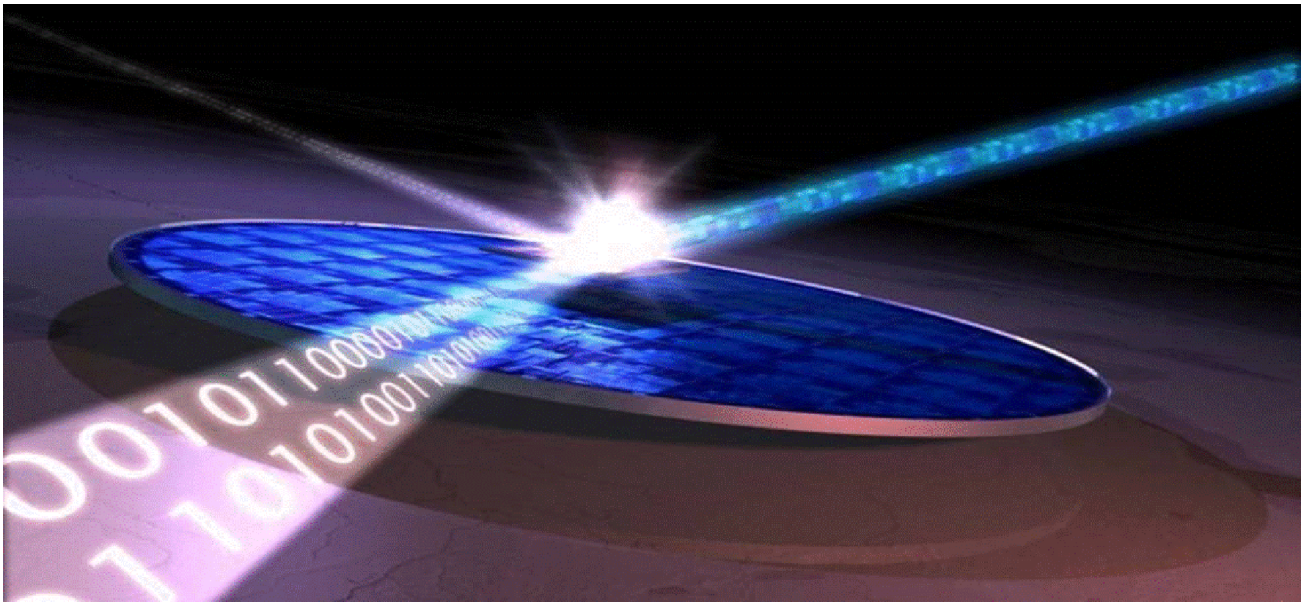
Р.Н. Квєтний, П.П. Повідайко,
М.М. Компанець, В.В. Гармаш,
Я.А. Кулик

АРИФМЕТИЧНІ ОСНОВИ ПРОЕКТУВАННЯ МІКРОПРОЦЕСОРНИХ СИСТЕМ



Міністерство освіти і науки України
Вінницький національний технічний університет

**Р.Н. Квєтний, П.П. Повідайко,
М.М. Компанець, В.В. Гармаш,
Я.А. Кулик**

АРИФМЕТИЧНІ ОСНОВИ ПРОЕКТУВАННЯ МІКРОПРОЦЕСОРНИХ СИСТЕМ

Навчальний посібник

Вінниця
ВНТУ
2017

УДК 004.312.44 (075)
ББК 32.971.32 - 043я73
А 81

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 3 від «21» листопада 2016 р.)

Рецензенти:
А.М. Петух, доктор технічних наук, професор
М.М. Биков, кандидат технічних наук, професор
Ю.П. Гульчак, кандидат технічних наук, доцент

Квєтний Р.Н.
А 81.. Арифметичні основи проектування мікропроцесорних систем: навчальний посібник / Р.Н. Квєтний, П.П. Повідайко, М.М. Компанець, В.В. Гармаш, Я.А. Кулик – Вінниця : ВНТУ, 2017. – 111 с.

В навчальному посібнику в простій формі викладені основні питання теорії систем числення: формування кодів та одержання різних за видом модифікацій цих кодів, контролю виконання логічних та арифметичних операцій, а це саме те, що застосовують та викликає найбільше запитань, розробники прикладного програмного забезпечення при проектуванні мікропроцесорних систем.

УДК 004.312.44 (075)
ББК 32.971.32 - 043я73

ЗМІСТ

ВСТУП

Всі стали помічати, що мікропроцесорна техніка стала використовуватись в реальних виробах. Очевидно, що сфери застосування МП стрімко розширюються. В любій галузі промисловості мікропроцесори вже здійснили свій вплив. Складається враження, що кожен день з`являються нові процесори, нові апаратні та програмні засоби для роботи з ними, нові пристрої до складу яких входять мікропроцесори, так як вони мають великі функціональні можливості та з успіхом можуть бути використані для побудови широкого класу засобів цифрової автоматики.

Мікропроцесори є невід`ємними компонентами сучасних пристроїв та систем автоматичного управління передусім тому, що їж висока надійність, відносно низька вартість, гнучкість при застосуванні, можливість організації складних алгоритмів управління досить простими засобами та створення систем з якісно новими обчислювальними можливостями та високими техніко-економічними показниками. Разом з тим, щоб розробити систему управління на базі мікропроцесора необхідно виконати цілу низку складних та разом з тим рутиних дій та операцій при проектувані, успішне виконання яких призводить до створення «інтелектуальних» систем, при роботі яких складається враження, що поруч працює «досвідчений спеціаліст», який миттєво фіксує будь-які зміни режимів роботи, контролює, інформує, реагує та управляє всім процесом з високою швидкодією та точністю при обробці даних, які представлені в різних за виглядом формах (системи числення) та виконуються арифметико-логічні перетворення та здійснює контроль над ними.

Навчальний посібник орієнтований на студентів, які вивчають комп’ютерні науки, зокрема системи числення, виконання арифметико-логічних операцій та їх контролю. Так, як при використанні мікропроцесорів основна увага приділяється при розробці не на апаратні засоби, а програмному забезпеченню: оброблюванню та перетворенню даних.

Навчальний посібник побудований слідуючим чином. Розділ 1 − є вступом в розгляд систем числення, наводяться основні визначення та класифікація та розглянуто позиційні, непозиційні, символічні та системи числення спеціального призначення. Наведені приклади переведення чисел із однієї позиційної системи числення в іншу.

У розділі 2 розглядаються правила виконання арифметичних операцій в пози-ційних системах числення. Спочатку приділено увагу вибору основи системи числення для застосування в системах числення. Далі розглянуто доповнення та віднімання чисел без знаку за допомогою доповнення.

Двійкова система числення та основні правила роботи із нею описана у розділі 3. В кінці розділу дається оцінювання точності представлення чисел в ци-фрових пристроях.

У розділі 4 розглянуті коди двійкових чисел: прямий, обернений, доповняльний.

У розділі 5 розглянуто виконання арифметичних операцій додавання та відні-мання в кодах (прямому та оберненому).

Двійково-десяткова арифметика, додавання в прямих, інверсних, зсув, множення та ділення чисел в Д-кодах, а також переведення чисел в Д-кодах розглядається у розділі 6.

У розділі 7 розглянуто питання контролю виконання логічних та арифметичних операцій.

Можливість розумного та гнучкого управління простими пристроями та складними системами тепер досягається за помірну платню, а це, в свою чергу, є закликом до усіх читачів відкрити цей навчальний посібник.

1 СИСТЕМИ ЧИСЛЕННЯ

1.1 Основні визначення. Класифікація систем числення

Системою числення називається сукупність цифрових знаків і правил їх з'єднання для позначення (запису) чисел. Тобто, в загальному випадку це спеціальна мова, алфавітом якої є символи, що називаються цифрами, а синтаксисом – правила, що дозволяють однозначно сформувати запис чисел.

Запис числа у будь-якій системі числення називають кодом числа. Стисло число може бути записано таким чином:

$$A = a_{n-1}...a_2a_1a_0. \qquad (1.1)$$

Окрему позицію у зображенні числа називають розрядом, а номер позиції – номером розряду. Число розрядів у записі числа називається розрядністю і співпадає з його довжиною.

Існує багато способів запису чисел цифровими знаками, тобто багато систем числення. В залежності від способу запису чисел і способу обчислення їх кількісного еквівалента системи числення можна класифікувати так (рис. 1.1).

У залежності від мети застосування використовуються різні системи. Наприклад, людиною для підрахунку і виконання дій над числами застосовується десяткова система числення, для обчислення часу – система обчислення часу, для нумерації – римська система числення тощо. Однак, будь-яка система числення, що призначена для практичного застосування, повинна забезпечити:

- можливість представлення будь-якого числа у заданому діапазоні чисел;
- однозначність представлення чисел;
- стислість і простоту запису чисел;
- простоту і зручність оперування числами в системі.

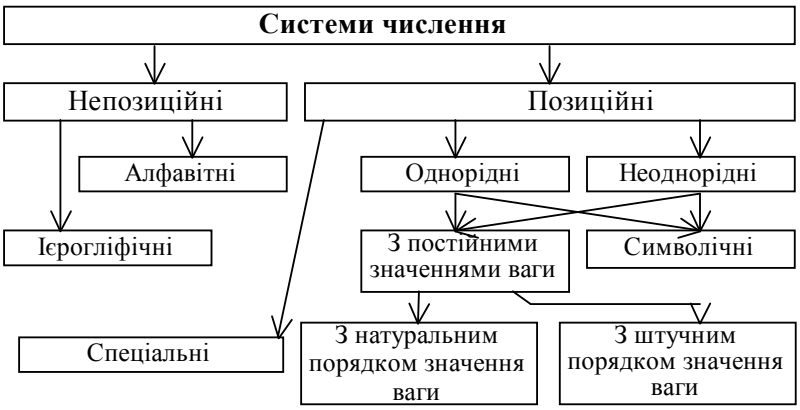


Рисунок. 1.1 – Класифікація систем числення

В основному системи числення будуються за таким принципом:

$$A(p) = a_1 p_1 + a_2 p_2 + ... + a_n p_n, \qquad (1.2)$$

де $A(p)$ – запис числа в системі із базисом p_i, a_i – база або послідовність цифр системи числення із p_i-ічним алфавітом, p_i – базис (основа) системи числення (сукупність значень ваг окремих розрядів системи).

База системи числення може бути додатною і змішаною. В першому випадку в ній як значення цифр використовується набір цифр 0, 1,..., ($p-1$). У другому випадку в системі на рівні з додатними цифрами є і від'ємні. Наприклад, для симетричної бази з нулем число додатних значень цифр дорівнює числу від'ємних. Значення цифр алфавіту в цьому випадку при $p = 2t + 1$ (тобто при непарній основі) становить такий ряд:

$$-t,\ (-t+1),\ ...\ -1,\ 0,\ 1,\ ...\ (t-1),\ t.$$

Системи числення зі змішаною базою можуть бути і при парній основі, але тоді можливо або застосування симетричних алфавітів без нуля (наприклад, при p = 2 можливий алфавіт +1, -1), або застосування алфавітів, у яких число від'ємних значень цифр не дорівнює числу додатних (наприклад, при p = 4 можливий такий алфавіт: -1, 0, 1, 2).

Основою системи числення називається кількість різних символів (цифр), що використовуються в кожному із розрядів числа для його зображення у даній системі числення.

Базис системи числення – це сукупність значень ваг окремих розрядів системи числення. Наприклад, базис десяткової системи являє собою таку послідовність: 1, 10, 10^2, ... 10^n.

Вага розряду R_i числа в будь-якій системі числення – це відношення $R_i = p_i / p_0$. Тому цифру a_i розряду з великим значенням i називають більш значущою, ніж цифру a_i розряду з меншим значенням i.

1.2 Непозиційні системи числення

Непозиційними називаються такі системи числення, алфавіт яких містить необмежену кількість символів (цифр), причому кількісний еквівалент будь-якої цифри постійний і залежить тільки від її зображення, але не від позиції у записі числа. Такі системи будуються за принципом адитивності, тобто кількісний еквівалент числа визначається, як сума рядом розташованих цифр:

$$K(A(Q)) = Q_1 + Q_2 + \ \ldots \ + Q_k = \sum_1^k Q_i, \qquad (1.3)$$

де Q_i – символи, що створюють базис системи.

Найбільш відомими представниками непозиційних систем числення є ієрогліфічні та алфавітні системи числення.

Ієрогліфічні системи числення – це такі системи числення, у яких кожна цифра представлена своїм символом, значком або ієрогліфом. Найбільш відомою з них є римська система числення, у якій застосовуються, наприклад, такі значення символів:

десяткова система:	1	5	10	0	100	500	1000
римська система	I	V	X	L	C	D	M

Значення записаного числа в римській системі визначається як сума послідовно записаних цифр, наприклад, XXV = 10 + 10 + 5 = $25_{(10)}$. Якщо зліва від цифри стоїть менша цифра, то значення останньої береться зі знаком мінус, наприклад, IX = X - I = $9_{(10)}$. Підкреслення запису числа зверху означає збільшення його у 1000 разів, наприклад, $\overline{XL} = 40000$. Тобто, тут існує відхилення від правила незалежності значення цифри від її розташування у числі.

Запис чисел в алфавітних системах будується за таким же принципом.

До основних недоліків непозиційних систем числення можна віднести:

- відсутність нуля;
- необхідність вмісту нескінченної кількості символів;
- складність виконання арифметичних дій над числами.

1.3 Позиційні системи числення

1.3.1 Загальні визначення

Позиційними називаються системи числення, алфавіт яких містить обмежену кількість символів, причому значення кожної цифри в числі визначається не тільки її зображенням, але і знаходиться в суворій залежності від місця її знаходження (позиції) у записі числа.

Наприклад, у десятковій системі числення число 555_{10} містить три однакові цифри, але значення кожної з них визначається її позицією, тобто $555 = 5 \cdot 10^2 + 5 \cdot 10^1 + 5 \cdot 10^0$.

Позиційні системи мають ряд переваг у порівнянні із непозиційними, основним з яких є зручність виконання арифметичних операцій.

У загальному вигляді число A в позиційній системі числення може бути представлено таким чином:

$$A = a_{n-1} p_{n-2} \ldots p_0 + a_{n-2} p_{n-3} \ldots p_0 + \ldots + a_1 p_0 + a_0, \qquad (1.4)$$

де a_i – цифра i-го розряду числа, причому, $a_i = \overline{0,\ p_j - 1}$ є базою системи числення, p_j – основа системи числення, $p_i = \prod_0^i p_j$ – вага i-го розряду числа.

Як видно з (1.4), такі системи будуються не тільки за принципом адитивності, але і за принципом мультиплікативності, тобто кількісний еквівалент числа визначається як сума рядом розташованих цифр зі своєю вагою.

Позиційні системи числення, в свою чергу, поділяються на ряд підкласів (рис. 1.1).

1.3.2 Однорідні позиційні системи числення

Однорідні позиційні системи числення є окремим випадком позиційних систем при $p_i = p_j$ для всіх i і j, тобто в них ваги окремих розрядів являють собою ряд членів геометричної прогресії зі знаменником p. Тому число в однорідних системах може бути представлено поліномом такого виду:

$$\begin{aligned} A = {} & a_{n-1} p^{n-1} + \ldots a_i p^{i1} + \ldots + a_1 p^1 + a_0 p^0 + \\ & + a_{-1} p^{-1} + \ldots + a_{-m} p^{-m} = \sum_{i=-m}^{n-1} a_i p^i \end{aligned} \qquad (1.5)$$

або у скороченій формі:

$$(A)_p = (a_{n-1} \ldots a_i \ldots a_1 a_0, a_{-1} \ldots a_{-m})_p = (C_A,\ V_A)_p = (C_A)_p, (V_A)_p,$$

де p-основа системи числення, $p \in \{2,3\ldots\}$, a_i – цифра i-го розряду, $a_i = \overline{0,1,2\ldots(p-1)}$, n – кількість цифр (розрядів) цілої частини числа, m – кількість цифр (розрядів) дробової частини числа.

Кома у (1.5) відділяє цілу частину C_A числа A від його дробової частини V_A.

Цифри a_{n-1} і a_{-m} мають спеціальні назви:

- a_{n-1} – цифра найбільшого значення (англ. Most Signifi-cant Digit

(MSD));

- a_{-m} – цифра найменшого значення (англ. Least Signifi-cant Digit (LSD)).

Очевидно, що основою однорідної позиційної системи може бути будь-яке ціле число, оскільки у визначенні позиційних систем числення не накладено ніяких обмежень на величину основи. Тому можлива нескінченна кількість однорідних позиційних систем числення, а назва системи визначає її основу: $p = 10$ – десяткова позиційна система числення, $p = 8$ – вісімкова позиційна система числення, $p = 2$ – двійкова позиційна система числення тощо.

За допомогою (1.5) можна отримати вирази для обчислення максимального значення числа A. Підставимо у вираз (1.5) значення $a_i = (p-1)$ і отримаємо:

$$A_{\max} = \sum_{i=-m}^{n-1}(p-1)p^i = (p-1)\sum_{i=-m}^{n-1}p^i = (p-1)\frac{p^{n-1}-p^{-m}}{p-1} = \\ = p^{n-1} - p^{-m}. \tag{1.6}$$

Із (1.6) видно, що значення мінімального числа, яке відрізняється від нуля, матиме такий вигляд:

$$A_{\min} = p^{-m}. \tag{1.7}$$

З урахуванням (1.6) і (1.7) можна визначити діапазон представлення числа A (максимальна кількість варіантів числа A):

$$D = \frac{A_{\max}}{A_{\min}} = p^{n+m-1} - 1. \tag{1.8}$$

Абсолютна похибка представлення числа A може бути визначена, як мінімальна різниця двох різних чисел:

$$\Delta = \min|A_1 - A_2| = p^{-m} \text{ для } A_1 \neq A_2. \tag{1.9}$$

В цифрових системах найбільше розповсюдження отримали такі однорідні позиційні системи числення:

- двійкова система числення (бінарна, англ. binary):

$$p = 2; \quad a_i = 0,1.$$

Загальна формула для запису числа у цій системі числення матиме вигляд:

$$[A]_2 = a_{n-1} \cdot 2^{n-1} + \quad ... \quad a_i \cdot 2^i + \quad ... \quad a_1 \cdot 2^1 + a_0 \cdot 2^0 +$$

$$+ a_{-1} \cdot 2^{-1} + \quad ... \quad + a_{-m} \cdot 2^{-m}.$$

Ваги розрядів, рухаючись вліво від коми, дорівнюють відповідно 1, 2, 4, 8, 16, ..., а вправо від коми – 1/2, 1/4, 1/8 ...

Наприклад, двійкове число $110010{,}101_2$ відповідає такому представленню у десятковій системі числення:

$$110010{,}101_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 +$$

$$+ 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^3 = 50{,}625_{10}.$$

- вісімкова система числення (октавна, англ. octal):

$$p = 8; \quad a_i = 0,\ 1,\ 2,\ ...\ 7.$$

Загальна формула для запису числа у цій системі числення матиме вигляд:

$$[A]_8 = a_{n-1} \cdot 8^{n-1} + \quad ... \quad a_i \cdot 8^i + \quad ... \quad a_1 \cdot 8^1 + a_0 \cdot 8^0 +$$

$$+ a_{-1} \cdot 8^{-1} + \quad ... \quad + a_{-m} \cdot 8^{-m}.$$

Ваги розрядів, рухаючись вліво від коми, дорівнюють відповідно 1, 8, 64, 256, ..., а вправо від коми – 1/8, 1/64, 1/256 ... .

Наприклад, вісімкове число $317{,}25_8$ відповідає такому представленню у десятковій системі числення:

$$317{,}25_8 = 3 \cdot 8^2 + 1 \cdot 8^1 + 7 \cdot 8^0 + 2 \cdot 8^{-1} + 5 \cdot 8^{-2} = \left(207\frac{21}{64}\right)_{10}.$$

- десяткова система числення (десяткова, англ. decimal):

$$p = 10; \quad a_i = 0,\ 1,\ 2,\ ...\ 9.$$

Загальна формула для запису числа у цій системі числення буде така:

$$[A]_{10} = a_{n-1} \cdot 10^{n-1} + \quad ... \quad a_i \cdot 10^i + \quad ... \quad a_1 \cdot 10^1 + a_0 \cdot 10^0 +$$

$$+ a_{-1} \cdot 10^{-1} + \quad ... \quad + a_{-m} \cdot 10^{-m}.$$

Ваги розрядів, рухаючись вліво від коми, дорівнюють відповідно 1, 10, 100, ..., а вправо від коми – 1/10, 1/10, ...

- шістнадцяткова система числення (гексагональна, англ. hexadecimal):

$$p = 16; \quad a_i = 0,\ 1,\ 2,\ ...\ 9,\ A,\ B,\ C,\ D,\ E,\ F.$$

Загальна формула для запису числа, при цьому, матиме вигляд:

$$[A]_{16} = a_{n-1} \cdot 16^{n-1} + \quad ... \quad a_i \cdot 16^i + \quad ... \quad a_1 \cdot 16^1 + a_0 \cdot 16^0 +$$

$$+ a_{-1} \cdot 16^{-1} + \quad ... \quad + a_{-m} \cdot 16^{-m}.$$

Ваги розрядів, рухаючись вліво від коми, дорівнюють відповідно 1, 16, 256, ..., а вправо від коми – 1/6, 1/256 ... .

Наприклад, шістнадцяткове число $A7B{,}C8_{16}$ відповідає такому представленню у десятковій системі числення:

$$A7B,C8_{16} = 10\cdot 16^2 + 7\cdot 16^1 + 11\cdot 16^0 + 12\cdot 16^{-1} + 8\cdot 16^{-2} =$$
$$= \left(2683\frac{25}{32}\right)_{16}.$$

Із розглянутого видно, що вага *i*-го розряду R_i числа у позиційній системі числення являє собою відношення $R_i = p_i / p_0$. Якщо розряд має вагу, наприклад, $R_i = 10^m$, то наступний старший розряд має вагу $R_{i+1} = 10^{m+1}$, а молодший $R_{i-1} = 10^{m-1}$. Такий взаємозв'язок розрядів вимагає передачі інформації між ними при виконанні арифметичних операцій (перенесень при додаванні та позик при відніманні).

В таблиці 1.1 наведені представлення шістнадцяти перших десяткових цифр у вказаних системах числення.

Таблиця 1.1 – Представлення десяткових цифр у різних системах числення

Системи числення			
$p = 10$	$p = 2$	$p = 8$	$p = 16$
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	*A*
11	1011	13	*B*
12	1100	14	*C*
13	1101	15	*D*
14	1110	16	*E*
15	1111	17	*F*

1.3.3 Неоднорідні позиційні системи числення

У неоднорідних позиційних системах числення p_j не залежить один від одного і може набувати будь-яких значень (ці системи ще називають системами зі змішаною основою).

У неоднорідних системах числення в кожному *i*-му розряді кількість допустимих символів може бути різна, при цьому $0 \le a_i < p_{i-1}$, де p_{i-1} - основа системи числення в *i*-му розряді. Запис цілого числа в таких системах проводиться у відповідності із (1.1).

Прикладом неоднорідної позиційної системи числення може бути система обчислення часу, для якої p_0 = 1 с.; p_1 = 60 с., p_2 = 60 хв., p_3 = 24 год., p_4 = 365 діб.

Наприклад, час у 3 роки, 16 діб, 12 годин, 27 хвилин, 31 секунду в одиницях молодшого розряду, визначиться згідно з (1.2) таким чином:

$$T = 3 \cdot 365 \cdot 24 \cdot 60 \cdot 1 + 16 \cdot 24 \cdot 60 \cdot 1 + 12 \cdot 60 \cdot 1 + \; + \; 27 \cdot 60 \cdot 1 + \; + 31 \cdot 1 \text{ (секунд)}.$$

Спеціально для застосування в ЕОМ була створена неоднорідна двійково-п’ятіркова система числення, у якій в непарних розрядах основа p_1 = 5 ($a_i = \overline{0,4}$), а в парних розрядах основа p_2 = 2 ($a_i = 0,1$). Оскільки добуток двох сусідніх (парного і непарного) розрядів дорівнює 10, то двома двійково-п’ятірковими розрядами можна кодувати одну десяткову цифру (табл. 1.2).

Таблиця 1.2 □ Кодування однієї десяткової цифри

a_{10}	a_{2-5}	a_{10}	a_{2-5}
0	00	5	10
1	01	6	11
2	02	7	12
3	03	8	13
4	04	9	14

Наприклад, число 297_{10} у двійково-п’ятірковій системі числення буде представлено так: $A = 297_{10} = (02 \cdot 14 \cdot 12)_{2\text{-}5}$. Тут n = 6, основи p_1 = 5; p_2 = 2; p_3 = 5; p_4 = 2; p_5 = 5; p_6 = 2, а цифри a_1 = 2; a_2 = 1; a_3 = 4; a_4 = 1; a_5 = 2; a_6 = 0. Для обчислення кількісного еквівалента числа $A_{2\text{-}5,}$ підставимо ці значення в (1.4):

$$A = 0 \cdot 5 \cdot 2 \cdot 5 \cdot 2 \cdot 5 \; + 2 \cdot 2 \cdot 5 \cdot 2 \cdot 5 + 1 \cdot 5 \cdot 2 \cdot 5 + 4 \cdot 2 \cdot 5 + \; + 1 \cdot 5 + 2 = \; 0 + 200 + 50 + 40 + 5 + 2 = 297_{10}.$$

1.3.4 Кодовані позиційні системи числення

В кодованих позиційних системах числення цифри однієї системи числення кодуються за допомогою цифр іншої системи числення. Число в загальному вигляді записується таким чином:

$$\begin{aligned} A = {} & (a_{n-1,k} p^{n-1} + a_{n-2,k} p^{n-2} + \ldots + a_{1,k} p^1 + a_{0,k} p^0) P^k + \\ & + (a_{n-1,k-1} p^{n-1} + \ldots + a_{1,k-1} p^1 + a_{0,k-1} p^0) P^{k-1} + \ldots \\ & + (a_{n-1,0} p^n + \ldots + a_{1,0} p^1 + a_{0,0} p^0) P^0, \end{aligned} \qquad (1.10)$$

де p – основа системи числення, символами якої кодуються числа, P – основа заданої системи числення.

При побудові кодованих позиційних систем як ваги розрядів можуть бути вибрані члени геометричної прогресії (тобто ваги однорідної позиційної системи числення), а також довільні числа. У першому випадку системи називаються кодованими системами числення з природною вагою розрядів, у другому – кодованими системами числення зі штучною вагою розрядів.

Прикладом системи числення з природною вагою розрядів може бути двійково-десяткова система (англ. Dinary Coded Decimal - BCD) з вагами 8-4-2-1. У ній кожна цифра десяткового числа кодується двійковою тетрадою (чотирма двійковими розрядами). Наприклад, десяткове число 7 у двійково-десятковій системі числення матиме такий вигляд:

$$7 = 0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1.$$

Прикладом системи числення зі штучною вагою розрядів може бути двійково-десяткова система числення з вагами 2-4-2-1. Десяткове число 7 в цій системі матиме, відповідно,

такий вигляд:

$$7 = 1 \cdot 2 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1.$$

Як відомо, цей код є самодоповнюючимся, тобто в такій системі кодування десяткове число 9 кодується як 1111, що дозволяє найбільш раціонально будувати десяткові лічильники та арифметичні схеми.

Код називається самодоповнюючимся, якщо двійкові коди будь-яких двох десяткових цифр, доповнюючи одна одну до 9 (тобто якщо їх десяткова сума дорівнює 9: $a'_{10} + a''_{10} = 9$), доповнюють один одного до $15_{10} = 1111_2$ (тобто їх сума $a'_2 + a''_2 = 1111$).

Ваги розрядів у коді можуть бути не тільки додатними, але і від'ємними. Наприклад, у коді 8-4-(-2)-(-1) число 7 буде записано так:

$$7 = 1 \cdot 8 + 0 \cdot 4 + 0 \cdot (-2) + 1 \cdot (-1).$$

Ще одним самодоповнюючимся кодом є код 8421 плюс 3 (код з надлишком три, англ. EXCESS-3). Він отримується з природного коду 8-4-2-1 додаванням до нього числа $3_{10} = 0011_2$ за такою формулою:

$$D = 8 \cdot a_4 + 4 \cdot a_2 + 2 \cdot a_2 + 1 \cdot a_1 + 3.$$

Внаслідок такого додавання отримується система з непостійною вагою розрядів. Дійсно, наприклад, цифра 1_{10} кодується як 0100_2, тобто в третьому двійковому розряді 1 має вагу одиниці. Тоді при кодуванні цифри $9_{10} = 1100_2$ старша двійкова цифра повинна мати вагу 8, тоді як при кодуванні цифри $5_{10} = 1000_2$ вона має вагу 5.

Відмітимо, що усі наведені коди є системами однорідними, оскільки в кожному двійковому розряді може бути дві цифри: 0 або 1.

В таблиці 1.3 наведені десяткові цифри, представлені в деяких двійково-десяткових кодах.

Необхідно підкреслитити, що двійково-десяткові коди мають певну надмірність, оскільки для кодування десяткових цифр використовується тільки 10 комбінацій із 16 можливих комбінацій, що може бути використано для виявлення деяких видів похибок.

Таблиця 1.3 – Десяткові цифрии в двійково-десяткових кодах

Десяткова цифра	Коди BCD					
	8421	2421	84-2-1	5421	7421	8421+3
0	0000	0000	0000	0000	0000	0011
1	0001	0001	0111	0001	0001	0100
2	0010	0010	0110	0010	0010	0101
3	0011	0011	0101	0011	0011	0110
4	0100	0100	0100	0100	0100	0111
5	0101	1011	0100	1000	0101	1000
6	0110	1100	1010	1001	0110	1001
7	0111	1101	1001	1010	1000	1010
8	1000	1110	1000	1011	1001	1011
9	1001	1111	1111	1100	1010	1100

У загальному вигляді десяткове число A, задане своїми (n + m) розрядами

$$A = \sum_{i=-m}^{n} a_i 10^i$$

може бути представлено у двійково-десятковому коді таким виразом:

$$A = \sum_{i=-m}^{n} T_i 10^i, \qquad (1.11)$$

або виразом

$$A = {}^{A}T_{n-1}\,{}^{A}T_{n-2}\ \ldots\ {}^{A}T_1\,{}^{A}T_0\,{}^{A}T_{-1}\,{}^{A}T_{-2}\ldots\,{}^{A}T_{-m}, \qquad (1.12)$$

де $T_i = (a_{i3}a_{i2}a_{i1}a_{i0})$ – i-та тетрада,

$i = (n-1),\ \ldots\ 1,0,-1,-2,\ \ldots\ -m$, $a_{ij} = 0,1;\ \ j = 0,1,2,3.$

Наприклад, десяткове число $A = 197_{10}$ може бути записано у вказаних кодах таким чином:

- - у коді 8421 $A_{8421} = \underbrace{0001}_{1}\ \underbrace{1001}_{9}\ \underbrace{0111}_{7},$
- - у коді 2421 $A_{2421} = 0001\ 1111\ 1101,$
- - у коді 84-2-1 $A_{84-2-1} = 0111\ 1111\ 1001,$
- - у коді 5421 $A_{5421} = 0001\ 1100\ 1010,$

- у коді 8421+3 $A_{8421+3} = 0100\ 1100\ 1010.$

Для кодування десяткових цифр можуть також бути використані коди з числом розрядів, більшим за чотири (табл. 1.4).

