МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

ЩЕРБАКОВ А.Н., ПРОСКУРІН М.П., ГРУШКО С.С.

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

ЧАСТИНА 1
КОМП'ЮТЕРНА АРИФМЕТИКА

ТЕКСТИ ЛЕКЦІЙ
для студентів усіх форм навчання спеціальностей
8.091501 – «Комп'ютерні системи і мережі» і
7.091503 – «Спеціалізовані комп'ютерні системи»
кафедри «Комп'ютерні системи і мережі»

2010

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ, частина 1 КОМП'ЮТЕРНА АРИФМЕТИКА. Тексти лекцій для студентів усіх форм навчання спеціальностей 8.091501 – «Комп'ютерні системи і мережі» і 7.091503 – «Спеціалізовані комп'ютерні системи» кафедри «Комп'ютерні системи і мережі» /Укладачі: ЩЕРБАКОВ А.М., ПРОСКУРІН М.П., ГРУШКО С.С.– ЗАПОРІЖЖЯ: ЗНТУ, 2010.– 102с.

Видання друге на укр. мові, виправлене та доповнене.

Рекомендовано до видання НМК як тексти лекцій з дисципліни «Прикладна теорія цифрових автоматів», частина 1 «Комп'ютерна арифметика» для спеціальностей 8.091501 – «Комп'ютерні системи і мережі» і 7.091503 – «Спеціалізовані комп'ютерні системи» на засіданні кафедри «Комп'ютерні системи і мережі».

Протокол №3 від 09 грудня 2009.

Укладачі: А.Н. Щербаков, доцент, к.т.н.; М.П. Проскурін, доцент, к.т.н.; С.С. Грушко, асистент.

Рецензент: А.К. Тимовський, доцент, к.т.н.

Відповідальний за випуск: М.П. Проскурін, доцент, к.т.н.

Затверджений на засіданні кафедри
«Комп'ютерні системи і мережі
Протокол №3 від 09 грудня 2009.

ЗМІСТ

ВСТУП

Поява електронних обчислювальних машин (ЕОМ) – це подія, яка ознаменувала початок другої промислової революції. Комп'ютерна техніка допомогла значно підвищити ефективність розумової праці людини, що пов'язана з обчислювальними і кібернетичними процесами, з моделюванням і дослідженнями. Вона поставила на новий, вищий технічний рівень питання, пов'язані з автоматизацією технологічних виробничих процесів, контролем і управлінням, накопиченням і обробкою інформації. Важко знайти область техніки або науки, куди не проникла або не могла б проникнути обчислювальна техніка (ОТ).

З'явившись близько 70-ти років тому, вона порівняно швидко пройшла стадії зміни застарілих поколінь комп'ютерів новими, підвищуючи, при цьому, швидкодію, надійність в роботі, якість, ступінь складності вирішуваних задач, адаптацію користувача.

Автоматизовану арифметичну машину, яку побудував англійський учений Ч. Беббідж [1-21], можна вважати прообразом сучасних комп'ютерів оскільки вона містила "склад" зберігання чисел (аналог пам'яті), "млини" виробництва арифметичних операцій (аналог арифметично-логічного пристрою АЛП), пристроїв управління (ПУ) та введення і виведення інформації (ПВВІ). Швидкість виконання операцій складання (віднімання) складала одну секунду, множення (ділення) – одну хвилину.

У 1947 р. була побудована релейна обчислювальна машина "Марк-2", що містила 13 тисяч реле з двома стійкими станами, тобто вперше використовувалася двійкова система числення. Операція складання складала 0,125 секунди, множення – 0,25 секунди.

У 1943 р. у Гарвардському університеті США Мочлі Джон і Преспер Екерт приступили до створення першої ЕОМ "Марк-1 "на електронних лампах, а у 1945 р. закінчили її будівництво. Вона мала величезні розміри, містила 18 тисяч електронних ламп, 1500 реле, споживала близько 150 кВт. В ній підвищилася швидкість до 5000 елементарних операцій в секунду, але програми виконання операцій набиралися вручну. У 1945 р. до процесу створення ЕОМ був підключений знаменитий математик Джон фон Нейман, що виклав основні принципи побудови ЕОМ.

У 1953 р. в СРСР була побудована цифрова електронна обчислювальна машина (ЕОМ) з швидкодією 8000 операцій в секунду (БЕСМ), краща для Європи того часу. За ними слідують БЕСМ-2, М-2, "Стрела", "Урал", М-3, «Минск-1», ін.

Основу другого покоління склали напівпровідникові діоди і транзистори. Їх швидкодія досягла 10-20 тис. операцій в секунду.

Найбільш потужною для того часу була ЕОМ БЕСМ-6, створена під керівництвом С.О. Лебедєва. Всі схеми машини були записані в рівняннях булевої алгебри. Вона була нескладна в експлуатації та мала високу для того часу надійність. Вона не була копією чиєїсь раніш створеною ЕОМ і використовувалася довгий час при моделюванні і математичних розрахунках.

У ЕОМ третього покоління використані великі інтегральні схеми (ВІС), четвертого покоління – надвеликі інтегральні схеми (НВІС). У одній такий НВІС об'ємом в долі кубічних сантиметрів вміщується ціла шафа ЕОМ першого покоління. При цьому, швидкодія зросла до сотень мільйонів операцій в секунду (один із основних показників продуктивності праці ЕОМ).

Вважають, що в комп'ютерах майбутнього можуть використовувати оптоелектроніку на основі когерентного випромінювання, оскільки швидкість світла вища за швидкість електронів. Перспективними напрямами є побудова біосистем людина – ЕОМ, створення інтелектуальних систем штучного розуму і багато інших.

Пошук нової елементної бази, методів і принципів створення нових поколінь комп'ютерів і комп'ютерних систем, вдосконалення вже відомих теоретичних основ виконання арифметичних і логічних операцій, пошук нових фізичних принципів зберігання і передачі інформації – ось далеко не повний перелік завдань, які вирішуватимуть фахівці в області створення комп'ютерної техніки (КТ) і комп'ютерних систем (КС).

1 ОСНОВНІ ВИЗНАЧЕННЯ І СКЛАД КОМП'ЮТЕРНИХ СИСТЕМ

1.1 Мета дисципліни

Мета дисципліни – викласти студентам теоретичні і практичні основи комп'ютерної арифметики, алгебри логіки і виконання арифметичних і логічних операцій обчислювальними пристроями, основи синтезу комбінаційних схем і мікропрограмних автоматів, що розглядаються з погляду теорії цифрових автоматів.

Курс умовно роздільний на дві частини:

- частина 1 - «Комп'ютерна арифметика»;
- частина 2 - «Логічні основи і синтез цифрових автоматів».

У першій частині курсу вивчаються форми і методи представлення чисел в сучасних КС, алгоритми перетворення чисел різних систем числення, методи виконання основних арифметичних операцій в різних системах числення, методи виконання арифметичних операцій в бінарно- кодових і спеціальних системах численнях, структурні і операційні схеми основних арифметичних пристроїв.

У другій частині курсу вивчаються логічні основи цифрових автоматів, математичний апарат булевої алгебри, методи мінімізації булевих рівнянь, описи цифрових автоматів (ЦА), методи синтезу і аналізу ЦА (в т.ч. канонічні) на логічних елементах різних базисів, графи і граф схеми автоматів (ГСА). Контроль правильності виконання арифметичних і логічних операцій пристроями. Студенти повинні отримати практичні навики проектування елементарних, логічних і принципових електричних схем типових комбінаційних цифрових автоматів і автоматів з пам'яттю.

1.2 Загальні визначення і математична модель цифрових автоматів

У загальному випадку, всі електронні пристрої, у тому числі і ЕОМ, ділять на аналогові, цифрові і змішані.

Під комп'ютером розуміють цифрову обчислювальну машину (ЦОМ), що представлена комплексом технічних і програмних засобів, об'єднаних загальним управлінням і

призначених для введення інформації, її переробки за заданим алгоритмом програми і виведення результатів для користувача або інших пристроїв.

При роботі комп'ютер виконує перетворення початкової інформації, при цьому вигляд інформації (тобто як вона надана, закодована) визначає велику роль у формуванні алгоритмів арифметичних і логічних операцій.

Інформація є однією з основних філософських категорій і понять нашого миру (поруч з такими поняттями як матерія, енергія і ін.). Кібернетика, як наука про автоматизацію процесів управління базується на теоретичних основах інформації.

У словнику С.І. Ожегова наведено: інформація – відомості про навколишній світ і процеси, що протікають в ньому, сприймані людиною або спеціальними пристроями.

Комітет науково-технічної термінології Академії наук дає наступне визначення: інформація – це відомості, зберігання, що є об'єктом передачі і перетворення [1-21]. Можна зупинитися на такому визначенні: інформація – це відомості про яку-небудь подію або предмет, що поступає до одержувача ззовні в результаті його взаємодії з навколишнім середовищем [1-21].

Як усяка категорія інформація характеризується наступними властивостями [1-21]:

- інформація приносить знання про навколишній світ, яких в даному місці не було до її отримання;
- інформація не матеріальна, але вона виявляється у формі матеріальних носіїв (дискретних знаків, символів, сигналів і ін.);
- інформація може бути поміщена як в знаках, як таких, так і в їх взаєм-ному розташуванні (наприклад: знаки у вигляді літер "Т, Р, С, О" можуть принести інформацію у вигляді різних слів: сорт, торс, тор, трос, ін.);
- знаки і сигнали несуть інформацію тільки для одержувача, що здатен розпізнати їх.

Знаками назвемо реальні помітні одержувачем матеріальні об'єкти: літери, цифри, символи, предмети, умовні рухи, їх комбінації [1-21].**Сигналами** називатимемо динамічні процеси будь-якої природи, що змінюються в часі (зміна значення напруги, тиску, електромагнітного поля і ін.). Поняття "знак" і "сигнал" часто

взаємозамінні, проте знаки частіше використовують для зберігання інформації, а сигнали для передачі повідомлень з однієї точки простору в іншу або перетворення інформації з однієї форми в іншу.

Із знаків і сигналів будуються **повідомлення**. Елементарним повідомленням є кожний із знаків або сигналів. Послідовність елементарних повідомлень несуть одержувачеві інформацію. Кінцеву кількість усіх знаків або сигналів називають ще **алфавітом.** При передачі повідомлень знаки перетворюють на сигнали, що однозначно зіставляються - **"коди".** Процес зіставлення називають - **кодуванням.** Коди бувають різними, але кожен відповідає своїм правилам (алгоритмам) перетворення. Обернений процес називають **декодуванням**.

Розрізняють інформацію безперервну і дискретну.

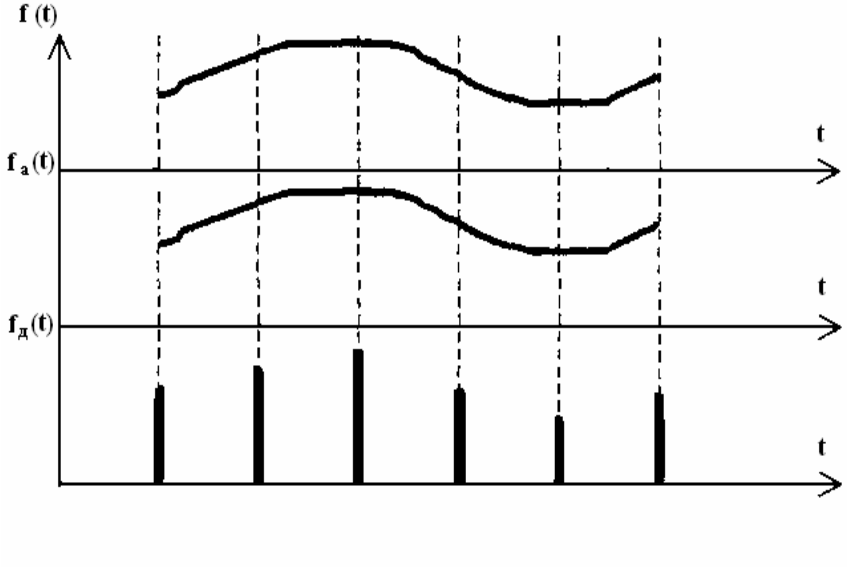


Рисунок 1.1 – Інформація функції f(t) представлена в безперервному (аналоговому) і дискретному вигляді

У аналоговому уявленні функція має значення параметра в будь-якій точці аргументу *t*. У дискретному вигляді функція *f*(*t*) визначена лише на конкретних значеннях *t*.

Вид інформації, що переробляється, впливає на структуру ОМ і принцип їх дії. Як наголошувалося раніше, ОМ ділять на три основні класи: аналогові, цифрові, гібридні.

Аналогова ОМ – та, що оперує інформацією, представленою у вигляді безперервних змін деяких фізичних величин (струм, напруга, опір і ін.). Багато процесів в природі можна моделювати аналогічно при однакових математичних моделях їх опису.

Цифрова ОМ – та, що оперує інформацією, представленою в дискретному вигляді. Вона більш універсальна оскільки методи чисельного рішення дають можливість рішення будь-яких математичних і логічних задач.

Останніми роками при розробці деяких ОМ вчені суміщають пози-тивні сторони ЦОМ і АОМ. Такі машини отримали назву гібридних ОМ.

Ми розглядатимемо арифметичні і логічні основи ЕОМ з погляду теорії **цифрових автоматів**. Відомий математик Джон фон Нейман у 1945р. у своїй доповіді описав, як повинен бути влаштований комп'ютер. У спрощеному вигляді, він повинен мати наступну структуру (рис 1.2).

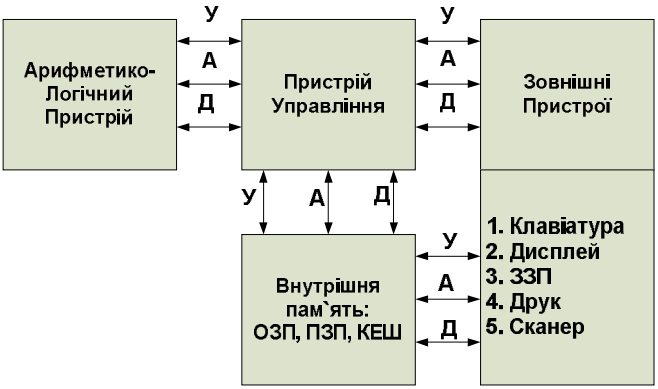


Рисунок 1.2 – Структура ЕОМ

В неї входять: арифметичний логічний пристрій (АЛП); пристрій керування (ПК), який організує процес виконання програм; внутрішні пристрої, що запам'ятовують інформацію (оперативний запам'ятовуючий пристрій – ОЗП, постійний запам'ятовуючий пристрій – ПЗП), для зберігання даних, проміжних рішень, програм і др.; зовнішні пристрої для введення/виведення інформації і її

тривалого зберігання, архівації – зовнішній запам'ятовуючий пристрій (ЗЗП).

Нас цікавлять теоретичні основи методів виконання арифметичних і логічних операцій АЛП. Загалом, будь-який пристрій ЕОМ можна представити як цифровий мікропрограмний автомат.

Під цифровим автоматом (ЦА) розуміють пристрій, що оперує з цифровою дискретною інформацією і характеризується кінцевою множиною внутрішніх станів $Z=\{z_0,z_1,z_2,\ldots,z_n\}$, в які він переходить під впливом множин сигналів: вхідних $X=\{x_1,x_2,...,x_k\}$ і вихідних $Y=\{y_1,y_2,...,y_m\}$, при цьому є кінцева множина правил переходу з одного стану в інший: $W=\{w_1,w_2,...,w_L\}$ - функція переходів ЦА із стану Z_{i-1} у Z_i по правилам формування Z_i, станів ЦА; а також функція $\lambda=\{\lambda_1,\lambda_2...,\lambda_j\}$ формування Y- виходів ЦА, що в загальному випадку може бути пов'язана з його станом Z_i і вхідними сигналами $Y=\lambda[Z_i, X_i]$). Всі множини є кінцеві, звідси кінцевий ЦА.

Математичною моделлю ЦА є деякий абстрактний автомат, заданий таким чином: у початковий момент t_0 автомат знаходиться в стані Z_0. Під впливом вхідного коду X з множини вхідних слів, автомат переходить зі стану Z_0 в Z_1, при цьому може виникнути вихідне слово $Y=\lambda[Z_i,X_i]$, а перехід $Z_i=W[Z_0,X_i]$ здійснюється функцією переходів W, дії вхідного коду X і попереднього стану $Z_{0\ (або\ n)}$ і так далі (алгоритм перетворення).

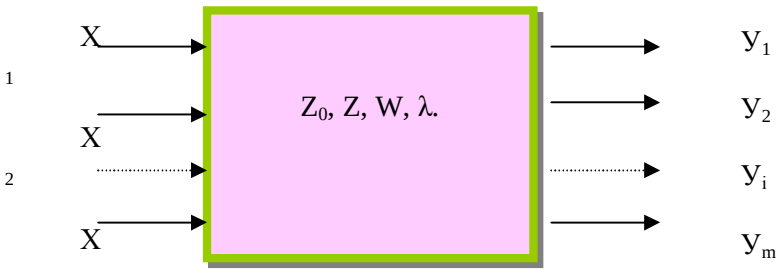


Рисунок 1.3 – Модель абстрактного автомата

У загальному випадку математична модель абстрактного автомата записується як кортеж (сукупність): $A=\{X, Z_0, Z, W, \lambda, Y\}$.

Алгоритм – кінцева послідовність точно сформульованих правил рішення якоїсь задачі (узбецький математик Аль-Хорезмі в IX столітті сформулював правила арифметичних дій). Можуть бути

словесними, математичними, заданий програмою, ін.. Бувають детерміновані алгоритми, випадкові алгоритми, чисельні алгоритми, логічні алгоритми.

Арифметико-логічний пристрій (АЛП) – функціональний пристрій ЕОМ, що виконує арифметичні і логічні дії, необхідні для обчислення або виконання логічних операцій над заданою в ОЗП інформацією відповідно до встановлених алгоритмів.

АЛП характеризується:

- часом виконання елементарних операцій або середньою швидкодією, тобто кількістю арифметичних або логічних операцій виконаних в одиницю часу (секунду);
- набором арифметичних дій і логічних команд, які він виконує;
- видом арифметичного базису;
- тактовою частотою (часова сітка) виконання операцій, її швидкодія.

Наприклад, частота 100 МГц позначає, що виконується 100 мл. тактів в секунду (якщо операція виконується за два такти, то швидкодія буде 50 мл. операцій в секунду).

Одиницею інформації в комп'ютері є один біт, тобто двійковий розряд, який може приймати значення 0 або 1. Вісім послідовних або паралельних бітів складають байт. У одному байті кодується значення одного повідомлення з 256 можливих. Продуктивність АЛП процесора додатково характеризується наступними параметрами: внутрішньою і зовнішньою розрядністю даних, що обробляються (чим вище розрядність, тим вище інформаційна продуктивність одночасно обробляємої інформації); об'ємами пам'яті, адресацією центрального процесорного пристрою CPU (central processor unit); ступенем інтеграції транзисторів, ін..

2 СИСТЕМИ ЧИСЛЕННЯ

2.1 Огляд деяких систем числення

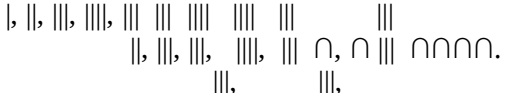
Сам процес числення (нумерація) – сукупність певних прийомів (правил, алгоритмів) представлення натуральних чисел. Систем числення існує багато.

У будь-якій системі числення прийняті деякі символи (знаки, слова) для позначення певних чисел, ці знаки називають **базовими**

(вузловими). Наприклад, в 10-ій системі базовими є знаки (зображення) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, що позначають цілі натуральні числа від 0 до 9 і дроби за допомогою коми (крапки).

Решта чисел уявного діапазону числової вісі мають назву алгоритмічні і створюються в результаті виконання дій за певними правилами над базовими числами (10, 123, 543). Тому системи числення відрізняються одна від одної вибором базових (вузлових) чисел і способами утворення алгоритмічних. При письмовому позначенні вони відрізняються характером використаних числових знаків і принципами їх запису. Наприклад, у стародавніх вавілонян базовими були числа 1,10,60; у племені маорі (Нова Зеландія) – 1, 11, 112, 113; у римській системі числення вузловими були числа 1, 5, 10, 50, 100, 500, 1000, які позначалися знаками I, V, X, L, C, D, M: приклад MCMXCVII - 1997.

Системи числення, в яких алгоритмічні числа утворюються складанням вузлових, називаються адитивними. Наприклад, в стародавньоєгипетській системі числення числа 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 19, 40 позначалися відповідно:



У римській системі числення числа виходять шляхом складання і віднімання вузлових.

У сучасних числах (в т.ч. європейських, українській і російській мовах, ін.) застосований адитивно-мультиплікативний спосіб. В деяких країнах були прийняті алфавітні системи числення – наприклад, греки, стародавні слов'яни, ін. (цифри від 1 до 9 зображалися буквами алфавіту з межею зверху).

Будь-яка призначена для практичних розрахунків система числення повинна забезпечувати:

- однозначність відображення числа;
- представлення всіх чисел в діапазоні числової осі;
- простоту запису чисел і виконуваних операцій над ними;
- систему правил запису і читання чисел.

Системи числення діляться на позиційні і непозиційні.

Позиційна система числення - система, в якій значення базових знаків залежить від їх місцеположення в числі, розрядності. (Тобто 0…9 в розряді одиниць не дорівнює 0…9 в розряді десятків, сотень, тисяч, ін. – бо прийнято, що вага сусідніх розрядів таких чисел різниться на порядок).

Непозиційна система числення- система, в якій значення базових знаків не залежить від їх місцеположення в числі, розрядності.

Принцип побудови таких систем полягає у виборі базових знаків і при читанні (або запису) проводяться операції складання (іноді і віднімання). Наприклад: Робінзон Крузо застосував непозиційну систему числення для підраховування діб. Базовий знак – вертикальна риска, які він складав і отримував числа. Ця система неефективна з двох причин:

- запис числа може виявитися довгим і незручним;
- для виконання підрахунків необхідно використовувати іншу систему.

Проте вона дуже проста.

Римську систему числення можна назвати частково позиційною, хоча вона вважається непозиційною системою. Так, наприклад, в числах LX і XL знак X (10) приймає два значення ±10 і залежить від свого місцеположення відносно L (справа чи зліва).

2.2 Система залишкових класів

Однією з непозиційних систем числення, розробленою спеціально для застосування в спеціалізованих ЕОМ, є система в залишкових класах (СЗК).

Ідейне коріння СЗК пов’язане з працями Ейлера, Гауса і Чебишева з теорії порівнянь. Значний внесок в розвиток СЗК внесли чеські вчені математики М. Валах і А. Свобода, що працювали над представленням чисел у вигляді сукупності ненегативних вирахувань по групі взаємнопростих основ [1-21].

Хай необхідно записати число N, задане в 10-ій системі в системі СЗК.

Кінцеву множину позитивних цілих чисел, взаємно простих між со-бою, назвемо базисом основ системи числення в залишкових класах [14] $P = \{p_1, p_2, \ldots, p_n\}$.

Під системою числення в залишкових класах будемо розуміти таку систему, в якій ціле число представляється у вигляді набору залишків (вирахувань) по вибраних основах p_i.

Тоді число N в СЗК буде записано як:

$$N=\{a_1,a_2,...,a_n\}, \qquad (2.1)$$

$$\text{де } \alpha_з = \mathrm{N} - \left[\left[\frac{\mathrm{N}}{\mathrm{p_i}}\right]\right] \mathrm{p_i}, \qquad (2.2)$$

тобто a_i – є відображення числа N у вигляді позитивного залишку від ділення N націло на основу P_i.

При цьому буде завжди справедлива нерівність $a_i<p_i$. (де i=1,2…n).

У відомій з теорії чисел першій і другій функціональній теоремі Гауса, доведено, що якщо числа p_i взаємно прості між собою, то опис $\{a_1, a_2, a_n\}$, що представляє N, є єдиним.

Нагадаємо, що взаємно простими числами називають числа, що мають **найменший загальний дільник (НЗД)**, що дорівнює 1. Наприклад, 15 і 22 є взаємно прості числа (оскільки НЗД рівний 1).

Приклад. Нехай вибрані основи СЗК: p={3,5,7}. Представимо десяткове число N=67 у СЗК.

Рішення. Знайдемо залишки згідно виразу 2.2

$$a_1 = 67-\left[\frac{67}{3}\right]\cdot 3 = 67-66 = 1;$$

$$a_2 = 67-\left[\frac{67}{5}\right]\cdot 5 = 67-65 = 2;$$

$$a_3 = 67-\left[\frac{67}{7}\right]\cdot 7 = 67-63 = 4.$$

Отже, число 67 записується в СЗК як {1,2,4}, або в двійковій – {001,010,100}.

