МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІГІВСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

МАШИННІ РОЗРАХУНКИ

Методичні вказівки
для самостійної роботи та виконання РГР з дисципліни
Комп'ютерна арифметика
для студентів напряму підготовки
6.050102 – «Комп'ютерна інженерія»

Обговорено та рекомендовано
на засіданні кафедри
Інформаційних та комп'ютерних систем
Протокол № 6
від 25 грудня 2014 р.

Чернігів ЧНТУ 2015

Машинні розрахунки. Методичні вказівки для самостійної роботи та виконання розрахунково-графічної роботи з дисципліни «Комп'ютерна арифметика» для студентів напряму підготовки 6.050102 – «Комп'ютерна інженерія». /Укл. Ульченко Д. О. – Чернігів: ЧНТУ. – 2015. – 44 с.

Укладач: УЛЬЧЕНКО ДМИТРО ОЛЕГОВИЧ, кандидат технічних наук, доцент кафедри інформаційних та комп'ютерних систем

Відповідальний за випуск: КАЗИМИР ВОЛОДИМИР ВІКТОРОВИЧ, завідувач кафедри інформаційних та комп'ютерних систем, доктор технічних наук, професор

Рецензент: ІВАНЕЦЬ СЕРГІЙ АНАТОЛІЙОВИЧ, кандидат технічних наук, доцент кафедри промислової електроніки Чернігівського національного технологічного університету

Зміст

Вступ

Мета виконання розрахунково-графічної роботи полягає у закріпленні, поглибленні та узагальненні теоретичного матеріалу та отримання практичних навичок у виконанні основних математичних операцій в ЕОМ.

Для вдалого виконання розрахунково-графічної роботи студент повинен знати основи подання даних в ЕОМ, елементарні методи обробки чисел з фіксованою та плавучою комою. Мати уявлення про сучасний стан комп'ютерної реалізації математичних операцій.

Тематика розрахунково-графічної роботи відповідає задачам дисципліни «Комп'ютерна арифметика» і тісно пов'язана з практичними потребами спеціаліста направлення підготовки «Комп'ютерна інженерія». Варіанти завдань студент отримує у відповідності до номеру залікової книжки та порядкового номера у списку групи. Варіанти є індивідуальними для кожного студента і не можуть повторюватись.

Студент виконує РГР самостійно під контролем викладача протягом семестру до початку залікового тижня. Відповідальність за якість та своєчасність виконання несе автор – студент.

1 Теоретичні засади комп'ютерної арифметики

1.1 Системи числення

Системою числення називається сукупність способів і правил запису чисел цифровими знаками. Розрізняють системи числення непозиційні й позиційні. У непозиційних системах числення значення кожного символу не залежить від його позиції в числі. Прикладом непозиційної системи є римська система числення. Недоліками непозиційних систем є необмежена кількість різних символів, необхідних для запису будь-якого числа, і складність арифметичних дій із числами.

Позиційними називаються системи числення, що містять обмежену кількість символів, причому значення кожної цифри в числі знаходиться в чіткій залежності від її позиції в числі. Будь-яка позиційна система числення характеризується основою. Основа (базис) p позиційної системи числення – кількість знаків або символів, що використовуються для зображення числа в даній системі. Число в однорідних позиційних системах числення представляється поліномом виду:

$$A = a_n p^n + a_{n-1} p^{n-1} + \cdots + a_i p^i + \cdots + a_1 p^1 + a_0 p^0 + a_{-1} p^{-1} + \cdots + a_{-k} p^{-k},$$

або

$$A = \sum_{i=-k}^{n} a_i p^i, \tag{1.1}$$

де p – основа системи числення;
$a_i = \overline{0, p-1}$;
$n+1, k$ – кількість цілих і дробових розрядів.

Ціле число в системі числення з основою p записується як:

$$A = a_n p^n + a_{n-1} p^{n-1} + \cdots + a_i p^i + \cdots + a_1 p^1 + a_0 p^0 = \sum_{i=0}^{n} a_i p^i,$$

Правильний дріб в системі числення з основою p має вид:

$$A = a_{-1} p^{-1} + a_{-2} p^{-2} + \cdots + a_{-k} p^{-k} = \sum_{i=-k}^{-1} a_i p^i,$$

На практиці використовують скорочений запис чисел в однорідній позиційній системі числення:

$$A = a_n a_{n-1} a_{n-2} ... a_1 a_0 a_{-1} ... a_{-k}.$$

Вага розряду числа (R_i) в однорідній позиційній системі числення виражається співвідношенням:

$$R_i = \frac{p^i}{p^0} = p^i,$$

де i — номер розряду при відліку справа наліво.

Надалі для простоти викладу будемо вживати термін «система числення», маючи на увазі однорідні позиційні системи числення.

В обчислювальній техніці використовуються в основному наступні системи числення: двійкова, вісімкова, шістнадцяткова, двійково-десяткова. Крім того, застосовуються системи числення спеціального призначення: двійкова система числення із цифрами $1, -1$ і надлишкова двійкова система числення із цифрами $1, 0, -1$.

1.2 Класична двійкова система числення

Під класичною двійковою системою числення розуміється така система, у якій для відображення чисел використовуються два символи 0 і 1, а ваги розрядів змінюються за законом $2^{\pm k}$, де k - довільне ціле число. Широке застосування двійкової системи числення обумовлене наступними причинами: різноманітністю й простотою технічної реалізації елементів із двома стійкими станами; гарним розрізненням двох станів, що зменшує можливість викривлення сигналів і збоїв; простотою виконання арифметичних операцій; економічністю устаткування.

Відповідно до (1.1), при умові що $p = 2$, у загальному вигляді всі двійкові числа представляються у вигляді полінома:

$$A = \sum_{i=-k}^{n} a_i 2^i,$$

де $a_i = \overline{0, 1}$.

Додавання у двійковій системі числення проводиться за правилами додавання поліномів. Тому при додаванні чисел A и B i-й розряд суми S_i й перенос Π_i із даного розряду в $(i+1)$-й буде визначатися у відповідності до наступного виразу:

$$a_i + b_i + \Pi_{i-1} = S_i + 2 \cdot \Pi_i \tag{1.2}$$

де $\Pi_i \in \{0{,}1\}$ й $S_i \in \{0{,}1\}$.

Виразу (1.2) відповідає таблиця 1.1 додавання однорозрядних чисел. Відповідно до цієї таблиці можна додавати багаторозрядні двійкові числа.

Таблиця 1.1 — Додавання однорозрядних двійкових чисел

a_i	b_i	Π_{i-1}	S_i	Π_i
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Операція однорозрядного множення виконується відповідно до таблиці 1.2.

Таблиця 1.2 — Правила множення однорозрядних двійкових чисел

a	b	x
0	0	0
0	1	0
1	0	0
1	1	1

Навички використання двійкових чисел

Далі наведені деякі правила швидкого розрахунку кількісного еквіваленту двійкових чисел:

1) Число $1\underbrace{00\ldots00}_{k\text{ нулів}} = 2^k$.
2) Число $\underbrace{111\ldots11}_{k\text{ одиниць}} = 2^k - 1$.
3) Двійкове число

$$\underbrace{a_5a_4a_3a_2a_1}_{\text{мале число } a}\underbrace{00\ldots00}_{k\text{ нулів}} = a \cdot 2^k$$

4) Двійкове число

$$\overbrace{a_5a_4a_3a_2a_1}^{\text{мале число } a}\underbrace{00\ldots00\overbrace{b_5b_4b_3b_2b_1}^{\text{мале число } b}}_{k\text{ розрядів}} = a \cdot 2^k + b$$

5) Якщо в n-розрядному числі багато одиниць і мало нулів, то для визначення його кількісного еквівалента можна з n-розрядного числа, записаного одними одиницями, відняти мале число, у якому розряди зі значенням 1 відповідають розрядам вихідного числа з нульовим значенням і навпаки.
6) Розрахунок необхідної кількості двійкових розрядів для запису десяткового числа виконується за виразом:

$$]\log_2 A_{10}[= n,$$

де $]X[$ — найменше більше від X ціле число;
n — кількість двійкових розрядів.
7) Читання двійкових дробів.
Дріб $A = 0,\underbrace{00\ldots00}_{k-1\text{ нулів}}1 = 2^{-k}$.
Дріб $A = 0,\underbrace{11\ldots11}_{k\text{ одиниць}} = 1 - 2^{-k}$.
Двійковий дріб читається по тим же правилам, що й десятковий: розряди праворуч від коми читаються як ціле число, яке є чисельником; знаменник читається як ціле число, що є k-им ступенем двійки, причому k – номер молодшого розряду праворуч від коми.

8) Двійкові дроби можуть бути періодичними. Наприклад, періодичними є дроби виду:

$$\frac{1}{2^k-1} = 0{,}(\underbrace{00\ldots001}_{k \text{ розрядів}})(\underbrace{00\ldots001}_{k \text{ розрядів}})\ldots$$

$$\frac{1}{2^k+1} = 0{,}(\underbrace{0\ldots00}_{k \text{ нулів}}\,\underbrace{1\ldots11}_{k \text{ одиниць}})(\underbrace{0\ldots00}_{k \text{ нулів}}\,\underbrace{1\ldots11}_{k \text{ одиниць}})\ldots$$

1.3 Шістнадцяткова система числення

Система числення з основою 16 називається шістнадцятковою системою. З 16 цифр шістнадцяткової системи числення перші 10 одержали позначення від 0 до 9 (десяткові значення від 0 до 9), а інші шість — від A до F (десяткові значення від 10 до 15). У таблиці 1.3 наведені двійкові, вісімкові й десяткові значення кожної шістнадцяткової цифри.

