МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

ПРЕПЕЛИЦЯ Г. П.
ПОНОМАРЕНКО О. Л.

КОМП'ЮТЕРНА СХЕМОТЕХНІКА ТА АРХІТЕКТУРА КОМП'ЮТЕРІВ

Конспект лекцій

Одеса
Одеський державний екологічний університет
2016

УДК 004.235
П71

Рекомендовано методичною радою Одеського державного екологічного університету Міністерства освіти і науки України як конспект лекцій (протокол №10 від 04.07. 2016 р.)

Препелиця Г. П., Пономаренко О. Л.
Комп'ютерна схемотехніка та архітектура компютерів: конспект лекцій. Одеса, Одеський державний екологічний університет, 2016. 121с.

В конспекті лекцій розглядаються основи теорії побудови та функціонування пристроїв, вузлів, базових елементів та архітектури сучасної комп'ютерної техніки, що виконані на базі інтегральної технології. Рекомендовано для студентів галузі знань 12 "Інформаційні технології" першого (бакалаврського) рівня.

ISBN 978-966-186-094-9

ЗМІСТ

1. ЦИФРОВІ КОМП’ЮТЕРИ 4
1.1. Функціональна структура комп’ютера 5
1.2. Пристрої та процес введення-виведення 13
1.3. Принципи організації систем переривання программ 16
1.4. Прямий доступ до пам’яті 20
2. ПРИСТРОЇ ПАМ'ЯТІ 23
2.1. Класифікація запам'ятовуючих пристроїв 23
2.2. Типи і характеристика пристроїв пам'яті 24
2.2.1. Принцип побудови запам'ятовуючих пристроїв з довільним доступом 26
2.2.2. Організація пам'яті з лінійною адресацією 28
2.3. Організація пам'яті з двовимірною адресацією 31
2.4. Статична оперативна пам'ять на біполярних- і МОП-транзисторах ... 34
2.5. Динамічні запам'ятовуючі пристрої 34
2.6. Оперативний запам'ятовуючий пристрій 36
2.7. Постійні запам'ятовуючі пристрої 38
2.8. Флеш-пам'ять 43
2.9. Пам'ять з послідовним доступом на основі регістрів зсуву 46
2.10. Асоціативна пам'ять 47
2.11. Принципи контролю роботи пам'яті 48
2.11.1. Код з перевіркою парності 49
2.11.2. Контроль роботи пам'яті ЕОМ 51
3. ПРОЦЕСОРИ 54
4. СУПЕРКОМП’ЮТЕРИ. ПАРАЛЕЛЬНІ ОБЧИСЛЮВАЛЬНІ СИСТЕМИ 64
4.1. Паралелізм на рівні команд 66
4.2. Паралелізм на рівні процесорів 71
5. УНІВЕРСАЛЬНІ МІКРОПРОЦЕСОРИ. СХЕМИ ПІДТРИМКИ МП НА СИСТЕМНИХ ПЛАТАХ 81
5.1. Характеристики мікропроцесорів 81
5.2. Структура і особливості роботи мікропроцесорів 82
5.3. Алгоритм роботи мікропроцесора 85
5.4. Системні керуючі сигнали 89
5.5. Схеми підтримки мп на системних платах ЦАП і АЦП 90
5.6. Імпульсні джерела електроживлення 104
6. АРХІТЕКТУРИ З ПОВНИМ І СКОРОЧЕНИМ НАБОРОМ КОМАНД. 110
6.1. Основні риси RISC-архітектури 111

6.2. Регістри в RISC-процесорах 113
6.3. Переваги та недоліки RISC 118

1 ЦИФРОВІ КОМП’ЮТЕРИ

Цифровий комп'ютер (англ. Digital computer, цифрова електронна обчислювальна машина, ЦЕОМ) - комп'ютер, що обробляє дискретні дані (інформацію), представлені у вигляді цифр. У зв'язку з можливістю подання безперервної інформації в дискретному вигляді (наприклад, у вигляді послідовності цифр) з певною точністю перетворення (дискретизації) і в зв'язку з відносною простотою роботи з дискретною інформацією переважна більшість сучасних комп'ютерів є цифровими. Тому визначення «цифровий» найчастіше опускають. Уточнення проводиться у випадках, коли це особливо необхідно, наприклад, в складних обчислювальних системах, що включають комп'ютери різних видів.

Існуючі в даний час типи комп'ютерів розрізняються розмірами, вартістю, обчислювальною потужністю і призначенням. Найбільш поширеним типом комп'ютерів є персональні комп'ютери.

Настільні комп'ютери - найбільш популярна форма персональних комп'ютерів.

Портативним комп'ютером (ноутбуком) називається компактна версія персонального комп'ютера, в якій усі компоненти розміщуються в одному блоці, який має розмір невеликого тонкого портфеля.

Робочі станції, що мають розмір настільних комп'ютерів, мають значно більшу обчислювальну потужність, ніж персональні комп'ютери. Вони мають графічні вхідні і вихідні пристрої з високою роздільною здатністю. Вони часто використовуються при виконанні інженерних розрахунків, в першу чергу для вирішення завдань автоматизованого проектування. Існує ще цілий спектр великих і дуже потужних комп'ютерних систем - від корпоративних серверів, до суперкомп'ютерів. Корпоративні сервери і мейнфрейми використовуються для обробки ділових даних у великих корпораціях, яким необхідні значно більша обчислювальна потужність і ємність запам'ятовуючих пристроїв (ЗУ), ніж можуть забезпечити робочі станції.

Сервери використовуються для зберігання баз даних і можуть обробляти велику кількість запитів. Вони широко використовуються в сфері освіти, в бізнесі, в системах Інтернету.

Суперкомп'ютери призначені для проведення великомасштабних числових обчислень при аналізі, наприклад, метеорологічних систем або при конструюванні літальних апаратів.

1.1 Функціональна структура комп'ютера

Комп'ютер складається з п'яти головних, функціонально незалежних частин: пристрою введення, пристрою пам'яті, арифметико-логічного пристрою, пристрою виведення і пристрою управління.

Пристрій введення приймає через цифрові лінії зв'язку закодовану інформацію від операторів, електромеханічних пристроїв типу клавіатури або від інших комп'ютерів мережі. Отримана інформація або зберігається в пам'яті комп'ютера для подальшого застосування, або негайно використовується арифметичними і логічними схемами для виконання необхідних операцій. Послідовність кроків обробки визначається програмою, яка зберігається в пам'яті. Отримані результати відправляються назад, до зовнішнього світу, за допомогою пристрою виводу. Всі ці дії координуються блоком управління.

Рисунок 1.1 - Базові функціональні пристрої комп'ютера

На рисунку не показані зв'язки між функціональними пристроями, тому що такі зв'язки можуть реалізовуватись по-різному.

Арифметичні і логічні схеми в комплексі з головними керуючими схемами називають «процесором», а все разом узяте обладнання для введення і виведення часто називають «пристроєм введення-виведення» (input-output unit).

Оброблювана комп'ютером інформація поділяється на дві основні категорії: команди і дані.

Команди, або машинні команди - це явно задані інструкції, які:

- керують пересиланням інформації всередині комп'ютера, а також між комп'ютером і його пристроями введення-виведення;

- визначають арифметичні і логічні операції, які підлягають виконанню

Список команд, які виконують деяку задачу, називається «програмою». Зазвичай програма зберігається в пам'яті. Процесор по черзі витягує команди програми з пам'яті і реалізує визначені ними операції. Комп'ютер повністю управляється програмою, що зберігається, якщо не брати до уваги можливість зовнішнього втручання оператора і приєднаних до машини пристроїв введення-виведення.

Дані - це числа і закодовані символи, які використовуються в якості операндів команд. Однак термін «дані» часто використовується для позначення будь-якої цифрової інформації. За цим визначенням сама програма також може вважатися даними, якщо вона обробляється іншою програмою. Таким прикладом обробки однієї програми іншою є компіляція вихідної програми, написаної на мові високого рівня, в список машинних команд, що складають програму на машинній мові, яка називається об'єктною програмою. Фактично вона являє собою набір нулів і одиниць, званих бітами. Для завдання чисел зазвичай використовується позиційне двійкове подання. Іноді застосовується двійковий-десятковий формат, відповідно до якого кожна десяткова цифра кодується окремо за допомогою чотирьох біт.

Найбільш поширений неупакований формат подання чисел і букв, званий кодом ASCII (American Standard Code for Information Interchange - американський стандартний код для обміну інформацією).

Пристрій введення

Комп'ютер приймає кодовану інформацію через пристрій введення, завданням якого є читання даних. Найбільш поширеним пристроєм введення є клавіатура. Коли користувач натискає кнопку, відповідна літера чи цифра автоматично перетворюється в певний двійковий код і по кабелю пересилається або в пам'ять, або процесору.

Існує і ряд інших пристроїв введення, серед яких джойстики, трекболи та миші. Вони використовуються спільно з дисплеєм в якості графічних вхідних пристроїв. Для введення звуку можуть використовуватися мікрофони. Звукові коливання, які сприймаються ними, вимірюються і конвертуються в цифрові коди для зберігання і обробки.

Блок пам'яті

Завданням блоку пам'яті є зберігання програм і даних. Існує два класи запам'ятовуючих пристроїв - первинні (або внутрішні) і вторинні (або зовнішні).

Первинний запам'ятовуючий пристрій (ЗП) - це пам'ять, швидкодія якої визначається швидкістю роботи електронних схем. Поки програма виконується, вона повинна зберігатися в первинній пам'яті. Ця пам'ять складається з великого числа напівпровідникових комірок, кожна з яких може зберігати один біт інформації. Комірки рідко зчитуються окремо - зазвичай вони обробляються групами фіксованого розміру, званими словами. Пам'ять організована так, що вміст одного слова, яке містить n-біт, може записуватися або зчитуватися за одну базову операцію.

Для полегшення доступу до слів в пам'яті, з кожним словом пов'язується окрема адреса. Адреса - це числа, які вказують конкретні місця розташування слів в пам'яті. Для того щоб прочитати слово з пам'яті або записати його в пам'ять, необхідно вказати його адресу і задати керуючу команду (читання або запису), яка почне відповідну операцію.

Кількість бітів в кожному слові називають довжиною машинного слова. Зазвичай слова мають довжину від 16 до 64 біт.

Одним з факторів, що характеризують клас комп'ютера, є ємність його пам'яті. Малі машини зазвичай можуть зберігати лише кілька десятків мільйонів слів, тоді як середні і великі - сотні мільйонів слів.

Типовими одиницями вимірювання кількості оброблюваних машиною даних є слово, кілька слів або частина слова. Як правило, за час одного звернення до пам'яті зчитується або записується тільки одне слово.

Дуже важливою є можливість швидкого доступу до будь-якого слова пам'яті. Пам'ять, до будь-якої точки якої можна отримати доступ за короткий і фіксований час, називається пам'яттю з довільним доступом (Random Access Memory, RAM). Час, необхідний для доступу до одного слова, називається часом доступу до пам'яті. Цей час завжди однаковий, незалежно від того, де розташовується потрібне слово. Час доступу до пам'яті в сучасних RAM становить від 100 нс до декількох нс. Пам'ять комп'ютера зазвичай являє собою ієрархічну структуру, яка складається з трьох або чотирьох рівнів RAM-елементів з різною швидкістю і різним розміром. Найбільш швидкодіючим типом RAM-пам'яті є кеш-пам'ять (або просто кеш). Вона безпосередньо пов'язана з процесором і часто розташована на одному з ним інтегрованому чипі, завдяки чому робота процесора значно прискорюється. Пам'ять більшої місткості, але менш швидка, називається основною пам'яттю (main memory).

Первинні ЗП досить дорого коштують. Тому комп'ютери обладнуються додатковими, більш дешевими вторинними ЗП, які використовуються для зберігання великих обсягів даних і великої кількості програм. З них найбільшого поширення набули магнітні диски, магнітні стрічки і оптичні диски.

Арифметико-логічний пристрій

Більшість комп'ютерних операцій виконується в арифметико-логічному пристрої (АЛП) процесора. Якщо, наприклад, необхідно скласти два числа, які знаходяться в пам'яті, то ці числа пересилаються в процесор, де АЛП виконує їх складання. Отримана сума може бути записана в пам'ять або залишена в процесорі для негайного використання.

Будь-які інші арифметичні або логічні операції, в тому числі множення, ділення і порівняння чисел, починаються з пересилання цих чисел в процесор, де АЛП має виконати відповідну операцію. Коли операнди переносяться в

процесор, вони зберігаються у високошвидкісних елементах пам'яті, що називаються регістрами. Кожен регістр може зберігати одне слово даних. Час доступу до регістрів процесора навіть менший від часу доступу до найшвидшої кеш-пам'яті.

Керуючі і арифметико-логічні пристрої працюють у багато разів швидше, ніж всі інші пристрої, підключені до комп'ютерної системи. Це дозволяє одному процесору контролювати безліч зовнішніх пристроїв, таких як клавіатура, дисплеї, магнітні та оптичні диски, сенсори і механічні керуючі пристрої.

Блок виведення

Функція блоку виведення протилежна функції блоку введення: він направляє результати обробки у зовнішній світ. Типовий приклад пристрою виведення - принтер (механічні, струменеві, лазерні). Існують принтери, здатні друкувати до 10000 рядків в хвилину. Для механічного пристрою це величезна швидкість, але в порівнянні з швидкодією процесора вона мізерно мала.

Деякі пристрої, наприклад графічні дисплеї, виконують одночасно і функцію введення, і функцію виведення. Тому вони називаються пристроями введення-виведення.

Блок керування

Роботу пристроїв пам'яті, АЛП, введення і виведення необхідно координувати. Цю функцію виконує блок керування. Він передає керуючі сигнали іншим пристроям і відстежує їх стан.

У блоці управління керуючими схемами виробляються так звані синхронизуючі сигнали, які управляють пересиланням. Синхронизуючі сигнали - це сигнали, які визначають, коли має бути виконано дану дію. Крім того, за допомогою синхронізуючих сигналів, що генеруються блоком управління, здійснюється передача даних між процесором і пам'яттю.

Блок управління можна представити як окремий пристрій, що взаємодіє з іншими частинами машини. Але на практиці більшість управляючих схем фізично розподілена по різних місцях комп'ютера.

Особливості функціонування комп'ютера

Для виконання конкретного завдання в пам'ять комп'ютера записується відповідна програма, що складається з ряду команд. Команди по черзі пересилаються з пам'яті в процесор, який їх виконує. Дані, що використовуються в якості операндів команд, також зберігаються в пам'яті.

Нехай необхідно виконати команду Add ADR, R0.

Ця команда додає операнд, що зберігається в пам'яті за адресою ADR, з операндом, що зберігаються в регістрі R0 процесора, і поміщає результат в цей же регістр. Початковий вміст пам'яті за адресою ADR не змінюється, а вміст регістра R0 перезаписується. Ця команда виконується в кілька етапів. Спочатку вона пересилається з пам'яті в процесор. Потім операнд команди зчитується з пам'яті за адресою ADR і додається до даного, який міститься у регістрі R0, після чого сума записується в регістр R0.

У розглянутій команді Add об'єднуються дві операції:

- доступ до пам'яті;
- операція АЛП.

У багатьох сучасних комп'ютерах ці два типи операцій виконуються за допомогою окремих команд. Такий поділ пов'язаний з прагненням збільшити продуктивність комп'ютера. Тому розглянута команда може реалізовуватися двома командами:

Load ADR, R1

Add R1, R0

Команда Load копіює вміст пам'яті за адресою ADR в регістр R1, а Add - додає вміст регістрів R1 і R0 і поміщає суму в регістр R0.

В результаті виконання двох команд початковий вміст двох регістрів R1, R0 знищується, а вміст пам'яті за адресою ADR зберігається.

Крім АЛП і керуючих схем процесор містить безліч регістрів, призначених для різних цілей. У регістрі команд IR (Instruction Register) міститься код команди, яка виконується в даний момент.

Ще один спеціалізований регістр, званий лічильником команд PC (Program Counter), служить для контролю за ходом виконання програми. У ньому міститься адреса наступної команди, що підлягає вибірці і виконанню.

Поки виконується чергова команда, вміст PC оновлюється - в нього записується адреса наступної команди. Кажуть, що регістр PC вказує на команду, яка повинна бути обрана з пам'яті.

Крім зазначених регістрів, в процесорі містяться регістри загального призначення від R0 до Rn-1. У сучасних комп'ютерах число РЗП від 8 до 32, а іноді і більше. Доступ до даних в цих регістрах здійснюється набагато швидше, ніж до пам'яті, в тому числі і КЕШ, оскільки регістри розташовуються усередині процесора. Для того, щоб вказати, який саме регістр бере участь в операції, достатньо всього кілька біт. Так для завдання одного з 32 регістрів потрібно 5 біт. Це набагато менше, ніж для завдання адреси в пам'яті.

Таким чином, оскільки РЗП дозволяють швидше обробляти результати і застосовувати більш короткі команди, вони широко використовуються для тимчасового зберігання даних, що оброблюються .

Ще два регістри забезпечують взаємодію з пам'яттю. Це регістр адреси MAR (Memory Address Register) і регістр даних MDR (Memory Data Register). У регістрі MAR міститься адреса, за якою проводиться звернення до пам'яті, а в регістрі MDR - дані, які повинні бути записані в пам'ять або прочитані з пам'яті за цією адресою.

Процесор з усіма його елементами зазвичай реалізується у вигляді однієї мікросхеми, на якій розташовується як мінімум один пристрій кеш-пам'яті. Такі чіпи (мікросхеми) називаються HBIC, або VLSI (Very Large Scale Integration).

Структура шини

Комп'ютер зможе працювати з достатньою швидкістю лише за умови, коли слово даних пересилається між пристроями паралельно, тобто, коли одночасно пересилаються все його біти. Кожен біт пересилається за своїм проводом (лінією), так що для пересилання слова потрібно декілька паралельних ліній. Група ліній, що утворюють з'єднання між кількома пристроями, називається шиною (BUS). Поряд з лініями, за якими пересилаються дані, шина містить лінії для передачі адреси і керуючих сигналів.

Для з'єднання декількох функціональних пристроїв комп'ютера найпростіше використовувати загальну шину.

До цієї шині приєднуються всі пристрої комп'ютера. Оскільки за один раз по шині може пересилатися тільки одне слово даних, в кожен конкретний момент шину можуть використовувати тільки два пристрої. Головною перевагою архітектури із загальною шиною є її низька вартість і гнучкість по відношенню до підключення периферійних пристроїв. При наявності в системі декількох шин можливо одночасне виконання декількох операцій пересилання даних, завдяки чому така система працює швидше. Але і вартість вище.

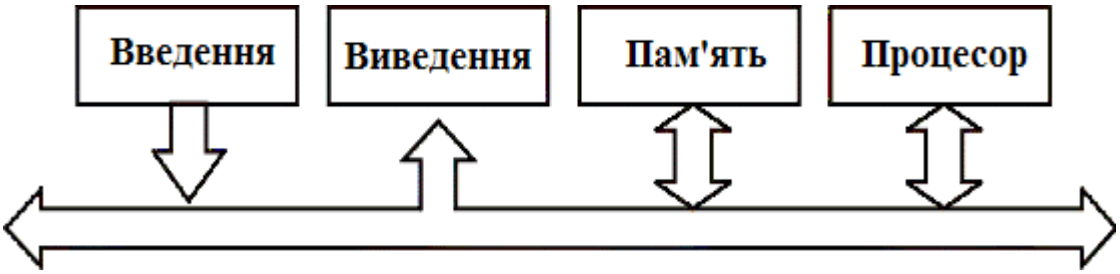


Рисунок 1.2 - Архітектура системи із загальною шиною

Приєднані до шини пристрої можуть помітно відрізнятися один від одного за швидкістю функціонування. Деякі електромеханічні пристрої (клавіатура, принтер і ін.) працюють відносно повільно. Значно вища швидкість роботи магнітних і оптичних дисків. А пам'ять і процесор функціонують зі швидкістю електронних схем і є найшвидшими частинами комп'ютера. Оскільки всі ці три типи пристроїв можуть взаємодіяти між собою через шину, необхідний такий механізм пересилання даних, який би згладжував різницю в швидкості роботи процесора, пам'яті і зовнішніх пристроїв.

Найпоширеніший підхід до вирішення цього завдання заснований на використанні буферних регістрів, які вбудовуються в зовнішні пристрої для зберігання одержуваної ними інформації.

Розглянемо, наприклад, як проходить процес передачі коду символу від процесора до принтера.

Процесор пересилає дані по шині в буфер принтера. Оскільки буфер є електронним регистром, процес пересилання відбувається дуже швидко.

Коли буфер заповнений, принтер починає друкувати, і втручання процесора більше не потрібно.

Шина і процесор звільняються для іншої роботи, яка може виконуватися одночасно з друкуванням символу, що зберігається в буфері принтера. Таким чином, використання буферних регістрів запобігає блокування високошвидкісного процесора повільними пристроями на весь час виконання операцій введення-виведення. Процесор може швидко перемикатися від одного пристрою до іншого.

Програмне забезпечення

Щоб користувач міг працювати з прикладною програмою, в пам'яті комп'ютера має міститися системне програмне забезпечення. Воно відповідає за координування всіх операцій, які виконуються в комп'ютерній системі.

Прикладні програми зазвичай пишуться на мовах програмування високого рівня, таких як C, C ++, Java і інших, що дозволяють програмісту задати дії, які повинна виконувати програма.

Програмісту, який використовує мову високого рівня, не потрібно знати машинні команди і особливості їх використання. Спеціальна системна програма, яка називається компілятором, транслює програму, написану на мові високого рівня, в програму на машинній мові. Інша важлива системна програма називається текстовим редактором. Вона призначена для введення і редагування прикладних програм.

На основі вищевикладеного можна зробити висновок, що характеристики комп'ютера визначаються особливостями структури процесора, пристроями та організацією введення-виведення, пристроями та організацією пам'яті, програмним забезпеченням, способами адресації і розподілом адресного простору.

1.2 Пристрої і процес введення-виведення

Підтримка пристроїв введення-виведення є однією з найважливіших функцій комп'ютера. Завдяки цьому, оператор може використовувати клавіатуру і дисплей для роботи з текстом і графікою. Комп'ютери застосовуються для взаємодії з іншими комп'ютерами через Інтернет, використовуються на виробництві, в навчальних аудиторіях, в різних транспортних, банківських, комерційних системах. Дані можуть надходити в

комп'ютер від різних приладів: відеокамер, мікрофонів, з пультів сигналізацій. Вихідними ж даними можуть служити закодована цифрова команда, яка змінює режим роботи двигуна, що відкриває вентиль або змушує робота виконувати певний рух або, наприклад, звуковий сигнал, що направляється в колонки.

Таким чином, ми бачимо, що комп'ютер повинен мати здатність обмінюватися інформацією з широким діапазоном пристроїв.

Доступ до пристроїв введення-виведення

Всі пристрої, підключені до шини, можуть обмінюватися між собою інформацією. Зазвичай шина складається трьох наборів ліній, призначених для передачі адрес, даних і керуючих сигналів. Кожному пристрою введення-виведення присвоюється унікальна адреса. Коли процесор видає на адресні лінії конкретну адресу, то відповідний пристрій вводу-виводу розпізнає цю адресу і відповідає на команду, вміщену(помещенную) на керуючі лінії. Процесор запитує або операцію читання, або операцію запису, і запитані дані пересилаються по лініях даних.

У більшості комп'ютерних систем пристрій введення-виведення і пам'ять мають єдиний адресний простір. Тому будь-які машинні команди, що виконують звернення до пам'яті, можуть бути задіяні і для обміну даними з пристроями введення-виведення.

Рисунок 1.3 - Архітектура системи із загальною шиною

Припустимо, що адреса вхідного буферного регістра, пов'язаного з клавіатурою - DATAIN. Тоді для зчитування даних з DATAIN і розташування їх в регістрі процесора R0 можна використовувати команду пересилання:

MOVE DATAIN, R0

Якщо ж адреса вихідного буфера дисплея або принтера-DATAOUT, то для пересилання вмісту регістра процесора R0 за адресою DATAOUT можна використовувати команду:

MOVE R0, DATAOUT.

Організація системи введення-виведення, при якій пристрій введення-виведення і пам'ять мають єдиний адресний простір, називається введенням-виведенням з відображенням в пам'ять.

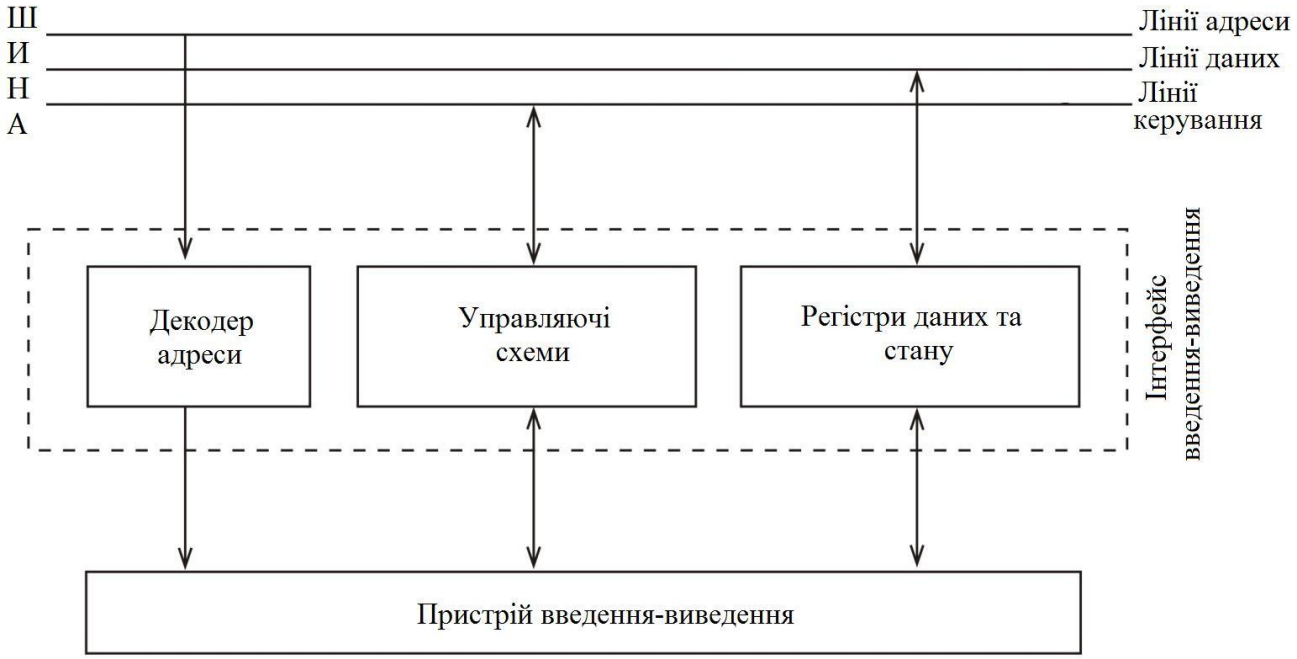


Рисунок 1.4 - Інтерфейс пристрою введення-виведення

Деякі процесори, наприклад сімейства Intel, мають спеціальні команди, призначені виключно для введення-виведення - IN і OUT, а також окремий 16-розрядний адресний простір для пристроїв введення-виведення. У цих командах задаються адреси пам'яті, розташованої в цьому окремому адресному просторі. Такі операції називаються ізольованим введенням-виведенням.

Одним з переваг наявності адресного окремого простору введення-виведення є те, що відповідні пристрої можуть використовувати менше число адресних ліній.

Створюючи комп'ютерну систему на основі процесорів типу Intel, конструктор може або з'єднати всі пристрої так, що б вони використовували спеціальний адресний простір введення-виведення, або ввести їх в єдиний адресний простір. Другий підхід дозволяє спростити програмне забезпечення, тому використовується значно частіше.

Апаратні елементи, необхідні для приєднання пристроїв введення-виведення до шини, представлені на рисуноку 1.4

Коли адреса пристрою з'являється на адресних лініях, пристрій може її розпізнати за допомогою дешифратора (декодера) адреси. Дані, якими пристрій обмінюється з процесором, зберігаються в регістрах даних. Регістр стану містить інформацію, що відноситься до функціонування пристрою введення-виведення. Регістри даних і стану з'єднуються шиною даних і їм присвоюються унікальні адреси. Дешифратор адреси, регістри даних і стану, керуючі схеми, необхідні для координації операцій введення-виведення, складають схему сполучення, або інтерфейс пристрою.

1.3 Принципи організації систем переривання програм. Призначення систем переривання програм

Під час виконання ЕОМ поточної програми всередині машини і в пов'язаному з нею зовнішньому середовищі (технологічний процес) можуть виникати події, що потребують негайної реакції на них з боку машини (готовність ОП до введення або виведення даних, поява помилок при введенні - виведення, аварійна ситуація) .

Реакція полягає в тому, що машина перериває обробку поточної програми і переходить до виконання іншої програми, спеціально призначеної для даної події. По завершенні цієї програми ЕОМ повертається до виконання перерваної програми.

Цей процес, званий «перериванням програм», пояснюється рисунком.

Моменти виникнення подій, які потребують переривання програм, заздалегідь невідомі і тому не можуть враховуватись при програмуванні.

Кожна подія, яка потребує переривання, супроводжується сигналом, що сповіщає ЕОМ і називається запитом переривання.

Програму, викликану запитом переривання, називають перериваючою програмою. Програма, яка переривається – це програма, що виконувалася машиною до появи запиту.

До запитів на переривання, що виникають всередині самої ЕОМ, відносяться запити при виникненні в ЕОМ таких подій, як збої в апаратурі, переповнення розрядної сітки, спроби поділу на «0», затребування

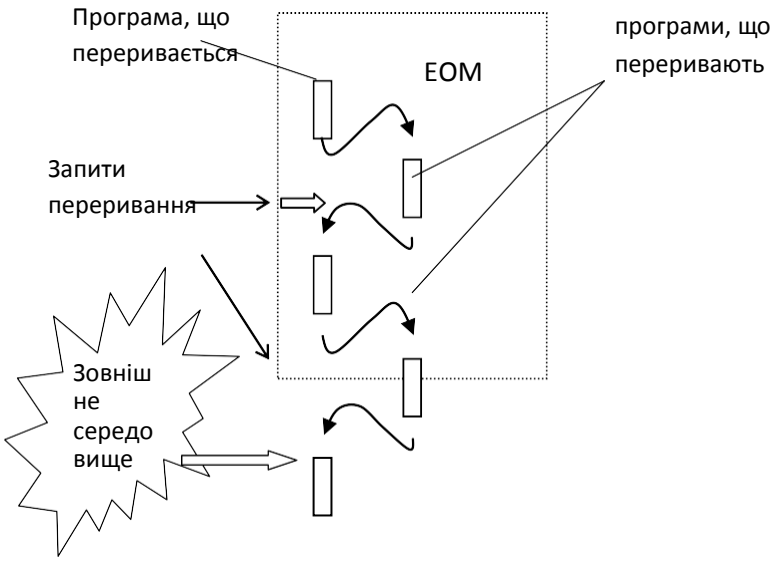


Рисунок 1.5 – Ілюстрація процесу переривання поточної програми

периферійним пристроєм операції введення-виведення та ін. Моменти їх появи неможливо передбачити.

Запити в зовнішньому середовищі можуть виникати від інших ЕОМ, від аварійних і деяких інших датчиків технологічного процесу і т.п.

Сукупність апаратних і програмних засобів, що реалізують з високою швидкодією переривання програм, називається системою переривання програми або контролером переривань.

Основними функціями системи переривання є:

- запам'ятовування стану програми, що переривається, і здійснення переходу до перериваючої програми;

- відновлення стану перерваної програми і повернення до неї.

Стан програми характеризується словом стану програми (або процесора), або, що те ж саме, вектором стану.

Слово стану програми, або вектор стану, в кожен момент часу повинен містити інформацію, достатню для продовження виконання програми з точки, відповідної моменту формування даного вектора стану. Вектор стану формується у відповідних регістрах процесора, зазнаючи зміни після виконання кожної команди.

Набори інформаційних елементів, що утворюють вектори стану, відрізняються у ЕОМ різних типів. Найбільш просто вектор стану виглядає у мікропроцесорів малої і середньої потужності. Як правило, він включає в себе вміст лічильника команд (адреса чергової команди), вміст регістра ознак, вміст акумулятора і інші елементи.

Більш складні мікропроцесори містять більш складні структури вектора стану.

При наявності декількох джерел запитів переривання між запитами (і відповідними перериваючими програмами) повинні встановлюватись пріоритетні співвідношення, що визначають, який з кількох запитів, що надійшли, підлягає обробці в першу чергу. Крім того, пріоритетні співвідношення встановлюють, має право або не має даний запит переривати ту чи іншу програму.

Характеристики систем переривання

Для оцінки ефективності систем переривання можуть використовуватись такі характеристики:

- загальна кількість запитів переривання (входів в систему переривання)
- час реакції - час між появою запиту переривання і початком виконання програми, що перериває.

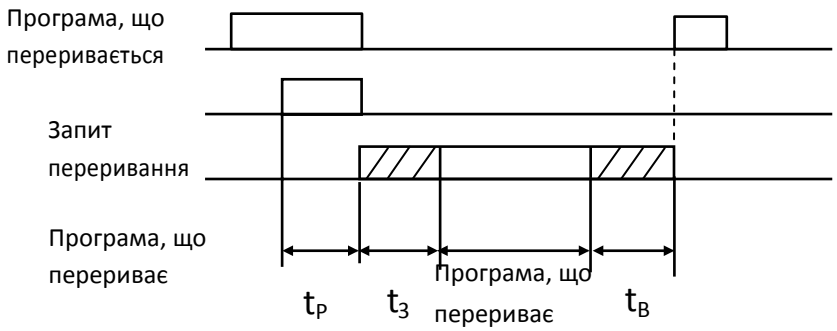


Рисунок 1.6 - Часова діаграма процесу переривання

$t_З$- запам'ятовування стану перерваної програми;

$t_В$- відновлення стану перерваної програми;

$t_Р$- час реакції.

Спрощена часова діаграма процесу переривання зображена на малюнку 1.6.

- $t_{ВИТ}$ - витрати часу на перемикання програм (витрати переривання), які дорівнюють сумарним витратам часу на запам'ятовування і відновлення стану програми

$$t_{ВИТ} = t_З + t_В$$

- глибина переривання - максимальне число програм, які можуть переривати одна - одну.

Якщо після переходу до програми, що перериває і аж до її закінчення прийом інших запитів забороняється, то кажуть, що система має глибину переривання, рівну одиниці.

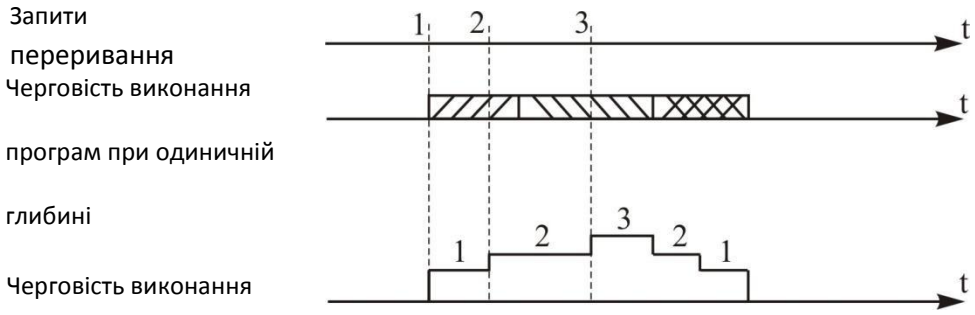


Рисунок 1.7 - Черговість виконання програм при різній глибині переривання

Глибина дорівнює n, якщо допускається послідовне переривання до n програм.