Таблиця 1.4 – Коди з числом розрядів, більшими за 4

Десяткова цифра	Код				
	2 із 5		2 із 7		1 із 10
	I	II	I	II	
0	00101	11000	0100001	0000101	0000000001
1	00101	00011	0100010	0000110	0000000010
2	00110	00101	0100100	0001001	0000000100
3	01001	00110	0101000	0001010	0000001000
4	01010	01001	0110000	0010001	0000010000
5	01100	01010	1000001	0010010	0000100000
6	10001	01100	1000100	0100001	0001000000
7	10010	10001	1000100	0100010	0010000000
8	10100	10010	1001000	1000001	0100000000
9	11000	10100	1010000	1000010	1000000000

1.4 Системи числення спеціального призначення

Перевагою позиційних систем числення, що спеціально створені для спрощення або прискорення обчислень в ЕОМ, є простота алгоритмів виконання деяких арифметичних операцій, а недоліком – необхідність переведення із класичних позиційних систем у спеціальні. Їх застосовують для реалізації деяких обчислювальних процесів, в яких не потрібно змінювати систему числення при введенні і виведенні, або ця зміна досягається простими засобами.

Наприклад, позиційні системи числення з від'ємною основою дозволяють представити без використання знаку будь-яке дійсне число, додатне або від'ємне. Однією із самих цікавих є врівноважена трійкова система числення, тобто система числення з основою 3 із цифрами +1, 0, -1 (або $\bar{1}$). Приклади запису чисел у цій системі можуть бути такі:

$$10\bar{1}_3 = 1\cdot 3^2 + 0\cdot 3^1 + (-1)\cdot 3^0 = +8_{10};$$

$$1\bar{1}\bar{1}0,1\bar{1} = 1\cdot 3^3 + (-1)\cdot 3^2 + (-1)\cdot 3^1 + 0\cdot 3^0 + 1\cdot 3^{-1} + (-1)\cdot 3^{-2} = \left(15,\frac{2}{9}\right)_{10};$$

$$\bar{1}1\,10,\bar{1}1 = (-1)\cdot 3^3 + 1\cdot 3^2 + 1\cdot 3^1 + 0\cdot 3^0 + (-1)\cdot 3^{-1} + 1\cdot 3^{-2} = -\left(15,\frac{2}{9}\right)_{10}.$$

Перевагами врівноваженої трійкової системи можна вважати такі:

- знак числа задається його найбільшою значущою (старшою) ненульовою цифрою;
- перехід до числа з протилежним знаком проводиться заміною усіх 1

на $\bar{1}$ і навпаки;

- операція округлення до найближчого цілого зводиться до відкидання дробової частини.

Виконувати операції додавання у цій системі дуже просто, якщо врахувати, що:

$$\bar{1}+\bar{1}=\bar{1}1,\ 1+\bar{1}=0,\ 1+1=1\bar{1}.$$

Операція віднімання проводиться переходом до числа, протилежного по знаку, і подальшого використання операції додавання. Правила множення на +1 звичайні, а при множенні на $\bar{1}$ знак часткового добутку змінюється на протилежний.

Приклад. Помножити числа $A = 19_{10}$ і $B = 33_{10}$

$$A = 1\bar{1}01 = 19_{10}$$
$$B = 11\bar{1}0 = 33_{10}$$
$$\begin{array}{r} 0000 \\ \bar{1}101 \\ 1\bar{1}01 \\ 1\bar{1}01 \\ \hline 10\bar{1}\bar{1}1\bar{1}0 \end{array} = 627_{10}.$$

1.5 Позиційні системи числення з непостійною вагою розрядів

Можливе використання позиційних систем числення з непостійною вагою розрядів. На практиці такі системи числення використовуються як спеціальні коди, наприклад код Грея.

У двійковому коді при переході від зображення одного числа до зображення наступного може відбуватися одночасна зміна цифр у декількох розрядах, що може бути джерелом похибок в роботі апаратури у деяких випадках (наприклад, при переході від коду цифри 7 до коду 8).

У коді Грея сусідні числа відрізняються цифрою тільки в одному розряді (табл. 1.5).

Двійкові розряди в коді Грея не мають постійної ваги. Наприклад, для числа $3_{10,}$ представленого у коді Грея, одиниця другого розряду має вагу, що дорівнює трьом, а для числа 7_{10} одиниця третього розряду має вагу, що дорівнює семи.

Код Грея має ряд характерних особливостей:

- при послідовному переході від одного числа до наступного числа в ньому немає одночасної зміни цифр в декількох розрядах;
- зміна значень кожного розряду при послідовному переході від комбінації до комбінації відбувається вдвічі рідше, ніж у звичайному двійковому коді (в молодшому розряді двійкового коду відбувається чергування елементів 0-1-0-1- .., у другому розряді – 00-11-00-11..., в четвертому –0000-1111-0000..., а в коді Грея для молодшого розряду відбувається чергування елементів 00-11-00-11..., для другого – 0000-

1111-0000... тощо;

- у коді Грея можна виділити осі симетрії, відносно яких спостерігається ідентичність стану деяких розрядів. Головна вісь симетрії розташована між кодами $(2^{n-1}-1)$ і 2^{n-1} (звідси походить: назва відображений або рефлексний код).

Таблиця 1.5 – Код Грея

Десяткова система числення	Двійкова система числення	Код Грея	Десяткова система числення	Двійкова система числення	Код Грея
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

Недоліки коду Грея та інших позиційних систем числення з непостійною вагою розрядів зумовлені тим, що в них вага символу “1” не визначається номером розряду. Через це їх важко застосувати для обробки інформації в ЕОМ. Тому перед введенням в ЕОМ або декодування даних, представлених в позиційних системах числення з непостійною вагою розрядів, їх перетворюють у більш простий і зручний двійковий код.

1.6 Символічні системи числення

Позиційні системи числення мають чіткі і певні міжрозрядні зв’язки. Цю їх властивість можна розглядати як перевагу, оскільки вона забезпечує простоту виконання арифметичних операцій, і, разом з тим, як недолік, оскільки ця властивість через наявність міжрозрядних перенесень призводить до обмеження технічної швидкості виконання арифметичних операцій. Тому розробка непозиційних систем числення, в яких відсутні міжрозрядні зв’язки і просто здійснюються арифметичні операції, дозволила б підвищити швидкодію пристроїв для виконання арифметичних операцій.

У символічних системах, на відміну від позиційних, цифри є символами, кожен із яких окремо ніяк не характеризує будь-яке число. Певним комбінаціям цифр умовно поставлені у відповідність певні числа. Прикладами символічних систем числення є знакологарифмічна система числення і система представлення чисел у залишках або система залишкових класів (СЗК).

Головна відмінність СЗК від розглянутих вище систем полягає у тому, що в ній використовується не одна, а декілька основ:

$$b_i = (b_1, b_2, b_3 \ldots b_n),$$

причому *i*-та цифра a_i будь-якого числа A являє собою остачу від ділення заданого числа на *i*-ту основу b_i:

$$a_i = A - \left[\frac{A}{b_i}\right] \cdot b_i \qquad (1.13)$$

де $\left[\frac{A}{b_i}\right]$ – ціла частина дробу A/b_i.

Якщо використовується одна основа b, то за допомогою остач від ділення на неї однозначно можна відобразити тільки числа в діапазоні від 0 до $(b-1)$. У цьому випадку СЗК не відрізняється від звичайної позиційної системи числення з основою b. Якщо взяти дві основи (b_1, b_2), то за допомогою остач однозначно можна відобразити числа в діапазоні від 0 до $(b_1 b_2 - 1)$. При збільшенні кількості і (або) абсолютної величини основ можна збільшити діапазон представлення чисел до необхідної величини N:

$$N \le b_1 b_2 b_3 \ldots b_{n-1} b_n - 1. \qquad (1.14)$$

При цьому накладається умова, щоб усі b_i були простими числами.

У таблиці 1.6 на основі виразу (1.13) наведені дворозрядні числа для п'ятнадцяти десяткових чисел, представлених в СЗК з вагою розрядів, відповідно $b_1 = 3$ і $b_2 = 5$.

Таблиця 1.6 – Дворозрядні числа

Десяткова цифра	СЗК		Десяткова цифра	СЗК	
	a_1 $(b_1 = 3)$	a_2 $(b_2 = 5)$		a_1 $(b_1 = 3)$	a_2 $(b_2 = 5)$
0	0	0	8	2	3
1	1	1	9	0	4
2	2	2	10	1	0
3	0	3	11	2	1
4	1	4	12	0	2
5	2	0	13	1	3
6	0	1	14	2	4
7	1	2			

Із аналізу табл. 1.6 видно, що в діапазоні від 0 до 14 ні одне сполучення остач не повторюється, тобто такі сполучення остач можуть бути використані для зображення (кодування) усіх чисел в заданому виразом (1.14) діапазоні.

Із сказаного зрозуміло, що на відміну від позиційних систем числення в СЗК немає міжрозрядних зв'язків, тобто цифри у кожному розряді формуються незалежно одна від іншої.

Таким чином, усі арифметичні дії (за виключенням ділення) в СЗК є порозрядними і виконуються за простими правилами арифметики, що дозволяє спростити апаратуру для виконання операцій і збільшити її швидкодію.

1.7 Переведення чисел із однієї позиційної системи числення в іншу

1.7.1 Методи переведення

Існує два основних методи переведення числа з однієї системи числення в іншу: табличний і розрахунковий.

Табличний метод прямого переведення базується на співставленні таблиць відповідності чисел різних систем числення. Цей метод дуже громіздкий і вимагає великого об'єму пам'яті для збереження таблиць, але може застосовуватись для будь-яких систем числення (не тільки для позиційних). Суть іншого виду табличного методу полягає в тому, що є таблиці еквівалентів у кожній системі тільки для цифр, тобто баз цих систем і ступенів основи (додатних і від'ємних), тобто базису систем. Задача переведення зводиться до того, що у загальний вираз поліномів (1.4) чи (1.5) для початкової системи числення підставити еквіваленти з нової системи для всіх цифр і їх ваг розрядів і виконати дії (множення і додавання) за правилами арифметики при новій основі. Отриманий при цьому результат буде зображати число в новій системі числення.

Приклад. Число $A_{10} = 127$ перевести в систему числення з основою $p = 2$. Еквіваленти для двійкової і десяткової систем числення представлені в табл. 1.7.

Таблиця 1.7 – Еквіваленти для двійкової і десяткової систем числення

Десяткове число	Двійкове число
10^0	0001
10^1	1010
10^2	1100100

$$\text{Тоді } A_{10} = 127 = 1\cdot10^2 + 2\cdot10^1 + 7\cdot10^0 = 0001\cdot1100100 + 0010\cdot1010 + 0111\cdot0001 = 1111111_2.$$

Цей метод застосовується тільки для позиційних систем числення.

Розрахунковий метод застосується тільки для однорідних позиційних систем числення.

1.7.2 Переведення цілих чисел з однієї позиційної системи числення в іншу

Нехай задана ціла частина $(C)_p$ числа A в довільній позиційній системі числення з основою *p:*

$$(C)_p = (a_{n-1}a_{n-2} \ \dots \ a_i \ \dots \ a_1a_0)_p,$$

і його необхідно перевести в нову систему числення з основою *r*, тобто звести до такого вигляду:

$$\begin{aligned}(C)_r &= (a'_{k-1}a'_{k-2} \ \dots \ a'_i \ \dots \ a'_1a'_0)_r = \\ &= a'_{k-1}r^{k-1} + a'_{k-2}r^{k-2} + \dots + a'_ir^i + \dots + a'_1r^1 + a'_0r^0.\end{aligned} \tag{1.15}$$

Цифри a'_i представляють собою остачу від ділення на нову основу r, задану в позиційній системі числення з основою *p,* цілої частини $(C)_p$ числа A і наступних цілих частин частки I_i аж до отримання цілої частини від ділення $I_{k-1} = 0$ згідно з такою схемою:

$$\begin{aligned}\frac{(C)_p}{r} &= I_0 + \frac{a'_o}{r}; \\ \frac{I_0}{r} &= I_1 + \frac{a'_1}{r}; \\ \frac{I_i}{r} &= I_{i+1} + \frac{a'_{i+1}}{r}; \\ \frac{I_{k-3}}{r} &= I_{k-2} + \frac{a'_{k-2}}{r}; \\ \frac{I_{k-2}}{r} &= 0 + \frac{a'_{k-1}}{r}.\end{aligned} \tag{1.16}$$

Доведення такого алгоритму очевидне. Із (1.16) можна записати:

$$(C)_p = rI_0 + a'_0.$$

Із врахуванням того, що

$$I_i = rI_{i+1} + a'_{i+1}$$

для $i = 0,1,2, \ \dots \ (k-2)$, можна отримати:

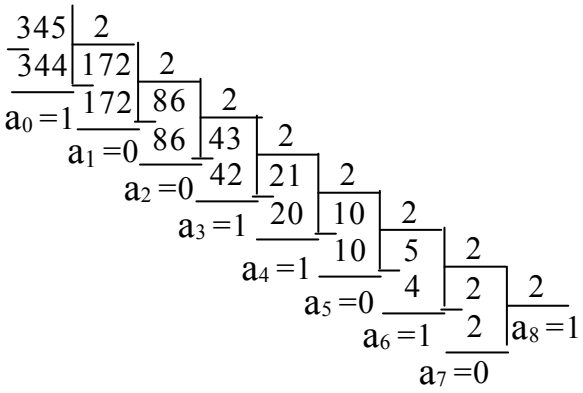
$$\begin{aligned}(C)_p &= rI_0 + a'_0 = I_1r^2 + a'_1r + a'_0 = I_2r^3 + a'_2r^2 + a'_1r + a'_0 = \ \dots \\ \dots &= a'_{k-1}r^{k-1} + a'_{k-2}r^{k-2} + \ \dots \ + a'_ir^i + \ \dots \ + a'_1r^1 + a'_0r^0 = (C)_r.\end{aligned}$$

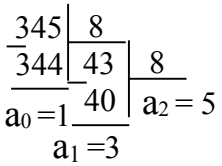
Тобто остачі від ділення згідно з алгоритмом (1.16) єцифрами представлення цілої частини $(C)_p$ числа A у позиційній системі числення з основою r, що і необхідно було довести.

Правило переведення цілих чисел із однієї позиційної системи числення в іншу формулюється таким чином: щоб перевести ціле число з однієї позиційної системи числення в іншу, необхідно початкове число послідовно ділити на основу нової системи числення, записану в початковій системі числення, до отримання частки, яка дорівнює нулю. Число у новій системі числення записується із остач від ділення, починаючи з останньої.

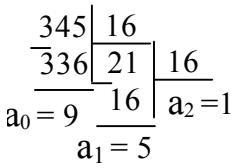
Приклад. Перевести десяткове число C_{10} = 207 у двійкову, вісімкову і шістнадцяткову системи числення.

Виконаємо дії згідно з наведеним вище алгоритмом і отримаємо:





Тобто $207_{10} = 101011001_2$.



…… $207_{10} = 531_8$

…… $207_{10} = 159_{16}$.

1.7.3 Переведення правильних дробів з однієї позиційної системи числення в іншу

Нехай задана дробова частина $(U)_p$ числа A (правильний дріб) у довільній позиційній системі числення з основою p:

$$(U)_p = (0, a_{-1}a_{-2} \ \dots \ a_{-i} \ \dots \ a_{-m})_p$$

і її необхідно перевести в нову систему числення з основою r, тобто звести до такого вигляду:

$$(U)_r = (0, a'_{-1}a'_{-2} \ \dots \ a'_{-i} \ \dots \ a'_{-s})_r + \delta_s =$$
$$= a'_{-1}r^{-1} + a'_{-2}r^{-2} + \dots + a'_{-i}r^{-i} + \dots + a'_{-s}r^{-s} + \delta_s, \qquad (1.17)$$

де $\delta_s = \varepsilon \cdot r^{-s}$ – похибка округлення ($\varepsilon < 1$).

Цифри a'_{-i} представляють собою цілу частину, яка утворюється при множенні на нову основу r, задану в позиційній системі числення з

основою *p,* дробової частини $(U)_p$ числа A і наступних дробових частин I_i аж до отримання заданої кількості цифр із врахуванням необхідної точності згідно з наступною схемою:

$$\begin{aligned}
(U)_p r &= I_1 + a'_{-1};\\
I_1 r &= I_2 + a'_{-2};\\
&\dots\\
I_i r &= I_{i+1} + a'_{-i-1};\\
I_{s-2} r &= I_{s-1} + a'_{-s+1};\\
I_{s-1} r &= I_s + a'_{-s} = \varepsilon + a'_{-s}.
\end{aligned} \qquad (1.18)$$

Доведення правильності такого алгоритму теж очевидне. Із (1.18) отримаємо:

$$(U)_p = a'_{-1} r^{-1} + I_1 r^{-1}.$$

З урахуванням того, що

$$I_i = I_{i+1} r^{-1} + a'_{-i-1} r^{-1}$$

для $i = 1,2, \dots (s-1)$ можна отримати

$$(U)_p = a'_{-1} r^{-1} + I_1 r^{-1} = a'_{-1} r^{-1} + a'_{-2} r^{-2} + I_2 r^{-2} = \dots$$

$$\dots = a'_{-1} r^{-1} + a'_{-2} r^{-2} + \dots + a'_{-i} r^{-i} + \dots + a'_{=s} r^{-s} + \varepsilon r^{-s} = (U)_r.$$

Тобто цілі частини від множення згідно з алгоритмом (1.18) є цифрами представлення дробової частини $(U)_p$ числа A у позиційній системі числення з основою r, що і необхідно було довести.

Правило переведення правильних дробів із однієї позиційної системи числення в іншу формулюється таким чином: щоб перевести правильний дріб із однієї позиційної системи числення в іншу, необхідно початкове число послідовно множити на основу нової системи числення, записану в початковій системі числення. Число у новій системі числення записується із цілих частин від множення, починаючи із першої. Процедуру необхідно продовжувати до отримання заданої точності.

Як видно з останнього виразу, на відміну від цілих чисел, точне переведення можливе не для всіх правильних дробів. Похибка при переведенні визначається одиницею молодшого розряду запису числа в системі числення з основою r.

Приклад. Перевести правильний дріб $U_{10} = 0{,}357$ із десяткової системи числення у двійкову, вісімкову і шістнадцяткову системи числення.

Процедура переведення згідно із розглянутим алгоритмом представлена у табл. 1.8.

Таблиця 1.8 □ Основа системи числення

$r = 2$		$r = 8$		$r = 16$	
Ціла частина	Дробова частина	Ціла частина	Дробова частина	Ціла частина	Дробова частина
0	357	0	357	0	357
	$\times 2$		$\times 8$		$\times 16$
0	714	2	856	5	712
	$\times 2$		$\times 8$		$\times 16$
1	428	6	848	4	272
	$\times 2$		$\times 8$		$\times 16$
0	856	6	784	4	352
	$\times 2$		$\times 8$		
1	712	6	272		
	$\times 2$				
1	424				
	$\times 2$				
0	848				
	$\times 2$				
1	696				
	$\times 2$				
1	392				
	$\times 2$				
0	78				
	$\times 2$				
1	568				

Із врахуванням результатів табл. 1.8, отримаємо:
$0{,}357_{10} = 0{,}0101101101_2 = 0{,}2666_8 = 0{,}544_{16}$.

1.7.4 Переведення неправильних дробів

При переведенні неправильних дробів необхідно окремо перевести цілу і дробову частини числа за вищеозначеними правилами, записати їх у новій системі числення, залишивши незмінним розташування коми.

У тих випадках, коли бажана одноманітність дій, що необхідні для переведення, задане число A спочатку ділять або на r^n (n - ціле додатне) так, щоб $A/r^n < 1$, або множать на r^k (k - число необхідних розрядів дробової частини числа A, представленого в новій системі числення з основою r) і округляють Ar^k до найближчого цілого числа. Потім отриманий дріб або ціле число переводять в систему числення з основою r згідно з алгоритмами (1.16) або (1.18). Для збереження кількісного еквівалента отриманий результат необхідно помножити або розділити відповідно на r^n або r^k. Практично це означає перенесення коми на n

розрядів праворуч в першому випадку і на k розрядів ліворуч у другому випадку.

1.7.5. Переведення чисел із систем числення з кратними основами

В цифровій техніці широке застосування знаходять вісімкова і шістнадцяткова системи числення, основи яких є степенями числа 2 (основи двійкової системи числення).

Якщо основи систем числення кратні один одному, тобто пов'язані залежністю

$$r = p^m, \tag{1.19}$$

то кожна цифра системи числення з основою r може бути зображена m цифрами в системі числення з основою p (хоча загальні правила переведення чисел із однієї позиційної системи числення в іншу за допомогою послідовного ділення і множення залишаються в силі і в цьому випадку).

Доведення розглянемо на прикладі переведення числа A із вісімкової системи числення у двійкову.

Згідно з (1.5) можна записати:

$$\begin{aligned}(A)_8 = \sum_{i=-m}^{n-1} a_{i(8)} \cdot 10^i_{(8)} = a_{n-1} \cdot 8^{n-1} + a_{n-2} \cdot 8^{n-2} + \ \ldots \\ + a_i \cdot 8^i + \ \ldots \ + a_1 \cdot 8^1 + a_0 \cdot 8^0 + a_{-1} \cdot 8^{-1} + \ \ldots \ + a_{-m} \cdot 8^{-m},\end{aligned} \tag{1.20}$$

де $10_{(8)} = 8_{(10)} = 2^3_{(10)}$.

Будь-яка вісімкова цифра може бути представлена у вигляді трирозрядного двійкового числа, тобто у вигляді тріади двійкових цифр:

$$0_{(8)} = 000_{(2)}; \quad 1_{(8)} = 001_{(2)}; \quad 2_{(8)} = 010_{(2)}; \quad 3_{(8)} = 011_{(2)};$$

$$4_{(8)} = 100_{(2)}; \quad 5_{(8)} = 101_{(2)}; \quad 6_{(8)} = 110_{(2)}; \quad 7_{(8)} = 111_{(2)},$$

або в загальному випадку у вигляді такої суми

$$a_{i(8)} = a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0, \tag{1.21}$$

де $a_{i(2)} = 0,1$ – ображення будь-якої двійкової цифри.

Підставимо значення (1.20) у формулу (1.19):

$$A = \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{3n-3} +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{3n-6} + \ \ldots \ +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^3 +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^0 +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{-3} +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{-6} + \ \ldots \ +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{-3m+3} +$$
$$+ \left(a_{i(2)} \cdot 2^3 + a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0\right)2^{-3m}.$$

Розкриємо дужки і отримаємо:

$$A = a_{i(2)} \cdot 2^{3i-1} + a_{i(2)} \cdot 2^{3i-2} + a_{i(2)} \cdot 2^{3i-3} +$$
$$+ a_{i(2)} \cdot 2^{3i-4} + a_{i(2)} \cdot 2^{3i-5} + a_{i(2)} \cdot 2^{3i-6} + \ \ldots \ +$$
$$+ a_{i(2)} \cdot 2^5 + a_{i(2)} \cdot 2^4 + a_{i(2)} \cdot 2^3 +$$
$$+ a_{i(2)} \cdot 2^2 + a_{i(2)} \cdot 2^1 + a_{i(2)} \cdot 2^0 +$$
$$+ a_{i(2)} \cdot 2^{-1} + a_{i(2)} \cdot 2^{-2} + a_{i(2)} \cdot 2^{-3} +$$
$$+ a_{i(2)} \cdot 2^{-4} + a_{i(2)} \cdot 2^{-5} + a_{i(2)} \cdot 2^{-6} + \ \ldots \ +$$
$$+ a_{i(2)} \cdot 2^{-3m+5} + a_{i(2)} \cdot 2^{-3m+4} + a_{i(2)} \cdot 2^{-3m+3}$$
$$+ a_{i(2)} \cdot 2^{-3m+2} + a_{i(2)} \cdot 2^{-3m+1} + a_{i(2)} \cdot 2^{-3m}.$$

Виконавши процедуру згортання останнього виразу, отримаємо:

$$A = \sum_{i=-m}^{n-1} a_{i(8)} \cdot 10_{(8)}^{i} = \sum_{i=-3m}^{3n-1} a_{i(2)} \cdot 10_{(2)}^{i}, \qquad (1.21)$$

що і необхідно було довести.

Виконавши аналогічні перетворення при переведенні із шістнадцяткової системи числення у двійкову, можна отримати:

$$A = \sum_{i=-m}^{n-1} a_{i(16)} \cdot 10_{(16)}^{i} = \sum_{i=-m}^{4n-1} a_{i(2)} \cdot 10_{(2)}^{i}. \qquad (1.22)$$

Отже, для того щоб перевести число з початкової системи числення в нову систему, основа якої кратна основі початкової системи, досить кожну цифру числа, що переводиться, записати за допомогою m цифр у новій системі числення, якщо основа початкової системи числення більша від основи нової системи числення. У іншому разі кожні m цифр початкового

числа необхідно записати за допомогою однієї цифри в новій системі числення, починаючи для цілих чисел з молодшого розряду і зі старшого розряду для правильних дробів.

Наприклад, при переведенні вісімкового числа 316_8 у двійкову систему числення досить кожну цифру вісімкового числа записати у вигляді двійкової тріади, оскільки $8 = 2^3$:

$$316_8 = \underbrace{011}_{3}\ \underbrace{001}_{1}\ \underbrace{110}_{6} = 011001110_2.$$

При переведенні, наприклад, двійкового числа в шіснадцяткову систему числення досить кожну тетраду початкового числа записати у вигляді шістнадцяткової цифри:

$$100101111\,0001011101_2 = \underbrace{1001}_{9}\ \underbrace{0111}_{7}\ \underbrace{1100}_{C}\ \underbrace{0101}_{5}\ \underbrace{1101}_{D} = 97C5D_{16}.$$

1.7.6 Інші методи переведення чисел із однієї системи числення в другу

Виконувати процедуру ділення в системах числення з малою основою (наприклад, у двійковій системі числення) важко. Тому на практиці при необхідності переведення чисел із системи числення з малою основою в систему із великою основою можна користуватися загальним записом чисел у вигляді полінома (1.5), у якому виразити усі цифри a_i і основу p цифрами у системі числення з основою r:

$$(A)r = \sum_{i=-m}^{n} (a_i)_r (p)_r^i. \qquad (1.22)$$

Наприклад, при переведенні двійкових чисел в десяткову систему числення на практиці звичайно підраховують суму ступенів основи 2, при яких коефіцієнти a_i дорівнюють одиниці. Розрахунки ведуться при цьому в десятковій системі числення.

Приклад. Перевести число $(A)_2 = 1011{,}011$ у десяткову систему числення:

$$(A)_{10} = 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2} + 1\cdot 2^{-3} =$$
$$= 8 + 2 + 1 + 0{,}25 + 0{,}125 = 11{,}375_{10}.$$

При застосуванні такого методу може бути використана таблиця, що наведена у додатку 1.

У загальному випадку можна обчислити многочлен (1.22) безпосередньо у такому вигляді:

$$(A)_p = ((\ldots(a_{n-1}p + a_{n-2})p + \ldots +)p + a_1)p + a_0 +$$
$$= + p^{-1}\left(a_{-1} + p^{-1}\left(a_{-2} + \ \ldots\ + p^{-1}\left(a_{-m+1} + p^{-1}a_{-m}\right)\ldots\ \right)\right), \qquad (1.23)$$

у якому необхідно представити усі значення a_i і p в системі числення з основою r:

$$a_i = (a_i)_r,$$
$$p = (p)_r,$$

і виконати усі дії за правилами арифметики із основою r.

Приклад. Для $(A)_2$ =1011,011 в десятковій системі числення, отримаємо:

$$(C)_2 = \quad 1 \quad 0 \quad 1 \quad 1$$
$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$
$$(C)_{10} = ((1\cdot 2+0)2+1)2+1 = 11$$
$$(U)_2 = 0,\ 0 \quad 1 \quad 1$$
$$\downarrow \quad \downarrow \quad \downarrow$$
$$(U)_{10} = 2^1\left(0+2^{-1}\left(1+2^{-1}\cdot 1\right)\right) = 0{,}25(1+0{,}5) = 0{,}375.$$

Тобто $(A)_{10}$ = 11,375

2 ПРАВИЛА ВИКОНАННЯ АРИФМЕТИЧНИХ ОПЕРАЦІЙ В ПОЗИЦІЙНИХ СИСТЕМАХ ЧИСЛЕННЯ

2.1 Вибір основи системи числення для застосування в цифрових пристроях

Очевидно, що непозиційні системи непридатні для практичного застосування в цифрових пристроях внаслідок своєї громіздкості і труднощі виконання арифметичних операцій.

З позиційних систем найбільш зручні однорідні системи числення, оскільки однорідна основа, тобто однакова кількість символів у всіх розрядах приводить до найбільш раціонального використання обладнання і найбільш простих алгоритмів виконання арифметичних операцій. Тому проаналізуємо однорідні позиційні системи числення на предмет їх застосування в цифрових пристроях. При цьому будемо враховувати наступні чинники:

- можливість простого і однозначного представлення чисел в заданому діапазоні;
- простота виконання арифметичних операцій і швидкодія цифрових систем;
- економічність системи, тобто простота технічної реалізації цифрової системи.

Таким чином, задача вибору системи числення для застосування в цифрових пристроях і МП зводиться, по суті, до задачі вибору раціональної величини основи системи p. Тому розглянемо з вказаних вище позицій системи числення із різними основами.

