Лекція 1. Головні складові архітектури високопродуктивних обчислювальних систем.

План лекції:

1. Історія розвитку високопродуктивних обчислювальних систем.
2. Класифікація паралельних архітектур по Флінну
3. Основні елементи архітектури високопродуктивних обчислювальних систем.
   3.1. Конвеєри
   3.2. RISC-процесори
   3.3. Суперскалярні процесори
   3.4. Суперскалярні процесори зі наддовгим командним словом
4. Архітектура багатоядерних процесорів
5. Різниця між многоядерною архітектурою і технологією гіперпоточності
6. Векторно-конвеєрні комп'ютери
7. Симметричні мультипроцесорні системи SMP
8. Системи з масовим паралелізмом (MPP)

Архітектура комп'ютера це опис основних комп'ютерних комплектуючих та їх взаємозв'язку.

Традиційна фон-нейманівська архітектура базується на таких принципах:

- програма зберігається на комп'ютері;
- програма під час виконання та необхідні для її роботи дані розташовані в оперативній пам'яті;
- існує арифметико-логічний пристрій, який виконує арифметичні та логічні операції над даними;
- існує пристрій управління, який інтерпретує команди, вибрані з пам'яті, та виконує їх;
- пристрої вводу та виводу (ВВ) використовуються для введення програм і даних і для відображення результатів обчислення. Працює під управлінням пристрою управління.

Інші методи підвищення продуктивності засновані на розширеннях традиційної архітектури фон Неймана, у тому числі:

- конвеєрній обробці команд і даних;
- використанні процесорів зі скороченим набором команд (RISC). Процесорами RISC більшість команд виконується за 1-2 такти;
- використанні суперскалярних процесорів;
- векторній обробці даних;
- використанні процесорів з наддовгим командним словом;
- використанні багатопроцесорних конфігурацій.

Найбільш перспективним класом високопродуктивних систем є багатопроцесорні системи. В організації багатопроцесорних обчислювальних систем можна підкреслити наступні ключові моменти:

- кількість окремих процесорів та їх архітектура;
- структура та організація доступу до оперативної пам'яті;
- топологія комунікаційної мережі і її швидкодія;

- робота з пристроями вводу-виводу.

Найбільш важливою характеристикою багатопроцесорної обчислювальної системи є її масштабованість. Масштабованість — це міра, яка вказує на те, чи можна вирішити дану проблему швидше, якщо збільшити кількість процесорних елементів. Цю властивість має як апаратне, так і та програмне забезпечення.

1. Історія розвитку високопродуктивних обчислювальних систем

У процесі розвитку суперкомп'ютерних технологій ідею підвищення продуктивності обчислювальної системи за рахунок збільшення числа процесорів використовували неодноразово. Якщо не вдаватися занадто глибоко в історичний екскурс і обговорення всіх таких спроб, то можна таким чином коротенько описати розвиток подій.

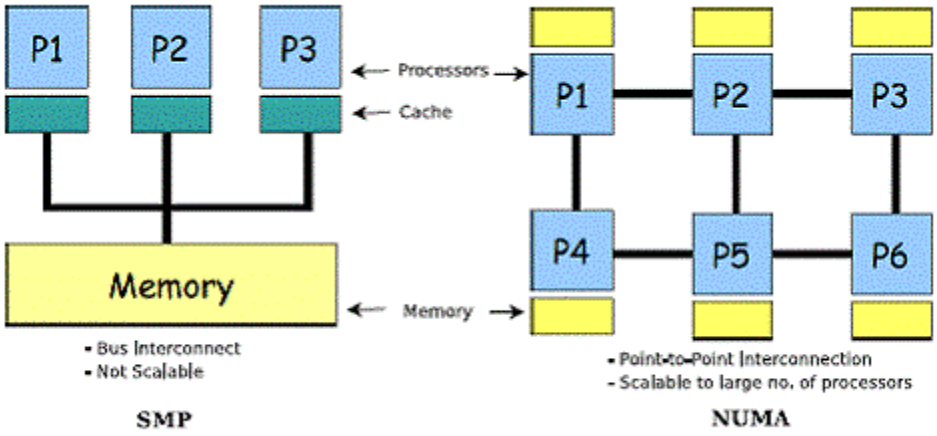
Експериментальні розробки по створенню багатопроцесорних обчислювальних систем почалися в 70-х роках 20 століття. Однією з перших таких систем стала розроблена в Іллінойському університеті МВС ILLIAC IV, яка включала 64 (у проекті до 256) процесорних елемента (ПЕ), що працюють за єдиною програмою, застосовуваної до вмісту власної оперативної пам'яті кожного ПЕ. Обмін даними між процесорами здійснювався через спеціальну матрицю комунікаційних каналів. Зазначена особливість комунікаційної системи дала назву "матричні суперкомп'ютери" відповідного класу МВС. Відзначимо, що більш широкий клас МВС з розподіленою пам'яттю і з довільною комунікаційною системою отримав згодом назву "багатопроцесорні системи з масовим паралелізмом", або МВС з МРР-архітектурою (МРР - MassivelyParallelProcessing). При цьому, як правило, кожен з ПЕ МРР системи є універсальним процесором, чинним за своєю власною програмою (на відміну від загальної програми для всіх ПЕ матричної МВС).

Перші матричні МВС випускалися буквально поштучно, тому їх вартість була фантастично високою. Серійні ж зразки подібних систем, такі як ICL DAP, що включали до 8192 ПЕ, з'явилися значно пізніше, проте не набули широкого поширення зважаючи на складність програмування МВС з одним потоком управління (з однією програмою, спільною для всіх ПЕ).

Перші промислові зразки мультипроцесорних систем з'явилися на базі векторно-конвеєрних комп'ютерів в середині 80-х років. Найбільш поширеними МВС такого типу були суперкомп'ютери фірми Cray. Однак такі системи були надзвичайно дорогими і вироблялися невеликими серіями. Як правило, в подібних комп'ютерах об'єднувалося від 2 до 16 процесорів, які мали рівноправний (симетричний) доступ до загальної оперативної пам'яті. У зв'язку з цим вони отримали назву симетричні мультипроцесорні системи (SymmetricMulti-Processing - SMP).

