А.М.Пєтух В.В.Войтко

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

Ч.1

Міністерство освіти і науки України
Вінницький державний технічний університет

А.М.Петух **В.В.Войтко**

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

Затверджено на засіданні Ученої ради Вінницького державного технічного університету як навчальний посібник для студентів спеціальності "Програмне забезпечення автоматизованих систем" Протокол № 5 від 27 грудня 2000 року.

Вінниця ВДТУ 2001

УДК 681.32 (075)
П 31

Рецензенти:
Р.Н.Квєтний, доктор технічних наук, професор
В.П.Кожем'яко, доктор технічних наук, професор
В.І.Сачанюк, голова правління ВАТ "Інфракон"

Рекомендовано до видання Ученою радою Вінницького державного технічного університету Міністерства освіти і науки України

А.М.Пєтух, В.В.Войтко
П 31 **Прикладна теорія цифрових автоматів.** Навчальний посібник. – Вінниця: ВДТУ, 2001 – 77с.

Навчальний посібник "Прикладна теорія цифрових автоматів" призначений для студентів спеціальності "Програмне забезпечення автоматизованих систем", складається з двох частин. У першій частині посібника розглядаються особливості використання різних систем числення, проводиться аналіз різних методів виконання арифметичних операцій у двійковій системі числення з урахуванням апаратних засобів для реалізації операцій в операційному автоматі.

УДК 681.32 (075)

ЗМІСТ

ВСТУП

Дисципліна "Прикладна теорія цифрових автоматів" передбачена навчальним планом для студентів бакалаврського напрямку "Комп'ютерні науки"; базується на курсах: "Основи програмування та алгоритмічних мов", "Організація і функціонування ЕОМ та систем"; є теоретичною основою курсів: "ЕОМ і мікропроцесорні системи", "Методи і засоби комп'ютерних інформаційних технологій", "Схемотехніка ЕОМ". У результаті вивчення дисципліни студенти повинні вміти здійснювати переведення чисел із однієї системи числення в інші; виконувати арифметичні операції над числами з плаваючою та фіксованою комами; розробляти алгоритми виконання основних арифметичних операцій для різних систем числення в різних кодах з оцінкою точності; виконувати перетворення булевих функцій в різних базисах; проводити мінімізацію абстрактних та структурних цифрових автоматів; складати на цій основі структурні схеми комбінаційних цифрових автоматів та автоматів з пам'яттю.

Автори поставили за мету викласти в стислому вигляді основний зміст дисципліни в рамках діючої програми і дати рекомендації для розширеного її опанування. Велика увага приділяється аналізу різних методів виконання арифметичних операцій у двійковій системі числення та алгоритмізації процесу вибору і реалізації арифметичних операцій. В алгоритмах та прикладах враховуються особливості використання апаратних засобів для реалізації арифметичних операцій у процесі розробки операційного та керуючого автоматів.

1 ОСНОВНІ ПОНЯТТЯ ПРО СИСТЕМИ ЧИСЛЕННЯ

Системою числення називається сукупність цифрових знаків і правил їх запису, яка використовується для однозначного зображення чисел. Алфавіт, на основі якого записуються числа в деякій системі числення, складається з кінцевої кількості елементів (цифр).

Зазвичай усі системи числення розбивають на два класи: непозиційні і позиційні. Таке розбиття є умовним, оскільки можливе існування систем, в яких ці два принципи поєднуються (так звані частково–позиційні системи).

Непозиційними називаються такі системи, в яких застосовується необмежена кількість цифр, причому значення кожної цифри не залежить від її позиції в числі. Історично першими системами числення були саме непозиційні системи, одним з основних недоліків яких являється складність запису великих чисел. Непозиційні системи числення сьогодні використовуються нечасто, в основному для нумерації. Прикладом такої системи є римська система числення.

Позиційною системою числення є широко розповсюджена десяткова система. Саме позиційні системи знайшли широке застосування в практичній діяльності людини. Позиційними називаються такі системи, в яких застосовується обмежений набір цифр, причому значення кожної цифри знаходиться в суворій залежності від її позиції в цифровій комбінації. Кількість різних цифр, які застосовуються в даній системі, називається її основою.

1.1 Двійкова система числення

Система числення з основою $N=2$ нічим не відрізняється від позиційної системи числення з будь-якою основою, якщо мова йде про систему числення взагалі. Але для ЕОМ ця система має істотну перевагу, оскільки її алфавіт містить усього два символи: $x_j = \{0,1\}$. Таким чином, будь-яке число в двійковій системі записується у вигляді комбінації нулів та одиниць. Для фіксації цих символів досить мати деякий "пристрій", що містить два істотно різних стійких стани.

Для людини двійкова система, звісно, є громіздкою у використанні. Їй звичною є десяткова система числення, в якій вироблені прийоми запису чисел за «іменем», прийоми додавання, віднімання, множення і ділення будь-яких чисел. У двійковому записі числа важко визначити його значення, оскільки відсутнє поняття «імені» саме двійкового числа, важко зіставити ланцюжок, особливо при великій його довжині, зі змістом. Виникає потреба перетворити двійковий запис у десятковий і навпаки, що не так то і легко. Але ці операції є необхідними тоді, коли людина змушена з якихось причин користуватися двійковою системою поза ЕОМ.

Переведення чисел із двійкової системи числення в десяткову

Переведення числа, поданого в двійковій системі числення, у десяткову найпростіше здійснити, записавши його у вигляді прогресії (1.1) зі знаменником $N=2$ і додавши її члени.

$$S_m N^m + S_{m-1} N^{m-1} + \cdots + S_1 N^1 + S_0 N^0 \qquad (1.1)$$

Приклад 1.1. Перевести ціле двійкове число 101101111_2 у десяткову систему числення.

Записуємо прогресію:

$$1\cdot 2^8 + 0\cdot 2^7 + 1\cdot 2^6 + 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0$$

і підсумовуємо її члени:

$$256+64+32+8+4+2+1=367_{10}$$

Десятковий еквівалент двійкового числа 101101111_2 знайдений і дорівнює 367_{10}. Тут і далі індексом будемо позначати основу системи числення.

Нічим не відрізняється переклад будь-якого змішаного числа з двійкової системи числення в десяткову.

Приклад 1.2. Перевести змішане двійкове число $1010,\ 011_2$ у десяткову систему.

Прогресія буде мати вигляд:

$$1\cdot 2^3+0\cdot 2^2+1\cdot 2^1+0\cdot 2^0+0\cdot 2^{-1}+1\cdot 2^{-2}+1\cdot 2^{-3},$$

а сума її членів складе десяткове число:

$$8+2+0{,}25+0{,}125=10{,}375.$$

Переведення числа з десяткової системи в двійкову

Подання числа у вигляді прогресії зі знаменником, що є основою системи числення, дозволяє виявити спосіб (алгоритм) переведення чисел з однієї системи числення в іншу (з іншим значенням основи). Розглянемо алгоритм переведення числа з десяткової системи числення в двійкову. Для цього приведемо запис прогресії, що являє собою двійкове число, до такого виду:

а) ціла частина числа:

$$((...((S_m 2+S_{m-1})2+S_{m-2})2+...+S_2)2+S_1)2+S_0;$$

б) дробова частина числа:

$$2^{-1}(S+2^{-1}(S_{-2}+2^{-1}(S_{-3}+...+2^{-1}(S_{-(n-1)}+2^{-1}S_{-n}))...)),$$

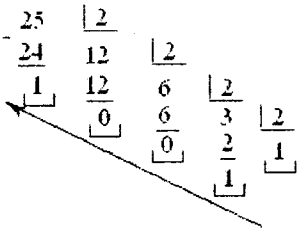
де S - один із символів двійкової системи, тобто 0 чи 1.

З цього запису видно, що для визначення молодшого розряду цілої частини двійкового числа цілу частину десяткового числа необхідно розділити на 2. Залишок від розподілу і буде шуканим значенням цього розряду, тобто S_0. Частка являє собою число, що у двійковій формі можна виразити прогресією

$$((...((S_m 2+S_{m-1})2+S_{m-2})2+...+S_3)2+S_2)2+S_1.$$

Тут знову можна застосувати операцію ділення й отримати наступний розряд двійкового числа. Повторюючи операцію ділення до моменту, коли частка виявиться меншою від основи N (тобто 2), отримуємо всі розряди.

Приклад 1.3. Перевести ціле десяткове число 25 у двійкову систему числення.



Остання частка є самим старшим розрядом двійкового числа. Код двійкового числа записуємо за напрямком стрілки. Таким чином, десятковому числу 25_{10} відповідає двійковий еквівалент 11001_2. Правильність результату можна перевірити переведенням двійкового числа в десяткову систему.

Для переведення дробової частини десяткового числа в дробову частину двійкового, як це видно з перетвореного запису прогресії, дробову частину десяткового числа необхідно помножити на 2 (тобто на основу). Отримаємо розряд S_{-1}. Частина, що залишилася, є дробовою і записується у вигляді:

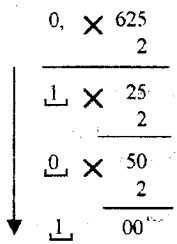
$$2^{-1}(S_{-2} + 2^{-1}(S_{-3} + ... + 2^{-1}(S_{-(n-1)} + 2^{-1}S_{-n}))...)).$$

Якщо її знову помножити на 2, буде отриманий наступний розряд двійкового запису дробового числа.

Багаторазове застосування операції множення дробової частини може привести до ситуації, коли виходить деяка ціла частина і нуль у дробовій. Це означає, що, по-перше, операція переведення завершена, і, по-друге, десяткове число точно переводиться в двійкове. Але дуже часто десятковий дріб точно не переводиться в двійковий. Тоді процес переведення доводиться переривати примусово.

Приклад 1.4. Перевести десятковий дріб 0,625 у двійкову систему.

Останнє множення в дробовій частині дало нуль, отже, процес переведення закінчено. Отримано двійковий дріб $0{,}101_2$. Запис розрядів дробу здійснюється за напрямком стрілки. Приклад 1.4 ілюструє випадок, коли десятковий дріб точно переводиться в двійковий.



Приклад 1.5. Перевести десятковий дріб $0{,}57_{10}$ у двійкову систему.

0,57
2
1,14
2
0,28 (*)
2
0,56
2
1,12
2
0,24
2
0,48
2
0,96
2
1,82
2
1,64
2
1,28 (*)
. . .

Отримано результат, що вказує на періодичність двійкового дробу. Знаком (*) відзначені однакові дробові числа. Подальші операції безглузді. Можна записати двійковий дріб у вигляді періодичного дробу 0,10(01000111). В обчислювальній машині переведення буде продовжуватися до заповнення всіх двійкових розрядів комірки.

Виникає питання точності подання такого дробу. Якщо обмежитися тільки отриманими десятьма розрядами, вони дають десяткове число 0,569335937. Похибка складає 0,000664063. Збільшення кількості двійкових розрядів приведе до істотного зменшення похибки, але вона

завжди буде, що варто обов'язково враховувати при розв'язанні різних інженерних задач.

При необхідності для перевірки правильності переведення десяткового дробу в двійкову систему чи для перевірки отриманої точності здійснюється зворотне переведення. У даному випадку мова йде про роботу з цифровою інформацією поза ЕОМ.

1.2 Вісімкова і шіснадцяткова системи числення

В обчислювальній техніці і програмуванні велике місце займають вісімкова і шіснадцяткова системи числення. У вісімковій системі в якості цифри використовуються символи: 0, 1, 2, 3, 4, 5, 6, 7. У шіснадцятковій системі потрібно 16 символів, у якості яких використовуються арабські цифри і п'ять букв латинського алфавіту, що утворюють послідовність (з урахуванням ваги шіснадцяткових цифр):

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.$$

Переведення чисел з десяткової системи числення у вісімкову чи шіснадцяткову здійснюється за правилом, що викладено для двійкової системи числення. Необхідно тільки пам'ятати, що в шіснадцятковій системі є цифри *A, B, C, D, E, F*, що трохи незвично.

Переведення з вісімкової і шіснадцяткової систем у десяткову найпростіше здійснити записом прогресії зі знаменником, що відповідає основі системи числення.

Приклад 1.6. Шіснадцяткове число $BA52_{16}$ перевести в десяткову систему. Запишемо прогресію, заміняючи символи *B, A* їх десятковими еквівалентами. Тоді прогресія буде мати вигляд:

$$11\cdot16^3 + 10\cdot16^2 + 5\cdot16^1 + 2\cdot16^0.$$

Визначимо суму членів прогресії:

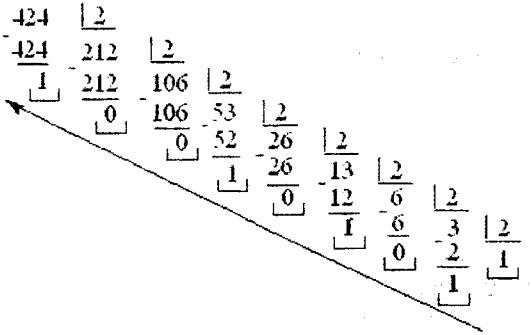
$$11\cdot4096 + 10\cdot256 + 5\cdot16 + 2\cdot1 = 47698_{10}.$$

Шіснадцятковому числу *BA52* відповідає десятковий запис 47698_{10}.

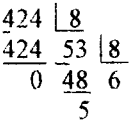
Зручно використовувати шіснадцяткову чи воісімкову систему числення і за необхідності швидкого переведення десяткового числа в двійкову систему і навпаки. Покажемо це на прикладі.

Приклад 1.7. Перевести десяткове число 424 у двійкову і вісімкову системи.

Скористаємося звичайним алгоритмом:



Десятковому числу 424 відповідає двійковий код 110101000. Виконаємо переведення у вісімкову систему:



Отримано вісімкове число $650_8 = 424_{10}$.

Замінимо кожну цифру вісімкового числа трирозрядним двійковим еквівалентом, тобто:

Тепер зрозуміло, що переведення з вісімкової системи числення в двійкову здійснюється простим прийомом: записом кожної вісімкової цифри двійковою триадою. Справді, послідовність 110101000 являє собою той результат, що раніше був отриманий переведенням числа 424_{10} у двійкову систему числення, але для отримання результату там необхідно було виконати вісім операцій ділення. Для переведення у вісімкову систему треба буде всього дві операції ділення. Переведення ж вісімкових цифр у двійкову триаду виробляється подумки. Це має велике значення при необхідності роботи з великими числами при досить великій їх кількості.

В операціях переведення можна користуватися і шіснадцятковою системою.

Приклад 1.8. Двійкове число 110101000 легко перевести в шіснадцяткове. Для цього подамо код у вигляді:

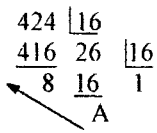
0001 1010 1000

Замінимо кожну четвірку двійкових символів цифрою шіснадцяткової системи. Буде отримана послідовність цифр:

1 A 8

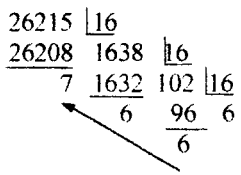
Це і є $1A8_{16} = 424_{10} = 560_8 = 110101000_2$.

Переведення 424_{10} у шіснадцяткову систему можна отримати діленням:

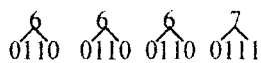


Кількість операцій ділення невелика. На великих десяткових цифрах можна отримати істотну економію по операціях ділення і при переведенні у вісімкову систему. А переведення із шіснадцяткової системи в двійкову дуже просте: кожну цифру шіснадцяткової системи варто замінити чотирирозрядним двійковим еквівалентом.

Приклад 1.9. Десяткове число 26215 перевести в двійкову систему через шіснадцяткову.



Ланцюжок 6667 і є шіснадцяткове число, тобто $6667_{16} = 26215_{10}$. Тепер легко отримати і двійковий код цього числа, замінивши кожну шіснадцяткову цифру двійковою тетрадою:



Отже, $110011001100111_2 = 6667_{16} = 26215_{10}$.

У комірках пам'яті шіснадцяткові числа подаються двійковими кодами. Кожна цифра кодується тетрадою. Важливе значення має і той факт, що над шіснадцятковими числами можна здійснювати операції додавання, віднімання, множення і ділення, використовуючи двійкову арифметику, а потім інтерпретувати отриманий результат у шіснадцятковій формі. Це дозволяє істотно збільшити діапазон подання чисел в оперативній пам'яті обчислювальних машин.

Контрольні запитання і задачі

1. Що розуміють під поняттям «система числення»? Розкрийте зміст позиційної та непозиційної системи числення.
2. Що розуміють під «основою системи числення»?
3. Запишіть числа 123,54 та 5678,098 у вигляді геометричної прогресії з основою 10.
4. Наведіть алгоритм переведення чисел з десяткової системи в іншу систему числення і навпаки.
5. Які ви знаєте методи переведення числа із десяткової системи числення в двійкову?
6. Складіть таблицю, в якій десятковим числам від 0 до 20 поставлені у відповідність числа в двійковій, вісімковій та шіснадцятковій системах числення.
7. Нехай дано числа у двійковій системі числення: 11000.11001; 111000.11000101; 10110.001101. Переведіть їх у вісімкову, шіснадцяткову, десяткову системи числення та зробіть перевірку отриманого результату.
8. Десяткове число 4567,85 переведіть у п'ятіркову та шіснадцяткову системи числення. Зробіть перевірку правильності отриманого результату.
9. Переведіть число $578906{,}456_{10}$ у двійкову систему числення через вісімкову та шіснадцяткову системи.
10. Переведіть двійкове число 1100010101.1111001 у десяткову систему числення через вісімкову та шіснадцяткову системи.

2 ПОДАННЯ ІНФОРМАЦІЇ В ЕОМ

Обчислювальні машини обробляють інформацію різного характеру. Для виконання операцій по запису, зберіганню, передачі, накопиченню, логічних і арифметичних операцій усю інформацію та програми її обробки необхідно подавати в одному вигляді, оскільки ЕОМ має свою специфічну внутрішню мову, на якій тільки і можливе подання інформації. Теорією інформації і кодування створений апарат відображення будь-якої інформації в строго визначену форму внутрішньої мови ЕОМ. За основним законом кодоутворення коди поділяють на два класи: числові (чи арифметичні), які базуються на відповідних системах числення, і комбінаторні (нечислові), які базуються на законах комбінаторики і теорії з'єднань. У будь-якому випадку між інформацією і кодом, що її відображує, існує специфічний зв'язок: кожному елементу інформації ставиться у взаємно однозначну відповідність строго визначена кодова послідовність із кінцевого набору специфічних символів.

2.1 Подання чисел в ЕОМ із фіксованою і плаваючою комою

У цифрових обчислювальних машинах використовують дві форми продання чисел: природну і напівлогарифмічну. Природну форму зображення часто називають поданням з фіксованою комою, а напівлогарифмічну – поданнням з плаваючою комою.

2.1.1 Подання чисел в ЕОМ із фіксованою комою

При зображенні чисел з фіксованою комою кількість розрядів, відведена для запису дробової і цілої частин числа, суворо фіксована. Нехай, наприклад, у машині для запису числа відведено n розрядів, перенумерованих зліва на право, починаючи з 1 до n, а кома «зафіксована» після i-го розряду. Оскільки положення коми у розрядній сітці залишається незмінним для всіх чисел, будь-яке число x може бути подане в такий спосіб:

$$x = \sum_{i=1}^{n} \left[\langle a_i \rangle\right]_j \cdot 2^{k-i}, \qquad (2.1)$$

де i - номер розряду, а $\left[\langle a_i \rangle\right]_j$ може приймати значення або "0", або "1" ($j = 1,2$).

Для того, щоб можна було подавати і негативне число в такій системі числення, необхідно вирішити проблему кодування знаку чисел. У зв'язку з тим, що місце розташування знаку може бути зафіксоване в розрядній сітці машини, а використання спеціальних символів для кодування знаку числа, тобто розширення алфавіту цифр системи числення, є небажаним, у розрядній сітці машини виділяється спеціальний знаковий розряд, в якому знак кодується цифрами з алфавіту тієї ж самої системи числення (рис. 2.1). Звичайно під знак відводиться самий лівий розряд. Знак «плюс» кодується цифрою "0", а «мінус» – "1".

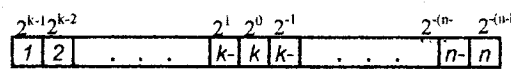


Рисунок 2.1 Розрядна сітка машини з поданням чисел з фіксованою комою

На практиці при поданні чисел із фіксованою комою, кома «закріплюється» або перед самим лівим (у розрядній сітці) розрядом числа, або після самого правого. У першому випадку все число за модулем менше одиниці:

$$|x| = \sum_{i=1}^{n} \left[\langle a_i \rangle \right] \cdot 2^{-i}, \qquad (2.2)$$

а в другому випадку все число ціле:

$$|x| = \sum_{i=1}^{n} \left[\langle a_i \rangle \right] \cdot 2^{n-i}. \qquad (2.3)$$

Таким чином, можуть бути записані числа від $-(1-2^{-n})$ до -2^{n} і від $+2^{n}$ до $+(1-2^{-n})$, а також нуль. Числа, для яких $|x| < 2^{-n}$, не можуть бути подані в такій формі і приймаються рівними нулю. Не можуть бути також подані і числа, для яких $|x| \geq 1$. При поданні, яке відповідає (2.3), крім нуля, можуть бути записані числа, для яких $1 \leq |x| < 2^{n}$.