Глибина переривання зазвичай збігається з числом рівнів пріоритету в системі переривання. На рисунку 1.7 передбачається, що пріоритет кожного наступного запиту вищий від попереднього. Системи з великим значенням глибини переривання забезпечують більш швидку реакцію на термінові запити.

- насичення системи переривання.

Якщо запит виявиться не оброблений до моменту приходу нового запиту від того ж джерела, то це - насичення системи переривання. Старий запит втрачається, що неприпустимо. Щоб цього не було, треба узгоджувати швидкодію ЕОМ, характеристики системи переривання, число джерел переривання, частоту виникнення запитів.

- Допустимі моменти переривання програм.

Найчастіше переривання допускаються після закінчення будь-якої поточної команди. Однак в ЕОМ, які працюють в реальному масштабі часу, допускаються переривання після будь-якого такту виконання команди. Однак при цьому зростає кількість інформації, що підлягає запам'ятовуванню і відновленню при перемиканні.

- Число рівнів переривання.

У ЕОМ число різних запитів (причин) переривання може досягати декількох десятків або сотень. У таких випадках часто запити поділяють на окремі рівні або класи.

1.4 Прямий доступ до пам'яті

При програмно-керованому введенні-виведенні пересилання даних між процесором і пристроями введення-виведення здійснюється командами типу MOVE DATAIN, R0. Ця команда виконується лише після того, як процесор визначить, що пристрій введення-виведення готовий до чергової операції. Для цього процесор опитує прапор стану в інтерфейсі пристрою.

Введення-виведення з керуванням по перериваннях - це великий крок вперед в порівнянні з програмованим введенням-виведенням, але все ж він далеко не досконалий. Справа в тому, що переривання потрібно для кожного

символу, що передається. Отже, потрібно якимось чином позбутися від більшості переривань.

Рішення лежить в поверненні до програмованого введення-виведення. Але роботу по введенню-виведенню повинен здійснювати не процесор, а інший пристрій. Тому комп'ютер може містити спеціальний керуючий пристрій, що дозволяє пересилати блоки даних між зовнішнім пристроєм і основною пам'яттю без постійної участі процесора. Ця технологія називається прямим доступом до пам'яті (ПДП), англійською - Direct Memory Access (DMA).

Операції ПДП виконуються керуючої схемою, що входить до складу інтерфейсу пристрою введення-виведення. Ця схема називається контролером прямого доступу до пам'яті (КПДП). Вона виконує ту ж задачу, що і процесор, який звертається до основної пам'яті.

Мікросхема ПДП, як показано на рисунку 1.8, має, щонайменш, 4 регістра. Хоча контролер ПДП працює без участі процесора, він управляється програмою, яка виконується процесором.

У перший регістр завантажується початкова адреса блоку пам'яті, який потрібно зчитати або записати.

Другий регістр містить число, яке показує кількість байтів або слів, які передаються.

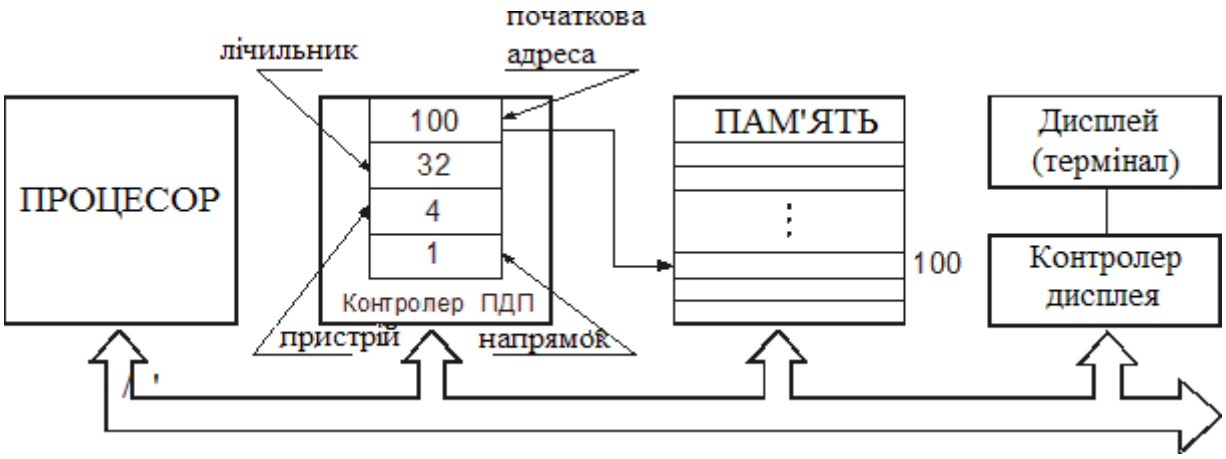


Рисунок 1.8 - Комп'ютерна система з контролером ПДП

Третій регістр містить номер або адресу пристрою введення-виведення, визначаючи таким чином, який саме пристрій нам потрібен.

Четвертий регістр повідомляє, чи повинні дані зчитуватися з пристрою або записуватися в нього, тобто напрямок передачі.

Щоб записати блок з 32 байтів із адреси пам'яті 100 на термінал (наприклад, пристрій 4), центральний процесор записує числа 100, 32 і 4 в перші три регістра КПДП і код запису (наприклад, 1) в четвертий регістр, як показано на рисунку 1.8 . Таким чином, відбувається ініціалізація контролера ПДП.

Потім контролер робить запит на доступ до шини, щоб зчитати байт з комірки пам'яті за адресою 100 точно так же, якби центральний процесор сам зчитував цей байт. Отримавши потрібний байт, контролер ПДП посилає пристрою 4 запит на введення-виведення, щоб записати на нього байт.

Після завершення цих двох операцій контролер ПДП збільшує значення регістра адреси на 1 і зменшує значення регістра лічильника байтів на 1. До тих пір, поки значення регістра лічильника буде більше "0", то наступний байт зчитується з пам'яті і записується на пристрій вводу-виводу. Коли значення лічильника доходить до "0", контролер ПДП зупиняє передачу даних та інформує про це процесор з допомогою сигналу переривання.

Поки контролер ПДП продовжує пересилання даних, программа, що запросила таку операцію, не може продовжувати свою роботу, і процесор часто використовується для виконання іншої програми. Після закінчення пересилання, за сигналом переривання від КПДП, процесор може повернутися до вихідної програми.

2 ПРИСТРОЇ ПАМ'ЯТІ

Пристрій, що запам'ятовує - носій інформації, призначений для запису і зберігання даних. В основі роботи запам'ятовуючого пристрою, може лежати будь-який фізичний ефект, що забезпечує приведення системи до двох або більше стійких станів.

2.1 Класифікація запам'ятовуючих пристроїв

За стійкістю запису і можливістю перезапису ЗП поділяються на:

- постійні ЗП (ПЗП), зміст яких не може змінюватись кінцевим користувачем (наприклад, DVD-ROM). ПЗП в робочому режимі допускає тільки зчитування інформації.
- записуючі ЗП, в які кінцевий користувач може записати інформацію тільки один раз (наприклад, DVD-R).
- багаторазово перезаписуючі ЗП (наприклад, DVD-RW).
- оперативні ЗП (ОЗП) забезпечує режим запису, зберігання і зчитування інформації в процесі її обробки.

За типом доступу ЗП поділяються на:

- пристрої з послідовним доступом (наприклад, магнітні стрічки);
- пристрої з довільним доступом (RAM) (наприклад, оперативна пам'ять);
- пристрої з прямим доступом (наприклад, жорсткі магнітні диски);
- пристрої з асоціативним доступом (спеціальні пристрої, для підвищення продуктивності БД).

За геометричним виконанням:

- дискові (магнітні диски, оптичні, магнітооптичні);
- стрічкові (магнітні стрічки, перфострічки);
- барабанні (магнітні барабани);
- карткові (магнітні картки, перфокарти, флеш-карти, й ін.)
- дрекovані плати (карти DRAM).

За фізичним принципом:

- перфораційні (перфокарта; перфострічки);
- з магнітним записом (феритові осердя, магнітні диски, магнітні стрічки, магнітні карти);

- оптичні (CD, DVD, HD-DVD, Blu-ray Disc);

- які використовують ефекти в напівпровідниках (флеш-пам'ять) та інші.

За формою записаної інформації виділяють аналогові і цифрові запам'ятовуючі пристрої.

2.2 Типи і характеристика пристроїв пам'яті

Найважливішими характеристиками окремих пристроїв пам'яті є ємність пам'яті та її швидкодія. Ємність пам'яті визначається кількістю даних, які можуть в ній зберігатися. Ємність вимірюється в бітах і в машинних словах (півслова - один байт, слово - два байта, подвійне, четверне слово і так далі). При цьому ємність пам'яті визначають через число К = 210 = 1024 (кілобіт, кілобайт). 1024 Кбайта = 1 Мбайт; 1024 Мбайт = 1 Гбайт.

Швидкодія пам'яті визначається тривалістю операції звернення, тобто часом, що витрачається на пошук одиниці інформації в пам'яті та її зчитування - $t_{обр.\ счит.}$, або часом на пошук місця в пам'яті для зберігання даної одиниці інформації і на її запис в пам'ять - $t_{обр.\ зап}$

У деяких пристроях пам'яті зчитування інформації викликає її руйнування. В такому випадку цикл звернення повинен містити опрацію відновлення прочитаної інформації на колишньому місці в пам'яті.

Тривалість циклу звернення до пам'яті вимірюється величиною

$$t_{обр.} = \max\left\{ t_{обр.счит}; t_{обр.зап.} \right\}.$$

Розрізняють пам'ять з довільним зверненням (можливі як зчитування, так і запис даних) і пам'ять тільки для зчитування інформації (постійну).

За способом організації доступу розрізняють пристрої пам'яті з довільним, прямим (циклічним) і послідовним доступом.

У пам'яті з довільним (безпосереднім) доступом час доступу не залежить від місця розташування елемента пам'яті. Цикл звернення в таких системах зазвичай становить від сотень до одиниць нс. Число розрядів, які зчитуються або записуються в пам'яті з безпосереднім доступом паралельно в часі за одну операцію звернення, називається шириною вибірки.

У пристроях пам'яті з прямим доступом - накопичувачах на гнучкому (НГМД) і на жорсткому магнітному диску (НМД) використовуються компакт-диски. Завдяки безперервному обертанню носія інформації можливість звернення до деякої ділянки носія для зчитування або запису циклічно повторюється. Час доступу - від часток секунди до мілісекунд.

У пам'яті з послідовним доступом виконується послідовний перегляд ділянок носія інформації, поки потрібна ділянка не займе деяке потрібне положення. Тому час доступу носія найбільший.

Ієрархічна структура пам'яті

Ієрархічна структура пам'яті є традиційним розв'язанням проблеми зберігання великих обсягів даних (рисунок 2.1). На самому верху ієрархії знаходяться регістри процесора.

Рисунок 2.1 П'ятирівнева організація пам'яті

Доступ до регістрів здійснюється найшвидше. Далі йде кеш-пам'ять, обсяг якої зараз становить від 32 Кбайт до декількох мегабайтів. Потім слідує основна пам'ять, яка в даний час може вміщати від 16 Мбайт до десятків гігабайтів. Потім йдуть магнітні диски і, нарешті, накопичувачі на магнітній стрічці і оптичні диски, які використовуються для зберігання архівів.

У міру просування зверху вниз по ієрархії змінюються три параметри. По-перше, збільшується час доступу. Доступ до регістрів займає кілька наносекунд, доступ до кеш-пам'яті - трохи більше, доступ до основної пам'яті - кілька десятків наносекунд. Далі йде великий розрив: доступ до дисків займає принаймні 10 мкс, а час доступу до магнітних стрічок і оптичних дисків взагалі може вимірюватися в секундах (оскільки ці накопичувачі інформації ще потрібно помістити у відповідний пристрій).

По-друге, зростає обсяг пам'яті. Регістри можуть містити в кращому випадку 128 байт, кеш-пам'ять - кілька мегабайтів, основна пам'ять - десятки тисяч мегабайтів, магнітні диски - від декількох одиниць до декількох десятків гігабайтів. Магнітні стрічки і оптичні диски зберігаються автономно від комп'ютера, тому їх сукупний обсяг обмежується тільки фінансовими можливостями власника.

По-третє, збільшується кількість бітів, яке ви отримуєте за 1 долар. Вартість обсягу основної пам'яті становить кілька доларів за мегабайт.

Пристрої памяті з довільним та послідовним доступом

2.2.1 Принцип побудови запам'ятовуючих пристроїв з довільним доступом

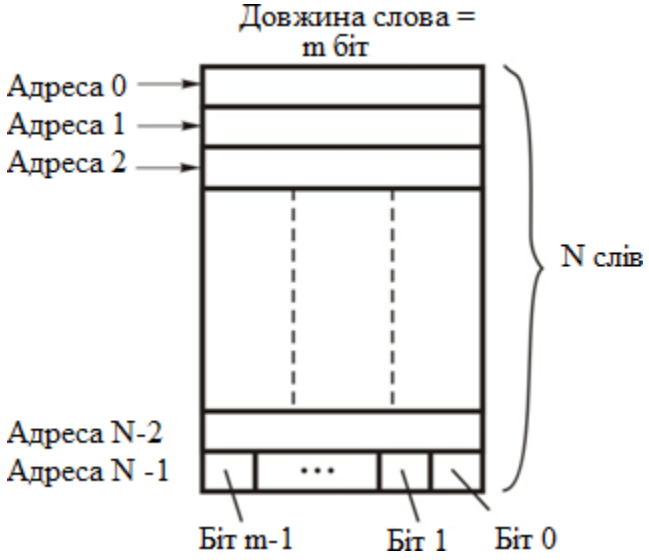


Рисунок 2.2 - Організація основної пам’яті ЕОМ

Швидкодіюча - основна пам'ять ЕОМ, як оперативна - ОЗП, так і постійна - ПЗП організована у вигляді слів фіксованої довжини і представлена на рисунку 2.2. Ця пам'ять поділяється на N слів, N - зазвичай деякий ступінь 2, а кожному слову привласнюється адреса в пам'яті. Кожне слово містить однакове число бітів, зване довжиною слова. Адреса пам'яті - послідовно зростаючі числа, починаючи з 0 і закінчуючи найбільшою адресою. За цими адресами ЕОМ зчитує

слово з будь-якої комірки пам'яті (ПЗП) або зчитує і записує слово в будь-яку комірку пам'яті (ОЗП).

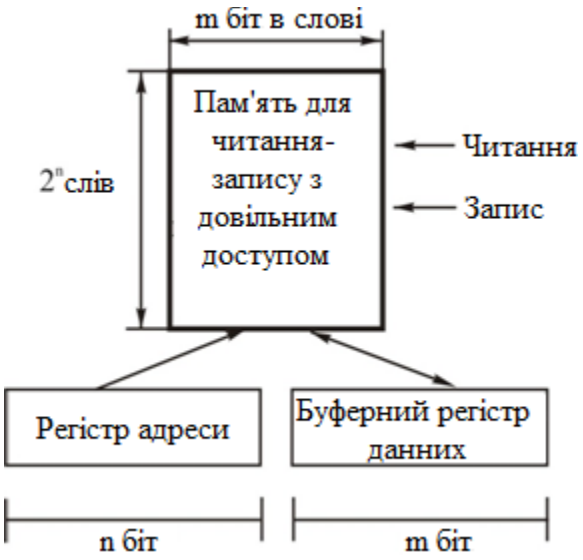


Рисунок 2.3 – Блок-схема оперативної пам’яті

На рис. 2.3 зображена блок-схема оперативної пам'яті. При запису ЕОМ поміщає адресу комірки, в яку повинні бути записані дані, в регістр адреси. Цей регістр складається з n тригерів, тому адресна область становить 2n елементів пам'яті. Дані, що підлягають запису в пам'ять, поміщаються в буферний регістр даних, розрядність якого відповідає розрядності пам'яті. Для здійснення запису пам'ять отримує команду «Писати» у вигляді сигналу 1 на лінію ЗАПИС.

Після цього пам'ять запам'ятовує вміст буферного регістра даних в комірці, адреса якої вказана в регістрі адреси. Щоб зчитати слово, адресу комірки поміщають в регістр адреси. Потім на лінію «Читання» подається команда «Читати» у вигляді сигналу «1». Пам'ять передає вміст цієї комірки в буферний регістр даних.

Сукупність певним чином з'єднаних елементів пам'яті ЕП утворює запам'ятовуючу матрицю або запам'ятовуючий масив, де кожен ЕП зберігає біт інформації. Запам'ятовуючий масив має систему адресних і розрядних ліній (провідників). Адресні лінії використовуються для виділення за адресою сукупності запам'ятовуючих елементів, яким встановлюється режим зчитування або запису. Виділення окремих розрядів здійснюється розрядними лініями, за якими передається в ЕП інформація, яка записується в нього або прочитується з нього.

Адресні і розрядні лінії носять спільну назву ліній вибірки. Залежно від числа таких ліній, з'єднаних з одним ЕП, розрізняють ЗП типу 2D з одновимірною (лінійною) адресацією, але двовимірною вибіркою, ЗП типу 3D з двовимірною адресацією, але тривимірною вибіркою. (D - перша буква

англійського слова dimension - розмірність). Є також ряд модифікацій, наприклад, 2,5 D.

2.2.2 Організація пам'яті з лінійною адресацією

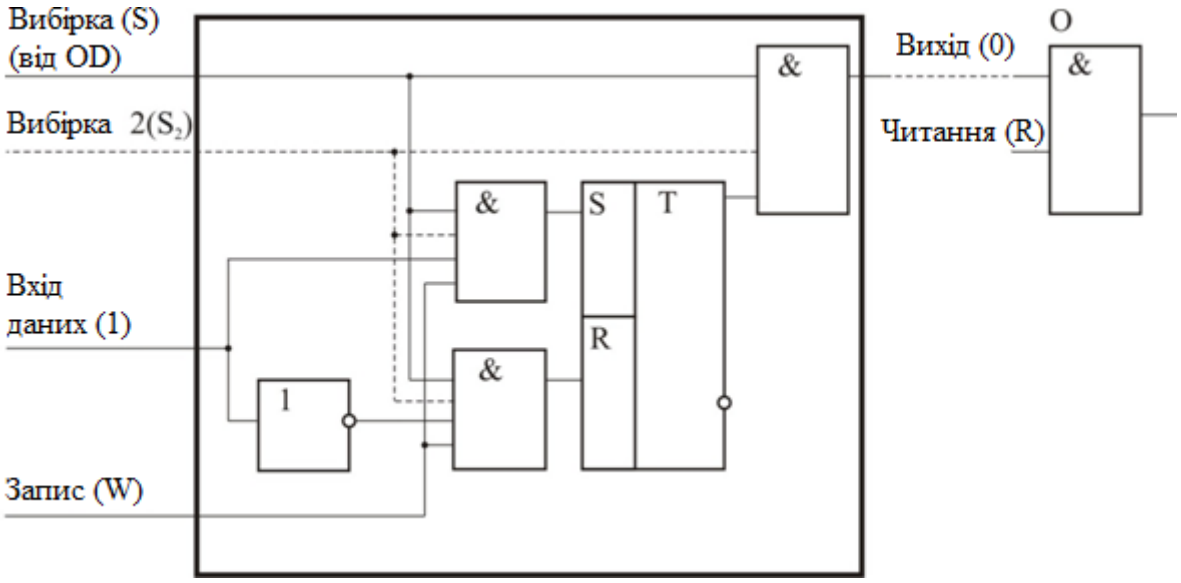


Рисунок 2.4 – Варіант базової комірки пам'яті

Моделі пам'яті, які ми будемо розглядати, будуть в деякій мірі ідеалізовані. У будь-якій пам'яті обов'язково повинна бути базова комірка пам'яті. Виберемо її відповідно до рис.2.4. Тоді при подачі сигналів "Вибірка (S) = 1" і "Запис (W) = 1" біт вхідних даних записується в тригер T, що є елементом базової комірки пам'яті. Зчитування інформації здійснюється з вихідного елемента O (Out) при подачі сигналів "Вибірка (S) = 1" і "Читання (R) = 1".

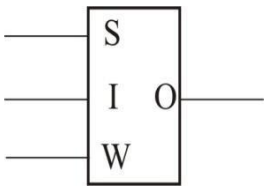


Рисунок 2.5 – Умовне позначення базової комірки

На рис.2.4 вхід, зазначений пунктиром, в даному розділі не враховувати. Тому покажемо базову комірку у вигляді, представленому на рис.2.5.

На основі такої базової комірки пам'яті організуємо пам'ять з лінійною вибіркою. На рис.2.6 зображена чотири адресна пам'ять з трирозрядним словом.

У будь-який заданий момент часу регістр адреси вибирає комірку пам'яті. Якщо на лінію «Читання» подається одиничний потенціал, то вміст трьох комірок обраного слова зчитується з вихідної лінії Out - O1, O2, O3, O4. Якщо на лінію «Запис» подається одиничний потенціал, то в пам'ять будуть записані дані значення з ліній I1, I2, I3.

Кон'юнктори (рис.2.3), підключені до ліній «ВИХІД» комірок пам'яті, повинні мати здатність зберігати на виході високий потенціал, коли кілька вихідних ліній схем «І» з'єднані разом. Тоді, якщо хоча б на одному виході є одиничний потенціал, на лінії буде «1», в іншому випадку - «0».

Таке з'єднання називають монтажним «АБО». Така пам'ять буде зберігати дані протягом будь-якого періоду часу (при наявності напруги живлення) та може виконувати операції зі швидкістю, яку допускають логіка і тригери. Проблема полягає в складності пам'яті. Базова комірка складна, а при великій ємності пам'яті потрібен великий дешифратор.

У регістр адреси записується в двійковому коді номер комірки, до якої необхідно звернутися. Для кожного коду, що надходить в регістр, буде обрана вихідна лінія дешифратора з одиничним потенціалом. На інших лініях деши фратора буде «0», і не будуть вибиратися кон'юнктори (схеми «І»), що знаходяться на входах і виходах тригерів (див. Рис. 2.4) в цих горизонтальних рядах (рядках). Кожен рядок з трьох комірок пам'яті становить трирозрядне слово.

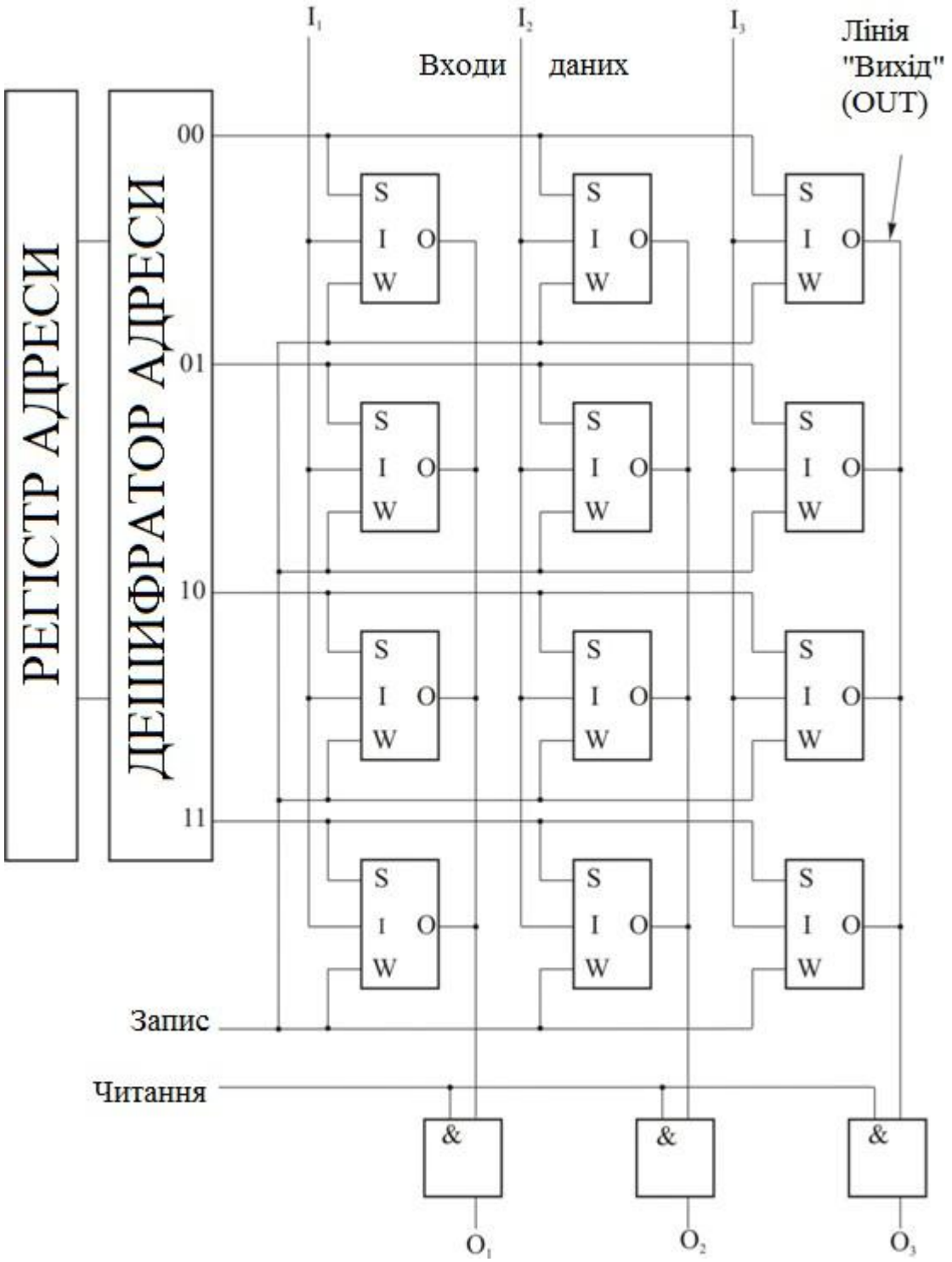


Рисунок 2.6 – Організація пам’яті з одновимірною адресацією

2.3 Організація пам'яті з двовимірною адресацією

В якості базової виберемо комірку пам'яті (рис.2.4), до якої доданий ще один вхід «Вибірка 2» (показаний пунктиром). Тепер для вибору тригера на обидва входи «Вибірка» і «Вибірка 2» повинні бути подані «1». Для спрощення структури дешифратора він розбивається на дві частини - дешифратор X і дешифратор Y. Тоді загальна кількість елементів в обох дешифраторах буде у багато разів менша, ніж для єдиного дешифратора при лінійної адресації. Структурна схема ОЗП з двовимірною адресацією зображена на рис.2.7.

Дешифратор X іноді називають дешифратором рядка, дешифратор Y-стовпчика.

Модернізована базова комірка матиме вигляд, представлений на рис.2.8.

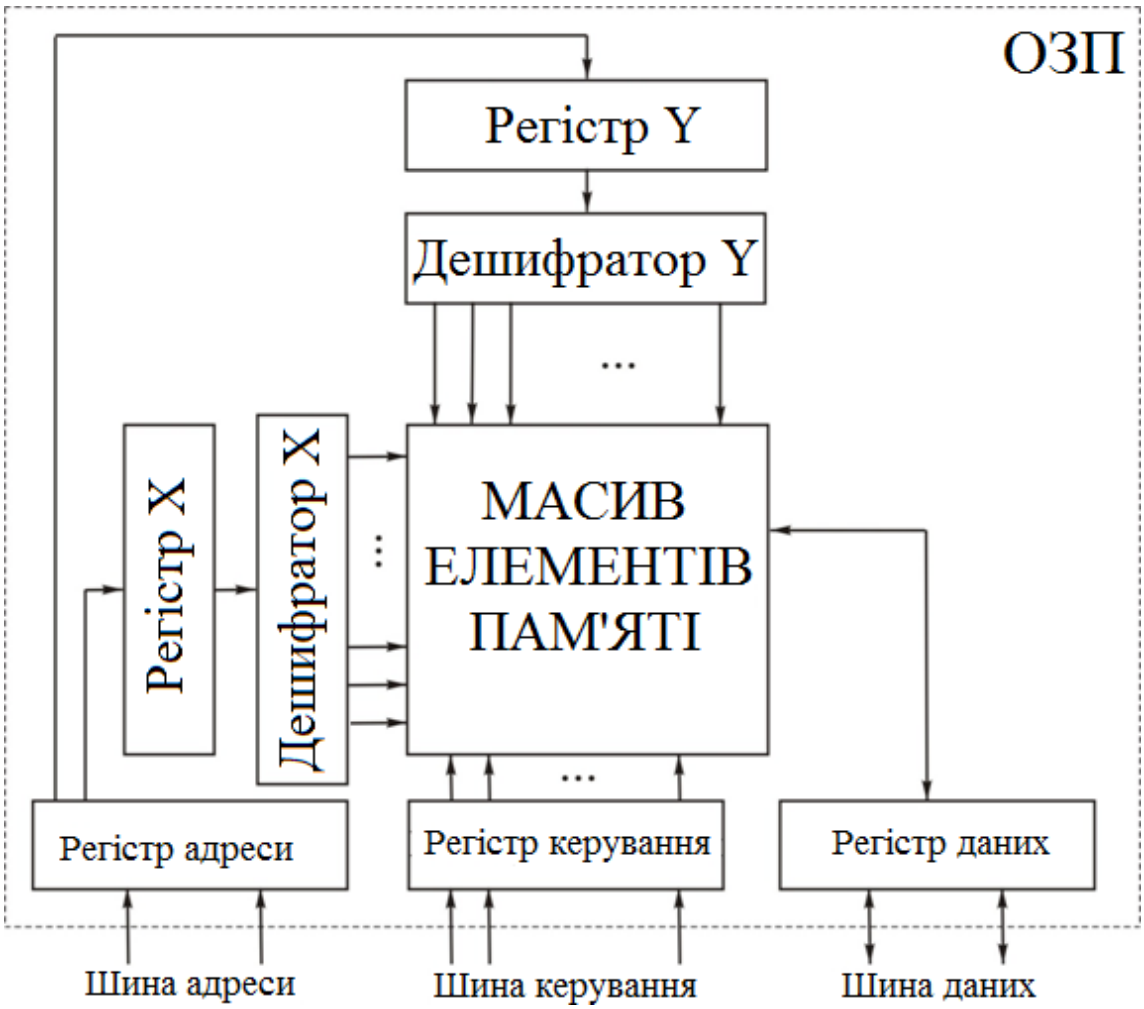


Рисунок 2.7 – Структурна схема ОЗП з двовимірною адресацією

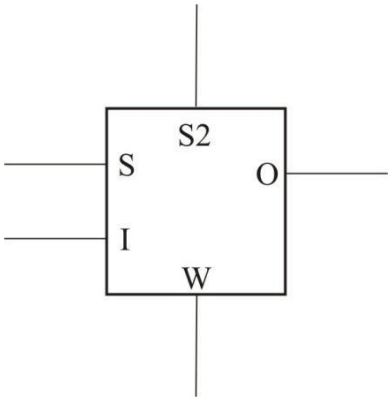


Рисунок 2.8 - Умовне позначення модернізованої базової комірки

Нехай масив елементів пам'яті ЕП - 16 однобітових слів (див. рис.2.9). Тоді адресація повинна здійснюватися за допомогою чотирирозрядного регістра, що приймає 16 станів. Після розбивки регістра на дві частини регістр X і регістр Y матимуть по 2 розряди, а дешифратори (DC) X і Y будуть дешифрувати по 4 стани кожного регістру.

Всі виходи W комірок повинні бути приєднані до шини «ЗАПИС». Всі входи I комірок повинні бути приєднані до шини «Вхід» (шина показана пунктиром).

Нехай в регістрі адреси записана адреса 0111.

Це означає, що код 01 записаний в регістрі X, а код 11 - в регістрі Y. Це призводить до вибору другого рядка в DC «X» і крайнього правого стовпця в DC «Y». В результаті тільки у комірки (тригера) на перетині другого рядка і крайнього правого стовпця будуть активізовані обидві її лінії «Вибірка» і логіка «І» (див. Рис 2.4). В результаті тільки ця комірка буде обрана, і тільки в цей тригер можна зробити запис або здійснити зчитування з нього.

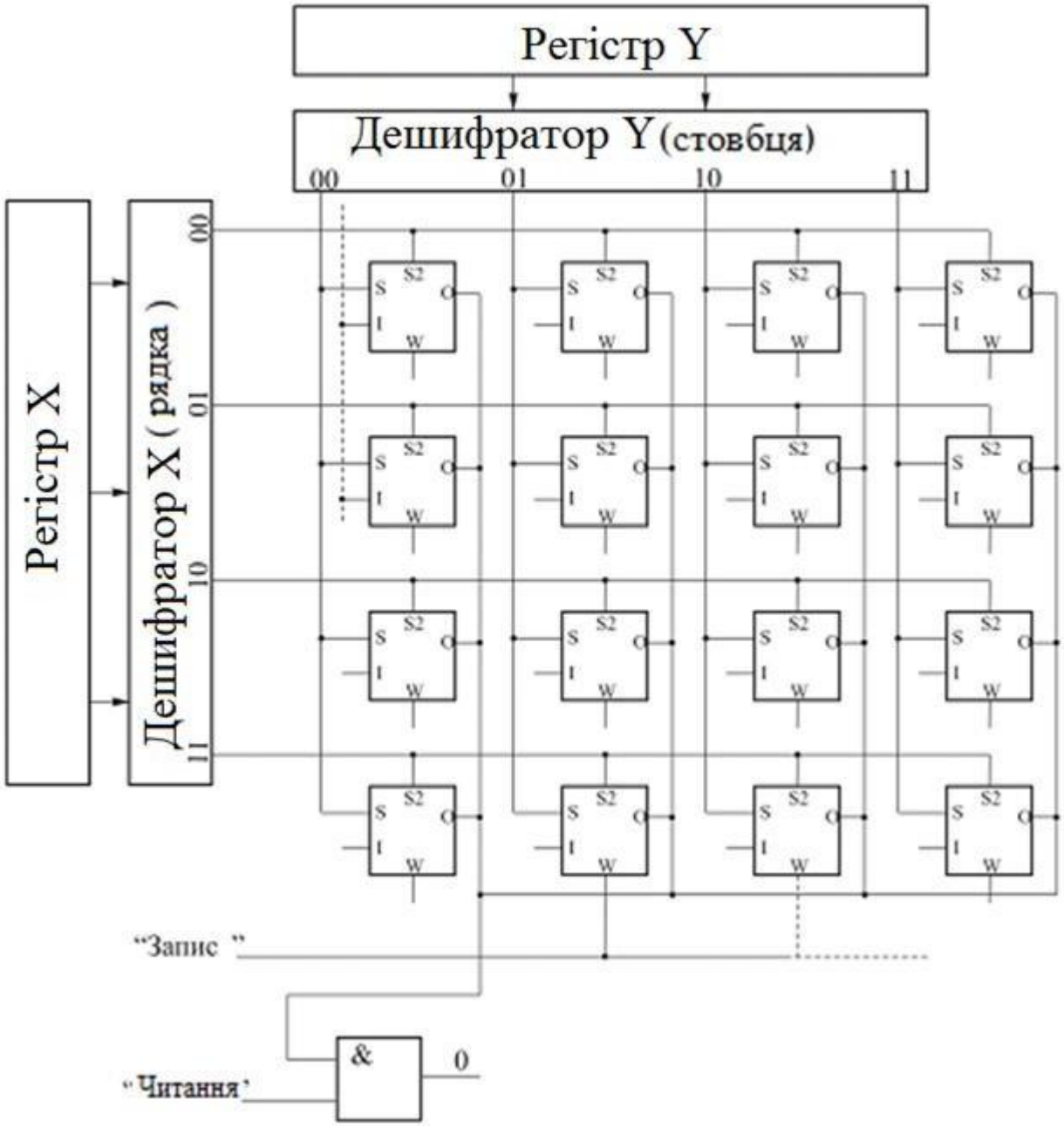


Рисунок 2.9 – Організація пам’яті з двовимірною адресацією

2.4 Статична оперативна пам'ять на біполярних- і МОП-транзисторах

Є кілька основних видів пам'яті, які виконуються на інтегральних схемах:

- біполярні запам'ятовуючі пристрої (ЗП). Це швидкодіючі, але досить дорогі ЗП з тригерами, виготовленими з використанням стандартних транзисторів з p-n переходами;

- статичні ЗП на МОП-транзисторах. У тригерних схемах цих пристроїв використовуються польові МОП-транзистори. Вони мають меншу швидкодію, ніж біполярні, але дешевше, витрачають менше енергії і мають високу щільність упаковки;

- ЗП на компліментарних МОП-транзисторах (КМОП). У КМОП-транзисторах використовуються p- і n-канальні транзистори на одній і тій же підкладці. Тому технологія виготовлення ускладнюється. Швидкодія їх вище, ніж у p- або n-канальних МОП-транзисторів, але і вартість вище;

- ЗП на сапфіро-силіконових елементах. Ці ЗП подібні КМОП, виконані на ізолюючої підкладці з сапфіра. Це зменшує ємність пристроїв (щільність упаковки) і збільшує швидкодію. Ця память - найдорожча;

- ЗП на інтегральної инжекційній логіці (И2Л). На схемах И2Л відсутні резистори навантаження джерела струму, наявні в схемах ТТЛ-типу. Це скорочує витрати електроенергії і збільшує щільність упаковки в порівнянні з біполярними ЗП. У них поєднується швидкодія біполярних ЗП з щільністю упаковки ЗП на МОП транзисторах. Це - ЗП середньої вартості.