Як альтернатива таким дорогим мультипроцесорним системам на базі векторно-конвеєрних процесорів була запропонована ідея будувати еквівалентні за потужністю багатопроцесорні системи з великого числа дешевих серійно випускаємих мікропроцесорів. Однак дуже скоро виявилося, що SMP архітектура має дуже обмежені можливості з нарощуванням числа процесорів в системі, через різке збільшення числа конфліктів при зверненні до загальної шини пам'яті. У зв'язку з цим виправданою представлялася ідея забезпечити кожен процесор власною оперативною пам'яттю, перетворюючи комп'ютер у об'єднання незалежних обчислювальних вузлів. Такий підхід значно збільшив ступінь масштабованості багатопроцесорних систем, але у свою чергу зажадав розробки спеціального способу обміну даними між обчислювальними вузлами, реалізованого зазвичай у вигляді механізму передачі повідомлень (MessagePassing). Комп'ютери з такою архітектурою є найбільш яскравими представниками MPP систем. В даний час ці два напрямки (або якісь їхні комбінації) є домінуючими в розвитку суперкомп'ютерних технологій.

Щось середнє між SMP і MPP представляють собою NUMA-архітектури (NonUniformMemoryAccess), в яких пам'ять фізично розділена, але логічно загальнодоступна. При цьому час доступу до різних блоків пам'яті стає неоднаковим. В одній з перших систем цього типу Cray T3D час доступу до пам'яті іншого процесора був у 6 разів більший, ніж до своєї власної.

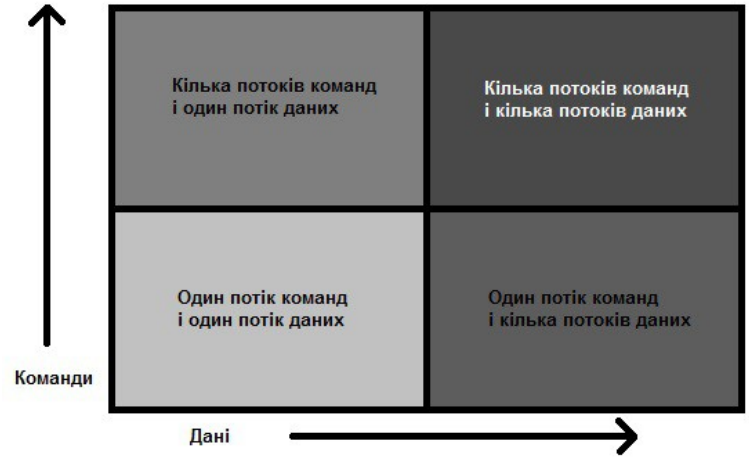


Мал.1.1. SMP і NUMA-архітектури

В даний час розвиток суперкомп'ютерних технологій йде за чотирма основними напрямками: векторно-конвеєрні суперкомп'ютери, SMP системи, MPP системи і кластерні системи. Розглянемо основні особливості перерахованих архітектур.

2. Класифікація паралельних архітектур по Флінну

Загальна класифікація архітектур ЕОМ за ознаками наявності паралелізму в потоках команд і даних була запропонована Майклом Флінном в 1966 році і розширена в 1972 році. Все розмаїття архітектур ЕОМ в цій таксономії зводиться до чотирьох класів:



Мал.1.2. Класифікація паралельних архітектур по Флінну

ОКОД - Обчислювальна система з одиночним потоком команд і одиночним потоком даних

(SISD, SingleInstructionstreamover a SingleDatastream).

ОКМД - Обчислювальна система з одиночним потоком команд і множинним потоком даних

(SIMD, SingleInstruction, MultipleData).

МКОД - Обчислювальна система з множинним потоком команд і одиночним потоком даних

(MISD, MultipleInstructionSingleData).

МКМД - Обчислювальна система з множинним потоком команд і множинним потоком даних

(MIMD, MultipleInstructionMultipleData).

Розглянемо цю класифікацію більш докладно.

SISD - комп'ютери (мал. 1.3) - це звичайні послідовні комп'ютери , що виконують в кожен момент часу тільки одну операцію над одним елементом даних. Більшість сучасних персональних ЕОМ належать саме цій категорії .

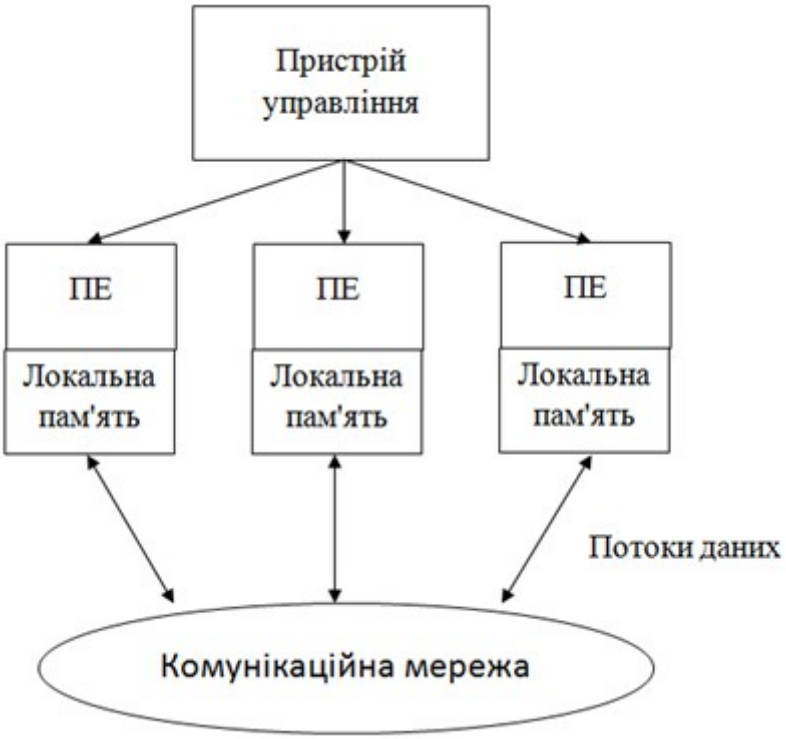
Мал.1.3. Схема SISD - комп'ютера

Багато сучасних обчислювальні системи включають в свій склад кілька процесорів , але кожен з них працює зі своїм незалежним потоком даних , що належать до незалежної програмою . Такий комп'ютер є, фактично, набором SISD - машин, що працюють з незалежними множинами даних.