Таблиця 1.3 — Шістнадцяткова система числення

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

1.4 Переведення чисел з однієї системи числення в іншу

1.4.1 Переведення цілих чисел

Для переведення цілого числа з однієї системи числення в іншу, необхідно, діючи у вихідній системі числення, вихідне число послідовне ділити на основу нової системи числення, записану у вихідній системі, до одержання частки, рівної нулю. Число в новій системі числення записується із залишку від ділення, починаючи з останнього.

Приклад:

Необхідно перевести десяткове число 98_{10} у двійкову систему числення:

$$\begin{array}{r|l} 98 & 2 \\ \cline{2-2} 98 & 49 \\ \hline a_0 = 0 & \end{array} \quad \begin{array}{r|l} 49 & 2 \\ \cline{2-2} 48 & 24 \\ \hline a_1 = 1 & \end{array} \quad \begin{array}{r|l} 24 & 2 \\ \cline{2-2} 24 & 12 \\ \hline a_2 = 0 & \end{array} \quad \begin{array}{r|l} 12 & 2 \\ \cline{2-2} 12 & 6 \\ \hline a_3 = 0 & \end{array}$$

$$\begin{array}{r|l} 6 & 2 \\ \cline{2-2} 6 & 3 \\ \hline a_4 = 0 & \end{array} \quad \begin{array}{r|l} 3 & 2 \\ \cline{2-2} 2 & 1 \\ \hline a_5 = 1 & \end{array} \quad \begin{array}{r|l} 1 & 2 \\ \cline{2-2} 0 & 0 \\ \hline a_6 = 1 & \end{array}$$

Отже: $\quad 98_{10} = 1100010_2$

1.4.2 Переведення правильних дробів

Для переведення правильного дробу (без цілої частини) з однієї системи числення в іншу, необхідно, діючи у вихідній системі числення, помножити вихідне число на основу нової системи числення, записану у вихідній системі. У результату відокремити цілу частину, а дробову частину, що залишилась, знову помножити на цю основу. В отриманому результаті відокремити цілу частину і т.д.

Зазначені дії повторюються до одержання потрібної кількості значущих цифр. Результат записується як $0, \ldots$ і цілі частини добутків у порядку їх одержання.

Перевірку правильності переведення можна виконати шляхом розкладання отриманого двійкового дробу у поліном (1.1). При цьому можливе накопичення похибки в наслідок врахування обмеженої кількості двійкових розрядів.

Приклад:

Перевести десятковий дріб $0{,}625_{10}$ у двійкову систему числення:

$$
\begin{array}{rr|l}
 & 0, & 625_{10} \\
 & & \quad 2 \\
\hline
a_1 = & 1, & 250_{10} \\
 & & \quad 2 \\
\hline
a_2 = & 0, & 500_{10} \\
 & & \quad 2 \\
\hline
a_3 = & 1, & 000_{10} \\
 & & \quad 2 \\
\hline
a_4 = & 0, & 000_{10}
\end{array}
$$

$$\text{Отже:} \quad 0{,}625_{10} = 0{,}1010_2$$

1.4.3 Переведення неправильних дробів

Для переведення неправильного дробу (тобто дробу, що містить цілу частину) з однієї системи числення в іншу, необхідно здійснити роздільне переведення її цілої й дробової частин, а результати записати послідовно, відокремивши цілу частину від дробової частини за допомогою коми.

Приклад:

Перевести десятковий дріб $98{,}625_{10}$ у двійкову систему числення. Результати переведення цілої й дробової частин нам уже відомі з попередніх прикладів: $98_{10} = 1100010_2$, $0{,}625_{10} = 0{,}1010_2$. Остаточний результат одержуємо, записавши поруч цілу й дробову частини й відокремивши їх комою: $98{,}625_{10} = 1100010{,}1010_2$.

1.4.4 Переведення чисел із системи числення в систему із кратною основою

Якщо основи систем числення кратні одна одній, тобто пов'язані залежністю $q = p^m$, то кожна цифра системи числення з основою q може бути зображена m цифрами в системі з основою p.

Правило переведення:

Для того щоб перевести число з вихідної системи числення в нову систему, основи яких кратні, достатньо кожну цифру перекладного числа записати за допомогою m цифр у новій системі числення, якщо основа вихідної системи числення більше основи нової системи. А якщо ні, то кожні m цифр вихідного числа необхідно записати за допомогою однієї цифри, у новій системі числення, починаючи для цілих чисел з молодшого розряду й зі старшого для правильних дробів.

Приклад:

Перевести шістнадцяткове $1FA{,}C24_{16}$ й вісімкове число $763{,}224_8$ у відповідні їм двійкові числа:

$$\begin{array}{cccccccccccccc}
1 & F & A & , & C & 2 & 4 & & 7 & 6 & 3 & , & 2 & 2 & 4 \\
\downarrow & \downarrow & \downarrow & , & \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \downarrow & , & \downarrow & \downarrow & \downarrow \\
0001 & 1111 & 1010 & , & 1100 & 0010 & 0100 & & 111 & 110 & 011 & , & 010 & 010 & 100
\end{array}$$

Нулі в старших і молодших розрядах не змінюють кількісного еквівалента числа, тому їх можна вилучити або дописати. Отримаємо у результаті:

$$1FA{,}C24_{16} = 111111010{,}1100001001_2$$
$$763{,}224_8 = 111110011{,}0100101_2$$

1.5 Форми запису чисел в ЕОМ

Під машинним (автоматним) зображенням числа розуміють запис числа А у розрядній сітці ЕОМ (цифрового автомату). Машинне зображення числа умовно позначають символом $[\ldots]$. При цьому справедливе співвідношення:

$$A = [A] \cdot K,$$

де K – коефіцієнт, величина якого залежить від форми запису числа у автоматі.

Існують три форми запису чисел: природна, з фіксованою комою і з плаваючою комою (нормальна).

1.5.1 Природна форма запису числа

Природною формою запису числа називається запис числа у вигляді поліному, представленого у скороченому вигляді:

$$A = a_n a_{n-1} \ldots a_1 a_0 a_{-1} a_{-2} \ldots a_{-k}.$$

При природній формі запису кома ставиться на строго визначеному місці -– між цілою й дробовою частиною числа. При природній формі число записується у природному натуральному вигляді, наприклад: 71405_{10} і 10011_2 — цілі числа; $0{,}004389_{10}$ і $0{,}1001101_2$ — правильні дроби; $87{,}97238_{10}$ і $101{,}00101_2$ — неправильні дроби.

Недоліками запис чисел у формі із природною комою є:

— місце положення коми повинно бути передбачене в кожному розряді, для чого необхідно додаткове устаткування;
— ускладнення арифметичних ланцюгів та труднощі оперування з дуже великими або дуже малими за абсолютною величиною числами.

1.5.2 Запис чисел у формі з фіксованою комою

При даному представленні чисел в ЕОМ встановлюється довжина розрядної сітки, довжина цілої та довжина дробової частин. При цьому розподіл розрядів між цілою та дробовою частинами не змінюється і залишається завжди постійним, незалежно від величини числа.

У машинах з фіксованою комою числа часто представлені у вигляді правильних дробів, тобто кому фіксують перед старшим розрядом числа: $-1 < [A]_{\text{ФК}} < 1$. При цьому числа, більші одиниці приводяться до такого вигляду за допомогою масштабного коефіцієнту $\text{К}_\text{А}$. Запис чисел у вигляді правильних дробів обумовлений необхідністю зменшити можливість переповнення розрядної сітки машини.

Формат (розрядна сітка) для чисел з фіксованою перед старшим розрядом комою розбивається на знакову частину та поле числа, як показано на рисунку 1.1.

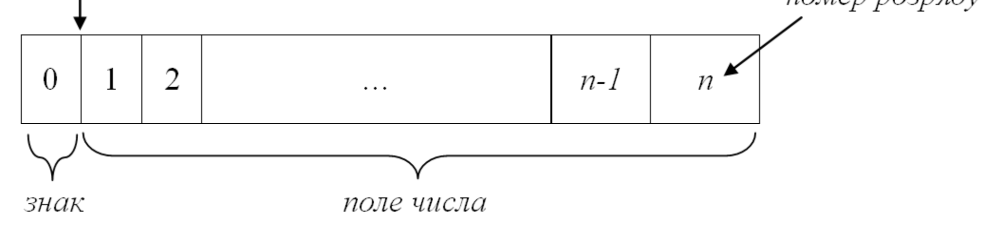


Рисунок 1.1 — Формат розрядної сітки

У знакову частину записується інформація про знак числа. Знак числа звичайно кодується в такий спосіб: знаку «+» відповідає 0 у знаковому розряді, знаку «–» відповідає 1.

Для запису n-значного дробу розрядна сітка повинна містити $n + 1$ розряд.

Величини чисел, що записуються в машинах з фіксованою перед старшим розрядом комою, лежать у межах:

$$2^{-n} \leqslant |A| \leqslant 1 - 2^{-n},$$

де n – кількість розрядів у полі числа.

