

Розділ 7

Арифметико-логічний пристрій

7.1. Функції арифметико-логічного пристрою

Арифметико-логічний пристрій (АЛП) призначений для виконання арифметичних, логічних та інших операцій обробки даних над операндами, які представляють собою двійкові числа з фіксованою та рухомою комою, двійково-десяtkові числа, команди, адреси, логічні коди, алфавітно-цифрові коди. АЛП є одним з основних вузлів процесора. Інтерфейс АЛП, тобто його зв'язки з іншими вузлами процесора, показано на рис. 7.1.



Рис. 7.1. Інтерфейс АЛП

Вхідні дані поступають в АЛП з регістрового файла процесора або з основної пам'яті (залежно від типу архітектури комп'ютера), до яких записуються і вихідні дані. Код операції поступає з поля коду операції виконуваної команди, яка зберігається в регистрі команди РгК, а сигнали станів АЛП повідомляють пристрій керування про стан ходу виконання операцій та фіксуються в регистрі слова стану програми регістрової пам'яті процесора.

Арифметико-логічний пристрій процесора – це комбінаційна схема (КС) без внутрішньої пам'яті, яка здатна виконувати набір елементарних операцій та деяку множину складних операцій, які ініціюються командами обробки даних з системи команд комп'ютера. В потужних комп'ютерах, а останнім часом і в багатьох однокристальних комп'ютерах, використовуються багатоблокові АЛП з внутрішньою регістровою пам'яттю на основі табличних, однотактових, багатотактових та конвеєрних операційних пристрій. Тип виконуваної операції вказується кодом на вході керування АЛП. Типово в АЛП виконуються такі операції: зсув – зміщення кодів, які зберігаються в регистрах, вліво або вправо на задане число розрядів; додавання до слова 1 або -1 – операція рахунку; дешифрування – перетворення двійкового коду в однорядний код; шифрування

– перетворення однорядного коду в двійковий; порівняння – визначення відношення старшинства двох слів або їх рівності; порозрядне доповнення – формування оберненого коду; порозрядні логічні множення і додавання двох слів; порозрядне додавання двох слів за модулем; сума двох чисел. В багатьох комп'ютерах цей перелік розширеній більш складними операціями, наприклад арифметичними, відношення, обробки рядків символів, обчислення елементарних функцій і т. д.

Залежно від способу обробки операндів АЛП діляться на послідовні, послідовно-паралельні та паралельні. В першому випадку обробка операндів в АЛП здійснюється послідовно в часі над кожним розрядом, тоді як в останньому операції здійснюються паралельно в часі над всіма розрядами операндів.

За способом представлення чисел розділяють АЛП з фіксованою та з рухомою комою, причому перші можуть бути орієнтовані на обробку цілих або дробових чисел.

Залежно від способу виконання операцій АЛП діляться на однотактові, коли задана операція виконується за один такт, та багатотактові, коли для виконання операції потрібно виконати деяку кількість тактів.

АЛП можуть бути конвеєрними або скалярними. Використання конвеєрного принципу обробки даних дозволяє суттєво підвищити продуктивність АЛП та комп'ютера в цілому.

За характером використання елементів АЛП діляться на одно- та багатоблокові. В одноблокових (багатофункціональних) АЛП всі операції над всіма типами операндів виконуються тими ж вузлами, які комутуються відповідним чином залежно від потрібного режиму роботи. В багатоблокових АЛП окремі групи операцій над кожним типом операндів виконуються окремими блоками. Це дозволяє підвищити продуктивність АЛП за рахунок паралельного виконання операцій.

7.2. Способи обробки даних в арифметико-логічному пристрой

Залежно від способу обробки операндів АЛП діляться на послідовні, послідовно-паралельні та паралельні.

В послідовних АЛП обробка операндів здійснюється послідовно в часі над кожним розрядом, як це показано на рис. 7.2.

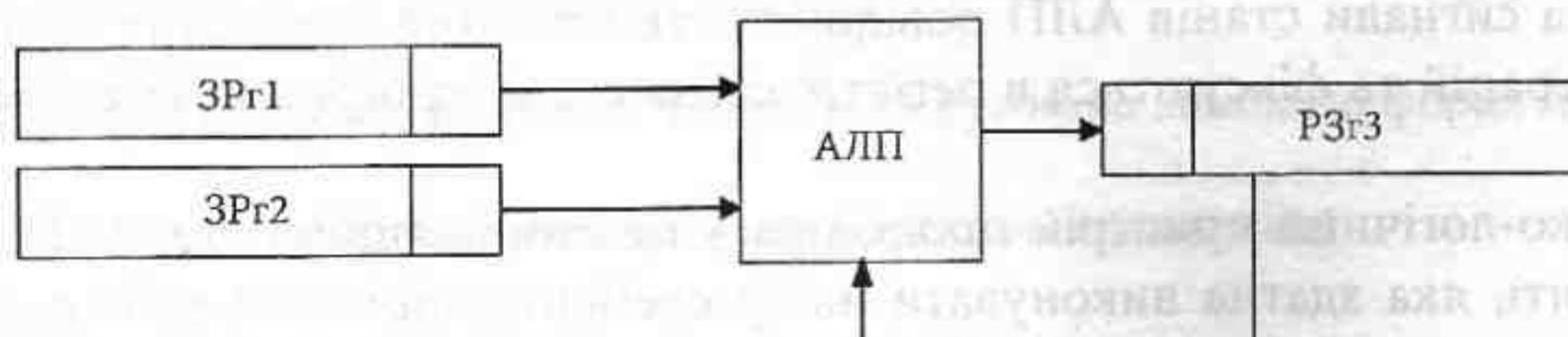


Рис. 7.2. Послідовний спосіб обробки даних в АЛП

Тут на вході АЛП є зсувні реєстри ЗPr1 та ЗPr2, з яких дані порозрядно поступають на обробку. Результат з АЛП також порозрядно поступає в вихідний зсувний реєстр ЗPr3. В кожному такті операнди в зсувних реєстрах зміщуються на один розряд вправо. Крім того, можливий зворотний зв'язок з вихідного реєстра до входу АЛП. Оскільки обробка здійснюється порозрядно, то для отримання результату потрібно як мінімум n тактів, де n – розрядність операндів. Для складних операцій кількість тактів може становити

n^2 і більше. Тобто, при використанні цього способу АЛП характеризується малою швидкодією. Разом з тим, він знаходить досить широке застосування при проектуванні малогабаритних комп'ютерів завдяки малим витратам обладнання на побудову таких АЛП.

В паралельних АЛП операції виконуються одночасно над всіма розрядами операндів, як це показано на рис. 7.3.

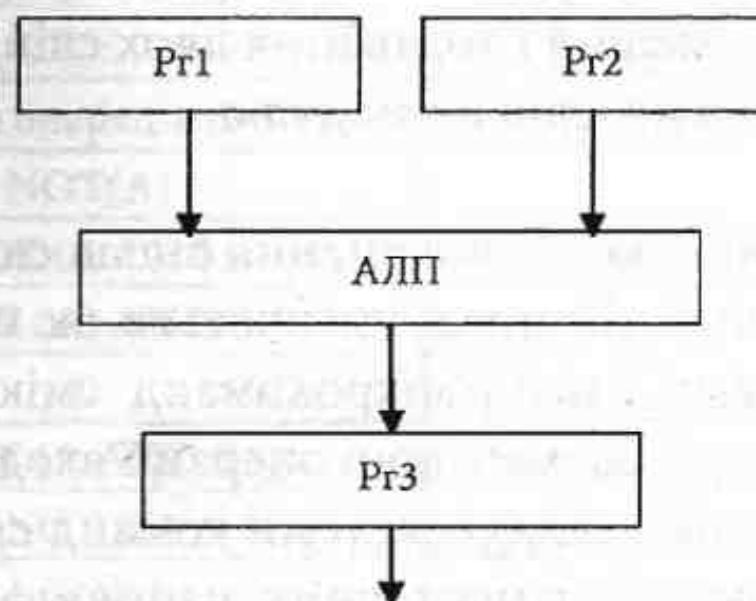


Рис. 7.3. Паралельний спосіб обробки даних в АЛП

Тут на вході АЛП є регістри Pr1 та Pr2, з яких дані паралельно поступають на обробку. Результат також паралельно поступає в вихідний регістр Pr3. Оскільки обробка здійснюється паралельно, вона виконується протягом лише одного такту незалежно від розрядності операндів. Тобто АЛП з паралельним способом обробки даних характеризується високою швидкодією, що і є причиною його широкого використання. Разом з тим, такий АЛП характеризується великими витратами обладнання на його побудову.

Послідовно-паралельний спосіб обробки даних є проміжним стосовно швидкодії та затрат обладнання в порівнянні з вище розглянутими послідовним та паралельним способами. Тут одне з вхідних даних може поступати на обробку в АЛП паралельно, а інше послідовно з видачею проміжного результату в паралельному коді, як це показано на рис. 7.4 а, або вхідні дані можуть поступати в АЛП групами по k і m розрядів, як це показано на рис. 7.4 б, та подаватись в вихідний регистр паралельно, або також групами.

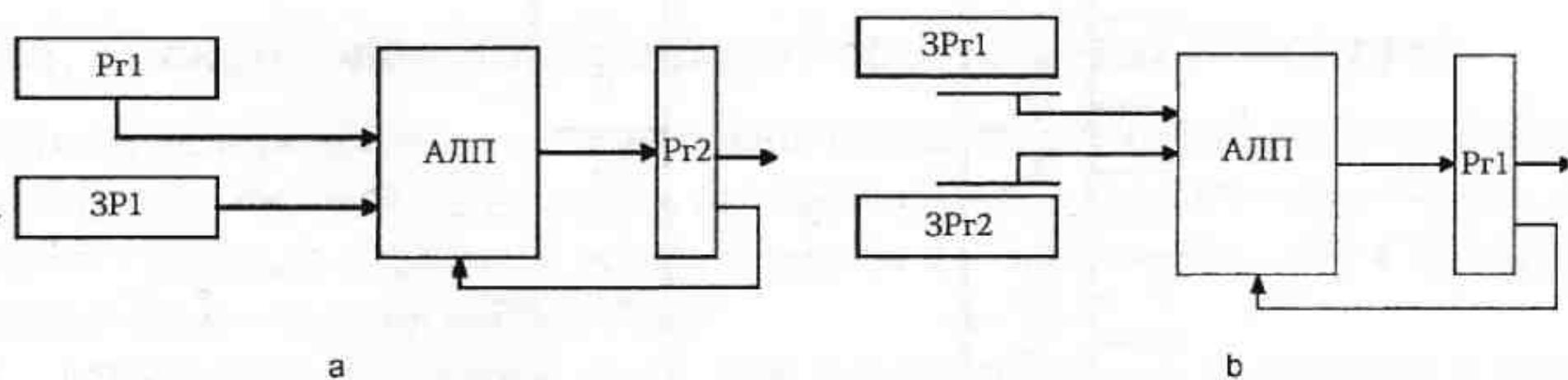


Рис. 7.4. Послідовно-паралельний спосіб обробки даних в АЛП

7.3. Елементарні операції арифметико-логічного пристрою

Складні операції в АЛП реалізуються як послідовність елементарних, тому АЛП будується на основі комбінаційних схем КС, які виконують елементарні операції. До типових елементарних операцій належать:

- зсув – зміщення кодів, які зберігаються в регистрі, вліво або вправо на задане число розрядів;

- додавання до слова 1 або -1 – операція рахунку;
- дешифрування перетворення слів в сигнали (однорядний код);
- шифрування ^{коду} перетворення однорядного коду в двійковий;
- порівняння визначення відношення старшинства двох слів або їх рівності;
- порозрядне доповнення формування оберненого коду;
- порозрядне логічне множення і додавання двох слів;
- порозрядне додавання двох слів по модулю;
- сума двох чисел

Елементарні операції є основою для виконання більш складних операцій процесора. Алгоритми виконання цих операцій представляються як послідовність елементарних, які називаються мікрокомандами, а набір мікрокоманд – мікропрограмами. Більше того, в більшості сучасних комп’ютерів елементарні операції входять до складу їх системи команд, не дивлячись на наявність в складі системи команда складних операцій, які вимагають виконання великої кількості елементарних, наприклад операцій компресії даних, шифрування даних і т. д. Це пояснюється двома причинами: наявність в складі системи команда комп’ютера команда виконання елементарних операцій забезпечує його універсальність, і, крім того, ці операції виконуються гранично швидко, що дозволяє досягти високих тактових частот роботи процесора.

На основі комбінаційних схем для виконання вищезазначених елементарних операцій синтезуються вузли АЛП для виконання складних операцій, що буде показано далі.

Арифметико логічний пристрій для виконання елементарних операцій наявний в кожному універсальному комп’ютері. Розглянемо побудову стандартного 4-розрядного АЛП, функціональне позначення та входи-виходи якого показано на рис. 7.5. Інтерфейс АЛП включає дві вхідні (A і B) та одну вихідну 4-розрядну шину даних. Дані з вхідних шин обробляються в АЛП відповідно до значення двійкового коду на входах керування M та S₀-S₃. Результат обробки поступає на вихідну шину F.

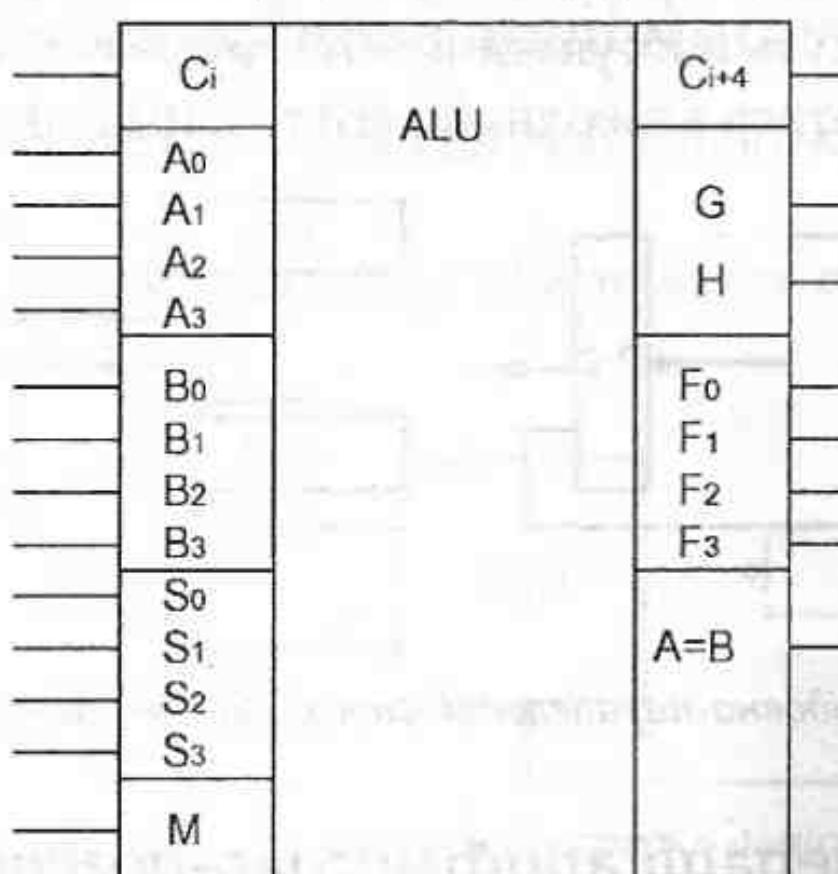


Рис. 7.5. Входи-виходи стандартного 4-розрядного АЛП

Для нарощування розрядності такі АЛП об’єднуються шляхом відповідного з’єднання входів переносу C_i та C_{i+4} . Основою АЛП служить суматор, що виконує операцію додавання двох чисел, схема якого доповнюється відповідними логічними елементами для розширення функцій та забезпечення переключення з однієї операції на іншу. Вхід

M (mode) задає тип виконуваної операції: логічна ($M=1$) чи арифметико-логічна ($M=0$). Виходи G і H задають функції генерації і прозорості, які використовуються для організації паралельних переносів при нарощуванні розрядності АЛП з використанням схем прискореного переносу. Вихід $A=B$ є виходом порівняння кодів A та B на збіжність.

Перелік операцій, виконуваних описаним АЛП, приведено в табл. 7.1.

Таблиця 7.1

S	Логічна операція ($M=1$)	Арифметико-логічна операція ($M=0$)
0000	NOT(A)	$A + C_i$
0001	NOT (A OR B)	$A \text{ OR } B + C_i$
0010	NOT (A) AND B	$A \text{ OR NOT } (B) + C_i$
0011	0	$1 + C_i$
0100	NOT (A AND B)	$A + A \text{ AND NOT } (B) + C_i$
0101	NOT (B)	$A \text{ OR } B + A \text{ AND NOT } (B) + C_i$
0110	A XOR B	$A + \text{NOT } (B) + C_i$
0111	A AND NOT (B)	$A \text{ AND NOT } (B) + 1 + C_i$
1000	NOT (A) OR B	$A + A \text{ AND } B + C_i$
1001	NOT (A XOR B)	$A + B + C_i$
1010	B	$A \text{ OR NOT } (B) + A \text{ AND } B + C_i$
1011	A AND B	$A \text{ AND } B + 1 + C_i$
1100	1	$A + A + C_i$
1101	A OR NOT (B)	$A \text{ OR } B + A + C_i$
1110	A OR B	$A \text{ OR NOT } (B) + A + C_i$
1111	A	$A + 1 + C_i$

В таблиці прийняті наступні позначення: OR – операція диз'юнкції, AND – операція кон'юнкції, XOR – операція нерівнозначності, “+” – операція додавання, “–” – операція віднімання. Позначенням 1 та 0 в таблиці відповідають двійкові коди відповідно 1111 та 0000. Вхідний перенос поступає в молодший розряд слова, тобто до слова додається код $000C_i$.

7.4. Складні операції арифметико-логічного пристрою

Крім вище перерахованих елементарних операцій, в АЛП виконується велика кількість складних операцій, тобто таких, які реалізуються на основі елементарних. Можна виділити наступний перелік складних операцій АЛП, сформований на основі аналізу системи команд сучасних комп'ютерів:

- логічні операції (логічне множення, логічне додавання, інверсія і т. д.) над двійковими числами;
- операції зсуву (вправо, вліво) на задану кількість розрядів, причому в одному такті зсув може бути здійснено як на один розряд, так і на декілька розрядів;
- арифметичні операції (додавання, віднімання, множення та ділення) над двійковими числами;
- операції відношення: менше, більше, рівне, менше-рівне, більше-рівне;
- операції обчислення елементарних функцій типу $\exp X$, $\ln X$, $\sin X$, $\cos X$, $\operatorname{Sh} X$, $\operatorname{Ch} X$, піднесення до степеня A^m ; $\operatorname{arctg} y/x$;
- операції обробки символів та рядків символів.

Потрібно відзначити, що розглянуті в розділі 4 операції перетворення даних (перетворення із формату з фіксованою в формат з рухомою комою і навпаки, перетворення з двійково-десяtkового коду в двійковий та навпаки і т. д.), так само як операції реорганізації масивів і визначення їх параметрів: сортування, пошук максимуму або мінімуму, вибір заданого масиву, зсув елементів масиву, стиск масиву, а також операції пошуку символу, зсув, заміна символів в рядку, пакування рядків символів, порівняння рядків символів виконуються в процесорі на основі елементарних та основних арифметичних і логічних операцій. Разом з тим, в останніх комп'ютерах з метою підвищення продуктивності та в зв'язку з широким використанням засобів телекомуникацій та мультимедіа до складу АЛП вводяться окремі блоки для виконання вищезазначених складних операцій, а також операцій типу кодування, компресії, шифрування даних і т. д.

Розглянемо питання реалізації в АЛП складних операцій більш детально.

7.5. Використання графа алгоритму при побудові арифметико-логічного пристрою

Щоб виявити ефективні підходи до реалізації алгоритму виконання складної операції в АЛП, необхідно представити його в формі, яка розкриває його базові обчислюальні та структурні характеристики. Обчислюальні характеристики описуються повним набором функціональних операторів алгоритму і відповідними їм обчислюальними затримками. Структурні характеристики описуються зв'язками між функціональними операторами алгоритму і їх взаємозалежністю.

Однією із можливих форм представлення алгоритму є графічна. На рис. 7.6а представлений граф деякого алгоритму.

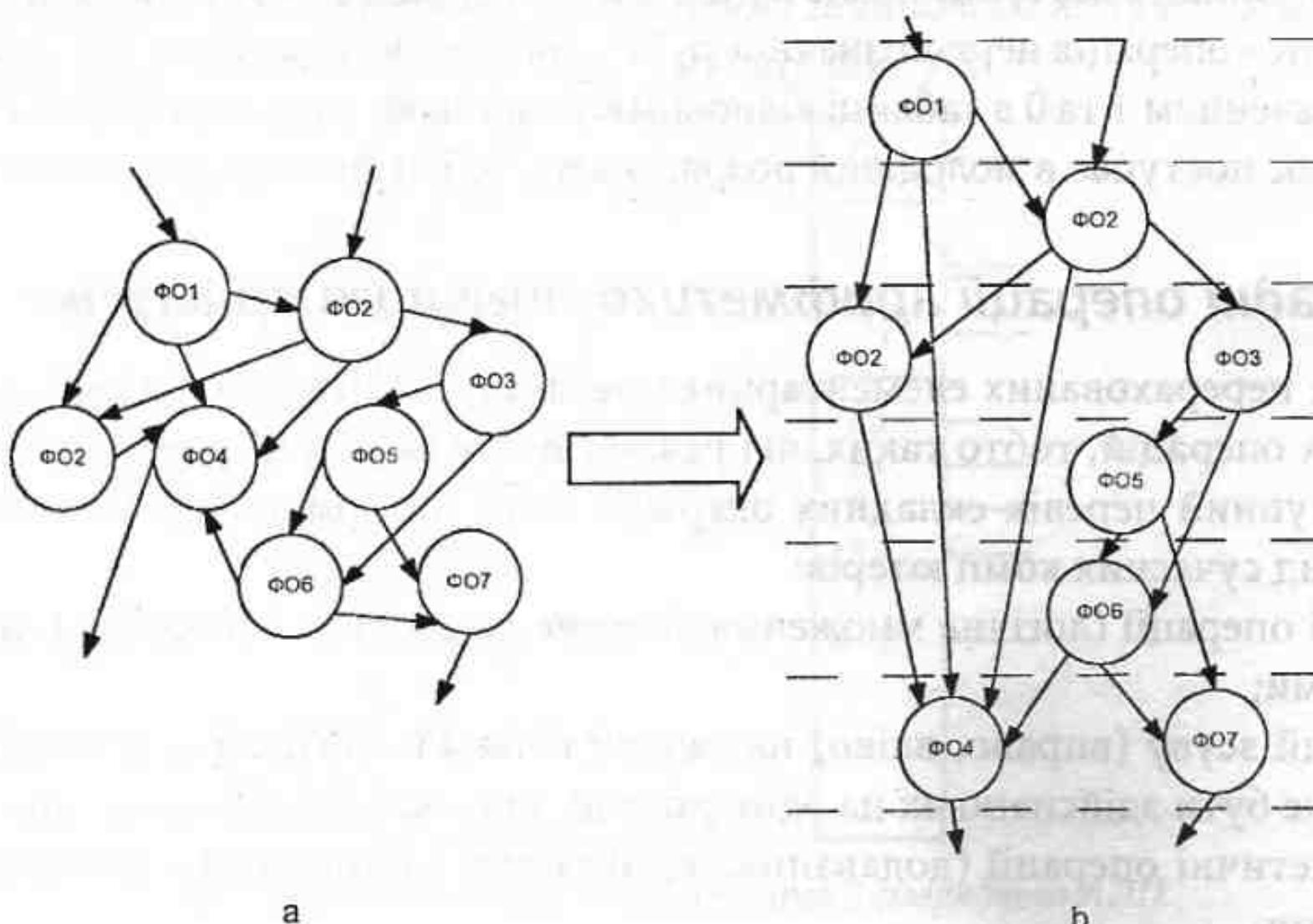


Рис. 7.6. Граф алгоритму

Кожна вершина графа представляє функціональний оператор алгоритму, кожна дуга – зв'язок між функціональними операторами. Вершина, яка відповідає функціональному оператору Φ_i , з'єднується з вершиною, яка відповідає функціональному оператору

Ф_j, тільки в тому випадку, коли результат, одержаний після виконання оператора Ф_i, є одним із аргументів для оператора Ф_j. Описана графічна форма представлення алгоритму дозволяє оцінити його обчислювальні характеристики, тому широко використовується при синтезі АЛП.

Представлення графа алгоритму в такому вигляді не дозволяє в повній мірі оцінити його структурні характеристики, а значить і можливості його розпаралелення та способи апаратурної реалізації.

Виявити в алгоритмі паралелізм і навіть керувати ним, забезпечуючи тим самим можливість знаходження компромісних просторово-часових співвідношень, що є головним при виборі структури обчислювального пристрою, дозволяє представлення графа алгоритму в потоковій (ярусно-паралельній) формі. Таке представлення графа означає розділення всіх його вершин по ярусах таким чином, що в i-му ярусі розміщені тільки функціональні оператори, які залежать хоча б від одного функціонального оператора (i-1)-го яруса і не залежать від функціональних операторів, які не входять в яруси з меншими ніж i номерами. Всередині яруса функціональні оператори між собою не мають з'єднань. Описаний граф називають потоковим графом (ПГ) алгоритму. На рис. 7.6b представлена ПГ раніше розглянутого алгоритму, де штриховими лініями розділені яруси графа.

ПГ алгоритму характеризується набором функціональних операторів Ф_{ij} ($i=1, 2, \dots, P$; $j=1, 2, \dots, L_i$), де i – номер яруса, P – кількість ярусів, j – номер функціонального оператора в ярусі, L_i – їх кількість в ярусі, а також набором каналів K_{ij} зв'язку між функціональними операторами.

При розгляді питань проектування пристрою для виконання деякого алгоритму у вигляді надвеликої інтегральної схеми (НВІС), використовують його конкретизований потоковий граф. В конкретизованому потоковому графі алгоритму всі функціональні оператори входять до бібліотеки функціональних елементів НВІС, на базі якої реалізується алгоритм. Тобто кожному функціональному оператору алгоритму може бути поставлений у відповідність як мінімум один функціональний елемент із бібліотеки функціональних елементів НВІС. Таке представлення алгоритму забезпечує можливість пошуку найефективнішого варіанту його апаратурної реалізації.