Діапазон чисел, що представляються, дорівнює:

$$R = p_1 \cdot p_2 \cdot p_3 \cdot ... \cdot p_n; \qquad (R = 3\cdot 5\cdot 7 = 105).$$

Зворотний переклад числа з СЗК в десяткову систему проводиться за формулою:

$$N = (a_1 \cdot B_1 + a_2 \cdot B_2 + ... + a_n \cdot B_n) - r \cdot R, \qquad (2.3)$$

де r – ранг, що приймає значення 0, 1, 2, … так, щоб права частина виразу 2.3 була менше R;

B_i – ортогональний базис, що визначає при виборі базису СЗК.

$$B_i = k_i \cdot \frac{R}{p_i},$$

де k_i – ціле позитивне число (k_i=1, 2, …, p_i–1), при цьому k_i вибирається таким, щоб **залишок від ділення B_i на p_i дорівнював одиниці** (тобто вага базису k_i вибирається виходячи з рівності):

$$B_i = \frac{m_i \cdot R}{p_i} = 1 = k_i \cdot p_i + 1, \qquad (2.4)$$

Приклад. Визначимо для СЗК з основами {3,5,7} ортогональні базиси B_i та переведемо число {1,2,4} в десяткову систему.

$$B_1 = \frac{k_1 \cdot 105}{3} = k_1 \cdot 35 = (2 \cdot 35); \qquad B_1 = 70;$$

(для p_1=3 умова 2.4 виконується при k=2),

$$B_2 = \frac{k_2 \cdot 105}{5} = (1 \cdot 21); \qquad B_2 = 21;$$

(для p_2 =5 умова 2.4 виконується при k=1),

$$B_3 = \frac{k_3 \cdot 105}{7} = (1 \cdot 15); \qquad B_3 = 15;$$

(для p_3 =7 умова 2.4 виконується при k=1), тоді N = $(a_1 \cdot B_1 + a_2 \cdot B_2 + a_3 \cdot B_3) - r \cdot R = (1 \cdot 70 + 2 \cdot 21 + 4 \cdot 15) - r \cdot 105 = 172 - 1 \cdot 105 = 67$.

Запишемо число 67 в двійковій системі 1000011 і в СЗК 001_ 010_ 100. Цифри СЗК в кожному розряді вирахувань по-перше, незалежні один від одного і не залежать від позиції, а лише залежать від базису; по-друге, кожен розряд вирахувань несе інформацію про число в цілому. Ця властивість СЗК використовується для відновлення спотвореної інформації при передачі по лінії зв'язку.

3 ПОЗИЦІЙНІ СИСТЕМИ ЧИСЛЕННЯ

3.1 Представлення чисел в позиційних системах числення

Система числення, в якій значення базисної цифри визначається її положенням в розрядах числа, називається позиційною. Наприклад, в десятковій системі числення в числі 222, перша цифра зліва від коми означає дві одиниці, друга – два десятки, третя – дві сотні.

Позиційна система числення визначається своєю основою.

Основа q позиційної системи числення - кількість знаків, що використовуються для відображення числа в даній системі.

Якщо q – ціле позитивне число, то позиційна система числення називається природна.

Позиційних систем числення може бути багато оскільки за основу можна прийняти будь-яке число і утворити нову систему числення.

Число в позиційній системі числення визначається рівністю:

$$A(q) = a_n q^n + a_{n-1} q^{n-1} + \ldots + a_0 q^0 + a_{-1} q^{-1} + \ldots + a_{-m} q^{-m},$$

або $$A(q) = \sum_{i=-m}^{i=n} a_i q^i,$$

де a_i – коефіцієнти розрядів в знаках системи; q – основа системи числення; $A(q)$ – довільне число; m, n – кількість дробових і цілих розрядів.

При записі числа, запис основи і знак "+" опускають, а ступінь визначають по розряду від коми,

тобто $A(q) = a_n a_{n-1} \ldots a_1 a_0, a_{-1} \ldots a_{-m}$.

Наприклад, число 934,125 записано в десятковій системі числення. Число $124{,}57_{(8)}$ записано у 8-й системі числення (беруться цифри 0, 1, ..., 7).

У двійковій системі числення використовують цифри 0, 1.

Наприклад, число $1001{,}1 = 1 \cdot 2^3 + 1 \cdot 2^0 + 1 \cdot 2^{-1} = 17{,}5$ (див. таблицю 3.1).

Вага розряду P_i числа в позиційній системі числення- це відношення $P_i = \frac{q^i}{q^0} = q^i$, де i – це номер розряду справа наліво.

Вага розрядів росте справа наліво. Якщо візьмемо розряд $P_i=10^K$, то наступний старший матиме вагу $P_{i+1}=10^{K+1}$, а молодший $P_{i-1}=10^{K-1}$. Такий взаємозв'язок розрядів припускає, при виконанні операцій, передачу інформації між ними.

Таблиця 3.1 – Еквіваленти десяткових чисел в різних системах числення

Десяткова цифра	q=2	q=3	q=5	q=8	q=16
0	0000	000	00	00	0
1	0001	001	01	01	1
2	0010	002	02	02	2
3	0011	010	03	03	3
4	0100	011	04	04	4
5	0101	012	10	05	5
6	0110	020	11	06	6
7	0111	021	12	07	7
8	1000	022	13	10	8
9	1001	100	14	11	9
10	1010	101	20	12	A
11	1011	102	21	13	B
12	1100	110	22	14	C
13	1101	111	23	15	D
14	1110	112	24	16	E
15	1111	120	30	17	F

Якщо в даному розряді накопичилося значення одиниць (десятків, сотень, і т.д.) рівне або більше q, то повинна відбуватися передача одиниці в сусідній, старший розряд. При складанні, такі передачі назвемо ***переносом одиниці в старший розряд***, а при відніманні – ***позикою одиниці із старшого розряду***. Приклад: 9 + 2 = 11; 17 – 8 = 9.

Глибина числа – це кількість знаків (зображень), які використано при ***записі числа в одному розряді.*** Поняття «глибина числа» співпадає з поняттям «основа» і дорівнює значенню q – тобто для 10-ї, 8-ї, 2-ї систем числення, відповідно: 0, 1, 2, …, 9 (десять); 0, 1, 2, …, 7 (вісім); 0, 1 (два).

Довжина числа – це кількість розрядів в записі числа. Часто вжива-ється термін – довжина розрядної сітки пристрою (кількість розрядів n).

Для різних систем характерна своя довжина розрядної сітки для запису одного і того ж числа. Наприклад, $96_{(10)} = 140_{(8)} = 10120_{(3)} = 1100000_{(2)}$. Чим менша основа, тим більше потрібно розрядів для запису числа і навпаки.

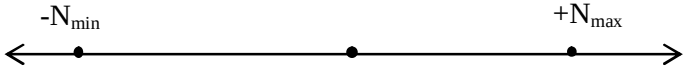


Рисунок 3.1 Числова вісь діапазону чисел, що представляються

Величина розрядної сітки n, що прийнята в ЦА, у свою чергу накладає обмеження на уявний діапазон чисел (яким він може оперувати) – це інтервал числової вісі між максимальним і мінімальним числом (рис. 3.1). У загальному випадку, максимальне (мінімальне) число уявного діапазону, залежить від основи системи числення і кількості виділених розрядів n: $N(q)_{max} = (q^n - 1)$, $N(q)_{min} = -(q^n - 1)$. Наприклад, три розряди десяткової системи забезпечать діапазон чисел (з урахуванням знаку) $N_{min} = -999$, $N_{max} = 999$.

3.2 Вибір системи числення комп'ютера

3.2.1 Переваги двійкової системи

Розробку комп'ютера починають з вибору системи числення, методів виконання арифметичних і логічних операцій, елементної бази. Ці питання є важливими, оскільки повинні забезпечити задану точність і швидкість обчислень, надійність в роботі, діапазон чисел.

Вибір системи числення у великій мірі обумовлюється наступними основними причинами:

1. Основу системи визначає число знаків (їх зображень) в одному розряді. Тут перевагу має двійкова система, а не десяткова оскільки вона вимагає всього два знаки (стани), а десяткова – десять.

Елементи, з десятьма станами (декатрони), що є в наш час, мають малу швидкість перемикання, громіздкі, мають невисоку надійність в роботі.

2. З одного боку система числень повинна забезпечити ***простоту арифметичних операцій***, а з іншого великий діапазон уявних чисел. Ці дві суперечності вирішуються на користь ***простоти арифметичних операцій*** при представленні чисел. Насправді, в двійковій системі не треба додаткових пристроїв представлення чисел. Сам сигнал несе інформацію про число і має двійкову природу: є сигнал або його немає - "1 " або "0". Крім того, майже всі елементи електронних схем (є струм/ напруга або ні) мають двійкову природу (реле, діод, транзистор, тригер, фотоприймач, стан оптоволокна, ін.).

Якщо прийняти показник ефективності системи з погляду витрат устаткування:

$$C = q \cdot n,$$

де q – основа (глибина) системи, n – довжина (розрядність) системи.

Розрядність n визначається з діапазону уявних чисел. Максимальне число з основою q: $N_{(q)\max} = q^n - 1$, звідси знайдемо n:

$$n = \log_q(N_{(q)\,max} + 1),$$

тоді витрати на устаткування підраховують за формулою:

$$C = q \log_q (N(q)_{\max} + 1).$$

Якщо за одиницю устаткування прийняти умовний елемент з одним стійким станом, то можна порівняти системи числення з погляду витрат.

Графік відносного показника приведений на рис. 3.2.

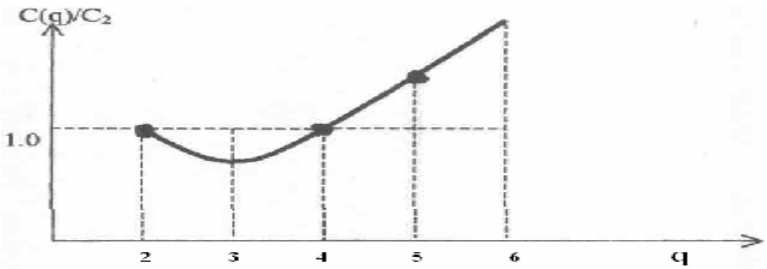


Рисунок 3.2 – Відносні витрати при різних значеннях основ

Наймінімальніші витрати будуть при основі e = 2,72, тобто, з погляду витрат, виходить найекономнішою буде система з q=3. Вона застосовується в ЕОМ "Сетунь". Проте, більшість ЕОМ використовують двійкову системи через простоту арифметичних операцій в ній.

Розглянемо це на прикладі складання в цій системі:

0+0=0, 1+0=1, 0+1=1, 1+1=10.

Складання цілих чисел без знаку представлено схемою (рис. 3.3).

Приклад. Знайти $C=A+B$. A=+1011 (+11); B=+1001 (+9); C=+10100 (+20).

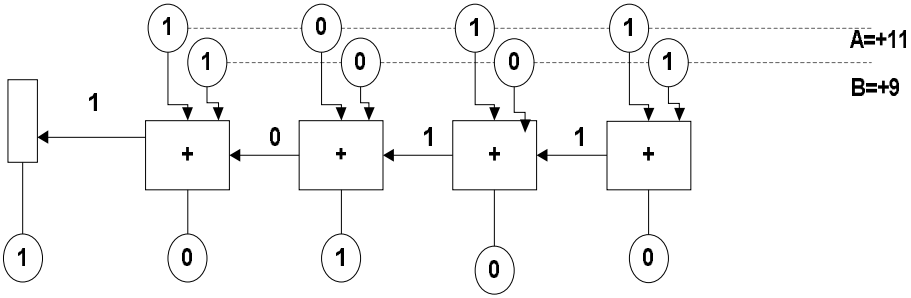


Рисунок 3.3 – Структурна схема бінарного (двійкового) суматора

Множення проводиться в двійковій системі ще простіше: 0·0=0, 0·1=0, 1·0=0, 1·1=1.

4 МЕТОДИ ПЕРЕКЛАДУ ЧИСЕЛ З ОДНІЄЇ ПОЗИЦІЙНОЇ СИСТЕМИ ЧИСЛЕННЯ В ІНШУ

Раніше ми відзначали, що будь-яке число можна представити поліномом з основою q_1, але це ж число можна представити іншим поліномом (див. рис. 3.3) з основою q_2, інакше: $N(q_1)= N(q_2)$. Представимо це для загального числа (що має цілу і дрібну частини – неправильний дріб) у наступному вигляді:

$$N(q_1)=a_n q_1^n + a_{n-1} q_1^{n-1} + \ldots + a_1 q_1^1 + a_0 q_1^0 + a_{-1} q_1^{-1} + \ldots + a_{-m} q_1^{-m} =$$

$$= b_k q_2^k + b_{k-1} q_2^{k-1} + \ldots + b_1 q_2^1 + b_0 q_2^0 + b_{-1} q_2^{-1} + \ldots + b_{-s} q_2^{-s} = N(q_2)$$

4.1 Методи перекладу цілих чисел

4.1.1 Метод підбору коефіцієнтів

Завдання перекладу числа з основою q_1 в число з основою q_2 зводиться до відшукування коефіцієнтів полінома нової основи. Цю задачу можна вирішити методом підбору коефіцієнтів полінома.

Правило. ***На початку беремо число з максимальною степінню основи, перевіряємо її вхід до даного числа так, щоб воно не перевищувало задане, потім підбираємо коефіцієнти менших степіней полінома, так щоб сума давала початкове число. Всі дії повинні виконуватися за правилами q_i-ї арифметики, тобто за правилами початкової системи числення.***

Наприклад. Перевести десяткове число 96 у трійкову систему.

Рішення. $96 = 0\cdot3^5+1\cdot3^4 + 0\cdot3^3+1\cdot3^2+2\cdot3^1 + 0\cdot3^0 = 010120_{(3)}$

Проведемо перевірку методом підстановки значень ваги розрядів, множення його на підібраний коефіцієнт і обчислення загальної суми.

$96=0\cdot243 + 1\cdot81 + 0\cdot27 + 1\cdot9 +2\cdot3 + 0\cdot1.$

Цей прийом застосовується тільки при "ручних" перекладах. Машинні алгоритми використовують метод ділення на основу нової системи числення.

4.1.2 Метод ділення на основу нової системи

Поліном цілого числа $N(q_1)$ можна записати за схемою Горнера в іншому вигляді:

$$N(q_2) = (...((b_k q_2 + b_{k-1})q_2 + b_{k-2})q_2 + ... + b_1)q_2 + b_0$$

де $b_0 \ldots b_k$ -коефіцієнти молодшого b_0, старшого b_k розрядів числа $N(q_2)$.

Якщо праву частину розділити на q_2, то отримаємо цілу частину (у лапках) і залишок з b_i ($0<i<k$). Повторюючи ділення k+1 разів отримуємо кожного разу залишки b_1, b_2, ..., $b_к$ і цілі частини. Останній залишок $b_к$ є старшим розрядом залишку числа, представленого в основі q_2 (неподільний залишок).

Приклад. $12_{(10)}$ перевести в бінарну (двійкову) систему числення.

Рішення. Ділимо на q=2 (у дужки поміщений залишок):

12:2=6(0)
6 : 2=3(0)
3 : 2=1(1)

Частка (1) менше основи 2, тому це теж залишок, причому старший, тому припиняємо ділення. Відповідь: 1100 ($12_{(10)}$).

4.1.3 Метод ділення на основу в будь-якій позитивній степіні

Попередній метод має один недолік. При великих числах, операція ділення має багато ітерацій. Це знижує швидкодію. Метод ділення на основу нової системи в будь-якому позитивному степіні аналогічний попередньому. Тут беруть для ділення число (з новою основою q_k) близьке до заданого числа $N(q_i)$, але що не перевищує його. Кожен залишок від ділення записують в двійкових розрядах, число яких рівне узятому степіні.

Наприклад. Перевести в бінарну систему числення число N=523 $_{(10)}$.

Рішення. Вибираємо, найближче до заданого, число 2^9 = 512 і ділимо:

523 : 512=1(залишок 11).

Отримали два залишки: старший – 1, молодший – 11. Кожний із залишків розписуємо в дев'яти бінарних розрядах: 1000000000 і 000001011. Потім сполучаємо (додаємо) записи (старші нулі можна не записувати) і отримуємо результат- число 1000001011.

В двійковій системі (для цілого числа) b_0 завжди «0» або «1». В першому випадку $N(q_2)$ – це парне число, в другому – непарне.

4.1.4 Метод віднімання найближчих, менших степінних ваг

Метод полягає в наступному. Вибирають значення найближчої, розрядної ваги бінарного (двійкового) числа, менше заданого.

Віднімаючи його від заданого числа, отримують залишок. У розряді вибраної ваги, ставиться 1. Потім порівнюють залишок з новим меншим ваговим розрядом. Якщо залишок менший, то в цьому ваговому розряді ставиться 0, якщо залишок більший, то в цьому розряді ставиться 1, а із залишку віднімається вага цього розряду. Виходить новий залишок, який знову порівнюється з наступними меншими вагами. Так продовжується до останнього (молодшого) вагового бінарного розряду і отримають число.

Приклад. Перевести десяткове число 1125 в двійкове. Метод – віднімання менших найближчих розрядних ваг. Найближчою меншою розрядною вагою буде число $1024=2^{10}$. Ставимо в десятому розряді 1 (зліва від коми, де міститься 2^{10}).

Віднімаємо 1125–1024 =101, де 101 – десятковийзалишок, який порівнюємо з числом $2^9= 512$. Залишок 101 менше 512, це означає, що на місці 2^9 ставимо 0. Порівнюємо наступну розрядну вагу $2^8= 256>101$. Знову на місці розрядної ваги 2^8 ставимо 0. Аналогічно буде і для розряду 2^7 – ставимо 0. Порівнюємо наступну розрядну вагу $2^6=64<101$. У цьому розряді ставимо 1 і віднімаючи 101–64=37 отримуємо новий залишок, який порівнюємо з розрядною вагою $2^5=32<37$. В цьому розряді ставимо 1, і віднімаючи 37–32=5, отримуємо новий залишок, який легко розписати в чотирьох розрядах, що залишилися: 0101. Таким чином, отримуємо число $10001100101_{(2)}=1125_{(10)}$. Перевіримо: 1024+64+32+4+1=1125.

Метод виключає громіздку операцію ділення і при невеликому досвіді легко виконується користувачем. Метод придатний для перекладу як цілої, так і дробової частини числа.

4.2 Переклад правильних дробів множенням на основу системи числення

Дробову частину числа можна записати в новій системі:

$$N(q_2) = b_{-1}q_2^{-1} + b_{-2}q_2^{-2} + \ldots + b_{-k}q_2^{-k}, \qquad (4.1)$$

цей вираз можна переписати по схемі Горнера:

$$N(q_2) = q_2^{-1}\left(b_{-1} + q_2^{-1}\left(b_{-2} + \ldots + q_2^{-1}\left(b_{-(k-1)} + q_2^{-1}b_{-k}\right)\right)\ldots\right).$$

Якщо праву частину множити послідовно на q_2, то знаходитимемо новий неправильний дріб, в цілій частині якої будуть числа b_{-1}, b_{-2}, ..., b_{-k}, при цьому всі дії повинні виконуватися за правилами q_1-арифметики, і отже в цілій частині дробів, що виходять, з'являтимуться еквіваленти чисел нової системи числення, записані в початковій системі числення.

Приклад: Перевести десятковий дріб 0,625 в двійкову систему числення. Рішення:

0,	625·2
1	250·2
0	500·2
1	000·2
0	000

Відповідь: N= 0,1010. Перевірити за формулою 4.1.

Правило: ***для перекладу правильного дробу (без цілої частини) необхідно, діючи в арифметиці початкової системи числення, помножити число, що переводиться, на основу нової системи, у результату відокремити цілу частина, а дробову частину, що залишилася, знову помножити на цю основу і так до отримання потрібного числа значущих цифр. Результат записувати як 0, ... і дробову частину у порядку отримання.***

Приклад. Перевести двійковий дріб 0,1101 в десятковий.

Рішення:

	0,	1101·1010
b_{-1}	1000,	0010·1010
b_{-2}	0001,	0100·1010
b_{-3}	0010,	1000·1010
b_{-4}	0101,	0000

Відповідь: N=0,8125.

При перекладі правильних дробів з однієї системи числення в іншу, може вийде дріб у вигляді нескінченного ряду або ряду, що розходиться. Тому, процес перекладу необхідно закінчувати:

- при появі в дробовій частині по всіх розрядах нулів;
- якщо буде досягнута задана точність дробу (тобто необхідне число роз рядів).

Приклад. Перевести дріб 0,543 з шісткової системи в трійкову.

Рішення:

0,	$543_{(6)} \cdot 3$
2,	513·3
2,	343·3
1,	513

Відповідь: $N = 0{,}543_{(6)} \sim 0{,}221_{(3)} = 0{,}532_{(6)}$

4.3 Переклад неправильних дробів

Правило. ***Для перекладу неправильного дробу (тобто такого дробу, що містить цілу частину) з однієї системи числення в іншу необхідно здійснити роздільно переклад її цілої і дробової частин, а результати записати послідовно, відокремивши цілу частина від дробової комою.*** Наприклад: $98{,}625_{(10)}$ дорівнює $1100010{,}1010_{(2)}$.

4.4 Переклад з 16-ої і 8-ої систем в 2-ву і навпаки

При перекладі чисел з десяткової системи в 2-ву, часто використовують проміжну 8-ву систему. Це дає економію в числі операцій.

Таблиця 4.1 – Перевод у бінарну систему

$q_2=8$		$q_2=2$		
Число	залишок	число	залишок	тріада
121	1	121	1	
15	7	60	0	1
1	1	30	0	
3 кроки		15	1	
		7	1	7
		3	1	
		1	1	1
		7 кроків		

Приклад. Перевести десяткове число 121 в двійкове (див. табл. 4.1) через проміжну 8-у систему. Для порівняння наведен переклад через основу q=2.

Правило. ***Щоб перевести число з 16-ї і 8-ї системи в 2-у необхідно кожну цифру числа, що переводиться, представити відповідно чотирирозрядним або трирозрядним двійковим кодом (тетрадами або тріадами), розташував їх на місцях (розрядах) цих цифр. Нулі в старших і молодших розрядах, що не змінюють значення числа можна опускати.***

Приклади: $171_{(8)}$ в двійкове N=001 111 001;
$753{,}335_{(8)}$ в двійкове
N=111 101 011,011 011 101.

Правило. ***Для перекладу чисел з двійкової системи числення в шістнадцяткову (вісімкову) слід двійкові цифри числа, що переводиться, згрупувати по чотири (три) в обидві сторони від коми (при необхідності неповні групи доповнити нулями), кожну групу двійкових цифр замінити відповідною їй цифрою в новій системі числення. Нові цифри розташувати на місцях замінюваних кодів.***

Приклад: $111111010{,}1100001001_{(2)}$ перевести в 16-у і 8-у системи числення.

а) 0001 1111 1010 , 1100 0010 0100$_{(2)}$
1 F A , 3 2 4$_{(16)}$
Відповідь: $1FA{,}C24_{(16)}$

б) 111 111 010, 110 000 100 100$_{(2)}$
7 7 2 , 6 0 4 4$_{(8)}$
Відповідь: $772{,}6044_{(8)}$

Узагальнюючи, робимо висновок: у якості проміжних систем числення, доцільно використати системи з основою $q=2^k$, для спрощення перетворення їх в двійкову систему і навпаки (де цифри замінюють їх еквівалентами).

4.5 Переклад двійково-десяткових систем числення

Двійкові розряди розбиваються на тетради з вагою розрядів наприклад, 8421, а в тетрадах записують цифри від 0 до 9 у двійково-десяткових кодах, тобто 1 треба ставити у тих розрядах, вага яких враховується в записуваній цифрі. Переклад з однієї системи до іншої проводять по тетрадах.

Приклад: $N=995_{(10)} \rightarrow 1001/1001/0101$.

4.6 Форма представлення чисел в комп'ютері

У десятковій системі існують багато форм представлення чисел. Ми, вміло користуючись цим, самі вибираємо ту або іншу форму представлення чисел. Наприклад, число 0,25 можна уявити, як 1/4, коли виконуємо операції з такими ж дробами. Можна уявити, як $25 \cdot 10^{-2}$ або $2500 \cdot 10^{-4}$ або $0{,}0025 \cdot 10^{2}$ або 0,3 при округленні і т.д.

Всю різноманітність запису чисел розбивають на природні і нормалізовані (нормальні) форми.

Нормалізованою формою запису будь-якого числа N звичайно називають математичний вираз типу $N=\pm 0{,}mxq^{(\pm n)}$, де $\pm m$ – мантиса (у вигляді правильного дробу зі значенням у старшому розряді, що не дорівнює «0»), q – основа, $\pm n$ – порядок числа.

При природній формі, число записують в натуральному вигляді, наприклад: для позитивних чисел: 125 (ціле число); 0,125 (правильний десятинний дріб); 125,125 (неправильний десятинний дріб). При норма-лізованій формі запису одне і те ж число може приймати різний вигляд і його форма залежить **від прийнятих правил (обмежень),** що діють при записі. Наприклад, 12500 може бути записано $12{,}5 \cdot 10^{3} = 0{,}125 \cdot 10^{5} = 125000 \cdot 10^{-1}$ і т.д.

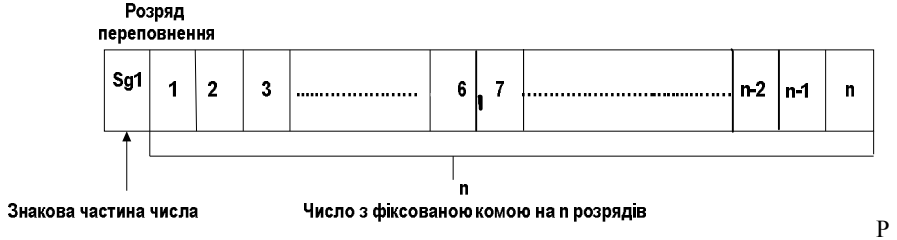
Така різноманітність форм представлення чисел є причиною ускладнення пристроїв і алгоритмів функціонування ЦА обчислювачів.