Нерідко кому фіксують після молодшого розряду числа. Тоді всі дані представляються у вигляді цілих чисел. У цьому випадку також необхідне масштабування вихідних даних.

У розрядній сітці для цілих чисел один розряд виділяється під знак числа, а наступні розряди утворюють поле числа.

Величини чисел, записаних в ЕОМ при фіксації коми після молодшого розряду, лежать у межах

$$1 \leqslant |A| \leqslant 2^n - 1.$$

1.5.3 Запис чисел у формі із плаваючою комою

У машинах із плаваючою комою числа записуються у нормальній формі, у вигляді добутку цілого ступеню основи системи числення та правильного дробу:

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

де q – показник ступеня при основі р називається порядком числа;

m – цифрова дробова частина називається мантисою.

Для запису числа у комірці машини виділяється $k + 1$ розряд для фіксації порядку зі своїм знаком і $n + 1$ розряд для мантиси зі своїм знаком.

Формат машинного зображення числа із плаваючою комою, показано на рисунку 1.2.

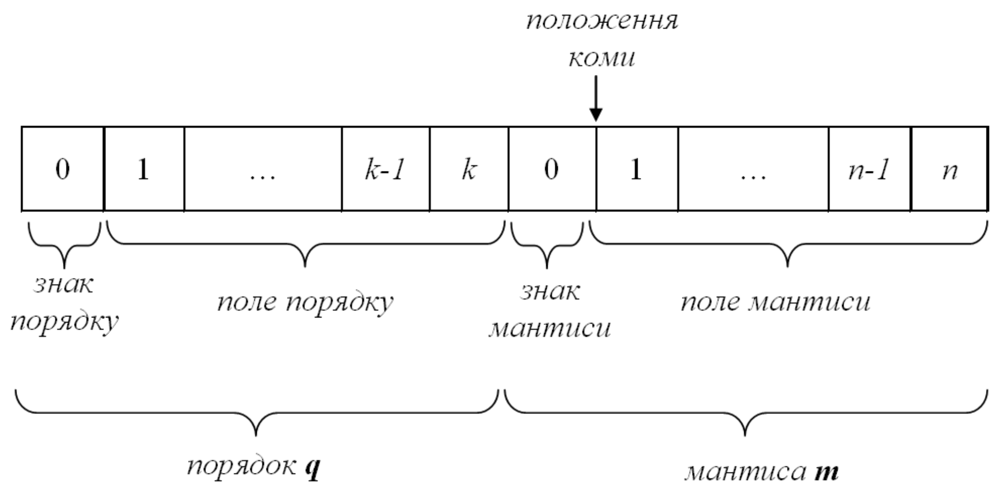


Рисунок 1.2 — Формат розрядної сітки

Для підвищення точності запису чисел мантиси в машинах із плаваючою комою представлені в *нормалізованому вигляді*, при якому діапазон мантис лежить у межах (1.3) для $p = 2$.

$$2^{-1} \leqslant |m| \leqslant 1 - 2^{-n}. \tag{1.3}$$

Тобто старший розряд мантиси є завжди значущий.

Діапазон порядку чисел лежить у межах:

$$-(2^k - 1) \leqslant q \leqslant 2^k - 1.$$

Мінімальне та максимальне числа для p = 2, які можна записати в комірці машини, визначаються в такий спосіб (1.4) (так як n зазвичай велике, то $1 - 2^{-n} \approx 1$)

$$\begin{aligned} |A_{min}| &= |m_{min}| \cdot 2^{-|q_{max}|} = 2^{-1} \cdot 2^{-(2^k-1)} = 2^{-2^k}; \\ |A_{max}| &= |m_{max}| \cdot 2^{+|q_{max}|} = (1 - 2^{-n}) \cdot 2^{(2^k-1)} \approx 2^{(2^k-1)}. \end{aligned} \tag{1.4}$$

1.6 Кодування від'ємних чисел

При виконанні арифметичних операцій в обчислювальних пристроях виникає задача запису від'ємних чисел. Для машинного запису від'ємних чисел використовують коди: прямий, додатковий і зворотний. Позитивні числа не міняють свого зображення в цих кодах. При кодуванні машинне зображення чисел має знакову й цифрову частини.

1.6.1 Прямий код

У прямому коді всі цифрові розряди від'ємного числа залишаються незмінними, а в знаковій частині записується одиниця.

Прямий код правильного дробу $A = -0{,}a_1a_2\ldots a_n$ — це машинне зображення цього числа у вигляді $[A]_{\text{пр}} = 1{,}a_1a_2\ldots a_n$.

Приклад: Записати числа А = −0,101110 і В = 0,110101 у прямому коді:

$$[A]_{\text{пр}} = 1{,}101110; \qquad [B]_{\text{пр}} = 0{,}110101.$$

Прямий код цілого від'ємного числа $A = -a_na_{n-1}\ldots a_1$ — це машинне зображення цього числа у вигляді $[A]_{\text{пр}} = 1a_na_{n-1}\ldots a_1$.

Приклад: Записати числа А = –1101011 і В = 1011101 у прямому коді:

$$[A]_{\text{пр}} = 11101011; \qquad [B]_{\text{пр}} = 01011101.$$

1.6.2 Додатковий код

Кодування правильних дробів

Додатковий код для правильних дробів є математичним доповненням до основи системи числення р:

$$|A| + [A]_{\text{д}} = p,$$

де |А| – абсолютне значення числа А.

Так як позитивні числа не змінюють свого зображення в додатковому коді, то правила перетворення в додатковий код правильних дробів можна записати так:

$$[A]_{\text{д}} = \begin{cases} A, & \text{якщо } A \geqslant 0 \\ p - |A|, & \text{якщо } A < 0 \end{cases}.$$

Нуль у додатковому коді має єдине значення 0,00...00.

Для отримання додаткового коду від'ємного двійкового дробу необхідно в його знаковому розряді записати 1, усі цифри вихідного числа замінити на інверсні (1 на 0 і 0 на 1) і додати одиницю до молодшого розряду перетвореного числа.

Кодування цілих чисел

Функція кодування цілих n-розрядних чисел у додатковому коді записується в такий спосіб:

$$[A]_{\text{д}} = \begin{cases} A, & \text{якщо } A \geqslant 0 \\ p^{n+1} - |A|, & \text{якщо } A < 0 \end{cases}.$$

де n – кількість цифрових розрядів машинного зображення числа;

р – основа системи числення.

Щоб знайти додатковий код цілого від'ємного двійкового числа, треба взяти його позитивну форму, обернути кожен біт (тобто замінити 1 на 0, а 0 на 1), а потім додати до отриманого результату 1.

1.6.3 Зворотній (обернений) код

Кодування правильних дробів

Зворотний код правильного двійкового дробу $A = -0{,}a_1 a_2 \ldots a_n$ – це таке машинне зображення цього числа $[A]_{\text{об}} = 1{,}\bar{a}_1\bar{a}_2 \ldots \bar{a}_n$, для якого $\bar{a}_i = 1$, якщо $a_i = 0$ і $\bar{a}_i = 0$, якщо $a_i = 1$.

У зворотному коді від'ємного правильного двійкового дробу в знаковому розряді записується 1, усі розряди вихідного числа приймають інверсне (зворотне) значення, тобто всі нулі замінюються на одиниці, а всі одиниці – на нулі.

У зворотному коді максимальне позитивне дрібне число записується у вигляді $[A]_{\text{об}} = 0{,}11111 \ldots 111$, а найбільше дрібне *негативне* число $A = -0{,}11111 \ldots 111$ — у вигляді $[A]_{\text{об}} = 1{,}00 \ldots 00$.

Кодування цілих чисел

Зворотний код цілого від'ємного двійкового числа є інверсним зображенням самого числа, у якому всі розряди вихідного числа приймають інверсне (зворотне) значення, тобто всі нулі замінюються на одиниці, а всі одиниці – на нулі.

1.7 Операція додавання

1.7.1 Додавання чисел в прямому коді

Додавання чисел в прямому коді відбувається за наступними правилами.

Якщо обидва доданки мають однакові знаки, то їх числові розряди додаються у відповідності до таблиці 1.1, а в знаковому розряді суми записується знак одного з них. Якщо доданки мають різні знаки, то з числових розрядів більшого за абсолютною величиною числа віднімається менше, а в знаковому розряді результату записується знак числа, більшого за абсолютною величиною.

При додаванні чисел в прямому коді числові розряди коду обробляються окремо від знакових.

Ознака переповнення – наявність одиниці переносу із старшого розряду цифрової частини. В цьому випадку повинен вироблятися сигнал переповнення, за яким відбувається автоматичне зупинення машини та корекція масштабних коефіцієнтів.

1.7.2 Додавання чисел в додатковому коді

У додатковому коді операція віднімання замінюється операцією алгебраїчного додавання. При алгебраїчному додаванні в додатковому коді знакова і цифрова частини числа розглядаються як одне ціле. Машина оперує з кодами як з неправильними дробами. Знак суми виходить автоматично, як результат додавання вмісту знакових розрядів операндів та одиниці переносу з цифрової частини, якщо вона є.

Сумою додаткових кодів чисел є додатковий код результату.

Якщо додаються два числа з протилежними знаками, з яких по абсолютній величині більше позитивне, то результат виходить в прямому коді. Отримана сума відрізняється від істинної на 2. Корекція на 2 здійснюється автоматично, оскільки для зображення 2 в розрядній сітці ЕОМ немає місця.