Однозначність представлення чисел у заданому діапазоні визначається необхідною розрядністю n представлення максимально можливого числа $A_{\max}$. З урахуванням (1.6) для цілих чисел можна отримати:

$$n = \frac{\ln\left(A_{\max} + 1\right)}{\ln p}. \qquad (2.1)$$

Будь-який із символів, що застосовуються для запису чисел, повинен зображатися у вигляді одного або декількох станів якогось фізичного елемента. Очевидно, що елемент буде тим простішим, чим менше станів йому потрібно мати, тобто чим менше основа системи числення. Наприклад, для реалізації двійкової системи числення можна застосувати реле, конденсатори, магнітні, напівпровідникові елементи тощо.

Трійкову систему числення можна ще реально реалізувати при допомозі конденсаторів і магнітних елементів, а p-позиційні елементи (при $p > 3$) носять штучний характер і при збільшенні p їх реалізація

ускладнюється. Так, для реалізації десяткового елемента потрібно використати не менше 10 транзисторів.

Таким чином, по цьому критерію найбільш придатною для використання в цифрових пристроях є двійкова система числення.

Знайти оптимальне значення основи системи числення (з точки зору економічності основи системи числення) досить складно, оскільки, чим більше основа, тим менше кількість розрядів і, значить, елементів потрібно для зображення числа, однак тим більшу кількість символів повинен відображати кожен елемент, тобто мати більшу кількість стійких станів, що призводить до його ускладнення.

Для оцінки економічності системи числення як критерій виберемо кількість цифро-розрядів $W(p)$, необхідну для зображення числа в *i*-й системі числення, виходячи з умови, що $W(p)$ пропорційне кількості деталей обладнання, тобто

$$W(p) = pn. \tag{2.2}$$

Правомірність останнього твердження зумовлена тим, що, наприклад, при виконанні операції множенні час множення на один розряд пропорційний основі p, а на всі n розрядів-величині $(p \cdot n)$.

З урахуванням (2.1) отримаємо:

$$W(p) = p\frac{\ln(A_{\max}+1)}{\ln p}. \tag{2.3}$$

Припускаючи (умовно) безперервність функції $W(p)$, можна обчислити

$$\frac{dW}{dp} = \ln(A_{\max}+1)\frac{(\ln p - 1)}{\ln^2 p}. \tag{2.4}$$

Із (2.4) витікає, що функція $W(p)$ досягає мінімального значення при $p = e = 2{,}718281\ldots$ (e – основа натурального логарифму).

З урахуванням необхідності застосування цілих значень числа p з останнього виразу можна визначити, що найбільш оптимальними є двійкова і трійкова системи числення, а

$$\frac{W(2)}{W(3)} = \frac{2\ln 3}{3\lg 2} = 1{,}056.$$

Однак в сучасних цифрових системах використовується в основному двійкова система числення. Це визначається простотою схемних рішень цифрових елементів, які можуть знаходитись в двох станах (включено/виключено), (це можна трактувати, як значення цифр двійкової системи числення).

Порівняння десяткової і двійкової систем числення показує, що остання є в 1,5 рази економнішою, ніж десяткова:

$$\frac{W(10)}{W(2)} = \frac{10\ln 2}{2\lg 10} = 1{,}5.$$

Необхідно також звернути увагу на той факт, що двійково-десяткове кодування є біль продуктивним, ніж кодування двійкове. Продуктивність кодування при різних основах систем числення можна порівняти, визначаючи число бітів I_p, необхідних для представлення довільного числа в діапазоні $(0, A_{\max})$ в системах числення з різними основами.

При безпосередньому двійковому кодуванні

$$I_2 = \log_2(A_{\max} + 1). \qquad (2.5)$$

При двійковому кодуванні в системі числення з основою *p*

$$I_p = B_p \log_p(A_{\max} + 1), \qquad (2.6)$$

де B_p – кількість бітів, необхідних для однозначного представлення (кодування) цифр заданої системи числення.

З урахуванням (2.5) і (2.6) можна записати:

$$\frac{I_p}{I_2} = \frac{B_p}{\log_2 p}.$$

Для p = 10 і $B_p = 4$ (наприклад, для кодів 8421, EXCESS-3 тощо) отримаємо:

$$\frac{I_p}{I_2} = \frac{4}{\log_2 10} = 1{,}2.$$

Це означає збільшення приблизно на 20% кількості бітів, необхідних для представлення десяткових чисел в двійково-десяткових кодах (4 біти для кожної цифри) у порівнянні з двійковими кодами вказаних цифр.

Перевагами двійково-десяткової системи відносно двійкової є:

- відсутність необхідності переведення початкових даних і результатів розрахунків з однієї системи в іншу; зручність контролю проміжних результатів шляхом виведення на індикацію для візуального спостереження;
- більш широкі можливості для автоматичного контролю через наявність в двійково-десятковому коді надмірних комбінацій.

Перевагами двійкової системи числення відносно двійково-десяткової є:

- економія близько 20 % обладнання;
- майже в 1,5 рази більш висока швидкодія;
- спрощення логічної побудови і значна економія обладнання в схемах управління і у допоміжних ланцюгах.

Математичним апаратом для аналізу і синтезу обчислювальних пристроїв, що дозволяє відносно просто і економічно будувати вузли і блоки, є алгебра логіки. Найбільший розвиток і закінченість вивчення, внаслідок своєї простоти і широкого практичного застосування, отримала двійкова логіка. При цьому всі арифметичні і керуючі ланцюги можна будувати на основі двійкових елементів, застосовуючи для їх аналізу і синтезу один і той же математичний апарат, що значно полегшує проектування схем цифрових пристроїв. Ця обставина також дозволяє зробити висновок на користь двійкової системи числення у порівнянні з іншими системами.

Таким чином, виходячи з перерахованих критеріїв, найбільш прийнятною для застосування в цифрових пристроях є однорідна позиційна система числення з основою, яка дорівнює двом. Однак в деяких конкретних випадках при синтезі обчислювального пристрою якому-небудь критерію може надаватись більше значення, ніж іншим. Тоді для застосування вибирається система числення, оптимальна за вибраним критерієм. Наприклад, в деяких випадках перевагу віддають десятковій системі числення, керуючись при цьому не міркуваннями економічності вибираного числення, а зручністю спілкування людини з цифровим пристроєм.

Двійкову систему числення застосовують у великих і середніх ЕОМ і ПЕОМ, призначених для вирішення науково-технічних задач, для яких характерний великий об’єм обчислень і порівняно малий об’єм початкових даних і результатів обчислень. Її також доцільно застосовувати в ЕОМ, призначених для управління технологічними процесами.

Двійково-десяткову систему числення застосовують для вирішення економічних задач, які характеризуються великим об’ємом початкових даних, порівняльною простотою і малим об’ємом перетворень, що виконуються над ними і великою кількістю результатів обчислень. Цю систему доцільно також застосовувати в калькуляторах і ЕОМ, призначених для інженерних розрахунків, а також в персональних ЕОМ.

2.2 Загальні правила виконання арифметичних операцій в системах числення з основою *p*

Як показано в першій главі, позиційні системи числення мають чіткі і певні міжрозрядні зв’язки. Цю їх властивість можна розглядати як перевагу, оскільки вона забезпечує простоту виконання арифметичних операцій, але, разом з тим, і як недолік, оскільки ця властивість через наявність міжрозрядних перенесень призводить до обмеження технічної швидкості виконання арифметичних операцій.

Арифметичні операції додавання, віднімання і множення двох $(n+m)$ розрядних чисел $(X)_p$ і $(Y)_p$

$$\begin{cases} (X)_p = (x_{n-1} \ \dots \ x_i x_{i-1} \ \dots \ x_{-m})_p, \\ (Y)_p = (y_{n-1} \ \dots \ y_i y_{i-1} \ \dots \ y_{-m})_p, \end{cases} \tag{2.7}$$

в системі числення з основою p можна розглядати, як виконання вказаних операцій над окремими розрядами заданих чисел.

Результатом виконання операцій над цифрами i-го розряду x_i і y_i в системі числення з основою p є дві цифри:

- s_i - результат операції у i-му розряді;
- r_{i+1} - перенесення в наступний (i+1)-й розряд (для додавання і множення).

При виконанні основних арифметичних операцій можна сформулювати такі алгоритми формування цифр s_i і r_{i+1}:

- додавання

$$\left.\begin{aligned} (s_i)_д &= \begin{cases} x_i + y_i + r_i & для \quad x_i + y_i + r_i < p, \\ x_i + y_i + r_i - p & для \quad x_i + y_i + r_i \geq p, \end{cases} \\ (r_{i+1})_д &= \begin{cases} 0 & для \quad x_i + y_i + r_i < p, \\ 1 & для \quad x_i + y_i + r_i \geq p. \end{cases} \end{aligned}\right\} \tag{2.8}$$

- віднімання

$$\left.\begin{aligned} (s_i)_в &= \begin{cases} x_i - y_i - r_i & для \quad x_i - y_i - r_i \geq 0, \\ x_i - y_i - r_i + p & для \quad x_i - y_i - r_i < 0, \end{cases} \\ (r_{i+1})_в &= \begin{cases} 0 & для \quad x_i - y_i - r_i \geq 0, \\ 1 & для \quad x_i - y_i - r_i < 0. \end{cases} \end{aligned}\right\} \tag{2.9}$$

- множення

$$\left.\begin{aligned} (s_i)_м &= \begin{cases} x_i \cdot y_i + r_i & для \quad x_i \cdot y_i + r_i < p, \\ x_i \cdot y_i + r_i - \left\lfloor \dfrac{x_i \cdot y_i + r_i}{p} \right\rfloor p^{1)} & для \quad x_i \cdot y_i + r_i \geq p, \end{cases} \\ (r_{i+1})_м &= \begin{cases} 0 & для \quad x_i \cdot y_i + r_i < p, \\ \left\lfloor \dfrac{x_i \cdot y_i + r_i}{p} \right\rfloor & для \quad x_i \cdot y_i + r_i \geq p. \end{cases} \end{aligned}\right\} \tag{2.10}$$

У двійковій системі числення операція множення значно спрощується у зв'язку з тим, що завжди перенесення $(r_{i+1})_m = 0.$

Для обчислення s_i і r_{i+1} при виконанні решта операцій у двійковій системі числення можна скористатись табл. 2.1.

Таблиця 2.1 – операції для обчислення s_i і r_{i+1}

x	y	$x+y$		$x-y$		xy
		$(s_i)_д$	$(r_{i+1})_д$	$(s_i)_в$	$(r_{i+1})_в$	$(s_i)_м$
0	0	0	0	0	0	0
0	0	1	0	1	1	0
1	1	1	0	1	0	0
1	1	0	1	0	0	1

Реалізація операції ділення двох чисел не може бути виконана над однорозрядними цифрами.

У випадку виконання операцій над двійково-десятковими цифрами необхідно взяти до уваги результат, який виникає з урахуванням особливостей кодування.

Розглянемо виконання операції додавання двох двійко-во-десяткових цифр X_{BCD} і Y_{BCD}

$$\begin{cases} X_{BCD} = {}^{X}T_{n-1}\,{}^{X}T_{n-2} \ \ldots\ {}^{X}T_i \ \ldots \\ Y_{BCD} = {}^{Y}T_{n-1}\,{}^{Y}T_{n-2} \ \ldots\ {}^{Y}T_i \ \ldots, \end{cases} \qquad (2.10)$$

а також можливі варіанти перенесення C_i з $(i-1)$-ої тетради.

У випадку додавання

$${}^{X}T_i + {}^{Y}T_i + C_i$$

можна отримати результат, який дорівнює 0,1, ... ,18 або 19.

В табл. 2.2 - 2.4 наведені результати додавання еквівалентів двійкових цифр (T'), а також їх дійсна сума (T) для кодів:

1. 8421 (таблиця 2.2);
2. 8421 + 3 (таблиця 2.3);
3. 2421 (таблиця 2.4).

Алгоритми додавання для вказаних кодів можна сформулювати наступним чином:

1. Код 8421

- для $0000 \le T' \le 1001$, маємо $T = T'$ (корекція не потрібна);
- для $1001 \le T' \le 10011$, маємо $T = T' + 0110.$

2. Код 8421+3

- для $0110 \le T^{'} \le 1111$, маємо $T = T^{'} - 0011$ або $T = T^{'} + 1101$ (при цьому перенесення, що виникає у тетраді, належить опускати);

- для $10000 \le T^{'} \le 11001$, маємо $T = T^{'} + 0011$.

У цьому випадку значення необхідної корекції залежить від перенесення C_{i+1}:

- якщо $C_{i+1} = 0$, то $T = T^{'} - 0011$ або $T = T^{'} + 1101$ (перенесення з тетради опускається);

- якщо $C_{i+1} = 1$, то $T = T^{'} + 0011$.

3. Код 2421

- для $0000 \le T^{'} \le 0100$, $1011 \le T^{'} \le 10100$ (при ${}^{X}T_i \ge 5$, ${}^{Y}T_i < 5$), $T^{'} \ge 11001$, маємо $T = T^{'}$;

- для $1011 \le T^{'} \le 1001$ (при ${}^{X}T_i < 5$, ${}^{Y}T_i < 5$), маємо $T = T^{'} + 0110$;

- для $10110 \le T^{'} \le 11010$ (при ${}^{X}T_i \ge 5$, ${}^{Y}T_i \ge 5$), маємо $T = T^{'} - 0110$ або $T = T^{'} + 1010$ (перенесення з тетради необхідно опускати).

Таблиця 2.2 – еквіваленти двійкових цифр $(T^{'})$, і їх дійсна сума для коду 8421

Сума десяткова	$\left({}^{X}T_i + {}^{Y}T_i + C_i\right)_{8421}$		Примітки
	$T^{'}$	T	
0	0000	0000	$T = T'$
1	0001	0001	
2	0010	0010	
3	0011	0011	
4	0100	0100	
5	0101	0101	
6	0110	0110	
7	0111	0111	
8	1000	1000	
9	1001	1001	
10	1010	1 ← 0000	$T = T'$+0110
11	1011	1 ← 0001	
12	1100	1 ← 0010	
13	1101	1 ← 0011	
14	1110	1 ← 0100	
15	1111	1 ← 0101	
16	1 0000	1 ← 0110	
17	1 0001	1 ← 0111	
18	1 0010	1 ← 1000	
19	1 0011	1 ← 1001	

Таблиця 2.4 – еквіваленти двійкових цифр (T'), і їх дійсна сума для коду 8421+3

Сума десяткова	$\left({}^{X}T_i + {}^{Y}T_i + C_i\right)_{8421+3}$		Примітки
	T'	T	
0	0110	0011	$T = T'$–0011
1	0111	0100	або
2	1000	0101	$T = T'$+1101
3	1001	0110	(при цьому
4	1010	0111	перенесення,
5	1011	1000	що виникає
6	1100	1001	із тетради, не
7	1101	1010	враховується
8	1110	1011	$C_{i+1} = 0$)
9	1111	1100	
10	1 0000	1 ← 0011	
11	1 0001	1 ← 0010	
12	1 0010	1 ← 0101	
13	1 0011	1 ← 0110	
14	1 0100	1 ← 0111	$T = T'$+0011
15	1 0101	1 ← 1000	
16	1 0110	1 ← 1001	
17	1 0111	1 ← 1010	
18	1 1000	1 ← 1011	
19	1 1001	1 ← 1100	

Таблиця 2.5 – еквіваленти двійкових цифр (T'), і їх дійсна сума для коду 2421

<table>
<tr><td rowspan="2">Сума десяткова</td><td colspan="3">$\left({}^{X}T_i + {}^{Y}T_i + C_i\right)_{2421}$</td><td colspan="2" rowspan="2">Примітки</td></tr>
<tr><td colspan="2">T'</td><td>T</td></tr>
<tr><td>0
1
2
3
4</td><td colspan="2">0000
0001
0010
0011
0100</td><td>0000
0001
0010
0011
0100</td><td colspan="2">$T = T'$</td></tr>
<tr><td></td><td>${}^{X}T_i \geq 5,\ {}^{Y}T_i < 5$</td><td>${}^{X}T_i < 5, {}^{Y}T_i < 5$</td><td></td><td rowspan="3">$T = T'$</td><td></td></tr>
<tr><td>5
6
7
8
9</td><td>1011
1100
1101
1110
1111</td><td>1011
0110
0111
1000
1001</td><td>1011
1100
1101
1110
1111</td><td>$T = T' +$
$+ 0110$</td></tr>
<tr><td></td><td></td><td>${}^{X}T_i \geq 5,\ {}^{Y}T_i < 5$</td><td></td><td>$T = T' -$
$- 0110$
або
$T = T' +$
$+ 0110$</td></tr>
<tr><td>10
11
12
13
14</td><td>1 0000
1 0001
1 0010
1 0011
1 0100</td><td>1 0110
1 0111
1 1000
1 1001
1 1010</td><td>1←0000
1←0001
1←0010
1←0011
1←0100</td><td colspan="2"></td></tr>
<tr><td>15
16
17
18
19</td><td colspan="2">1 1011
1 1100
1 1101
1 1110
1 1111</td><td>1←1011
1←1100
1←1101
1←1110
1←1111</td><td colspan="2">$T = T'$</td></tr>
</table>

З аналізу таблиць видно, що:

1. правила додавання залежать від типу коду, що застосовується,
2. в деяких випадках необхідно провести корекцію результату.

Приклад. Виконати операції додавання

$$\begin{array}{r} 17 \\ +\,85 \\ \hline 102 \end{array}$$

з використанням розглянутих BCD кодів.

а) код 8421

0000 0001 0111 (17)
+ 0000 1000 0101 (85)
———————————
0000 1001 1100
+ 0110 (корекція 1)
———————————
0000 1010← 0010
+ 0110 (корекція 2)
———————————
0001 0000 0010 (102)

б) код 8421 + 3

0011 0100 1010 (17)
+ 0011 1011 1000 (85)
———————————
0111 ← 0000 ← 0010
+ 1101 0011 0011 (корекція)
———————————
(не враховується) ← 0100 0011 0101 (102)

в) код 2421

0000 0001 1101 (17)
+ 0000 1110 1011 (85)
———————————
0001← 0000 ←1000
+ 1010 (корекція)
———————————
0001 0000 ↓ 0010 (102)
(не враховується)

Тобто виявляється, що введення поправки (корекції) є багатогранним процесом і не можна у цьому випадку поставити в залежність введення поправки від наявності перенесення.

2.3 Доповнення в позиційних системах числення

В позиційних системах числення з основою p для $(n+m)$ - розрядного числа

$$A = a_{n-1}a_{n-2} \ \dots \ a_1a_0, a_{-1} \ \dots \ a_{-m}$$

вводиться два визначення доповнення:

- доповнення до p

$$\overline{\overline{A}} = p^n - A \quad \text{для} \quad A \neq 0,$$
$$(\overline{\overline{A}} = 0 \quad \text{для} \quad A = 0) \tag{2.11}$$

- доповнення до (p - 1)

$$\overline{A} = p^n - A - p^{-m} \tag{2.12}$$

В залежності від значення основи системи числення застосовуються такі назви доповнень:

- доповнення до двох (двійкове) і доповнення до одиниці (одиничне) (p = 2);

- доповнення до восьми (вісімкове) і доповнення до семи (сімкове) (p = 8);

- доповнення до десяти (десяткове) і доповнення до дев'яти (дев'яткове) (p = 10);

- доповнення до шістнадцяти (шістнадцяткове) і доповнення до п'ятнадцяти (п'ятнадцяткове) (p = 16).

Варто зазначити, що двократне доповнення заданого числа є самим числом, тому що:

- доповнення до p доповнення до p числа A визначається так

$$\overline{\overline{p^n - A}} = p^n - \left(p^n - A\right) - A = A; \tag{2.13}$$

- доповнення до (p - 1) доповнення до (p - 1) числа A визначається так:

$$\overline{p^n - A - p^{-m}} = p^n - \left(p^n - A - p^{-m}\right) - p^{-m} = A. \tag{2.14}$$

Необхідно відмітити, що доповнення до p можна обчислити, як суму доповнення до (p - 1) і одинички молодшого розряду p^{-m}

$$\overline{\overline{A}} = p^n - A = \overline{A} + p^{-m}. \tag{2.15}$$

Приклад.

а) для десяткового числа A = $1636{,}671_{10}$ маємо

$$n = 4, \quad m = 3,$$

- доповнення десяткове $\overline{\overline{A}} = 10^4 - 1636{,}671 = 8363{,}329,$

- доповнення дев'яткове $\overline{A} = \overline{\overline{A}} - 10^{-3} = 8363{,}328;$

б) для двійкового числа A = 110,10 маємо:

$$n = 3, \quad m = 2,$$

- двійкове доповнення $\overline{\overline{A}} = 2^3 - 110{,}10 = 1{,}10$,

- одиничне доповнення $\overline{A} = \overline{\overline{A}} - 2^{-2} = 1{,}01$.

З наведених визначень і розглянутого прикладу випливає, що для числа *A*, заданого в позиційній системі числення з основою *p,* будуть справедливі такі співвідношення:

$$A + \overline{\overline{A}} = \underbrace{100\ \ldots\ 0{,}0\ \ldots\ 0}_{(n+m)} \tag{2.16}$$

$$A + \overline{A} = \underbrace{(p-1)(p-1)\ \ldots\ (p-1),(p-1)\ \ldots\ (p-1)}_{(n+m)}. \tag{2.17}$$

Як легко зауважити, доповнення до (p - 1) для числа, заданого в позиційній системі числення з основою *p,* найпростіше обчислити за допомогою віднімання кожної цифри числа від (p - 1). Тобто

$$\overline{A} = \sum_{i=-m}^{n-1}(p-1-a_i)p^i = \sum_{i=-m}^{n-1}\overline{a_i}\,p^i; \tag{2.18}$$

де $\overline{a_i}$ - доповнення до (p - 1) цифри a_i.

Тоді, з урахуванням (2.17), отримаємо

$$\overline{\overline{A}} = \sum_{i=-m}^{n-1}(p-1-a_i)p^i + p^{-m} = \sum_{i=-m}^{n-1}\overline{a_i}\,p^i + p^{-m}. \tag{2.19}$$

Алгоритм такого перетворення розглянемо на наступних прикладах.

Приклад.

а) двійкова система числення: $A = 1\ 0\ 1\ 1\ 0,\ 0\ 1\ 0\ 1$

$\Downarrow$

$1 - a_i$

$\Downarrow$

доповнення одиничне

$\overline{A} = 0\ 1\ 0\ 0\ 1,\ 1\ 0\ 1\ 0$

$+\ 2^{-4} = \qquad\qquad 1$

доповнення двійкове $\overline{\overline{A}} = 0\ 1\ 0\ 0\ 1,\ 1\ 0\ 1\ 1$

в) десяткова система числення: $A = 3\ \ 5\ \ 4\ \ 8,\ \ 9\ \ 2\ \ 7\ \ 0$

$\Downarrow$

$9 - a_i$

$\Downarrow$

доповнення дев'яткове $\overline{A} = 6\ \ 4\ \ 5\ \ 1,\ \ 0\ \ 7\ \ 2\ \ 9$

$+10^{-4} = \qquad\qquad\qquad\qquad 1$

доповнення десяткове $\overline{\overline{A}} = 6\ \ 4\ \ 5\ \ 1,\ \ 0\ \ 7\ \ 3\ \ 0$

с) шістнадцяткова система: $A = B\ \ 7\ \ C,\ \ F\ \ 3\ \ 6$

$\Downarrow$

$F - a_i$

$\Downarrow$

доповнення п'ятнадцяткове $\overline{A} = 4\ \ 8\ \ 3,\ \ 0\ \ C\ \ 9$

$+16^{-3} = \qquad\qquad 1$

доповнення шістнадцяткове $\overline{\overline{A}} = 4\ \ 8\ \ 3,\ \ 0\ \ C\ \ A.$

У випадку використання кодування десяткових чисел двійковими кодами, що самодоповнюються, досить легко обчислити дев'яткове доповнення десяткових цифр. Це доповнення отримується через одиничне доповнення кожного біту коду (заміна 0 на 1 і 1 на 0 у кожному біті).

Приклад. Виконаємо вказану процедуру для числа $A = 79{,}4$, заданого в коді 8421 плюс 3: $A_{8421+3} = 1010\ \ 1100,\ \ 0111$

$\Downarrow$

$1 - a_{ij}$

$\Downarrow$

- доповнення дев'яткове $\overline{A}_{8421+3} = 0101\ \ 0011,\ \ 1000$

$+1$

- доповненнядесяткове $\overline{\overline{A}}_{8421+3} = 0101\ \ 0011,\ \ 1001$

Якщо для кодування використовується код 8421, який не має властивості самодоповнення, розглянутий вище спосіб необхідно модифікувати за допомогою введення корекції, яка полягає у додаванні до кожної тетради коректуючого коду 0110.

Приклад. Десяткове число $A = 79{,}4$ в коді 8421 має такий вигляд:

$$A_{8421} = 0111\ \ 1001,\ \ 0100$$

$$+0110\ \ 0110,\ \ 0110$$

$$1101\ \ 1111,\ \ 1010$$

$$\Downarrow$$

$$1 - a_{ij}$$

$$\Downarrow$$

- доповнення дев'яткове $\overline{A}_{8421} = 0010\ \ 0000,\ \ 0101$

$$+1$$

- доповнення десяткове $\overline{\overline{A}}_{8421} = 0010\ \ 0000,\ \ 0110.$

Використання розглянутих понять доповнень дозволяє спростити виконання операцій віднімання в позиційних системах числення з основою *p*.

2.4 Віднімання чисел без знаку за допомогою доповнення

2.4.1 Загальний підхід

Операція віднімання двох чисел $(A - B)$, заданих в одній і тій же системі числення з основою *p:*

$$A = a_{n-1}a_{n-2}\ \ \ldots\ \ a_1a_0, a_{-1}\ \ \ldots\ \ a_{-m},$$

$$B = b_{n-1}b_{n-2}\ \ \ldots\ \ b_1b_0, b_{-1}\ \ \ldots\ \ b_{-m},$$

може бути виконана за допомогою додавання до зменшуваного доповнення до *p* від'ємника $\left(A + \overline{\overline{B}}\right)$ або доповнення до (p – 1) від'ємника $\left(A + \overline{B}\right)$ з застосуванням певних правил. Визначимо ці правила окремо для кожного варіанта заміни операції віднімання на операції додавання.

2.4.2 Застосування доповнення до *p*

Для двох $(m+n)$-розрядних чисел, заданих в системі числення з основою *p*, обчислимо значення

$$A + \overline{\overline{B}} = A + \left(p^n - B\right) = A + p^n - B.$$

а) Якщо $A \geq B$, то $A + p^n - B \geq p^n$.

Це означає, що виникає перенесення. Опускаючи ймовірність перенесення (відкидання перенесення), отримаємо

$$A + \overline{\overline{B}} = A - B \qquad \text{(результат додатній).}$$

б) Якщо $A < B$, то $A + p^n - B < p^n$.

Це означає, що перенесення не виникає. Обчисливши доповнення до p для виразу $\left(A + p^n - B\right)$ зі знаком мінус, отримаємо:

$$\overline{\overline{-\left(A + p^n - B\right)}} = -\left\{p^n - \left(A + p^n - B\right)\right\} = -(B - A).$$

Для цього випадку

$$A + \overline{\overline{B}} = -(B - A) \quad \text{(результат від'ємний).}$$

Осмислення вищенаведеного дозволяє реалізувати операцію віднімання двох додатних чисел $(A - B)$ із застосуванням наведених в пунктах а) і в) правил.

Приклади.

1. Для двох десяткових чисел A = 29 і B = 38 обчислити вирази $(A - B)$ і $(B - A)$ із застосуванням десяткового доповнення:

а)

$$\begin{array}{r} A = 29 \\ \overline{\overline{B}} = 62 \end{array} \Big\} + \qquad \begin{array}{r} B = 38 \\ \overline{\overline{A}} = 71 \end{array} \Big\} +$$

$$0 \leftarrow 91 \qquad 1 \leftarrow \lceil 09$$

(перенесення відсутнє) (перенесення не враховується)

$$A - B = -\overline{\overline{91}} = -09 = -9 \qquad B - A = +09 = +9.$$

2. Для вказаних чисел, представлених у двійковій системі числення (A = 011101 і B = 100110) обчислити їх різницю із застосуванням двійкового доповнення:

а) $A - B$

$$\begin{array}{r} A = 011101 \\ \overline{\overline{B}} = 011010 \end{array} \Big\} +$$

$$0 \leftarrow 110111$$

$$A - B = -\overline{\overline{110111}} = -001001,$$

б) $B - A$

$$\begin{array}{r} B = 100110 \\ \overline{\overline{A}} = 100011 \end{array} \Big\} +$$

$$1 \leftarrow \lceil 001001$$

$$B - A = +001001.$$

3. Для вказаних десяткових чисел, представлених в кодах 8421 і 8421+3, обчислити їх різницю:

а[1])

$$A_{8421} - B_{8421}$$

$$\left.\begin{array}{l} A_{8421} = 0010 \quad 1001 \\ \overline{\overline{B}}_{8421} = 0110 \quad 0010 \end{array}\right\} +$$

$$1000 \quad 1011$$

$$+ \quad 0110 \text{ (корекція)}$$

$$0 \leftarrow 1001 \quad 0001$$

$$A_{8421} - B_{8421} = -1001 \quad 0001 = -0000\ 1001_{8421} \hat{=} (-9)_{10}$$

$$1001 \quad 0001$$

$$+ 0110 \quad 0110 \quad \text{(корекція)}$$

$$1111 \quad 0111$$

$$0000 \quad 1000$$

$$+ \quad 1$$

$$\underbrace{0000 \quad 1001}$$

$$9_{10}$$

б[1])

$$B_{8421} - A_{8421}$$

$$\left.\begin{array}{l} B_{8421} = 0011 \quad 1000 \\ \overline{\overline{A}}_{8421} = 0111 \quad 0001 \end{array}\right\} +$$

$$1010 \quad 1001$$

$$+ \quad 0110 \qquad \text{(корекція)}$$

$$1 \leftarrow \quad 0000 \quad 1001$$

$$B_{8421} - A_{8421} = +0000 \quad 1001_{8421} \hat{=} (+9)_{10}$$

а[2])

$$A_{8421+3} - B_{8421+3}$$

$$\left.\begin{array}{l} A_{8421+3} = 0101 \quad 1100 \\ \overline{B}_{8421+3} = 1001 \quad 0101 \end{array}\right\} +$$

$$\begin{array}{r} 0\urcorner \quad 1\urcorner \\ 1111 \quad 0001 \\ + \quad 1101 \quad 0011 \ \text{(корекція)} \\ \hline 1\leftarrow 1100 \quad 0100 \end{array}$$

$$A_{8421+3} - B_{8421+3} = -1100 \ \ 0100 = -0011 \ \ 1100_{8421+3} \hat{=} (-9)_{10}.$$

б[2])

$$B_{8421+3} - A_{8421+3}$$

$$\left.\begin{array}{l} B_{8421+3} = 0110 \quad 1011 \\ \overline{A}_{8421+3} = 1010 \quad 0100 \end{array}\right\} +$$

$$\begin{array}{r} \ulcorner \ 1\urcorner \quad 0\urcorner \\ 0000 \quad 1111 \\ + \quad 0011 \quad 1101 \ \text{(корекція)} \\ \hline \end{array}$$

$$B_{8421+3} - A_{8421+3} \quad = \quad + \ 0011 \quad 1100 \quad \hat{=} (+9)_{10}.$$

2.4.3 Застосування доповнення до (*p*-1)

Для двох $(m+n)$ - розрядних чисел, заданих в системі числення з основою *p*, обчислимо значення

$$A + \overline{B} = A + p^n - B - p^{-m}.$$

а) Якщо $A > B$, то $A + p^n - B - p^{-m} \geq p^n$.