SIMD - комп'ютери (мал. 1.4 і 1.5) складаються з одного командного процесора (керуючого модуля) , званого контролером , і декількох модулів обробки даних, званих процесорними елементами (ПЕ) . Кількість модулів обробки даних таких машин може бути від 1 024 до 16 384. Керуючий модуль приймає , аналізує і виконує команди.

Типовими представниками SIMD є векторні архітектури.

Мал.1.4. Схема SIMD - комп'ютера з роздільною пам'ятью



Мал.1.5. Схема SIMD - комп'ютера з розподіленою пам'ятью

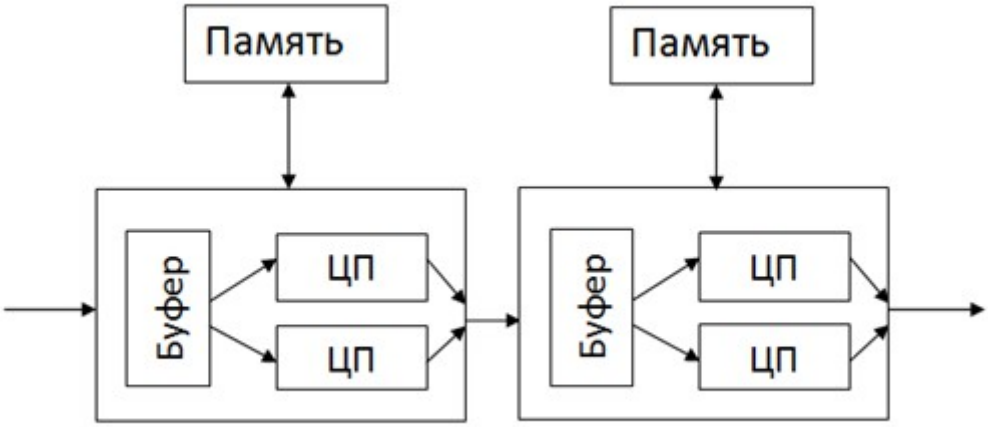
Якщо в команді зустрічаються дані , контролер розсилає на всі ПЕ команду, і ця команда виконується або на декількох , або на всіх процесорних елементах . Процесорні елементи в SIMD - комп'ютерах мають відносно простий пристрій , вони містять арифметико - логічний пристрій (АЛП), що виконує команди, що надходять з устрою управління (УУ) , кілька регістрів і локальну оперативну пам'ять. Однією з переваг даної архітектури вважається ефективна реалізація логіки обчислень .

До половини логічних команд звичайного процесора пов'язане з управлінням процесом виконання машинних команд , а решта їх частина відноситься до роботи з внутрішньою пам'яттю процесора.

У SIMD - комп'ютері управління виконується контролером , а " арифметика" віддана процесорним елементам. Підкласом SIMD - комп'ютерів є векторні комп'ютери . Приклад такої обчислювальної системи - Hitachi S3600 .

Інший приклад SIMD - комп'ютера - матричні процесори (Array Processor) . Як приклад можна привести обчислювальну систему Thinking Machines CM - 2 , де 65536 ПЕ пов'язані між собою мережею комунікацій з топологією " гіперкуб " . Часто комп'ютери з SIMD - архітектурою спеціалізовані для вирішення конкретних завдань , що допускають матричне подання . Це, наприклад, можуть бути завдання обробки зображень , де кожен модуль обробки даних працює на отримання одного елемента кінцевого результату .

Клас MISD включає в себе багатопроцесорні системи, де процесори обробляють множинні потоки даних. Обчислювальних машин такого класу мало. Один з небагатьох прикладів – систолічний масив процесорів , в якому процесори знаходяться в вузлах регулярної решітки. Роль ребер в ній грають міжпроцесорні з'єднання, всі ПЕ управляються загальним тактовим генератором. У кожному циклі роботи кожен ПЕ отримує дані від своїх сусідів , виконує одну команду і передає результат сусідам. На мал. 1.6 дана схема фрагмента систолічного масиву.



Мал.1.6. Схема MISD - комп'ютера

Належність конкретних машин до конкретного класу сильно залежить від точки зору дослідника. Так, конвеєрні машини можуть бути віднесені і до класу SISD (конвеєр - єдиний процесор), і до класу SIMD (векторний потік даних з конвеєрним процесором) і до класу MISD (безліч процесорів конвеєра обробляють один потік даних послідовно), і до класу MIMD - як виконання послідовності різних команд (операцій щаблів конвеєра) над множинним скалярним потоком даних (вектором).