Ознакою позитивного переповнення є наявність переносу з цифрової частини в знаковий розряд суми за відсутності переносу з її знакового розряду.

Ознакою негативного переповнення є наявність переносу із знакового розряду суми за відсутності переносу в її знаковий розряд з цифрової частини.

Таким чином, переповнення відсутнє, якщо обидва переноси є чи ні. Переповнення присутнє, якщо є тільки один з переносів.

Для виявлення переповнення розрядної сітки вводять допоміжний розряд в знакову частину зображення числа, який називають розрядом переповнення. Таке представлення числа називається **мо-**

дифікованим. Модифікований додатковий код відрізняється від звичайного використанням двох знакових розрядів.

Знак «+» числа кодується як 00, знак «−» – як 11.

Ознакою переповнення розрядної сітки при складанні в модифікованому додатковому коді буде відмінність знакових розрядів суми. Лівий знаковий розряд завжди зберігає правильний код знаку результату. Тому для усунення виникнувшого переповнення необхідно зсунути результат на один розряд вправо, власне збільшивши масштабний коефіцієнт, а в правому знаковому розряді результату продублювати вміст лівого знакового розряду.

1.7.3 Додавання чисел в зворотному коді

У зворотному коді, як і в додатковому, операція віднімання замінюється операцією складання. Знаковий розряд та цифрова частина числа розглядаються як єдине ціле. Машина оперує з кодами, як з неправильними дробами. Правильний знак результату виходить автоматично в процесі додавання цифр знакових розрядів операндів і одиниці переносу з цифрової частини, якщо вона є. Характерною особливістю зворотного коду є наявність циклічного переносу, якщо він виникає, із знакового розряду в молодший розряд цифрової частини.

Сума позитивна, вона виходить в прямому коді. Отримана сума відрізняється від істинної на $(2-2^{-n})$. Корекція на 2 виходить автоматично, оскільки для зображення 2 в розрядній сітці ЕОМ немає місця. Корекцію на 1 молодшого розряду отримують шляхом додавання до молодшого розряду суми одиниці переповнення, яка виникає при додаванні цифр знакових розрядів. Це виконується так званий циклічний перенос.

Для спрощення виявлення переповнення розрядної сітки в ЕОМ використовується модифікований зворотний код. Ознакою переповнення в цьому випадку є різні цифри в знакових розрядах суми.

1.8 Операція множення

1.8.1 Множення двійкових чисел у прямому коді

Множення двійкових чисел найбільш просто виконується в прямому коді. Знак добутку визначається шляхом додавання знакових цифр співмножників по модулю 2.

Процесом накопичення суми часткових добутків можна керувати за допомогою цифр множника відповідно до виразу:

$$C = A \cdot B = A \cdot b_1 \cdot 2^{-1} + A \cdot b_2 \cdot 2^{-2} + \ldots + \\ + A \cdot b_{n-i} \cdot 2^{-(n-i)} + \ldots + A \cdot b_n \cdot 2^{-n} = A \cdot \sum_{i=0}^{n-1} b_{n-i} \cdot 2^{-(n-i)}, \quad (1.5)$$

де C – шуканий добуток; A – множене; B – множник.

З формули (1.5) видно, що керування процесом множення може починатися як з молодших розрядів множника, так й зі старших. При цьому повну суму (добуток) можна отримати двома шляхами:

a) зсувом множеного на необхідну кількість розрядів та додаванням отриманого чергового часткового добутку до раніше накопиченої суми;
b) зсувом суми раніше отриманих часткових добутків на кожному кроці на один розряд та наступним додаванням до зсунутої суми нерухомого множеного або 0.

Це дає підставу для виникнення різних методів реалізації операції множення.

Метод 1 – множення молодшими розрядами множника зі зсувом накопичуваної суми часткових добутків вправо.

Вираз (1.5) для знаходження добутку C = A · B зводиться до n-кратного виконання циклу за наступним виразом:

$$C_{i+1} = (C_i + A \cdot b_{n-i}) \cdot 2^{-1}$$

за початкових умов $i = 0; C_0 = 0$.

Процес формування добутку починається з обнулення суми часткових добутків $C_0 = 0$. У кожному циклі множене A або додається до суми часткових добутків (якщо $b_i = 1$) або ні (якщо $b_i = 0$). Після цього сума часткових добутків зсувається на один розряд вправо.

По закінченню n-го циклу утворюється шуканий добуток $\mathrm{C}_n = \mathrm{C} = \mathrm{A} \cdot \mathrm{B}$.

Метод 2 – множення молодшими розрядами множника зі зсувом множеного вліво.

Для даної схеми обчислення виразу (1.5) зводиться до n-кратного виконання циклу за наступним виразом:

$$C_{i+1} = C_i + A_i \cdot b_{n-i},$$

де $\mathrm{A}_i = 2 \cdot \mathrm{A}_{i-1}$, за початкових значень $i = 0; C_0 = 0; A_0 = \mathrm{A}$.

У кожному циклі, за винятком першого, множене зсувається на один розряд вліво та передається в суматор при $b_i = 1$, або ні при $b_i = 0$.

Метод 3 – множення старшими розрядами множника зі зсувом суми часткових добутків вліво.

Для даної схеми обчислення виразу (1.5) зводиться до n-кратного виконання циклу за наступним виразом:

$$C_{i+1} = (C_i + A \cdot b_{n+1}) \cdot 2,$$

з початковими значеннями $i = 0; \mathrm{C}_0 = 0$.

Процес формування добутку починається з обнулення суми часткових добутків $\mathrm{C}_0 = 0$. У кожному циклі сума часткових добутків зсувається на один розряд вліво, і до неї або додається множене А при $b_i = 1$, або ні при $b_i = 0$.

Метод 4 – множення старшими розрядами множника зі зсувом множеного вправо.

Для даної схеми обчислення виразу (1.5) зводиться до n-кратного виконання циклу за наступним виразом:

$$C_{i+1} = C_i + A_{i+1} \cdot b_{n+1},$$

де $\mathrm{A}_{i+1} = A_i \cdot 2^{-1}$, за початкових умов $i = 0; \mathrm{A}_0 = \mathrm{A}; \mathrm{C}_0 = 0$.

У кожному циклі, *у тому числі й у першому*, множене зсувається на один розряд вправо й в залежності від значення старшого керуючого розряду множника або передається в суматор ($b_i = 1$), або ні ($b_i = 0$).

1.9 Операція ділення двійкових чисел

Найбільш просто ділення в ЕОМ виконується в прямому коді. Знак частки при цьому визначається як сума по модулю 2 знакових цифр діленого і дільника. Частка визначається шляхом ділення модулів початкових чисел. Для реалізації алгоритму ділення двійкових чисел, представлених у формі з фіксованою комою, необхідно, щоб виконувалася умова $|\mathrm{A}| < |\mathrm{B}|$, де A – ділене, B – дільник. Це забезпечить відсутність переповнення розрядної сітки ЕОМ з фіксованою комою.

Розглянемо два основних алгоритми виконання ділення в двійковій системі: ділення чисел з відновленням залишку і ділення без відновлення залишку.

1.9.1 Ділення з відновленням залишків

Нехай потрібно розділити A на B з точністю до i-го розряду, тобто

$$C_i = \frac{A}{B} = 0{,}c_1c_2c_3\ldots c_i$$

При будь-якому значенні i залишок від ділення $r = (A - B \cdot C_i)$ не може бути від'ємним та більшим або рівним дільнику B, помноженому на 2^{-i}, тобто $0 < r < B \cdot 2^{-i}$. Перенесемо 2^{-i} по інший бік нерівності та запишемо останню нерівність у вигляді:

$$0 \leqslant R_i < B, \tag{1.6}$$

де $R_i = (A - B \cdot C_i) \cdot 2^i$ – залишок (*англ.* remain).

Цифри частки визначаються послідовно, починаючи зі старшого розряду. Враховуючи те, що частка є двійковим числом, поточний розряд частки може бути тільки нулем або одиницею.

Розглянемо як зміниться нерівність (1.6) на якомусь $(i+1)$ кроці в двох випадках: коли значення поточного розряду частки $c_{i+1} = 1$ та коли $c_{i+1} = 0$.

Якщо отриманий розряд частки $c_{i+1} = 0$, то частка $C_{i+1} = C_i$. Тоді залишок

$$R_{i+1} = (A - B \cdot C_{i+1}) \cdot 2^{i+1} = (A - B \cdot C_i) \cdot 2^i \cdot 2 = 2R_i. \tag{1.7}$$

Звідси нерівність (1.6) приймає наступний вигляд:

$$0 \leqslant R_{i+1} < B.$$

$$0 \leqslant 2R_i < B. \tag{1.8}$$

Якщо отриманий розряд частки $c_{i+1} = 1$, то

$$C_{i+1} = C_i + 2^{-(i+1)}.$$

Отже залишок

$$\begin{aligned} R_{i+1} = (A - BC_{i+1}) \cdot 2^{i+1} = \\ = (A - B(C_i + 2^{-(i+1)})) \cdot 2^{i+1} = \\ = (A - BC_i - B \cdot 2^{-(i+1)}) \cdot 2^{i+1} = \\ = (A - BC_i) \cdot 2^{i+1} - B \cdot 2^{-(i+1)} \cdot 2^{i+1} = \\ = (A - BC_i) \cdot 2^i \cdot 2 - B = 2R_i - B. \quad (1.9) \end{aligned}$$

Звідси нерівність (1.6) приймає наступний вигляд:

$$0 \leqslant 2R_i - B < B,$$

або

$$B \leqslant 2R_i < 2B. \quad (1.10)$$

Отже, для визначення поточної цифри частки досить перевірити одну з умов (1.8) або (1.10). Зазвичай перевіряється умова (1.8). Ліва частина цієї нерівності виконується заздалегідь, згідно (1.6). Для перевірки правої частини виконується порівняння виразу $(2R_i - B)$ з нулем.