2.5 Динамічні запам'ятовуючі пристрої

В якості основи для динамічної системи пам'яті зазвичай використовуються МОП - комірки. У динамічних ЗП двійкові коди зберігаються на «запам'ятовуючих ємностях», в якості яких використовуються міжелектродні ємності МОП-транзисторів. Відсутність так званого об'ємного заряду на запам'ятовуючій ємності означає стан «0», а наявність - «1». В такому випадку інформація, що зчитується полягає у визначенні, заряджені чи ні «запам'ятовучі ємності».

Запам'ятовуюча ємність може невизначено довго зберігати стан «0» (заряд відсутній). Стан «1» зберігається обмежений час через витік заряду.

Тому в розглянутих ЗП необхідно періодично, через кожні (8 - 64) мілісекунд, проводити відновлення інформації, що зберігається. Операція динамічного відновлення інформації називається регенерацією або рефрешем.

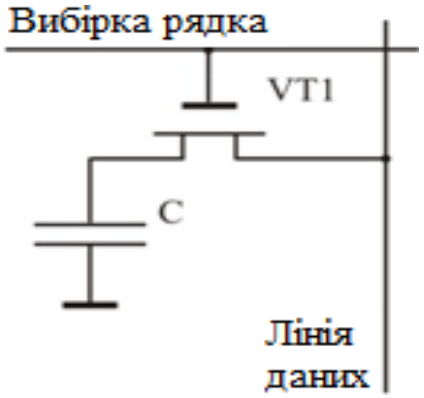


Рисунок 2.10 - Спрощена схема динамічного елемента пам'яті

Запам'ятовуючою ємністю є міжелектродна ємність С затвора МОП - транзистора.

На рис 2.10 зображена найпростіша запам'ятовуюча комірка з одним перемикаючим МОП - транзистором і конденсатором С, який виконує функцію запам'ятовуючого елемента.

Лінія «Вибірка рядка» виконує ті ж функції, що і в інших ЗП - вибір комірки здійснюється шляхом подачі на цю лінію високого потенціалу. Лінія даних використовується для зчитування з комірки. Комірки компонуються в двовимірний масив з підсилювачами зчитування, підключеними до кожної лінії даних.

Коли лінія «Вибірка» приводить рядок в дозволений стан, усі транзистори в цьому рядку переходять в провідний стан. Відкриті транзистори переносять будь-який заряд з конденсатора С на лінію даних, здійснюючи зчитування з руйнуванням. Підсилювач стовпчика підсилює зафіксований рівень до логічної «1» (або видає «0», якщо немає заряду).

Для запису інформації в комірку на лінію даних подається висока (1) або низька (0) напруга, а потім підвищується напруга на лінії "Вибірка рядка". При цьому конденсатор (міжелектродна ємність) отримує одиничний або нульовий заряд.

Незважаючи на необхідність регенерації, ці схеми дуже широко використовуються в пристроях пам'яті завдяки простоті, дешевизні, високій щільності упаковки.

2.6 Оперативний запам'ятовуючий пристрій

Оперативна пам'ять (також оперативний запам'ятовуючий пристрій, ОЗП) - призначена для тимчасового зберігання даних і команд, необхідних процесору для виконання ним операцій (рисунок 2.11). Оперативна пам'ять передає процесору дані безпосередньо, або через кеш-пам'ять. Кожна комірка оперативної пам'яті має свою індивідуальну адресу.

ОЗП може виготовлятися як окремий блок або входити в конструкцію однокристальної ЕОМ або мікроконтролера.

Рисунок 2.11 – Зовнішній вигляд оперативної пам’яті

На сьогодні найбільшого поширення набули два види ОЗП: SRAM (Static RAM) і DRAM (Dynamic RAM).

SRAM - ОЗП, зібраний на тригерах, називається статичною пам'яттю з довільним доступом або просто статичною пам'яттю. Перевагою цього виду пам'яті є швидкість. Оскільки тригери зібрані на вентилях, а час затримки вентиля дуже мала, то і перемикання стану тригера відбувається дуже швидко. Даний вид пам'яті не позбавлений недоліків. По-перше, група транзисторів, що входять до складу тригера, обходиться дорожче, навіть якщо вони витравляються мільйонами на одній кремнієвій підложці. Крім того, група

транзисторів займає набагато більше місця, оскільки між транзисторами, які утворюють тригер, повинні бути витравлені лінії зв'язку.

DRAM - більш економічний вид пам'яті. Для зберігання розряду (біта або тріта) використовується схема, що складається з одного конденсатора і одного транзистора (в деяких варіаціях конденсаторів два). Такий вид пам'яті вирішує, по-перше, проблему дорожнечі (один конденсатор і один транзистор дешевше декількох транзисторів) і по-друге, компактності (там, де в SRAM розміщується один тригер, тобто один біт, можна вмістити вісім конденсаторів і транзисторів). Є і свої мінуси.

По-перше, пам'ять на основі конденсаторів працює повільніше, оскільки якщо в SRAM зміна напруги на вході тригера відразу ж призводить до зміни його стану, то для того щоб встановити в одиницю один розряд (один біт) пам'яті на основі конденсатора, цей конденсатор потрібно зарядити , а для того щоб розряд встановити в нуль, відповідно, розрядити. А це набагато більш тривалі операції (в 10 і більше разів), ніж перемикання тригера, навіть якщо конденсатор має вельми невеликі розміри. Другий істотний мінус - конденсатори схильні до «стікання» заряду; простіше кажучи, з часом конденсатори розряджаються. Причому розряджаються вони тим швидше, чим менше їх ємність. У зв'язку з цією обставиною, щоб не втратити вміст пам'яті, заряд конденсаторів необхідно регенерувати через певний інтервал часу - для відновлення. Регенерація виконується шляхом зчитування заряду (через транзистор). Контролер пам'яті періодично припиняє всі операції з пам'яттю для регенерації її вмісту, що значно знижує продуктивність даного виду ОЗП. Пам'ять на конденсаторах отримала свою назву Dynamic RAM (динамічна пам'ять) якраз за те, що розряди в ній зберігаються не статично, а «стікають» динамічно в часі.

Таким чином, DRAM дешевше SRAM і її щільність вище, що дозволяє на тому самому просторі кремнієвої підложки розміщувати більше бітів, але при цьому її швидкодія нижче. SRAM, навпаки, більш швидка пам'ять, але зате і дорожче. У зв'язку з цим звичайну пам'ять будують на модулях DRAM, а SRAM використовується для побудови, наприклад, кеш-пам'яті в мікропроцесорах.

2.7 Постійні запам'ятовуючі пристрої (ПЗП)

Постійний запам'ятовуючий пристрій - дуже важлива складова частина будь-якої мікропроцесорної системи. ПЗП в робочому режимі допускає тільки зчитування інформації, що зберігається. У порівнянні з ЗП з довільним зверненням, конструкція ПЗП значно простіше, швидкодія і надійність, вище, а вартість нижча.

ПЗП призначено для зберігання постійної програмної та довідкової інформації. Дані в ПЗП заносяться при виготовленні. Інформацію, що зберігається в ПЗП, можна тільки зчитувати, але не змінювати. ПЗП широко використовуються для зберігання незмінної (або рідко змінної) інформації - системного програмного забезпечення (BIOS), таблиць - наприклад знакогенераторов графічних адаптерів, програм, призначених для вирішення певного набору завдань, для яких є відпрацьовані алгоритми. Прикладами можуть служити бортові ЕОМ літаків, ракет і космічних кораблів, а також обчислювальні комплекси, що керують технологічними процесами.

У ПЗУ знаходяться:

- програма управління роботою процесора;
- програма запуску і зупинки комп'ютера;
- програми тестування пристроїв, перевіряючі при кожному включенні комп'ютера правильність роботи його блоків;
- програми управління дисплеєм, клавіатурою, принтером, зовнішньою пам'яттю;
- інформація про те, де на диску знаходиться операційна система.

ПЗП є енергонезалежною пам'яттю, при відключенні живлення інформація в ньому зберігається.

Залежно від типу запам'ятовуючих елементів розрізняють резисторні, ємнісні, індуктивні, напівпровідникові та інші ПЗП. Найбільш поширеним типом є напівпровідникові інтегральні ПЗП.

Напівпровідникові ПЗП є незалежними і мають велику ємність на одному кристалі.

Запам'ятовуючий масив утворюється системою взаємно перпендикулярних ліній, в перетині яких встановлюються елементи, що

запам'ятовують (ЗЕ). За типом ЗЕ, що встановлюють або розривають зв'язок (контакт) між горизонтальними і вертикальними лініями, розрізняють діодні, біполярні і МОП-схеми ПЗП.

Існують чотири типи ПЗП різного призначення.

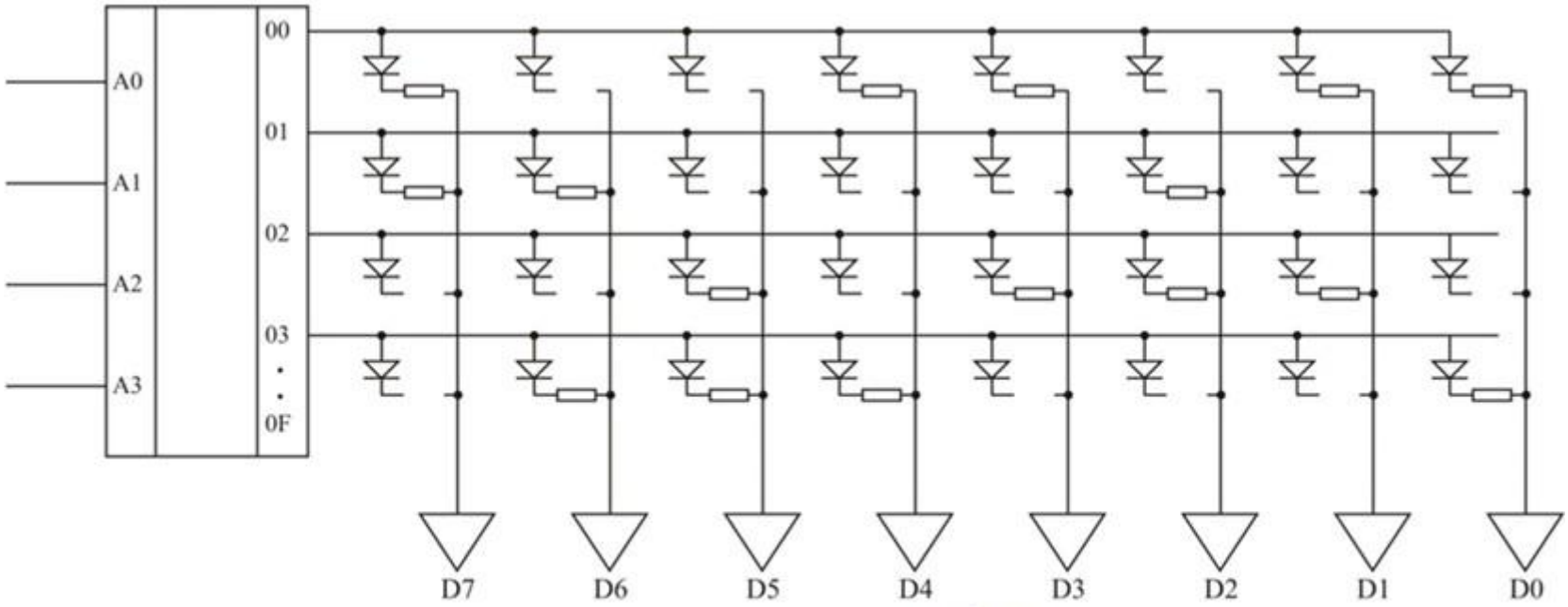
Постійні запам'ятовуючі пристрої з масочним програмуванням - це пристрої, в яких інформація записана раз і назавжди в процесі виготовлення напівпровідникових БІС. Запис проводиться шляхом металізації проміжків, що дозволяють з'єднати через діоди (або через МОП-транзистори) відповідні лінії рядків і стовпців (рис.2.1). Це робиться за допомогою масочних фотошаблонів. Вони задають ділянки металізації, які потрібні для кодування тієї чи іншої інформації. Цим способом виготовляють ПЗП мікропрограм, для перетворення двійкового коду в коди символів (російських, українських і латинських букв, цифр).

Масочний діодний ПЗП, який зображений на рисунку 2.12, містить фрагмент програми ємністю чотири байти, який розташований в адресному просторі ØØh - Ø3h. Схема складається з дешифратора, що має адресні входи і прямі виходи, а також систему адресних і розрядних ліній.

Кількість n адресних входів дешифратора визначається розмірами адресного простору і визначає число виходів дешифратора, рівне 2n. Так, наприклад, для адресації комірок пам'яті, розташованих в адресному просторі ØØh ÷ Ø7h, досить трьох адресних входів АØ - А2. При виході за цей простір кількість адресних входів збільшується.

Залежно від коду, поданого на адресні входи, на одній з адресних ліній з'являється одиничний (високий) потенціал. На всіх інших лініях в цей момент буде низький потенціал. Цей високий потенціал через відкриті діоди по ланцюгу анод - катод і ділянки металізації надходить на розрядні лінії і далі - на підсилювачі зчитування.

Таким чином, формується відповідний код команди або символу.



Адрес		Вміст ПЗП							
00	(9Bh)	1	0	0	1	1	0	1	1
01	(C4h)	1	1	0	0	0	1	0	0
10	(2Eh)	0	0	1	0	1	1	1	0
11	(71h)	0	1	1	1	0	0	0	1

Рис. 2.12 – Масочно-діодний ПЗП

В даний час велика частина ПЗП виготовляється з використанням МОП-технології. Якщо затвор МОП-транзистора приєднаний до обраної рядку, то транзистор відкривається і шунтує напругу рівня «1» на «землю». Тому на відповідній розрядній лінії формується сигнал «0». Якщо затвор МОП-транзистора до обраного рядку не приєднаний, то транзистор не відкривається, і на виході даного розряду ПЗП сигнал має значення, рівне «1».

Крім масочних ПЗП використовуються також ПЗП, програмовані користувачем (ППЗП). Вони відрізняються тим, що при їх виготовленні всі діоди з'єднуються з відповідними стовпцями за допомогою плавких перемичок, як показано на рисунку 2.13.

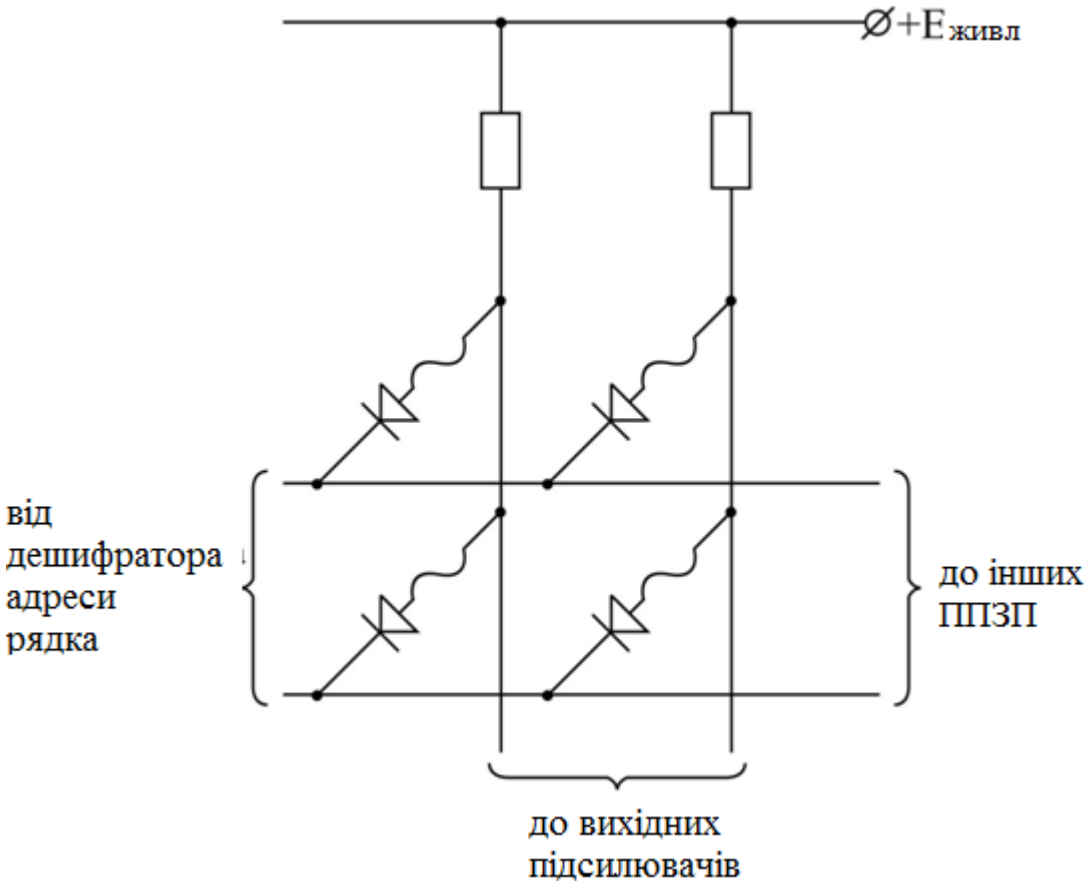
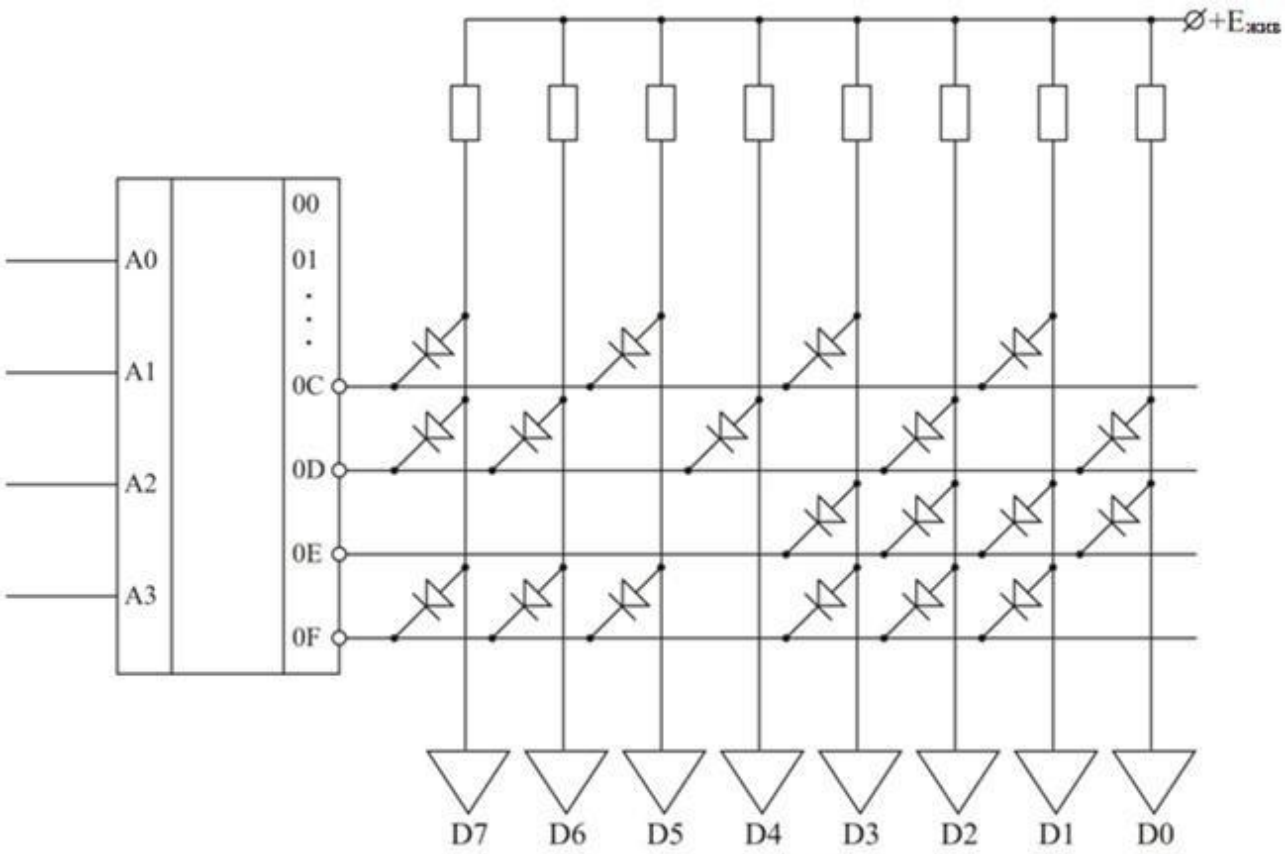


Рисунок 2.13 – Програмований ПЗП с плавкими перемичками

Програмування такого ППЗП полягає в тому, що на нього послідовно подаються адреси слів, а імпульсами струму руйнуються перемички в тих місцях, де вони не потрібні.

В результаті виходить структура пам'яті, зображена на малюнку 2.14. Тут ППЗП містить фрагмент пам'яті, який розташовується в адресному просторі ØCh ÷ ØFh і який складається з чотирьох байтів.

Дешифратор з інверсними виходами подає сигнал «0» на обраний рядок. На тих вихідних вертикальних лініях, на яких збережені діодні зв'язки з обраної горизонтальною лінією, формуються сигнали логічного «0», на інших - логічної «1».



Адрес		Вміст ППЗП							
0C	(55h)	0	1	0	1	0	1	0	1
0D	(2Ah)	0	0	1	0	1	0	1	0
0E	(F0h)	1	1	1	1	0	0	0	0
0F	(11h)	0	0	0	1	0	0	0	1

Рисунок 2.14 – Програмований діодний ПЗП

При проектуванні ППЗП можна керуватися тими ж рекомендаціями, що і в попередньому випадку.

Третім типом ПЗП є перепрограмовані ПЗП (РППЗП). Вони дозволяють проводити запис і стирання інформації. Організація РППЗП відрізняється від організації ППЗП тим, що між лініями рядків і стовпців встановлені не діоди з плавними перемичками, а спеціальні МОП-транзистори з ізольованим затвором. Після виготовлення всі МОП-транзистори мають дуже великий опір (тобто закриті). Подачею імпульсу великий амплітуди МОП-транзистор переводиться в провідний стан, в якому він може зберігати більше 10 років.

Для повернення МОП-транзисторів у попередній (закритий) стан їх треба піддати тривалому впливу ультрафіолетових променів. Групове опромінення всіх МОП-транзисторів здійснюється на спеціальних установках протягом 10 ÷ 30 хв. через прозоре вікно в корпусі мікросхеми. Після цього схема РППЗУ виявляється в початковому стані, і її можна знову програмувати.

Значного поширення набув четвертий тип ПЗП - електрично змінюваний постійний запам'ятовуючий пристрій (ЕЗПЗП). В таких схемах після програмування можна повернути в початковий стан електричним сигналом будь-який окремо взятий МОП-транзистор. ЕЗПЗП енергонезалежні, однак вони не забезпечують довготривалого зберігання інформації. Крім того, вони мають найбільшу вартість і найменшу щільність розміщення інформації. Для стирання, запису і читання даних в них потрібна різна напруга.

2.8 Флеш-пам'ять

В останні роки з'явився новий тип електрично змінюваного ПЗП - ФЛЕШ-пам'ять. Комірка такої пам'яті містить, подібно комірці EEPROM, транзистор, керований "захопленим" зарядом.

Однак технології флеш-пам'яті і EEPROM, незважаючи на велику схожість, істотно розрізняються. Пам'ять EEPROM дозволяє зчитувати і записувати вміст одної комірки. Флеш-пам'ять дає можливість зчитувати комірки по одній, а записувати тільки блоками. Перед записом початковий вміст блоку комірок стирається. Флеш-пам'ять має більшу щільність комірок,

а, отже, більшу ємність і меншу вартість в перерахунку на біт. Для неї достатньо напруги живлення одного рівня, і до того ж вона економічніша.

Флеш-пам'ять (англ. Flash-Memory) - різновид твердотільної напівпровідникової незалежної перезаписувальної пам'яті.

Вона може бути прочитана скільки завгодно раз, але писати в таку пам'ять можна лише обмежене число раз (максимально - близько мільйона циклів). Поширена флеш-пам'ять, що витримує близько 100 тисяч циклів перезапису - набагато більше, ніж здатна витримати дискета або CD-RW.

Не містить рухомих частин, так що, на відміну від жорстких дисків, більш надійна і компактна.

Завдяки своїй компактності, дешевизні і низькому енергоспоживанню флеш-пам'ять широко використовується в цифрових портативних пристроях (рисунок 2.15). Вона застосовується в портативних комп'ютерах, стільникових телефонах, цифрових відеокамерах і плеєрах.

У разі застосування в портативних комп'ютерах і стільникових телефонах флеш-пам'ять містить програмне забезпечення, замінюючи собою носії пам'яті. У цифрових камерах вона використовується для зберігання зображень, а в плеєрах - для зберігання звуку.

Флеш-пам'ять використовується в вигляді великих модулів, що складаються з безлічі мікросхем. Існують дві популярні різновиди таких модулів: флеш-карти і більші модулі - флеш-диски.

Рисунок 2.15 – Різновиди флеш-накопичувачей

Флеш-пам'ять зберігає інформацію в масиві транзисторів з плаваючим затвором, званих комірками. У традиційних пристроях з однорівневими комірками, кожна з них може зберігати тільки один біт. Деякі нові пристрої з багаторівневими комірками можуть зберігати більше одного

біта, використовуючи різний рівень електричного заряду на плаваючому затворі транзистора.

В основі типу флеш-пам'яті NOR лежить АБО-НЕ елемент (англ. NOR), тому що в транзисторі з плаваючим затвором низька напруга на затворі позначає одиницю.

Транзистор має два затвори: керуючий і плаваючий. Останній повністю ізольований і здатний утримувати електрони до 10 років. В комірці є також стік і джерело. При програмуванні напругою на керуючому затворі створюється електричне поле і виникає тунельний ефект. Деякі електрони туннелируют через шар ізолятора і потрапляють на плаваючий затвор, де і будуть перебувати. Заряд на плаваючому затворі змінює «ширину» каналу стік-витік і його провідність, що використовується при читанні.

Програмування та читання комірок сильно розрізняються в енергоспоживанні: пристрої флеш-пам'яті споживають досить великий струм при записі, тоді як при читанні витрати енергії малі.

Для стирання інформації на керуючий затвор подається висока негативна напруга, і електрони з плаваючого затвора переходять (туннелірують) на джерело.

В NOR-архітектурі до кожного транзистора необхідно підвести індивідуальний контакт, що збільшує розміри схеми. Ця проблема вирішується за допомогою NAND-архітектури.

В основі NAND-типу лежить І-НЕ елемент (англ. NAND). Принцип роботи такий же, від NOR-типу відрізняється лише розміщенням комірок і їх контактами. В результаті вже не потрібно підводити індивідуальний контакт до кожної комірки, так що розмір і вартість NAND-чіпа може бути істотно менша. Також запис і стирання відбувається швидше. Однак ця архітектура не дозволяє звертатися до довільної комірки.

NAND і NOR-архітектури зараз існують паралельно і не конкурують одна з одною, оскільки знаходять застосування в різних областях зберігання даних.

2.9 Пам'ять з послідовним доступом на основі зсувних регістрів

Безадресне завдання операндів реалізується пам'яттю магазинного типу. Пам'ять магазинного типу, як показано на рис.2.16 а) і б), утворюється з послідовно з'єднаних регістрів.

Якщо запис в реєстрову структуру (а) проводиться через один регістр, а зчитується через інший, то така пам'ять є аналогом лінії затримки і працює за принципом «Перший увійшов - перший вийшов» (FIFO - first input, first output).

Якщо ж запис і читання здійснюється через один і той же регістр (б), такий пристрій називається стековой пам'яттю, що працює за принципом «Перший увійшов - останнім вийшов» (FILO - first input, last output). При запису числа в стекову пам'ять спочатку вміст стека зміщується в бік останнього К-го регістра (якщо стек був повністю заповнений, то вміст К-го регістра втрачається), а потім число заноситься в вершину стека - регістр 1. читання здійснюється теж через вершину стека . Після того, як число з вершини прочитано, стек зсувається в бік регістра 1.

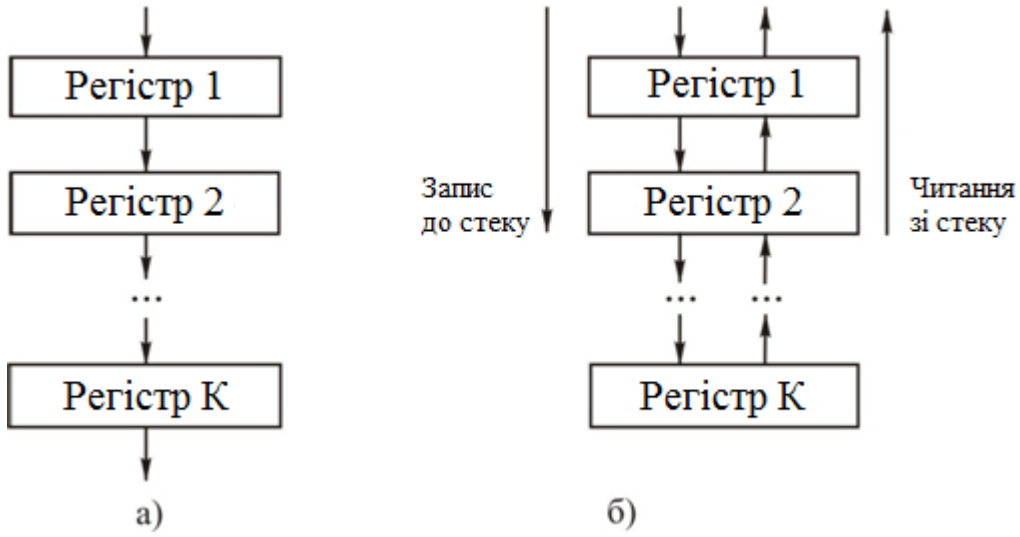


Рисунок 2.16 – Память а) магазинного типа FIFO;
б) стекового типа LIFO

Стекова пам'ять набула широкого поширення. Для її реалізації в ЕОМ розроблені спеціальні мікросхеми. Але часто робота стекової пам'яті емулюється в основній пам'яті ЕОМ - за допомогою програм операційної системи виділяється частина пам'яті під стек (в IBM PC з цією метою виділяється 64 Кбайта).

Спеціальний регістр мікропроцесора - покажчик стека - постійно зберігає адресу комірки оперативної пам'яті, яка виконує функції вершини стека. Читання числа завжди проводиться з вершини стека, після чого покажчик стека змінюється і вказує на чергову комірку стекової пам'яті. Фактично стек залишається нерухомим, а переміщається вершина стека.

При запису числа в стек спочатку номер комірки в покажчику стека модифікується так, що б він вказував на чергову вільну комірку, після чого проводиться запис числа за цією адресою. Така робота покажчика стека дозволяє реалізувати принцип «першим увійшов - останнім вийшов». В стек може бути завантажений в певній послідовності ряд даних, які в наслідку зчитуються з стека вже в зворотному порядку. На цій властивості побудована система арифметичних перетворень інформації, відома під назвою «логіка Лукашевича».

2.10 Асоціативна пам'ять

Пам'ять з вибіркою за змістом (асоціативна пам'ять) є безадресною. Звернення до неї здійснюється за спеціальною маскою, яка містить пошукову область. Інформація зчитується з пам'яті, якщо частина її відповідає образу,який шукається, зафіксованого в масці. Наприклад, якщо в таку пам'ять записана інформація, що містить дані про місце проживання, і необхідно знайти відомості про жителів певного району, то назва цього району (тобто двійковий код) поміщається в маску і дається команда «читання». З пам'яті вибираються всі записи, які стосуються заданого району. У мікропроцесорах асоціативні ЗП використовуються в складі кеш-пам'яті для зберігання адресної частини команд і операндів програми, що виконується. При цьому немає необхідності звертатися до ОП за допомогою такої команди або необхідним операндом: досить помістити в маску потрібну

адресу. Якщо шукана інформація є в СОЗУ, то вона відразу буде видана. Звернення до ОП буде необхідно лише при відсутності необхідної інформації в СОЗУ. За рахунок такого використання СОЗУ скорочується число звернень до ОП, а це дозволяє економити час, так як звернення до СОЗУ вимагає в 2-10 разів менше часу, ніж звернення до ОП.

Кеш-пам'ять може бути розміщена в кристалі процесора (кеш-пам'ять I рівня) або виконана у вигляді окремої мікросхеми (зовнішня кеш-пам'ять II рівня). Вбудована кеш-пам'яті (I рівня) в процесорах Pentium має обсяг десятків Кбайт, час доступу до 5 нс, працює з 32-бітними словами. Зовнішня кеш-пам'ять (II рівня) має обсяг від сотень Кбайт до одиниць Мбайт, час доступу до 10 нс, працює з 64-бітними словами. Конструктивно виконується у вигляді багатоконтактні мікросхеми, або в модулі розширення.

2.11 Принципи контролю роботи пам'яті

Якщо усі розряди слова служать для представлення інформації, код називається простим (не надмірним). Коди, в яких лише частина кодових слів використовується для представлення інформації, називається надлишковим. Частина слів в надлишкових кодах є забороненими, і поява таких слів свідчить про наявність помилки.