Щоб уникнути цього, в ЦА прийняті ***свої нормалізовані форми запису і відображення чисел***, що дозволяють мінімізувати вартість апаратних і алгоритмічних засобів.

5 ФОРМАТИ ПРЕДСТАВЛЕННЯ ЧИСЕЛ В КОМП'ЮТЕРІ

5.1 Представлення чисел з фіксованою комою (крапкою)

Автоматне зображення числа – це представлення числа N в розрядній сітці ЦА, в наперед заданому форматі і правилами відображення. На рис. 5.1 представлений звичайний машинний формат числа з фіксованою комою на n-розрядів (для знакової частини числа надано один додатковий розряд $Sg1$).



Р

исунок 5.1 – Формат машинного представлення числа з фіксованою комою (крапкою) – ФФК

При представленні числа у форматі з фіксованою комою (ФФК) в двійковій системі числення увесь формат n-розрядної сітки розбивається наперед на дві основні частини-знакову і числову:

- один (або два зліва) розряди для представлення знаку числа (0 або 00 плюс «+»; 1 або 11 мінус «–»);
- наступні k розрядів визначають для розміщення цілої частини числа;
-інші (n–2–k) розрядів відводять для розміщення дробової частини числа.

Дійсно, положення коми строго фіксоване в розрядній сітці ЦА.

Позначимо машинне зображення числа N через N_m, тоді число N_m дорівнює: $N_m = \dfrac{N}{K_\Phi}$,

де K_ϕ – масштабний коефіцієнт прийнятого формату, величина якого ***залежить від числа розрядів цілої частини***, тобто для рис 5.1 $K_\Phi=2^k=2^6$.

Приклад. Записати число N=+11,00111000111 (природний запис) у форматі машини з $K_\Phi=2^6$, кількість розрядів n=16. $N_m= N\cdot 2^{-6}$.

$[N]_m$=0/000011,001110001 (машинний запис числа N, останні два розряди 11 – втрачені із за відсутності знакомісць для їх розміщення).

Для сприйняття машинної форми числа N_m знакову його частину (крайні зліва один або два розряди) відокремлюють знаком «слеш» / чи \.

Навпаки, число N визначається як $N=N_m \cdot K_Ф$=+11,001110001. Знак «+» для позитивного числа (природний запис) не записується.

Діапазон уявних чисел залежить від обраного формату цілої частини і складає:

$$R=\pm 2^{k-1}+2^{k-2}+...+2^1+2^0,$$

де k – кількість розрядів числа.

При записі чисел у форматі з фіксованою комою, якщо число виходить за межі розрядної сітки числа, молодші розряди відкидаються. При цьому, можуть виникати значні погрішності, зокрема, ділення на «0», що приводить до невизначеності і необхідності втручання у розрахунки. Проте, при роботі ЕОМ у форматі з фіксованою комою швидкість виконання арифметичних і логічних операцій висока.

Помилка представлення чисел зменшується при правильному виборі (розрахунку) масштабних коефіцієнтів $K_Ф$. У деяких ЦА при переповненні розрядної сітки автомата, виводять один розряд «переповнення» із зупинкою розрахунків, що порушує його нормальне функціонування. (Наприклад, у ЄС ЕОМ ХХХХ АЛП виконуються операції окремо з цілими числами і дробами, а переповнення від складання дробів додається до цілої частини).

5.2 Представлення чисел у форматі з рухомою комою (ФРК)

Іншою найбільш поширеною нормальною формою є представлення чисел у формі з рухомою комою. В цьому випадку в нормальній формі число записується як

$$N=\left(\pm m\right)\left(\pm p\right),$$

N – нормалізоване число; m – мантисса, а p – характеристика (порядок) числа.

Таке уявлення, в загальному випадку, не є однозначним. Тому, для нормалізованого числа вводять, як правило, обмеження. Найбільш зручною і поширеною є вимога вигляду:

$$q^{-1} \le |m| < 1, \qquad (5.1)$$

де q – основа системи.

Правило. ***Нормалізованою формою представлення чисел є форма числа, для якої справедлива умова 5.1.***

В цьому випадку абсолютне значення мантиси може бути в межах від $0{,}q^{-1}$ до $(0{,}q^{-1}+0{,}q^{-2}+\ldots+0{,}q^{-n})$, тобто до 1 при $n\to\infty$, де n – кількість розрядів для запису мантиси без знаку.

Розташування коми у числі, мантису якого помножено на порядок, не є постійним. Тому і форма запису називається з рухомою комою (крапкою).

Формат машинного відображення чисел з рухомою комою представлений на рис. 5.2 для 16-и розрядної сітки.

16	15	14	13	12	11	10		8	7	6	5	4	3	2	1
Зн. мант.		Мантиса								Зн. поряд.		Порядок			

Рисунок 5.2 – Формат машинного представлення числа з рухомою комою (крапкою)

Формат уявлення цілої частини числа визначається знаком порядку і числом порядку і може змінюватися. Але завжди формат числа повинен містити: ***знак числа, мантису, знак порядку, порядок*** (де порядок визначає скільки розрядів мантиси необхідно відлічити вправо або вліво, щоб поставити кому, яка відокремлює дробову частину у кінцевому вигляді числа).

Отже, перш ніж записати число в ЦА з його розрядною сіткою, це число ***нормалізують***, тобто приводять до вигляду, коли в старшому розряді мантиси немає «нуля», а є «одиниця» (для двійкової системи числення). Решта всіх форм називається ***ненормалізованими***.

Порядок – це показник степіні для обранної системи числення, на який потрібно помножити (для позитивного знаку порядку) або розділити (для негативного знаку порядку) мантису для перетворення, тобто, $P=q^{\pm p}$,

де P – число, яке записане в двійкових розрядах порядку. При нормалізації число зрушують порозрядно вправо (якщо є ціла частина числа) або вліво (до одиниці після коми). При зрушенні вправо порядок (тобто степінь основи) збільшується з кожним зрушенням на 1. При зрушенні вліво порядок зменшується кожного разу на 1.

Приклади. Записати двійкове число в нормальній формі (звичайний машинний код-один розряд під код знаку):

+11101,011 (природна форма); 0\,11101011·2^{+5} (напівмашинна форма); 0\11101011\0\0101(звичайний машинний код).

Для прийнятого формату (рис.5.2) число запишеться у машинній формі (модифікований машинний код- два розряди під код знаку):

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	1
Зн. мант.		Мантиса								Зн. поряд.		Порядок			

При записі мантиси зручно щоб старший розряд був зліва, а в полі порядку розташування старшого розряду обумовлено коментарем (визначено для окремого ЦА особливо). Приклад:

+0,00011111 (природна форма) = 0\11111·2^{-3} (напівмашинна форма) = 0\11111000\1\000011 (звичайний машинний код, наявність коми припускається, але в ЦА вона за «слешем» відсутня).

Оскільки основа $2_{(10)}=10_{(2)}$ завжди постійна, то її запис в характеристиці числа опускається, а діапазон уявних чисел визначається розрядною сіткою.

Максимальне уявне число буде при позитивних знаках мантиси і порядку, і при максимальних значеннях їх чисел, іншими словами, при запису числа – по всіх розрядах одиниці. Швидкість обчислень у ФРК менша ніж у ФФК, а точність і діапазон – вища.

Мінімальне уявне число має негативні знаки мантиси і порядку, при мінімальному значенні мантиси і максимальному (тобто одиниці по всіх розрядах) порядку.

$$N_{\max} = \left(1 - 2^{-n}\right) \cdot 2^{\left(2^{m}-1\right)},$$

$$N_{\min} = -2^{-1} \cdot 2^{\left(2^{m}-1\right)}.$$

5.3 Погрішності представлення чисел

Представлення чисел в ЦА, як ми бачимо, завжди спричиняє за собою появу погрішностей в розрахунках, величина яких залежить від обмежень, що накладаються на автомат по розрядній сітці та формі представлення чисел.

5.3.1 Абсолютна похибка представлення чисел DN

Абсолютна похибка представлення чисел ΔN називається різниця між дійсним значенням числа N_i та його значенням N_m, отриманим після машинного відображення, операцій АЛП.

Приклад: двійкове число +11,00111000111 при n=15 і $K_Ф=2^k$, k=6, тоді N_m = 0\000011,001110001 [останні два розряди 11 – втрачені]. Перевівши в десяткові числа, отримаємо:

N =11,00111000111 (+3,2221678);

N_m =0\000011,001110001 (+3,2207031);

$\Delta N = N - N_m$ = +3,2221678– (+3,2207031)= +0,0014647.

Правило. ***Максимальна похибка*** **D*N*** ***для чисел формату з фіксованою комою не перевищує одиниці молодшого розряду сітки:***

$$\Delta N = 2^{[n-(k+1)]},$$

де n – розрядність автомата;

k – степінь масштабного коефіцієнта $K_Ф$;

N_m – значення машинного числа в двійковій системі після його переводу (тобто з N в N_m).

5.3.2 Відносна похибка представлення числа dN

Відносна похибка представлення числа dN. Це відношення абсолютної погрішності ΔN до числа N в процентному відношенні, тобто:

$$d N = \frac{\Delta N}{N} \cdot 100 \ \% \ .$$

Зазвичай N визначається математично після виконання арифметичних операцій в десятковій системі.

Після визначення абсолютної погрішності визначаємо відносну похибку в %. Приклад: з попереднього рішення ΔN=0,0014647, тоді

$$dN = \frac{0{,}0014647}{3{,}2221678} \cdot 100\% = 0{,}04546\,\%.$$

6 БІНАРНА (ДВІЙКОВА) АРИФМЕТИКА

У арифметиці будь-якого типу беруть участь завжди два або більше чисел. Як результат виконання арифметичних операцій з'являється нове число. Формально це можна представити у вигляді:

$$C = A \nabla B_0 \nabla B_1 \nabla ... \nabla B_n, \qquad (6.1)$$

де ∇ – знак будь-якої арифметичної дії з чотирьох: складання, віднімання, множення, ділення.

Будь-яке з чисел A, B_0, B_1, B_n є операндами, тобто числами, що беруть участь в арифметичних операціях, які виконуються ЦА.

Необхідно звернути увагу на те, що у виразі 6.1 опущено індекс m – машинне число. Але всі операнди і результат C є машинними числами. Правила виконання арифметичних операцій на рівні розрядів операндів:

$$A = a_1, a_2, a_3, ..., a_n,$$
$$B = b_1, b_2, b_3, ..., b_n,$$
$$C = c_1, c_2, c_3, ..., c_n,$$

де n–розрядна сітка автомата.

Алгоритми виконання арифметичних операцій в системі числення з основою q аналогічні алгоритмам в десятковій системі числення.

Результат виконання операції ∇ цифр a_i та b_i в i-м розряді представляється двома цифрами: цифрою C_i – результатом відповідної операції в даному розряді та цифрою $П_i$ – перенесенням в старший розряд або позики зі старшого розряду (при відніманні).

Результат C_i і перенесення $П_i$ формуються за наступними правилами:

Додавання:

$$c_i = \begin{cases} a_i + b_i, \text{ якщо } a_i + b_i < q; \\ a_i + b_i - q, \text{ якщо } a_i + b_i \geq q. \end{cases}$$

$$\ddot{I}_{\ з} = \begin{cases} 0, \ddot{y} \leftrightarrow a_i + b_i < q; \\ 1, \ddot{y} \leftrightarrow a_i + b_i > q. \end{cases}$$

Віднімання:

$$\tilde{n}_{з} = \begin{cases} \grave{a}_{з} - b_i, \text{ ÿ} \leftrightarrow a_i \geq b_i; \\ a_i - b_i + q, \text{ ÿ} \leftrightarrow a_i < b_i. \end{cases}$$

$$\ddot{I}_{з} = \begin{cases} 0, \text{ ÿ} \leftrightarrow a_i \geq b_i; \\ 1, \text{ ÿ} \leftrightarrow a_i < b_i. \end{cases}$$

Множення: $$c_i = \begin{cases} a_i \cdot b_i, \text{ ÿ} \leftrightarrow a_i \cdot b_i < q; \\ a_i \cdot b_i - \left[\dfrac{a_i \cdot b_i}{q} \right] \cdot q, \text{ ÿ} \leftrightarrow a_i \cdot b_i \geq q. \end{cases}$$

Виконання операції не може бути зведене до операції над окремими цифрами розрядів. Тому ділення виконується по аналогії з діленням в десятковій системі.

Двійковий напівсуматор (HS)- пристрій, що виконує арифметичну дію складання без урахування переносу $П_i$ з попереднього розряду, тобто на його вхід подаються тільки числа розрядів a_i та b_i без переносу $П_i$ (наприклад, наймолодші розряди операндів А і В).

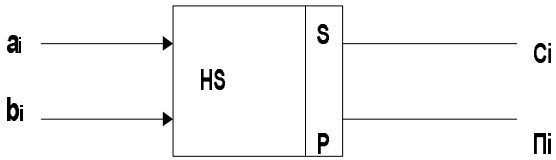


Рисунок 6.1 – Двійковий напівсуматор

Правила складання напівсуматора можна представити у вигляді таблиці 6.1, що має назву таблиця істинності (ТІ).

Таблиця 6.1 – Напівсуматор

a_i	b_i	c_i	$П_i$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Тут порозрядне складання виконується за формулою:

$a_i + b_i = C_i + П_i$,

де a_i, b_i – розряди чисел *A*, *B*;

C_i – сума *i*-го розряду;

$П_i$ – перенос у наступний розряд.

Але не враховувати переносів при складанні багаторозрядних чисел не можна. Тому, при складанні користуються пристроєм, що має назву двійковий суматор. Це пристрій, що виконує арифметичні дії складання з урахуванням ***переносу*** **(**$П_{i-1}$). від попереднього розряду і передачі ***переносу*** **(**$П_i$). в наступний розряд. Його робота і структурна схема була розглянута в розділі 3 (рис. 3.3).

ТІ двійкового суматора представлена в таблиці 6.2.

Таблиця 6.2 – Суматор

a_i	b_i	$П_{(i-1)}$	C_i	$П_i$
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Тут порозрядне складання виконується за формулою:

$a_i+b_i+П_{i-1}=C_i+П_i$,

де a_i, b_i – розряди чисел *A*, *B*;

C_i – сума *i*-го розряду;

$П_{i-1}$ – перенос з попереднього розряду;

$П_i$ – перенос у наступний розряд.

При виконанні арифметичної операції– віднімання в двійковій системі числення (в десятковій– аналогічно), нерідко проводиться ***позика***, рівно-сильна відніманню одиниці зі старшого розряду. Запозичення зі старшого розряду еквівалентно додаванню до молодшого розряду величини основи.

Крім того, цю позику необхідно враховувати при відніманні цифр наступного старшого розряду (тобто чи була позика в молодшому розряді). З урахуванням цього, і пам'ятаючи, що від'ємник

завжди повинен бути менше зменшуваного можна представити правила у вигляді таблиці 6.3.

Таблиця 6.3 – Від'ємник бінарних чисел

a_i	b_i	Z_i	C_i	$Z_{(i+1)}$
0	0	0	0	0
1	0	0	1	0
1	1	0	0	0
0	1	0	1	-1
0	0	-1	1	-1
1	0	-1	0	0
1	1	-1	1	-1
0	1	-1	0	-1

Порозрядне віднімання виконується за формулою:

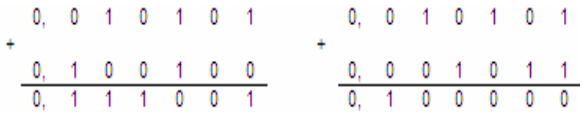
$$a_i - b_i + Z_i = C_i + Z_{i+1},$$

де a_i, b_i – операнди розряду;
C_i – різниця розряду;
Z_i – позика із молодшого i-го розряду;
Z_{i+1} – позика у старшого розряду.

Порівняння даних табл. 6.2 і 6.3 свідчать про те, що при відніманні операнду B від A порозрядно АЛП повинен мати додаткові розряди для зберігання знаку «–» для Z_i , $Z_{(i+1)}$ що ускладнює схему АЛП (при тому що a_i, b_i і C мають знак «+»). Тому перевагу надано двійковому суматору.

Розглянемо приклади на складання і віднімання для чисел з фіксованою комою (ФФК). Оперуватимемо з шестизначними, позитивними числами за модулем менше одиниці.

Приклади: а) виконати операцію складання бінарних чисел $+A$ і $+B$ (двох операндів A і B):



б) виконати операцію віднімання бінарних чисел $+C$ і $–D$ (двох операндів C і D):

```
  0,  1  1  1  1  1  0
-
  0,  0  1  1  1  1  1
  0,  0  1  1  1  1  1
```

в) віднімання в 8-ій системі:

```
  0,  7  5  7  4  7  3
-
  0,  2  7  3  6  5  4
-------------------------
  0,  4  6  3  6  1  7
```

г) віднімання в 3-ій системі:

```
  0,  2  2  2  2  2  2
-
  0,  1  0  2  2  1  0
-------------------------
  0,  1  2  0  0  1  2
```

д) виконати операцію множення бінарних чисел $+E$ і $+F$:

```
                        0,  1  0  0  1  0  1
                    *   0,  1  1  1  1  1  0
                    ---------------------------
                        0,  0  0  0  0  0  0
                  0,    1  0  0  1  0  1
               0,       1  0  0  1  0  1
            0,          1  0  0  1  0  1
         0,             1  0  0  1  0  1
      0,                1  0  0  1  0  1
   0,                   1  0  0  1  0  1
0,                      1  0  0  1  0  1
---------------------------------------------
0,  1  0  0  0  1  1  1  1  0  1  1  0
```

е) виконати ділення бінарних чисел $+G$ і $+H$ (у вигляді неправильного дробу $+G$ і цілого числа $+H$):

```
_1  1  0  1,  0  0  1  |  1  0  1  0
1  0  1  0             |  1  0  1  0,  1
0  0 _1  1  0  0       |
     1  0  1  0        |
  ------------------   |
     0  0  1  0  1  0  |
```

Проте, таке віднімання, ділення і множення для автоматів є складним оскільки при виконанні цих операцій в обчислювальних ЦА виникає проблема представлення негативних чисел. Для машинного представлення негативних чисел використовують коди**: прямий, обернений і доповняльний.**

Одним із способів виконання операцій за допомогою двійкового суматора, є заміна операції віднімання операцією суми із оберненим або доповняльним кодом негативного числа:

$$A - B = A + (-B) = A + B^{*}.$$

Заміна віднімання складанням ставить проблему представлення негативного числа в цифровому форматі. Це може бути вирішено,

наприклад, за допомогою використання властивості **циклічності** чисел при їх переліку, рахунку, зображенні (**два**, **три**, …, **два**надцять, **три**надцять…).

7 КОДИ БІНАРНИХ ЧИСЕЛ

7.1 Прямий код числа

Один метод вже розглянуто. Він використовує **прямий код числа,** $N = -0, a_1 a_2 ... a_n$ (природний запис числа у вигляді правильного дробу), де для машинного його зображення $N_M = 1 \setminus a_1 a_2 ... a_n$ використовують 1 (або 11) для позначення знаку «–» негативного числа.

Для систем числення з іншою основою q знак «–» (мінус) машинного зображення числа кодується старшим зображенням (вісьмирічна 7, шістнадцятирічна F, т.д.).

Прямий код негативного числа є код, всі цифрові розряди якого залишаються незмінними, записаними в нормалізованій формі, а в знаковій частині числа записується "1"(або 11 для модифікованого коду). Наприклад: N = – 0,101110 то N_M = 1\101110.

Позитивне число в прямому коді не міняє свого зображення.

Прямий код позитивного числа є код, всі цифрові розряди якого залишаються незмінними (наприклад, записаними в нормалізованій формі), а в знаковій частині числа записується 0 (або 00 для модифікованного коду).

Приклад: N= +0,110101, то $N_{M\,пр}$= 0\,110101.

У прямому коді в числову сітку автомата можна записати

$$N_{M\max} = 0 \setminus 111...1.. = 1 - 2^{-n},$$

де n – кількість розрядів. Діапазон чисел прямого коду (для правильного дробу), лежить в межах, визначених виразом:

$$-(1 - 2^{-n}) \le N_M \le (1 - 2^{-n}), \qquad (7.1)$$

Але в для операндів A і B в прямому коді виконуються тільки два випадки арифметичні операцій (з чотирьох наявних), тобто:

$C_{пр}$= +$A_{пр}$+$B_{пр}$ та $C_{пр}$= -$A_{пр}$-$B_{пр}$ = - ($A_{пр}$+$B_{пр}$).

Іншим методом представлення негативних чисел є уявлення їх в оберненому або доповняльному коді.

7.2 Обернений код числа

Оберненим кодом числа $N_M=1\backslash a_1a_2\ldots a_n$ називається таке машинне зображення числа для якого $a_i=0$ якщо воно дорівнювало "1" і навпаки, $a_i=1$, якщо воно дорівнювало "0".

Інакше, оберненим кодом **двійкового** числа є **інверсне** зображення цього числа, тобто всі розряди початкового бінарного числа, приймають обернене значення.

Інверсія, інвертування *- процедура переходу до протилежного значення розрядів числа в двійковій системі: інверсія «0» дає «1»; інверсія «1» дає «0». Позначається знаком риски ¯(або хвил ї˜, ін.) над числом, його розрядами.*

Приклад: N=–101110 (природний запис), то $N^{м}_{об}$=1\010001 (машинний запис в оберненому звичайному коді).

Узагальнюючи правила утворення оберненого коду на всі основи систем числення можна вважати, що **обернений код негативного числа виходить при відніманні з $(q-1)$ цифр по кожному розряду числа за винятком знакових розрядів, які замінюються значенням $(q-1)$ тобто:**

$$N_{їð} = 0 \backslash a_1a_2\ldots a_n,$$

$$N_{їá} = (q-1) \backslash \overline{a_1}\,\overline{a_2}\ldots\overline{a_n},$$

$$\overline{a_i} = (q-1) - a_i.$$

Приклад: а) $-0{,}286357_{(10)}$(природний) ↔ $1\backslash{,}286357_{(10)пр}$ (машинний прямий) ↔ $9\backslash{,}713642_{(10)об}$(машинний обернений).

б) $-0{,}1010111101_{(2)пр}$↔ $1\backslash{,}0101000010_{(2)об}$

Особливо звернути увагу: **якщо в знаковому розряді машинного запису знаходиться $(q-1)$ - тобто число негативне, то всі цифри числа, виключаючи знаковий, замінюються вирахуванням з $(q-1)$ значення розряду;**

якщо в знаковому розряді знаходиться нуль - тобто число позитивне, то перетворення не проводяться.

Приклади: (записи наведено без слеша «\», іноді його може заміняти крапка, кома або інший знак):

$$-0.243476_{(10)пр} \rightarrow 9.756523_{(10)об}$$
$$1.0111000111_{(2)пр} \rightarrow 1.1000111000_{(2)об}$$
$$0.943890_{(10)пр} \rightarrow 0.943890_{(10)об}$$
$$0.1010110101_{(2)пр} \rightarrow 0.1010110101_{(2)об}$$
$$00.110000111_{(2)пр} \rightarrow 00.110000111_{(2)об}$$
$$11.110100000_{(2)пр} \rightarrow 11.001011111_{(2)об}$$

Перехід від оберненого коду до прямого проводиться за аналогічним правилом: із значення $(q-1)$ віднімається значення по кожному розряду, окрім знакових (що остаються незмінними).

Для бінарної системи числення (просто щасливий випадок), можна перейти до прямого коду простим інвертуванням розрядів оберненого коду, окрім розрядів знаків.

Приклад: $A_{об}$= 11.1100110
$A_{пр}$= 11.0011001

7.3 Доповняльний код числа

Доповняльний код числа N=–0,$a_1a_2..a_n$ – таке машинне уявлення, в якому число $N_m = 1 \setminus a_1 a_2 ... a_n a_{n-1}$ записується оберненим кодом $N_m = 1 \setminus \bar{a}_1 \bar{a}_2 ... \bar{a}_n (\bar{a}_{n-1} + 1)$ із збільшенням в молодшому розряді на +1.

Правило перекладу з прямого коду в додатній код наступне:

- якщо в знаковому розряді знаходиться $(q-1)$ - тобто число негативне, то всі цифри числа, окрім розрядів знаків, замінюються вирахуваннями з $(q-1)$ значення розряду, а потім до цифри останнього молодшого розряду додається одиниця;

- якщо в знаковому розряді знаходиться 0 (або 00) - тобто число позитивне, то перетворення цифр не відбувається.

Приклади:

а) $-243476_{(10)}$; $9\setminus243476_{(10)пр}$ = $9\setminus756523_{об}$ + 0000001 = $9\setminus756524_{(10)доп}$

б) $-0111000111_{(2)}$; $1\setminus0111000111_{(2)пр.}$; $1\setminus1000111000_{(2)об}$ + $0\setminus0000000001_{(2)}$= $1\setminus1000111001_{(2)доп}$

Перевірка:

$1\backslash 1000111001_{доп}$

$+\underline{1\backslash 0111000111_{пр}}$

$10.0000000000\ \ (2_{(10)})$

в) –0,101110 ; $1\backslash 101110_{пр}$; $1\backslash 010001_{об} + 0\backslash 000001 = 1\backslash 010010_{доп.}$

г) $+425736_{(10)}$; $0\backslash 425736_{(10)пр} = 0\backslash 425736_{(10)доп} = 0\backslash 425736_{(10)об}$

Таким чином, для позитивних чисел прямий, оберненій і доповняльний коди співпадають, для негативного числа вони різні. Для позитивного числа в розряді знаку завжди встановлюють 0 або 00, а для негативного 1 або 11 (для довільної основи *q* знак «-» має код (*q* - 1)).