Враховуючи, що виникаюче перенесення буде додано до результату як 1 у молодшому розряді

$$\left.\begin{array}{r} p^n + A - B - p^{-m} \\ + p^{-m} \end{array}\right\} +$$

$$A \ - B$$

отримаємо, що результат цієї операції додатній, тобто:

$$A + \overline{B} = A - B \qquad \text{(результат додатній).}$$

б) Якщо $A \le B$, то $A + p^n - B - p^m < p^n$.

Це означає, що перенесення не наступає. Виконавши обчислення доповнення до (p - 1) виразу $(A + p^n - B - p^m)$ зі знаком мінус отримаємо:

$$\overline{-\left(A + p^n - B - p^m\right)} = -\left\{p^n - \left(A + p^n - B - p^m\right) - p^m\right\} =$$
$$= -(B - A).$$

Тобто операція віднімання (A - B) може бути замінена на операцію додавання згідно з правилами, наведеними в пунктах а) і в).

Приклад.

1. Використовуючи дев'яткове доповнення для десяткових чисел $A = 29_{10}$ і $B = 38_{10}$, обчислити їх різницю:

а) A - B

$$\left.\begin{matrix} A = 29 \\ \overline{B} = 61 \end{matrix}\right\} +$$

$$0 \leftarrow 90$$

(перенесення = 0)

$$\downarrow$$

$$A - B = -\overline{90} = -09 = -9_{10}$$

б) B - A

$$\left.\begin{matrix} B = 38 \\ \overline{A} = 70 \end{matrix}\right\} +$$

$$1 \leftarrow 08$$

(перенесення = 1) + 1

$$\downarrow 09$$

$$B - A = +09 = +9_{10}$$

2. Використовуючи одиничне доповнення для цих же чисел у двійковій системі числення A = 011101 і B = 1000110, обчислити їх різницю:

а) A - B

$$\left.\begin{matrix} A = 011101 \\ \overline{B} = 011001 \end{matrix}\right\} +$$

$$0 \leftarrow 110110$$

(перенесення = 0)

$$\downarrow$$

$$A - B = -\overline{110110} =$$
$$= -001001 = -1001,$$

б) B - A

$$\left.\begin{matrix} B = 100110 \\ \overline{A} = 100010 \end{matrix}\right\} +$$

$$1 \leftarrow 001000$$

(перенесення = 1) + 1

$$\downarrow \overline{\qquad}$$

$$B - A = +001001 =$$
$$= +1001.$$

3. Використовуючи дев'яткове доповнення для десяткових чисел A = 29 і B = 38, представлених в кодах 8421 і 8421 плюс 3, обчислити їх різницю:

а[1])

$$A_{8421} - B_{8421}$$

$$\left.\begin{array}{l} A_{8421} = 0010 \quad 1001 \\ \overline{B}_{8421} = 0110 \quad 0001 \end{array}\right\} +$$

$$\begin{array}{r} 1000 \quad 1010 \\ + \quad 0110 \ \text{(корекція)} \\ \hline 0 \leftarrow 1001 \quad 0000 \end{array}$$

$$\downarrow$$

$$A_{8421} - B_{8421} = -\overline{1001\ 0000} = -0000\ 1001_{8421} \hat{=} (-9)_{10}$$

б[1])

$$B_{8421} - A_{8421}$$

$$\left.\begin{array}{l} B_{8421} = 0011 \quad 1000 \\ \overline{A}_{8421} = 0111 \quad 0000 \end{array}\right\} +$$

$$\begin{array}{r} 1010 \quad 1000 \\ + \quad 0110 \quad \text{(корекція)} \\ \hline 1 \leftarrow \quad 0000 \quad 1000 \\ \qquad\qquad\quad 1 \end{array}$$

$$B_{8421} - A_{8421} = +\overline{0000 \quad 1000}_{8421} \hat{=} (+9)_{10}$$

а[2])

$$A_{8421+3} - B_{8421+3}$$

$$\left.\begin{array}{l} A_{8421+3} = 0101 \quad 1100 \\ \overline{B}_{8421+3} = 1001 \quad 0100 \end{array}\right\} +$$

$$\begin{array}{r} 0\urcorner \quad 1\urcorner \\ \downarrow \quad 1111 \quad 0000 \\ + \quad 1101 \quad 0011 \ \text{(корекція)} \\ \hline 1100 \quad 0011 \end{array}$$

$$A_{8421+3} - B_{8421+3} = -\overline{1100\ \ 0011} = -0011\ \ 1100_{8421+3} \hat{=} (-9)_{10}.$$

б[2])

$$\begin{array}{r} B_{8421+3} - A_{8421+3} \\ \left.\begin{array}{l} B_{8421+3} = 0110\ \ 1011 \\ \overline{A}_{8421+3} = 1010\ \ 0011 \end{array}\right\} + \\ \hline \left|\begin{array}{r} \leftarrow 1 \quad 0 \\ 0000\ \ 1110 \\ \longrightarrow \quad 1 \end{array}\right. \end{array}$$

$$B_{8421+3} - A_{8421+3} = +\ 0011\ \ 1100 \hat{=} (+9)_{10}.$$

2.4.4 Загальні правила віднімання додатних чисел за допомогою доповнення

Застосування доповнення дозволяє сформулювати такі правила віднімання додатних чисел:

а) віднімання двох додатних чисел $(A - B)$, заданих в позиційній системі числення з однаковою основою, можна реалізувати за допомогою додавання до зменшуваного доповнення до *p* від'ємника $\left(A + \overline{\overline{B}}\right)$, при чому:

- якщо при додаванні виникає перенесення у старшому розряді, воно не враховується,
- якщо перенесення у старшому розряді не виникає, то результат операції дорівнює доповненню до *p* суми, взятому зі знаком мінус.

б) віднімання двох додатних чисел $(A - B)$, заданих в позиційній системі числення з однаковою основою, можна реалізувати за допомогою додавання до зменшуваного доповнення до (*p*- 1) від'ємника $\left(A + \overline{B}\right)$, при чому:

- якщо при додаванні виникає перенесення у старшому розряді, то необхідно додати 1 до молодшого розряду суми;
- якщо перенесення у старшому розряді не виникає, то результат операції дорівнює доповненню до (*p*- 1) суми, взятому зі знаком мінус.

Необхідно звернути увагу на той факт, що існує два варіанти арифметичного доповнення до (*p*- 1), тому що

$$A - A = A + \overline{A},$$

але

$$A - A = 00\ \ldots\ 0,0\ \ldots\ 0,$$

в той час, як

$$A + \overline{A}. = (p-1)(p-1)\ \ldots\ (p-1),(p-1)\ \ldots\ (p-1).$$

Це може привести до певних непорозумінь при виконанні арифметичних операцій.

Необхідно також зауважити, що додавання двох чисел може бути реалізовано за допомогою операції віднімання від доданку доповнення доданку при урахуванні наведених вище особливостей виконання операцій з застосуванням доповнення.

3 ДВІЙКОВА СИСТЕМА ЧИСЛЕННЯ

3.1 Загальні визначення і правила

Під двійковою системою числення розуміють таку систему, в якій для зображення чисел використовуються два символи, а ваги розрядів міняються згідно із законом $2^{\pm k}$ (де k – довільне ціле число).

Класичною двійковою системою є система з символами0,1. Її двійкові цифри часто називають бітами.

Щоб оволодіти будь-якою системою числення, треба уміти складати і множити в ній будь-які цифри. Арифметичні операції в двійковій системі числення виконуються так само, як і в любій позиційній системі числення з основою p відповідно до правил, розглянутих в попередньому розділі (див. табл. 2.1).

Як вже відмічалось, в загальному вигляді всі двійкові числа представляються у вигляді полінома

$$A = \sum_{i=-m}^{n-1} a_i \cdot 2^i \qquad (3.1)$$

Переведення в двійкову систему числення з десяткової проводиться або за загальним правилом переведення чисел з однієї позиційної системи числення в іншу, або десяткові числа переводяться у вісімкову систему за загальним правилом, а потім вісімкові числа переводяться в двійкові за правилом переведення чисел для систем з кратною основою. Зворотний переклад проводиться аналогічно або за допомогою загального вигляду запису числа (3.1) у вигляді полінома.

Додавання у двійковій системі числення проводиться за правилами додавання поліномів. Тому при додаванні чисел X і Y $i-$й розряд суми S_i і перенесення P_i із даного розряду в $(i+1)-$й буде визначатись у відповідності із наступним правилом:

$$x_i + y_i + P_{i-1} = S_i + 2P_i, \qquad (3.2)$$

де $P_i \in \{0,1\}, \quad S_i \in \{0,1\}.$

Виразу (3.2) відповідає табл. 3.1 додавання однороз-рядних двійкових чисел. У відповідності з табл. 3.1 можна підсумовувати багаторозрядні двійкові числа.

Таблиця 3.1 – додавання однорозряднх двійкових чисел

a_i	b_i	P_{i-1}	S_i	P_i	a_i	b_i	P_{i-1}	S_i	P_i
0	0	0	0	0	0	0	1	1	0
0	1	0	1	0	0	1	1	1	1
1	0	0	1	0	1	0	1	1	1
1	1	0	0	1	1	1	1	1	1

Приклад. Задано $X = 10_{10} = 1010_2$, і $Y = 4_{10} = 0100_2$. Знайти суму $Z = X + Y$. З урахуванням порозрядного додавання чисел X і Y отримаємо:

$$\begin{array}{r} 1010 \\ +\ 0100 \\ \hline \end{array}$$

$$Z = 1110_2 = 14_{10}.$$

3.2 Інші варіанти двійкової системи числення

3.2.1 Двійкова система з цифрами $1,\bar{1}$

З визначення двійкової системи числення витікає, що для зображення чисел можуть бути використані не тільки символи 0, 1, але і інші, наприклад, (1, -1) або (0, -1).

Символ (-1) звичайно зображають як $\bar{1}$, а систему, в якій використовуються символи (1,-1), називають системою $(1,\bar{1})$. Число $A = 37_{10}$ запишеться, наприклад, в цій системі як $A = 1\bar{1}\bar{1}1\bar{1}1$, якщо вважати в (3.1), що a_i приймає значення 1 або $\bar{1}$.

Особливістю двійкової системи числення з цифрами 1 і $\bar{1}$ є представлення єдиним кодом як додатних, так і від'ємних чисел. Однак в ній відсутній нуль і немає можливості представлення деяких чисел у вигляді кінцевої множини.

У системі $(1, \bar{1})$ деякі цілі і дробові числа, наприклад парне $(20 = 1\bar{1}\bar{1}1\bar{1},111...)$, представляють у вигляді нескінченних дробів, тобто з певною похибкою.

Разом з тим існують числа, які не мають єдиного зображення, наприклад, число 1 може бути представлене у вигляді

$$1 = \underbrace{1\ \bar{1}\ \bar{1}...\bar{1}}_{n-1} = 1\ \underbrace{0\ 0\ 0...0}_{n-1} - \underbrace{1\ 1\ 1...1}_{n-1} = 2^{n-1} - (2^{n-1} - 2^0), \qquad (3.3)$$

де $n = 0,1,2...N$.

Співвідношення (3.3) виражає зв'язок між звичайною двійковою системою числення іі системою з цифрами $(1,\bar{1})$.

Приклад. Перевести число A = 100101 в систему (1, $\bar{1}$). Використовуючи співвідношення (3.3), зводимо переклад до заміни комбінацій 001 і 01 комбінаціями відповідно $1\bar{1}\bar{1}$ і $1\bar{1}$, тобто $A = 11\bar{1}\bar{1}1\bar{1}$.

Для отримання кінцевого представлення як парних, так і непарних чисел в системі $(1,\bar{1})$ був запропонований наступний запис чисел:

$$A_{(1,\bar{1})} = \sum_{i=-m}^{n-1} a_i \cdot 2^i - 2^m \qquad (3.4)$$

У цьому випадку через адитивну добавку додатних чисел буде менше на одне число, ніж від'ємних. Всього при $N = n + m$ можна зобразити 2^N чисел: 2^{N-1} від'ємних, (2^{N-1} – 1) додатних і нуль замість (2^N – 1) чисел у двійковій системі із цифрами 0, 1, бо в останній число нуль представляється неоднозначно (+0, -0).

Для переведення двійкових чисел в систему (1, $\bar{1}$) існує декілька методів [ ], у більшості з яких необхідно враховувати наступне: у разі непарного числа переведення здійснюють за правилом (3.3), а потім в розряд $i = -m$ записують одиницю; при переведенні парного числа його спочатку перетворюють в непарне доданням одиниці в молодший розряд і тільки після цього переводять в систему $(1,\bar{1})$ за правилом (3.3) як непарне число. Потім до отриманого результату в розряд $i = -m$ записують $\bar{1}$.

Приклад. Перевести в систему (1, $\bar{1}$) двійкове число А = 11000. За вказаним правилом це число спочатку перетворюємо в непарне число 11001. Після цього замінюємо в зображенні числа комбінацію 001 на комбінацію $1\bar{1}\bar{1}$, приписуємо в розряд після коми цифру $\bar{1}$ і отримуємо $A_{1,\bar{1}}$ = $111\bar{1}\bar{1}$**,**$\bar{1}$.

Правило переведення із двійкової системи числення з цифрами 0, 1 в систему з цифрами 1, $\bar{1}$ можна формалізува-ти таким чином.

Для переведення додатних чисел спочатку до початкового числа приписується праворуч ще один розряд, значення якого дорівнює 1. Після чого у початковому зображенні виділяють конструкції, що складаються із послідовності нулів і одиниці праворуч, тобто конструкції вигляду 00...01. Ці конструкції на основі (3.3) перетво-рюються до вигляду $1\bar{1}...\bar{1}\bar{1}$, тобто самий старший нульовий розряд замінюється на 1, а інші розряди, включаючи 1 в молодшому розряді конструкції, замінюються на $\bar{1}$.

Приклад. Число $A_2 = 10001{,}10101$ записати в системі $(1,\bar{1})$.

Приписуємо до заданого числа праворуч розряд зі значенням 1:

$$A' = 10001,\ 101011.$$

Виділяємо конструкції виду 000...01.

$$A' = 1\ \underline{0001},\ 1\ \underline{01}\ \underline{01}\ 1. \quad .$$

Перетворюємо виділені конструкції за правилом (3.3) і отримуємо:

$$A_{1,\bar{1}} = 11\bar{1}\bar{1}\bar{1},\ 11\bar{1}1\bar{1}1.$$

У цьому випадку доданням 1 справа до початкового запису реалізується врахування адитивної поправки. Значення цієї поправки повинне бути одне і те ж для всіх чисел, тому, якщо в початковому записі дробова частина числа має менше, ніж $(m-1)$ значущих цифр, то її необхідно заповнити нулями до $(m-1)$ розрядів і вже в $m-$й розряд записати 1.

Приклад. Число $A_2 = 10{,}1$ записати в системі $(1,\bar{1})$ при $n = 3, m = 2.$

Спочатку підготуємо запис числа до перетворення: $A_2 = 0010{,}10$.

Доповнимо A розрядом справа зі значенням 1 і виділимо конструкції вигляду 00...01.

$$A' = \underline{001}\ \underline{0{,}1}\ \underline{01}.$$

Перетворюємо виділені конструкції за правилом (3.3) і отримуємо:

$$A_{1,\bar{1}} = 1\bar{1}\bar{1}1, \bar{1}1\bar{1}.$$

Для представлення від'ємних чисел в системі з цифрами 1, $\bar{1}$ спочатку перетворюється запис абсолютної величини числа A шляхом дописування зліва і справа необхідного числа нулів. Потім в цьому зображенні всі 1 замінюються на $\bar{1}$, а до отриманого зображення дописують 1 справа (адитивна добавка). Далі виділяються конструкції вигляду $00...0\bar{1}$ і 00...01, які перетворюються відповідно до вигляду $\bar{1}1...11$ і. $1\bar{1}...\bar{1}\bar{1}$

Приклад. Число $A_2 = -10{,}1$ записати в системі $(1,\bar{1})$.

$$A_2 = -0010{,}10;$$

$$A' = \underline{00\bar{1}}\ \underline{0{,}\bar{1}}\ \underline{01};$$

$$A_{1,\bar{1}} = \bar{1}1\,1\bar{1}{,}1\,1\bar{1}.$$

Двійкову систему числення з цифрами 1, $\bar{1}$ раціонально використати як проміжну при виконанні операцій множення і ділення. Пряма реалізація арифметичних дій в цій системі помітно складніша, ніж в системі з цифрами 0, 1.

Існують інші двійкові системи числення, наприклад, система із символами 1, 0, $\bar{1}$, яка є надмірною (надлишковою).

3.2.2 Надлишкова двійкова система

Як відомо, надлишковою системою числення з основою p називається система, в якій для запису чисел використовується кількість символів, більше, ніж p.

Надлишкова двійкова (симетрична) система числення пов'язана із звичайною двійковою системою таким співвідношенням:

$$\underbrace{1\ 1\ 1...1}_{k} = \sum_{i=0}^{k-1} 2^i = 2^k - 2^0 = 1\ \underbrace{0\ 0\ 0...0\ \bar{1}}_{k} \qquad (3.5)$$

На основі виразу (3.5) проводиться перехід від двійкової системи з цифрами (0, 1) до систем з цифрами (1,0,$\bar{1}$) і навпаки, наприклад, $A = 11110001_2 = 1000\bar{1}0001_{1,0,\bar{1}}$.

У будь-якій надлишковій системі числення одні і ті ж числа можна представити декількома способами. Наприклад $A = 0{,}01110011 = 0{,}0100\bar{1}0011 = 0{,}0100\bar{1}00\bar{1}0\bar{1}1.$

При цьому в надлишковій системі можна зменшити кількість одиниць в зображенні числа. Симетричні надлишкові системи числення дозволяють в ряді випадків спростити виконання арифметичних дій. Наприклад, надлишкову двійкову систему числення використовують в деяких алгоритмах прискорення операції множення.

В двійковій системі числення з цифрами (1, 0, $\bar{1}$), як і в системі числення з цифрами (1, $\bar{1}$), для позначення від'ємної величини до числа не потрібно приєднувати додатковий знак, отже, при виконанні арифметичних дій над числами не потрібно використовувати ще і правила операцій із знаками. Крім того, в цій системі при виконанні операції можна уникнути поширення перенесення між розрядами далі, ніж на два (і навіть один) сусідній розряди.

Іншими словами, в цій системі за рахунок надлишковості, що вводиться, можна реалізувати методи додавання, в яких кожний розряд суми є функцією тільки суміжних справа розрядів складових величин. Це означає, що час виконання операції додавання не залежить від розрядності операндів і еквівалентний часу додавання трьох або навіть двох розрядів. Звідси витікає, що коди чисел при додаванні можуть поступати в порядку від старших розрядів до молодших, тобто починаючи зі старших розрядів.

Для доведення розглянемо наступний алгоритм додавання знакорозрядних двійкових чисел *A* і *B*.

На першому етапі проводиться порозрядне додавання операндів *A* і *B*:

$$a_i + b_i = P_i' + S_i'$$

Значеннями P_i' і S_i' можуть бути $\bar{1}$, 0, 1, але $|P_i'| = 1$, якщо $|a_i + b_i| \geq 1$ (табл. 3.2).

Таблиця 3.2 – Значення $P_i^{'}$ і $S_i^{'}$

a_i	b_i	$P_i^{'}$	$S_i^{/}$	S_i	P_{i-1}	$S_i^{''}$	$P_i^{''}$
1	1	1	0	1	1	0	1
1	0	1	1	1	0	1	0
1	1	0	0	1	1	0	0
0	1	1	1	0	1	1	0
0	0	0	0	0	0	0	0
0	1	1	1	0	1	1	0
1	1	0	0	1	1	0	0
1	0	1	1	1	0	1	0
1	1	1	0	1	1	0	1

На другому етапі до отриманого значення розрядної суми $S_i^{'}$ додається перенесення $P_{i-1}^{'}$ із попереднього розряду:

$$S_i^{'} + P_{i-1}^{'} = 2P_i^{''} + S_i^{''}.$$

Тут вже $\left|P_i^{''}\right| = 1$, якщо $S_i^{'} + P_{i-1}^{'} \geq 2$ (табл. 3.3).

На третьому етапі додаються значення $S_i^{''}$ і $Pi_i^{'}$

$$S_i^{''} + P_{i-1}^{''} = 2P_i^{''} + S_i.$$

Нехай це підсумовування здійснюється точно так само, як на другому етапі, тобто $S_i \in \{\bar{1},\ 0,\ 1\}$ і $\left|P_i^{'}\right| = 1$, якщо $S_i^{'} + P_{i-1}^{'} \geq 2$. Можна показати, що при цих умовах $P_i^{''} \equiv 0$.

Таблиця 3.3 – Значення $P_i^{'}$ і $S_i^{'}$

$a_i + b_i$	P_{i-1}	P_i	$P_i^{'}$	$S_i^{'}$
2	0	1	0	0
1	0	0	0	1
0	0	0	0	0
-1	0	0	1	1
-2	0	0	1	0
2	1	1	0	1
1	1	1	0	0
0	1	0	0	1
-1	1	0	0	0
-2	1	0	1	1

Дійсно, $P_i^{''} \neq 0$, якщо $S_i^{''} = P_{i-1}^{''} = \pm 1$. Нехай спочатку $P_{i-1}^{''} = 1$, але тоді $S_{i-1}^{'} = P_{i-2}^{'}$. а раз $S_{i-1} = 1$, то обов'язково повинне бути $P_{i-1} = \bar{1}$. Це, в свою чергу, приводить до того, що $S_i^{'} \neq 1$ (див. табл. 3.2). Також само доводиться, що $P_{i-1}^{''} \neq -1$; отже, $P_i^{''} \equiv 0$ і тоді $S_i = S_i \cup P_{i-1}$.

Таким чином, при додаванні знакорозрядних двійкових операндів перенесення може розповсюджуватися не більш ніж на два розряди і, отже, максимальний час підсумовування операндів довільної розрядності становить не більше $3\tau_s$, вважаючи, що в кожному онорозрядному суматорі час $\tau_{s,}$ формування суми більший, ніж час τ_p формування перенесення.

Для перенесення допустимими значеннями були $\bar{1}$, 0, 1. Можна вважати, що при додаванні використовуються два типи перенесень додатній P_i зі значеннями 0, 1 і від'ємний P_i зі значеннями 0, $\bar{1}$. При наявності двох типів перенесень можна перебудувати алгоритм підсумовування таким чином. Спочатку знаходять суму a_i, b_i і додатнього перенесення з попереднього розряду P_i:

$$a_i + b_i + P_{i-1} = 2P_i + 2P_i^{'} + S_i^{'},$$

де для S_i допустимі значення 0, 1 (табл. 3.3). Потім отримане значення розрядної суми $S_i^{'}$ уточнюється шляхом додавання до неї від'ємного перенесення з попереднього розряду:

$$S_i = S_i + P_{i-1}^{'}.$$

Очевидно, що при такому додаванні перенесення бути не може. Хоч даний алгоритм і не приводить до прискорення додавання, бо тут перенесення також розповсюджується на два розряди, але при практичній реалізації він може привести до менших витрат обладнання.

Для двійкової системи числення можна добитися поширення перенесеня на один розряд, якщо один операнд представити у знакорозрядній формі, а другий - у звичайній двійковій системі числення. У цьому випадку на першому етапі порозрядно підсумовуються a_i і b_i:

$$a_i + b_i + P_{i-1} = 2P_i^{'} + S_i^{'}.$$

Нехай $a_i \in \{\bar{1},\ 0,\ 1\}$, а $b_i \in \{0,\ 1\}$. Цікаво, що можна вибрати значення $P_i^{'} + S_i^{'}$ так, що $P_i^{'} \in \{0,\ 1\}$, а $S'_i \in \{0, \bar{1}\}$. На другому етапі додаються значення $S_i^{'}$ і $P_{i-1}^{'}$:

$$S_i = S_i + P_{i-1}^{'}.\bullet$$

Очевидно, що при цьому ніколи не буде переповнення і $S_i \in \{1,\ 0,\ \bar{1}\}$.

Таким чином, використання знакорозрядного представ-лення двійкових чисел дозволяє реалізувати алгоритм додавання, при якому час виконання операції не залежить від розрядності операндів і не перевищує $3\tau_s$ (коли обидва операнди представлені у знакорозрядній формі) і навіть $2\tau_s$, (коли тільки один операнд представлений в знакорозрядній формі).

При множенні, на відміну від звичайної двійкової системи, утворення добутку може починатися також зі старших розрядів. У цьому випадку процес виконання операції множення може бути зупинений, коли буде

отримана необхідна кількість цифр добутку, тобто досягнута необхідна точність.

3.2.3 Основні правила роботи з двійковими числами

Хоч усі правила виконання операцій в двійковій системі числення дуже прості, але проте при роботі з двійковими числами через відсутність навичок виникають різного роду незручності. Нижче приведені деякі прості прийоми, які дозволяють досить вільно працювати з двійковими числами.

Правило 1. Число $\underbrace{100...00}_{k \text{ нулів}} = 2^k$.

Бажано знати напам'ять десяткові значення цих чисел від $k = 0$ хоча б до $k == 12$. Вони приведені в табл. 3.4.

Таблиця 3.4 – десяткові значення цифр

k	2^k	k	2^k	k	2^k	k	2^k
0	1	4	16	8	256	12	4096
1	2	5	32	9	512	13	8192
2	4	6	64	10	1024	14	16384
3	8	7	128	11	2048	15	32768

Правило 2. Число

$$\underbrace{11\ldots11}_{k \text{ одиниць}} = \left(\underbrace{11\ldots11}_{k \text{ одиниць}} + 1 \right) - 1 = 2^k - 1.$$

Правило 3. Бажано знати напам'ять десяткові значення двійкових чисел від 0 до 15 включно (табл. 3.5). Ці числа надалі будемо називати "малими числами".

Таблиця 3.5 – Десяткові значення двійкових чисел від 0 до 15

Двійкове число	Десяткове число	Двійкове число	Десяткове число
0000	0	1000	8
0001	1	1001	9
0010	2	1010	10
0011	3	1011	11
0100	4	1100	12
0101	5	1101	13
0110	6	1110	14
0111	7	1111	15

Правило 4. Двійкове число

$$\underbrace{a_{n-k+5}a_{n-k+4}a_{n-k+3}a_{n-k+2}a_{n-k+1}}_{\text{мале число } a}\underbrace{0\ 0...0\ 0}_{k \text{ нулів}} = a\cdot 2^k.$$

Приклади. $1101000 = 1101\cdot 2^3 = 13\cdot 8 = 104$;
$101000000 = 101\cdot 2^6 = 5\cdot 64 = 320.$

Правило 5. Двійкове число

$$\underbrace{a_{n-k+5}a_{n-k+4}a_{n-k+3}a_{n-k+2}a_{n-k+1}}_{\text{мале число } a}0\ 0...0\ 0\underbrace{bbbb}_{\substack{\text{мале число } b \\ k \text{ розрядів}}} = a\cdot 2^k + b$$

Приклади. $10110000101 = 1011\cdot 2^7 + 101 =$
$= 11\cdot 128 + 5 = 1413;$
$101000001101 = 101\cdot 2^9 + 1101 =$
$= 5\cdot 512 + 13 = 2573.$

Правило 5. Читання двійкових дробів

$$A = 0,\ \underbrace{0\ 0\ ...0\ 0}_{(m-1)\text{ нулів}}1 = 2^{-m},$$

$$A = 0,\ \underbrace{1\ 1\ ...1\ 1}_{m \text{ одиниць}} = 1 - 2^{-m}.$$

Двійковий дріб читається за тими ж правилами, що і десятковий: розряди справа від коми читаються як ціле число, яке є чисельником; знаменник читається як ціле число, що є k-им ступенем двох, причому k-номер молодшого розряду праворуч від коми.

Приклад. $A = 0{,}11011 = 27/2^5 = 27/32$ (читається “двадцять сім тридцять других");

$A = 0{,}00110110 = 27/2^8 = 54/256$ (читається “п'ятдесят чотири двісті п'ятдесят шостих").

Правило 6. Двійкові дроби можуть бути періодичними. Наприклад, періодичними є дроби:

$$\frac{1}{2^k-1} = 0,\ \underbrace{(0\ 0...0\ 0\ 1)}_{k \text{ розрядів}},\quad \frac{1}{2^k+1} = 0,\ \underbrace{(0\ 0...0\ 0}_{k \text{ нулів}}\ \underbrace{1\ 1...1\ 1)}_{k \text{ одиниць}}.$$

Приклади.

$$\frac{1}{3} = \frac{1}{2^1+1} = \frac{1}{2^2-1} = 0,(01) = 0{,}010101...$$

$$\frac{1}{5} = \frac{1}{2^2+1} = 0,(0011) = 0{,}00110011...$$

$$\frac{1}{7} = \frac{1}{2^3-1} = 0,(001) = 0{,}001001...$$

$$\frac{1}{9} = \frac{1}{2^3+1} = 0,(000111) = 0{,}000111000111...$$

$$\frac{1}{15} = \frac{1}{2^4 - 1} = 0,(0001) = 0,00010001...$$

3.3 Форми подання двійкових чисел в цифрових системах

3.3.1 Загальні визначення

Під машинним зображенням числа розуміють подання числа A в розрядній сітці цифрової системи.

Машинне зображення числа умовно позначають символом [A]. При цьому

$$A = [A]K_A \tag{3.6}$$

де K_A □ коефіцієнт, величина якого залежить від форми подання числа в системі.

Під формою подання чисел розуміють збір правил, що дозволяє встановити взаємну відповідність між записом числа і його кількісним еквівалентом.