Належність слова до дозволених або заборонених слів визначається правилами кодування, і для різних кодів ці правила різні.

Здатність коду виявляти або виправляти помилки визначається так званою кодовою відстанню. Кодовою відстанню між двома словами називається число розрядів, в яких символи слів не збігаються. Якщо довжина n, то кодова відстань може приймати значення від 1 до n. Мінімальною кодовою відстанню даного коду називається мінімальна відстань між двома будь-якими словами в цьому коді. Якщо є хоча б одна пара слів, що відрізняються один від одного тільки в одному розряді, то мінімальна відстань даного коду дорівнює 1.

Простий (не надмірній) код має мінімальну відстань dmin = 1. Для надлишкових кодів dmin> 1. Якщо dmin≥2, то будь-які два слова в даному коді відрізняються не менш ніж у двох розрядах, отже, будь-яка одиночна помилка призведе до появи забороненого слова і може бути виявлена.

2.11.1 Код з перевіркою парності

Код з перевіркою парності утворюється додаванням до групи інформаційних розрядів, які представляють простий (не надмірний) код, одного надлишкового (контрольного) розряду.

При формуванні коду слова в контрольний розряд записується 0 або 1, таким чином, щоб сума одиниць в слові, включаючи надлишковий розряд, була парною (при контролі по парності) або непарній (при контролі за непарності). Надалі під час запису в пам'ять або зчитуванні слово передається разом зі своїм контрольним розрядом. Якщо при передачі інформації приймальний пристрій виявляє, що в прийнятому слові значення контрольного розряду не відповідає парності суми одиниць слова, то це сприймається як ознака помилки.

Мінімальна відстань коду dmin = 2, тому код з перевіркою парності виявляє все поодинокі помилки, а, крім того, всі випадки непарного числа помилок (3, 5, 7 і т.д.). При одночасному виникненні двох або будь-якого іншого парного числа помилок код з перевіркою парності не може виявити помилок.

Таблиця 2.1 - Значення контрольного розряду

A_{10}	A_2	**Значення контрольного розряду при контролі**	
		по парності	**по непарності**
0	0000	0	1
1	0001	1	0
2	0010	1	0
3	0011	0	1
4	0100	1	0
5	0101	0	1
6	0110	0	1
7	0111	1	0
8	1000	1	0
9	1001	0	1

Доцільно число одиниць в кодовому наборі з виявленням одиночної помилки вибирати непарним (здійснювати контроль за непарності). Тоді будь-яке кодове подання, в тому числі і для нуля матиме хоча б одну одиницю. Це дає можливість відрізнити повна відсутність (пропажа) інформації від передачі нуля. Кодове слово, яке складається з нулів, в цьому випадку, буде ставитися до заборонених.

Код з перевіркою парності має невелику надмірність і не потребує великих витрат обладнання для реалізації контролю. Цей код широко застосовується в обчислювальних машинах для контролю інформації, що зчитується в оперативній пам'яті.

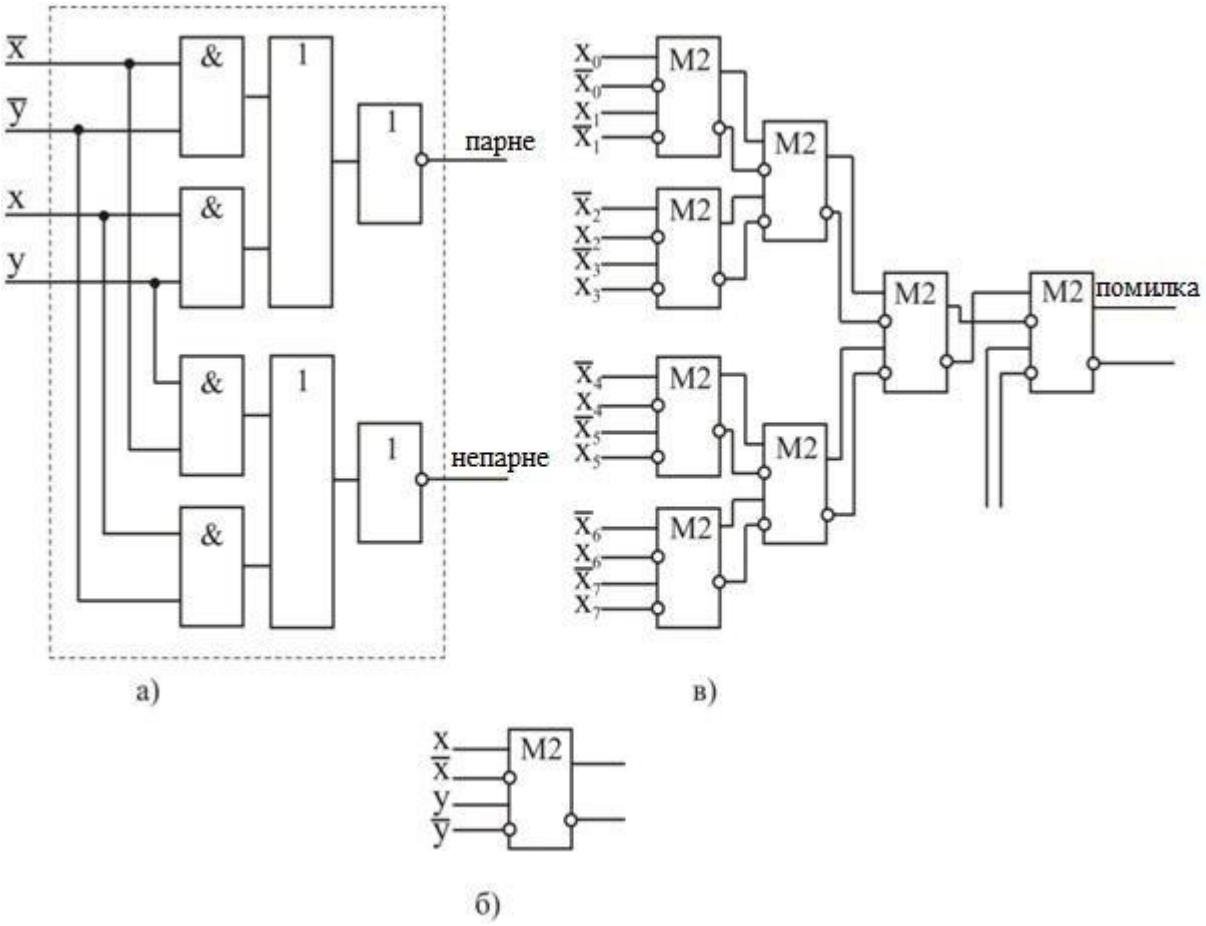


Рисунок 2.17 – Схеми визначення парності

Для побудови схеми визначення парності одного слова використовують логічні елементи з парафазними виходами (рис. 2.17а, 2.17б). Ці схеми виконують операції додавання по модулю 2 (умовне позначення - М2) для двійкових змінних х і у. На рис. 2.17в показана схема визначення ознаки парності байта.

Кожен інформаційний символ повинен задаватись прямим і інверсним кодом. Схема перевірки парності є багатоступеневою, тобто слово ділиться на кілька груп розрядів, в кожній з яких перевірка парності проводиться прямим способом (перший ступінь). Далі проводиться перевірка парності для груп другого ступеня, утворених з груп першого ступеня, парності яких в цьому випадку розглядаються як звичайні двійкові розряди і т.д. до остаточної перевірки парності суми одиниць всього слова. В останньому ступені парність байта порівнюється зі значенням контрольного розряду КР.

Якщо кількість одиниць в слові має бути парною, то в контрольний розряд записується прямий код суми по модулю 2 всіх інформаційних розрядів слова. При контролі на парність в контрольний розряд заноситься зворотний код зазначеної суми.

2.11.2 Контроль роботи пам'яті

У будь-якої з багатьох мільйонів комірок пам'яті можливий випадковий збій або остаточна відмова, що приводить до помилки. Імовірність помилки зростає зі збільшенням обсягу пам'яті.

У перших моделях РС, коли мікросхеми пам'яті мали суттєво гірші характеристики надійності в порівнянні з сучасними, обов'язково застосовувався контроль парності. При його використанні кожен байт пам'яті супроводжувався бітом паритету, що доповнює кількість одиниць в байті до непарного. Значення контрольного біта (біта паритету) апаратно генерується під час запису в пам'ять і перевіряється при зчитуванні. При виявленні помилки схемою контролю виробляється незамасковане переривання, в результаті чого на екран виводиться повідомлення про помилку, вказується адреса збійної комірки і зупиняється процесор. Спеціальними методами помилку тестування можна ігнорувати.

Згодом якість мікросхем пам'яті покращилась, і з метою їх здешевлення стали відмовлятися від застосування контролю парності. З'явилася маса моделей системних плат, в яких контролю паритету немає взагалі. Модулі пам'яті стали випускатися як з паритетом, так і без нього, а для обману плат, що вимагають наявності біта паритету, стали випускати моделі з підробленим паритетом. У цих моделях використовується генератор паритету - логічна схема суматора по модулю 2, формує завжди хороший біт паритету незалежно від наявності помилок в самій пам'яті. У позначенні типу модуля з генератором паритету входять буквосполучення BP, VT, GSM, MPEC. Ніякі програмні засоби тестування пам'яті не можуть відрізнити пам'ять з дійсним паритетом від пам'яті з фіктивним паритетом.

Навіть коли контроль паритету дійсно існує, він не всесильний.

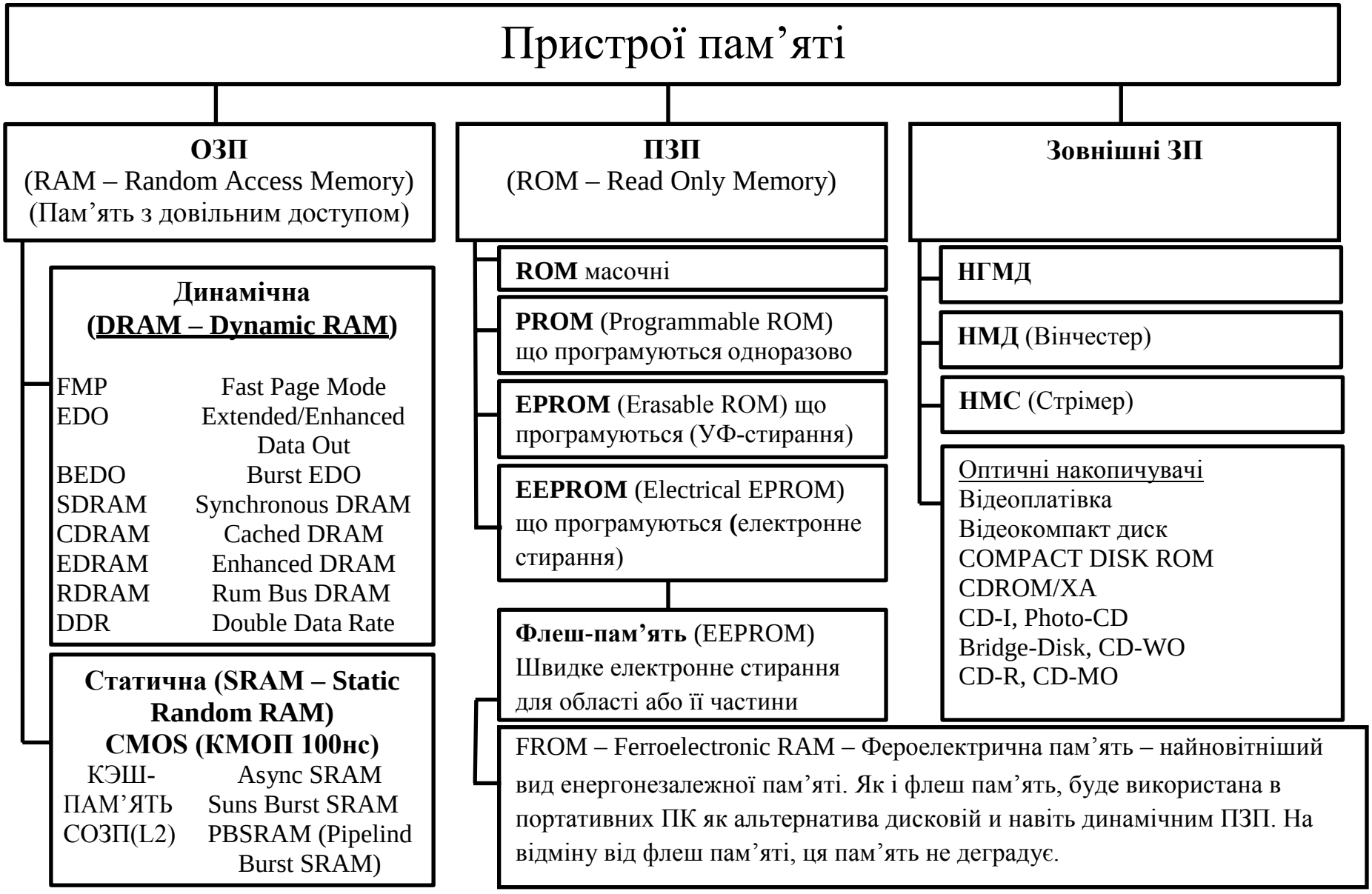
Він виявляє в межах кожного байта помилки тільки непарної кратності (спотворення 1, 3, 5 або 7 біт). Правда, одночасна відмова двох біт в одному байті малоймовірна.

Можливий збій в самому контрольному біті.

У комп'ютерах особливо відповідального застосування використовують пам'ять з виявленням і виправленням помилок - ECC Memory (Error Checking and Correcting). У такому випадку для кожного записуваного інформаційного слова пам'яті (а не байта, як при контролі паритету) за певними правилами обчислюється функція згортки, результат якої розрядністю в кілька біт також зберігається в пам'яті. Для 64-бітного слова зазвичай використовують 7-8 додаткових біт. При цьому схема контролю здатна виявляти помилки з різною кратністю і виправляти одноразові помилки.

У найсерйозніших системах ECC застосовується не тільки для ОЗП, але і для КЕШ - пам'яті.

КЛАСИФІКАЦІЯ ПРИСТРОЇВ ПАМ'ЯТІ



3 ПРОЦЕСОРИ

Центральний процесор.

Найважливіший елемент комп'ютера - центральний процесор. Часто його називають ЦПП або CPU (Central Processor Unit - Центральне Процессорное Пристрій), а також кристал, камінь, хост-процесор, де проводяться всі необхідні обчислення і звідки надходять команди для управління комп'ютером. І чим потужніший процесор, тим швидше працює комп'ютер.

Перші процесори впаюються в материнську плату. Надалі з метою можливої швидкої заміни процесорів були розроблені спеціальні роз'єми , в які можна було б швидко встановити необхідний процесор. Роз'єм - сокет (Soket), в який вставляються мікросхеми процесора, які знаходяться на всій поверхні роз'єму, і слот (Slot), у якого контакти розташовані по периметру або на одній лінії.

Сучасні персональні комп'ютери використовують, як правило, певний алгоритм обробки даних, званий архітектурою Фон Неймана, коли інструкції і самі дані зберігаються в одній пам'яті, а сам процес обробки побудований на циклічній послідовній обробці даних. Правда, саме послідовність обробки є вузьким місцем такої архітектури, оскільки будь-яке дане повинно послідовно пройти через процесор, хоча саме обчислення може бути однотипним.

Цифровий комп'ютер складається з пов'язаних між собою процесорів, пам'яті і пристроїв введення-виведення.

Центральний процесор - це мозок комп'ютера. Його завдання - виконувати програми, що знаходяться в основній пам'яті. Він викликає команди з пам'яті, визначає їх тип, а потім виконує їх одну за одною. Компоненти з'єднані шиною, що представляє собою набір паралельно пов'язаних дротів, по яких передаються адреси, дані і сигнали управління. Шини можуть бути зовнішніми (зв'язують процесор з пам'яттю і пристроями введення-виведення) і внутрішніми. На рис.3.1 показана схема пристрою центрального процесора.

Процесор складається з декількох частин: блоку управління, арифметико-логічного пристрою і регістрів.

Блок управління відповідає за виклик команд з пам'яті і визначення їх типу.

Арифметико-логічний пристрій ALU, працює з цілими числами, виконує арифметичні операції, такі, як додавання, віднімання, множення і так далі. Крім цього, він може виконувати ряд додаткових операцій, наприклад, логічний зсув,

коли значення зсуваються на один розряд вліво або вправо, порівняння чисел і інші операції.

У процесорі є кілька регістрів, які відрізняються своїми функціональними можливостями.

Усередині центрального процесора знаходиться пам'ять для зберігання проміжних результатів і деяких команд управління. Ця пам'ять складається з декількох регістрів, кожен з яких виконує певну функцію. Зазвичай все регістри однакового розміру. Кожен регістр містить одне число, яке обмежується розміром регістра. Регістри зчитуються і записуються дуже швидко, оскільки вони знаходяться всередині центрального процесора.

Найважливіший регістр - лічильник команд, який вказує, яку команду потрібно виконувати далі. Назва «лічильник команд» не відповідає дійсності, оскільки він нічого не рахує, але цей термін вживається повсюдно.

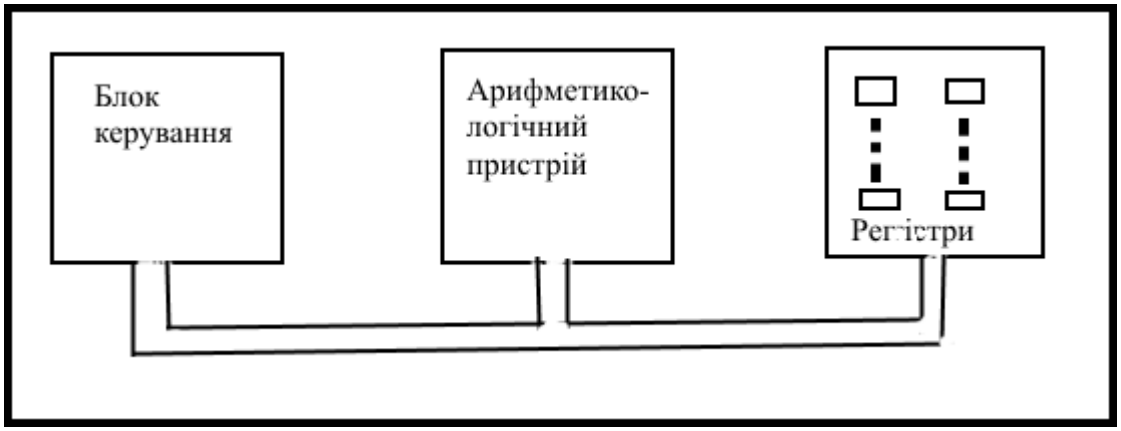


Рисунок 3.1 Схема устрою центрального процесора

Ще є регістр команд, в якому знаходиться команда, яка виконується в даний момент. У більшості комп'ютерів є і інші регістри, одні з них багатофункціональні, інші виконують тільки будь-які специфічні функції.

Регістри, які зберігають дані, отримані з оперативної пам'яті, мають назви AX, BX, CX, DX (де X - від слова eXtended - розширений). Вони розділені на дві частини: наприклад, для AX ліва частина - AH і права - AL, для BX - BH і BL і так далі. Розмір регістра залежить від типу процесора. Перші моделі мали 16 розрядів, а останні - 64. Регістри цікаві тим, що визначають можливості центрального процесора. Чим більші вони за розміром, тим більше даних можна обробити, але для цього повинне бути відповідне програмне забезпечення. У перших моделях було 14 регістрів. У сучасних комп'ютерах

більше регістрів, ніж в перших моделях, частина з них видна для програміста, тобто він може ними скористатися, частина прихована від програми, але використовується модулями центрального процесора. Сучасні процесори програмно скасовують ім'я регістра, що підвищує продуктивність центрального процесора. Перейменування регістрів використовується для запису проміжних результатів, що дозволяє виконувати декілька інструкцій одночасно.

Крім регістрів, в процесорі існує регістр прапорів. У ньому знаходиться інформація про стан процесора (чи відповідати на зовнішні переривання, порядок обробки даних та ін.), адреса наступної команди, результат порівняння, переповнення при додаванні і ін., що повністю описує стан завдання в поточний момент часу.

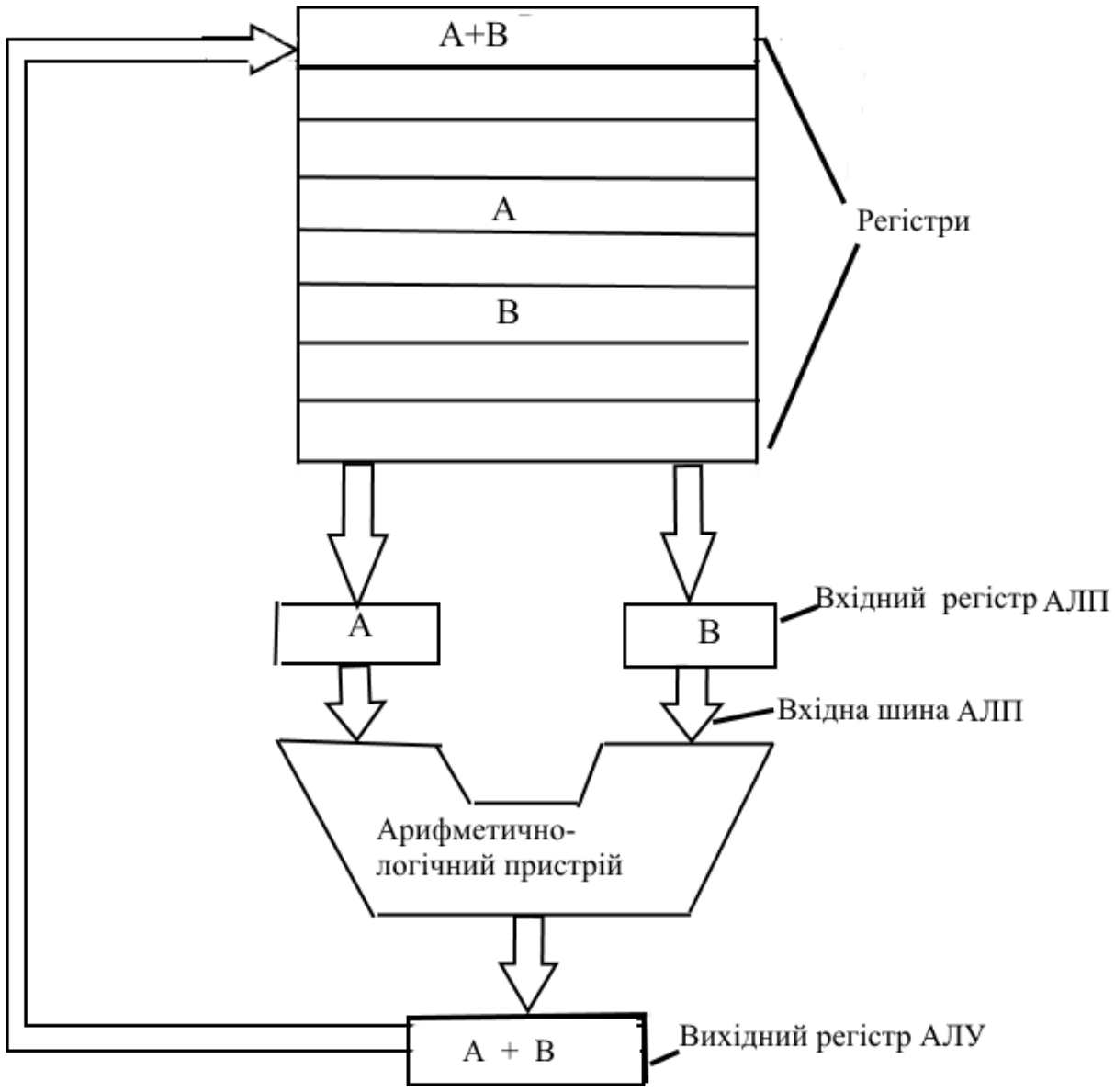


Рисунок 3.2 Тракт даних фон-неймановського процесора

Внутрішній устрій тракту даних типового фон-неймановского процесора показано на рис. 3.2.

Тракт даних складається з регістрів (зазвичай від 1 до 32), АЛП

(Арифметико-логічного пристрою) та кількох з'єднуючих шин. Вміст регістрів надходить у вхідні регістри АЛУ, які на рис. 2 позначені буквами А і В. У них знаходяться вхідні дані АЛП, поки АЛП робить обчислення. Тракт даних - важлива складова частина всіх комп'ютерів.

АЛП може додавати, віднімати і інші прості операції над вхідними даними і поміщає результат у вихідний регістр. Цей вихідний регістр може міститися назад в один з регістрів. Він може бути збережений в пам'яті, якщо це необхідно. На рис. 2 показана операція додавання. Відзначимо, що вхідні і вихідні регістри є не у всіх комп'ютерів.

Більшість команд можна розділити на дві групи: команди типу регістр-пам'ять і типу регістр-регістр. Команди першого типу викликають слова з пам'яті, поміщають їх в регістри, де вони використовуються в якості вхідних даних АЛП.

(Слова - це такі елементи даних, які переміщаються між пам'яттю і регістрами. Слова зазвичай відповідає розрядності регістру даних. Так, наприклад, у 16-бітних мікропроцесорів 8086 і 8088 слово було 16-бітовим, а у 32-бітних мікропроцесорів слово має довжину 32 біта. Словом може бути ціле число. Інші команди цього типу поміщають регістри назад в пам'ять.)

Команди другого типу викликають два операнда з регістрів, поміщають їх у вхідні регістри АЛП, виконують над ними якусь арифметичну або логічну операцію і переносять результат назад в один з регістрів. Цей процес називається циклом тракту даних. В якійсь мірі він визначає, що може робити машина. Чим швидше відбувається цикл тракту даних, тим швидше комп'ютер працює.

Виконання команд

Центральний процесор виконує кожну команду за кілька кроків:

1) викликає наступну команду з пам'яті і переносить її в регістр команд;

2) змінює положення лічильника команд, який тепер повинен вказувати на наступну команду (Це відбувається після декодування поточної команди, а іноді і після її виконання.);

3) визначає тип викликаної команди;

4) якщо команда використовує слово з пам'яті, визначає, де знаходиться це слово;

5) переносить слово, якщо це необхідно, в регістр центрального процесора (Бувають команди, які потребують завантаження з пам'яті чималої кількості слів і їх обробки в рамках однієї-єдиної команди.);

6) виконує команду;

7) переходить до кроку 1, щоб почати виконання наступної команди.

Така послідовність кроків (вибірка-декодування-виконання) є основою роботи всіх комп'ютерів.

Характеристики центрального процесора

Існують наступні основні характеристики центрального процесора: тактова частота, кількість ядер, встановлена кеш пам'ять, вид оперативної пам'яті, з якою працює процесор, сокет (роз'єм) і частота системної шини.

Тактова частота визначає, на якій частоті працює центральний процесор. За один такт може виконуватися кілька операцій. Чим вища частота, тим швидше працює комп'ютер. У 90х роках і на початку 2000х основний фактор збільшення продуктивності комп'ютера було збільшення тактової частоти. Однак згодом виявилося, що існує фізична межа збільшення тактової частоти. Сучасні продуктивні процесори випускаються з тактовою частотою від 1.8 до 4 МГц (2 Мгц позначає, що за секунду відбувається 2 мільйони коливань, під час яких відбувається робота процесора). Бюджетні варіанти мають меншу частоту, яка менша від зазначеної вище.

Тому високопродуктивні процесори стали випускатися з кількома конвеєрами, потім ядрами. Кожне ядро практично являє собою окремий процесор. Чим більше ядер, тим швидше працює комп'ютер. Конвеєр це проміжна ланка між одноядерним і багатоядерним процесором. На комп'ютері зазвичай працює кілька завдань, наприклад, операційна система, антивірусна програма, браузер та інше. Якщо кожна з них буде працювати на своєму ядрі, то вони будуть працювати паралельно один з одним. Можна розпаралелити і звичайну програму, але виробники математичного забезпечення поки що рідко це роблять. Але вже з'являються перші програми, які можуть працювати з декількома ядрами одночасно, наприклад, Photoshop.

Наступною характеристикою є сокет (або роз'єм), в який вставляється центральний процесор. Якщо процесор призначений для певного виду сокета, наприклад, Socket 478, то його не можна встановити в Socket 479. Тим часом, на

материнській платі знаходиться тільки один сокет для центрального процесора і він повинен відповідати типу процесора.

Також основною характеристикою є частота системної шини. Чим більша частота системної шини, тим більше даних передається за відрізок часу. За один такт можна передати для старих комп'ютерів один біт, для сучасних кілька. Є інший показник - пропускна здатність шини. Він дорівнює частоті системної шини, помноженої на кількість біт, які можна передати за один такт. Якщо частота системної шини дорівнює 100 Мгц, а за один такт передається два біти, то пропускна здатність буде 200 Мбіт / с. Якщо можна передати за один такт 8 біт, то пропускна здатність дорівнює 800 Мбіт / сек. Ясно, що швидкодія в другому випадку буде вище, незважаючи на те, що частота системної шини залишилася одна і та ж. В даний час пропускна здатність шини починає обчислюватися в гігабітах (або в десятках гігабітів) в секунду. Чим вище цей показник, тим краще.

Типи процесорів. Основний компанією, яка випускає центральні процесори для персональних комп'ютерів, є компанія Intel. Практично всі персональні комп'ютери починають відлік часу з появи першого процесора 8086. І хоча в той час випускалися і інші види процесорів, саме ця модель стала завойовувати популярність як серед виробників пристроїв для них, так і серед користувачів. Як приклад можна навести комп'ютер компанії Apple, який має дещо іншу модель процесора і не набув такого широкого поширення в силу її закритої системи.

Наступна модель називалася 80286, потім 80386, з часом цифра 80 стали опускати і процесори стали називати трьома цифрами: 286, 386, 486. Тому часто покоління розроблених процесорів називають сімейством x86. В даний час випускаються і інші моделі процесорів, наприклад, сімейства Alpha, Power PC і інші.

Основними характеристиками процесорів є частота і кількість розрядів, за якими можна адресувати дані. Частота вимірюється в герцах, і чим більша вона, тим швидше працює процесор. Один герц позначає один цикл в секунду і зазвичай вказується швидкість роботи в кілогерцах (Кгц або 1 000 циклів в секунду), або мегагерцах (Мгц або 1 000 000 циклів в секунду), або гегагерцах (Ггц дорівнює 1 000 000 циклів в секунду). Підвищення розрядності покращує продуктивність комп'ютера. Розглянемо основні типи процесорів, які можуть бути: 8086, 80286, 80386, 80486, Pentium, Pentium Pro, Pentium MMX, Pentium II, Pentium III і Pentium IV. Celeron позначає урізаний варіант процесора Pentium. Після назви зазвичай наводиться тактова частота процесора, наприклад, Celeron

450, що позначає тип процесора (Celeron) і тактову частоту (450 Мгц), на якій він працює.

Кількість оброблюваних даних одночасно. Однією з характеристик процесора є кількість даних, які обробляються за один такт. Чим більше даних може бути оброблено, тим вища продуктивність у процесора, тим швидше вони обробляються. У перших процесорах серії 8086 процесор обробляв по 16 біт даних. Ця характеристика безпосередньо пов'язана з розміром регістрів всередині ЦП. Якщо розмір регістрів 16 біт, то центральний процесор обробляє 16 біт одночасно.

Іншою головною характеристикою процесора є кількість даних, якими він може обмінюватися із зовнішніми пристроями або пропускна здатність шини. Відзначимо, що оперативна пам'ять є зовнішнім пристроєм для процесора. При цьому, чим більше даних одночасно буде відправдено / отримано, тим вища продуктивність процесора. Ця характеристика визначається кількістю ліній системної шини для одночасної передачі даних. Чим їх більше, тим більше даних може бути передано. У перших процесорах було від 8 до 16 подібних ліній, потім 32, а для зв'язку з оперативною пам’ятью - 64.

З появою нових видів процесорів діє евристичне правило Гордона Мура (одного із засновників компанії Intel), яке свідчить, що кожне десятиліття кількість елементів в процесорі збільшується в 100 разів, а ціни на процесори за півтора року падають в два рази.

Кожен новий вид процесора мав переваги перед попередніми моделями. Як правило, це стосується його швидкодії, наприклад, впроваджується новий вид команд, скажімо, MMX для того, щоб підвищити продуктивність процесора при роботі з графікою (як правило, це потрібно для ігрових програм). Крім того, можуть вводитися нові елементи в сам процесор (наприклад, кеш всередині процесора), які не змінюють принципів роботи процесора, але забезпечують його підвищену продуктивність.

Комп'ютери 486 серії мали тактову частоту 25 і 33 Мгц. У перших моделях Pentium основні частоти становили 50, 60, 66 Мгц. Сучасні комп'ютери випускаються з частотами 100 і 133 МГц і вище. В силу того, що процесор працює на власній частоті, яка перевищує частоту системної шини, вводиться помножуваний коефіцієнт, який свідчить про кількість тактів, вироблених процесором за один такт системної шини. Наприклад, процесор Pentium 120 має тактову частоту 120 Мгц, а частоту системної шини 60 Мгц, тобто за один такт системної шини відбудеться два такти у процесорі. Цей коефіцієнт може бути не цілим числом, наприклад, у процесора Pentium 166 частота процесора

становить 166 МГц, а частота системної шини 66 Мгц, тобто помножуваний коефіцієнт дорівнює 2,5. У цьому випадку за два такти системної шини відбувається 5 тактів у центральному процесорі.

Основні види процесорів.

Процесори персонального комп'ютера випускаються в форматі CISC (Complex Instruction Set Computer - комп'ютер зі складним набором інструкцій), тобто кожна машинна інструкція виконується безпосередньо процесором. На відміну від даного виду процесорів, існує інший підхід: процесори RISC (Reduced Instruction Set Computer - комп'ютер зі зменшеним набором інструкцій), які мають команди однієї довжини. Якщо на вхід комп'ютера потрапить команда з розширеного набору, то вона виконується кількома інструкціями. Кожен з цих підходів має свої переваги і недоліки. RISC процесори працюють швидше, але коли зустрічається команда, яку потрібно транслювати, вона виконується повільніше, проте сам процесор влаштований простіше, ніж CISC. Крім того, RISC процесори виконують за один такт кілька команд, а деякі CISC процесори потребують кількох тактів.

Тому розробники пішли на випуск нових процесорів, які мають і RISC, і CISC підходи. В майбутньому будуть розроблені процесори з VLCW обробкою (Very Long Computer Word - дуже довге машинне слово), в яких кілька інструкцій поміщаються в один запис і подаються на вхід процесора, який обробляє кілька команд одночасно, будуть реалізовані й інші підходи.

Процесор, крім внутрішньої роботи, має шини, через які він отримує (надсилає) дані. У самому процесорі є пристрій інтерфейсу шини, який відповідальний за прийом / передачу даних, зокрема, підсилює вихідний сигнал для того, щоб сигнал дійшов до пункту призначення, при цьому посилюючи і вхідні сигнали, щоб їх можна було розпізнати на іншому кінці шини. Крім того, у нього є багато додаткових функцій, таких, як узгодження сигналу та ін.

Продуктивність

Продуктивність процесорів, аж до Pentium, майже в 1,5-2,0 рази вища при однакових тактових частотах у порівнянні з попередніми моделями комп'ютерів. Так, 80286 швидше, ніж 8086 приблизно в 1,5 рази, 80386 швидше 80286 приблизно 1,5 разу і так далі. Винятком є процесори серій Pentium MMX,

Pentium II і Pentium Pro , оскільки їх продуктивність буде залежати від видів додатків, з якими працює комп'ютер.