7.6. Виконання складних операцій в арифметико-логічному пристрой

Виконання деякої операції можна здійснити цілим рядом способів, для яких характерна єдина семантична основа, що визначається графом алгоритму цієї операції, тобто одні і ті ж операції можуть виконуватися по різному та з різним ступенем розпаралелювання.

Якщо використовувати для виконання графа алгоритму багатофункціональний АЛП, який виконує елементарні операції, наприклад, відповідно до табл. 7.1, то це потребує перетворення всіх процесів, які підлягають виконанню за даним алгоритмом, в послідовну процедуру переробки і передачі інформації. Більш того, значна частина функціональних операторів алгоритму реалізується в такому АЛП з використанням достатньо великого числа елементарних операцій. Разом з цим, просторова передача інформації між вершинами графа повинна бути перетворена в послідовну в часі передачу інформації між АЛП і основною пам'яттю або регістрами процесора. Таким чином, послідовно-паралельний процес виконання алгоритму згідно з його графом, що має просторову структуру, перетвориться в

послідовну процедуру переробки і передачі інформації в багатофункціональному АЛП. Це призводить до того, що швидкість обчислення складних операцій в АЛП є низькою.

Висока продуктивність обробки даних в сучасних комп'ютерах досягається завдяки використанню просторового і часового паралелізму з урахуванням досягнень інтегральної технології. Серед основних ідей слід виділити конвеєрну організацію обчислень, поєднання обробки на різних рівнях, використання декількох паралельно працюючих АЛП з конвеєрним принципом обробки даних, застосування секцій векторних реєстрів в реєстровій пам'яті, розшарування пам'яті, застосування найбільш швидкої елементної бази з тією метою, щоб в максимальній мірі забезпечити розпаралелювання виконання операцій. Особливо цікавою в цьому плані є ідея підвищення продуктивності та ефективності за рахунок статичної і динамічної програмної зміни структури системи в цілях її наближення до структури графа алгоритму, тобто її реконфігурація. В основі концепції АЛП з реконфігурованою структурою лежить принцип, згідно з яким структура АЛП, функції його операційних пристройів, а також способи організації обчислювального процесу повинні відповісти структурі графа виконуваного алгоритму, а не навпаки, як це має місце у традиційних АЛП, коли в цілях реалізації алгоритму в рамках жорсткої структури АЛП алгоритм модифікується так, що йому надається форма, відповідна структурі АЛП. Чим більше адаптована структура АЛП до специфіки вирішуваних задач, тим суттєвіше покращуються його характеристики. Така адаптація досягається шляхом перекладення обчислювальних процедур з програмного забезпечення на апаратну частину, чому сприяють вражаючі успіхи в області інтегральної технології. Це, зокрема, поява програмованих логічних інтегральних схем (ПЛІС), що дозволяє в найкоротші терміни реалізувати на їх базі крупні функціональні вузли. Це і можливість створення спеціалізованих НВІС місткістю в десятки і сотні мільйонів транзисторів, що привело до радикальних змін як в області технологічних та інструментальних засобів, так і в принципах проектування.

Таким чином, розглядаючи принципи побудови АЛП, доцільно це робити в тісному зв'язку з графом виконуваного алгоритму, що дозволяє виявити всі форми паралелізму, наявні в ньому. Граф алгоритму, а особливо його потокова форма, в наочній формі відображає фундаментальні обмеження, що визначають внутрішню семантику ряду різних алгоритмів для отримання одних і тих же результатів. Найбільш високі параметри досягаються в АЛП, структура яких більше наближена до структури графа виконуваного алгоритму.

7.7. Структура арифметико-логічного пристрою

В більшості комп'ютерів АЛП виконує операції над двома входними даними, тобто є двомісним, та видає один вихідний результат, як це показано на рис. 7.7. При цьому спочатку операнди A та B записуються у входні реєстри Pr1 і Pr2, та поступають на входи АЛП через мультиплексори МП1 і МП2, які керуються сигналами Y1 та Y2. Після цього в АЛП виконується задана операція, тип якої задається кодом операції. Результат операції поступає на вихід АЛП та записується у вихідний реєстр Pr3. З виходу вказаного реєстра результат поступає в реєстровий файл процесора, а крім того, якщо він потрібний для виконання наступної операції, він поступає через мультиплексори МП1 або МП2 на один з входів АЛП, що здійснюється шляхом подання відповідних значень керуючих сигналів на ходи мультиплексорів.

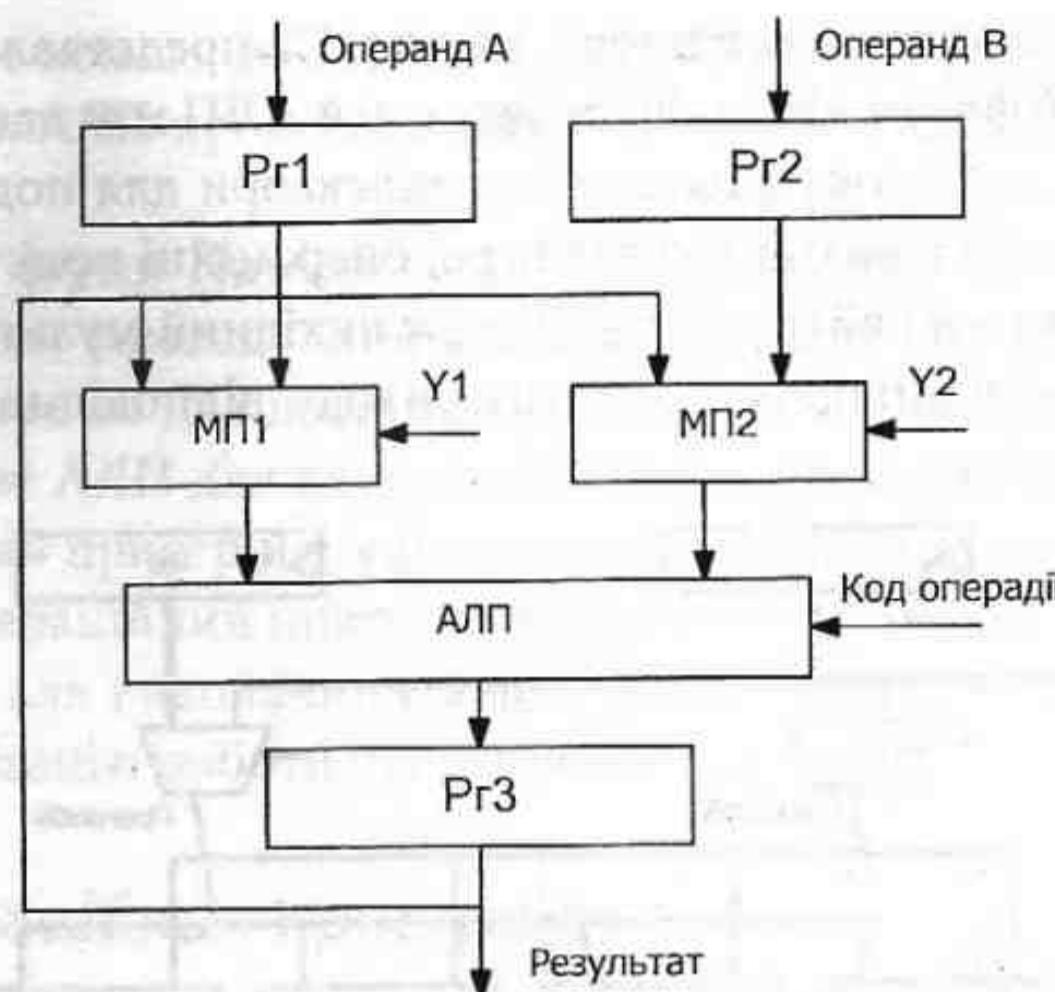


Рис. 7.7. Типова структура АЛП

В сучасних комп’ютерах АЛП є багатоблоковими. В них окрім групи операцій над кожним типом операндів виконуються окремими блоками, які називаються операційними пристроями. Це дозволяє підвищити продуктивність АЛП за рахунок паралельного виконання операцій. Узагальнена структура АЛП сучасного комп’ютера представлена на рис. 7.8.



Рис. 7.8. Структура АЛП сучасного комп’ютера

До складу АЛП, крім вищеописаного пристрою для виконання елементарних операцій, входять і операційних пристройів ОП1, ОП2, ..., ОП n , які виконують складні операції. Кількість і цих пристройів та їх функції визначаються конструкторами комп’ютера залежно від сфери його використання. Входи та виходи операційних пристройів АЛП підключаються до його інформаційних входів та виходів за допомогою комутуючих мереж, якими керує код виконуваної операції. Цим же кодом вибирається тип виконуваної операції в пристройі для виконання елементарних операцій та в операційному пристройі, якщо він може виконувати декілька операцій.

Як приклад АЛП реального комп'ютера, на рис. 7.9 представлено АЛП програмованого процесора NIOS 2.0 фірми Altera. Як бачимо, цей АЛП має наступні блоки: два вхідних регістри RA та RB, два двовходових мультиплексори для подачі даних на обробку або з вхідних регістрів, або з вихідного регістра, операційні пристрої – арифметичний, логічний, зсуву та виділення байтів і слів, а також вихідний мультиплексор, необхідний для підключення до входу вихідного регістра вихіду відповідного операційного пристрою, і сам вихідний регістр.

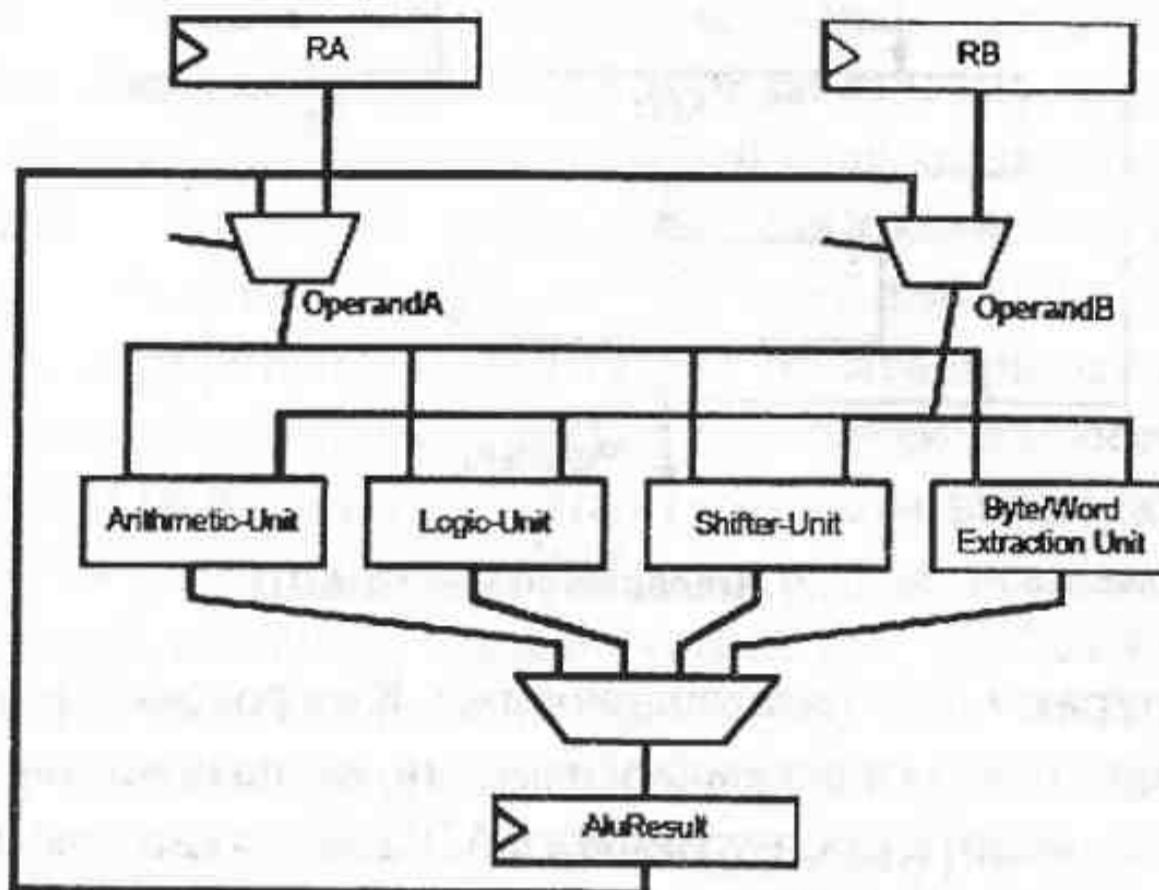


Рис. 7.9. Структура АЛП програмованого процесора NIOS 2.0 фірми Altera

Ще більшу кількість паралельних блоків мають АЛП процесорів UltraSPARC фірми Sun Microsystems та PA-8000 фірми Hewlett-Packard, структури яких наведено на рис. 7.10a та рис. 7.10b відповідно.

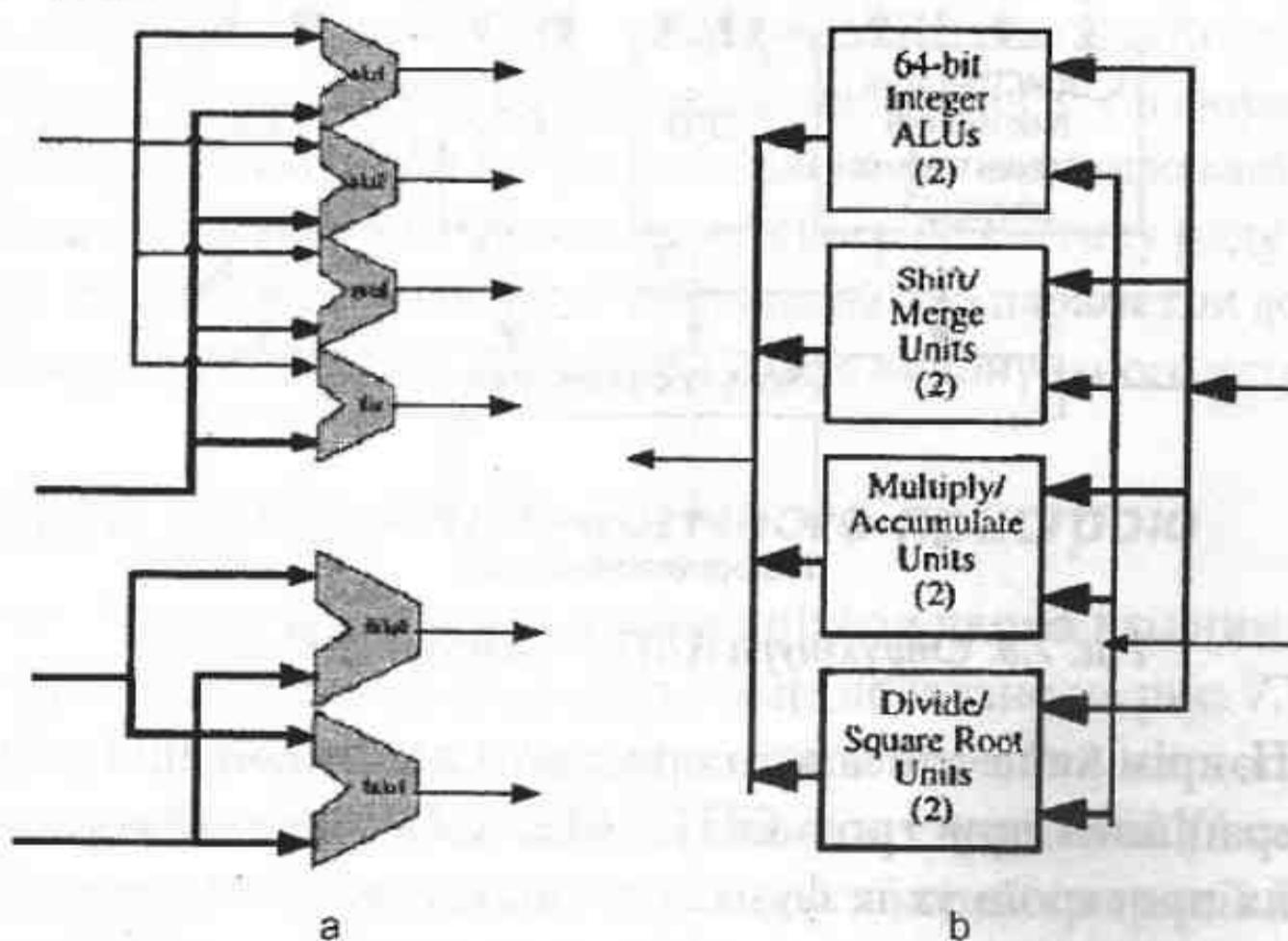


Рис. 7.10. Структури АЛП процесорів UltraSPARC фірми Sun Microsystems (a) та PA-8000 фірми Hewlett-Packard (b)

АЛП процесора UltraSPARC фірми Sun Microsystems має два блоки для виконання елементарних операцій над цілими числами (ALU1 та ALU2), перемножувач та поділь-

ник чисел з фіксованою комою (MUL, DIV), два АЛП для обробки даних з рухомою комою (FALU0 та FALU1). АЛП процесора UltraSPARC може виконувати до чотирьох операцій за один такт.

Процесор PA-8000 фірми Hewlett-Packard є суперскалярним процесором з невпорядкованим виконанням команд, який може виконувати до чотирьох команд за один такт. Його АЛП включає по два наступних блоки (длочки, розміщені зверху донизу на рис. 7.10b): 64-розрядне АЛП для виконання елементарних операцій над цілочисловими даними, операційний пристрій зсуву та сортування, операційний пристрій множення та накопичення, операційний пристрій ділення та добування квадратного кореня.

Функції пристрою для виконання елементарних операцій раніше було розглянуто. Розглянемо далі організацію роботи операційних пристрій.

7.8. Типи операційних пристрій

Залежно від принципів побудови можна провести наступну класифікацію операційних пристрій:

- Табличні операційні пристрої – операційні пристрої, в яких значення результату шукається за адресою, рівною значенню операнда, в наперед сформованій таблиці, в ролі якої використовується пам'ять.
- Алгоритмічні операційні пристрої – операційні пристрої, в яких задана операція виконується шляхом апаратної реалізації алгоритму цієї операції.
- Таблично-алгоритмічні операційні пристрої – стиснута в таблиці (пам'яті) інформація відновлюється шляхом виконання елементарних та складних операцій.

Під апаратною реалізацією алгоритму розуміється його виконання без участі програміста (програми, яка зчитується з основної пам'яті). При цьому алгоритмічні та таблично-алгоритмічні операційні пристрої в свою чергу діляться на:

- Багатотактові операційні пристрої – операційні пристрої, в яких задана операція виконується шляхом послідовного потактового виконання функціональних операторів алгоритму цієї операції. Потактове виконання операцій в таких пристроях задається мікропрограммою, яка складається з мікрокоманд, що виконують елементарні операції.
- Однотактові операційні пристрої – операційні пристрої, в яких апаратно відображається граф виконуваного алгоритму і задана операція виконується шляхом одноразового проходження даних через операційні вузли, які виконують функціональні оператори алгоритму цієї операції. Виходячи з принципів побудови, такі пристрої можна назвати граф-алгоритмічними операційними пристроями.
- Конвеєрні операційні пристрої – операційні пристрої, в яких апаратно відображається потоковий граф виконуваного алгоритму з розділенням ярусів графа регістрами. Виходячи з принципів побудови, такі пристрої можна назвати конвеєрними граф-алгоритмічними операційними пристроями.

Розглянемо спочатку будову та особливості найпростіших з названих табличних операційних пристрій, далі принципи побудови багатотактових, однотактових, та конвеєрних операційних пристрій, основні ідеї, закладені в таблично-алгоритмічних операційних пристроях, а після цього питання організації роботи операційних пристрій, побудованих за принципами приведеної вище класифікації, для виконання конкретних операцій.

7.9. Табличний операційний пристрій

Наявність на ринку швидкодіючих широкорозрядних пристріїв напівпровідникової пам'яті великого об'єму в інтегральному виконанні, та можливість розміщення такої пам'яті на кристалі разом з процесором, зробили реальним (при невисоких вимогах до точності), використання табличного методу виконання операцій. За цим методом значення результату шукається в наперед сформованій таблиці, в ролі якої використовується пам'ять, за адресою, рівною значенню операнда.

Розглянемо принцип формування таблиці на прикладі виконання операції $Y = X^2$. В табл. 7.2 наведені двійкові коди значень operandів X , які одночасно є адресами, та значень operandів Y , які є вмістом відповідних комірок пам'яті.

Таблиця 7.2

Двійкове значення X (адреса)	Двійкове значення X (вміст комірок пам'яті)
000	000000 0^2
001	000001 1^2
010	000100 2^2
011	001001 3^2
100	010000 4^2
101	011001 5^2
110	100100 6^2
111	110001 7^2

Тут розрядності X та Y вибрані рівними відповідно 3 та 6 бітам.

Запис у пам'ять необхідної інформації може здійснюватися або розроблювачем цифрового пристрою при використанні оперативних та перепрограмових постійних запам'ятовуючих пристрій (ОЗП та ППЗП), або виготовлювачем замовних ПЗП. Важливим при цьому є те, що виробництво замовних ПЗП не вимагає від виготовлювача тих великих витрат, з якими звичайно зв'язана розробка і виробництво спеціалізованих НВІС, тому що зводиться до виключення частини елементів заздалегідь розробленої топології типової НВІС. При використанні перепрограмових ПЗП, що допускають електричний перезапис збереженої інформації, з'являється можливість зміни виконуваних пристроєм функцій без зміни його структури.

Структура табличного операційного пристроя показана на рис. 7.11.

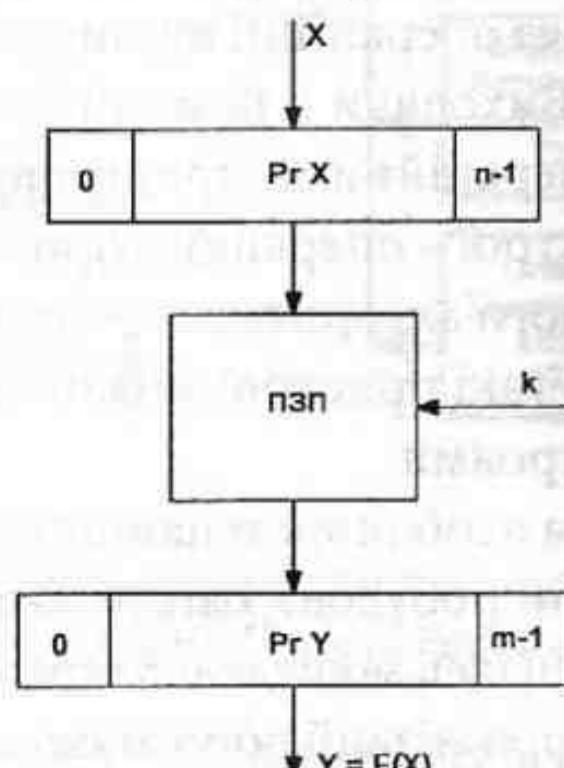


Рис. 7.11. Структура табличного операційного пристроя

Як видно з рисунку, пристрій складається з вхідного n -роздрядного реєстра PrX , де n – розрядність вхідних даних, та вихідного m -роздрядного реєстра PrY , де m – розрядність вихідних даних, а також ПЗП для зберігання табличних значень функції $Y = F(X)$.

При необхідності обчислення K функцій, таблицю кожної з них можна записати до ПЗП. Тоді додатково на вхід ПЗП потрібно подати k -роздрядний код ($k = \log_2 K$) для задання типу обчислюваної функції.

Застосування табличного методу дозволяє забезпечити мінімально можливий час обчислення, обумовлений часом вибірки з пристрою пам'яті $T = t_{\text{ПЗП}}$.

Однак табличний метод має й недоліки, які обмежують його використання. Основним з них є значний об'єм пам'яті, який визначається з виразу $Q = Km2^n$. При малих розрядностях аргументів об'єм ПЗП є незначним, однак при обробці аргументів з великою розрядністю n та при великій кількості K обчислюваних функцій, застосування описаного методу є проблематичним, або й нереальним.

Розглянемо приклад. Нехай розрядність вхідних та вихідних даних рівна $n=m=8$ бітів, а кількість виконуваних функцій $K=4$. Тоді об'єм ПЗП буде рівним $Q = 4*8*2^8 = 1\text{KB}$, що цілком прийнятно для реалізації.

Розглянемо інший приклад. Нехай розрядність вхідних та вихідних даних рівна $n=m=16$ бітів, а кількість виконуваних функцій $K=4$. Тоді об'єм ПЗП буде рівним $Q = 4*16*2^{16} = 0.5\text{MB}$, що дещо проблематично, але також прийнятно для реалізації.

Зрозуміло, що при більших розрядностях аргумента використання описаного методу є недоцільним.

Вище показано структуру табличного операційного пристрою для виконання одномісних операцій, тобто обчислення функцій одного аргументу. Його можна використати також і для виконання багатомісних операцій, тобто для обчислення функцій більшої кількості аргументів. На рис. 7.12 показана структура табличного операційного пристрою для виконання операцій над двома аргументами.