Узагальнемо і отримуємо вираз перекладу чисел у доповняльний код: $N_{äîï} = \begin{cases} N, \text{ ÿêùî } N \ge 0; \\ q + (-N), \text{ ÿêùî } N < 0. \end{cases}$

Приклади:

а) $+0{,}275936_{(10)}$; $0\backslash{,}275936_{(10)\,пр} = 0\backslash{,}275936_{(10)доп}$ (бо позитивне).

б) $-0{,}275936_{(10)}$; $10{,}000000_{(10)} + (-0{,}275936_{(10)}) = 9\backslash{,}724064_{доп}$

Звідси $N_{доп} = q + (-N)$, тобто додатній код є математичним доповненням числа до основи системи числення.

Покажемо, що доповняльний код – це доповнення до *q*, тобто: $|A| + A_{äîï} = q$. Звідси, $A_{äîï} = q - |A|$ враховуючи це, будемо віднімати.

Приклад:

A = –0,1011; тоді $A_{доп}$ = _ 10,0000

$\underline{\quad 0{,}1011}$

1\,0101 Отримано доповняльний код.

Покажемо, що обернений код – це доповнення до $q - q^{-n}$, тобто:

$|A| + A_{îá} = q - q^{-n}$, якщо A негативне, то $A_{об} = q - q^{-n} - |A|$. Звідси, треба віднімати з q негативне число і молодшу 1 (тобто q^{-n}).

Для довільної основи q це означає, що обернений код числа $A_{об}$ порозрядно формується від`ємом значення $A_{пр}$ в і-му розряді від (q-1).

Максимальне додатне число, що представляється при цьому дорівнює $(1 - 2^{-n})$, або N_{max} = – 0,111...11 тоді додатній код буде $N_{дод}$ =1\00... 01. Якщо в наймолодший розряд числа N_{max} додати 1 то отримаємо: –1,000...00. Перетворивши це число в додатній код, отримаємо, $N_{äîä\ \min} = -1.000..0$ тобто: $-1 \le N_{дод} \le (1 - 2^{-n})$.

7.4 Складання чисел, представлених у формі з фіксованою комою, на двійковому суматорі прямого коду

Двійковим суматором прямого коду (ДСПК) є суматор, в якому відсутній ланцюг порозрядного перенесення між старшим цифровим і знаковим розрядами (рис. 7.3).

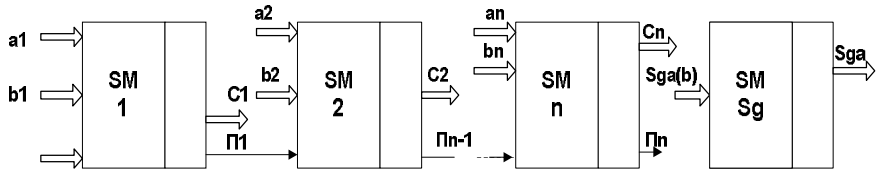


Рисунок 7.3 – Двійковий суматор прямого коду (ДСПК)

На ДСПК можна складати тільки числа, що мають однакові знаки, а результат складання < 1 (для правильних дробів) або менше виразу 2^n–1 (для цілого числа). Аналіз показує, що такий суматор виконує не всі операції складання, а лише дві з чотирьох: $A+B$ і $-(A+B)$. Сигнал пере-повнення $П_n$ старшого n-розряду служить лише для індикації факту, що воно було, покладаючи подальші турботи на користувача або ЦА, а знак суми C визначається за знаком будь-якого операнда, наприклад Sg_A (Sg_B).

Нехай задані операнди:

$$A_{їд} = Sg_A \backslash a_1 a_2 \ldots a_n$$

$$\hat{A}_{їд} = Sg_B \backslash b_1 b_2 \ldots b_n$$

де Sg_A і Sg_B – вміст знакових розрядів. Якщо Sg_A=Sg_B , то сума чисел матиме знак будь-якого з доданків, а цифрова її частина вийде порозрядним складанням операндів.

Приклади.

а) Скласти A = 0\1011 (+11); B = 0,0100 (+4). Тут Sg_A=0; Sg_B=0

+1011 (+11)

0100 (+4) C = 0\1111

0\1111 (+15)

б) Скласти A =–0\0101 (-5); B =–0\1001 (-9). Тут Sg_A=1; Sg_B=1

+0101 (–5)

1001 (–9) C = 1\1110

1\1110 (–14)

При складанні чисел на ДСПК можливі випадки, коли абсолютне значення суми операндів перевищує одиницю, тобто має місце переповнення розрядної сітки ЦА. Ознакою переповнення (γ=1) буде наявність одиниці перенесення $П_n$ із старшого розряду цифрової частини суматора. За цим сигналом повинні відбуватися автоматичний «останов» процесу рахування і коректування масштабних коефіцієнтів операндів A і B з розрахунком, щоб уникнути переповнення (наприклад зміна K_ϕ).

8 АЛГЕБРАЇЧНЕ СКЛАДАННЯ БІНАРНИХ ЧИСЕЛ

8.1 Складання чисел на двійковому суматорі доповняльного коду

Двійковим суматором доповняльного коду (ДСДК) називається суматор, що оперує числами, що представлені в доповняльному коді.

Основною особливістю ДСДК є наявність ланцюга перенесення ***одиниці переповнення*** $П_n$ зі старшого розряду цифрової частини в знаковий розряд (рис. 8.1) і відсутність зворотнього зв'язку перенесення ***одиниці переповнення*** $П_{Sg}$ із знакового розряду в наймолодший розряд числа.

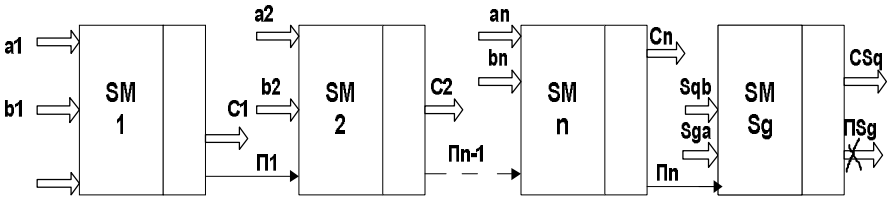


Рисунок 8.1 –Двійковий суматор доповняльного коду (ДСДК)

Для визначення правил складання чисел в ДСДК розглянемо **наступну теорему**: **якщо результат суми доповняльних кодів чисел негативний, то він представлений у доповняльному коді.**

При доведенні теореми, припускаємо, що числа представлені у формі з фіксованою комою, що стоїть перед старшим розрядом. Розглянемо наступні випадки (всі числа є цілими):

Випадок 1. $A>0$, $B>0$, а $A+B<1$ числа позитивні і немає переповнення. Для цього випадку $A_{доп}=A$, $B_{доп}=B$, то

$A_{доп}+B_{доп}=A+B=[A+B]_{доп}$ – результат позитивний. Покажемо це на прикладі: A = +0,1010; B = +0,0100.

$$\begin{array}{rl} + & \grave{A}äîï = 0\backslash,1010 \\ & Bäîï = 0\backslash,0100 \\ \hline & Cïð = 0\backslash,1110 \end{array}$$ C=+0,1110.

Випадок 2. A<0, B>0, а $|A|>B$. Для цього випадку $A_{доп}=q+|-A|$, $B_{доп}=B$; тоді $A_{доп}+B_{доп}=q+|A|+B=[A+B]_{доп}$ – результат негативний.

A=–0,1011; B=+0,0100

$$\begin{array}{l} + A_{äîï} = 1\backslash,0101 \\ B_{äîï} = 0\backslash,0100 \\ \tilde{N}_{äîï} = \overline{1\backslash,1001} \qquad C_{ïð} = 1\backslash,0111 \end{array}$$ C= - 0,0111.

Отриманий результат негативний, це означає, що він представлений у доповняльному коді, і його необхідно перевести в прямий, щоб отримати дійсний результат. Переклад числа з доповняльного коду в прямий здійснюємо за наступним алгоритмом: усі розряди числа (окрім знакових) інвертуються (тобто береться обернений код результату) і в молодший розряд додається 1. Знак суми C зберігається.

Випадок 3. A<0, B>0, а $|A|<B$. для цього випадку $A_{доп}=q+|A|$, $B_{доп}=B$; тоді $A_{доп}+B_{доп}=q+|A|+B$. Оскільки значення цієї суми більше q, то з'являється одиниця перенесення в знаковий розряд, що дорівнює вилученню з суми q одиниці, тобто результат: $A_{доп}+B_{доп}=A+B=C_{пр}$.

Приклад. А =–0,0100; В =+0,1011.

$$\begin{array}{l} + A_{äîï} = 1\backslash,1100 \\ B_{äîï} = 0\backslash,1011 \\ \hline C_{ïð} = 0\backslash,0111 \end{array}$$ C= + 0,0111.

Випадок 4. A<0, B<0, а $|A+B|<1$. Для цього випадку $A_{доп}=q+(-A)$; $B_{доп}= q+(-B)$, тоді $A_{доп}+B_{доп}=q+(-A)+q+(-B)=[A+B]_{доп}$ – результат негативний і з'явиться одиниця перенесення із знакового розряду.

Приклад. А = –0,0100; В = –0,1011.

$A_{äîï} = 1\backslash, 1100$

$\hat{A}_{äîï} = 1\backslash, 0101$

$\tilde{N}_{äîï} = 1\backslash, 0001$

$C_{пр}$=1\,1111. C= -0,1111

Висновок. Теорема справедлива для всіх випадків, в яких не виникає переповнення розрядної сітки, що дозволяє складати в ДСДК машинні зображення чисел за правилами двійкової арифметики. Приклади:

Випадок 1. A>0, B>0, C<1.

а) $A_{(10)} = +0{,}372914_{(10)пр}$; $B_{(10)} = +0{,}564019_{пр}$

$A_{доп} = 0\backslash{,}372914$

$+B_{доп} = 0\backslash{,}564019$

$C_{доп} = 0\backslash{,}936933 = C_{пр}$

б) Для цілих чисел: $A_{пр}$= 0\1001110101 (+629) $B_{пр}$=0\0101110011(+371)

$A_{доп} =$ 0\1001110101

$+ B_{доп} =$ 0\0101110011

$C_{доп} =$ 0\1111101000 $= C_{пр}$ $1000_{(10)}$

Перевірка: 629+371=1000

Випадок 2. A>0, B>0, C≥1 (переповнення розрядової сітки):

$A_{пр}$=0\672914$_{(10)}$, $B_{пр}$=0\564019$_{(10)}$.

$A_{доп}$ = 0\672914

$+B_{доп}$= 0\564019

$C_{доп}$=1\236933→0\1236933 (проведено зсув вправо мантиси на один розряд та відновлено знак суми)

Поява в знаковому розряді одиниці позитивного переповнення говорить про переривання роботи для зміни $K_Ф$ або зсув мантиси вправо на один розряд, а до порядку результату додається одиниця (тобто проводиться нормалізація результату).

$A_{пр}$ = 0\1101110101 $B_{пр}$ = 0\0101110011

$A_{доп}$ = 0\1101110101

$+ B_{доп}$ = 0\0101110011

$C_{доп}$ = 1\0011101000 (далі зсув вправо на один розряд)

$C_{пр}$ = 0\10011101000.

Перевірка: +885 + 371 =1256

Відбулося переповнення (g=1), тобто в знаковому розряді Sg_{C0} з'явилася ознака (1). Що це? ЦА не може розібрати ситуацію, це знак «мінус» суми C або ознака переповнення. Для вирішення цього треба в поле знаків додати ще один розряд (використовується ***модифікований*** доповняльний код, див. гл. 9).

Зсув вправо (вліво) числа зменшує (збільшує) його на порядок: тобто змінеться $K_ф$ (для ФФК) або порядок P (для ФРК) і цю зміну треба відкорегувати, щоб значення числа не змінилось.

Випадок 3. A>0, B<0, $|A|>|B|$: тобто $C = A+q+B$ оскільки $C>q$, то q повинне бути відібране, що робиться шляхом відкидання сигналу перенесення, що виникає в знаковому розряді.

$A_{(10)пр}$=+0.732904; $B_{(10)пр}$= -0.042703(зрівняно кількість розрядів A і B), $B_{(10)доп}$=9\,957297

$[A]_{доп}$ = 0\,732904

\+ $[B]_{доп}$ = 9\,957297

$[C]_{доп}$ =**1**0\,690201; $C_{пр}$ = +0,690201. При цьому в ДСДК переповнення ***1*** розряду SgC не використовується (відкидається).

При визначенні доповняльного коду $B_{(10)доп}$, спочатку провели вирівнювання розрядності чисел, додавши 0 в старший розряд числа $B_{пр}$, а потім віднімали з (q–1)=9 значення по кожному розряду і отримали обернений код, потім до його молодшого розряду додали 1.

Розглянемо ще приклад:

$A_{(2)пр}$=0\1101110101 (+885); $B_{(2)пр}$ =1\0110110101 (–437).

\+ $A_{доп}$ = 0\1101110101

$B_{доп}$ = 1\1001001011

$C_{доп}$=**1**0\0111000000

$C_{пр}$ = 0\0111000000 (+448)

Одиниця переповнення ***1*** відкидається (бо ДСДК), знак результату C позитивний і означає прямий код.

Випадок 4. $A>0$, $B<0$, $|A|<|B|$, $C = A+q+B$, оскільки $C<0$, то $C=q+(A+B)$.

а) $A_{(10)пр}$ = +0,324761 $B_{(10)пр}$ = -0,560129

+ $A_{доп}$ = 0\,324761

$B_{доп}$ = 9\,439871

$C_{доп}$ = 9\,764632 ; $C_{пр}$ = 9\,235368

б) $A_{(2)пр}$ = 0\0110011001 (+409); $B_{(2)пр}$ = 1\1010011101 (–669).

+ $A_{доп}$= 0\0110011001

$B_{доп}$ = 1\0101100011

$C_{доп}$ = 1\1011111100 – Необхідно перетворення в прямий код, тому що результат суми від'ємний.

$C_{об}$ =1\0100000011 – Інверсний (обернений) код результату.

Далі в молодший розряд додаємо 1, тобто +0\0000000001. Тоді $C_{пр}$=1\0100000100 = $-260_{(10)}$; (Формат числа надан в ФФЗ. При представленні його в ФРК необхідна буде нормалізація $C_{пр}$: представлення мантиси у вигляді правильного дробу з зсувом вліво на 10розрядів і відповідним корегуванням значення порядку Р на +10, а потім зсув вліво на 1 розряд, бо маємо 0 після коми, а далі – корекція порядку на -1). $C_{пр}$ = 1\100000100. Перевірка: –669+409 =–260

Випадок 5. $A<0$, $B<0$, $|A|+|B|<1$, $C=q+A+q+B<0$. При цьому, одна основа повинна бути відкинута за рахунок втрати сигналу перенесення в знаковому розряді. Тоді $C=q+(A+B)$.

$A_{(10)пр}$ = -0,432096 $B_{(10)пр}$ = -0,392671

+ $A_{доп}$ = 9\,567904

$B_{доп}$ = 9\,607329

$C_{доп}$ =***1***9\,175233 $C_{пр}$ = 9\,824767

Переповнення ***1*** в знаковому розряді пропадає, оскільки використовується

Випадок 6. $A<0$, $B<0$, але $|A|+|B|>0$.

Це також випадок переповнення розрядної сітки, але на відміну від випадку 2 переповнення тут негативне.

$A_{(10)пр}$ = $-0,610892_{(10)}$; $B_{(10)пр}$ = $-0,843507_{(10)}$

$A_{доп}$ = 9\,389108

+ $B_{доп}$ = 9\,156493

$C_{доп}$ =18\,545601

Поява цифри 8 в знаковому розряді говорить про те, що відбулося переповнення Sg_{C0}, необхідний зсув вправо з відновленням знаку і перетворення результату в прямий код.

$C_{доп}$=18\,545601; після зсуву $C_{доп}$=9\,8545601 (проведено зсув на 1 розряд вправо і відновлення знаку), $C_{пр}$=9\,1454399.

Розглянемо приклад в двійковій системі числення.

$A_{пр}$ = -1101110100; $B_{пр}$ = -1110111010

$A_{доп}$ = 1\0010001100

\+ $B_{доп}$ = 1\0001000110

$C_{доп}$ =10\0011010010

Було переповнення. Необхідний зсув вправо, для відновлення знаку з подальшим перетворенням результату в прямий код.

$C_{доп}$ =10\0011010010=1\00011010010;

$C_{пр}$=1\11100101110. Перевірка: –884–954= –1838.

У звичайному доповняльному коді є проблема із зображенням числа «–1». Тут розглядають два варіанти:

1. Виробляти сигнал переповнення, якщо в знаковому розряді (q–1), а в цифрових розрядах є нулі тобто $A_{доп}$= 1.0000…0.

2. Вважати –1 допустимим значенням, але при цьому її зображення в прямому коді співпадатиме із зображенням у доповняльному коді.

$-1_{(10)доп}$ = 9.00000…0 = 9.99999…9 + 0.0000…01 = 9.00000…0 = $-1_{(10)пр}$

$-1_{(2)доп}$ = 1.00000…0 = 1.11111…1 + 0.0000…01 = $-1_{(2)пр}$

Щоб відрізняти –1 від +1, використовують ***модифікований доповняльний код*** (див. розділ 9).

8.2 Складання чисел на суматорі оберненого коду

Двійковим суматором оберненого коду (ДСОК) є суматор, що оперує з числами в оберненому коді.

Структурна схема ДСОК приведена на рис. 8.2.

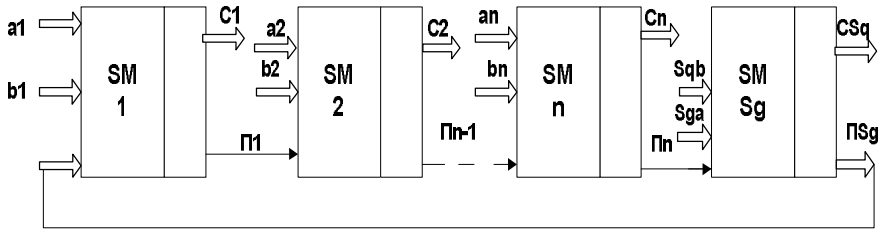


Рисунок 8.2 – Двійковий суматор оберненого коду (ДСОК)

Основною особливістю ДСОК є наявність ланцюга перенесення ***одиниці переповнення*** $\boldsymbol{\Pi_n}$ із старшого розряду цифрової частини в знаковий розряд (рис. 8.2) і наявність зворотнього зв'язку перенесення ***одиниці пере-повнення*** $\boldsymbol{\Pi_{Sg}}$ із старшого знакового розряду $\boldsymbol{Sg_1}$ в молодший розряд числа.

ДСОК має: n суматорів (по кількості розрядів мантиси); суматор знакових розрядів; a_i, b_i – цифрові розряди числа в машинному коді; перенесення із старших розрядів мантиси в знаковий розряд; нарешті, обернене перенесення із старшого знакового розряду в молодший розряд цифрової частини числа. Щоб сформувати правила складання чисел в обернених кодах на ДСОК розглянемо теорему.

Теорема: сума обернених кодів чисел є обернений код результату.

При доведенні цієї теореми припускаємо, що числа представлені у формі з фіксованою комою, що стоїть перед старшим цифровим розрядом.

Розглянемо наступні випадки складання на суматорі ДСОК.

Випадок 1. $A>0$, $B>0$, $A+B<1$ (тобто обидва числа позитивні).

Тоді $A_{об}+B_{об}=[A+B]_{об}=A+B$, тобто складання проводиться в прямих кодах.

Приклад: A = 0\0101 (+5)

B = 0\0111 (+7)

C = 0\1100 (+12)

Випадок 2. $A<0$, $B>0$, $|A|>B$, $A_{об}=q-q^{-n}+A$, $B_{об}=B$, тоді $A_{об}+B_{об}=q-q^{-n}+A+B=[A+B]_{об}$, тобто результат негативний і в оберненому коді, треба повертатися до $C_{пр}$ шляхом інверсії розрядів числа.

Приклад: A = –0\1011 (–11), B = 0\0111 (+7).

$A_{об}$=1\0100

$+B_{об}$=0\0111

$C_{об}$=1\1011 $C_{пр}$=1\0100 (–4)

Випадок 3. $A<0$, $B>0$, $A<B$, тут $A_{об}$ = q–q^{-n}+A. Тоді $A_{об}+B_{об}=q-q^{-n}+A+B$. Оскільки сума $(A+B)$ позитивна, то права частина цього виразу стає більше q, що викликає появу одиниці перенесення із знакового розряду в молодший розряд числа (величина перенесення при цьому, рівна $q-q^{-n}$), тоді $A_{об}+B_{об}=A+B$. Результат позитивний, це і є $C_{пр}$.

Приклад: A = –0\0101 (–5); B = 0\0111 (+7).

$A_{об}$ = 1\1010

$+B_{об}$ = 0\0111

$C_{об}$ = 10\0001

$^{+}$ **а** →→1

$C_{пр}$=0\0010(+2). Оскільки $B>|A|$, то їх алгебраїчна різниця позитивна.

Випадок 4. $A<0$, $B<O$, $|A + B|<1$, тут $A_{об}=q-q^{-n}+A$, $B_{об}=q-q^{-n}+B$. Тоді $A_{об}+B_{об}=q-q^{-n}+A+q-q^{-n}+B$. Тут з'являється одиниця перенесення, що дорівнює вилученню з суми $-q^{-n}$, тобто $A_{об}+B_{об}=q-q^{-n}+A+B=[A+B]_{об}$

Приклад: A = –0,0101 ; B = –0,1000.

$A_{об}$=1\,1010

$+B_{об}$=1\,0111

11\,0001

$^{+}$ **а** →→1

$C_{об}$=1\,0010; $C_{пр}$= 1\,1101 (-0,8125).

9 МОДИФІКОВАНІ БІНАРНІ КОДИ

9.1 Переповнення розрядної сітки

Ми не раз спостерігали, як при складанні чисел з однаковими знаками, представлених у формі з фіксованою комою (ФФК), може виникнути переповнення розрядної сітки. При цьому, щоб уникнути спотворення резуль-тату, автомат повинен фіксувати переповнення і відповідно реагувати на це.

9.1.1 Переповнення при складанні прямих кодів

Ознакою переповнення розрядної сітки суматора прямого коду є поява одиниці перенесення із старшого розряду цифрової частини числа.

Приклади. Суматор ДСПК.

а) A=+1010, B=+1101.

$A_{пр}$= 0\1010 (+10)

$+B_{пр}$= 0\1101 (+13)

$C_{пр}$= 0\0111 (+7≠ +23). Отримано спотворений (невірний) результат із-за втрати перенесення П зі старшого цифрового розряду.

б) A=–1100, B= –1010.

$A_{пр}$= 1\1100

+$B_{пр}$= 1\1010

$C_{пр}$= 1\0110 (-6≠-22). Тут теж отриманий невірний результат із-за втрати одиниці перенесення.

9.1.2 Переповнення при складанні доповняльних кодів

Ознакою переповнення розрядної сітки суматора доповняльного коду при складанні позитивних чисел є негативний знак результату, а при складанні негативних чисел – позитивний знак результату.

Приклади:

а) A= +0\1011; B= +0\1010 (перевірку отриманого результату проведіть самостійно).

$A_{доп}$=0\1011
+$B_{доп}$=0\1010
$C_{доп}$≠1\0101

б) A= –0\1011; B= –0\1001 (перевірку отриманого результату проведіть самостійно).

$A_{доп}$= 1\0101
+$B_{доп}$= 1\0111
$C_{пр}$= 0\1100

9.1.3 Переповнення при складанні в обернених кодах

Ознакою переповнення розрядної сітки суматора оберненого коду є знак результату, протилежний знакам операндів.

Приклад:

1) A=0\0111; B=0\1101.

$A_{об}$=0\0111
+$B_{об}$=0\1101
$C_{об}$=1\0100

Знак числа спотворений (невірний):

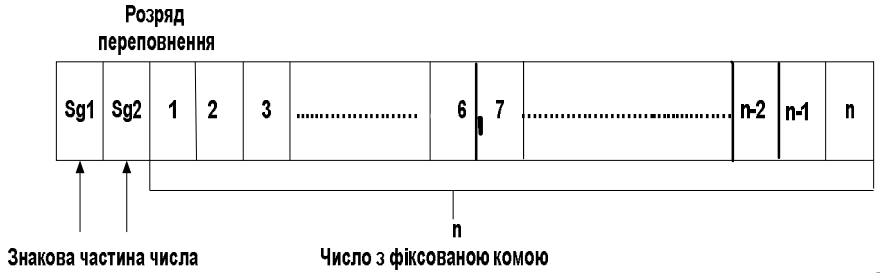
2) A= -0\0110; B= -0\1101.

$A_{об}$=1\1001
$B_{об}$=1\0010
$C_{пр}$≠0\1011

Знак числа спотворений (невірний)

Для усунення втрат від неврахування перенесення одиниці та формування стану переповнення, у складі ЦА повинні бути передбачені додаткові засоби, що дозволяють фіксувати і аналізувати переповнення за їх ознаками з метою усунення спотворення результату.

Щоб фіксувати переповнення розрядної сітки в ДСПК, ДСДК, ДСОК, вводять допоміжний розряд в знакову частину зображення числа, який називають розрядом переповнення числа. Таке представлення числа називається модифікованим (див. рис. 9.1).

