

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

КОМП'ЮТЕРНА ЛОГІКА. ЧАСТИНА 2

Практикум

Видання третє перероблене і доповнене

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою «Комп'ютерні системи та мережі»
спеціальності 123 Комп'ютерна інженерія

Укладачі: В. І. Жабін, В.В. Жабіна, О. А. Верба

Електронне мережне навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2025

УДК 004.31 (075.8)
К

Укладачі: *Жабін Валерій Іванович, д-р техн. наук, проф.*
Жабіна Валентина Валеріївна, канд. техн. наук, доц.
Верба Олександр Андрійович, канд. техн. наук

Рецензент: *Дичка І. А., д-р техн. наук, проф., декан факультету прикладної математики, науковий керівник кафедри програмного забезпечення комп'ютерних систем КПІ ім. Ігоря Сікорського*

Відповідальний редактор *Кулаков Ю.О., д-р. техн. наук, проф.*

*Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 5 від 6.03.2025 р.)
за поданням Вченої ради факультету інформатики та обчислювальної техніки
(протокол № 8. від 24.02.2025 р.)*

Комп'ютерна логіка. Частина 2. Практикум. Видання третє перероблене і доповнене. [Електронний ресурс]: навч. посібн. для здобувачів ступеня бакалавра за освітньою програмою «Комп'ютерні системи та мережі» спеціальності 123 «Комп'ютерна інженерія» / Укладачі: В. І. Жабін, В. В. Жабіна, О. А. Верба; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 1,53 Мбайт). – Київ: КПІ ім. Ігоря Сікорського, 2025.
– 103 с.

Навчальний посібник призначено для вивчення арифметичних основ побудови вузлів і блоків цифрових систем, способів представлення чисел в комп'ютерах, виконання арифметичних операцій з фіксованою і плаваючою комою, способів машинного переводу чисел між різними системами числення, а також виконання домашніх завдань з курсу для здобувачів ступеня бакалавра за освітньою програмою «Комп'ютерні системи та мережі» спеціальності 123 «Комп'ютерна інженерія» та для інших освітніх програм факультетів: інформатики та обчислювальної техніки, прикладної математики тощо.

УДК 004.31 (075.8)

Реєстр. № НП ХХ/ХХ-ХХХ. Обсяг 5,0 авт. арк.
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056 <https://kpi.ua>
Свідцтво про внесення до Державного реєстру видавців, виготовлювачів
і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

© КПІ ім. Ігоря Сікорського, 2023

ЗМІСТ

ВСТУП	4
1. ЗАНЯТТЯ №1. ДОСЛІДЖЕННЯ МЕТОДІВ ПОДАННЯ ДАНИХ ТА ВИКОНАННЯ ОДНОТАКТНИХ ОПЕРАЦІЙ В КОМП'ЮТЕРАХ.....	5
2. ЗАНЯТТЯ №2. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЇВ ДЛЯ МНОЖЕННЯ ЧИСЕЛ.....	27
3. ЗАНЯТТЯ №3. ПОБУДОВА ФУНКЦІОНАЛЬНИХ СХЕМ ВИКОНАННЯ АРИФМЕТИЧНИХ ОПЕРАЦІЙ.....	41
4. ЗАНЯТТЯ №4. ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЇВ ДЛЯ ДІЛЕННЯ ЧИСЕЛ.....	46
5. ЗАНЯТТЯ №5. ДОСЛІДЖЕННЯ ОПЕРАЦІЙ З ЧИСЛАМИ У ФОРМАТІ З ПЛАВАЮЧОЮ КОМОЮ	55
6. ЗАНЯТТЯ №6. ДОСЛІДЖЕННЯ ОПЕРАЦІЙ ДОДАВАННЯ ТА ВІДНІМАННЯ В ДВІЙКОВО-КОДОВАНИХ СИСТЕМАХ ЧИСЛЕННЯ.....	68
7. ЗАНЯТТЯ №7. ДОСЛІДЖЕННЯ ОПЕРАЦІЙ ПЕРЕВОДУ ЧИСЕЛ МІЖ СИСТЕМАМИ ЧИСЛЕННЯ З РІЗНОЮ ОСНОВОЮ.....	77
8. ЗАНЯТТЯ №8. ОБЧИСЛЕННЯ ФУНКЦІЙ.....	86
ДОДАТОК А. ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ МОДЕЛЮВАННЯ ЛОГІЧНИХ СХЕМ	98

ВСТУП

Дисципліна „Комп'ютерна логіка” відноситься до циклу професійної та практичної підготовки студентів спеціальності 123 Комп'ютерна інженерія. Освітній компонент «Комп'ютерна логіка. Частина 2» присвячений вивченню принципів виконання арифметичних і логічних операцій над машинними словами в комп'ютерах. Видання третє доповнене теоретичними відомостями і практичними завданнями для вивчення принципів представлення даних і виконання арифметичних операцій у форматі з плаваючою комою і орієнтоване на виконання практичних робіт. Виконання практичних робіт дозволяє розширити і закріпити теоретичні знання за курсом. Кожній практичній роботі повинна передувати самостійна підготовка, в процесі якої докладно вивчається опис практичної роботи, відповідний розділ конспекту лекцій і літературні джерела. В процесі підготовки складається звіт про практичну роботу, в якому повинні бути відображені виконані пункти теоретичного завдання, а також заготовлені необхідні для експериментальної частини практичної роботи таблиці, діаграми і т. ін.

Перед початком практичної роботи результати підготовки перевіряються викладачем. При цьому студент повинний сформулювати мету і порядок виконання практичної роботи, представити заготовлений звіт і відповіді на контрольні питання.

По результатам роботи студент представляє оформлений звіт, який повинний містити умови і результати підготовки до роботи, визначення варіанту і виконання завдання, усі схеми, формули, таблиці, діаграми, графіки, отримані при виконанні завдання та в процесі експериментального дослідження схем, відповіді на контрольні запитання а також висновки по роботі. Залік по практичній роботі студент одержує після співбесіди по тематиці виконаної роботи.

1. ЗАНЯТТЯ №1

ДОСЛІДЖЕННЯ МЕТОДІВ ПОДАННЯ ДАНИХ ТА ВИКОНАННЯ ОДНОТАКТНИХ ОПЕРАЦІЙ В КОМП'ЮТЕРАХ

Мета роботи: вивчення методів та засобів подання чисел в комп'ютерах з використанням машинних кодів, одержати навички побудови та опису операційних схем для виконання однотоктних операцій, оволодіти програмним комплексом моделювання та дослідження цифрових пристроїв.

Теоретичні відомості

Кодування чисел в комп'ютерах

Найбільше поширення в обчислювальній техніці (ОТ) має двійкова однорідна позиційна система числення з цифрами $\{0, 1\}$ та природним порядком ваги розрядів. Для подання чисел та виконання операцій з числами, що мають знаки, використовують спеціальні машинні коди:

- прямий код (ПК),
- обернений код (ОК),
- доповняльний код (ДК).

Будемо вважати, що розрядна сітка містить число, яке має n -розрядну цілу частину, k -розрядну дробову частину і знаковий розряд зліва від основних розрядів (рис. 2.1).

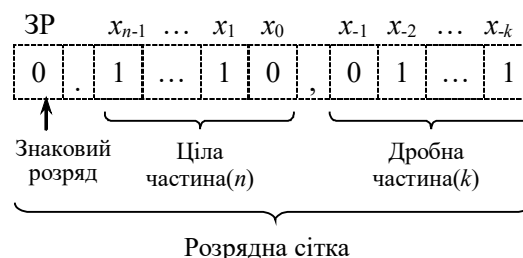


Рис. 1.1. Приклад запису числа зі знаком, що має цілу та дробову частину

Старший розряд цілої частини має вагу 2^{n-1} , а молодший розряд дробової частини – вагу 2^{-k} .

Подання числа X у *прямому кодi* визначається виразом:

$$[X]_{\text{ПК}} = \begin{cases} X, & \text{якщо } X \geq 0; \\ 2^n + |X|, & \text{якщо } X \leq 0. \end{cases}$$

Під час утворення прямого коду знаковий розряд дорівнює 0, якщо число додатне і 1, якщо число від'ємне.

Під час запису числа знаковий розряд відокремлюється від основних розрядів крапкою, а ціла частина числа від дробової – комою. У випадку, коли числа не мають цілої частини, то знаковий розряд відокремлюється від основних розрядів комою.

Приклад 1.1. Записати числа $A=10,101011$; $B=-10,0111010$ у ПК.

Виконання завдання:

$$[A]_{\text{ПК}} = 0.10, 101011; [B]_{\text{ПК}} = 1.10, 0111010$$

Приклад 1.2. Записати числа $A=0,101011$; $B=-0,0111010$ у ПК.

Виконання завдання:

$$[A]_{\text{ПК}} = 0, 101011; [B]_{\text{ПК}} = 1, 0111010$$

ПК застосовується для зберігання чисел в пам'яті комп'ютера та виконання деяких операцій (наприклад, множення, ділення та ін.). Для операцій додавання і віднімання ПК не використовується.

Під час перетворення від'ємного числа в *обернений код*, у знаковий розряд записується 1, а значення основних розрядів інвертуються, тобто, у кожному розряді 0 замінюється на 1, а 1 замінюється на 0. Додатне число у ОК збігається із числом у ПК, тобто основні розряди не інвертуються, у знаковий розряд записується 0.

Формула перетворення чисел у ОК код має вигляд:

$$[A]_{\text{OK}} = \begin{cases} A, \text{ якщо } A \geq 0; \\ 2^{n+1} - 2^{-k} - |A| = 2^{n+1} - 2^{-k} + A, \text{ якщо } A \leq 0. \end{cases}$$

Приклад 1.3. Записати числа $A=10,1011$; $B=-01,11010$ в ОК.

Виконання завдання:

$$[A]_{\text{OK}} = 0.10,1011; [B]_{\text{OK}} = 1.10,00101.$$

Приклад 1.4. Записати числа $A=101011$; $B=-0111010$ в ОК.

Виконання завдання:

$$[A]_{\text{OK}} = 0.101011; [B]_{\text{OK}} = 1.1000101.$$

Приклад 1.5. Записати числа $A=0,101011$; $B=-0,0111010$ в ОК.

Виконання завдання:

$$[A]_{\text{OK}} = 0,101011; [B]_{\text{OK}} = 1,1000101.$$

Під час перетворення від'ємного числа в *доповняльний код*, у знаковий розряд записується 1, а значення основних розрядів інвертуються, після чого до молодшого розряду додається 1 (з поширенням переносів між розрядами). Додатне число в ДК збігається з числами у ПК і ОК, тобто в знаковий розряд записується 0, а основні розряди не змінюються.

Формула перетворення чисел у доповнювальний код має вигляд:

$$[A]_{\text{ДК}} = \begin{cases} A, \text{ якщо } A \geq 0; \\ 2^{n+1} - |A| = 2^{n+1} + A, \text{ якщо } A < 0. \end{cases}$$

Приклад 1.6. Записати числа $A = -010,010$; $B = 010,010$ у ПК, ОК і ДК.

Виконання завдання:

$$\begin{array}{l} [A]_{\text{ПК}} = 1.010,010; \\ [A]_{\text{ОК}} = 1.101,101; \\ \quad + \quad \underline{\quad 1} \\ [A]_{\text{ДК}} = 1.101,110. \end{array} \qquad \begin{array}{l} [B]_{\text{ПК}} = 0.010,010; \\ [B]_{\text{ОК}} = 0.010,010; \\ [B]_{\text{ДК}} = 0.010,010. \end{array}$$

Приклад 1.7. Записати числа $A = -0,010010$; $B = 0,010010$ у ПК, ОК і ДК.

Виконання завдання:

$$\begin{array}{l} [A]_{\text{ПК}} = 1, 010010; \\ [A]_{\text{ОК}} = 1, 101101; \\ \quad + \quad \underline{\quad 1} \\ [A]_{\text{ДК}} = 1, 101110. \end{array} \qquad \begin{array}{l} [B]_{\text{ПК}} = 0, 010010; \\ [B]_{\text{ОК}} = 0, 010010; \\ [B]_{\text{ДК}} = 0, 010010. \end{array}$$

Додавання чисел із знаками у машинних кодах

Операції алгебраїчного підсумовування і віднімання неможливо виконувати в прямому коді із використанням звичайного суматора, оскільки знакові розряди і основні розряди повинні оброблятися по-різному. З використанням ОК та ДК операції додавання і віднімання можна виконувати за допомогою звичайних багаторозрядних суматорів, на яких оброблюються як основні, так і знакові розряди. Операція віднімання замінюється операцією додавання з числом, що має протилежний знак. Наприклад, операція $S = A - B$ виконується як $S = A + (-B)$.

Під час додавання чисел із однаковими знаками може виникнути переповнення розрядної сітки, що приводить до втрати знака числа.

Для виявлення переповнення використовують *модифіковані машинні коди*, в яких вводиться додатковий (другий) знаковий розряд ZP_2 ліворуч основного (першого) розряду ZP_1 . У разі переповнення сітки старший

знаковий розряд завжди зберігає знак результату. Ознака переповнення визначається функцією $OVR=3P_2 \oplus 3P_1$.

Формули подання модифікованих кодів мають вигляд:

$$[A]_{OK} = \begin{cases} A, \text{ якщо } A \geq 0; \\ 2^{n+2} - 2^{-k} + A, \text{ якщо } A < 0. \end{cases}$$

$$[A]_{DK} = \begin{cases} A, \text{ якщо } A \geq 0; \\ 2^{n+2} + A, \text{ якщо } A < 0. \end{cases}$$

В модифікованих кодах знакові розряди і основні розряди обробляються як єдине ціле. Правильний знак суми утворюється автоматично в процесі підсумовування цифр знакових і основних розрядів з урахуванням переносів.

Характерною рисою ОК є циклічний перенос зі старшого знакового розряду в молодший розряд суми. Перенос виникає автоматично, коли корекція результату потрібна.

Приклад 1.9. Задано $A=0,10101$; $B=-0,01001$. Знайти суму C з використанням ОК.

Виконання завдання:

$$\begin{array}{r} [A]_{OK} = 0\ 0,10101 \\ + [B]_{OK} = 1\ 1,10110 \\ \hline 0\ 0,01011 \\ + \quad \quad \quad 1 \text{ (перенос)} \\ \hline [C]_{OK} = 0\ 0,01100 \end{array}$$

Приклад 1.10. Задано $A = 0.01001$; $B = -0.10101$. Знайти суму C з використанням ОК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ОК}} = 00,01001 \\ + [B]_{\text{ОК}} = 11,01010 \\ \hline [C]_{\text{ОК}} = 11,10011 \\ \text{(Перенос відсутній)} \end{array}$$

Приклад 1.11. Задано обернені коди чисел $A = -0.10101$ і $B = -0.01001$. Знайти обернений код суми C .

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ОК}} = 11,01010 \\ + [B]_{\text{ОК}} = 11,10110 \\ \hline 11,00000 \\ + \quad \quad \quad 1 \text{ (перенос)} \\ \hline [C]_{\text{ОК}} = 11,00001 \end{array}$$

Приклад 1.12. Задано $A = 0.10101$ і $B = 0.01110$. Знайти суму C з використанням ОК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ОК}} = 00,10101 \\ + [B]_{\text{ОК}} = 00,01110 \\ \hline [C]_{\text{ОК}} = 01,00011 \text{ – додатне переповнення} \end{array}$$

Приклад 1.13. Задано $A = -0.10101$; $B = -0.01110$. Знайти суму C .

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ОК}} = 11,01010 \\ + [B]_{\text{ОК}} = 11,10001 \\ \hline 10,11011 \\ + \quad \quad \quad 1 \text{ (перенос)} \\ \hline [C]_{\text{ОК}} = 10,01100 \text{ – від'ємне переповнення} \end{array}$$

Функціональна схема пристрою, що реалізує операцію додавання і віднімання в ОК наведена на рис. 1.2.

Операнди надходять на вхід суматора SM з регістрів RGx і RGy . Операція віднімання виконується шляхом додавання зменшеного до від'ємника, який інвертується за допомогою елемента ВИКЛЮЧНЕ АБО. Для цього на вхід COM подається одиничний сигнал. Під час додавання на цей вхід потрібно подати сигнал 0. Суматор формує основні розряди суми (OP) і два знакових розряди $ЗР_2$ і $ЗР_1$.

Корекція суми здійснюється за допомогою ланцюга циклічного переносу зі знакового розряду в молодший розряд суматора ($CO \rightarrow CI$). Цей ланцюг завжди замкнений. Перенос автоматично виникає тільки тоді, коли потрібно реалізувати корекцію суми. При значенні $OVR = 0$, переповнення розрядної сітки відсутнє, у випадку $OVR = 1$ наявне переповнення розрядної сітки.

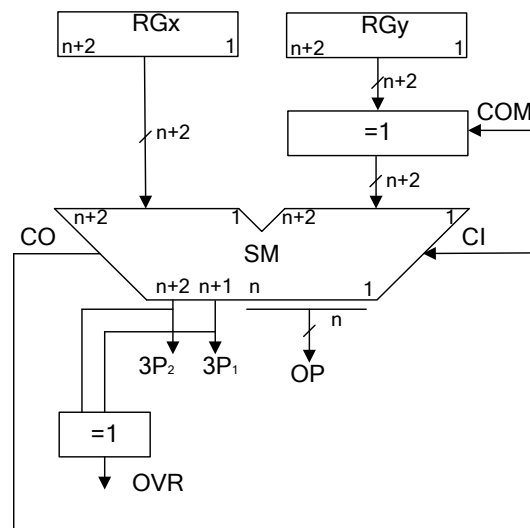


Рис. 1.2. Схема виконання операцій додавання і віднімання в обернених кодах

Підсумовування операндів у ДК автоматично формує суму в ДК з урахуванням знаку. За застосування модифікованого коду корекції роботи не треба при будь-якому сполученні знаків доданків. Факт переповнення

розрядної сітки можна визначити за розбіжністю значень знакових розрядів. Старший знаковий розряд завжди зберігає знак результату.

Приклад 1.14. Задано $A = 0,10101$; $B = 0,01001$. Знайти суму C в ДК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 00,10101 \\ + [B]_{\text{ДК}} = 00,01001 \\ \hline [C]_{\text{ДК}} = 00,11110 \end{array}$$

Приклад 1.15. Задано $A = 0,10101$; $B = -0,01001$. Знайти суму C в ДК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 00,10101 \\ + [B]_{\text{ДК}} = 11,10111 \\ \hline [C]_{\text{ДК}} = 00,01100 \end{array}$$

Приклад 1.16. Для $A = 0,01001$ і $B = -0,10101$ знайти суму C в ДК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 00,01001 \\ + [B]_{\text{ДК}} = 11,01011 \\ \hline [C]_{\text{ДК}} = 11,10100 \end{array}$$

Приклад 1.17. Задано $A = -0,10101$; $B = -0,01001$. Знайти C в ДК.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 11,01011 \\ + [B]_{\text{ДК}} = 11,10111 \\ \hline [C]_{\text{ДК}} = 11,00010 \end{array}$$

Приклад 1.18. Задано $A = 0,10101$; $B = 0,01110$. Знайти суму C в ДК.

Визначити знак результату.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 00,10101 \\ + [B]_{\text{ДК}} = 00,01110 \\ \hline [C]_{\text{ДК}} = 01,00011 \text{ – додатне переповнення} \end{array}$$

У цьому випадку наявне додатне переповнення розрядної сітки, та за застосування модифікованого коду старший розряд зберігає знак результату. Отже отримане в результаті обчислень число є додатним.

Приклад 1.19. Задано $A = -0.10101$ і $B = -0.01110$. Знайти суму C в ДК. Визначити знак результату.