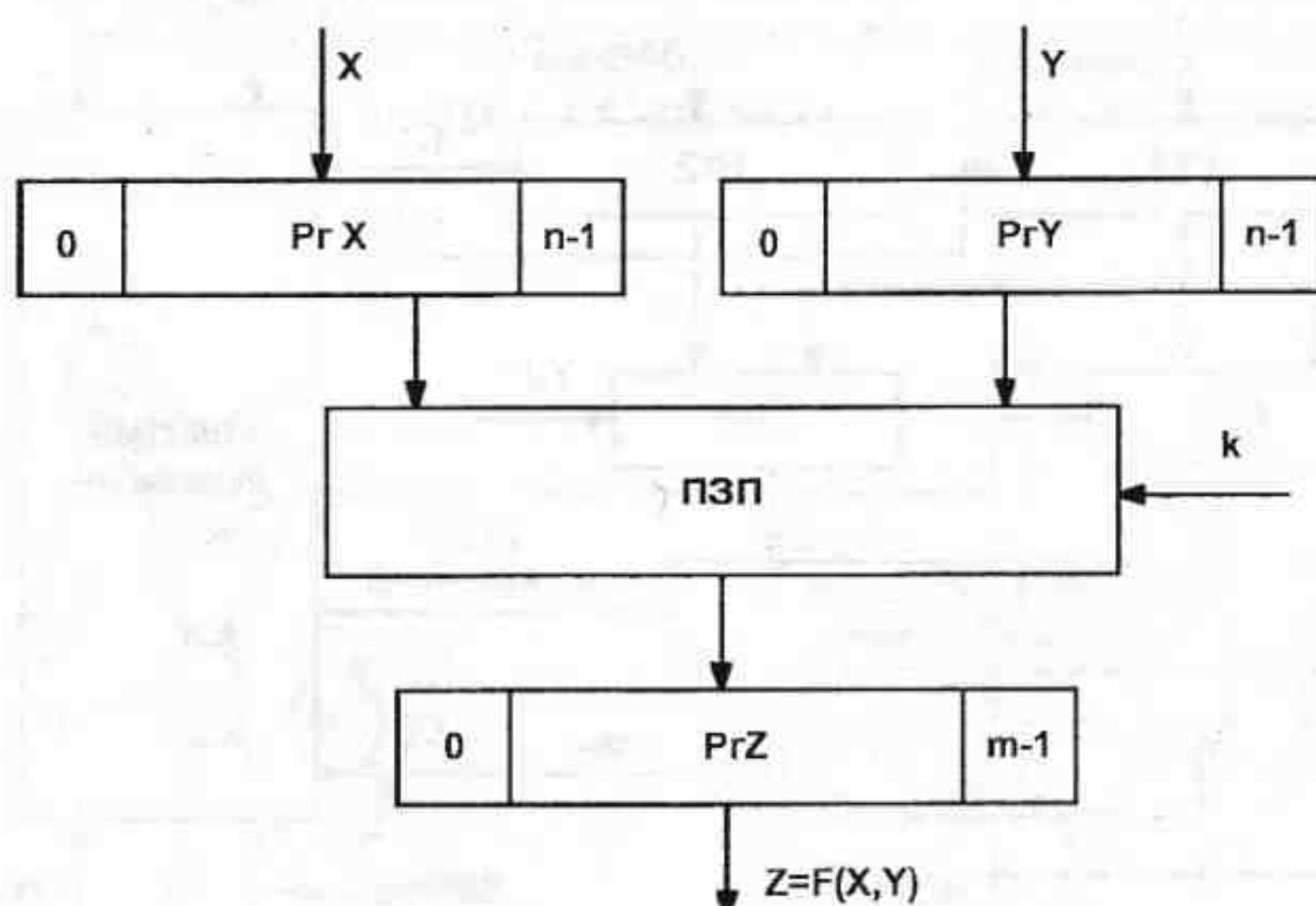


Рис. 7.12. Структура табличного операційного пристрою для виконання операцій над двома аргументами

Об'єм пам'яті для даного пристрою визначається з виразу $Q = Km2^{2n}$. Для багатомісних операцій об'єм ПЗП визначається з виразу $Q = Km2^{ln}$, де l – кількість аргумент-

тів. Вже навіть при невеликих розрядностях аргументів об'єм ПЗП для даного випадку є значним, тому цей метод рідко застосовується для виконання операцій над більш ніж одним аргументом.

Розглянемо приклад. Нехай розрядність вхідних та вихідних даних рівна $n=m=8$ бітів, а кількість виконуваних функцій $K=4$. Тоді об'єм ПЗП буде рівним $Q = 4 \cdot 8 \cdot 2^{16} = 256\text{-KB}$, що цілком прийнятно для реалізації.

Розглянемо інший приклад. Нехай розрядність вхідних та вихідних даних рівна $n=m=16$ бітів, а кількість виконуваних функцій $K=4$. Тоді об'єм ПЗП буде рівним $Q = 4 \cdot 16 \cdot 2^{32} = 32\text{GB}$, що реалізувати на даний час неможливо.

До інших недоліків табличного методу, які також обмежують його використання, належать:

- зростання часу звертання до пам'яті при збільшенні її об'єму;
- великий обсяг попередніх обчислень для розрахунку вмісту таблиць;
- великі витрати часу на запис обчислених значень у ПЗП.

Таким чином, використання табличних операційних пристрій є доцільним при маліх розрядностях аргументів.

7.10. Багатотактовий операційний пристрій

До складу багатотактового операційного пристрою, структура якого приведена на рис. 7.13, входять: багатофункціональна комбінаційна схема, вхідні і вихідні регістри, а також мультиплексори, необхідні для створення каналів передавання даних. Формування керуючих сигналів для запису даних до регістрів, задання коду операції та керування пропуском даних через мультиплексори, здійснює пристрій керування операційним пристроєм, принципи побудови якого будуть розглянуті в наступному розділі.

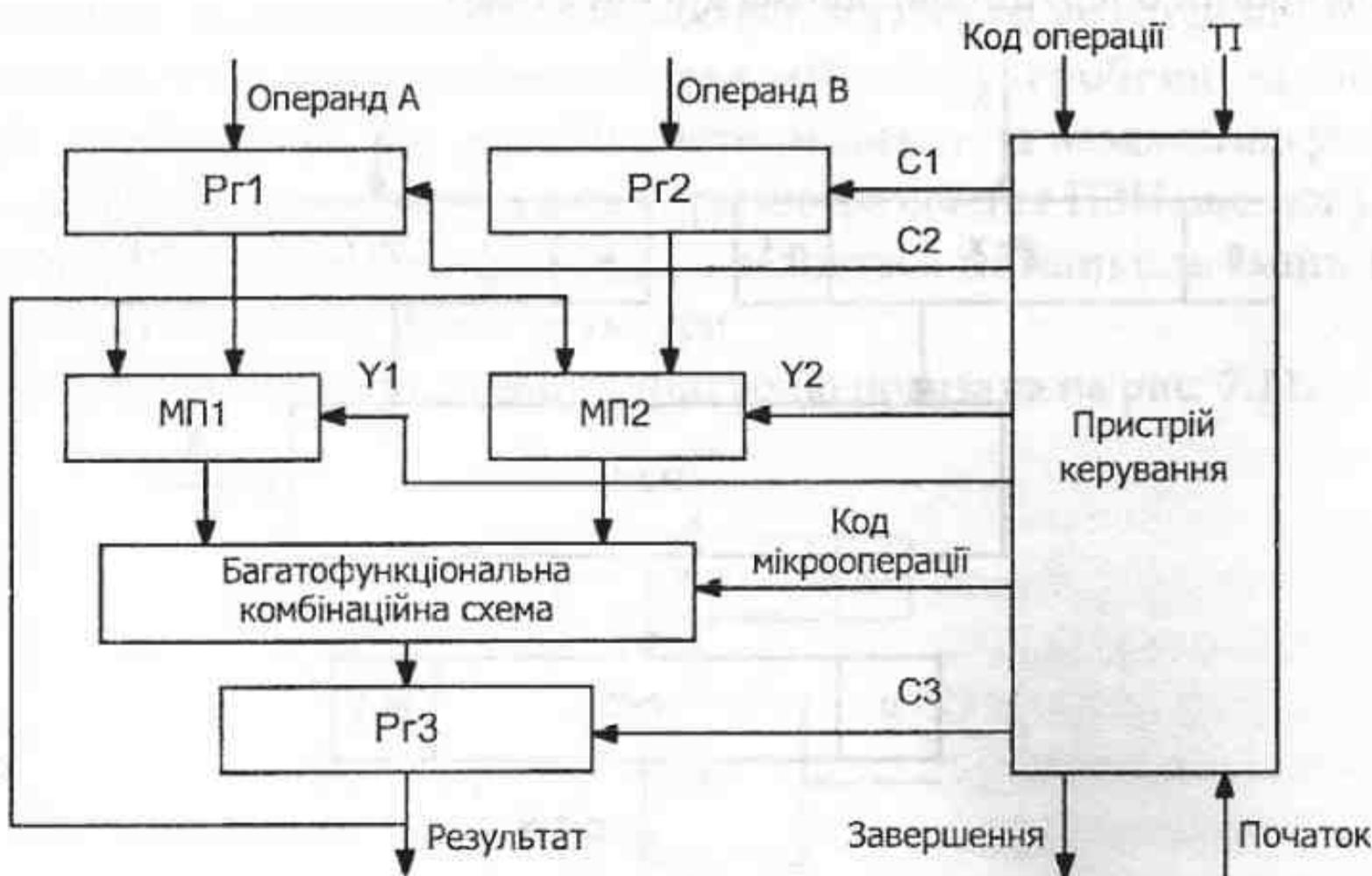


Рис. 7.13. Структура багатотактового операційного пристроя

Багатотактовий операційний пристрій виконує операції над двома вхідними даними, тобто є двомісним, та видає один вихідний результат, як це показано на рис. 7.13. При цьому спочатку операнди A та B записуються у вхідні регістри Pr1 і Pr2, та по-

ступають на входи багатофункціональної комбінаційної схеми через мультиплексори МП1 та МП2, які керуються сигналами Y_1 та Y_2 , що поступають з пристрою керування. Після цього в багатофункціональній комбінаційній схемі виконується задана мікрооперація, тип якої задається кодом мікрооперації, що поступає з пристрою керування. Результат мікрооперації поступає на вихід багатофункціональної комбінаційної схеми та записується у вихідний регистр Рг3. З виходу вказаного регистра проміжний результат поступає через мультиплексори МП1 або МП2 на один з входів багатофункціональної комбінаційної схеми для виконання наступної мікрооперації, що здійснюється шляхом подання з пристрою керування відповідних значень керуючих сигналів Y_1 та Y_2 на входи мультиплексорів. Запис даних до registrів здійснюється сигналами С1-С3 з пристрою керування. Пристрій керування починає виконання операції після поступлення на його входи сигналу початку роботи, коду операції та тактових імпульсів ТІ. Після завершення виконання операції на виході регистра Рг3 буде результат операції, а пристрій керування сформує сигнал завершення роботи.

При побудові багатотактового пристроя виникає дві задачі: синтез багатофункціональної комбінаційної схеми, яка б забезпечувала реалізацію всіх функціональних операторів алгоритму, та синтез пристроя керування, який генерує сигнали керування для забезпечення організації виконання функціональних операторів алгоритму шляхом тимчасового запам'ятовування проміжних результатів в регистрах та подання їх на входи багатофункціональної комбінаційної схеми в потрібному такті.

Для синтезу багатофункціональної комбінаційної схеми та пристроя керування потрібно визначити всі мікрооперації, які необхідно виконати для реалізації заданої операції та послідовність їх виконання. Це можна зробити, зокрема, шляхом переходу від просторового графа алгоритму виконання операції до часового графа цього алгоритму, як це показано на рис. 7.14. На основі представленого на рис. 7.14а просторового графа алгоритму можна визначити кількість та типи функціональних операторів, які мають бути виконані багатофункціональною комбінаційною схемою та послідовність їх виконання шляхом зведення графа до однієї вершини $\Phi_O = \{\Phi_{O_i}, i = 1, 2, \dots, 7\}$ (рис. 7.14б).

Тобто часовий граф будеться шляхом об'єднання всіх функціональних операторів алгоритму та їх зв'язків.

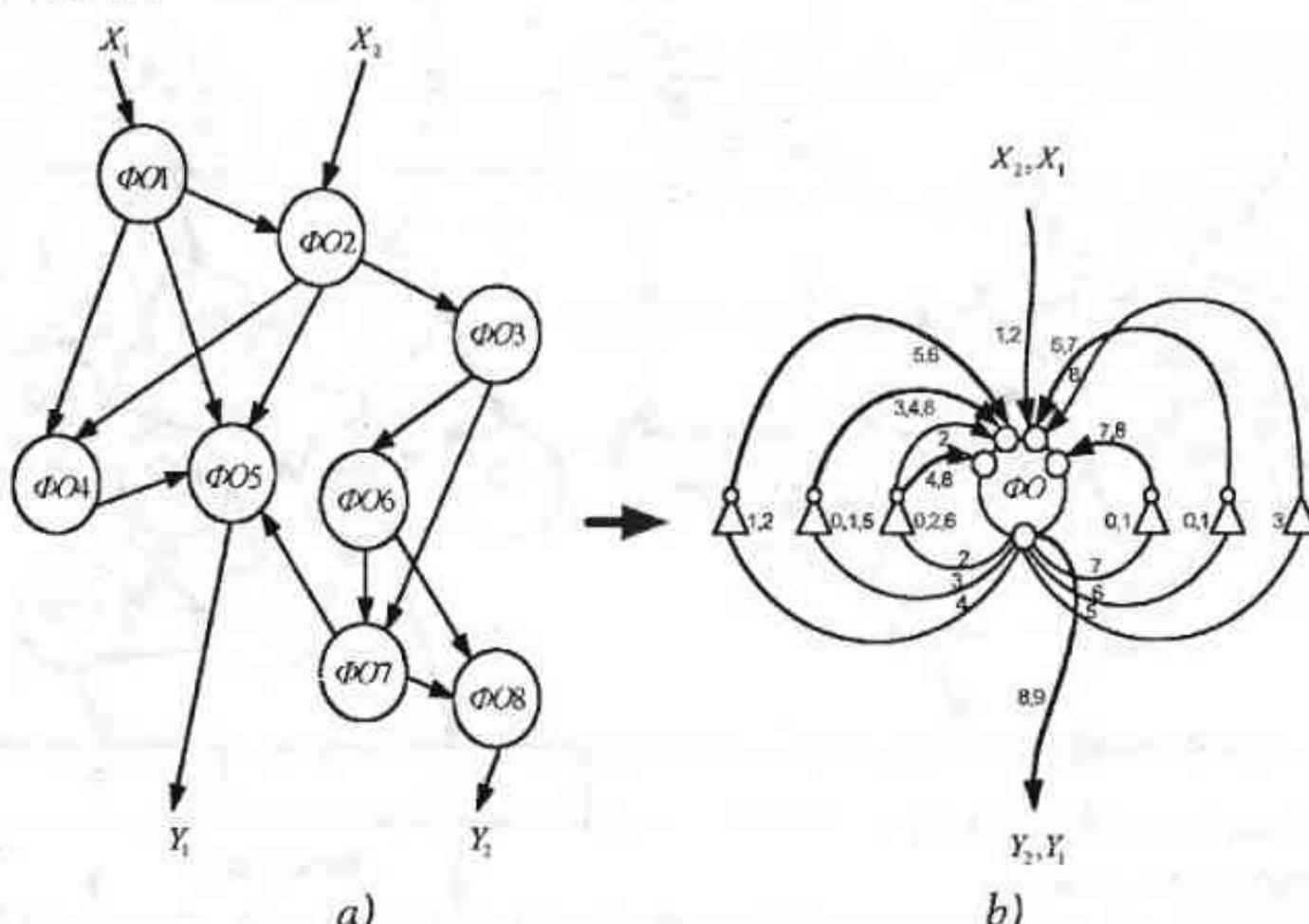


Рис. 7.14. Перехід від просторового графа алгоритму до часового

Просторовий граф алгоритму складається з функціональних операторів, яким відповідають обчислювальні операції, та дуг, якими здійснюється передача даних для обробки. Часовий граф алгоритму складається з однієї вершини, яка послідовно виконує операції просторового графу. Даний граф характеризується наявністю вхідних та вихідних вузлів, які є незалежними входами (виходами) вершини графу. Подавання вхідних даних та видача вихідних даних здійснюється у певні проміжки часу (такти). Вхідні дані X_1 та X_2 поступають на 1 та 2 такті, а видача вихідних результатів Y_1 та Y_2 відбувається на 8 та 9 тактах. На вхідних та вихідних дугах графу зображені числа, які вказують порядок спрацювання дуги в часі. Дуги, біля яких записано кілька цифр, спрацьовують більше одного разу. Трикутні вершини із вказаними значеннями затримки даних відповідають за здійснення одночасної подачі даних з вихідного вузла до вхідних вузлів вершини графу. Трикутні вершини, які мають на виході вузли, пропускають більше одного даного.

Тоді для побудови багатотактового пристрою потрібно синтезувати багатофункціональну комбінаційну схему, яка б дозволяла виконати всі функціональні оператори ФО, а пристрій керування повинен задати тип виконуваного в конкретний момент часу функціонального оператора та забезпечити пересилання на входи багатофункціональної комбінаційної схеми потрібних в цей момент даних.

Багатотактові операційні пристрой характеризуються малими затратами обладнання та, відповідно, невисокою продуктивністю.

Питання побудови багатотактового операційного пристрою в цілому формалізовані і детально розглянуті в літературі.

7.11. Однотактовий операційний пристрій

Побудова однотактових операційних пристрой передбачає повністю апаратне відображення просторового графа виконуваного алгоритму комбінаційними схемами, які виконують функціональні оператори алгоритму і з'єднані між собою відповідно до графа алгоритму, як це показано на рис. 7.15.

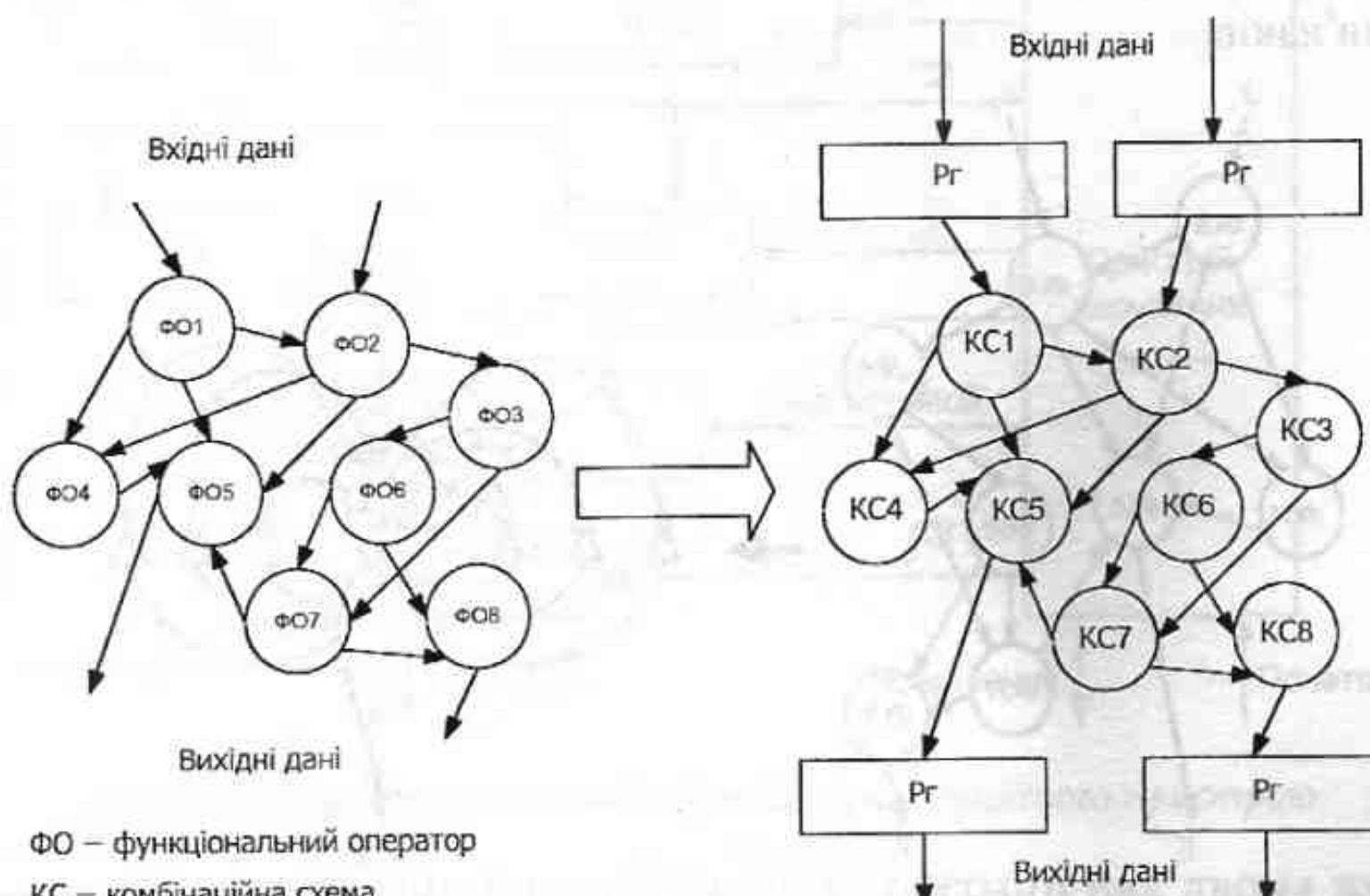


Рис. 7.15. Просторовий граф алгоритму та його апаратне відображення

Кожному функціональному оператору алгоритму ФО поставлена у відповідність комбінаційна схема КС, яка його виконує. На вході та виході операційного пристрою включені регистри.

Часова затримка в однотактовому операційному пристрой $T_{оп}$ визначається сумою затримок t_{ij} комбінаційних елементів, що лежать на найдовшому шляху проходження сигналу

$$T_{оп} = \sum_i MAX t_{ij}$$

Цей шлях (їх може бути декілька) називається критичним шляхом. Обчислення часу виконання алгоритму в однотактовому операційному пристрой зводиться, таким чином, до знаходження критичного шляху.

Затрати обладнання $W_{оп}$ на однотактовий операційний пристрой визначаються сумою затрат обладнання W_i на реалізацію комбінаційних схем $КС_i$, тобто

$$W_{оп} = \sum_i W_i$$

а також на вхідні та вихідні регистри.

7.12. Конвеєрний операційний пристрй

Конвеєрний принцип обробки передбачає суміщення в часі виконання операторів алгоритму над різними даними. Одним із можливих підходів тут є конвеєризація однотактових ОП, структура яких базується на апаратному відображені потокового графа виконуваного алгоритму. На рис. 7.16 показано структуру конвеєрного операційного пристроя, який реалізує потоковий граф алгоритму, наведений на рис. 7.6.

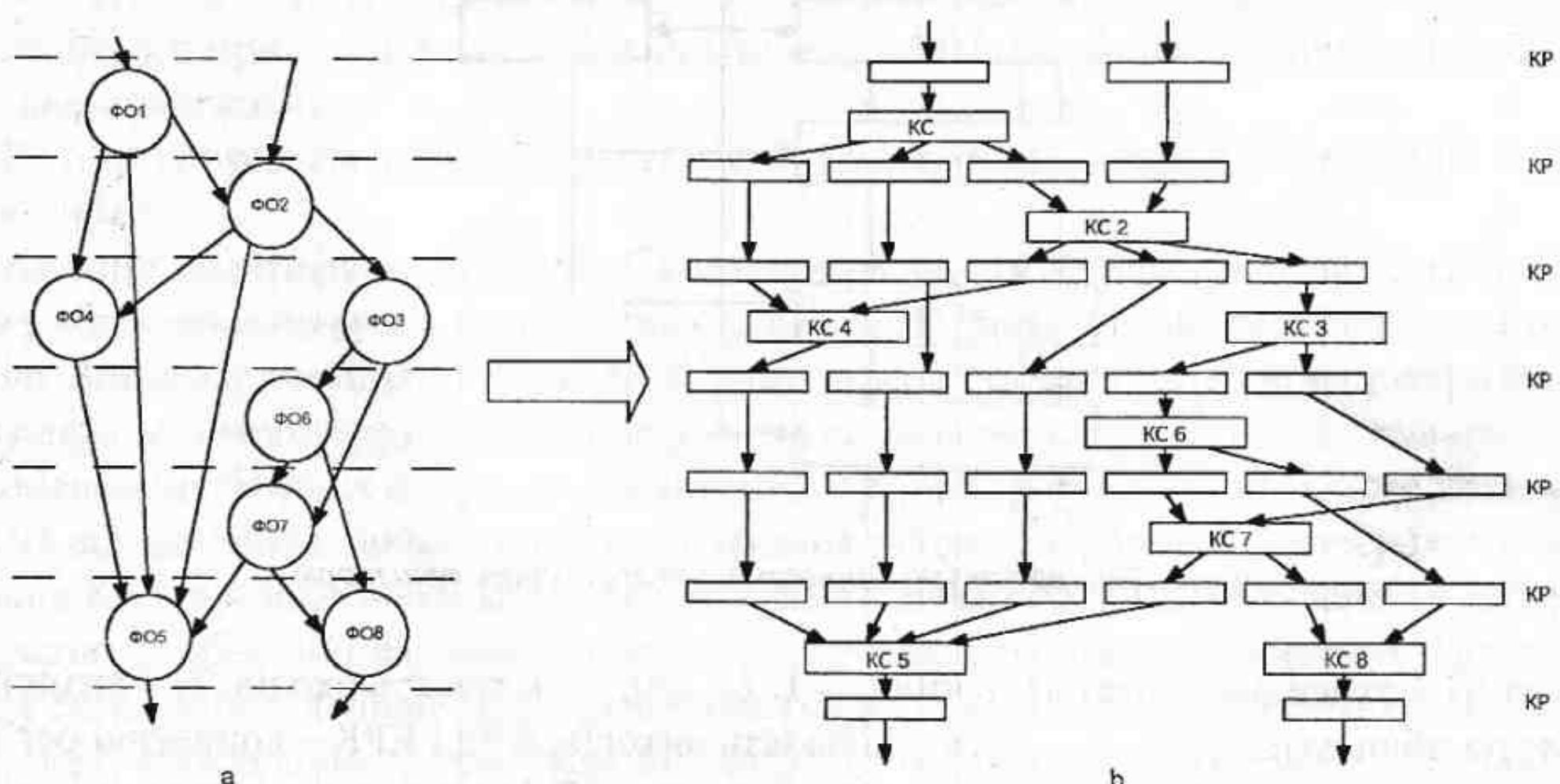


Рис. 7.16. Потоковий граф алгоритму а) та структура конвеєрного операційного пристроя, який його реалізує б)

Як ми вже вияснили, потоковим графом алгоритму називають таке представлення графа, яке передбачає розділення всіх його вершин по ярусах таким чином, що в i-му

ярусі розміщені тільки функціональні оператори, які залежать хоча б від одного функціонального оператора ($i-1$)-го яруса і не залежать від функціональних операторів, які не входять в яруси з меншими ніж i номерами. Всередині яруса функціональні оператори між собою не мають з'єднань. На рис. 7.16а штриховими лініями розділені яруси графа. При побудові конвеєрного операційного пристрою кожному функціональному оператору алгоритму поставлена у відповідність комбінаційна схема КС, яка його виконує, і, крім того, комбінаційні схеми, які виконують функціональні оператори ярусів потокового графа алгоритму, розділяються конвеєрними регістрами КР, як це показано на рис. 7.16б.

