

МІНІСТЕРСТВО ОСВІТИ Й НАУКИ УКРАЇНИ
Запорізький національний технічний університет

Методичні вказівки

до виконання контрольних завдань
з дисципліни «Прикладна теорія цифрових автоматів»
для студентів заочної форми навчання спеціальностей
«Комп'ютерні системи і мережі» і
«Спеціалізовані комп'ютерні системи»
кафедри «Комп'ютерні системи і мережі»

2010

Методичні вказівки до виконання контрольних завдань з дисципліни «Прикладна теорія цифрових автоматів» для студентів заочної форми навчання спеціальностей «Комп'ютерні системи і мережі» і «Спеціалізовані комп'ютерні системи» кафедри «Комп'ютерні системи і мережі» / Укладачі: Щербаков А.М., Проскурін М.П., Грушко С.С.– Запоріжжя: ЗНТУ, 2010.– 66 с.

Видання друге на укр. мові, виправлене та доповнене.

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

Рецензент: А.І. Вершина, доцент, к.т.н.

Відповідальний за випуск: С.С. Грушко, асистент

Затверджено
на засіданні кафедри
«Комп'ютерні системи і мережі»
Протокол № 3
від «9» грудня 2010 р.

ЗМІСТ

ВСТУП.....	5
ВИМОГИ ДО ОФОРМЛЕННЯ КОНТРОЛЬНОЇ РОБОТИ.....	6
1 СИСТЕМИ ЧИСЛЕННЯ КОМП'ЮТЕРНОЇ АРИФМЕТИКИ.....	7
1.1 Представлення чисел у позиційних системах числення	7
1.2 Вибір системи числення комп'ютера.....	9
2 МЕТОДИ ПЕРЕВЕДЕННЯ ЧИСЕЛ ІЗ ОДНІЄЇ СИСТЕМИ ЧИСЛЕННЯ ДО ІНШОЇ	11
2.1 МЕТОДИ ПЕРЕВЕДЕННЯ ЦІЛИХ ЧИСЕЛ.....	11
2.1.1 Переведення чисел методом підбору коефіцієнтів	11
2.1.2 Метод ділення числа на основу нової системи	12
2.1.3 Метод ділення числа на основу в будь-якому додатному степені	12
2.1.4 Метод віднімання найближчої меншої ваги степеня	13
2.2 ПЕРЕВЕДЕННЯ ДРОБОВОЇ ЧАСТИНИ ЧИСЛА В ІНШУ СИСТЕМУ ЧИСЛЕННЯ МЕТОДОМ МНОЖЕННЯ ЇЇ НА НОВУ ОСНОВУ	14
2.3 ПЕРЕВЕДЕННЯ НЕПРАВИЛЬНОГО ДРОБУ	16
2.4 ПЕРЕВЕДЕННЯ ЧИСЕЛ ІЗ ШІСТНАДЦЯТКОВОЇ І ВІСІМКОВОЇ СИСТЕМ У ДВІЙКОВУ	16
3 ФОРМАТИ ПРЕДСТАВЛЕННЯ ЧИСЕЛ У КОМП'ЮТЕРІ.....	18
3.1 Подання чисел у форматі з фіксованою комою.....	18
3.2 Подання чисел у форматі з рухомою комою.....	20
3.3 Похибки подання чисел	22
3.3.1 Абсолютна похибка подання чисел ΔN	22
3.3.2 Відносна похибка подання числа dN	23
4 ДВІЙКОВА АРИФМЕТИКА.....	24
4.1 ФОРМАЛЬНІ ПРАВИЛА ДВІЙКОВОЇ АРИФМЕТИКИ	24
4.2 Подання від'ємних чисел	28
4.2.1 Прямий код числа	29
4.2.2 Обернений код числа.....	29
4.2.3 Доповняльний код числа	31
5 ДОДАВАННЯ ДВІЙКОВИХ ЧИСЕЛ.....	33
5.1 Додавання чисел, поданих у формі з фіксованою комою, на двійковому суматорі прямого коду	33
5.2 Додавання чисел на двійковому суматорі оберненого коду.....	34

5.3 Додавання чисел на двійковому суматорі доповняльного коду	38
6 МОДИФІКОВАНІ БІНАРНІ КОДИ	42
6.1 Переповнення розрядної сітки	42
6.1.1 Переповнення при складанні прямих кодів	42
6.1.2 Переповнення при складанні доповняльних кодів	42
6.1.3 Переповнення при складанні в обернених кодах	42
6.2 Модифіковане додавання чисел у форматі із рухомою комою (крапкою)	44
7 СКЛАДАННЯ ЧИСЕЛ ПРИ РІЗНИХ ЗНАЧЕННЯХ ПОРЯДКІВ	47
8 МНОЖЕННЯ ДВІЙКОВИХ ЧИСЕЛ	50
8.1 Методи множення бінарних чисел	50
8.2 Множення чисел з фіксованою крапкою (комою) на ДСПК	51
8.3 Множення чисел з рухомою комою	53
9 МНОЖЕННЯ ЧИСЕЛ НА ДСДК	55
9.1 Множення чисел на ДСДК при позитивному множнику	55
9.2 Множення чисел на ДСДК при негативному множнику	55
10 ДІЛЕННЯ БІНАРНИХ ЧИСЕЛ	57
10.1 Метод ділення у бінарних чисел	57
10.2 Ділення чисел з фіксованою комою з відновленням залишку	58
11 ДІЛЕННЯ ЧИСЕЛ З ФІКСОВАНОЇ КОМОЮ БЕЗ ВІДНОВЛЕННЯ ЗАЛИШКУ	61
11.1 Алгоритм ділення у без відновлення залишку	61
11.2 Ділення чисел з рухомою комою	63
12 КОНТРОЛЬНЕ ЗАВДАННЯ	64
ЛІТЕРАТУРА	66

ВСТУП

Методичні вказівки до виконання контрольних завдань із курсу «Прикладна теорія цифрових автоматів» містять у короткій формі теоретичні основи комп'ютерної арифметики.

Розглянуті методи перетворення чисел з десяткової системи числення в різні форми представлення інформації в комп'ютері. Представлені алгоритми виконання основних арифметичних операцій алгебраїчного додавання, множення і ділення. Розглянуті формати представлення бінарних чисел з фіксованою і рухомою комою, прямі, обернені і доповняльні коди.

Контрольне завдання містить базові числа для проведення обчислень, завдання для обчислень і оцінки абсолютної і відносної погрішності

ВИМОГИ ДО ОФОРМЛЕННЯ КОНТРОЛЬНОЇ РОБОТИ

Пояснювальна записка до контрольної роботи повинна бути виконана відповідно до ДСТУ 3008-95, і включати:

- титульний аркуш;
- вступ;
- завдання;
- основні теоретичні визначення, застосовувані в розрахунках правила, алгоритми;
- опис послідовності обчислень відповідно до алгоритмів виконання арифметичних операцій варіанта завдання;
- виводи;
- перелік використаної літератури.

Контрольна робота в надрукованому виді на аркушах формату А4 надається викладачеві курсу ПТЦА кафедри КСМ не пізніше, ніж за 15 днів до початку строку модульного контролю або залікової сесії. Для студентів дистанційного навчання допускається електронне пересилання.

1 СИСТЕМИ ЧИСЛЕННЯ КОМП'ЮТЕРНОЇ АРИФМЕТИКИ

1.1 Представлення чисел у позиційних системах числення

Сам процес числення (нумерація) - сукупність певних прийомів (правил, алгоритмів) представлення натуральних чисел і виконання арифметичних операцій.

У будь-якій системі числення прийняті деякі символи (знаки, слова) для позначення певних чисел, ці знаки називають базовими (вузловими). Приклад, у десятковій системі базовими є знаки 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 натуральні числа, що позначають цілі, від 0 до 9 і дробові за допомогою коми (крапки).

Інші (алгоритмічні) числа утворюються у результаті виконання дій за певними правилами над базовими числами.

Системи числення відрізняються одна від одної вибором базових (вузлових) чисел і способами утвору алгоритмічних.

Системи числення, у яких алгоритмічні числа утворюються додаванням вузлових, називаються аддитивними.

У числах десяткової, двійкової, вісімкової і ін. системах застосований аддитивно-мультиплікативний спосіб.

Системи числення ділять на позиційні і непозиційні.

Система числення, у якій значення базисної цифри визначається її положенням у розрядах числа, називається позиційною.

Приклад, у десятковій системі числення в числі 222 перша цифра праворуч означає дві одиниці, друга (ліворуч) - два десятки, третя (праворуч) - дві сотні.

Позиційна система числення визначається своєю основою.

Основа q позиційної системи числення визначається кількістю знаків, використаних для відображення числа в даній системі.

Непозиційна система числення - система, у якій значення базових знаків не залежить від їх місця розташування в числі, розрядності. Наприклад, римську систему числення називають непозиційною системою, хоча вона і містить деякі елементи позиційної. Так, у числах IX (9) і XI (11) знак I = 1 приймає два значення ± 1 і залежить від свого місця розташування відносно X.

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

Число в позиційній системі числення визначається рівністю:

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

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

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

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

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

Таблиця 1.1 – Еквіваленти 10-х чисел в системах числення

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

Приклад. Число $934,125_{(10)}$ записане в десятковій системі числення. Число $124,57_{(8)}$ записане у вісімковій системі числення (тут беруться базові цифри 0, 1, ..., 7).

У двійковій системі числення використовують цифри 0,1. Приклад. Число $1001,1 = 1*2^3 + 1*2^0 + 1*2^{-1} = 17,5$ (див. таблицю 1.1).

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

розрядів зростає справа наліво. Якщо беремо розряд $P_i = 10^k$, то наступний старший буде мати вагу $P_{i+1} = 10^{k+1}$, а молодший $P_{i-1} = 10^{k-1}$. Такий зв'язок розрядів припускає (при виконанні операцій додавання, віднімання) **передачу інформації** між ними. Якщо в даному розряді нагромадилося значення одиниць, що дорівнює або більше основи q (переповнення), то повинна відбуватися передача одиниці в сусідній старший розряд. Це має назву переніс в старший розряд (при складанні) і позика із старшого розряду при відніманні).

Приклад. $9 + 2 = 11 = (9 + 1) + 1 = 10* + 1 = 11$.

Додавши $(1 + 9)$, отримали **0** в розряді одиниць (* означає переповнення розряду одиниць) і формується переніс **1** в розряд десятків, далі проводиться операція додавання **0** + 1 в розряді одиниць.

1.2 Вибір системи числення комп'ютера

При розробці комп'ютера проводиться вибір системи числення, методів виконання арифметичних і логічних операцій, елементної бази і ін.

Вибір системи числення обумовлен наступними причинами:

- основа системи визначає число знаків, що представлені в одному розряді. Тут перевагу має **двійкова система**, а не десяткова тому що вона вимагає всього два знаки (0,1), або два рівні.

- система числення повинна забезпечити точність арифметики, великий діапазон і простоту представлення чисел. Сигнал передачі біта інформації і майже всі радіоелементи мають двійкову природу включено/виключено (напруга, струм, світло є/немає: «1»/ «0») та легко відображають числа у двійковій системі.

Розглянемо приклад поразрядного додавання двійкових чисел:

$$0 + 0 = 0; \quad 1 + 0 = 1; \quad 0 + 1 = 1; \quad 1 + 1 = 10.$$

Додавання цілих чисел без знака можна представити за схемою рисунка 1.1.

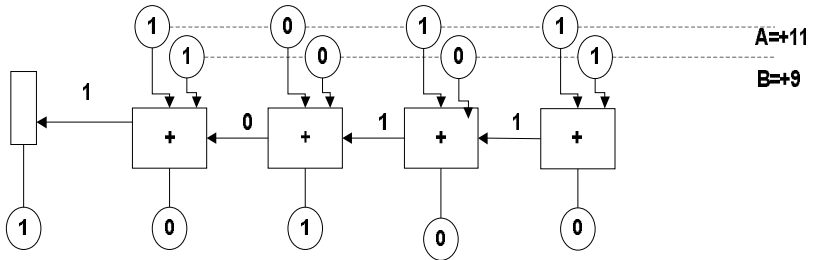


Рисунок 1.1 – Структурна схема сумматора

2 МЕТОДИ ПЕРЕВЕДЕННЯ ЧИСЕЛ ІЗ ОДНІЄЇ СИСТЕМИ ЧИСЛЕННЯ ДО ІНШОЇ

Раніше ми відзначали, що будь-яке число можна зобразити поліномом з основою q_1 . Але це ж число можна записати іншим поліномом з основою q_2 , інакше: $N(q_1) = N(q_2)$. Представимо це як:

$$\begin{aligned} N(q_1) &= a_n q_1^n + a_{n-1} q_1^{n-1} + \dots + a_1 q_1^1 + a_0 q_1^0 + a_{-1} q_1^{-1} + \dots + a_{-m} q_1^{-m} = \\ &= b_k q_2^k + b_{k-1} q_2^{k-1} + \dots + b_1 q_2^1 + b_0 q_2^0 + a_{-1} q_2^{-1} + \dots + a_{-s} q_2^{-s} = N(q_2). \end{aligned}$$

Таким чином, задача переведення числа з основою q_1 у число з основою q_2 зводиться до пошуку коефіцієнтів b_i поліному нової основи.