Якщо довільне дійсне число A' таке, що $A' = [A]K_A$, то вважають, що таке число точно подається в машині при заданій формі подання чисел. Якщо ж $A' \neq [A]K_A$, то довільне дійсне число A' може бути подане в машині приблизно або взагалі не може бути подане.

При наближеному поданні дійсне число A' замінюється деяким числом $[A]$, що належить множині машинних чисел. Множині машинних чисел належать тільки числа, кратні двом, оскільки будь-які два попарно сусідніх машинних числа відрізняються один від одного на величину 2^{-n}, де n - кількість розрядів. Число, відмінне від нуля, але подане в цифровому пристрої як нуль. тобто число $|A| < A_{min}$, називають машинним нулем. Числа, більші A_{max}, в цифрових пристроях також не можуть бути подані. У цьому випадку кажуть про переповнення розрядної сітки.

Існує три форми подання чисел: природна, з фіксованою комою і нормальна.

Природною формою запису числа називається запис числа у вигляді полінома, представленого в скороченому вигляді так:

$$A = a_{n-1}a_{n-2}...a_1a_0a_{-1}a_{-2}...a_{-m}. \tag{3.7}$$

При цьому відлік ваги розрядів ведеться від коми. При природній формі запису кома ставиться на суворо певному місці між цілою і дробовою частиною числа.

Числа, подані в такій формі, називають числами з природною комою. При цій формі подання для кожного числа необхідно вказати положення його коми в одному з розрядів коду, тобто місце положення коми повинне

бути передбачене в кожному розряді, для чого необхідне додаткове обладнання. Недоліки подання чисел в формі з природною комою зумовлені також ускладненням арифметичних вузлів і трудністю оперування з дуже великими або дуже малими по абсолютній величині числами. Тому цю форму застосовують лише в кальку ля-торах, незважаючи на звичність подання чисел.

Необхідність у вказівці положення коми повністю відпадає, якщо місце коми в розрядній сітці машини заздалегідь фіксоване. Така форма подання чисел називається подання з фіксованою комою. Тому цифрові пристрої і системи, які використовують запис чисел в цій формі, називаються пристроями і системами з фіксованою комою.

Крім форми запису (3.7) число можна подати у вигляді добутку цілої степені основи системи і цифрової частини, що є правильним дробом:

$$A = p^k \sum_{i=-m}^{n-1} a_i p^{i-k} \qquad (3.8)$$

Така форма запису називається нормальною, показник степені при основі p називається порядком числа, а цифрову дробову частину-мантисою. Оскільки обмежень на величину мантиси немає, то положення коми в ній може мінятися при відповідній зміні порядку. Внаслідок цього цифрові пристрої і системи, що використовують нормальний запис числа, називаються пристроями з плаваючою комою.

3.3.2 Подання чисел у формі з фіксованою комою

У цифрових пристроях з фіксованою комою числа можуть бути подані у вигляді правильних дробів або у вигляді цілих величин, тобто кому фіксують або перед старшим розрядом числа, або після молодшого розряду:

$$A'_{фк} = \pm 0, a_{-1} a_{-2} ... a_{-m}; \qquad (3.9)$$

$$A''_{фк} = \pm a_{n-1} a_{n-2} ... a_1 a_0,. \qquad (3.10)$$

В першому варіанті числа, більші одиниці, приводяться до стандартного вигляду за допомогою масштабного коефіцієнта K_A. Подання чисел у вигляді правильних дробів зумовлене необхідністю зменшити можливість переповнення розрядної сітки цифрових пристроїв, тобто зменшити небезпеку втрати значущих цифр старших розрядів при виконанні арифметичних операцій (особливо операцій множення).

При фіксації коми після молодшого розряду також необхідно проводити масштабування початкових даних.

Для запису і збереження даних в цифрових пристроях і системах використовуються комірки пам'яті або регістри заданої розрядності.

У пристроях і системах, які оперують з фіксованою перед старшим розрядом комою, число записується в розрядну сітку у вигляді значущої частини дробу зі своїм знаком, тобто для запису $m-$розрядного дробу розрядна сітка повинна містити $(m+1)$ розряд (рис.3.1).

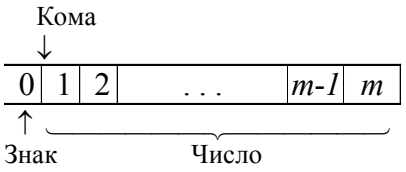


Рисунок 3.1 □ Формат запису числа з фіксованою комою(кома перед старшим розрядом)

Окремих розрядів для запису цілої частини числа (0) і коми не виділяється, оскільки їх положення зумовлене формою запису чисел (при записі даних у вигляді цілих чисел знак також є старшим розрядом).

Знак числа звичайно кодується таким чином: знаку "+" (додатні числа) відповідає 0 у знаковому розряді, знаку "- " (від'ємні числа) – 1 у знаковому розряді.

Величини чисел, що подаються в цифрових пристроях з фіксованою перед старшим розрядом комою, лежать в межах

$$2^{-m} \le |A| \le 1 - 2^{-m}. \tag{3.11}$$

При чому числа, менші значення $[-A_{\max}] = -1 + 2^{-m}$ і більші значення $[+A_{\max}] = 1 - 2^{m}$ пристроєм не сприймаються. Область чисел, котрі знаходяться в межах $(-2^{-m} < A < 2^{-m})$, називається областю нечутливості і визначається похибкою подання чисел.

Величини чисел, представлених при фіксації коми після молодшого розряду, лежать в межах

$$1 \le |A| \le 2^{n} - 1 \tag{3.12}$$

Для оцінки можливостей різних форм подання чисел вводиться поняття діапазону подання чисел:

$$D = \frac{|A_{\max}|}{|A_{\min}|}, \tag{3.13}$$

який визначає максимальну кількість можливих варіантів запису числа для заданої розрядності, яка застосовується в цифровому пристрої.

Діапазон подання чисел при фіксації коми перед старшим розрядом дорівнює

$$D'_{фк} = \frac{|A_{\max}|}{|A_{\min}|} = \frac{1-2^{-m}}{2^{-m}} = \frac{2^m - 1}{1} \cong 2^m. \qquad (3.14)$$

Діапазон подання чисел при фіксації коми після молодшого розряду дорівнює

$$D''_{фк} = \frac{|A_{\max}|}{|A_{\min}|} = \frac{2^n - 1}{1} \cong 2^n. \qquad (3.15)$$

При $m = n$ діапазон подання чисел в цифрових пристроях з фіксованою комою однаковий.

Наприклад, при $n = m = 32,\ D_{фк} = 2^{32} \approx 4{,}3 \cdot 10^9.$

3.3.3 Подання чисел у формі з плаваючою комою

В цифрових пристроях з плаваючою комою числа подаються в нормальній формі

$$A = p^{\beta_A} M_A, \qquad (3.16)$$

де $M_A = \pm 0, a_{-1} a_{-2} \ldots a_{-m}$ – мантиса числа A (правильний дріб), p^{β_A} – характеристика числа A, $\beta_A = \pm a_{k-1} a_{k-2} \ldots a_1 a_0$ – порядок (ціле число зі знаком).

Для підвищення точності запису чисел, мантиси в цифрових пристроях з плаваючою комою подаються в нормалізованому вигляді, при якому діапазон подання мантиси лежить в межах:

$$p^{-1} \le \left| M_A^{н} \right| < 1, \qquad (3.17)$$

тобто старший розряд мантиси є завжди значущим.

Для запису числа в цифрових пристроях відводиться $(k+1)$ розряд для фіксації порядку зі своїм знаком і $(m+1)$ розряд для мантиси, тобто місця, що займається в коді числа кодом мантиси і кодом порядку, заздалегідь фіксовані (рис. 3.2).

	k	*k-2*	...	*1*	*0*		1	2	…	*m*
Знак порядку	Порядок					Знак числа	Мантиса			

Рисунок 3.2 □ Формат запису числа з плаваючою комою

Діапазон подання порядку числа у формі з плаваючою комою лежить в межах $-(p^k - 1) \le p^{\beta_A} \le p^k - 1$. Отже, мінімальне число для $p = 2$, яке можна

подати в елементах пам’яті цифрових пристроїв, визначиться таким чином:.

$$\left|A_{\min}\right| = 2^{-|\beta_{\max}|} \cdot \left| M^{н}_{\min} \right| = 2^{-(2^k-1)} \cdot 2^{-1} = 2^{-2^k}.$$

Максимальне число при цьому буде:

$$\left|A_{\max}\right| = 2^{+|\beta_{\max}|} \cdot \left|M^{н}_{\max}\right| = 2^{2^k-1}(1-2^{-m}) \approx 2^{2^k-1},$$

(оскільки m звичайно велике, то $1-2^{-m} \approx 1$).

Очевидно, що діапазон чисел, що подватись в цифрових пристроях з плаваючою комою, значно більший, ніж в пристроях з фіксованою комою:

$$D_{пк} = \frac{\left|A_{\max}\right|}{\left|A_{\min}\right|} \approx \frac{2^{2^k-1}}{2^{-2^k}} = 2^{2^{k+1}-1} \qquad (3.18)$$

Наприклад, якщо з 32 розрядів 7 + 1 займе порядок числа, а інші 24 □ мантиса, то діапазон подання чисел складе:

$$D_{пк} = 2^{2^8-1} = 2^{255} \approx 10^{26}.$$

Співставляючи між собою розглянуті дві основні форми подання чисел, можна прийти до наступних висновків.

Діапазон подання чисел в цифрових пристроях з фіксованою комою значно менший, ніж в пристроях з плаваючою комою, а точність обчислень залежить від величин початкових чисел. Програмування для цифрових систем з фіксованою комою значно складніше, оскільки доводиться вводити масштабні коефіцієнти, щоб уникнути переповнення розрядної сітки при виконанні арифметичних операцій. При цьому від правильності вибору масштабних коефіцієнтів залежить помилка подання. Отже, їх обчислення повинно проводитися таким чином, щоб виключити можливість появи не тільки переповнення розрядної сітки, але і машинного нуля, що буває часом складно здійснити навіть досвідченому програмісту. Крім того, необхідно пам’ятати, що правильний результат алгебраїчного додавання можливий тільки для однакових масштабних коефіцієнтів, а операції множення і ділення формують нові масштабні коефіцієнти результатів.

Одна цифрові системи з плаваючою комою конструктивно більш складні, оскільки необхідно вводити додаткове обладнання для виконання операцій над порядками чисел, а також передбачити операцію нормалізації і вирівнювання порядків чисел. Плаваюча кома менш зручна, ніж фіксована, і в тих випадках, коли розподіл розрядів в розрядній сітці не відповідає тій схемі, яка заздалегідь прийнята для чисел з плаваючою комою. Найбільш важливий з цих випадків – це операції над командами Крім того, час виконання операцій над числами в цифрових пристроях з плаваючою комою більший, ніж в аналогічних пристроях з фіксованою комою, що зумовлено необхідністю роботи з порядками.

Як і при фіксованій комі, тут можливе переповнення розрядної сітки, яке виражається в тому, що результат якої-небудь операції має порядок, більший допустимого. Це приводить до аварійної ситуації. При виконанні операцій можливе отримання чисел, що мають порядок, менший допустимого і нормалізовану мантису. Ці числа розглядаються як машинні нулі, так само як і числа, що мають нульову мантису і допустимий порядок.

3.4 Точність подання чисел в цифрових пристроях

При рішенні різних задач потрібна різна точність кінцевих результатів. Так, при рішенні інженерних задач достатня точність до (3 – 4) десяткових знаків (10 – 13) двійкових), при рішенні наукових задач – до (8 – 9) десяткових або (25 – 30) двійкових знаків, а при рішенні особливо точних задач – до (15 – 20) десяткових, тобто (50 – 60) двійкових розрядів.

При обмеженій довжині машинних слів множина чисел, які можна подати в цифровому пристрої, є кінцевим. Тому подання чисел в цифрових системах, як правило, спричиняє за собою появу похибок, величина яких залежить як від форми подання чисел, так і від довжини розрядної сітки. Необхідно зазначити, що запас точності в цифрових системах береться звичайно в 1,5-2 рази більший, ніж це необхідно, що зумовлено крім всього іншого накопиченням похибок в ході обчислень.

Точність подання числа характеризується абсолютною і відносною похибками.

Абсолютна похибка це різниця між істинним значенням вхідної величини A і її значенням, отриманим з машинного зображення $[A]$

$$\Delta = A - [A]. \tag{3.19}$$

Усереднена абсолютна похибка подання чисел в цифрових пристроях з фіксованою комою визначається, як середня арифметична між мінімальним числом, яке може бути подано в даній розрядній сітці, і його мінімальною втратою. Якщо кома фіксована перед старшим розрядом, то абсолютна похибка дорівнює:

$$\Delta = \frac{A_{\min} + 0}{2} = \frac{2^{-m} + 0}{2} = 2^{-(m+1)}. \tag{3.20}$$

Тобто в машинах з фіксованою комою абсолютна погрішність постійна і дорівнює половині молодшого розряду.

Відносна похибка подання чисел визначається, як відношення усередненої абсолютної похибки до самого числа:

$$\varepsilon = \left| \frac{\Delta}{A} \right|. \tag{3.21}$$

Оскільки саме число A міняється в межах

$$A_{\min} = 2^{-m} \le |A| \le 1 - 2^{-m} = A_{\max},$$

те і відносна погрішність є величиною змінною, що міняється відповідно в межах

$$\varepsilon_{\min} \le \varepsilon \le \varepsilon_{\max} \tag{3.22}$$

Для цифрових пристроїв з фіксованою комою вона визначається таким чином:

$$\begin{cases} \varepsilon_{\min} = \left| \dfrac{\Delta}{A_{\max}} \right| = \dfrac{2^{-(m+1)}}{1-2^{-m}} \approx 2^{-(m+1)}; \\ \varepsilon_{\max} = \left| \dfrac{\Delta}{A_{\min}} \right| = \dfrac{2^{-(m+1)}}{2^{-m}} \approx 2^{-1}. \end{cases} \tag{3.23}$$

Таким чином, відносна похибка для машин з фіксованою комою залежить від величини числа і коливається в межах від $2^{-(m+1)}$ для великих чисел, до 2^{-1} для малих чисел.

В цифрових пристроях з плаваючою комою абсолютна похибка подання числа визначається таким чином:

$$\Delta = \Delta a \cdot 2^{\beta} \tag{3.24}$$

де Δa – похибка подання мантиси, яка визначається так само, як і абсолютна похибка подання чисел в пристроях з фіксованою комою, тобто $\Delta a = 2^{-(m+1)}$, β – порядок числа, який змінюється в межах

$$-(2^k - 1) \le \beta \le +(2^k - 1)$$

Отже, на відміну від цифрових пристроїв з фіксованою комою, в пристроях з плаваючою комою абсолютна похибка подання чисел залежить від порядку числа: похибка мінімальна при найбільшому від'ємному порядку $-\beta_{\max}$ і максимальна при найбільшому додатному порядку $\beta_{\max}$.

Мінімальне і максимальне значення абсолютної похибки визначаються таким чином:

$$\begin{cases} \Delta_{\min} = 2^{-(m+1)} \cdot 2^{-(2^k-1)} = 2^{-(m+2^k)}; \\ \Delta_{\max} = 2^{-(m+1)} \cdot 2^{(2^k-1)} = 2^{(2^k-m-2)}. \end{cases} \tag{3.25}$$

Відносна похибка подання чисел в цифрових пристроях з плаваючою комою визначається за загальним правилом:

$$\varepsilon = \left| \frac{\Delta}{A} \right| = \frac{\Delta a \cdot 2^k}{a \cdot 2^k} = \frac{\Delta a}{a},$$

тобто не залежить від порядку числа і змінюється в межах:

$$\begin{cases} \varepsilon_{\min} = \dfrac{\Delta a}{a_{\max}} = \dfrac{2^{-(m+1)}}{1-2^{-m}} = 2^{-(m+1)} \\ \varepsilon_{\max} = \dfrac{\Delta a}{a_{\min}} = \dfrac{2^{-(m+1)}}{2^{-1}} = 2^{-m} \\ \varepsilon_{\min} = \dfrac{\Delta a}{a_{\max}} = \dfrac{2^{-(m+1)}}{1-2^{-m}} = 2^{-(m+1)} \\ \varepsilon_{\max} = \dfrac{\Delta a}{a_{\min}} = \dfrac{2^{-(m+1)}}{2^{-1}} = 2^{-m} \end{cases} \qquad (3.26)$$

Отже, в цифрових пристроях з плаваючою комою, на відміну від пристроїв з фіксованою комою, відносна похибка зображення чисел у всьому діапазоні подання практично постійна і для чисел з нормалізованою мантисою залежить від кількості розрядів мантиси: чим їх більше, тим менше похибка подання.

У деяких обчислювальних засобах інформаційною одиницею є не окремі числа, а їх блоки або масиви, тобто послідовності, що складаються з сотень і тисяч чисел. У цих випадках нерідко застосовується проміжна форма поданя чисел, так зване представлення з поблочно плаваючою комою, при якому всьому масиву чисел привласнюється загальний порядок і масив вважається нормалізованим, якщо хоч би одне його слово є нормалізованим. Природно, що відносна похибка подання окремих елементів масиву буде при цьому різною. Як і у разі подання з фіксованою комою, максимальний по абсолютній величині елемент буде заданий з мінімальною похибкою, в той час як мінімальний по абсолютній величині елемент масиву – з максимальною відносною похибкою. Однак це не має істотного значення, оскільки основне інформаційне навантаження в цих випадках несуть максимальні елементи масивів. Разом з тим завдяки поданню чисел з поблочно-плаваючою комою вдається при прийнятній точності обчислень значно скоротити об’єм обладнання, а головне – час виконання операції, оскільки дії над порядками в цьому випадку виконуються тільки один раз за час обробки всього масиву чисел.

З цього витікає, що не можна віддати перевагу якій-небудь одній формі подання чисел. Звичайно в обчислювальних машинах загального призначення застосовують нормальну форму. Цим забезпечується великий діапазон подання чисел, висока точність обчислень, простота

програмування. Ускладнення апаратури цих машин має другорядне значення.

В мікропроцесорах і спеціалізованих ЕОМ частіше застосовують фіксовану або поблочно-плаваючу кому, якщо інформація обробляється окремими масивами, оскільки ці форми забезпечують простоту конструкції цифрової системи. Програма для реалізації алгоритмів обчислень складається тільки один раз, діапазон зміни величин відомий заздалегідь, масштабні коефіцієнти підбираються один раз, необхідна точність обчислень також відома заздалегідь і визначає довжину розрядної сітки.

В сучасних ЕОМ використовуються обидві форми подання чисел. При цьому в більшості випадків формат чисел з фіксованою комою служить для подання цілих двійкових і десяткових чисел і виконання операцій над ними, що, наприклад, необхідно для операцій над кодами адрес (операції індексної арифметики).

В режимі з плаваючою комою звичайно використовується система числення з основою 16. У цьому випадку число A вважається нормалізованим, якщо хоч би один з чотирьох старших двійкових розрядів відрізняється від 0. Це дещо зменшує точність подання чисел, але дозволяє різко збільшити діапазон чисел, що подаютья в машині і прискорити виконання деяких операцій, зокрема нормалізації, оскільки зсув мантиси проводиться відразу на 4 розряди. При цьому кожний зсув мантиси на 4 розряди ліворуч або праворуч вимагає відповідної зміни порядку шістнадцяткового числа всього лише на одиницю.

Точність подання в цьому випадку підвищують за рахунок використання формату подвійної довжини (подання з подвоєною точністю)

4 КОДИ ДВІЙКОВИХ ЧИСЕЛ

4.1 Загальні відомості

Цифрові пристрої і МП призначені, в першу чергу, для виконання арифметичних і логічних операцій. У зв'язку з цим у складі цих пристроїв завжди виділяється один із самих важливих функціональних пристроїв – арифметично-логічний пристрій (АЛП) або суматор. Числа які беруть участь в арифметичних операція, називаються операндами.

При проектуванні цифрових пристроїв і систем необхідно вирішувати питання представлення додатних і від'ємних чисел, виконання основних арифметичних операцій, оцінки можливості отримання заданої точності обчислень і визначення ознаки переповнення розрядної сітки.

Ці задачі вирішуються за допомогою використання спеціальних кодів, при застосуванні яких можна спростити апаратну реалізацію АЛП, розширити їх функціональні можливості і підвищити швидкодію.

Для представлення двійкових чисел в цифрових пристроях і МП застосовують прямий, обернений і доповняльний коди. У всіх цих кодах додатні числа мають один і той же вигляд, а від'ємні – різний. Розглянемо ці коди і правила виконання в них основних арифметичних операцій.

4.2 Прямий код

Самим простим машинним кодом є прямий код, у якому знак числа кодується нулем (додатні числа) і одиницею (від'ємні числа), а цифрова частина числа залишається без змін. Тобто прямим кодом від'ємного числа називається його зображення у звичайній формі запису, у якому в знаковому розряді ставиться одиниця. Прямий код додатного двійкового числа співпадає з його звичайним зображенням у природній формі, тому що знак кодується нулем.

Згідно визначенню, функція кодування чисел в прямому коді для правильних дробів вигляду: $A = a_{зн}a_{-1}a_{-2}...a_{-m}$ запишеться таким чином:

$$[A]_{пр} = \begin{cases} A & при \quad A \geq 0; \\ 1 + |A| & при \quad A \leq 0. \end{cases} \tag{4.1}$$

Величина числа A буде визначатись в прямому коді таким виразом:

$$A = (1 - 2a_{зн}) \sum_{i=-1}^{-m} a_i p^i, \tag{4.2}$$

при цьому знаковому розряду не приписується ніякої ваги.

Очевидно, що діапазон зміни машинних зображень для прямого коду лежить в межах

$$-\left(1-2^{-m}\right) \le [A]_{пр} \le \left(1-2^{-m}\right). \qquad (4.3)$$

Приклад. Записати числа A = + 0,1000110101101101 і B = – 0,0111010110010101 в прямому коді:

$$[A]_{пр} = 0{,}1000110101101101;$$
$$[B]_{пр} = 1{,}0111010110010101.$$

В прямому коді нуль має два значення: додатне 0,000... і від'ємне 1,000... Звичайно в ЕОМ використовується додатній нуль, але в процесі обчислення може виникнути і його від'ємне зображення. Ці зображення повністю еквіва-лентні і використання будь-якого з них не приводить до помилки.

4.3 Обернений код

Обернений код двійкового числа (правильний дріб) записується у відповідності з формулою:

$$[A]_{об} = \begin{cases} A & при \quad A \ge 0; \\ 2-2^{-m}+A & при \quad A \le 0, \end{cases} \qquad (4.4)$$

де m – розрядність числа A.

Для додатних чисел обернений код співпадає з його прямим кодом.

Для запису оберненого коду від'ємного числа необхідно виконати інверсію усіх розрядів числа, а в знаковому розряді записати одиницю. Ця особливість є важливою перевагою оберненого коду, тому що перетворення числа із прямого коду в обернений і навпаки є порозрядною операцією інвертування, що спрощує і прискорює виконання операції перетворення кодів.

Приклад. Записати числа A = + 0,1000110101101101 і B = – 0,0111010110010101 в оберненому коді:

$$[A]_{об} = 0{,}1000110101101101;$$
$$[B]_{об} = 1{,}1000101001101010.$$

Зауваження щодо двох значень коду нуля, визначене раніше при розгляді прямого коду, справедливі і для оберненого коду.

4.4 Доповняльний код

Обернений код двійкового числа (правильний дріб) записується у відповідності з формулою:

$$[A]_д = \begin{cases} A & при \quad A \geq 0; \\ 2 + A & при \quad A \leq 0. \end{cases} \tag{4.5}$$

Для додатних чисел обернений код співпадає з його прямим кодом.

Для визначення правила запису доповняльного коду для від'ємного числа перетворимо вираз (4.5) таким чином:

$$A_д = 2 - |A| = 2 - 2^{-m} - |A| + 2^{-m} = A_{об} + 2^{-m}. \tag{4.6}$$

Тобто, для обчислення доповняльного коду від'ємного числа, необхідно записати його обернений код і додати до його молодшого розряду одиницю.

Згідно з (4.5), нуль в доповняльному коді має одно значення:

$$[0]_д = 0{,}000...0,$$

$$[-0]_д = [A]_{об} + 2^{-m} = 1{,}111...1 + 0{,}000...0 = 10{,}000...0 = 0{,}000...0,$$

↓

одиниця не враховується.

Приклад. Записати числа A = + 0,1000110101101101 і B = – 0,0111010110010101 в оберненому коді:

$$[A]_д = 0{,}1000110101101101;$$

$$[B]_д = 1{,}1000101001101010 + 0{,}0000000000000001 =$$

$$= 1{,}1000101001101011.$$

Для практичного застосування при проведенні обчислень і розрахунків може бути використане таке правило для запису доповняльного коду від'ємного числа: для запису доповняльного коду від'ємного числа необхідно провести інверсію усіх його розрядів, починаючи із старшого, до останньої одиниці в записі числа (не включаючи її), а в знаковому розряді записати одиницю.

Приклад. З застосуванням наведеного правила записати число B = – 0,0111010110010101 в оберненому коді:

$$[B]_д = 1{,}1000101001101011$$

↓

не інвертується.

Це правило безпосередньо витікає із порівняння абсолютних величин однакових чисел, представлених в оберненому і доповняльному кодах.

5 ВИКОНАННЯ АРИФМЕТИЧНИХ ОПЕРАЦІЙ ДОДАВАННЯ І ВІДНІМАННЯ В КОДАХ

5.1 Основні поняття

Важливою функцією багатьох цифрових пристроїв і МП є виконання арифметичних операцій додавання і віднімання. В зв'язку з цим в ЕОМ і МП виділяють спеціальний функціональний блок – арифметичний пристрій (АП), призначений для виконання операцій над числовими кодами. Числа, які беруть участь в арифметичних операціях, називаються операндами.

Для позиційних систем числення із звичайною вагою всі допустимі числа є поліномами із степенями p (основи системи числення). Отже, всі арифметичні дії в цьому випадку виконуються за правилами алгебраїчного додавання поліномів.

Додавання двійкових чисел виконується у відповідності з правилами, визначеними у розділі 2.2 (таблиця 2.1). Але необхідно врахувати, що фактично у порозрядному додаванні приймають участь не дві, а три цифри. Третім доданком є цифра перенесення з сусіднього молодшого розряду (див. (2.8)). Таблиця додавання у цьому випадку буде мати вигляд, представлений у табл.5.1.

Кожна сума у таблиці 5.1 записана двома розрядами, молодший з яких є безпосередньо сума у поточному i – му розряді, а старший – цифра перенесення у наступний старший розряд p_{i+1} (значення цього розряду продубльовано у останньому стовпці таблиці 5.1).

Таблиця 5.1 – Додавання

p_i	a_i	b_i	S_i	p_{i+1}
0	0	0	00	0
0	0	1	01	0
0	1	0	01	0
0	1	1	10	1
1	0	0	01	0
1	0	1	10	1
1	1	0	10	1
1	1	1	11	1

При обчисленні суми двох чисел можливі два випадки: доданки мають однакові знаки; доданки мають різні знаки. У відповідності з цим алгоритми одержання суми для кожного з варіантів значно відрізняються

між собою. Так, спрощений алгоритм додавання двох чисел з однаковими знаками можна визначається таким чином:

- додати два числа;
- сумі присвоїти знак одного із доданків.

Алгоритм одержання алгебраїчної суми можна записати таким чином:

- порівняти знаки доданків і, якщо вони однакові, то виконувати додавання за першим алгоритмом;
- порівняти доданки за абсолютною величиною, якщо знаки доданків різні;
- якщо є необхідність, переставити числа місцями (щоб віднімати від більшого менше);
- провести віднімання двох чисел;
- результату присвоїти знак більшого доданку.

З цього випливає, що перший алгоритм значно простіший другого. Отже, бажано перетворити від'ємні числа таким чином, щоб операцію віднімання замінити додаванням, тобто виконати додавання двох чисел таким чином:

$$X - Y = X + (-Y),$$

$$-X - Y = (-X) + (-Y).$$

Для цього необхідно зобразити додатні та від'ємні числа єдиним натуральним кодом, що можливо досягти при використанні систем числення з додатньою основою та симетричною базою або з від'ємною основою та невід'ємною базою. При використанні однорідних позиційних систем числення з додатньою основою та невід'ємною базою натуральним кодом можна представити тільки додатні числа та нуль, тобто в цьому випадку не існує єдиного натурального коду для додатніх та від'ємних чисел. Це приводить до того, що проблему представлення чисел зі знаком приходиться вирішувати на рівні слів за допомогою спеціальних кодів: прямого, оберненого та доповняльного, визначення яких наведено у главі 4.

5.2 Виконання операцій алгебраїчного додавання в прямому коді

Правила додавання чисел в прямому коді не відрізняються від звичайних правил додавання, наведених вище. Тобто, якщо числа мають однаковий знак, то їх числові розряди додаютьсяя, а сумі присвоюється знак одного (любого) з доданків. Якщо розряди мають різні знаки, то процедура виконання операції ускладнюється (добавляється операція порівняння модулів доданків). Після цього з числових розрядів більшого за модулем числа віднімається меньше за модулем число, а сумі присвоюється знак більшого за модулем числа. При цьому знакові розряди чисел не беруть участі у виконанні операції віднімання, оскільки вони не мають ваги.

Наступні приклади ілюструють чотири основні варіанти додавання чисел у прямому коді при виконанні умови $|X| >> |Y|$ і $|X|+|Y| < 1$.

Завдання для самоконтролю

Варіант 1. $X > 0;\ Y > 0;\ Z > 0;$

$[X]_{пр} = X;\ [Y]_{пр} = Y;\ [Z]_{пр} = X + Y.$

Приклад 1.

$X = 0{,}10101010,\ Y = 0{,}00011101,\ [Z]_{пр} = ?$

$$[Z]_{пр} = X + Y = +\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \end{cases}.$$
$$0{,}11000111$$

Варіант 2. $X > 0;\ Y < 0;\ Z > 0;$

$$[X]_{пр} = X;\ [Y]_{пр} = 1 + |Y|;\ [Z]_{пр} = X - |Y|.$$

Приклад 2.