Таким чином, апаратне відображення алгоритму означає послідовне з'єднання комбінаційних схем, кожна з яких виконує операції одного яруса алгоритму. В конвеєрному операційному пристрой комбінаційні схеми з'єднані між собою через конвеєрні регістри.

Заданий алгоритм виконується над вхідними даними при їх однократному проходжені через конвеєрний операційний пристрій.

Найбільша довжина конвеєра такого конвеєрного операційного пристрою дорівнює кількості P ярусів алгоритму. Такий конвеєрний операційний пристрій має найвищу частоту, яка може бути досягнута при апаратній реалізації заданого конкретизованого ПГ алгоритму.

Узагальнена структура конвеєрного операційного пристрою наведена на рис. 7.17.

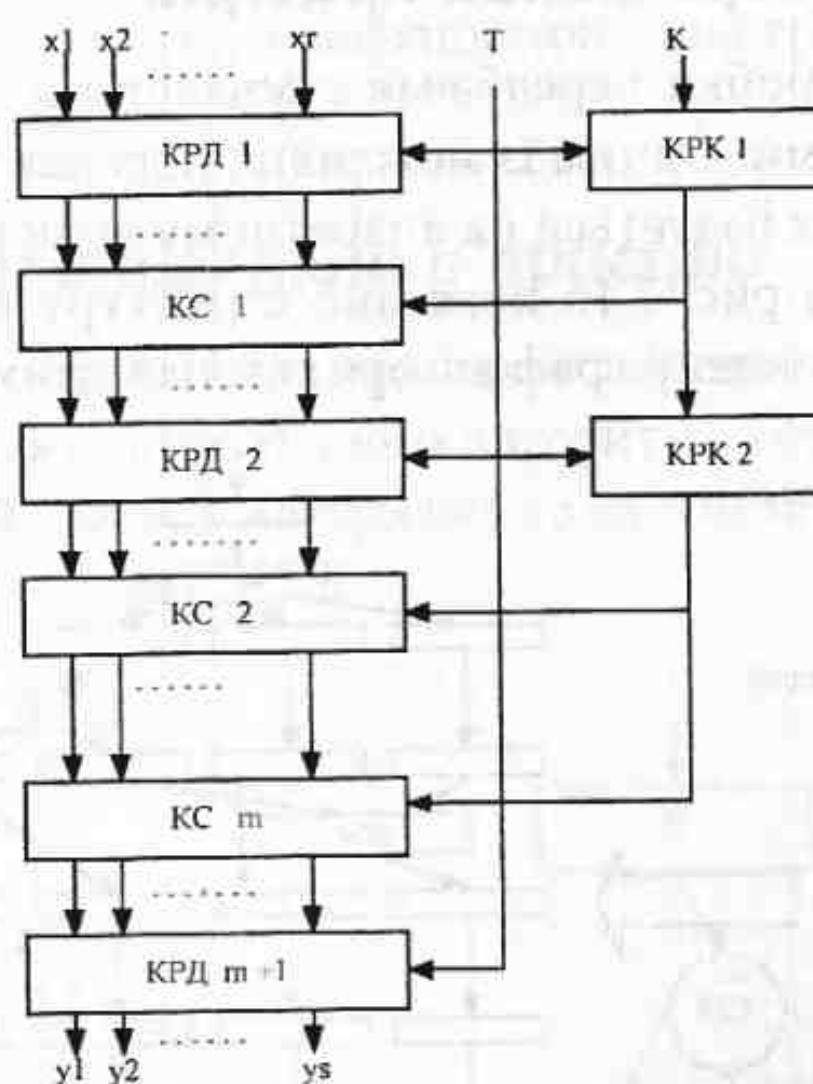


Рис. 7.17. Структура конвеєрного операційного пристрою

Тут X_i – входи поступлення даних ($i = 1, 2, \dots, r$), r – кількість входів, Y_j – виходи результатів обчислень ($j = 1, 2, \dots, s$), s – кількість виходів, КРД і КРК – конвеєрні регістри відповідно до даних D і команд K , m – кількість ярусів ПГ алгоритму, n – кількість ярусів конвеєра конвеєрного операційного пристрою. Якщо структура конвеєрного операційного пристрою орієнтована на реалізацію алгоритму, який обчислює одну функцію, він називається однофункціональним, групу функцій – багатофункціональним. В багатофункціональному конвеєрному операційному пристрої програмуються канали передачі

інформації та функцій, які виконуються комбінаційними схемами відповідно до графа багатофункціонального алгоритму

В однофункціональному конвеєрному операційному пристрої вихідні дані $Y = y_1, y_2, \dots, y_s$, однозначно визначаються входними даними $X = x_1, x_2, \dots, x_r$ і функцією Φ , зашитою в його структурі: $Y = \Phi(X)$. В загальному випадку $Y = \{Y_{kf}\}$, $X = \{X_{jf}\}$, тобто обробці підлягають матриці даних. При цьому $k = 1, 2, \dots, s$; $j = 1, 2, \dots, r$; $f = 1, 2, \dots, N$, де r, s – кількість входів і виходів конвеєрного операційного пристрою, N – кількість груп чисел, які поступають в конвеєрний операційний пристрій. Для кожного яруса однофункціонального конвеєрного операційного пристрою можна записати $A = \Phi(A_i)$, де A_i – вмістиме реєстрів i -го яруса, Φ – операція, що визначається функціональними операторами $(i+1)$ -го яруса ПГ алгоритму і виконується в $(i+1)$ -му ярусі конвеєрного операційного пристрою. Так як $A_0 = X$; $A_1 = \Phi_1(A_0)$; $A_2 = \Phi_2(A_1)$ і т. д., то $Y = \Phi_m(\Phi(\dots(\Phi_1(X))\dots))$, тобто функція Φ виконується над даними, які надходять в конвеєрний операційний пристрій, при їх проходженні через всі яруси конвеєра

Нехай в конвеєрному операційному пристрої обробляється масив чисел $X = \{X_1, X_2, \dots, X_r\}$, $r = 1, 2, \dots, N$. В першому такті тактовим імпульсом T в реєстр KRD_1 записується значення X_1 . Над ним в комбінаційній схемі KS_1 першого яруса виконується операція Φ_1 . Другим тактовим імпульсом результати цієї операції перепишуться в реєстр KRD_2 , а в реєстр KRD_1 запишеться значення X_2 . В комбінаційних схемах KS_1, KS_2 над значеннями, що зберігаються відповідно в реєстрах KRD_1 і KRD_2 виконуються операції Φ_1 і Φ_2 . В третьому такті результати із KS_2 запищуться в KRD_3 і т. д. На реєстри ярусів конвеєра, котрі звільняються, в кожному такті засилуються нові дані оброблюваного масиву, над якими виконуються ті ж операції. При повній загрузці конвеєра одночасно виконуються оператори всіх m ярусів алгоритму. Стан конвеєрного операційного пристрою в t -му такті його роботи при обробці N чисел визначається вектором $A(t) = |A_0(t) A_1(t) \dots A_m(t)|$, де $t = 1, 2, \dots, (m + N)$. При цьому $A_m(t) = Y_t$, тобто в кожному такті на виході конвеєрного операційного пристрою з'являється результат обчислення, крім перших m тактів, коли заповнюється конвеєр

Регістри ярусів мають розрядність, рівну розрядності вихідної інформації i -го яруса ПГ алгоритму

Кожний багатофункціональний конвеєрний операційний пристрій повинен мати деяку кількість елементів, призначених для забезпечення налаштування на задану операцію. Зокрема, комбінаційні схеми KS ярусів можуть вміщувати комутатори прямих зв'язків, з допомогою яких під дією керуючих сигналів виконується налаштування KS на виконання необхідних операцій. Відміна структури багатофункціонального конвеєрного ОП від однофункціонального заключається також в наявності конвеєрних реєстрів команд КРК, що зберігають код операції, а також зв'язків для налаштування на потрібну операцію комбінаційних схем ярусів конвеєрного операційного пристрою. Тут по конвеєру синхронно з даними під керуванням тих же тактових імпульсів T по реєстрах команд КРК просуваються і команди. Команда з виходу відповідного реєстра налаштовує комбінаційні схеми цього яруса на виконання необхідної операції. Для ярусів конвеєра в цьому випадку можна записати: $A_{i+1} = \Phi_i(A_i, K_i)$, де K_i – вмістиме реєстра КРК i -го яруса конвеєра. Виконувані в такому конвеєрному операційному пристрої перетворення над входними даними X можна записати виразом

$$Y = \Phi_m(K_m, \Phi_m(K_m, \Phi_m(\dots, \Phi(K, X)\dots))).$$

7.13. Алгоритмічні операційні пристрой

7.13.1. Пристрой додавання і віднімання двійкових чисел з фіксованою комою

Пристрой для додавання двійкових чисел називається суматором. Суматор використовується в комп'ютері як складова частина АЛП, а також як елемент інших вузлів комп'ютера. Коротко розглянемо основні питання побудови суматора. На рис. 7.18 представлена схема однотактового алгоритмічного операційного пристрою (АОП) додавання і віднімання чисел з фіксованою комою з послідовною обробкою двійкових чисел.

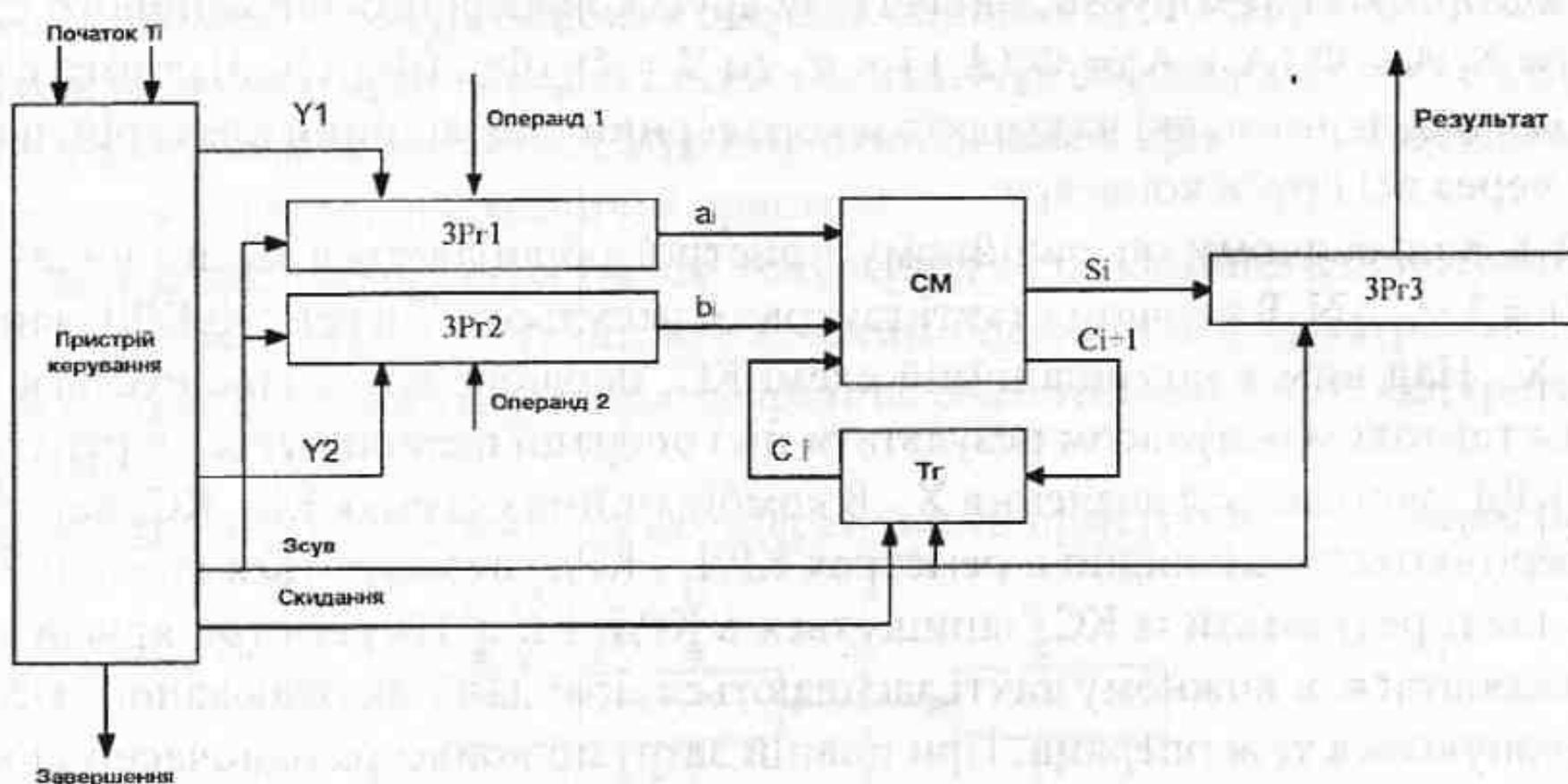


Рис. 7.18. Послідовний АОП додавання і віднімання чисел з фіксованою комою

Тут вхідні операнди 1 та 2 записуються сигналами Y_1 та Y_2 з пристрою керування до зсувних реєстрів $3Pr_1$ та $3Pr_2$, виходи молодших розрядів яких з'єднані з входами однорозрядного суматора CM , вихід суми S_i , якого з'єднаний з входом старшого розряду зсувного реєстра $3Pr_3$. Вихід переносу суматора з'єднаний з входом тригера Tg , який зберігає значення переносу C відповідного розряду протягом одного такту для подачі його на вхід однорозрядного суматора. Перед початком роботи вміст тригера Tg встановлюється в нуль. В кожному такті сигналом зсуву з пристрою керування вміст зсувних реєстрів зсувается на один розряд вправо, а в тригер записується значення переносу з чергового розряду суми. Однорозрядний суматор працює відповідно до таблиці істинності з розділу 4 (табл. 4.5). Розряд суми з виходу однорозрядного суматора записується до зсувного реєстра $3Pr_3$. Для додавання двох n -розрядних чисел в приведеному пристрої необхідно n тактів. Після цього на виході пристрою керування з'явиться сигнал завершення роботи, а в зсувному реєстрі $3Pr_3$ буде знаходитись результат операції.

Перевагою такого пристрою є простота та малі затрати обладнання на реалізацію суматора.

Значно частіше в комп'ютерах використовуються однотактові алгоритмічні ОП для додавання та віднімання двійкових чисел (рис. 7.19), в якому всі розряди операндів поступають на паралельний суматор з вхідних реєстрів Pr_1 та Pr_2 одночасно. Тим самим, за рахунок паралельної обробки досягається значно вища швидкодія порівняно з порозрядним додаванням операндів.

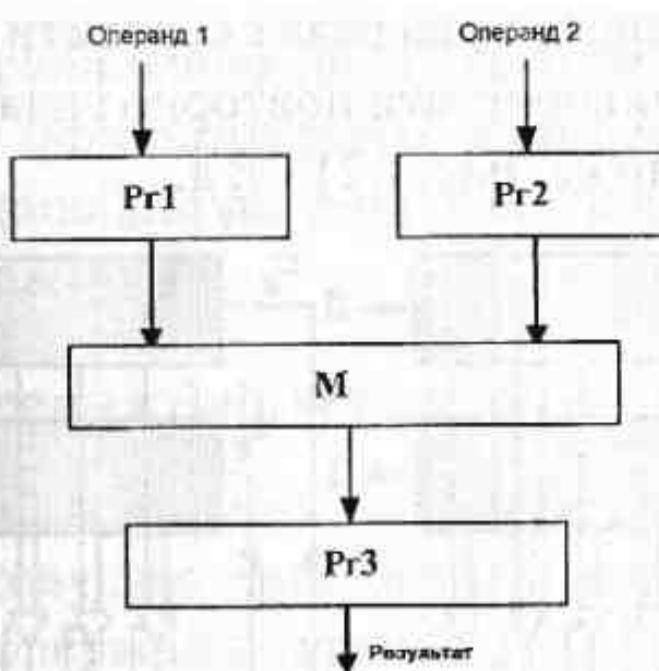


Рис. 7.19. Паралельний АОП для додавання та віднімання двійкових чисел

Однотактовий суматор будується шляхом апаратного відображення графа алгоритму додавання двійкових чисел, тобто шляхом заміни функціональних операторів алгоритму відповідними комбінаційними схемами.

Найпростішим з однотактових суматорів є однотактовий суматор, який апаратно відображає граф алгоритму додавання двійкових чисел з послідовним переносом і відповідно називається суматором з послідовним переносом. Схема цього суматора повторює граф відповідного алгоритму n-роздрядного додавання двійкових чисел з послідовним переносом (рис. 4.5). Недоліком суматора з послідовним переносом є значна затримка при формуванні переносу, оскільки для отримання останнього розряду суми перенос повинен бути сформованим всіма попередніми однорозрядними суматорами.

Для зменшення кількості операцій, які знаходяться на критичному шляху, створено ряд алгоритмів додавання, в яких скорочено кількість послідовних операцій при формуванні переносу та відповідних структур суматорів. Це, зокрема, суматори з наскрізним, частково груповим і груповим переносами, в яких здійснюється паралельне формування переносів для всіх розрядів, або для груп розрядів. Досить простою та ефективною є реалізація в суматорі алгоритму за методом вибору переносу.

Оскільки питанню побудови ефективних алгоритмів додавання двійкових чисел приділено достатньо уваги в літературі, коротко розглянемо останній з названих вище підходів, тобто реалізацію алгоритму за методом вибору переносу. Розірвавши в довільному місці тракт проходження переносу та сформувавши два тракти його подальшого проходження із значенням вхідного переносу 0 та 1 відповідно, з послідовним вибором результата за допомогою мультиплексора за значенням реального переносу в місці розриву, отримаємо показану на рис. 7.20 схему суматора.

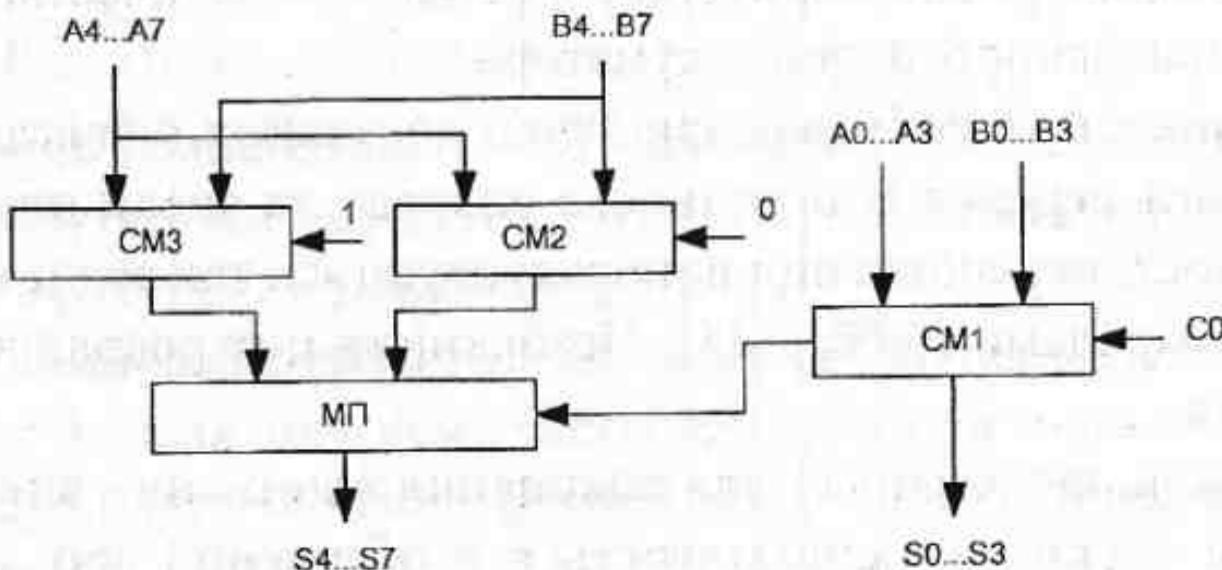


Рис. 7.20. Однотактовий суматор двійкових чисел за методом вибору переносу

Тим самим доволі просто вдалося в два рази скоротити критичний шлях формування переносу. Цей же підхід можна використати повторно і для прискорення роботи n/2-розрядних суматорів, як це показано на рис. 7.21 і т. д.

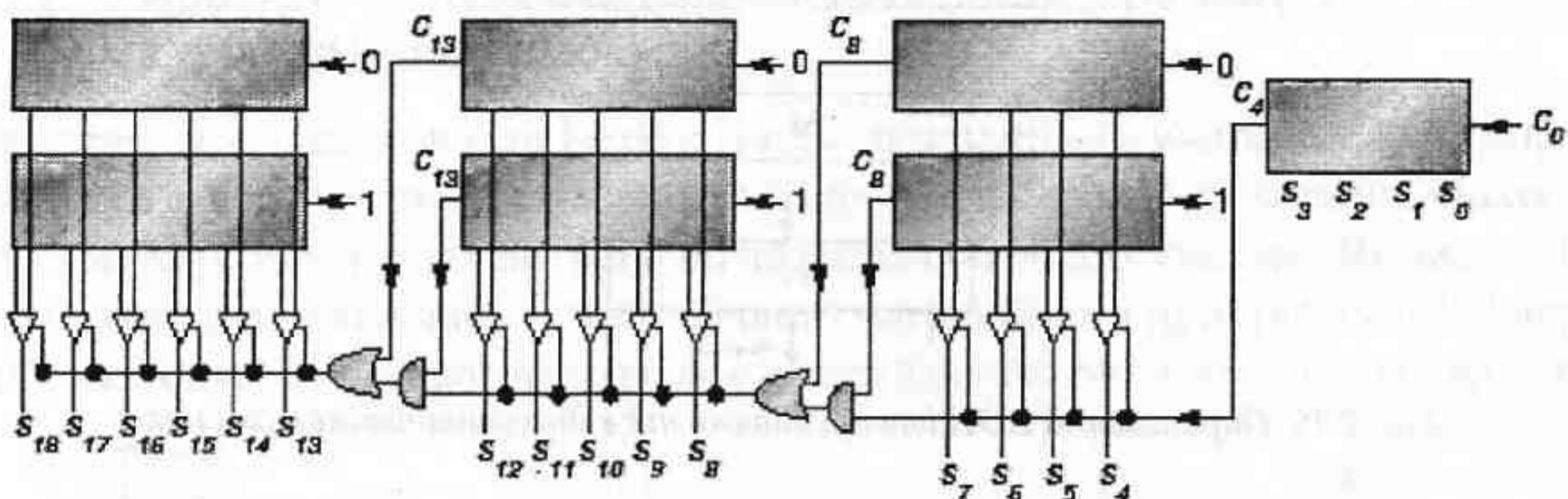


Рис. 7.21. Однотактовий суматор двійкових чисел за методом вибору переносу з чотирикратним прискоренням

Часто в комп'ютерах використовується пристрій для накопичення двійкових чисел, тобто для послідовного багатомісного додавання N чисел (рис. 7.22). В такому операційному пристрої є входний Рг1 та вихідний Рг2 регістри, а також суматор, причому один з входів суматора з'єднаний з виходом вихідного Рг2 регістра. Для забезпечення коректної роботи пристрою розрядність суматора повинна бути розширеною в сторону старших розрядів на $\lceil \log_2 N \rceil$ бітів.

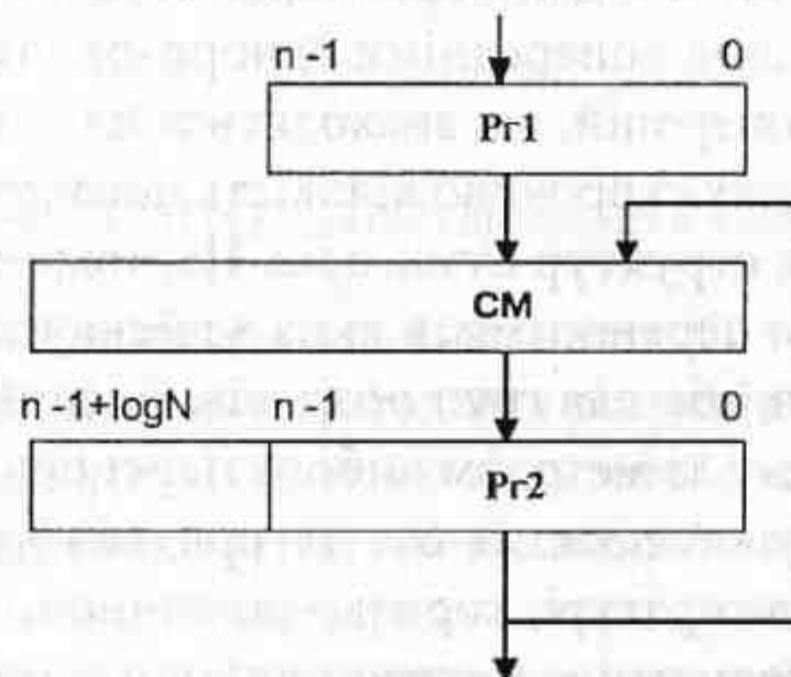


Рис. 7.22. АОП накопичення двійкових чисел

Для прискорення накопичення чисел використовують запам'ятовування не тільки сум з виходу суматора, а й переносів (так звані суматори з запам'ятовуванням переносів). В цьому випадку тakt роботи пристрою в режимі накопичення чисел буде визначатися затримкою лише однорозрядного суматора.

Важливим питанням при побудові пристрій додавання є фіксація переповнення. Для цього аналізується перенос із останнього розряду та знаки операндів. Для спрощення аналізу наявності переповнення використовуються так звані модифіковані коди з двома знаковими розрядами (0-00, 1-11). Незбіжність цих розрядів говорить про наявність переповнення.