2.1 Методи переведення цілих чисел

Розглянемо основні методи переведення чисел з однієї системи числення до іншої. Основну увагу будемо приділяти методам переведення чисел із десяткової системи числення до двійкової. Методи переведення цілої і дробової частини числа різні. Тому, для початку, розглянемо методи переведення цілої частини числа.

2.1.1 Переведення чисел методом підбору коефіцієнтів

Цю задачу можна вирішити методом підбору коефіцієнтів нового поліному.

Правило послідовності переведення десяткових чисел наступне. На початку беремо число нового поліному зі степенем основи 2^n таке, щоб воно не перевищувало задане десяткове, але було близьким і меншим за нього. У старшому розряді нового двійкового числа ставимо 1. Потім підбираємо коефіцієнти менших степенів поліному так, щоб сума значень ваг розрядів де стоять 1, дорівнювала десятковому числу. Усі дії повинні виконуватися за правилами q_1 -арифметики, тобто за правилами десяткової системи числення.

Наприклад. Перевести десяткове число 96 у двійкову систему.

Рішення.

$$96 = 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 1100000_{(2)}.$$

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

$$96 = 1 \cdot 64 + 1 \cdot 32 + 0 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 0 \cdot 1.$$

Метод за цим правилом можна використовувати для будь-яких позиційних систем числення, підставивши в розрядах базисні цифри нової системи числення.

Наприклад. Перевести десяткове число 96 у трійкову систему.

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

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

$$96 = 0 \cdot 243 + 1 \cdot 81 + 0 \cdot 27 + 1 \cdot 9 + 2 \cdot 3 + 0 \cdot 1.$$

Машинні алгоритми переведення частіше використовують метод ділення числа на основу нової системи числення.

2.1.2 Метод ділення числа на основу нової системи

Поліном цілої частини числа можна записати за схемою Горнера:

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

використовуючи для цього круглі дужки.

Якщо праву частину розділити на нову основу – q_2 , то одержимо цілу частину і залишок b_0 . Повторюючи ділення $k+1$ раз, одержимо інші цілі частини і залишки $b_1, b_2, \dots, b_k$. Останній залишок (неподільна ціла частина) є старшим розрядом числа, поданого залишками нової базової основи – q_2 .

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

Рішення. Ділимо число 12 на $q = 2$:

$$\begin{array}{r} \underline{12 \mid 2} \\ \underline{12 \quad 6 \mid 2} \\ 0 \quad \underline{6 \quad 3 \mid 2} \\ 0 \quad 2 \quad \underline{1 \text{ Ст. роз.}} \\ 1 \end{array}$$

Частка 1 менше основи 2, і є старшим залишком. Відповідь: $12 = 1100$.

2.1.3 Метод ділення числа на основу в будь-якому додатному степені

Попередній метод має один недолік. При великих числах операція ділення має багато ітерацій. Це знижує швидкодію. Метод

ділення на основу в будь-якому позитивному степені аналогічний попередньому. Тут беруть для ділення число рівне q^n , близьке до заданого числа, але що не перевищує його. Виконують ділення числа на вибрану основу q^n . Після цього кожен залишок від ділення записують у розрядах двійкової системи, число яких дорівнює обраному степеню n .

Наприклад. Перевести в двійкову систему числення десяткове число 523.

Рішення. Вибираємо найближче до заданого число $2^9=512$ і ділимо число на проміжну основу – 512:

$$\begin{array}{r} \underline{523} \overline{)512} \\ \underline{512} \\ 11 \end{array}$$

Одержали два залишки: $523:512=1(11)$. Старший – 1, молодший – 11. Кожен із залишків розписуємо в дев'ятьох двійкових розрядах: 000000001 000001011. Потім з'єднуємо запис (старші нулі в розрядах залишків можна не записувати) і одержуємо число 1000001011. Цей метод потребує лише одну операцію ділення, а запис залишків у двійковій системі числення можна виконувати з таблиць, які зберігаються в пам'яті.

2.1.4 Метод віднімання найближчої меншої ваги степеня

Метод полягає в наступному. Вибирають значення найближчої меншої розрядної ваги двійкового числа. Віднімаючи його від заданого числа, одержують залишок. У розряді обраної ваги, ставиться 1. Потім порівнюють залишок із новим меншим ваговим розрядом. Якщо залишок менший, то в цьому ваговому розряді ставиться 0, якщо залишок більший, то в цьому розряді ставиться 1, а з залишку віднімається вага цього розряду. Виходить новий залишок, що знову порівнюється з наступною меншою вагою. Так продовжується до останнього (молодшого) вагового двійкового розряду. Отримане двійкове число і буде шуканим.

Приклад. Перевести десяткове число 1125 у двійкове. Метод – віднімання найближчої меншої розрядної ваги. Найближчою меншою розрядною вагою буде число $1024=2^{10}$.

Ставимо в десятому розряді 1. Віднімаємо $1125-1024=101$, де 101 залишок, що порівнюємо з числом $2^9=512$. Залишок 101 менший ніж 512, значить на місці 2^9 ставимо 0. Порівнюємо наступну розрядну вагу $2^8=256>101$. Знову на місці розрядної ваги 2^8 ставимо 0. Аналогічно буде і для розряду 2^7 – ставимо 0. Порівнюємо з розрядною вагою $2^6=64<101$. У цьому розряді ставимо 1 і, віднімаючи $101-64=37$ одержуємо новий залишок, що порівнюємо з розрядною вагою $2^5=32<37$. Отже, в цьому розряді ставимо 1 і, віднімаючи $37-32=5$, отримуємо новий залишок, що легко розписати в чотирьох розрядах, що залишилися, 0101. Таким чином, одержуємо число $10001100101_{(2)}=1125_{(10)}$. Перевіримо: $1024+64+32+4+1=1125$. Цей метод виключає громіздку операцію ділення, і вже при невеликому досвіді, легко виконується користувачем. Метод придатний для переводу як цілої, так і дробової частини числа.

2.2 Переведення дробової частини числа в іншу систему числення методом множення її на нову основу

Переведення дробової частини числа системи q_1 в нову систему числення q_2 виконується окремо від цілої частини, шляхом множення дробу системи q_1 на нову основу q_2 .

Це можна показати, використовуючи форму загального запису дробу. Дробову частину числа можна записати в новій системі:

$$N(q_2) = b_{-1}q_2^{-1} + b_{-2}q_2^{-2} + \dots + b_{-k}q_2^{-k} + \dots, \quad (2.1)$$

цей вираз можна переписати за схемою Горнера [7]:

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

Якщо праву частину послідовно множити на q_2 , то будемо знаходити новий неправильний дріб, у цілій частині якого будуть залишки $b_{-1}, b_{-2}, \dots, b_{-k}$, при цьому всі дії повинні виконуватися за правилами q_1 -арифметики, і отже, в цілій частині дробу, що виходить поза розрядність, будуть з'являтися еквіваленти чисел нової системи числення q_2 , починаючи зі старшого розряду.

Правило. Для переводу правильного дробу (без цілої частини) необхідно, діючи в арифметиці системи числення q_1 , помножити дробову частину на основу нової системи q_2 , відокремити у результаті цілу частину, а дробову частину, що залишилася, знову

помножити на цю основу і так до отримання потрібного числа цифр для необхідної точності. Результат записується як 0,... і дробова частина в порядку одержання переповнень, починаючи зі старшого розряду.

Приклад. Перевести десятковий дріб 0,625 у двійкову систему ($q_2=2$).

Рішення:

$$\begin{array}{r}
 0,625 \\
 \hline
 1 \ 250 \quad \text{Ст. роз.} \\
 \hline
 0 \ 500 \\
 \hline
 1 \ 000 \\
 \hline
 0 \ 000
 \end{array}$$

Відповідь: $N=0,1010$. Перевірити можна за формулою (2.1).

Приклад. Перевести двійковий дріб 0,1101 у десятковий ($q_2=1010$).

$$\begin{array}{r}
 0,1101 \\
 \hline
 *1010 \\
 \hline
 1000 \mid 0010 \quad b_1 = 8 \\
 * \quad 1010 \\
 \hline
 0001 \mid 0100 \quad b_2 = 1 \\
 * \quad 1010 \\
 \hline
 0010 \mid 1000 \quad b_3 = 2 \\
 * \quad 1010 \\
 \hline
 0101 \mid 0000 \quad b_4 = 5
 \end{array}$$

$b_1=8, b_2=1, b_3=2, b_4=5$. Відповідь: $N=0,8125$.

При переведенні правильних дробів із однієї системи числення в іншу може вийти дріб у виді нескінченного або розбіжного ряду. Тому процес переведення необхідно закінчувати:

- з появою в дробовій частині в усіх розрядах нулів;

- якщо буде досягнута задана точність дробу, тобто одержано необхідне число розрядів після коми.

2.3 Переведення неправильного дробу

Правило. Для переведення неправильного дробу (дробу з цілою частиною) з однієї системи числення в іншу необхідно окремо здійснити переведення його цілої і дробової частин, а результати записати послідовно, відокремивши цілу частину від дробової комою.

2.4 Переведення чисел із шістнадцяткової і вісімкової систем у двійкову

При переведенні чисел із десяткової системи у двійкову, часто використовують проміжну вісімкову систему. Це значно зменшує число операцій.

Приклад. Перевести десяткове число 123 у двійкове через проміжну вісімкову систему. Рішення подане у табл. 2.1.

У табл. 2.1 приведені залишки при переведенні через основу 2. Щоб перейти від залишків вісімкової проміжної системи до двійкової, треба кожний залишок записати у тріаду двійковим кодом.

Таблиця 2.1 – Перевага проміжних систем переводу чисел

Проміжна система $q=8$			Двійкова система $q=2$	
Число	Залишок	Тріада	Число	Залишок
123:8	3	1	123:2	1
		1	61:2	1
		0	30:2	0
15:8	7	1	15:2	1
		1	7:2	1
		1	3:2	1
1	1	1	1	1 ст.р.
		0 0 ст.р		
Ітерацій - 2			Ітерацій - 6	

Щоб перевести число з 16-вої і 8-вої систем у двійкову, необхідно кожну цифру числа, яке переводиться, подати відповідно

чотирма (або трьома) двійковими розрядами у двійковому коді (тетрадами або тріадами), розташували їх на місцях (розрядах) цих цифр. Нулі в старших і молодших розрядах, що не змінюють значення числа, можна опускати.

Приклад:

$$1,171_{(8)} = 001,001\ 111\ 001_{(2)};$$

$$753,335_{(8)} = 111\ 101\ 011,011\ 011\ 101_{(2)};$$

$$1FA, C_{(16)} = 0001\ 1111\ 1010, 1100_{(2)}.$$

Для переведення чисел із двійкової системи числення в шістнадцяткову (вісімкову) треба двійкові цифри числа, яке перекладається, згрупувати по чотири (три) в обидва боки від коми (при необхідності неповні групи доповнити нулями). Кожну одержану групу двійкових цифр замінити відповідною їй цифрою в новій системі числення. Нові цифри розташувати на місцях заміненних кодів.

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

$$0001\ 1111\ 1010, 1100\ 0010\ 0100_{(2)}$$

$$1\ F\ A, \ C\ 2\ 4_{(16)}$$

$$\text{Відповідь: } 1FA, C24_{(16)}$$

$$111\ 111\ 010, 110\ 000\ 100\ 100_{(2)}$$

$$7\ 7\ 2, \ 6\ 0\ 4\ 4_{(16)}$$

$$\text{Відповідь: } 772, 6044_{(8)}$$

Узагальнюючи правила переведення чисел, можна зробити висновок, що в якості проміжних систем числення доцільно використовувати системи з основою $q=2^n$, де $n=1, 2, 3, 4, 5, 6, \dots$ Це спрощує переведення чисел у двійкову систему і навпаки.

З ФОРМАТИ ПРЕДСТАВЛЕННЯ ЧИСЕЛ У КОМП'ЮТЕРІ

У десятковій системі існує багато форм подання чисел. Ми, користуючись цим, самі вибираємо ту або іншу форму. Наприклад, число 0,25 можна відобразити як $1/4$, при виконанні операцій з такими ж дробовими числами. Можна відобразити як $25 \cdot 10^{-2}$ або $2500 \cdot 10^{-4}$, або $0,0025 \cdot 10^2$.

Усю різноманітність запису чисел поділяють на природну і нормалізовану (нормальну) форми.

При природній формі число записують у природному (натуральному) виді, наприклад 125 – ціле число, 0,125 – правильний дріб, 125,125 – неправильний дріб. При нормалізованій формі запис числа може приймати різний вигляд і його форма залежить від прийнятих діючих правил (обмежень). Наприклад, 12500 може бути записано тільки як $12,5 \cdot 10^3$ або $0,125 \cdot 10^5$ і т.д.

Така велика кількість форм подання чисел може стати причиною ускладнення пристроїв і алгоритмів функціонування цифрових автоматів.

Щоб уникнути цього, у цифрових автоматах прийнято свої нормалізовані форми запису і відображення чисел, що дозволяє мінімізувати вартість апаратних і алгоритмічних засобів.

Найбільше поширення одержали формати з фіксованою та з рухомою комою (крапкою) [17].

3.1 Подання чисел у форматі з фіксованою комою

Зображення комп'ютерного числа – це подання числа N у вигляді n -розрядного коду у наперед заданому форматі за встановленими правилами.