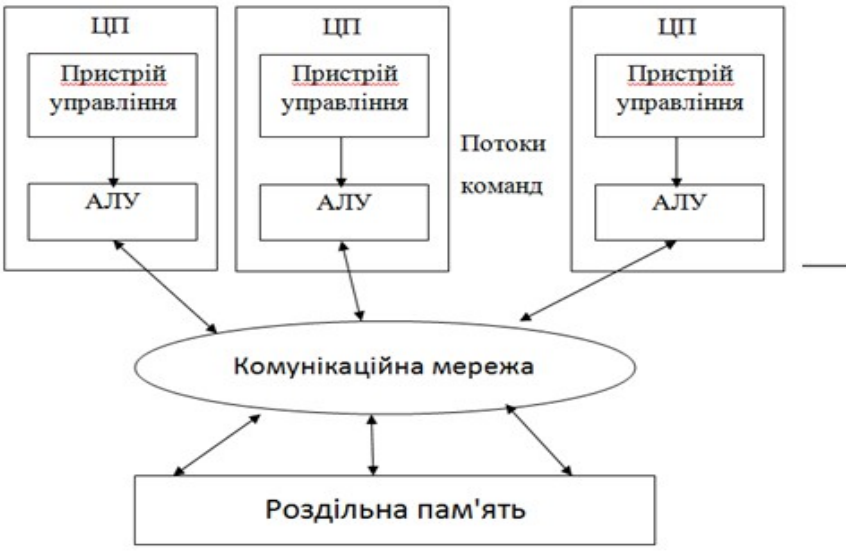
MIMD-комп’ютери.

Цей клас архітектур (рис. 1.7 і 1.8) найбільш багатий прикладами успішних реалізацій . У нього потрапляють симетричні паралельні обчислювальні системи , робочі станції з декількома процесорами , кластери робочих станцій і т . д. Досить давно з'явилися комп'ютери з декількома незалежними процесорами , але спочатку на них був реалізований тільки принцип паралельного виконання завдань, т . е. На різних процесорах одночасно виконувалися незалежні програми .

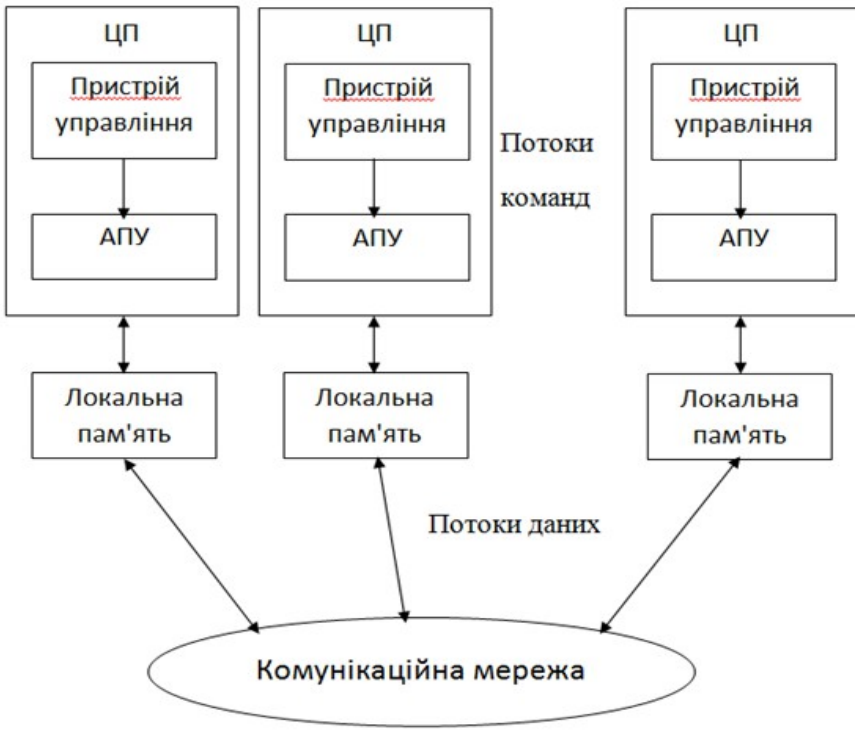
Розробці перших комп'ютерів для паралельних обчислень були присвячені проекти під умовною назвою CM * і C.MMP в університеті Карнегі (США). Технічною базою для цих

проектів були процесори DEC PDP - 11 . На початку 90 -х років минулого століття саме MIMD - комп'ютери вийшли в лідери на ринку високопродуктивних обчислювальних систем .

Розвитком концепції MIMD - архітектури з розподіленою пам'яттю є розподілена обробка , коли замість набору процесорів в одному корпусі використовуються комп'ютери , пов'язані досить швидкої мережею . Концептуального відмінності від MIMD - архітектури з розподіленою пам'яттю немає, а особливістю є повільне мережеве з'єднання.



Мал.1.7. Схема MIMD - комп'ютера з роздільною пам'ятью



Мал.1.8. Схема MIMD - комп'ютера з розподіленою пам'ятью

3. Основні елементи архітектури високопродуктивних обчислювальних систем.

3.1. Конвеєри

Ідея конвеєра полягає в тому, щоб складну операцію розбити на кілька більш простих, таких, які можуть виконуватися одночасно. Операція підсумовування, наприклад, включає віднімання порядків, вирівнювання порядків, складання мантис і нормалізацію. Кожна з підопераций може виконуватися на окремому блоці апаратури. При русі об'єктів по конвеєру одночасно на різних його ділянках (сегментах) виконуються різні підоперації, що дає збільшення продуктивності за рахунок використання паралелізму на рівні команд. При досягненні об'єктом кінця конвеєра він виявиться повністю обробленим. Конвеєри застосовуються як при обробці команд (конвеєри команд), так і в арифметичних операціях (конвеєри даних). Оскільки використання конвеєрної обробки ускладнює конструкцію процесора, ефективним конвеєр даних може бути при виконанні векторних операцій. Операція над одним елементом даних в конвеєрі буде виконуватися довше, ніж у звичайному АЛП.

Для ефективної реалізації конвеєра повинні виконуватися наступні умови:

• система виконує повторювану операцію;

• операція може бути розділена на незалежні частини;

• трудомісткість підопераций приблизно однакова.

Мал.1.9. Конвеєр команд

3.2. RISC-процесори

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 даній області є Intel (процесори Pentium). Ідеї RISC-архітектури використовувалися ще в комп'ютерах CDC6600 (розробники - Крей, Торнтон та ін.). Обидва варіанти відносяться до протилежних кордонів семантичного розриву - зростаючого розриву між програмуванням на мовах високого рівня і програмуванням на рівні машинних команд. 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 користувача програм.