Потрібно зауважити, що зазвичай для виконання віднімання використовується додавання, перед яким від'ємник перетворюється в обернений або доповняльний код. Пряме віднімання використовується рідко. Якщо ж така потреба є, то взамін суматора

використовується віднімач, правила побудови якого не відрізняються від правил побудови суматора, а для синтезу однорозрядного віднімача використовується залежність між входами та виходами відповідно до табл. 4.6. Якщо виконуються обидві операції – розробляється суматор-віднімач.

7.13.2. Пристрой множення двійкових чисел з фіксованою комою

Послідовне (багатотактове) множення базується на послідовному утворенні суми часткових добутків, використовуючи один або декілька суматорів, і реалізується шляхом послідовного виконання операцій додавання.

Як ми вже бачили в розділі 4, процес множення може починатися з молодших і старших розрядів множника. При цьому повну суму часткових добутків (тобто добуток) можна отримати двома шляхами:

- зсувом множеного на потрібну кількість розрядів і додаванням отриманого чергового часткового добутку до раніше накопиченої суми;
- зсувом суми раніше отриманих часткових добутків на кожному кроці на один розряд і наступним додаванням нерухомого множеного або 0 до зсунutoї суми.

Таким чином, існує 4 методи множення двійкових чисел, на основі яких можна побудувати 4 алгоритми ітераційного виконання цієї операції та 4 базових структури багатотактових АОП множення двійкових чисел:

- множення починаючи з молодших розрядів множника зі зсувом суми часткових добутків вправо;
- множення починаючи з молодших розрядів множника зі зсувом множеного вліво;
- множення починаючи з старших розрядів множника зі зсувом суми часткових добутків вліво;
- множення починаючи з старших розрядів множника зі зсувом множеного вправо.

Розглянемо їх детальніше.

7.13.2.1. Багатотактовий пристрой множення двійкових чисел з молодших розрядів множника при нерухомому множеному з зсувом суми часткових добутків

Алгоритм множення двійкових чисел, який реалізує метод множення починаючи з молодших розрядів множника з зсувом суми часткових добутків вправо, описується наступним ітераційним виразом:

$$Z_{i+1} = \frac{(Z_i + X \cdot Y_{(i)})}{2},$$

$$\text{де } Z_0 = 0; \quad i = \overline{0, n-1}; \quad Z_n = Z = X \cdot Y.$$

Тут вжито наступні позначення: X, Y, Z – множене, множник і добуток відповідно, Z_i – сума часткових добутків на i -му етапі, $Y(i)$ – i -й розряд множника, n – кількість розрядів операндів без врахування знакового розряду.

В кожному циклі множене додається до суми часткових добутків, якщо $Y_i = 0$, і не додається, якщо $Y_i = 1$, після чого сума часткових добутків множиться на 2^{-1} , тобто зсувается на один розряд вправо. Після закінчення n -го циклу утворюється шуканий добуток, тобто $Z_n = Z = XY$.

Алгоритм можна представити блок-схемою, показаною на рис. 7.23.

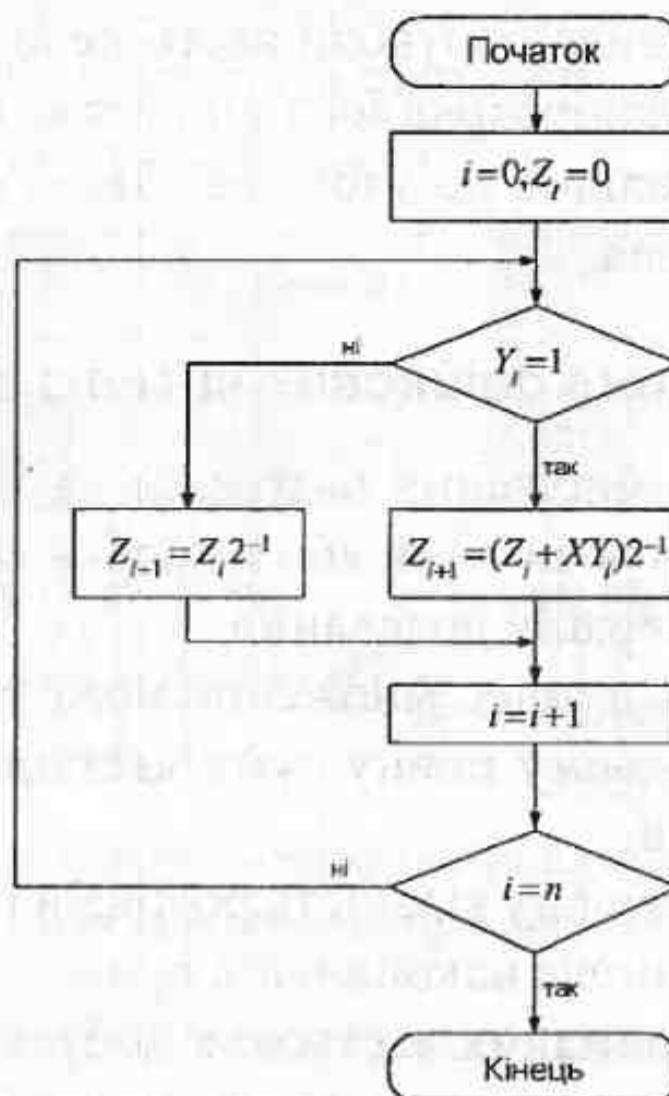


Рис. 7.23. Блок-схема алгоритму множення першим методом

Приклад:

Необхідно помножити два числа (без знакового розряду):

$$X = 0101\ 0101; \quad Y = 0110\ 1011;$$

Хід операцій проілюстровано в табл. 7.3:

Таблиця 7.3

i	Z_i	$Y(Y(i))$	$X \cdot Y(i)$	$Z_i + X \cdot Y(i)$	Z_{i+1}
0	0000 0000 0000 0000	0110 1011	0101 0101	0101 0101 0000 0000	0010 1010 1000 0000
1	0010 1010 1000 0000	0110 1011	0101 0101	0111 1111 1000 0000	0011 1111 1100 0000
2	0011 1111 1100 0000	0110 1011	0000 0000	0011 1111 1100 0000	0001 1111 1110 0000
3	0001 1111 1110 0000	0110 1011	0101 0101	0111 0100 1110 0000	0011 1010 0111 0000
4	0011 1010 0111 0000	0110 1011	0000 0000	0011 1010 0111 0000	0001 1101 0011 1000
5	0001 1101 0011 1000	0110 1011	0101 0101	0111 0010 0011 1000	0011 1001 0001 1100
6	0011 1001 0001 1100	0110 1011	0101 0101	1000 1110 0001 1100	0100 0111 0000 1110
7	0100 0111 0000 1110	0110 1011	0000 0000	0100 0111 0000 1110	0010 0011 1000 0111

Таким чином $0101\ 0101 \cdot 0110\ 1011 = 0010\ 0011\ 1000\ 0111$.

Базова структура багатотактового АОП множення двійкових чисел за описаним методом наведена на рис. 7.24.



Рис. 7.24. Базова структура багатотактового АОП множення першим методом

Тут СМЧД – суматор часткових добутків. Множник зберігається в регістрі множника, а множене – в регістрі множеного. Обидва ці регістри є n-роздрядними. Суматор част-

кових добутків є накопичувальним суматором, тобто на його виході є регистр з оберненим зв'язком, як це показано на рис. 7.22. В кожному такті вміст цього регистра та регистра множника зсуваються на один розряд вправо в сторону молодших розрядів. Розряд в крайньому правому тригері регистра множника випадає, а на його місце поміщається наступний розряд множника, який керує операцією СМЧД, тобто вказує, чи є в даному такті додавання, чи його немає. При зсуві молодший розряд СМЧД може випадати, а може і зберігатися, залежно від того, якої розрядності потрібен результат – n -розрядний, чи $2n$ -розрядний. Завдяки тому, що в даній структурі регистри множеного і множника є n -розрядними, а не $2n$ -розрядними, як це є в базових структурах АОП множення іншими методами, перший метод множення використовується найчастіше.

7.13.2.2. Багатотактовий пристрій множення двійкових чисел з молодших розрядів при нерухомій сумі часткових добутків з зсувом множеного вліво

Алгоритм множення двійкових чисел, який реалізує цей метод, описується наступними ітераційними виразами:

$$\begin{cases} Z_{i+1} = Z_i + X_i \cdot Y_{(i)}, \\ X_{i+1} = 2 \cdot X_i, \end{cases}$$

де $X_0 = X$; $Z_0 = 0$; $i = \overline{0, n-1}$; $Z_n = Z = X \cdot Y$.

Тут вжито наступні позначення: X , Y , Z – множене, множник і добуток відповідно, Z_i – сума часткових добутків на i -му етапі, $Y(i)$ – i -й розряд множника, n – кількість розрядів операндів без врахування знакового розряду.

Алгоритм можна представити блок-схемою, показаною на рис. 7.25.

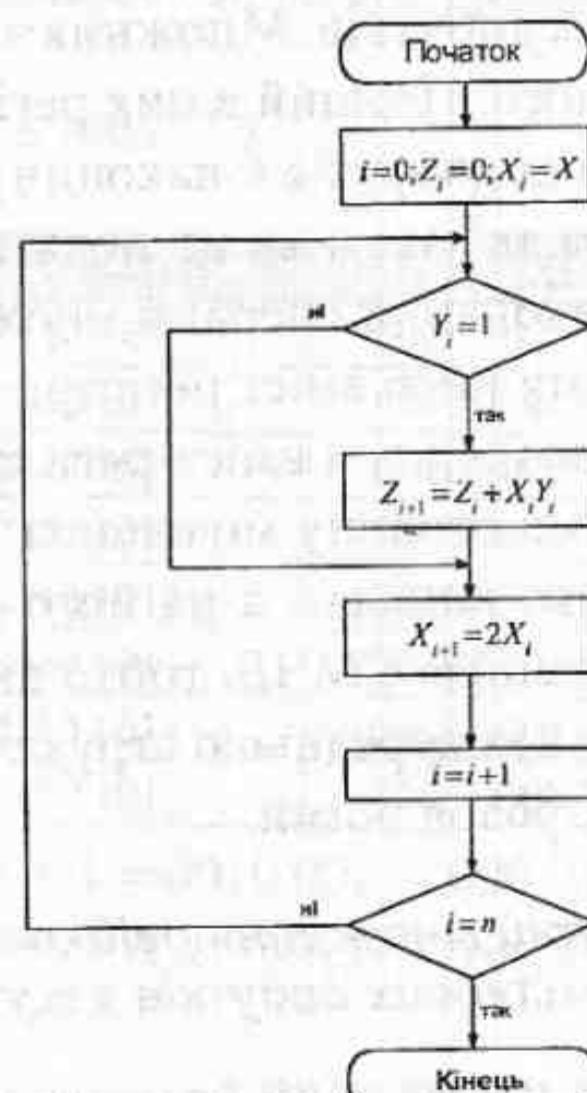


Рис. 7.25. Блок-схема алгоритму множення другим методом

Приклад:

Необхідно помножити два числа (без знакового розряду):

$X=0101\ 0101$; $Y=01101011$.

Хід операцій проілюстровано в табл. 7.4.

Таблиця 7.4

i	Z _i	Y(Y(i))	X	Z _{i+1} = Z _i + X • Y(i)
0	0000 0000 0000 0000	0110 1011	0000 0000 0101 0101	0000 0000 0101 0101
1	0000 0000 0101 0101	0110 1011	0000 0000 1010 1010	0000 0000 1111 1111
2	0000 0000 1111 1111	0110 1011	0000 0001 0101 0100	0000 0000 1111 1111
3	0000 0000 1111 1111	0110 1011	0000 0010 1010 1000	0000 0011 1010 0111
4	0000 0011 1010 0111	0110 1011	0000 0101 0101 0000	0000 0011 1010 0111
5	0000 0011 1010 0111	0110 1011	0000 1010 1010 0000	0000 1110 0100 0111
6	0000 1110 0100 0111	0110 1011	0001 0101 0100 0000	0010 0011 1000 0111
7	0010 0011 1000 0111	0110 1011	0010 1010 1000 0000	0010 0011 1000 0111

Таким чином $0101\ 0101 \cdot 01101011 = 0010\ 0011\ 1000\ 0111$.

Базова структура багатотактового АОП множення двійкових чисел за описаним методом наведена на рис. 7.26.

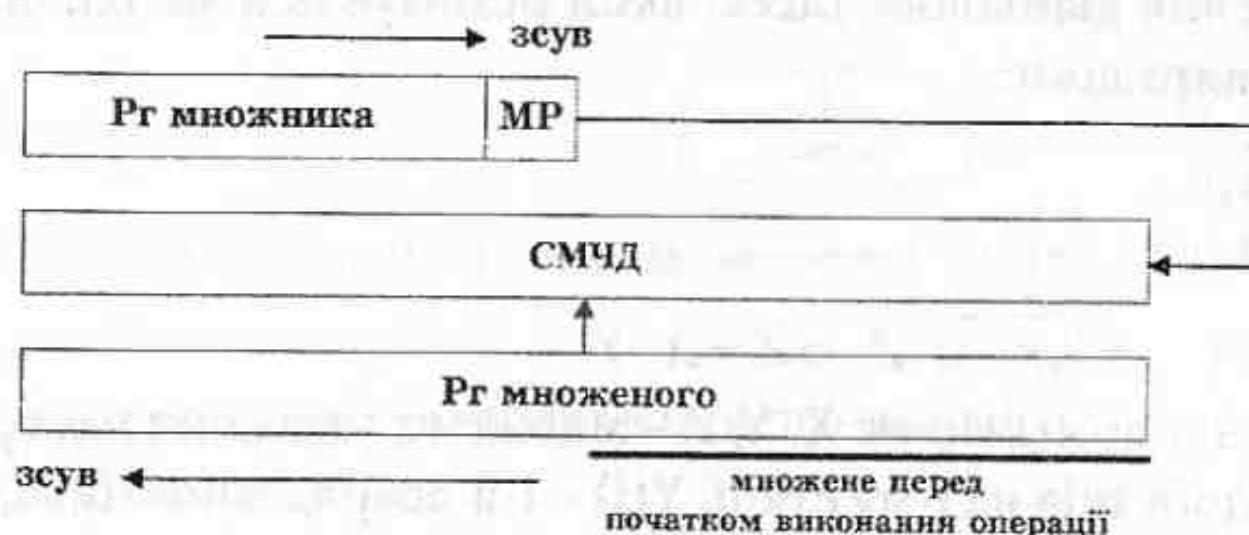


Рис. 7.26. Базова структура багатотактового АОП множення другим методом

Тут СМЧД – суматор часткових добутків. Множник зберігається в регістрі множника, а множене – в регістрі множеного. Перший з цих регістрів є n-роздрядним, а другий – 2n-роздрядним. Суматор часткових добутків є накопичувальним суматором, тобто на його виході є регістр з оберненим зв'язком як це показано на рис. 7.22, який також є 2n-роздрядним. Перед початком виконання операції множене знаходиться в правій частині регістра множеного. В кожному такті вміст регістра множеного зсувається на один розряд вліво в сторону старших розрядів, а вміст регістра множника в кожному такті зсувається на один розряд вправо в сторону молодших розрядів. Розряд в крайньому правому тригері регістра множника випадає, а на його місці поміщається наступний розряд множника, який керує операцією СМЧД, тобто вказує чи є в даному такті додавання, чи його немає. В порівнянні з попередньою структурою тут регістр множеного та СМЧД обов'язково мають бути 2n-роздрядними.

7.13.2.3. Багатотактовий пристрій множення двійкових чисел з старших розрядів при нерухомій сумі часткових добутків з зсувом множеного вправо

Алгоритм множення двійкових чисел, який реалізує цей метод, описується наступними ітераційними виразами:

$$\begin{cases} Z_{i+1} = Z_i + X_{i+1} \cdot Y_{(n-i-1)}, \\ X_{i+1} = \frac{X_i}{2}, \end{cases}$$

де $X_0 = X$; $Z_0 = 0$; $i = \overline{0, n-1}$; $Z_n = Z = X \cdot Y$.

Тут вжито наступні позначення: X, Y, Z – множене, множник і добуток відповідно, Z_i – сума часткових добутків на i -му етапі, $Y_{(n-i-1)}$ – $(n-i-1)$ -й розряд множника, n – кількість розрядів операндів без врахування знакового розряду.

Алгоритм можна представити блок-схемою, показаною на рис. 7.27.

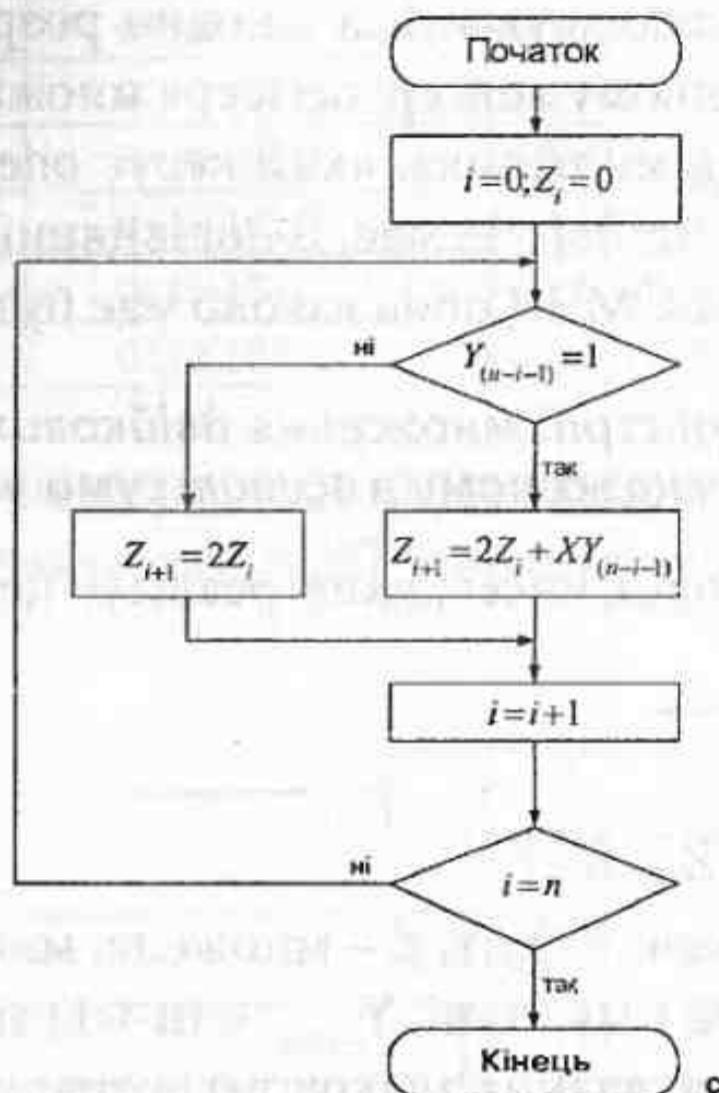


Рис. 7.27. Блок-схема алгоритму множення третім методом

Приклад:

Необхідно помножити два числа (без знакового розряду):

$X=0101\ 0101$; $Y=0110\ 1011$.

Хід операцій проілюстровано в табл. 7.5.

Таблиця 7.5

i	Z _i	Y (Y _(n-i-1))	X _{i+1}	Z _{i+1} = Z _i + X _{i+1} • Y _(n-i-1)
0	0000 0000 0000 0000	0110 1011	0010 1010 1000 0000	0000 0000 0000 0000
1	0000 0000 0000 0000	0110 1011	0001 0101 0100 0000	0001 0101 0100 0000
2	0001 0101 0100 0000	0110 1011	0000 1010 1010 0000	0001 1111 1110 0000
3	0001 1111 1110 0000	0110 1011	0000 0101 0101 0000	0001 1111 1110 0000
4	0001 1111 1110 0000	0110 1011	0000 0010 1010 1000	0010 0010 1000 1000
5	0010 0010 1000 1000	0110 1011	0000 0001 0101 0100	0010 0010 1000 1000
6	0010 0010 1000 1000	0110 1011	0000 0000 1010 1010	0010 0011 0011 0010
7	0010 0011 0011 0010	0110 1011	0000 0000 0101 0101	0010 0011 1000 0111

Таким чином $0101\ 0101 \cdot 0110\ 1011 = 0010\ 0011\ 1000\ 0111$.

Базова структура багатотактового АОП множення двійкових чисел за описаним методом наведена на рис. 7.28.

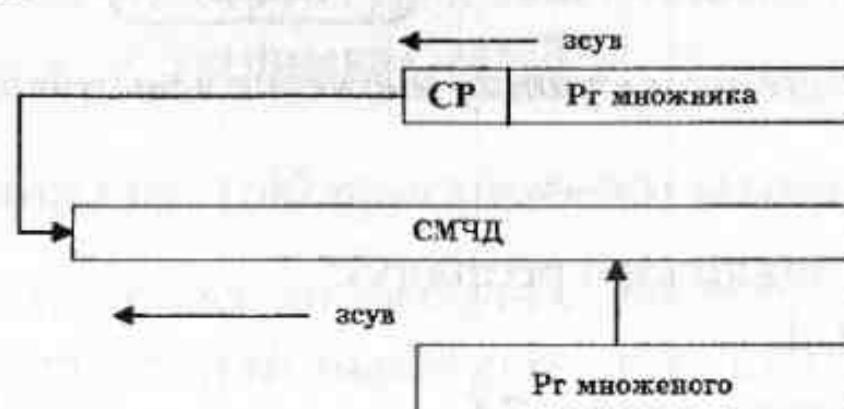


Рис. 7.28. Базова структура багатотактового АОП множення третім методом

Тут СМЧД – суматор часткових добутків. Множник зберігається в реєстрі множника, а множене – в реєстрі множеного. Обидва ці реєстри є n-роздрядними. Суматор часткових добутків є накопичувальним суматором, тобто на його виході є реєстр з оберненим зв'язком як це показано на рис. 7.22, який є 2n-роздрядним. В кожному такті вміст реєстрів множника та множеного зсувается на один розряд вліво в сторону старших розрядів. Розряд в крайньому лівому тригері реєстра множника випадає, а на його місце поміщається наступний розряд множника, який керує операцією СМЧД, тобто вказує чи є в даному такті додавання, чи його немає. В порівнянні з базовою структурою АОП множення першим методом тут СМЧД обов'язково має бути 2n-роздрядним.

7.13.2.4. Багатотактовий пристрій множення двійкових чисел з старших розрядів при нерухомому множеному з зсувом суми часткових добутків вліво

Алгоритм множення двійкових чисел, який реалізує цей метод, описується наступним ітераційним виразом:

$$Z_{i+1} = 2 \cdot Z_i + X \cdot Y_{(n-i-1)},$$

$$\text{де } Z_0 = 0; \quad i = \overline{0, n-1}; \quad Z_n = Z = X \cdot Y.$$

Тут вжито наступні позначення: X, Y, Z – множене, множник і добуток відповідно, Z_i – сума часткових добутків на i-му етапі, $Y_{(n-i-1)}$ – (n-i-1)-й розряд множника, n – кількість розрядів операндів без врахування знакового розряду.

Алгоритм можна представити блок-схемою, показаною на рис. 7.29.

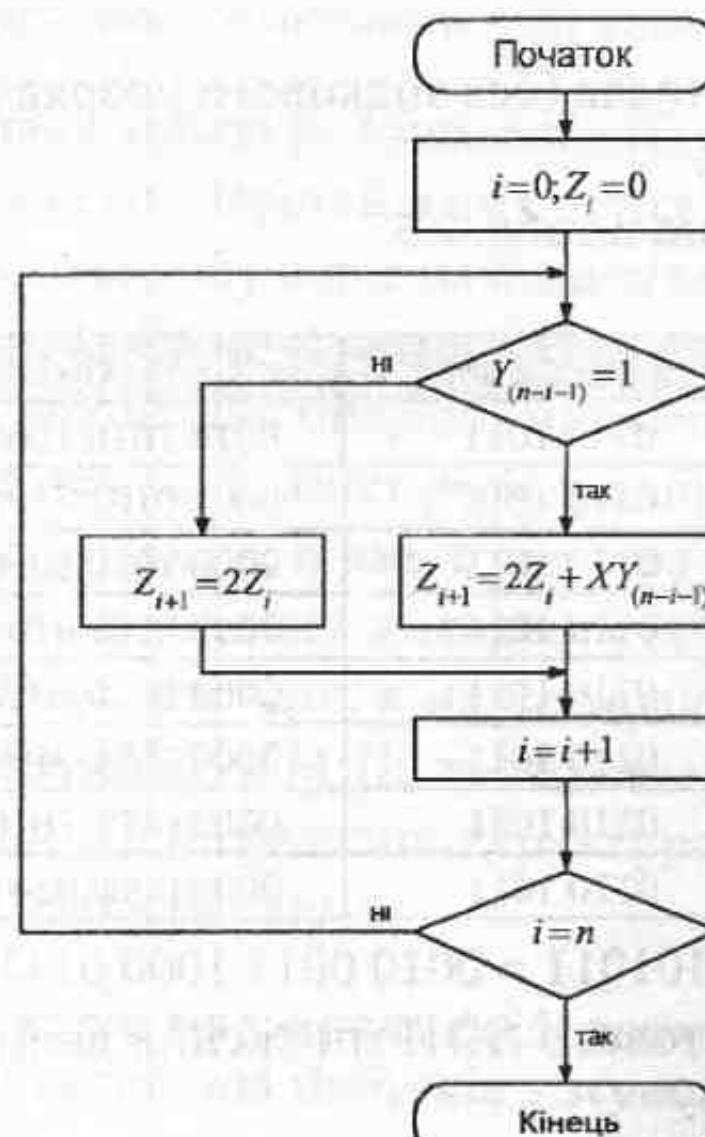


Рис. 7.29. Блок-схема алгоритму множення четвертим методом

Приклад:

Множимо два числа (без знакового розряду):

$$X=0101\ 0101; \quad Y=01101011.$$

Хід операцій проілюстровано в табл. 7.6.