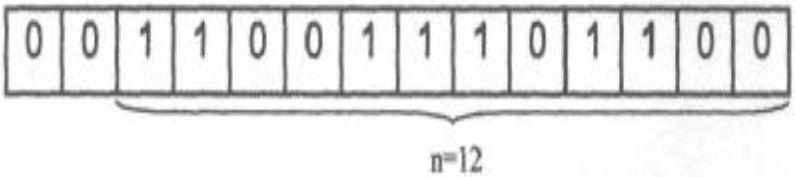


Р

исунок 9.1 – Модифікований код числа для ФФК

Приклади:

а) $[A]_{пр}$=00\1100111011, тоді, оскільки A>0 маємо: $[A]_{М.доп}$=00\1100111011

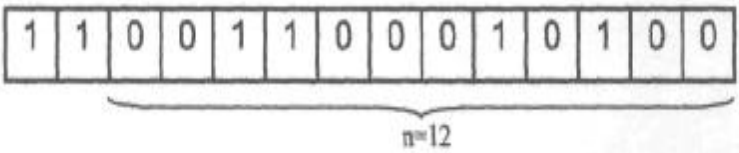


б) $A_{пр}$ = –0\1100111011, оскільки A<0, то $A_{м.доп}$ =100+($–A_{пр}$)=q^2–A

_100\0000000000

0\1100111011

11\0011000101



ПРАВИЛО: для переводу бінарного числа з прямого коду в модифіковний доповняльний код, аналізують знак числа. Якщо в знаковому розряді знаходиться мінус (одиниця), то в два знакових розряди записують одиниці, а всі цифри числа інвертують (одиниці заміняють нулями, а нулі – одиницями), потім до останньої молодшій цифрі (молодшого розряду) числа додають одиницю. Якщо в знаковому розряді стоїть нуль, то додають ще один, а число не змінюють.

ПРАВИЛО: для переводу бінарного числа з прямого коду в модифіковний обернений код, аналізують знак числа. Якщо в знаковому розряді знаходиться мінус (одиниця), то в два знакових розряди записують одиниці, а всі цифри числа інвертують (одиниці заміняють нулями, а нулі – одиницями). Коли в знаковому розряді стоїть нуль, то додають ще один, а число не змінюють.

При виконанні операцій алгебраїчного додавання або вирахування два знакових розряди приймають участь в операції, як рівноправні цифрові розряди. Після виконання операцій, стан знакових розрядів (знак результату) встановлює **лівий знаковий розряд**, а наявність чи відсутність переповнення – **правий знаковий розряд**.

Знакові розряди можуть мати наступні комбінації станів при будь- яких значеннях операндів *A* і *B*:

- 00 – результат числа позитивний, переповнення немає;
- 01 – результат числа позитивний, було переповнення;
- 11 – результат числа негативний, переповнення немає;
- 10 – результат числа негативний , було переповнення.

У машинному представленні чисел, представлених в модифікованому коді, використовується (n+2) розрядів: з них два знакових, кома опускається, встановлюється постійний коефіцієнт формату $K_ф$. Сигнал переповнення g виробляється по наступній математичній моделі (тут знак ∧ позначає логічну операцію множення «і», а риска над символом Sg – логічну операцію інверсії

γ = **1**, якщо $\begin{cases} Sg\mathbf{1} \wedge \overline{Sg}\mathbf{2}= \mathbf{1} \\ \overline{Sg}\mathbf{1} \wedge Sg\mathbf{2}= \mathbf{1}, \end{cases}$ і γ = **0** в інших випадках.

Приклади: а) A= 0\1011 (+11), B= 0\1010 (+10).

$A^м_д$=00\1011
+$B^м_д$=00\1010
$C^м_д$= 0**1**\0101=00\10101 (+21)

Тут ознакою переповнення є **1** в знаковому розряді. За цією ознакою проводиться зрушення числа вправо з одночасним збільшенням коефіцієнта формату $K_ф$ на 1 і відновленням Sg_1 (по значенню Sg_2)обох розрядів Sg_1Sg_2\, тобто знаку 00 результату С числа, далі перевіряється нормалізація числа (наявність 1 після коми в числі).

б) A= –0\1011 (-11), B= –0\1001(-9). Перетворимо в машинну форму: $A^м_{пр}$=11\1011, $B^м_{пр}$=11\1001. Оскільки обидва числа негативні і операція складання виконується на ДСДК, то в суматор числа поступають в доповняльному коді.

$A^м_д$=11\0101
+ $B^м_д$=11\0111
$C^м_д$=10\1100 Було переповнення. Потрібен зсув вправо і відновлення знака. $C^м_д$=11\01100 результату і перетворення його в прямий код.
$C^м_{пр}$=11\10011
\+ 1, тоді $C_{пр}$= 11\10100 (–20).

Аналіз розрядів знаку результату (10 – див. третю строку) показав, необхідність відновлення його (до 11), зрушуючи число вправо на один розряд, з одночасним збільшенням коефіцієнта формату $K_ф$ на 1. Результат негативний, і тому представлений у доповняльному коді. Тому результат перетвориться в істиний (прямий код) через інвертування (обернений код) і з додаванням +1 в молодший розряд.

9.2 Модифіковане складання чисел у форматі з рухомою крапкою (комою)

Числа, представлені у форматі з рухомою крапкою (комою) – ФРК, мають дві часті: мантису і порядок. Тому, операція додавання (складання) виконується окремо над мантисою і над порядком. Отже, в ЦА може бути два суматори: для мантиси і для порядку.

Для чисел з рухомою крапкою справедлива умова нормалізації:

$$q^{-1} \le |m_A| < 1, \qquad (9.1)$$

де q – основа системи числення; m_A – мантиси числа.

Це **нормалізоване представлення числа**, яке вимагає, щоб в старшому розряді мантиси 2-го числа була одиниця. Для двійкової системи це означатиме, що мантиса завжди знаходиться в межах:

$$0{,}5 \le |m_A| < 1, \qquad (9.2)$$

При виконанні автоматом операцій над числами, нормалізують як вхідні доданки A і B, так і вихідний результат C.

Операція нормалізації числа складається з умови нормалізації (9.1) і здійснюється методом зрушення мантиси числа в ту або іншу сторону.

Зрушення можуть проводитися вліво або управо в межах розрядної сітки ЦА за правилами представлених моделлю таблиці 9.1.

Таблиця 9.1 –Зсув звичайних кодів числа (один розряд знаку)

Початкове число	Зсув вліво на один розряд (від порядку вираховується 1)	Зсув вправо на один розряд (до порядку додається 1)
$0/a_1\ a_2 \ldots\ a_n$ $1/a_1\ a_2 \ldots\ a_n$	$a_1/a_2 \ldots\ a_n\ 0$ $a_1/a_2 \ldots\ a_n \varepsilon$	$0/0\ a_1\ a_2 \ldots\ a_n$ $0/1\ a_1\ a_2 \ldots\ a_n$

Величина e залежить від коду. Для доповняльного коду ε=0, для оберненого коду e=1. При складанні чисел результат може вийти з нормалізації як зліва, так і справа (див. табл. 9.1, 9.2).

Ознакою порушення нормалізації числа справа γ=1 (коли результат має переповнення) є наявність різнойменних комбінацій в знакових розрядах сумматора:

$$\gamma=1, \text{ якщо } \begin{cases} Sg1\ \dot{\vee}\ \bar{}Sg2 = 1 \\ \bar{}Sg1\ \dot{\vee}\ Sg2 = 1, \end{cases} \text{ і } \gamma=0 \text{ у решті випадків.}$$

Таблиця 9.2 –Зсув модифікованих кодів числа (два розряди знаку)

Початкове число	Зсув вліво на один розряд (від порядку вираховується 1)	Зсув вправо на один розряд (до порядку додається 1)
$00/\alpha_1,\alpha_2,...,\alpha_n$	$0\alpha_1/\alpha_2,...,\alpha_n 0$	$00/0,\alpha_1,\alpha_2,...,\alpha_n$
$01/\alpha_1,\alpha_2,...,\alpha_n$	$1\alpha_1/\alpha_2,...,\alpha_n 0$	$00/1,\alpha_1,\alpha_2,...,\alpha_n$
$10/\alpha_1,\alpha_2,...,\alpha_n$	$0\alpha_1/\alpha_2,...,\alpha_n \varepsilon$	$11/0,\alpha_1,\alpha_2,...,\alpha_n$
$11/\alpha_1,\alpha_2,...,\alpha_n$	$1\alpha_1/\alpha_2,...,\alpha_n\varepsilon$	$11/1,\alpha_1,\alpha_2,...,\alpha_n$

(де **γ** – ознака порушення нормалізації числа справа, вказує на необхідність зсуву числа вправо на один розряд для відновлення знаку числа).

Ознакою порушення нормалізації числа зліва δ=1 (коли результат по абсолютній величині виявляється менше 1/q) є наявність однакових комбінацій в розряді переповнення і старшому розряді (R1) цифрової частини суматора:

δ=1, якщо $\begin{cases} Sg2 \text{ Ù } R1 = 1 \\ `Sg2 \text{ Ù } `R1 = 1, \text{ і } \delta=0 \text{ у решті випадків,} \end{cases}$

(де δ – ознака порушення нормалізації, що вказує на необхідність зсуву числа вліво на один розряд). Зазвичай Sg1 (або Sg0)– старший, а Sg2 (або Sg1)– молодший знакові розряди.

Таким чином, операція нормалізації отриманого числа (суми двох операндів) складатиметься з сукупності перевірки наявності ознак порушення γ і δ та зсувів (вправо, вліво) числа.

Отже, розглянемо складання чисел $A=m_Ap_A$ и $B=m_Bp_B$, що мають однаковий порядок $p_A=p_B$. Обидві мантиси задовольняють умові нормалізації. Складання мантиси здійснюють на суматорі ДСДК або ДСОК за правилом складання чисел представлених у формі з фіксованою комою. Якщо після складання мантиса результату задовольняє умові нормалізації (тобто δ=0, γ=0), то до цього результату приписується порядок будь-якого з операндів. Інакше відбувається нормалізація числа.

Приклад 1. Знайти суму чисел: A=0,1000·2^{-3} та B=–0,1011·2^{-3}.

Мантиси і порядок обробляються на ДСДК у ФРК.

$[m_A]_д$=00/,1000 $[P_A]_д$=11/101

$+[m_B]_д$=11/,0101 $[P_в]_д$=11/101

$[m_c]_д$=11/,1101

Тут $Sg2\&R1=1$, тобто $\delta=1$, $\gamma=0$. Це означає, що необхідний зсув мантиси $[m_c]_o$ вліво на 1 розряд.

$[m'_c]=11/,1010$ Перевіряємо. Знову $\delta=1$, $\gamma=0$ – необхідний ще зсув вліво на 1 розряд.

$[m''_c]_o$= 11/,0100. Перевіряємо, все гаразд $\delta=0$, $\gamma=0$.

Одночасно зі зрушенням потрібна корекція порядку на мінус $2_{(10)}=11/010_{(2).}$ (що рівнозначно збільшенню порядку на дві одиниці – у доповняльному коді 11/110).

$+[P_c]_o=11/101$

$[\Delta P_c]_o=11/110$ $\qquad$ $[\Delta P_c]_{np}=11/010$

$[P''_c]_o=11/011$ $\qquad$ $P_{C\,np}=-11/101$ (тобто $P= 2^{-5}$)

тоді результат (число) дорівнює:

$[m''_c]_д$= 11/,0100

$[m''_c]_{об}=11/,1011$

$+\quad 1$

11/,1100.

Відповідь: $m_{C\,np}=11/,1100$. З урахуванням порядку запис для 16-ти розрядної ЕОМ C_{np}= 11/,11000000/11/0101/.

Приклад 2. Знайти суму чисел $A=-0,1100\cdot2^4$ та $B=-0,1000\cdot2^4$. Мантиси і порядок обробляються на ДСОК у ФРК (шість розрядів мантиси і чотири для порядку). Порядки $[P_A]=[P_B]=[P_C]_{об}=$ 0,100 (оскільки позитивні і рівні обидва порядки операндів A і B).

$+[m_A]_{об}=11/,0011$

$[m_B]_{об}=11/,0111$

$[m_C]_{об}=110/,1010$

$+\ \mathbf{a}\rightarrow\rightarrow 1$

10/,1011 $\quad$ ($\delta=0$, $\gamma=1$)

Порушення нормалізації справа (тобто $\gamma=1$ означає, що було переповнення). Число зсуваємо вправо на один розряд, відновлюючи знак 11 і одночасно збільшуючи порядок P на 1.

Тобто $[m_C]_{об}=11/,0101$, тепер $\delta=0$, $\gamma=0$. Одночасно коректуємо і порядок:

$+[Pc]=$ 00/100

$[\Delta Pc]=00/001$

00/101 (+5), тоді відповідь: $-0,1010\cdot2^{+5}$ (Самостійно зробіть запис C_{np} для 16-ти розрядної ЕОМ).

10 СКЛАДАННЯ ЧИСЕЛ ПРИ РІЗНИХ ЗНАЧЕННЯХ ПОРЯДКІВ

Для операції складання чисел необхідною умовою є зіставлення вагів розрядів операндів один одному. Тому спочатку потрібно вирівняти порядки, що спричинить за собою тимчасове порушення нормалізації одного з доданків (A або B).

Вирівнювання порядків означає, що порядок меншого числа треба збільшити на величину $\Delta P=P_A-P_B$, що означає зсув мантиси меншого числа вправо на кількість розрядів, рівне $P_A \geq P_B$. Тому, цифровий автомат повинен спочатку розпізнати, який порядок з двох чисел є меншим. На це вкаже знак ΔP. Якщо $P_A \geq P_B$ – знак позитивний і навпаки, якщо знак негативний, то $P_A < P_B$. Цю операцію іноді виконують шляхом порівняння чисел.

10.1 Алгоритм операції складання у форматі з рухомою крапкою (комою)

Отже, операція складання (віднімання) виконується в наступній послідовності.

1. Перевести операнди в двійкові доповняльні (або обернені) модифіковані коди, перевіривши перед цим нормалізацію початкових чисел.

2. Визначити різницю порядків $\Delta P = P_A - P_B$

3. Якщо $\Delta P>0$, зсунути мантису числа B на ΔP розрядів вправо; якщо $\Delta P<0$, зсунути мантису числа A на ΔP розрядів вправо; якщо $\Delta P=0$, мантиси не зсуваються (розряди виходять за межі розрядної сітки мантиси втрачаються).

4. Виконати операцію алгебраїчного складання (віднімання) над мантисами. Алгебраїчне складання виконується за наступним правилом. Аналізується знак числа:

- якщо знак позитивний (00), то в суматор мантиси число поступає в прямому коді;
- якщо знак негативний (11), то в суматор мантиси число поступає в доповняльному (оберненому) коді;

Складання проводиться порозрядно, за правилами бінарної арифметики, з перенесенням одиниці переповнення в старший розряд. Знакові розряди також беруть участь в складанні. Одиниця переповнення в старшому знаковому розряді для суматорів доповняльного коду пропадає, для суматорів оберненого коду по оберненому зв'язку додається до молодшого розряду мантиси.

До представлення результату в прямому коді, після складання, аналізується знак результату:

- якщо знак 01(або 10), то знак спочатку відновлюється шляхом зсуву мантиси вправо на один розряд з одночасним збільшенням порядку на 1;
- якщо знак 00, то результат- число позитивне, представлено в прямому коді;
- якщо знак 11, то результат- число негативне і представлено в оберненому (доповняльному) коді.

В цьому випадку, потрібне перетворення числа із оберненого (доповняльного) в прямий код, шляхом інвертування числа (окрім знаку) для ДСОК і інвертування числа (окрім знаку) з додаванням 1 до молодшого розряду мантиси для ДСДК. Результату встановлюється порядок більшого за модулем числа.

5. Проводиться перевірка числа на нормалізацію (для нормалізованих чисел, після знаку мантиси повинна стояти одиниця). Якщо після знаку мантиси стоїть нуль, то число зсувається вліво на один розряд (з одночасним зменшенням порядку на 1). Перевіряється нормалізація (перевіряємо **ознаки порушення γ і δ)**. Цикл може повторюватися до досягнення умов нормалізації **γ=0 і δ=0)**. За цим алгоритмом виконують операції складання і віднімання АЛП.

Приклад. Скласти A=+0,1011·2^{-2} и B = –0,1001·2^{-3}

Числа задані в природному вигляді. В АЛП використовується два суматори оберненого коду: суматор мантис (шість розрядів, включаючи знак); суматор порядків (разом із знаком – чотири розряди).

РІШЕННЯ.

1. Враховуючи, що числа задані вже в нормалізованому вигляді (після коми стоїть 1), представимо їх в машинному вигляді у форматі з рухомою крапкою (модифікований код): $A^{м}_{пр}$=00.1011.11.10; $B^{м}_{пр}$=11.1001.11.11.

2. Переводимо мантиси чисел в обернені коди: $m_{Аоб}$=00.1011; $m_{Воб}$=11.0110.

3. Для вирівнювання порядків чисел, а значить, і їх вагових розрядів, необхідно з'ясувати, яке із заданих чисел підлягає тимчасовій денормалізації. Для цього автоматом визначається різниця порядків чисел: $\Delta P=P_A–P_B$. Переведемо порядки чисел в обернені коди.

$P_{Аоб}$=11.01 ; $P_{Воб}$=11.00. Тоді, $\Delta P=P_{Аоб}–P_{Воб}$ =11.01–11.00

Замінимо операцію віднімання на операцію складання.

$\Delta P=P_{Аоб}–P_{Воб} = P_{Аоб} + \overline{P}_{Впр}$ =11.01 + 00.11 (де $P_{Впр}$ – інверсний код $P_{Воб.,}$ включаючи і знакову частину).

Виконаємо додавання.

11.01
+00.11
100.00
+ **a** →1
00.01

Величина ΔP позитивна, тому $P_A > P_B$. Треба зсувати мантису числа вправо на ΔP розрядів, тобто на один розряд, збільшивши одночасно порядок на 1.

4. Зсунемо мантису числа B вправо на один розряд і складемо їх значення на ДСОК.

5. Перевіримо умову нормалізації мантиси результату справа, зліва.

$m_{Aоб}$=00.1011 (тут δ=0, γ=0)
$m_{Bоб}$=11.0110 (тут δ=0, γ=1, необхідно зробити зсув $m_{Bоб}$ вправо)
$m_{Bоб}$=11.1011
+$m_{Aоб}$=00.1011
$m_{Cоб}$=100.0110
a →→ 1
00.0111 (тут δ=1, γ=0; необхідно зробити зсув вліво)

$m_{Cоб}$=00.1110 і тому робимо корекцію порядку на ΔP=–1

$P_{Aоб}$=11.01
+$\Delta P_{об}$= 11.10 (ΔP представлено в оберненому коді)
110.11
+ **a** →1
$P_{Cоб}$=11.00

Результат $P_{Cоб}$ отриманий в оберненому коді, тобто знак негативний (11). Визначаємо прямий код порядку і записуємо результат $C_{пр}$=00.1110·2^{-3}.
Відповідь: $C_{пр}$=00.1110.11.11

Приклад. Скласти на ДСДК числа $A_{пр}$= +5, $B_{пр}$= –3, представлених в десятковій системі числення. $A_д$=00/5; $B_д$=99/7. Проведемо складання $A_д$+$B_д$=00/5+99/7=*1*00/2=00/2 (+$2_{(10)}$). Одиниця (*1*) пропадає, оскільки складання йде на ДСДК.

11 МНОЖЕННЯ ДВІЙКОВИХ ЧИСЕЛ

11.1 Методи множення бінарних чисел

Розглянемо основні способи виконання операції множення для різних систем. Найпоширенішим способом множення чисел є спосіб порозрядного множення множеного на множник, починаючи з молодшого розряду (1-й спосіб), починаючи зі старшого розряду (2-й спосіб). Розглянемо приклад:

```
  347
  532
  694
 +1041
+1735
 184604
```

```
  347   множене (A)
  532   множник (B)
1735 частковий (розрядний) добуток
+1041
 +694
184604  повний добуток (C)
```

Аналіз способів множення чисел в десятковій системі показує, що операція множення складається з порозрядного множення множеного на множник з перенесенням переповнення в старший розряд, зрушення часткових добутків на один розряд вліво (вправо), підсумовування часткових добутків.

У двійковому численні це завдання значно спрощується, оскільки множити порозрядно немає необхідності. Насправді, якщо множити множене на "1", то це повторення множеного із зсувом на один розряд вправо (вліво), а на "0" – записуються одні нулі із зрушенням.

```
   1101
 * 1101
   1101
  +0000
 +1101
+ 1101
 10101001
```

```
   1101
 * 1101
   1101
 +  1101
   + 0000
  +1101
 10101001
```

В обох випадках операція множення складається з ряду послідовних операцій зсуву і складання часткових добутків. Таким чином, операція множення зводиться до складання часткових

добутків, які виходять з множеного або нулів, якщо в розряді множника «нуль», або множеного, якщо в розряді множника «одиниця» і відповідним або зсувом.

Окрім операції складання, при виконанні множення, з'являється операція зсуву чисел. При цьому, можливі два варіанти:

- зсувати множене відповідно до вказівок множника;
- зсувати суму часткових добутків.

Розглянемо основні методи виконання операції множення ЦА.

МЕТОД 1. Нехай A – множене > 0, B – множник > 0, а C – добуток.

Запишемо числа у ФФК: $A=0,\alpha_1\alpha_2,...,\alpha_n$; $B=0,b_1,b_2,...,b_n$ (A і B правильні дроби). B – множник і він визначає розрядність C (чим він більший, тим більша розрядність C). Тоді, $B=b_1\cdot 2^{-1}+b_2\cdot 2^{-2}+...+b_n\cdot 2^{-n}$.

Звідси: $C=A\cdot B = 0,\alpha_1\alpha_{2...}\ \alpha_n(b_1\cdot 2^{-1}+b_2\cdot 2^{-2}+..\ .+b_n\cdot 2^{-n})=$

$$=(2^{-1}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_1+(2^{-2}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_2+...+(2^{-n}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_n.. \qquad (11.1)$$

Множник 2^{-n} означає зсув на n розрядів вправо числа, що укладено в дужки, тобто зсуваэться вправо множене і множення на множник починається зі старших розрядів. Структурно це зображено на рис. 11.1.

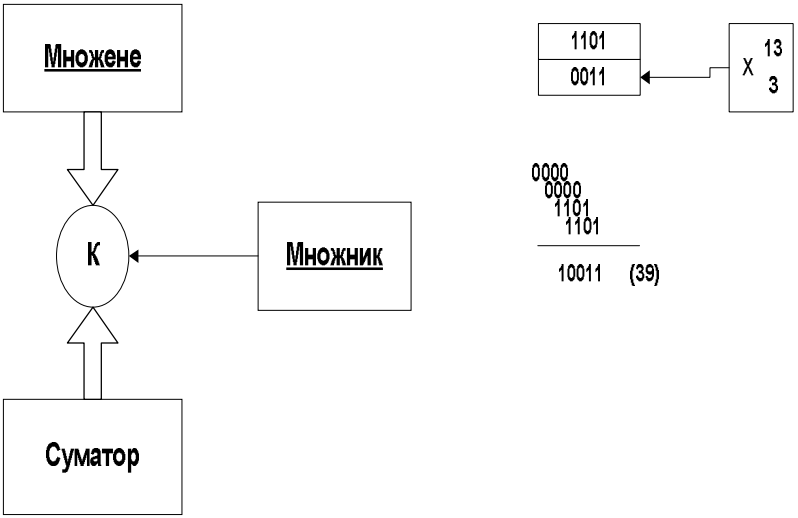


Рисунок 11.1 – Метод множення зі зрушенням вправо множеного

МЕТОД 2. Нехай $A=0,\alpha_1\alpha_{2...}\ \alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Перетворимо множник по методу Горнера:

$$B=(...((b_n\cdot 2^{-1}+b_{n-1})\cdot 2^{-1}+b_{n-2})\cdot 2^{-1}+...+b_2)\cdot 2^{-1}+b_1)\cdot 2^{-1}.$$

Тоді,

$$C=A\cdot B=(...(b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot 2^{-1}+b_{n-1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot 2^{1}+...+b_1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^{-1}, \quad (11.2)$$

За цією моделлю, множення починається з молодших розрядів і вправо зрушується. Структура наведена на рис. 11.2

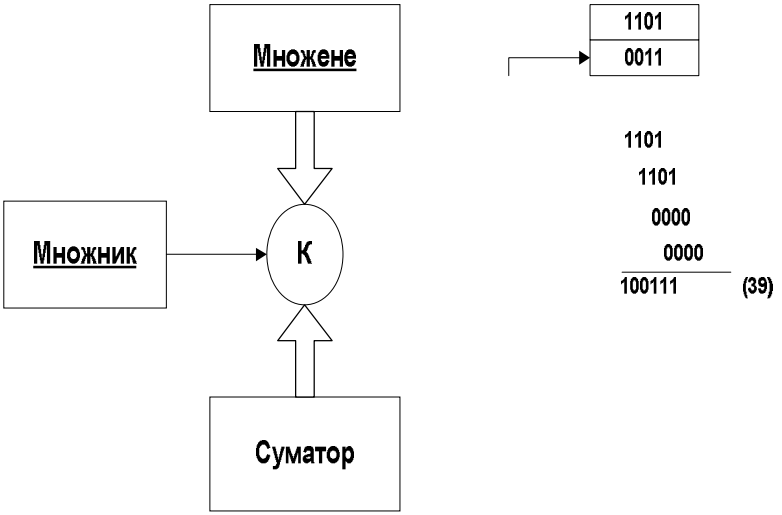


Рисунок 11.2 – Метод множення зі зрушенням вправо суми часткових добутків

МЕТОД 3. Нехай $A=0,\alpha_1\alpha_{2...}\alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Використовуючи метод Горнера, запишемо множник:

$$B=2^{-n}(b_1\cdot 2n^{-1}+b_2\cdot 2^{n-2}+...+b_{n-1}\cdot 2^1+b_n\cdot 2^0).$$ Тоді, $C=A\cdot B=$

$$=2^{-n}(b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n+(2^1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot b_{n-1}+...+(2^{n-1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot b_1 \quad (11.3)$$

Це означає: множення починається з молодших розрядів і множене зсувається вліво на один розряд у циклі. Структура представлена на рис.11.3.