При поданні числа *у форматі з фіксованою комою* весь формат n -розрядної сітки заздалегідь розбивається на три частини:

- два лівих розряди призначені для подання знаку числа:

- 1) 00 – додатне число;

- 2) 11 – від'ємне число;

- наступні k розрядів використовуються для розміщення цілої частини числа;

- наступні $n-(k+2)$ розрядів призначені для розміщення дробової частини числа.

На рис. 3.1 наведений формат числа з фіксованою комою.



Рисунок 3.1 – Формат числа з фіксованою комою

Дійсно, положення коми строго фіксовано в розрядній сітці автомата. Позначимо машинне зображення числа N через N_m , тоді

$$N_m = \frac{N}{K_\phi},$$

де K_ϕ – коефіцієнт формату, величина якого залежить від числа розрядів цілої частини. Для рисунку 5.1 $K_\phi = 2^k = 2^{10}$. Основу двійкової системи 2 за звичаєм не пишуть. Тоді $K_\phi = 10$, тобто, коефіцієнт формату вказує кількість розрядів цілої частини числа, починаючи від розрядів знаку, після яких потрібно ставити кому.

Приклад. Записати число $N = +11,00111000111$ у форматі 16 розрядної машини з $K_\phi = 6$.

$$N_m = N \cdot 2^6.$$

$N_m = 00.00001100111000$ (останні розряди 111 – пропадають за округленням). Тоді число N визначається як $N = N_m \cdot K_\phi = 11,00111000$. Знак «+» для додатного числа не записується.

Діапазон поданих чисел R залежить від обраного формату цілої частини і складає:

$$R = \pm 2^n - 1,$$

де n – кількість розрядів числа і один розряд відведений для знака.

При запису чисел у форматі з фіксованою крапкою, якщо число виходить за межі розрядної сітки числа, молодші розряди відкидаються. При цьому, можуть виникати похибки, зокрема, ділення на 0, що приводить до невизначеності і необхідності втручати у розрахунки. Однак, при роботі ПК у форматі з фіксованою крапкою швидкість виконання арифметичних і логічних операцій зростає.

Похибка подання чисел зменшується при правильному виборі масштабних коефіцієнтів.

У деяких автоматах при переповненні розрядної сітки, передбачається сигнал переповнення із зупинкою розрахунку, що порушує нормальне функціонування.

3.2 Подання чисел у форматі з рухомою комою

Іншою найбільш розповсюдженою нормалізованою формою є подання чисел у форматі з рухомою комою. У цьому випадку в нормальній формі число записується як:

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

N – машинне нормалізоване число; m – мантиса числа; p – характеристика числа, яка визначає його порядок.

Таке подання у загальному випадку не є однозначним, тому що не оговорені вимоги до форми мантиси та порядку.

Для нормалізованого числа, як правило, вводять обмеження.

Правило. Для нормалізованої форми подання чисел у форматі з рухомою комою, діє умова, яка передбачає наявність одиниці у старшому розряді мантиси.

У цьому випадку абсолютне значення мантиси може бути в межах від $0,1\dots$ до $0,1111\dots 111q^{-n}$, тобто, від $0,5$ до 1 при $n \rightarrow \infty$, де n кількість розрядів для запису мантиси без знаку.

Формат машинного відображення чисел з рухомою комою, для 16-ти розрядної сітки представлений на рис. 3.2.

Положення коми не є постійним. Тому і формат називається з рухомою комою. Число, подане мантисою числа, визначається знаком порядку і порядком числа, тому воно може змінюватися залежно від знаку та значення порядку. Але завжди формат числа повинний містити знак мантиси, мантису, знак порядку, порядок.

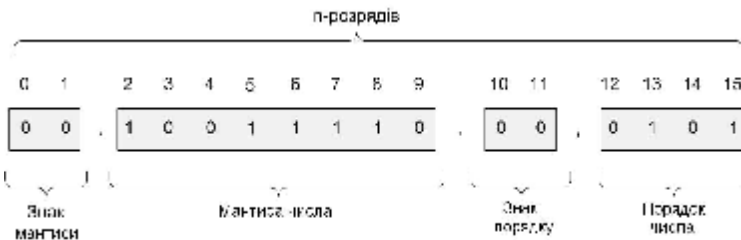


Рисунок 3.2 - Формат числа з рухомою комою

Порядок числа визначає скільки розрядів мантиси необхідно відрахувати щоб поставити кому, що відокремлює дробову частину від цілої. Отже, перш ніж записати число в цифровий автомат із заданою розрядною сіткою, це число нормалізують, тобто приводять до виду, коли в старшому розряді мантиси немає нуля, а стоїть 1. Всі інші форми називають **ненормалізованими (денормалізованими)**.

Порядок – це показник степеня основи системи числення, на який потрібно помножити або розділити (для від’ємного знаку порядку) мантису для визначення цілої частини числа, тобто:

$$P_m = q^{\pm P},$$

де P_m – характеристика порядку мантиси, p - ціле число, яке визначає порядок. Його записують у двійкових розрядах зі своїм знаком.

При нормалізації число зсувають вправо (якщо є ціла частина числа) або вліво (до одиниці після коми). При зсуві вправо, порядок (тобто степінь основи) збільшується з кожним зсувом на 1. При зсуві вліво порядок зменшується щораз на 1.

Приклад. Записати двійкове число в нормалізованій формі формату із рухомою комою:

$$11101,011 = 00.11101011 \cdot 2^5 = 00.11101011.00.0101.$$

Для прийнятого формату рис. 3.2 це число запишеться як

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	1	1	0	1	0	1	1	0	0	0	1	0	1

При запису мантиси зручно, щоб старший розряд мантиси і порядку був ліворуч. Іноді розташування старшого розряду визначають окремо. Приклад запису числа із від’ємним знаком порядку.

$$N = 00.00011111; N_m = 00.11111 \cdot 2^{-3} = 00.11111000.11.0011$$

Оскільки основа $q = 2_{(10)} = 010_{(2)}$ завжди постійна, то її запис у характеристиці числа опускається.

Діапазон поданих чисел визначається розрядною сіткою автомата.

Максимальне число буде при позитивних знаках мантиси і порядку і при максимальних значеннях їх чисел, тобто, коли у всіх розрядах будуть одиниці.

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

$$N_{\max}=(00.111\dots1)\cdot 2^{111\dots1}, \quad N_{\min}=(11.111\dots1)\cdot 2^{-111\dots1}$$

При використанні формату з рухомою комою, швидкість обчислень падає, але точність і діапазон подання чисел зростає.

3.3 Похибки подання чисел

Цифровий автомат, як ми бачимо, завжди спричиняє появу похибок в розрахунках, величина яких залежить від обмежень, що накладаються на автомат за розрядною сіткою, форм і подання чисел.

3.3.1 Абсолютна похибка подання чисел ΔN

Абсолютною похибкою подання чисел ΔN називається різниця між істинним значенням числа N і його значенням N_m , що отримано після машинного відображення операцій АЛП:

$$\Delta N = N - N_m.$$

Наприклад: задане двійкове число $+11,00111000111$ при $n=16$ і $K_\phi=2^k$, $k=6$, тоді $N_m=0000001100111000$ (останні три розряди 111 – втрачені).

Перевівши в десяткові числа, одержимо:

$$N=11,00111000111_{(2)}=3,222168_{(10)},$$

$$N_m=00.000011,00111000_{(2)}=3,21875_{(10)},$$

$$\Delta N=N-N_m=3,222168-3,21875=0.003418,$$

де n – розрядність автомата;

k – степінь масштабного коефіцієнту формату K_ϕ ;

N_m – машинне число після виконання арифметичних операцій у двійковій системі.

Правило. Максимальна похибка $\Delta N_{\max}$ при переведенні чисел із десяткової системи у двійкову для формату з фіксованої комою, не перевищує одиниці молодшого розряду сітки:

$$\Delta N_{\max} \leq 2^{-n}.$$

3.3.2 Відносна похибка подання числа dN

Відносна похибка подання числа dN – це відношення абсолютної похибки ΔN до числа N у відсотковому відношенні, тобто:

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

Звичайно N визначається математично після виконання аналогічних арифметичних операцій у десятковій системі.

Після визначення абсолютної похибки визначаємо відносну похибку у відсотках.

Наприклад, із попереднього рішення $\Delta N = 0.0014647$.

Визначимо відносну похибку

$$\delta N = \frac{0,003418}{3,222168} \cdot 100\% = 0,106\% .$$

Відносна похибка складає 0,106 %.

4 ДВІЙКОВА АРИФМЕТИКА

4.1 Формальні правила двійкової арифметики

В арифметиці будь-якого виду завжди беруть участь два або більше чисел. Як результат виконання арифметичних операцій з'являється нове число C . Формально це можна представити як:

$$C = A \otimes B \otimes B_1 \dots \otimes B_n, \quad (4.1)$$

де $\otimes$ – знак будь-якої з чотирьох арифметичних дій: додавання, віднімання, множення, ділення.

Кожне з чисел $A, B, B_1, \dots, B_n$ є операндом, тобто числом, яке бере участь у арифметичних операціях цифрового автомата. Необхідно звернути увагу на те, що у виразі (3.1) не вказано індекс позначення машинного числа. Проте, всі операнди і результат C є машинними числами і подаються в одному із форматів автомата.

Розглянемо арифметичні операції між операндами A і B , поданими на рівні розрядів, та їх результат C .

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

де n – розрядна сітка автомата.

Алгоритми виконання арифметичних операцій у системі числення з основою q аналогічні алгоритмам десяткової системі числення.

Результат виконання операції $\otimes$ над цифрами a_i і b_i у i -му розряді виражається двома цифрами: c_i – результатом відповідної операції в даному розряді і цифрою P_i – перенесенням у старший розряд. Результати порозрядного виконання операцій c_i і перенесення P_i формуються за наступними правилами.

Додавання:

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

$$P = \begin{cases} 0, & \text{при } a_i + b_i < q; \\ 1, & \text{при } a_i + b_i \geq q. \end{cases}$$

Віднімання

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

$$q = \begin{cases} 0, & \text{при } a_i \geq b_i; \\ 1, & \text{при } a_i < b_i. \end{cases}$$

Множення:

$$c_i = \begin{cases} a_i \cdot b_i, & \text{при } a_i \cdot b_i < q; \\ a_i \cdot b_i - \left[\frac{a_i \cdot b_i}{q} \right] \cdot q, & \text{при } a_i \cdot b_i \geq q. \end{cases}$$

Виконання операції ділення не може бути зведене до операції над окремими цифрами числа. Тому, ділення виконується за аналогією з десятиковою системою і буде розглянуто далі.

Двійковий напівсуматор – пристрій, що виконує арифметичну дію додавання без врахування переносів із попереднього розряду, тобто, на його вхід подаються тільки числа розряду a_i і b_i (рис. 4.1 а).

Проте, не враховувати переноси при підсумовуванні багаторозрядних чисел не можна.

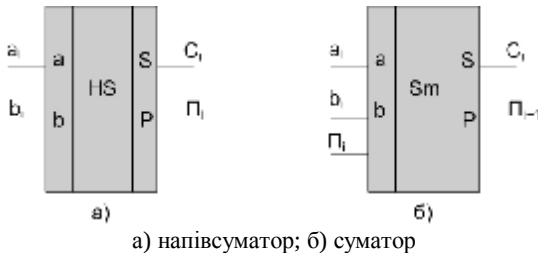


Рисунок 4.1 – Двійковий напівсуматор та суматор

Тому, при додаванні чисел використовується двійковий суматор – пристрій, що виконує арифметичну дію додавання з урахуванням

переносів від попереднього розряду і передаванні переносів у наступний. Його умовне графічне наведено на рис. 3.1 б.

Правила додавання для напівсуматора можна визначити табл. 4.1.

Таблиця 4.1 – Напівсуматор

a	b_i	c_i	Π_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Функціонування цифрового автомата – двійкового суматора визначається табл. 4.2.

Таблиця 4.2 – Робота суматора

a_i	b_i	Π_i	c_i	Π_{i+1}
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Тут виконується порозрядне додавання за формулою:

$$a_i + b_i + \check{I}_i = C_i + \check{I}_{i+1},$$

де a_i, b_i – розряди чисел A, B ;

c_i – сума i -го розряду;

Π_i – перенос із попереднього молодшого розряду;

Π_{i+1} – перенос до наступного старшого розряду.

При виконанні арифметичної операції віднімання у двійковій системі числення беруть позику одиниці зі старшого розряду, аналогічно як і в десятковій.

Крім того, позику необхідно враховувати при наступному відніманні цифр зі старшого розряду. З урахуванням цього і

пам'ятаючи, що від'ємник завжди повинен бути менше зменшуваного, можна визначити правила у вигляді табл. 4.3.

Таблиця 4.3 – Віднімання чисел

a_i	b_i	Z_i	c_i	Z_{i+1}
0	0	0	0	0
1	0	0	1	0
1	1	0	0	0
0	1	0	1	-1
0	0	-1	1	-1
1	0	-1	0	0
1	1	-1	1	-1
0	1	-1	0	-1

Формулу порозрядного віднімання можна записати:

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

де a_i, b_i – операнди розряду;

c_i – різниця розряду;

Z_i – позика із молодшого i -го розряду;

Z_{i+1} – позика із старшого розряду.