Форма подання чисел із фіксованою комою має декілька суттєвих недоліків. Насамперед, при організації виконання обчислювальних операцій над числами, поданими в такій формі, усі вихідні дані повинні бути масштабовані, тобто для кожного числа *x* повинен бути введений відповідний масштаб *Mx*, так що множення *Mx·x* потрапляє в прийнятий діапазон зображення чисел. Крім того, при виконанні арифметичних операцій необхідно враховувати, що числа можуть мати різноманітні масштаби і це може призвести до одержання невірних результатів.

Наприклад, якщо все число в машині за модулем менше одиниці, то числа 10,94 і 2,34 можуть бути подані у виді 0,1094 і 0,234 і просте додавання останніх без прийняття додаткових заходів не дає в результаті потрібну суму у відповідному записі, тобто 0,1328. При виконанні операцій над числами, навіть з однаковими масштабними множниками, може бути отримане число, що виявиться поза діапазоном подання чисел. Відбудеться так зване переповнення розрядної сітки. Наприклад, при додаванні чисел 0,412 і 0,731 отримане число перевищує одиницю і не може бути подане в розрядній сітці машини (рис. 2.2,а), і відбувається автоматичне переривання процесу обчислень. Усі ці положення повинні враховуватися людиною, що веде розрахунки на машині.

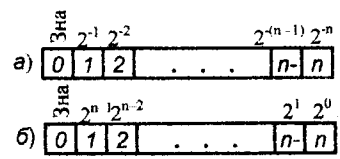


Рисунок 2.2 - Розрядні сітки ЦОМ з фіксованою комою: а) кома фіксована перед першим розрядом; б) кома фіксована після *n*-го розряду

Незважаючи на істотні недоліки, подання чисел із фіксованої комою

знайшло широке застосування, особливо в перших обчислювальних машинах, причому, як правило, у формі (2.2). Це було пов'язано з тим, що подання чисел у природній формі дозволяє спростити схеми машини, забезпечити високу швидкодію арифметичного пристрою. В даний час в універсальних обчислювальних машинах основною є форма подання чисел із плаваючою комою. Проте поряд з останньою застосовується також і подання з фіксованою комою, оскільки операції над числами в такій формі виконуються швидше. Звичайно використовується подання у формі (2.3), тобто виконання операцій із цілими двійковими числами.

2.1.2 Подання чисел в ЕОМ із плаваючою комою

При використанні чисел з плаваючою комою у розрядну сітку машини записується не тільки саме число в деякому масштабі, але і сам масштаб цього числа. У напівлогарифмічній формі подання будь-яке число x має вигляд:

$$x = S^{P_x} \cdot \hat{x}, \tag{2.4}$$

де S – основа системи числення, P_x – ціле число, яке називають порядком масштабу, $\hat{x}$ -мантиса числа, $|\hat{x}| < 1$.

У розрядну сітку числа звичайно записується не сам масштаб S^{P_x}, а тільки порядок P_x і мантиса числа. Наприклад, число $y = 24.12$ у напівлогарифмічній формі в звичайній десятковій системі числення буде мати вигляд:

$$y = 10^2 \cdot 0{,}2412 \text{ або } y = 10^3 \cdot 0{,}02412 \text{ і т.п.} \tag{2.5}$$

Оскільки масштаб числа при його поданні в напівлогарифмічній формі записується в розрядну сітку машини, то операції запам'ятовування, вирівнювання масштабів (порядків) і т.п. виконуються машиною і не потребують особистої участі з боку людини. Правда, після виконання операцій може виникнути потреба в нормалізації мантиси і відповідній зміні порядку. Але ці операції досить просто реалізуються в машині.

При виконанні операцій над числами, поданими у формі з плаваючою комою, можливо одержання результату, порядок якого більший допустимого в машині. Це також варто враховувати при проведенні обчислень. Наприклад, нехай потрібно обчислити $z = w \cdot v : u$, причому $w = 1010 \cdot 0{,}4$, $v = 108 \cdot 0{,}3$ і $u = 108 \cdot 0{,}2$, а максимально допустимий порядок у машині 10^{12}. Якщо спочатку підрахувати множення $w \cdot v$, то відбудеться так зване переповнення і машина припинить обчислення. Якщо ж спочатку виконати операцію ділення $y = w : u$, а потім множення $y \cdot v$, одержимо потрібний результат.

При поданні чисел у напівлогарифмічній формі також виникає проблема кодування знака. Вона вирішується так само, як і для подання чисел у формі з фіксованою комою, але додаткових розрядів потрібно вже два – для знака порядку і знака мантиси.

Якщо для подання чисел у напівлогарифмічній формі відведено n розрядів, не враховуючи розрядів для знаків мантиси і порядку, і під запис мантиси відведено m розрядів, то діапазон подання чисел буде визначатися такою нерівністю:

$$S^{S^{n-m}} \le |x| \le (1 - S^{-m}) \cdot S^{S^{n-m}-1}. \qquad (2.6)$$

Розглянемо як приклад розрядну сітку для подання двійкових чисел у 32 розрядах (рис. 2.3).

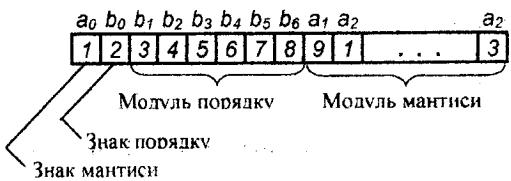


Рисунок 2.3. Розрядна сітка машини з плаваючою комою (з урахуванням розрядів під знаки мантиси і порядку)

Нехай запис числа має вигляд:

$$a_0 b_0 b_1 b_2 \ldots b_6 a_1 a_2 \ldots a_{24}. \qquad (2.7)$$

Перші два розряди виділяються під знаки мантиси (числа) і порядку відповідно, розряди з 3-го по 8-й виділяються під запис модуля порядку, а 24 розряди з 9-го по 32-й – під запис модуля мантиси. Тоді діапазон подання чисел буде визначатися в такий спосіб:

$$2^{-64} \le |x| \le (1 - 2^{-24}) \cdot 2^{63}, \qquad (2.8)$$

що приблизно відповідає діапазону чисел від 10^{-19} до 10^{19}.

У ряді машин необхідна точність обчислень викликає необхідність

передбачити можливість зміни розрядної сітки для подання чисел. Звичайно ця проблема вирішується шляхом збільшення розрядної сітки вдвічі за рахунок об'єднання двох регістрів машини. Таке подання чисел називається поданням із подвоєною точністю.

З нерівності (2.6) випливає, що чим більший розмір S, тим більший діапазон чисел, які можна подати в цій системі. У зв'язку з цим у деяких обчислювальних машинах отримало поширення подання чисел із плаваючою комою з основою, рівною цілому степеню числа 2, в основному з $S = 8$ і $S = 16$. Використання таких основ S призводить до деякого зменшення точності обчислень, але, крім розширення діапазону подання чисел, дозволяє прискорити деякі операції.

Як уже відзначалося, форма подання з плаваючою комою є основною формою подання чисел у сучасних обчислювальних машинах. До її недоліків відносять зменшення продуктивності машини в порівнянні з обчисленнями в природній формі, ускладнення її технічних пристроїв. Наприклад, повинні бути розроблені схеми нормалізації, передбачене устаткування для виконання операцій над порядками і мантисами чисел і т.п. Обчислювальні машини, у яких може бути використана і та й інша форми подання, дозволяють врахувати особливості розв'язуваної задачі і підвищити продуктивність обчислень.

2.2 Кодування чисел у двійковій системі числення

Будь-яку кінцеву послідовність нулів і одиниць прийнято називати двійковим кодом. У кожному конкретному випадку цей код має чітку однозначну інтерпретацію.

2.2.1 Прямий код числа у двійковій системі числення

Найпростішим машинним кодом є прямий код, який можна отримати при кодуванні знакової інформації.

Роздивимося двійкове число:

$$R = \pm 0, a1a2 \ldots an. \tag{2.9}$$

Відповідно до вище викладеного, число R кодується в такий спосіб:

$$R = \begin{cases} 0 \; a_1 a_2 \ldots a_n, & R \geq 0, \\ 1 \; a_1 a_2 \ldots a_n, & R \leq 0, \end{cases} \tag{2.10}$$

або в більш загальному випадку, якщо $S \neq 2$:

$$R = \begin{cases} 0 & a_1 a_2 \ldots a_n, \quad R \geq 0, \\ S-1 & a_1 a_2 \ldots a_n, \quad R \leq 0. \end{cases} \tag{2.11}$$

Оскільки $|R| < 1$, ці співвідношення можна переписати в такий спосіб:

$$[R]_{пр} = \begin{cases} R, & R \geq 0, \\ S - 1 + |R|, & R \leq 0. \end{cases} \tag{2.12}$$

Подання чисел відповідно до формули (2.12) називається прямим кодом числа R. Якщо S = 2, то (2.12) переписується у вигляді:

$$[R]_{пр} = \begin{cases} R, & R \geq 0, \\ 1 + |R|, & R \leq 0. \end{cases} \tag{2.13}$$

Таким чином, прямим кодом від'ємного числа називається його зображення в природній формі, при цьому в знаковому розряді проставляється одиниця. Прямий код додатного числа збігається з його звичайним зображенням у природній формі.

2.2.2 Доповняльний код числа

Введемо запис:

$$[R]_{д} = \begin{cases} R, & R \geq 0, \\ S + R, & R < 0. \end{cases} \tag{2.14}$$

Подання чисел відповідно до формули (2.14) називається доповняльним кодом числа R. Доповняльним кодом від'ємного двійкового числа називається його доповнення, яке отримується за таким правилом: у знаковому розряді записується одиниця, а в усіх інших розрядах цифри замінюються на взаємно обернені, після чого до молодшого розряду числа додається одиниця.

Доповняльний код додатного числа збігається з його прямим кодом.

2.3.3 Обернений код числа

Код, визначений за допомогою співвідношення:

$$[R]_{обр} = \begin{cases} R, & R \geq 0, \\ S + R - S^{n}, & R \leq 0. \end{cases} \qquad (2.15)$$

називається оберненим кодом числа R.

При S=2 для отримання оберненого коду від'ємного числа в знаковий розряд потрібно записати одиницю, а кожну цифру в записі числа замінити на її доповнення до S–1, тобто одиницю замінити на нуль, а нуль на одиницю.

Обернений код додатного числа збігається з його прямим кодом.

Зі співвідношень (2.14) і (2.15) випливає, що

$$[R]_{д} = [R]_{обр} + S^{-n}, \qquad (2.16)$$

тобто відмінність у записі доповняльного й оберненого кодів від'ємного числа полягає в останньому розряді. Якщо в останньому розряді у доповняльному коді записується $(S - x_n)$, то в оберненому – $(S - 1 - x_n)$. Звідси випливає, що для отримання доповняльного коду від'ємного числа потрібно до його оберненого коду додати одиницю.

З приведених визначень зрозуміло, що позитивне число в прямому, оберненому і доповняльному кодах збігаються.

Відзначимо, що при поданні чисел з плаваючою комою окремо кодуються мантиса і порядок числа. При цьому можливо подання мантис і порядків чисел у тому самому або різних кодах. Наприклад, порядок числа може бути поданий у прямому, а мантиса – у доповняльному кодах і т.п.

Таким чином, використовуючи обернений і доповняльний коди, операцію алгебраїчного додавання можна зводити до арифметичного додавання кодів чисел, що поширюється і на розряди знаків, які розглядаються як розряди цілої частини числа.

Приклад 2.1. Отримати прямий, обернений і доповняльний коди числа A= -10 у двійковій системі числення.

Прямий код числа має вигляд $A_{пр}$=1.1010 і отримується за алгоритмом переведення десяткових чисел у двійкову систему числення (п.2.1).

Обернений код числа A отримують інвертуванням кожного розряду інформаційної частини прямого коду: $A_{об}$=1.0101.

Доповняльний код числа A отримують шляхом додавання "1" до молодшого інформаційного розряду оберненого коду: $A_{доп}$=1.0110.

Приклад 2.2. Отримати прямий, обернений і доповняльний коди числа A= 10 у двійковій системі числення.

Оскільки число A є додатним, то прямий, доповняльний і обернений коди числа A є однаковими і мають вигляд: A=0.1010.

Контрольні запитання і задачі

1. Які є форми подання чисел в ЕОМ? Охарактеризуйте їх.
2. Охарактеризуйте особливості подання чисел з фіксованою комою, наведіть структуру розрядної сітки машини та приклади зображення двійкових чисел.
3. Назвіть недоліки використання форми подання чисел з фіксованою комою.
4. Охарактеризуйте особливості подання чисел з плаваючою комою, наведіть структуру розрядної сітки машини та приклади зображення двійкових чисел. Поясніть поняття "мантиси" та "порядку" числа.
5. Назвіть переваги та недоліки використання форми подання чисел з плаваючою комою.
6. Що розуміють під прямим кодом двійкового числа?
7. Як отримують обернений та доповняльний коди?
8. Як визначають знак двійкового числа?
9. Запишіть прямий, обернений та доповняльний коди у двійковій системі числення для десяткових чисел: -23; 12; -5; 7; 45; -19; -11.
10. Запишіть обернений та доповняльний коди для мантис та порядків двійкових чисел: $-1001*2^{-110}$; $10101*2^{-01101}$; $-11011001*2^{1101}$.

3 ПРАВИЛА ДОДАВАННЯ ТА ВІДНІМАННЯ ДВІЙКОВИХ ЧИСЕЛ

У запам'ятовувальних пристроях (ЗП) ЕОМ числа зберігаються у прямому чи доповняльному кодах. В оберненому коді числа зберігати недоцільно, оскільки при необхідності обернений код можна легко отримати з прямого шляхом порозрядного інвертування, для чого не потрібно додаткових затрат обладнання і часу. Із запам'ятовувального пристрою операнди пересилаються в арифметико-логічний пристрій у тому вигляді, у якому вони зберігалися в ЗП. Тому залежно від того, в яких кодах зберігаються числа в ЗП, розрізняють операції в прямих та доповняльних кодах.

При проектуванні підсумовувальних блоків (ПБ) для додавання і віднімання чисел у прямих кодах основні трудності пов'язані з реалізацією операції віднімання. Якщо у ПБ не використовується спеціальна схема-віднімач (її побудова подібна до побудови суматора), то віднімання реалізують шляхом його заміни операцією додавання прямого коду одного операнда з доповняльним чи оберненим кодом другого операнда.

Отриманий результат необхідно перевести у прямий код. Як відомо, для додатного числа прямий, обернений та доповняльний коди є однаковими. При отриманні від'ємного результату для утворення прямого коду з оберненого проводять операцію порозрядного інвертування. Якщо ж від'ємний результат додавання отримали у доповняльному коді, то для переведення його у прямий код після виконання операції порозрядного інвертування додають одиницю у молодший розряд результату.

3.1 Класифікація методів додавання двійкових чисел

Додавання чисел у двійковій системі числення класифікується таким чином:

- за формою подання чисел:
  - з фіксованою комою,
  - з плаваючою комою.
- за способом кодування двійкових чисел:
  - у прямому коді,
  - у доповняльному коді,
  - в оберненому коді.

Додавання також може відбуватися з переповненням розрядної сітки і без переповнення розрядної сітки.

3.2 Додавання чисел з фіксованою комою

Додавання чисел з фіксованою комою у цифрових обчислювальних машинах може виконуватися в одному з машинних кодів: прямому, оберненому або доповняльному. Суму також отримаємо в одному з цих кодів. При реалізації операції додавання знаковий розряд й інформаційна частина числа розглядаються як єдине ціле, в результаті чого, з від'ємними числами машина оперує як і з додатніми. Головна перевага такого методу полягає в тому, що правильний знак суми отримується автоматично в процесі додавання знакових цифр операндів і цифри переносу з сусіднього молодшого розряду. У випадку виникнення одиниці переносу зі знакового розряду суми її потрібно відкинути при додаванні в доповняльному коді і додати до молодшого інформаційного розряду суми при додаванні в оберненому коді (тобто виконати циклічний перенос одиниці переповнення). Блок-схема алгоритму методу додавання двійкових чисел подана у додатку А.

Для виявлення переповнення розрядної сітки при додаванні вводиться допоміжний розряд у знакову частину зображення числа, що називають розрядом переповнення. Таке подання числа називається модифікованим.

Знакова частина позитивного числа містить цифри 00, а від'ємного 11. Ознакою переповнення розрядної сітки є наявність у знаковій частині цифр 01 або 10.

Додавання у прямому коді виконується тільки над числами одного знаку. Числа з різними знаками підсумовують в оберненому або доповняльному коді.

У прикладах для наочності будемо використовувати модифіковані коди.

3.2.1 Додавання двійкових чисел у прямому коді

У прямому коді додаються лише додатні числа : A>0; B>0.

Операція додавання здійснюється за правилами двійкової арифметики. Основним ускладненням цього методу є можливість переповнення розрядної сітки. У цьому випадку варто передбачити можливість забезпечення додаткового старшого інформаційного розряду результату.

Особливістю цього методу є простота його реалізації. Оскільки додавання проводиться у прямих кодах, то і результат, зрозуміло, теж отримують у прямих кодах, тобто ніяких додаткових перетворень цей метод не потребує.

Алгоритм методу додавання двійкових чисел у прямих кодах подано в додатку А.

Приклад 3.1. Додамо числа A=8, B=4 у двійковій системі числення.

$A_{пр}$=0.1000, $B_{пр}$=0.0100

$$\begin{array}{rl} & 00.1000 \\ + & \underline{00.0100} \\ C_{пр} = & 00.1100 \end{array}$$

Приклад 3.1 показує механізм додавання двійкових чисел з фіксованою комою у прямому коді. У результаті додавання переповнення розрядної сітки не виникає.

Приклад 3.2. Додамо числа A=9, B=12 у двійковій системі числення.

$A_{пр}$=0.1001, $B_{пр}$=0.1100

$$
\begin{array}{rl}
 & 00.1001 \\
+ & \underline{00.1100} \\
C = & 01.0101 \\
C_{пр} = & 0.10101
\end{array}
$$

Знаковий розряд — Розряд переповнення

У прикладі 3.2 у результаті додавання виникає переповнення розрядної сітки. Для отримання правильного результату необхідно забезпечити можливість використання додаткового старшого інформаційного розряду (розряду переповнення).

3.2.2 Додавання двійкових чисел у доповняльному коді

Доповняльні коди використовуються при додаванні від'ємних чисел, тобто це також є одним із методів, що замінює операцію віднімання. Алгоритм методу подано в додатку А. Виконавши операцію додавання чисел у доповняльних кодах, перевіряємо, чи був перенос у знаковому розряді. Якщо перенос був, то старшу одиницю знакового розряду відкидаємо, якщо ж переносу не було, то переходимо до перевірки умови: чи було переповнення розрядної сітки. Якщо переповнення не було, то результат переводимо у прямий код: при умові, що в знаковому розряді отримали "1", інвертуємо код числа і до останнього інформаційного розряду результату додаємо одиницю, в наслідок чого отримуємо результат у прямому коді (знаковий розряд у цьому випадку не інвертується), якщо ж у знаковому розряді – "0", то результат уже отриманий у прямому коді, тому ніяких подальших перетворень проводити не потрібно. Якщо було переповнення розрядної сітки, то для отримання результату необхідно вважати додатковим старшим інформаційним розрядом розряд переповнення розрядної сітки.

Приклад 3.3. Додамо числа A=10, B=-13 у двійковій системі числення.

$A_{пр}$=0.1010, $B_{пр}$=1.1101

Запишемо число B у доповняльному коді:

$$\begin{array}{r} 1.0010 \\ +\quad 1 \\ \hline 1.0011 \end{array}$$

Виконаємо операцію додавання, використовуючи модифіковані коди:

$$\begin{array}{rl} & 00.1010 \\ + & 11.0011 \\ \hline C_{доп} = & 11.1101 \\ C_{пр} = & 11.0011 \end{array}$$

У прикладі 3.3 показано виконання операції додавання від'ємного та додатного чисел без переповнення розрядної сітки і з одиницею в знаковому розряді результату. Тому, відповідно до вище описаного алгоритму (додаток А), при переведенні результату у прямий код необхідно інвертувати інформаційну частину коду числа та додати "1" до молодшого інформаційного розряду результату.

Приклад 3.4. Додамо числа А=15, В=-14 у двійковій системі числення.