Виконання завдання:

$$\begin{array}{r} [A]_{\text{ДК}} = 11,01011 \\ + [B]_{\text{ДК}} = 11,10010 \\ \hline [C]_{\text{ДК}} = 10,11101 \text{ – від'ємне переповнення} \end{array}$$

У цьому випадку наявне від'ємне переповнення розрядної сітки, та за застосування модифікованого коду старший розряд зберігає знак результату. Отже отримане в результаті обчислень число є від'ємним.

Схема, що реалізує операцію додавання і віднімання в ДК, аналогічна схемі для зворотних кодів (рис. 1.2), але в цьому випадку відсутній циклічний ланцюг корекції результату ($CO \rightarrow CI$). За віднімання разом з одиничним сигналом на вхід COM подається одиниця на вхідний перенос CI суматора. Це необхідно для зміни знака від'ємника, яке здійснюється інвертуванням усіх розрядів від'ємника і додаванням одиниці до його молодшого розряду. Під час додавання на входи COM і CI потрібно подати значення 0.

Зсуви машинних кодів.

Існують два різновиди машинних зсувів:

- логічний зсув;
- арифметичний зсув.

Логічний зсув це зміщення розрядів машинного слова у просторі із втратою розрядів, що виходять за межі розрядної сітки. Розряди, що звільняються заповнюються нулями. Схема логічного зсуву чисел зображена на рис. 1.3.

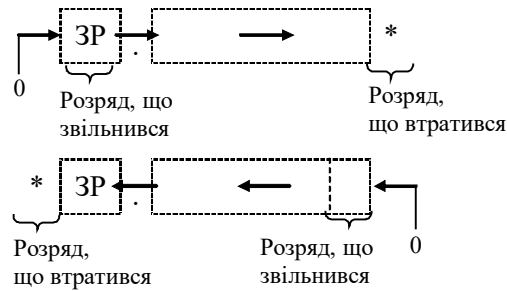


Рис. 1.3. Операційна схема логічного зсуву двійкових чисел

Приклад 1.20. Виконати логічний зсув двійкового числа вліво і вправо на один розряд.

$$A_{(2)} = 0.0101,1101.$$

Виконання завдання:

$$\begin{array}{l} A \\ A \rightarrow \end{array} \left| \begin{array}{l} 1.0101,1101 \\ 0.1010,1110 \end{array} \right| *$$

$$\begin{array}{l} A \\ \leftarrow A \end{array} \left| \begin{array}{l} 1.0101,1101 \\ *0.1011\ 1010 \end{array} \right|$$

Арифметичний зсув виконується з урахуванням знакового розряду. Правила зсуву чисел поданих у ПК, ОК та ДК відрізняються. Арифметичний зсув ліворуч означає множення числа на 2 (тобто на основу системи числення), а зсув праворуч – ділення числа на 2.

Арифметичний зсув чисел поданих у ПК

При цьому типі зсуву знаковий розряд не зсувається. Основні розряди зсуваються ліворуч або праворуч із заповненням нулями розрядів, що звільнилися при зсуві. Розряди, що вийшли за межі розрядної сітки втрачаються. За арифметичного зсуву вліво можлива втрата значимості числа. За зсуву праворуч виникає похибка. Схема арифметичного зсуву чисел поданих у ПК зображена на рис. 1.4



Рис. 1.4. Операційна схема арифметичного зсуву чисел у ПК

Приклад 1.21. Виконати арифметичний зсув вліво і вправо на один розряд двійкових чисел, поданих у ПК.

$$A_{(2)} = 1.10101; B_{(2)} = 0.11011.$$

Виконання завдання:

A	$1.$	1	0	101	
$A \rightarrow$	$1.$	0	1	010	* Похибка
$\leftarrow A$	$1.$	* 0	1	010	Втрата значимості

B	$0.$	1	1	011	
$B \rightarrow$	$0.$	0	1	101	* Похибка
$\leftarrow B$	$0.$	* 1	0	110	Втрата значимості

Арифметичний зсув ліворуч чисел, поданих у ОК.

Для визначення переповнення розрядної сітки при арифметичному зсуві використовують модифіковані коди.

Правило. Якщо під час зсуву від'ємного числа ліворуч за розрядну сітку виходить одиниця із знакового розряду, то необхідно виконати корекцію $K = +2^{-k}$.

На рис. 1.5 зображена операційна схема реалізації арифметичного зсуву ліворуч від'ємних чисел поданих у ОК.

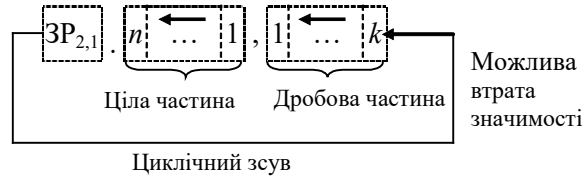
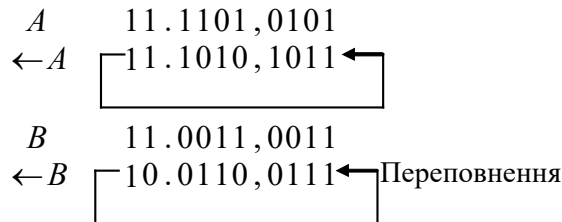


Рис. 1.5. Операційна схема арифметичного зсуву ліворуч від’ємних чисел у ОК

Приклад 1.22. Виконати арифметичний зсув ліворуч на один розряд двійкових чисел, поданих у ОК.

$$A_{(2)} = 11.1101,1010; B_{(2)} = 11.0011,0011.$$

Виконання завдання:



Арифметичний зсув праворуч чисел, поданих у ОК.

Правило. За зсуву праворуч від’ємного числа знаковий розряд переходить у поле основних розрядів і знов заповнюється тим самим значенням.

На рис. 1.6 зображена операційна схема реалізації арифметичного зсуву праворуч від’ємних чисел поданих у ОК.

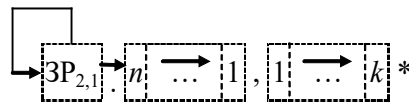


Рис. 1.6. Операційна схема арифметичного зсуву праворуч від’ємних чисел у ОК

Приклад 1.23. Виконати арифметичний зсув праворуч на один розряд двійкових чисел, поданих у ОК.

$$A_{(2)} = 00.1011,1001; B_{(2)} = 11.1010,0011.$$

Виконання завдання:

$$\begin{array}{l} A \quad \left[\begin{array}{l} \text{---} 00.1011,1001 \\ \rightarrow A \quad \left[\begin{array}{l} \text{---} 00.0101,1100^* \end{array} \right] \end{array} \right. \end{array}$$

$$\begin{array}{l} B \quad \left[\begin{array}{l} \text{---} 11.1010,0011 \\ \rightarrow B \quad \left[\begin{array}{l} \text{---} 11.1101,0001^* \end{array} \right] \end{array} \right. \end{array}$$

Арифметичний зсув ліворуч чисел, поданих у ДК

Правило. За зсуву ліворуч числа поданого у ДК розряди, що звільнились заповнюються нулями. Якщо знаковий розряд змінює значущість виникає втрата значимості числа.

На рис. 1.7 зображена операційна схема реалізації арифметичного зсуву ліворуч чисел поданих у ДК.



Рис. 1.7. Операційна схема арифметичного зсуву ліворуч від'ємних чисел у ДК

Арифметичний зсув праворуч чисел, поданих у ДК

Правило. За зсуву праворуч числа поданого у ДК знаковий розряд розповсюджується у поле основних розрядів і знов заповнюється тим самим значенням. В результаті може виникнути похибка.

Операційна схема реалізації арифметичного зсуву праворуч чисел поданих у ДК зображена на рис. 1.8.

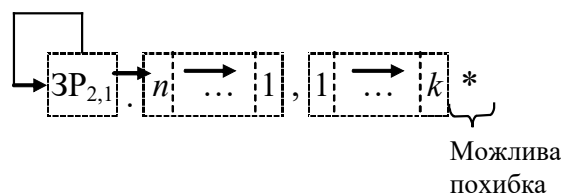


Рис. 1.8. Операційна схема арифметичного зсуву праворуч

від'ємних чисел у ДК

Приклад 1.24. Виконати арифметичний зсув праворуч і ліворуч на один розряд двійкових чисел, поданих у ДК.

$$A_{(2)} = 0.1011, 0111; B_{(2)} = 1.1011, 1001.$$

Виконання завдання:

$$\begin{array}{l} A \quad \left[\begin{array}{l} 00.1011, 0111 \\ 01.0110, 1110 \\ 00.0101, 1011^* \end{array} \right. \\ \leftarrow A \\ A \rightarrow \end{array}$$

$$\begin{array}{l} B \quad \left[\begin{array}{l} 11.1011, 1001 \\ 11.0111, 0010 \\ 11.1101, 1100^* \end{array} \right. \\ \leftarrow B \\ \rightarrow B \end{array}$$

Операційні схеми та мікроалгоритми.

Перетворення інформації в арифметико-логічних пристроях комп'ютерів провадиться шляхом послідовного виконання мікрооперацій над машинними словами (кодами чисел, символами та іншими об'єктами).

Під *мікроопераціями* (МО) розуміють елементарну дію, в результаті виконання якої можуть змінюватися значення машинних слів.

Машинне слово можна задати перерахуванням розрядів чи за допомогою ідентифікаторів. Наприклад, 01011001 – машинне слово, яке завдано переліком всіх 8-ми розрядів. Ідентифікатори можуть подаватися рядком символів (букв та цифр), починаючи з букви. Доцільно у якості ідентифікаторів використовувати позначення вузлів, на яких виконуються мікрооперації, наприклад: *RG1*, *CT*. Для визначення довжини слів та впорядкування номерів розрядів використовують додаткові дані у дужках, наприклад *RG1[0...31]*, *CT[7...0]*.

Алгебраїчні перетворення даних у цифрових пристроях виконуються за допомогою таких МО як *пересилання, підсумовування, зсув, підрахування (декремент, інкремент), інвертування.*

Для позначення мікрооперації будемо використовувати оператор присвоювання ($:=$). Інформаційний зв'язок між вузлами цифрових пристроїв пояснюється операційною схемою, на який визначається напрямок передачі даних. Управляючі сигнали не показують, бо це визначається особливістю елементної бази. Такі сигнали показують на функціональних і принципових схемах.

Приклади операційних схем для виконання МО пересилання $RG1:=DATA$, $RG1:=RG2$, $RG1[31...24]:=RG2[7...0]$ відповідно показані на рис. 1.9.

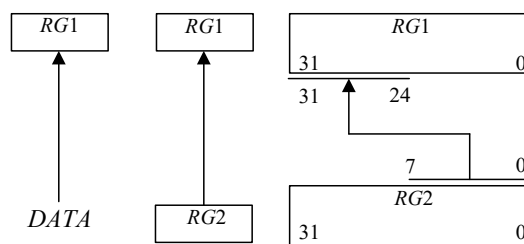


Рис. 1.9. Операційні схеми для виконання мікрооперацій пересилання даних

Для підсумовування слів в схемі використовують суматор. Приклади операційних схем, що відповідають мікроопераціям $RG1:=RG1+RG2$ і $RG1:=RG2+RG3+CI$ (де *CI* – перенос у молодший розряд суматора) відповідно показані на рис. 1.10. В схемах рис. 1.10, *а* і 1.10, *б* використовуються комбінаційні суматори, а в схемі рис. 1.12, *в* – накопичуючий суматор, який є композицією суматора і регістра.

Мікрооперації інкременту ($CT:=CT+1$) та декременту ($CT:=CT-1$) виконуються на лічильнику.

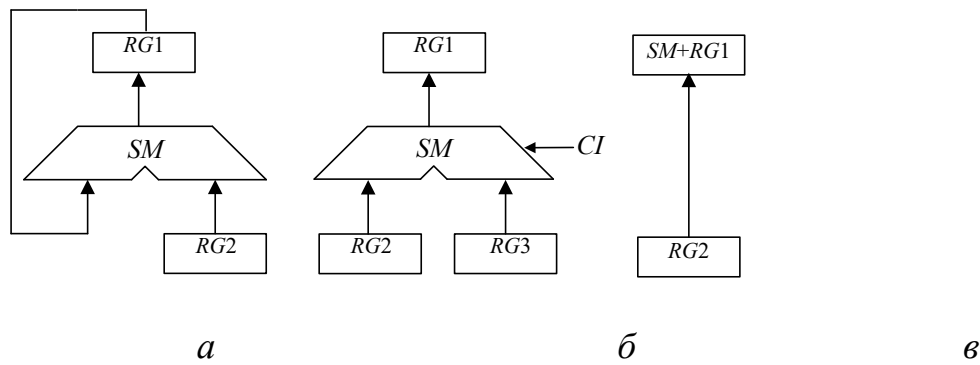


Рис. 1.10. Операційні схеми для виконання мікрооперацій

підсумовування чисел: *a*, *б* – із застосуванням комбінаційних суматорів;
в – з використанням накопичуючого суматора.

Мікрооперації зсуву слів можуть бути виконані на регістрі зсуву праворуч або ліворуч. Для означення напрямку зсуву використовують оператори зсуву *r* (*right*) та *l* (*left*). За допомогою складеного слова визначають значення розряду, який заповнюється внаслідок зсуву. Приклад операційної схеми, що відповідає мікрооперації зсуву $RG1 := 0.r(RG1)$ зображений на рис 1.11 *a*. Для реалізації зсуву, що відповідає операційній схемі на рис. 1.11, *б* необхідно виконати дві МО в одному такті $RG2 := l(RG2).0$ та $RG1 := l(RG1).RG2[7]$.

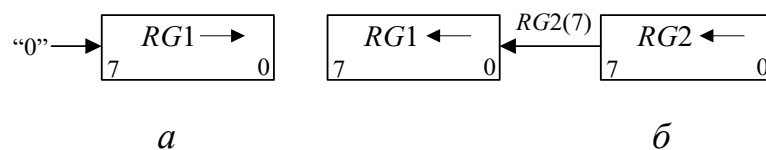


Рис. 1.11. Операційні схеми для виконання мікрооперацій зсуву

Інвертування розрядів двійкових чисел можна забезпечити інвертуванням розрядів регістру, що зберігає це число, або використанням лінійки логічних елементів при пересиланні числа, наприклад, елементів ВИКЛЮЧНЕ АБО (рис. 1.12).

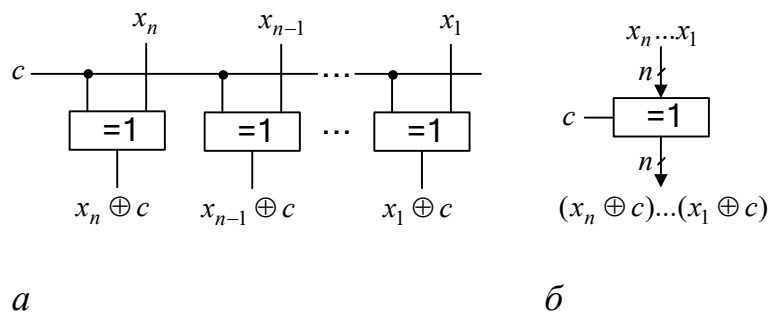
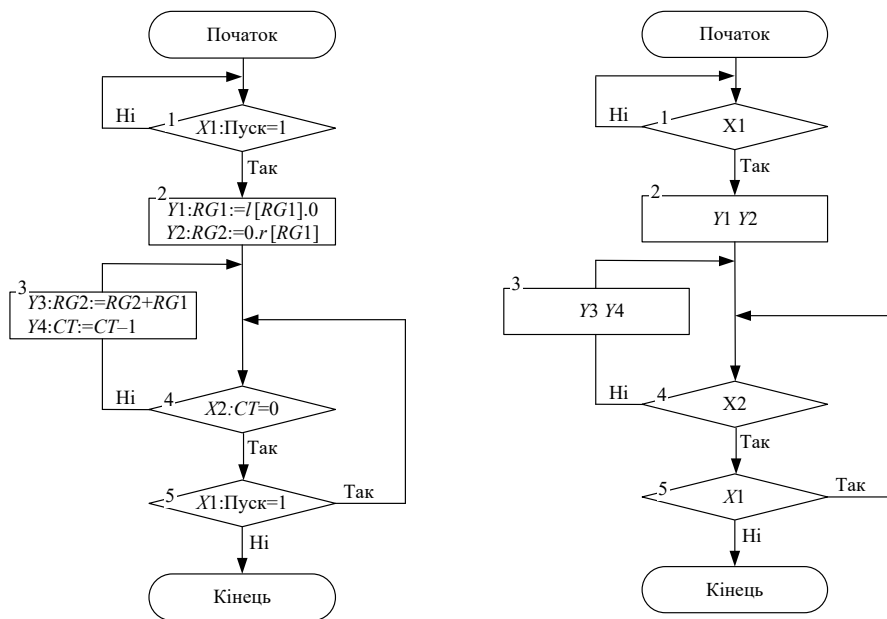


Рис. 1.12. Інвертування розрядів x_i двійкового числа: *a* – логічна схема;
б – умовне позначення (*c* – сигнал управління інвертуванням)

Послідовність мікрооперацій, що забезпечує задане перетворення інформації, називається *мікроалгоритмом* (МА).

Для опису МА використовують *схеми мікроалгоритмів*, які можуть бути описані мовами ГСА (графічна схема алгоритмів) і ЛСА (логічна схема алгоритмів).

Мікроалгоритми можуть складатися як у змістовній, так і в закодованій формі. Для складання змістовного мікроалгоритму мікрооперації записують у змістовній формі, наприклад, через оператори присвоєння або їх ідентифікаторів (рис. 1.13). Для розробки такого мікроалгоритму достатньо скласти операційну схему пристрою, який виконує задані МО.



а

б

Рис. 1.13. Приклад запису мікроалгоритму: *а* – з розширеним змістовним описом мікрооперацій; *б* – із скороченим описом мікрооперацій у вигляді ідентифікаторів

Для складання закодованого мікроалгоритму необхідна більш докладніша схема (наприклад, функціональна), яка пояснює спосіб управління кожною мікрооперацією, включаючи означення управляючих сигналів. Змістовні МО у закодованому мікроалгоритмі замінюються на сукупність управляючих сигналів, які забезпечують виконання мікрооперацій.

ПРАКТИЧНА РОБОТА №1

Підготовка до практичної роботи

1. Вивчити теоретичні відомості, інформацію в Додатку А та, при необхідності, відповідні розділи в літературі, що подається в списку.

2. Визначити свій варіант завдання. Для цього перевести десятковий номер варіанту (номер групи і номер студента у списку) в двійкову систему числення і виділити молодші розряди $a_7, a_6, a_5, a_4, a_3, a_2, a_1$ двійкового числа.

3. Визначити два двійкових числа: $F = 1a_7a_6a_5a_41$ і $G = 1011a_3a_2a_11$. Записати F і G через кому, надати числу знак «-». Одержане двійкове від'ємне число має вигляд $X = -F, G = -1a_7a_6a_5a_41, 1011a_3a_2a_11$.

4. Одержане двійкове число X з природною фіксованою комою записати у 15-розрядну сітку в машинних кодах: прямому, доповняльному і оберненому.

5. Подати модифіковані коди (доповняльний і обернений) у 16-розрядній сітці.

6. Виконати арифметичний зсув одержаних модифікованих кодів числа X на один розряд ліворуч і на один розряд праворуч. Перевірити переповнення розрядної сітки.

7. Одержати доповняльний та обернений коди числа $Y = X + 101a_4a_3, 1a_2a_110$.

8. Виконати підсумування $Z = X + Y$ в доповняльних і обернених кодах.

9. Виконати підсумування $N = X + (-Y)$ в доповняльних і обернених кодах.

10. Розробити функціональні схеми перетворення 15-розрядного числа, поданого в ПК, в 16-розрядні модифіковані ОК і ДК.

11. У відповідності з операційною схемою (рис.1.14) розробити функціональну схему, що виконує мікрооперації:

- перетворює 15-розрядні операнди E і H , подані ПК, в модифіковані 16-розрядні коди згідно з варіантом (ДК при $a_1=0$) і (ОК при $a_1=1$);

- виконує мікрооперації додавання і віднімання модифікованих ДК або ОК (за варіантом);

- записує результат додавання (віднімання) в регістр зсуву;

- виконує арифметичний зсув модифікованих кодів на один розряд ліворуч і праворуч.