$X = 0{,}10101010,\ Y = -0{,}00011101,\ [Z]_{пр} = ?$

$$[X]_{пр} = 0{,}10101010;\ [Y]_{пр} = 1{,}00011101;$$

$$[Z]_{пр} = X - Y = -\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \end{cases}$$
$$0{,}10001101$$

Варіант 3. $X < 0;\ Y > 0;\ Z < 0;$

$$[X]_{пр} = 1 + |X|;\ [Y]_{пр} = Y;\ [Z]_{пр} = 1 + (|X| - |Y|).$$

Приклад 3.

$$X = -0{,}10101010,\ Y = 0{,}00011101,\ [Z]_{пр} = ?$$
$$[X]_{пр} = 1{,}10101010;\ [Y]_{пр} = 0{,}00011101;$$

$$[Z]_{пр} = 1 + (|X| - |Y|) = 1 + \left(-\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \end{cases} \right) = 1{,}10001101.$$
$$0{,}10001101$$

Варіант 4. $X < 0;\ Y < 0;\ Z < 0;$

$$[X]_{пр} = 1 + |X|;\ [Y]_{пр} = 1 + |Y|;\ [Z]_{пр} = 1 + |X| + |Y|.$$

Приклад 4.

$$X = -0{,}10101010,\ Y = -0{,}00011101,\ [Z]_{пр} = ?$$
$$[X]_{пр} = 1{,}10101010;\ [Y]_{пр} = 1{,}00011101;$$

$$[Z]_{пр} = 1 + |X| + |Y| = 1 + \left(+\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \end{cases} \right) = 1{,}11000111.$$
$$0{,}11000111$$

Таким чином, у прямому коді цифрову частину і знаковий розряд не можна розглдати, як єдине ціле. При цьому виконання операцій ускладнюється тим, що в структурі цифрового пристрою необхідно, крім

суматора, мати ще і пристрій для віднімання кодів. Ці недоліки настільки серьозні, що прямий код для виконання операцій арифметичного додавання практично не застосовується.

5.3 Виконання операцій алгебраїчного додавання в оберненому коді

В оберненому коді знаковий розряд і цифрова частина числа розглядаються, як єдине ціле, внаслідок чого цифрові пистрої виконують операції з від'ємними числами, які мають вигляд неправильних дробів. Правильний знак суми визначається автоматично в процесі додавання цифр знакових розрядів операндів (і одиниці перенесення з цифрової частини, якщо вона тає місце).

Характерною особливістю оберненого кода є наявність циклічного перенесення (якщо він виникає) із знакового розряду в молодший розряд цифрової частини, завдяки чому виконується корекція результату на 2^{-m}. Це зумовлено тим, що додавання проводиться за модулем $2-2^{-m}$.

Загальні правила додавання двійкових операндів, представлених у формі з фіксованою комою (правильні дроби), в оберненому коді можна розглянути на підставі застосування доповнення до двох і доповнення до одиниці (див. розділ 3). Розгляд будемо вести для двох основних варіантів:

а) $Z = X - Y$.

Обчислимо цей вираз із застосуванням доповнень.

Як відомо, для правильних двійкових дробів

$$A^o = \begin{cases} 0\cdot 2^0 + |A| \quad \textit{для додатніх чисел,} \\ 1\cdot 2^0 + |\overline{A}| = 1\cdot 2^0 + 2^0 - 2^{-m} - |A| \quad \textit{для від'ємних чисел.} \end{cases}$$

Тоді

$$X^o + (-Y)^o = 0\cdot 2^0 + |X| + 1\cdot 2^0 + 2^0 - 2^{-m} - |Y| = 1\cdot 2^0 + 2^0 - 2^{-m} + \Delta_1,$$

де $\Delta_1 = |X| - |Y|$.

Розглянемо такі три випадки:

1) Якщо $\Delta_1 < 0$, то $2^0 - 2^{-m} - |\Delta_1| < X_{\max} = Y_{\max}$.

Це означає, що не виникає перенесення до знакового розряду, а, також, із знакового розряду в молодший розряд результату (циклічного перенесення).

Результат операції є від'ємний і записується в оберненому коді так:

$$X^o + (-Y)^o = 1\cdot 2^0 + 2^0 - 2^{-m} - |\Delta_1| = 1\cdot 2^0 + |\overline{\Delta_1}| = Z_o.4.$$

2) Якщо $\Delta_1 > 0$, то $2^0 - 2^{-m} + |\Delta_1| > X_{\max} = Y_{\max}$.

Це означає, що виникає перенесення до знакового розряду, а, також, із знакового розряду в молодший розряд результату. Додаючи перенесення із знакового розряду до отриманого результату з вагою 2^{-m}, отримаємо додатній результат, який записується в оберненому коді так:

$X^o + (-Y)^o = |\Delta_1| = 0 \cdot 2^0 + |\Delta_1|.$

3) Якщо $\Delta_1 = 0 \ \ (|X| = |Y|)$, то отримаємо результат:

$X^o + (-Y)^o = 1 \cdot 2^0 + 2^0 - 2^{-m} = 1.111...1$ – один із варіантів визначення нуля в оберненому коді.

б) $Z = -X - Y.$

Обчислимо в оберненому коді:

$$(-X)^o + (-Y)^o = 1 \cdot 2^0 + (2^0 - 2^{-m} - |X|) + 1 \cdot 2^0 + (2^0 - 2^{-m} - |Y|).$$

Вираз

$$(2^0 - 2^{-m} - |X|) + (2^0 - 2^{-m} - |Y|) =$$
$$2^0 + \{2^0 - 2^{-m} - (|X|) + |Y|\} - 2^{-m} =$$
$$= 2^0 + \{2^0 - 2^{-m} - \Delta_2\} - 2^{-m} > \ X_{\max} = Y_{\max},$$

де $\Delta_2 = |X| + |Y|.$

Це означає, що виникає перенесення до знакового розряду, а, також, із знакового розряду. Додаючи перенесення із знакового розряду до отриманого результату з вагою 2^{-m}, отримаємо правильний від'ємний результат, який записується в оберненому коді так:

$$(-X)^o + (-Y)^o = 1 \cdot 2^0 + 2^0 - 2^{-m} - \Delta_2 = 1 \cdot 2^0 + \overline{\Delta}_2 = Z^o.$$

Отримані результати дають право зформулювати таке правило додавання двох чисел в оберненому коді:

Сума двох чисел в оберненому коді дорівнює сумі їх обернених кодів з урахуванням знакових розрядів. У випадку виникнення перенесення із знакового розряду одиниця перенесення додається до результату, як одиниця молодшого розряду.

Наступні приклади ілюструють чотири основні варіанти додавання чисел у оберненому коді при виконанні умови $|X| > |Y|$ і $|X| + |Y| < 1.$

1) $X > 0; \ \ Y > 0; \ \ Z > 0;$

Приклад 1.

$X = 0{,}10101010, \ Y = 0{,}00011101, \ [Z]^o = ?$

$$[Z]^o = +\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \\ 0{,}11000111 \end{cases}.$$

У звязку з тим, що обернений код додатніх чисел співпадє з записом самих чисел, оприманий результат співпадає з результатом додавання цих чисел у прямому коді.

2) $X > 0; \ \ Y < 0; \ \ Z > 0;$

Приклад 2.

$X = 0{,}10101010,\ Y = -0{,}00011101,\ [Z]^{o} = ?$

$$[Z]^{o} = +\begin{cases} 0{,}10101010 \\ \underline{1{,}11100010} \end{cases}$$

$$10{,}10001100$$

$$\underline{\downarrow \_\_\_\_\_ +1}$$

$$0{,}10001101$$

Виникає перенесення в знаковий розряд і із знакового розряду. Необхідно провести корекцію результату.

3) $X < 0;\ \ Y > 0;\ \ Z < 0;$

Приклад 3.

$X = -0{,}10101010,\ Y = 0{,}00011101,\ [Z]^{o} = ?$

$$[Z]^{o} = +\begin{cases} 1{,}01010101 \\ \underline{0{,}00011101} \end{cases}$$

$$1{,}01110010$$

Сума від'ємна. Перенесення до знакового розряду немає. Корекція не потрібна.

4) $X < 0;\ \ Y < 0;\ \ Z < 0;$

Приклад 4.

$X = -0{,}10101010, Y = -0{,}00011101,\ [Z]^{o} = ?$

$$[Z]^{o} = +\begin{cases} 1{,}01010101 \\ \underline{1{,}11100010} \end{cases}$$

$$11{,}00110111$$

$$\underline{\downarrow \_\_\_\_\_ +1}$$

$$1{,}00111000$$

Сума від'ємна. Виникає перенесення в знаковий розряд і із знакового розряду. Необхідно провести корекцію результату.

Очевидно, що у всіх розглянутих випадках додавання операнди можна поміняти місцями, і при цьому результат не зміниться.

Таким чином, обернений код досить зручний для виконання операцій алгебраїчного додавання, якщо врахувати простоту переведення від'ємних чисел із прямого кода в обернений і навпаки.

5.4 Виконання операцій алгебраїчного додавання в доповняльному коді

Як і в оберненому коді, при використанні доповняль-ного коду операція віднімання замінюється операцією алгебраїчного додавання. При цьому знаковий розряд і цифрова частина числа розглядаються, як єдине

ціле, тобто з від'ємними числами цифровий пристрій оперує, як з неправильними дробами. Правильний знак суми формується автоматично в процесі додавання вмісту знакових розрядів операндів і одиниці перенесення з цифрової частини, якщо вона має місце.

Використаємо визначення доповнення до двох і доповнення до одиниці для форулювання загальних правил додавання двійкових операндів, представлених у формі з фіксованою комою (правильні дроби), в доповняльному коді. Розгляд будемо вести для двох основних варіантів:

а) $Z = X - Y.$

Обчислимо цей вираз із застосуванням доповнень.

Як відомо, для правильних двійкових дробів

$$A^{o} = \begin{cases} 0 \cdot 2^{0} + |A| \quad \textit{для додатніх чисел}, \\ 1 \cdot 2^{0} + \left|\overline{\overline{A}}\right| = 1 \cdot 2^{0} + 2^{0} - |A| \quad \textit{для від'ємних чисел}. \end{cases}$$

Тоді

$$X^{\partial} + (-Y)^{\partial} = 0 \cdot 2^{0} + |X| + 1 \cdot 2^{0} + 2^{0} - |Y| = 1 \cdot 2^{0} + 2^{0} + \Delta_1,$$

де $\Delta_1 = |X| - |Y|.$

Розглянемо два такі випадки:

1) Якщо $\Delta_1 \geq 0$, то $2^{0} + |\Delta_1| > X_{\max} = Y_{\max}.$

Це означає, що виникає перенесення до знакового розряду, і також, із знакового розряду. З метою отримання правильного результату необхідно не враховувати (відкинути) перенесення із знакового розряду. Тоді

$$X^{\partial} + (-Y)^{\partial} = 0 \cdot 2^{0} + \Delta_1,$$

2) Якщо $\Delta_1 < 0$, то $2^{0} - |\Delta_1| \leq X_{\max} = Y_{\max}.$

Це означає, що не виникає перенесення до знакового розряду, а, також, із знакового розряду.

Результат операції є від'ємний і записується в доповняльному коді так:

$$X^{\partial} + (-Y)^{\partial} = 1 \cdot 2^{0} + 2^{0} - |\Delta_1| = 1 \cdot 2^{0} + \left|\overline{\overline{\Delta_1}}\right| = Z^{\partial}.$$

б) $Z = -X - Y.$

Обчислимо в доповняльному коді:

$$(-X)^{\partial} + (-Y)^{\partial} = 1 \cdot 2^{0} + (2^{0} - |X|) + 1 \cdot 2^{0} + (2^{0} - |Y|).$$

Вираз

$$(2^{0} - |X|) + (2^{0} - |Y|) = 2^{0} + \{2^{0} - (|X|) + |Y|\} =$$
$$= 2^{0} + \{2^{0} - \Delta_2\} > X_{\max} = Y_{\max}.$$

Це означає, що виникає перенесення до знакового розряду, і також, із знакового розряду.

Результат операції є від'ємний і записується в доповняльному коді так:

$$(-X)^{\partial} + (-Y)^{\partial} = 1 \cdot 2^{0} + 2^{0} - |\Delta_1| = 1 \cdot 2^{0} + \left|\overline{\overline{\Delta_1}}\right| = Z^{\partial}.$$

Отримані результати дають право зформулювати таке правило додавання двох чисел в доповняльному коді:

Наступні приклади ілюструють чотири основні варіанти додавання чисел у доповняльному коді при виконанні умови $|X| > |Y|$ і $|X|+|Y| < 1$.

1) $X > 0;\ Y > 0;\ Z > 0;$

Приклад 1.

$$[Z]^{\rho} = +\begin{cases} 0{,}10101010 \\ \underline{0{,}00011101} \end{cases}.$$
$$0{,}11000111$$

У звязку з тим, що доповняльний код додатніх чисел співпадє з записом самих чисел, оприманий результат співпадає з результатом додавання цих чисел у прямому коді.

2) $X > 0;\ Y < 0;\ Z > 0;$

Приклад 2.

$X = 0{,}10101010,\ Y = -0{,}00011101,\ [Z]^{\rho} = ?$

$$[Z]^{\rho} = +\begin{cases} 0{,}10101010 \\ \underline{1{,}11100011} \end{cases}$$

відкинути $\leftarrow 10{,}10001101$

Виникає перенесення в знаковий розряд і із знакового розряду. Одиницю перенесення із знакового розряду необхіднр відкинути.

3) $X < 0;\ Y > 0;\ Z < 0;$

Приклад 3.

$X = -0{,}10101010,\ Y = 0{,}00011101,\ [Z]^{\rho} = ?$

$$[Z]^{\rho} = +\begin{cases} 1{,}01010110 \\ \underline{0{,}00011101} \end{cases}$$
$$1{,}01110011$$

Сума від'ємна. Перенесення до знакового розряду немає. Перенесення із знакового розрядк теж немає. Результат правильний.

4) $X < 0;\ Y < 0;\ Z < 0;$

Приклад 4.

$X = -0{,}10101010, Y = -0{,}00011101,\ [Z]^{\rho} = ?$

$$[Z]^{\rho} = +\begin{cases} 1{,}01010110 \\ \underline{1{,}11100011} \end{cases}$$

відкинути $\leftarrow 11{,}00111001$

Сума від'ємна. Одиницю перенесення із знакового розряду необхідно відкинути.

Таким чином, у всіх випадках знак суми формується автоматично в результаті застосування загальних правил порозрядного додавання знаковиз і цифрових розрядів операндів Очевидно, що у всіх розглянутих випадках додавання операнди можна поміняти місцями, і при цьому результат не зміниться.

6 ДВІЙКОВО-ДЕСЯТКОВА АРИФМЕТИКА

6.1. Додавання в прямих Д-кода

В двійковому поданні десяткового числа, що кодувалося кожна десяткова цифра зображається тетрадою двійкових символів $\alpha_i = \alpha_4^i \alpha_3^i \alpha_2^i \alpha_1^i$, де α, □ десяткова цифра 1-го розряду; α_j^i □ двійкова цифра 1-ї тетради. Отриманий таким чином десятковий код, що кодувався двійковими символами, для більш короткої назви вживають Д-кодом.

Є деяка множина Д-кодів. Вона зумовлюється наявністю всього 10 дозволених з 16 можливих комбінацій, що припускає тетрада. Наявність заборонених комбінацій в Д-кодах відрізняє їх від звичайних позиційних систем счислення, в яких всі комбінації □ дозволені. З всієї безлічі відомих Д-кодів найбільше розповсюдження в обчислювальній техніку отримали код Д1 прямого заміщення (система 8421) і код Д2 з надлишком 3 (система 8421+3).

Із-за заборонених комбінацій, при додаванні чисел в будь-якому з Д-кодів виникають необхідність в корекції результату і труднощі в формуванні десяткового переносу в наступну тетраду. Особливості додавання чисел в кожному з Д-кодів різноманітні, тому розглянемо їх окремо. При цьому будемо вважати, що задані числа $A=a_n a_{n-1}...a_1 a_0$ і $B=b_n b_{n-1}...b_1 b_0$, де a_i, b_i □ двійково-кодовані десяткові цифри (тетради). Необхідно отримати:

$A + B = C = c_{n+1} c_n ... c_1 c_0$, Причому

$$C_i=a_i+b_i+П_{i-1}-П_i p;\ c_{n+1}=П_n$$

де $П_i = \{0, 1\}$, $П_{i-1} = \{0, 1\}$ □ десяткові переноси; $p = 10$ □ підстава системи счислення.

Бо найбільше десяткове однорозрядне число рівно 9, то з урахуванням переносу в даний розряд, значення результату розрядного складання лежить в межах від 0 до 19. При цьому одиниця в другому розряді являє собою десятковий перенос в наступну тетраду, а сума одержується в двійковому коді, відмінному від необхідного двійково-десяткового представлення,тобто Вона вимагає корекції.

Код Д1

При додаванні чисел в коді Д1 можуть виникнути наступні випадки:

1 Якщо $a_i + b_i + П_{n-i} < 10$, то при виконанні дій над розрядами тетради по правилам двійкової арифметики відразу одержується правильний результат.

1 Якщо $a_i + b_i + П_{n-i} \geq 10$, то виникає десятковий перенос. Тому сума в даній тетраді повинна бути рівна:

$$a_i+b_i+П_{i-1}-П_i \bullet 10,$$

де $П_i = 1$.

При цьому признаком невірного результату є в одному випадку

виникнення потетрадного переносу $П_i'$ = 16, В другому-поява забороненої комбінації, якщо $15 \geq a_i + b_i + +П_{i-i} \geq 10$. В будь-якому з цих випадків необхідно скоректувати результат в даній тетраді введенням поправки + 0110, що призведе до виникненню потетрадного переносу і в другому випадку. Корекція зумовлена тим, що кожний перенос виносить з собою з даної тетради 16 одиниць, а приносить в наступну тільки 10 одиниць.

Приклад. Скласти тетради a_i = 1000 і b_i = 1001 при $П_{i-1}$ = 1.

$$c_i' = a_i + b_i + П_{i-i} = 10\ 010.$$

Так як $П_i$== 1, вимагається корекція результату

$$c_i = 0010 + 0110 = 1000;\ \ П_i = П_i' = 1.$$

Приклад. Скласти a_i = 1000 і b_i = 0110 при $П_{i-1}$ = 1.

$$c_i' = a_i + b_i + П_{i-i} = 1111.$$

Так як величина c_i' = 1111 належить до заборонених комбінацій, те необхідно ввести поправку виду 0110:

$$c_i = 1111+0110 = 10101,\ т.\ е.\ П_i = 1,\ c_i = 0101.$$

Таким чином, якщо в i-й тетраді сума цифр з переносом з (i-1)-й тетради менш 10, то додавання робиться без поправок; якщо ж сума цифр з переносом рівна або більше 10, те відбувається корекція результату тетради введенням поправки +0110, а при цьому перенос, що виникає додається до вмісту (i+1)-й тетради.

При цьому, якщо в декількох тетрадах, починаючи з (i+1)-й, розрядна сума рівна 1001, то перенос призведе до формування забороненої комбінації в (i+1)-й тетраді. В результаті цього зажадається корекція, що призведе до забороненої комбінації в (i+2)-й тетраді і т. п. Отже, із-за послідовного розповсюдження потетрадних переносів час додавання в коді Д1 складе в гіршому випадку n тактів, де n □ кількість тетрад. Звичайно схеми будують таким чином, щоб перенос, що виникає при доданні тетрадної поправки, минав скрізь тетради, в яких попередня сума рівна $9_{10}=1001_2$ і скидав їх в 0. При цьому сума завжди формується за два такту.

Приклад. Скласти числа

$A = 248_{10} = 0010\ \ 0100\ \ 1000$ і $B = 349_{10} = 0011\ \ 0100\ \ 1001$

0010 0100 1000
0011 0100 1001
0101 1001← 0001
0110
0101 1001 0111: $597_{10}=248_{10}+348_{10}$

Стрілкою показані одиниці потетрадних переносів.

Код Д2

При додаванні чисел в коді Д2 можливі наступні випадки с врахуванням того, що $a_i' = a_i+3$; $b_i' = b_i + 3$, де a_i' и b_i'□ тетради для коду Д2.

Якщо $a_i'+ b_i+П_{i-1}-1 \leq 15$, то $C_i=(a_i+ b_i+П_{i-1}+3)+3 = C_i + +6 = C_i' + 3$,

тобто результат необхідно скоректувати на величину (-0011). Якщо ж

a_i+ b_i+$П_{i-1}$ - 1>15, то C_i=(a_i+ b_i+$П_{i-1}$+6) +10 - - 16 = C_i' - 3. Тут виникає десятковий перенос, що при переході в старшу тетраду міняє свою вагу з 16 на 10. Тому в цьому випадку вимагається поправка результату на величину +0011.

Суматори не роблять операцію виднімання, тому поправка (□ 3_{10}) = (□ $0011)_2$ здійснюється доданням в дану тетраду числа $(13)_{10}$ = $(1101)_2$ і блокуванням при цьому переносу, що виникає з даної тетради в старшу, яка має значення $(16)_{10}$ □ $(10000)_2$:

(□3 = +13 □ 16)$_{10}$ или (□0011 = +1101□10000)$_2$.

Приклад. Задані в коді Д2

A = 46 = 0111 1001; B == 37 = 0110 1010.

Знайдемо суму

0111 1001
0110 1010
1110 0011
1101 0011
1 1011 0110 = 83
не реалізується

При додаванні в коді Д2 не виникає проблеми наскрізного переносу, і операція додавання виконується за два такти. Це пояснюється тим, що якщо в декількох тетрадах сума до додання поправки рівна $9_{10} = 1111_2$, то при надходженні переносу з (i □ 1)-й тетради в 1-ю, її вміст скинеться в 0 і перенос піде в (i+1)-у тетраду в цьому ж такті складання. В 1-у ж тетраду в другому такті складання буде додана поправка +0011. Тому корекція результату здійснюється потетрадно з блокуванням кіл потетрадних переносів. Таким чином, в коді Д2 завжди робиться корекція проміжного результату, отриманого шляхом додавання цифр доданків по правилам двійкової арифметики. При цьому, якщо при додаванні і-х тетрад не виникає переносу, тобто $П_i$ = 0, то поправка рівна -0011. Якщо ж виникає потетрадний перенос $П_i$ = 1, то поправка в 1-й тетраді дорівнює +0011.

6.2 Додавання чисел в інверсних Д-кодах

Д-коди можуть бути представлені в розрядній сітці машини в формі або з фіксованої, або з що плаваючою крапкою. При цьому негативні числа можуть бути представлені в прямому, зворотному і додатковому кодах. Тому, якщо

A = 0, $a_1a_2...a_n$, де a_i— тетради, то

$$[A]_{пр} = a_1a_2\ldots a_n;$$
$$[A]_{о} == \overline{\overline{a}}_1\overline{\overline{a}}_2...\overline{\overline{a}}_n;$$
$$[A]_{д} == \overline{\overline{a}}_1\overline{\overline{a}}_2...\overline{a}_n,$$

де a_i □ доповнення до p-1=9 в усіх тетрадах; $\overline{a}_n$ □ доповнення до p=10 в

молодшій тетраді.

Отже, $\overline{a}_n + a_i = p = 10$; $\overline{\overline{a}}_n + a_i = p - 1 = 9$. Зворотній код Д2 одержується простим інвертуванням цифр тетрад.

Приклад. Знайти зворотний і додатковий коди числа

A = -0.136 для коду Д2.

$$A_{Д2} = -0{,}0100\ \ 0110\ \ 1001;$$

$$[A_{Д2}]_о = 1{,}1011\ \ \ \ 1001\ \ 0110;$$

На основі співвідношення отримаємо додатковий код числа A:

$$[A_{Д2}]_д = 1{,}1011\ \ \ \ 1001\ \ 0111.$$

В коді Д1 пряме інвертування цифр тетрад означає отримання доповнення до $2^4 - 1 = 15$. Тому для отримання зворотного коду числа в коді Д1 в увсі тетради числа додають спочатку +0110, після чого роблять інвертування цифр тетрад.

Приклад. Для числа A= 0,148 = 0,0001 0100 1000 отримати зворотний і додатковий коди Д1.

$$0{,}0001\ \ 0100\ \ 1000$$
$$0110\ \ \ \ 0110\ \ 0110$$
$$0{,}0111\ \ 1010\ \ 1110.$$

Звідси $[A_{Д1}]_о = 1{,}10000101\ 0001;$

$[A_{Д1}]_д = 1{,}100001010010.$

Відзначимо, що додання 1 в молодший розряд при отриманні додаткового Д2 коду робиться значно складніше, ніж в коді Д1, бо ця операція виконується як звичайне додавання в Д2 коді. При додаванні в інверсних Д-кодах, як і при додаванні в прямих Д-кодах, необхідна корекція результату, що може здійснюватися або програмними, або апаратними засобами. На практиці звичайно застосовують другий засіб. При цьому в процесі виконання корекції ланцюга міжтетрадного переносу в суматорах блокуються для коду Д2, а при додаванні в коді Д1 не блокуються.

Приклад. Скласти в зворотному і додатковому кодах Д1 і Д2 числа

A = - 0,926 і B =+ 0,845.

Для коду Д1

A = - 0,1001 0010 0110 і B =+ 0,1000 0100 0101.

$$[A]_о = \ \ 1{,}0000\ \ \ \ 0111\ \ 0011$$
$$[B]_о = \ \ 0{,}1000\ \ \ \ 0100\ \ 0101$$
$$1{,}1000\ \ 1011\ \ 1000$$
$$+\ 0110$$
$$[C]_о = [A]_о + [B]_о = \ \ 0{,}1001\ \ \ \ 0001\ \ 1000$$
$$[C]_{пр} = \ 1{,}0000\ \ \ \ 1000\ \ 0001$$
$$[A]_д = \ \ 1{,}0000\ \ \ \ 0111\ \ 0100$$
$$[B]_д = \ \ 0{,}1000\ \ \ \ 0100\ \ 0101$$
$$1{,}1000\ \ 0100\ \ 0101$$
$$+\ 0110$$

$[C]_д$= 1,1001 0001 1001

$[C]_{пр}$= 1,0000 1000 0001

Для коду Д2

A= □ 0,1100 0101 1001

B=+ 0,1011 0111 1000

$[A]_о$= 1,0011 1010 0110

$[B]_о$= 0,1011 0111 1000

1,1111 0001 1110

+ 1101 0011 1101

$[C]_о$= 1,1100 0100 1011

$[C]_{пр}$= 1,0011 1011 0100

Негативні поправки вносяться в виді доповнення: 1101

Таким чином, код Д2, тобто код з надлишком 3, дозволяє надто просто формувати порозрядні доповнення до p □ 1 = 9, а також просто виконувати алгебраїчне додавання. Однак, як і код Д1, він вимагає корекції результату, що збільшує час реалізації операцій. Код Д2 володіє ще рядом корисних властивостей. Наприклад, цифри десяткової системи счислення, значення яких більше або рівні п'яті, відрізняються тим, що в старшому розряді двійково-десяткового коду завжди присутня одиниця. Завдяки цій властивості по значенню тільки одного старшого розряду тетради визначається необхідність округлення. Цією ж властивістю можна скористуватися для спрощення реалізації прискореного множення.

6.3 Зсув Д-кодів

При зсуві Д-коду числа на один десятковий розряд необхідно передбачити зсів двійкових цифр відразу на 4 розряду. Правила зсуву залишаються тими ж, що і для двійкових дробів, тобто зсув праворуч обмежується величиною допустимої помилки, яку можна зменшити шляхом округлення зсунутого числа. Для цього вводять додатковий десятковий розряд (ДДР). При цьому позитивні дроби округляють доданням числа 5 в ДДР, якщо цифра в ньому більше або рівна 5. Округлення негативного дробу, наведеного в зворотних Д-кодах, не робиться. Округленне в додаткових Д-кодах робиться тільки тоді, коли хвост числа більше 0,5 х 10^{-n}, де n □ число основних десяткових розрядів. Операція зсуву Д-коду ліворуч коректна до тих пір, доки не відбудеться переповнення розрядної сітки (табл.6.1).

При виконанні операцій множення і ділення Д-кодів нерідко виникає необхідність множення або ділення чисел на 2. Для множення Д1 коду на 2 необхідно зсунути задане число на один двійковий розряд ліворуч. Поправки вносяться як і при додаванні в коді Д1, тобто якщо в i-й тетраді в результаті зсуву виникла заборонена комбінація або з i-й тетради в (i+1)-ю була висунута одиниця, то до 1-ї тетради додається поправка 0110 і ланцюг

міжтетрадних переносів не блокується. В інших випадках корекція результату не робиться.

Приклад. Задане в коді Д1 число $A=0.159_{10}=0.0001\ 0101\ 1001$. Зсунути його на один двійковий розряд ліворуч

$$\begin{array}{l}\overleftarrow{A}=0,\quad 0010\ \ 1011\ \ 0010\\ \quad +\ 0110\ +\qquad 0110\\ \quad 0,0011\ \ 0001\ \ 1000\end{array}$$

Таким чином, $\overleftarrow{A}=0{,}0011\ 0001\ 1000=0{,}318_{10}$.

Приклад. Задане в коді Д1 число $A=0.15810=0.0001\ 0101\ 1000$. Зсунути його на один двочный розряд праворуч

$$\begin{array}{l}\overrightarrow{A}=0,\quad 0000\ \ 1010\ \ 1100\\ \quad +\ 1101\ +\qquad 1101\\ \quad 0,0000\ \ 0111\ \ 1001\end{array}$$

Таким чином, $\overrightarrow{A}==0{,}0000\ 0111\ 1001=0{,}079_{10}$.

При діленні коду Д1 на 2 необхідно зсунути його на один двійковий розряд праворуч і відняти поправку 0011 з тих тетрад, в які зсунулася одиниця з попередньої або додати поправку 1101 з блокуванням міжтетрадного переносу, що виникає.

При множенні коду Д2 на 2 необхідно зсунути його на один двійковий розряд ліворуч. Корекція вноситься як і при додаванні, тобто якщо з i-й тетради зсувається одиниця в (i+1)-ю, то до i-й тетради додається 0011, якщо ж одиниця в (i+1)-ю тетраду не зсувається, те до i-ї тетради додається 1101. При цьому ланцюги міжтетрадних переносів блокуються.

Приклад. Задане в коді Д2 число $A=0.159_{10}=0.0100\ 1000\ 1100$.

Зсунути його на один розряд ліворуч

$$\begin{array}{l}\overleftarrow{A}=0,\quad 1001\ \ 0001\ \ 1000\\ \quad +\ 1101\ \ 0011\ \ 0011\\ \quad 0,0110\ \ 0100\ \ 1011\end{array}$$

Таким чином, $\overleftarrow{A}=0{,}0110\ 0100\ 1011=0{,}318_{10}$.

При діленні коду Д2 на 2 необхідно зсунути його на один двійковий розряд праворуч і додати поправку згідно табл. 6.1.