$A_{пр}$=0.1111, $B_{пр}$=1.1110

Перетворюємо число В у доповняльний код.

$$\begin{array}{r} 1.0001 \\ +\quad 1 \\ \hline 1.0010 \end{array}$$

Виконаємо операцію додавання, використовуючи модифіковані коди:

$$\begin{array}{rl} & 00.1111 \\ + & 11.0010 \\ \hline C = & 100.0001 \\ C_{пр} = & 00.0001 \end{array}$$

У прикладі 3.4 здійснюється операція додавання двійкових чисел у доповняльних кодах. Переповнення розрядної сітки не було, у знаковому

розряді результату отримали "0". Тому одиницю переносу у знаковому розряді, що виходить за межі розрядної сітки, відкидаємо і отримуємо результат в прямому коді. Ніяких подальших перетворень робити не потрібно.

Приклад 3.5. Додамо числа A=-5, B=-12 у двійковій системі числення.

$A_{пр}$=1.0101, $B_{пр}$=1.1100

Записуємо числа A і B у доповняльному коді:

$A_{доп}$ = 1.1011, $B_{доп}$ =1.0100

Виконаємо операцію додавання, використовуючи модифіковані коди:

$$\begin{array}{r} 11.1011 \\ + \underline{\ 11.0100} \end{array}$$

$C_{доп}$=110.1111=1.01111

$C_{пр}$ = 1.10001

У прикладі 3.5 показано додавання двох від'ємних чисел у доповняльних кодах у випадку виникнення переповнення розрядної сітки. Тому при переведенні результату із доповняльного в прямий код слід враховувати добавлений старший інформаційний розряд (розряд переповнення), а одиницю переносу зі знакового розряду відкидаємо.

3.2.3 Додавання двійкових чисел в оберненому коді

В обернених, як і в доповняльних, кодах ми можемо оперувати з від'ємними числами (тобто можемо виконати операцію віднімання).

У випадку додавання чисел в оберненому коді згідно з алгоритмом (додаток А) від'ємні числа переводимо в обернений код шляхом інвертування кожного інформаційного розряду прямого коду. Додавши числа в обернених кодах перевіряємо умову, чи був перенос зі знакового розряду. Якщо перенос був, то старшу одиницю знакового розряду додаємо до молодшого інформаційного розряду результату. Перевіряємо умову "чи було переповнення розрядної сітки". При переповненні враховувати додатковий старший інформаційний розряд (розряд

переповнення), а результат необхідно перевести у прямий код. Якщо переповнення розрядної сітки не було, то аналізуємо знаковий розряд результату: якщо у знаковому розряді – одиниця, то переводимо результат у прямий код, інвертуючи лише інформаційні розряди; якщо у знаковому розряді – нуль, то результат уже отримали у прямому коді і ніяких подальших перетворень не потрібно. Блок-схема алгоритму методу подана в додатку А.

Приклад 3.6. Додамо числа A=-10, B=12 у двійковій системі числення.

$A_{пр}$=1.1010, $B_{пр}$=0.1100

$A_{об}$=1.0101, $B_{об}$=0.1100

Виконаємо операцію додавання, використовуючи модифіковані коди:

$$\begin{array}{r} 11.0101 \\ + \underline{00.1100} \\ 100.0001 \\ + \underline{\quad\quad\nearrow 1} \\ C_{пр} = \ 00.0010 \end{array}$$

У прикладі 3.6 здійснюється додавання від'ємного та додатного числа. У результаті додавання виник перенос зі знакового розряду, тому додаємо одиницю переносу до молодшого інформаційного розряду результату. Але переповнення розрядної сітки не було, і в знаковому розряді результату отримали нуль, тому результат отримали одразу в прямому коді.

Приклад 3.7. Додамо числа A=-11, B=-7 в двійковій системі числення.

$A_{пр}$=1.1011, $B_{пр}$=1.0111

$A_{об}$=1.0100, $B_{об}$=1.1000

Виконаємо операцію додавання, використовуючи модифіковані коди:

$$\begin{array}{r} 11.0100 \\ + \underline{11.1000} \\ 110.1100 \\ + \underline{\quad\quad\nearrow 1} \\ 10.1101 \\ C_{об} = \ 1.01101 \\ C_{пр} = \ 1.10010 \end{array}$$

У прикладі 3.7 виник перенос зі знакового розряду результату і переповнення розрядної сітки, тобто необхідно спочатку старшу одиницю знакового розряду додати до молодшого інформаційного розряду результату, а потім, враховуючи розряд переповнення як старший інформаційний розряд, перевести результат у прямий код, тобто інвертувати інформаційні розряди.

Приклад 3.8. Додамо числа A=-5, B=-2 у двійковій системі числення.

$A_{пр}$=1.0101, $B_{пр}$=1.0010
$A_{об}$=1.1010, $B_{об}$=1.1101

$$\begin{array}{r} 11.1010 \\ + \underline{11.1101} \\ 111.0111 \\ + \underline{\overset{\nearrow}{1}} \\ C_{об} = 11.1000 \\ C_{пр} = 11.0111 \end{array}$$

У прикладі 3.8 здійснюється операція додавання двох від'ємних чисел, результат ми отримали без переповнення розрядної сітки, але з одиницею у знаковому розряді, тому при переведенні результату в прямий код необхідно інвертувати лише інформаційні розряди.

3.3 Додавання двійкових чисел з плаваючою комою

Як відомо, числа, зображені у формі з плаваючою комою, відображаються у вигляді двох частин: мантиси та порядку. Перед додаванням мантис доданків необхідно привести числа з плаваючою комою до одного порядку.

Машинна операція додавання чисел з плаваючою комою має такі етапи:

- вирівнюються порядки доданків: менший порядок збільшується до більшого, а мантиса числа зсувається вправо на відповідну кількість розрядів;
- виконується перетворення мантис доданків в один з модифікованих кодів (обернений чи доповняльний);
- мантиси доданків додаються;
- у разі необхідності мантиса суми переводиться в прямий код;

- виконується нормалізація суми;
- виконується корекція порядку результату.

Число вважається нормалізованим, якщо старший інформаційний розряд дорівнює одиниці. А також, слід зауважити, що додавання мантис здійснюється в оберненому або доповняльному кодах. Якщо мантиса результату додавання нормалізована, то до цього результату приписуємо порядок будь-якого з операндів. В іншому випадку відбувається нормалізація числа. Порушення нормалізації може бути справа чи зліва. Ознакою порушення нормалізації числа справа є наявність різних цифр у знакових розрядах. У цьому випадку необхідно зсунути число вправо на один розряд. А ознака порушення нормалізації числа зліва – це присутність однакових цифр в розряді переповнення і в старшому розряді цифрової частини. Він показує на необхідність зсуву числа вліво на один розряд.

Кожний зсув мантиси вліво при нормалізації веде до зменшення порядку результату додавання на одиницю. А кожний зсув мантиси вправо – до збільшення. Таким чином відбувається корекція порядку.

Блок-схему алгоритму методу подано в додатку А .

Приклад 3.9. Додамо числа $A=27*e^6$ і $B=-46*e^5$ у двійковій системі числення.

Вирівнюємо порядки доданків:

$$A=0.11011*2^{110} \text{ і } B=1.01110*2^{101}=1.00111*2^{110}$$

Перетворюємо мантису числа В у доповняльний код:

$$\begin{array}{r} 1.11000 \\ + \quad\quad 1 \\ \hline 1.11001 \end{array}$$

Додаємо мантиси чисел А і В:

$$\begin{array}{rl} & 00.11011 \\ + & 11.11001 \\ \hline C_{доп}= & 100.10100 \end{array}$$

$$C_{пр} = 00.10100*2^{110}$$

У прикладі 3.9 ілюструється механізм додавання двійкових чисел з плаваючою комою. Отриману "1", що виходить за межі розрядної сітки знакового розряду результату, відкидаємо, керуючись правилами виконання операції додавання двійкових чисел у доповняльних кодах. Переповнення розрядної сітки не було, а у знаковому розряді результату отримали "0", тому можна зробити висновок, що результатом є додатне число, і виконувати додаткові операції для переведення його у прямий код не потрібно. У старшому інформаційному розряді результату записана "1", тому двійковий код результату є нормалізованим і не потребує додаткової операції нормалізації.

Контрольні запитання і задачі

1. У яких кодах зберігається інформація у запам'ятовувальних пристроях?
2. Як виконується операція віднімання у двійковій системі числення?
3. Поясніть алгоритм виконання операції додавання двійкових чисел з плаваючою комою.
4. Що означає поняття "переповнення розрядної сітки"?
5. У чому полягає основна різниця між методами додавання від'ємних чисел у доповняльних та обернених кодах?
6. Що розуміють під нормалізацією коду двійкового числа з плаваючою комою?
7. Додайте числа A=-24 і B=18 у двійковій системі числення у доповняльних та обернених кодах.
8. Додайте числа A=-14 і B=-9 у двійковій системі числення у доповняльних та обернених кодах.
9. Додайте числа $A_{пр}=0.11100101*2^{100}$ і $B_{пр}=1.10010011*2^{110}$ у доповняльних та обернених кодах.
10. Додайте числа $A_{пр}=0.00101101*2^{1001}$ і $B_{пр}=1.11100101*2^{1011}$ у доповняльних та обернених кодах.

4 ДІЛЕННЯ ЧИСЕЛ У ДВІЙКОВІЙ СИСТЕМІ ЧИСЛЕННЯ

Ділення чисел у двійковій системі числення класифікується таким чином:

- за формою подання чисел:
  - з фіксованою комою;
  - з плаваючою комою.

- за механізмом виконання операції:
  - з відновленням остачі;
  - без відновлення остачі;
- за швидкодією:
  - просте;
  - прискорене;
- за точністю результату:
  - з округленням результату;
  - без округлення результату.

При діленні двійкових чисел користуються алгоритмом: від діленого віднімають дільник і аналізують результат: 1) якщо він >0, то в результат ділення записують "1", потім результат віднімання домножують на 2 та віднімають дільник; 2) якщо результат віднімання <0, то в результат ділення записують "0", а результат віднімання домножують на 2 і до отриманого результату додають дільник. Потім знову аналізують знак отриманого результату віднімання, щоб визначити наступний результат ділення. Дії виконують до тих пір, поки не отримають результат ділення у двійковій системі із заданою точністю. Приклад 4.1 ілюструє реалізацію описаного методу ділення.

Приклад 4.1. Поділити число 5 на 7, отримати результат у двійковій системі числення з точністю до 6 розрядів. Для наочності числа 5 і 7 будемо розглядати як цілі числа.

	5 -7	-4 7	6 -7	-2 7	10 -7	6 -7	-2 7
Результат віднімання	- 2	3	- 1	5	3	- 1	5
Результат ділення	0,	1	0	1	1	0	1

Отже, отримали результат 0,101101.
Ціла частина, дробова частина

Якщо у цілій частині при діленні двійкових чисел з фіксованою комою отримали "1", то це означає, що виникло переповнення розрядної сітки, подальші операції виконувати не потрібно. Тому при діленні чисел з фіксованою комою слід дотримуватися правила:

$$|A| < B, \text{ тобто } |\text{ділене}| < |\text{дільника}|. \qquad (4.1)$$

Отже, результат ділення двійкових кодів з фіксованою комою – дробове число.

При діленні чисел з плаваючою комою умова (4.1) не є обов'язковою, оскільки можна провести операцію нормалізації, зсунувши результат на 1 розряд вліво і збільшивши порядок результату на "1".

4.1 Метод ділення двійкових чисел з відновленням остачі

Для того, щоб поділити двійкові числа з відновленням остачі необхідно виконати такі операції:

- визначити знак результату (за правилом алгебри логіки: «сума по модулю два» знакових розрядів операндів);
- у суматор записати прямий код числа А;
- у регістр В - прямий код числа В;
- у регістр С - доповняльний код числа В;
- у регістр Д будемо записувати результат.
- відбувається зсув у суматорі вліво.

У суматорі додаємо зміст суматора та регістра С. Якщо в результаті додавання в знаковому розряді отримали "0", то в регістр Д потрібно записати "1", а якщо в знаковому розряді отримали "1", то в регістр Д потрібно записати "0", а до суматора додати значення регістра В. Потім здійснюється зсув вліво на один розряд коду, записаного у суматорі, та коду, записаного у регістрі Д.

Операція ділення належить до розряду неточних операцій, оскільки результат, як правило, отримують з деякою похибкою. Тому ознакою закінчення операції ділення може бути або досягнення заданої точності (кількість розрядів у частці), або отримання чергової остачі, яка рівна нулю. Блок-схема алгоритму методу ділення двійкових чисел з відновленням остачі подано в додатку Б гілка А.

Приклад 4.2. Поділимо два числа А=0.101 і В=-0.111 у двійковій системі числення з точністю до 6 розрядів результату.

$A_{пр}$=00.101, $B_{пр}$=-00.111.
$A_{доп}$=00.101. $B_{доп}$=11.001.

Суматор (СМ)	Регістр (РгД)	Примітки
00.101	000000	[СМ]:= $[A]_{пр}$, [РгД]:=0; [РгВ]:= $[B]_{пр}$; [РгС] :=$[B]_{доп}$;
01.010	00000-	[СМ]←1; [РгД]←1;
+11.001		
00.011	000001	[СМ]:= [СМ]+ [РгС]; $Д_1$=1;

Суматор (СМ)	Регістр (РгД)	Примітки
00.110	00001-	[СМ]←1; [РгД]←1;
+11.001 11.111	000010	[СМ]:= [СМ]+ [РгС]; $Д_2$=0;
+00.111 00.110		Відновлення остачі [СМ]:= [СМ]+ [РгВ];
01.100	00010-	[СМ]←1; [РгД]←1;
+11.001 00.101	000101	[СМ]:= [СМ]+ [РгС]; $Д_3$=1;
01.010	00101-	[СМ]←1; [РгД]←1;
+11.001 00.011	001011	[СМ]:= [СМ]+ [РгС]; $Д_4$=1;
00.110	01011-	[СМ]←1; [РгД]←1;
+11.001 11.111	010110	[СМ]:= [СМ]+ [РгС]; $Д_5$=0;
+00.111 00.110		Відновлення остачі [СМ]:= [СМ]+ [РгВ];
01.100	10110-	[СМ]←1; [РгД]←1;
+11.001 00.101	101101	[СМ]:= [СМ]+ [РгС]; $Д_6$=1;

Відповідь: Д = 1 | 101101
Знаковий розряд | Інформаційні розряди

У прикладі 4.2 показано механізм виконання операції ділення двійкових чисел з відновленням остачі, здійснений за наданим вище алгоритмом. Знак результату визначається за логічним правилом "сума за модулем два": $S_D = 0 \oplus 1 = 1$. Отже, результат є від'ємним числом.

4.2 Метод ділення двійкових чисел без відновлення остачі

Для того, щоб поділити двійкові числа без відновлення остачі необхідно виконати такі операції:

- визначити знак результату (за логічним правилом: «сума за модулем два» знакових розрядів операндів);
- у суматор записати прямий код числа А;

- у регістр В - прямий код числа В;
- у регістр С - доповняльний код числа В;
- у регістр Д будемо записувати результат;
- зсуваємо вліво значення суматора і регістра Д.

Аналізуємо знаковий розряд суматора. Якщо в знаковому розряді – "0", то до суматора додаємо значення регістра С, якщо – "1", то до суматора додаємо значення регістра В. Якщо в результаті додавання у знаковому розряді суматора отримуємо "0", то в регістр Д записуємо "1", якщо – "1", то в регістр Д записуємо "0". Потім проводимо зсув суматора і регістра Д. Блок схема алгоритму методу подається у додатку Б гілка В.

Приклад 4.3. Поділимо числа А=0.101 і В=- 0.111 у двійковій системі числення з точністю до 6 розрядів результату.

$A_{пр}$=00.101; $B_{пр}$=-00.111

$A_{доп}$=00.101; $B_{доп}$=11.001

Суматор (СМ)	Регістр (РгД)	Примітки
00.101	000000	[СМ]:= $[A]_{пр}$; [РгД]:=0; [РгВ]:= $[B]_{пр}$; [РгС] :=$[B]_{доп}$;
01.010	00000-	[СМ]←1; [РгД]←1;
+11.001 00.011	000001	[СМ]:= [СМ]+ [РгС]; $Д_1$=1;
00.110	00001-	[СМ]←1; [РгД]←1;
+11.001 11.111	000010	[СМ]:= [СМ]+ [РгС]; $Д_2$=0;
11.110	00010-	[СМ]←1; [РгД]←1;
+00.111 00.101	000101	[СМ]:= [СМ]+ [РгВ]; $Д_3$=1;
01.010	00101-	[СМ]←1; [РгД]←1;
+11.001 00.011	001011	[СМ]:= [СМ]+ [РгС]; $Д_4$=1;
00.110	01011-	[СМ]←1; [РгД]←1;
+11.001 11.111	010110	[СМ]:= [СМ]+ [РгС]; $Д_5$=0;
11.110	10110-	[СМ]←1; [РгД]←1;
+00.111 00.101	101101	[СМ]:= [СМ]+ [РгВ]; $Д_6$=1;

Відповідь: Д = 1,101101
Знаковий розряд

У прикладі 4.3 продемонстровано механізм виконання алгоритму ділення двійкових чисел без відновлення остачі. Знак результату визначається за правилом: $S_D = 0 \oplus 1 = 1$. Отже, результат є від'ємним числом.

Результати ділення числа А = 0.101 на число В = -0.111 у двійковій системі числення за методом з відновленням остачі (приклад 4.2) і без відновлення остачі (приклад 4.3), звісно, отримали однаковий.

4.3 Метод прискореного ділення двійкових чисел

Для прискорення операції ділення використовують аналіз старших інформаційних розрядів. Якщо два старші інформаційні розряди дорівнюють одиниці, то у наступний розряд частки записуємо "1", якщо "0", то записуємо "0" і проводимо зсув суматора і регістра результату на два розряди вліво. Початок алгоритму прискореного ділення включає, звісно, ті ж операції, що і прості алгоритми ділення:

- визначити знак результату (за правилом алгебри логіки "сума за модулем два" знакових розрядів операндів);
- у суматор записати прямий код числа А;
- у регістр В - прямий код числа В;
- у регістр С - доповняльний код числа В;
- у регістр Д будемо записувати результат.

Зрозуміло, що алгоритм прискорення може накладатися як на алгоритм ділення з відновленням остачі, так і на алгоритм ділення без відновлення остачі.

У прикладі 4.4 показано механізм виконання методу прискореного ділення двійкових чисел за алгоритмом ділення без відновлення остачі з округленням результату.

4.4 Метод ділення двійкових чисел з округленням та без округлення результату

Після закінчення операції ділення двійкових чисел за обраним алгоритмом (з відновленням чи без відновлення остачі, з прискоренням чи без прискорення) для забезпечення округлення результату операцію ділення за обраним алгоритмом продовжують для визначення ще одного розряду результату. Потім аналізують молодший інформаційний розряд результату. Якщо у цьому розряді записана "1", то її додають до попереднього розряду результату, якщо "0", то останній інформаційний розряд результату просто ігнорують. При використанні методу ділення

двійкових чисел без округлення результату описані дії не проводять, обмежуючись виконанням алгоритму ділення.

У прикладі 4.4 розглядаються особливості виконання операції "округлення результату ділення двійкових чисел".

Приклад 4.4. Розділимо число A=0.101 на число B= -0.111 у двійковій системі числення прискореним методом ділення без відновлення остачі з точністю до 6 розрядів з округленням результату.

Для забезпечення операції округлення результату беремо семирозрядний регістр Д.

$A_{пр}$=00.101; $B_{пр}$=-00.111; $B_{доп}$=11.001

Суматор (СМ)	Регістр (РгД)	Примітки
00.101	000000	[СМ]:= $[A]_{пр}$; [РгД]:=0; [РгВ]:= $[B]_{пр}$; [РгС] :=$[B]_{доп}$;
01.010	00000-	[СМ]←1; [РгД]←1;
+11.001 00.011	000001	[СМ]:= [СМ]+ [РгС]; $Д_1$=1; аналіз двох старших розрядів
00.110	00001-	[СМ]←1; [РгД]←1;
+11.001 11.111	000010	[СМ]:= [СМ]+ [РгС]; $Д_2$=0; аналіз двох старших інформаційних розрядів: $Д_3$ = 1
11.110	000101-	[СМ]←2; [РгД]←2;
+00.111 00.011	0001011	[СМ]:= [СМ]+ [РгВ]; $Д_4$=1; аналіз двох старших інформаційних розрядів
00.110	001011-	[СМ]←1; [РгД]←1;
+11.001 11.111	0010110	[СМ]:= [СМ]+ [РгС]; $Д_5$=0; аналіз двох старших інформаційних розрядів : $Д_6$ = 1
11.100	101101-	[СМ]←2; [РгД]←2;
+00.111 00.011	1011011	[СМ]:= [СМ]+ [РгВ]; $Д_7$=1;

Визначимо знак результату: $S_D = S_A \oplus S_B = 0 \oplus 1 = 1$.