Команди процесора

Центральний процесор виконує команди, число яких не перевищує двох сотень. Основними з них є найпростіші арифметичні команди, які виконуються над даними, що знаходяться в регістрах. Регістр - це область пам'яті, розташована в центральному процесорі і має досить невелику ємність: до декількох байт, в залежності від типу процесора. Через те що центральний процесор виконує арифметичні операції в регістрах, то часто використовуються команди пересилання даних з оперативної пам'яті в регістри і назад. Щоб скласти два числа, які знаходяться в оперативній пам'яті, необхідно спочатку переслати їх в регістри, потім виконати операцію додавання, а потім переслати в оперативну пам'ять. Крім того, може зустрітися переповнення при додаванні чисел і виникає сітуація, коли результат не поміщається в виділений для нього регістр, і викликається програма обробки даної помилки. Тобто для однієї операції може виникнути необхідність виконання відразу декількох операцій. Оскільки процесор має велику продуктивність, то ці операції виконуються дуже швидко. Коли повідомляється, що процесор може працювати з частотою 400 мегагерц, то це не означає, що він виконає 400 мільйонів арифметичних операцій за секунду.

По-перше, для однієї операції додавання можуть знадобитися додаткові команди, наприклад, пересилання, і, по-друге, не всі команди виконуються за один такт, деякі складні команди можуть потребувати для виконання кількох тактів і, по-третє, при великій кількості операцій введення-виведення, тобто пересилання даних по системній шині, процесор, надіславши запит на дане з оперативної пам'яті, може простоювати до тих пір, поки шина не звільниться.

Для складання чисел є кілька різних видів команд, кожна з яких працює з певними видами даних в залежності від їх довжини і виду (двійкове, десяткове, упаковане і ін.). В результаті, хоча команд багато, існує досить невелика кількість груп команд, які досить легко систематизуються.

Крім зазначених, існують команди переходів для передачі управління команді, яка знаходиться в іншому місці програми (пам'яті). Існують умовні та безумовні переходи, тобто перехід за умови, що щось виконано, наприклад, лічильник досяг певного значення, або без всяких умов.

Існують і інші команди, наприклад, циклічний зсув розрядів числа, порожній оператор та ін., які використовуються досить рідко в порівнянні з

вищевказаними командами. Команди для центрального процесора мають розмір від 1 до 11 байт, в середньому 4-5 байт.

Дані, одержувані процесором, можна розділити на дві частини: це безпосередньо дані, які містяться в регістри процесора, і інструкції. Інструкції потрапляють до обробника інструкцій, який за першим байтом визначає, скільки байт займає команда, чи уся команда отримана з пам'яті. Якщо команда отримана не вся, то решта команди повинна бути зчитана з оперативної пам'яті. Далі необхідно перевірити, з якими даними працює команда. Якщо вони знаходяться в оперативній пам'яті, то їх потрібно звідти переслати, а коли команда готова до виконання, то виконати дану машинну команду. З внутрішнього сховища обираються мікрокоманди, що відповідають команді. Після цього мікрокоманди виконуються у центральному процесорі.

Кеш-пам'ять

Починаючи з 486 процесорів, стала використовуватися кеш-пам'ять, яка відіграє роль проміжного сховища даних. В силу того, що дані між процесором і кеш-пам'яттю переміщуються швидше, ніж між процесором і оперативною пам'яттю, то при необхідності записати в оперативну пам'ять дані, вони тимчасово записуються в кеш-пам'ять, а потім передаються в оперативну пам'ять, коли звільниться системна шина. При читанні даних процесор спочатку перевіряє, чи є дані в кеш-пам'яті, і, якщо є, то зчитуються з неї, а якщо немає, то відбувається звертання до оперативної пам'яті і таким чином, за рахунок зниження простою процесора, збільшується продуктивність комп'ютера. Вся робота з кеш-пам'яттю проводиться апаратними засобами процесора. Програміст, який написав програму, часто не знає, як це відбувається, тому ті програми, які були написані до введення кеш-пам'яті в комп'ютер, також будуть використовувати цю пам'ять, що збільшує їх продуктивність.

Кількість ядер

Від кількості ядер залежить, скільки процесів зможе реально виконуватись одночасно. Кількість ядер останнім часом міцно увійшла в основні характеристики процесора, і багато хто помилково вважає, що якщо ядер більше, то завжди буде приріст продуктивності. На жаль, якщо програма ще не оптимізована під 4 ядра, то вона 4 ядра використовувати не буде.

4 СУПЕРКОМП’ЮТЕРИ. ПАРАЛЕЛЬНІ ОБЧИСЛЮВАЛЬНІ СИСТЕМИ.

Суперкомп'ютер (Санглена. Supercomputer), - спеціалізована обчислювальна машина, яка значно перевершує за своїми технічними параметрами і швидкості обчислень більшість існуючих в світі комп'ютерів.

Як правило, сучасні суперкомп'ютери - це велике число високопродуктивних серверних комп'ютерів, з'єднаних один з одним локальною високошвидкісною магістраллю для досягнення максимальної продуктивності в рамках підходу розпаралелювання обчислювальної задачі.

Більшість суперкомп'ютерів 70-х оснащувалися векторними (матричними) процесорами, а на початку і в середині 80-х невелике число (від 4 до 16) паралельно працюючих векторних процесорів практично стало стандартним суперкомп'ютерним рішенням. Кінець 80-х і початок 90-х років XX ст. охарактеризувалися зміною магістрального напряму розвитку суперкомп'ютерів від векторно-конвеєрної обробки до великого та надвеликого числа паралельно з'єднаних скалярних процесорів.

Масово-паралельні системи стали об'єднувати в собі сотні і навіть тисячі окремих процесорних елементів. Більшість масивно-паралельних комп'ютерів створювалася на основі потужних процесорів з архітектурою RISC, на зразок PowerPC або PA-RISC.

В кінці 90-х років набули поширення комп'ютерні кластери. Ці системи характеризується використанням окремих вузлів на основі дешевих і широко доступних комп’ютерних комплектуючих для серверів і персональних комп'ютерів і об'єднаних за допомогою потужних комунікаційних систем і спеціалізованих програмно-апаратних рішень. Незважаючи на відносну простоту, кластери досить швидко зайняли досить великий сегмент суперкомп'ютерного ринку, забезпечуючи найвищу продуктивність при мінімальній вартості рішень.

В наш час суперкомп’ютерами прийнято називати комп'ютери з величезною обчислювальною потужністю. Такі машини використовуються для роботи з додатками, які потребують найбільш інтенсивних обчислень (наприклад, прогнозування погодно-кліматичних умов, моделювання ядерних випробувань і т. п.). Іноді суперкомп'ютери використовуються для

роботи з одним-єдиним додатком, що використовує всю пам'ять і всі процесори системи; в інших випадках вони забезпечують виконання великої кількості різноманітних додатків.

Продуктивність суперкомп'ютерів найчастіше оцінюється і виражається в кількості операцій з плаваючою точкою в секунду (FLOPS). Це пов'язано з тим, що завдання чисельного моделювання, під які і створюються суперкомп'ютери, найчастіше потребують обчислень, пов'язаних з числами з високим ступенем точності, а не з цілими числами. Тому для суперкомп'ютерів непридатна міра швидкодії звичайних комп'ютерних систем - кількість мільйонів операцій в секунду (MIPS). Попри всю свою неоднозначність і приблизність, оцінка у флопс дозволяє легко порівнювати суперкомп'ютерні системи одну з одною, спираючись на об'єктивний критерій.

Перші суперкомп'ютери мали продуктивність близько 1 кфлопс, тобто 1000 операцій з плаваючою точкою в секунду. У США комп'ютер, що мав продуктивність в 1 мільйон флопс (1 Мфлопс) (CDC 6600), був створений в 1964 році. Відомо, що в 1963 році в московському НДІ-37 (пізніше НДІ ДАР) був розроблений комп'ютер на основі модулярной арифметики з продуктивністю 2,4 млн. Оп / с. Це експериментальний комп'ютер другого покоління (на дискретних транзисторах) Т340-А (гл. Конструктор Д.І. Юдицький). Однак слід зазначити, що пряме порівняння продуктивності модулярних і традиційних ЕОМ некоректно. Модулярная арифметика оперує тільки з цілими числами. Подання дійсних чисел в модулярних ЕОМ можливо тільки в форматі з фіксованою комою, недоліком якого є істотне обмеження діапазону представлення чисел.

Планка в 1 мільярд флопс (1 гігафлопс) була подолана суперкомп’ютерами NECSX-2 в 1983 році з результатом 1.3 Гфлопс, і М-13 академіка Карцева з результатом в 2,4 Гфлопс.

Кордон в 1 трильйон флопс (1 Тфлопс) досягнутий в 1996 році суперкомп'ютером ASCIRed.

Рубіж 1 квадрильйон флопс (1 петафлопс) був узятий в 2008 році суперкомп'ютером IBMRoadrunner.

З 2010-го року декількома країнами ведуться роботи спрямовані на створення до 2020 року екзафлопсних комп'ютерів, здатних виконувати 1

квінтильйон операцій з плаваючою точкою в секунду, і споживати при цьому не більше кількох десятків мегават.

Що стосується програмного забезпечення, то в даний час межі між суперкомп'ютерним і загальновживаним програмним забезпеченням сильно розмиті і продовжують розмиватися ще більш разом з проникненням технологій паралелізації і багатоядерності в процесорні пристрої персональних комп'ютерів і робочих станцій. Виключно суперкомп'ютерним програмним забезпеченням сьогодні можна назвати лише спеціалізовані програмні засоби для управління і моніторингу конкретних типів комп'ютерів, а також унікальні програмні середовища, що створюються в обчислювальних центрах під «власні», унікальні конфігурації суперкомп'ютерних систем.

4.1 Паралелізм на рівні команд

Розробники комп'ютерів прагнуть до того, щоб підвищити продуктивність своїх машин. Один із способів змусити процесори працювати швидше - підвищення їх тактової частоти, однак при цьому існують деякі технологічні обмеження, пов'язані з конкретним історичним періодом. Тому більшість розробників для підвищення продуктивності при даній тактовій частоті процесора використовують паралелізм (виконання двох або більше операцій одночасно).

Існує дві основні форми паралелізму: паралелізм на рівні команд і паралелізм на рівні процесорів. У першому випадку паралелізм реалізується за рахунок запуску великої кількості команд кожну секунду. У другому випадку над одним завданням працюють одночасно кілька процесорів. Кожен підхід має свої переваги.

Конвеєр

Головною перешкодою високої швидкості виконання команд є їх виклик з пам'яті. Для вирішення цієї проблеми можна викликати команди з пам'яті заздалегідь і зберігати в спеціальному наборі регістрів. Ця ідея використовувалася ще в 1959 році при розробці комп'ютера Stretch компанії IBM, а набір регістрів був названий буфером вибірки з попередженням.

Таким чином, коли була потрібна певна команда, вона викликалася прямо з буфера, а звернення до пам'яті не відбувалося.

Насправді процес вибірки з попередженням поділяє виконання команди на два етапи: виклик і власне виконання команди. Ідея конвеєра ще більше просунула цю стратегію вперед. Тепер команда поділялась вже не на два, а на кілька етапів, кожен з яких виконувався певною частиною апаратного забезпечення, причому всі ці частини могли працювати паралельно.

На рис. 4.1 , а зображений конвеєр з 5 блоків, які називаються стадіями. Стадія С1 (блок С1) викликає команду з пам'яті і поміщає її в буфер, де вона зберігається до тих пір, поки не буде потрібна. Стадія С2 (блок С2) декодує цю команду, визначаючи її тип і тип операндів, над якими вона буде проводити певні дії. Стадія С3 (блок С3) визначає місце знаходження операндів і викликає їх або з регістрів, або з пам'яті. Стадія С4(блок С4) виконує команду, зазвичай шляхом проведення операндів через тракт даних.

І нарешті, стадія С5 записує результат назад в потрібний регістр.

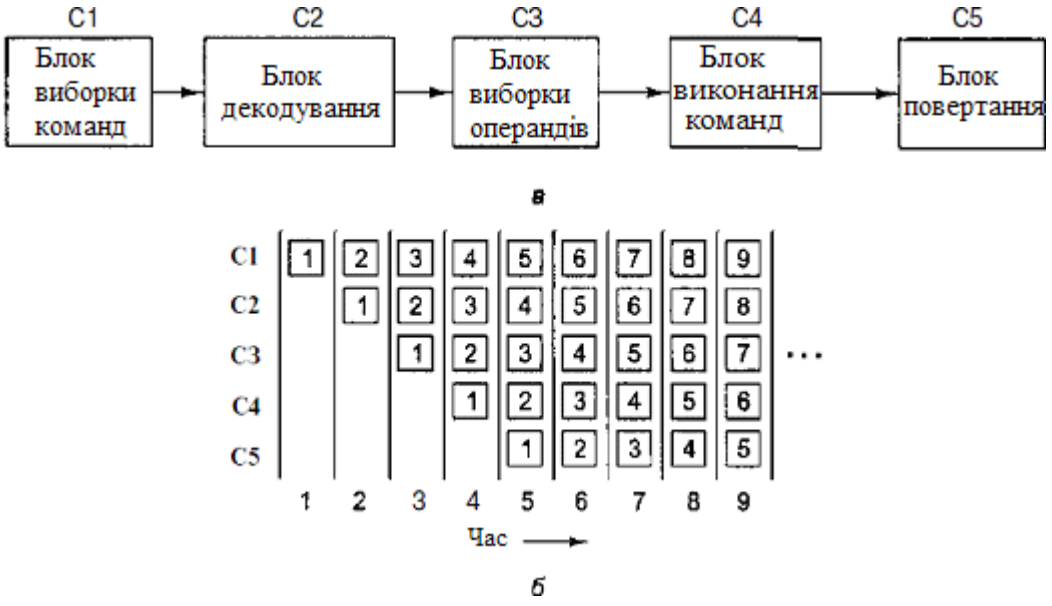


Рис. 4.1 Конвеєр з 5 стадій (а); стан кожної стадії в залежності від кількості пройдених циклів (б). Показано 9 циклів

На рис. 4.1, б ми бачимо, як діє конвеєр в часі. Під час циклу 1 стадія С1 працює над командою 1, викликаючи її з пам'яті. Під час циклу 2 стадія

С2 декодує команду 1, в той час як стадія С1 викликає з пам'яті команду 2. Під час циклу 3 стадія СЗ викликає операнди для команди 1, стадія С2 декодує команду 2, а стадія С1 викликає третю команду. Під час циклу 4 стадія С4 виконує команду 1, СЗ викликає операнди для команди 2, С2 декодує команду 3, а С1 викликає команду 4. Нарешті, під час п'ятого циклу С5 записує результат виконання команди 1 назад в регістр, тоді як інші стадії працюють над наступними командами.

Припустимо, що час циклу у цієї машини 2 нс. Тоді для того, щоб одна команда пройшла через весь конвеєр, потрібно 10 нс. На перший погляд може здатися, що такий комп'ютер може виконувати 100 млн. команд за секунду, в дійсності ж швидкість його роботи набагато вища. Під час кожного циклу (2 нс) завершується виконання однієї нової команди, тому машина виконує не 100 млн., а 500 млн. команд за секунду.

Конвеєри дозволяють знайти компроміс між часом очікування (скільки часу займає виконання однієї команди) і пропускною спроможністю процесора (скільки мільйонів команд в секунду виконує процесор). Якщо час циклу становить Т нс, а конвеєр містить n стадій, то час очікування складе nT нс, а пропускна здатність - 1000 / Т млн. команд в секунду.

Суперскалярні архітектури

Один конвеєр - добре, а два - ще краще. Одна з можливих схем процесора з подвійним конвеєром показана на рис. 4.2. В основі розробки лежить конвеєр, зображений на рис. 4.1. Тут загальний відділ виклику команд бере з пам’яті одночасно по дві команди і завантажує кожну з них в один з конвеєрів. Кожен конвеєр містить АЛП для паралельних операцій. Щоб виконуватися паралельно, дві команди не повинні конфліктувати при використанні ресурсів (наприклад, регістрів), і жодна з них не повинна залежати від результату виконання іншої. Як і у випадку з одним конвеєром, або компілятор повинен стежити, щоб не виникало неприємних ситуацій (наприклад, коли апаратне забезпечення видає некоректні результати, якщо команди несумісні), або ж конфлікти виявляються і усуваються прямо під час виконання команд завдяки використанню додаткового апаратного забезпечення.

Спочатку конвеєри (як подвійні, так і одинарні) використовувалися тільки у комп'ютерах RISC. У 386-го і його попередників їх не було. Конвеєри в процесорах компанії Intel з'явилися тільки починаючи з 486-ї моделі. 486-й процесор містив один конвеєр, а Pentium - два конвеєри з п'яти стадій. Схожа схема зображена на рис. 4.2, але поділ функцій між другою і третьою стадіями (вони називалися декодування 1 і декодування 2) було трохи іншим. Головний конвеєр (u-конвеєр) міг виконувати довільні команди. Другий конвеєр (v-конвеєр) міг виконувати тільки прості команди з цілими числами, а також одну просту команду з плаваючою точкою (FXCH).

Є складні правила визначення, чи є пара команд сумісними для того, щоб виконуватися паралельно. Якщо команди, що входять в пару, були складними або несумісними, виконувалася тільки одна з них (в і-конвеєрі). Та, що залишилася. друга команда, складала потім пару з наступною командою. Команди завжди виконувалися по порядку.

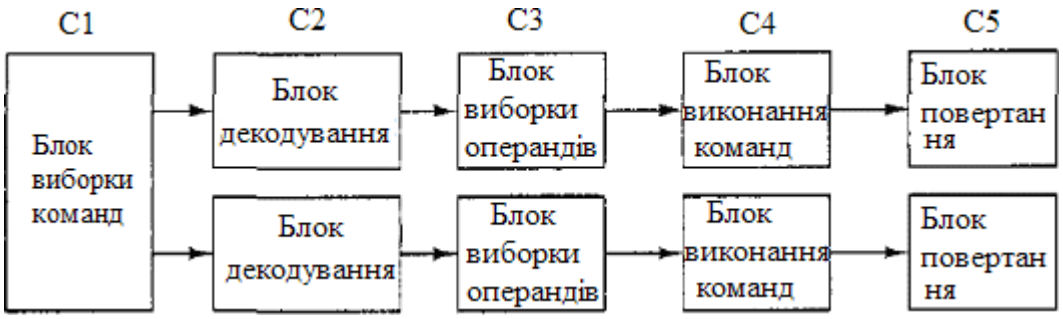


Рис. 4.2 Подвійний конвеєр з п'яти стадій із загальним відділом виклику команд

Таким чином, Pentium містив особливі компілятори, які об'єднували сумісні команди в пари і могли породжувати програми, що виконуються швидше, ніж в попередніх версіях. Вимірювання показали, що програми, які виробляють операції з цілими числами, на комп'ютері Pentium виконуються майже в два рази швидше, ніж на 486-му, хоча у нього така ж тактова частота. Поза всякими сумнівами, перевага в швидкості з'явилася завдяки другому конвеєру.

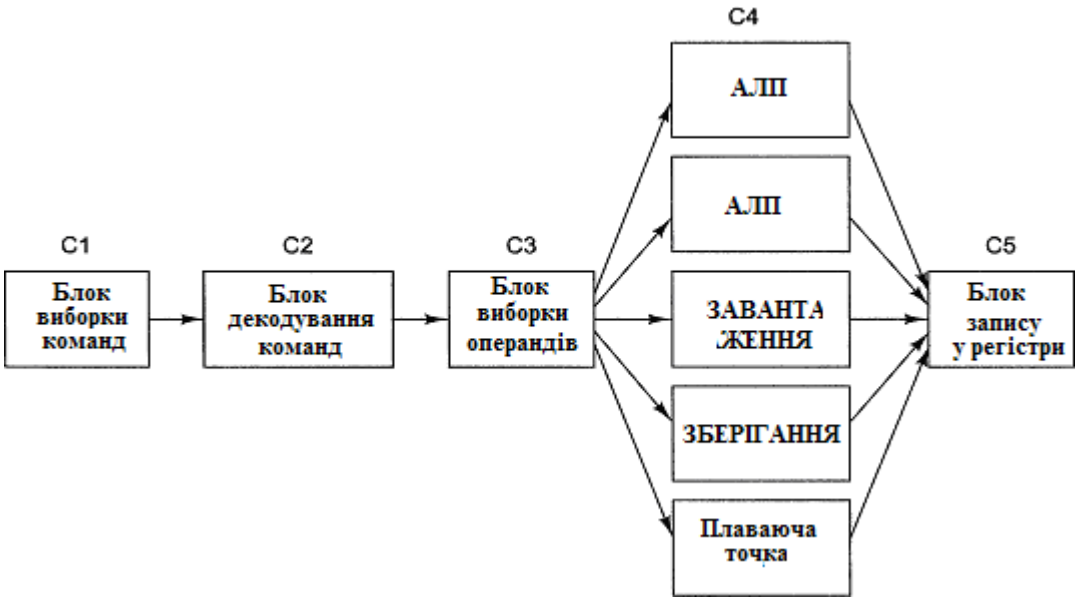


Рис.4.3 . Суперскалярний процесор з п'ятьма функціональними блоками

Перехід до чотирьох конвеєрів можливий, але це потребувало б створення громіздкого апаратного забезпечення. Замість цього використовується інший підхід. Основна ідея - один конвеєр з великою кількістю функціональних блоків, як показано па рис. 4.3 PentiumII, наприклад, має подібну структуру. У 1987 році для позначення цього підходу був введений термін суперскалярна архітектура. Однак подібна ідея знайшла втілення ще більше 30 років тому в комп'ютері CDC 6600. CDC 6600 викликав команду з пам'яті кожні 100 нс і поміщав її в один з 10 функціональних блоків для паралельного виконання. Поки команди виконувалися, центральний процесор викликав наступну команду.

Відзначимо, що стадія 3 випускає команди значно швидше, ніж стадія 4 здатна їх виконувати. Якби стадія 3 випускала команду кожні 10 нс, а усі функціональні блоки виконували б свою роботу також за 10 нс, то на четвертій стадії завжди функціонував би тільки один блок, що зробило б саму ідею конвеєра безглуздою. Насправді більшості функціональних блоків четвертої стадії для виконання команди потрібно значно більше часу, ніж займає один цикл (це блоки доступу до пам'яті і блок виконання операцій з

плаваючою точкою). Як видно з рис. 4.3, на четвертій стадії може бути кілька АЛП.

4.2 Паралелізм на рівні процесорів

Попит на комп'ютери, що працюють усе з більш і більш високою швидкістю, не припиняється. Астрономи хочуть з'ясувати, що сталося в першу мікросекунду після великого вибуху, економісти хочуть змоделювати всю світову економіку, підлітки хочуть грати в 3D інтерактивні ігри зі своїми віртуальними друзями через Інтернет. Швидкість роботи процесорів підвищується, але у них постійно виникають проблеми із швидкістю передачі інформації, оскільки швидкість поширення електромагнітних хвиль в мідних проводах і світла в оптико-волоконних кабелях як і раніше залишається 20 см / нс, незалежно від того, наскільки розумні інженери компанії Intel. Крім того, чим швидше працює процесор, тим сильніше він нагрівається, і потрібно оберігати його від перегріву.

Паралелізм на рівні команд допомагає в якоюсь мірою, але конвеєр та суперскалярна архітектура зазвичай збільшують швидкість роботи всього лише у 5-10 разів. Щоб поліпшити продуктивність в 50, 100 і більше разів, необхідно розробити комп'ютери з декількома процесорами.

Матричні комп'ютери

Багато задач в фізичних і технічних науках містять вектори, в іншому випадку вони мали б дуже складну структуру. Часто одні і ті ж обчислення виконуються над різними наборами даних в один і той же час. Структура цих програм дозволяє підвищувати швидкість роботи завдяки паралельному виконанню команд. Існує два методи, які використовуються для швидкого виконання великих наукових програм. Хоча обидві схеми у багатьох відношеннях схожі, одна з них вважається розширенням одного процесора, а інша - паралельним комп'ютером.

Матричний (масивно-паралельний) процесор (array processor) складається з великої кількості подібних процесорів, які виконують одну й ту ж послідовність команд, які відносяться до різних наборів даних. Першим в світі таким процесором був ILLIACIV (Університет Іллінойсу). Він

зображений на рис. 4.6. Спочатку передбачалося сконструювати машину, що складається з чотирьох секторів, кожен з яких містить решітку 8x8 елементів процесор / пам'ять. Для кожного сектора був один блок контролю. Він розсилав команди, які виконувалися усіма процесорами одночасно, при цьому кожен процесор використовував свої власні дані зі своєї власної пам'яті (завантаження даних проводилось під час ініціалізації). Через дуже високу вартість був побудований тільки один такий сектор, але він міг виконувати 50 млн. операцій з плаваючою точкою за секунду. Якби при створенні машини використали чотири сектори, то вона могла б виконувати 1 млрд. операцій з плаваючою точкою за секунду, і потужність такої машини в два рази перевищувала б потужність комп'ютерів усього світу.

Для програмістів векторний процесор (vector processor) дуже схожий на матричний процесор (array processor). Як і матричний, він дуже ефективний при виконанні послідовності операцій над парами елементів даних. Але, на відміну від першого (array processor), всі операції додавання виконуються в одному блоці підсумовування, який має конвеєрну структуру. Компанія Cray Research, засновником якої був Сеймур Крей, випустила багато векторних процесорів, починаючи з моделі Cray-1 (1974) і до цього дня. Cray Research входить до складу SGI.

Рис.4.6 Массивно-параллельный процессор ILLIAC IV

Обидва типи процесорів працюють з масивами даних. Обидва вони виконують одні і ті ж команди, які, наприклад, попарно складають елементи для двох векторів. Але якщо у матричного процесора (array processor) є стільки ж підсумкових пристроїв, скільки елементів в масиві, векторний процесор (vector processor) містить векторний регістр, який складається з набору стандартних регістрів.

Ці регістри послідовно завантажуються з пам'яті за допомогою однієї команди. Команда складання попарно складає елементи двох таких векторів, завантажуючи їх з двох векторних регістрів в пристрій, що підсумовує, з конвеєрною структурою. В результаті із підсумовуючого пристрою виходить інший вектор, який або поміщається в векторний регістр, або відразу використовується в якості операнда при виконанні іншої операції з векторами.

Матричні процесори в нащ час не випускаються, але принцип, на якому вони основані, як і раніше актуальний. Аналогічна ідея застосовується в наборах MMX- і SSE-команд процесорів Pentium 4, і вона успішно вирішує завдання прискореного виконання мультимедійних програм. В цьому відношенні комп'ютер ILLIAC IV можна вважати одним з прабатьків процесора Pentium 4.

Мультипроцесори

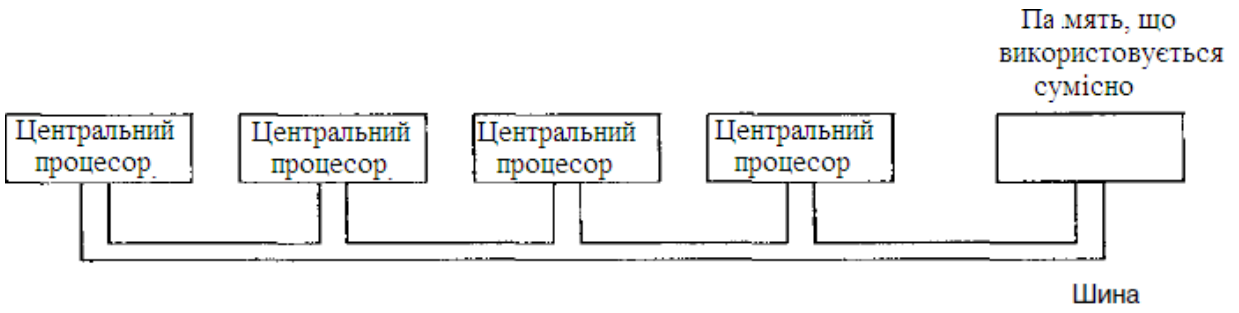
Елементи матричного процесора пов'язані між собою, оскільки їх роботу контролює один блок управління. Система декількох паралельних процесорів, які поділяють загальну пам'ять, називається мультипроцесором.

Оскільки кожен процесор може записувати або зчитувати інформацію із будб-яка частини пам'яті, їх робота повинна узгоджуватися програмним забезпеченням, щоб не допустити будь-яких перетинів.

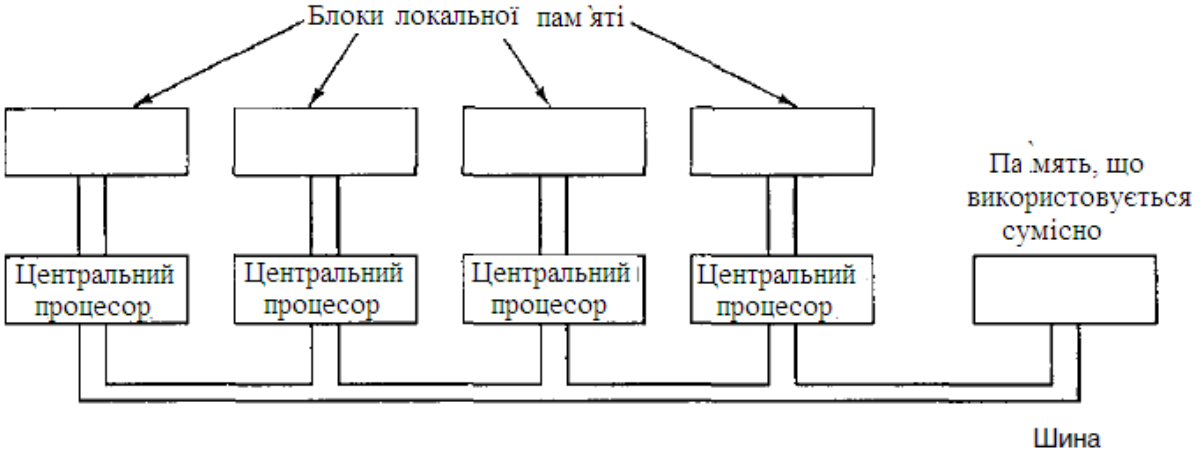
Можливі різні способи втілення цієї ідеї. Найпростіший з них наявність однієї шипи, що з'єднує кілька процесорів і одну загальну пам'ять.

Схема такого мультипроцесора показана на рис. 4.7 , а. Такі системи виробляють багато компаній. Неважко зрозуміти, що при наявності великої кількості швидко працюючих процесорів, які постійно намагаються отримати доступ до пам'яті через одну й ту ж саму шину, будуть виникати конфлікти.

Щоб вирішити цю проблему і збільшити продуктивність комп'ютера, були розроблені різні моделі. Одна з них зображена на рис.4.7, б. В такому комп'ютері кожен процесор має свою власну локальну пам'ять, яка недоступна для інших процесорів. Ця пам’ять використовується для програм і даних, які не потрібно розділяти між декількаами процесорами. При доступі до локальної пам'яті головна шина не використовується , і, таким чином, потік інформації в цій шині знижується. Можливі й інші варіанти вирішення проблеми (наприклад, кеш-пам'ять).



а



б

Рисунок. 4.7 Мультипроцесор з однією шиною і загальною пам'яттю (а); мультипроцесор, в якому для кожного процесора є власна локальна пам'ять (б)

Мультипроцесори мають перевагу перед іншими видами паралельних комп'ютерів, оскільки з єдиною розділеною пам'яттю дуже легко працювати. Наприклад, уявімо, що програма шукає ракові клітини на зробленому через мікроскоп знімку тканини. Фотографія в цифровому вигляді може

зберігатися в загальній пам'яті, при цьому кожен процесор обстежує якусь певну область фотографії. Оскільки кожен процесор має доступ до загальної пам'яті, обстеження клітини, яке починається в одній області і триває в іншій, не становить труднощів.

Мультикомп'ютери

Мультипроцесори з невеликим числом процесорів (<256) сконструювати досить легко, а от створення великих мультипроцесорів становить деякі труднощі. Складність полягає в тому, щоб зв'язати всі процесори із загальною пам'яттю. Тому багато розробників просто відмовились від ідеї розділяти пам'ять і стали створювати системи, що складаються з більшої кількості взаємопов'язаних комп'ютерів, у кожного з яких є своя власна пам'ять, а загальної пам'яті немає. Такі системи називаються мультикомп'ютерами. У них процесори є слабо пов'язаними, на противагу сильно пов'язаним процесорам в мультипроцесорних системах.

Процесори мультікомпьюиера відправляють один одному послання (це схоже на електронну пошту, але набагато швидше). Кожен комп'ютер не обов'язково зв'язувати з усіма іншими, тому зазвичай в якості топологій використовуються дво- та тривимірні решітки, а також дерева і кільця. Хоча щоб послання могли дійти до місця призначення, вони повинні проходити через один або кілька проміжних комп'ютерів, все одно час передачі займає всього кілька мікросекунд. Зараз вже працюють мультикомп'ютери, що містять близько 10 000 процесорів.

Оскільки мультипроцесори легше програмувати, а мультикомпьютери конструювати, виникла ідея створення гібридних систем, які містили б в собі переваги обох видів машин. Такі комп'ютери емулюють розділену пам'ять, хоча фізично такий блок не представлений.

Суперскалярні процесори

Оскільки можливості щодо вдосконалення елементної бази вже практично вичерпані, подальше підвищення продуктивності ОМ лежить в площині архітектурних рішень. Як уже зазначалося, один з найбільш ефективних підходів в цьому плані - введення в обчислювальний процес різних рівнів паралелізму. Раніше розглянутий конвеєр команд - типовий

приклад такого підходу. Тим же цілям служать і арифметичні конвеєри, де конвеєеризації піддається процес виконання арифметичних операцій. Додатковий рівень паралелізму реалізується в векторних і матричних процесорах, але тільки при обробці багатокомпонентних операндів типу векторів і масивів. Тут висока швидкодія досягається за рахунок одночасної обробки всіх компонентів вектора або масиву, проте подібні операнди характерні лише для досить вузького кола вирішуваних завдань. Основний обсяг обчислювального навантаження зазвичай припадає на скалярні обчислення, тобто на обробку одиночних операндів, таких, наприклад, як цілі числа. Для подібних обчислень додатковий паралелізм реалізується значно складніше, але тим не менше він можливий. Прикладом можуть служити суперскалярні процесори.

Суперскалярним (цей термін вперше був використаний в 1987 році) називається центральний процесор (ЦП), який одночасно виконує більш ніж одну скалярну команду. Це досягається за рахунок включення до складу ЦП кількох самостійних функціональних (виконавчих) блоків, кожен з яких відповідає за свій клас операцій і може бути присутнім в процесорі в декількох примірниках. Так, в процесорі Pentium III блоки цілочисельної арифметики і операцій з плаваючою точкою дубльовані, а в мікропроцесорах Pentium 4 і Athlon - троєровані. Структура типового суперскалярного процесора показана на рис. 4.8 .

Процесор містить в собі шість блоків: вибірки команд, декодування команд, диспетчеризації команд, розподілу команд по функціональних блоках, блок виконання і блок поновлення стану.

Блок вибірки команд витягує команди з основної пам'яті через кеш-пам'ять команд. Цей блок зберігає кілька значень лічильника команд і обробляє команди умовного переходу.