Табл 6.1 одержується на основі наступних міркувань. Якщо цифра а, записана в даній тетраді, парна, тобто $a = 2\ b$, то її двійковий код буде непарним, бо він рівний $a + 3 = 2b + 3$. Якщо a □ непарне, тобто $a = 2b + 1$, то двійковий код в даній тетраді буде парним, бо він рівний $2b + 4$.

Таблиця 6.1 – Зсув Д-коду

Цифра, що зсувається в дану тетраду	Цифра, що зсувається з даної тетради	Поправка
—	0	$П_0 = 0001$
—	1	$П_1 = 0010$
0	0	$П_{00}=0110$
0	1	$П_{01}=0111$
1	0	$П_{10} = 1001$
1	1	$П_{11}= 1010$

В старшу тетраду нічого не зсувається, але з неї в сусідню праворуч тетраду зсувається або 0, або 1 Якщо зсувається 0, значить $a = 2b + 1$, тобто код в ній парний і рівний $2b + 4$. Після зсуву він стане рівний $b + 2$, а повинен бути рівним $b + 3$, отже, необхідна поправка на величину $П_0$= 0001. Якщо з старшої тетради зсувається 1, то а парне (а == $2b$), і після зрушення повинен бути код b+3. Зсувається ж код —1/2(a+3) =1/2(2b+3) = b+1+1/2. Тут 1/2 переходить у вигляді 1 в тетраду праворуч, а поправка, очевидно, буде рівна $П_1$=0010.

Розглянемо зсув будь-який тетради. Якщо в неї зсувається 0, то це означає, те в неї повинна попасти 1 з вагою 5 (10/2). Якщо з тетради зсувається 0, то в ній число непарно, тобто $a = 2b + 1$, а код його парний, тобто рівний $2b + 1 + 3 = 2b + 4$. Після зсуву в тетраді буде код b+2, а повинно бути $b + 3$. Тому поправка в цьому випадку складе $П_{00} = (5 + 1)_{10}$ = 0110. Якщо ж з даної тетради зсувається код 1, то в ній число парне, тобто $a = 2b$, а код непарний і рівний $2b + 3$. Після зсуву в тетраді отримаємо $(2b+3)½ = b+1+½$. При цьому ½ в вигляді 1 зсувається в сусідню праворуч тетраду, а в даній залишається код $b + 1$, то повинно бути $b + 3$. Звідси поправка дорівнює $П_{01} == (5+ + 2)_{10} = 0111$.

Якщо в дану тетраду зсувається код 1, те це означає, що в неї повинен попасти 0, бо десяткова цифра в попередній тетраді парна (цифра, а не її код!). Ця 1 має вагу 8, тому в поправку необхідно ввести негативну величину (-8). Якщо з даної тетради зсувається 0, то код в ній парний і рівний $2b + 4$. Після зсуву він рівний b+2, тобто результуюча поправка повинна скласти 1 □ 8 = □7. Негативні поправки звичайно вносяться в вигляді доповнень до 16, тому $П_{10}$= 1001. Якщо з даної тетради зсувається 1, значить а парне, тобто $a = 2b$, а код в ній непарний і рівний $2b + 3$. Після зсуву отримаємо код $b + 1 + ½$. При цьому ½ переходить в вигдяді 1 в сусідню праворуч тетраду, а в даній залишається $b + 1$, тобто необхідна поправка коду тетради на величину 2 □ 8 = = □6, що також задається в виді доповнення до 16, тобто

$П_{11}$ = 1010. Необхідно пам'ятати, що в коді Д2 при реалізації поправок ланцюги міжтетрадних переносів блокуються.

Якщо дана тетрада молодша, то праворуч двійкова цифра, що зсувається з неї буде являти собою інверсний код залишку від ділення числа на 2: якщо цифра коду залишку 0, то цифра залишку числа рівна 1 і навпаки.

Приклад. Задане в коді Д2 $A = 526_{10}$ = 1000 0101 1001, Зсунути його на один двійковий розряд праворуч

A= 0100 0010 1100 1
0001 0111 1010
0101 1001 0110

Таким чином, $\vec{A}$ = 0101 1001 0110 = 526 : 2 = 263_{10}.

6.4 Множнення чисел в Д-кодах

Множення чисел в Д-кодах зводиться до послідовного складання часткових добутків, одержуваних при множенні множеного на чергову цифру множика. При цьому множення супроводжується розшифровкою значення черговий *i*-й тетради множника, що представляється в вигляді $(\beta_4\beta_3\beta_2\beta_1)_i$, і зсувом множника на чотири розряди одразу.

Самим простим засабом розшифровки тетради є послідовне виднімання одиниці з значення тетради до отримання нуля і відповідно додання множеного в суматор на кожному такті. Бо при множенні множеного на тетраду можливо переповнення разрядної сітки суматора (внаслідок того що множене додається до суми часткових добутків стільки раз, скільки одиниць міститься в даному десятковому розряді множника), то в ньому необхідно передбачити додаткову тетраду для врахування переносів, що виникають. З чотирьох можливих засобів множення в двійково-десятковій системі доцільно застосовувати тільки один: множення молодшими розрядами множника зі зсувом суми часткових добутків праворуч. Множення звичайно робиться в прямому коді, тобто знак результату визначається сумою цифр співмножників по модулю 2.

Приклад. Помножити в прямому Д2 коді числа

A = -0,45; $[A]_{пр}$=1,0111 1000;
B =+0,24; $[B]_{пр}$= 0,0101 0111.

Знак результату визначається так же, як і в випадку двійкової системи: $1 \oplus 0 = 1$. Для виконання операції множення необхідний трьохтетрадний суматор. На рисунку 6.1 зображена операція множення.

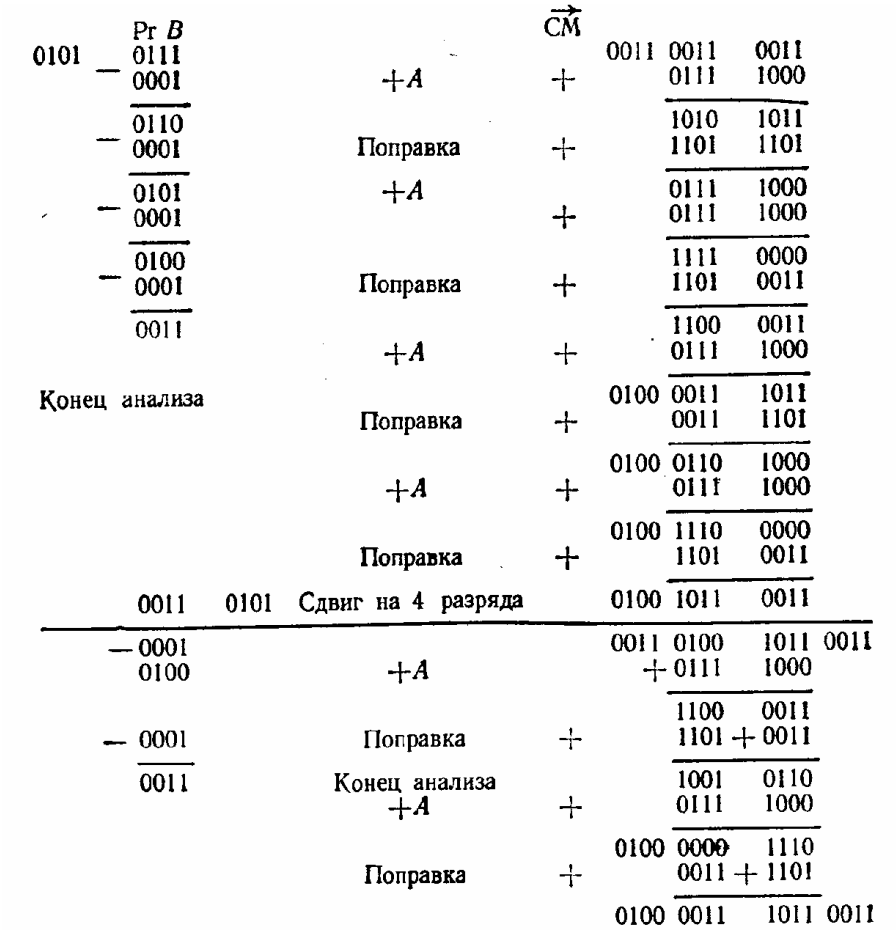


Рисунок 6.1 – Операція множення

Таким чином, $[A \bullet B]_{пр}$= 1,0100 0011 1011 0011.

Якщо співмножники мають по n десяткових розрядів, то в регістрі множеного повинно бути n розрядів (тетрад) праворуч від крапки, а в суматорі повинно бути n основних розрядів, одна тетрада переповненнь і один (n+1)-й додатковий десятковий розряд (ДДР), що напередвизначений для округлення результату (в коді Д2 для округлення достатньо мати один додатковий двійковий розряд). При необхідності зберегти всі 2n розрядів добутку праворуч від суматора повинен бути ще зсунутий регістр для молодших розрядів добутку. Регістр множника повинен мати n тетрад праворуч від крапки. Причому, якщо молодша тетрада цього регістру виконана в виді реверсивного лічильника, тобто лічильнка, який може додавати або віднімати одиницю, то операцію множення можна прискорити в середньому майже в два рази. В цьому випадку необхідно мати спеціальний розряд для запису 1, якщо при складанні в лічильнику-тетраді з'явиться код 10_{10}.

Такий склад обладнання пояснюється слідуючим чином. Якщо чергова цифра множника, яка знаходиться в молодшій тетраді (лічильникові) регістрі множника, рівна або менш 5_{10}, то робиться багаторазове додання множеного до суми часткових добутуів суматора. При кожному складанні

множеного віднімається 1 з лічильника. Такі дії виконуються до тих пір, доки на лічильникові не з'явиться код 0. Якщо ж чергова цифра множника рівна або більше 6_{10}, то робиться багаторазове віднімання (додавання в додатковому Д-коді) множеного з суми часткових добутків суматора. При кожному відніманні множеного до вмісту лічильника додається 1. Виднімання тривають до тих пір, доки в лічильникові не з'явиться код 1010. При цьому в спеціальний розряд записується 1.

По закінченню аналізу поточної тетради множника робиться зрушення на одну тетраду праворуч суми часткових добутків і множника. При цьому в лічильник регістру множника зсувається наступна тетрада множника, до якої додається вміст спеціального розряду, тобто 1, якщо вона була там записана. Операція множення закінчується зсувом множника і суми часткових творів. Всього повинно бути виконане n зрушень. На останньому зсуві в лічильник зсувається нуль множника. Якщо при цьому в нього додається 1 зі спеціального розряду, то операція множення закінчується доданням множеного до суми часткових добутків. В результаті одержується 2*n*-разрядний добуток.

Таким чином, описаний алгоритм дозволяє істотно скоротити час множення практично без модифікації складу обладнання. Якщо необхідний *n*-розрядний результат, то й твір округляється шляхом додання до ДДР, тобто до (*n*+1)-й тетради, числа 510.

Приклад. Задано $A=0{,}367_{10}$, $B=0{,}583_{10}$. Знайти добуток $C=AB$ і округлити до 3-х розрядів. Всі дії для простоти виконуємо в десятковому коді. Квадратною рамкою обведений лічильник в регістрі множника (Рг В), лічильник тактів (розрядів) множення (СЧТ) і спеціальний розряд (СПР). На суматору (СМ) самий лівий розряд (від крапки) двійковий (знаковий), а другий ліворуч (розряд переповнений) десятковий. Оперція множення зображена на рис.6.2.

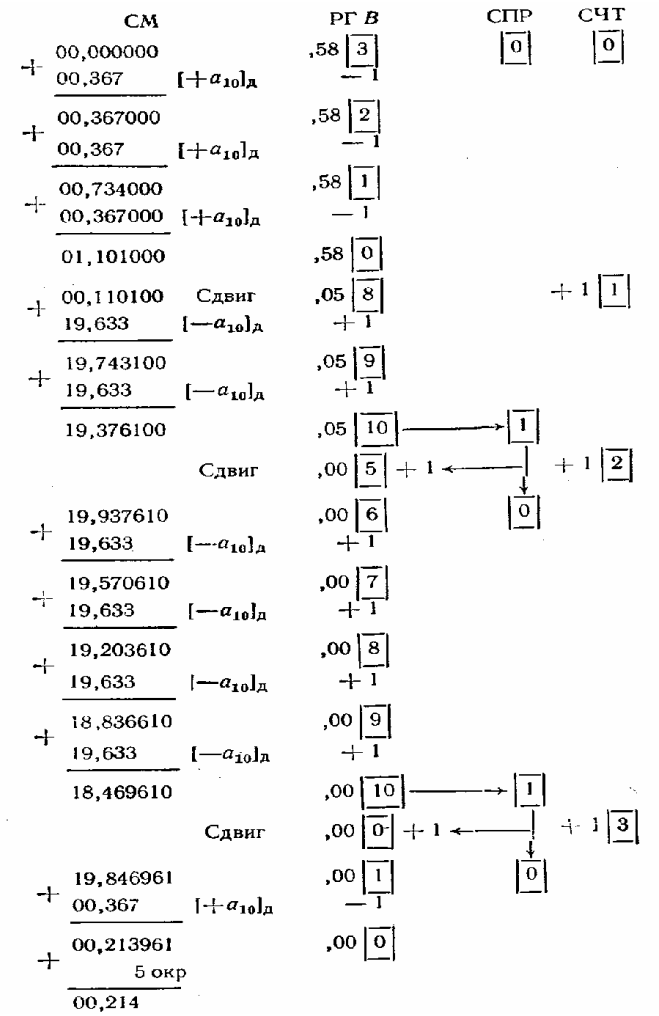


Рисунок 6.2 – Операція множення

Таким чином, $C = A \cdot B \approx 0{,}214_{10}$.

Подальшим розвитком описаного способу прискорення операції множення в Д-кодах є формування кратних множеного *A*. Наприклад, за наявності кратних ±2*A* і ±4*A* часний добуток формується слідуючим чином в залежності від поточної цифри множника В (табл. 6.2).

Таблиця 6.2 – Часнний добуток

Поточна цифра множника	Вид операції	Поточна цифра множника	Вид операції
0	0	5	+A+4A
1	+A	6	□4A
2	+2A	7	□A□2A
3	+A+2A	8	□2A
4	+4A	9	□A

При цьому, якщо попередня цифра В була більше п'яті, то дію на черговому кроку треба збільшити на +1А. Як видно з таблиці, в гіршому

випадку для даного способу зажадається всього 2 дії алгебраїчного додавання при отриманні одного часткового добутоку, а в середньому для цього необхідно 1, 2 додаваннь відніманнь. Спосіб дозволяє аналізувати чергову цифру множника простими апаратними засобами. Для його реалізації зажадається ще два регістри для зберігання 2А і 4А або комутатор для формування 4А і 2А шляхом зсуву А відповідно на 2 і 1 розряд

Інший прийом, що застосовується для прискореного множення, зводиться до наступних покрокових перетворень добутків:

$$AB = 2A\frac{B}{2},$$ якщо В-парне число;

$$AB = 2A\frac{B-1}{2} + A,$$ якщо В-непарне число.

Приклад. Задано $A = 35$, $B = 34$. Знайти $C = A \cdot B$.

35 • 34 = 2 • 35 • 34/2 = 70 • 17 = 140 • 8 + 70 = 280 • 4 + 70 = 560 • 2+70 = 1 120 • 1+70 = 1190.

Таким чином, $C=A \cdot B$= 1190.

При цьому подвоєння числа означає його зсув ліворуч, а ділення на 2 зсув праворуч. Бо двійкові ззуви робляться над двійковими кодами, то вимагається корекція тетрад на кожному кроці. Як бачимо, додавання роблять тільки при непарному множнику. Найбільша їхня кількість можлива при $B = 2_n$ 1 і дорівнює (n 1), а при $B = 2^n$ роблять тільки зсув множеного і множника.

Якісну оцінку ефективності розглянутих способів множення дасть порівняння числа додавань, що необхідні для отримання повного добутка по кожному з способів, наприклад, при B = 0,583:

1. Тільки складання множеного. Кількість додавань: 5 +8+3= 16.

2. Складання і виднімання множеного, що при цьому практично має вигляд $B = 1,\overline{4}\overline{2}3,$ де риска над цифрою означає віднімання. Тоді кількість додавань складе: 1+4+2+3 = 10.

3. Складання і виднімання кратних множених. Кількість додавань: 2+1+2 = 5.

4. Покрокове формування добутока шляхом двійкового зсуву множеного і множника. Кількість додавань: 1+1+1+1= 4.

Таким чином, якщо врахувати, що в реальних ЕОМ кількість розрядів значно більше трьох, то виграш в швидкодії при виконанні множення прискореними способами виявиться ще більш істотним.

6.5 Ділення чисел в Д-кодах

Ділення десяткових чисел звичайно виконується в прямих Д-кодах. Знак частки визначається так же, як і при двійковому діленні. Тому в подальшому розглянемо ділення позитивних десяткових дробів. З двох можливих способів ділення двійкових чисел для ділення десяткових чисел

доцільно застосовувати тільки спосіб при непорушному дільникові і зсуву залишків і частки ліворуч. При двійковому діленні в кожному такті одержувалася чергова цифра частки, що приймала одне з двох можливих значень (0 або 1). Для визначення цієї цифри дільник віднімався з діленого на нульовому кроці і зсунутого ліворуч на один розряд чергового залишку на наступних кроках. При десятковому діленні кількість віднімань дільника з зсунутого чергового залишку, як правило, більше 1, бо розряди частки можуть приймати значення від 0 до 9. Так же як і при множенні, на суматору ліворуч від крапки повинно бути п'ять двійкових розрядів: самий лівий знаковий, де фіксується знак залишку, і ще одна тетрада, в яку може зсуватися старша цифра поточного залишку при його зсуві ліворуч. Праворуч від крапки в суматорі повинна бути (n+1) тетрада, молодша з яких використовується тільки для округлення частки. В регістрі дільника праворуч від крапки повинно бути n тетрад. Регістр частки повинен бути з лівим зсувом і мати (n+1) тетраду. Причому молодша тетрада повинна являти собою реверсивний лічильник, в якому формується чергова десяткова цифра частки при виконанні чергового кроку ділення. Всіх циклів ділення повинно бути n+2, бо в першому циклі одержується цифра ліворуч від крапки, а праворуч від крапки необхідно сформувати n+1 цифру з урахуванням розряду округлення. Якщо на першому кроці цифра частки не рівна 0, то частки вийшла за межі розрядної сітки, тобто операція не коректна. При виконанні кожного кроку ділення до лічильника тактів додається 1, а операція завершується після виконання (n+2) циклів.

В вхідному стані діленее знаходиться в суматорі, дільник в регістрі дільника, а регістр частки встановлений в стан 0. Ділення починається з віднімання дільника з діленого на нульовому кроці і з ссунутих залишків на наступних кроках. Віднімання на кожному кроку замінюється додаванням в додатковому Д-коді і робиться до тих пір, доки не отримається негативний залишок. При цьому кожний раз при отриманні позитивного залишку додається одиниця в спеціальний лічильник, де накопичується чергова цифра частки. Після цього здійснюється зсув залишку на чотири двійкових розряди і додання до нього дільника до тих пір, доки не отримається позитивний залишок. Кількість додаванн (без останнього) є доповненням відповідної цифри частки до 9, яка заноситься в лічильник чергової цифри частки.

Таким чином, процес ділення складається з ряду послідовно перемінних циклів додавання і віднімання, що виконуються на суматорі, працюючому по правилам алгебраїчного додавання в відповідному Д-коді, і завершуються зсувом частки і залишків ліворуч на одну тетраду. При цьому в тетраду суматора ліворуч від крапки зсувається код з тетради праворуч від урапки, а вміст знакового розряду не міняється.

Приклад. Поділити число A = 0,154675 на число B = 0,55. Так як $A < B$, на нульовому кроку залишок повинен бути негативним, тобто операція коректна. Далі розглянемо тільки цифрову частину чисел:

1-ий крок зображено на рис. 6.3

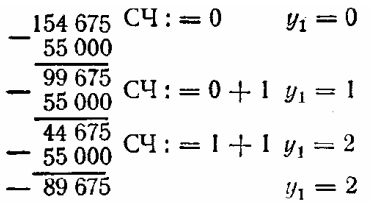


Рисунок 6.3 – 1-й крок операції ділення

Залишок < 0. Зсув частки ліворуч і дільника праворуч (або залишку ліворуч). 2-ий крок зоражено на рис. 6.4

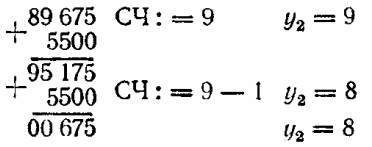


Рисунок 6.4 – 2-й крок операції ділення

Залишок > 0. Зсув частки ліворуч і дільника праворуч на 4 двійкових розряду. 3і-й крок зображено на рис. 6.5:

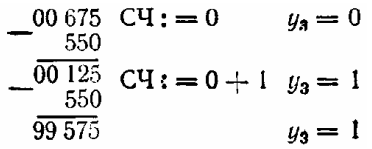


Рисунок 6.5 – 3-ій крок операції ділення

Залишок < 0. Зсув частки ліворуч і дільника праворуч на 4 двійкових розрядів. І так далі. Таким чином $Y = A/B \approx 0{,}281$.

По іншому способу при діленні чисел, заданих в Д-кодах, результат одержується в двійковій системі счислення.

Припустимо, при діленні A на B отримана частки у вигляді $Y = 0, y_1y_2\ldots y_n$.

Тоді $A = B(y_1 2^{-1} + y_2 2^{-2} + \ldots + y_n 2^{-n}) + R_n$ де R_n □ залишок від ділення.

Покладемо $R_n = 0$ і $y_1 = 1$, $y_2 = y_3 = \ldots = 0$. Тоді залишок на першому кроці $R_1 = A$ □ $2^{-1}B$.

Якщо $R_1 \geq 0$, то $y_1 = 1$; якщо $R_1 < 0$, то $y_1 = 0$. В наступному випадку відновлюється попередній позитивний залишок. Після цього приймемо, що $y_2 = 1$, а інші $y_i = 0$ і т. д. Залишок на будь-якому кроці складе $R_i = A - B\sum_{i-1}^{n} y_i 2^{-i}$. Слід врахувати, що при числах, наведених в Д-коді,

необхідно вводити поправки. Тому для зберігання дільника бажано мати самостійний суматор, в якому при передачі одиниці з однієї тетради в іншу автоматично вноситься поправка.

Приклад. Задані A = 0,2425; B = 0,5200. Знайти $Y = A/B$. Рішення заданого прикладу зображено на рис. 6.6.

Делитель B на t-м шаге	Сумматор (СМ)	Примечание	Цифры
0,5200	0,2425	$B \cdot 2^{-1}$; СМ: = [СМ] — $B \cdot 2^{-1}$	
0,2600	— 0,2600		
	9,9825	$R < 0$	$y_1 = 0$
	+ 0,2600	Восстановление R_1 т. е.	
	0,2425	$R_1 + B \cdot 2^{-2}$	
0,1300	— 0,1300	$B \cdot 2^{-2}$; СМ: = [СМ] — $B \cdot 2^{-2}$	
	0,1125	$R_2 > 0$	$y_2 = 1$
0,0650	— 0,0650	$B \cdot 2^{-3}$; $R_2 - B \cdot 2^{-3}$	
	0,0475	$R_3 > 0$	
0,0325	— 0,0325	$B \cdot 2^{-4}$; $R_3 - B \cdot 2^{-4}$	$y_3 = 1$
	0,0150	$R_4 > 0$	
	0,0150	$B \cdot 2^{-5}$; $R_4 - B \cdot 2^{-5}$	
0,0162	— 0,0162	$R_5 < 0$	$y_4 = 1$
	9,9938		
	+ 0,0162	Восстановление R_5	$y_5 = 0$
	0,0150	$B \cdot 2^{-6}$; $R_4 - B \cdot 2^{-6}$	
0,0081	— 0,0081	$R_6 > 0$	$y_6 = 1$
	0,0069		
	и т. д.		

Рисунок 6.6 – Рішення прикладу

Таким чином, Y= 0,011101.

6.6 Переведення чисел в Д-кодах

Розглянемо переведення цілих десяткових чисел, наведених в Д-коді, в двійкову систему счислення.

Нехай задане десяткове число $A = a_n a_{n-1} \ldots a_2 a_1$, де a_i □ десяткова цифра, що повинна бути представлена в Д-коді в вигдяді $a_i = (\alpha_4^i \alpha_3^i \alpha_2^i \alpha_1^i)$.

Використовуючи рівність 10 = 8+2 = (2^3 + 2^1), будь-яке десяткове ціле число можна записати

$$A=(\ldots(\alpha_4^n\alpha_3^n\alpha_2^n\alpha_1^n)(2^3+2^1) + \alpha_4^{n-1}\alpha_3^{n-1}\alpha_2^{n-1}\alpha_1^{n-1})\,(2^3+2^1)+$$
$$+\ldots+\alpha_4^2\alpha_3^2\alpha_2^2\alpha_1^2)\,(2^3+2^1)+\alpha_4^1\alpha_3^1\alpha_2^1\alpha_1^1.$$

Множення на 2^k означає зсув двійкового коду на k розрядів ліворуч. Отже, переведення зводиться до зсуву відповідних тетрад і їхнього наступного складання.

Приклад. Задане A_{10} = 254 або в коді Д1 A = 0010 0101 0100. Знайти A_2.

A_2 = (0010. (2^3 + 2^1)+0101) (2^3 + 2^1) + 0100 = (10000 + 0100+ +0101) (2^3 + 2^1) + 0100 = 10. 000. 000+100. 000+1. 001. 000+ +10.010+0. 100=11.

101.000+10. 110=11. 111. 110.

Для переведення правильних дробів можна застосувати наступний прийом. Задану *k*-розрядний десятковий дріб спочатку розглядають як ціле число і переводять по описаному алгоритму, а після цього ділять на 10^k, записане двійковими символами (10^k можна перевести в двійковий код по тому ж алгоритму).

Приклад.Задане A_{10}=0,37 або в коді Д1 A=0,0011 0111. Знайти A_2.Рішення зображено на рис. 6.7.

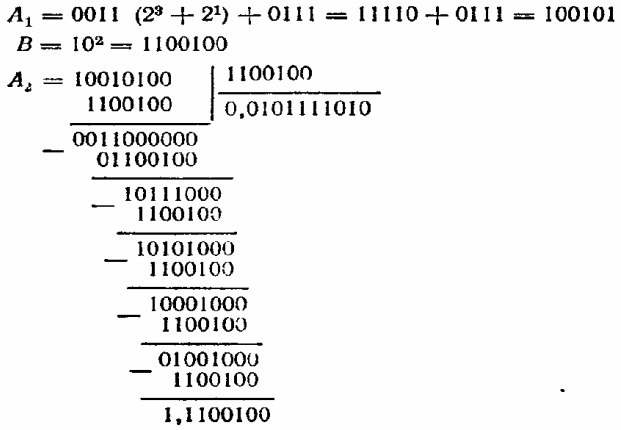
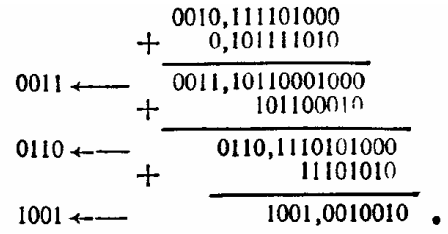


Рисунок 6.7 – Рішення прикладу

Переведення чисел з двійкової системи счислення в Д-код може здійснюватися різними способами. Наприклад, діленням цілих двійкових чисел на число 1010. При цьому десяткові цифри одержуються послідовно одна за інший. При дробових числах ця операція видозмінюється таким чином, щоб при множенні на число 1010 можна було отримати відповідні цифри десяткових дробів.

Приклад. Задане A_2 = 0,0101111010 = $0{,}37_{10}$. Знайти двійково-десятковий код цього числа. При визначенні коду кожної десяткової цифри числа множення A на 1010_2, Замінюємо додаванням $A \bullet 2^3$ і $A \bullet 2^1$. Рішення зображено на рис. 6.8.



Ричунок 6.8 - Рішення прикладу

Таким чином, $A_{2\text{-}10}$= 0,0011 0110 1001 = $0{,}369_{10}$.

Переведення з Д-коду в двійкову систему счислення і навпаки можна спростити наступним чином. Нехай число А задане в коді Д1, тобто

$$A = \sum_{-k}^{n} p^i \sum_{1}^{4} \alpha_j^i 2^{j-1},$$

Позначивши через b_j^i константи виду $p^i 2^{j-1}$ одержимо

$$A = \sum_{-k}^{n} \sum_{1}^{4} \alpha_j^i b_j{}^i,$$

Отже, якщо зберігати двійкові коди констант b_j^i в таблиці, то переведення з Д-коду в двійкову систему зводиться до формування часткових добутоків виду $a_j^i b_j^i$, де $a_j^i = \{0,1\}$, і їхнього наступного складання.

Приклад. В коді Д1 задане число A_{10}= 59, т. е. $A_{2\text{-}10}$= 0101 1001. Знайти двійковий код цього числа. Константи b_j^i в цьому випадку наступні:

$$b_1^0 = 1; b_2^0 = 2_{10} = 10_2; b_3^0 = 4_{10} = 100_2; b_4^0 = 8_{10} = 1000_2;$$
$$b_1^1 = 10_{10} = 1010_2; b_2^1 = 20_{10} = 10100_2; b_3^1 = 40_{10} = 101000_2;$$
$$b_4^1 = 80_{10} = 1010000_2.$$

Тому

$$A_2 = 0 \bullet 1010000 + 1 \bullet 101000 + 0 \bullet 10100 + 1 \bullet 1010 +$$
$$+ 1 \bullet 1000 + 0 \bullet 0100 + 0 \bullet 10 + 1 \bullet 1 = 111011.$$

Ці ж константи b_j^i дозволяють спростити переклад числа А з двійкової системи счислення в Д-код. При цьому спочатку з числа А віднімається найбільша константа b_4^n. Якщо різниця позитивна, то a_4^n = 1, в противному випадку a_4^n = 0. На наступному кроці з залишку віднімається наступна константа b^n, а значення a_3^n визначається знаком нового залишку. Ця процедура повторюється до тих пір, доки не будуть знайдені всі a_j^i. Слід відзначити, що при негативному, поточному залишку до нього на наступному кроці нова константа додається.