12. Описати мікроалгоритми, що виконуються пристроєм, за допомогою ГСА в змістовних мікроопераціях (як на рис. 1.13). Одержати закодований мікроалгоритм, в якому змістовні мікрооперації замінені на управляючі сигнали, що забезпечують їх виконання, наприклад: W – запис коду в регістр, SL – зсув коду в регістрі ліворуч, SR – зсув коду в регістрі праворуч.



Рис. 1.14. Операційна схема пристрою

Порядок виконання роботи

1. За допомогою моделюючого комплексу ПРОГМОЛС-2 вивчити спосіб управління всіма логічними елементами і вузлами (регістрами, суматорами), що необхідні для побудови розроблених функціональних схем.
2. Побудувати і відлагодити всі розроблені функціональні схеми.
3. Промодельовати роботу пристроїв для різних наборів операндів.
4. Зробити висновки по роботі.

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, порядок виконання роботи, всі схеми, формули і таблиці, отримані при виконанні теоретичного завдання і в процесі моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Контрольні запитання

1. Яким чином представляються числа із знаками в комп'ютерах?
2. Які машинні коди використовують для виконання операцій додавання і віднімання?
3. Поясніть правила подання чисел із знаками в різних машинних колах.
4. Поясніть правила зсуву чисел в ПК, ОК і ДК.
5. Поясніть правила додавання та віднімання чисел в ОК і ДК.
6. Як можна виявити переповнення розрядної сітки при виконанні операцій з машинними кодами?

7. Яким чином можна подати мікрооперації і мікроалгоритми?

Список літератури

1. Жабін В.І., Арифметичні та управляючі пристрої цифрових ЕОМ: Навч. Посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, С.Г. Стіренко. – К.: ВЕК+, 2008. – 176 с.

2. Жабін В.І. Прикладна теорія теорія цифрових автоматів: Навч. Посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, В.В. Ткаченко. – К.: Вид-во НАУ, 2009. – 364 с.

3. Мельник А.О. Архітектура комп'ютера / А.О.Мельник. – Луцьк; Волинська обл. друкарня, 2008. – 470 с.

4. Матвієнко М.П. Комп'ютерна логіка : підручник / М.П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

2. ЗАНЯТТЯ №2

ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЇВ ДЛЯ МНОЖЕННЯ ЧИСЕЛ

Ціль роботи – вивчити методи реалізації операції множення чисел в прямих кодах, одержати навички в дослідженні операційних пристроїв.

Теоретичні відомості

При множенні чисел у прямих кодах знакові та основні розряди обробляються роздільно. Для визначення знаку добутку здійснюють підсумовування за модулем 2 цифр, записаних в знакових розрядах співмножників. Будемо вважати, що множене Y і множник X – правильні двійкові дроби виду $X = x_1x_2\dots x_n$, $Y = y_1y_2\dots y_n$, де двійкові розряди $x_i, y_i \in \{0,1\}$. Тоді добуток Z модулів чисел дорівнює

$$Z = YX = Yx_12^{-1} + Yx_22^{-2} + \dots + Yx_i2^{-i} + \dots + Yx_n2^{-n}. \quad (2.1)$$

Множення Y і X може бути реалізоване шляхом виконання циклічного процесу, характер якого залежить від конкретної форми виразу (2.1). Один цикл множення складається з додавання чергового часткового добутку, що представляє собою добуток множеного на одну цифру множника, до суми часткових добутків. Розрізняють чотири способи множення.

Перший спосіб множення

Вираз (2.1) можна представити у вигляді

$$Z = YX = (((...((0 + Yx_n)2^{-1} + Yx_{n-1})2^{-1} + \dots + Yx_i)2^{-1} + \dots + Yx_1)2^{-1} .$$

Звідси випливає, що отримані суми часткових добутків в i -му циклі ($i = \overline{1, n}$) зводиться до обчислення

$$Z_i = (Z_{i-1} + Yx_{n-i+1})2^{-1}$$

з початковими значеннями $i=1, Z_0=0$, причому $Z_n=Z=YX$.

Множення здійснюється з молодших розрядів множника, сума часткових добутків зсувається вправо, а множене залишається нерухомим.

Другий спосіб множення.

Запишемо (2.1) у вигляді

$$Z = (((...((0 + Y2^{-n}x_n) + Y2^{-n+1}x_{n-1}) + \dots + Y2^{-1}x_1.$$

Очевидно, що процес множення може бути зведений до n -кратного виконання циклу

$$Z_i = Z_{i-1} + Y_i x_{n-i+1}, \quad Y_i = 2Y_{i-1},$$

з початковими значеннями $i=1, Y_0=Y2^{-n}, Z_0=0$. Множення здійснюється з молодших розрядів, множене зсувається вліво, а сума часткових добутків залишається нерухомою.

Третій спосіб множення.

Представимо (10.1) у виді

$$Z = (((...((0 + Y2^{-n}x_1)2 + Y2^{-n}x_2)2 + \dots + Y2^{-n}x_i)2 + \dots + Y2^{-n}x_n.$$

Отже, суму часткових добутоків у i -м циклі ($i=\overline{1,n}$) можна одержати по формулі

$$Z_i = 2Z_{i-1} + Y2^{-n} x_i.$$

Початковими значеннями є $i=1, Z_0=0$. Множення здійснюється зі старших розрядів множника, сума часткових добутоків зсувається вліво, а множене нерухоме.

Четвертий спосіб множення.

$$Z = (((...((0 + Y2^{-1} x_1) + Y2^{-2} x_2) + \dots + Y2^{-i} x_i) + \dots + Y2^{-n} x_n).$$

Процес множення може бути зведений до n -кратного виконання циклу

$$Z_i = Z_{i-1} + Y_{i-1} x_i, \quad Y_i = Y_{i-1} 2^{-1}$$

с початковими значеннями $i=1, Y_0=Y2^{-1}, Z_0=0$.

Множення виконується зі старших розрядів множника, сума часткових добутоків залишається нерухомою, а множене зсувається вправо.

Принцип побудови пристроїв, що реалізують різні способи множення, показаний на рис. 2.1, де $RG3$ – реєстр множеного, $RG1$ – реєстр добутку, $RG2$ – реєстр множника. Цифрами зазначені номери розрядів SM і реєстрів, а стрілками показаний напрямок зсуву кодів у реєстрах.

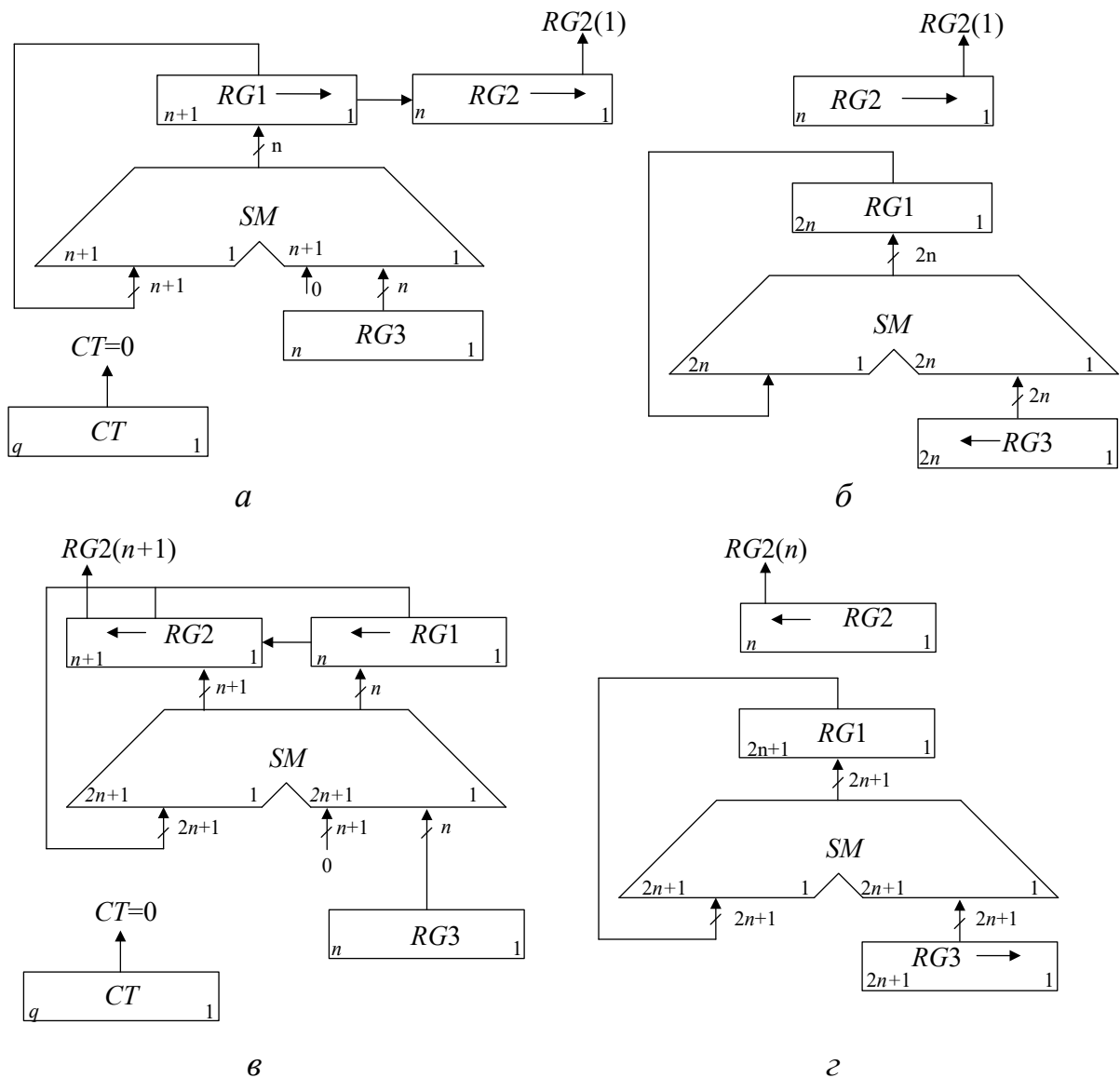


Рис. 2.1. Операційні схеми пристроїв для множення чисел: *a* – перший спосіб; *б* – другий спосіб; *в* – третій спосіб; *г* – четвертий спосіб

Під час множення *першим способом* (рис. 2.1, *a*) в першому такті *i*-го циклу аналізується значення $RG2[1]$ – молодшого (*n*-го) розряду регістру $RG2$, в якому знаходиться чергова цифра множника. Вміст $RG3$ додається до суми часткових добутоків, що знаходяться в регістрі $RG1$, якщо $RG2[1]=1$, або не додається, якщо $RG2[1]=0$. В другому такті здійснюється правий зсув у регістрах $RG1$ і $RG2$, що еквівалентно множенню їхнього вмісту на 2^{-1} . При зсуві цифра молодшого розряду регістру $RG1$ записується у

вивільнюваний старший розряд регістру $RG2$. Після виконання n циклів молодші розряди $2n$ -розрядного добутку будуть записані в регістр $RG2$, а старші – у $RG1$.

Максимальний час множення, якщо не застосовуються методи прискорення операції, визначається виразом $t_m = n(t_{\Pi} + t_3)$, де t_{Π} і t_3 – тривалості тактів підсумовування та зсуву відповідно.

Перед початком множення *другим способом* (рис. 2.1, б) множник X записують в регістр $RG2$, а множене Y – в молодші розряди регістру $RG3$ (тобто в регістрі $RG3$ установлюють $Y_0 = Y2^{-n}$). В кожному i -му циклі множення додаванням кодів $RG3$ і $RG1$ управляє цифра $RG2[1]$, а в регістрі $RG3$ здійснюється зсув вліво на один розряд, в результаті чого формується величина $Y_i = 2Y_{i-1}$. Оскільки сума часткових добутків в процесі множення нерухома, зсув в регістрі $RG3$ можна виконати суміщення в часі з підсумовуванням (як правило, $t_{\Pi} \geq t_3$). В цьому випадку $t_m = nt_{\Pi}$. Завершення операції множення визначається за нульовим вмістом регістру $RG2$, що також приводить до збільшення швидкодії, якщо множник ненормалізований.

При множенні *третім способом* (рис. 2.1, в) множник X записується в старші розряди $RG2$, при цьому $RG2[1]=0$. Вага молодшого розряду $RG3$ дорівнює 2^{-2n} , тому код в регістрі $RG3$ являє собою значення $Y2^{-n}$. В кожному циклі множення підсування виконується при $RG2[n+1]=1$. В регістрах $RG1$ і $RG2$ виконується лівий зсув. В результаті підсумовування вмісту $RG3$ і $RG1$ може виникнути перенос в молодший розряд регістру $RG2$, що реалізується на SM . Збільшення довжини $RG2$ на один розряд усуває можливість поширення переносу в розряди множника. Після виконання n циклів молодші розряди добутку будуть знаходитися в регістрі $RG1$, а старші – в регістрі $RG2$. Час множення третім способом визначається аналогічно першому способу.

Перед множенням *четвертим способом* (рис. 2.1, г) множник записують в регістр $RG2$, а множене – в старші розряди регістру $RG3$ (тобто в $RG3$ установлюють $Y_0=Y2^{-1}$). В кожному циклі цифра $RG2[n+1]$, що знаходиться в старшому розряді регістру $RG2$, управляє підсумовуванням, а в $RG3$ здійснюється правий зсув на один розряд, що еквівалентно множенню вмісту цього регістра на 2^{-1} . Час виконання множення *четвертим способом* складає $t_m=nt_p$, визначається аналогічно другому способу.

В ЕОМ при роботі із дробовими числами часто потрібно обчислювати не $2n$, а тільки $(n+1)$ цифр добутку й округляти його до n розрядів. В цьому випадку при реалізації другого способу можна зменшити довжину SM і $RG1$, а при реалізації четвертого – зменшити довжину SM , $RG1$ і $RG3$. Для того щоб похибка від відкидання молодших розрядів не перевищила половини ваги n -го розряду результату, в перерахованих вузлах досить мати тільки по l додаткових молодших розрядів, де l вибирається з умови

$$l \geq 1 + \log_2(n - l - 1).$$

Операція округлення здійснюється звичайно шляхом додавання одиниці до $n+1$ -го розряду результату після коми і відкиданням усіх розрядів, розташованих праворуч від n -го.

В операційних пристроях, що реалізують другий і четвертий способи множення, можна без пересилань кодів між регістрами обчислювати суму K попарних добудків $\sum_j^K X_j Y_j$, для чого досить черговий результат операції залишати в регістрі $RG2$, який в цьому випадку повинен мати додаткові старші розряди.

При реалізації другого, третього та четвертого способу можна обчислювати функцію $F = XY + G$, де $G = 0, g_1 g_2 \dots g_n$ – дробове число в такій самій формі, як і X та Y . Для цього необхідно перед множенням записати число G у регістр $RG1$ з урахуванням ваги розрядів. Наприклад, для третього

способу – в молодші розряди регістра $RG1$. Для можливості підключення до входів $RG1$ в різних циклах різні коди (спочатку операнд G , а потім в кожному циклі виходи SM) необхідно мати мультиплексор.

Найбільш простими є пристрої, що реалізують перший спосіб, а найбільш швидкодіючими – другий і четвертий. Однак другий спосіб не має особливих переваг порівняно з четвертим і, крім того, вимагає більших апаратних витрат при реалізації множення до n розрядів після коми.

Етапи побудови операційних пристроїв для множення чисел.

1. Вивчити алгоритм множення чисел заданим методом.
2. Побудувати операційну схему пристрою.
3. Розробити змістовний (функціональний) мікроалгоритм з використанням операторів присвоєння та зсуву.
4. Виконати логічне моделювання роботи пристрою за допомогою таблиці станів вузлів (регістрів, лічильника) у кожному такті. Перевірити правильність вибору розрядності вузлів на операційній схемі.
5. Побудувати функціональну схему з відображенням управляючих сигналів для всіх вузлів.
6. Розробити структурний мікроалгоритм, в якому змістовні мікрооперації замінюються на сукупність управляючих сигналів, що забезпечують виконання мікрооперацій. Сигнали, що формуються завжди разом, можна подати одним символом. Це зменшує кількість функцій вихідних сигналів при синтезі пристроїв управління.
7. Побудувати і відлагодити схему в системі ПРОГМОЛС-2 (AFDK).

Розглянемо побудову пристрою множення третім способом по наведеній вище схемі.

Операційна схема множення представлена на рис. 2.1, в, а функціональний мікроалгоритм подано на рис. 2.2, а.

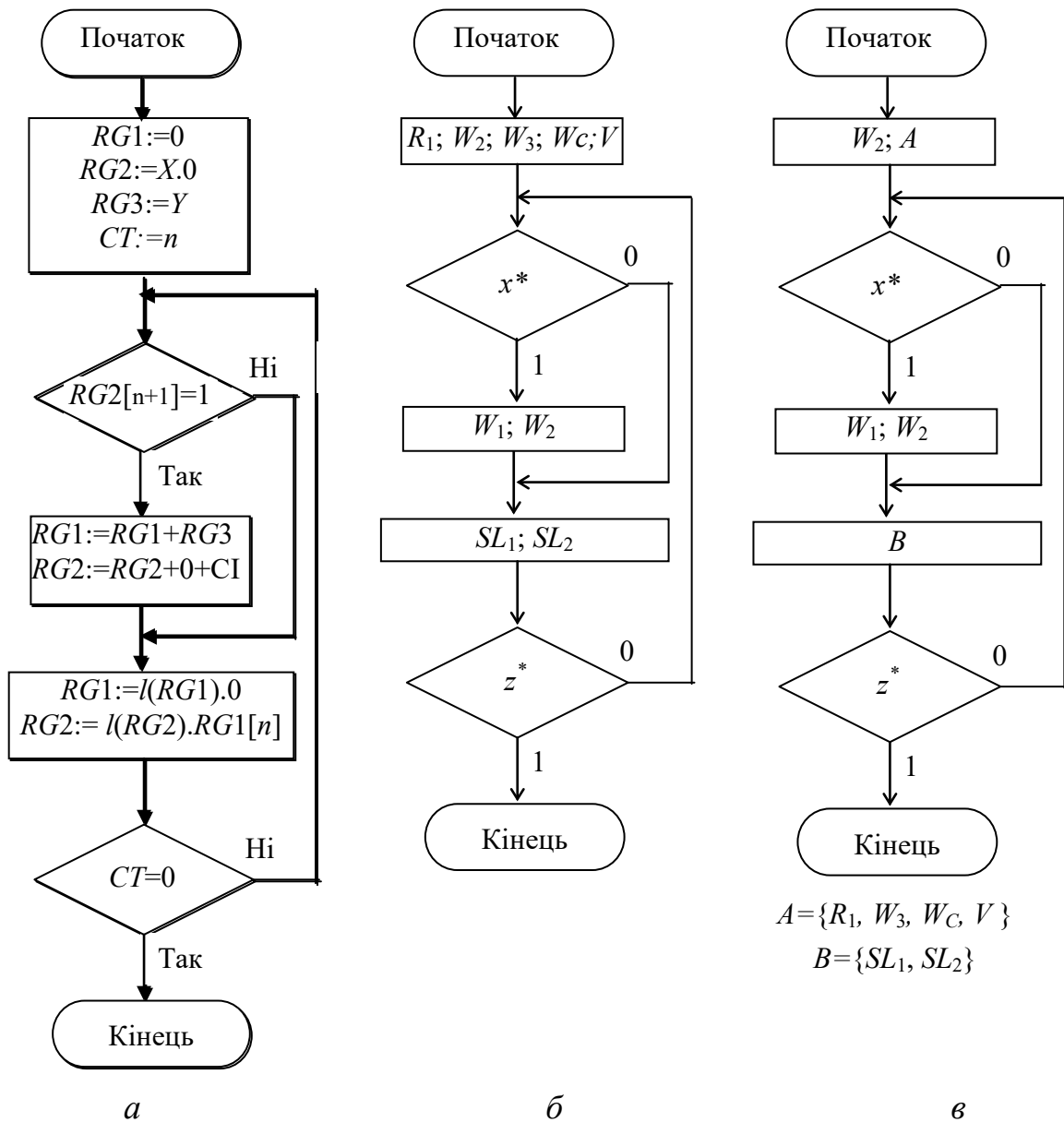


Рис. 2.2. Мікроалгоритми множення чисел за третім способом: а – функціональний; б – структурний; в – з об’єднаними сигналами

Для додатних операндів $X = 0,1011$ і $Y = 0,1101$, що мають по чотири основних дробових розрядів, значення станів регістрів і лічильника в кожному такті показано в табл. 2.1. У результаті моделювання перевіряється розрядності вузлів при $n = 4$.

