїв з управління можуть бути великими.

Оптимізація роботи конвеєра: У більшості сучасних процесорів використовуються спеціальні пристрої, які виконують вибірку команд і поміщають їх в спеціальні черги ще до того, як вони можуть знадобитися. Функціональні вузли конвеєра вміють розпізнавати команди безумовного переходу, визначають адреси наступних команд і вибирають ці команди з черги, що дозволяє забезпечити безперервний потік команд в конвеєрі. Умовні переходи важче піддаються оптимізації, але і тут є способи підвищення продуктивності конвеєра при обробці умовних переходів.

1. **Процесори для побудови систем високої продуктивності**

У багатьох обчислювальних системах, поряд з конвеєром команд, **використовуються** і **конвеєри даних**. Поєднання цих двох конвеєрів дає можливість досягти дуже високої продуктивності на певних класах завдань, особливо якщо використовується кілька різних конвеєрних процесорів, здатних працювати одночасно і незалежно один від одного.

В даний час створені однокристальні векторно-конвеєрні процесори, такі як SX-6.

Процесор SX-6

Основними компонентами мікропроцесора є скалярний процесор і 8 ідентичних векторних пристроїв, сумарна продуктивність яких становить 64 GFLOPS.

(GFLOPS / с - "гігафлоп в секунду" одиниця виміру швидкодії процесора, складова один мільярд операцій з плаваючою крапкою в секунду).

Прикладом комп'ютера з суперскалярним процесором є IBM RISC/6000. При тактовій частоті 62,5 МГц швидкодія системи на обчислювальних тестах досягала 104 MFLOPS/c.

Суперскалярний процесор не вимагає спеціальних векторизуючих компіляторів, хоча компілятор повинен в цьому випадку враховувати особливості архітектури.

Підвищення продуктивності за рахунок збільшення числа функціональних пристроїв і використання низькорівневого паралелізму можна ефективно реалізувати, лише забезпечивши їх збалансовану завантаження. Тут можливі два підходи:

Перший - динамічне завантаження, засноване на аналізі програмного коду під час його виконання.

Другий підхід - статичний, підтримуваний компілятором.

Процесори з RISC архітектурою CISC

В основі RISC-архітектури (RISC - Reduced Instruction Set Computer) процесора лежить ідея збільшення швидкості його роботи за рахунок спрощення набору команд.

Протилежну тенденцію представляють CISC-архітектури, процесори зі складним набором команд (CISC - Complete Instruction Set Computer). Основоположником архітектури CISC є компанія IBM, а в даний час лідером у цій області є Intel (процесори Pentium).

1. **Процесори для побудови систем високої продуктивності**

Особливості RISC-архітектури

В рамках CISC-підходу набір команд включає команди, близькі до операторів мови високого рівня. **В рамках RISC-підходу** набір команд спрощується і оптимізується під реальні потреби користувацьких програм.

В основу RISC-архітектури покладені наступні принципи і ідеї:набір команд повинен бути обмеженим і включати тільки прості команди, час виконання яких після вибірки і декодування один такт або трохи більше; використовується конвеєрна обробка.

Ефективність RISC-архітектури:

Конструкція пристрою управління у разі RISC-архітектури спрощується, і це дає можливість процесору працювати на великих тактових частотах. Використання простих команд дозволяє ефективно реалізувати і конвеєрну обробку даних, і виконання команд.

1. **Процесори для побудови систем високої продуктивності**

Ефективність RISC-архітектури:

RISC-процесор має час виконання команд майже удвічі менше.

Складні команди RISC-процесором виконуються довше, але їх кількість відносно невелика.

В RISC-процесорах невелике число команд адресується до пам'яті. Вибірка даних з оперативної пам'яті вимагає більш одного такту. Велика частина команд працює з операндами, що знаходяться в регістрах.

Усі команди мають уніфікований формат і фіксовану довжину. Це спрощує і прискорює завантаження і декодування команд, оскільки, наприклад, код операції і поле адреси завжди знаходяться в одній і тій же позиції.

1. **Процесори для побудови систем високої продуктивності**

Ефективність RISC-архітектури:

Завдяки спрощеній архітектурі RISC-процесора, на мікросхемі з'являється місце для розміщення додаткового набору регістрів. Розподіл реєстрової пам'яті під змінні виконується компілятором.

Не дивлячись на згадані переваги RISC-архітектури, немає простої і однозначної відповіді на питання про те, що краще - RISC або CISC.

Прикладом CISC-процесора є процесор Pentium (кількість команд більше 200, довжина команди 1-11 розрядів, є 8 регістрів загального призначення), а в якості прикладу RISC-процесора можна привести SunSPARC (кількість команд близько 50, довжина команди 4 розряди, є 520 регістрів загального призначення).