Приклад. Задане $A = 59_{10} = 111011_2$. Знайти представлення числа А в коді Д1. В даному випадку n = 1, тому зажадаються тільки константи $b_1^0 - b_4^0$ і $b_1^1 - b_4^1$, за допомогою двійкових кодів яких за 8 кроків отримаємо всіх вісем коефіцієнтів a_j^i:

0111011

1-й крок — 1010000

— 010101 Зсув $\alpha_4^1 = 0$

2-й крок + 101000

10011 Зсув $\alpha_3^1 = 0$

3-й крок — 10100

— 00001 Зсув $\alpha_2^1 = 0$

4-й крок + 1010

+ 1001 Зсув $\alpha_1^1 = 0$

5-й крок — 1000

+ 0001 Зсув $\alpha_4^0 = 1$

6-й крок — 100

— 011 Зсув $\alpha_3^0 = 1$

7-й крок + 10

— 01 Зсув $\alpha_2^0 = 1$

8-й крок + 1

00 Зсув $\alpha_1^0 = 1$

Таким чином, $A_{2\text{-}10}$=0101 1001.

Перевагою даного способу переведення є використання одних і тих же констант для прямого і зворотного перетворень, що спрощує його апаратурну реалізацію. При цьому переведення в Д-код зводиться по суті до виконання операції ділення з змінним дільником, а переведення з Д-коду до виконання операції множення з змінним множеним.

7 КОНТРОЛЬ ВИКОНАННЯ ОПЕРАЦІЇ

7.1 Загальні положення

Контроль правильності виконання операцій може бути здійснений застосуванням спеціальних арифметичних кодів, ідея побудови яких базуються на властивостях порівняння по модулю.

Річ у тім, що при розгляді різноманітних арифметичних виразів вхідні числа, яці входять в ці вирази, можна замінювати на інші, порівняні із ними по вибраному модулю *p*. В частковості, кожне число може бути замінене своїм вирахуванням. При цьому всі машинні числа вважаються умовно цілими.

Розрізняють два способу отримання контрольного коду: числовий і цифровий. При числовому способі контролю контрольний код заданого числа визначається як найменший позитивний залишок від ділення числа *A* на вибраний модуль *p*:

$$r_A = A - \{A/p\}, \tag{7.1}$$

де $\{A/p\}$ □ ціла частина від ділення числа *A* на *p*.

Величина модуля p істотно впливає на якість контролю. Якщо при числовому контролі $p = q$, де q □ основа системи счислення, в якій уявлена число, то контролюється тільки молодший розряд числа і контроль як такий не має сенсу. Для $p = q^m$ Справедливі аналогічні міркування, бо знов не всі розряди числа (при $m<n$) беруть участь в контролі і помилки в розрядах старше m взагалі не сприймаються.

Для числового способу контролю по mod *p* справедливі основні властивості порівнянь. Тому, якщо

$$\begin{aligned} A &\equiv r_A (\mathrm{mod}\ p); \\ B &\equiv r_B (\mathrm{mod}\ p). \end{aligned} \tag{7.2}$$

де

$$0 \le r_A \le p\text{-}1;\ 0 \le r_B \le p\text{-}1;\ \text{то}\ A+B \equiv r_A + r_B (\mathrm{mod}\ p),$$

Звідки

$$r_{A+B} \equiv r_A + r_B (\mathrm{mod}\ P). \tag{7.3}$$

Аналогічним образом доводиться справедливість і наступних співвідношень:

$$r_{A-B} \equiv r_A - r_B (\text{mod } p);$$
$$r_{AB} \equiv r_A r_B (\text{mod } p). \qquad (7.4)$$

Приклад. Задані $A = 125$; $B = 89$; $p = 11$. Знайти $r_A(\text{mod } p)$ і $r_B(\text{mod } p)$.

$$r_A == 125 — \{125/11\} \bullet 11 \equiv\equiv 4 (\text{mod } 11);$$
$$r_A = 89 — \{89/11\} \bullet 11 \equiv\equiv 1 (\text{mod } II).$$

Недоліком числового способу контроля по mod p є використання операції ділення для визначення залишку, що вимагає більших затрат машинного часу.

При цифровому способі контролю контрольний код числа утвориться діленням суми цифр числа на вибраний модуль при виконанні умов

$$r'_A = \sum_i a_i - \left\{ \frac{\sum_i a_i}{p} \right\} \bullet p \text{ або } r'_A \equiv \sum_i a_i (\text{mod } p) \qquad (7.5)$$

Можливі два шляхи отримання контрольного коду: 1) безпосереднє ділення суми цифр на модуль p; 2) складання цифр по mod p. Другий шлях значно простіше, бо якщо $a_i < p$, то контрольний код одержується тільки операцією складання. В цьому полягає перевага цифрового способу контролю перед цисловим.

Приклад. Задані $A = 153$; $B = 41$; $p = 11$. Знайти r'_A і r'_B.

$$r_A^1 = \sum a_i = 9; r_B^1 = \sum b_i = 5.$$

Однак при цифровому способі властивості порівнянь не завжди справедливі. Відбувається це із-за наявності переносу (позик) при виконанні арифметичних дій над числами. Тому отримання контрольного коду результату операція повинно супроводжуватися його корекцією.

Припустимо, задані числа A і B їхні контрольні коди

$$r'_A \equiv \sum_i a_i (\text{mod } p), r'_B \equiv \sum_i b_i (\text{mod } p),$$
$$a \quad також \quad C = A + B, \qquad (7.6)$$

Вимагається знайти контрольний код r'_C. За наявності результату C код r'_C знаходиться по загальному правилу:

$$r'_C \equiv \sum_i c_i (\text{mod } p) \qquad (7.7)$$

Для порівняння контрольних кодів необхідно отримати r_C' через контрольні коди доданків.

Суму цифр результату можна знайти, знаючи цифри a_i і b_i і кількість

переносу в кожному розряді. Кожний перенос уносить із одного розряду q одиниць, де q − основа системи счислення, і додає одну 1 в наступний розряд, тобто сума цифр зменшиться на величину (q-1) на кожний перенос. Тоді

$$\sum_{i=1}^{n} c_i = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i - k(q-1), \tag{7.8}$$

де k − кількість переносу, виниклих при додаванні. Таким чином

$$r'_A \equiv \sum_i a_i (\mathrm{mod}\, p);$$
$$r'_B \equiv \sum_i b_i (\mathrm{mod}\, p);$$
$$r'_C \equiv \sum_i c_i (\mathrm{mod}\, p).$$

Підставивши ці значення в (7.8), отримаємо

$$r'_C \equiv [r'_A + r'_B - k(q-1)] (\mathrm{mod}\, p). \tag{7.10}$$

Аналогічні міркування можна навести і для різності чисел C = A - B.

Приклад. Задані $A = 565$; $B = 278$; $p = 11$. Знайти r_C' і r_{A+B}',

$$\sum a_i = 5 \oplus 6 \oplus 5 \equiv 5(\mathrm{mod}11),$$
$$r'_A = 5(\mathrm{mod}11);$$
$$\sum b_i = 2 \oplus 7 \oplus 8 = 6(\mathrm{mod}11),$$
$$C = A + B = 843; k = 2; r'_C = 8 \oplus 4 \oplus 3 \equiv 4(\mathrm{mod}11);$$
$$r'_{A+B} = 5 \oplus 6 - 2(10-1) \equiv 4(\mathrm{mod}11).$$

7.2 Вибір модуля для контролю

Як було відзначене вище, справедливість властивостей порівнянь для контрольних кодів розповсюджується тільки на числовий спосіб контролю. В цьому складається його гідність. Гідність цифрового способу контролю полягає в простоті отримання контрольних кодів. Щоб зберегти обидва ці гідності, необхідно виконати умову

$$r_A = r'_A$$

Так як $r_A \equiv A (\mathrm{mod}\, p); r'_A = \sum_i a_i (\mathrm{mod}\, p)$, то в цьому випадку повинна виконуватись умова

$$\sum_i a_i q_i \equiv \sum_i a_i (\mathrm{mod}\, p). \tag{7.11}$$

Це можливо тільки тоді, коли кожний елемент кожної з частин вираження рівний:

$$a_i q^j \equiv a_i (\mathrm{mod}\ p)$$

або

$$q^i \equiv 1(\mathrm{mod}\ p).$$

Звідси випливає, що

$$q \equiv 1(\mathrm{mod}\ p).$$

або

$$q = mp+1,$$

де m − ціле число.

Звідки одержуємо

$$p = (q—1)/m \qquad (7.12)$$

тобто для збереження умови $r_A = r'_A$ необхідно накласти певні обмеження на модуль p. До величини модуля звичайно подаються наступні вимоги:

1) величина модуля повинна забезпечувати викриття будь-якої з одинничних арифметичних або логічних помилок і якомога більшого числа помилок більш високої кратності;

2) утворення контрольного коду повинно здійснюватися найбільш простим і швидким способом, тобто простими засобами;

3) величина модуля p повинна бути по можливості невелика, з тим щоб залишки r від ділення на нього будь-яких чисел не вимагали великого обсягу обладнання для зберігання і обробки.

Аналіз (7.11) з урахуванням викладених вимог показує, що для двійкової системи счислення немає цілочисельного рішення. Тоді, з урахуванням того що цифрова інформація в ЕОМ представляється двійковими символами, для контролю доцільно перейти до систем счислення з основою $q = 2^S$, де s − деяке ціле позитивне число ($s \geq 2$). Цей перехід здійснюється розбиттям вхідної двійкової інформації на групи по s розрядів з наступним складанням цих груп по модулю $p = (2^S - 1)/m$ або при $m = 1$, $p = 2^S - 1$. Цей процес називається коагулюванням числа по модулю p.

На практиці вміст контрольних розрядів, тобто пакунку двійкового числа визначається слідуючим чином:

1) все число, починаючи з молодших розрядів, розбивається на $i=n/s$ груп по s розрядів кожна;

2) всі гурти послідовно, починаючи з молодшої, складаються на звичайному двійковому s-разрядном суматорі по модулю $p=2^S - 1$, тобто з циклічним переносом одиниці переповнення в молодший розряд (суматор зворотного коду). Отримана на суматору пакунку і буде цим залишком в двійковій системі счислення.

Розглянемо часткові випадки утворення пакунок при різноманітних значеннях модуля p.

1. Контроль по mod 3 (m = 1, s = 2, p = 3). Так як $2^2 \equiv 1(\text{mod } 3)$, то контрольована інформація представляється символами четверичної системи, а пакунки утворяться складанням діад по модулю 3. Для цього вимагається двухрозрядний двійковий суматор з ланцюгом циклічного переносу з старшого розряду в молодший.

Приклад. Задані $A = 53_{10} = 110101_2$; $B = 30_{10} = 011110_2$; $p = 3$.

$$r_A = 11 \oplus 01 \oplus 01 \equiv 10(\text{mod } 3);$$
$$r_B = 01 \oplus \text{II} \oplus 10 = \text{II} \ (\text{mod } 3).$$

2. Контроль по mod 7 (m = 1; s = 3; p = 7). Так як $2^3 \equiv 1(\text{mod } 7)$, то контрольована інформація представляється символами вісьмиричної системи, а для отримання пакунків необхідний трьохрозрядний двійковий суматор з ланцюгом циклічного переносу.

Приклад. Задане $A = 249_{10} = 011111001_2$; $p = 7$.

$$r_A = 011 \oplus 111 \oplus 001 \equiv 100(\text{mod } 7).$$

Можна показати, що з зростанням модуля підвищується корекційна спроможність однак при цьому росте обсяг контрольного обладнання.

7.3 Контроль логічних операцій

До логічних операцій відносяться операції зсуву, логічного додавання і множення.

Нехай задане число $A = a_n a_{n-1} \ldots a_1$, що має контрольний код $r_A = a_{ks} \ldots a_{k1}$.

Позначимо код числа A, ссунутий ліворуч, через $\bar{A}$ (без циклічного переносу) і через $\bar{A}_ц$ з циклічним перенесенням (при зсуві праворуч стрілка в позначці буде повернута праворуч). Тоді контрольні коди можна позначити відповідно:

$$A \equiv r_A(\text{mod } p); A \equiv r_{\bar{A}}(\text{mod } p); \vec{A} \equiv r_{\vec{A}}(\text{mod } p);$$
$$\vec{A}_M \equiv r_{\vec{A}_M}(\text{mod } p) \qquad (7.13)$$

Зсув ліворуч на один розряд двійкового числа еквівалентно множенню його на два. Так як при зсуві числа відбувається втрата деяких його розрядів, то можна припустити, що контрольний код ссунутого числа зміниться на величину Δ:

$$r_{\bar{A}} = \bar{r}_A + \Delta(\text{mod } p), \qquad (7.14)$$

де $\bar{r}_A = 2r_A$ − ссунутий вліво контрольный код.

Величина A залежить від значень a_n і a_{ks}, що при зсуві виходять за межі розрядної сітки. Вихід старшої одиниці за межі розрядної сітки при зсуві n-

разрядного числа еквівалентному видніманню $a_n\sigma_{n+1}$ одиниць з контрольного коду зсунутого числа, де σ_{n+1} − вага $(n + 1)$-го разряду.

Вихід за межі розрядної сітки розряду a_{ks}, = 1 при зсуві контрольного коду ліворуч еквівалентному його зменшенню на $2^S \equiv 1 \pmod{(2^S - 1)}$, тобто цю втрату можна відновити доданням одиниці до контрольного коду. Тоді можна записати

$$r_{\bar{A}} \equiv (\bar{r}_A - a_n\sigma_{n+1} + a_{ks}) \bmod(2^s - 1). \qquad (7.15)$$

Ваги розрядів числа, наведеного в системі з основою 2^S, призначаються слідуючим чином, наприклад, при $s = 3$: розряди a_i:$a_n\, a_{n-1}\, a_{n-2}\, a_{n-3}\, a_{n-4}\, a_{n-5} \ldots a_3\, a_2\, a_1$, вага σ: $2^2\ 2^1\ 2^0\ 2^2\ 2^3\ 2^1\ 2^0 \ldots 2^2\ 2^1\ 2^0$, тобто $\sigma_{n-1} \equiv 1(\mathrm{mod}(2^S-1))$

Внаслідок цього значення поправок Δ для виконання лівого зсуву по модулю (2^S-1) будуть такими, як в табл. 7.1.

Таблиця 7.1 – Значеня поправок

Значення a_n	0	1	0	1
Значення a_{ks}	0	0	1	1
Поправка Δ	0	-1	+1	0

Значення поправки Δ = -1 звичайно замінюють її доповненням по модулю p.

Для виконання зсуву ліворуч з циклічним переносом зі старшого розряду в молодший необхідно зменшити контрольний код на величину a_n $(\sigma_{n+1} - 1)$. Так як $\sigma_{n+1} \equiv \equiv 1(\mathrm{mod}\ p)$, то цей член рівний 0. Тоді можна записати

$$r_{\bar{A}_ц} = (\bar{r}_A + a_{ks})(\mathrm{mod}(2^s - 1)). \qquad (7.16)$$

Приклад. Задане A = 1,01011010; s = 3; p = 7. Знайти $r_{\bar{A}}$ і $r_{\bar{A}_ц}$.

$$r_A = 101 \oplus 011 \oplus 010 \equiv\equiv 011(\mathrm{mod}7);$$

$$\bar{A} = 0{,}10110100;\ \bar{r}_A = 110.$$

Так як a_n = 1, a_{ks}= 0, то

$$r_{\bar{A}} = 111 - 1 + 000 \equiv\equiv 101(\mathrm{mod}\,7).$$

При циклічному зсуві $\bar{A}_ц$ = 0,10110101. Тоді

$$\bar{A}_ц = 110 \oplus 000 \equiv\equiv 110\ (\mathrm{mod}\ 7).$$

Таким чином, для того щоб не порушувалася відповідність між основним і контрольним кодами при циклічному зсуві на один розряд

ліворуч, необхідно до зсунутого контрольного коду додати значення старшого його розряду, що він мав до зсуву. Для цієї мети служить ланцюг циклічного переносу з старшого розряду в молодший в суматору контрольного коду.

При зсуві праворуч відбувається втрата молодших розрядів числа і його контрольного коду. Контрольний код зсунутого праворуч числа можна отримати по формулі

$$r_{\vec{A}} \equiv (\vec{r}_A + \Delta) \bmod(2^s - 1), \qquad (7.17)$$

тільки поправка Δ буде при цьому інший.

В випадку простого зсуву поправка Δ до контрольного коду залежить від модуля p і для p=7 приймає наступні значення (табл. 7.2).

Таблиця 7.2 – Значення поправки у випадку простого зсуву

Значення a_i	0	0	1	1
Значення a_{k1}	0	1	0	1
Поправка Δ	000	100	011	000

При зсуві, що модифікувався праворуч, який виконується по правилу A = 1, a_{n-1} ... a_2a_1; $\vec{A}_M$ == 1,1 a_n ... a_3a_2 Також відбувається втрата молодших розрядів кодової комбінації числа і контрольного коду. Для цього випадку контрольний код обчислюється по (7.13), але поправки Δ для p = 7 приймуть вигляд (табл. 7.3).

Таблиця 7.3 – Поправка для коду, що модифікувався праворуч

Значення a_i	0	0	1	1
Значення a_{k1}	0	1	0	1
Поправка Δ	100	001	000	100

Приклад. Задане A = 1,01110111101; p= 7. Знайти $r_{\vec{A}}$ і $r_{\vec{A}_M}$.

$$r_A = 101 \oplus 110 \oplus 111 \oplus 101 \equiv 010 \ (\text{mod } 7);$$

$$\vec{A} = 0,10111011110; \ \vec{r}_A = 001.$$

Так як a_1 = 1, a_{k1} = 0, то Δ = 011. Тоді $r_{\vec{A}}$ = 001⊕011≡100(mod 7)

Модифікований зсув числа A:

$$\vec{A}_M = 1,10111011110.$$

При a_1=1; a_{k1}=0 поправка Δ=000. Тоді $r_{\vec{A}_M}$ =001⊕ 000=001(mod 7).

Таким чином, однім зі способів реалізації відповідності між основним і контрольним кодами при зсуві праворуч є корекція контрольного коду в відповідності з наведеної вище табл. 7.3.

Другим способом реалізації відповідності між основним і контрольним кодами при зсуві праворуч є подовження регістру основного коду на n розрядів, з тим щоб при зсуві праворуч не пропадали молодші розряди коду. При цьому здійснюється контроль по модулю p всього $2n$-разрядного регістру основного коду.

Операцію додавання по mod 2 можна представити через інші операції, наприклад $A \oplus B = A + B - 2(A \wedge B)$.

Це вираження можна уявити інакше:

$$A \oplus B = (A + B) + \overline{A \wedge B_{сдв}}.$$

Тоді, використовуючи перехід від арифметичних виразів до порівнянь, отримаємо наступну формулу для утворення контрольного коду:

$$r_{\oplus} = r_{A+B} + \bar{r}_{\wedge} \pmod{p}, \qquad (7.18)$$

де r_{A+B} − контрольный код суммы двух чисел; $\bar{r}_{\Delta}$ − інверсія контрольного коду логічного добутку двох чисел зі зсувом ліворуч на один розряд.

Приклад. Задані $A = 010000111$; $B = 101110011$; $p = 7$.

Знайти $r_{\oplus}$. Знаходимо: $r_A = 010$; $r_B = 000$; $r_{A+B} = 0{,}10$. Потім одержимо:

$$A \wedge B = 0000000011: r_{\wedge} = 011;$$

$$(A \wedge B)_{сдв} = 0000000110;\ \bar{r}_{\wedge} = 110.$$

Після цього визначимо $\bar{r}_{\wedge} = 001$ і по формулі (7.13) отримаємо

$$r_{\oplus} = 010 \oplus 001 \equiv 011 \pmod{7}.$$

7.4 Контроль арифметичних операцій

Як відомо, арифметичні операції виконуються в ЕОМ в прямому, зворотному і додатковому кодах. Якщо зображення чисел зберігаються в машині в відповідному коді, те їх можна розглядати як єдині кодові комбінації, до яких можна застосовувати всі що формулювалися вище правила отримання пакунків. При цьому вимагається обов'язкова кратність загального числа розрядів вибранному модулю.

При додаванні в прямому коді складаються тільки цифрові частини зображенн чисел, а знаки обробляються окремо. Тому в цьому випадку можливий контроль двома способами:

1) роздільний контроль знакової і цифрової частин зображенн результату;

2) узагальнений контроль всього зображення. При роздільному способі

контроль знакових розрядів роблять звичайно за допомогою схеми, що виявляє переповнення, бо в коді, що модифікувався поява одиночної помилки в знакових розрядах призводить до незбігу інформації в них. Для цифрових частин зображеннь маємо

$$r_{|[A \pm B]_{пр}|} \equiv r_{|A|} \pm r_{|B|}. \qquad (7.19)$$

При узагальненому способі контролю вимагається корекція контрольного коду результату внаслідок того, що знак результату при додаванні чисел з рівними знаками повторює знак доданків, тобто контрольний код суми буде рівний

$$r_{(A+B)пр} \equiv (r_A + r_B - S_q \sigma_s), \qquad (7.20)$$

де S_q − значення знакового розряду операндів; σ_s − вага старшого розряду свертки.

Приклад. Задані $[A]_{пр}$ = 1,01101011; $[B]_{пр}$ = 1,00110010; p=7. Знайти $r_{[A+B]_{пр}}$.

Маємо: r_a = 110; r_B = 101; $[A + B]_{пр}$= 1,10011101; $r_{[A+B]_{пр}}$ = 111.

На основі (7.19) отримаємо

$$r_{[A+B]_{пр}} = 110+101 - 1 \bullet 100 \equiv 111(\text{mod } 7).$$

Узагальнений спосіб може бути застосований і для контролю виконання операції додавання в зворотному коді. В цьому випадку

$$r_{[A+B]_o} \equiv r_{[A]_o} + r_{[B]_o}$$

Приклад. Задані $[A]_o$ = 1,011001001; $[B]_o$ = 0,110001111; p = 3. Знайти $r_{[A+B]_o}$.

Розв'язок: $r_{[A]_o}$ = 10; $r_{[B]_o}$ = 11; $[A+B]_o$ = 0,001011001; $r_{[A+B]_o}$ = 10.

Перевірка:

$$r_{[A+B]_o} \equiv r_{[A]_o} + r_{[B]_o} \equiv 10(\text{mod} 3).$$

При додаванні чисел в додатковому коді необхідна корекція контрольного коду результату в випадку, якщо виникає одиниця переносу зі знакового розряду, тобто контрольний код суми буде рівний

$$r_{[A+B]_Д} \equiv r_{[A]_Д} + r_{[B]_Д} - \alpha \qquad (7.21)$$

де α — корекція (α = 1, якщо виник перенос із знакового розряду, і α = 0, якщо переносу немає).

Приклад. Задані: $[A]_д$ = 1,00110001110; $[B]_д$ = 1,11101110111; p = 15. Знайти $r_{[A+B]_Д}$. Розв'язок: $r_{[A]_Д}$ = 0001; $r_{[B]_Д}$ = 1110; $[A+B]_д$= 1,00100000101;

$r_{[A+B]_Д} = 1110 \pmod{15}$.

Перевірка:

$$r_{[A+B]_Д} \equiv r_{[A]_Д} + r_{[B]_Д} - \alpha \equiv 0001 + 1110 — 0001 \equiv 1110 \pmod{15}.$$

В зв'язку з алгебраїчним додаванням чисел в інверсних кодах виникає питання про перетворення контрольного коду при перетворенні прямого коду чисел в інверсний і назад. Для конкретності будемо вважати, що негативні числа зображаються в зворотному коді і $p = 2^S - 1$.

Нехай до перетворення в регістрі зберігався код

$$[A]_{пр} = 2^n + |A|$$

тобто в знаковому розряді була записана 1 (числа умовно цілі). Після перетворення цього коду в зворотний в регістрі отримаємо код

$$[A]_o = 2^n + (2^n - 1 - |A|),$$

Виразив $[A]_o$ через $[A]_{пр}$, одержимо

$$[A]_o = 3 \bullet 2^n - 1 - [A]_{пр} = 2[A]_{пр} + 2 \pmod 3$$

При перетворенні прямого коду в зворотний (або навпаки) необхідно виконати такі ж перетворення з контрольним кодом, тобто

$$r_{A2} \equiv 2r_{A1} + 2 \pmod 3, \qquad (7.22)$$

де r_{A1} − контрольний код до перетворення; r_{A2} − контрольний код після перетворення.

Таким чином, при виконанні операції алгебраїчного додавання в зворотному і додатковому кодах обробка основних і контрольних кодів може робитися в двох різних суматорах паралельно, що дозволяє в значному ступені розпаралелити процеси в основному і контрольному АУ, спростити алгоритми арифметичних операцій і зменшити час їхньої реалізації. При додаванні у зворотному коді суматори працюють незалежно один від одного при додаванні в додатковому коді одиниця переносу зі знакового розряду суми використовується для корекції контрольного коду останньої.

При використанні позиційної системи счислення всі арифметичні операції зводяться до послідовності додаваннь кодів. Тому достатньо перевіряти тільки правильність всіх сум, одержуваних в машині (остаточних при алгебраїчному додаванні і проміжних при множенні і діленні). Якщо формується $2n$-разрядный результат, то операцію множення можна перевіряти також і прямим способом − порівнянням добутока залишків співмножників з залишком їхнього добутка.

Можливість порівняння основного і контрольного кодів перевіряється тільки після закінчення операції. При цьому в кінці операції перевіряється можливість порівняння також основного і контрольного кодів операндів. Це зумовлене тим, що поодинока помилка в операнді може виявитися джерелом багаторазових помилок в результаті операції. Наприклад,

помилка, яка викривила один розряд множеного при його зсуві, викличе серію невірних його передач в суматорі. Тому цю помилку можна виявити або тільки при перевірці всіх часткових добутоків, або шляхом перевірки відповідності основної і контрольної кодів множеного після закінчення операції.

СПИСОК ЛІТЕРАТУРИ

1. Майоров С.А. Введение в микро ЭВМ / Майоров С.А., Кириллов В.В., Приблуда А.А. – Ленинград: Машиностроение, – 1988. – 304 с.

2. Касаткин В.Н. Введение в кибернетику / Касаткин В.Н. – Киев: Радянська школа, – 1976. – 176 с.

3. Шестаков А.П. Введение в информатику / Шестаков А.П. – Пермь: Пермский университет, – 1999. – 71 с.

4. Андреева Е.Н. Системы счисления и компьютерные арифметика / Андреева Е.Н., Фалина И.Г. − М.: Лаборатория базовых знаний, − 1999. – 256 с.

5. Булос Д. Вычислимость и логика / Булос Д., Джефри Р. − М.: Мир, − 1994. – 396 с.

6. Микушин А.В. Занимательно о микроконтроллерах. / Микушин А.В. − СПб: БХВ-Петербург, − 2006. – 429 с.

7. Дадаев Ю. Г. Теория арифметических кодов / Дадаев Ю. Г. − М.: Радио и связь, − 1981. – 272 с.

8. Угрюмов Е.П. Цифровая схемотехника / Угрюмов Е.П. – СПб: БХВ-Петербург, − 2004. – 800 с.

9. Учебное пособие: Системы счисления [Електронный ресурс]. – Режим доступу: http://www.numeration.ru

10. Гашков С. Б. Арифметика, алгоритмы, сложность вычислений / Гашков С. Б., Чубариков В. Н. − М.: Высшая школа, − 2000. – 320 с.

11. Сэвидж Д. Э. Сложность вычислений / Севидж Д. Э. − М.: Факториал, − 1998. – 368 с.

12. Петцольд Ч. Код / Петцольд Ч. − М.: Русская редакция Майкрософт Пресс, − 2001. – 492 с.

13. Фомин С. В. Системы счисления / Фомин С. В. − М.: Наука, − 1980. – 48 с.

14. Гук М.Ю. Апаратные средства IBM PC / Гук М.Ю. – СПб.: Питер, − 2006. – 1072 с.

15. Алфутова Н. Б. Алгебра и теория чисел: Сборник задач. / Алфутова Н. Б., Устинов А. В. − М.: МЦНМО, − 2002. – 269 с.

16. Воробьёв Н. Н. Признаки делимости / Воробьёв Н. Н. − М.: Наука, − 1988. − 96 с.

17. Кнут Д. Искусство программирования: том 2 / Кнут Д. − М.: Вильямс, − 2000. – 788 с.

18. Стахов А. П. Коды золотой пропорции / Стахов А. П. − М.: Радио и связь, − 1984. – 152 с.

Навчальне видання

Роман Наумович Квєтний
Петро Михайлович Повідайко
Микола Миколайович Компанець
Володимир Володимирович Гармаш
Ярослав Анатолійович Кулик

АРИФМЕТИЧНІ ОСНОВИ ПРОЕКТУВАННЯ МП СИСТЕМ

Навчальний посібник

Редактор С.А.Малішевська

Оригінал-макет підготовлено авторами

Підписано до друку .....
Формат 29,7×42¼. Папір офсетний.
Гарнітура Times New Roman.
Друк різографічний. Ум. друк. арк. ….
Наклад …. пр. Зам. № 2015-

Вінницький національний технічний університет,
навчально-методичний відділ ВНТУ.
21021, м. Вінниця, Хмельницьке шосе, 95.
ВНТУ, к. 2201.
Тел. (0432) 59-87-36.
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.

Віддруковано у Вінницькому національному технічному університеті
в комп'ютерному інформаційно-видавничому центрі
21021, м. Вінниця, Хмельницьке шосе, 95.
ВНТУ, ГНК, к. 114.
Тел. (0432) 59-87-38,
publish.vntu.edu.ua; email: kivc.vntu@gmail.com.
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.

Міністерство освіти і науки України
Вінницький національний технічний університет

Арифметичні основи проектування мікропроцесорних систем

НАВЧАЛЬНИЙ ПОСІБНИК

Усі цитати, цифровий, фактичний матеріал та бібліографічні відомості перевірені, написання одиниць відповідає стандартам.

Зауваження рецензентів враховані.

Відповідальність за наявність плагіату несуть автори.

Вимогам, які висуваються до навчальної літератури, відповідає.

До друку і в світ дозволяю на підставі § 2 п. 15 «Єдиних правил...»

Перший проректор з науково-педагогічної роботи по організації навчального процесу та його науково-методичного забезпечення

О. Н. Романюк

Автори: _____ Р.Н. Квєтний
(підпис)
_____ П.М. Повідайко
(підпис)
_____ М.М. Компанець
(підпис)
_____ В.В. Гармаш
(підпис)
: _____ Я.А. Кулик
(підпис)

Затверджено
на засіданні кафедри АІВТ
Протокол № 5 від 15.11.2016 р.
Зав. кафедрою
_____ Р.Н. Квєтний
(підпис)

Вінниця ВНТУ 2017