Таблиця 7.6

i	Z_i	$Y(Y(n-i-1))$	$2 \cdot Z_i$	$Z_{i+1}=2 \cdot Z_i + X_{i+1} \cdot Y(n-i-1)$
0	0000 0000 0000 0000	0110 1011	0000 0000 0000 0000	0000 0000 0000 0000
1	0000 0000 0000 0000	0110 1011	0000 0000 0000 0000	0000 0000 0101 0101
2	0000 0000 0101 0101	0110 1011	0000 0000 1010 1010	0000 0000 1111 1111
3	0000 0000 1111 1111	0110 1011	0000 0001 1111 1110	0000 0001 1111 1110
4	0000 0001 1111 1110	0110 1011	0000 0011 1111 1100	0000 0100 0101 0001
5	0000 0100 0101 0001	0110 1011	0000 1000 1010 0010	0000 1000 1010 0010
6	0000 1000 1010 0010	0110 1011	0001 0001 0100 0100	0001 0001 1001 1001
7	0001 0001 1001 1001	0110 1011	0010 0011 0011 0010	0010 0011 1000 0111

Таким чином $0101\ 0101 \cdot 01101011 = 0010\ 0011\ 1000\ 0111$.

Базова структура багатотактового АОП множення двійкових чисел за описаним методом наведена на рис. 7.30.

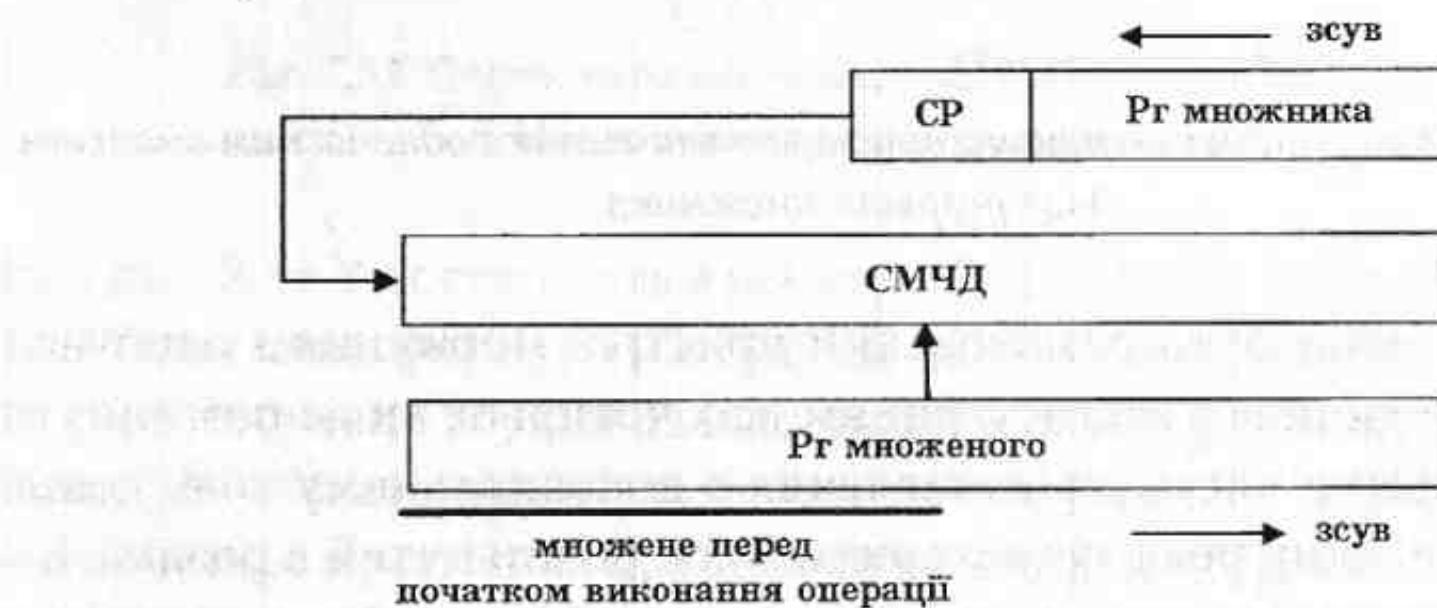


Рис. 7.30. Базова структура багатотактового АОП множення четвертим методом

Тут СМЧД – суматор часткових добутків. Множник зберігається в регістрі множника, а множене – в регістрі множеного. Перший з цих регістрів є n-роздрядним, а другий – 2n-роздрядним. Суматор часткових добутків є накопичувальним суматором, тобто на його виході є регістр з оберненим зв’язком як це показано на рис. 7.22, який також є 2n-роздрядним. Перед початком виконання операції множене знаходиться в лівій частині регістра множеного. В кожному такті вміст регістра множеного та вміст СМЧД зсуваніся на один розряд вправо в сторону молодших розрядів. Розряд в крайньому лівому тригері регістра множника випадає, а на його місце поміщається наступний розряд множника, який керує операцією СМЧД, тобто вказує чи є в даному такті додавання, чи його немає. В порівнянні з базовою структурою АОП множення першим методом тут, як в базовій структурі АОП множення другим методом, регістр множеного та СМЧД обов’язково мають бути 2n-роздрядними.

В усіх чотирьох розглянутих структурах АОП множення двійкових чисел час виконання операції $t_{mn} = nt_{cm}$, де t_{cm} – затримка СМЧД.

7.13.2.5. Багатотактовий пристрій прискореного множення

Одним із методів прискореного множення є одночасний аналіз декількох розрядів множника. Це може бути одночасний аналіз двох, трьох і більшої кількості розрядів. Для пояснення суті методу на рис. 7.31 показана схема багатотактового пристрою множення

з одночасним аналізом двох розрядів множника. Тут схема аналізу СА проводить аналіз двох розрядів множника і вказує СМЧД тип виконуваної операції: додавання відсутнє, додається значення множеного, додається подвоєне значення множеного, додається потроєне значення множеного. Зсув в реєстрах відбувається одночасно на два розряди.

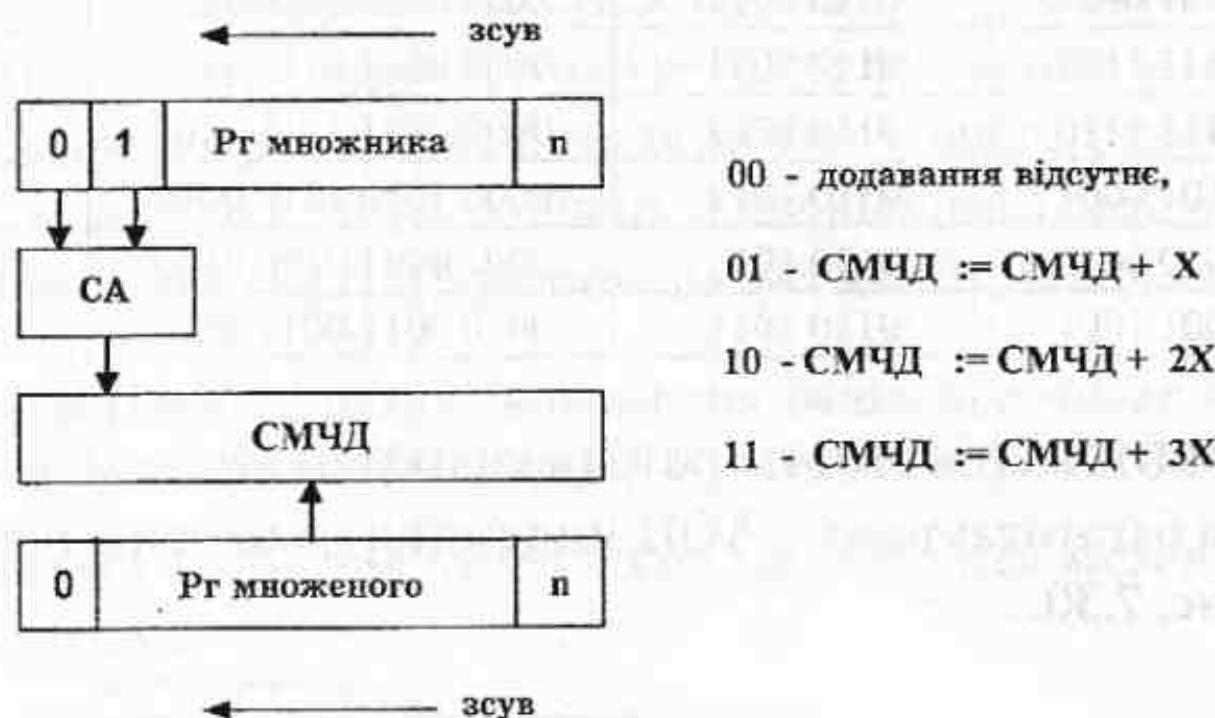


Рис. 7.31. Схема багатотактового пристрою множення з одночасним аналізом двох розрядів множника

Подібним чином може бути реалізований пристрій множення з одночасним аналізом двох розрядів множника з застосуванням всіх чотирьох вище розглянутих методів множення. При множенні чисел, представлених в доповняльному коді, доцільно використовувати пристрій, який реалізує алгоритм Бута, розглянутий в розділі 4.

Часто застосовуються асинхронні пристрої множення з одночасним аналізом всіх розрядів множника. В таких пристроях кількість тактів визначається кількістю одиниць в множнику.

Приклад:

$$X = 101001110001$$

$$Y = 110000111110.$$

На позиціях нулів у множнику Y додавання не виконується, лише зсув на відповідну кількість розрядів.

Множник Y можна представити в вигляді $Y = 1100010000(-1)0$. Тоді взамін 7 додавань необхідно виконати 3 додавання і одне віднімання. Особливо ефективний цей метод при виконанні множення на константи.

7.13.2.6. Однотактові пристрої множення двійкових чисел з фіксованою комою

Як вже було показано, побудова однотактових операційних пристрій передбачає апаратне відображення просторового графа алгоритму виконання операції комбінаційними схемами, які виконують функціональні оператори алгоритму і з'єднані між собою відповідно до графа алгоритму. Тому структура однотактового пристроя множення двійкових чисел з фіксованою комою повторить відповідну структуру графа алгоритму, наведеного на рис. 4.7, як це показано на рис. 7.32.

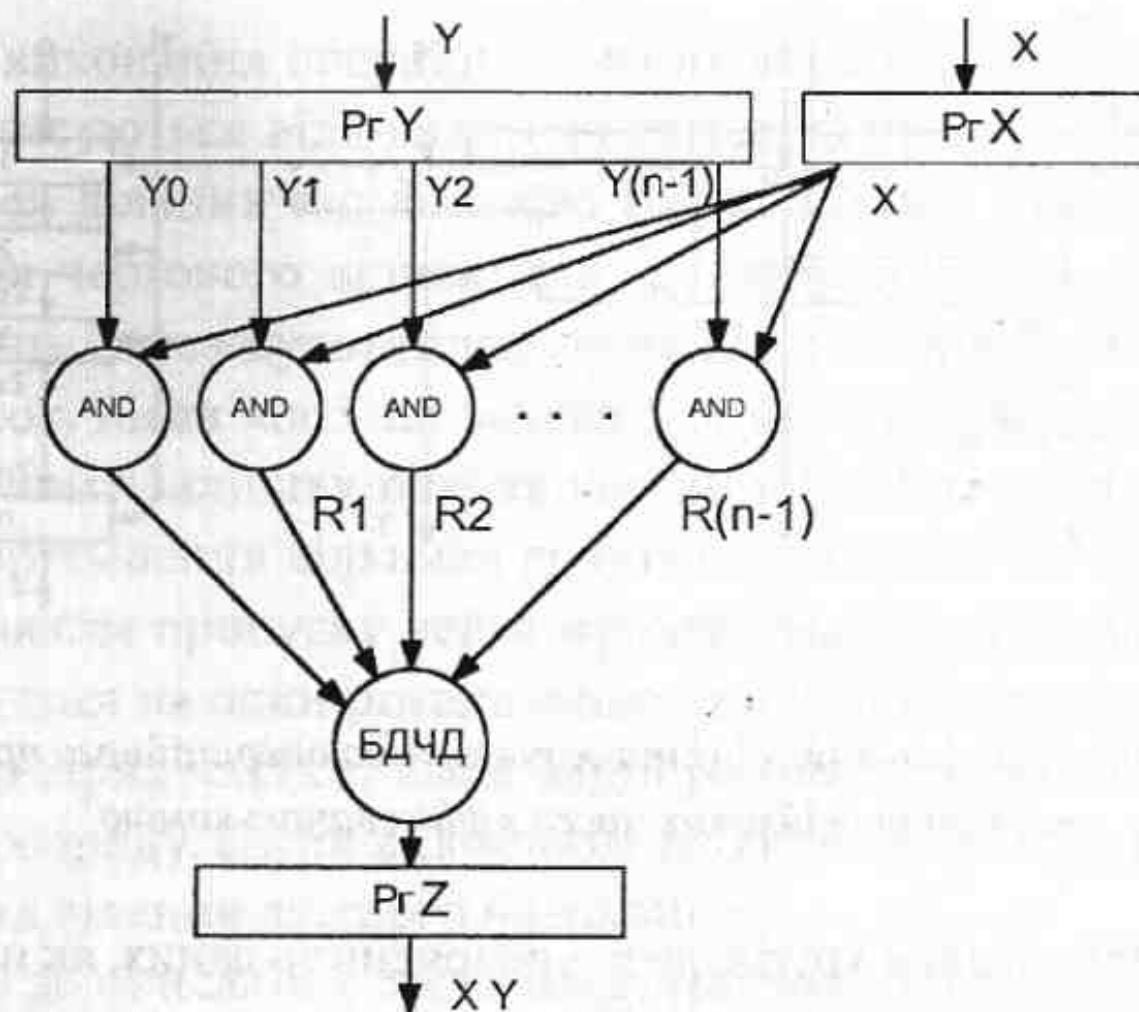


Рис. 7.32. Структура однотактового пристрою множення двійкових чисел з фіксованою комою

Тут вхідні дані X та Y поступають в реєстри PrX та PrY , а з них на пристрой логічного множення AND, на яких формуються логічні добутки множеного X на розряди множника Y . Ці логічні добутки з зсувом на відповідну кількість розрядів поступають на входи комбінаційної схеми багатомісного додавання часткових добутків БДЧД, результат множення з якої поступає в реєстр PrZ , а з нього на вихід пристрою.

Комбінаційна схема багатомісного додавання часткових добутків БДЧД реалізує алгоритми, детально розглянуті в п. 4.4.4.2, де кожному оператору двомісного однорозрядного двійкового додавання має бути поставлений у відповідність однорозрядний суматор двійкових чисел, який реалізує логічні вирази відповідно до табл. 4.5.

7.13.2.7. Конвеєрні пристрої множення двійкових чисел з фіксованою комою

При побудові конвеєрного операційного пристрою множення двійкових чисел з фіксованою комою кожному функціональному оператору алгоритму ставиться у відповідність комбінаційна схема, яка його виконує, і, крім того, комбінаційні схеми, які реалізують функціональні оператори ярусів потокового графа алгоритму, розділяються конвеєрними реєстрами. Алгоритм множення виконується над вхідними даними при їх однократному проходженні через конвеєрний операційний пристрій.

Якщо вибрати для реалізації граф алгоритму послідовного попарного додавання часткових добутків, отриманих починаючи з аналізу молодших розрядів множника, який представлений на рис. 4.8, то структура i -го яруса конвеєрного операційного пристрою множення двійкових чисел з фіксованою комою буде мати вигляд, показаний рис. 7.33.

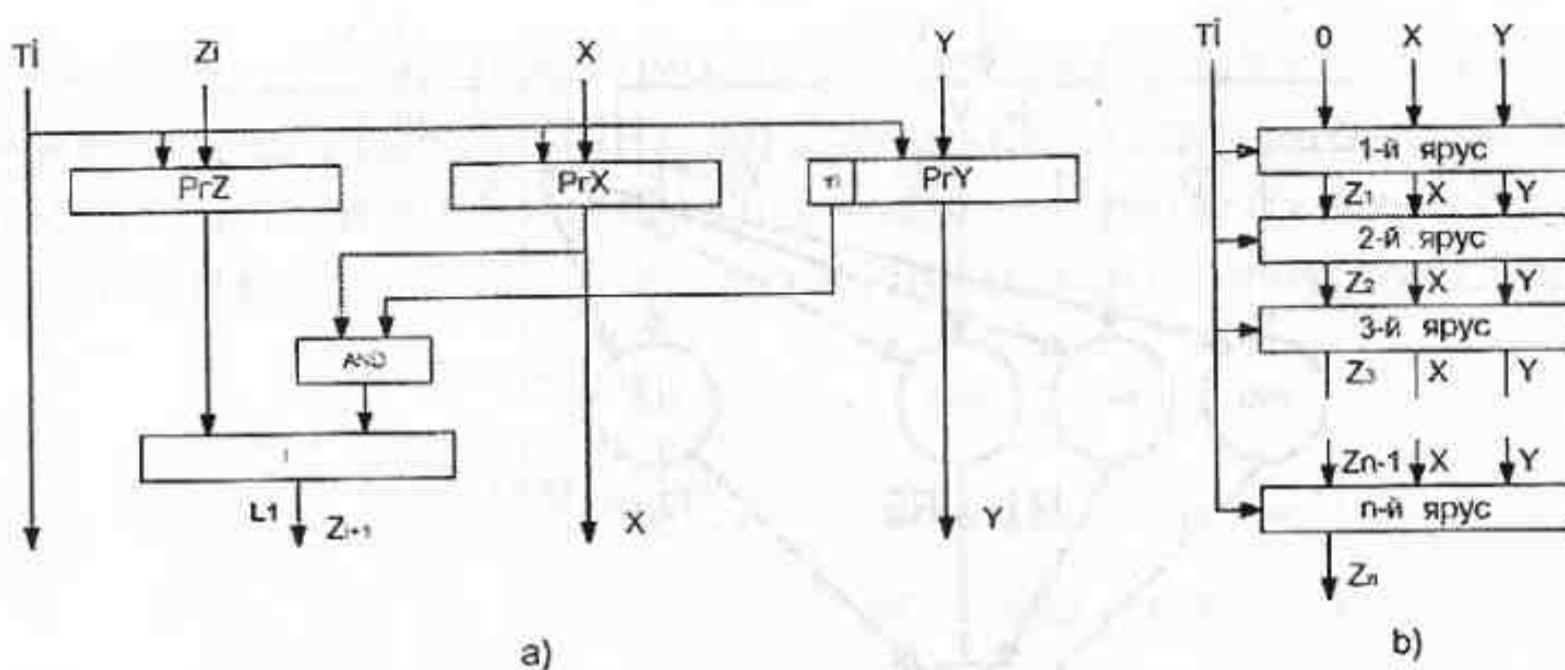


Рис. 7.33. Структура i -го ярусса (а) та конвеєрного операційного пристрою (б) множення двійкових чисел з фіксованою комою

Послідовно з'єднавши n таких ярусів, де n – розрядність даних, як показано на рис. 7.33б, отримаємо структуру конвеєрного операційного пристрою множення двійкових чисел з фіксованою комою.

Аналогічним чином можна побудувати конвеєрні операційні пристрої множення двійкових чисел з фіксованою комою на основі операторів попарного n -розрядного додавання двох чисел відповідно до інших алгоритмів множення, розглянутих в п. 4.4.4.2 розділу 4.

Не є складною і побудова потокового графа алгоритму паралельного матричного виконання багатомісної операції додавання часткових добутків, наприклад з діагональним розповсюдженням переносу відповідно до рис. 4.12, а також реалізація відповідного конвеєрного пристрою багатомісного додавання часткових добутків.

7.13.3 Пристрої ділення двійкових чисел з фіксованою комою

7.13.3.1. Багатотактові пристрої ділення двійкових чисел з фіксованою комою

Як це вже було показано в розділі 4, існує два основних варіанти виконання операції ділення: з зсувом залишків вліво та з зсувом дільника. Для реалізації АОП перший варіант вигідніший, так як вимагає використання n -розрядного віднімача, тоді як другий варіант вимагає використання $2n$ -розрядного віднімача. При цьому перший варіант може бути виконаний двома способами: з відновленням і без відновлення залишку. Схема багатотактового пристрою ділення за алгоритмом з відновленням залишку, який працює відповідно до блок-схеми, наведеної на рис. 4.15, показана на рис. 7.34.

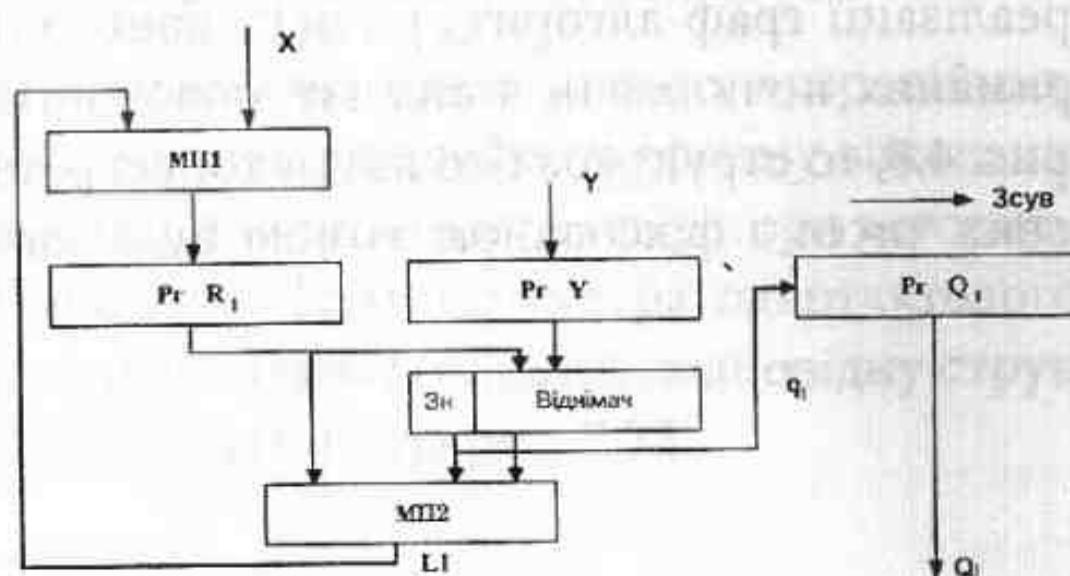


Рис. 7.34. Структура багатотактового АОП ділення двійкових чисел з відновленням залишку

Перед початком виконання операції значення дільника Y та діленого X (через мультиплексор МП1) записуються відповідно до регістрів PrR_i та PrY . В кожному такті послідовно віднімається дільник від діленого і проводиться аналіз значення поточного залишку. Якщо після чергового віднімання залишок додатній, то відповідний розряд частки рівний одиниці. Через мультиплексор МП2 пропускається значення з виходу віднімача, тобто залишок, після чого він зсувається на один розряд вліво і процес повторюється. При від'ємному залишку розряд частки рівний нулю. В цьому випадку виконується коригуюче збільшення дільника до поточного залишку (відновлення залишку), що здійснюється шляхом пропуску через мультиплексор МП значення з регістра PrR_i , після чого він зсувається на один розряд вліво і процес повторюється. В кожному такті визначається один розряд частки, який записується в старший розряд регістру PrQ_i на місце зсунутого розряду. Після виконання n тактів в регістрі PrQ_i буде знаходитись n -розрядна частка від ділення діленого на дільник.

Досить подібною до описаної є схема багатотактового пристроя ділення без відновлення залишку, представлена на рис. 7.35.

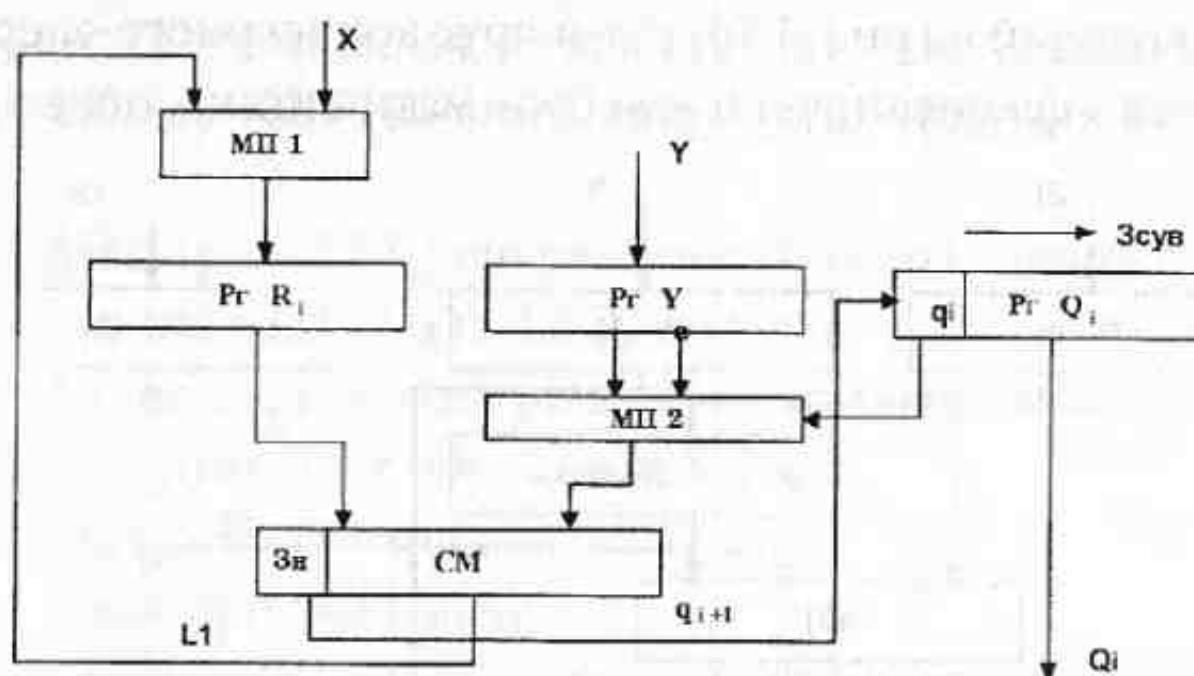


Рис. 7.35. Структура багатотактового АОП ділення двійкових чисел без відновлення залишку

Як і в попередньо розглянутому пристрої, перед початком виконання операції значення дільника Y та діленого X записуються відповідно до регістрів PrY та PrR_i . В кожному такті залежно від значення розряду частки, отриманого на попередньому такті, через мультиплексор МП на суматор СМ проходить прямий або інверсний код дільника, і тим самим дільник додається або віднімається від діленого. Якщо після чергової операції додавання або віднімання залишок додатній, то відповідний розряд частки рівний одиниці, при від'ємному залишку розряд частки рівний нулю. Після виконання операції значення з виходу суматора зсувається на один розряд вліво і процес повторюється. В кожному такті визначається один розряд частки, який записується в старший розряд регістру PrQ_i на місце зсунутого розряду. Після виконання n тактів в регістрі PrQ_i буде знаходитись n -розрядна частка від ділення діленого на дільник.