Розглянемо приклади арифметичних дій двійкової арифметики для чисел із фіксованою комою.

Будемо оперувати із шестизначними додатними числами за модулем меншим одиниці.

Приклади:

- виконати операцію додавання двійкових чисел:

$$\begin{array}{r} \text{а) } 0,010101 \\ + 0,100100 \\ \hline 0,111001 \end{array} \quad \begin{array}{r} \text{б) } 0,010101 \\ + 0,001011 \\ \hline 0,100000 \end{array};$$

- виконати операцію віднімання двійкових чисел:

$$\begin{array}{r} 0,111110 \\ - 0,011111 \\ \hline 0,011111 \end{array}$$

а) віднімання у 8-ій системі; б) віднімання у 3-ій системі:

$$\begin{array}{r} \text{а) } \underline{0,757473} \\ 0,273654 \\ \hline 0,463617 \end{array} \quad \begin{array}{r} \text{б) } \underline{0,222222} \\ 0,102210 \\ \hline 0,120012 \end{array} ;$$

- виконати ділення двійкових чисел:

$$\begin{array}{r} \underline{1101,001} \overline{) 1010} \\ \underline{1010} \\ 001100 \\ \underline{1010} \\ 001010 \\ \underline{1010} \\ 0000 \end{array}$$

- виконати операцію множення:

$$\begin{array}{r} 0,100101 \\ * 0,111110 \\ \hline 0,000000 \\ + 0,100101 \\ 0,100101 \\ 0,100101 \\ 0,100101 \\ 0,100101 \\ \hline 0,100011110110 \end{array}$$

Наведене вище виконання арифметичних дій ускладнює структуру автомата. Реалізацію цих операцій в комп'ютерах можна значно спростити, якщо вирішити проблему заміни подання від'ємних чисел додатними.

4.2 Подання від'ємних чисел

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

$$A - B = A + (-B_{\text{дв}}).$$

Заміна операції віднімання на додавання ставить проблему подання від'ємного числа в цифровому форматі.

Це питання вирішується використанням спеціальних кодів для зображення чисел. За допомогою цих кодів операція віднімання замінюється арифметичним додаванням. Для машинного подання двійкових чисел використовують прямий, обернений і доповняльний коди.

4.2.1 Прямий код числа

Від'ємне двійкове число з комою, фіксованою перед старшим розрядом виглядає як:

$$N = -0, a_1 a_2 \dots a_n ,$$

тоді його машинне зображення:

$$N_M = 1.a_1 a_2 \dots a_n .$$

Отже, прямий код від'ємного числа є код, усі цифрові розряди якого залишаються незмінними, записаними в нормалізованій формі, а в знаковій частині числа записується 1 (або 11 для модифікованого коду).

Наприклад: $N = -0,101110$, тоді $N_M = 1.101110$.

Додатне число в прямому коді не міняє свого зображення.

Наприклад: $N = 0,110101$, тоді $N_M = 0.110101$.

У прямому коді в числову сітку автомата можна записати:

$$N_{Mmax} = 0.111\dots 1 = 2^n - 1 ,$$

де n – кількість розрядів, а діапазон поданих чисел для прямого коду лежить у межах:

$$-(2^n - 1) \leq N_M \leq (2^n - 1). \quad (4.2)$$

Іншим методом подання від'ємних чисел є запис їх у оберненому або доповняльному коді.

4.2.2 Обернений код числа

Оберненим кодом двійкового числа $N_i = 11.a_1 a_2 \dots a_n$ називається таке машинне зображення числа, для якого $a_i = 0$, якщо воно дорівнювало 1 і навпаки, $a_i = 1$, якщо воно дорівнювало 0. Знакові розряди приймають значення 1 або значення 11 для модифікованого знаку.

Інакше, оберненим кодом двійкового числа є інверсне зображення цього числа, тобто, всі розряди вихідного двійкового числа, окрім знакових, приймають обернене значення.

Наприклад. $N = -0,101110$, тоді $N_{об.} = 1.010001$.

Узагальнюючи правила утворення оберненого коду для всіх позиційних систем числення, можна дати наступне визначення. Обернений код від'ємного числа отримаємо при відніманні від значення $(q-1)$ цифри за кожним розрядом числа за винятком знакових, у яких залишається значення $(q-1)$, тобто,

$$N_{іоб.} = (q-1).a_1 a_2 \dots a_n,$$

де $\bar{a}_i = (q-1) - a_i$.

Наприклад. Записати число у оберненому коді.

$$-286357_{(10)} = 9.713642_{(10)об.}$$

$$-0,1010111101_{(2)} = 1.0101000010_{(2)об.}$$

Особливо слід звернути увагу на наступне.

Якщо при перетворенні в знаковому розряді машинного зображення числа, що подане прямим кодом, перебуває $(q-1)$, то всі цифри числа, крім знакового, замінюються відніманням з $(q-1)$ значення розряду; якщо в знаковому розряді перебуває нуль, то перетворення не відбувається.

Наприклад:

Прямий код	Обернений код
1.0111000111	1.1000111000
0.1010110101	0.1010110101
00.110000111	00.110000111
11.110100000	11.001011111
9.243476	9.75623
0.943890	0.943890

Перехід від оберненого коду до прямого виконується за аналогічним правилом: від значення $(q-1)$ віднімається значення кожного розряду, крім знакових. Для двійкової системи числення можна перейти до прямого коду простим інвертуванням розрядів оберненого коду, крім розрядів знаків.

Приклад:

$$A_{об.} = 11.1100101$$

$$A_{пр.} = 11.0011010$$

4.2.3 Доповняльний код числа

Доповняльний код числа $N = -0, a_1 a_2 \dots a_n$ – це таке машинне подання, у якому число записується оберненим кодом $N_{\text{іа.}} = (q-1) \cdot \bar{a}_1 \bar{a}_2 \dots (\bar{a}_n + 1)$ із додаванням до молодшого розряду одиниці.

Правило переходу від прямого до доповняльного коду наступне:

- якщо в знаковому розряді міститься $(q-1)$, то всі цифри числа, крім розрядів знаків, замінюються відніманнями з $(q-1)$ значення розряду, а потім до цифри останнього молодшого розряду (якщо є дробові розряди, то до молодшого розряду дробу) додається одиниця;
- якщо в знаковому розряді знаходиться 00 (або 0), то перетворення не відбувається.

Наприклад:

$$\text{а) } -243476_{(10)} = 99.756523_{\text{об.}} + 1 = 99.756524_{\text{доп.}};$$

$$\text{б) } -0,111000111_{(2)} = 11.000111000_{\text{об.}} + 1 = 11.000111001_{\text{доп.}};$$

$$\text{в) } -0,101110_{(2)} = 11.010001_{\text{об.}} + 1 = 11.010010_{\text{доп.}}$$

$$\text{г) } 425736_{(10)} = 00.425736_{\text{об.}} = 00.425736_{\text{доп.}}$$

Для перевірки, (наприклад, б) потрібно до доповняльного коду від'ємного числа додати абсолютну величину прямого коду і одержати значення основи системи числення.

Перевірка.

$$\begin{array}{r} + 1.000111001_{\text{доп.}} \\ 0.111000111_{\text{пр.}} \\ \hline 10.000000000 = 2_{(10)} \end{array}$$

Таким чином, для додатних чисел прямий і доповняльний код мають одне і теж значення (приклад г), для від'ємного числа вони різні. Для додатного числа у розряді знаку завжди встановлюють нуль (або 00), а для від'ємного – $(q-1)$ або $(q-1)(q-1)$.

Узагальнюючи, можна визначити це математичним виразом, який підтверджує правило переведення чисел у доповняльний код.

$$N_{\text{доп.}} = \begin{cases} N, & \text{якщо } N \geq 0; \\ q - |N|, & \text{якщо } N < 0. \end{cases}$$

Звідси робимо висновок, що $N_{\text{доп.}} + |N| = q$, тобто, доповняльний код є математичним доповненням числа до основи системи числення.

Покажемо, що доповняльний код є доповненням до q – основи системи числення. Якщо до прямого коду, взятого по модулю, додати доповняльний, то одержимо значення основи системи числення. Для прикладу візьмемо:

$A_{\text{пр.}} = 1.1011$. Тоді, $|A_{\text{пр.}}| = 0.1011$.

$$\begin{array}{r} + A_{\text{пр.}} = 0.1011 \\ A_{\text{доп.}} = 1.0101 \\ \hline q = 10.0000 = 2_{(10)} \end{array}$$

Максимальне за модулем від'ємне число, яке можна при цьому записати, дорівнює $N_{\text{max}} = -0,111\dots11$ або $2^n - 1$, тоді доповняльний код буде $N_{\text{доп.}} = 1.00\dots01$. Мінімальне за модулем від'ємне число – це $-0,000\dots01$, що в доповняльному коді запишеться як $1.11\dots11$. Тому діапазон подання чисел у доповняльному коді має вигляд:

$$-2^n - 1 \leq N_{\text{max}} \leq -1.$$

5 ДОДАВАННЯ ДВІЙКОВИХ ЧИСЕЛ

5.1 Додавання чисел, поданих у формі з фіксованою комою, на двійковому суматорі прямого коду

Двійковим суматором прямого коду (ДСПК) називається суматор, у якому відсутній ланцюг міжрозрядного переносу між старшим цифровим і знаковим розрядами (рис. 5.1).

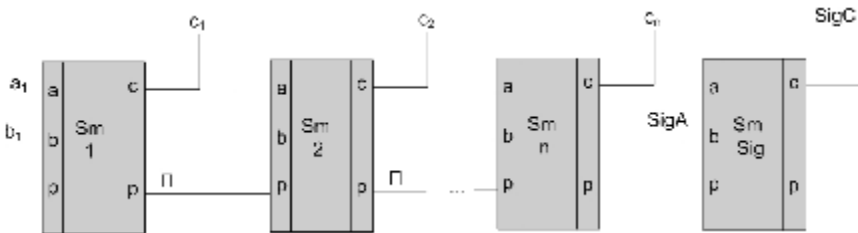


Рисунок 5.1 – Суматор прямого коду

На суматорах прямого коду можна складати тільки числа, що мають однакові знаки, а результат додавання менший за 2^n , тобто такий суматор не виконує операцію алгебраїчного додавання. При переповненні старшого n -розряду подається сигнал переповнення і подальші турботи перекладаються на користувача, а знак визначається за знаком будь-якого операнду, наприклад, *SigA*. Нехай задані операнди:

$$A_{\text{пр.}} = \text{Sig } A. a_1 a_2 \dots a_n,$$

$$B_{\text{пр.}} = \text{Sig } B. b_1 b_2 \dots b_n,$$

де *SigA*, *SigB* - вміст знакових розрядів. Якщо *SigA*=*SigB*, то сума чисел буде мати знак будь-якого з двох чисел які складаються, а цифрова частина визначиться порозрядним додаванням.

Приклад 1: Скласти $A_{\text{пр.}}=0,1011$ і $B_{\text{пр.}}=0,0100$.

Тут *SigA*=0, *SigB*=0, звідси *SigC*=0.

$$\begin{array}{r} 1011 \text{ (11)} \\ + 0100 \text{ (4)} \\ \hline 1111 \text{ (15)} \end{array}$$

Відповідь: $C=0.1111$.

Приклад 2: Скласти $A = -0,0101$ і $B = -0,1001$.

Тут $SigA=1$, $SigB=1$, звідси $SigC=1$.

$$\begin{array}{r} 0101 \text{ (5)} \\ 1001 \text{ (9)} \\ \hline 1110 \text{ (14)} \end{array}$$

Відповідь: $C_{пр.} = 1.1110$, $C = -0,1110$.

При додаванні чисел на ДСПК можливі випадки, коли абсолютне значення суми перевищує одиницю, тобто має місце переповнення розрядної сітки автомата. Ознакою переповнення $g=1$ буде наявність одиниці переносу зі старшого розряду цифрової частини суматора. За цим сигналом відбувається автоматична зупинка і корегування масштабних коефіцієнтів з таким розрахунком, щоб уникнути переповнення. Взагалі, це не дуже зручно. Тому, щоб уникнути переповнення та виконувати алгебраїчне додавання, використовують суматори доповняльного та оберненого кодів.

Алгебраїчне додавання двійкових чисел виконується на суматорах оберненого та доповняльного коду з урахуванням знаків та значення їх абсолютної величини за модулем. Тобто, результат буде мати знак того числа, яке за абсолютною величиною більше.

5.2 Додавання чисел на двійковому суматорі оберненого коду

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

Структурна схема ДСОК приведена на рис. 4.2.

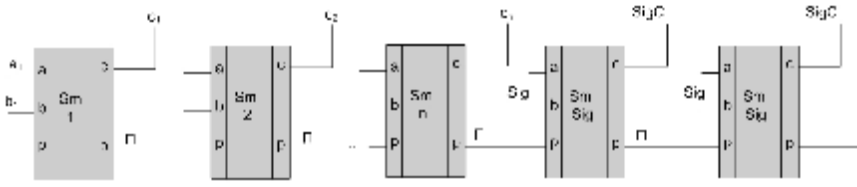


Рисунок 5.2 – Суматор оберненого коду

Суматор оберненого коду має:

- n суматорів за кількістю розрядів числа;
- два суматора знакових розрядів $SmSig$;
- ланцюги переносу одиниці переповнення із молодших розрядів числа до старших;

- ланцюг переносу одиниці переповнення зі старшого розряду числа до знакового розряду;

- ланцюг переносу одиниці переповнення зі старшого розряду знаку, за допомогою якого вона додається до молодшого розряду числа.