Дослідження показали, що 33% команд типової програми складають пересилання даних, 20% - умовні розгалуження і ще 16% - арифметичні і логічні операції. У переважній більшості команд обчислення адреси може бути виконано швидко, за один цикл. Більш складні режими адресації використовуються приблизно в 18% випадків. Близько 75% операндів є скалярними, т. Е. Змінними цілого, речового, символьного типу і т. Д., А інші є масивами і структурами. 80% скалярних змінних - локальні, а 90% структурних є глобальними. Таким чином, більшість операндів - це локальні операнди скалярних типів. Вони можуть зберігатися в регістрах.

Згідно зі статистикою, більша частина часу витрачається на обробку операторів CALL (виклик підпрограми) і RETURN (повернення з підпрограми). При компіляції ці оператори породжують довгі послідовності машинних команд з великим числом звернень до пам'яті, тому навіть якщо частка цих операторів становить всього 15%, вони споживають основну частину процесорного часу. Тільки близько 1% підпрограм мають більше шести параметрів, а близько 7% підпрограм містять більше шести локальних змінних.

У результаті вивчення цієї статистики був зроблений висновок про те, що в типовій програмі домінують прості операції: арифметичні, логічні та пересилання даних. Домінують і прості режими адресації. Велика частина операндів - це скалярні локальні змінні. Одним з найважливіших ресурсів підвищення продуктивності є оптимізація операторів CALL і RETURN.

Аналіз коду програм, що генерується компіляторами мов високого рівня, показав, що найчастіше використовується тільки обмежений набір простих команд форматів "регістр, регістр → регістр" і "регістр ↔ пам'ять". Компілятори не в змозі ефективно використовувати

складні команди. Це спостереження сприяло формуванню концепції процесорів з скороченим набором команд, так званих RISC-процесорів (RISC - ReducedInstructionSetComputer).

Дейв Паттерсон і Карло Секуін сформулювали 4 основних принципи побудови RISC-процесорів:

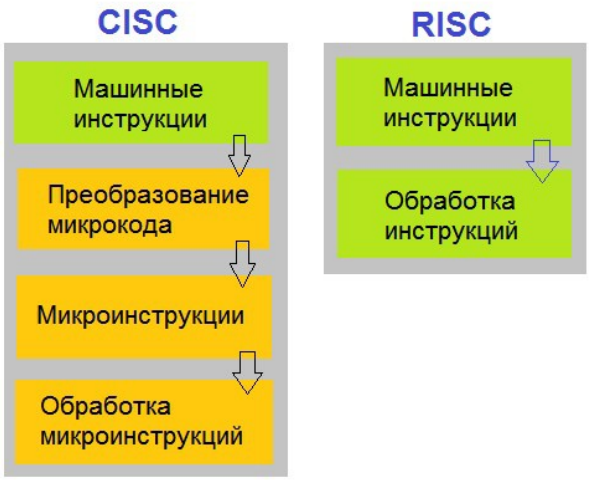
1. Будь-яка операція повинна виконуватися за один такт, незалежно від її типу.

2. Система команд повинна містити мінімальну кількість найбільш часто використовуваних найпростіших інструкцій однакової довжини.

3. Операції обробки даних реалізуються тільки у форматі "регістр - регістр" (операнди вибираються з оперативних регістрів процесора, і результат операції записується також у регістр, а обмін між оперативними регістрами й пам'яттю виконується тільки за допомогою команд завантаження / запису).

4. Склад системи команд повинен бути "зручний" для компіляції операторів мов високого рівня.

Таким чином, RISC-процесори комп'ютерів з скороченим набором команд мають команди обробки типу "регістр ← регістр, регістр" і команди збереження (store) і завантаження (load) типу "пам'ять ← регістр" і "регістр ← пам'ять" відповідно. Функціональні перетворення можуть виконуватися тільки над вмістом регістрів, а результат поміщається тільки в регістр.



Мал.1.10. Порівняння CISC і RISC процесорів

3.3. Суперскалярні процесори

У суперскалярних процесорах реалізована конвеєрна обробка і паралельне виконання команд. У звичайному конвеєрі паралельне виконання команд можливо тільки при знаходженні команд на різних стадіях обробки (у різних сегментах конвеєра). Суперскалярні процесори

дозволяють виконувати декілька команд в одному сегменті конвеєра. Кілька команд одночасно можуть виконатися протягом одного такту.

У суперскалярних процесорах використовується декілька конвеєрів, що працюють паралельно, і пристрої для інтерпретації команд, забезпечені логікою, що дозволяють визначити, чи є команди незалежними.

Паралельне виконання команд не завжди можливо через ті ж конфлікти, що і в конвеєрі. Для вирішення можливих конфліктів використовують методи позачергової вибірки і завершення команд, прогнозування переходів, умовне виконання команд та ін. Застосовується динамічний розподіл команд, причому порядок їх вибірки може не збігатися з порядком проходження в програмі. Зрозуміло, результат виконання повинен збігатися з результатом строго послідовного виконання .

У суперскалярних комп'ютерах питання про паралелізм команд вирішується апаратно. Програми для них сумісні на рівні бінарних (виконуваних) файлів.

Недоліки: складність апаратної частини; обмежений розмір вікна виконання, що зменшує можливості визначення потенційно паралельних команд.

Представниками суперскалярних процесорів є Pentium, PowerPC, K6 / K7, Alpha.

3.4. Суперскалярні процесори зі наддовгим командним словом

Суперскалярні процесори зі наддовгим командним словом – VLIW (VeryLongInstructionWord) в процесі роботи виявляють паралелізм команд під час трансляції. Транслятор, аналізуючи програму, виділяє операції, які можна виконувати паралельно, і будує з них «великі» команди. Як тільки «велика» команда надходить на виконання, звичайні команди з неї виконуються паралельно. Така реалізація веде до ряду недоліків (немає сумісності на рівні бінарних файлів, необхідний канал з великою розрядністю, збільшується виконуваний код).