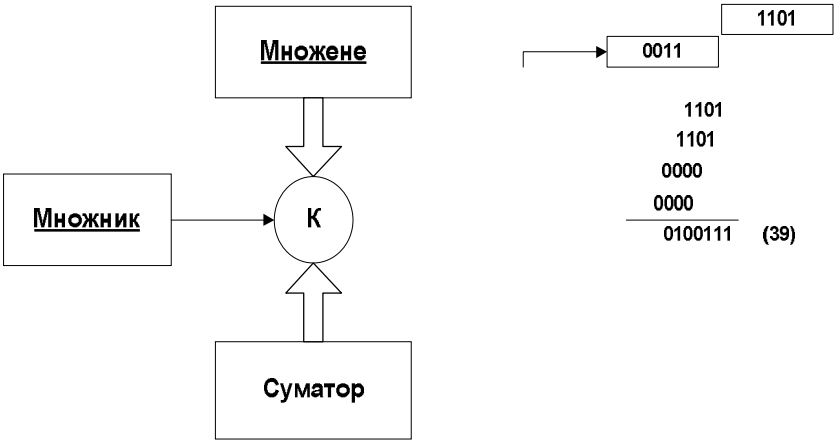


Рисунок 11.3 – Метод множення зі зрушенням вліво множеного на один розряд у циклі

МЕТОД 4. Нехай $A=0,\alpha_1\alpha_{2...}\alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Запишемо B – множник по методу Горнера, як у попередньому методі. Тоді, $C=A\cdot B=$

$=2^{-n}(...(2^1(b_1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)+b_2\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^1+\ldots$

$+b_{n1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^1+ +b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n).$ (11.4)

За цією моделлю множення починається зі старшого розряду і в кожному циклі сума часткових добутків зрушується вліво. Схема такого множного пристрою наведена на рис.11.4.

Рисунок 11.4 – Метод множення зі зрушенням вліво суми часткових добутків

Таким чином, для реалізації операції множення необхідно мати, як мінімум: суматор, регістри для зберігання множеного й множника, схему аналізу розрядів множника. Суматор і регістри повинні мати ланцюги зсуву вмісту в ту або іншу сторону відповідно до прийнятого методу множення.

11.2 Множення чисел з фіксованою крапкою (комою) на ДСПК

Запишемо машинне зображення множеного і множника у формі з фіксованою комою в прямому коді. $A_{пр}=Sg,\alpha_1\alpha_2...\alpha_n$; $B_{пр}=Sg,b_1b_2...b_n$. Тоді, їхній добуток запишеться як $C_{пр}=Sg,c_1c_2...c_n$, де $Sg_C=Sg_A\oplus Sg_B$, де $\oplus$ – знак додавання функції «Σ по mod2» (її ТІ: $1\oplus1=0$, $0\oplus0=0$, $1\oplus0=1$, $0\oplus1=1$).

Таким чином, при використанні ДСПК, знак добутку визначається окремо від цифрової частини, потім виконується операція множення. Вона виконується відповідно до заданої структури множного пристрою (див. наприклад, рис. 11.5) і методу множення (див. наприклад, метод 2).

За методом 2 множення починається з молодшого розряду і зсувається вправо сума (Σ) часткових добутків.

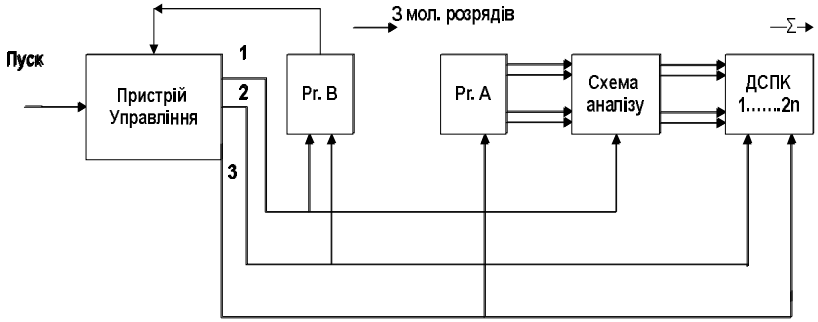


Рисунок 11.5 – Структура ЦА для пристрою множення чисел з фіксованою комою

Приклад. Помножити $A_{пр}$=11/11010 (–26); $B_{пр}$ =00/11001 (25), С= –650.

РІШЕННЯ: Визначається знак добутку С: $1\oplus0$ =1.

Приймемо:

1) суматор має 10 розрядів (без знака).

2) регістри мають по 5 розрядів (без знака).

Послідовність дій представимо таблицею 11.1

Для спрощення запису таблиць, приймемо наступні умовні позначки:

- оператор := привласнення значення (блоку ліворуч привласнюється значення, що є праворуч);

- позначення, наприклад, [См] – вміст суматора;

¾ ®

- оператор[РгА]зсуву вмісту, наприклад, регістра А вправо наодин розряд;

- позначення В. П. – вхідне положення;

-позначення $A_{пр}$, $B_{пр}$- цифрова частина множенного, множника (прямий код).

Відповідь: $C_{пр}$= 11/1010001010.

Таблиця 11.1 – Приклад рішення

Суматор [см]	Регістр [РгВ]	Коментар
+0000000000	11001	і.П.[см]:=0; РгА:= А;
11010		РгВ:= В
1101000000		b_5=1; [см]+[РгА];
0110100000	-1100	[РгВ]; [см];
0011010000	--110	b4=0; [РгВ]; [см];
+0001101000	---11	b3=0; [РгВ]; [см];
11010		b2=1; [см]:=[см]+ [РгА];
1110101000		[см]; [РгВ];
+0111010100	----1	b1=1; [см]:=[см]+ [РгА];
11010		
10100010100	-----	[РгВ]; [см];
1010001010		Кінець

Якщо при множенні виникає одиниця переносу зі старшого розряду, то її зберігають шляхом зсуву суматора (Σ), тобто необхідно передбачати в ЦА стробування (фіксування) сигналу переповнення для виробу зсуву на один розряд вправо. Цей спосіб одержав найбільше поширення в практиці ЦА.

11.2.1 Множення чисел з рухомою комою

Числа у ФРК представляються мантисою і порядком, тому виконання операції множення складається із двох дій:

- перемножування мантис;
- додавання порядків.

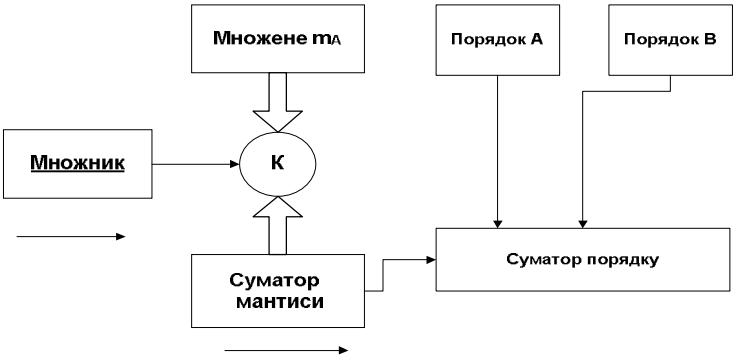


Рисунок 11.6 – Структура ЦА для пристрою множення чисел у ФРК

Результат множення може вийти денормалізованим, тому потрібна перевірка на нормалізацію числа (критерій δ) та, при необхідності, його нормалізація з відповідною корекцією порядку P результату. Пристрій множення чисел із рухомою комою представлено ЦА зі структурою рис. 11.6.

Таблиця 11.2 – Перемножування мантис A і B.

Знак результату	Суматор [см]	Регістр m_b	Коментар
$Sg_C=Sg_A \oplus Sg_B=$ $1 \oplus 0=1$	+00000	10011	і.П. $P_r mB:=[m_B]$; $P_r mA:=[m_A]$; $P_r PA:=[P_A]$;$P_r PB:=[P_B]$; $C_M m_C=0$
	11001		$b_5=1$; См:=[$C_M m_C$]+[$P_r mA$];
	11001	01001	$[\overline{Prm_B}]$; $[\overline{смm_C}]$;
	+011001		b4=1;$[\overline{смm_C}]$:=$[\overline{смm_C}]$+[$P_r mA$];
	11001		$[\overline{смm_C}]$;$[\overline{Prm_B}]$
	1001011	00100	
	01001011	00010	b3=0; $[\overline{смm_C}]$;$[\overline{Prm_B}]$.
	001001011	00001	b2=0; $[\overline{смm_C}]$;$[\overline{Prm_B}]$.
	+001001011		b1=1; $[\overline{смm_C}]$:=$[\overline{смm_C}]$+[$P_r mA$];
	11001		
	111011011	00000	$[\overline{Prm_B}]$; $[\overline{смm_C}]$;
	0111011011		Кінець

Приклад. A= –0,11001·2^{-3}; B=0,10011·2^{1} (перемноження мантис здійсніть самостійно зг. п. 11.2. Суматор – 10 розрядів, Рг для A і B – по 5 розрядів). Додавання порядків робимо на ДСОК у ФРК.

$[P_A]_{об}$ =11/100
$[P_B]_{об}$=00/001
$[P_C]_{об}$=11/101 $[C]_{пр}$ = 11/010; $C_{пр}$=–010$_{(2)}$

Тому що мантиса результату не задовольняє нормалізації (тобто δ=1, γ =0), то робимо зсув мантиси вліво на один розряд (див. табл.9.2) і проводимо корекцію порядку: m=11/1110110111.

$_{+}[P_C]_{об}$ = 11/101
$[\Delta P]_{об}$ = 11/110
111/011
а →→1
$[P_C]'_{об}$ = 11/100; $[P_C]'_{пр}$=11/011 (–3).

Відповідь: C= –0,1110110111·2^{-3}

11.2.2 Особливі випадки при множенні

При множенні можуть спостерігатися наступні особливі випадки:

- один зі співмножників = 0. Операція блокується з видачею результату C = 0;

- порядок результату дорівнює найбільшій негативній величині. Формується "0";

- множене найбільше і негативне число. Збільшується множене на 2^{n-1} тобто, зсувається вліво.

Корекції роблять за допомогою аналізаторів, що вводять у ЦА.

ПРАВИЛО. ***Якщо в якості множника виступає число зі степінного ряду вагових значень бінарної системи, то множення проводиться зсувом числа вліво на число розрядів, що дорівнює ступені множника.***

Приклад. Помножити A=+00011 (+3); B=+00100 (+4=2^2). C=0\00011 ·0\00100= 0\00011$^{\chi}$(2^2)= 0\01100.

12 МНОЖЕННЯ ЧИСЕЛ НА ДСДК

З одного боку, якщо числа зберігаються в ЦА в доповняльних кодах, то і операцію множення зручно робити на ДСДК. З іншого боку, при множенні на ДСДК виникають проблеми, усунути які можна з огляду на певні правила.

12.1 Множення чисел на ДСДК при позитивному множнику

При позитивному множнику для ДСДК діє наступне правило.

Правило. Добуток доповняльних кодів співмножників дорівнює допо-вняльному коду результату тільки при позитивному множнику B>0.

Дійсно: нехай A – будь-яке число, тоді $A = A_{доп}$ і якщо B>0, тоді маємо:

$$A\cdot B=A_{доп}\cdot 0,b_1b_2...b_n-A_{доп}\cdot b_1\cdot 2^{-1}+A_{доп}\cdot b_2\cdot 2^{-2}+...+A_{доп}\cdot b_n\cdot 2^{-n}.$$

На основі раніше розглянутої теореми про додавання доповняльних кодів, робимо висновок, що результат множення представлений у доповняльному модифікованому коді.

Можна сформулювати алгоритм множення чисел на ДСДК.

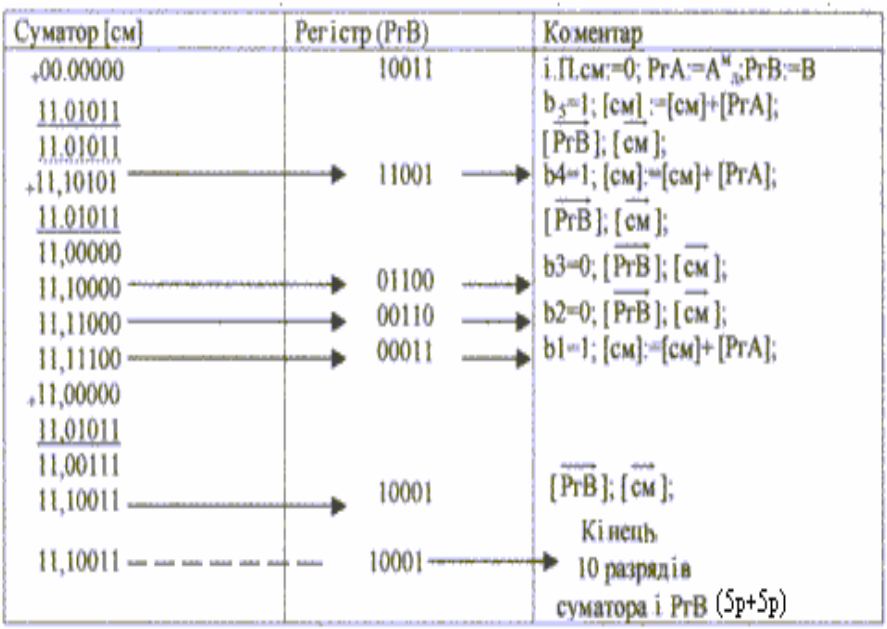
Якщо множник більше "0", то множення на суматорі доповняльного модифікованого коду полягає в наступному. Аналізується розряд множника, починаючи з молодшого. При B_i = 1, до вмісту суматора додається множене. При B_i = 0, до вмісту суматора нічого не додається. Після кожного аналізу і додавання виробляється модифіковане зсув суматора і множника вправо на один розряд.

Приклад. Помножити числа A=–0\10101 (–21), B=0\10011 (+19).

Метод 2. ДСДК. Запишемо числа $A^{м}_{доп}$ =11\01011; $B^{м}_{доп}$=00\10011. Суматор має 7 розрядів, РгB 5 розрядів. Рішення наведено в таблиці 12.1.

Результат отриманий у доповняльному коді зі знаком мінус (11). Після перетворення його в прямий код($C_{пр}$=11/0110001111) знак зберігається Для зменшення апаратних засобів, результат зберігається в суматорі і РгB (при зсувах молодші розряди суматора надходять у старші розряди РгB).

Таблиця 12.1 – Множення чисел на ДСДК (метод 2, B >0)



12.2 Множення чисел на ДСДК при негативному множнику

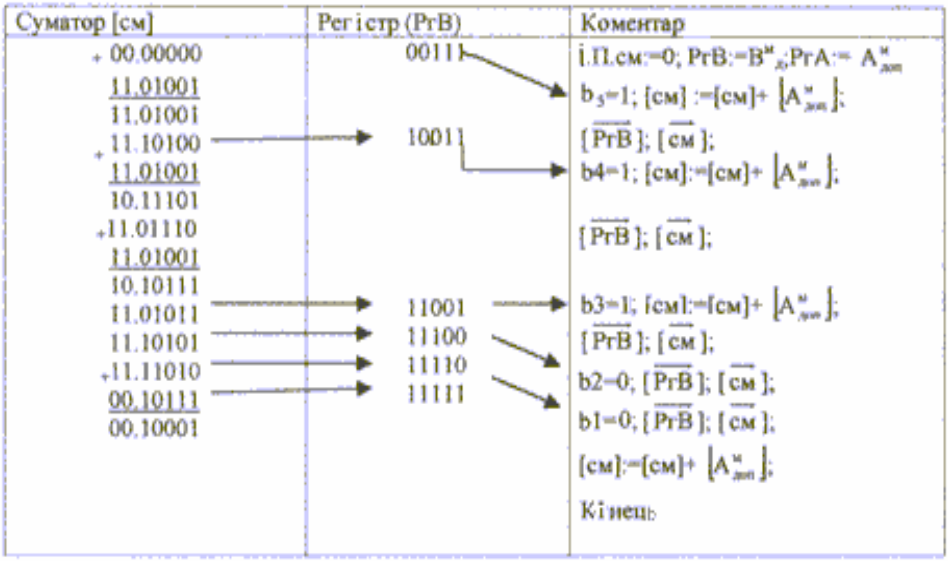
Другий випадок, коли A – будь-яке число, а множник B<0. $B_{äîï} = 1,\overline{\hat{a}_1}\,\overline{\hat{a}_2}\ldots\overline{\hat{a}_n}\,\grave{}$, де $\overline{\hat{a}_n}\,\grave{}$ означає $\overline{\hat{a}_n}$ +1

На основі $|A|+[A_{доп}] = q$, де $|A|$ – абсолютне значення числа A. Доповняльний код є математичним доповненням до числа основи системи числення [1]. Тому (для двійкового коду) можна записати, що $-B=B_м = B_{доп}-2$ (тому що $-(B+[B_{доп}])= -2$). Тоді, $A\cdot B=A\cdot (B_{доп}-2)$.

Отже, добуток чисел $A\cdot B = A(1,\overline{\hat{a}_1}\,\overline{\hat{a}_2}\ldots\overline{\hat{a}_n}\,\grave{}-1) = A\cdot 1,\overline{\hat{a}_1}\,\overline{\hat{a}_2}\ldots\overline{\hat{a}_n}\,\grave{}-A$. (З урахуванням того, що число B негативне *і значущі цифри його доповняльного коду мають значення* $|[B_{доп}]|=1-|B|$).

Таким чином, ***при негативному множнику добуток доповняльних кодів операндів не дорівнює доповняльному коду результату.*** Якщо $(-A)$ замінити на $(\overline{A})$, то можна ввести правило.

Таблиця 12.2 – Множення чисел на ДСДК при $B<0$



ПРАВИЛО. ***Якщо множник негативний, то добуток чисел на суматорі доповняльного коду дорівнює додатку виправлення*** $[\overline{A^{і}_{вп}}]$ **+1** ***до добутку доповняльних кодів співмножників. При цьому, виправлення - це повністю інвертоване*** $\overline{A^{і}_{вп}}$ ***, включаючи і знак з додаванням*** **1** ***до молодшого розряду.***

Приклад. Помножити числа A=–10111 (–23) і B=–11001 (–25) на ДСДК (метод 2).

Підготовчі роботи. Запишемо: $A^{м}_{доп}$ =11\01001; $B^{м}_{доп}$ =11\00111; $(-A)=\overline{A_{вп}}$ =00\10111.

Рішення наведено в таблиці 12.2. C = 00\1000111111 (+575).

Перевірка: (-23)·(–25)=+575

Таким чином, при множенні чисел на ДСДК одержуємо одночасно знакову і цифрову частини добутку $C=A\cdot B$.

13 МНОЖЕННЯ ЧИСЕЛ НА ДСОК

13.1 Множення чисел на ДСОК при позитивному множнику

За аналогією із ДСДК, при множенні операндів заданих у оберненому модифікованому коді, розглянемо два випадки: $B>0$ і $B<0$.

Випадок $B>0$. ***Добуток обернених кодів співмножників дорівнює оберненому коду результату тільки у випадку позитивного множника.***

Доказ. Нехай множене $A=A_{об.}$, а множник $B>0$. Тоді: $A\cdot B=A_{об}\cdot 0,b_1b_2...b_n = A_{об}\cdot b_1\cdot 2^{-1}+A_{об}\cdot b_2\cdot 2^{-2}+...+A_{об}\cdot b_n\cdot 2^{-n}$. У правій частині виходить обернений код результату тому що $B=B_{об}$. Це означає, що множник B позитивний.

Алгоритм. Множення на ДСОК (при B>0) виконується в наступній послідовност (наведена в таблиці 13.2):

- у суматор записуються 11.111...1 (машинний нуль оберненого коду);

- у регістр *B* записується множник у прямому коді. Ведеться аналіз розрядів Рг*B* (починаючи з молодшого), і якщо там "1", то до вмісту суматора додається обернений код множеного. Якщо там 0, то до суматора нічого не додається;

- після циклу додавання і аналізу, виробляється зсув вправо вмісту суматора і Рг*B* на один розряд (можна з переносом молодших розрядів суматора в старші розряди Рг*B*);

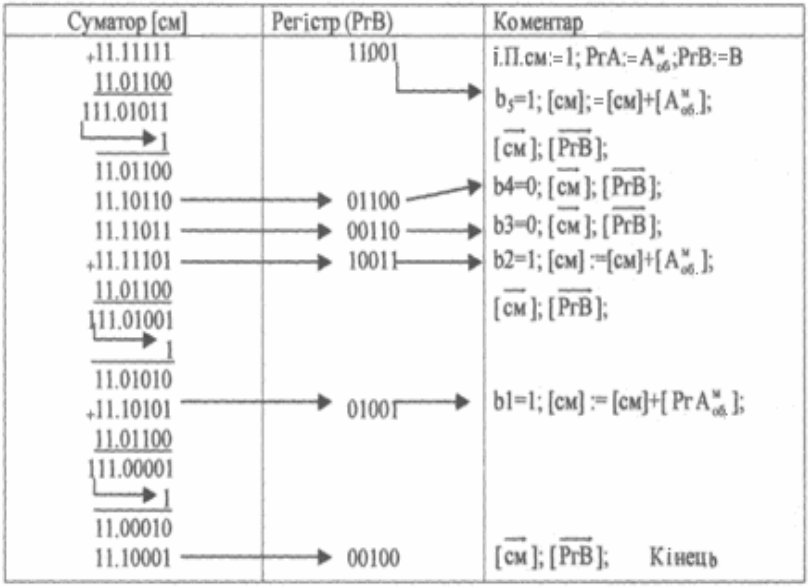
- якщо знак результату негативний, то він у оберненому коді і необхідне перетворення в прямий код;

-після перетворення в прямий код, перевіряємо нормалізацію результату.

Приклад. Помножити на ДСОК числа: A = –0/10011; B = –0/11001. Запишемо числа в машинному зображенні:

$A^{м}_{об}$=11/01100; $B^{м}_{об}$=00/11001.

Таблиця 13.1 – Множення на ДСОК при позитивному множнику B>0



Відповідь $C^{м}_{об}$=11/1000100100; $C_{пр}$= –111011011.

13.2 Множення чисел на ДСОК при негативному множнику

Випадок B<0. ***Якщо множник негативний, то добуток чисел на суматорі ДСОК є додатком виправленя [`A] і [A]$_{об}$×2^{-n} до отриманого добутку обернених кодів співмножників.***

Доказ. Нехай A=[A]$_{об}$ і B<0, тоді $[B]_{їá} = 1, \hat{a}_1, \hat{a}_2 \ldots \hat{a}_n$; відповідно до подання у ФРК, фіксовано перед старшим розрядом |B|+[B]$_{об}$=q–q^{-n}=2–2^{-n}; звідси [B]$_{об}$=2+B–2^{-n}; тож, $B = 1, \hat{a}_1, \hat{a}_2 \ldots \hat{a}_n$ + 2^{-n}–1, добуток дорівнює: $A \times B = A_{об} \times 1, \hat{a}_1, \hat{a}_2 \ldots \hat{a}_n + [A]_{об} \times (-2^{-n}) + \overline{\hat{A}_{îá}}$; є два виправлення: перше $[A]_{об} \times (-2^{-n})$ порівняне мале (тому часто ігнорується) і друге $+\overline{\hat{A}_{îá}}$.

Алгоритм. Множення на ДСОК (при В<0) виконується в наступній послідовності (наведена в таблиці 13.2)**:**

- у суматор записуються 11.111...1 (машинний нуль оберненого коду);

- у регістр *В* записується множник у оберненому коді (без знака);

- у суматор додається множене у оберненому коді $A_{об}$;

- ведеться аналіз розрядів Рг*В* (починаючи з молодшого), і якщо там «1» («0»), то до вмісту суматора *додається* (нічого не додається) *обернений код множеного*;

- після циклу додавання і аналізу робиться зсув вправо вмісту суматора і Рг*В* на один розряд (можна з переносом молодших розрядів суматора в старші розряди Рг*В*);

- після аналізу і зсуву старшого розряду множника до результату додають виправлення [$\overline{\hat{A}_{iá}}$] - інверсію $A_{об}$, включаючи і знак;

- якщо знак результату негативний, то він у оберненому коді і необхідне перетворення у прямий код;

- після перетворення в прямий код, необхідно перевірити нормалізацію результату.

Приклад. Метод 2, множення чисел на ДСОК. *A*=–0/110101; *B*=–0/101000; *A*·*B*=+2120. Рішення: Запишемо машинне зображення чисел.