Якщо ця різниця від'ємна, то в $(i+1)$-ому розряді частки записується 0. При цьому для наступного $(i+2)$-го кроку ділення вихідними даними є поточний залишок $R_{i+1} = (2R_i - B)$, який, відповідно до (1.7), необхідно відновити до $2R_i$ наступним чином:

$$R'_{i+1} = (2R_i - B) + B = 2R_i.$$

Якщо різниця $(2R_i - B)$ додатна, то в $(i+1)$-ому розряді частки записується 1.

А в якості вихідних даних для наступного $(i+2)$-го кроку ділення, відповідно до (1.9), використовується вже вирахуване значення поточного залишку $R_{i+1} = (2R_i - B)$.

Вихідними даними для 1-го кроку ділення є $\mathrm{C}_0 = 0$ і $R_0 = A$.

Виходячи з вищесказаного сформулюємо алгоритм ділення двійкових чисел з відновлення залишків:

1) Дільник віднімається від діленого й визначається знак нульового (по порядку) залишку.
   — Якщо *залишок позитивний*, тобто $|A| > |B|$, то в псевдознаковому розряді частки проставляється **1**, з появою якої формується ознака ***переповнення розрядної сітки ЕОМ***, й операція ділення припиняється.

— Якщо *залишок негативний*, то в псевдознаковому розряді частки записується **0**, а потім проводиться відновлення діленого шляхом додавання до залишку дільника.

2) Далі виконується зсув відновленого діленого на один розряд вліво (множення на 2) й повторне віднімання дільника. Знак одержуваного в такий спосіб залишку визначає першу значущу цифру частки:
   — якщо *залишок позитивний*, то в першому розряді частки записується **1**,
   — якщо *залишок негативний*, то записується **0**.
3) Далі, якщо залишок позитивний, то він зсувається вліво на 1 розряд і від нього віднімається дільник для визначення наступної цифри частки.
4) Якщо залишок негативний, то до нього додається дільник *для відновлення попереднього залишку*, а потім відновлений залишок зсувається на 1 розряд вліво й від нього віднімається дільник для визначення наступної цифри частки.
5) Далі повторюємо кроки 3 та 4 до одержання необхідної кількості цифр частки з врахуванням одного додаткового розряду для округлення (для забезпечення необхідної точності ділення).

1.9.2 Ділення без відновлення залишків

Спосіб ділення з відновленням залишків є аритмічним процесом зі змінним числом кроків того чи іншого виду: 3 кроки при $2R_i < B$ і 2 кроки при $2R_i > B$.

Алгоритм ділення без відновлення залишків дозволяє спростити операцію ділення і отримати кожну цифру частки за 2 кроки.

Алгоритм наступний:

1) Дільник віднімається від діленого й визначається знак нульового (по порядку) залишку.
   — Якщо *залишок позитивний*, тобто $|A| > |B|$, то в псевдознаковому розряді частки проставляється **1**, з появою якої формується ознака ***переповнення розрядної сітки ЕОМ***, й операція ділення припиняється.
   — Якщо *залишок негативний*, то в псевдознаковому розряді частки записується **0**, а потім проводиться відновлення діленого шляхом додавання до залишку дільника.

2) Далі, щоб визначити наступну цифру частки, необхідно зсунути поточний залишок вліво на один розряд, а потім алгебраїчно додати до нього *модуль* дільника, якому приписується знак, *протилежний знаку поточного залишку.*
   Знак, отриманого в такий спосіб наступного залишку, визначає наступну цифру частки:
   — якщо *залишок позитивний*, то в частці записується **1**,
   — якщо *залишок негативний*, то записується **0**.
3) Далі повторюємо крок 2 поки не одержимо необхідної кількості цифр частки з врахуванням одного додаткового розряду для округлення (для забезпечення необхідної точності ділення).

2 Завдання на розрахунково-графічну роботу та вимоги до її оформлення

2.1 Вимоги до виконання та оформлення РГР

1) Завдання виконуються в зошиті в «клітинку» власноручно кожним студентом без використання комп'ютерних засобів.
2) Кожне завдання оформляється на окремому аркуші. На початку вказується саме завдання, далі йде його розв'язок, наприкінці приводиться відповідь із кінцевим значенням, отриманим у результаті виконання завдання. Після відповіді виконується перевірка отриманого результату кожним з відомих методів. Ці вимоги не діють, якщо результатом є таблиця або послідовність операцій.
3) У постановці завдання використовуються наступні позначення:
   а) G — остання цифра з номера групи;
   б) XX — останні дві цифри з номера залікової книжки.
4) Точність операцій із правильними й неправильними дробами — **6** знаків після коми.
5) У відповіді контролюються всі знаки із зазначеною точністю, якщо хоча б в одному знаку не буде відповідності правильній відповіді, завдання вважається не виконаним.

2.2 Завдання на РГР

Завдання 1. Для двійкової зсунутої системи числення із цифрами $\{0, 1\}$ і зі штучним порядком ваг, що задаються як $G19XX$, утворювати всі можливі цілі позитивні числа.

Завдання 2. Утворити цілі десяткові числа A, B, C і D за наступними правилами:

$$\begin{aligned}
A &= 2^7 + G \cdot 2^4 + XX;\\
B &= 2^8 - G \cdot 2^4 - XX;\\
C &= 2^7 + G \cdot 2^4 + H;\\
D &= 2^8 - G \cdot 2^4 - H;
\end{aligned} \tag{2.1}$$

Тут H дорівнює сумі цифр у числі $G19XX$. *Наприклад*, якщо $G = 3, XX = 13$, то $H = 3+1+9+1+3 = 17$, А $= 128+3\cdot16+13 = 189$.

Утворити дробові десяткові числа X, Y, V і W за наступними правилами:

$$\begin{aligned} X &= A \cdot 2^{-8}; \\ Y &= B \cdot 2^{-8}; \\ V &= C \cdot 2^{-8}; \\ W &= D \cdot 2^{-8}; \end{aligned} \tag{2.2}$$

обмеживишсь 6-ю десятковими цифрами після коми.

Завдання 3. Перевести цілі числа A, B, C и D у двійкову систему числення.

Завдання 4. Перевести дробові числа X,Y,V і W у двійкову систему числення, обмеживишсь у всіх випадках 6-ю двійковими розрядами після коми.

Завдання 5. Перевести, у системи з основами 8 і 16 двійкові числа A, B, C, D, X, Y, V, W, отримані при виконанні завдань 3 і 4.

Завдання 6. Записати в прямому, зворотному й додатковому кодах двійкові числа $+A, -A, +B, -B, \ldots, +W, -W$, отримані при виконанні завдань 3 і 4.

Результат представити у вигляді таблиці 2.1.

Таблиця 2.1 — Запис чисел в різних кодах

Числа	Прямий код	Зворотній код	Додатковий код
$+A$			
$-A$			
$+B$			
$-B$			
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Завдання 7. Виконати операцій $A+B$, $-B+C$, $C-D$, $-C-A$, $X+Y$, $-Y+V$, $V-W$, $-W-X$ з використанням модифікованих зворотного й додаткового кодів. Відзначити можливі випадки переповнення розрядної сітки.

Завдання 8. Помножити двійкові числа X на Y і V на W. (*Примітка*: у кожній парі операндів першим зазначений множник, другим — множене).

Завдання 9. Виконати операцію ділення в додатковому коді чисел $-X$ і W, а також V і $-Y$ двома методами: з відновленням залишку і без відновлення залишку. У якості діленого в кожній парі вибрати менше по абсолютній величині число, а ділення проводити до одержання дев'яти цифр частки після коми, результат потім округлити до восьми цифр.

3 Приклад виконання розрахунково-графічної роботи

Розглянемо виконання розрахунково-графічної роботи на прикладі варіанта: $G = 6$, $XX = 31$. В кожному завданні буде розглянуто тільки по одному прикладу, але при виконанні РГР необхідно буде виконати і розписати абсолютно всі приклади з кожного завдання.

3.1 Завдання 1

Відповідно до 1-го завдання необхідно записати всі можливі числа у двійковій позиційній системі числення зі штучним порядком ваг, що задається як $G19XX$. За наведеним варіантом порядок ваг розрядів: 6 1 9 3 1. Необхідні числа отримують відповідно до (1.1) якщо замість p^i підставити значення ваг за варіантом.

Наприклад число 10 можна розкласти на розряди відповідно до наступного виразу:

$$a_4 \cdot G + a_3 \cdot 1 + a_2 \cdot 9 + a_1 \cdot X + a_0 \cdot X = \\ = 1 \cdot 6 + 0 \cdot 1 + 0 \cdot 9 + 1 \cdot 3 + 1 \cdot 1 = 10_{10}. \quad (3.1)$$

Як видно з (3.1) число 10_{10} можна отримати ще трьома комбінаціями двійкового коду змінюючи значення a_i. Усі можливі п'ятирозрядні двійкові числа в данній системі числення та відповідні їм десяткові числа наведені в таблиці 3.1.