Архітектура VLIW - одна з реалізацій внутрішнього паралелізму в мікропроцесорах. Швидкодію можна підвищувати за рахунок збільшення тактової частоти або за рахунок кількості операцій, які виконуються за один такт. Перший спосіб можна реалізувати при використанні «швидких» технологій (використовувати замість кремнію арсенід галію) і глибокою конвеєризацією (задіяти всі логічні елементи кристала). Щоб реалізувати другий спосіб, необхідно розмістити на одному чипі кілька функціональних модулів та забезпечити надійність при паралельному виконанні інструкцій.

Говорячи про надійність, мається на увазі вірність результату. Якщо необхідно виконати наступні дії A = B + C, B = E + H. Очевидно що результат першого виразу залежить від результату другого. При плануванні порядку обчислень необхідно враховувати такі нюанси.

Якщо не буде подібного контролю, то отримати вірний результат обчислення можна лише з якоюсь імовірністю.

Мал.1.11. Схема процесора з наддовгим командним словом

4. Архітектура багатоядерних процесорів

Архітектура багатоядерних процесорів в багато чому повторює архітектуру симетричних мультипроцесорів (SMP-машин) тільки в менших масштабах і зі своїми особливостями. У багатоядерних процесорах тактова частота, як правило, навмисно знижена. Це дозволяє зменшити енергоспоживання процесора без втрати продуктивності. У деяких процесорах тактова частота кожного ядра може змінюватися в залежності від його індивідуального навантаження. Ядро є повноцінним мікропроцесором, що використовують всі досягнення мікропроцесорної техніки: конвеєри, позачергове виконання коду, багаторівневий кеш, підтримка векторних команд. Суперскалярність в ядрі не використовується, так як вона реалізована самою наявністю декількох ядер в процесорі.

Багатоядерні процесори можна підрозділити по наявності підтримки когерентності (загальної) кеш-пам'яті між ядрами. Бувають процесори з такою підтримкою і без неї. Спосіб зв'язку між ядрами:

- розділяєма шина;
- мережа (Mesh) на каналах точка-точка;
- мережа з комутатором;

- загальна кеш-пам'ять;

Кеш-пам'ять: У всіх існуючих на сьогодні багатоядерних процесорах кеш-пам'яттю 1-го рівня володіє кожне ядро окремо, а кеш-пам'ять 2-го рівня існує в декількох варіантах:

- розділювана - розташована на одному кристалі з ядрами і доступна кожному з них у повному обсязі. Використовується в процесорах сімейств IntelCore.
- індивідуальна - окремі кеши рівного об'єму, інтегровані в кожне з ядер. Обмін даними з кешем 2-го рівня між ядрами здійснюється через контролер пам'яті - інтегрований (Athlon 64 X2, Turion X2, Phenom) або зовнішній (використовувався в Pentium D, надалі Intel відмовилася від такого підходу).

Багатоядерні процесори також мають гомогенну або гетерогенну архітектуру:

- гомогенна архітектура - всі ядра процесора однакові і виконують одні й ті ж завдання. Типові приклади: IntelCoreDuo, Sun SPARC T3, AMD Opteron
- гетерогенна архітектура - ядра процесора виконують різні завдання. Типовий приклад: процесор Cell альянсу IBM, Sony і Toshiba, у якого з дев'яти ядер одне є ядром процесора загального призначення PowerPC, а вісім інших - спеціалізованими процесорами, оптимізованими для векторних операцій, які використовуються в ігровій приставці SonyPlayStation 3.
-

5. Різниця між многоядерною архітектурою і технологією гіперпоточності

Hyper-threading (англ. Hyper-threading - гіперпоточність, офіційна назва - hyper-threadingtechnology, HTT або HT) - технологія, розроблена компанією Intel для процесорів на мікроархітектурі NetBurst. HTT реалізує ідею «одночасної мультипоточності» (англ. Simultaneousmultithreading, SMT). HTT є розвитком технології суперпоточності (англ. Super-threading), що з'явилася в процесорах IntelXeon в лютому 2002 і в листопаді 2002 доданої в процесори Pentium 4. Після включення HTT один фізичний процесор (одне фізичне ядро) визначається операційною системою як два окремих процесора (два логічних ядра). За певних робочих навантаженнях використання HTT дозволяє збільшити продуктивність процесора. Суть технології: передача «корисної роботи» (англ. Usefulwork) бездіяльним виконавчим пристроям (англ. Executionunits).

Процесор, що підтримує технологію hyper-threading:

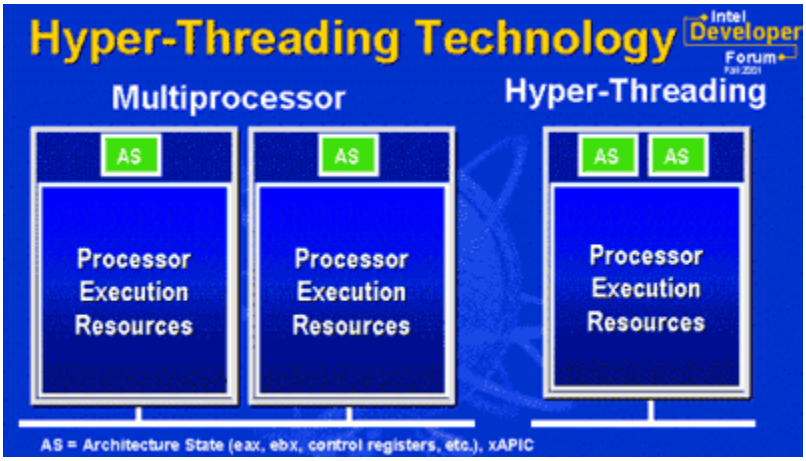
- може зберігати стан відразу двох потоків;
- має по одному набору регістрів і по одному контролеру переривань (APIC) на кожен логічний процесор;

Для операційної системи це виглядає як наявність двох логічних процесорів (англ. Logicalprocessor). У кожного логічного процесора є свій набір регістрів і контролер переривань (APIC). Інші елементи фізичного процесора є загальними для всіх логічних процесорів.