Таблиця 2.1. Стани регістрів і лічильника при множенні

№ циклу	RG2	RG1	RG3	CT	Мікрооперації
0	1011.0	0000	1101	100	$RG1:=0; RG2:=X.0; RG3:=Y; CT:=n$
1	+00000	+1101	←	011	$RG1:=RG1+RG3; RG2:=RG2+0+CI$
	10110	1101			$RG1:=l(RG1).0; RG2:=l(RG2).RG1[n]$
	01101	1010			
2	11011	0100		010	$RG1:=l(RG1).0; RG2:=l(RG2).RG1[n]$
3	+00000	+1101	←	001	$RG1:=RG1+RG3; RG2:=RG2+0+CI$
	11100	0001			$RG1:=l(RG1).0; RG2:=l(RG2).RG1[n]$
	11000	0010			
4	+00000	+1101	←	000	$RG1:=RG1+RG3; RG2:=RG2+0+CI$
	11000	1111			$RG1:=l(RG1).0; RG2:=l(RG2).RG1[n]$
	10001	111.0			

Результат

На функціональній схемі (рис.2.3) відображаються управляючі сигнали і логічні умови x^*, z^* .

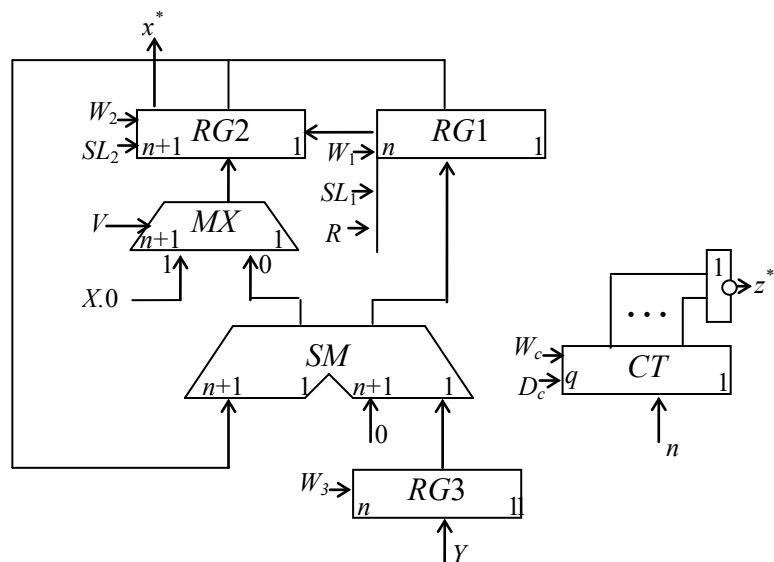


Рис.2.3. Функціональна схема пристрою множення

Занесення інформації в $RG2$ відбувається з двох джерел через мультиплексор MX (при $V=1$ заноситься операнд X в старші розряди, а в молодший записується 0). При $V=0$ в $RG2$ заноситься інформація з виходів SM). Інші управляючі сигнали позначаються символами: W – занесення даних; SL – зсув ліворуч; D – декремент; R – скид у нуль вмісту регістра.

Враховуючи позначення сигналів на схемі пристрою, будемо структурний мікроалгоритм та мікроалгоритм із об'єднанням сигналів, що формуються разом (рис. 2.2, б і 2.2. в).

На базі одержаних даних (рис. 2.2, в та рис. 2.3) в програмному комплексі ПРОГМОЛС-2 можна набрати та налагодити схему пристрою.

ПРАКТИЧНА РОБОТА №2

Підготовка до роботи

1. Варіант завдання визначається молодшими двійковими розрядами $a_6, a_5, a_4, a_3, a_2, a_1$ десяткового номера варіанту (номер групи і номер студента у списку) відповідно табл.2.2.

2. Відповідно до запропонованої послідовності етапів синтезу пристроїв множення виконати всі пункти побудови пристрою множення згідно таблиці варіантів (табл. 2.2).

3. Для побудови функціональної схеми можна використовувати комбінаційний суматор, асинхронні регістри і лічильник, що мають окремі входи керування різними мікроопераціями (W, SL, SR, D і т.і.) На схемі повинна бути зазначена розрядність регістрів. Третій спосіб множення використовується для обчислення функції $F = XY + G$, а інші методи – для функції $F = XY$, тобто тільки для множення. Операнд G перед початком циклів множення записується в $RG1$ через MX . Для цього виконується окрема мікрооперація, яка повинна бути відображена в мікроалгоритмі.

4. Згідно варіанту завдання (табл. 2.2) треба виконати числовий приклад обчислень у вигляді таблиці станів вузлів в кожному такті. Таблиця станів може використовуватися у якості тесту при налагодженні пристрою.

Таблиця 2.2. Таблиця варіантів

a_3	a_2	a_1	Спосіб множення, розрядність операндів	Значення додатних операндів			Повна операція
				X	Y	G	
0	0	0	1-й, 7	,1 $a_6 a_5 a_4$ 101	,1001101	-	$F=XY$
0	0	1	2-й, 5	,1 $a_6 a_5 a_4$ 0	,10011	-	$F=XY$
0	1	0	3-й, 7	,1 $a_6 a_5 a_4$ 001	,0100111	,1101011	$F=XY+G$

0	1	1	4-й, 6	,1 $a_6 a_5 a_4$ 01	,110011	-	$F=XY$
1	0	0	1-й, 6	,11 $a_6 a_5 a_4$ 0	,101111	-	$F=XY$
1	0	1	2-й, 6	,10 $a_6 a_5 a_4$ 1	,111010	-	$F=XY$
1	1	0	3-й, 6	,10 $a_6 a_5 a_4$ 1	,101111	,100101	$F=XY+G$
1	1	1	4-й, 5	,10 $a_6 a_5 a_4$,11001	-	$F=XY$

Порядок виконання роботи

1. Набрати в комплексі ПРОГМОЛС-2 розроблену функціональну схему пристрою множення. Входами пристрою є інформаційні входи операндів X, Y, G та всі управляючі входи: W, SL, SR і т.і. (рис.2.4). Виходами пристрою є логічні умови x^*, z^* .

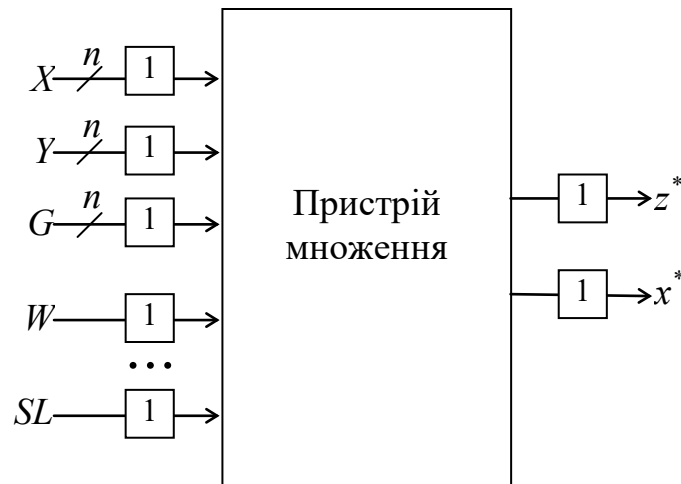


Рис. 2.4. Підготовка схеми для відлагодження

2. Відлагодити схему і виконати підготовлений цифровий приклад. Сигнали подаються мишею вручну безпосередньо на входи пристрою.

Послідовність сигналів залежить від логічних умов відповідно схемі мікроалгоритму.

4. Дослідити зазначені викладачем часові параметри схеми.

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, порядок виконання роботи, всі схеми, формули і таблиці, отримані при виконанні теоретичного завдання і в процесі моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Контрольні запитання

1. Охарактеризуйте чотири основні методи множення чисел.
2. Як розрахувати розрядність вузлів операційного пристрою?
3. Що таке мікрооперація і мікроалгоритм операції?
4. Охарактеризуйте основні етапи проектування пристроїв множення.
5. Як перейти від функціонального (змістовного) мікроалгоритму до структурного мікроалгоритму?
6. Як визначити необхідну тривалість керуючих сигналів?
7. Як визначити тривалість циклу множення?
8. Порівняйте за часом виконання операції різні способи множення.
9. Які додаткові обчислювальні функції мають різні методи множення.

10. Як перейти від операційної схеми до функціональної?

Література

1. Жабін В.І., Арифметичні та управляючі пристрої цифрових ЕОМ: Навч. Посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, С.Г. Стіренко. – К.: ВЕК+, 2008. – 176 с.
2. Жабін В.І. Прикладна теорія теорія цифрових автоматів: Навч. Посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, В.В. Ткаченко. – К.: Вид-во НАУ, 2009. – 364 с.
3. Мельник А.О. Архітектура комп'ютера / А.О.Мельник. – Луцьк; Волинська обл. друкарня, 2008. – 470 с.
4. Матвієнко М.П. Комп'ютерна логіка : підручник / М.П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

3. ЗАНЯТТЯ №3.

ПОБУДОВА ФУНКЦІОНАЛЬНИХ СХЕМ ВИКОНАННЯ АРИФМЕТИЧНИХ ОПЕРАЦІЙ

Побудова функціональних схем за тематикою курсової роботи

3.1 побудувати на базі операційної схеми функціональну схему з відображенням управляючих сигналів, входів для запису операндів при ініціалізації пристрою і схем формування внутрішніх логічних умов; схему подати як рисунок у пояснювальній записці;

3.2 у моделюючій програмі ПРОГМОЛС (AFDK) побудувати функціональну схему виконання операції, виконати числовий приклад, управляючі сигнали подавати вручну, надати скріншот схеми з числовим результатом;

3.3 розробити структурний мікроалгоритм (мікрооперації замінюються управляючими сигналами виду W, SL, SR тощо), а також закодований мікроалгоритм (сигнали, що завжди формуються разом, позначаються одним символом);

3.4 для операції з парним номером $x_3x_2x_1$ подати граф управляючого автомата Мура з відображенням кодів вершин, а для непарного номера – граф автомата Мілі;

3.5 забезпечити подвійну тривалість управляючого сигналу додавання, що виконується у циклі; надати таблиці кодування станів автомата, управляючих сигналів і логічних умов.

3.5 розробити управляючий автомат на тригерах та елементах булевого базису. Відповідно із кодом x_2x_1 , що приймає значення 00,01,10,11, вибрати відповідно JK-, RS-, T-, D-тригери для побудови автомата; виконати сумісну мінімізацію функцій управління тригерами та вихідних сигналів;

3.6 у моделюючій програмі ПРОГМОЛС (AFDK) побудувати керуючий автомат виконання даної операції, надати часову діаграму роботи автомата за алгоритмом з ілюстрацією вихідних сигналів і виходів тригерів.

3.7 у форматі А2 або А3 виконати креслення функціональної схеми управляючого автомата з урахуванням діючих стандартів для схем Е2.

Функціональна схема визначає основні функціональні частини виробу, їх призначення і взаємозв'язок, роз'ясняє визначені процеси, що протікають в окремих функціональних частинах або у виробі в цілому. Функціональні частини таких схем зображуються, як правило, у виді прямокутників.

У функціональних схемах використовуються умовні графічні позначення (УГП) функціональних частин. УГП елемента має форму прямокутника, до якого підводять лінії виводів. УГП елемента в загальному випадку може містити три поля: основне і два додаткових, котрі розташовують ліворуч і праворуч від основного (рис. 4.1).

В основному полі УГП поміщають позначення функції, яка реалізується елементом (табл. 4.1). В додаткових полях поміщають інформацію про призначення виводів (мітки виводів, вказівники).

Крім виду, зазначеного на рис 4.1, УГП може також складатися:

- тільки з основного поля;
- з основного поля і одного додаткового (праворуч або ліворуч від основного);

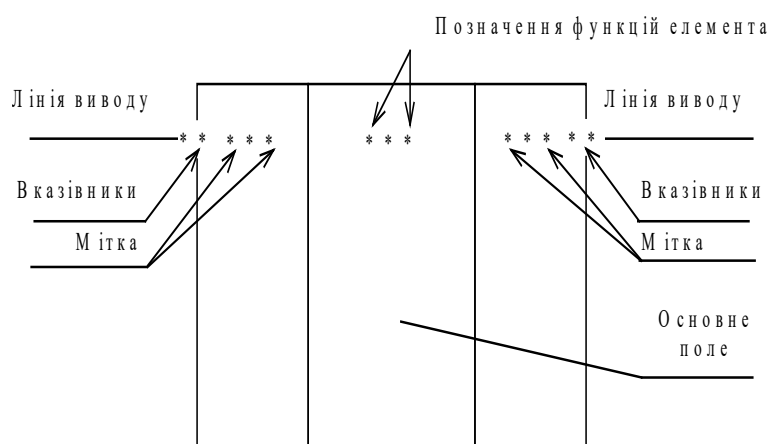


Рисунок 4.1 - Умовне графічне позначення елемента

Допускаються додаткові поля розділяти на зони, відокремлені горизонтальною рисою. Основне і додаткове поля можуть бути не відділені

лінією. При цьому відстань між буквенними, цифровими і буквенно-цифровими позначеннями, поміщеними в основне і додаткові поля, визначається однозначністю розуміння кожного позначення.

Входи елемента зображують з лівої сторони УГП, виходи – з правої сторони УГП. Двонаправлені виводи і виводи, що не несуть логічної інформації, зображують з правої або лівої сторони УГП.

При підведенні ліній виводів до контуру УГП не допускається:

- проводити їх на рівні сторін прямокутника;
- проставляти на них стрілки, що вказують напрямок передачі інформації.

Таблиця 4.1 - Позначення елементів

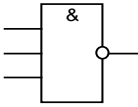
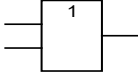
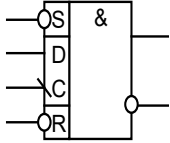
Найменування	Позначення
Тригер	T
Двоступінчастий тригер	TT
Логічне “І”	&
Логічне “АБО”	1
Виключне “АБО”	=1
Генератор імпульсів	G

Таблиця 4.2 - Позначення входів елементів

Найменування	Позначення
Прямий статичний вхід	
Інверсний статичний вхід	
Прямий динамічний вхід	
Інверсний динамічний вхід	

Таблиця 4.3 – Приклади УГП елементів

Найменування	Позначення
--------------	------------

Елемент 3І-НЕ	
Елемент 2АБО	
D-тригер	

Розміри сторін УГП і відстані між выводами повинні бути кратні М. Діаметр вказівника інверсного виводу повинний дорівнювати М. При виконанні курсової роботи прийняти М=5 мм.

Позначення функцій, виконуваних елементом, утворюють із прописних букв латинського алфавіту, арабських цифр і спеціальних знаків, записаних без пробілів. Кількість знаків у позначенні функції не обмежена, однак варто прагнути до їх мінімального числа при збереженні однозначності розуміння кожного позначення.

В таблиці 4.1 приведені деякі стандартні позначення функцій, яких варто дотримуватись при виконанні курсової роботи.

Виводи елементів, які несуть логічну інформацію, підрозділяють на статичні і динамічні, а також на прямі та інверсні. Властивості виводів позначають за допомогою міток відповідно до табл. 4.2.

Приклади деяких УГП приведені в табл. 4.3.

Література

Базова:

1. Жабін В.І., Верба О.А. Комп'ютерна логіка. Курсова робота. Навчальний посібник. <https://ela.kpi.ua/handle/123456789/50134>

2. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів: Навч. посібник.— К.: Вид-во НАУ, 2009.— 364 с.
<https://campus.kpi.ua/tutor/index.php?mode=mob&show&irid=200206>

3. Жабін В.І., Ткаченко В.В. Цифрові автомати. Практикум. – К.: ВЕК+, 2004.— 160 с.

Допоміжна:

5. Матвієнко М.П. Комп'ютерна логіка. Підручник. Вид. 2-ге перероб. та доп. – Київ: Видавництво Ліра – К, 2017. – 324 с.

6. Лупенко С.А., Пасічник В.В., Тиш Є.В. Комп'ютерна логіка. Навчальний посібник для ВНЗ. Вид. Магнолія, 2017.— 354 с.

7. ДСТУ 3008-2015 «Державний стандарт України. Документація. Звіти в сфері науки і техніки. Структура і правила оформлення».

8. ДСТУ ISO 5457:2006 (ISO 5457:1999, IDT) Національний стандарт України. Документація технічна на вироби. Кресленики. Розміри та формати.

9. ДСТУ ГОСТ 2.702:2013 ЄСКД. Правила виконання електричних схем (ГОСТ 2.702-2011, IDT).

10. ДСТ 2.104-2006. ЄСКД. Основні написи (Міждержавний. В Україні - ДСТУ 2.104-2006).

11. Рамки і штампи. <https://sprosi.xyz/articles/ramka-dlya-kursovyh-diplomov-i-referata/>

4. ЗАНЯТТЯ №4

ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ПРИСТРОЇВ ДЛЯ ДІЛЕННЯ ЧИСЕЛ

Ціль роботи – вивчити методи ділення чисел в прямих кодах і способи їх апаратурної реалізації, придбати навички в налагодженні та дослідженні операційних пристроїв.

Теоретичні відомості

Існують два основних методи ділення чисел: ділення з відновленням і без відновлення від'ємної остачі. Реалізація цих методів вимагає приблизно однакового обсягу устаткування, але при діленні першим методом потрібно більше часу для виконання операції. Тому метод ділення чисел без відновлення залишку є кращим.

Нехай ділене X и дільник Y є n -розрядними правильними дробами, представленими в прямому коді. В цьому випадку знакові й основні розряди операндів обробляються окремо. Знак результату визначається шляхом підсумовування за модулем 2 цифр, записаних в знакових розрядах.

Алгоритм ділення чисел без відновлення залишку зводиться до виконання наступних дій.

1. Одержати різницю $R_0 = X - Y$. Якщо $R_0 \geq 0$, то цифра Z_0 частки, що має вагу 1, а при $R_0 < 0$ – дорівнює 0. Різниця R_0 є залишком.
2. Подвоїти залишок (одержати $2R_i$).
3. Якщо $2R_i < 0$, то додати, а якщо $2R_i \geq 0$, то відняти Y . Якщо знову отриманий залишок $R_{i+1} \geq 0$, то $Z_{i+1} = 1$, інакше $Z_{i+1} = 0$.

4. Повторити пп. 2 і 3 $n-1$ раз.

Пункт 2 алгоритму можна замінити пунктом “зменшити в два рази дільник”. Наявність двох інтерпретацій другого пункту дає два основних варіанти реалізації ділення.

Перший спосіб ділення.