1. **Процесори для побудови систем високої продуктивності**

Альтернативою суперскалярним процесорам є **процесори з наддовгим командним словом** (VLIW - Very Large Instruction Word). Робота VLIW-процесора заснована на виявленні паралелізму команд під час трансляції. **Транслятор,** аналізує програму, визначаючи, які операції можуть виконуватися паралельно.

Такі операції "упаковуються" в одну велику команду

Особливості VLIW-процесорів

У трансляторах для VLIW-процесорів застосовуються такі спеціальні прийоми, як розгортка циклів, пророкування розгалужень і планування видачі команд.

Метод трасування планування полягає в тому, що з послідовності звичайних команд вихідної програми довгі команди генеруються шляхом перегляду (трасування) лінійних (без розгалужень) ділянок програми.

1. **Процесори для побудови систем високої продуктивності**

Особливості архітектури VLIW-процесорів

На відміну від традиційної архітектури, регістри не резервуються для певних цілей, їх використання визначається під час компіляції. Висока швидкість роботи оперативної пам'яті досягається її розшаруванням. Прикладами процесорів з VLIW-архітектурою є Itanium, а також деякі процесори, орієнтовані на мультимедіа-додатки.

Векторний процесор за один цикл виконання команди зробить поелементне складання масивів і присвоїть отримані значення відповідних елементів масиву. Кожен операнд при цьому зберігається в особливому, векторному регістрі. Звичайному, послідовному процесору довелося б багаторазово виконати операцію додавання елементів двох масивів.

Векторний процесор виконує лише одну команду.

1. **Процесори для побудови систем високої продуктивності**

Векторна обробка даних

Векторний процесор за один цикл виконання команди зробить поелементне складання масивів і присвоїть отримані значення відповідних елементів масиву. Кожен операнд при цьому зберігається в особливому, векторному регістрі. Звичайному, послідовному процесору довелося б багаторазово виконати операцію додавання елементів двох масивів.

Векторний процесор виконує лише одну команду.

Особливості архітектури: Векторні операції можуть бути реалізовані з використанням паралелізму, який забезпечують конвеєрізовані функціональні вузли. Такі архітектури і називаються векторними процесорами. Можна вважати, що векторний процесор побудований на основі SISD-архітектури, в якій реалізовані векторні операції. В результаті цього отримуємо SIMD-архітектуру

1. **Процесори для побудови систем високої продуктивності**

Векторний модуль

Векторний модуль містить декілька векторних регістрів загального призначення, регістр довжини вектора (для зберігання довжини оброблюваного вектора), регістр маски.

Регістр маски містить послідовність бітів - виконавчі розряди вектора, яким відповідають нульові біти маски, в певних ситуаціях ігноруються при виконанні векторних операцій.

Мультимедійні програми зазвичай працюють з великими масивами даних, що складаються з коротких (8- або 16-розрядних) значень з фіксованою точкою.

Такі додатки представляють величезний потенціал векторного (SIMD) паралелізму, тому нові покоління мікропроцесорів загального призначення забезпечуються мультимедійними командами.

1. **Процесори для побудови систем високої продуктивності**

Мультимедійні розширення

Мультимедійні розширення включені в системи команд процесорів Intel (MMX-розширення системи команд Pentium і нові SIMD-команди Pentium), AMD (3D Now!), Sun (VIS SPARC), Compaq (Alpha MVI), Hewlett Packard (PA-RISC MAX2), SGI (MDMX), Motorola (PowerPC AltiVec).

У Pentium введена паралельна обробка в режимі SIMD-процесора чотирьох 32-розрядних операндів в форматі з плаваючою точкою.

Багатопотокова архітектура

У даний час з'явилися і розвиваються процесори з багатопотоковою архітектурою (MTA - MultiThread Architecture).

Ідея полягає в тому, що програма виконується в декількох потоків і кожному потоку віддається спеціальний регістровий файл.

Завдання створення і управління безліччю потоків вирішується транслятором.

Приклад комп'ютера з багатопотоковою архітектурою - Cray MTA-2.

Різновидом даного підходу є одночасна багатопоточність (SMT - Simultaneous MultiThreading), коли один фізичний процесор бачиться операційній системі як два «логічних» процесора.

1. **Процесори для побудови систем високої продуктивності**

Комунікаційні процесори

Комунікаційні процесори - це мікрочіпи, що являють собою щось середнє між жорсткими спеціалізованими інтегральними мікросхемами і гнучкими процесорами загального призначення.

Комунікаційні процесори програмуються, як і звичні нам ПК-процесори, але побудовані з урахуванням мережевих завдань, оптимізовані для роботи в мережі, і на їх основі виробники - як процесорів, так і обладнання - пишуть програмне забезпечення для специфічних додатків.