Розглянемо приклад. Фізичний процесор виконує потік команд першого логічного процесора. Виконання потоку команд припиняється по одній з наступних причин:

- відбулась помилка при зверненні до кешу процесора;
- виконано невірне передбачення розгалуження;
- очікується результат попередньої інструкції.

Фізичний процесор не буде бездіяльним, а передасть управління потоку команд другого логічного процесора. Таким чином, поки один логічний процесор очікує, наприклад, дані з пам'яті, обчислювальні ресурси фізичного процесора будуть використовуватися другим логічним процесором.



Мал.1.12. Порівняння багатоядерної архітектури з технологією гіперпоточності

Кожне ядро також може використовувати технологію SMT для почергового виконання декількох потоків, створюючи ілюзію декількох «логічних процесорів» на основі кожного ядра. На процесорах компанії Intel ця технологія носить назву Hyper-threading і подвоює число логічних процесорів в порівнянні з фізичними. На процесорах SunUltraSPARC таке збільшення може досягати 8 потоків на ядро.

Перевагами HTT вважаються:

- можливість запуску декількох потоків одночасно (багатопотоковий код);
- зменшення часу відгуку;
- збільшення числа користувачів, що обслуговуються сервером.

6. Векторно-конвеєрні комп'ютери

Перший векторно-конвеєрний комп'ютер Cray-1 з'явився в 1976 році. Архітектура його виявилася настільки вдалою, що він поклав початок цілому сімейству комп'ютерів. Назву цього сімейства комп'ютерів дали два принципи, закладені в архітектурі процесорів:

- конвеєрна організація обробки потоку команд;
- запровадження в систему команд набору векторних операцій, які дозволяють оперувати з цілими масивами даних.

Мал.1.13. Перший векторно-конвеєрний комп'ютер Cray-1

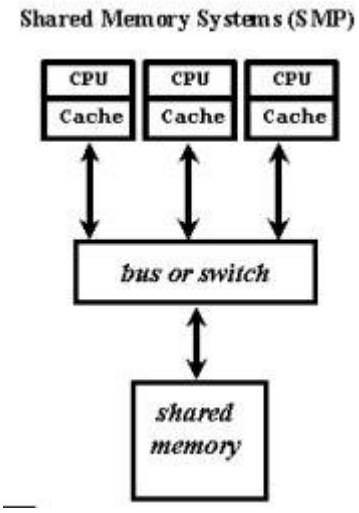
Довжина одночасно оброблюваних векторів в сучасних векторних комп'ютерах складає, як правило, 128 або 256 елементів. Очевидно, що векторні процесори повинні мати набагато більш складну структуру і, по суті справи, містити безліч арифметичних пристроїв. Основне призначення векторних операцій полягає в розпаралелюванні виконання операторів циклу, в

яких в основному і зосереджена велика частина обчислювальної роботи. Для цього цикли піддаються процедурі векторізації з тим, щоб вони могли реалізовуватися з використанням векторних команд. Як правило, це виконується автоматично компіляторами при виготовленні ними виконуваного коду програми. Тому векторно-конвеєрні комп'ютери не вимагали якоїсь спеціальної технології програмування, що і стало вирішальним фактором у їхньому успіху на комп'ютерному ринку. Тим не менш, вимагали дотримання деяких правил при написанні циклів з тим, щоб компілятор міг їх ефективно розпаралелювати.

Історично це були перші комп'ютери, до яких повною мірою було застосовне поняття суперкомп'ютер. Як правило, кілька векторно-конвеєрних процесорів (2-16) працюють в режимі із загальною пам'яттю (SMP), утворюючи обчислювальний вузол, а кілька таких вузлів об'єднуються за допомогою комутаторів, утворюючи або NUMA, або MPP систему. Типовими представниками такої архітектури є комп'ютери CRAY J90 / T90, CRAY SV1, NEC SX-4 / SX-5. Рівень розвитку мікроелектронних технологій не дозволяє в даний час виробляти однокристальні векторні процесори, тому ці системи досить громіздкі і надзвичайно дорогі. У зв'язку з цим, починаючи з середини 90-х років, коли з'явилися досить потужні суперскалярні мікропроцесори, інтерес до цього напрямку був значною мірою ослаблений. Суперкомп'ютери з векторно-конвеєрною архітектурою стали програвати системам з масовим паралелізмом. Однак у березні 2002 р корпорація NEC представила систему EarthSimulator з 5120 векторно-конвеєрних процесорів, яка в 5 разів перевищила продуктивність попереднього власника рекорду - MPP системи ASCI White з 8192 суперскалярних мікропроцесорів. Це, звичайно ж, змусило багатьох по-новому поглянути на перспективи векторно-конвеєрних систем.

7. Симметричні мультипроцесорні системи SMP

Характерною рисою багатопроцесорних систем SMP архітектури є те, що всі процесори мають прямий і рівноправний доступ до будь-якої точки загальної пам'яті. Перші SMP системи складалися з декількох однорідних процесорів і масиву загальної пам'яті, до якої процесори підключалися через загальну системну шину. Однак дуже скоро виявилося, що така архітектура непридатна для створення масштабних систем. Першою виникла проблема - велике число конфліктів при зверненні до загальної шини. Гостроту цієї проблеми вдалося частково зняти поділом пам'яті на блоки, підключення до яких за допомогою комутаторів дозволило розпаралелити звернення від різних процесорів. Однак і в такому підході неприйнятно великими здавалися накладні витрати для систем більш ніж з 32-ма процесорами.