Отримали результат 1.1011011. Округлимо результат до 6 розрядів:

$$\begin{array}{r} 1.101101 \\ +\quad\quad 1 \\ \hline 1.101110 \end{array}$$

Отримали результат: Д = 1.101110.

4.5. Особливості виконання операції ділення двійкових чисел з плаваючою комою

При діленні двійкових чисел з плаваючою комою спочатку визначають знак результату за правилом алгебри логіки "сума за модулем два", потім проводять корекцію форми запису чисел ($|mA| < |mB|$) та визначають порядок результату за формулою:

$$P_C = P_A - P_B,$$

де P_C – порядок результату; P_A – порядок числа А; P_B – порядок числа В. Далі виконують операцію ділення мантиси числа А на мантису числа В за правилами ділення двійкових чисел з фіксованою комою за одним з обраних алгоритмів (з відновленням чи без відновлення; з округленням чи без округлення; просте чи прискорене). Отриманий результат нормалізують.

Приклад 4.5. Розділити число А=0.101·2 на число $1.111 \cdot 2^{100}$ простим методом ділення з відновленням остачі без округлення.

Для забезпечення можливості визначення цілої частини результату при діленні візьмемо семирозрядний регістр Рг Д для отримання результату.

Визначимо порядок результату:

P_A=0.101

P_B=0.100

$P_A - P_B$=0.101–0.100=0.101+(- 0.100).

$P_{Вдоп.}$=1.100

$$\begin{array}{r} 0.101 \\ \underline{1.100} \\ 10.001 \end{array}$$

Отже, $P_Д$=0.001.

Визначимо знак результату:

$S_Д = S_{m_A} \oplus S_{m_B} = 0 \oplus 1 = 1$, тобто результат – число від'ємне.

Тут $S_Д$ - знак результату;

S_{m_A} - знак мантиси числа А;

S_{m_B} - знак мантиси числа В.

Поділимо мантису числа А на мантису числа В:
$[m_A]_{пр.}$=0.101
$[m_B]_{пр.}$=- 00.111
$[m_B]_{доп.}$=11.001

Суматор (СМ)	Регістр (РгД)	Примітки
00.101	0000000	[СМ]:=$[m_A]_{пр}$; [РгД]:=0; [РгА]:=$[m_B]_{пр}$; [РгВ]:=$[m_B]_{доп}$;
	000000.-	[РгД]←1;
+11.001 11.110	000000.0	[СМ]:=[СМ]+[РгВ]; $Д_1$=0;
+00.111 00.101		Відновлення остачі [СМ]:=[СМ]+[РгА];
01.010	00000.0-	[СМ]←1; [РгД]←1;
+11.001 00.011	00000.01	[СМ]:=[СМ]+[РгВ]; $Д_2$=1;
00.110	0000.01-	[СМ]←1; [РгД]←1;
+11.001 11.111	0000.010	[СМ]:=[СМ]+[РгВ]; $Д_3$=0;
+00.111 00.110		Відновлення остачі [СМ]:=[СМ]+[РгА];
01.100	000.010-	[СМ]←1; [РгД]←1;
+11.001 00.101	000.0101	[СМ]:=[СМ]+[РгВ]; $Д_4$=1;
01.010	00.0101-	[СМ]←1; [РгД]←1;
+11.001 00.011	00.01011	[СМ]:=[СМ]+[РгВ]; $Д_5$=1;
00.110	0.01011-	[СМ]←1; [РгД]←1;
11.001 11.111 +00.111 00.110	0.010110	[СМ]:=[СМ]+[РгВ]; $Д_6$=0; Відновлення остачі [СМ]:=[СМ]+[РгА];
01.100	010110-	[СМ]←1; [РгД]←1;
+11.001 00.101	0.101101 ціле число / дробова частина	[СМ]:=[СМ]+[РгВ]; $Д_7$=1;

Оскільки переповнення розрядної сітки не було, то отриманий результат: $Д=1.101101\cdot 2^{001}$. Результат отримали у нормалізованому вигляді.
Знаковий розряд
Відповідь: $Д=1.101101\cdot 2^{001}$.

Контрольні запитання і задачі

1. У чому різниця алгоритмів ділення двійкових чисел з відновленням та без відновлення остачі?
2. У чому полягає алгоритм прискорення виконання операції ділення двійкових чисел?
3. Поясніть особливості реалізації алгоритму ділення двійкових чисел з округленням результату.
4. Які додаткові операції містить алгоритм виконання операції ділення двійкових чисел з плаваючою комою?
5. Поділіть число A=0.01001011 на число B=1.11001101 за методами ділення без відновлення остачі та з відновленням остачі. Порівняйте отримані результати.
6. Поділіть число A=0.01110011 на число B=1.10101101 за методами ділення з округленням та без округлення результату. Порівняйте отримані результати та зробіть висновки.
7. Поділіть число A=1.10010101 на число B=1.10011011 за простим та прискореним методами ділення двійкових чисел з відновленням остачі. Порівняйте отримані результати та зробіть висновки.
8. Поділіть число A=1.11010011 на число B=0.11101101 за простим та прискореним методами ділення двійкових чисел без відновлення остачі. Порівняйте отримані результати та зробіть висновки.
9. Поділіть число $A=1.00110111*2^{1001}$ на число $B=1.10100101*2^{-1100}$ за простим та прискореним методами ділення двійкових чисел з плаваючою комою без відновлення остачі. Порівняйте отримані результати та зробіть висновки.
10. Поділіть число $A=0.11001011*2^{-0101}$ на число $B=0.10011101*2^{1011}$ за простим та прискореним методами ділення двійкових чисел з плаваючою комою з відновленням остачі. Порівняйте отримані результати та зробіть висновки.
11. Поділіть число $A=0.10110110*2^{0110}$ на число $B=1.11100100*2^{-1101}$ за прискореним методом ділення двійкових чисел з плаваючою комою без відновлення та з відновленням остачі. Порівняйте отримані результати та зробіть висновки.

5 МНОЖЕННЯ ЧИСЕЛ У ДВІЙКОВІЙ СИСТЕМІ ЧИСЛЕННЯ

Методи множення чисел у двійковій системі числення можна класифікувати таким чином:

- За формою подання чисел:
  - методи множення чисел з фіксованою комою;

- методи множення чисел з плаваючою комою.

- За швидкодією:
  - методи простого множення;
  - методи прискореного множення.
- За видом використаного суматора:
  - методи множення на суматорі прямого коду;
  - методи множення на суматорі доповняльного коду;
  - методи множення на суматорі оберненого коду (використовується рідко).
- За аналізом розрядів множника:
  - множення з молодших розрядів;
  - множення зі старших розрядів.

Усі перераховані методи можуть накладатися один на одного, що дає змогу вибрати потрібний метод множення двійкових чисел з урахуванням вимог до швидкості виконання операції та до використання апаратних затрат на реалізацію алгоритму.

5.1 Методи простого множення на суматорі прямого коду

Розрізняють такі методи простого множення на суматорі прямого коду :

- множення з молодших розрядів;
- множення зі старших розрядів.

Розглянемо особливості реалізації цих методів.

5.1.1 Множення з молодших розрядів

Для реалізації методу простого множення двійкових чисел на суматорі прямого коду, починаючи з молодших розрядів, необхідно використовувати регістр для зберігання множника і суматор часткових добутків, які повинні мати кола зсуву вправо. Регістр множеного може не мати кіл зсуву. Для виконання операції множення використовують регістри РгА і РгВ та суматор. При використанні суматора прямого коду

знаковий розряд результату множення визначається за допомогою спеціальної логічної операції:

$$Sg_C = Sg_A \oplus Sg_B,$$

де $\oplus$ - знак додавання за модулем два. У цьому випадку додатний результат буде за умови, що знаки обох множників однакові.

У першу чергу регістру РгА присвоюємо значення множника А в прямому коді (РгА:=А), а регістру РгВ – прямий код числа В (РгВ:=В). Суматор розпочинає операції з обнулення (СМ:= 0).

Аналізуємо молодший розряд регістра РгВ. Якщо у молодшому розряді регістра РгВ знаходиться "1", то до суматора додаємо значення регістра РгА і проводимо зсув См і РгВ вправо на один розряд. Якщо в молодшому розряді регістра РгВ – "0", то просто проводиться зсув См і РгВ вправо на один розряд. При чому зсув проводиться наскрізно, тобто молодші розряди суматора записують в старші розряди РгВ. Такі дії проводять циклічно до тих пір, поки не проаналізують усі розряди РгВ. Якщо мають справу з двійковими числами з плаваючою комою, то отриманий результат множення нормалізують. Результат множення міститься у суматорі і регістрі РгВ. Наочно ці дії можна побачити на структурній блок-схемі алгоритму по гілці А, додаток В. Приклад 5.1 – приклад множення двійкових чисел за описаним методом.

Приклад 5.1. Помножити два числа з фіксованою комою: А=1.111; В=0.101.

См	РгВ	Примітки
000	101	[См]:=0; [РгА]:=А;
+111		[См]:=[См]+[РгА];
111		
011	110	[См]→ 1; [РгВ]→ 1;
001	111	[См]→ 1; [РгВ]→ 1;
+111		
1000	111	[См]:=[См]+[РгА];
100	011	[См]→ 1; [РгВ]→ 1;

Відповідь: $C_{пр} = 1.100011_2$. Знак результату визначають за логічним правилом "сума за модулем два":

$Sg_C = Sg_A \oplus Sg_B = 1 \oplus 0 = 1$, тобто результат – від'ємне число.

5.1.2 Множення зі старших розрядів

Перед виконанням операції множення зі старших розрядів ініціалізують дані. Регістр множника і суматор повинні мати кола зсуву вліво. Суматор часткових добутків повинен мати подвійну довжину. Метод потребує більше обладнання і ніяких особливих переваг не дає. Якщо старший розряд регістра РгВ рівний одиниці, то до суматора додають значення регістра РгА і проводять зсув регістра РгВ та суматора вліво на один розряд, інакше проводиться тільки зсув регістра РгВ та суматора. Дії відбуваються до тих пір, поки не проаналізують усі розряди РгВ. Зауважимо, що при аналізі останнього розряду зсув суматора та РгВ не проводиться. При множенні з старших розрядів зсув суматора та регістра РгВ проводиться не наскрізно. Зрозуміло, якщо мають справу з двійковими числами з плаваючою комою, то отриманий результат множення нормалізують. Кінцевий результат міститься в суматорі. Ілюстрацію виконання такого множення показано на структурній блок-схемі алгоритму по гілці А, додаток В.

У прикладі 5.2 ілюструється виконання операції простого множення на суматорі прямого коду зі старших розрядів:

Приклад 5.2. Помножити два числа з фіксованою комою А=1.111 і В=1.101.

См	РгВ	Примітки
000000	101	[См]:=0; $[РгВ]:=[B]_{пр}$ $[РгА]:=[A]_{пр};$
+000111 000111 001110	 01-	См]:=[См]+[РгА] [См]← 1; [РгВ]← 1
011100	1--	[См]← 1; [РгВ]← 1
+000111 100011		[См]:=[См]+[РгА]

Відповідь: С:=0.100011. Результат множення С=А*В=1.111*1.101=0.100011. Знак результату визначають за логічним правилом "сума за модулем два":

$Sg_C = Sg_A \oplus Sg_B = 1 \oplus 1 = 0$, тобто результат – додатне число.

5.2 Просте множення на суматорі доповняльного коду

Добуток обернених (доповняльних) кодів співмножників дорівнює оберненому (доповняльному) коду результату тільки у випадку додатного множника.

Нехай множник $A = [A]_{об}$, а множник $B>0$. Тоді

$$A*B = [A]_{об}\, 0, b_1b_2 \ldots b_n = [A]_{об}\, b_1\, 2^{-1} + [A]_{об}\, b_2\, 2^{-2} + \ldots + [A]_{об}\, b_n\, 2^{-n}.$$

За теоремою про додавання обернених кодів у правій частині даного рівняння отримується обернений код результату. В обернених кодах знак добутку визначається автоматично, знаковий розряд бере участь в операції множення разом з інформаційними розрядами.

Якщо множник від'ємний, то добуток чисел на суматорі оберненого коду отримують додаванням поправок $[A]_{об}$ та $[A]_{об}\cdot 2^{-n}$ до добутку обернених кодів співмножників.

При множенні чисел на суматорах оберненого та доповняльного кодів одночасно отримують знакову та цифрову частини. Аналізуючи знак результату, проводять корекцію результату: якщо у знаковому розряді "0", то результат – додатне число і ніяких змін інформаційна частина не потребує, якщо у знаковому розряді "1", то необхідно інформаційну частину перевести в прямий код, і результат множення буде від'ємним.

5.2.1 Множення з молодших розрядів за умови, що значення числа B > 0

Для множення двох чисел у доповняльних кодах з молодших розрядів роблять ініціалізацію: в регістр РгА записують доповняльний код числа А, в РгВ – доповняльний код числа В, у суматор записують "0". Аналізуємо молодший розряд регістра РгВ. Якщо у молодшому розряді регістра РгВ знаходиться " 1 ", то до суматора додаємо значення регістра РгА і проводимо зсув См і РгВ вправо на один розряд. Якщо в молодшому розряді регістра РгВ знаходиться "0", то просто проводиться зсув См і РгВ вправо на один розряд. При чому зсув проводиться наскрізно, тобто молодші розряди суматора записують у старші розряди РгВ. Ці дії проводять циклічно до тих пір, поки не проаналізують всі розряди РгВ. Потім аналізують знаковий розряд: якщо у знаковому розряді отримали "1", то результат переводять у прямий код і нормалізують (при множенні чисел з плаваючою комою), якщо –"0", то результат тільки нормалізують (при множенні чисел з плаваючою комою). Результат множення міститься у суматорі (старші розряди) і регістрі РгВ (молодші розряди). Наочно ці дії

можна побачити на блок-схемі алгоритму по гілці В додатку В. Приведемо приклад множення двійкових чисел за описаним методом.

Приклад 5.2. Помножити два числа з фіксованою комою: А=-0.101; В=0.111

$A_{пр}$ =-00.101; $B_{пр}$=00.111;

$A_{доп}$=11.011; $B_{доп}$=00.111.

СМ	РгВ	Примітки
00.000	111	[См]:=0;[Рг В]:=$[B]_{доп}$; [РгА]:=$[A]_{доп}$;
+11.011		[См]:=[См]+[РгА];
11.011		
11.101	111	[См]→1 ;[РгВ]→1;
+11.011		[См]:=[См]+[РгА];
111.000		
11.100	011	[См]→1; [РгВ]→1;
11.011		[См]:=[См]+[РгА];
110.111		
11.011	101	[См]→1; [РгВ]→1.

$C_{доп}$=11.011101;

$C_{пр}$ =-0.100011;

Відповідь: С =-0.100011.

5.2.2 Множення з молодших розрядів за умови, що значення множника В < 0

Множення чисел у доповняльному коді з молодших розрядів у випадку, коли множник В<0 виконується аналогічно п.5.2.1. Але є, звичайно, суттєва відмінність методів на завершальному етапі виконання операції множення. Після завершення операції множення, до суматора додають доповняльний доповняльного коду код числа А, якщо А>0, та прямий код числа А, якщо А<0; операції зсуву не проводять. Потім перевіряють знаковий розряд для забезпечення можливості переведення результату у прямий код. Якщо операція множення виконувалась над числами з плаваючою комою, то проводять нормалізацію результату. Алгоритм методу поданий на блок-схемі додатку В (гілка В).

Приклад 5.3. Помножити два числа з фіксованою комою: А=-0.101; В=-0.111.

$A_{пр}$=-00.101 $B_{пр}$=-00.111

$A_{доп}$=11.011 $B_{доп}$=11.001

СМ	РгВ	Примітки
00.000	001	[См]:=0; [РгВ]:=$[B]_{доп}$; [РгА]:=$[A]_{доп}$; [РгС]:=$[A]_{пр}$;
+11.011		[См]:=[См]+[РгА];
11.011		
11.101	100	[См]→ 1;[Рг В]→ 1;
11.110	110	[См]→ 1;[Рг В]→ 1;
11.111	011	[См]→ 1;[Рг В]→ 1;
+00.101		[См]:=[См]+[РгС].
00.100	011	

Відповідь: С:= 00.100011. Оскільки у знаковому розряді результату записано "0", то результат отримали у прямому коді, тому ніяких перетворень непотрібно. Результат – додатне число.

Приклад 5.4. Помножити два числа з фіксованою комою: А=0.101; В= -0.111.

$A_{пр}$=00.101; $B_{пр}$= -00.111;

$A_{доп}$=00.101; $B_{доп}$=11.001;

$[A_{доп}]_{доп}$=11.011.

СМ	РгВ	Примітки
00.000	001	[См]:=0; [РгВ]:=$[B]_{доп}$; [РгА]:=$[A]_{доп}$; [РгС]:=$[A_{доп}]_{доп}$;
+00.101		[См]:=[См]+[РгА];
00.101		
00.010	100	[См]→ 1; [РгВ]→ 1;
00.001	010	[См]→ 1; [РгВ]→ 1;
00.000	101	[См]→ 1; [РгВ]→ 1;
+11.011		[См]:=[См]+[РгС].
11.011	101	

$C_{доп}$=11.011101

$C_{пр}$= -00.100011

Відповідь: $C_{пр}$= -00.100011.

5.2.3 Множення зі старших розрядів за умови, що значення множника B > 0

При множенні чисел, починаючи зі старших розрядів у доповняльних кодах використовують (як і при множенні в прямих кодах зі

старших розрядів) подвійну розрядність суматора, причому ініціалізація інших даних така ж, як і при множенні з молодших розрядів: у регістр РгА записуємо доповняльний код числа А, в РгВ – доповняльний код числа В, суматор (подвійної розрядності) обнулюють. Аналізують старший розряд регістра РгВ: якщо старший розряд регістра РгВ рівний одиниці, то до суматора додають значення регістра РгА і проводять зсув регістра РгВ та суматора вліво на один розряд, інакше проводиться тільки зсув регістра РгВ та суматора. Дії відбуваються до тих пір, поки не проаналізують усі розряди РгВ. При чому при аналізі останнього розряду зсув суматора та РгВ не проводиться. При множенні зі старших розрядів зсув суматора та регістра РгВ проводиться ненаскрізно. Приведення результату відбувається аналогічно до множення з молодших розрядів, тільки кінцевий результат міститься у суматорі. Оскільки значення множника В>0, то корегування результату не проводиться. На блок-схемі ці дії зображені по гілці В додатку В.

Приклад 5.5. Помножити два числа з фіксованою комою:

$A = -0.101$; $B = 0.111$.

$A_{пр} = -0.00010$; $B_{пр} = 0.111$;

$A_{доп} = 1.111011$; $B_{доп} = 0.111$.

СМ	РгВ	РгА	Примітки
0.000000	0.111	1.1110.11	[См]:=0; [Рг В]:=$[B]_{доп}$; [Рг А]:=$[A]_{доп}$;
0.000000	111-		[См]← 1; [Рг В]← 1;
+1.111011 11.111011 11.110110	11--		[См]:=[См]+[РгА]; [См]← 1; [Рг В]← 1;
+11.111011 11.110001 11.100010	1---		[См]:=[См]+[РгА]; [См]← 1; [Рг В]← 1;
+11.111011 11.011101			[См]:=[См]+[РгА];

$C_{доп} = 1.011101$

$C_{пр} = -0.100011$.

Відповідь: Отримали результат – від'ємне число -0.100011.

5.2.4 Множення зі старших розрядів за умови, що значення числа В < 0

Спочатку у РгВ записуємо доповняльний код числа В, у РгА – доповняльний код числа А, суматор (з подвійною розрядністю)

обнулюємо. Оскільки при множенні в доповняльних кодах беруть участь і знакові розряди, то при аналізі старшого розряду РгВ аналізуємо знаковий розряд. Оскільки в ньому "1", то проводимо операцію корегування: якщо число A>0, то до суматора додаємо значення $[A_{доп}]_{доп}$, якщо число A<0, то до суматора додаємо прямий код числа А. Потім проводиться зсув суматора та регістра РгВ вліво на один розряд і виконується операція множення аналогічно множенню у прямих кодах зі старших розрядів: якщо у старшому розряді РгВ "1", то до суматора додаємо $[A]_{доп}$ і проводимо зсув См та РгВ на один розряд вліво, якщо у старшому розряді РгВ – "0", то проводиться лише зсув См та РгВ на один розряд вліво. Ці операції проводяться циклічно, поки не проаналізують усі розряди РгВ. При аналізі останнього розряду РгВ зсув См та Рг В не проводять. Потім аналізують знак результату та отримують прямий код результату.