1. **Процесори для побудови систем високої продуктивності**

Особливості побудови

Комунікаційний процесор має власну пам'ять і оснащений високошвидкісними зовнішніми каналами для з'єднання з іншими процесорними вузлами. Його присутність дозволяє в значній мірі звільнити обчислювальний процесор від навантаження, пов'язаного з передачею повідомлень між процесорними вузлами.

Швидкісний комунікаційний процесор з RISC-ядром дозволяє управляти обміном даних з кількох незалежних каналах, підтримувати практично всі поширені протоколи обміну, гнучко і ефективно розподіляти і обробляти послідовні потоки даних з тимчасовим поділом каналів.

2 ПАМ’ЯТЬ СИСТЕМ ВИСОКОЇ ПРОДУКТИВНОСТІ.

Пам'ять комп'ютерних систем

У сучасних КС використовуються такі різновиди пам'яті.

- Оперативна пам'ять
- Колективна пам'ять
- Почергова пам'ять

Прийнята така *класифікація паралельних комп'ютерів по архітектурі підсистем оперативної пам'яті*:

системи з розділеною пам'яттю, у яких є одна велика віртуальна пам'ять, і все процесори мають однаковий доступ до даних і командам, що зберігаються в цій пам'яті;

системи з розподіленою пам'яттю, у яких кожен процесор має свою локальну оперативну пам'ять, і до цієї пам'яті у інших процесорів немає доступу.

2. Пам’ять систем високої продуктивності

Проблема когерентності

У багатопроцесорних обчислювальних системах виникає проблема кеш-когерентності.

Вона полягає в тому, що ***якщо двом або більше процесорам знадобилося значення однієї і тієї ж змінної, воно буде зберігатися у вигляді декількох копій в кеш-пам'яті всіх процесорів.***

Один з процесорів може змінити це значення в результаті виконання своєї команди і воно буде передано в оперативну пам'ять комп'ютера. Але в кеш-пам'яті інших процесорів все ще зберігається старе значення.

Отже, необхідно забезпечити своєчасне оновлення даних в кеш-пам'яті всіх процесорів комп'ютера.

Рішення проблеми когерентності

Проблема кеш-когерентності може вирішуватися **програмним шляхом - на рівні транслятора і операційної системи.** Транслятор, наприклад, може визначати моменти безпечної синхронізації кеш-пам'яті при виконанні програми.

Інший підхід заснований на **апаратному рішенні проблеми кеш-когерентності**. У цьому випадку застосовуються спеціальні протоколи, кеш-пам'ять використовується більш ефективно, все відбувається "прозоро" для програміста.

Протоколи управління кеш-пам'яттю

Протоколи каталогів реалізують збір та облік інформації про копії даних в кеш-пам'яті. Ця інформація зберігається в спеціальній області оперативної пам'яті - ***каталозі***.

Запити перевіряються за допомогою каталогу, потім виконуються необхідні пересилання даних. Використання даного підходу ефективно в великих системах зі складними схемами комунікації.

Протоколи спостереження розподіляють "відповідальність" за когерентність кешей між контролерами кеш-пам'яті. Контролер визначає зміну змісту кеш-пам'яті і передає інформацію про це іншим модулям кеш-пам'яті.

ВИСНОВОК

1.Процессор є основним елементом обчислювальної системи, що визначає рівень його швидкодії.
2.В традиційному послідовному комп'ютері, швидкодія центрального процесора може бути збільшено за рахунок збільшення тактової частоти, і швидкодії мікросхем оперативної пам'яті
3.Другі методи підвищення швидкодії засновані на розширеннях традиційної фон-нейманівскої архітектури, що включають: 1) конвеєрну обробку даних і команд; 2) використання процесорів з скороченим набором команд (RISC-процесорів), де велика частина команд виконується за 1-2 такти; 3) використання суперскалярних процесорів; 4) векторну обробку даних; 5) використання процесорів з наддовгим командним словом; 6) - використання багатопроцесорних конфігурацій.

ВИСНОВОК

4.Характеристика оперативної пам'яті і особливості її пристрою є найважливішим фактором, від якого залежить швидкодія комп'ютерної системи.
5.Принята наступна класифікація паралельних комп'ютерів по архітектурі підсистем оперативної пам'яті:
- системи з розділеною пам'яттю, у яких є одна велика віртуальна пам'ять, а всі процесори мають однаковий доступ до даних і команд, що зберігаються в цій пам'яті;
- системи з розподіленою пам'яттю, у яких кожен процесор має свою локальну оперативну пам'ять, і до цієї пам'яті у інших процесорів немає доступу.