Суматор оберненого коду складається з n однорозрядних суматорів числової частини та двох розрядів знаку. Усі розряди розташовані паралельно, на входи a_i та b_i за синхроімпульсом одночасно на всі розряди подаються числа A і B . За принциповою схемою усі суматори однакові. Оскільки найбільше поширення одержали комбінаційні суматори, які не мають інерції щодо затримки сигналів виходу, то майже одночасно з сигналами на вході, на виході з'являються сигнали суми розрядів s_i , перенос між розрядами P_i та перенос переповнення знакових розрядів до молодшого розряду числа. **Основна властивість суматора оберненого коду – це наявність циклічного переносу між старшим знаковим розрядом та молодшим розрядом числа.** Цією властивістю він відрізняється від суматора доповняльних кодів, у якого переповнення знакових розрядів зникає.

Для обернених суматорів розглянемо наступну теорему.

Теорема. Якщо результат суми обернених кодів від'ємний, то він поданий оберненим кодом.

При доведенні цієї теореми припускаємо, що числа представлені у формі з фіксованою комою, що знаходиться перед старшим цифровим розрядом. Це гарантує рівність коефіцієнтів формату доданків $K_{\phi}A = K_{\phi}B$, які беруть участь у операції додавання.

Розглянемо наступні випадки.

Випадок 1. $A > 0$, $B > 0$, $A + B < 2^{n+1}$, тобто обидва числа додатні. Тоді $A_{\text{іа.}} + B_{\text{іа.}} = [A + B]_{\text{іа.}} = A + B$, тобто додавання здійснюється в прямих кодах.

Приклад до випадку 1:

Додати два числа $A=5$ і $B=7$.

Запишемо їх у двійковій системі. Оскільки обернений код додатного числа дорівнює прямому коду, то:

$$\begin{array}{r}
 A_{\text{об.}} = A_{\text{пр.}} = 00.0101 \\
 + B_{\text{об.}} = B_{\text{пр.}} = 00.0111 \\
 \hline
 C_{\text{об.}} = C_{\text{пр.}} = 00.1100
 \end{array}$$

Щоб запобігти переповненню розрядної сітки, ми використовуємо чотири розряди мантиси і два розряди знаку (модифікований код).

Відповідь: $C = 1100$ ($12_{(10)}$).

Випадок 2. $A < 0$, $B > 0$, $|A| > \hat{A}$, тут $A_{\text{іа.}} = 1.\bar{a}_1\bar{a}_2, \dots, \bar{a}_n$, $B_{\text{іа.}} = B_{\text{ід.}}$, тоді $A_{\text{іа.}} + B_{\text{іа.}} = [A + B]_{\text{іа.}}$, тобто результат буде від'ємний і у оберненому коді, треба перейти до $C_{\text{пр.}}$ шляхом інверсії розрядів числа

Приклад до випадку 2:

Додати числа $A = -11$ і $B = 7$ на ДСОК.

Переведемо числа у двійкову систему числення. $A_{\text{пр.}} = 11.1011$, $B_{\text{пр.}} = 00.0111$. Обернені коди будуть:

$$\begin{array}{r}
 A_{\text{об.}} = 11.0100 \\
 + B_{\text{об.}} = 00.0111 \\
 \hline
 C_{\text{об.}} = 11.1011
 \end{array}$$

Аналізуємо знак результату. Знак від'ємний, отже результат поданий оберненим кодом. Треба перевести його до прямого коду. $C_{\text{пр.}} = 11.0100$.

Відповідь: $C_{\text{пр.}} = 11.0100$, $C = -4_{(10)}$.

Випадок 3. $A < 0$, $B > 0$, $A < B$, тут $A_{\text{іа.}} = q - q^{-n} + A$. Тоді $A_{\text{іа.}} + B_{\text{іа.}} = q - q^{-n} + A + B$. Сума $(A + B)$ позитивна, тому права частина цього виразу стає більше q , що викликає появу одиниці переносу зі знакового розряду в молодший розряд числа (величина переносу при цьому, дорівнює $(q - q^{-n})$, тоді $A_{\text{іа.}} + B_{\text{іа.}} = A + B$. Результат додатний. Це і є $C_{\text{пр.}}$.

Приклад до випадку 3:

Додати числа $A = -5$ і $B = 7$ на ДСОК.

Переводимо числа у двійкову систему числення. $A_{\text{пр.}} = 11.0101$, $B_{\text{пр.}} = 00.0111$. Аналізуємо знаки чисел:

- якщо знак 11, то до суматора число надходить в оберненому коді;
- якщо знак 00, то в прямому коді. Тоді:

$$\begin{array}{r}
 + \quad A_{об.} = 11.1010 \\
 \quad B_{об.} = B_{пр.} = 00.0111 \\
 \hline
 C_{об.} = C_{пр.} = 100.0001 \\
 + \quad \underline{\quad 1 \quad} \\
 \hline
 00.0010
 \end{array}$$

Аналізуємо знак результату:

- якщо знак результату 11, то результат від'ємний і записаний в оберненому коді. Необхідно виконати переведення в прямий код.
- якщо знак результату 00, то результат додатний і поданий прямим кодом.

Відповідь: $C = 00.0010$ ($2_{(10)}$).

Випадок 4. $A < 0$, $B < 0$, але $|A + B| < 2^{n+1}$.

Тут $A_{об.} = q - q^{-n} + A$, $B_{об.} = q - q^{-n} + B$. Тоді $A_{об.} + B_{об.} = q - q^{-n} + A + q - q^{-n} + B$. Тут з'являється одиниця перенесення, що рівнозначно вилученню з суми $q - q^{-n}$, тобто

$$A_{об.} + B_{об.} = q - q^{-n} + A + B = [A + B]_{об.}.$$

Приклад до випадку 4.

Знайти суму чисел $A = -5$; $B = -8$, використовуючи ДСОК. Запишемо числа у машинних прямих кодах. $A_{пр.} = 11.0101$, $B_{пр.} = 11.1000$. Числа від'ємні, тому переводимо їх до обернених кодів. $A_{об.} = 11.1010$, $B_{об.} = 11.0111$. Складаємо на ДСОК:

$$\begin{array}{r}
 + \quad A_{об.} = 11.1010 \\
 \quad B_{об.} = 11.0111 \\
 \hline
 C_{об.} = 111.0001 \\
 + \quad \underline{\quad 1 \quad} \\
 \hline
 C_{об.} = 11.0010
 \end{array}$$

Аналізуємо знак результату. Знак результату від'ємний. Це значить, що знак числа від'ємний, а число треба перевести з оберненого коду до прямого. $C_{пр.} = 11.1101$.

Висновок. Теорема справедлива для всіх розглянутих випадків, навіть для тих, де виникає переповнення розрядної сітки. Це дозволяє виконувати на суматорі обернених кодів алгебраїчне додавання чисел за правилами двійкової арифметики з автоматичним визначенням знаку суми.

Сформулюємо правило алгебраїчного додавання чисел формату з фіксованою комою на ДСОК при рівних коефіцієнтах формату доданків $K_{\Phi}A=K_{\Phi}B$.

Правило. При алгебраїчному додаванні чисел на суматорі обернених кодів аналізується знак кожного доданку. Якщо знак додатний, то до суматора число разом зі знаком надходить у прямому коді. Якщо знак від'ємний, то до суматора число надходить у оберненому коді. Після складання доданків аналізується знак результату. Якщо знак результату додатний, то число додатне і записано в прямому коді. Якщо знак від'ємний, то число від'ємне і подане в оберненому коді і його необхідно перевести до прямого коду.

Зауважимо, що одиниця переповнення у знаковому розряді не зникає, а завжди додається до молодшого розряду числа (якщо є дріб, то до молодшого розряду дробу).

5.3 Додавання чисел на двійковому суматорі доповняльного коду

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

Основною особливістю ДСДК є наявність кола переносу одиниць переповнення зі старшого розряду цифрової частини c_n у знаковий розряд (рис. 5.3), і відсутність зворотного зв'язку переносу одиниць переповнення зі старшого знакового розряду в самий молодший розряд числа.

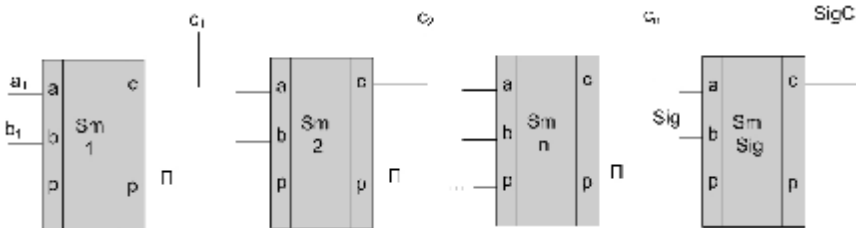


Рисунок 5.3 – Суматор доповняльного коду

Для визначення правил додавання чисел у ДСДК необхідно розглянути наступну теорему, доказ якої базується на розгляді конкретних випадків.

Теорема: *Якщо результат суми чисел доповняльних кодів від'ємний, то він поданий у доповняльному коді.* При доведенні теореми, припускаємо, що числа подані у формі з фіксованою комою, що знаходиться перед старшим розрядом числа, тобто, після знакових розрядів. Це гарантує рівність коефіцієнтів формату додатків $K_{\Phi}A = K_{\Phi}B$, які беруть участь у операції додавання.

Розглянемо наступні випадки.

Випадок 1. $A > 0$, $B > 0$, а $A + B < 2^{n+1}$ (n – номер старшого незнакового розряду, рис. 8.1). Числа додатні, переповнення до знакових розрядів нема. Для цього випадку $A_{\text{дп.}} = \hat{A}_{\text{дп.}}$, $\hat{A}_{\text{дп.}} = \hat{A}_{\text{дп.}}$, тоді $A_{\text{дп.}} + B_{\text{дп.}} = A_{\text{пр.}} + B_{\text{пр.}} = C_{\text{пр.}}$ – результат додатний. Покажемо це.

Приклад до випадку 1:

$A_{\text{пр.}} = 0.11011$; $B_{\text{пр.}} = 0.00100$. Знаки чисел додатні, тому

$$\begin{array}{r} A_{\text{дп.}} = 0.11011 \\ + B_{\text{дп.}} = 0.00100 \\ \hline C_{\text{дп.}} = 0.11111 \end{array}$$

У цьому випадку $C_{\text{дп.}} = C_{\text{пр.}} = 0.11111$.

Випадок 2. $A < 0$, $B > 0$, але $|A| > |B|$. Для цього випадку

$$A_{\text{дп.}} = q + |-A|, B_{\text{дп.}} = B_{\text{пр.}},$$

тоді, $A_{\text{дп.}} + B_{\text{дп.}} = q + |-A_{\text{пр.}}| + B_{\text{пр.}} = C_{\text{дп.}}$ – результат буде з від'ємним знаком, число треба перевести в прямий код.

Приклад до випадку 2:

$A_{\text{пр.}} = 1.1011$; $B_{\text{пр.}} = 0.100$. Тоді $A_{\text{дп.}} = 1.0101$, $B_{\text{дп.}} = B_{\text{пр.}} = 0.0100$.

$$\begin{array}{r} A_{\text{дп.}} = 1.0101 \\ + B_{\text{дп.}} = 0.0100 \\ \hline C_{\text{дп.}} = 1.1001 \end{array}$$

Отриманий результат від'ємний, це значить, що він поданий у доповняльному коді і його необхідно перевести в прямий, щоб користуватись ним для подальших перетворень

$C_{\text{доп.}}=1.1001$; $C_{\text{об.}}=1.0110$; $C_{\text{пр.}}=1.0111$.

Переведення числа з доповняльного коду до прямого здійснюється за наступним алгоритмом: усі розряди числа (крім знакових) інвертуються (тобто, береться обернений код результату) і в молодший розряд додається одиниця. Від'ємний знак зберігається.

Випадок 3. $A < 0$, $B > 0$, але $|A| < |B|$. Для цього випадку $A_{\text{доп.}} = q + |-A|$, $B_{\text{доп.}} = B_{\text{пр.}}$, тоді, $A_{\text{доп.}} + B_{\text{доп.}} = q + |-A_{\text{пр.}}| + B_{\text{пр.}}$. Значення цієї суми більше q , тому з'являється одиниця переносу в знаковий розряд, що рівнозначно вилученню із суми q , тобто результат має бути додатний, а код прямий.

Приклад до випадку 3.

$A_{\text{пр.}}=1.0100$; $B_{\text{пр.}}=0.1011$. Запишемо в кодах машини. $A_{\text{пр.}}=1.0100$; $B_{\text{пр.}}=0.1001$. У доповняльному коді: $A_{\text{доп.}}=1.1100$; $B_{\text{доп.}}=B_{\text{пр.}}$. Тоді

$$\begin{array}{r} A_{\text{доп.}}=1.1100 \\ + B_{\text{доп.}}=0.1011 \\ \hline C_{\text{доп.}}=0.0111=C_{\text{пр.}} \end{array}$$

Одержали додатний знак результату. Це означає: число додатне, а код прямий.