Приклад 5.6. Помножити числа з фіксованою комою:

A=-0.101; B= - 0.111.
$A_{пр}$= -0.000101; $B_{пр}$= -0.111;
$A_{доп}$= 1.111011; $B_{доп}$=1.001.

СМ	РгВ	РгА	РгС	Примітки
0.000000	1.001	1.111011	0.000101	[См]:=0; $[РгВ]:=[B]_{доп}$; $[РгА]:=[A]_{доп}$; $[РгС]:=[A]_{пр}$;
+0.000101				[См]:=[См]+[РгС];
0.000101				
0.001010	0.01-			[См]← 1; [РгВ]← 1;
0.010100	0.1--			[См]← 1; [РгВ]← 1;
0.101000	1.---			[См]← 1; [РгВ]← 1;
+1.111011				[См]:=[См]+[РгА].
0.100011				

Відповідь: С:=0.100011. Оскільки у знаковому розряді "0", то корегування результату не проводиться. Отже, результат отримали одразу у прямому коді.

Приклад 5.7. Помножити числа з фіксованою комою:

A=0.101; B= -0.111.
$A_{пр}$=0.000101; $B_{пр}$= -0.111;
$A_{доп}$=0.000101; $B_{доп}$ =1.001.
$[A_{доп}]_{доп}$ = 1.111011.

СМ	РгВ	РгА	РгС	Примітки
0.000000	1.001	0.000101	1.111011	[См]:=0; [РгВ]:=$[B]_{доп}$; [Рг А]:=$[A]_{доп}$; [РгС]:=$[A_{доп}]_{доп}$;
+1.111011 1.111011 1.110110	0.01-			[См]:=[См]+[РгС]; [См]← 1; [РгВ]← 1;
1.101100	0.1--			[См]← 1; [РгВ]← 1;
1.011000	1.---			[См]← 1; [РгВ]← 1;
+0.000101 1.011101				[См]:=[См]+[РгА].

$C_{доп}$=1,011101; $C_{пр}$= -0.100011.
Відповідь: -0.100011.

5.3 Прискорене множення двійкових чисел

5.3.1 Метод множення з розбиттям множника на дві частини

Для підвищення швидкості реалізації операції множення використовують прискорене множення, що є важливим, оскільки операція множення, як і додавання, зустрічається в програмах досить часто. Однак, час виконання операції множення в значно триваліший від часу виконання операції додавання. Тому продуктивність арифметико-логічного пристрою (АЛП) практично визначається часом виконання операції множення. Зрозуміло, що при проектуванні АЛП велика увага приділяється використанню методів прискореного множення. Це прискорення, як правило, супроводжується збільшенням апаратних витрат. Оскільки блок множення (БМ) складається з підсумовувального блока (ПБ) та блока місцевого керування (БМК), то в залежності від того, яка частина БМ ускладнюється, розрізняють логічні, апаратні та комбіновані способи прискорення.

До логічних методів належать: пропуск тактів додавання у випадках, коли чергова цифра множника нуль; згрупування розрядів множника; послідовне перетворення цифр множника.

До апаратних методів належать: ділення множника на дві частини: множення з запам'ятовуванням переносу та ін. Прискорене множення з розбиттям множника на дві частини можна виконувати як зі старших, так і з молодших розрядів.

Алгоритми виконання прискореного множення двійкових чисел наведені у додатку В (гілка Д).

5.3.1.1 Прискорене множення зі старших розрядів з розбиттям множника на дві частини

Такий метод множення вимагає два суматори однакової розрядності. У регістр Рг1 заносять старші розряди множника В, у регістр Рг2 – молодші; у РгА – код числа А. Одночасно аналізується старший розряд Рг1 та Рг2. Проводять відповідні дії у суматорах: якщо цей розряд "1", то до суматора додають зміст РгА та зсувають суматори та Рг1 і Рг2 вліво на один розряд; якщо розряд "0" –зсувають тільки СМ1, СМ2, Рг1, Рг2 вліво на один розряд. Ці операції, звісно, проводять до тих пір, поки не проаналізують усі розряди Рг1 (Рг2). Для отримання результату здійснюють додавання значень СМ1 та СМ2 (зсунутого вправо на стільки розрядів, скільки розрядів має Рг2 та Рг1), отриманий результат можна перетворювати за необхідністю у прямий код та нормалізувати. Зауважимо, що при множенні на суматорах доповняльного коду у Рг1 старший розряд – це знаковий розряд, що вказує на знак числа В. Отже, якщо у цьому розряді "0" (В>0), то корегування не проводиться (зсув вправо СМ1 та Рг1), а якщо у старшому розряді Рг1 є "1" (В<0), тоді проводимо корегування: до СМ 1 додаємо код $[A_{доп}]_{доп}$, якщо А>0 (приклад 5.9.), та код $[A]_{кор}$, якщо А < 0 (приклад 5.10). Далі операції проводяться за правилами множення з старших розрядів на суматорах доповняльного коду. Результат множення отримують шляхом додавання значень суматора СМ1 і значення суматора СМ2, зсунутого на стільки розрядів вправо, скільки розрядів має регістр Рг1 чи Рг2. Множення зі старших розрядів можна наочно проглянути на структурній блок-схемі по гілці D, E додатку В.

Приклад 5.8. Помножити прискореним методом числа А=0.1010 і В= - 0.1011 на суматорі прямого коду при діленні множника на частини.

СМ1	Рг1	СМ2	Рг2	Примітки
00000	10	00000	11	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди $B_{пр}$; [Рг2]:=мол. розряди $B_{пр}$; [РгА]:=$[A]_{пр}$.
+001010 001010 010100	 0	+001010 001010 010100	 1	[СМ1]:=[СМ1]+[РгА]; [СМ2]:=[СМ2]+[РгА]; [СМ1]←1; [СМ2]←1; [Рг1]←1; [Рг2]←1;
		+001010 011110		[СМ2]:=[СМ2]+[РгА].

Отримаємо результат множення:

[СМ 1]= 01010000
+[СМ 2]= 011110
01101110

Результат С=1.01101110

Знаковий розряд результату отримуємо за правилом: $S_D = S_A \oplus S_B = 0 \oplus 1 = 1$.

Відповідь: С=1.01101110

Приклад 5.9. Помножити числа А=0.1010 і В= -0.1011 на суматорі доповняльного коду прискореним методом множення, починаючи зі старших розрядів при діленні множника на дві частини.

$A_{пр}$= 0.00101; $B_{пр}$= -0.111;
$A_{доп}$=0.00101; $B_{доп}$=1.001;
$[A_{доп}]_{доп}$=1.11011.

СМ1	Рг1	СМ2	Рг2	Примітки
0.00000	10	00000	10	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди числа В; [Рг2]:=мол. розряди числа В; [РгА]:=$A_{доп}$; [РгС]:=$[A_{доп}]_{доп}$;
+1.11011 1.11011 1.10110	0	00000	1-	[СМ1]:=[СМ1]+[РгС]; [СМ1]←1; [Рг1]←1; [СМ2]←1; [Рг2]←1;
		+00101 00101		[СМ2]:=[СМ2]+[РгА];

Отримаємо результат множення:

[СМ 1]= 1.1011000
+[С1М 2]= 00101
1.1011101

Отже, результат множення $C_{доп}$=1.1011101. Перевівши результат у прямий код, отримали: С= -0.0100011.

Приклад 5.10. Помножити число А= -0.101 і число В= -0.111 на суматорі доповняльного коду прискореним методом множення з діленням множника на дві частини.

Операцію множення починати зі старших розрядів.

$A_{пр}$= -0.00101; $B_{пр}$= -0.111;
$A_{доп}$=1.11011; $B_{доп}$=1.001;
$A_{кор}$=1.11011.

СМ1	Рг1	СМ2	Рг2	Примітки
0.00000	10	0.00000	01	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди числа $B_{доп}$; [Рг2]:=мол. розряди числа $B_{доп}$; [РгА]:=$A_{доп}$; [РгС]:=$A_{кор}$;
+1.1110 1 1.11101 1.11010	0-	000000	1-	[СМ1]:=[СМ1]+[РгС]; [СМ1]←1; [Рг1]←1; [СМ2]←1; [Рг2]←1;
		+1.11011 1.11011		[СМ2]:=[СМ2]+[РгА];

СМ 1= 1.1101000
+СМ 2= 1.11011
С= 0.0100011

Оскільки у знаковому розряді результату отримали "0", то результат отримали у прямому коді.

Відповідь: С= 0.0100011.

Приклад 5.11. Помножити числа А= -0.1010 і В=0.1011 прискореним методом множення на суматорі доповняльного коду з діленням множника на дві частини. Операцію почати зі старших розрядів.

$A_{пр.}$= -0.001010 $B_{пр.}$=0.1011
$A_{доп.}$=1.110110 $B_{доп.}$=0.1011

СМ1	Рг1	СМ2	Рг2	Примітки
0.000000	10	000000	11	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди $B_{доп}$; [Рг2]:=мол. Розряди $B_{доп}$; [РгА]:=$A_{доп}$.
+1.110110 1.110110 1.101100	0-	+110110 110110 101100	1-	[СМ1]:=[СМ1]+[РгА]; [СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [Рг1]→1; [СМ2]→1; [Рг2]→1;
		+110110 100010		[СМ2]:=[СМ2]+[РгА].

СМ 1=1.101100
+СМ 2=1.11100010
$C_{доп.}$=1.10010010
$C_{пр}$=1.01101110

5.3.1.2 Прискорене множення з молодших розрядів з розбиттям множника на дві частини

Прискорене множення з молодших розрядів виконують аналогічно методу, описаному в п.5.3.1.1, з тією лише різницею, що аналізують розряди Рг1 та Рг2, починаючи з молодших розрядів та зсув См1 і Рг1 та См2 і Рг2 проводиться вправо наскрізно, тобто молодші розряди См1 записуються у старші розряди Рг1, а молодші розряди См2 – у старші розряди Рг2. Суматори СМ1 та СМ2 мають таку ж розрядність, як і число А. Корегування звісно виконують на останньому такті в См1, аналізуючи останній розряд Рг1. За правилами, описаними у п.5.3.1.1, якщо B<0, а A>0, то код корегування $A_{доп}$, якщо B<0, а A<0, то код корегування $A_{пр}$. Результат отримують при додаванні коду, що знаходиться у См2 та Рг2, зсунутого вліво на стільки розрядів, скільки розрядів у Рг1. Отриманий результат перетворюють за необхідністю в прямий код та нормалізують. На блок-схемі по гілці D0 показано всі вищеперераховані дії.

Приклад 5.12. Помножити число A=-0.1010 на B=0.1011 на суматорі прямого коду прискореним методом множення з молодших розрядів з використанням ділення множника на дві частини.

A=1010; B=1011.

СМ1	Рг1	СМ2	Рг2	Примітки
0000	10	0000	11	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди числа B; [Рг2]:=мол. розряди числа B; [РгА]:=A;
0000	01	+1010 1010 0101	01	[СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [СМ2]→1; [Рг1]→1; [Рг2]→1;
+1010 1010 0101	00	+1010 1111 0111	10	[СМ1]:=[СМ1]+[РгА]; [СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [СМ2]→1; [Рг1]→1; [Рг2]→1.

СМ1=01010000
+СМ2= 011110
С= 01101110

Визначимо знак результату: $S_C = S_A \oplus S_B = 1 \oplus 0 = 1$.
Отже, С=1.01101110.

Приклад 5.13. Помножити число А= -0.1010 і число В=0.1011 на суматорі доповняльного коду прискореним методом множення, починаючи з молодших розрядів, використовуючи метод ділення множника на дві частини.

A=1010; B=1011

СМ 1	Рг 1	СМ 2	Рг 2	Примітки
0.0000	10	0.0000	11	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди числа $B_{доп}$; [Рг2]:=мол. розряди числа $B_{доп}$; [РгА]:=$A_{доп}$;
0.0000	01	+1.0110 1.0110 1.1011	01	[СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [Рг1]→1; [СМ2]→1; [Рг2]→1;
+1.0110 1.0110 1.1011	00	+1.0110 1.0001 1.1000	10	[СМ1]:=[СМ1]+[РгА]; [СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [СМ2]→1; [Рг1]→1; [Рг2]→1.

СМ 1= 1.10110000
+СМ 2= 1.11100010
$C_{доп}$= 1.10010010
$C_{пр}$= -0.01101110

Відповідь: $C_{пр}$= - 0.01101110.

Приклад 5.14. Помножити число А=-0.1010 на число В=-0.1011 прискореним методом множення з діленням множника на дві частини. Операцію множення реалізувати на суматорі доповняльного коду, починаючи з молодших розрядів.

$A_{пр}$= -0.1010; $B_{пр}$= -0.1011;
$A_{доп}$=1.0110; $B_{доп}$=1.0101.

СМ1	Рг1	СМ2	Рг2	Примітки
0.0000	$\bar{1}.01$	0.0000	01	[СМ1]:=0; [СМ2]:=0; [Рг1]:=ст. розряди числа $B_{доп}$; [Рг2]:=мол. розряди числа $B_{доп}$; [РгА]:=$A_{доп}$; [РгВ]:=$A_{пр}$;
+1.0110 1.0110 1.1011	$0\bar{1}.0$	+1.0110 1.0110 1.1011	00	[СМ1]:=[СМ1]+[РгА]; [СМ2]:=[СМ2]+[РгА]; [СМ1]→1; [Рг1]→1; [СМ2]→1; [Рг2]→1;
1.1101	$10\bar{1}$	1.1101	10	[СМ1]→1; [Рг1]→1; [СМ2]→1; [Рг2]→1.
+0.1010 0.0111	10			Корегування результату [СМ1]:=[СМ1]+[РгВ];

СМ 1= 0.011110
+СМ 2= 1.11110110
$C_{пр}$= 0.01101110

Відповідь: $C_{пр}$= 0.01101110

Приклад 5.15. Помножити числа А=0.1010 і В=-0.1011 прискореним методом множення на суматорі доповняльного коду з діленням множника на дві частини. Операцію почати з молодших розрядів.

$A_{пр.}$=0.1010 $B_{пр.}$= -0. 1011
$A_{доп.}$=0.1010 $B_{доп.}$=1.0101
$A_{кор.}$=1.0110

СМ1	Рг1	СМ2	Рг2	Примітки
0.0000	$(\bar{1})$ 01	0000	01	[СМ1]:=0; [СМ2]:=0; [РгА]:=$A_{доп}$; [РгВ]:=$A_{кор}$; [Рг1]:=ст.розряди числа $B_{доп}$; [Рг2]:=мол.розряди число$B_{доп}$;
+0.1010 0.1010 0.0101	0 $(\bar{1})$0		00	[СМ1]:=[СМ 1]+[РгА]; [СМ2]:=[СМ 2]+[РгА]; [СМ1]→1; [СМ2]→1; [Рг1]→1; [Рг2]→1;
0.0010	10$(\bar{1})$	0010	10	[СМ1]→1; [Рг1]→1; [СМ2]→1; [Рг2]→1.
+1.0110 1.1000	10			Корегування результату [СМ1]:=[СМ1]+[РгВ];

СМ 1 = 1.100010
+СМ 2 = .001010
$C_{доп}$= 1.10010010
$C_{пр}$= 1.01101110

Відповідь: $C_{пр}$=1.01101110.

5.3.2 Прискорене множення із запам'ятовуванням проміжної суми та проміжного переносу

5.3.2.1 Прискорене множення з молодших розрядів з запам'ятовуванням проміжної суми і проміжного переносу

Прискорене множення можна виконувати ще одним методом, множення з запам'ятовуванням проміжних суми та переносу. Розбиття множника не відбувається. Розглянемо множення починаючи з молодших розрядів. Аналізують послідовно три розряди РгВ, починаючи з молодших. Додають три часткові добутки без розповсюдження переносу, сформувавши при цьому проміжну суму за правилом ⊕ та проміжний перенос. Потім додають без розповсюдження переносу проміжну суму, проміжний перенос та наступний частковий добуток, сформувавши чергову проміжну суму та проміжний перенос. На останньому кроці додають проміжну суму та проміжний перенос; та при множенні чисел з плаваючою комою нормалізують результат. Гілка D, F, M блок-схеми додатку В ілюструє цей алгоритм.

Приклад 5.16. Помножити числа A=0.1010 і B=0.1011 прискореним методом множення з формуванням проміжної суми і проміжного переносу. Операцію множення починати з молодших розрядів.

1010
+1010
0000
Si = 011110
+Pi = 000000
1010
Si=1001110
+ Pi =0100000
1101110

Визначимо знак результату : $S_C = S_A \oplus S_B = 0 \oplus 0 = 0$
Відповідь : C = 0.1101110.

5.3.2.2 Прискорене множення зі старших розрядів із запам'ятовуванням проміжної суми і проміжного переносу

Якщо виконується множення з старших розрядів, то результат знаходиться тільки в суматорі. Множення з старших розрядів виконується аналогічно множенню з молодших розрядів; відмінністю є те, що аналіз розрядів РгВ проводиться починаючи зі старших розрядів.

Приклад 5.17. Помножити числа : A=0.1010 і B=0.1011 прискореним методом множення з формуванням проміжної суми переносу. Операцію множення починати зі старших розрядів.

```
      1010
    * 1011
      1010
    +  0000
         1010
  Si = 100010
 +Pi = 010000
         1010
  Si =1101110
+ Pi = 0000000
       1101110
```

Визначимо знак результату : $S_C = S_A \oplus S_B = 0 \oplus 0 = 0$

Відповідь: C=0.1101110.

5.3.3 Прискорене множення з групуванням розрядів множника

Метод з групуванням розрядів множника виконує перехід до надлишкової двійкової системи числення з цифрами 0, 1, $\bar{1}$, що дозволяє зменшити кількість одиниць в зображенні множника, та скоротити кількість операцій додавання, що дозволяє отримати виграш у швидкості виконання операції. Алгоритм реалізації цієї операції наводиться у додатку В (гілки D, F, M).

5.3.3.1 Прискорене множення з молодших розрядів з групуванням розрядів множника

Правила перетворення множника при множенні чисел, починаючи з молодших розрядів, при групуванні враховують такі дії: комбінації виду 00, 10, 01 не перетворюються; комбінація виду 11 замінюється

комбінацією вигляду 1.0 $\bar{1}$, що означає запам'ятовування одиниці для пари цифр множника.

Якщо молодший розряд регістра РгВ містить одиницю, до суматора додають прямий код числа А і проводять наскрізний зсув вправо на один розряд См і РгВ. Якщо ж молодший розряд регістру РгВ містить $\bar{1}$, то до суматора додають доповняльний код числа А і проводять наскрізний зсув регістра РгВ і суматора; якщо молодший розряд РгВ дорівнює нулю, то проводять зсув РгВ і суматора на один розряд вправо. За необхідністю нормалізують результат. Суматор і регістр РгВ містять вихідний результат.

Приклад 5.18. Помножити число А=0.1010 на число В=0.1011 прискореним методом з групуванням розрядів множника. Операцію множення можна починати з молодших розрядів.

$$A_{пр} = 1010; \quad B_{пр} = 1011;$$
$$A_{доп} = 0110$$

Групуємо розряди множника В: $10\underline{11}$

$$\underline{10\bar{1}}$$
$$10\bar{1}$$
$$B=10\bar{1}0\bar{1}$$

См	РгВ	Примітки
0000	$10\ \bar{1}0\ \bar{1}$	[См]:=0; [РгВ]:=$B_{пр}$; [РгА]:=$A_{доп}$; [РгС]:=$A_{пр}$.
$+\underline{0110}$ 0110		[См]:=[См]+[РгА];
1011	$0101\bar{0}$	[См]→1 ;[РгВ]→1; Оскільки додавали доповняльні коди, то при зсуві вправо у старші розряди суматора записується "1"
1101	$1010\bar{1}$	[См]→1 ;[РгВ]→1;
$+\underline{0110}$ 0011		[См]:=[См]+[РгА];
1001	11010	[См]→1 ;[РгВ]→1;
1100	11101	[См]→1 ;[РгВ]1;
$+\underline{1010}$ 0110		[См]:=[См]+[РгС];
0011	01110	[См]→1 ;[РгВ]→1; Оскільки додавали $A_{пр}$, то при зсуві вправо у старші розряди суматора записується "0"

Відповідь: С=0.1101110. Знак результату визначаємо за правилом: $S_C = S_A \oplus S_B = 0 \oplus 0 = 0$, тобто результат – число додатне.

5.3.3.2 Прискорене множення зі старших розрядів з групуванням розрядів множника

Прискорене множення з групуванням розрядів множника, починаючи з старших розрядів включає такі дії: розряди множника B групують по два і роблять заміну кожної одиниці комбінацією 1. $\bar{1}$, розрядність суматора, зрозуміло, повинна бути вдвічі більшою, в регістрі A записуємо прямий код множника A, в РгВ – доповняльний код A, а в регістрі C міститься отриманий код $B_{тр}$.