Блок декодування розшифровує код операції, що міститься у витягнутих з кеш-пам'яті командах. У деяких суперскалярних процесорах, наприклад в мікропроцесорах фірми Intel, блоки вибірки і декодування суміщені.

Блоки диспетчеризації і розподілу взаємодіють між собою і в сукупності відіграють в суперскалярному процесорі роль контролера трафіка. Обидва блоки зберігають черги декодованих команд.

Черга блоку розподілу часто розосереджується по кількох самостійних буферах - накопичувачам команд або схемам резервування (reservation station), - призначеним для зберігання команд, які вже декодовані, але ще не виконані.

Кожен накопичувач команд пов'язаний зі своїм функціональним блоком (ФБ), тому число накопичувачів дорівнює числу ФБ, але якщо в процесорі використовується кілька однотипних ФБ, то їм надається загальний накопичувач. По відношенню до блоку диспетчеризації накопичувачі команд виступають в ролі віртуальних функціональних пристроїв. Обидва види черг показані на рис. 4.9. У деяких суперскалярних процесорах вони об'єднані в єдину чергу.

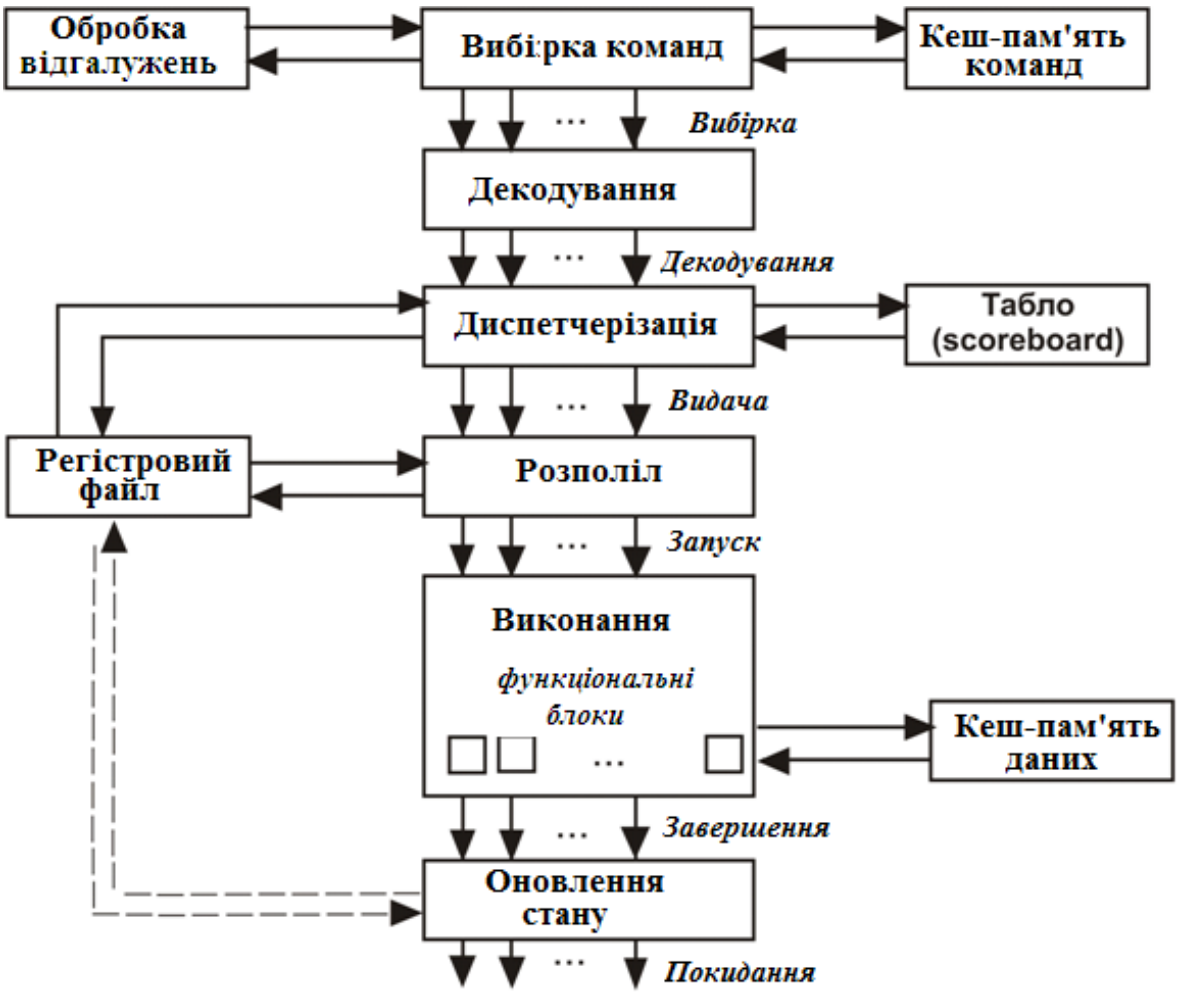


Рисунок 4.8 – Архітектура суперскалярного процессора

На додаток до черги, блок диспетчеризації зберігає також список вільних функціональних блоків, званий табло (scoreboard). Табло використовується для відстеження стану черги розподілу. Один раз за цикл блок диспетчеризації витягує команди зі своєї черги, зчитує з пам'яті або регістрів операнди цих команд, після чого, в залежності від стану табло, поміщає команди і значення операндів в чергу розподілу. Ця операція називається видачею команд.

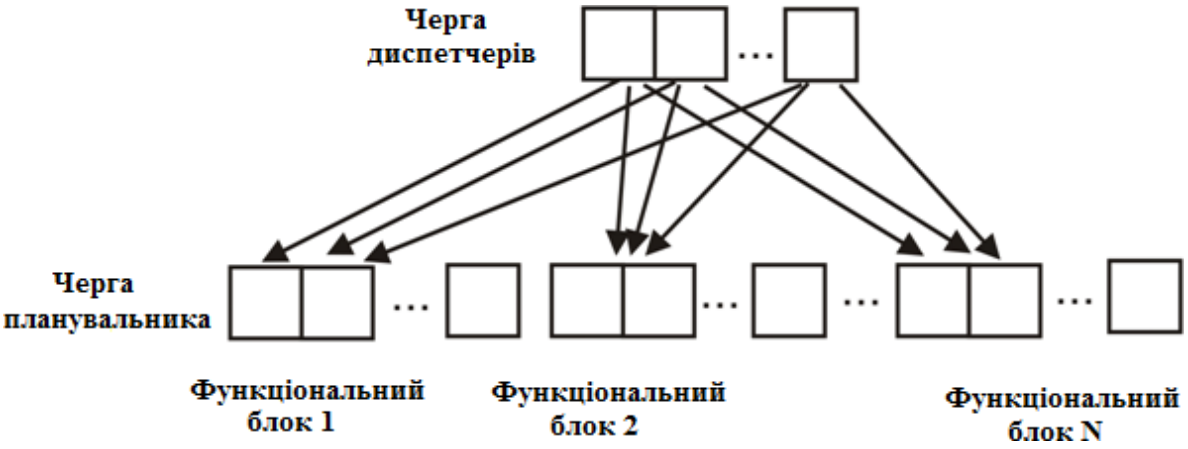


Рисунок 4.9 – Черга диспетчеризації та розподілу

Блок розподілу в кожному циклі перевіряє кожну команду в своїх чергах на наявність всіх необхідних для її виконання операндів і при позитивній відповіді починає виконання таких команд у відповідному функціональному блоці.

Блок виконання складається з набору функціональних блоків. Прикладами ФБ можуть служити цілочисельні операційні блоки, блоки множення і складання з плаваючою комою, блок доступу до пам'яті. Коли виконання команди завершується, її результат записується і аналізується блоком поновлення стану, який забезпечує облік отриманого результату тими командами в чергах розподілу, де цей результат є одним з операндів.

Як було зазначено раніше, суперскалярність передбачає паралельну роботу максимального числа виконавчих блоків, що можливо лише при одночасному виконанні декількох скалярних команд. Остання умова добре

поєднується з конвеєрною обробкою, при цьому бажано, щоб в суперскалярному процесорі було кілька конвеєрів, наприклад два або три.

Подібний підхід реалізований в процесорі Intel Pentium, де є два конвеєри, кожен зі своїм АЛП (рис. 4.10). Відзначимо, що тут, на відміну від стандартного конвеєра, в кожному циклі необхідно робити вибірку більш ніж однієї команди. Відповідно, пам'ять ОМ повинна допускати одночасне зчитування кількох команд і операндів, що найчастіше забезпечується за рахунок її модульної побудови.

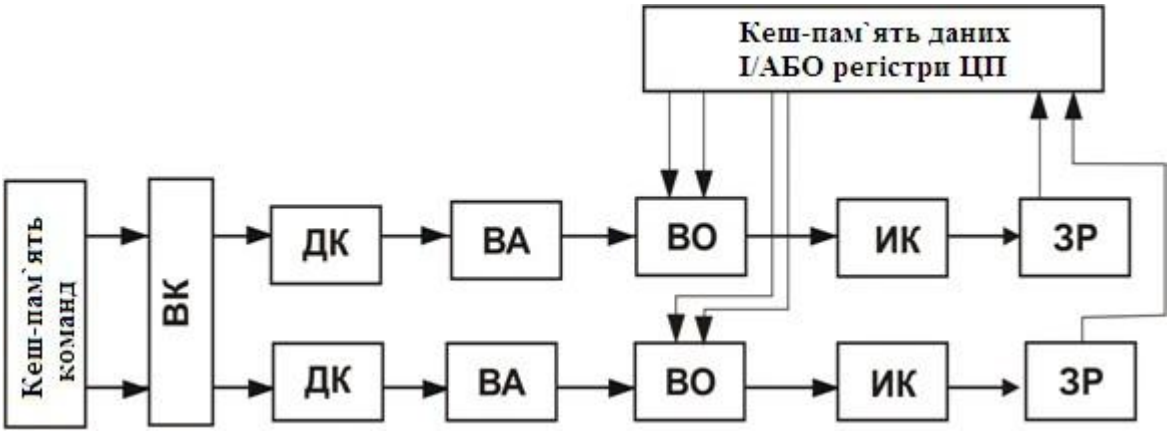


Рисунок 4.10 – Суперскалярний процесор з двома конвеєрами

Більш інтегрований підхід до побудови суперскалярного конвеєра показаний на рис. 4.11. Тут блок вибірки (ВК) витягує з пам'яті більше однієї команди і передає їх через ступені декодування команди і обчислення адрес операндів в блок вибірки операндів (ВО). Коли операнди стають доступними, команди розподіляються по відповідних виконавчих блоках.

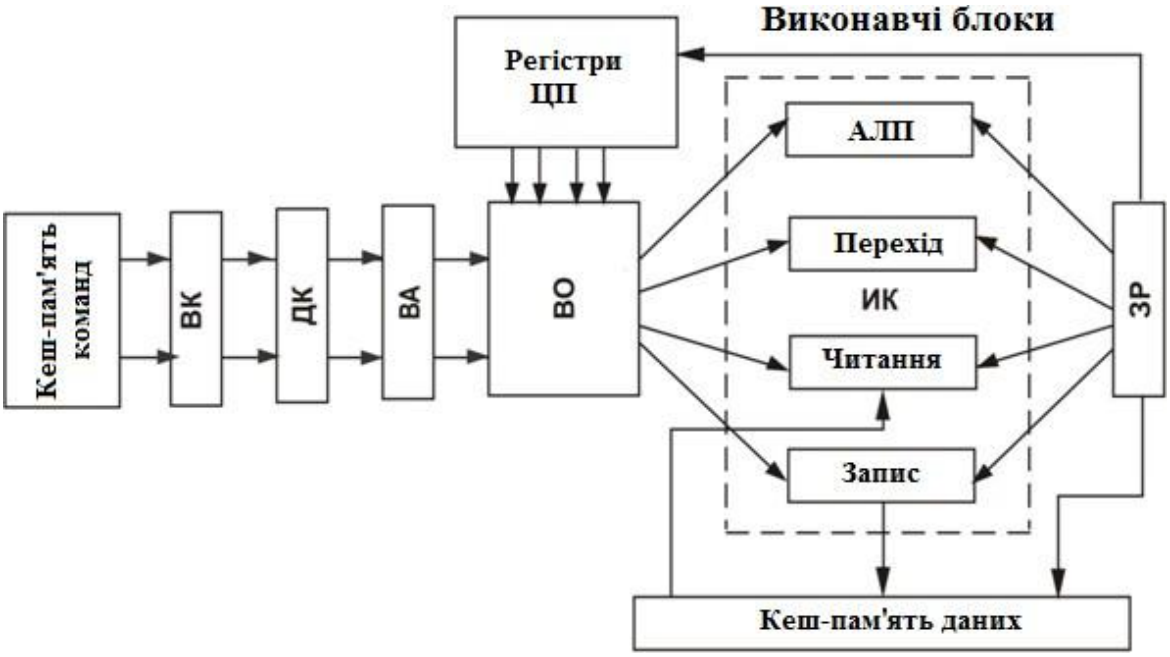


Рисунок 4.11 – Суперскалярний конвеєр зі спеціалізованими виконавчими блоками

5 УНІВЕРСАЛЬНІ МІКРОПРОЦЕСОРИ. СХЕМИ ПІДТРИМКИ МП НА СИСТЕМНИХ ПЛАТАХ

Мікропроцесор - це процесор, виконаний у вигляді великої інтегральної схеми і розміщений у герметичному корпусі. В основі будь-якої ПЕОМ лежить використання мікропроцесорів.

Мікропроцесор є «мозком» комп'ютера. Він здійснює виконання програм, що працюють на комп'ютері, і керує роботою інших пристроїв комп'ютера. Основними характеристиками мікропроцесора є швидкодія і розрядність. Швидкодія - це число виконуваних операцій в секунду.

Розрядність характеризує обсяг інформації, який мікропроцесор обробляє за одну операцію: 8-розрядний процесор за одну операцію обробляє 8 біт інформації, 32-розрядний - 32 біта. Швидкість його роботи багато в чому визначає швидкодія комп'ютера. У IBM PC використовуються мікропроцесори, розроблені фірмою Intel, або сумісні з ними процесори інших фірм. Перший мікропроцесор Intel 4004 був створений в 1971 році.

Процесорів в комп'ютері багато, крім центрального процесора в комп'ютері знаходяться і інші процесори. Процесор - це не тільки скопище транзисторів, а ціла система безлічі важливих пристроїв. На будь-якому процесорному кристалі знаходяться: власне процесор-головний обчислювальний пристрій, співпроцесор -спец. блок, який застосовується для особливо точних і складних розрахунків, кеш-пам'ять першого рівня -невелика надшвидка пам'ять, призначена для проміжних результатів обчислень, кеш-пам'ять другого рівня - ця пам'ять трохи повільніша, зате більша.

5 .1 Характеристики мікропроцесорів

Мікропроцесори відрізняються один від одного двома характеристиками: типом (моделлю) і тактовою частотою. Однакові моделі мікропроцесорів можуть мати різну тактову частоту - чим вища тактова частота, тим вища продуктивність і ціна мікропроцесора .

Тактова частота показує, скільки елементарних операцій (тактів) мікропроцесор виконує в одну секунду. Тактова частота вимірюється в

мегагерцах (МГц). Тактова частота позначається цифрою в назві процесора. Найбільшою популярністю користуються процесори з частотою від 800 до 1200 МГц.

Кеш-пам'яті в процесорі є двох видів. Найшвидша - кеш-пам'ять першого рівня. Існує ще трохи менш швидка, але об'ємна кеш-пам'ять другого рівня.

Перелічимо основні функції мікропроцесора:

• вибірка команд із ОЗП;

• декодування команд (тобто визначення призначення команди, способу її виконання і адрес операндів);

• виконання операцій, закодованих в командах;

• управління пересиланням інформації між своїми внутрішніми регістрами, оперативною пам'яттю і зовнішніми (периферійними) пристроями;

• обробка внутрішньопроцесорних і програмних переривань;

• обробка сигналів від зовнішніх пристроїв та реалізація відповідних переривань;

• управління різними пристроями, що входять до складу комп'ютера.

5.2 Структура і особливості роботи мікропроцесорів

Структурна схема базової моделі МП фірми Intel наведена на рис. 5.1 Умовно МП можна розділити на 2 частини - виконавчий блок (ExecutionUnit - EU) і пристрій сполучення з системною магістраллю (BusInterfaceUnit - BIU).

У виконавчому блоці знаходяться: арифметичний блок і РОН. Арифметичний блок включає АЛП, допоміжні регістри зберігання операндів і регістр прапорів.

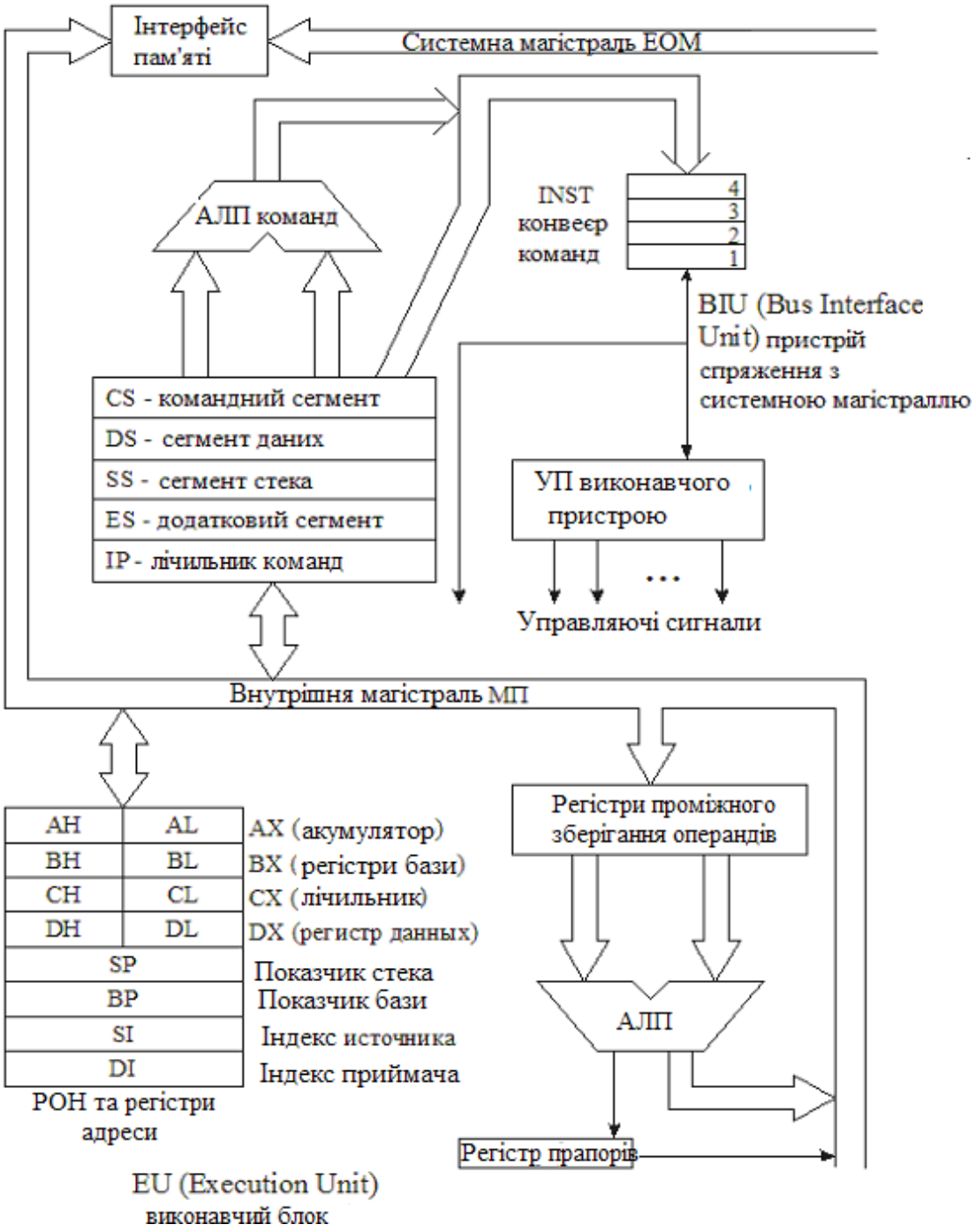


Рисунок 5.1 – Структурна схема мікропроцесора

Вісім регістрів виконавчого блоку (AX, BX, CX, DX, SP, BP, SI, DI), що мають довжину, рівну машинному слову, діляться на 2 групи.

Першу групу складають РОН, AX, BX, CX і DX. Кожен з них являє собою реєстрову пару, складену з двох регістрів довжиною в 0,5 машинного слова. Акумулятор, або регістр AX, складається з регістрів AH і AL. Регістр бази (BaseRegister) BX - з BH і BL. Регістр даних (DataRegister) DX - DH і DL. Кожен з коротких регістрів може використовуватися самостійно або в складі реєстрової пари. Умовні назви - (акумулятор, регістр бази ...) не обмежують застосування цих регістрів. Вони говорять про найбільш часте використанні їх в тій чи іншій команді.

Другу групу складають адресні регістри SP, BP, SI і DI. У старших моделях кількість адресних регістрів збільшено. Ці регістри активно використовуються за функціональним призначенням, і в інших цілях їх застосовувати не рекомендується.

РОН BX часто використовується в якості адресного регістра.

Пристрій сполучення з системною магістраллю містить керуючі регістри, конвеєр команд, пристрій управління виконавчим блоком МП і інтерфейс пам'яті (що з'єднує внутрішню магістраль МП з системною магістраллю ПЕОМ).

Керуючі регістри BIU: CS - покажчик командного сегмента, DS - покажчик сегмента даних, SS - покажчик сегмента стека, ES - покажчик додаткового сегмента та ін. - служать для визначення фізичних адрес операндів і команд ОЗП. Регістр IP (InstructionPointer) є покажчиком адреси команди, яка буде вибиратися в конвеєрі команд в якості чергової команди. Конвеєр команд МП зберігає кілька команд, що дозволяє при виконанні лінійних програм поєднати підготовку чергової команди з виконанням поточної.

До керуючих регістрів МП відноситься і регістр прапорів, кожен розряд якого має певне призначення. Зазвичай розряди регістра прапорів встановлюються апаратно при виконанні чергової операції в залежності від одержуваного результату в АЛП. При цьому фіксуються такі властивості отриманого результату, як нульовий результат, негативне число, переповнення розрядної сітки АЛП і т.д. Але деякі розряди регістра прапорів можуть встановлюватися за спеціальними командами. Деякі розряди мають чисто

службове призначення (наприклад, зберігають розряд, який випав з АЛП під час зсуву) або є резервними (тобто не використовуються в даному МП, але плануються використовуватися при модернізації МП).

Всі прапори молодшого байта регістра встановлюються арифметичними або логічними операціями МП. Всі прапори старших байтів, за винятком прапора переповнення, встановлюються програмним шляхом, для цього в МП є команди установки прапорів (STC, STD, STI), скидання (CLC, CLD, CLI), інвертування (CMC).

5.3 Алгоритм роботи мікропроцесора

Команди в мікропроцесорі (МП) , як відомо, виконуються по машинних циклах (МЦ), тривалість яких складає 3-5 машинних тактів. Число МЦ та загальна кількість тактів, необхідна для виконання команди, визначається типом команди, що виконується. При цьому число МЦ дорівнює сумарному числу звертань до оперативної пам'яті (ОП) і пристроїв введення-виведення (ПВВ), необхідних для вибірки та виконання даної команди.

Перший МЦ (М1) завжди є циклом вибірки першого байта команди та складається з 4-5 тактів. Три наступних МЦ (М2, М3, М4) виконуються завжди за три такти, а п'ятий МЦ – за 3-5 тактів.

Кожний МЦ супроводжується видаванням сигналу синхронізації (СИНХ) у такті Т1 (рис. 5.2). При цьому на ШД видається байт стану (БС), що визначає дії, які будуть виконуватися в даному МЦ. По збігу сигналів Ф1 та СИНХ байт стану фіксується в зовнішньому 8-розрядному регістрі та використовується в мікропроцесорній системі (МПС) для керування циклом. Змістовий граф алгоритму роботи керуючого автомата МП показаний на рис. 5.3.

З такту Т1 МП завжди переходить у такт Т2, в якому аналізуються вхідні сигнали "Готовність" (ГТ), "Захват" (ЗХ) та сигнал "Підтвердження зупину" (ПЗУП) з байта стану. Якщо ПЗУП=1, то МП переходить у стан зупину СЗУП, вийти з якого можна тільки по початковому скиданню, режиму прямого доступу до пам'яті (ПДП) чи перериванню (сигналами Скидання, ЗХ, "Запит переривання" – ЗПР). Перехід в СЗУП відбувається в другому циклі команди HLT (рис. 5.2).

Якщо $\overline{ПОСТ}\&\overline{ГТ}=1$, то МП переходить у стан очікування СОЧК, з якого він може вийти по сигналу ГТ=1.

Якщо $\overline{ПОСТ\&ГТ}=1$, то МП аналізує вхідний сигнал ЗХ. При ЗХ=0 виконується такт Т3, протягом якого відбувається введення чи виведення інформації з МП. Після завершення такту Т3 аналізується внутрішній сигнал кінця машинного циклу (КМЦ), з якого здійснюють подальші переходи МП. Такти Т4 та Т5, якщо вони в даному циклі присутні, використовують для завершення виконання команди за рахунок внутрішніх операцій МП (без звертання до пам'яті та інших пристроїв МПС).

Якщо ЗХ=1, то по фазі Ф2 такту Т2 цей сигнал фіксується на внутрішньому тригері захвату шин ТЗХ, що забезпечує перевід ШД і ША у високоімпедансний стан та перехід МП, після завершення поточного машинного циклу команди, у стан захвату шин СЗХ по одиничному стану ТЗХ.

Переведення ШД і ША в високоомний стан відбувається по фазі Ф2 відповідно у такті Т3 та в наступному за ним такті. Це переведення ініціюється одиничним вихідним сигналом підтвердження захвату шин – ПЗХ.

Вихід МП зі стану захвату відбувається по сигналу ЗХ=0, що скидає в 0 внутрішній тригер захвату шин, і МП розпочинає виконувати наступний цикл з такту Т1, так як при переводі ША і ШД у стан захвата поточний машинний цикл завжди завершується.

Після завершення циклу та виходу зі стану захвату, якщо воно було, відбувається аналіз внутрішнього сигналу кінця виконання команди (КВК). Якщо виконання поточної команди не скінчено, здійснюється перехід до такту Т1 наступного циклу даної команди. Якщо ж виконання команди завершено, то аналізується зовнішній сигнал запиту переривання (ЗПР) і внутрішній сигнал дозволу переривання (ДПР). Ці ж сигнали аналізуються і при стані зупину СЗУП, якщо сигнал ЗХ=0 (рис. 5.3).

Якщо ЗПР&РПР=0, то МП переходить на початок першого машинного циклу М1 чергової команди чи в стан зупин СЗУП. Якщо ж ЗПР& &РПР=1, то внутрішній тригер запиту переривання (ТПР) встановлюється в 1 (УТПР), і МП

переходить до виконання машинного циклу підтвердження переривання, що є першим циклом виконання команди RST (рестарт).

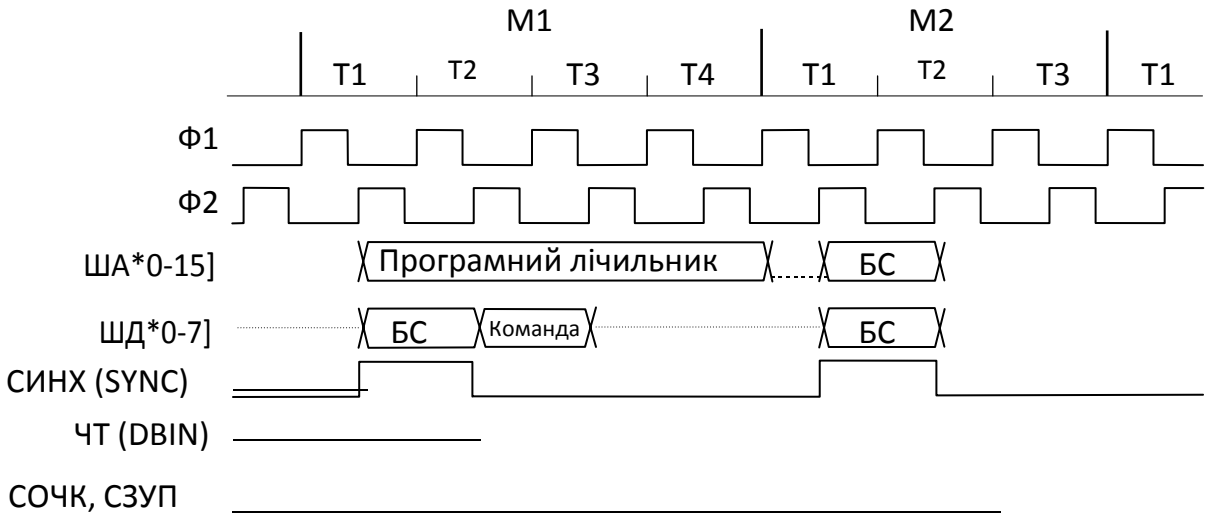


Рисунок. 5.2 – Часова діаграма роботи МП (команда HLT)

При цьому, якщо МП приступає до виконання переривання, знаходячись у СЗУП (рис. 5.3), то заздалегідь скидається в 0 керуючий сигнал підтвердження зупину ПЗУП з байта стану. Таким чином, МП переходить у режим переривання в останньому такті останнього машинного циклу поточної команди, тобто повністю завершивши її виконання. По сигналу Ф2 такту Т1 наступного МЦ вихідний сигнал ДПР (що керується також командами EI та DI) і тригер ТПР скидаються в 0, забороняючи тим самим нові переривання з одного і того ж запиту.

Зняття заборони на переривання у подальшому здійснюється програмно шляхом включення команди дозволу переривання EI в програму обробки (обслуговування) переривання, що дозволяє реалізувати вкладені переривання.

Нагадаємо, що перехід у режим прямого доступу до пам'яті ПДП, на відміну від режиму переривання, відбувається всередині команди, що виконується, по закінченні поточного машинного циклу. Це можливо, так як МП повністю від'єднується від системи, припиняючи виконання яких-небудь дій.

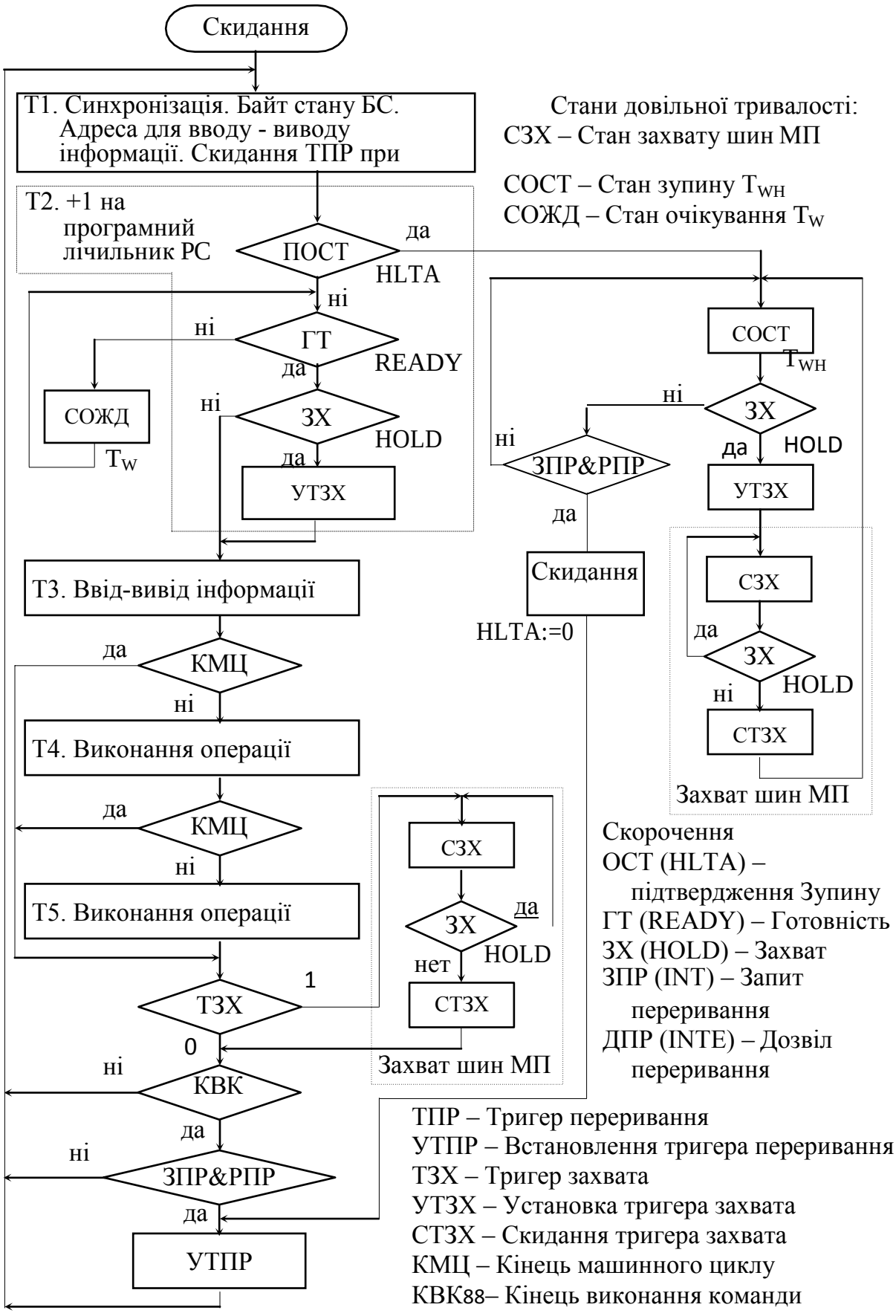


Рисунок 5.3 – Алгоритм роботи керуючого автомата МП

Запуск МП після ввімкнення напруги живлення і появи тактових сигналів Ф1 і Ф2 здійснюється подаванням 1 на вхід "СКД", внаслідок чого програмний лічильник ПЛ, регістр команд РГК та внутрішні тригери переривання ТПР, захвату ТЗХ і очікування ТОЧК скидаються в нульовий стан. Вміст регістрів загального призначення РЗП, акумулятора та регістра прапорів РП змінюється тільки у процесі виконання команд. По закінченню сигналу "СКД" МП починає виконувати цикл М1 і видає на ША нульову адресу.

5.4 Системні керуючі сигнали

Для нормальної роботи такої складної системи, якою є мікропроцесорна система (МПС), необхідний набір спеціальних керуючих сигналів, що організують взаємодію різноманітних периферійних пристроїв з МП. Як відомо, деякі сигнали керування генеруються безпосередньо самим МП у відповідь на зміну вхідних сигналів запиту обслуговування конкретних пристроїв, а також для обміну інформацією між ними. Наприклад, МП виробляє стробуючі сигнали запису (WRITE), читання (DBIN) з пам'яті чи пристроїв введення-виведення та синхронізації (SYNC), сигнали підтвердження захвату шин (HLDA) при запиті пристроїв прямого доступу до пам'яті чи дозволу переривання (INTE) при запиті переривання, а також сигнал підтвердження стану очікування (WAIT) при обміні МП з повільнодіючою периферією.