3.2 Завдання 2

Відповідно до варіанту $G = 6$, $XX = 31$ розрахуємо вихідні числа за рівняннями (2.1), (2.2). Отримаємо наступне:

$$A = 2^7 + 6 \cdot 2^4 + 31 = 255_{10};$$
$$B = 2^8 - 6 \cdot 2^4 - 31 = 129_{10};$$
$$C = 2^7 + 6 \cdot 2^4 + 20 = 244_{10};$$
$$D = 2^8 - 6 \cdot 2^4 - 20 = 140_{10};$$

Таблиця 3.1 — Числа в двійковій системі зі штучним порядком ваг

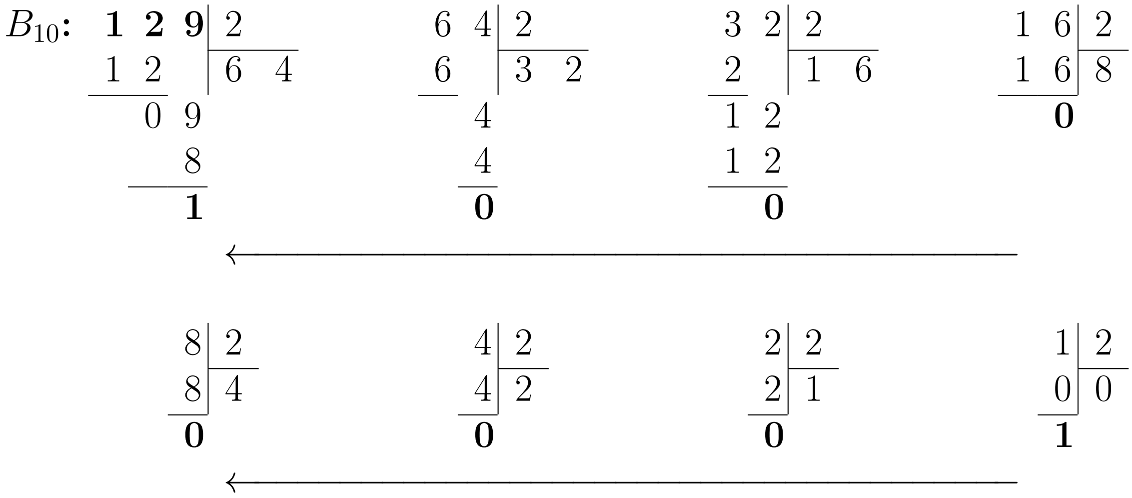
Dec	6-1-9-3-1	*Dec*	6-1-9-3-1
0	0 0 0 0 0	11	0 1 1 0 1
1	0 1 0 0 0		1 1 0 1 1
	0 0 0 0 1	12	0 0 1 1 0
2	0 1 0 0 1	13	0 1 1 1 0
3	0 0 0 1 0		0 0 1 1 1
4	0 1 0 1 0	14	0 1 1 1 1
	0 0 0 1 1	15	1 0 1 0 1
5	0 1 0 1 1	16	1 1 1 0 0
6	1 0 0 0 0		1 0 1 0 1
7	1 1 0 0 0	17	1 1 1 0 1
	1 0 0 0 1	18	1 0 1 1 0
8	1 1 0 0 1	19	1 1 1 1 0
9	0 0 1 0 1		1 0 1 1 1
	1 0 0 1 0	20	1 1 1 1 1
10	0 1 1 0 0		
	0 0 1 0 1		
	1 1 0 1 0		
	1 0 0 1 1		

$$X = 255 \cdot 2^{-8} = 0{,}996094_{10};$$
$$Y = 129 \cdot 2^{-8} = 0{,}503906_{10};$$
$$V = 244 \cdot 2^{-8} = 0{,}953125_{10};$$
$$W = 140 \cdot 2^{-8} = 0{,}546875_{10}.$$

3.3 Завдання 3

Відповідно до правила наведеного в підрозділі 1.4.1 для переведення цілого числа в іншу систему числення необхідно богаторазово ділити це число на основу нової системи числення поки частка не буде дорівнювати 0. Результат записується як послідовність залишків від ділення починаючи з останнього.

Приклад переведення цілого числа наведено на рисунку 3.1.



$$B = \mathbf{10000001}_2$$

Рисунок 3.1 — Переведення цілого десяткового числа в двійкову систему числення

Перевірка: Перевірку виконаємо шляхом розкладання отриманого двійкового числа в поліном відповідно до (1.1). Дивись вираз (3.2).

Перевірка:

$$\begin{aligned} B = 1 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + \\ + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 128 + 1 = 129_{10}. \end{aligned} \quad (3.2)$$

Для решти цілих чисел запишемо зразу результат.

$$A = 255_{10} = 11111111_2;$$
$$B = 129_{10} = 10000001_2;$$
$$C = 244_{10} = 11110100_2;$$
$$D = 140_{10} = 10001100_2.$$

3.4 Завдання 4

Відповідно до правила наведеного в підрозділі 1.4.2 для переведення правильного дробу в іншу систему числення необхідно багаторазово множити цей дріб на основу нової системи числення поки не набереться необхідна кількість розрядів. Результат записується як послідовність цілих частин, що з'являлися під час множення, по мірі їх виникнення.

Приклад переведення правильного дробу наведено на рисунку 3.2.

$$
\begin{array}{rl}
0, & 996094_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 992188_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 984376_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 968752_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 937504_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 975008_{10} \\
\times & 2 \\
\hline
\mathbf{1}, & 950016_{10}
\end{array}
$$

$$\text{Отже:} \quad X = 996094_{10} \approx 0{,}111111_2$$

Рисунок 3.2 — Переведення десяткового правильного дробу в двійкову систему числення

Перевірка: Перевірку виконаємо шляхом розкладання отриманого двійкового числа в поліном.

$$
\begin{aligned}
B = 1 \cdot 2^{-1} + 0 \cdot 2^{-1} + 0 \cdot 2^{-1} + 0 \cdot 2^{-1} + 0 \cdot 2^{-1} + 0 \cdot 2^{-1} = \\
= 0.5 + 0.25 + 0.125 + 0.0625 + 0.03125 + 0.015625 = \\
= 0.984375_{10}.
\end{aligned}
$$

Для решти дробів запишемо зразу результат:

$$
\begin{aligned}
X &= 0{,}996094_{10} = 0{,}111111_2; \\
Y &= 0{,}503906_{10} = 0{,}100000_2; \\
V &= 0{,}953125_{10} = 0{,}111101_2; \\
W &= 0{,}546875_{10} = 0{,}100011_2.
\end{aligned}
$$

3.5 Завдання 5

Відповідно до правила наведеного в підрозділі 1.4.4 для переведення двійкового числа до вісімкової системи числення необхідно відповідно до таблиці 1.3, починаючи від коми кожні 3 двійкові розряди ($2^3 = 8$) замінити одним вісімковим розрядом. А для переведення двійкового числа до вісімкової системи числення необхідно відповідно до тієїж таблиці 1.3, починаючи від коми, кожні 4 двійкові розряди ($2^4 = 16$) замінити одним шістнадцятковим розрядом.

$$\begin{array}{rlccclccc}
C_2 & = & \underbrace{011} & \underbrace{110} & \underbrace{100} & W_2 & = 0, & \underbrace{100} & \underbrace{011} \\
 & & \downarrow & \downarrow & \downarrow & & & \downarrow & \downarrow \\
C_8 & = & 3 & 6 & 4 & W_8 & = 0, & 4 & 3 \\
 & & & & & & & & \\
C_2 & = & & \underbrace{1111} & \underbrace{0100} & W_2 & = 0, & \underbrace{1000} & \underbrace{1100} \\
 & & & \downarrow & \downarrow & & & \downarrow & \downarrow \\
C_{16} & = & & F & 4 & W_{16} & = 0, & 8 & C
\end{array}$$

Отже:

$$C = 11110100_2 = 364_8 = F4_{16}; \quad W = 0{,}100011_2 = 0{,}43_8 = 0{,}8C_{16}.$$

Для решти чисел запишемо зразу результат:

$$\begin{aligned}
A &= 11111111_2 &= 377_8 &= FF_{16};\\
B &= 10000001_2 &= 201_8 &= 81_{16};\\
C &= 11110100_2 &= 364_8 &= F4_{16};\\
D &= 10001100_2 &= 214_8 &= 8C_{16};\\
X &= 0{,}111111_2 &= 0{,}77_8 &= 0{,}FC_{16};\\
Y &= 0{,}100000_2 &= 0{,}40_8 &= 0{,}80_{16};\\
V &= 0{,}111101_2 &= 0{,}75_8 &= 0{,}F4_{16};\\
W &= 0{,}100011_2 &= 0{,}43_8 &= 0{,}8C_{16}.
\end{aligned}$$

3.6 Завдання 6

Відповідно до правил наведених в підрозділі 1.6 необхідно записати всі числа, що були отримані в завданнях 3 та 4, та їх від'ємні значення в прямому, оберненому та додатковому кодах.

Результат виконання завдання необхідно оформити у вигляді таблиці 3.2.