В залежності від старшого розряду регістра C виконують такі операції:

"1" – до суматора додають значення регістра A та проводять зсув суматора та РгС вліво на один розряд вліво.

"$\bar{1}$"– до суматора додають значення регістра B та проводять зсув суматора та РгС вліво на один розряд вліво

"0" – проводять зсув суматора і РгС вліво на один розряд.

Результат буде міститися в суматорі.

Приклад 5.19. Помножити число A=0.1010 на число B=0.1011 прискореним методом з групуванням розрядів множника. Операцію множення можна починати з старших розрядів.

$A_{пр}$=00001010; B=1011; $A_{доп}$=11110110;

Проведемо групування розрядів множника:

B:	1	0	1	1
			1	$\bar{1}$
		1	$\bar{1}$	
	1	$\bar{1}$		
B=1	$\bar{1}$	1	0	$\bar{1}$

См	Рг В	Примітки
00000000	1 $\bar{1}$10 $\bar{1}$	[См]:=0; [РгС]:=$A_{доп}$; [РгА]:=$A_{пр}$; [РгВ]:=Впр.
00001010 00001010		[См]:=[См]+[РгА];
00010100	$\bar{1}$10$\bar{1}$	[См]←1 ; [Рг В]←1;
11110110 00001010		[См]:=[См]+[РгВ];
00010100	10 $\bar{1}$ - -	[См]←1 ; [Рг В]←1;
00001010 00011110		[См]:=[См]+[РгА];
00111100	0 $\bar{1}$ - - -	[См]←1 ; [РгВ]←1;
01111000	$\bar{1}$ - - - -	[См]←1 ; [РгВ]←1;
11110110 01101110		[См]:=[См]+[РгС];

Знак результату визначаємо за правилом: $S_C = S_A \oplus S_B = 0 \oplus 0 = 0$. Отже, результат є числом додатним.
Відповідь: C=0.01101110.

5.4 Особливості множення чисел, що подані у формі з плаваючою комою

При множенні чисел з плаваючою комою користуються формулою:

$A \cdot B = m_a \cdot 2^{Pa} \cdot m_b \cdot 2^{Pb} = (m_a \cdot m_b) \cdot 2^{(Pa+Pb)}$, що обумовлює виконання алгоритму:

1. Додавання порядків.
2. Множення мантис операндів за правилами множення двійкових чисел з фіксованою комою.
3. Нормалізація мантиси добутку, оскільки мантиса в загальному випадку може мати ненормалізовану форму, та корекція порядку.

При нормалізації мантиси проводять її зсув вліво на необхідну кількість розрядів, при чому корегування порядку полягає у зменшенні його на таку ж кількість розрядів.

Приклад 5.20. Помножити число $A=-0.11 \cdot 2^{001}$ на число $B=0.100 \cdot 2^{110}$ простим методом множення на суматорі доповняльного коду, починаючи з молодших розрядів.

$m_{Апр.} = -0.011$; $m_{Впр.}=0.100$; $P_{Апр.}=0.101$;

$m_{Адоп.}=1.101$; $m_{Вдоп.}=0.100$; $P_{Впр.}=0.110$.

Визначимо порядок результату:

$$\begin{array}{r} 0.001 \\ +\underline{0.110} \\ P_{Сдоп.} = 0.111 \end{array}$$

Визначимо мантису результату:

СМ	Рг С	Примітки
0.000	100	[СМ]:=0; [РгА]:=$m_{Адоп.}$; РгС=$m_{Вдоп.}$;
0.000	010	[СМ]→1; [РгС]→1;
0.000	001	[СМ]→1; [РгС]→1;
+1.101 1.101		[СМ]:=[СМ]+[РгА];
1.110	100	[СМ]→1; [РгС]→1;

$m_{Сдоп.}$=1.110100
$m_{Вдоп.}$=1.001100

$C_{пр.}$= -0.001100·2^{111}.

Нормалізуємо результат:

$C_{пр.}$= -0.110000·2^{101}.

Відповідь: $C_{пр.}$= -0.110000·2^{101} .

Контрольні запитання і задачі

1. Які методи множення двійкових чисел Ви знаєте ?

2. Наведіть блок-схему узагальненого алгоритму виконання операції множення двійкових чисел з плаваючою комою.

3. Порівняйте методи множення, починаючи зі старших та з молодших розрядів на суматорі прямого та доповняльного коду з точки зору апаратних затрат на реалізацію.

4. Виконайте операцію множення чисел у двійковій системі числення на суматорі прямого коду, починаючи зі старших та з молодших розрядів:

A=0.100101 ; - 0.00101101 ; - 0.01011;
B=-0.001010 ; - 0.00100101 ; - 0.00111.

5. Виконайте операцію множення чисел у двійковій системі числення на суматорі доповняльного коду, починаючи: з молодших; зі старших розрядів:

A=0.100101 ; - 0.00101101 ; - 0.01011;
B=-0.001010 ; - 0.00100101 ; - 0.00111.

6. Виконайте операцію прискореного множення шляхом ділення множника на дві частини :

A=0.100101 ; - 0.00101101 ; - 0.01011;
B=-0.001010 ; - 0.00100101 ; - 0.00111.

Операцію множення виконайте на суматорі прямого коду, починаючи зі старших і з молодших розрядів.

7. Виконайте операцію прискореного множення шляхом ділення множника на дві частини чисел :

A = 0.100101 ; - 0.00101101 ; - 0.01011;
B = - 0.001010 ; - 0.00100101 ; - 0.00111.

Операцію множення виконайте на суматорі доповняльного коду, починаючи зі старших і з молодших розрядів.

8. Помножте числа : A= 0.00101101 ; 0.01011;
B= 0.00101101 ; 0.00111

прискореним методом множення з групуванням розрядів множника; операцію виконати, починаючи зі старших і з молодших розрядів.

9. Помножте числа : A= 0.00101101 ; 0.01011;

B= 0.00100101 ; 0.00111

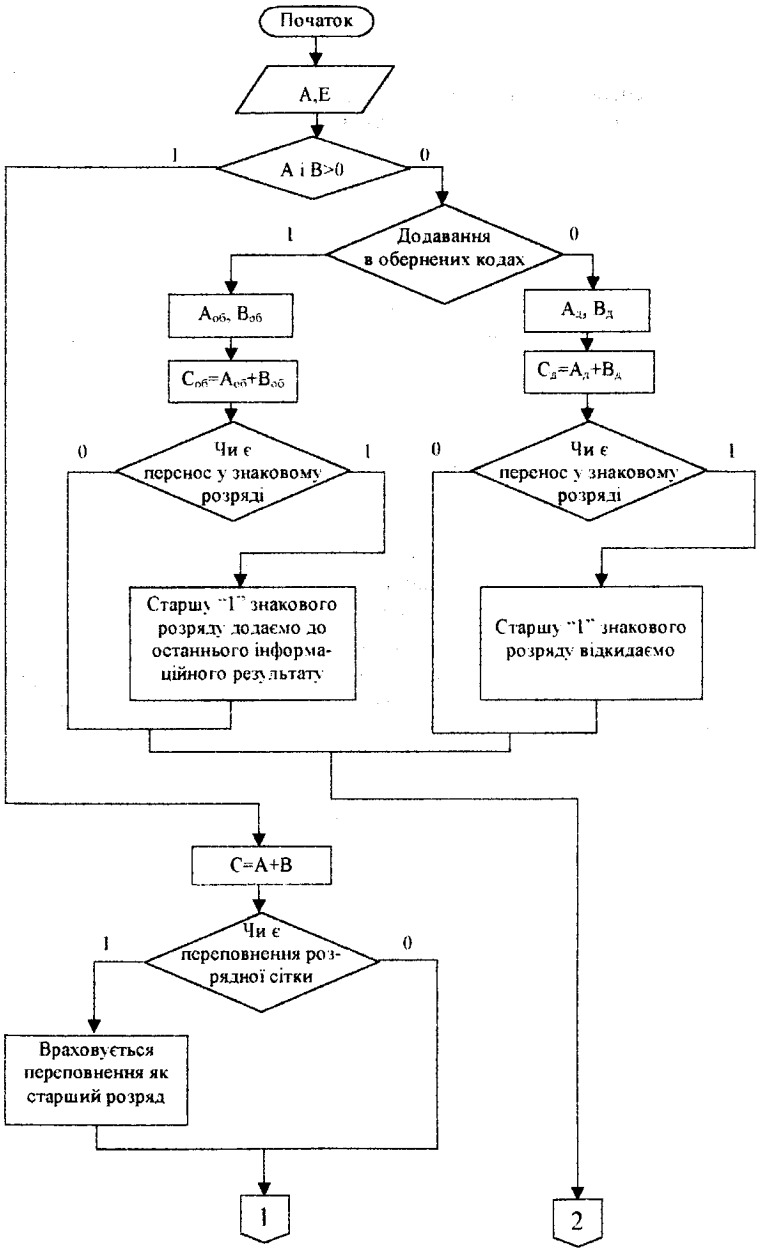
прискореним методом множення з формуванням проміжної суми і проміжного переносу, починаючи зі старших і з молодших розрядів.

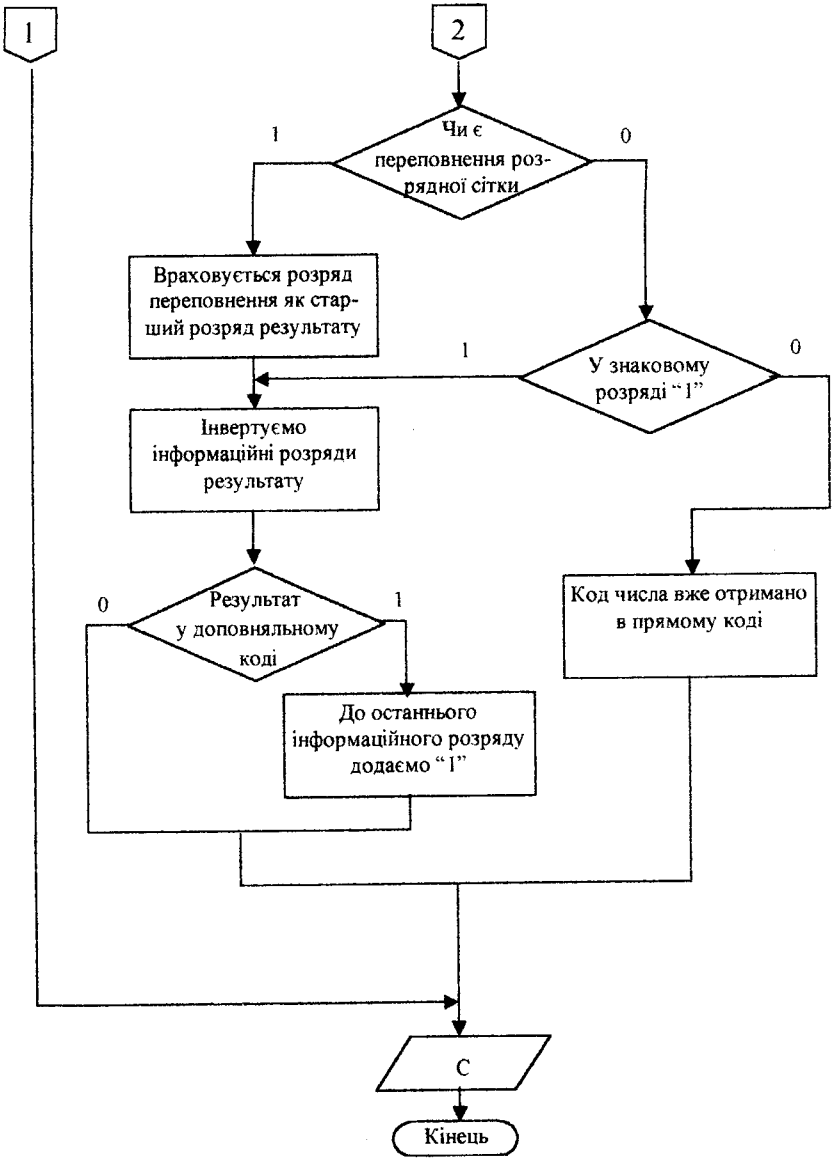
10. Помножте числа: $A=-0.1001*2^{100}$; $B=-0.1110*2^{-101}$ на суматорі прямого та доповняльного кодів, починаючи зі старших і з молодших розрядів.

Література

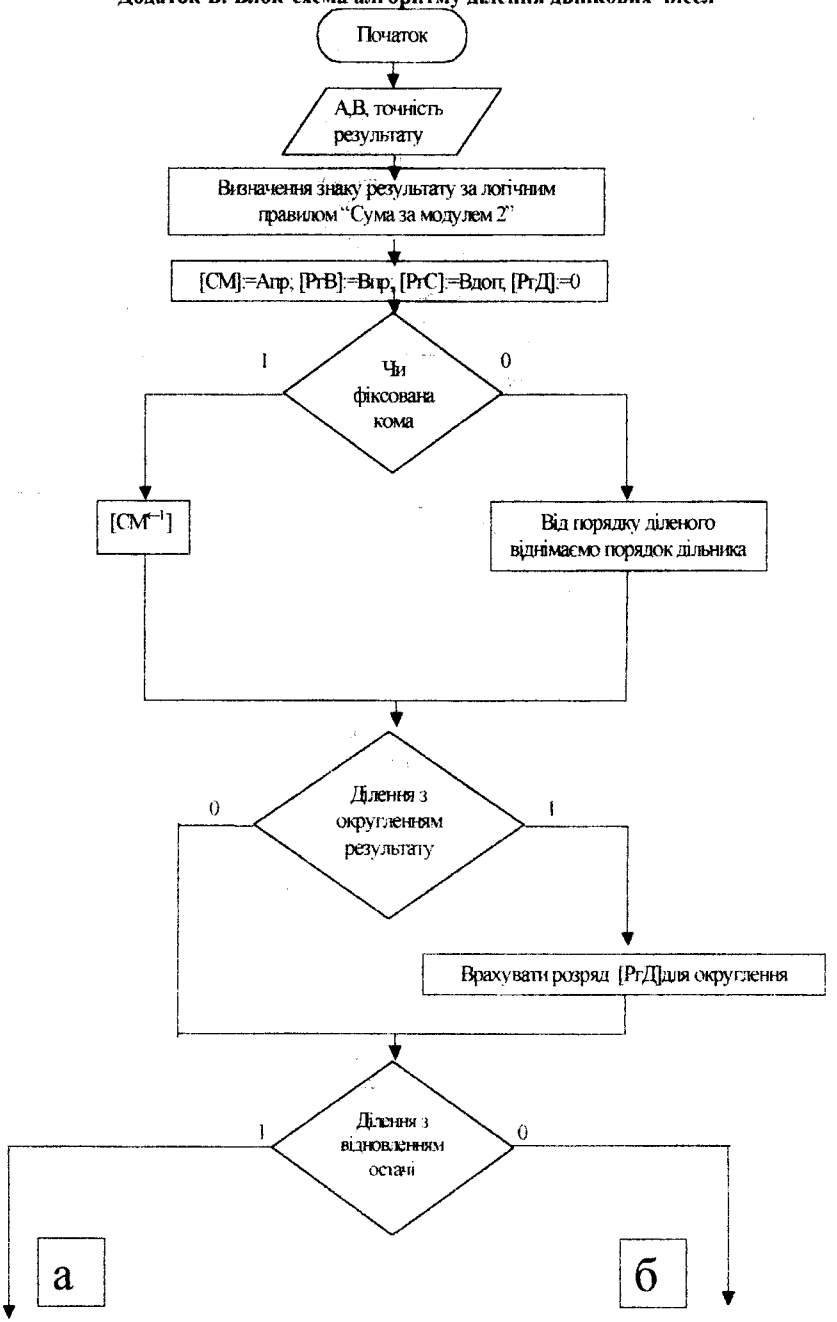
1. Самофалов К.Г., Корнейчук В.И., Тарасенко В.П. Цифровые ЭВМ: Теория и проектирование.– К.:Вища шк., 1989.–424с.
2. Савельев А.Я. Прикладная теория цифровых автоматов.–М.: Высш.шк., 1987.–272с.
3. Прикладная теория цифровых автоматов / К.Г. Самофалов, А.М.Романкевич, В.Н.Валуйский и др.–К.: Вища шк., 1987.–224с.
4. Каган Б.М. Электронные вычислительные машины и системы.–М.: Энергоатомиздат, 1985.– 552с.
5. Савельев А.Я. Арифметические и логические основы цифровых автоматов: Учебник.– М.: Высш.шк., 1980.–255с.
6. Лысиков В.К. Арифметические и логические основы цифровых автоматов.– Минск: Вышэйшая шк., 1980.–382с.
7. Лужецкий В.А. Методические указания к практическим занятиям по курсу «Прикладная теория цифровых автоматов».– Винница: ВПИ, 1986.–60с.

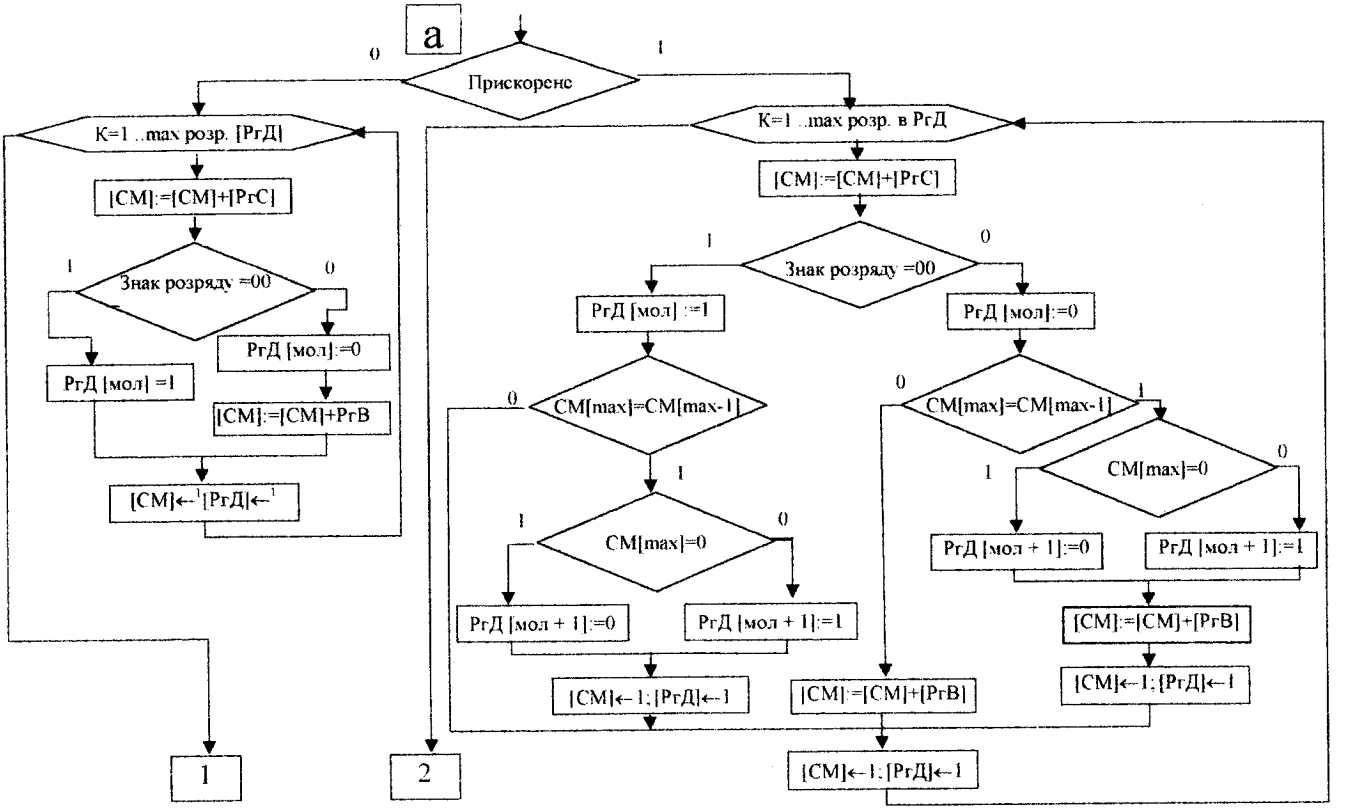
Додаток А. Блок- схема алгоритму додавання двійкових чисел