При реалізації ділення за першим варіантом здійснюється зсув вліво залишку при нерухомому дільнику. На рис. 4.1 показана можлива побудова пристрою ділення. Чергова остача формується в регістрі $RG2$ (у вихідному стані в цьому регістрі записаний X). Виходи $RG2$ підключені до входів суматора SM безпосередньо, тобто ланцюги видачі коду з $RG2$ не потрібні. Дільник Y знаходиться в регістрі $RG1$. Результат формується в регістрі $RG3$ за $(n + 1)$ циклів. Знак остачі визначається розрядом $RG2[n+2]$. Розряд $RG3[n+1]$ використовується для визначення кінця операції, ознакою цього є маркерний нуль на виході розряду. Максимальний час одержання цифри результату визначається виразом $t_{ц} = t_{д} + t_{з}$, де $t_{д}$ – тривалість виконання мікрооперації додавання/віднімання; $t_{з}$ – тривалість виконання мікрооперації зсуву. Час для одержання $n+1$ цифри частки визначається виразом $t = (n+1) t_{ц}$.

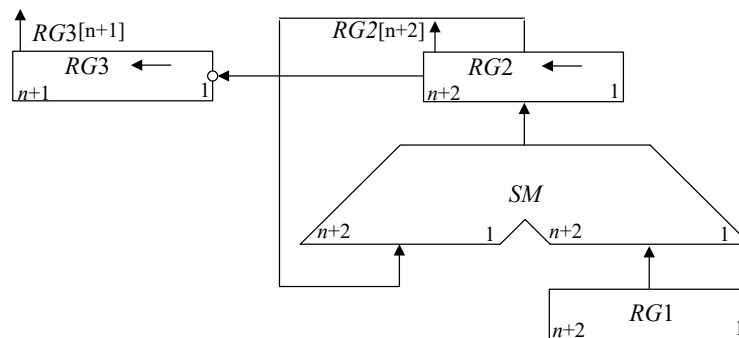


Рис. 4.1. Операційна схема ділення за першим способом із зсувом остачі

Другий спосіб ділення.

При реалізації ділення другим способом (із зсувом дільника) збільшується розрядність регістрів $RG1$, $RG3$ і суматора SM (рис. 4.2). В даному випадку процеси додавання/віднімання і зсуву можуть бути суміщені у часі. Отже,

для ділення за другим способом час одержання цифри результату дорівнює $t_{Ц} = t_{д}$. Цифра результату формується на виході переносу суматора $SM(p)$. Загальний час ділення визначається як $t = (n + 1)t_{д}$.

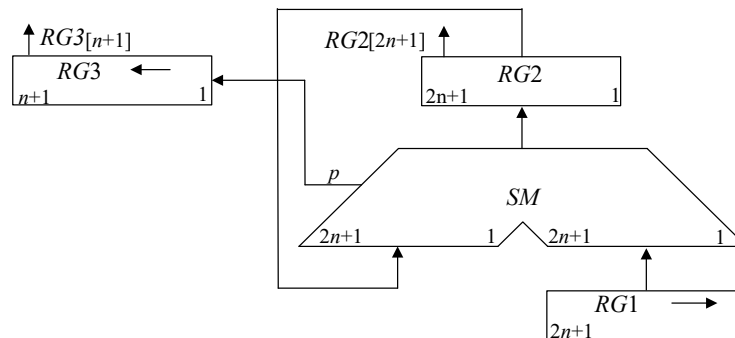


Рис. 4.2. Операційна схема пристрою ділення другим способом із зсувом дільника

При діленні чисел з фіксованою комою повинна бути передбачена можливість фіксації переповнення розрядної сітки. Ознака переповнення формується, якщо в першому циклі цифра результату дорівнює одиниці.

Етапи розробки операційного пристрою для ділення чисел.

1. Вивчити алгоритм ділення чисел заданим методом.
2. Побудувати операційну схему пристрою.
3. Розробити змістовний (функціональний) мікроалгоритм з використанням операторів присвоєння, зсуву тощо.
4. Виконати логічне моделювання роботи пристрою за допомогою таблиці станів регістрів у кожному такті. Перевірити правильність вибору розрядності вузлів на операційній схемі.
5. Побудувати функціональну схему з відображенням управляючих сигналів виконання мікрооперацій для всіх вузлів.
6. Розробити структурний мікроалгоритм, в якому змістовні мікрооперації замінюються на сукупність управляючих сигналів, що забезпечують виконання мікрооперацій. Сигнали, що формуються завжди разом, можна

подати одним символом. Це зменшує кількість функцій вихідних сигналів при синтезі пристроїв управління.

7. Побудувати і налагодити схему в системі ПРОГМОЛС-2 (AFDK).

Змістовний мікроалгоритм ділення за другим способом подано на рис.4.2. Залежно від знакового розряду $RG2[2n+1]$ до регістра $RG2$ додається або віднімається код із регістра $RG1$. Віднімання забезпечується інверсією коду із $RG1$ і додаванням одиниці ($D=1$) на вхід переносу суматора SM . Чергова цифра результату записується в $RG3$ із виходу переносу суматора при зсуві.

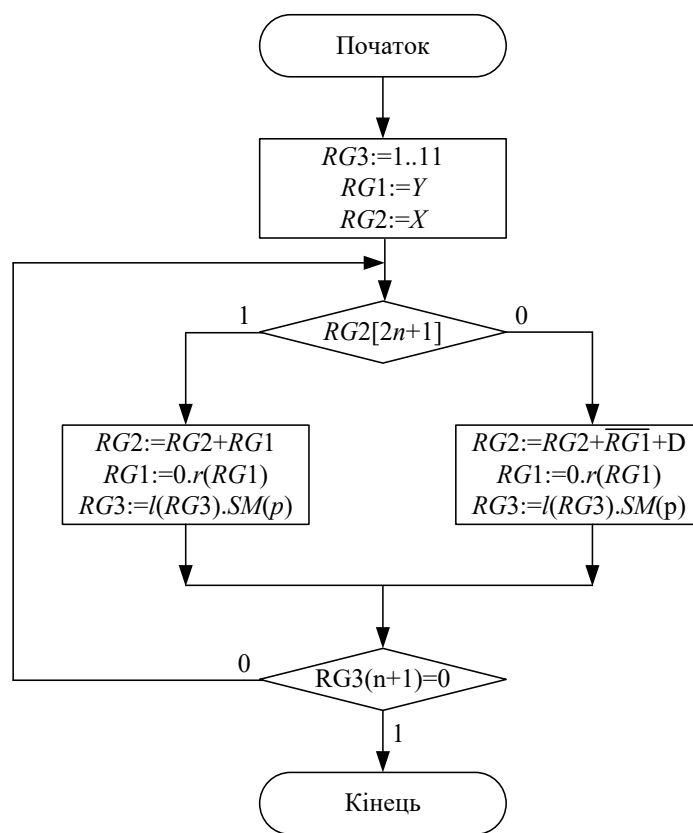


Рис. 4.3. Змістовний мікроалгоритм виконання операції ділення із зсувом дільника

Цифрова діаграма стану вузлів при діленні $Z = Y / X$ двох додатних мантис наведена в табл. 3.1, в якій $Y = 0,1001$; $X = 0,1100$; $0 < Y, X < 1$; $Y < X$.

При побудові функціональної схеми на базі операційної схеми (рис. 4.1 і рис. 4.2) необхідно мати на виходах $RG1$ перетворювач дільника у від'ємне число в доповняльному коді для реалізації мікрооперації віднімання.

Перед початком виконання циклів ділення необхідно записати ділене в *RG2*. В процесі ділення в цей регістр передається інформація з виходів суматора *SM*. Забезпечити запис даних в *RG2* з двох напрямків дозволяє використання мультиплексора.

Таблиця 4.1. Стани регістрів при ділення чисел

№ циклу	<i>RG3</i>	<i>RG2</i>	<i>RG1</i>	Мікро-операція
	1 1111	0 10010000	0 11000000	
1		0 10010000 1 01000000 1 11010000		-Y
	1 1110		0 01100000	Зсув
2	1 1110	1 11010000 0 01100000 0 00110000	0 01100000	+Y
	1 1101		0 00110000	
3	1 1101	0 00110000 1 11010000 0 00000000	0 00110000	-Y
	1 1011		0 00011000	Зсув
4	1 1011	0 00000000 1 11101000 1 11101000	0 00011000	-Y
	1 0110		0 00001100	Зсув
5	1 0110	1 11101000 0 00001100 1 11110100	0 00001100	+y
	0 1100		0 00000110	Зсув

Для визначення кінця операції використовується маркерний нуль. Якщо перед початком обчислень в усі розряди регістру *RG3* записати одиниці, то перший нуль в цьому розряді після зсуву означає кінець операції. В першому циклі ділення (якщо немає переповнення розрядної сітки) завжди формується нульова цифра частки. Такий підхід дозволяє спростити пристрій за рахунок усунення лічильника циклів.

ПРАКТИЧНА РОБОТА №4

Підготовка до роботи

1. Варіант завдання визначається шістьма молодшими двійковими розрядами $a_6, a_5, a_4, a_3, a_2, a_1$ десяткового номера варіанту (номер групи і номер студента у списку) відповідно табл. 4.2.

2. Виконати всі запропоновані вище пункти синтезу пристрою ділення згідно таблиці варіантів (табл. 4.2).

3. Для побудови функціональної схеми можна використовувати комбінаційний суматор, асинхронні регістри і лічильник, що мають окремі входи керування різними мікроопераціями (W, SL, SR, D тощо). На схемі повинна бути зазначена розрядність регістрів.

4. Згідно варіанту завдання (табл. 4.2) треба виконати числовий приклад обчислень у вигляді таблиці станів вузлів в кожному такті. Таблиця станів може використовуватися у якості тесту при налагодженні пристрою.

Таблиця 4.2. Таблиця варіантів

a_3	a_2	a_1	Спосіб ділення, розрядність операндів	Додатні дробові операнди	
				X	Y
0	0	0	1-й, 7	,1 $a_6 a_5 a_4$ 101	,1111101
0	0	1	2-й, 5	,10 $a_6 a_5 a_4$,11011
0	1	0	1-й, 6	,10 $a_6 a_5 a_4$ 1	,110111
0	1	1	2-й, 6	,10 $a_6 a_5 a_4$ 0	,110011
1	0	0	1-й, 5	,10 $a_6 a_5 a_4$,11011
1	0	1	2-й, 7	,10 $a_6 a_5 a_4$ 01	,1110101
1	1	0	1-й, 4	,1 $a_6 a_5 a_4$,1111

1	1	1	2-й, 4	,0 $a_6 a_5 a_4$,1110
---	---	---	--------	------------------	-------

Порядок виконання роботи

1. Набрати в комплексі ПРОГМОЛС-2 розроблену функціональну схему пристрою ділення. Входами пристрою є інформаційні входи операндів X, Y та всі управляючі входи: W, SL, SR, \dots (рис.4.4). Виходами пристрою є логічні умови – старші розряди регістрів $RG2$ і $RG3$ (відповідно x^* і z^*).
2. Відлагодити схему і виконати підготовлений цифровий приклад. Сигнали подаються мишею вручну безпосередньо на входи пристрою. Послідовність сигналів залежить від логічних умов відповідно схемі мікроалгоритму.
3. Дослідити зазначені викладачем часові параметри схеми.

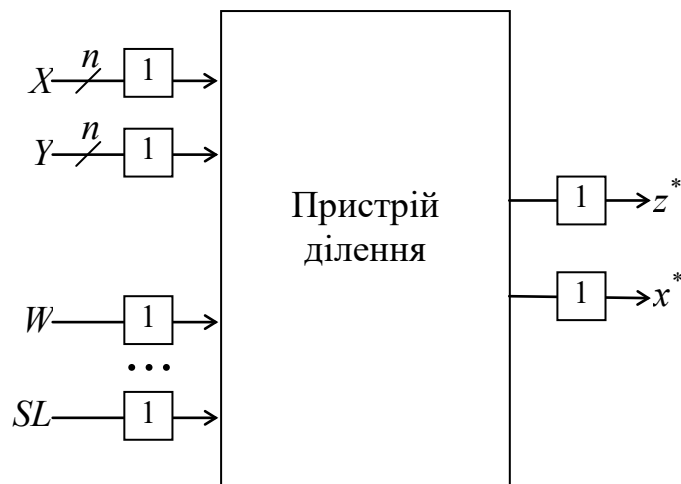


Рис. 3.4. Підготовка схеми для налагодження

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, порядок виконання роботи, всі схеми, формули і таблиці,

отримані при виконанні теоретичного завдання і в процесі моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Контрольні запитання

1. Охарактеризуйте два основні методи ділення чисел.
2. Як розрахувати розрядність вузлів операційного пристрою?
3. Що таке мікрооперація і мікроалгоритм операції?
4. Охарактеризуйте основні етапи проектування пристроїв ділення.
5. Як перейти від функціонального (змістовного) мікроалгоритму до структурного мікроалгоритму?
6. Як визначити необхідну тривалість керуючих сигналів?
7. Як визначити тривалість циклу ділення, часу виконання операції?
8. Порівняйте за часом виконання операції різні способи ділення.
9. В яких пристроях можна суміщати мікрооперації підсумовування/віднімання і зсуву? Чому це можливо?
10. Чи можна зменшити довжину регістрів операційного пристрою при реалізації ділення чисел за другим варіантом, якщо результат повинний бути представлений q розрядами ($q < n$)?
11. Як перейти від операційної схеми до функціональної?

Література

1. Жабін В.І., Арифметичні та управляючі пристрої цифрових ЕОМ: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, С.Г. Стіренко. – К.: ВЕК+, 2008. – 176 с.

2. Жабін В.І. Прикладна теорія теорія цифрових автоматів: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, В.В. Ткаченко. – К.: Вид-во НАУ, 2009. – 364 с.

3. Мельник А.О. Архітектура комп'ютера / А.О.Мельник. – Луцьк; Волинська обл. друкарня, 2008. – 470 с.

4. Матвієнко М. П. Комп'ютерна логіка : підручник / М. П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

5. ЗАНЯТТЯ №5

ДОСЛІДЖЕННЯ ОПЕРАЦІЙ З ЧИСЛАМИ У ФОРМАТІ З ПЛАВАЮЧОЮ КОМОЮ

Мета роботи: Дослідити машинні методи виконання операцій з числами в форматі з плаваючою комою.

Теоретичні відомості

Форми подання чисел у ЕОМ

В ОТ переважно використовуються дві форми подання чисел:

- подання чисел з фіксованою комою;
- подання чисел із плаваючою комою.

Будь-яке число X у позиційній системі числення з основою k можна записати як

$$X = k^P M,$$

де M – мантиса числа X , а P – порядок числа X .

Якщо порядок фіксований для всіх чисел, то говорять, що число подане у формі з фіксованою комою. В цьому випадку числа задаються тільки одним об'єктом – мантисою.

При поданні чисел у формі з фіксованою комою положення коми залишається незмінним для всіх чисел, з якими оперує машина, тобто спеціальні розряди для коми не виділяються. З метою спрощення обчислення масштабних коефіцієнтів кому звичайно фіксують перед старшим розрядом мантиси або після молодшого розряду.

При фіксації коми перед старшим розрядом мантиси ЕОМ оперує із числами, які менше одиниці. Якщо число розрядів для запису мантис становить n , то мінімальне (за абсолютною величиною) число, що може бути подане в машині, дорівнює k^{-n} , а максимальне дорівнює $1 - k^{-n}$.

У другому випадку, при фіксації коми після молодшого розряду, в машині виконуються операції над цілими числами, абсолютні величини яких перебувають у межах від 0 до $k^n - 1$.

Розрядна сітка для подання чисел у ЕОМ з фіксованою комою складається із двох частин: один розряд для подання знака, інші розряди для подання мантиси.

Якщо кожне число задається парою P і M , то таке подання називають формою із плаваючої комою.

При поданні чисел у формі із плаваючою комою порядок P може бути додатним або від'ємним цілим числом. Мантиса ж у більшості випадків є додатним або від'ємним правильним дробом, причому

$$k^{-1} \leq M \leq 1. \quad (1)$$

Це означає, що старший розряд модуля мантиси завжди дорівнює 1 (мантиса нормалізована). Якщо в ЕОМ для запису порядку використовується m розрядів, а для запису мантиси – n розрядів, то в цій машині може бути подане наступне максимальне (за абсолютною величиною) число

$$k^{k^m-1}(1-k^{-n}) = k^{k^m-1} - k^{k^m-n-1}.$$

У свою чергу, мінімальне за абсолютною величиною число, що відрізняється від нуля, у такій машині дорівнює

$$k^{-k^m+1}k^{-1} = k^{-k^m}.$$

Крім зазначених вище розрядів, для подання порядку й мантиси, у розрядній сітці ЕОМ із плаваючою комою є також розряди для подання знаків порядку й мантиси (рис. 5.1).

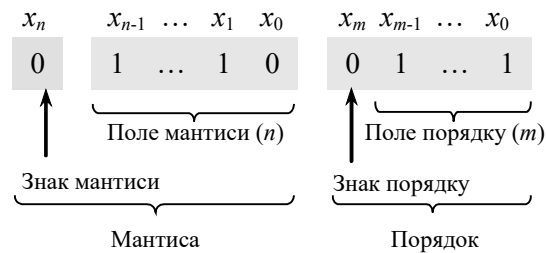


Рис. 5.1. Приклад запису числа у формі з плаваючою комою

Приклад.

Записати у формі із плаваючою комою ($m=n=4$) двійкові доповнювальні коди чисел: $X=9_{(10)}$, $Y=-9_{(10)}$, $Z=-9/256_{(10)}$.

$P_{(X)} = 0, 1 0 0$	$M_{(X)} = 0, 1001$
$P_{(Y)} = 0, 1 0 0$	$M_{(Y)} = 1, 0111$
$P_{(Z)} = 1, 1 0 0$	$M_{(Z)} = 1, 0111$

У ЕОМ, де використовується подання чисел у формі із плаваючою комою, арифметичні операції виконуються як над мантисами чисел, так і над їхніми порядками. Часто також необхідно виконувати операцію нормалізації чисел, сутність якої складається у виконанні умови $k^{-1} \leq M \leq 1$. Тому машини із

плаваючою комою є більш складним і менш швидкодіючим, ніж машини з фіксованою комою. З іншого боку, у машинах з фіксованою комою через масштабування виникають труднощі при програмуванні.

Для універсальних комп'ютерів розроблений стандарт ANSI/IEEE 754-2008 на подання чисел із плаваючою комою. В цьому стандарті визначені різні формати чисел. У кожному форматі використовують схований старший розряд мантиси з вагою 2^{-1} , який завжди дорівнює 1 для додатних чисел. Таким чином, мантиси змінюються в межах $1 \leq M < 2$. Вводиться “зміщений порядок” $P_{zm} = P + (2^{m-1} - 1)$, де m – кількість розрядів для подання зміщеного порядку. Для 32-розрядної та 64-розрядної сітки визначені два основні базові формати: короткий та довгий

Короткий формат (рис. 5.2) має 32 розряди, з яких 8 розрядів призначені для подання порядку, 23 розряди - для подання мантиси й один розряд - для знака мантиси. Спеціального розряду для знаку порядку немає. Таким чином, коди нулевого та від’ємних порядків P будуть подані “зміщеними порядками” P_{zm} , які є додатними числами з нулем в старшому розряді, а коди додатних порядків – з одиницями в старшому розряді. Введення “зміщеного порядку” дозволяє звести операції над порядками до арифметичної дії над цілими додатними числами. Для короткого формату $P_{zm} = P + (2^{8-1} - 1) = P + 127$.

Отже, максимальний додатний порядок числа в короткому форматі дорівнює

$$11111111_{(2)} - 01111111_{(2)} = 10000000_{(2)} = 128_{(10)},$$

а максимальний за модулем від’ємний порядок

$$00000000_{(2)} - 01111111_{(2)} = -01111111_{(2)} = -127_{(10)}.$$

ЗМ	2^7	2^6	...	2^0	2^{-2}	2^{-3}	...	2^{-24}
Знак мантиси	Зміщений порядок (8 розрядів)				Мантиса (23 розряди)			

а

ЗМ	2^{10}	2^9	...	2^0	2^{-2}	2^{-3}	...	2^{-53}
Знак мантиси	Зміщений порядок (11 розрядів)				Мантиса (52 розряди)			

б

Рис. 5.2. Формати чисел із плаваючою комою: а – короткий формат; б – довгий формат.