Таблиця 3.2 — Прямі, зворотні та додаткові коди

Числа	Прямий код	Зворотній код	Додатковий код
$+A$	0 11111111	0 11111111	0 11111111
$-A$	1 11111111	1 00000000	1 00000001
$+B$	0 10000001	0 10000001	0 10000001
$-B$	1 10000001	1 01111110	1 01111111
$+C$	0 11110100	0 11110100	0 11110100
$-C$	1 11110100	1 00001011	1 00001100
$+D$	0 10001100	0 10001100	0 10001100
$-D$	1 10001100	1 01110011	1 01110100
$+X$	0,111111	0,111111	0,111111
$-X$	1,111111	1,000000	1,000001
$+Y$	0,100000	0,100000	0,100000
$-Y$	1,100000	1,011111	1,100000
$+V$	0,111101	0,111101	0,111101
$-V$	1,111101	1,000010	1,000011
$+W$	0,100011	0,100011	0,100011
$-W$	1,100011	1,011100	1,011101

3.7 Завдання 7

Правила виконання операції додавання розписані в підрозділі 1.7. Наведемо приклад виконання операції додавання для цілих чисел ($A + B$, $-B + C$) з використанням модифікованих оберненого (зворотнього) та додаткового кодів.

$$
\begin{array}{cllll}
 & [A]^{\text{м}}_{\text{д}} & = & 00 & 11111111 \\
+ & [B]^{\text{м}}_{\text{д}} & = & 00 & 10000001 \\
\hline
 & [Res]^{\text{м}}_{\text{д}} & = & 01 & 10000000 \\
 & & & \uparrow &
\end{array}
$$

Переповнення в додатньому результаті

$$
\begin{array}{clllll}
 & [-B]^{\text{м}}_{\text{об}} & = & & 11 & 01111110 \\
+ & [C]^{\text{м}}_{\text{об}} & = & & 00 & 11110100 \\
\hline
 & & & \boxed{1} & 00 & 01110010 \\
+ & & & & & \longrightarrow 1 \\
\hline
 & [Res]^{\text{м}}_{\text{об}} & = & & 00 & 01110011
\end{array}
$$

При винекненні переповнення розрядної сітки необхідно збільшити маштабний коефіцієнт K і зсунути результат на один розряд вправо та відновити лівий знаковий розряд:

$$K = 2^1; \qquad [(A+B)\cdot 2^1]^{\text{м}}_{\text{д}} = 00\ 11000000{,}0 \cdot 2^1.$$

Аналогічно для дробових чисел $(-Y+V,\ -X-W)$:

$$\begin{array}{rrrl} & [-Y]^{\text{м}}_{\text{об}} = & 11, & 011111 \\ + & [V]^{\text{м}}_{\text{об}} = & 00, & 111101 \\ \hline & \boxed{1} & 00, & 011100 \\ + & & \longrightarrow & 1 \\ \hline & [Res]^{\text{м}}_{\text{об}} = & 00, & 011101 \end{array} \qquad \begin{array}{rrrl} & [-W]^{\text{м}}_{\text{д}} = & 11, & 011101 \\ + & [-X]^{\text{м}}_{\text{д}} = & 11, & 000001 \\ \hline & [Res]^{\text{м}}_{\text{д}}\ \ 1 & 10, & 011110 \\ & & \uparrow & \end{array}$$

Переповнення у від'ємному результаті

Знову ж таки, при винекненні переповнення розрядної сітки необхідно збільшити маштабний коефіцієнт K і зсунути результат на один розряд вправо та відновити лівий знаковий розряд:

$$K = 2^1; \qquad [(-W-X)\cdot 2^1]^{\text{м}}_{\text{д}} = 11,\ 001111 \cdot 2^1.$$

Аналогічно інші пари чисел.

3.8 Завдання 8

Операцію множення розберемо на прикладі множення чисел W і V за 1-м методом «Множення молодшими розрядами множника зі зсувом накопичуваної суми часткових добутків вправо». У якості множника візьмемо число W. Процес множення повинен бути оформлений у вигляді таблиці 3.3.

Отже: $Res = W \cdot V = 0{,}100001010111_2$.

3.9 Завдання 9

Операцію ділення розберемо на прикладі ділення чисел $-X$ та W. В якості дільника виберемо число X, тому що воно більше ніж W по абсолютній величині. Відповідно до завдання необхідно виконати ділення двома способами. Розглянемо кожен з них окремо.

Таблиця 3.3 — Множення W на V за 1-м методом

Регистр W	Дія	CM, Регистр V	
0,10001 **1**	$CM = 0$	0,0000 0000 0000	CM
		0,1111 01	V
	$CM + V$	0,1111 0100 0000	CM
	$CM \longrightarrow$	0,0111 1010 0000	CM
0,1000 **1**	$W \longrightarrow$	0,1111 01	V
	$CM + V$	1,0110 1110 0000	CM
	$CM \longrightarrow$	0,1011 0111 0000	CM
0,100 **0**	$W \longrightarrow$		
	$CM + 0$		
	$CM \longrightarrow$	0,0101 1011 1000	CM
0,10 **0**	$W \longrightarrow$		
	$CM + 0$		
	$CM \longrightarrow$	0,0010 1101 1100	CM
0,1 **0**	$W \longrightarrow$		
	$CM + 0$		
	$CM \longrightarrow$	0,0001 0110 1110	CM
0, **1**	$W \longrightarrow$	0,1111 01	V
	$CM + V$	1,0000 1010 1110	CM
	$CM \longrightarrow$	0,1000 0101 0111	CM

Ділення з відновленням залишків

1) Визначимо знак частки (результату) шляхом додавання по модулю 2 знакових розрядів діленого та дільника:

$$1 \oplus 0 = 1.$$

Одиниця означає що результат ділення від'ємний.

2) Визначимо наявність/відсутність переповнення розрядної сітки ЕОМ. Для цього алгебраїчно додамо до діленого дільник, якому приписується знак протилежний знаку діленого.

$$\begin{array}{cccc} & [W]_{\text{д}} & = & 0,\ 100011 \\ + & [-X]_{\text{д}} & = & 1,\ 000001 \\ \hline & R_0 & = & \boxed{1},\ 100100 \end{array}$$

Отримали від'ємне число, знаковий розряд якого записується у псевдознаковий розряд частки. При цьому, якщо ділене додатне, то знаковий розряд отриманого числа інвертується і тільки потім записується у псевдознаковий розряд частки. А якщо ділене від'ємне, то знаковий розряд отриманого числа записується в псевдознаковий розряд частки без інверсії.
В нашому випадку ділене додатне, тому у псевдознаковий розряд частки заносимо 0 (знаковий розряд R_0-го з інверсією).

$$[Res] = 0, \ldots$$

Нуль у псевдознаковому розряді частки означає *відсутність переповнення розрядної сітки ЕОМ*. Можемо продовжувати розрахунки.

3) Отримане на попередньому кроці число R_0 є від'ємним, тому необхідно його відновити до позитивного шляхом додавання модуля дільника.

$$\begin{array}{rlcrl} & R_0 & = & 1, & 100100 \\ + & [X]_{\text{д}} & = & 0, & 111111 \\ \hline & R_0' & = & 10, & 100011 \end{array}$$

Усі перенесення із знакового розряду у додатковому коді відкидаються.
Тепер, за алгоритмом, необхідно відновлений залишок R_0' зсунути вліво на один розряд (що еквівалентно множенню числа на 2) і потім, для визначення наступного залишку, відняти від нього модуль дільника.

$$\begin{array}{rlcrl} & 2 \cdot R_0' & = & 1, & 000110 \\ + & [-X]_{\text{д}} & = & 1, & 000001 \\ \hline & R_1 & = & 1\boxed{0}, & 000111 \end{array}$$

Знаковий розряд отриманого залишку R_1 заносимо до першого числового розряду частки. При цьому, якщо дільник від'ємний, то знаковий розряд залишку переноситься без інверсії. А якщо дільник додатній, то знаковий розряд залишку переноситься з інверсією.
В нашому випадку дільник від'ємний і тому усі знакові розряди наступних залишків записуються до результату без інверсії.

$$[Res] = 0,\boxed{0}\ldots$$

Поточний залишок R_1 є додатнім і тому його відновлювати не треба.
Далі зсуваємо залишок на один розряд, тобто множимо на 2, і віднімаємо від нього дільник для визначення наступного розряду частки.
Наступні кроки розрахунку числових розрядів частки покажемо у вигляді таблиці 3.4.