Випадок 4. $A < 0$, $B < 0$, а $|A+B| < 2^{n+1}$. Для цього випадку $A_{\text{доп.}} = q + (-A)$, $B_{\text{доп.}} = q + (-B)$, тоді, $A_{\text{доп.}} + B_{\text{доп.}} = [A+B]_{\text{доп.}}$ – результат від'ємний і з'явиться одиниця переносу зі знакового розряду.

Приклад до випадку 4.

Додати числа на ДСДК. $A_{\text{пр.}}=1.0100$; $B_{\text{пр.}}=1.1011$. Доповняльні коди будуть:

$$\begin{array}{r} A_{\text{доп.}}=1.1100 \\ + B_{\text{доп.}}=1.0101 \\ \hline C_{\text{доп.}}=1.0001 \end{array}$$

Аналізуємо знак результату. Він від'ємний. Тобто число від'ємне, а код доповняльний. Потрібно перейти до прямого коду. $C_{\text{пр.}}=1.1111$;

Висновок. Теорема справедлива для всіх випадків, навіть там де виникає переповнення розрядної сітки. Це дозволяє складати на машині числа за правилами двійкової арифметики.

Сформулюємо правило алгебраїчного додавання чисел формату із фіксованою комою при рівних коефіцієнтах формату доданків $K_{\phi}A=K_{\phi}B$.

Правило. *При додаванні чисел на суматорі доповняльних кодів аналізується знак кожного доданку. Якщо знак додатний, то до суматора число надходить у прямому коді. Якщо знак від'ємний, то до суматора число надходить у доповняльному коді. Після додавання аналізується знак результату. Якщо знак результату додатний, то число додатне і записане в прямому коді. Якщо знак від'ємний, то число від'ємне і подане в доповняльному коді і його необхідно перевести до прямого коду.*

Правила додавання для суматора доповняльного коду можна поширити і на інші системи числення, наприклад, на десяткову, в якій знак числа визначають:

- додатний, як 00;
- від'ємний, як 99.

Це те саме правило ($q-1$) для знаку, а значення кожного розряду доповняльного коду одержуємо за алгоритмом, від ($q-1$) віднімаємо значення за кожним розрядом числа прямого коду. Одержимо обернений код, а потім до молодшого розряду додамо одиницю.

Приклад для десяткових чисел. Додати десяткові числа в доповняльних кодах: $A=732904$; $B=-42703$. Урівняємо розрядність чисел і запишемо їх у кодах машини. $A_{\text{пр.}}=00.732904$; $B_{\text{пр.}}=99.042703$. Переведемо числа у доповняльний код та додамо їх.

$$\begin{array}{r} + \text{Адоп.} = 00.732904 \\ \text{Вдоп.} = 99.957297 \\ \hline \text{Сдоп.} = 00.690201 = \text{Спр.} \end{array}$$

Одиниця переповнення у знакових розрядах пропадає. Результат легко перевірити.

6 МОДИФІКОВАНІ БІНАРНІ КОДИ

6.1 Переповнення розрядної сітки

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

6.1.1 Переповнення при складанні прямих кодів

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

Приклад. $A = +1010$, $B = +1101$.

$$A_{np} = 0 \setminus 1010 \quad (+10)$$

$$+B_{np} = 0 \setminus 1101 \quad (+13)$$

$C_{np} = 0 \setminus 0111 \quad (+7 \neq +23)$. Отримано спотворений (невірний) результат через втрату переносу П зі старшого цифрового розряду при складанні на ДСПК.

6.1.2 Переповнення при складанні доповняльних кодів

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

Приклад. $A_m = 0 \setminus 1011 \quad (+11)$, $B_m = 0 \setminus 1010 \quad (+10)$ (перевірку отриманого результату проведіть самостійно).

$$\begin{array}{r} A_{don} = 0 \setminus 1011 \\ + B_{don} = 0 \setminus 1010 \\ \hline C_{don} \neq 1 \setminus 0101 \end{array}$$

6.1.3 Переповнення при складанні в обернених кодах

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

Приклад. $A_m = 0 \setminus 0111 \quad (+7)$, $B_m = 0 \setminus 1101 \quad (+11)$.

$$\begin{array}{r} A_{oo} = 0 \setminus 0111 \\ + B_{oo} = 0 \setminus 1101 \\ \hline C_{oo} = 1 \setminus 0100 \end{array}$$

Знак числа спотворений (невірний).

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

У машинному представленні чисел, представлених в модифікованому коді, використовується $(n+2)$ розрядів: з них два знакових $Sg1$ (старший) і $Sg2$ (молодший), кома опускається, встановлюється постійний коефіцієнт формату K_ϕ .

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

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

Щоб фіксувати переповнення розрядної сітки в ДСПК, ДСДК, ДСОК, вводять допоміжний розряд в знакову частину зображення числа, який називають розрядом переповнення числа. Таке представлення числа називається модифікованим (рис. 6.1).

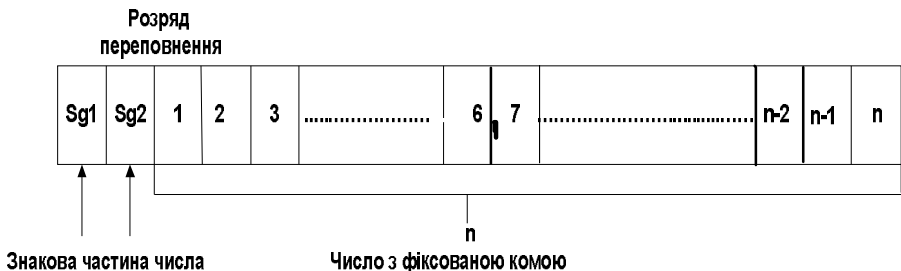


Рисунок 6.1 – Модифікований код числа для ФФК

Приклади. а) $A_m = +1011 (+11)$, $B_m = +1010 (+10)$.

$$\begin{aligned}
 A_o^m &= 00\backslash 1011 \\
 + B_o^m &= 00\backslash 1010 \\
 C_o^m &= 01\backslash 0101 = 00\backslash 0101 \quad (+21)
 \end{aligned}$$

Тут ознакою переповнення є **1** в знаковому розряді. За цією ознакою проводиться зсув числа вправо з одночасним збільшенням коефіцієнта формату K_ϕ на 1 і відновленням Sg_1 (за значенням Sg_2) обох розрядів $Sg_1 Sg_2$, тобто знаку 00 результату C числа, далі перевіряється нормалізація числа (наявність 1 після коми в числі).

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

$$\begin{array}{r} A^m_\phi = 11 \setminus 0101 \\ + B^m_\phi = 11 \setminus 0111 \\ \hline C^m_\phi = 10 \setminus 1100 \end{array} \quad \begin{array}{l} \text{Було переповнення. Потрібен зсув вправо і відновлення} \\ \text{знака. } C^m_\phi = 11 \setminus 01100 \text{ результату і перетворення його в прямий код.} \\ C^m_{np} = 11 \setminus 10011 \\ + \quad \quad 1, \text{ тоді } C_{np} = 11 \setminus 10100 \text{ } (-20). \end{array}$$

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

6.2 Модифіковане додавання чисел у форматі із рухомою комою (крапкою)

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

Для чисел з рухомою комою (крапкою) справедлива умова нормалізації:

$$q^{-1} \leq |m_A| < 1, \quad (6.1)$$

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

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

$$0,5 \leq |m_A| < 1, \quad (6.2)$$

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

Операція нормалізації числа складається з умови нормалізації (6.1) і здійснюється методом зсуву мантиси числа в ту або іншу сторону.

Зсув може проводитися вліво або управо в межах розрядної сітки ЦА за правилами представлених моделлю таблиці 5.1.

Таблиця 6.1 – Зсув звичайних кодів числа (один розряд знаку)

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

Модифікований зсув - це операція над модифікованим зображенням числа, яка виконується за моделлю таблиці 9.2.

Таблиця 6.2 - Зсув модифікованих кодів числа (два розряди знаку)

Початкове число	Зсув вліво на один розряд (від порядку вираховується 1)	Зсув вправо на один розряд (до порядку додається 1)
$00/a_1, a_2, \dots, a_n$	$0a_1/a_2, \dots, a_n 0$	$00/0, a_1, a_2, \dots, a_n$
$01/a_1, a_2, \dots, a_n$	$1a_1/a_2, \dots, a_n 0$	$00/1, a_1, a_2, \dots, a_n$
$10/a_1, a_2, \dots, a_n$	$0a_1/a_2, \dots, a_n \varepsilon$	$11/0, a_1, a_2, \dots, a_n$
$11/a_1, a_2, \dots, a_n$	$1a_1/a_2, \dots, a_n \varepsilon$	$11/1, a_1, a_2, \dots, a_n$

Величина ε залежить від коду. Для доповняльного коду $\varepsilon=0$, для оберненого коду $\varepsilon=1$. При складанні чисел результат може вийти з нормалізації як зліва, так і справа (див. табл. 6.1, 6.2).

Позначемо через γ ознаку порушення нормалізації числа справа (коли переповнення), вказує на необхідність зсуву числа вправо на один розряд для відновлення знаку числа.

Ознакою порушення нормалізації числа зліва $\delta=1$ (коли результат по абсолютній величині виявляється менше $1/q$) є

наявність однакових комбінацій в розряді переповнення і старшому розряді (R1) цифрової частини суматора.

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

Приклад 1. Знайти суму чисел: $A = +0,1000 \cdot 2^{-3}$ та $B = -0,1011 \cdot 2^{-3}$.

Мантиси і порядок обробляються на ДСДК у ФРК.

$$\begin{array}{ll} [m_A]_0 = 00/1000 & [P_A]_0 = 11/101 \\ + [m_B]_0 = 11/0101 & [P_B]_0 = 11/101 \\ \hline [m_c]_0 = 11/1101 \end{array}$$

Тут $Sg2 \& R1 = 1$, тобто $\delta=1, \gamma=0$. Це означає, що необхідний зсув мантиси $[m_c]_0$ вліво на 1 розряд.

$[m'_c]_0 = 11/1010$ Перевіряємо. Знову $\delta=1, \gamma=0$ – необхідний ще зсув вліво на 1 розряд.

$[m''_c]_0 = 11/0100$. Перевіряємо, все гаразд $\delta=0, \gamma=0$.

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

$$\begin{array}{ll} + [P_c]_0 = 11/101 & \\ \hline [\Delta P_c]_0 = 11/110 & [\Delta P_c]_{np} = 11/010 \\ [P''_c]_0 = 11/011 & P_{c\ np} = -11/101 \text{ (тобто } P = 2^{-5}) \end{array}$$

тоді результат (число) дорівнює:

$$\begin{array}{l} [m''_c]_d = 11/0100 \\ [m''_c]_{006} = 11/1011 \\ + \underline{\quad 1 \quad} \\ 11/1100. \end{array}$$

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

7 СКЛАДАННЯ ЧИСЕЛ ПРИ РІЗНИХ ЗНАЧЕННЯХ ПОРЯДКІВ

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

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

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

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

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

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

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

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

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

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

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

- якщо знак 01(або 10), то знак спочатку відновлюється шляхом зсуву мантиси вправо на один розряд з одночасним збільшенням порядку на 1;

- якщо знак 00, то результат - число позитивне, представлено в прямому коді;

- якщо знак 11, то результат - число негативне і представлено в оберненому (доповняльному) коді.

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

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

Приклад. Скласти $A = +0,1011 \cdot 2^{-2}$ і $B = -0,1001 \cdot 2^{-3}$

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

Рішення. 1. Враховуючи, що числа задані вже в нормалізованому вигляді (після коми стоїть 1), представимо їх в машинному вигляді у форматі з рухомою крапкою (модифікований код): $A_{np}^m = 00.1011.11.10$; $B_{np}^m = 11.1001.11.11$.

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

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

$P_{Aob} = 11.01$; $P_{Bob} = 11.00$. Тоді, $\Delta P = P_{Aob} - P_{Bob} = 11.01 - 11.00$

Замінімо операцію віднімання на операцію складання.

$\Delta P = P_{Aob} - P_{Bob} = P_{Aob} + P_{Bnp} = 11.01 + 00.11$ (де P_{Bnp} – інверсний код P_{Bob} , включаючи і знакову частину).

Виконаємо додавання.

$$\begin{array}{r}
 11.01 \\
 +00.11 \\
 \hline
 100.00 \\
 + \mathbf{a} \rightarrow 1
 \end{array}$$

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

$$\begin{array}{l}
 m_{Aob}=00.1011 \quad (\text{тут } \delta=0, \gamma=0) \\
 m_{Bob}=11.0110 \quad (\text{тут } \delta=0, \gamma=1, \text{ необхідно зробити зсув } m_{Bob} \text{ вправо}) \\
 m_{Bob}=11.1011 \\
 +m_{Aob}=00.1011 \\
 \hline
 m_{Cob}=100.0110 \\
 \mathbf{a} \rightarrow \rightarrow 1 \\
 \hline
 00.0111 \quad (\text{тут } \delta=1, \gamma=0; \text{ необхідно зробити зсув ліво}) \\
 m_{Cob}=00.1110 \quad \text{і тому робимо корекцію порядку на } \Delta P=-1 \\
 P_{Aob}=11.01 \\
 +\Delta P_{ob}=11.10 \quad (\Delta P \text{ представлено в оберненому коді}) \\
 \hline
 110.11 \\
 + \mathbf{a} \rightarrow 1
 \end{array}$$

$P_{Cob}=11.00$ Результат P_{Cob} отриманий в оберненому коді, тобто знак негативний (11). Визначаємо прямий код порядку і записуємо результат $C_{np}=00.1110 \cdot 2^{-3}$. Відповідь: $C_{np}=00.1110.11.11$

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

5. Перевіримо умову нормалізації мантиси результату справа, зліва.