Мал.1.14. SMP структура

Наявність загальної пам'яті значно спрощує організацію взаємодії процесорів між собою і спрощує програмування, оскільки паралельна програма працює в єдиному адресному просторі. Однак за цією уявною простотою ховаються великі проблеми, властиві системам цього типу. Всі вони, так чи інакше, пов'язані з оперативною пам'яттю. Справа в тому, що в даний час навіть у однопроцесорних системах найвужчим місцем є оперативна пам'ять, швидкість роботи якої значно відстала від швидкості роботи процесора. Для того щоб згладити цей розрив, сучасні процесори забезпечуються швидкісний буферною пам'яттю (кеш-пам'яттю), швидкість роботи якої значно вище, ніж швидкість роботи основної пам'яті.

Очевидно, що при проектуванні багатопроцесорних систем ці проблеми ще більш загострюються. Крім добре відомої проблеми конфліктів при зверненні до загальної шини пам'яті виникла і нова проблема, пов'язана з ієрархічною структурою організації пам'яті сучасних комп'ютерів. У багатопроцесорних системах, побудованих на базі мікропроцесорів з вбудованою кеш-пам'яттю, порушується принцип рівноправного доступу до будь-якої точки пам'яті. Дані, що знаходяться в кеш-пам'яті деякого процесора, недоступні для інших процесорів. Це означає, що після кожної модифікації копії деякої змінної, що знаходиться в кеш-пам'яті якого-небудь процесора, необхідно виробляти синхронну модифікацію самої цієї змінної, розташованої в основній пам'яті.

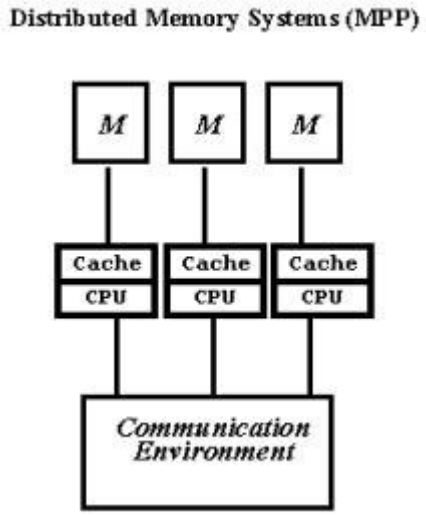
Ще одним неприємною властивістю SMP систем є те, що їх вартість зростає швидше, ніж продуктивність при збільшенні числа процесорів в системі.

8. Системи з масовим паралелізмом (MPP)

Проблеми, властиві багатопроцесорним системам із загальною пам'яттю, простим і природним чином усуваються в системах з масовим паралелізмом. Комп'ютери цього типу

являють собою багатопроцесорні системи з розподіленою пам'яттю, в яких за допомогою деякого комунікаційного середовища об'єднуються однорідні обчислювальні вузли.

Кожен з вузлів складається з одного або декількох процесорів, власної оперативної пам'яті, комунікаційного устаткування, підсистеми вводу / виводу, тобто володіє всім необхідним для незалежного функціонування. При цьому на кожному вузлі може функціонувати або повноцінна операційна система (як у системі RS / 6000 SP2), або урізаний варіант, що підтримує тільки базові функції ядра, а повноцінна ОС працює на спеціальному керуючому комп'ютері (як в системах Cray T3E, nCUBE2).



Мал.1.15. MPP структура

Процесори в таких системах мають прямий доступ тільки до своєї локальної пам'яті. Доступ до пам'яті інших вузлів реалізується зазвичай за допомогою механізму передачі повідомлень. Така архітектура обчислювальної системи усуває одночасно як проблему конфліктів при зверненні до пам'яті, так і проблему когерентності кеш-пам'яті. Це дає можливість практично необмеженого нарощування числа процесорів в системі, збільшуючи тим самим її продуктивність. Успішно функціонують MPP системи з сотнями і тисячами процесорів (ASCI White - 8192, BlueMountain - 6 144). Продуктивність найбільш потужних систем досягає 10000000000000 оп / сек (10 Tflops). Важливою властивістю MPP систем є їх висока ступінь масштабованості. Залежно від обчислювальних потреб для досягнення необхідної продуктивності потрібно просто зібрати систему з потрібним числом вузлів.

На практиці все, звичайно, набагато складніше. Усунення одних проблем, як це зазвичай буває, породжує інші. Для MPP систем на перший план виходить проблема ефективності комунікаційного середовища. Легко сказати: "Давайте зберемо систему з 1000 вузлів". Але яким чином поєднати в єдине ціле таку безліч вузлів? Найпростішим і найефективнішим було би з'єднання кожного процесора з кожним. Але тоді на кожному вузлі було б потрібно 999 комунікаційних каналів, бажано двонаправлених. Очевидно, що це нереально. Різні виробники

MPP систем використовували різні топології. У комп'ютерах IntelParagon процесори утворювали прямокутну двовимірну сітку. Для цього в кожному вузлі достатньо чотирьох комунікаційних каналів. У комп'ютерах Cray T3D / T3E використовувалася топологія тривимірного тора. Відповідно, у вузлах цього комп'ютера було шість комунікаційних каналів. Фірма nCUBE використовувала в своїх комп'ютерах топологію n-мірного гіперкуба.

Системи з розподіленою пам'яттю ідеально підходять для паралельного виконання незалежних програм, оскільки при цьому кожна програма виконується на своєму вузлі і ніяким чином не впливає на виконання інших програм. Однак при розробці паралельних програм доводиться враховувати складнішу, ніж в SMP системах, організацію пам'яті. Оперативна пам'ять в MPP системах має 3-х рівневу структуру:

- кеш-пам'ять процесора;
- локальна оперативна пам'ять вузла;
- оперативна пам'ять інших вузлів.

При цьому відсутня можливість прямого доступу до даних, розташованим в інших вузлах. Для їх використання ці дані повинні бути попередньо передані в той вузол, який в даний момент їх потребує. Це значно ускладнює програмування. Крім того, обміни даними між вузлами виконуються значно повільніше, ніж обробка даних в локальній оперативної пам'яті вузлів. Тому написання ефективних паралельних програм для таких комп'ютерів являє собою більш складну задачу, ніж для SMP систем.