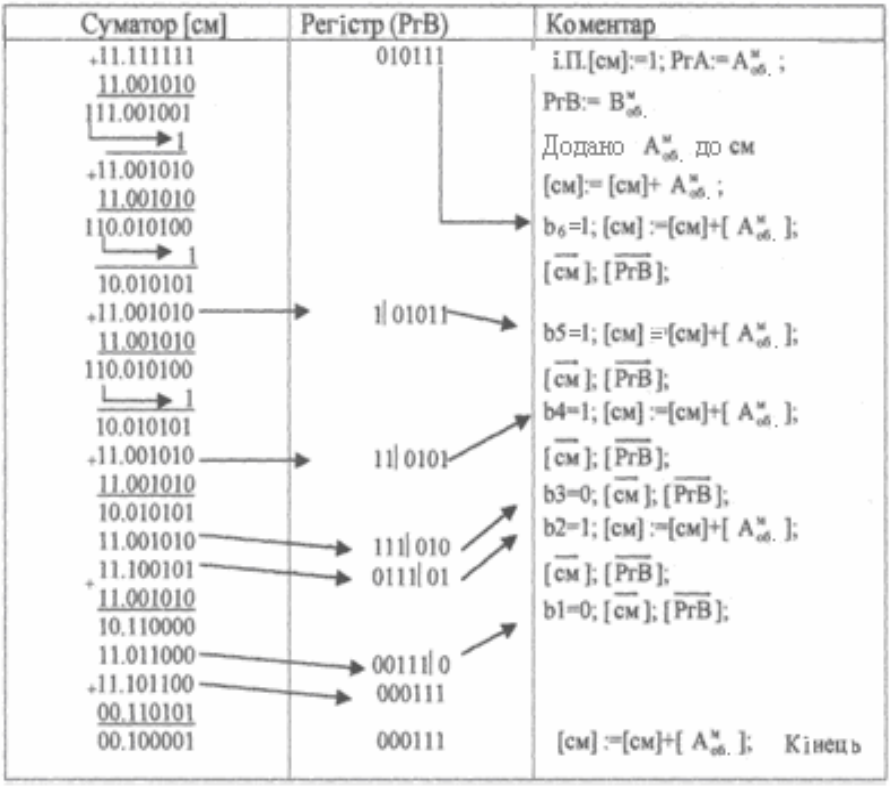
$A^{м}_{об}$ = 11/001010; $B^{м}_{об}$ = 11/010111; $\left\lfloor \overline{\hat{A}^{i}_{iá}} \right\rfloor$= 00/110101.

Послідовність виконання операції множення наведена в таблиці 13.2.

З урахуванням розрядів Рг*В* будемо мати 100\100001000111. Переповнення «1» із знакового розряду Sg_1 додається до молодшого розряду числа. Тоді, одержимо відповідь: $C_{пр}$=00\100001001000=$2120_{(10)}$.

Існує ряд методів множення заснованих на роздільному під-сумковуванні груп часткових доданків з наступним об’єднанням сум з урахуванням переносів. Роздільна обробка проміжних сум і переносів вимагає так званого "дерева суматорів" (використано в IBM-360).

Таблиця 13.2 – Множення на ДСОК при негативному множнику



Існують також **прискорені методи множення**, засновані на використанні матриць проміжних результатів. Розглянемо схему множення:

$$A = a_5,\ a_4,\ a_3,\ a_2,\ a_1$$
$$*B = b_5,\ b_4,\ b_3,\ b_2,\ b_1$$
$$a_5b_1\ a_4b_1\ a_3b_1\ a_2b_1\ a_1b_1$$
$$+.....$$
$$a_5b_5\ a_4b_5\ a_3b_5\ a_2b_5\ a_1b_5$$
$$C_{10}\ C_9.....C_2\ C_1$$

Цю схему можна представити також у вигляді матриці таблиці 13.3. Кожний елемент цієї матриці дорівнює "0" або "1".

Таблиця 13.3- Матричне множення чисел

B_i	a_i				
	a_5	a_4	a_3	a_2	a_1
B_1	a_5B_1	a_4B_1	a_3B_1	a_2B_1	a_1B_1
B_2	a_5B_2	a_4B_2	a_3B_2	a_2B_2	a_1B_2
B_3	a_5B_3	a_4B_3	a_3B_3	a_2B_3	a_1B_3
B_4	a_5B_4	a_4B_4	a_3B_4	a_2B_4	a_1B_4
B_5	a_5B_5	a_4B_5	a_3B_5	a_2B_5	a_1B_5

Добуток двох чисел можна одержати, якщо підсумувати елементи матриці в діагональному порядку. Складати доводиться тільки 0 або 1, тому операцію додавання можна виконувати за допомогою напівсуматорів і суматорів на один двійковий розряд.

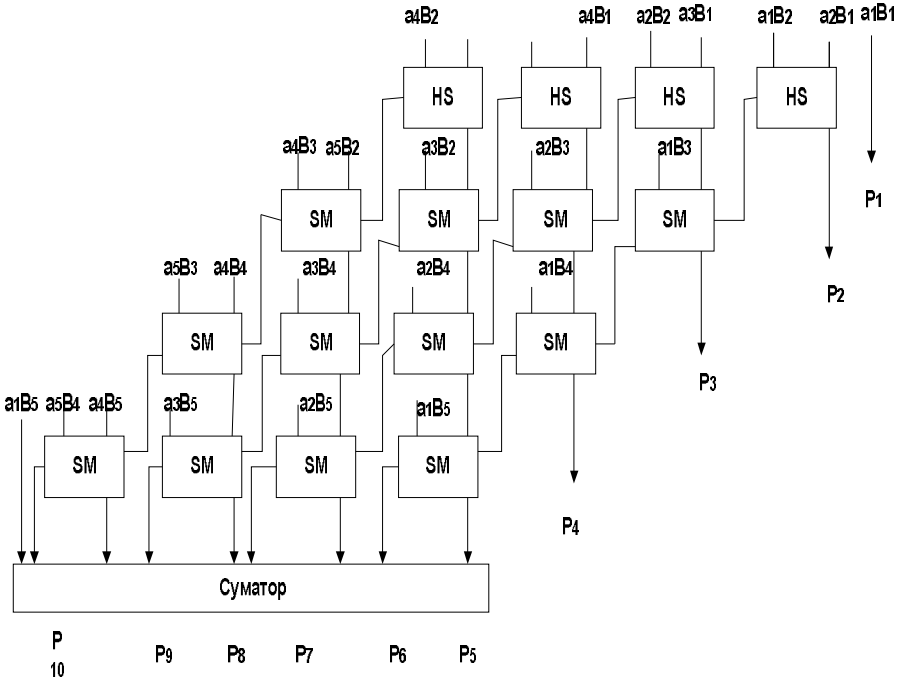


Рисунок 13.1 – Структурна схема пристрою множення для реалізації матричного алгоритму

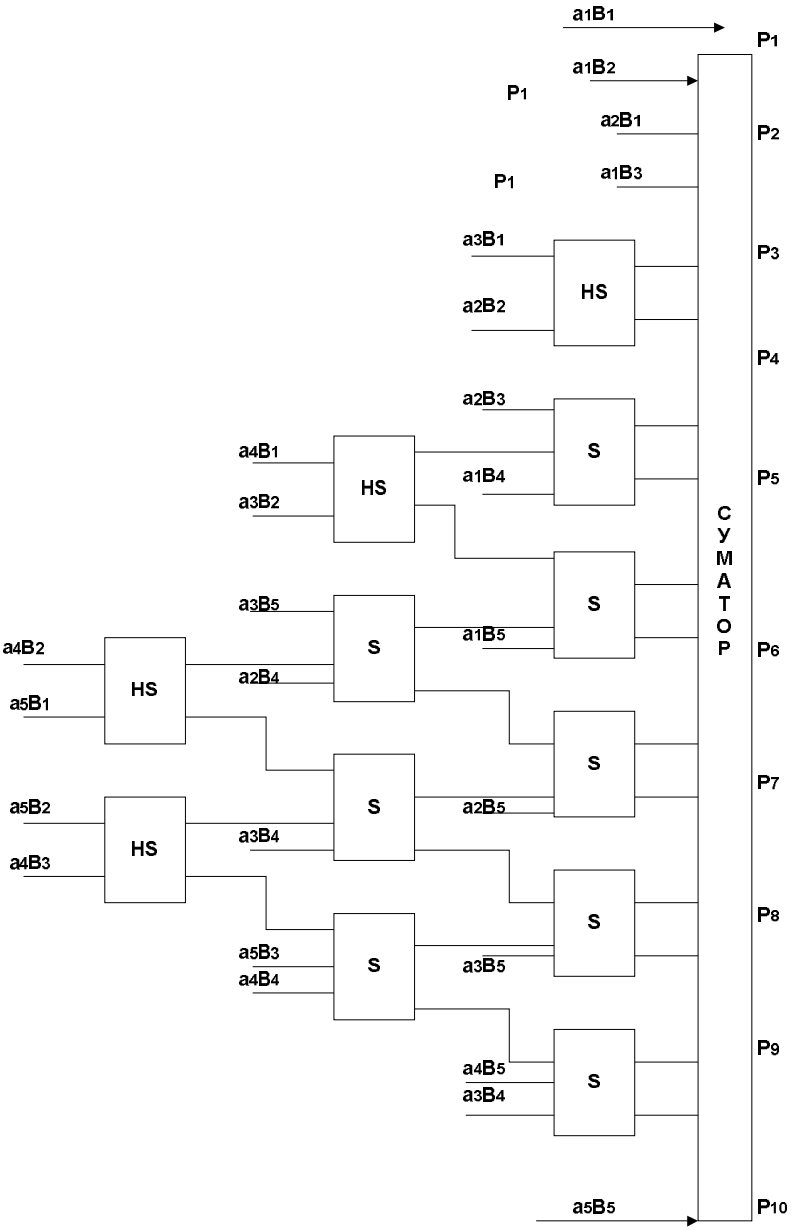


Рисунок 13.2 – Схема множного пристрою за алгоритмом Дадда

Найбільше поширення одержали алгоритми Дадда, Уоллеса (матричний) і алгоритм збереження переносів [1-21]. На рис. 13.1 представлена структурна схема матричного пристрою множення для реалізації матричного алгоритму.

Алгоритми відрізняються друг від друга групуванням часткових добут-ків, кількістю етапів перетворень. Для матричного алгоритму – чотири етапи перетворення, для алгоритму Дадда – три етапи. Однак, матричні методи множення, незважаючи на більший об'єм устаткування, дають великий виграш часу, а використання ВІС значно знижує обмеження на устаткування.

Розглянуті методи множення знайшли широке застосування в практиці бінарної арифметики.

14 ДІЛЕННЯ БІНАРНИХ ЧИСЕЛ

14.1 Методи ділення бінарних чисел

Найбільше поширення одержали наступні методи виконання операції ділення чисел:

1. На кожному кроці з діленого віднімається дільник (починаючи зі старших розрядів) стільки разів, скільки це можливо до одержання залишку менше дільника. У частці записується цифра, рівна числу цілих частин дільника N. L – залишок. A: B= $N \cdot B+L$

2. Інший метод ділення полягає в множенні діленого на обернену величину $3=A:B=A \cdot B^{-1}=A \cdot 1/B$.

Тут виникає проблема знаходження обернених величин шляхом розкладання в біноминальний ряд Ньютона, використовуючи модель розкладання бінома ступеня m:

$$(Z_1+Z_2)^m=\sum_{k=0}^{m} C_m^k * Z_1^{m-k} * Z_2^k = Z_1^m + \frac{m}{1!} * Z_1^{m-1} * Z_2 + \frac{m(m-1)}{2!} * Z_1^{m-2} * Z_2^2 + \ldots + Z_2^m$$

де C^k_m - біномінальні коефіцієнти число різних сполучень із m різних змінних по k у кожному сполученні:

$$C_m^k = \frac{m!(m-k)!}{k!}$$

3. У третьому методі використовують наближені формули знаходження частки, які зводять операцію розподілу до операції додавання, вирахування, множення.

Перший розглянутий метод відносять до “шкільних” алгоритмів ділення з відновленням залишку. Розглянемо приклад розподілу в бінарній арифметиці.

Приклад: $A_м$=00/1100100. ($100_{(10)}$); $B_м$=00/1010. ($10_{(10)}$). Рішення наведене нижче (використано правила бінарного віднімання).

Правила віднімання

C_i– результат віднімання в i-му розряді;

Z_i – запозичення старшому розряді

$$C_i = \begin{cases} a_i - b_i, \text{ при } a_i \geq b_i \\ q + a_i - b_i, \text{ при } a_i < b_i \end{cases}$$

$$Z_i = \begin{cases} 0, \text{ при } a_i > b_i \\ 1, \text{ при } a_i < b_i \end{cases}$$

14.1.1 Алгоритм ділення з відновленням залишку

Формально алгоритм описується в такий спосіб. Нехай A – ділене, B – дільник, C – частка: $A=0,\alpha_1\alpha_2\ldots\alpha_n$; $B=0,b_1b_2\ldots b_m$, $C=0,c_1c_2\ldots c_r$.

На кожному кроці визначається залишок $A_i=A_{i-1}-B\cdot 2^{-i}$, проводиться аналіз: якщо залишок $A_i>0$, то в старший розряд частки записується C_i=l, робиться лівий зсув і перехід до визначення наступного залишку. Якщо $A_i<0$, то $C_i=0$ і відновлюється залишок $A_i=A_{i-1}+B\cdot 2^{-i}$, а на наступному кроці після зсуву визначається новий залишок і т.д.

Даний алгоритм ділення з відновленням залишку реалізується на двійкових суматорах оберненого (ДСОК) або доповняльного (ДСДК) кодів. Структурна схема наведена на рис. 14.1.

Частковий залишок (частка) виходить в результаті послідовного виконання операції віднімання (із заміною віднімання на додавання в доповняльному коді), тому частіше застосовується суматор ДСДК для алгебраїчного додавання.

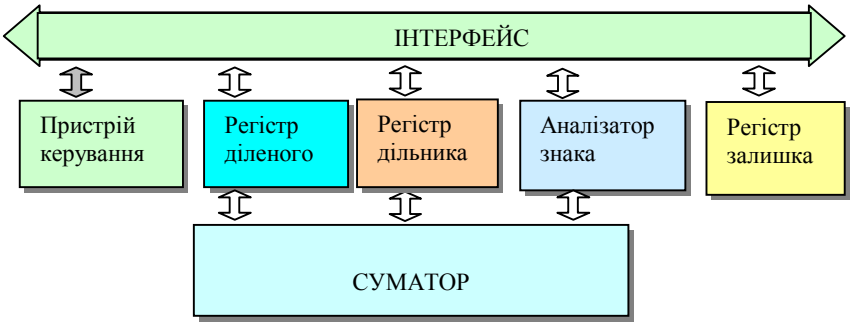


Рисунок 14.1 – Структурна схема ЦА операції ділення

14.2 Ділення чисел з фіксованою комою з відновленням залишку

Ділення виконується за алгоритмом із відновленням залишку на суматорі доповняльного коду (ДСДК) у наступній послідовності:

- визначається знак частки по формулі $Sg_C=Sg_A \oplus Sg_B$;
- проводиться подання діленого і дільника в машинних кодах, коли ділене завжди, незалежно від його знака, береться в прямому коді з позитивним знаком, а дільник завжди, незалежно від його знака, береться у доповняльному коді з негативним знаком;
- усунення дробової частини в дільнику, шляхом переносу коми вправо на *n* розрядів (за аналогією з десятковою системою числення). Щоб дріб не змінився, у діленому також переносять вправо кому на *n* розрядів;
- починаючи зі старших розрядів, до діленого додають дільник у доповняльному коді, що рівнозначно вирахуванню з діленого дільника й аналізують знак проміжного залишку:

1) якщо знак проміжного залишку 00 (позитивний), то в регістр частки Рг*C* записується 1, починаючи зі старшого розряду. Залишок зсувається на один розряд вліво (просто знакову крапку перенести вправо на один розряд), зноситься наступний розряд діленого (що не брав участь до цього в ділені). Після цього, проміжний залишок підготовлений до наступного додатка діленого у доповняльному коді;

2) якщо знак проміжного залишку 11 (негативний), то в регістр частки Рг*C* записується 0, починаючи зі старшого розряду. Залишок відновлюється шляхом додатка до нього дільника в прямому коді з

позитивним знаком. Відновлений залишок зсувається вліво на один розряд (крапку, що відокремлює знак, перенести вправо на один розряд), зноситься наступний розряд діленого (що не брав участь до цього в ділені). Після цього, проміжний залишок підготовлений до наступного додатка дільника у доповняльному коді;

- дії попередніх пунктів повторюються до одержання машинного нуля або заданої точності обчислення (кількість розрядів дробу після коми за цілою частиною числа). Кома дробу встановлюється в частці *C* після зносу останнього розряду цілої частини діленого.

- результат ділення представлений у регістрі частки *C* в прямому коді.

Знак результату привласнюється відповідно до пункту 1.

Приклад. Розділити число A= –100111(–39) на B= –11(–3), представлених у (ФФЗ) . Розрядність суматора і регістрів дорівнює 6.

Рішення: Операція робиться на ДСДК (модифікований код) за алгоритмом із відновленням залишку.

1. Визначаємо знак частки: $Sg_C=Sg_A \oplus Sg_B$ =1⊕1=0. Знак позитивний.

2. Записуємо машинні зображення чисел: $A^{м}_{пр}$=00/100111; $B^{м}_{пр}$=11/11; $B^{м}_{доп}$=11/01;

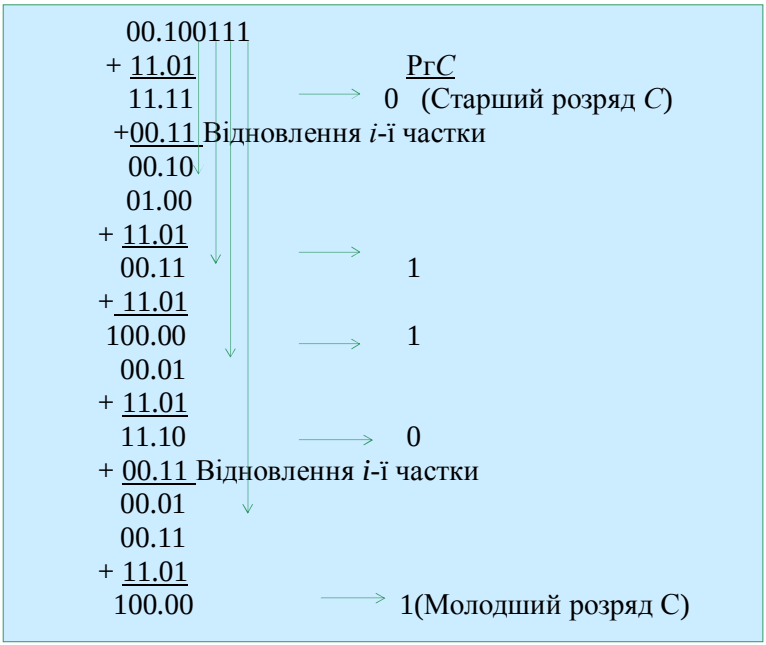
3. Виконуємо послідовність дій над числами за методом алгоритму з відновленням залишку (таблиця 14.1).

Примітка: ДСДК не використовує перенос $П_{Sg1}$ при переповненні суматора старшого розряду знака (див. рис. 8.1).

Відповідь: $C_{пр}$=11/01101 (–13).

У таблиці 14.1 вертикальними стрілками показані знесення наступних розрядів діленого в проміжні залишки. Горизонтальні стрілки відображають розряди запису результату в регістр частки Рг*C*. Одиниці переповнення, що одержані після підсумовування, пропадають, тому що використається ДСДК. При виконанні операції лівого зсуву залишків, переноситься вправо крапка, що відокремлює знак числа(при цьому, старший розряд знака суматора *Sg* пропадає).

Таблиця 14.1 – Приклад ділення чисел A на B з відновленням залишку



Відповідь: $C_{пр}$=11/01101 (–13).

15 ДІЛЕННЯ ЧИСЕЛ З ФІКСОВАНОЮ КОМОЮ БЕЗ ВІДНОВЛЕННЯ ЗАЛИШКУ

Аналізуючи попередній алгоритм, бачимо, що у випадку, коли C_i – чергова цифра частки дорівнює "0", то проводиться відновлення залишку по формулі $A_{i-1} = A_i + B \cdot 2^{-i}$ (на i-му кроці алгоритм ділення $A_i = A_{i-1} - B \cdot 2^{-i}$, де A_{i-1} – попередній залишок, A_i – поточний залишок). Відновлений залишок приймається за A'_i і процес ділення триває, тобто:

$$A_{i+1}=A'_i - B \cdot 2^{-(i+1)} =(A_i+B \cdot 2^{-i})-B \cdot 2^{-(i+1)}$$

Перетворюючи цей вираз, одержимо:

$$A_{i+1}=A_i+B \cdot 2^{-(i+1)} . \qquad (15.1)$$

З формули 15.1 витікає, що відновлювати залишок немає необхідності. Це дає можливість одержати інший метод ділення бінарних чисел, що і розглядається далі.

15.1 Алгоритм ділення без відновлення залишку

Метод ділення бінарних чисел без відновлення проміжних залишків виконується в послідовності:

- визначити знак частки за формулою: $Sg_C=Sg_A \oplus Sg_B$;
- представити числа (операнди) у доповняльному коді в машинному зображенні, ділене (завжди), незалежно від його знака, береться в прямому коді з позитивним знаком, а дільник (завжди), незалежно від його знака, береться в доповняльному коді з негативним знаком;
- привласнити суматору значення См:=$A^м_{доп}$; РгB:=$B^м_{доп}$; РгC:=0;
- усунути дробову частину в дільнику, переносячи кому вправо на n розрядів (за аналогією з десятковою системою числення) і, щоб дріб не змінився, у діленому також перенести вправо кому на n розрядів;
- починаючи зі старших розрядів, до діленого додають дільник у доповняльному коді, що рівнозначно вирахуванню з діленого дільника й аналізують знак проміжного залишку:

1) якщо знак проміжного залишку 00 (позитивний), то в регістр частки РгC записується 1, починаючи зі старшого розряду. Залишок

зсувається на один розряд вліво (знакову крапку перенести вправо на один розряд), зноситься наступний розряд діленого, що не брав участі до цього в діленні. Після цього, проміжний залишок підготовлений до наступного додатка діленого у доповняльному коді;

2) якщо знак проміжного залишку 11 (негативний), то в регістр частки РгС записується 0, починаючи зі старшого розряду. Залишок зсувається на один розряд вліво (знакову крапку перенести вправо на один розряд), зноситься наступний розряд діленого, що не брав участі до цього в діленні. Після цього, проміжний залишок підготовлений до наступного додавання до нього дільника в прямому коді зі знаком 00;

- дії попереднього пункту повторюються до одержання машинного нуля або заданої точності обчислення (кількість розрядів дробу після коми цілої частини числа). Кома дробу встановлюється в частці після зносу останнього розряду цілої частини діленого.

- знак результату привласнюється відповідно до пункту 1. Результат ділення представлений у регістрі частки в прямому коді.

Приклад 1. Розділити на ДСДК числа (ФФЗ): A=16,25$_{(10)}$; B=–3,25$_{(10)}$

Рішення: виконуємо ділення методом без відновлення залишку.

- визначаємо знак частки С: Sg_0C=0 ⊕ 1=1. У старші розряди регістра частки C заносимо значення негативного знака 11/;

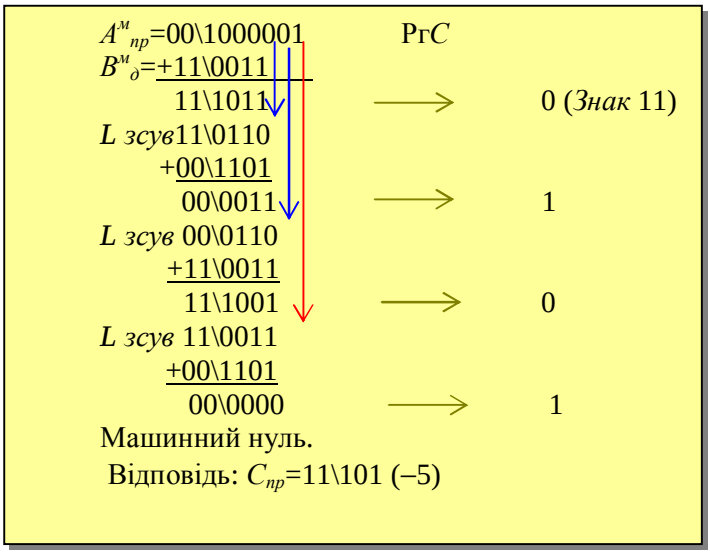
- встановлюємо регістри РгA, РгB і См у нульовий стан, очистивши їх від попередньої інформації;

- переводимо десяткові числа в бінарні, прямі і доповняльні коди (п.15.1):

$A_{пр}$=00\10000,01 (+16,25); $B_{пр}$=11\11,01 (–3,25); $B^{м}_{доп}$=11\00,11; $B_{пр.від.}$=00\11,01. Для вирахування використаємо $B^{м}_{доп}$=11\00,11. Для дільника в прямому коді зі знаком «+» беремо $B^{м}_{пр}$=00\11,01. (Щоб провести ділення з операндами А і В, як з цілими числами, перенесемо зап`яту на два розряди вліво в А і В).

Для порівняння кількості кроків при виконанні операції ділення проведіть самостійно операцію ділення на ДСДК вказаних чисел A= +16,25$_{(10)}$ і B= –3,25$_{(10)}$ за алгоритмом без відновлення залишку.