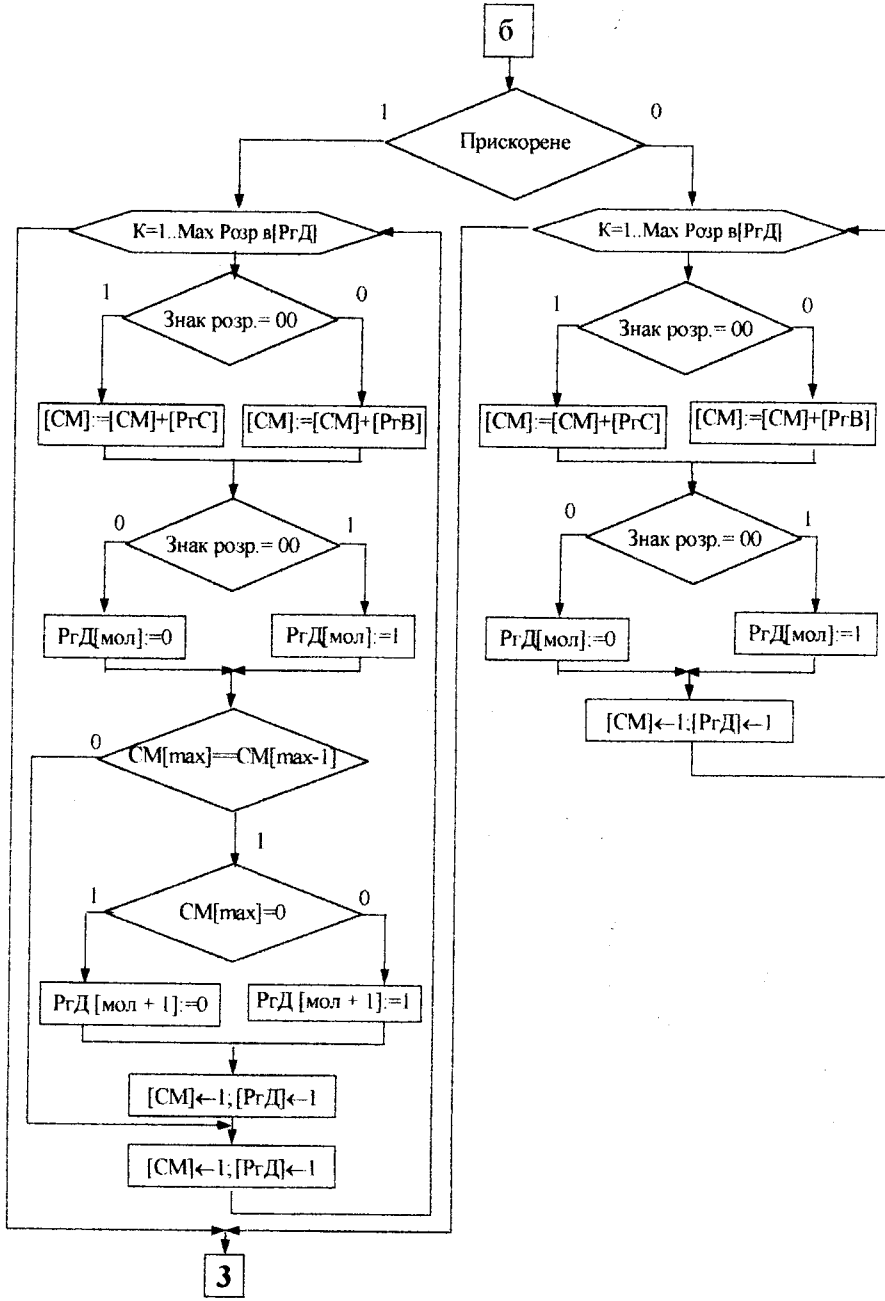


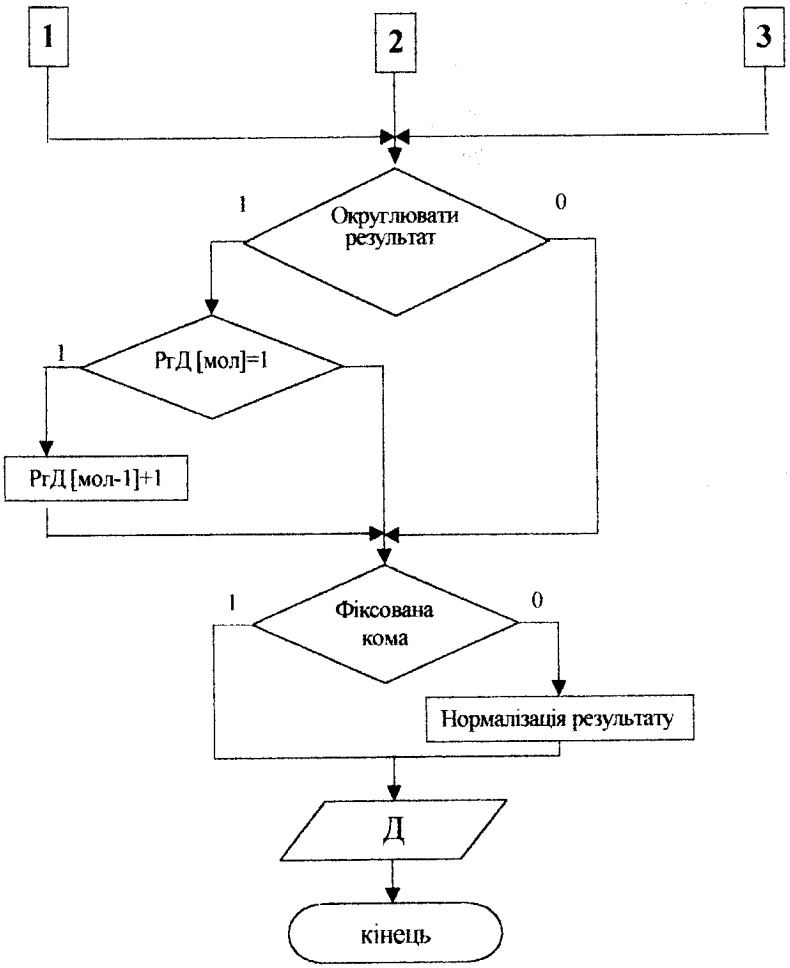

1
2
Чи є переповнення розрядної сітки
1
0
Враховується розряд переповнення як старший розряд результату
У знаковому розряді "1"
1
0
Інвертуємо інформаційні розряди результату
Результат у доповняльному коді
0
1
До останнього інформаційного розряду додаємо "1"
Код числа вже отримано в прямому коді
C
Кінець

Додаток Б. Блок-схема алгоритму ділення двійкових чисел

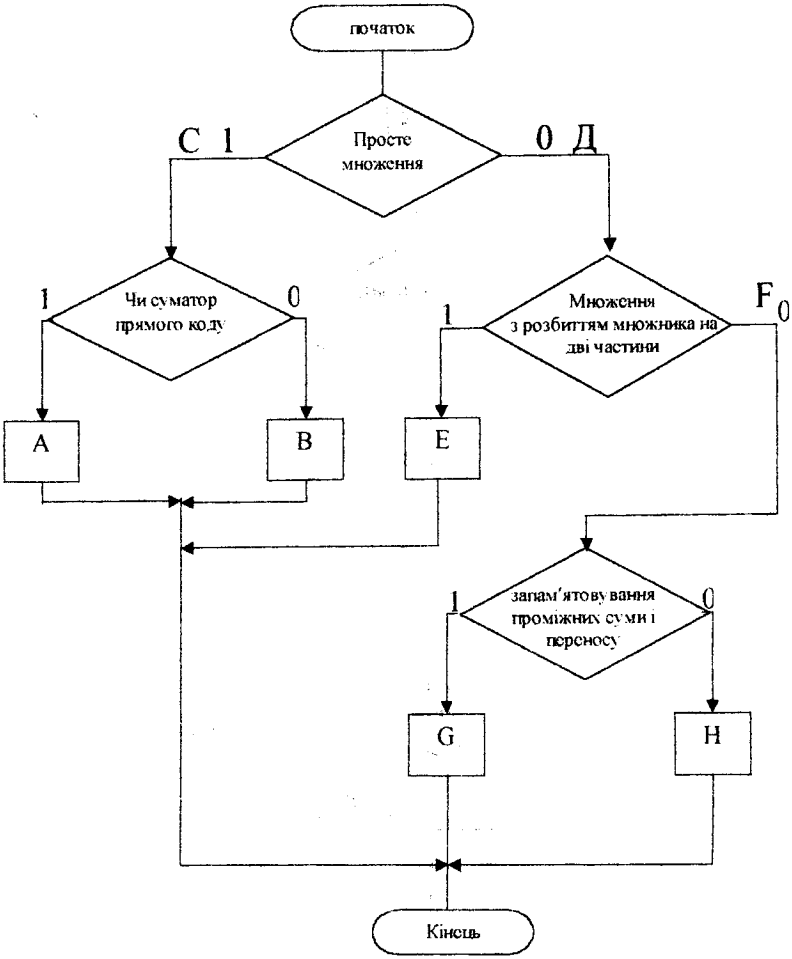



a
Прискорене
0
1
K=1 ..max розр. [РгД]
[СМ]:=[СМ]+[РгС]
Знак розряду =00
РгД [мол] =1
РгД [мол]:=0
[СМ]:=[СМ]+РгВ
[СМ]←1;[РгД]←1
1
K=1 ..max розр. в РгД
[СМ]:=[СМ]+[РгС]
Знак розряду =00
РгД [мол] :=1
РгД [мол]:=0
CM[max]=CM[max-1]
CM[max]=0
РгД [мол + 1]:=0
РгД [мол + 1]:=1
[СМ]←1;[РгД]←1
[СМ]:=[СМ]+[РгВ]
2


б
Прискорене
1
0
К=1..Max Розр в[РгД]
Знак розр.= 00
[СМ]:=[СМ]+[РгС]
[СМ]:=[СМ]+[РгВ]
РгД[мол]:=0
РгД[мол]:=1
СМ[max]=СМ[max-1]
СМ[max]=0
РгД [мол + 1]:=0
РгД [мол + 1]:=1
[СМ]←1;[РгД]←1
[СМ]←1;[РгД]←1
К=1..Max Розр в[РгД]
Знак розр.= 00
[СМ]:=[СМ]+[РгС]
[СМ]:=[СМ]+[РгВ]
РгД[мол]:=0
РгД[мол]:=1
[СМ]←1;[РгД]←1
3


1
2
3
Округлювати результат
1
0
РтД [мол]=1
1
РтД [мол-1]+1
Фіксована кома
1
0
Нормалізація результату
Д
кінець

Додаток В. Загальний алгоритм вибору методу множення двійкових чисел



С – методи простого множення.

С, А – методи простого множення на суматорі прямого коду.

С, В – методи простого множення на суматорі доповняльного коду.

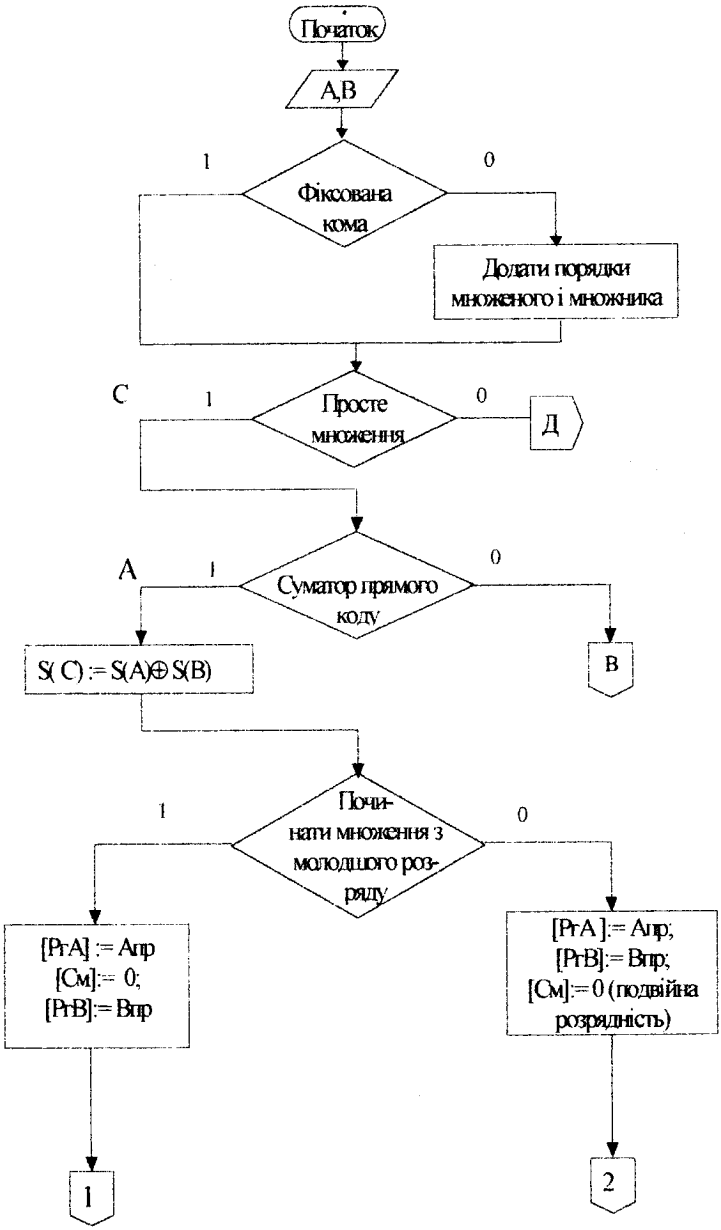
Д – методи прискореного множення.

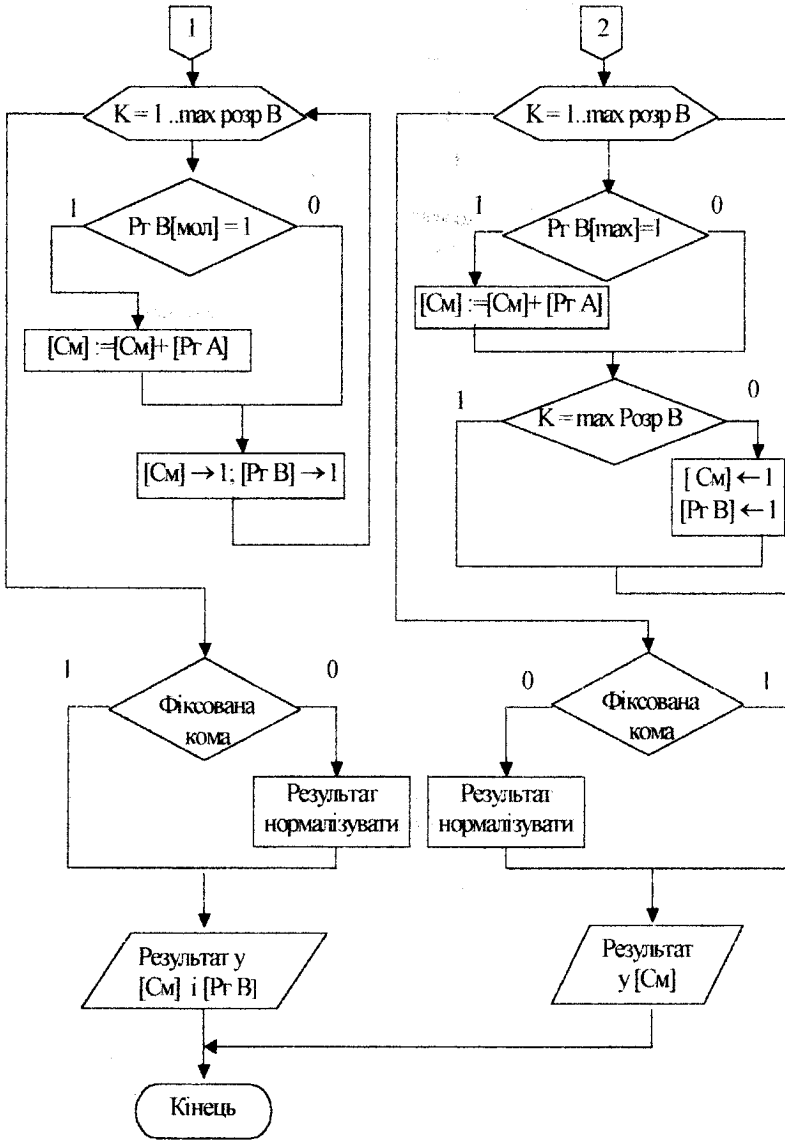
Д, Е – методи прискореного множення з розбиттям множника на дві частини.

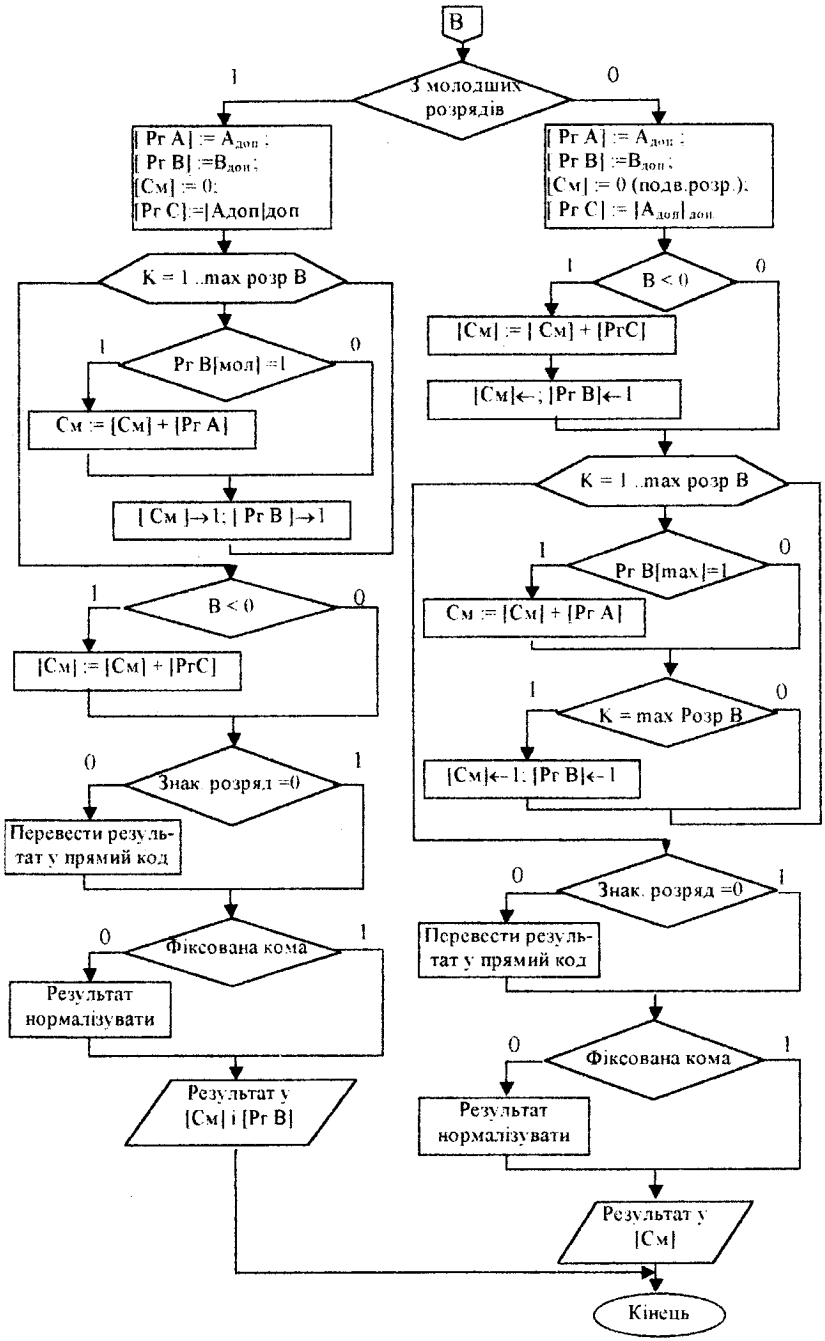
Д, F, G – методи прискореного множення із запам'ятовуванням проміжної суми і проміжного переносу.