Однак зазначених сигналів явно недостатньо для нормального функціонування МП-системи. Тому додатково діють сигнали байта стану МП, значення яких визначають поточним машинним циклом. Необхідні розряди байта стану заводяться на відповідні входи схем сполуки (адаптерів) з пристроями введення-виведення, визначаючи тим самим режим їх роботи відповідно до поточного стану МП.

І, нарешті, поєднання деяких сигналів, що генеруються МП, та сигналів байта стану складають групу системних керуючих сигналів, що безпосередньо забезпечують приймання та передавання кодів між МП, пам'яттю та зовнішніми пристроями в певні інтервали часу у відповідності з змістовим графом керуючого автомата МП. Дані сигнали безпосередньо мікропроцесором не

виробляються. Для їх формування використовують сигнали приймання ЧТ (DBIN) та запису ЗП (WR) з МП та необхідні ознаки з байта стану – MEMR (Д7), OUT (Д4), INP (Д6), INTA (Д0). Інші ознаки байта стану можуть використовуватися при тестуванні МП.

П'ять системних керуючих сигналів формують у відповідності з нижчеподаними виразами:

$\overline{MEMR} = \overline{DBIN \& MEMR}$ – читання з пам'яті,

$\overline{MEMW} = \overline{\overline{WR} \& \overline{OUT}}$ – запис у пам'ять,

$\overline{I/OR} = \overline{DBIN \& INP}$ – введення інформації,

$\overline{I/OW} = \overline{\overline{WR} \& OUT}$ – виведення інформації,

$\overline{INTA} = \overline{DBIN \& INTA}$ – підтвердження переривання.

5.5 Схеми підтримки мп на системних платах ЦАП і АЦП

Цифро-аналогові перетворювачі (ЦАП, DAC - Digital-to-Analog Converter) і аналого-цифрові перетворювачі (АЦП, ADC -Analog-to-Digital Converter) головним чином застосовуються для сполучення цифрових пристроїв і систем із зовнішнім аналоговим сигналами, з реальним світом . При цьому АЦП перетворює аналогові сигнали у вхідні цифрові сигнали, що надходять на цифрові пристрої для подальшої обробки або зберігання, а ЦАП перетворює вихідні цифрові сигнали цифрових пристроїв в аналогові сигнали.

При цьому застосування ЦАП і АЦП постійно розширюється в міру переходу від аналогових пристроїв до цифрових пристроїв. У якості ЦАП і АЦП зазвичай застосовуються спеціалізовані мікросхеми, що випускаються багатьма вітчизняними і зарубіжними фірмами.

Призначення і види цифро-аналогових перетворювачів.

Цифро-аналоговим перетворювачем (ЦАП) називається електронний пристрій, призначений для перетворення цифрової інформації в аналогову. Вони використовуються для формування сигналу у вигляді напруги або струму, функціонально пов'язаного з керуючим кодом. У більшості випадків ця

функціональна залежність є лінійною. Найбільш часто ЦАП використовуються для сполучення пристроїв цифрової обробки сигналів з системами, що працюють з аналоговими сигналами. Крім цього, ЦАП використовуються в якості вузлів зворотного зв'язку в аналого-цифрових перетворювачів і в пристроях порівняння цифрових величин з аналоговими. Області застосування ЦАП досить широкі. Вони застосовуються в системах передачі даних, в вимірювальних приладах і випробувальних установках, в синтезаторах напруги і генераторах складних функцій, для формування зображень на екрані дисплеїв і ін. У зв'язку з цим розроблено і випускається велика кількість інтегральних мікросхем ЦАП.

Схеми ЦАП можна класифікувати за різними ознаками: принципом дії, виду вихідного сигналу, полярності вихідного сигналу, елементній базі і ін. За принципом дії найбільшого поширення набули ЦАП наступних видів: зі складанням струмів, з розподілом напруги і зі складанням напруг. У мікроелектронному виконанні застосовуються тільки перші два типи.

По виду вихідного сигналу ЦАП ділять на два види: з струмовим виходом і виходом по напрузі. Для перетворення вихідного струму ЦАП в напругу зазвичай використовуються операційні підсилювачі. За полярності вихідного сигналу ЦАП прийнято ділити на однополярні і двохполярні.

Керуючий код, що подається на вхід ЦАП, може бути різним: двійковим, двійковим-десятковим, Грея, унітарним і ін. Крім того, різними можуть бути і рівні логічних сигналів на вході ЦАП.

При формуванні вихідної напруги ЦАП під дією керуючого коду зазвичай використовуються джерела опорної напруги. Залежно від виду джерела опорного напруги ЦАП ділять на дві групи: з постійною опорною напругою і із змінною опорною напругою. Крім цього, ЦАП ділять за основними характеристиками: кількістю розрядів, швидкодією, точністю перетворення, споживаною потужністю.

Основні параметри ЦАП. Всі параметри ЦАП можна розділити на дві групи: статичні і динамічні. До статичних параметрів ЦАП відносять: роздільну здатність, похибка перетворення, діапазон значень вихідного сигналу, характеристики керуючого коду, зсув нульового рівня і деякі інші.

До динамічними показниками ЦАП прийнято відносити: час встановлення вихідного сигналу, граничну частоту перетворення, динамічну похибку.

Застосування ЦАПУ загальному випадку мікросхему ЦАП можна представити у вигляді блоку (рис. 5.4), що має кілька цифрових входів і один аналоговий вхід, а також аналоговий вихід.

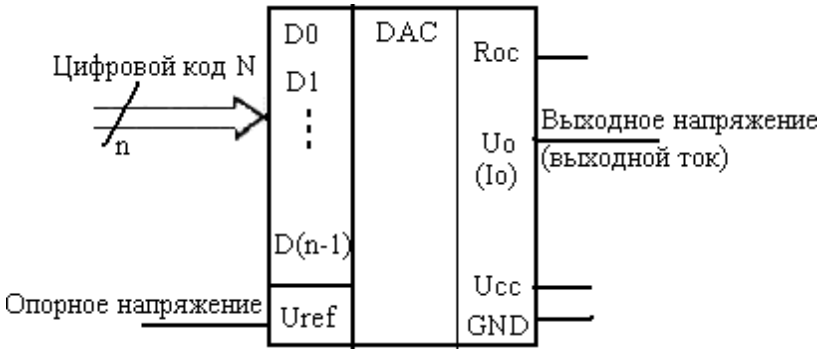


Рисунок 5.4 – Мікросхема ЦАП

На цифрові входи ЦАП подається n-розрядний код N, на аналоговий вхід-опорна напруга Uon (інше поширене позначення - UREF). Вихідним сигналом є напруга Івих (інше позначення - Uo) або ток Івих (інше позначення - Io). При цьому вихідний струм або вихідна напруга пропорційні вхідному коду і опорної напруги. Для деяких мікросхем опорна напруга повинне мати строго заданий рівень, для інших допускається змінювати його значення в широких межах, в тому числі і змінювати його полярність (позитивну на негативну і навпаки). ЦАП з великим діапазоном зміни опорного напруги називається (умножающим) ЦАП, так як його можна легко використовувати для множення вхідного коду на будь-яку опорну напругу.

Крім інформаційних сигналів мікросхеми ЦАП потребує також підключення одного або двох джерел живлення і загального проводу. Зазвичай цифрові входи ЦАП забезпечують сумісність зі стандартними виходами мікросхем ТТЛ.

У разі, коли ЦАП має струмовий вихід, його вихідний струм зазвичай перетворюється на вихідну напругу за допомогою зовнішнього операційного підсилювача і вбудованого в ЦАП резистора Roc, один з виводів якого виведений на зовнішній вивод мікросхеми (рис. 5.5). Тому, за відсутності іншої домовленості, ми будемо надалі вважати, що вихідний сигнал ЦАП - напруга Uo.

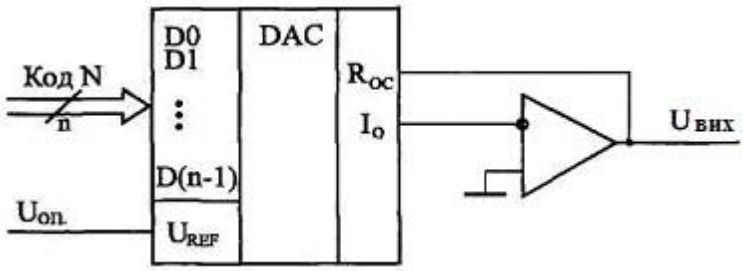


Рисунок 5.5 – Перетворення вихідного тока ЦАП
в вихідну напругу

Суть перетворення вхідного цифрового коду на вихідний аналоговий сигнал досить проста. Вона полягає в поєднанні декількох струмів (по числу розрядів вхідного коду), кожен наступний з яких вдвічі більший від попереднього. Для отримання цих струмів використовуються або транзисторні джерела струму або резистивні матриці, комутовані транзисторними ключами.

Мікросхеми ЦАП, наявні на ринку, розрізняються кількістю розрядів (від 8 до 24), величиною затримки перетворення (від одиниць наносекунд до одиниць мікросекунд), допустимою величиною опорного напруги (зазвичай - одиниці вольт), величинами похибок перетворення та іншими параметрами. Розрізняються вони також технологією виготовлення і особливостями внутрішньої структури, що нерідко накладає обмеження на їх використання. Тому вибирати мікросхему ЦАП для конкретного застосування необхідно з використанням докладної довідкової інформації, що надається фірмами-виробниками. Ми ж будемо говорити тільки про загальні принципи включення ЦАП в цифрові схеми без урахування їх частинних особливостей.

Іноді буває необхідно зменшити кількість розрядів ЦАП. Для цього треба подати сигнали логічного нуля на потрібне число молодших розрядів ЦАП (але ніяк не старших розрядів).

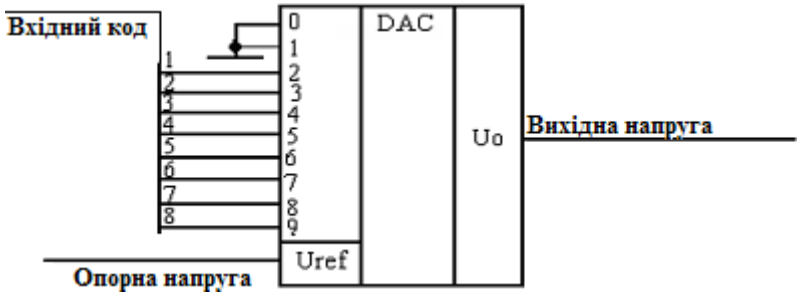


Рисунок 5.6 – Зменшення розрядності ЦАП

Основне застосування мікросхем ЦАП полягає в отриманні аналогового сигналу з послідовності цифрових кодів. Як правило, коди подаються на входи ЦАП через паралельний регістр, що дозволяє забезпечити одночасність зміни всіх розрядів вхідного коду ЦАП. При неодночасному зміні розрядів вхідного коду на виході ЦАП виникають великі короткі імпульси напруги, рівні яких не відповідають жодному з кодів.

Однак навіть при одночасній зміні всіх розрядів вхідного коду ЦАП рівень напруги, відповідний даному коду, встановлюється не відразу, а за час встановлення ЦАП tуст, що пов'язано з тим, що внутрішні елементи ЦАП не ідеальний. Вихідний струм ЦАП, як правило, встановлюється значно швидше ніж вихідна напруга, оскільки він не залежить від інерційності операційного підсилювача. Зрозуміло, що умова належного функціонування ЦАП полягає в тому, щоб тривалість збереження вхідного коду була більша, ніж час встановлення ЦАП tуст, інакше вихідний сигнал не встигне набрати значення, відповідне вхідному коду.

Якщо подавати коди на вхід ЦАП рідко, то наведена схема може використовуватися, наприклад, в керованому джерелі живлення, вихідна напруга якого задається вхідним кодом. Правда, при цьому необхідно ще забезпечити великий вихідний струм джерела живлення, застосувавши зовнішній підсилювач струму.

Якщо ж подавати коди на вхід ЦАП з високою частотою, то можна отримати генератор (він же синтезатор) аналогових сигналів довільної форми. В цьому випадку коди, що надходять на ЦАП, називають кодами вибірок (тобто миттєвих значень) генерованого аналогового сигналу.

У найпростішому випадку в якості джерела вхідних кодів ЦАП можна використовувати звичайний двійковий лічильник (рис. 5.6). Вихідна напруга ЦАП буде зростати при цьому на величину $2U_{REF}$. З кожним тактовим імпульсом, формуючи пилковидні вихідні сигнали амплітудою U_{REF}- Тривалість кожної сходинки дорівнює періоду тактового генератора Т, а період всього вихідного сигналу дорівнює 2 Т. Кількість сходинок в періоді вихідного сигналу дорівнює 2n. Якщо в даній схемі використовувати синхронні лічильники з синхронним перенесенням, то вхідний регістр ЦАП не потрібен, так як всі розряди лічильника перемикаються одночасно. Якщо ж використовуються асинхронні лічильники або синхронні лічильники з асинхронним перенесенням, то вхідний регістр ЦАП необхідний.

Аналогово-цифрові перетворювачі. Види аналого-цифрових перетворювачів і їх особливості.

Аналого-цифрові перетворювачі (АЦП) являють собою пристрої, призначені для перетворення електричних величин (напруги, струму, потужності, опору, ємності та ін.) В цифровий код. Найбільш часто вхідний величиною є напруга. Всі інші величини перед подачею на такий АЦП потрібно попередньо перетворювати на напругу. Однак на практиці знаходять застосування також перетворювачі, наприклад, опору або ємності в цифровий код без проміжного перетворення в напругу. Зазвичай це дозволяє зменшити похибку перетворення, але ускладнює проектування перетворювача і його виготовлення. Останнє пояснюється тим, що серійні промислові мікросхеми АЦП призначені тільки для роботи з напругою. Тому в подальшому будуть розглянуті тільки перетворювачі напруги на цифровий код.

У загальному випадку напруга характеризується її миттєвим значенням. Однак для оцінки напруги можна також користуватися його середнім за вибраний проміжок часу Т значенням.

У зв'язку з цим всі типи АЦП можна розділити на дві групи: АЦП миттєвих значень напруги і АЦП середніх значень напруги. Оскільки операція усереднення передбачає інтегрування миттєвого значення напруги, то АЦП середніх значень часто називають інтегруючими.

При перетворенні напруги на цифровий код використовуються три незалежних операції: дискретизація, квантування і кодування. Процедура аналого-цифрового перетворення безперервного сигналу - це перетворення неперервної функції напруги u (t) на послідовність чисел віднесених до деяких фіксованих моментів часу. При дискретизації безперервна функція u (t) перетворюється на послідовність її відліків.

Друга операція, яка називається квантуванням, полягає в тому, що миттєві значення функції u (t) обмежуються лише певними рівнями, які називаються рівнями квантування. В результаті квантування безперервна функція u (t) набуває вигляду ступінчастої кривої.

Третя операція, яка називається кодуванням, представляє дискретні квантовані величини у вигляді цифрового коду, тобто послідовності цифр, підпорядкованих певним законом. За допомогою операції кодування здійснюється умовне уявлення чисельного значення величини.

Вихідною величиною АЦП є цифровий код, т. з. послідовність цифр, за допомогою якої представляються дискретні квантовані величини. У АЦП використовують чотири основних типи кодів: натуральний двійковий, десятковий, двійковій-десятковий і код Грея. Крім цього, АЦП, призначені для виведення інформації в десятковому коді, видають на своєму виході спеціалізований код для управління семисегментними індикаторами.

Основні характеристики АЦП. Будь-який АЦП є складним електронним пристроєм, який можна виконати у вигляді однієї інтегральної мікросхеми або вмістити велику кількість різних електронних компонентів. У зв'язку з цим характеристики АЦП залежать не тільки від його побудови, але і від характеристик елементів, які входять до його складу. Проте, більшість АЦП оцінюють за їх основним метрологічним показниками, які можна розділити на дві групи: статичні і динамічні.

До статичних характеристик АЦП відносять: абсолютні значення і полярності вхідних сигналів, вхідний опір, значення і полярності вихідних

сигналів, вихідний опір, значення напруг і струмів джерел живлення, кількість довічних або десяткових розрядів вихідного коду, похибки перетворення постійної напруги і ін. До динамічних параметрах АЦП відносять: час перетворення, максимальну частоту дискретизації, апертурний час, динамічну похибку і ін.

Цифро-аналогові перетворювачі з підсумовуванням струмів. Існує кілька схем, які є базою для побудови багатьох різновидів ЦАП відповідного класу. Для формування відповідних рівнів вихідної напруги (або струму) до виходу ЦАП підключається необхідну кількість опорних сигналів E1, E2 ... En (або струмів I1, I2 ... In) або встановлюють відповідне дискретне значення коефіцієнта ділення K1, K2 ... Kn.

На рис. 5.7 наведена схема ЦАП з підсумовуванням струмів. У цій схемі використовуються n опорних джерел струму R1, R2 ... Rn. Вхідний код ь1, Ь2 ... Ьn управляє ключами S1, S2 ... Sn, які або підключають джерела струму до навантаження, або замикають їх накоротко. При цьому якщо bi = 0, то відповідне джерело закорочено і в роботі схеми участі не приймає. Якщо ж bi = 1, то відповідне джерело струму підключено до навантаження.

Результуючий струм дорівнює сумі струмів опорних джерел, для яких bi = 1. Напруга на виході дорівнюватиме результуючому току IΣ помноженому на опір RH, тобто.

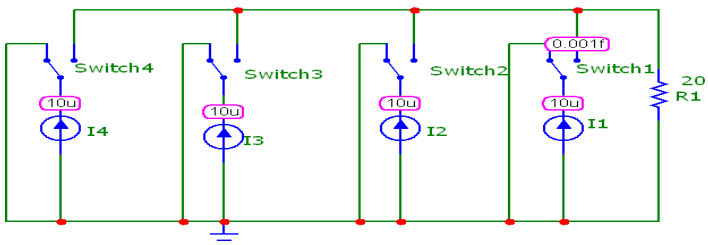
U вих = IΣ RH

Так, наприклад, якщо вхідний код є двійковим, то результуючий струм визначається виразом:

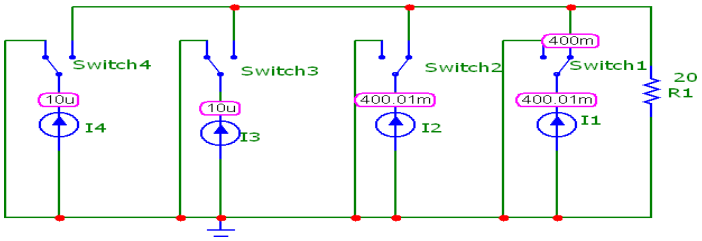
IΣ = I0 (b1 2n-1 + b2 2n-2 + ... + bn20) = I0N,

де n - число двійкових розрядів вхідного струму, N - n-розрядне цифрове слово.

а)



б)



в)

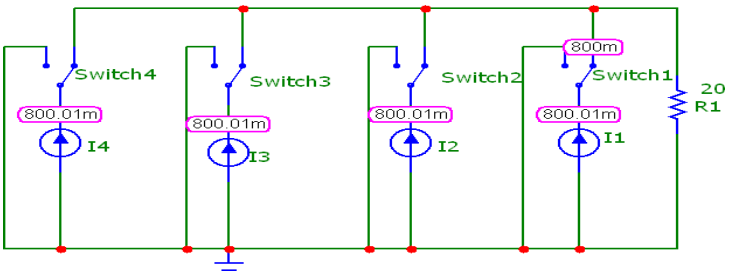


Рисунок 5.7 – Спрощена схема ЦАП зпідсумуванням токів:
а) – де всі гілки напруги підключені до землі;
б) – 2 джерела струму підключені до навантаження;
в) – де всі джерела струму підключені до навантаження;

На рис. 5.7 (а, б, в) наведені схеми ЦАП з підсумовуванням струмів, з різною кількістю підключених до навантаження джерел.

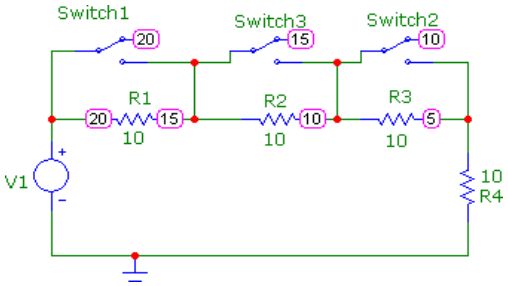
Спрощені схеми ЦАП з розподілом напруги

Спрощена схема ЦАП з розподілом опорної напруги E0 наведена на рис. 5.8. У цій схемі є одине джерело опорної напруги і набір каліброваних опорів R1, R2 ... Rn, за допомогою яких напруга опорного джерела може бути розділена до значення, відповідного вхідного коду.

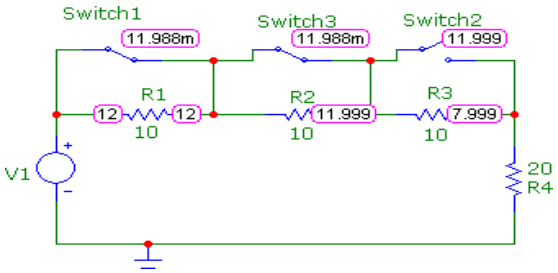
Вихідна напруга для схеми, наведеної на рисунку 5.8, визначається формулою: U вих = (E0 RH) / (RΣ + RH),

де RΣ - результуючий опір, який встановлюється за допомогою ключів S1, S2 ... Sn, які управляються вхідним кодом.

а)



б)



в)

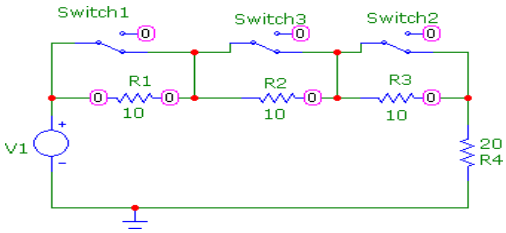


Рисунок 5.8 – Спрощена схема ЦАП з розподілом напруги:
а) - всі гілки напруги підключені до землі; б) - 2 джерела струму підключені до напруги живлення; в) - все джерела струму підключені до напруги живлення

При RH = 0 ця схема перетворюється на кероване джерело струму, тобто працює так само, як схема зі складанням струмів. Практично виконати RH = 0 можна за допомогою операційного підсилювача з паралельним зворотним зв'язком.

На рис. 5.8 (а, б, в) наведені спрощені схеми ЦАП з розподілом напруги, з різною кількістю підключених джерел струму до напруги.

Схеми ЦАП зі складанням струмів на резистивній матриці типу R-2R

Практична схема ЦАП зі складанням струмів зазвичай виконується на джерелі опорного напруги. На рис. 5.9 наведена схема ЦАП з підсумовуванням струмів, в якому використано одно джерело опорної напруги E0 і резистивна матриця типу R-2R. Особливість резистивної матриці полягає в тому, що при будь-якому положенні ключів S1, S2 ... Sn вхідний опір матриці завжди дорівнює R, а отже, струм, який впадає в матрицю, дорівнює I0 = E0 / R. Далі він послідовно ділиться в вузлах за допомогою бінарного закону. Двійковий

закон розподілу струмів в гілках резистивної матриці дотримується за умови рівності нулю опору навантаження. Оскільки навантаженням резистивної матриці є операційний підсилювач ОУ, охоплений негативним зворотним зв'язком через опір Roc, то його вхідний опір дорівнює нулю з досить високою точністю.

Напруга на виході операційного підсилювача визначається виразом U вих = ((E0 Roc) | (R2n)) (b1 2 n-1 + b2 2 n-2 + ... + bn20) = ((E0 Roc) | (R2n)) N, де bi = 1, якщо ключ Si знаходиться в положенні, при якому струм протікає в загальний вивід, n - число розрядів перетворювача.

Максимальне значення вихідної напруги має місце при всіх bi = 1 і визначається за формулою:

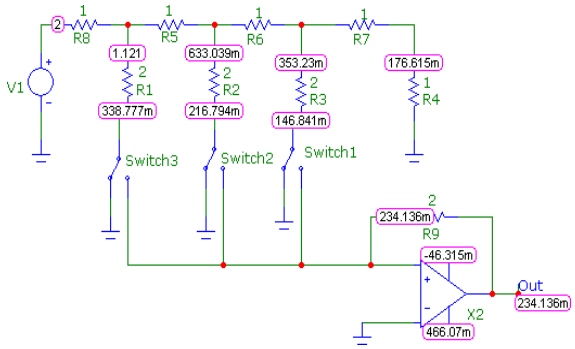
U вих.max = (E0 Roc (1-2-n)) / Roc = (E0 Roc) / R - h,

де h - крок квантування, тобто приріст вихідної напруги при зміні вхідного коду на одиницю молодшого розряду:

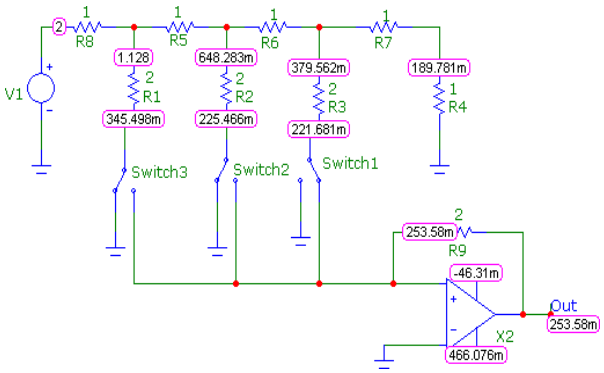
h = (E0 Roc) / R2n.

З формули, вихідна напруга ЦАП залежить не тільки від вхідного коду N, але і від напруги E0 опорного джерела.

а)



б)



в)

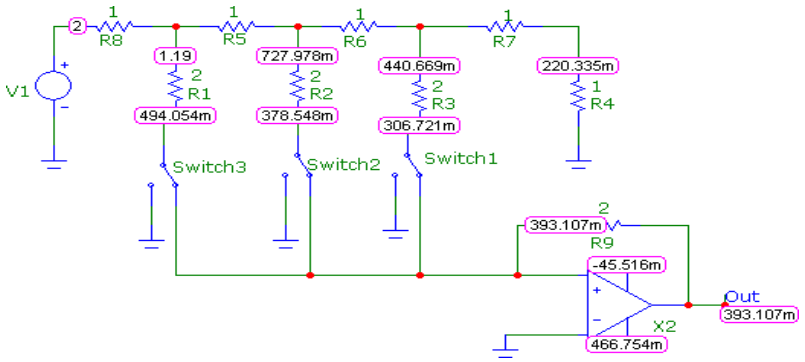


Рисунок 5.9 – Спрощені схеми ЦАП зі складанням струмів на резистивних матрицях типу R-2R:
а) - всі гілки напруги підключені до землі;
б) - 1 джерело струму підключене до операційного підсилювача;
в) - всі джерела струму підключені до операційного підсилювача

На рис. 5.9 (а, б, в) наведені спрощені схеми ЦАП зі складанням струмів на резистивних матрицях типу R-2R, з різною кількістю підключених джерел струму до напруги. Результати моделювання представлені у вигляді рожевого еліпса, в якому видно навантаження.

Аналого-цифрові перетворювачі

Види аналого-цифрових перетворювачів і їх особливості. Аналого-цифрові перетворювачі (АЦП) являють собою пристрої, призначені для перетворення електричних величин (напруги, струму, потужності, опору, ємності та ін.) в цифровий код. Найбільш часто вхідною величиною є напруга. Всі інші величини перед подачею на АЦП потрібно попередньо перетворювати на напругу. Однак на практиці знаходять застосування також перетворювачі, наприклад, опору або ємності в цифровий код без проміжного претворення на напругу. Зазвичай це дозволяє зменшити похибку претворення, але ускладнює проектування перетворювача і його виготовлення. Останнє пояснюється тим, що серійні промислові мікросхеми АЦП призначені тільки для роботи з напругою. Тому далі будуть розглянуті тільки перетворювачі напруги в цифровий код. У загальному випадку напруга характеризується миттєвим

значенням u (t). Однак для оцінки напруги можна також користуватися середнім за вибраний проміжок часу T значенням:

$$U_{ср} = U = \frac{1}{T}\int_0^T u(t)dt \quad .$$

У зв'язку з цим типи АЦП можна розділити на дві групи. АЦП миттєвих значень напруги і АЦП середніх значень напруги. Оскільки операція усереднення передбачає інтегрування миттєвого значення напруги, то АЦП середніх значень часто називають інтегруючим. При перетворенні напруги на цифровий код використовуються три незалежних операції: дискретизація, квантування і кодування. Процедура аналого-цифрового перетворення безперервного сигналу являє собою перетворення безперервної функції напруги u(t) в послідовність чисел $u(t_n)$, де n = 0, 1, 2 ..., віднесених до деяких фіксованих моментів часу. При дискретизації безперервна функція u(t) перетворюється на послідовність її відліків и (tn), як показано на рис. 5.10а.

Друга операція, яка називається квантуванням, полягає в тому, що миттєве значення функції u(t) обмежуються лише певними рівнями, які називаються рівнями квантування. В результаті квантування безперервна функція u(t) набирає вигляду ступінчастої кривої u_k (t) показаної на рис. 5.11.

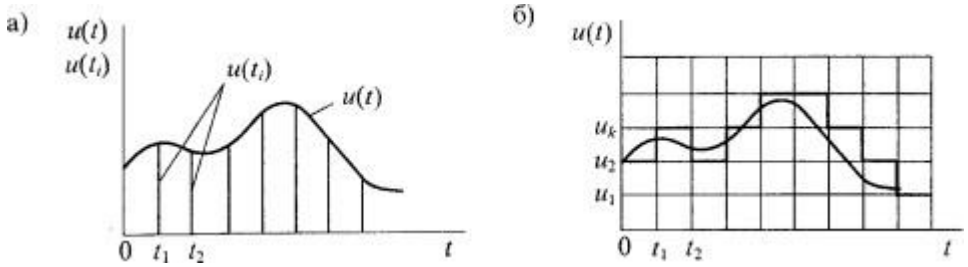


Рисунок 5.10 – Процес дискретизації (а) та квантування (б) безперервного сигналу $u(t)$

Третя операція, яка називається кодуванням, представляє дискретні квантовані величини у вигляді цифрового коду, тобто послідовності цифр, підпорядкованих певним законам. За допомогою операції кодування здійснюється умовне представлення чисельного значення величини.

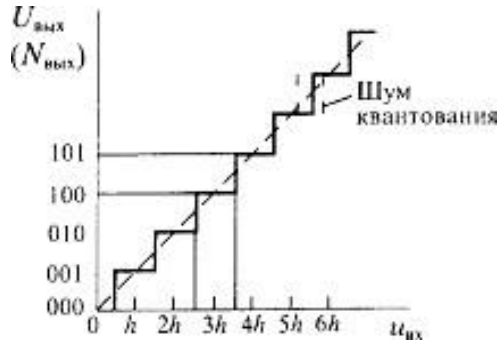


Рисунок 5.11 – Характеристика ідеального квантування

В основі дискретизації сигналів лежить принципова можливість подання їх у вигляді зважених сум:

$$u(t) = \sum_n a_n f_n(t) \quad ,$$

де a_n - деякі коефіцієнти чи відліки, які характеризують початковий сигнал в дискретні моменти часу, $f_n(t)$ - набір елементарних функцій, які використовуються при відновленні сигналу за його відліками.

Дискретизація буває рівномірна і нерівномірна. При рівномірній дискретизації період відліків Т залишається постійним, а при нерівно-мірній - період може змінюватися. Нерівномірна дискретизація найчастіше обумовлена швидкістю зміни сигналу і тому називається адаптивною.

В основі рівномірної дискретизації лежить теорема відліків, згідно якої в якості коефіцієнтів an потрібно використовувати миттєві значення сигналу u (tn) в дискретні моменти часу t_n = Tn, а період дискретізації вибирати з умови $T = (2 f_m)^{-1}$, де f_m – максимальна частота в спектрі вихідного сигналу

На відміну від дискретизації, яка теоретично є оборотною операцією, квантування є необоротне перетворення вихідної послідовності і супроводжується появою неминучих похибок. Характеристика ідеального квантувача наведена на рис. 5.11. При рівномірному квантуванні відстань між двома сусідніми значеннями робиться постійною, як показано на рис. 5.10 Різниця між двома сусідніми значеннями квантованной величини називається кроком квантування h.

По суті квантування є операцію округлення безперервної величини до найближчого цілого значення. В результаті максимальна похибка квантування дорівнює ± 0,5h (рис. 5.10). Однак при перетворенні довільного сигналу максимальна похибка зустрічається порівняно рідко, тому в більшості випадків для оцінки якості АЦП використовують не максимальну, а середньоквадратичну похибку $\sigma_{кв} = h/\sqrt{12}$, яка приблизно в 3,5 раза менша від максимальної. У АЦП похибка квантування визначається як одиниця молодшого значущого розряду (ОМР).

Вихідною величиною АЦП є цифровий код, тобто послідовність цифр, за допомогою якої представляються дискретні квантовані величини. У АЦП використовують чотири основних типи кодів: натуральний двійковий, десятковий, двійковий-десятковий і код Грея. Крім цього, АЦП, призначені для виведення інформації в десятковому коді, видають на своєму виході спеціалізований код для управління семисегментними індикаторами.

5.6 Імпульсні джерела електроживлення

Види і особливості імпульсних джерел електроживлення.

Імпульсні джерела електроживлення в наш час набули поширення не менше, ніж лінійні стабілізатори напруги. Їх основними перевагами є: високий коефіцієнт корисної дії, малі габарити і маса, висока питома(удельная) потужність. Всі перераховані властивості ці джерела живлення отримали завдяки застосуванню ключового режиму при роботі силових елементів. У ключовому режимі робоча точка транзистора більшу частину часу перебуває в області насичення або області відсічення, а зону активного (лінійного) режиму проходить з високою швидкістю за дуже малий час перемикання. При цьому в області насичення напруга на транзисторі близько до нуля, а в режимі відсічення в транзисторі відсутній струм. завдяки чому втрати в транзисторі виявляються досить малими. Все це призводить до того, що середня за період комутації потужність, яка розсіюється в ключовому транзисторі, виявляється набагато менша, ніж в лінійному регуляторі.

Малі втрати в силових ключах призводять до зменшення або повного виключення охолоджуючих радіаторів.

Поліпшення масогабаритних характеристик джерела живлення обумовлено перш за все тим, що зі схеми джерела живлення виключається силовий трансформатор, який працює на частоті 50 Гц. Замість нього в схему вводиться високочастотний трансформатор або дросель, габарити і маса якого набагато менша від низькочастотного силового трансформатора.

До недоліків імпульсних джерел електроживлення зазвичай відносять: складність схеми, наявність високочастотних шумів і перешкод, збільшені пульсації вихідної напруги, великий час виходу на робочий режим. Порівняльні характеристики звичайних (тобто з силовим трансформатором) і імпульсних джерел живлення показує, що ККД імпульсних джерел живлення збільшується в порівнянні з лінійними у співвідношенні 2:1, а питома потужність зростає у співвідношенні 4:1. При підвищенні частоти перетворення з 20кГц до 200кГц питома потужність збільшується в співвідношенні 8:1, тобто майже в два рази. Імпульсні джерела живлення мають більший час утримання вихідної напруги при раптовому відключення живлення. Це обумовлено тим що в мережевому випрямлячі використовуються конденсатори великої ємності і з високою робочою напругою (до 400 В). При цьому розміри конденсатора ростуть пропорційно добутку CU, а енергія конденсатора пропорційна CU^2. Цієї енергії конденсатора мережевого випрямляча досить для підтримки в робочому стані джерела живлення до 30 мс, що дуже важливо для збереження інформації в комп'ютерах при раптовому відімкненні живлення.