Довгий формат, що використовується для обчислень із підвищеною точністю, має 64 розряди. Для порядку виділяється 11 розрядів, а для мантиси – 52 розряди.

Як видно з рис. 2, старший розряд мантиси з вагою 2^{-1} не записується. В цьому розряді, якщо мантиса нормалізована, завжди буде одиниця і вона не записується для економії числа розрядів. Ця одиниця носить назву «прихована одиниця».

Додавання чисел із плаваючою комою

Суму двох чисел $X = 2^{P_x} M_X$ і $Y = 2^{P_y} M_Y$, поданих у форматі із плаваючою комою, можна записати у вигляді

$$2^{P_x} M_X + 2^{P_y} M_Y = 2^{P_z} M_Z.$$

У загальному випадку дійсні ваги однакових розрядів мантиси різних чисел із урахуванням порядків чисел можуть відрізнятися. Тому для додавання чисел із плаваючою комою необхідно привести їх до загального порядку $P_{\text{заг}}$, в якості якого зручно обрати більший порядок з двох доданків

$$P_{\text{заг}} = \max(P_x, P_y).$$

При цьому зменшенню за рахунок зсуву праворуч підлягає мантиса числа з меншим порядком. В протилежному випадку виникає переповнення розрядної сітки мантиси числа, що перетворюється. Після цього суму двох чисел можна подати у вигляді

$$2^{P_{\text{заг}}} M_X + 2^{P_{\text{заг}}} M'_Y = 2^{P_{\text{заг}}} (M_X + M'_Y),$$

де за M'_Y прийнято перетворену мантису числа з меншим порядком.

Можна визначити наступні етапи додавання чисел із плаваючою комою. Будемо вважати, що мантиси чисел зберігаються у пам'яті у прямих кодах.

1. *Зрівняння порядків.* На цьому етапі виконується віднімання порядків доданків з метою визначення доданку з меншим порядком. Отримана різниця порядків і її знак вказують, порядок якого доданку менше і на скільки розрядів необхідно зсунути його мантису праворуч. Далі у якості загального обирається більший порядок, а мантиса числа з меншим порядком зсувається вправо на різницю порядків. При цьому молодші розряди мантиси можуть втрачатися, вслід чого у доданок вноситься похибка.

2. *Додавання мантис.* Виконується додавання мантис доданків за правилами складання чисел із знаками в модифікованих обернеїх або доповнювальних кодах.

3. *Нормалізація результату.* Мантиса результату може опинитися відразу нормалізованою або подана з порушенням нормалізації вліво чи вправо. В двох останніх випадках проводиться нормалізація результату, тобто приведення мантиси до вигляду що відповідає виразу (1).

Якщо при додаванні мантис виникає переповнення розрядної сітки, то таке переповнення говорить о *порушенні нормалізації вліво*. Ознакою порушення нормалізації вліво є розбіжність у знакових розрядах модифікованих кодів чисел. Для приведення результату до нормалізованої форми необхідно зсунути мантису результату вправо на один розряд, а загальний порядок збільшити на одиницю.

Нормалізовані мантиси додатних чисел в розряді з вагою 2^{-1} завжди мають одиницю. Нормалізовані мантиси від'ємних чисел, що подані обернеім або доповнювальним кодом, мають в цьому розряді нуль. Збіжність цифр у знаковому і старшому розряді мантиси у додатних і від'ємних чисел і говорить о *порушенні нормалізації вправо*. Для приведення до нормалізованого вигляду мантиса отриманого результату зсувається вліво до появи одиниці або нуля (залежно від використовуємих кодів) у старшому розряді, а із загального порядку результату віднімається число одиниць, що дорівнює кількості зсувів мантиси.

Слід зазначити, якщо у результаті додавання отриманий нульовий результат, процес зсуву мантиси може виявитись нескінченним, тому кількість зсувів обмежується кількістю розрядів мантиси і результат операції подається *машичним нулем*.

4. *Формування результату*. На останньому етапі мантиса результату, що отримана у доповнювальному або обернеому коді подається у прямому коді. Далі до мантиси приписується загальний порядок доданків.

Вслід зсуву мантиси на етапі зрівняння порядків частина розрядів виходить з меж розрядної сітки і втрачається, тому після підсумовування може бути зроблено округлення результату, для цього до розряду, що вийшов із розрядної сітки останнім, додається одиниця з розповсюдженням переносив у старші розряди, що представляють результат у рамках розрядної сітки. Округлення може визвати порушення нормалізації вліво, тобто повторну нормалізацію.

Приклад.

Завдання. Виконати додавання чисел $Z = X + Y$, що подані у формі із плаваючою комою.

$$X = 0\ 1000\ 0\ 10111,$$

$$Y = 0\ 1010\ 0\ 11100.$$

Виконання завдання.

$$P_{(X)} = 0, 1000 \qquad M_{(X)} = 0, 10111$$

$$P_{(Y)} = 0, 1010 \qquad M_{(Y)} = 0, 11100$$

1. Зрівняння порядків.

Виконаємо віднімання порядків доданків у вигляді $\Delta = P_{(X)} + (-P_{(Y)})$ з використанням доповнювального коду.

$$\begin{array}{r} P_{(Y) \text{ [ДК]}} \quad 1, 0110 \\ + \\ P_{(X)} \quad \underline{0, 1000} \\ (P_{(X)} - P_{(Y)}) \text{ [ДК]} \quad 1, 1110 \end{array}$$

$$\begin{array}{r} (P_{(X)} - P_{(Y)}) \text{ [ПК]} \quad 1, 0010 \end{array}$$

Отримана в результаті від'ємна різниця порядків ($\Delta = -2_{(10)}$) вказує на те, що порядок числа Y більший за порядок числа X . Тому порядок цього числа обираємо як загальний порядок:

$$P_{(\text{заг})} = 0, 1010$$

Приводимо порядок числа X до загального порядку, для чого збільшуємо його на різниця порядків і зсуваємо мантису числа X на два розряди вправо:

$$P_{(\text{заг})} = 0, 1010 \quad M_{(X)} = 0, 0010111$$

2. Додавання мантис:

Мантиси є додатними числами, тому підсумовування виконуємо за правилами додавання звичайних додатних чисел, при чому застосовуємо модифікований код подання чисел:

$$\begin{array}{r} M_{(X)} \quad 00, 0010111 \\ + \\ M_{(Y)} \quad \underline{00, 1110000} \\ M_{(Z)} \quad 01, 0000111 \end{array}$$

3. Нормалізація результату.

Отримана мантиса результату $M_{(Z)}$ не є нормалізованою, оскільки присутня розбіжність у знакових розрядах. Це говорить про переповнення розрядної сітки, тобто відбулося порушення нормалізації вліво.

Для приведення результату до нормалізованої форми зсунемо мантису результату на один розряд вправо, збільшимо загальний порядок на одиницю.

$$\begin{array}{r} P'_{(\text{заг})} = P_{(\text{заг})} + 1 \\ P'_{(\text{заг})} = 0, 1011 \quad M'_{(Z)} = 0, 10000111 \end{array}$$

4. Формування результату.

Округляємо мантису результату, для чого до розряду що вийшов із розрядної сітки останнім додається одиниця:

$$\begin{array}{r|l} M'_{(Z)} & 0,10000111 \\ + & \\ \hline M_{(Z)} & 0,10001 \end{array}$$

До отриманої мантиси результату приписуємо загальний порядок:

$$P'_{(\text{заг})} = P_{(Z)}$$

$$P_{(Z)} = 0,1011 \qquad M_{(Z)} = 0,10001$$

Відповідь:

$$Z = 0\ 1011\ 0\ 10001.$$

Множення чисел із плаваючою комою

Множення двох чисел $X = 2^{P_x} M_x$ і $Y = 2^{P_y} M_y$, що задані у форматі із плаваючою комою, можна подати у вигляді

$$2^{P_z} M_z = 2^{P_x} M_x \cdot 2^{P_y} M_y = 2^{(P_x+P_y)} \cdot (M_x \cdot M_y)$$

При виконання операції множення мантис можна отримати результат із порушенням нормалізації вправо. Дійсно, якщо мантиси множників подані у нормалізованій формі

$$2^{-1} \leq M_x, M_y < 1,$$

то результат може знаходитися у границях

$$2^{-2} \leq M_z < 1.$$

Етапи множення чисел із плаваючою комою.

1. Одержання порядку результату у вигляді суми порядків множників

$$P_z = P_x + P_y.$$

2. Знаходження мантиси результату

$$M_z = M_x \cdot M_y.$$

3. Нормалізація результату, тобто приведення мантиси результату до вигляду

$$2^{-1} \leq M_z < 1.$$

Приклад.

Завдання. Подати числа у формі з плаваючою комою и знайти добуток $Z = X \cdot Y$, якщо:

$$X = 10,101,$$

$$Y = 0,1011.$$

Виконання завдання.

Приведемо задані числа до форми:

$$X = 2^{P_X} M_X = 2^2 \cdot 0,10101$$

$$Y = 2^{P_Y} M_Y = 2^0 \cdot 0,1011$$

Отримали порядки і мантиси:

$$P_{(X)} = 0, 10 \quad M_{(X)} = 0, 10101$$

$$P_{(Y)} = 0, 00 \quad M_{(Y)} = 0, 10110$$

Визначимо порядок результату, виконавши додавання порядків

множників $P_Z = P_X + P_Y = 2 + 0 = 2$. Після множення мантис отримаємо:

$$P_Z = 0, 10 \quad M_Z = 0, 0111001|1|1|0$$

Мантиса результату не є нормалізованою. Для виконання нормалізації мантису результату зсуваємо ліворуч на один розряд і виконуємо корекцію порядку $P_Z = P_Z - 1 = 1$. Отримаємо:

$$P_{(Z)} = 0, 01 \quad M_Z = 0, 1110011|1$$

Відповідь одержуємо після округлення мантиси:

$$Z = 0 \ 01 \ 0 \ 11101.$$

Ділення чисел із плаваючою комою

Результат ділення двох чисел $X = 2^{P_X} M_X$ і $Y = 2^{P_Y} M_Y$, що задані у форматі із плаваючою комою, можна записати у вигляді

$$2^{P_z} M_z = \frac{2^{P_x} M_x}{2^{P_y} M_y} = 2^{(P_x - P_y)} \cdot \frac{M_x}{M_y}$$

Ділення мантис повинно виконуватись при виконанні умови

$$M_x < M_y, \quad (2)$$

яка не завжди виконується при поданні мантис у нормалізованій формі. Тому перед началом ділення мантису діленого завжди зсувають вправо, чим забезпечують зменшення її у два рази. Відповідно до порядку діленого додається одиниця, тобто

$$2^{P_x} M_x = 2^{P_x+1} \cdot M_x \cdot 2^{-1}. \quad (3)$$

Після цього маємо

$$2^{-2} \leq M_x < 2^{-1},$$

$$2^{-1} \leq M_y < 1,$$

$$2^{-2} \leq M_z < 1.$$

Етапи ділення чисел із плаваючою комою.

1. Одержання порядку результату із урахуванням виконання умови (2) шляхом віднімання

$$P_z = P_x - P_y.$$

2. Одержання мантиси результату. На цьому етапі виконується ділення мантис:

$$M_z = \frac{M_x}{M_y}.$$

3. Нормалізація результату. Виконується аналогічно нормалізації при множенні чисел із плаваючою комою.

Приклад.

Завдання. Виконати ділення чисел $Z = \frac{X}{Y}$, що подані у формі із плаваючою комою.

$$X = 0\ 10\ 0\ 1011,$$

$$Y = 0\ 01\ 0\ 1001.$$

Виконання завдання.

Маємо ділене і дільник, задані у формі:

$$P_x = 0, 10 \quad M_x = 0, 1011$$

$$P_y = 0, 01 \quad M_y = 0, 1001$$

Застосуємо перетворення (3) для забезпечення умови (2). Отримаємо змінену мантису і порядок діленого X :

$$P_X = 0, 11 \quad M_{(X)} = 0, 01011$$

Визначимо порядок результату.

$$P_Z = P_X - P_Y = 3 - 1 = 2, \text{ тобто}$$

$$P_Z = 0, 10$$

Визначимо мантису результату діленням мантис і одержуємо результат:

$$M_Z = 0, 1100 \overline{1}$$

Отриману в результаті мантису частки округляємо і усікаємо до розмірів розрядної сітки:

$$M_Z = 0, 1101 \overline{1}$$

Мантиса результату є нормалізованою, тому сформуємо результат обчислення:

$$P_Z = 0, 10 \quad M_Z = 0, 1100$$

Відповідь:

$$Z = 0 \ 01 \ 0 \ 1100.$$

ПРАКТИЧНА РОБОТА №5

Підготовка до роботи

Перевести номер варіанту (номер групи і номер студента у списку) в двійкову систему. Записати два 10-розрядних двійкових числа:

$$X = -x_7x_61x_5x_40,x_31x_2x_1 \quad \text{і} \quad Y = +x_91x_8x_7,x_6x_5x_4x_3x_2x_1,$$

де x_i - двійкові цифри варіанту у двійковій системі числення (x_1 - молодший розряд).

Таблиця 5.1. Визначення варіанту

x_2, x_1	Спосіб множення	x_3	Спосіб ділення	x_5, x_4	Машинні коди в АЛУ
0 0	1-й, 4-й	0	1-й	0 0	Додавання в ОК
0 1	2-й, 3-й	1	2-й	0 1	Додавання в ДК
1 0	1-й, 3-й			1 0	Віднімання в ОК
1 1	2-й, 4-й			1 1	Віднімання в ДК

Завдання

1. Числа X і Y в прямому коді записати у формі з плаваючою комою у класичному варіанті (з незміщеним порядком і повною мантисою). Вважати, що у пам'ятті комп'ютера числа зберігаються у прямому коді з одним знаковим розрядом. На мантису відвести 7 розрядів (з урахуванням знакового розряду). На порядок кількість розрядів вибрати самостійно з урахуванням можливого розширення розрядності порядків при виконанні послідовності операцій.

Записати числа X і Y також за стандартом ANSI/IEEE 754-2008 в короткому 32-розрядному форматі).

2. Виконати 4 операції (Табл. 5.1) з числами, що подані з плаваючою комою в класичному варіанті (два з чотирьох способів множення, один з двох способів ділення та додавання/віднімання). Операндами для способу множення є задані числа X та Y . Для кожної наступної операції першим операндом є результат попередньої операції, а другим операндом завжди є

число Y . (Наприклад, для ділення першим операндом є результат множення, для операції додавання/віднімання першим операндом є результат ділення).

Вихідні операнди з пам'яті приймаються в прямому коді з одним знаковим розрядом. Модифікований код формується при необхідності в АЛУ.

Для обробки мантис кожної операції, подати:

- 2.1 теоретичне обґрунтування способу;
- 2.2 операційну схему;
- 2.3 змістовний (функціональний) мікроалгоритм;
- 2.4 таблицю станів регістрів (лічильника), довжина яких забезпечує одержання 6 основних розрядів мантиси результату;
- 2.5 функціональну схему з відображенням управляючих сигналів, входів для запису операндів при ініціалізації пристрою і схем формування внутрішніх логічних умов;
- 2.6 обробку порядків (показати у довільній формі);
- 2.7 форми запису нормалізованого результату з плаваючою комою в пам'ять комп'ютера в прямому коді для класичного формату.

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 (AFDK) побудувати і налагодити розроблені пристрої.

2. В асинхронному режимі виконати числові приклади із заданими значеннями операндів. Керуючі сигнали подавати на операційну схему в ручному режимі.

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, завдання, порядок виконання роботи, всі схеми, формули і таблиці, отримані при виконанні теоретичного завдання і в процесі

моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Література

1. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів: Навч. посібник. – К.: Вид-во НАУ, 2009. – 364 с.
2. Жабін В.І., Жуков І.А., Клименко І.А., Стіренко С.Г. Арифметичні та управляючі пристрої цифрових ЕОМ. – К.: ВЕК+, 2008. – 176 с.
3. Матвієнко М. П. Комп'ютерна логіка : підручник / М. П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

Контрольні запитання

1. Як перейти від функціонального (змістовного) мікроалгоритму до структурного мікроалгоритму?
2. Поясніть правила додавання та віднімання чисел в ОК і ДК.
3. Поясніть правила зсуву кодів в ОК і ДК.
4. Як можна виявити переповнення розрядної сітки при виконанні операцій з машинними кодами?
5. Яким чином можна подати мікрооперації і мікроалгоритми?
6. Форми подання чисел у ЕОМ.
7. Виконання операцій з числами в форматі з фіксованою комою.
8. Етапи додавання чисел з плаваючою комою.
9. Дайте означення мікрооперації та мікроалгоритму?
10. Охарактеризуйте основні етапи проектування пристрою для виконання операції.
11. Як визначити необхідну тривалість керуючих сигналів?
12. Як перейти від операційної схеми до функціональної?
13. Як розрахувати розрядність вузлів операційного пристрою?
14. Особливості формату чисел за стандартом ANSI/IEEE 754-2008.
15. Коли необхідно виконувати округлення результату операції, як це зробити?

6. ЗАНЯТТЯ №6

ДОСЛІДЖЕННЯ ОПЕРАЦІЙ ДОДАВАННЯ ТА ВІДНІМАННЯ В ДВІЙКОВО-КОДОВАНИХ СИСТЕМАХ ЧИСЛЕННЯ

Ціль роботи: Оволодіння способами подання десяткових чисел зі знаками. Дослідження операції додавання та віднімання чисел в системах числення з двійково-кодованим поданням цифр.

Теоретичні відомості

Додавання чисел в двійково-кодованих системах числення.

В системах числення з основою $k > 2$ та цифрами $x_i \in \{0, 1, \dots, k-1\}$ для подання цифр використовують двійкову систему числення. Для представлення однієї цифри необхідно мати не менше ніж $m = \lceil \log_2(k-1) \rceil$ двійкових розрядів ($\lceil \cdot \rceil$ – функція округлення числа до найближчого цілого). Наприклад, для десяткової системи числення цифри кодуються не менш ніж чотирма двійковими розрядами (тетрадами), хоча двійково-десяткові коди (ДДК) можуть мати і більше розрядів, якщо це дає переваги при виконанні певних операцій (табл. 6.1)

Таблиця 6.1. Двійково-десяткові коди

Десяткова цифра	Двійково-десятковий код						
	8421	2421	5421	7421	8421+3	2 із 5	Томсона
0	0000	0000	0000	0000	0011	11000	00000
1	0001	0001	0001	0001	0100	00011	10000
2	0010	0010	0010	0010	0101	00101	11000
3	0011	0011	0011	0011	0110	00110	11100
4	0100	0100	0100	0100	0111	01001	11110
5	0101	1011	1000	0101	1000	01010	11111
6	0110	1100	1001	0110	1001	01100	01111
7	0111	1101	1010	1000	1010	10001	00111
8	1000	1110	1011	1001	1011	10010	00011
9	1001	1111	1100	1010	1100	10100	00001

Наприклад в коді 2 із 5 спрощується дешифрування кодів цифр, а також визначення помилок (зміна будь-якого одного розряду дає заборонену комбінацію двійкових розрядів). Для коду Томсона лічильник будується на

реєстрі зсуву з інверсним циклічним ланцюжком, що прискорює мікрооперацію інкремент порівняно з лічильниками, що мають міжрозрядні логічні схеми. Крім того, це також спрощує дешифрування десяткових цифр. Для арифметичних операцій вказані коди не застосовують, бо вони не є адитивними і зваженими.

Адитивними є коди для яких сума кодів двох чисел представляє код суми. В *зважених кодах* кожний розряд в тетраді має постійну вагу.