Таблиця 3.4 — Розрахунок числових розрядів частки

Розрахунки	Частка
$\begin{array}{llcrl} & 2\cdot R_1 & = & 0, & 001110 \\ + & [-X]_д & = & 1, & 000001 \\ \hline & R_2 & = & \boxed{1}, & 001111 \end{array}$	$[Res] = 0{,}0\boxed{1}\ldots$
$\begin{array}{llcrl} & R_2 & = & 1, & 001111 \\ + & [X]_д & = & 0, & 111111 \\ \hline & R_2' & = & 10, & 001110 \end{array}$	Відновлення залишку
$\begin{array}{llcrl} & 2\cdot R_2' & = & 0, & 011100 \\ + & [-X]_д & = & 1, & 000001 \\ \hline & R_3 & = & \boxed{1}, & 011101 \end{array}$	$[Res] = 0{,}01\boxed{1}\ldots$
$\begin{array}{llcrl} & R_3 & = & 1, & 011101 \\ + & [X]_д & = & 0, & 111111 \\ \hline & R_3' & = & 10, & 011100 \end{array}$	Відновлення залишку
$\begin{array}{llcrl} & 2\cdot R_3' & = & 0, & 111000 \\ + & [-X]_д & = & 1, & 000001 \\ \hline & R_4 & = & \boxed{1}, & 111001 \end{array}$	$[Res] = 0{,}011\boxed{1}\ldots$
$\begin{array}{llcrl} & R_4 & = & 1, & 111001 \\ + & [X]_д & = & 0, & 111111 \\ \hline & R_4' & = & 10, & 111000 \end{array}$	Відновлення залишку
$\begin{array}{llcrl} & 2\cdot R_4' & = & 1, & 110000 \\ + & [-X]_д & = & 1, & 000001 \\ \hline & R_5 & = & 1\boxed{0}, & 110001 \end{array}$	$[Res] = 0{,}0111\boxed{0}\ldots$

Продовження таблиці 3.4

Розрахунки	Частка
$\begin{array}{rlrl} & 2\cdot R_5 & = & 1, & 100010 \\ + & [-X]_{д} & = & 1, & 000001 \\ \hline & R_6 & = & 1\boxed{0}, & 100011 \end{array}$	$[Res] = 0{,}01110\boxed{0}\ldots$
$\begin{array}{rlrl} & 2\cdot R_6 & = & 1, & 000110 \\ + & [-X]_{д} & = & 1, & 000001 \\ \hline & R_7 & = & 1\boxed{0}, & 000111 \end{array}$	$[Res] = 0{,}011100\boxed{0}\ldots$
$\begin{array}{rlrl} & 2\cdot R_7 & = & 0, & 001110 \\ + & [-X]_{д} & = & 1, & 000001 \\ \hline & R_8 & = & \boxed{1}, & 001111 \end{array}$	$[Res] = 0{,}0111000\boxed{1}\ldots$
$\begin{array}{rlrl} & R_8 & = & 1, & 001111 \\ + & [X]_{д} & = & 0, & 111111 \\ \hline & R'_8 & = & 0, & 001110 \end{array}$	Відновлення залишку
$\begin{array}{rlrl} & 2\cdot R'_8 & = & 0, & 011100 \\ + & [-X]_{д} & = & 1, & 000001 \\ \hline & R_9 & = & \boxed{1}, & 011101 \end{array}$	$[Res] = 0{,}01110001\boxed{1}$

4) Після підрахунку необхідної кількості розрядів частки (плюс один додатковий) виконаємо округлення результату та переведення його у додатковий код. Округлення виконується шляхом додавання одиниці у молодший додатковий розряд та відсіканням цього додаткового розряду.

$$\begin{array}{rcr|l} & & & \text{ДР} \\ [Res] & = & 0{,}01110001 & 1 \\ & + & & 1 \\ \hline [Res]_{д} & = & 0{,}01110010 & 0 \end{array}$$

5) Результату дописуємо правильний знак, що було визначено на першому кроці.

$$[Res]_{д} = 1{,}01110010.$$

Після переведення результату у прямий код та запису його в природній формі отримуємо наступне:

$$Res = \frac{W}{-X} = -0{,}10001110.$$

Ділення без відновлення залишків

1) Визначимо знак частки (результату) шляхом додавання по модулю 2 знакових розрядів діленого та дільника:

$$1 \oplus 0 = 1.$$

Одиниця означає що результат ділення від'ємний.

2) Визначимо наявність/відсутність переповнення розрядної сітки ЕОМ. Для цього алгебраїчно додамо до діленого дільник, якому приписується знак протилежний знаку діленого.

$$\begin{array}{cllll} & [W]_{д} & = & 0, & 100011 \\ + & [-X]_{д} & = & 1, & 000001 \\ \hline & R_0 & = & \boxed{1}, & 100100 \end{array}$$

Отримали від'ємне число. В нашому випадку ділене додатне, тому у псевдознаковий розряд частки заносимо 0 (знаковий розряд R_0-го з інверсією).

$$[Res] = 0, \ldots$$

Нуль у псевдознаковому розряді частки означає *відсутність переповнення розрядної сітки ЕОМ*. Можемо продовжувати розрахунки.

3) Особливістю ділення без відновлення залишків є те, що не треба відновлювати залишок якщо на i-му кроці він вийшов від'ємним. Не залежно від знаку поточного залишку його необхідно зсунути на один розряд вліво (тобто помножити на 2). А потім додати модуль дільника з приписаним до нього знаком протилежним знаку поточного залишку.
Враховуючи те, що в нашому випадку дільник від'ємний, кожний знаковий розряд залишку буде переноситись до результату без інверсії.
Розрахунки подамо у вигляді таблиці 3.5.

Таблиця 3.5 — Розрахунок числових розрядів частки

Розрахунки	Частка
$\begin{array}{lrcl} & 2\cdot R_0 & = & 1,\ 001000 \\ + & [X]_д & = & 0,\ 111111 \\ \hline & R_1 & = & 1\boxed{0},\ 000111 \end{array}$	$[Res] = 0{,}\boxed{0}\ldots$
$\begin{array}{lrcl} & 2\cdot R_1 & = & 0,\ 001110 \\ + & [-X]_д & = & 1,\ 000001 \\ \hline & R_2 & = & \boxed{1},\ 001111 \end{array}$	$[Res] = 0{,}0\boxed{1}\ldots$
$\begin{array}{lrcl} & 2\cdot R_2 & = & 0,\ 011110 \\ + & [X]_д & = & 0,\ 111111 \\ \hline & R_3 & = & \boxed{1},\ 011101 \end{array}$	$[Res] = 0{,}01\boxed{1}\ldots$
$\begin{array}{lrcl} & 2\cdot R_3 & = & 0,\ 111010 \\ + & [X]_д & = & 0,\ 111111 \\ \hline & R_4 & = & \boxed{1},\ 111001 \end{array}$	$[Res] = 0{,}011\boxed{1}\ldots$
$\begin{array}{lrcl} & 2\cdot R_4 & = & 1,\ 110010 \\ + & [X]_д & = & 0,\ 111111 \\ \hline & R_5 & = & 1\boxed{0},\ 110001 \end{array}$	$[Res] = 0{,}0111\boxed{0}\ldots$
$\begin{array}{lrcl} & 2\cdot R_5 & = & 1,\ 100010 \\ + & [-X]_д & = & 1,\ 000001 \\ \hline & R_6 & = & 1\boxed{0},\ 100011 \end{array}$	$[Res] = 0{,}01110\boxed{0}\ldots$
$\begin{array}{lrcl} & 2\cdot R_6 & = & 1,\ 000110 \\ + & [-X]_д & = & 1,\ 000001 \\ \hline & R_7 & = & 1\boxed{0},\ 000111 \end{array}$	$[Res] = 0{,}011100\boxed{0}\ldots$
$\begin{array}{lrcl} & 2\cdot R_7 & = & 0,\ 001110 \\ + & [-X]_д & = & 1,\ 000001 \\ \hline & R_8 & = & \boxed{1},\ 001111 \end{array}$	$[Res] = 0{,}0111000\boxed{1}\ldots$
$\begin{array}{lrcl} & 2\cdot R_8 & = & 0,\ 011110 \\ + & [X]_д & = & 0,\ 111111 \\ \hline & R_9 & = & \boxed{1},\ 011101 \end{array}$	$[Res] = 0{,}01110001\boxed{1}$

4) Після підрахунку необхідної кількості розрядів частки (плюс один додатковий) виконаємо округлення результату та переведення його у додатковий код.

$$\begin{array}{rcr|l}
 & & & \text{ДР} \\
[Res] & =_{+} & 0{,}01110001 & 1 \\
 & & & 1 \\
\hline
[Res]_{\text{д}} & = & 0{,}01110010 & 0
\end{array}$$

5) Результату дописуємо правильний знак, що було визначено на першому кроці.

$$[Res]_{\text{д}} = 1{,}01110010.$$

Після переведення результату у прямий код та запису його в природній формі отримуємо наступне:

$$Res = \frac{W}{-X} = -0{,}10001110.$$

Даний результат співпадає з результатом, отриманим іншим способом. Значить розрахунки виконано вірно.

Рекомендована література

1. Корнейчук В.И., Тарасенко В.П. Основы компьютерной арифметики. – К.: «Корнейчук», 2002. – 176 с.
2. Матвієнко М.П. Комп'ютерна логіка. Навчальний посібник. – К.: Видавництво Ліра-К, 2012. – 288 с.
3. Основы компьютерной арифметики и логики: Учеб. пособие. / В. И. Потапов, О.П. Шафеева, И.В. Червенчук. – Омск: Изд-во ОмГТУ, 2004. – 172 с.
4. Самофалов К.Г., Романкевич А.М. и др. Прикладная теория цифровых автоматов. – К.: В.Шк., 1987. – 375 с.
5. Дж. Ф. Уэйкерли. Проектирование цифровых устройств. – М.: Постмаркет, 2002. – 528 с.
6. Шауман А. М. Основи машинной арифметики.— Л.: Изд-во Ленингр. ун-та, 1979. 312 с.
7. Behrooz Parhami. Computer Arithmetic: Algorithms and Hardware Designs, Second Edition (Oxford Series in Electrical and Computer Engineering) / Oxford University Press, USA, 2009