В обох розглянутих пристроях час виконання ділення дорівнює $T_d = n (t_{MP} + t_{CM} + t_{Pr})$, де складові суми є затримками в мультиплексорі, суматорі та регістрі відповідно.

Потрібно відзначити, що досить близькими до розглянутих алгоритмів і пристрой в ділення є алгоритми і пристрой добування квадратного кореня.

7.13.3.2. Однотактові та конвеєрні пристрої ділення двійкових чисел з фіксованою комою

Подібно до операції множення, побудова однотактових операційних пристрій ділення передбачає повністю аппаратне відображення просторового графа алгоритму виконання операції комбінаційними схемами, які виконують функціональні оператори алгоритму і з'єднані між собою відповідно до графа алгоритму. Тому структура однотактового пристроя множення двійкових чисел з фіксованою комою повторить відповідну структуру графа алгоритму, наведено на рис. 4.16.

При побудові конвеєрного операційного пристроя ділення двійкових чисел з фіксованою комою кожному функціональному оператору алгоритму ставиться у відповідність комбінаційна схема, яка його виконує, і, крім того, комбінаційні схеми, які реалізують функціональні оператори ярусів потокового графа алгоритму, розділяються конвеєрними регістрами. Алгоритм ділення виконується над входними даними при їх однократному проходженні через конвеєрний операційний пристрій.

Якщо вибрати для реалізації граф алгоритму ділення двійкових чисел з відновленням залишку, який представлений на рис. 4.16, то i -й ярус конвеєрного операційного пристроя ділення двійкових чисел з фіксованою комою буде мати вигляд, показаний на рис. 7.36.

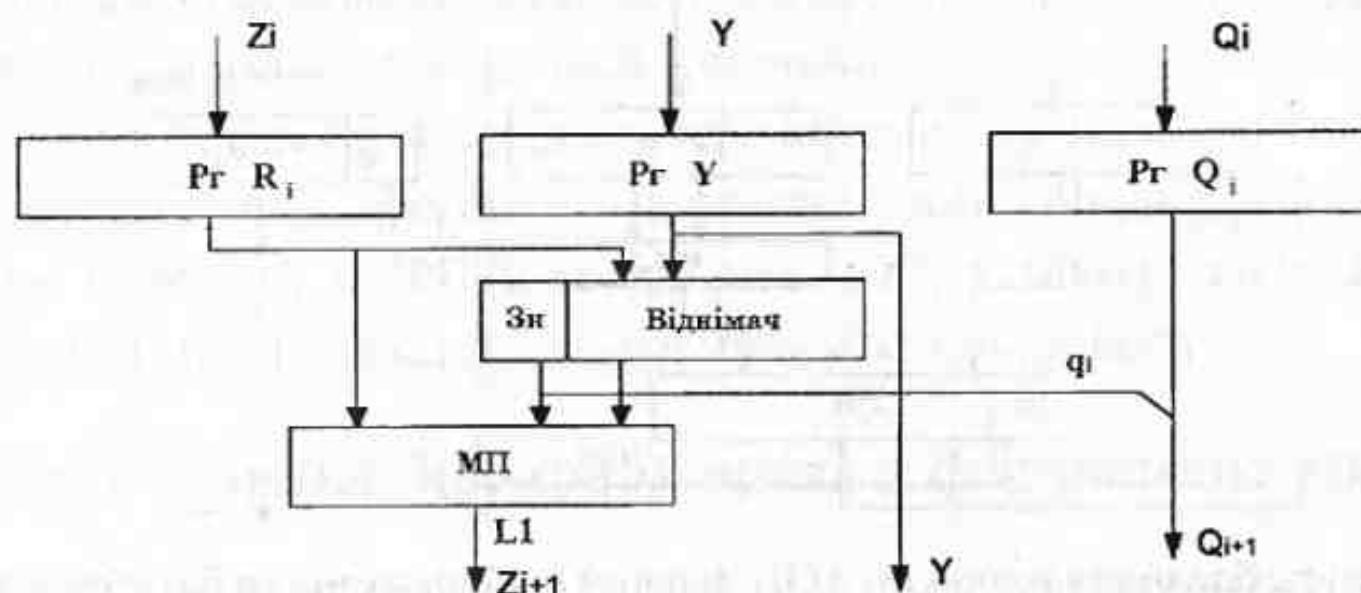


Рис. 7.36. i -й ярус конвеєрного операційного пристроя ділення двійкових чисел з фіксованою комою за алгоритмом з відновленням залишку

Якщо вибрати для реалізації граф алгоритму ділення двійкових чисел без відновлення залишку, то структура i -го яруса конвеєрного операційного пристроя ділення двійкових чисел з фіксованою комою буде мати вигляд, показаний на рис. 7.37a.

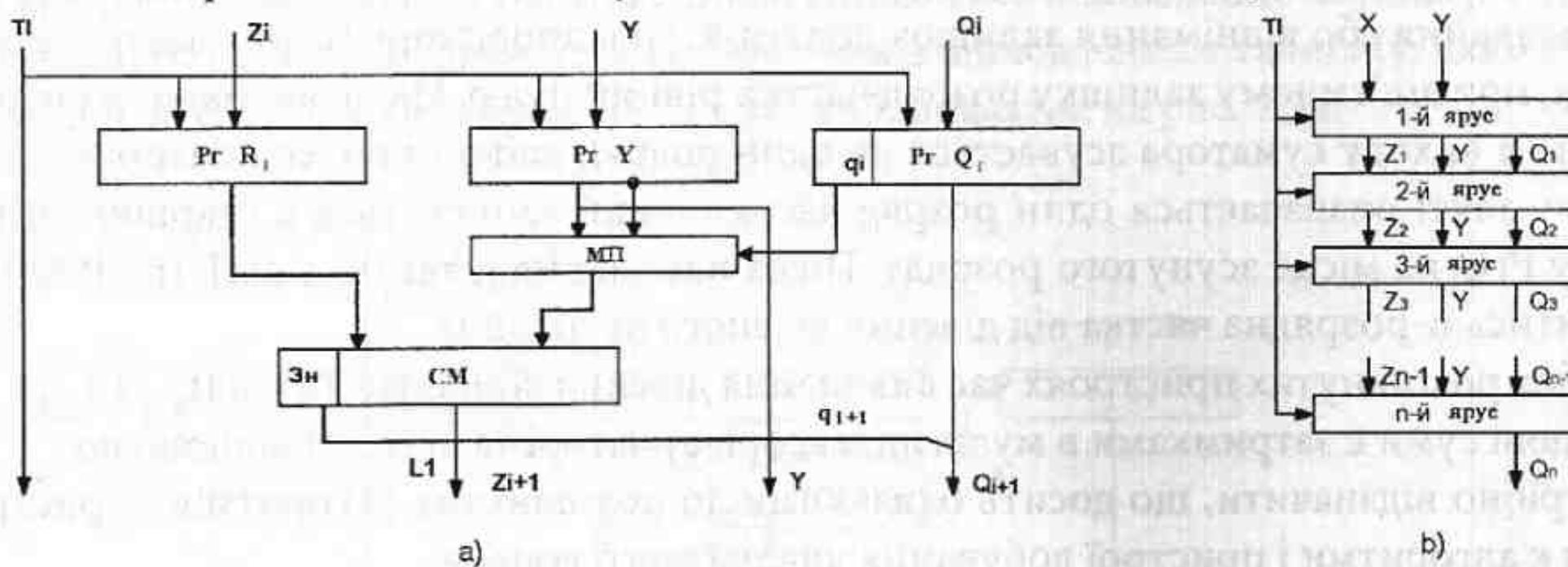


Рис. 7.37. Структура i -го яруса (a) та конвеєрного операційного пристроя (b) ділення двійкових чисел з фіксованою комою за алгоритмом без відновлення залишку

Послідовно з'єднавши п таких ярусів, де n – розрядність частки, як показано на рис. 7.37b, отримаємо структуру конвеєрного операційного пристрою ділення двійкових чисел з фіксованою комою за алгоритмом без відновлення залишку.

7.13.4. Пристрої обчислення елементарних функцій методом "цифра за цифрою"

7.13.4.1. Багатотактовий пристрій обчислення елементарних функцій методом "цифра за цифрою"

В системі команд сучасних комп’ютерів присутня велика кількість команд обчислення елементарних функцій типу $\exp X$, $\ln X$, $\sin X$, $\cos X$, $\operatorname{Sh} X$, $\operatorname{Ch} X$, піднесення до степеня A^m ; $\arctg y/x$ тощо. Виконання цих команд на універсальному АЛП, яке виконує елементарні команди, вимагає значних витрат часу. Навіть реалізація на універсальному АЛП досить простого за складом базових операцій методу "цифра за цифрою" не дає відчутного виграшу в швидкодії внаслідок його специфіки, що знайшла відззеркалення, наприклад, в необхідності виконання зсувів на змінне число розрядів. Тому в ряді сучасних комп’ютерів до складу АЛП вводять операційні пристрої для обчислення елементарних функцій.

Багатотактовий АОП (рис. 7.38), що реалізовує метод "цифра за цифрою" відповідно до ітераційних рівнянь, наведених в п. 4.5.2, містить:

PrX , PrY , PrZ , PrC – регістри для зберігання початкових значень X_0 , Y_0 , Z_0 , та констант C_i , а також результатів проміжних обчислень X_i , Y_i , Z_i .

$C31$, $C32$ – схеми зсуву на i розрядів ($i=1,2,\dots,n$);

$CB1$, $CB2$, $CB3$ – суматори-віднімачі;

ПЗП – постійний запам’ятовуючий пристрій для зберігання констант;

МП – мультиплексор.

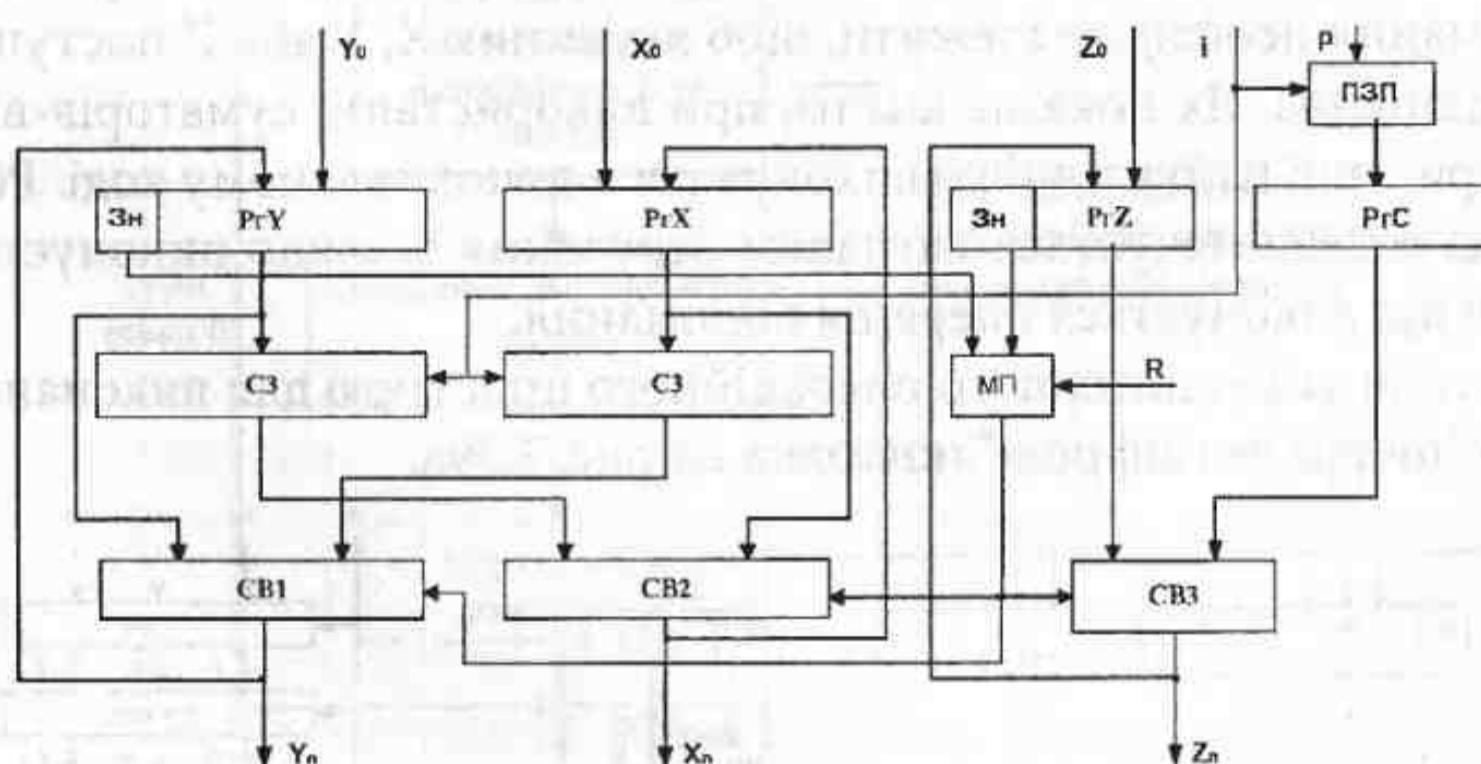


Рис. 7.38. Структура багатотактового АОП обчислення елементарних функцій методом "цифра за цифрою"

Будь-яка з елементарних функцій може бути реалізована на даній структурі за час $T = n (t_{cb} + t_{c3} + t_{pr})$, де t_{cb} – час затримки на суматорі-віднімачі, t_{c3} – час затримки в схемі зсуву, t_{pr} – час запису даних до реєстра, n – розрядність операндів. Окрім розглянутої

тут структури з паралельною обробкою даних, можуть бути реалізовані менш швидкодіючі структури пристройів з порозрядною обробкою, а також багато варіантів проміжних структур.

До основних переваг даного методу з точки зору можливостей його реалізації в АОП належать:

- простота обчислювальних алгоритмів, що базуються лише на трьох операціях: додавання/віднімання, зсув і вибірка з ПЗП;
- однотипність обчислювальних алгоритмів для обчислення майже всіх елементарних функцій;
- однотипність виконання кожної ітерації;
- можливість побудови багатофункціональних АОП, що використовують даний метод;
- похибки цього методу достатньо повно досліджені та легко компенсируються шляхом введення додаткових розрядів.

7.13.4.2. Однотактовий та конвеєрний операційні пристрої обчислення елементарних функцій методом "цифра за цифрою"

Постійність констант і кількості розрядів зсуву на кожній ітерації дозволяє при реалізації однотактового операційного пристрою видалити ПЗП для зберігання констант і схеми зсуву, які необхідні в багатотактових операційних пристроях (рис. 7.38).

Зсуви здійснюються шляхом відповідного з'єднання розрядів операційних елементів, а константи формуються на вході операційного елементу шляхом подачі логічного нуля або одиниці у відповідний розряд. Таким чином, для реалізації однієї ітерації алгоритму "цифра за цифрою" необхідно виконати одну операцію віднімання і дві операції додавання, якщо оператор f_i має значення -1, або одну операцію додавання і дві операції віднімання, якщо оператор f_i має значення +1. Операції додавання і віднімання кодів чисел можна виконувати використовуючи комбінаційну схему суматора-віднімача. При виконанні віднімання необхідно стежити, щоб від'ємник X, Y або Z поступав на другий вхід суматора-віднімача. Як показав аналіз, при використанні суматорів-віднімачів обробку даних в пристрої найвигідніше виконувати в доповняльному коді. Режим роботи суматора-віднімача забезпечується сигналом керування 0, якщо виконується операція додавання, і 1, якщо виконується операція віднімання.

Структура i-го яруса конвеєрного операційного пристрою для виконання однієї ітерації алгоритму "цифра за цифрою" показана на рис. 7.39а.

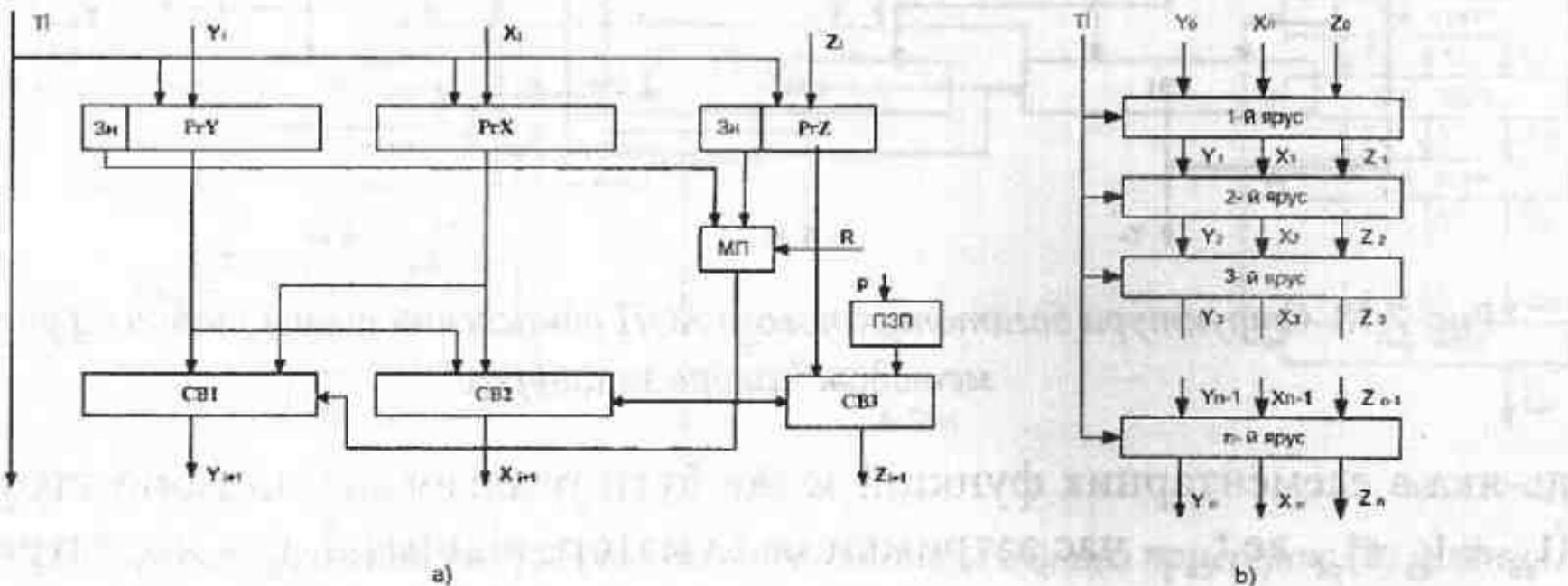


Рис. 7.39. Структура пристрою для виконання однієї ітерації алгоритму "цифра за цифрою"

Параметр r управлює подачею на суматор-віднімач СВЗ констант C_i . Параметр f_i управляє суматорами-віднімачами, поступаючи із знакового розряду Y_i або Z_i через комутатор КМ1 залежно від коду операції R .

Включивши послідовно n таких пристрій, де n – кількість ітерацій, одержимо конвеєрний операційний пристрій виконання алгоритму “цифра за цифрою” (рис. 7.39б). Якщо з описаного пристрою видалити конвеєрні регістри, отримаємо однотактовий операційний пристрій виконання алгоритму “цифра за цифрою”.

7.13.5. Пристрої для виконання арифметичних операцій над числами з рухомою комою

7.13.5.1. Пристрої додавання і віднімання чисел з рухомою комою

Як відомо, формат з рухомою комою передбачає наявність двох частин числа – порядку (P) та мантиси (M). Числа представляються у вигляді $X = M_x 2^{P_x}$, $Y = M_y 2^{P_y}$.

При виконанні додавання та віднімання порядки операндів вирівнюються, а мантиси додаються. Порядки вирівнюються збільшенням порядку меншого операнду до значення порядку більшого операнду. Цей порядок є порядком результату. Щоб при вирівнюванні порядків величина операнда не змінювалася, його мантиса одночасно зменшується. Після виконання вирівнювання порядків виникають додаткові молодші розряди мантиси, які до, або після додавання, відкидаються або заокруглюються.

Блок-схема додавання та віднімання двійкових чисел з рухомою комою наведена на рис. 7.40.



Рис. 7.40. Блок-схема додавання та віднімання двійкових чисел з рухомою комою

Детальніший розпис алгоритму додавання та віднімання двійкових чисел з рухомою комою наведено нижче, а структура однотактового операційного пристрою додавання та віднімання двійкових чисел з рухомою комою наведена на рис. 7.41. Послідовність кроків алгоритму та відповідні їм вузли пристрою є наступними:

- Віднімання порядків чисел з метою знаходження їх різниці та визначення, яке з двох чисел є більшим, а яке меншим, для чого в схемі використано віднімач.
- Зсув праворуч мантиси числа, яке має менший порядок. Для цього знак з віднімача керує мультиплексорами, які пропускають на вхід схеми зсуву відповідну мантису.
- Додавання іншої мантиси до зсунутої, яке виконується на суматорі.
- Визначення кількості нулів в старших розрядах отриманої суми з метою визначення кількості розрядів зсуву при нормалізації, для чого в пристрій введена відповідна схема.
- Зсув на схемі зсуву цієї суми вліво на кількість розрядів, рівну кількості нулів в її старших розрядах.
- Віднімання від більшого порядку числа, рівного кількості нулів в старших розрядах отриманої суми, тобто коригування порядку.

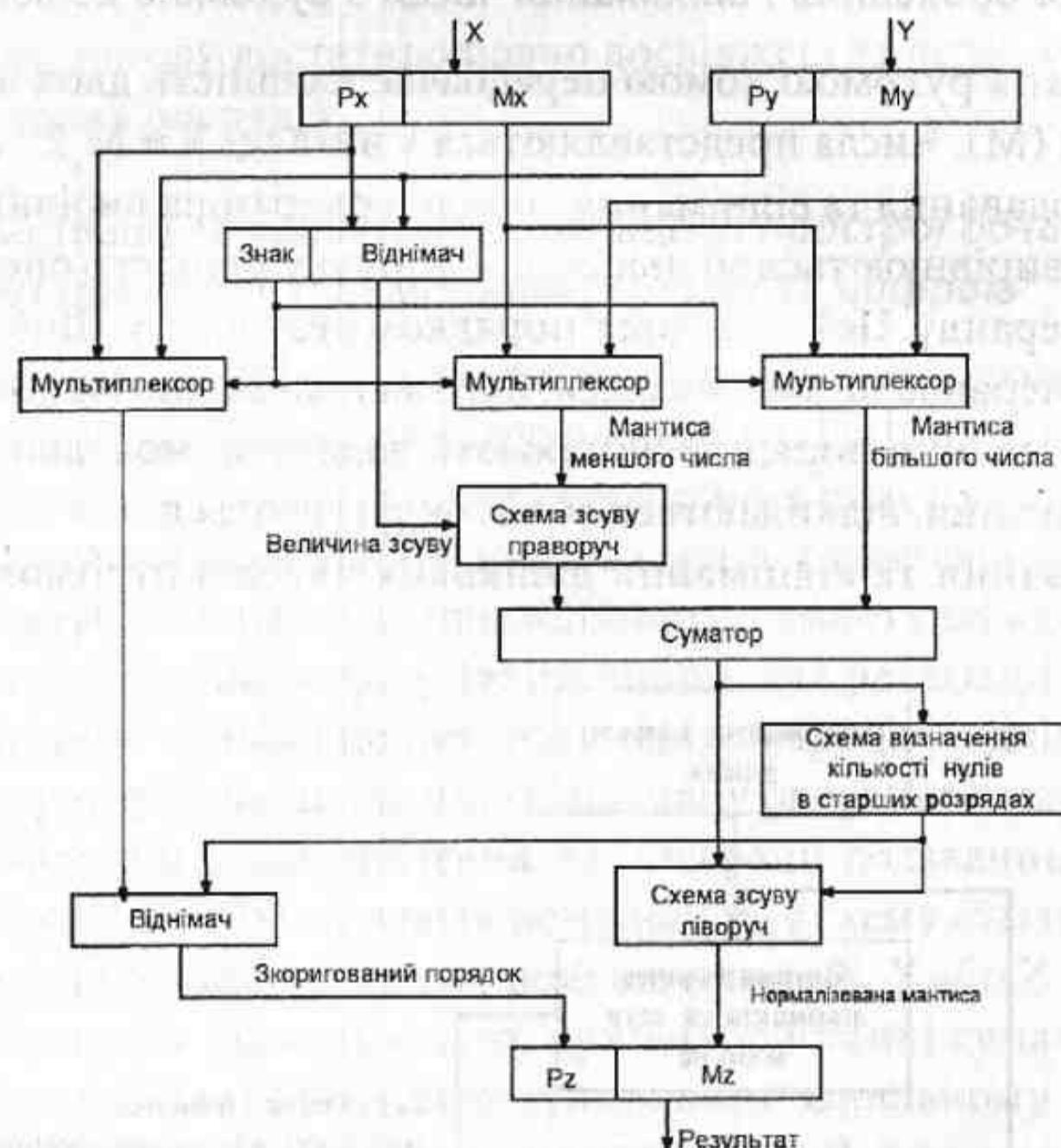


Рис. 7.41. Структура однотактового операційного пристрою додавання та віднімання двійкових чисел з рухомою комою

При необхідності представлений на рис. 7.41 операційний пристрій додавання та віднімання двійкових чисел з рухомою комою може бути конвеєризований шляхом введення конвеєрних реєстрів між його ярусами.

7.13.5.2. Пристрой множення та ділення чисел з рухомою комою

Пристрой множення та ділення чисел з рухомою комою будуються на основі суматорів, піднімачів, перемножувачів та подільників з фіксованою комою, які вже були розглянуті, а також пристроя нормалізації чисел. Побудова пристроя множення та ділення чисел з рухомою комою на основі названих пристріїв не є складною, оскільки алгоритми є досить простими. Тому розглянемо далі лише питання забезпечення коректного виконання цих операцій.