У той же час пульсації вихідної напруги в імпульсних джерелах живлення більші, ніж у лінійних, що обумовлено складністю глушіння коротких імпульсів при роботі імпульсного перетворювача. Інші характеристики у цих джерел практично збігаються.

Узагальнена структурна схема імпульсного джерела живлення наведена на рис.5.12. Вона складається з чотирьох основних блоків:

мережевого випрямляча з ємнісним фільтром;

високочастотного інвертора випрямленої напруги мережі;

пристрою управління високочастотним інвертором (зазвичай це спеціалізована мікросхема управління);

вихідного високочастотного випрямляча з ємнісним фільтром.

Високочастотний інвертор і пристрій управління спільно утворюють імпульсний перетворювач, який може бути індуктивним або ємнісним. Найбільшого поширення в імпульсних ІВЕП отримали індуктивні імпульсні перетворювачі, які можна розділити на дросельні (або автотрансформаторні) і трансформаторні. Ємнісні (конденсаторні) перетворювачі мають обмежене застосування - для інверсії полярності або подвоєння (множення) напруги. Зазвичай вони представляють собою пристрої з конденсаторами, що перемикаються і використовуються для живлення малопотужних навантажень.

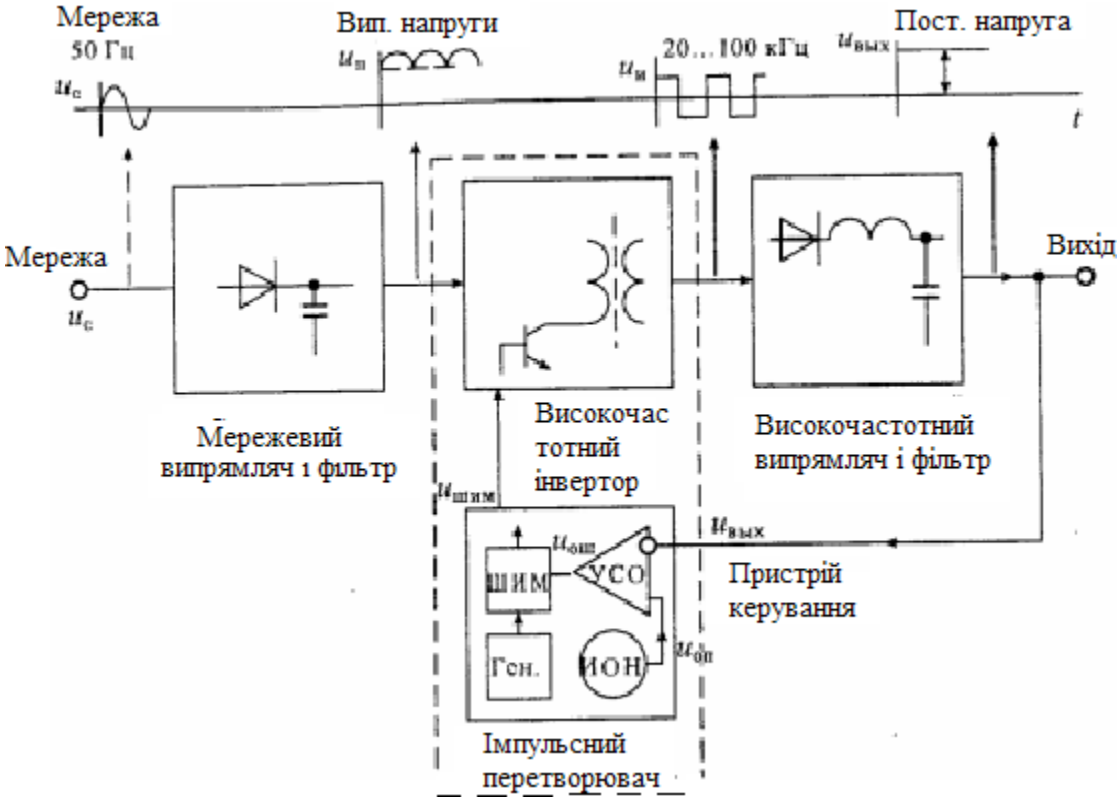


Рисунок 5.12 – Узагальнена структурна схема імпульсного джерела електроживлення

Дросельні і автотрансформаторні перетворювачі відносять до розряду імпульсних стабілізаторів напруги, які ділять на три групи: понижуючі, підвищуючі та інвертувальні.

Особливістю імпульсних стабілізаторів є їх гальванічний зв'язок з силовою мережею живлення. Для виключення гальванічного зв'язку на вході імпульсного стабілізатора іноді включають силовий трансформатор, однак це знижує питому потужність.

Трансформаторні імпульсні перетворювачі не мають гальванічного зв'язку з мережею, проте їх питома потужність нижча, ніж у дросельних. Трансформаторні перетворювачі можна розділити на однотактні і двотактні. У однотактних перетворювачах енергія передається на вихід тільки протягом однієї частини періоду перетворення, якщо енергія передасться при включеному силовому ключі, то такий перетворювач називають прямоходовим (Forward). Якщо ж енергія передасться при вимкненому стані силового ключа, то перетворювач називають зворотноходовим (Flyback).

Двотактні перетворювачі ділять на двофазні (Push-Pull), мостові (Full-Bridge) і напівмостові (Half-Bridge). У двотактних перетворювачах використовуються обидві частини періоду перетворення. На відміну від однотактних двотактні перетворювачі працюють без підмагнічування осердя трансформатора постійним струмом.

Розглянемо роботу ключового джерела живлення, користуючись узагальненою, структурною схемою, наведеною на рис. 5.12. Гармонійна напруга мережі (50 або 60 Гц) випрямляється мережевим випрямлячем і заряджає конденсатор фільтра, який має досить велику ємність. Велика ємність фільтра мережевого випрямляча забезпечує низькі пульсації випрямленої напруги і збільшує час утримання вихідної напруги. При ємності фільтра 100мкФ і споживаної потужності 100Вт час утримання становить приблизно 30мс. При напрузі мережі живлення 220В напруга на ємності становить приблизно 300В.

Ця напруга надходить на вхід імпульсного перетворювача, який перетворює її на високочастотні імпульси прямокутної форми. Частота імпульсного напруги зазвичай лежить в межах від 20 до 200кГц. Зі збільшенням частоти перетворення збільшується питома потужність, але одночасно

зростають втрати в елементах перетворювача, що призводить до зниження ККД.

З виходу перетворювача напруга надходить на високо частотний випрямляч з ємнісним фільтром. При високій частоті перетворення до елементів випрямляча і фільтра ставляться дуже жорсткі вимоги: час відновлення зворотного опору випрямних діодів має лежати в межах від 10 до 100нс, а ємності фільтра не повинні мати індуктивності.

У більшості випадків високочастотний інвертор працює на фіксованій частоті, а регулювання вихідної напруги забезпечується за допомогою широтно-імпульсної модуляції управляючих сигналів. Широтно-імпульсне регулювання виконується за допомогою схеми управління, на вхід якої подається вихідна напруга. Для забезпечення гальванічного відділення виходу від силової мережі в трансформаторних схемах інверторів зазвичай використовуються різні типи пристроїв гальванічної розв'язки: оптрони, трансформатори, ізолюючі підсилювачі та ін. Форми керуючих сигналів при широтно-імпульсної модуляції наведені на рис. 5.13. Глибина широтно-імпульсной модуляції характеризується коефіцієнтом заповнення $\gamma = t_i / T$,

де t_i тривалість імпульса управління, а $T=f^{-1}$ - період повторення. Якщо тривалість імпульсу становить половину періоду, то $\gamma = 0,5$, тобто 50%. При збільшенні тривалості імпульсу коефіцієнт заповнення зростає до 100% . У загальному випадку коефіцієнт заповнення $0 < \gamma < 100$ %.

Спосіб отримання широтно-модульованих імпульсів показаний на рис. 5.13. У схемі, наведеній на рис. 5.12, спочатку формується сигнал помилки Uош, (рассогласованія). Для цього на вхід схеми управління подається вихідна напруга Uвих, яке порівнюється в підсилювачі сигналу помилки (УСО) з опорною напругою Uоп, створюваною спеціальним джерелом опорної напруги (ДОН).

У схемі широтно-імпульсного модулятора (ШІМ) сигнал помилки Uощ порівнюється з лінійно зростаючою напругою пилкоподібної форми Uпm. Якщо за початковий стан ШІМ прийняти, що Uош, = Uпm /2, де Uпm - максимальне значення пилкоподібної напруги, то отримаємо, що в початковому стані коефіцієнт заповнення $\gamma_o = 50\%$.

При збільшенні вихідної напруги Uвих > Uвих.ном., сигнал помилки також збільшується Uош > Unm / 2, а тривалість імпульсу управління зменшується, як показано на рис. 5.13 б.

При зменшенні вихідної напруги Uвих < Uвих.ном сигнал помилки зменшується Uош < Unm/2, а тривалість імпульсу збільшується.

Зміна тривалості імпульсу t_i призводить до зміни часу включеного стану силового транзисторного ключа і, до пропорційної зміни вихідної напруги. Таким чином в регульованому ШІМ-інверторі забезпечується стабілізація вихідної напруги.

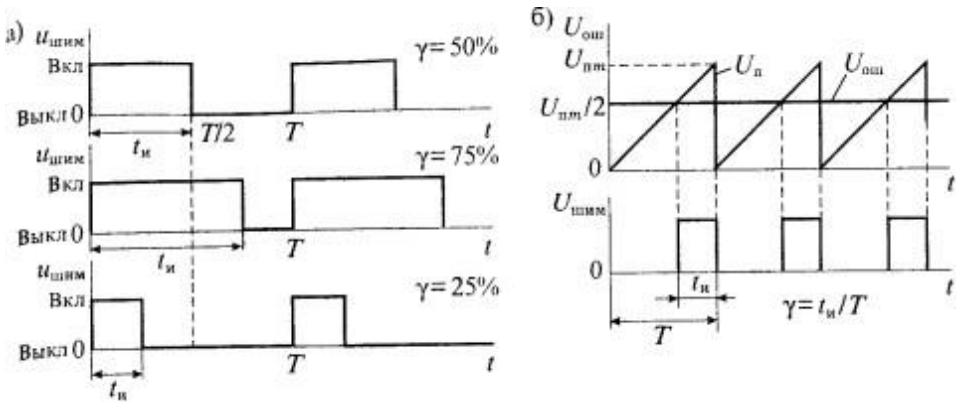


Рисунок 5.13 Форма імпульсів при широтно-імпульсній модуляції (а) та засоб їх отримання (б)

6 АРХІТЕКТУРИ З ПОВНИМ І СКОРОЧЕНИМ НАБОРОМ КОМАНД (RISC- ТА CISC-ПРОЦЕСОРИ)

Сучасна технологія програмування орієнтована на мови високого рівня (МВР), головне завдання яких - полегшити процес написання програм. Більше 90% всього процесу програмування здійснюють на мовах високого рівня. На жаль, операції, характерні для мов високого рівня, відрізняються від операцій, що реалізуються машинними командами. Ця проблема отримала назву семантичного розриву і веде вона до недостатньо ефективного виконання програм. Намагаючись подолати семантичний розрив, розробники ОМ розширюють систему команд, доповнюючи її командами, що реалізують складні оператори мов високого рівня на апаратному рівні, вводять додаткові види адресації і т. п. Обчислювальні машини, де це реалізовано, прийнято називати ОМ з повним набором команд (CISC - Complex Instruction Set Computer). До типу CISC можна віднести практично всі ОМ, що випускалися до середини 80-х років і значну частину з тих що випускаються в даний час.

Характерні для CISC способи вирішення проблеми семантичного розриву разом з тим ведуть до ускладнення архітектури ОМ, головним чином пристрою управління, що, в свою чергу, негативно позначається на продуктивності в цілому. Крім того, в CISC дуже складно організувати ефективний конвеєр команд, який, як уже зазначалося, є одним з найбільш перспективних шляхів підвищення продуктивності ОМ. Все це змусило більш уважно проаналізувати програми, одержувані після компіляції з мов високого рівня. Були зроблені комплекс досліджень, в результаті яких виявилися цікаві закономірності:

- реалізація складних команд, еквівалентних операторам МВР, вимагає збільшення ємності керуючої пам'яті в мікропрограмному пристрої управління УП. Мікропрограми складних команд можуть займати до 60% керуючої пам'яті, в той час як їх частка в загальному обсязі програми часто не перевищує 0,2%;

- у скомпільованій програмі оператори мов високого рівня реалізуються у вигляді процедур (підпрограм), тому на операції виклику

процедури і повернення з неї припадає від 15 до 45% обчислювального навантаження;

- при виклику процедури програма, яка викликає, передає цій процедурі кілька аргументів. У 98% випадків число переданих аргументів не перевищує шести. Приблизно таке ж становище склалося і з параметрами, які процедура повертає програмі, яка викликає. Більше 80% змінних, які використовує програма, є локальними, тобто створюються при вході в процедуру і знищуються при виході з неї. Кількість локальних змінних, що створюються окремою процедурою, в 92% випадків не перевищує шести.

- майже половину операцій в ході обчислень становить операція присвоювання, що зводиться до пересилання даних між регістрами, осередками пам'яті або регістрами і пам'яттю.

Детальний аналіз результатів досліджень призвів до перегляду традиційних архітектурних рішень, наслідком чого стала поява архітектури зі скороченим набором команд (RISC - Reduced Instruction Set Computer). Термін «RISC» вперше був використаний Паттерсоном і Дітцелем в 1980 році.

6.1 Основні риси RISC-архітектури

Головні зусилля в архітектурі RISC спрямовані на побудову максимально ефективного конвеєра команд, тобто такого, де всі команди витягуються із пам'яті і надходять в центральний процесор (ЦП) на обробку у вигляді рівномірного потоку. При цьому жодна команда не повинна перебувати в стані очікування, а ЦП повинен залишатися завантаженим протягом усього часу. Крім того, ідеальним буде варіант, коли будь-який етап циклу команди виконується протягом одного тактового періоду. Останню умову можна доволі просто реалізувати для етапу вибірки. Необхідно лише, щоб всі команди мали стандартну довжину, яка б дорівнювалась ширині шини даних, що з'єднує ЦП і пам'ять. Уніфікація часу виконання для різних команд - значно складніше завдання, оскільки поряд з командами звернення до регістрів існують також команди із зверненням до пам'яті.

Крім однакової довжини команд, важливо мати відносно просту підсистему декодування і управління, тому що складний пристрій управління (ПУ) буде вносити додаткові затримки в формування сигналів управління. Очевидний шлях істотного спрощення УП - скорочення числа виконуваних команд, форматів команд і даних, а також видів адресації.

Зрозуміло, що в скороченому списку команд повинні залишатися ті, які використовуються найбільш часто. Дослідження показали, що 80-90% часу виконання типових програм припадає на відносно малу частину команд (10-20%). До найбільш часто дій, які потребуються, належать пересилання даних, арифметичні і логічні операції. Основна причина, що перешкоджає приведенню всіх етапів циклу команди до одного тактовою періоду, - потенційна необхідність доступу до пам'яті для вибірки операндів і / або запису результатів. Слід максимально скоротити число команд, що мають доступ до пам'яті. Це міркування додає до раніше згаданих принципів RISC ще два:

- доступ до пам'яті під час виконання здійснюється тільки командами «Читання» і «Запис»;

- всі операції, крім «Читання» і «Запис», мають тип «регістр-регістр».

Для спрощення виконання більшості команд і приведення їх до типу «регістр-регістр» потрібно забезпечити ЦП значним числом регістрів загального призначення. Велике число регістрів в регістровому файлі ЦП дозволяє забезпечити тимчасове зберігання проміжних результатів, які використовуються як операнди в подальших операціях, і веде до зменшення числа звернень до пам'яті, прискорюючи виконання операцій. Мінімальна кількість регістрів, рівна 32, взята як стандарт де-факто більшістю виробників RISC-комп'ютерів.

Підсумовуючи вище сказане, концепцію RISC-комп'ютера можна звести до таких положень:

- виконання всіх (або, принаймні, 75% команд) за один цикл;

- стандартна однослівна довжина всіх команд, що дорівнює природній довжині слова і ширині шини даних і допускає уніфіковану потокову обробку всіх команд;

- мале число команд (не більше 128);

- мала кількість форматів команд (не більше 4);

- мале число способів адресації (не більше 4);

- доступ до пам'яті тільки за допомогою команд «Читання» і «Запис»;

- всі команди, за винятком «Читання» і «Записи», використовують внутрішньопроцесорне міжрегістрове пересилання;

- пристрій управління з «жорсткою» логікою;

- відносно великий (не менше 32) процесорний файл регістрів загального призначення (число регістрів загального призначення в сучасних RISC-мікропроцесори може перевищувати 500).

6.2 Регістри в RISC-процесорах

Відмітна риса RISC-архітектури - велика кількість регістрів загального призначення (РЗП), що пояснюється прагненням звести все пересилання до типу «регістр-регістр». Але збільшення числа РЗП здатне дати ефект лише при розумному їх використанні. Оптимізація використання регістрів в RISC-процесорах забезпечується програмними і апаратними засобами.

Програмна оптимізація виконується на етапі компіляції програми, написаної на МВР. Компілятор прагне так розподілити регістри процесора, щоб розмістити в них ті змінні, які протягом певного періоду часу будуть використовуватися найбільш інтенсивно.

На початковому етапі компілятор виділяє кожній змінній віртуальний регістр. Число віртуальних регістрів в принципі не обмежена. Потім компілятор відображає віртуальні регістри на обмежену кількість фізичних регістрів. Віртуальні регістри, використання яких не перекривається, відображаються на один і той же фізичний регістр. Якщо в певному фрагменті програми фізичних регістрів не вистачає, то їх роль для решти віртуальних регістрів виконують комірки пам'яті. В ході обчислень вміст кожної такої комірки за допомогою команди «Читання» тимчасово засилається в регістр, після чого командою «Запис» знову повертається в комірку пам'яті.

Завдання оптимізації полягає у визначенні того, яким змінним в даній точці програми найвигідніше виділити фізичні регістри. Найбільш

поширений метод, застосовуваний для цієї мети, відомий як розфарбування графа. У загальному випадку метод формулюється в такий спосіб. Є граф, що складається з вузлів і ребер. Необхідно розфарбувати вузли так, щоб сусідні вузли мали різний колір і щоб, при цьому, загальна кількість залучених кольорів було мінімальним. У нашому випадку роль вузлів виконують віртуальні регістри. Якщо два віртуальних регістра одночасно присутні в одному і тому ж фрагменті програми, вони з'єднуються ребром. Робиться спроба розфарбувати граф в n кольорів, де n - число фізичних регістрів. Якщо така спроба не увінчалася успіхом, то вузлам, які не вдалося розфарбувати, замість фізичних регістрів виділяються комірки в пам'яті.

На рис. 6.1 наведено приклад розмальовки графа, в якому шість віртуальних регістрів відображаються на три фізичних. Показана тимчасова послідовність активного залучення в роботу кожного віртуального регістру (рис. 6.1, а) і розфарбований граф (рис.6.1, б).

Як видно, не вдалося розфарбувати тільки віртуальний регістр F, його доведеться відображати на комірку пам'яті.

Апаратна оптимізація використання регістрів в RISC-процесорах орієнтована на скорочення витрат часу при роботі з процедурами. Найбільше час в програмах, написаних на мовах високого рівня , витрачається на виклики процедур і повернення з них. Пов'язано це зі створенням і обробкою великого числа локальних змінних і констант. Одним з механізмів для боротьби з цим ефектом є так звані регістрові вікна. Головне їхнє завдання - спростити і прискорити передачу параметрів від процедури, що викликає до викликаємої і назад.

Регістровий файл розбивається на групи регістрів, звані вікнами. Окреме вікно призначається глобальним змінним. Глобальні регістри доступні всім процедурам, які виконуються в системі в будь-який час. З іншого боку, кожній процедурі виділяється окреме вікно в регістровому файлі. Всі вікна мають однаковий розмір (зазвичай по 32 регістра) і складаються з трьох полів. Ліве поле кожного регістрового вікна одночасно є і правим полем попереднього вікна (рис. 6.2). Середнє поле служить для зберігання локальних змінних і констант процедури.

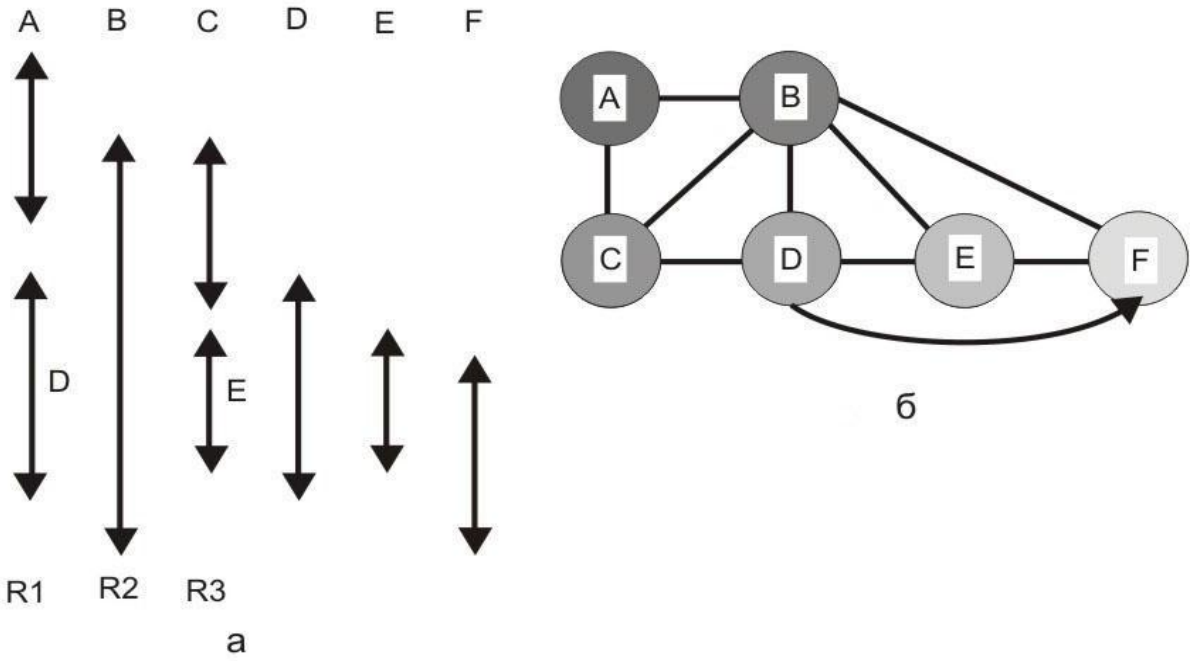


Рисунок 6.1 – Ілюстрація методу розфарбовки графа:

а – часова послідовність активного використання віртуальних регістрів;

б – граф взаємного використання регістрів

База вікна (перший в послідовності регістрів вікна) вказується полем, званим покажчиком поточного вікна (CWP, Current Window Pointer), зазвичай розташованим в регістрі (слові) стану ЦП. Якщо поточній процедурі призначено реєстрове вікно j, CWP містить значення j.

Кожній знову викликаній процедурі виділяється реєстрове вікно, яке безпосередньо слідує за вікном процедури, яка її викликала. Останні k регістрів вікна j одночасно є першими k регістрами вікна j + 1. Якщо процедура, що займає вікно j, звертається до процедури, якії в цій архітектурі має бути призначено вікно j + 1, вона може передати в процесі виклику k аргументів.

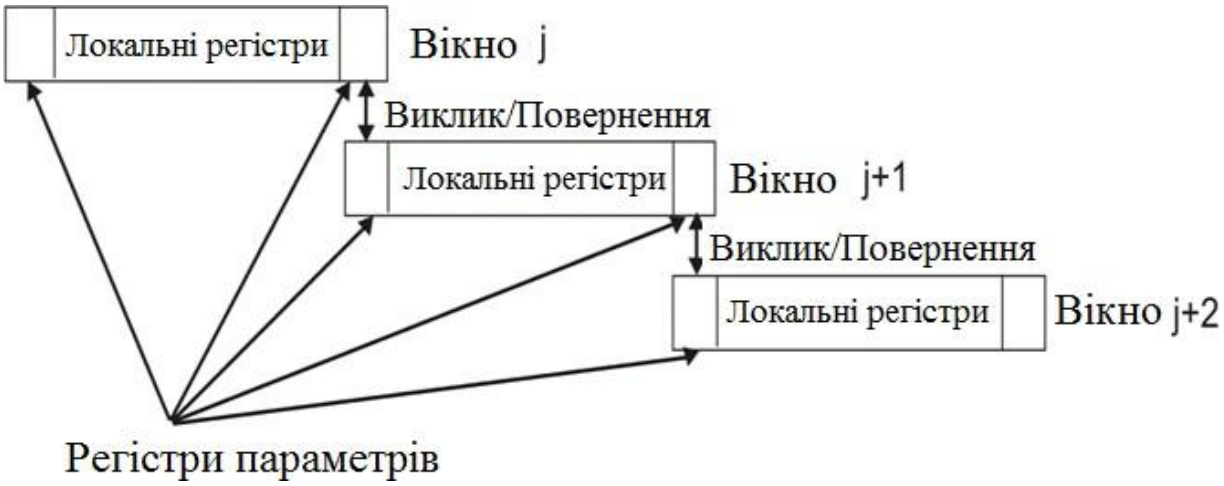


Рисунок 6.2 – Перекриття регістрових вікон

Згадані k регістрів відразу ж будуть доступні викликаній процедурі без всяких пересилань. Природно, виклик призведе до збільшення вмісту поля CPW на одиницю.

Глибина вкладення процедур одна в іншу може бути досить велика, і бажано, щоб кількість реєстрових вікон не було стримуючим фактором. Це досягається за рахунок організації вікон у вигляді циклічного буфера.

На рис. 6.3 показаний циклічний буфер з шести вікон, заповнений на глибину 4 (процедура A викликала B, B викликала C, C викликала D). Покажчик поточного вікна (CWP) ідентифікує вікно активної на даний момент процедури - D, тобто вікно O_3. При виконанні процедури всі посилання на регістри в командах перетворюються в зміщення щодо CWP. Покажчик збереженого вікна (SWP, Saved Window Pointer) містить номер останнього з вікон, збережених в пам'яті через переповнення циклічного буфера. Якщо процедура D тепер викличе процедуру E, аргументи для неї вона помістить в загальне для обох поле реєстрових вікон (перетин вікон O_3 і O_4), а значення CWP збільшиться на одиницю, тобто CWP буде показувати на вікно O_4.

Якщо далі процедура E викличе процедуру F, то цей виклик при існуючому стані буфера не може бути виконаний, оскільки вікно для F (O_5) перекривається з вікном процедури A (O_0). Отже, при спробі F почати

завантажувати праве поле свого вікна будуть втрачені параметри процедури A ($A_{вх}$). Тому, коли CWP збільшується на одиницю (операція виконується по модулю 6) і виявляється рівним SWP, виникає переривання, і вікно процедури A зберігається в пам'яті (запам'ятовуються лише поля $A_{вх}$ і $A_{лок}$). Далі значення CWP інкрементується і здійснюється виклик процедури F. Аналогічне переривання відбувається і при поверненні, наприклад коли виконується повернення з процедури B в A. CWP зменшується на одиницю (по модулю 6) і співпаде зі значенням SWP. Обробка переривання призведе до відновлення вмісту вікна процедури A з пам'яті.

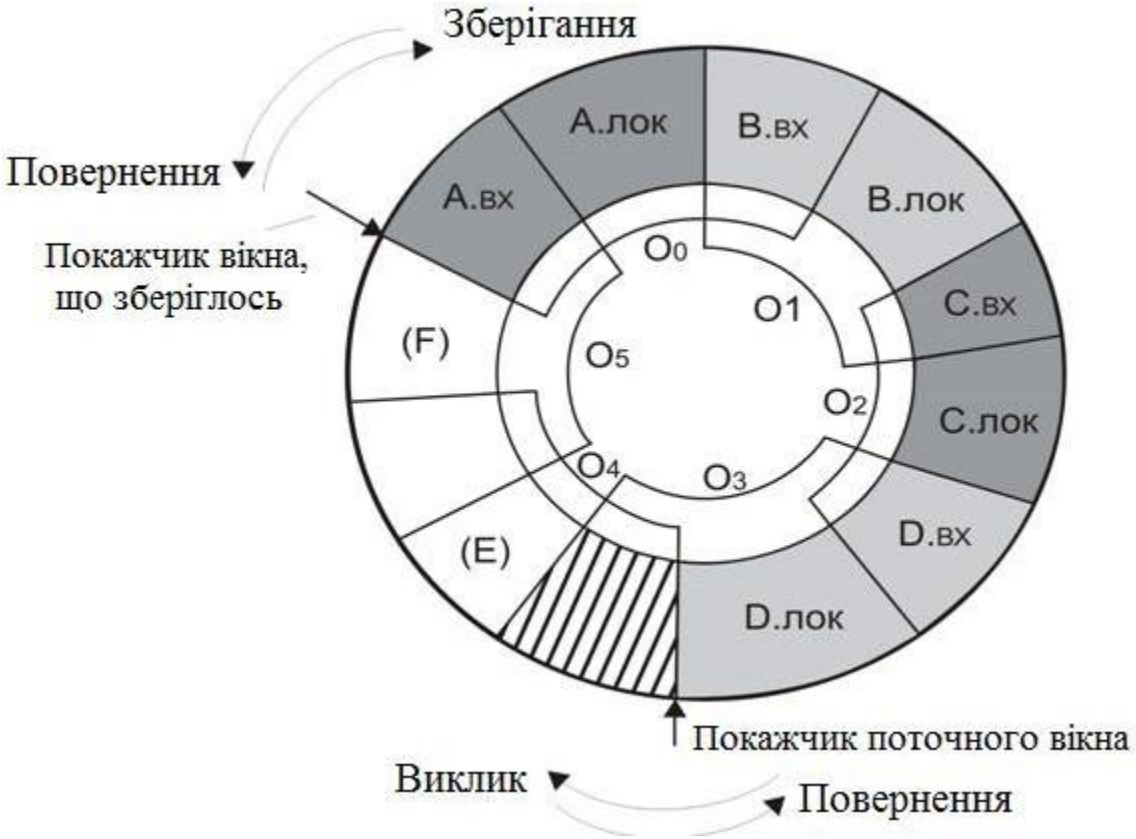


Рисунок 6.3 – Циклічний буфер із регістрових вікон, що перетинаються

Як видно з прикладу, регістровий файл із n вікон здатний підтримувати n - 1 виклик процедури. Число n не повинно бути великим. При 8 регістрових вікнах збереження і відновлення вікон в пам'яті потрібно лише для 1%

операцій виклику процедур. У ОМ Pyramid, наприклад, використовується 16 вікон по 32 регістра в кожному.

Теоретично такий прийом не виключений і в CISC. Однак пристрій управління CISC-процесора займає на кристалі більше 50% площі, залишаючи мало місця для інших підсистем, зокрема для великого файлу регістрів. УП RISC займає близько 10% поверхні кристала, надаючи можливість мати великий регістровий файл.

Інша техніка апаратної оптимізації, яка часто зустрічається, використання регістрів - надання деяким з них спеціальної якості: такі регістри в змозі приймати на себе ім'я будь-якого регістра загального призначення . Набір регістрів, що які мають подібні властивості, називають буфером перейменування. Прийом виявляється дуже зручним при конвеєризації команд і дозволяє запобігти конфліктам, коли одна команда хоче скористатися регістром, в даний момент зайнятий іншою командою.

6.3 Переваги і недоліки RISC

Порівнюючи переваги і недоліки CISC і RISC, неможливо зробити однозначний висновок про незаперечну перевагу одній архітектури над іншою. Для окремих сфер використання ОМ кращої виявляється та чи інша. Проте нижче наводиться основна аргументація «за» і «проти» RISC-архітектури.

Для технології RISC характерна порівняно проста структура пристрою управління. Площа, що виділяється на кристалі мікросхеми для реалізації ПУ, істотно менше. Так, в RISC I вона становить 6%, а в RISC II - 10%. Як наслідок, з'являється можливість розмістити на кристалі велике число регістрів ЦП (138 в RISC II). Крім того, залишається більше місця для інших вузлів ЦП та для додаткових пристроїв: кеш-пам'яті, блоку арифметики з плаваючою комою, частини основної пам'яті, блоку управління пам'яттю, портів введення / виводу.

Уніфікація набору команд, орієнтація на потокову конвеєрну обробку, уніфікація розміру команд і тривалості їх виконання, усунення періодів очікування в конвеєрі - всі ці чинники позитивно позначаються на загальній

швидкодії. Простий пристрій управління має не багато вентилів і, отже, короткі лінії зв'язку для передачі сигналу управління. Мале число команд, форматів і режимів призводить до спрощення схеми декодування, і воно відбувається швидше. У RISC застосовується пристрій управління з «жорсткою» логікою, швидший за мікропрограмний. Високої продуктивності сприяє і спрощення передачі параметрів між процедурами. Таким чином, застосування RISC веде до скорочення часу виконання програми (або збільшення швидкості) за рахунок скорочення числа циклів на команду.

Простота УП, супроводжувана зниженням вартості і підвищенням надійності, також свідчить на користь RISC. Розробка УП займає менше часу. Простий УП буде містити менше конструктивних помилок і тому більш надійний.

Багато сучасних CISC-машини, такі як VAX 11/780, VAX-8600, мають багато засобів для прямої підтримки функцій мов високого рівня, найбільш частих в цих мовах (управління процедурами, операції з масивами, перевірка індексів масивів, захист інформації, управління пам'яттю і т . д.). Архітектура RISC також має низку засобів для безпосередньої підтримки мов високого рівня і спрощення розробки компіляторів МВР. Завдяки цьому, архітектура RISC в плані підтримки МВР ні в чому не поступається CISC.

Недоліки RISC прямо пов'язані з деякими перевагами цієї архітектури. Принциповий недолік - скорочене число команд: на виконання ряду функцій доводиться витрачати кілька команд замість однієї як в CISC. Це подовжує код програми, збільшує завантаження пам'яті і трафік команд між пам'яттю і ЦП. Дослідження показали, що RISC-програма в середньому на 30% довша від CISC-програми, яка реалізує ті ж функції.

Хоча велика кількість регістрів дає істотні переваги, сама по собі вона ускладнює схему декодування номера регістра, тим самим збільшується час доступу до регістрів.

УП з «жорсткою» логікою, реалізований в більшості RISC-систем, менш гнучкий, більш схильний до помилок, ускладнює пошук і виправлення помилок, поступається при виконанні складних команд.

Однослівна команда виключає пряму адресацію для повної 32-бітової адреси. Тому ряд виробників допускають невелику частину команд подвійної довжини.

Навчальне електронне видання

ПРЕПЕЛИЦЯ ГЕОРГІЙ ПЕТРОВИЧ
ПОНОМАРЕНКО ОЛЕНА ЛЕОНІДІВНА

КОМП’ЮТЕРНА СХЕМОТЕХНІКА ТА АРХІТЕКТУРА КОМП’ЮТЕРІВ

Конспект лекцій

Видавець і виготовлювач
Одеський державний екологічний університет
вул. Львівська, 15, м. Одеса, 65016
тел./факс: (0482) 32-67-35
E-mail: info@odeku.edu.ua
Свідоцтво суб’єкта видавничої справи
ДК № 5242 від 08.11.2016