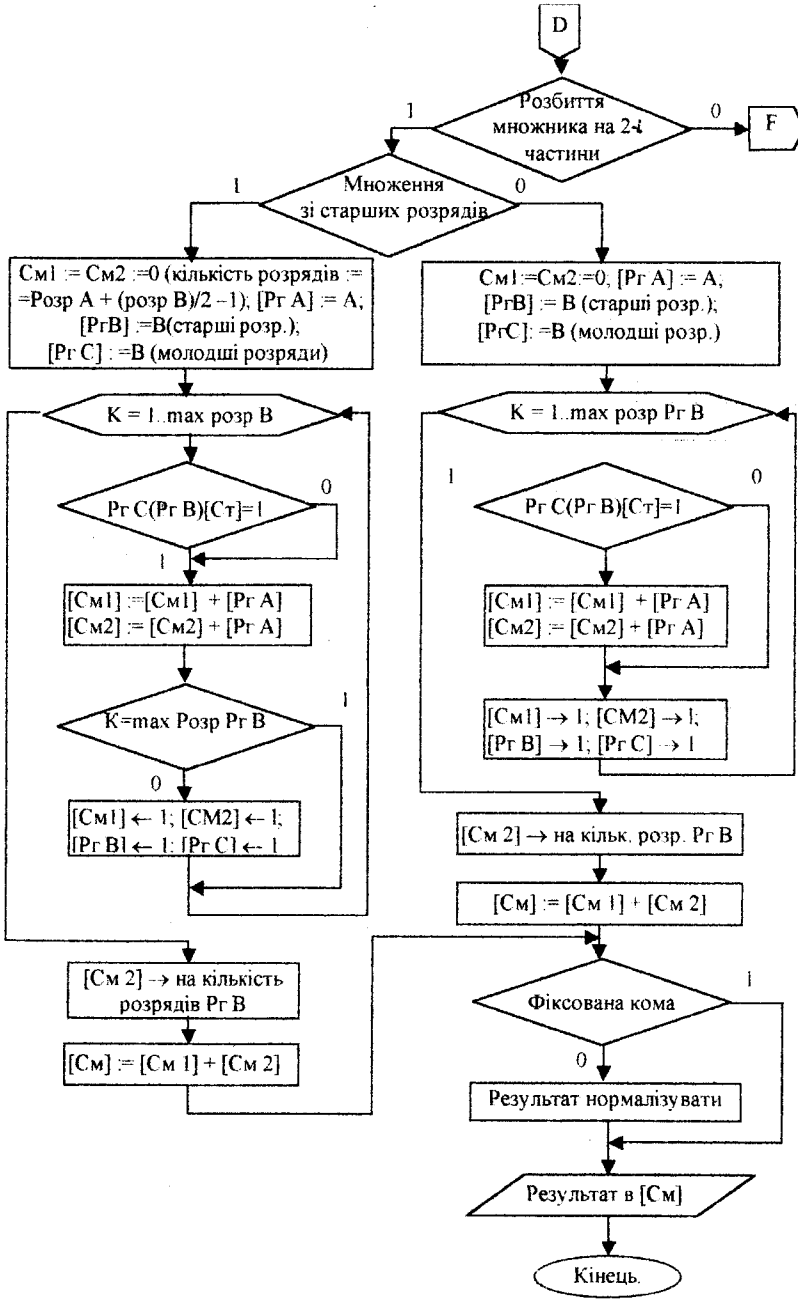
Д, F, H – методи прискореного множення з групуванням розрядів множника.

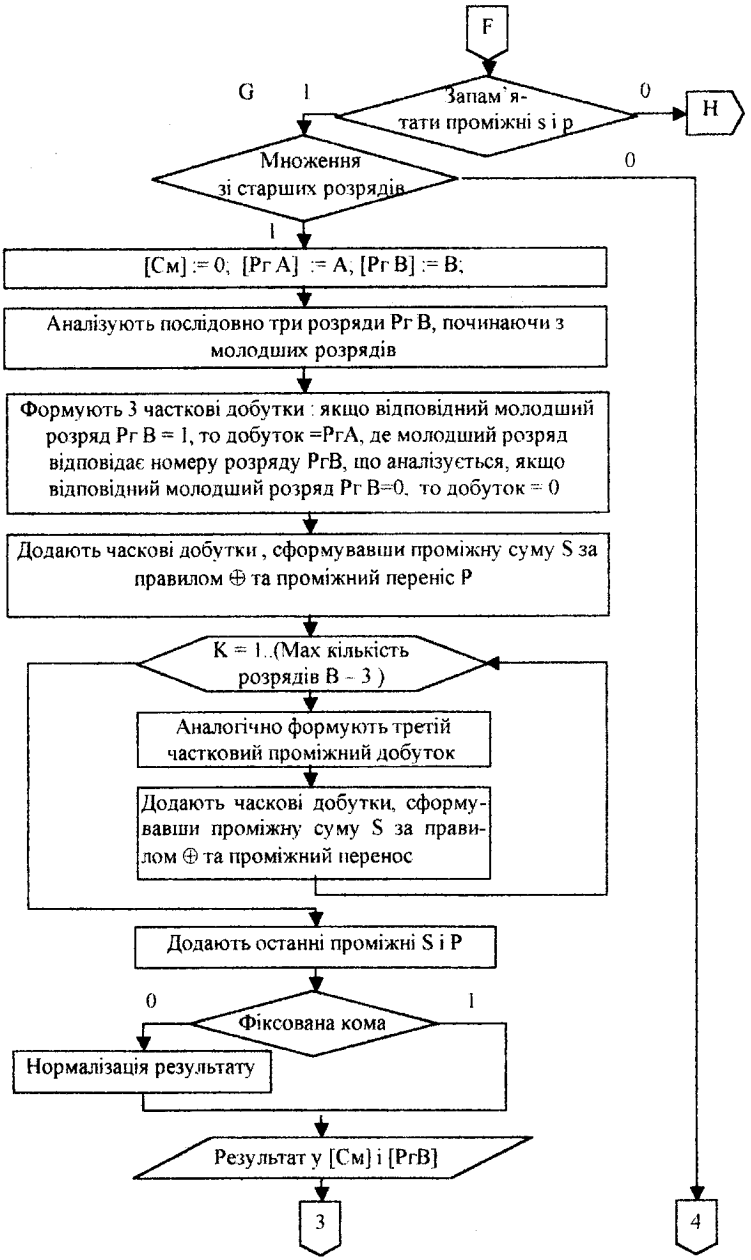
Множення двійкових чисел

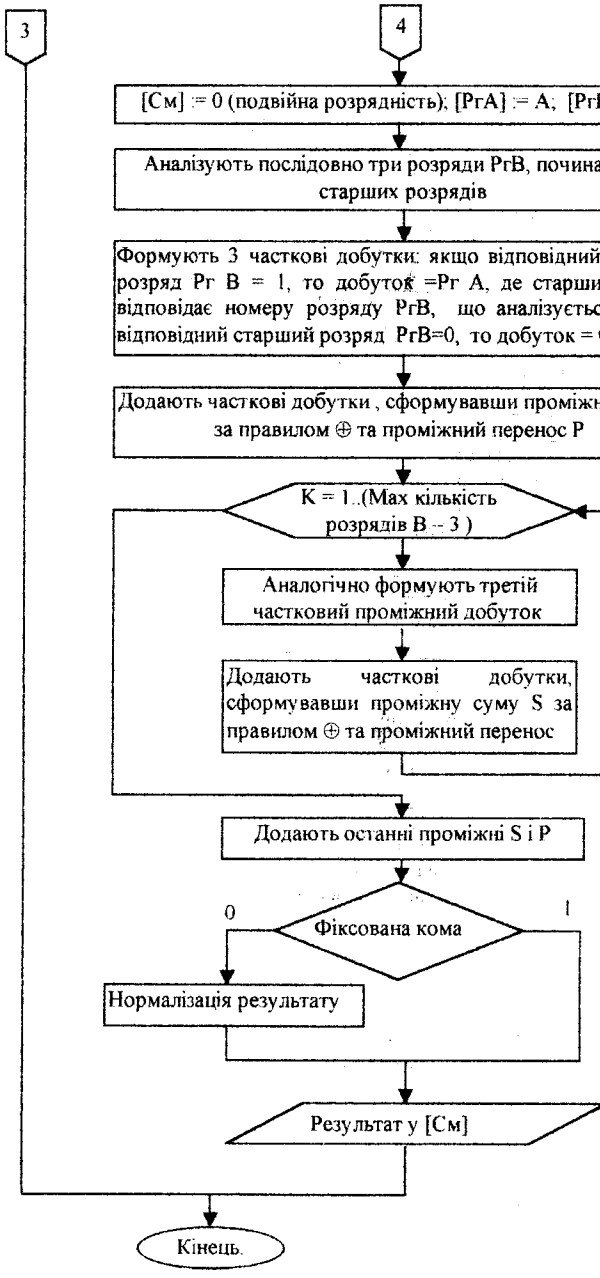


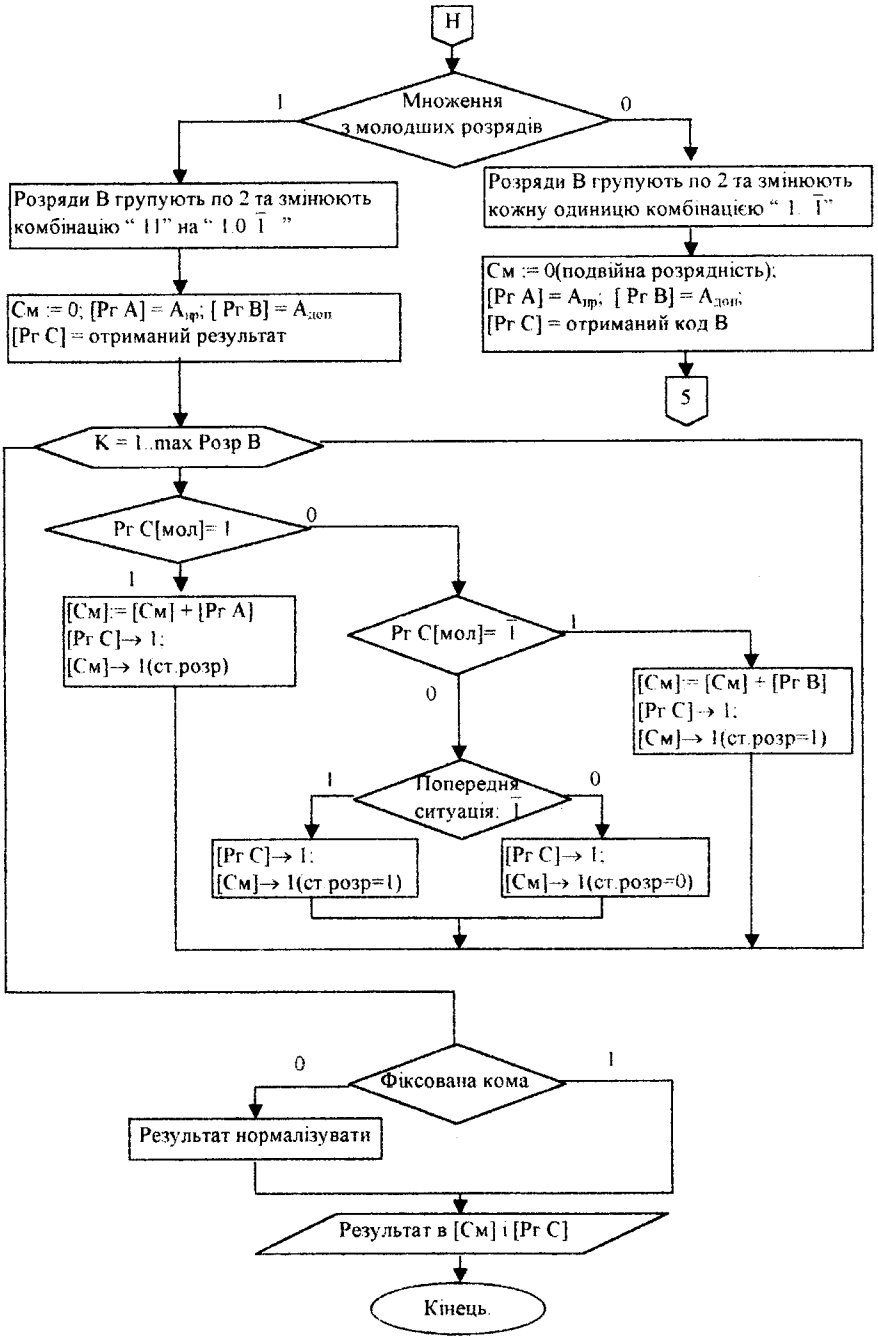

1
K = 1 ..max розр B
1
Рг B[мол] = 1
0
[См] :=[См]+ [Рг A]
[См] → 1; [Рг B] → 1
2
K = 1..max розр B
1
Рг B[max]=1
0
[См] :=[См]+ [Рг A]
1
K = max Розр B
0
[См] ← 1
[Рг B] ← 1
1
Фіксована кома
0
Результат нормалізувати
0
Фіксована кома
1
Результат нормалізувати
Результат у [См] і [Рг B]
Результат у [См]
Кінець

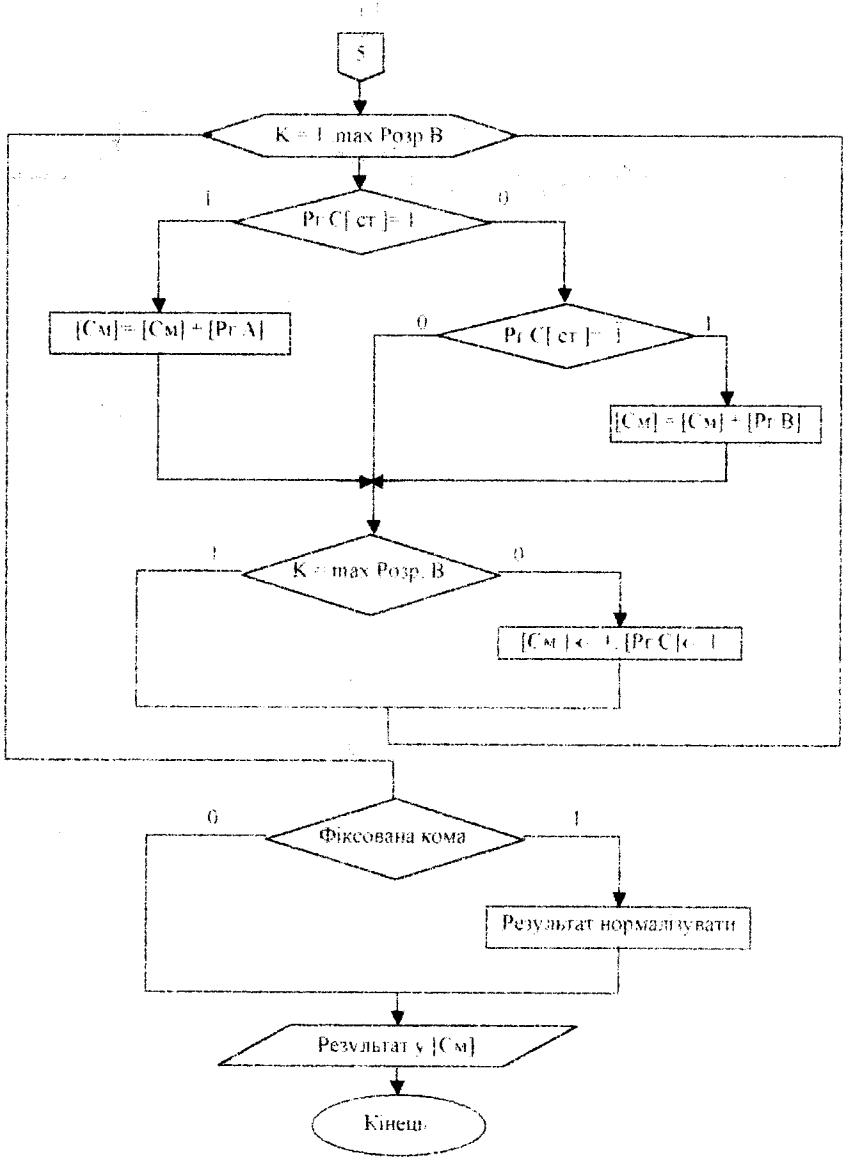

В
3 молодших розрядів
1
0
| Рг А| := Адоп ;
| Рг В| :=Вдоп ;
|См| := 0;
|Рг С|:=|Адоп|доп
| Рг А| := Адоп ;
| Рг В| :=Вдоп ;
|См| := 0 (подв.розр.);
| Рг С| := |Адоп|доп
К = 1 ..max розр В
Рг В|мол| =1
См := |См| + |Рг А|
| См |→1; | Рг В |→1
В < 0
|См| := |См| + |РгС|
Знак розряд =0
Перевести результат у прямий код
Фіксована кома
Результат нормалізувати
Результат у |См| і |Рг В|
В < 0
|См| := | См| + |РгС|
|См|←; |Рг В|←1
К = 1 ..max розр В
Рг В|max|=1
См := |См| + |Рг А|
К = max Розр В
|См|←1; |Рг В|←1
Знак. розряд =0
Перевести результат у прямий код
Фіксована кома
Результат нормалізувати
Результат у |См|
Кінець


D
Розбиття множника на 2-і частини
1
0
F
Множення зі старших розрядів
1
0
См1 := См2 :=0 (кількість розрядів := =Розр А + (розр В)/2 –1); [Рг А] := А; [РгВ] :=В(старші розр.); [Рг С] : =В (молодші розряди)
См1:=См2:=0; [Рг А] := А; [РгВ] := В (старші розр.); [РгС]: =В (молодші розр.)
К = 1..max розр В
К = 1..max розр Рг В
Рг С(Рг В)[Ст]=1
0
1
[См1] :=[См1] + [Рг А] [См2] := [См2] + [Рг А]
К=max Розр Рг В
[См1] ← 1; [СМ2] ← 1; [Рг В] ← 1; [Рг С] ← 1
[См1] → 1; [СМ2] → 1; [Рг В] → 1; [Рг С] → 1
[См 2] → на кількість розрядів Рг В
[См 2] → на кільк. розр. Рг В
[См] := [См 1] + [См 2]
Фіксована кома
Результат нормалізувати
Результат в [См]
Кінець.


F
G
1
Запам'я-
тати проміжні s і p
0
H
Множення
зі старших розрядів
0
1
[См] := 0; [Рг А] := А; [Рг В] := В;
Аналізують послідовно три розряди Рг В, починаючи з молодших розрядів
Формують 3 часткові добутки : якщо відповідний молодший розряд Рг В = 1, то добуток =РгА, де молодший розряд відповідає номеру розряду РгВ, що аналізується, якщо відповідний молодший розряд Рг В=0, то добуток = 0
Додають часкові добутки , сформувавши проміжну суму S за правилом ⊕ та проміжний переніс Р
K = 1..(Мах кількість розрядів В - 3)
Аналогічно формують третій частковий проміжний добуток
Додають часкові добутки, сформувавши проміжну суму S за правилом ⊕ та проміжний перенос
Додають останні проміжні S і P
0
Фіксована кома
1
Нормалізація результату
Результат у [См] і [РгВ]
3
4


3
4
[См] := 0 (подвійна розрядність); [РгА] := А; [РгВ] := В
Аналізують послідовно три розряди РгВ, починаючи зі старших розрядів
Формують 3 часткові добутки: якщо відповідний старший розряд Рг В = 1, то добуток =Рг А, де старший розряд відповідає номеру розряду РгВ, що аналізується , якщо відповідний старший розряд РгВ=0, то добуток = 0
Додають часткові добутки , сформувавши проміжну суму S за правилом ⊕ та проміжний перенос Р
K = 1..(Max кількість розрядів В -- 3)
Аналогічно формують третій частковий проміжний добуток
Додають часткові добутки, сформувавши проміжну суму S за правилом ⊕ та проміжний перенос
Додають останні проміжні S і P
0
Фіксована кома
1
Нормалізація результату
Результат у [См]
Кінець.


Н
Множення з молодших розрядів
1
0
Розряди В групують по 2 та змінюють комбінацію " 11" на " 1 0 1 "
Розряди В групують по 2 та змінюють кожну одиницю комбінацією " 1 1"
См := 0; [Рг А] = А_пр; [Рг В] = А_доп
[Рг С] = отриманий результат
См := 0(подвійна розрядність);
[Рг А] = А_пр; [Рг В] = А_доп;
[Рг С] = отриманий код В
5
К = 1..max Розр В
Рг С[мол]= 1
[См]:= [См] + [Рг А]
[Рг С]→ 1;
[См]→ 1(ст.розр)
Рг С[мол]= 1
[См]:= [См] + [Рг В]
[Рг С]→ 1;
[См]→ 1(ст.розр=1)
Попередня ситуація: 1
[Рг С]→ 1;
[См]→ 1(ст розр=1)
[Рг С]→ 1;
[См]→ 1(ст.розр=0)
Фіксована кома
Результат нормалізувати
Результат в [См] і [Рг С]
Кінець.


5
K = 1 max Розр B
1
Pr C[ст]= 1
0
[См] = [См] + [Pr A]
0
Pr C[ст]= 1
1
[См] = [См] + [Pr B]
K = max Розр. B
0
[См]<- [Pr C]<- 1
0
Фіксована кома
1
Результат нормалізувати
Результат у [См]
Кінець

Навчальне видання

А.М.Пєтух **В.В.Войтко**

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

Частина 1

Навчальний посібник

Оригінал-макет підготовлено авторами

Редактор С.А. Малішевська

Підписано до друку 27.12.2004р.
Формат 29,7х42 1/4 Гарнітура Times New Roman
Друк різографічний Ум. друк. арк. 4.09
Тираж 75 прим.
Зам. № 2001-267

Віддруковано в комп'ютерному інформаційно-видавничому центрі
Вінницького державного технічного університету
21021, м.Вінниця, Хмельницьке шосе, 95, ВДТУ, ГНК, 9-й поверх
Тел. (0432) 44-01-59