ДДК з вагами розрядів 8421 називають кодом прямого заміщення. Завдяки адитивності та зваженості зручність цього коду проявляються при машинному переводі з десяткової системи в двійкову і назад, а також при підсумовуванні на звичайних двійкових суматорах завдяки його адитивності (сума кодів двох чисел представляє код суми).

Коди 2421, 5421 і 7421 не є адитивними. Код з вагами 2421 дозволяє простим інвертуванням розрядів одержувати обернений, а при ще й додаванні одиниці – доповняльний код числа. Це зручно для виконання алгебраїчних операцій. В кодах 5421 і 7421 ряд комбінацій має менше одиниць ніж код 8421. Це може зменшити споживання енергії в динамічній елементній базі.

ДДК з надлишком 3 не є зваженим, що ускладнює перетворення чисел в системи з різною основою, але він дозволяє автоматично формувати перенос в старшу тетраду при підсумуванні чисел на звичайних суматорах.

Способи додавання чисел з основою $k = 10$ на базі двійкових суматорів.

Алгебраїчні операції з десятковими числами, як і з двійковими, виконуються на суматорах з використанням обернених або доповняльних кодів. Для одержання оберненого коду необхідно кожен розряд числа замінити на цифру, що є доповненням до 9. Наприклад, 8 замінюють на 1, 3 – на 6, 5 –

на 4 і т.і. Додавання одиниці в молодший розряд переводить обернений код в доповняльний. Додатні числа в знакових розрядах мають 0, а від’ємні – 1.

Наприклад, операції $542+223=765$, $542-223=319$, $223-542=-319$ в доповняльних кодах виконуються наступним чином:

0.542	0.542	0.223
<u>+0.223</u>	<u>+1.777</u>	<u>+1.458</u>
0.765	0.319	1.681
Результат=+765	Результат=+319	Результат=-319

Віднімання $Z = X - Y$ замінюється на додавання $Z = X + (-Y)$, що дозволяє виконувати операції на суматорах. Використання різних адитивних ДДК для подання десяткових цифр потребує корекції при підсумуванні розрядів. В коді 8421 корекція в десяткових розрядах виконується, якщо після підсумування тетрад виникає перенос в старшу тетраду або результат в тетраді стає більше 9. Необхідна корекція в тетрадах «+6», що пояснюється наступним. Одиниця, що залишає тетраду, повинна забрати з неї $k = 10$ одиниць, а фактично забирає 16, тобто подвійну вагу старшого двійкового розряду тетради (для коду 8421 ця вага дорівнює 8). Тому для компенсації до тетради додається число +6 (0110_2) з розповсюдженням переносу в старшу тетраду. На рис. 6.1 показано варіанти корекції при додаванні 2-розрядних двійково-десяткових чисел.

$$\begin{array}{r}
 38_{2-10}=0011\ 1000 \\
 +16_{2-10}=0001\ 0110 \\
 \hline
 0100\ 1110 \\
 \text{Корекція } +0000\ 0110 \\
 \hline
 \text{а) } 54_{2-10}=0101\ 0100
 \end{array}
 \qquad
 \begin{array}{r}
 29_{2-10}=0010\ 1001 \\
 +58_{2-10}=0101\ 1000 \\
 \hline
 1000\ 0001 \\
 \text{Корекція } +0000\ 0110 \\
 \hline
 \text{б) } 87_{2-10}=1000\ 0111
 \end{array}$$

Рис. 6.1. Корекція результату: а – заборонена цифра 14 (1110_2) в молодшому розряді; б – є перенос в старшу тетраду

Двійково-десяткові суматори.

Двійково-десяткові суматори в кожному десятковому розряді повинні реалізувати п'ять перемикальних функцій (рис. 6.2). Чотири з них відповідають двійково-кодованій десятковій сумі $S = s_4s_3s_2s_1$ і одна - переносу в старший десятковий розряд P_4 . Ці функції залежать від десятичних цифр доданків $X = x_4x_3x_2x_1$ і $Y = y_4y_3y_2y_1$, а також переносу з молодшої тетради P_0 , тобто від 9 аргументів. Нормальні форми функцій дуже громіздкі і погано мінімізуються. Тому підсумування ДДК доцільно виконувати відповідно до схеми на рис. 6.2.

На першому етапі на двійковому суматорі підсумовують ДДК десятичних цифр за правилами двійковій арифметики. Потім на другому етапі за допомогою ще одного суматора роблять корекцію отриманого результату шляхом додавання або віднімання деякої константи, що визначається комбінаційною схемою КС, а також виділяють десятковий перенос в старшу тетраду. ДДК повинен мати властивості адитивності.

Такою властивістю володіють ДДК 8421 і $8421+\Delta$, де Δ ціле число, що назване надлишком. Якщо використовується ДДК, що не володіє

властивостями адитивності, то цифри доданків слід перед підсумовуванням перетворити в адитивний ДДК.

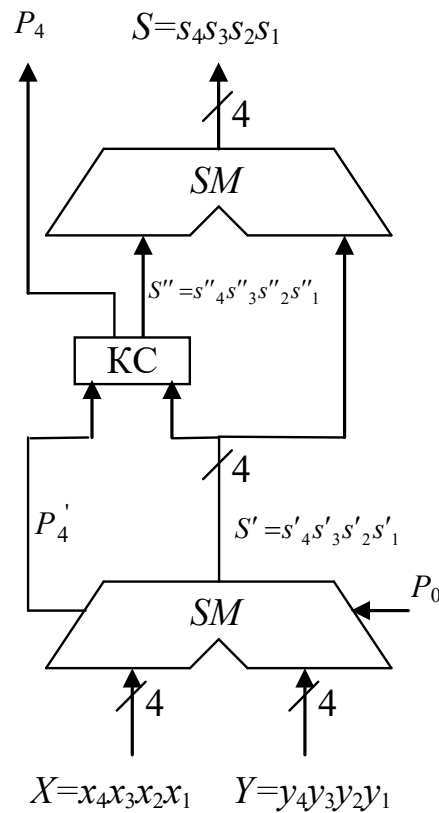


Рис. 6.2. Структура одного розряду десятичного суматора

Схема корекції і виділення переносу може бути визначена шляхом порівняння P'_4 і $S' = s'_4 s'_3 s'_2 s'_1$, отриманих при підсумовуванні цифр доданків, і необхідного результату, тобто P_4 і $S = s_4 s_3 s_2 s_1$. Нехай в якості ДДК використовується код 8421. Тоді стани вихідних сигналів за схемою на рис. 4.2 можна описати за допомогою табл. 4.2, де Σ_d – сума в десятичному вигляді.

З таблиці 6.2 видно, що в залежності від суми, отриманої на першому етапі, корекція результату для ДДК 8421 складається в додаванні 0 або 6. Вважаючи функції $P''_4, s''_4, s''_3, s''_2, s''_1$ частково визначеними функціями 5-ти аргументів $P'_4, s'_4, s'_3, s'_2, s'_1$, можна після мінімізації знайти:
 $s''_4 = s''_1 = 0; s''_3 = s''_2 = P''_4 = P'_4 \vee s'_4 \cdot s'_2 \vee s'_4 \cdot s'_3$.

Знайдені функції у якості одного з доданків подаються на суматор другого ярусу, на виходах якого формується правильний результат, тобто

десятьова цифра. З десятикових однорозрядних суматорів, показаних на рис. 6.1, складається суматор на необхідну кількість розрядів. Для цього вихідний перенос i -го розряду подається на вхідний перенос $i + 1$ -го розряду.

Таблиця 6.2. Таблиця істинності комбінаційного двійково-десятькового суматора

Σ_d	Сума до корекції					Сума після корекції					Код корекція			
	P_4	S_4	S_3	S_2	S_1	P_4	S_4	S_3	S_2	S_1	S''_4	S''_3	S''_2	S''_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
2	0	0	0	1	0	0	0	0	1	0	0	0	0	0
3	0	0	0	1	1	0	0	0	1	1	0	0	0	0
4	0	0	1	0	0	0	0	1	0	0	0	0	0	0
5	0	0	1	0	1	0	0	1	0	1	0	0	0	0
6	0	0	1	1	0	0	0	1	1	0	0	0	0	0
7	0	0	1	1	1	0	0	1	1	1	0	0	0	0
8	0	1	0	0	0	0	1	0	0	0	0	0	0	0
9	0	1	0	0	1	0	1	0	0	1	0	0	0	0
10	0	1	0	1	0	1	0	0	0	0	0	1	1	0
11	0	1	0	1	1	1	0	0	0	1	0	1	1	0
12	0	1	1	0	0	1	0	0	1	0	0	1	1	0
13	0	1	1	0	1	1	0	0	1	1	0	1	1	0
14	0	1	1	1	0	1	0	1	0	0	0	1	1	0
15	0	1	1	1	1	1	0	1	0	1	0	1	1	0
16	1	0	0	0	0	1	0	1	1	0	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1	0	1	1	0
18	1	0	0	1	0	1	1	0	0	0	0	1	1	0
19	1	0	0	1	1	1	1	0	0	1	0	1	1	0

Знайдені функції у якості одного з доданків подаються на суматор другого ярусу, на виходах якого формується правильний результат, тобто десятикова цифра. З десятикових однорозрядних суматорів, показаних на рис. 6.2, складається суматор на необхідну кількість розрядів. Для цього вихідний перенос i -го розряду подається на вхідний перенос $i + 1$ -го розряду.

ПРАКТИЧНА РОБОТА №6

Підготовка до роботи

1. Для визначення варіанту завдання необхідно перевести повний десятковий номер варіанту (номер групи і номер студента у списку) в двійкову систему числення і виділити сім молодших розрядів $a_7, a_6, a_5, a_4, a_3, a_2, a_1$.

2. Відповідно з вихідними даними (табл. 6.3) подати таблицю кодування десяткових цифр $0, 1, \dots, 9$ в ДДК і таблицю істинності однорозрядного комбінаційного двійково-десятькового суматора з використанням заданого ДДК (по аналогії з табл. 6.2). Якщо ДДК не має властивості адитивності, необхідно синтезувати схеми перетворювачів заданого ДДК в адитивний код (наприклад, код 5421 в код 8421).

3. На основі одержаної таблиці істинності виконати синтез комбінаційної схеми формування корекції (див. КС на рис. 6.2). На двійкових суматорах та заданих логічних елементах (табл. 6.3) побудувати функціональну схему одного розряду двійково-десятькового суматора.

4. Подати один розряд синтезованого суматора у вигляді прямокутника і побудувати на їх основі структурну схему 4-розрядного десяткового суматора. Записати приклади додавання $Z = X + Y$ і віднімання у формі $Z = X + (-Y)$ та $Z = Y + (-X)$ 4-розрядних десяткових чисел на розробленому суматорі. Показати необхідну корекцію в розрядах. Для подання чисел використати доповняльний код. Значення операндів подано в табл. 4.3.

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити розроблений однорозрядний суматор.

2. Виконати по два приклади додавання цифр, що потребують і не потребують корекції результату в тетрадї. Відмітити результати у звіті.

3. В режимі синхронного моделювання визначити максимальний час підсумування розрядів.

Таблиця 6.3. Вихідні дані для побудови суматора

$a_7a_6a_5$	ДДК	$a_4a_3a_2$	Логічні елементи	$a_3a_2a_1$	Операнди	
					X	Y
000	8421+1	000	2І, 2АБО, НЕ	000	3174	-3442
001	8421+2	001	2І, 3АБО, НЕ	001	-3427	3626
010	8421+3	010	3І, 2АБО, НЕ	010	-5473	3672
011	8421+4	011	3І, 3АБО, НЕ	011	1234	-7333
100	8421+5	100	2АБО-НЕ	100	-5136	4437
101	2421	101	2І-НЕ	101	-4512	1312
110	5421	110	3АБО-НЕ	110	5484	-3445
111	7421	111	3І-НЕ	111	3342	-3912

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, порядок виконання роботи, всі схеми, формули і таблиці, отримані при виконанні теоретичного завдання і в процесі моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Контрольні запитання

1. В якій формі у ЕОМ подаються десяткові числа?
2. Як визначити кількість розрядів двійкового і двійково-десятькового числа з однаковим кількісним еквівалентом?
3. Поясніть, коли при додаванні чисел необхідна корекція результату.

4. Як визначити необхідну корекцію при додаванні двійково-десяткових чисел.
5. Наведіть склад апаратури для побудови двійково-десятькового суматора.
6. Яким вимогам повинні задовольняти ДДК, що використовуються у суматорі ?
7. У чому сутність властивості адитивності ДДК і до чого може привести відсутність такої властивості у ДДК?
8. У чому сутність властивості зваженості ДДК і до чого може привести відсутність такої властивості у ДДК?
9. Наведіть приклади ДДК, що володіють і не володіють властивістю адитивності.
10. Наведіть приклади ДДК, що володіють і не володіють властивістю зваженості.
11. Наведіть приклад додавання 4-розрядних двійково-десятькових чисел із знаками в заданому ДДК.

Литература

1. Жабін В.І., Арифметичні та управляючі пристрої цифрових ЕОМ: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, С.Г. Стіренко. – К.: ВЕК+, 2008. – 176 с.
2. Жабін В.І. Прикладна теорія теорія цифрових автоматів: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, В.В. Ткаченко. – К.: Вид-во НАУ, 2009. – 364 с.
3. Матвієнко М. П. Комп'ютерна логіка : підручник / М. П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

7. ЗАНЯТТЯ №7

ДОСЛІДЖЕННЯ ОПЕРАЦІЙ ПЕРЕВОДУ ЧИСЕЛ МІЖ СИСТЕМАМИ ЧИСЛЕННЯ З РІЗНОЮ ОСНОВОЮ

Ціль роботи: Дослідити машинні методи перевodu чисел між двійковою та двійково-кодованими системами числення.

Теоретичні відомості

Подання чисел в двійково-кодованих системах числення.

Для подання цифр десяткової системи числення застосовують двійково-десяткові коди (ДДК). Найбільш поширеним є ДДК прямого заміщення (табл. 5.1). Цифри кодуються чотирма двійковими розрядами (тетрадами), які мають відповідно вагу 8, 4, 2 і 1, починаючи зі старших розрядів (зазвичай код називають ДДК або (2-10)-код 8421).

Таблиця 7.1. Двійково-десятковий код прямого заміщення 8421

Десяткова цифра	0	1	2	3	4	5	6	7	8	9
ДДК	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Код прямого заміщення може бути застосовано для подання чисел в інших k -ічних системах. В системах числення з основою $k > 2$ та цифрами $x_i \in \{0, 1, \dots, k-1\}$ для подання цифр необхідно мати не менше ніж $m = \lceil \log_2(k-1) \rceil$ двійкових розрядів ($\lceil \cdot \rceil$ – функція округлення числа до найближчого цілого).

В лабораторній роботі розглядаються машинні методи перевodu чисел між двійковою системою та k -ічними системами з парною основою і цифрами з множини $\{0, 1, \dots, k-1\}$, для кодування в яких використовують код прямого заміщення. Наприклад, при $k = 6$ цифри належать множині $\{0, 1, \dots, 5\}$ і кодуються двійковими розрядами $\{000, 001, \dots, 101\}$. Аналогічно, при $k = 12$

цифри належать множині $\{0,1,\dots,8,9,A,B\}$ і кодуються двійковими розрядами $\{0000,0001,\dots,1011\}$.

Коди прямого заміщення відносяться до адитивних кодів. *Адитивними* є коди для яких сума кодів двох чисел завжди представляється одним кодом суми. Дана властивість суттєво спрощує перетворення чисел між різними системами числення апаратними засобами.

Машинні методи перетворення чисел в різні системи числення.

Розглянемо перетворення чисел між системами числення методом «зсуву-корекції» на прикладі переводу чисел з двійково-десятькової системи числення в двійкову і навпаки при використанні ДДК 8421.

Перехід від двійково-десятькової системи числення до двійкової відбувається шляхом ділення, а зворотний перехід – шляхом множення чисел на два. В двійковій системі числення ці операції зводяться відповідно до звичайного зсуву на один розряд вправо або вліво. В двійково-десятьковій системі, окрім зсуву, може бути необхідна корекція кодів в тетрадах.

Перетворення чисел із двійково-десятькової системи числення в двійкову методом «зсуву-корекції».

Алгоритм перетворення n -розрядних двійково-десятькових чисел в код 8421 у двійкову систему числення з природним порядком ваг розрядів складається у наступному.

1. Коди всіх N тетрад і двійкове число зсуваються праворуч з переходом двійкових розрядів із кожної старшої тетради в сусідню, а із наймолодшої тетради в розряди двійкового числа.

2. Якщо із старшої тетради в молодшу перейшов нуль, то корекція не потрібна. При переході в молодшу тетраду одиниці в цій тетраді виконується корекція, яка полягає у відніманні з коду тетради трійки.

3. Цифри двійкової системи формуються у процесі зсуву на виході молодшої тетради, починаючи із молодших двійкових розрядів. Перетворення закінчується, коли у всіх тетрадах двійково-десятькового числа будуть отримані нулі.

Значення корекції «-3» пояснюється наступним. Одиниця в тетраді двійково-десятькового числа дорівнює 10 одиницям сусідньої молодшої тетри, тобто основі системи числення k . Коли при зсуві вправо (діленні на два) із старшої тетри в молодшу тетраду переходить одиниця, то вона повинна перенести половину ваги свого десяткового розряду, що складає п'ять одиниць для молодшої тетри ($k/2=10/2=5$). Фактично одиниця потрапляє в двійковий розряд молодшої тетри, який має вагу 8 (вага старшого розряду коду 8421). Звідси корекція дорівнює «-3». Віднімання замінюють додаванням доповняльного коду від'ємного числа три (1101) без розповсюдження переносу в старшу тетраду. Операційна схема пристрою приведена на рис. 7.1.

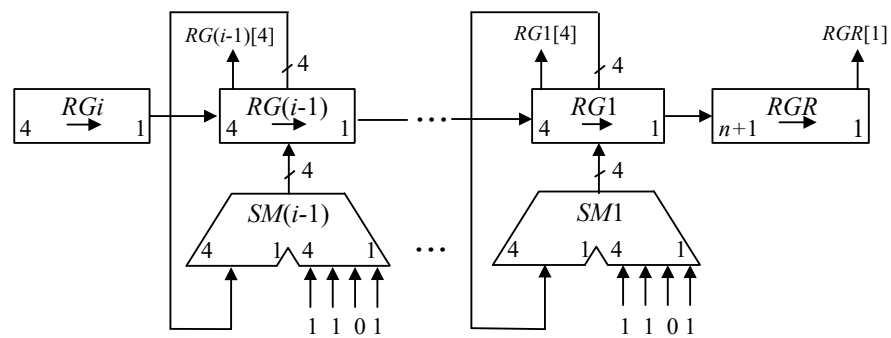


Рис. 7.1. Пристрій перетворення двійково-десятькового коду в двійковий

В кожному циклі перевіряється старший розряд регістра тетри $RG_i[4]$. Якщо він дорівнює 1, то до тетри додається 1101, інакше додавання в циклі не виконується. Кінець операції визначається за маркерною одиницею, коли вона потрапляє в розряд $RGR[1]$.

Цифрове моделювання роботи пристрою за допомогою цифрової діаграми для значення аргументу $A=125_{10}=0001\ 0010\ 0101_{2-10}$ представлено в табл. 5.2.

Перетворення чисел з двійкової системи числення в двійково-десятькову методом «зсуву-корекції».

Алгоритм перетворення двійкових чисел в двійково-десятькові числа в кодї 8421 складається у наступному.

1. Коди всіх N тетрад зсуваються ліворуч на один розряд з переходом двійкових розрядів із кожної молодшої тетради в сусідню старшу тетраду. Розряди двійкового числа переходять у наймолодшу тетраду.