Рішення наведене по алгоритму без відновленням залишку.



15.2 Ділення чисел з рухомою комою

При діленні чисел представлених у ФРК, ділення виконують над мантиссою $m_C=m_A/m_B$, а порядки віднімаються:

$P_C=P_A-P_B$ (15.2)

При цьому можливі два випадки:

- мантиса $|m_A|\geq|m_B|$;
- мантиса $|m_A|<|m_B|$.

Розподіл мантис виробляється в такому ж порядку, як і у форматі з фіксованою комою. При цьому, використаються методи з відновленням і без відновлення залишку. Результату привласнюється порядок P_C.

Примітка. В алгоритмах ділення застосовується правило: на першому кроці підготовки операндів A і B до ділення робиться так, щоб $|A|$ завжди було менше $|B|$ (тимчасово змінюючи вимогу нормалізації операнда A і, відповідно, корегуючи значення порядка P на +1) для того, щоб при першому кроці операції ділення не **було переповнення молодшого знакового розряду Sg_2**: порівняйте дії над нормалізованими двійковими операндами, наданими у ФРК: $A=+0{,}101\cdot2^{+1}$ і $B=+0{,}101\cdot2^{+1}$:

а) ділене $A^{м}_{пр}$=00\,1010\00\01\; дільник $B^{м}_{пр}$= 00\,1010\00\01; частка $C^{м}_{пр}$=01\,0000\00\00\– отримали ненормоване число і переповнення знакового розряду (тоді, згідно вимог g і **δ**, треба робити зсув **на першому же кроці алгоритму ділення**, що затримує процес у часі);

б) ділене $A^{м}_{пр}$=00\,0101\00\10\; дільник $B^{м}_{пр}$=00\,1010\00\01\; частка $C^{м}_{пр}$= 00\,1000\00\01\–отримали нормоване число С **автоматично** без переповнення знакового розряду Sg_2, бо вибрано $|A| < |B|$, (ф-ла 15.2).

6 БІНАРНО – КОДОВАНІ ДЕСЯТКОВІ СИСТЕМИ ЧИСЛЕННЯ

16.1 Загальні вимоги до БКДС

Візьмемо будь-яке десяткове число, наприклад, 689 і представимо кожний його розряд у бінарній (двійковій) системі числення. Тоді, 689=0110.1000.1001. Якщо кожний десятковий розряд представляти бінарною **тетрадою** (чотири молодших розряди цілих двійкових чисел), то будь-яке десяткове число буде представлене в бінарно-кодованій десятковій системі числення (БКДС). Узагальнюючи приклад для будь-якої системи числення, можна сказати, що будь-яке число A, представлене цифрами з основою $B \neq 2^k$, може бути записане в бінарно-кодованій системі числення (БКС) як:

$$A_{БКС} = \pm\Sigma_i(\Sigma_\ell d_\ell q_\ell)\cdot B^{p-1},$$

де d – бінарні розряди; q – вага кожного бінарного розряду. Вираз в дужках представляє ℓ-й розряд числа A, вага якого B, у свою чергу залежить від місця розташування в числі B^{p-1}.

На практиці в інформаційних системах найбільш широке поширення одержали БКДС (основні їх типи представлені в таблиці 16.1).

Взагалі, можна знайти багато варіантів і способів кодування десяткових чисел. Однак, для того щоб у БКДС можна було виконувати арифметичні операції, ефективно проводити кодування – декодування, необхідно, щоб вони відповідали ряду основних вимог:

- ***одиничності***, тобто кожна десяткова цифра повинна представлятися єдиною бінарною комбінацією, кодом;

- ***впорядкованості***, тобто більшій десятковій цифрі повинна відповідати більша двійкова, що забезпечує ефективність операції порівняння чисел;

- ***парності***, що полягає в тому, що парній (непарній) десятковій цифрі повинні відповідати парні (непарні) двійкові. Це забезпечує ефективність операцій округлення, множення, ділення;

- ***доповняльності*** – сума прямого і оберненого двійкового коду будь-якої десяткової цифри повинна рівнятися коду числа 9;

- ***зваженості*** (відповідності ваг) для двійкових і десяткових розрядів.

Таблиця 16.1 – Основні типи БКДС

Десяткове число	ЕКВІВАЛЕНТИ ДЕСЯТКОВИХ ЦИФР У КОДАХ БКДС								
	8421	8421+3	2421	7421	5211	5421	8-4-21	2 з 5	84-2-1
0	0000	0011	0000	0000	0000	0000	0000	11000	0000
1	0001	0100	0001	0001	0001	0001	0001	00011	0111
2	0010	0101	0010	0010	0011	0010	1110	00101	0110
3	0011	0110	0011	0011	0101	0011	1111	00110	0101
4	0100	0111	0100	0100	0111	0100	1100	01001	0100
5	0101	1000	1011	0101	1000	1000	1101	01010	1011
6	0110	1001	1100	0110	1010	1001	1010	01100	1010
7	0111	1010	1101	1000	1100	1010	1011	10001	1001
8	1000	1011	1110	1001	1110	1011	1000	10010	1000
9	1001	1100	1111	1010	1111	1100	1001	10100	1111

Перераховані обмеження дозволяють значно спростити виконання арифметичних операцій. Істотним у цьому випадку є простота подання інверсних кодів і простота виділення сигналу переносу з десяткового розряду при виконанні операції додавання.

16.2 Характеристики бінарно (двійково)-десяткових кодів

Розглянемо основні характеристики і область застосування двійково-десяткових кодів (ДДК) систем числення, наведених у таблиці 16.1.

16.2.1 Код із природними вагами 8421

У цьому коді кожна десяткова цифра представлена у звичайній бінарній системі числення. Використовується код при перекладі чисел з десяткової системи у двійкову, індикації роботи регістрів, суматорів, лі-чильників і ін. пристроїв. В арифметичних операціях застосовується рід-ко через труднощі виділення переносів і одержання доповняльності до 9.

Приклад. Записати число 765 у ДДК 8421.

Рішення. Записується кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=0111.0110.0101

16.2.2 Код 8421+3 (код з надлишком 3)

У цьому коді до кожної десяткової цифри додається 3. Код не відповідає вимозі виваженості. Застосовують його найчастіше в десятковій арифметиці, тому що легко виділяється десятковий перенос.

Приклад. Записати число 765 у ДДК 8421+3.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів і надлишком +3: 765=1010.1001.1000

16.2.3 Код 2421

Код 2421 (аналогічний 4221) задовольняє всім п’яти вимогам. Однак, не має властивість адитивності, що утруднює виконання арифметичних операцій. Застосовувався в закордонній ЕОМ «МАРКО-3».

Приклад. Записати число 765 у ДДК 2421.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=1101.110.1011.

16.2.4 Код 7421

У цьому коді будь-яка цифра зображується не більш ніж із двома одиницями, що знижує споживання струму. Не відповідає вимозі доповняльності.

Приклад. Записати число 765 у ДДК 7421.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=1000.0110.0101.

16.2.5 Код 5211

Стверджується, що пристрої (лічильники, регістри і ін.) побудовані на основі цього коду, більш економічні за устаткуванням. Застосовувався в ЕЦОМ типу «ПРОМІНЬ».

Приклад. Записати число 765 у ДДК 5211.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=1100.1010.1000.

16.2.6 Код 5421

Цей код використається для прискорення виконання деяких арифметичних операцій. Наприклад, при множенні на цифри 2 і 5 переніс поширюється на один розряд вперед, а при множенні на 4 - на два розряди вперед. Це дозволяє організувати множення на ці цифри апаратними засобами зі схемною реалізацією.

Приклад. Записати число 765 у ДДК-5421.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=1010.1001.1000.

16.2.7 Код 8-4-21

У цьому коді число, що перевищує дев’ять, виходить за межі тетради, що прискорює формування переносу при виконанні операцій підсумовування.

Приклад. Записати число 765 у ДДК 8-4-21.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів і їхніх знаків: 765=1011.1010.1101.

16.2.8 Код 2 з 5 (7421mod2)

Цей код виходить із коду 7421 шляхом додавання праворуч п’ятого розряду, куди проставляється сума цифр основної тетради, узятої по mod 2, за винятком цифри 0. Цифра 0 має штучне зображення 11000.

Відмінність коду полягає в тому, що всі двійкові еквіваленти десяткових цифр мають дві одиниці. Такі коди називають рівноважними, їх використовують для виявлення асиметричних, одиночних помилок. Імовірність компенсованих (подвійних) помилок надзвичайно мала. Ще однією достойністю коду є рівномірне навантаження на джерело живлення (використано в ЕОМ IBM-7070).

Приклад. Записати число 765 у ДДК 2 з 5.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів: 765=10001.01100.01010.

16.2.9 Код 84-2-1

Код 84-2-1 добре формує переноси в старший розряд при порозрядному додаванні, тому що вже цифра 9 має у всіх розрядах одиниці – 1111.

Приклад. Записати число 765 у ДДК 84-2-1.

Рішення. Записуємо кожний десятковий розряд числа 765 у двійковому коді з урахуванням вагових розрядів і їхніх знаків: 765=1001.1010.1011.

16.3 Виконання операції додавання в кодах ДДК

Правила операції додавання будемо розглядати тільки для конкретних кодів, тому що вони не можуть бути універсальними (тобто відповідати всім вимогам п. 16.1) через те, що кожний тип кодів має свою конкретну специфіку.

Нехай задані десяткові числа: $A=a_1a_2\ldots a_n$, $B=b_1b_2\ldots b_n$, і результат операції їх додавання $C=c_0c_1\ldots c_n$, де a_i, b_i, c_i - десяткові цифри, представлені бінарними кодами тетрад.

Операція додавання виконується порозрядно з урахуванням додавання до i-ого розряду переносу переповнення з попереднього молодшого розряду p_{i+1} і відрахування переповнення p_i при переносі його в старший p_{i-1} розряд:

$$c_i=a_i+в_i+p_{i+1}-(p_i\cdot 10).$$

В окремому випадку, переносів переповнення може і не бути.

16.3.1 Операція додавання в ДДК 8421

При додаванні десяткових чисел у ДДК 8421 потетрадно, виконується бінарне, порозрядне додавання, при цьому можуть виникнути наступні випадки.

1. Коли $c_i < 10$, тому переповнення немає і перенос у наступний розряд не виникає, тобто: $c_i=a_i+в_i+p_{i+1} < 10$.

Приклад.
+3 (0011)
+5 (0101)
+8 (1000).

2. Коли $c_i \geq 10$. Тут виникає десятковий перенос і сума повинна бути дорівнює $c_i=a_i+b_i+p_{i+1}-(p_i \cdot 10)$. Однак, якщо перенос виділився автоматично тобто $c_i \geq 16$, то в наступний розряд йде перенос рівний 16, а не 10. Якщо ж перенос не виділився автоматично, тобто $10 \leq c_i \leq 15$, то для c_i вийшла заборонена комбінація. В обох випадках до суми необхідно додавати корекцію 6 (0110), що автоматично усуває заборонені комбінації. При цьому, в i-ому розряді залишається необхідне значення двійкового коду десяткового числа, а в старший розряд переходить двійкова одиниця еквівалентна 10. Приклади.

5= 0101 +5= 0101 10= 1010 +6= 0110 1.0000	7=0111 +8=1000 15=1111 +6=0110 10101	679 = 0110.0111.1001 +465 = 0100.0110.0101 1010.1101.1110 +0110.0110.0110 144=1.0001.0100.0100

16.3.2 Операція додавання в ДДК 8421+3

У цьому коді при двійковому додаванні також можуть бути два випадки.

1. Сума розряду $c_i=a_i+b_i+p_{i+1}<10$. Тому що цифри a_i і b_i вже мали надлишок три кожна, то сума буде мати надлишок шість. Тому, для одержання правильного результату, із суми необхідно відняти три, а три залишаться як надлишок +3. У машинних алгоритмах замість вирахування трьох до суми додають число 13 (1101), тому що 13=16–3, і тим самим блокують поширення переносу p_i=16, що рівноцінно вирахуванню 16.

Приклади.

3 = 0110 +5 = 1000 1110 +13 =1101 8 =1011	3 = 0110 +6 = 1001 1111 +13 1101 9 =1100

2. У цьому випадку сума розряду перебуває в межах $10 \le c_i = a_i + b_i + p_{i+1} \le 19$, тому перенос формується автоматично. У наступний старший розряд переходить перенос $p_i = 16$, тому, для відновлення надмірності коду, до отриманої суми необхідно додати три.

Цей код використовувався в деяких моделях машин ЄС при виконанні операції додавання.

Приклади.

```
6 = 1001
+7 = 1010
     1.0011
   + 0011
13= 1.0110
```

```
   3862 = 0110.1011.1001.0101
+5974 = 1000.1100.1010.0111
  1111.1000.0011.1100
+        1101.0011.0011.1101
  9836 = 1100.1011.0110.1001
```

16.3.3 Операція додавання в ДДК 2421

Таблиця кодування цього коду наче би ділиться на дві частини: від 0 до 4 йдуть нормальні двійкові числа, починаючи з 5, кожна тетрада містить надлишок 0110.

При додаванні можуть виникнути три випадки.

1. Якщо складають цифри, що, менше 5 і результат додавання теж менше 5, то корекції не потрібно, тому що він представлений у звичайній бінарній системі.

Якщо $c_i = a_i + b_i + p_{i+1} \ge 5$, то результат повинен бути збільшений на шість (0110).

Ознакою даної корекції є одержання заборонної комбінації при додаванні. Приклади.

```
 1 = 0001
+2 = 0010
 3 = 0011
```

```
 3 = 0011
+4 = 0100
 1 = 0001
      1000
      0110
 8 =  1110
```

2. Якщо один з додатків більше 4 і результат перебуває в діапазоні $15>a_i+b_i+p_{i+1}\geq 5$, то корекція не потрібна, тому що в ньому є надлишок 6.

3. Якщо обидва з додатків більше 4 і $10\leq a_i+b_i+p_{i+1}<15$, то з результату треба відняти 6 (0110), якщо ж $a_i+b_i+p_{i+1}\geq 15$, то корекція не потрібна. Звичайно, вирахування шістки міняють на додавання десяти (1010) з відкиданням виниклого переносу.

Приклади.

```
 9 =  1111
+3 =  0011
12 =10010
```

```
 6 = 1100
+8 = 1110
  11010
  + 1010
14=10100
```

```
 7279 = 1101.0010.1101.1111
+3658 = 0011.1100.1011.1110
       1.0000.1111.1001.1101
     + 0000.0000.1010.0000
10937 =1.0000.1111.0011.1101
```

17 АРИФМЕТИЧНІ ОПЕРАЦІЇ В СИСТЕМІ ЗАЛИШКОВИХ КЛАСІВ

Розглянемо основні правила виконання арифметичних операцій у системі залишкових класів (див. гл. 2). Нехай задано набор основ: $p_1=3$; $p_2=5$; $p_3=7$.

Тоді діапазон чисел, що можна представити буде дорівнювати $P=p_1\cdot p_2\cdot p_3=105$. Числа будемо представляти в залишках основ $A=(a_1,a_2,\ldots,a_n)$, $B=(b_1,b_2,\ldots,b_n)$, при цьому n дорівнює кількості узятих основ-тобто трьом.

Результат будь-якої операції позначимо $A\cdot B=C$, де $C=(c_1,c_2,\ldots,c_n)$. При цьому, $0<A<P$, $0<B<P$, $0<C<P$.

При виконанні операцій у СЗК, необхідно враховувати наступні особливості:

- між розрядами СЗК відсутні зв’язки;
- всі операції виконуються порозрядно і обчислення можна проводити паралельно;
- для кожного розряду СЗК використають малорозрядні суматори, що працюють по модулю p_i;
- є можливість використати замість суматорів прості таблиці результатів розрядів СЗК.

17.1 Додавання чисел у СЗК

Правило додавання двох чисел представлених у СЗК можна сформулювати в такий спосіб:

- числа складаються по розрядах основ, підсумуються їхні залишки;

- з отриманої суми залишків, віднімається основа цього розряду до одержання вирахування (залишку) результату цієї основи.

Формально це можна представити як: $C=A+B$, або $(c_1,c_2,\ldots,c_n) = (a_1,a_2,\ldots,a_n)+(b_1,b_2,\ldots, b_n)$, де кожний розряд дорівнює:

$$c_i=(a_i+b_i)-кр_i,$$

де $к$=0, 1, 2 – ціле число раз вирахування основи до одержання залишку.

Приклад 1. Нехай p_1=3, p_2=5, p_3=7, P=105. Скласти числа A=67, B=13.

Рішення: Представимо числа в СЗК, де залишки записані в десятковій системі. Тоді A=67=(1,2,4), B=13=(1,3,6). Проведемо додавання по незалежних розрядах, тоді $c_1=1+1=2<p_1=3$, тому c_1=2;

$c_2=2+3=5=p_2$, тому $c_2=5-5=0$;

$c_3=4+6=10>7=p_3$, тому $c_3=10-7=3$. Тоді:

A = 67 = (1,2,4)
+B = 13 = (1,3,6)
C = 80 = (2,0,3)

Приклад 2. Нехай p_1=7, p_2=9, p_3=11, p_4=13, p_5=17. Скласти десяткові числа.

Рішення:

A = 39031 = (6,7,3,5,16)
+B = 67556 = (6,2,5,8,15)
C= 106587 = (5,0,8,0,14)

Якщо залишки представити у двійковій системі числення, то і результат одержимо у двійковій системі.

17.2 Вирахування чисел у СЗК

Розглянемо операцію вирахування позитивних десяткових чисел за умови, що зменшуване більше від'ємника. Операція $C=A-B$ реалізується порозрядно в такий спосіб:

$$c_i=a_i-b_i+\kappa p_i,$$

де $\kappa=0$, якщо $a_i>b_i$, і $\kappa=1$, якщо $a_i<b_i$.

Приклад 1. Нехай $p_1=3$, $p_2=5$, $p_3=7$, $P=105$. Десяткові числа $A=19=(1,4,5)$,

$B=13=(1,3,6)$. Потрібно вирахувати: $C=A-B$.

Рішення: $c_1=1-1+0\cdot3=0$, $c_2=4-3+0\cdot5=1$, $c_3 5-6+1\cdot7=6$. Тоді:

$$\begin{array}{r} A = 19 = (1,4,5) \\ \underline{-B = 13 = (1,3,6)} \\ C = 06 = (0,1,6). \end{array}$$

Приклад 2. Нехай $p_1=7$, $p_2=9$, $p_3=11$, $p_4=13$, $p_5=17$, $P=153153$. Десяткові числа $A=67556=(6,2,5,8,15)$; $B=39031=(6,7,3,5,16)$. Потрібно вирахувати: $C=A-B$.

Рішення:

$$\begin{array}{r} A = 67556 = (6,2,5,8,15) \\ \underline{-B = 39031 = (6,7,3,5,16)} \\ C = 28525 = (0,4,2,3,16) \end{array}$$

17.3 Множення чисел у СЗК

Операція множення чисел у СЗК реалізується порозрядно множенням залишків з наступним відрахуванням K раз основи розряду до одержання залишку результату. Формальне множення

$C=A\cdot B$ реалізується за формулою:

$$c_i = a_i\cdot b_i - \kappa p_i,$$

де $\kappa=0,1,2\ldots$; p_i – i-та основа СЗК.

Приклад 1. Нехай $p_1=3$, $p_2=5$, $p_3=7$, $P=105$. Помножити числа $A=4$, $B=13$.

Рішення. Представимо числа в СЗК: $A=4=(1,4,4)$; $B=13=(1,3,6)$.

Визначимо порозрядні добутки по кожній основі.

$c_1 = 1 \cdot 1 = 1$, що менше p_1, тому $c_1 = 1$;

$c_2 = 4 \cdot 3 = 12 > 2 \cdot 5$, тому $c_2 = 12 - (2 \cdot 5) = 2$;

$c_3 = 4 \cdot 6 = 24 > 3 \cdot 7$, тому $c_3 = 24 - (3 \cdot 7) = 3$.

Тоді, $A \cdot B = (1,4,4) \cdot (1,3,6) = (1,2,3) = 52$

Взявши ортогональні базиси $B_1=70$, $B_2=21$, $B_3=15$ (див. гл. 2.2), перевіримо: $70+2 \cdot 21+3 \cdot 15=157-105=52$.

Приклад 2. Нехай $p_1=7$, $p_2=9$, $p_3=11$, $p_4=13$, $p_5=17$, $P=153153$. Помножити десяткові числа.

Рішення:

A = 407 = (1,2,0,4,16)
*B = 279 = (6,0,4,6,7)
C = 113553 = (6,0,0,11,10)

Самостійно визначити ортогональний базис і перевірити рішення.

17.4 Ділення чисел у СЗК

Операцію ділення будемо виконувати тільки із цілими позитивними числами, що мають цілу частку C. При цьому умови алгоритму операції ділення $C = A/B$ наступні. Ділення виконується порозрядно у відповідності з формулою:

$$\tilde{n}_i = \frac{\grave{a}_i + \kappa\partial_i}{b_i},$$

де $\kappa = 0,1,\ldots,p_i - 1$.

При цьому, $b \neq 0$, а κ вибирається так, щоб не було залишку при розподілі.

Приклад 1. Нехай $p_1=3$, $p_2=5$, $p_3=7$, $P=105$. Необхідно розділити числа $A=64$, $B=8$.

Рішення. Запишемо числа в СЗК. $A=64=(1,4,1)$; $B=8=(2,3,1)$.

Визначимо порозрядні частки:

$c_1 = (1 + \kappa \cdot 3)/2 = (1+1 \cdot 3)/2 = 2$ ($\kappa = 1$);

$c_2 = (4 + \kappa \cdot 5)/3 = (4+1 \cdot 5)/3 = 3$ ($\kappa = 1$);

$c_3 = (1 + \kappa \cdot 7)/1 = (1+0 \cdot 7)/1 = 1$ ($\kappa = 0$). Тоді,

A = 64 = (1,4,1)
: B = 8 = (2,3,1)
C = 8 = (2,3,1)

Приклад 2. Нехай p_1=7, p_2=9, p_3=11, p_4=13, p_5=17. Розділити десяткові числа.

Рішення:

A = 18848 = (4,2,5,11,12)
: B = 152 = (5,8,9,9,16)
C = 124 = (5,7,3,7,5),
K = 3 6 2 4 4

Самостійно визначити ортогональний базис і перевірити рішення. ***Особливі випадки.*** Якщо b_i=0, то виникає невизначеність $A/0$. ЦА у цих випадках виробляє сигнал невизначеності і зупиняє рішення.

ПЕРЕЛІК ПОСИЛАННЬ

1. Бабич М.П., Жуков І.А. Комп`ютерна схемотехніка.-Київ.:МК-Прес, 2004.-412 с.: іл.

2. Буняк А. Електроніка та мікросхемотехніка.-Київ-Тернопіль: СМП «Астон», 2001.-382 с: іл.

3. Угрюмов Е. П. Цифровая схемотехника.- СПб.: БХВ – Санкт-Петербург, 2000. – 528 с.: ил.

4. Кудерметов Р.К., Щербаков А.М., Грушко С.С. Прикладна теорія цифрових автоматів. Навчальний посібник.-Запоріжжя, ЗНТУ, 2009.-190 с.: іл.

5. Савельев А. Я. Прикладная теория цифровых автоматов.-М. Высшая школа. 1987-272 с.

6. Акушский И.Я., Юдицкий Д. А. Машинная арифметика в остаточных классах. М. Высшая школа. 1968–453 с.

7. Глушков В. М. Синтез цифровых автоматов. М. Физматгиз. 1962– 475 с.

8. Евреинов Э. В., Бутыльский Ю. Т. Мамзелев И. А. и др. Цифровая и вычислительная техника. -М. Радио и связь. 1991.– 464 с.

9. Кудрявцев В. Б., Алешин С. В. Подколзин А. С. Введение в теорию автоматов. М. Наука. 1985– 320 с.

10. Кузин Л. Т. Основы кибернетики. Том1. М. Энергия. 1973– 504 с.

11. Лихтциндер Б.Я., Кузнецов В. М. Микропроцессоры и вычислительные устройства в радиотехники.-К. Высшая школа. 1988– 315 с.

12. Майоров С. А., Новиков Г. И., Немолочнов О. Ф. и др. Проектирование цифровых вычислительных машин. -М. Высшая школа. 1972– 345 с.

13. Поспелов Д. А. Логические методы анализа и синтеза схем. М. Энергия. 1974–368 с.

14. Поснов Н. Н. Арифметика вычислительных машин в упражнениях и задачах. Минск. Университетское. 1984–226 с.

15. Потемкин И. С. Функциональные узлы цифровой автоматики. М. Энергоатомиздат 1988 –320 с.

16. Савельев А. Я. Арифметические и логические основы цифровых автоматов. -М. Высшая школа, 1980–255 с.

17. Савельев А. Я. Основы информатики. -М. Высшая школа. 1991–235 с.

18. Соловьев В. В. Проектирование цифровых систем на основе программируемых логических интегральных схем. М. Горячая линия. 2001–636 с.

19. Самофалов К. Г. Цифровые ЭВМ. Практикум.-К. Высшая школа. 1989.

20. Самофалов К. Г., Романкевич А.М., Валуйский В.Н., Каневский Ю.С., Пиневич М.М. Прикладная теория цифровых автоматов.-К. Вища школа. 1987-376 с.

21. Шеннон К. Э. Работы по теории информации и кибернетике.-М. Иностранная литература. 1963–830 с.