Приклад. Скласти на ДСДК числа $A_{np}=+5$, $B_{np}=-3$, представлених в десятковій системі числення. $A_\delta=00/5$; $B_\delta=99/7$. Проведемо складання $A_\delta + B_\delta = 00/5 + 99/7 = 100/2 = 00/2 (+2_{(10)})$. Одиниця (1) пропадає, оскільки складання йде на ДСДК.

8 МНОЖЕННЯ ДВІЙКОВИХ ЧИСЕЛ

8.1 Методи множення бінарних чисел

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

$ \begin{array}{r} 347 \\ \underline{532} \\ 694 \\ +1041 \\ \underline{+1735} \\ 184604 \end{array} $	$ \begin{array}{r} 347 \text{ множене (A)} \\ \underline{532 \text{ множник (B)}} \\ 1735 \text{ частковий (розрядний) добуток} \\ +1041 \\ \underline{+694} \\ 184604 \text{ повний добуток (C)} \end{array} $
---	--

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

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

$ \begin{array}{r} 1101 \\ * 1101 \\ \hline 1101 \\ +0000 \\ +1101 \\ \hline + 1101 \\ \hline 10101001 \end{array} $	$ \begin{array}{r} 1101 \\ * 1101 \\ \hline 1101 \\ + 1101 \\ + 0000 \\ \hline + 1101 \\ \hline 10101001 \end{array} $
---	---

В обох випадках операція множення складається з ряду послідовних операцій зсуву і складання часткових добутків. Таким чином, операція множення зводиться до складання часткових добутків, які виходять з множеного або нулів, якщо в розряді множника «нуль», або множеного, якщо в розряді множника «одиниця» і відповідним зсувом.

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

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

Розглянемо основні методи виконання операції множення ЦА.

8.2 Множення чисел з фіксованою крапкою (комою) на ДСПК

Запишемо машинне зображення множеного і множника у формі з фіксованою комою в прямому коді. $A_{np} = Sg, \alpha_1 \alpha_2 \dots \alpha_n$; $B_{np} = Sg, b_1 b_2 \dots b_n$. Тоді, їхній добуток запишеться як $C_{np} = Sg, c_1 c_2 \dots c_n$, де $Sg_c = Sg_A \oplus Sg_B$, де $\oplus$ – знак додавання функції « $\Sigma \bmod 2$ » (її ТІ: $1 \oplus 1 = 0$, $0 \oplus 0 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$).

Таким чином, при використанні ДСПК, знак добутку визначається окремо від цифрової частини, потім виконується операція множення. Вона виконується відповідно до заданої структури множного пристрою (див. наприклад, рис. 7.1) і методу множення (див. наприклад, метод 2).

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

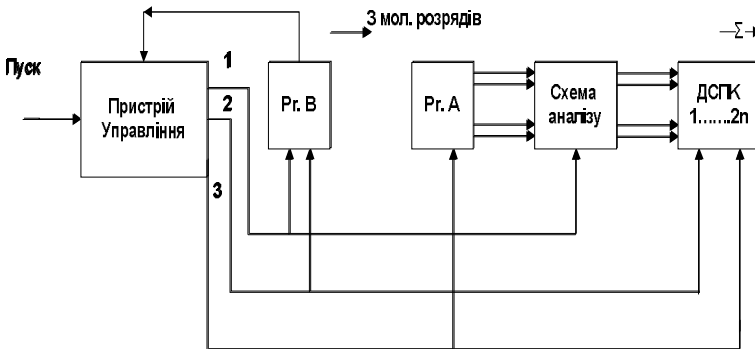


Рисунок 8.1 – Структура ЦА для пристрою множення чисел з фіксованою комою

Приклад. Помножити $[A_{np}] = 11/11010$ (-26); $[B_{np}] = 00/11001$ (25), $C = -650$.

Рішення. Визначається знак добутку C : $1 \oplus 0 = 1$.

Прийmemo: а) суматор має 10 розрядів (без знака).

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

Послідовність дій представимо таблицею 7.1

Для спрощення запису таблиць, прийmemo наступні умовні позначки:

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

- позначення, наприклад, $[C_m]$ – вміст суматора;

¾ ®

- оператор $[PrA]$ зсуву вмісту, наприклад, регістра A вправо на один розряд;

- позначення V, P – вхідне положення;

- позначення A_{np}, B_{np} - цифрова частина множеного, множника (прямий код).

Відповідь: $[C_{np}] = 11/1010001010$.

Таблиця 8.1 – Приклад рішення

Суматор $[C_m]$	Регістр $[PrB]$	Коментар
+0000000000	11001	$i.P.[C_m] := 0; PrA := A;$ $PrB := B$
11010		$b_5 = 1; [C_m] + [PrA];$
1101000000		$[PrB]; [C_m];$
0110100000	-1100	$b_4 = 0; [PrB]; [C_m];$
0011010000	--110	$b_3 = 0; [PrB]; [C_m];$
+0001101000	---11	$b_2 = 1; [C_m] := [C_m] + [PrA];$
11010		$[C_m]; [PrB];$
1110101000	----	$b_1 = 1; [C_m] := [C_m] + [PrA];$
+0111010100		$[C_m]; [PrB];$
11010		$b_1 = 1; [C_m] := [C_m] + [PrA];$
10100010100		$[PrB]; [C_m];$
1010001010		Кінець

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

8.3 Множення чисел з рухомою комою

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

- перемножування мантис;
- додавання порядків.

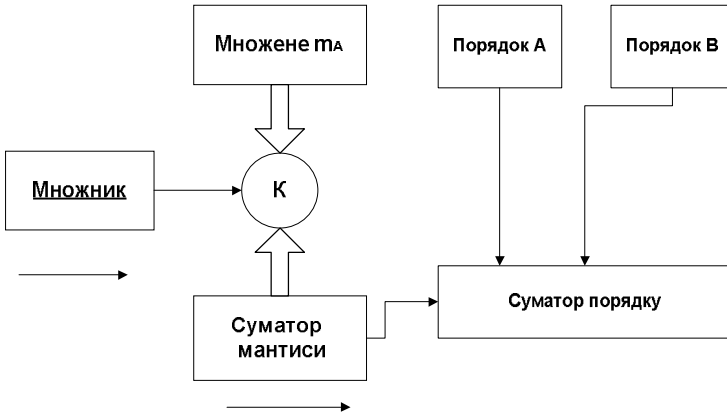


Рисунок 8.2 – Структура ЦА для пристрою множення чисел у ФРК

Таблиця 8.2 – Перемножування мантис A і B .

Знак результату	Суматор [см]	Регістр m_b	Коментар
$S_{gc} = S_{gA} \oplus S_{gB} = 1 \oplus 0 = 1$	+.00000	10011	і.п. $P_r m_B := \{m_B\}$; $P_r m_A := \{m_A\}$; $P_r P_A := \{P_A\}$; $P_r P_B := \{P_B\}$; $C_{mmC} = 0$
	<u>11001</u>		$b_5 = 1$; $C_m := [C_{mmC}] + [P_r m_A]$;
	11001	01001	$[P_r m_B]$; $[cmm_c]$;
	+.011001		$b_4 = 1$; $[cmm_c] := [cmm_c] + [P_r m_A]$;
	<u>11001</u>		$[cmm_c]$; $[P_r m_B]$
	1001011	00100	
	01001011	00010	$b_3 = 0$; $[cmm_c]$; $[P_r m_B]$.
	001001011	00001	$b_2 = 0$; $[cmm_c]$; $[P_r m_B]$.
	+.001001011		$b_1 = 1$; $[cmm_c] := [cmm_c] + [P_r m_A]$;
	<u>11001</u>		$[P_r m_B]$; $[cmm_c]$;
	111011011	00000	
	0111011011		Кінець

Результат множення може вийти денормалізованим, тому потрібна перевірка на нормалізацію числа (критерій δ) та, при необхідності, його нормалізація з відповідною корекцією порядку P результату. Пристрій множення чисел із рухомою комою представлено ЦА зі структурою рис. 8.2.

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

$$[P_A]_{o\delta} = 11/100$$

$$[P_B]_{o\delta} = 00/001$$

$$[P_C]_{o\delta} = 11/101 \quad [C]_{np} = 11/010; C_{np} = -010_{(2)}$$

Тому що мантиса результату не задовольняє нормалізації (тобто $\delta=1$, $\gamma=0$), то робимо зсув мантиси вліво на один розряд і проводимо корекцію порядку: $m=11/1110110111$.

$$+[P_C]_{o\delta} = 11/101$$

$$[\Delta P]_{o\delta} = \frac{11/110}{111/011}$$

$$\underline{a \rightarrow \rightarrow 1}$$

$$[P_C]'_{o\delta} = 11/100; [P_C]'_{np} = 11/011 (-3).$$

$$\text{Відповідь: } C = -0,1110110111 \cdot 2^{-3}$$

В машинному коді це буде відповідати коду: 11.1110110110.11.11 (при тому, що порядок P має тільки два розряди).

Примітка. Крпки (.), коми (,) слеш (\), що розділяють знак мантиси, мантису m , знак порядку, порядок P в машинному коді не проставляються і використовуються лише при запису чисел.

9 МНОЖЕННЯ ЧИСЕЛ НА ДСДК

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

9.1 Множення чисел на ДСДК при позитивному множнику

При позитивному множнику можна сформулювати алгоритм множення чисел на ДСДК.

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

Приклад. Помножити числа $A = -10101$ (-21), $B = 10011$ ($+19$).

Метод 2. ДСДК. Запишемо числа $A_{дон}^m = 11\backslash 01011$; $B_{дон}^m = 00\backslash 10011$. Суматор має 7 розрядів, RgB 5 розрядів. Рішення наведено в таблиці 9.1.

Результат отриманий у доповняльному коді зі знаком мінус (11). Після перетворення його в прямий код ($C_{пр} = 11\backslash 0110001111$) знак зберігається. Для зменшення апаратних засобів, результат зберігається в суматорі і RgB (при зсувах молодші розряди суматора надходять у старші розряди RgB). Цим і визначається проведений раніш вибір розрядності пристрою.

9.2 Множення чисел на ДСДК при негативному множнику

Другий випадок, коли A – будь-яке число, а множник $B < 0$. $B_{дон} = 1, \overline{e_1 e_2 \dots e_n}$, де $\overline{e_n}$ означає $e_n + 1$

При негативному множнику добуток доповняльних кодів операндів не дорівнює доповняльному коду результату.

Тому множення виконується аналогічно підрозділу 7.1, але з додаванням до результату поправки.

ПРАВИЛО. Якщо множник негативний, то добуток чисел на суматорі доповняльного коду дорівнює додатку виправлення

$[A_{\text{доп}}^M] + 1$ до добутку доповняльних кодів співмножників. При цьому, виправлення - це повністю інвертоване $A_{\text{доп}}^M$, включаючи і знак з додаванням 1 до молодшого розряду.

Таблиця 9.2 – Множення чисел на ДСДК при $B < 0$

Суматор [см]	Регістр (PrB)	Коментар
+ 00.00000	00111	$i. П. см := 0; PrB := B_{доп}^M; PrA := A_{доп}^M$
<u>11.01001</u>		$b_3 = 1; [см] := [см] + [A_{доп}^M];$
11.01001		
11.10100	10011	$[PrB]; [см];$
+ <u>11.01001</u>		$b_4 = 1; [см] := [см] + [A_{доп}^M];$
10.11101		
+ 11.01110		$[PrB]; [см];$
<u>11.01001</u>		
10.10111	11001	$b_3 = 1; [см] := [см] + [A_{доп}^M];$
11.01011	11100	$[PrB]; [см];$
11.10101	11110	$b_2 = 0; [PrB]; [см];$
+ 11.11010	11111	$b_1 = 0; [PrB]; [см];$
<u>00.10111</u>		$[см] := [см] + [A_{доп}^M];$
00.10001		Кінець

Таким чином на ДСДК одночас отримуємо і знакову і цифрову частини результату в обох випадках множення (при $B > 0$ і $B < 0$).

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

Підготовчі роботи. Запишемо: $A_{доп}^M = 11\backslash 01001$; $B_{доп}^M = 11\backslash 00111$; $(-A) = A_{\text{дир}} = 00\backslash 10111$.

Рішення наведено в таблиці 12.2. $C = 00\backslash 100011111$ ($+575$).

Перевірка: $(-23) \cdot (-25) = +575$

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

При виконанні операції множення, використовуємо режим економії апаратних засобів, тобто при зрушенні суматора інформація з молодших розрядів переходить у старші розряди регістру В.

10 ДІЛЕННЯ БІНАРНИХ ЧИСЕЛ

10.1 Метод ділення у бінарних чисел

Найбільше поширення одержав метод виконання операції ділення у чисел шляхом віднімання.