Операції множення та ділення чисел з рухомою комою описуються відповідно виразами:

$$Z = M_x 2^{P_x} \cdot M_y 2^{P_y} = M_x M_y 2^{(P_x+P_y)}$$

$$Z = X/Y = M_x 2^{P_x} / M_y 2^{P_y} = M_x / M_y 2^{P_x-P_y}$$

Блок-схеми виконання цих виразів представлено відповідно на рис. 7.42 а) та б).

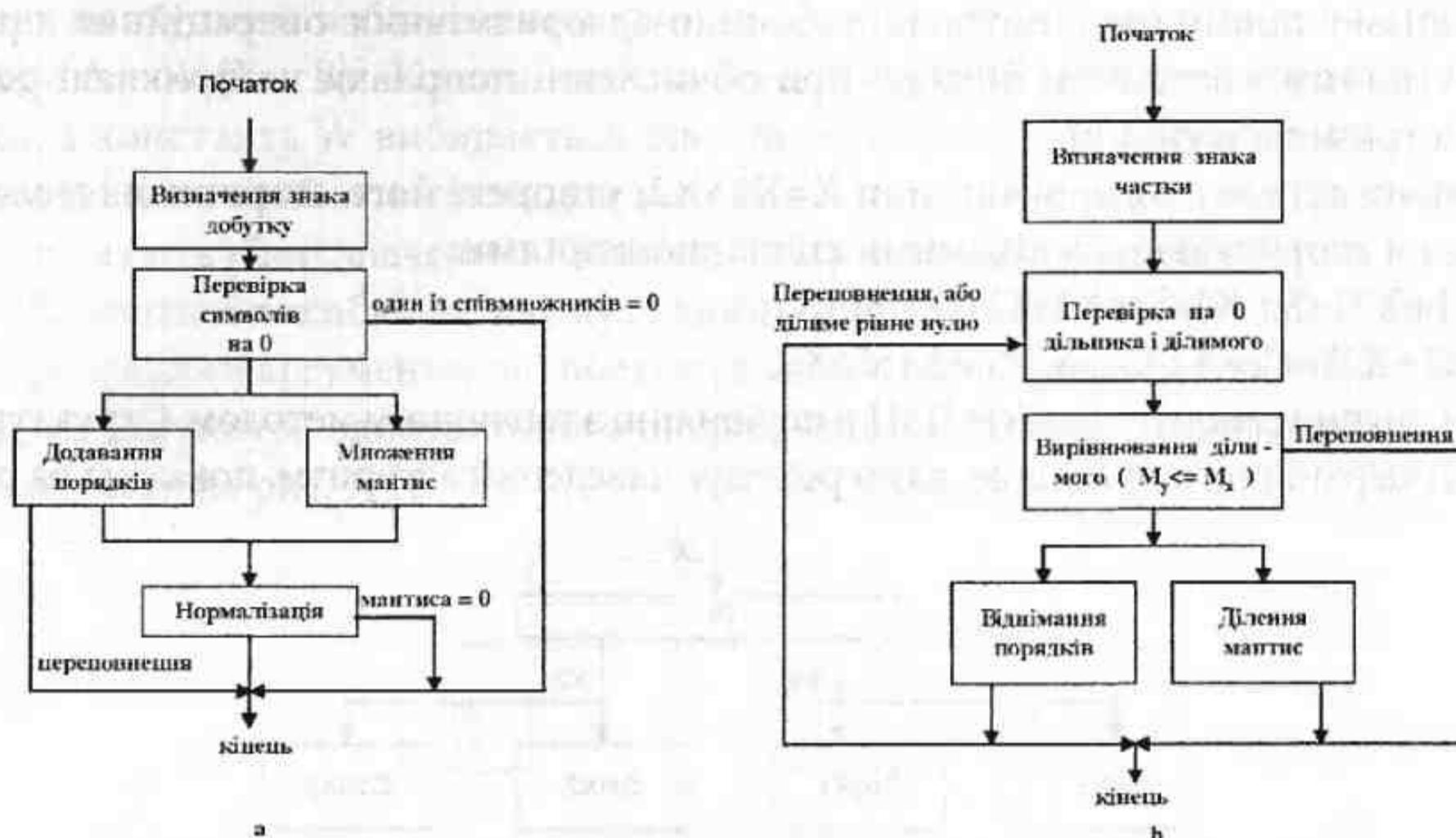


Рис. 7.42. Блок-схеми виконання операцій множення а) та ділення б) чисел з рухомою комою

7.14. Таблично-алгоритмічні операційні пристрої

Арифметичні операції в таблично-алгоритмічних операційних пристроях необхідні для обчислення поправки. Ці обчислення можуть базуватися на використанні різних методів, таких як: дробово-раціональні наближення, ітераційні процеси, розкладання в ряди, наближення поліномами, ланцюговими дробами і т. д. При обчисленні таблично-алгоритмічним методом аргумент звичайно розбивається на дві частини. Розбивка аргументу на більшу кількість частин при неконвеерній реалізації нераціональна через значну кількість додаткових операцій. При конвеерній же реалізації така розбивка в ряді випадків може виявитися вигідною.

Застосування конвеерного принципу обробки для виконання арифметичних операцій у таблично-алгоритмічних операційних пристроях дозволяє значно скоротити об'єми пам'яті в порівнянні з прямим табличним методом без зменшення, або і з підвищенням їх продуктивності. Продуктивність операційного пристроя, в якому застосований таблично-алгоритмічний метод, може бути вищою, ніж при використанні табличного методу, тому що швидкодія ПЗП залежить від розрядності операндів n , i , при великих вимогах по точності, може перевершувати затримку в одному ярусі операційного пристроя, яку можна довести до затримки в ПЗП значно меншого об'єму.

Основною проблемою при синтезі таблично-алгоритмічних операційних пристрояв є вибір із всіх існуючих методів обчислення даної функції чи набору функцій найбільш ефективного і визначення оптимальних співвідношень між об'ємом ПЗП і витратами обладнання на арифметичну частину при забезпеченні потрібної продуктивності.

В даний час існує багато таблично-алгоритмічних методів обчислення елементарних функцій, частина з яких була розглянута в розділі 4. Тут буде розглянута реалізація декількох таблично-алгоритмічних методів обчислення елементарних функцій, в яких застосований як загальний, так і частковий підхід при обчисленні поправки. Проектування таблично-алгоритмічних операційних пристрій за іншими, не розглянутими тут, алгоритмами може бути виконане аналогічно приведеному.

Розглянемо принципи побудови таблично-алгоритмічних операційних пристрій з використанням часткового підходу при обчисленні поправки на прикладі реалізації тригонометричних функцій.

Розбивши аргумент на дві частини $X=X_1+X_2$, утворені його старшими і молодшими розрядами, і скориставшись відомими співвідношеннями:

$$\sin(X_1+X_2)=\sin X_1 \cos X_2 + \cos X_1 \sin X_2;$$

$$\cos(X_1+X_2)=\cos X_1 \cos X_2 - \sin X_1 \sin X_2,$$

можна значно скратити об'єм ПЗП в порівнянні з табличним методом. Структура однотактового операційного пристрою, який реалізує наведений алгоритм, показана на рис. 7.43.

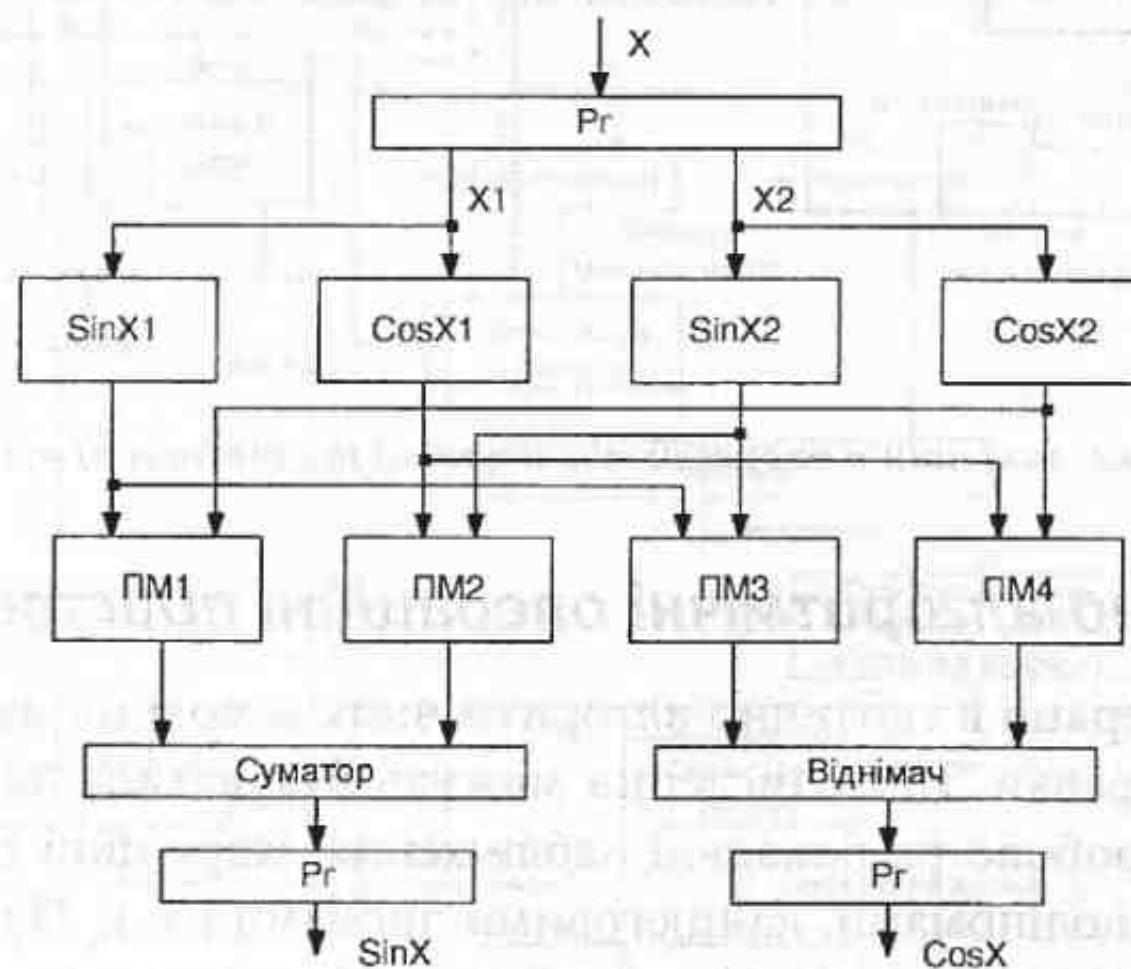


Рис. 7.43. Структура однотактового таблично-алгоритмічного операційного пристрою обчислення тригонометричних функцій

Якщо прийняти, що X_1 – ціле число градусів з дискретністю 1, а X_2 – дробове число градусів, з дискретністю 0,01, то неважко підрахувати, що таблиці, котрі зберігають $\sin X_1$ і $\cos X_1$, будуть містити по 90 15-розрядних слів, а таблиці, що зберігають $\sin X_2$ і $\cos X_2$ – по 1000 15-розрядних слів. Загальний об'єм ПЗП складе 5700 бітів, що в 24 рази менше, ніж при обчисленні табличним методом. В якості ПЗП з уже прошитими значеннями тригонометричних функцій можна використати мікросхеми ДО505РЕ30068-ДО505РЕ300071 ємністю 4 Кбіти кожна. Перші дві мікросхеми містять 8 старших розрядів 512 значень функції $\sin X$ у діапазоні 0... $\pi/4$, а дві інші мікросхеми містять значення цієї функції в діапазоні $\pi/4$... $\pi/2$.

Ввівши конвеєрні реєстри, включаючи і в перемножувачах ПМ, які також можна побудувати за конвеєрним принципом, можна підвищити продуктивність описаного пристрою. Зрозуміло, що зменшивши об'єми таблиць, тут збільшенні затрати обладнання на операційну частину, тобто введені чотири перемножувачі та суматор і віднімач.

Принципи побудови таблично-алгоритмічних операційних пристрой з використанням загального підходу при обчисленні поправки розглянемо на прикладі обчислення елементарних функцій з використанням раціонального наближення.

Нехай обробці підлягають нормалізовані числа, представлені у форматі з фіксованою комою. Обчислення елементарних функцій будемо робити на основі методу сегментної апроксимації, відповідно до якого діапазон зміни аргументу $[0,5;1]$ розбивається на інтервали з наступним наближенням функції на кожнім інтервалі за допомогою виразу $Y = F(X) = A + W(X + B)^2$. Константи A та B вибираються з умови мінімізації абсолютної похибки, а константа W вибирається рівною ступеню числа 2 (основа системи числення), що дозволяє замінити операцію множення операцією зсуву. На різних інтервалах константи мають різні значення. Кількість інтервалів також визначається з умови мінімізації абсолютної похибки, причому граници інтервалів визначаються к старшим двійковими розрядами аргументу, що полегшує здійснення адресації пам'яті, в яку записані константи. Структура однотактового операційного пристрою, який реалізує наведений вираз, показана на рис. 7.44.

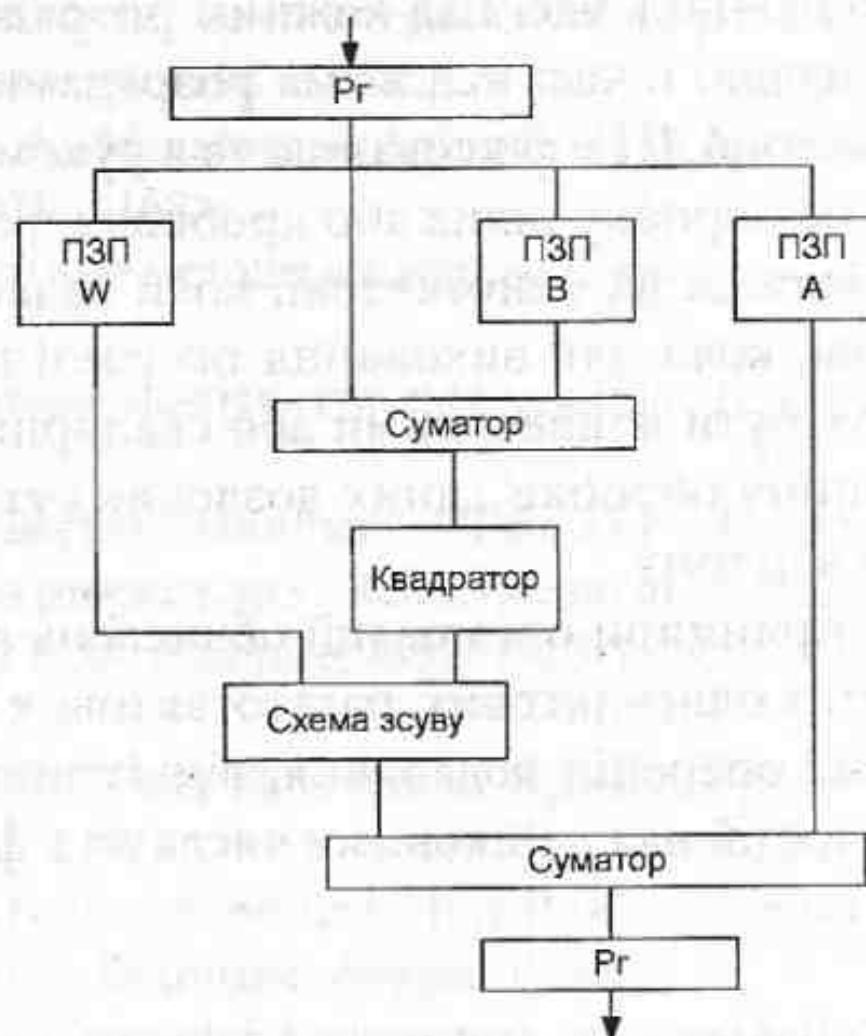


Рис. 7.44. Структура однотактового таблично-алгоритмічного операційного пристрою обчислення елементарних функцій

Для реалізації описаного необхідно три блоки постійної пам'яті для збереження коефіцієнтів, схему зсуву і два суматори. Для 12-ти розрядних операндів загальний об'єм ПЗП складає 8960 бітів. Константи A, B, W обчислюють з тим, щоб забезпечити задану точність обчислення. Вихідними даними для реалізації алгоритму розрахунку коефіцієнтів є: реалізована функція, величина інтервалу зміни аргументу, розрядність функції та аргументу. Після обчислення отримують наступні величини: сумарний об'єм ПЗП, число підінтервалів зміни аргументу, число розрядів, необхідних для кодування номера підінтервалу, значення констант A, B, W . У табл. 7.6 як приклад наведені значення констант, граници підінтервалів і абсолютнона похибка обчислень τ при обчисленні функції I/X . Пристрій, що реалізує формулу $y=A+W(X+B)^2$ є багатофункціональним. Для різних функцій константи мають різні значення. Включення конвеєра команд дозволяє обчислювати на такому пристрої одночасно k функцій від k даних.

Подібно до описаного, описані підходи можна застосувати до інших таблично-алгоритмічних методів обчислення елементарних функцій, наприклад, при використанні ланцюгових дробів, раціональних наближень і т. д.

7.15. Короткий зміст розділу

Розглянуто принципи побудови арифметико-логічного пристрою сучасних комп'ютерів, який є одним з основних вузлів процесора, призначеним для виконання арифметичних, логічних та інших операцій обробки даних. Розкрито структуру АЛП для виконання елементарних операцій та структуру багатоблокових АЛП, призначених для виконання складних операцій, які ініціюються командами обробки даних з системи команд комп'ютера. Описані багатоблокові АЛП з внутрішньою реєстровою пам'яттю на основі табличних, однотактових, багатотактових та конвеєрних операційних пристрій. Проведено класифікацію АЛП залежно від способу обробки операндів. АЛП діляться на послідовні, послідовно-паралельні та паралельні. В першому випадку обробка операндів в АЛП здійснюється послідовно в часі над кожним розрядом, тоді як в останньому операції здійснюються паралельно в часі над всіма розрядами операндів. За способом представлення чисел розділяють АЛП з фіксованою та з рухомою комою, причому перші можуть бути орієнтовані на обробку цілих або дробових чисел. Залежно від способу виконання операцій АЛП діляться на однотактові, коли задана операція виконується за один такт, та багатотактові, коли для виконання операції потрібно виконати деяку кількість тактів. АЛП можуть бути конвеєрними або скалярними. Показано, що використання конвеєрного принципу обробки даних дозволяє суттєво підвищити продуктивність АЛП та комп'ютера в цілому.

Розглянуто структури та принципи організації обчислень в табличних, алгоритмічних та таблично-алгоритмічних однотактових, багатотактових та конвеєрних операційних пристроях для виконання операцій додавання, віднімання, множення, ділення та обчислення елементарних функцій над двійковими числами в форматах з фіксованою та рухомою комою.

7.16. Література для подальшого читання

Питання побудови арифметико-логічного пристрою розглянуті в багатьох підручниках з організації комп'ютерів, зокрема в роботах [1–8]. Формалізації питань побудови багатотактових операційних пристрій присвячена робота [9], в якій вони названі операційними пристроями з закріпленими мікроопераціями. Опис стандартного 4-розрядного АЛП приведено в роботі [3]. В роботах [10–12] є опис структури арифметико-логічного пристрою процесорів Nios, UltraSPARC та PA-8000. Принципи побудови табличних та таблично-алгоритмічних операційних пристрій розглянуті в роботах [13–15]. Принципи побудови однотактових, багатотактових та конвеєрних операційних пристрій запропоновано в роботах [17–19].

7.17. Література до розділу 7

1. Благовещенский Ю. В., Теслер Г. С. Вычисление элементарных функций на ЭВМ. – К., "Техника", 1977. – 208 с.

2. Байков В. Д., Смолов В. Б. Аппаратурная реализация элементарных функций в ЦВМ. – Л. ЛГУ. – 96с.
3. Каган Б. М. Электронные вычислительные машины и системы. М.: Энергия, 1979. – 528 с.
4. Каган Б. М., Каневский М. М. Цифровые вычислительные машины и системы. М.: Энергия, 1974. – 680 с.
5. Майоров С. А., Новиков Г. И. Структура электронных вычислительных машин. Л. Машиностроение. 1979. – 384 с.
6. Мельник А.А. Выбор метода вычисления элементарных функций в процессорах обработки сигналов. Тезисы Всесоюзной конференции “Методы и микроэлектронные средства цифрового преобразования и обработки сигналов”. – Рига, 1983.
7. Мельник А.А. О вычислении одного класса элементарных функций путем конвейерной реализации метода Волдера. -Автоматика и вычислительная техника, 1983, N 6.
8. Мельник А.А. Использование алгоритма Волдера в высокопроизводительных вычислительных БПФ. - Автометрия, 1984, N 6, с. 85-87.
9. Мельник А.А. Процессоры обработки сигналов. Препринт N 29-89, ИППММ АН УССР, 1989, 63 с.
10. Мельник А.О. Спеціалізовані комп’ютерні системи реального часу. – Львів: Державний університет “Львівська політехніка”, 1996. – 54 с.
11. Коркішко Т., Мельник А., Мельник В. Алгоритми та процесори симетричного блокового шифрування. – Львів: БаK, 2003. – 168 с.
12. Оранский А.М. Аппаратные методы в цифровой вычислительной технике. -Минск, Из-во БГУ, 1977. – 208 с.
13. Справочник по цифровой вычислительной технике. Б.Н. Малиновский и др. К. Техніка, 1980. – 320 с.
14. Угрюмов Е.П. Цифровая схемотехника. – СПб.: БХВ – Санкт-Петербург, 2000. – 528 с.
15. Altera Corporation. Nios programmer’s Reference manual. March 2001.
16. Kane, Gerry. PA-RISC 2.0 Architecture, ISBN 0-13-182734-0, Prentice Hall, Englewood Cliffs, NJ, 1996.
17. Melnyk A. Synthesis of the Data Flow Graph Pipeline Operation Devices. IWK-95, Ilmenau, 1995.
18. Melnyk A.O. The Architecture of the Teal-Time Processing System Optimized to the Data Flow Intensity. International Conference, Zakopane, August, 1996.
19. D. Patterson, J. Hennessy. Computer Architecture. A Quantitative Approach. Morgan Kaufmann Publishers, Inc. 1996.
20. Patterson, D. A., & Hennessy, J. L. Computer Organization and Design, The Hardware/Software Interface, 2nd ed., San Mateo, CA: Morgan Kaufmann, 1997.
21. Stallings, W. Computer Organization and Architecture, 5th ed., New York, NY: Macmillan Publishing Company, 2000.
22. Sun Microelectronics. UltraSPARC I&II. Sun Microelectronics. 1997.
23. Tanenbaum, Andrew. Structured Computer Organization, 4th ed., Upper Saddle River, NJ: Prentice Hall, 1999.
24. Volder J.E. The CORDIC trigonometric computing technique. – “IRE Trans.”, 1959, 3, pp. 330-334.

7.18. Питання до розділу 7

1. Назвіть місце АЛП в комп’ютері.
2. Назвіть функції АЛП.
3. Яким чином АЛП взаємодіє з іншими вузлами процесора.
4. Наведіть класифікацію АЛП.

5. Порівняйте послідовний, паралельний та послідовно-паралельний способи обробки інформації в АЛП.
6. Поясніть роботу АЛП для виконання елементарних операцій.
7. Назвіть елементарні операції АЛП. Чому до складу системи команд сучасних комп'ютерів входять команди виконання елементарних операцій?
8. Назвіть складні арифметичні і логічні операції АЛП.
9. Поясніть, що таке граф алгоритму та як його можна використати при виборі структури операційного пристрою.
10. Приведіть класифікацію операційних пристройів.
11. Як організована робота табличних операційних пристройів?
12. Поясніть принципи роботи багатотактових операційних пристройів.
13. Поясніть принципи роботи однотактових операційних пристройів.
14. Поясніть принципи роботи конвеєрних операційних пристройів.
15. Наведіть схему та опишіть роботу послідовного АЛП додавання та віднімання двійкових чисел з фіксованою комою.
16. Наведіть схему та опишіть роботу паралельного АЛП додавання та віднімання двійкових чисел з фіксованою комою.
17. Як побудований однотактовий суматор двійкових чисел за методом вибору переносу?
18. Які є методи прискорення роботи паралельного АЛП додавання та віднімання двійкових чисел з фіксованою комою?
19. Назвіть чотири методи та чотири базові структури множення двійкових чисел з фіксованою комою.
20. Поясніть роботу багатотактового пристрою множення двійкових чисел з молодших розрядів множника при нерухомому множенному з зсувом суми часткових добутків.
21. Поясніть роботу багатотактового пристрою множення двійкових чисел з молодших розрядів при нерухомій сумі часткових добутків з зсувом множеного вліво.
22. Поясніть роботу багатотактового пристрою множення двійкових чисел з старших розрядів при нерухомій сумі часткових добутків з зсувом множеного вправо.
23. Поясніть роботу багатотактового пристрою множення двійкових чисел з старших розрядів при нерухомому множенному з зсувом суми часткових добутків вліво.
24. Як будується однотактовий пристрій множення двійкових чисел з фіксованою комою?
25. Наведіть структуру конвеєрного операційного пристрою множення двійкових чисел з фіксованою комою.
26. Наведіть структуру багатотактового АОП ділення двійкових чисел з відновленням залишку.
27. Наведіть структуру багатотактового АОП ділення двійкових чисел без відновлення залишку.
28. Наведіть структуру конвеєрного операційного пристрою ділення двійкових чисел з фіксованою комою за алгоритмом з відновленням залишку.
29. Наведіть структуру конвеєрного операційного пристрою ділення двійкових чисел з фіксованою комою за алгоритмом без відновлення залишку.
30. Поясніть роботу багатотактового пристрою для обчислення елементарних функцій методом "цифра за цифрою".
31. Поясніть роботу конвеєрного пристрою для обчислення елементарних функцій методом "цифра за цифрою".
32. Як будується пристрій додавання і віднімання чисел з рухомою комою?
33. Як будується пристрій множення та ділення чисел з рухомою комою?
34. Поясніть роботу операційного пристрою для обчислення елементарних функцій таблично-алгоритмічним методом.