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

Розглянутий метод ставиться до "шкільних" алгоритмів ділення з відновленням залишку. Приклад. $A = +1100100 = 100_{(10)}$, $B = +1010 = 10_{(10)}$.

Рішення наведено нижче.

Ділене	<u>1100100</u>	<u>1010</u>
Дільник	<u>1010</u>	1010
Залишок 1	<u>00101</u>	
	<u>1010</u>	
Залишок 2	<u>1011</u> *	
Відновлення	<u>1010</u>	
залишку	<u>01010</u>	
	<u>1010</u>	
	0000	

При цьому використовують правила бінарного віднімання. У комп'ютері операція віднімання замінюється на додавання в доповняльному коді, тому алгоритм ділення відрізняється. Знаком * позначено від'ємний залишок (тобто «Залишок 2» < 0).

Формально алгоритм описується в такий спосіб. Нехай A – ділене, B – дільник, C – частка: $A = 0, \alpha_1 \alpha_2 \dots \alpha_n$; $B = 0, b_1 b_2 \dots b_m$, $C = 0, c_1 c_2 \dots c_r$.

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

$A_i = A_{i-1} + B \cdot 2^{-i}$, а на наступному кроці після зсуву визначається новий залишок і т.д.

Даний алгоритм ділення з відновленням залишку реалізується на двійкових суматорах оберненого (ДСОК) або доповняльного (ДСДК) кодів.

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

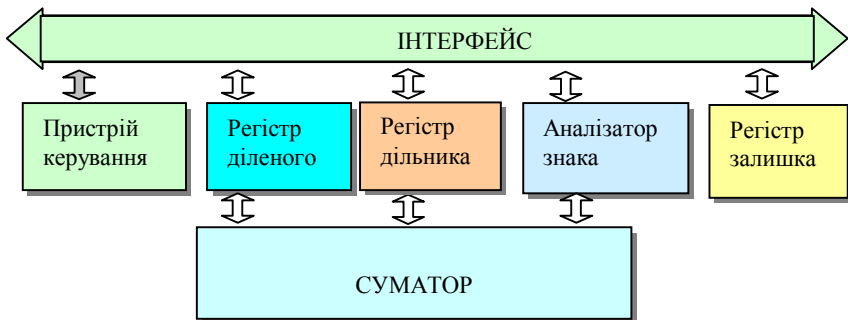


Рисунок 10.1 – Структурна схема ЦА операції ділення

10.2 Ділення чисел з фіксованою комою з відновленням залишку

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

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

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

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

2) якщо знак проміжного залишку 11 (негативний), то в регістр частки R_C записується 0, починаючи зі старшого розряду. Залишок відновлюється шляхом додатка до нього дільника в прямому коді з позитивним знаком. Відновлений залишок зсувається вліво на один розряд (крапку, що відокремлює знак, перенести вправо на один розряд), зноситься наступний розряд діленого (що не брав участь до цього в ділені). Після цього, проміжний залишок підготовлений до наступного додатка дільника у доповняльному коді;

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

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

Знак результату привласнюється відповідно до пункту 1.

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

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

1. Визначаємо знак частки: $Sg_C = Sg_A \oplus Sg_B = 1 \oplus 0 = 1$: негативний (-).

2. Записуємо машинні зображення чисел: $A^M_{np} = 00/100111$; $B^M_{np} = 11/11$; $B^M_{don} = 11/01$;

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

Примітка: ДСДК не використовує перенос Π_{Sg1} при переповненні суматора старшого розряду знака.

Відповідь: $C_{np} = 11/01101$ (-13).

У таблиці 10.1 вертикальними стрілками показані знесення наступних розрядів діленого в проміжні залишки. Горизонтальні стрілки відображають розряди запису результату в регістр частки R_C . Одиниці переповнення, що одержані після підсумовування, пропадають, тому що використовується ДСДК. При виконанні операції

лівого зсуву залишків, переноситься вправо крапка, що відокремлює знак числа(при цьому, старший розряд знака суматора Sg пропадає).

Таблиця 10.1 – Приклад ділення чисел A на B з відновленням залишку

00.100111		
+ 11.01	→	$\overline{PrC}$
11.11		0 (Старший розряд C)
+ 00.11		Відновлення i -ї частки
00.10	↓	
01.00		
+ 11.01	→	1
00.11		
+ 11.01	→	1
100.00		
00.01		
+ 11.01	→	0
11.10		
+ 00.11		Відновлення i -ї частки
00.01	↓	
00.11		
+ 11.01	→	1 (Молодший розряд C)
100.00		

Відповідь: $C_{np}=11/01101$ (-13).

11 ДІЛЕННЯ ЧИСЕЛ З ФІКСОВАНОЮ КОМОЮ БЕЗ ВІДНОВЛЕННЯ ЗАЛИШКУ

11.1 Алгоритм ділення у без відновлення залишку

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

- визначити знак частки за формулою: $Sg_C = Sg_A \oplus Sg_B$;
- представити числа (операнди) у доповняльному коді в машинному зображенні, ділене (завжди), незалежно від його знака, береться в прямому коді з позитивним знаком, а дільник (завжди), незалежно від його знака, береться в доповняльному коді з негативним знаком;
- привласнити суматору значення $Sm := A^m_{доп}$; $PrB := B^m_{доп}$; $PrC := 0$;
- усунути дробову частину в дільнику, переносячи кому вправо на n розрядів (за аналогією з десятковою системою числення) і, щоб дріб не змінився, у діленому також перенести вправо кому на n розрядів;
- починаючи зі старших розрядів, до діленого додають дільник у доповняльному коді, що рівнозначно вирахуванню з діленого дільника й аналізують знак проміжного залишку:
 - 1) якщо знак проміжного залишку 00 (позитивний), то в регістр частки PrC записується 1, починаючи зі старшого розряду. Залишок зсувається на один розряд вліво (знакову крапку перенести вправо на один розряд), зноситься наступний розряд діленого, що не брав участі до цього в діленні. Після цього, проміжний залишок підготовлений до наступного додавання діленого у доповняльному коді;
 - 2) якщо знак проміжного залишку 11 (негативний), то в регістр частки PrC записується 0, починаючи зі старшого розряду. Залишок зсувається на один розряд вліво (знакову крапку перенести вправо на один розряд), зноситься наступний розряд діленого, що не брав участі до цього в діленні. Після цього, проміжний залишок підготовлений до наступного додавання до нього дільника в прямому коді зі знаком 00;
- дії попереднього пункту повторюються до одержання машинного нуля або заданої точності обчислення (кількість розрядів дробу після коми цілої частини числа). Кома дробу встановлюється в частці після зносу останнього розряду цілої частини діленого.

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

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

Рішення: виконуємо ділення методом без відновлення залишку.

- визначаємо знак частки C : $Sg_0C=0 \oplus 1=1$. У старші розряди регістра частки C заносимо значення негативного знака 11\;

- встановлюємо регістри PrA , PrB і Sm у нульовий стан, очистивши їх від попередньої інформації;

- переводимо десяткові числа в 2-ї, прямі і доповняльні коди:

$A_{np}=00\backslash10000,01$ (+16,25); $B_{np}=11\backslash11,01$ (-3,25); $B_{don}^M=11\backslash00,11$; $B_{np,vid}^M=00\backslash11,01$. Для вирахування використаємо $B_{don}^M=11\backslash00,11$. Для дільника в прямому коді зі знаком «+» беремо $B_{np}^M=00\backslash11,01$. (Щоб провести ділення з операндами A і B , як з цілими числами, перенесемо зап'яту на два розряди вліво в A і B).

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

Таблиця 11.1 – Приклад ділення чисел A на B за алгоритмом без відновлення залишку

$A_{np}^M=00\backslash1000001$	PrC	
$B_{don}^M=+11\backslash0011$		
$11\backslash1011$	$\longrightarrow$	0 (Знак 11)
$L \text{ зсув } 11\backslash0110$		
$+00\backslash1101$	$\longrightarrow$	1
$00\backslash0011$		
$L \text{ зсув } 00\backslash0110$		
$+11\backslash0011$	$\longrightarrow$	0
$11\backslash1001$		
$L \text{ зсув } 11\backslash0011$		
$+00\backslash1101$	$\longrightarrow$	1
$00\backslash0000$		
Машинний нуль.		
Відповідь: $C_{np}=11\backslash101$ (-5)		

11.2 Ділення чисел з рухомою комою

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

$$P_C = P_A - P_B. \quad (11.1)$$

При цьому можливі два випадки:

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

Ділення мантис робиться в такому ж порядку, як і у форматі з ФФЗ. При цьому, використовуються методи з відновленням і без відновлення залишку. Результату привласнюється порядок P_C .

Примітка. В алгоритмах ділення застосовується правило: на першому кроці підготовки операндів А і В до ділення робиться так, щоб $|A|$ завжди було менше $|B|$ (тимчасово змінюючи вимогу нормалізації операнда А і, відповідно, корегуючи значення порядку Р на +1) для того, щоб при першому кроці операції ділення не **було переповнення молодшого знакового розряду Sg_2** : порівняйте дії над нормалізованими двійковими операндами, наданими у ФРК: $A = +0,101 \cdot 2^{+1}$ і $B = +0,101 \cdot 2^{+1}$:

а) ділене $A_{пр}^M = 00\backslash,1010\backslash00\backslash01\backslash$; дільник $B_{пр}^M = 00\backslash,1010\backslash00\backslash01\backslash$; частка $C_{пр}^M = 01\backslash,0000\backslash00\backslash00\backslash$ — отримали ненормоване число і переповнення знакового розряду (тоді, згідно вимог g і δ , треба робити зсув **на першому же кроці алгоритму ділення**, що затримує процес у часі);

б) ділене $A_{пр}^M = 00\backslash,0101\backslash00\backslash10\backslash$; дільник $B_{пр}^M = 00\backslash,1010\backslash00\backslash01\backslash$; частка $C_{пр}^M = 00\backslash,1000\backslash00\backslash01\backslash$ —отримали нормоване число **С автоматично** без переповнення знакового розряду Sg_2 , бо вибрано $|A| < |B|$.

12 КОНТРОЛЬНЕ ЗАВДАННЯ

Після вивчення теоретичних основ курсу «Комп'ютерної арифметики», студент виконує самостійно свій варіант контрольного завдання, керуючись даними методичними вказівкам. Оформлена контрольна робота направляється викладачеві дисципліни ПТЦА кафедри КСМ не пізніше, чим за два тижні до дати модульного контролю або залікової сесії.

Для всіх варіантів у контрольному завданні означені наступні базові числа:

$$A = -35, 62; \quad B = 2, 5$$

Використовуючи базові числа, студент визначає числа свого варіанта, необхідні для виконання контрольного завдання.

Числа варіанта визначаються шляхом додавання до базових чисел записаного в останніх двох розрядах номера залікової книжки студента. Сума береться по абсолютній величині (без врахування знака).

Приклад. У заліковій книжці студента в останніх двох розрядах є число 38, тоді числа варіанта будуть:

$$A = 35, 62 + 38 = 73, 62; \quad B = 2, 5 + 38 = 40, 5.$$

Над числами варіанта необхідно зробити наступні операції:

- виконати перетворення десяткових чисел у двійкові всіма методами, викладеними в розділі 2 (2.1, 2.2, 2.3) і провести перевірку перетворення шляхом підстановки ваг розрядів і їх підсумовування;
- виконати перетворення за п.2.4;
- представити числа варіанта завдання у форматах з фіксованою комою (крапкою) і рухомою комою (крапкою) в модифікованому коді при 16-й сітці ПК (п.п. 3.1, 3.2);
- виконати перетворення чисел варіанта із прямого коду у обернений і знову в прямий (п. 4.2.1);
- виконати перетворення чисел варіанта із прямого коду у обернений, потім у доповняльний і знову в прямий (п. 4.2.2, п. 4.3.3);
- виконати операцію додавання бінарних чисел у прямому коді (п. 5.1, знаки чисел брати однаковими);
- виконати операцію додавання бінарних чисел у доповняльному і оберненому коді (п.п. 5.2, 5.3). Визначити абсолютну й відносну погрішності результатів;
- виконати операцію множення бінарних чисел (9.1, 9.2);

- виконати операцію ділення бінарних чисел у доповняльному коді методами: з відновленням (11.1) і без відновлення залишку (11.2). Арифметичні операції робити з точністю до 4-го знака після коми (крапки).

ЛІТЕРАТУРА

1. Коштoев В.В. Основы прикладной теории цифровых автоматов (учебное пособие) / В.В. Коштoев, К.К. Кипиани. – М: Наука, 1999. – 155 с.
2. Прикладная теория цифровых автоматов / [Самофалов К.Г., Росанкевич А.М, Валуйский В.Н., Каневский Ю.С., Пиневич М.М.] – К: Вища шк. Головное изд-во, 1987. – 375 с.
3. Проектирование цифровых вычислительных машин: учебное пособие / [С. А. Майоров, Г. И. Новиков, О. Ф. Немолочнов и др.]. – М. : Высшая школа, 1972. – 344 с.
4. Савельев А. Я. Арифметические и логические основы цифровых автоматов / А. Я. Савельев. – М. : Высшая школа, 1980. – 255 с.