2. Коли із тетради у сусідню старшу переходить одиниця або код в тетраді стає більше за 9 (максимальну цифру $k - 1$ при основі k), то потрібна корекція, яка полягає у додаванні до коду тетради числа 6 з розповсюдженням переносу.

3. Цифри двійково-десятькової системи формуються у процесі зсуву в тетрадах, починаючи із старших розрядів. Перетворення закінчується, коли всі двійкові розряди (включаючи нулі) переходять у двійково-десятькові тетради.

Таблиця 7.2. Стани регістрів пристрою перетворення чисел

№ такт	Двійково-десятькове число			Двійкове Число	Мікрооперація
	$RG1$ 4 1	$RG2$ 4 1	$RG3$ 4 1	$RG4$ 12 0	
ПС	0001	0010	0101	1	
1	0000	1001 + 1101 0110	0010	11	$(RG1, RG2, RG3, RG4) \rightarrow$ $RG2 := RG2 + 1101$ Корекція (-3)
2	0000	0011	0001	011	$(RG1, RG2, RG3, RG4) \rightarrow$ $CT := CT - 1$
3	0000	0001	1000 + 1101 0101	1011	$(RG1, RG2, RG3, RG4) \rightarrow$ $RG3 := RG3 + 1101$ Корекція (-3)
4	0000	0000	1010 + 1101 0111	11011	$(RG1, RG2, RG3, RG4) \rightarrow$ $RG3 := RG3 + 1101$ Корекція (-3)
5	0000	0000	0011	111011	$(RG1, RG2, RG3, RG4) \rightarrow$
6	0000	0000	0001	1111011	$(RG1, RG2, RG3, RG4) \rightarrow$
7	0000	0000	0000	11111011	$(RG1, RG2, RG3, RG4) \rightarrow$
8	0000	0000	0000	011111011	$(RG1, RG2, RG3, RG4) \rightarrow$
9	0000	0000	0000	0011111011	$(RG1, RG2, RG3, RG4) \rightarrow$
10	0000	0000	0000	00011111011	$(RG1, RG2, RG3, RG4) \rightarrow$
11	0000	0000	0000	000011111011	$(RG1, RG2, RG3, RG4) \rightarrow$
12	0000	0000	0000	0000011111011	$(RG1, RG2, RG3, RG4) \rightarrow$
Результат: $A = 1111101_2$; 1 – маркерна одиниця					

Необхідність корекції (+6) в тетрадах пояснюється наступним. Одиниця, що залишає тетраду, повинна забрати з неї $k = 10$ одиниць, а фактично забирає 16, тобто подвійну вагу старшого двійкового розряду тетради (для коду 8421 ця вага дорівнює 8). Тому для компенсації до тетради додається число 6 (0110). Операційна схема пристрою показана на рис. 7.2.

На початку операції в молодшому розряді двійкового регістру $RGR[0]$ знаходиться маркерна одиниця. Кінець операції визначається по нульовому вмісту n молодших розрядів цього регістру. Моделювання роботи пристрою подано в табл. 7.3.

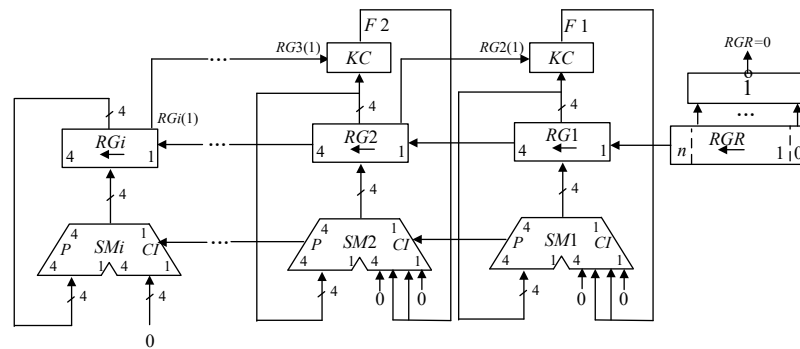


Рис.7.2. Пристрій перетворення двійкового коду в двійково-десятковий

Розглянуті методи переводу чисел в різні системи числення потребують використання адитивних кодів прямого заміщення. Більша за основою системи повинна мати парну основу $k > 2$, наприклад, 4, 6, 8, 10, 12, 14 ... Для систем з основами $k = 2^i$ корекція після зсуву кодів не потрібна.

Для інших систем корекція знаходиться аналогічно системі з основою $k = 10$. Оцінюється вага розряду, що потрапляє в k -ічний розряд або залишає цей розряд при зсуві і реальна кількість одиниць, яка повинна перейти або покинути k -ічний розряд.

Таблиця 7.3. Стани регістрів пристрою перетворення чисел

№ такт	Двійково-десятькове число			Двійкове число	Кінець операції	Мікрооперація
	RG1	RG2	RG3	RG4		
ПС	0000	0000	0000	11110100101	0	
1	0000	0000	0001	1110100101	0	(RG1, RG2, RG3, RG4) ←
2	0000	0000	0011	110100101	0	(RG1, RG2, RG3, RG4) ←
3	0000	0000	0111	10100101	0	(RG1, RG2, RG3, RG4) ←
4	0000	0000 + 0000 0001	1111 + 0110 0101	0100101	0	(RG1, RG2, RG3, RG4) ← RG3:= RG3+ 0110 Корекція (+6) Поширення переносу
	0000	0010 + 0000 0001	1010 + 0110 0000	100101	0	(RG1, RG2, RG3, RG4) ← RG3:= RG3+0110 Корекція (+6) Поширення переносу
6	0000	0110	0001	00101	0	(RG1, RG2, RG3, RG4) ←
7	0000 + 0000 0001	1100 + 0110 0010	0010	0101	0	(RG1, RG2, RG3, RG4) ← RG2:= RG2+ 0110 Корекція (+6) Поширення переносу
	0010	0100	0100	101	0	(RG1, RG2, RG3, RG4) ←
	0100	1000	1000	01	0	(RG1, RG2, RG3, RG4) ←
	1001	0001 + 0110 0111	0000 + 0110 1000	10000000000	1	RG2:= RG2+ RG5 RG2:= RG2+ RG5 Корекція (+6)
Результат: $A=1001\ 0111\ 0010_{2-10} = 972_{10}$; 1 – маркерна одиниця						

ПРАКТИЧНА РОБОТА №7

Підготовка до роботи

1. Для визначення варіанту завдання необхідно перевести повний десятковий номер варіанту (номер групи і номер студента у списку) в двійкову систему числення і виділити сім молодших розрядів $a_6, a_5, a_4, a_3, a_2, a_1$. Вихідні дані для виконання завдання подані в табл. 5.4.

2. Виходячи з основи системи k і розрядності k -ічного числа визначити максимальну розрядність двійкового числа. Розрядність k -ічного числа задана в табл. 5.4 безпосередньо (наприклад, 345_6 – три розряди, $9A_{18}$ – два розряди). Виходячи зі значення заданого k -ічного числа визначити значення двійкового числа, що має такий самий кількісний еквівалент (наприклад, $135_6=111011_2$; $126_{10}=1111110_2$). Одержані числа використати для цифрового моделювання пристроїв перетворення чисел у вигляді таблиці станів регістрів.

3. Визначити кількість двійкових розрядів для кодування однієї k -ічної цифри за формулою $m = \lceil \log_2(k-1) \rceil$. Визначити вагу кожного двійкового розряду в коді прямого заміщення (наприклад, $[16]8421$ для основи $k=18$, 8421 для основи $k=12$, 421 для основи $k=6$). Визначити множину цифр та їх позначення для k -ічної системи (наприклад, для $k=14$ маємо $x_i \in \{0,1,2,3,4,5,6,7,8,9,A,B,C,D\}$). Подати таблицю цифр в коді прямого заміщення, наприклад, для $k=14$ коди змінюються від 0000_2 до $1101_2=13_{10}$; для $k=18$ коди змінюються від 00000_2 до $10001_2=17_{10}$.

3. Для заданої операції перетворення чисел на базі операційної схеми (рис. 7.1 або рис. 7.2) розробити змістовний мікроалгоритм і побудувати таблицю станів регістрів в кожному такті виконання операції.

4. Побудувати функціональну схему з позначенням розрядності регістрів і сигналів управління. Подати структурний мікроалгоритм, в якому змістовні мікрооперації замінені на управляючі сигнали, що забезпечують їх виконання (наприклад, W – запис даних в регістр; SL – зсув даних в регістрі ліворуч; SR – зсув праворуч).

5. Подати таблицю станів регістрів при зворотному переводу числа, одержаного в п. 3, у вихідну систему числення. Наприклад, якщо в п. 3 виконується перетворення двійкового числа в двійково-десятькове, то зробити зворотний перехід в двійково-десятькову систему.

Таблиця 7.4. Вихідні дані для побудови пристрою

$a_5a_4a_3a_2$	Значення максимального k -ічного числа				Основа k системи числення	Операція
	a_6a_1					
	00	01	10	11		
0000	541	551	553	533	6	Перетворення з двійкової системи числення в двійково-кодовану з основою k
0001	453	443	425	435		
0010	342	342	412	442		
0011	452	454	451	453		
0100	97B	77B	4A7	5A7	12	
0101	29A	39A	6A3	5A3		
0110	7B0	8B0	749	739		
0111	615	685	615	635		

1000	54A	74A	A51	A55	14	Перетворення з двійково-кодованої системи числення з основою k в двійкову систему
1001	699	6A9	62C	64C		
1010	31B	3BB	A12	A32		
1011	4D7	4D8	4BC	2BC		
1100	4E	7E	5A	7A	18	
1101	6A	CA	F9	F9		
1110	4F	7F	7B	5B		
1111	6G	8G	DG	DA		

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити розроблений пристрій.
2. В асинхронному режимі виконати числовий приклад із заданими значеннями операндів.
3. В режимі синхронного моделювання визначити максимальний час виконання операції.

Зміст звіту

Звіт повинний включати назву роботи, мету роботи, підготовку до практичної роботи, порядок виконання роботи, всі схеми, формули і таблиці, отримані при виконанні теоретичного завдання і в процесі моделювання схем, відповіді на контрольні запитання. Сформулювати висновки по роботі.

Контрольні запитання

1. В якій формі у ЕОМ подаються десяткові числа?
2. Як визначити кількість розрядів двійкового і двійково-десятькового числа з однаковим кількісним еквівалентом?
3. Поясніть, коли при перетворенні чисел необхідна корекція результату.
4. Як визначити необхідну корекцію при перетворенні чисел для заданої основи системи числення.
5. Наведіть склад апаратури для побудови пристрою перетворення чисел в різні системи числення.

6. Яким вимогам повинні задовольняти коди, що використовуються у пристроях перетворення чисел?

7. У чому сутність властивості адитивності кодів і до чого може привести відсутність такої властивості?

8. Наведіть приклади кодів, що володіють і не володіють властивістю адитивності.

9. Наведіть приклад переводу 3-розрядних двійково-десяткових чисел в двійкові і навпаки.

Література

1. Жабін В.І., Арифметичні та управляючі пристрої цифрових ЕОМ: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, С.Г. Стіренко. – К.: ВЕК+, 2008. – 176 с.
2. Жабін В.І. Прикладна теорія теорія цифрових автоматів: Навч. посібник / В.І. Жабін, І.А. Жуков, І.А. Клименко, В.В. Ткаченко. – К.: Вид-во НАУ, 2009. – 364 с.
3. Матвієнко М. П. Комп'ютерна логіка : підручник / М. П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

8. ЗАНЯТТЯ №8. ОБЧИСЛЕННЯ ФУНКЦІЙ

8.1 Метод обчислення квадратного кореня

Розглянемо метод обчислення функції $Y = \sqrt{X}$ при послідовному введенні коду аргументу X зі старших розрядів. Такий метод дозволяє виконувати процеси введення аргументу та обчислювання функції у режимі суміщення.

Будемо вважати, що значення аргументу подане нормалізованим n -розрядним числом у позиційній системі числення з основою 2.

Процес обчислення квадратного кореня по запропонованому алгоритму закінчується безпосередньо після введення останнього розряду коду аргументу. Це дозволяє зменшити час обчислення в тих випадках, коли швидкість надходження чергових розрядів аргументу визначається зовнішніми стосовно операційного блоку факторами.

Добування кореню додатного числа

Операнд:

$X = 0,110110$

Теоретичне обґрунтування способу виконання операцій

Найбільш простий алгоритм обчислення квадратного кореня з n -розрядної мантиси числа зводиться до підбору цифр результату розряд за розрядом, розпочинаючи зі старшого 2^{-1} -го розряду. За цього обчислення i -ї цифри результату X відбувається наступним чином. Після отримання чергової $(i-1)$ -ї цифри a_{i-1} в i -й розряд A розміщується одиниця. Обчислюється різниця $(B - A_i^2) = R_i$. Якщо $R_i \geq 0$, то A_i є число, де цифри всіх розрядів збігаються з цифрами результату A . Якщо $R_i < 0$, то в i -му розряді a_i необхідно поставити нуль і переходити до обчислення $(i+1)$ -го розряду.

Операційна схема

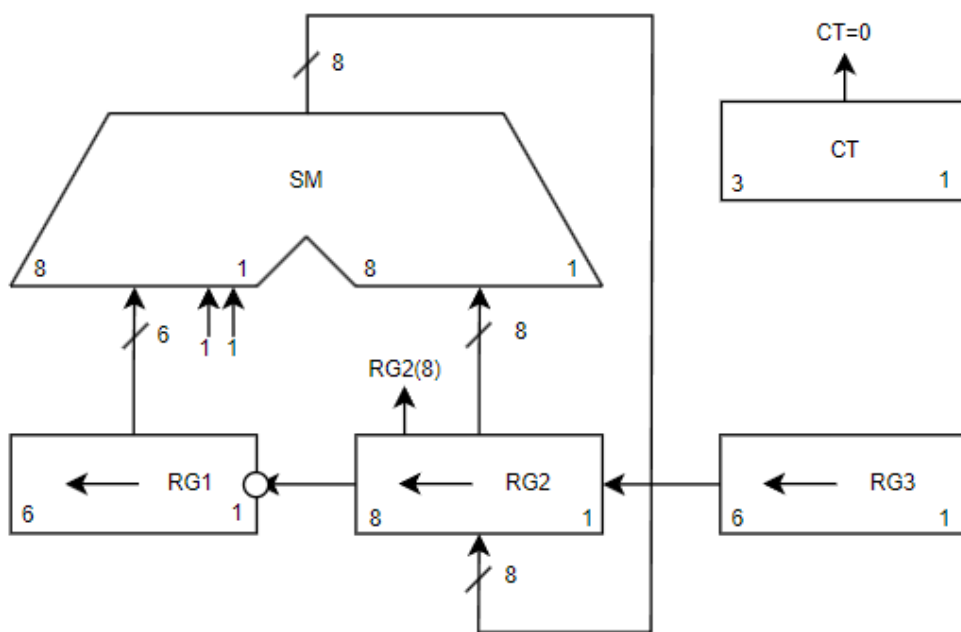


Рисунок 8.1 – Операційна схема добування кореня

Змістовний мікроалгоритм

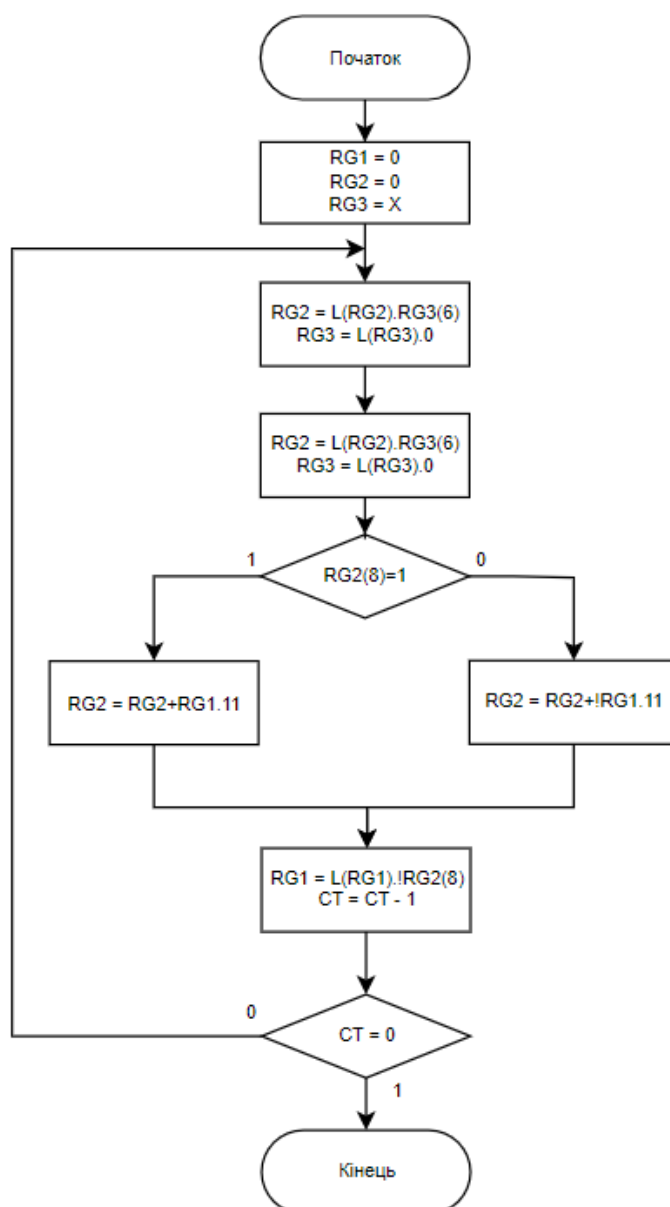


Рисунок 8.2 – Змістовний мікроалгоритм добування кореня

Таблиця станів регістрів

$$\sqrt{|M_X|} = \sqrt{110110}$$

1) $RG_1 = 0; RG_2 = 0; RG_3 = X;$

2) $RG_2 = L(RG_2).RG_3(6); RG_3 = L(RG_3).0; RG_2 = L(RG_2).RG_3(6);$

$RG_3 = L(RG_3).0;$

$RG_2 = RG_2 + !RG_1.11; RG_1 = L(RG_1).!RG_2(8); CT = CT - 1$

3) $RG_2 = L(RG_2).RG_3(6); RG_3 = L(RG_3).0; RG_2 = L(RG_2).RG_3(6);$

$$RG_3 = L(RG_3).0;$$

$$RG_2 = RG_2 + RG_1.11; RG_1 = L(RG_1).!RG_2(8); CT = CT - 1$$

Таблиця 8.1 – Стани реєстрів під час добування кореня

№	RG ₁	RG ₂		RG ₃	CT	Мікрооперації
		8-3	2-1			
-	000000	000000	00	110110	110	1
1	000000 000001	000000 000000 000000 + 111111 <u>000000</u>	00 01 11 <u>11</u> 10	110110 101100 011000	101	2
2	000001 000011	000000 000001 000010 + 111110 <u>000001</u>	10 00 01 <u>11</u> 00	011000 110000 100000	100	2
3	000011 000111	000001 000010 000100 + 111100 <u>000001</u>	00 01 10 <u>11</u> 01	100000 000000 000000	011	2
4	000111 001110	000001 000010 000101 + 111000 <u>111101</u>	01 10 00 <u>11</u> 11	000000 000000 000000	010	2
5	001110 011101	111101 111011 110111 + 001110 <u>000101</u>	11 10 00 <u>11</u> 11	000000 000000 000000	001	3
6	011101 111010	000101 001011 010111 + 100010 <u>111001</u>	11 10 00 <u>11</u> 11	000000 000000 000000	000	2

Відповідь: 0,111010

8.2 Розробка операційного пристрою для обчислення функції $D=2A^2+0,5B$.

Операційна схема для виконання означеної операції зображена на рис. 8.3, де $RG1$, $RG2$ – реєстри, SM – суматор, CT – лічильник. Будемо вважати, що операнди і результат операції додатні і мають тільки дробову частину, що досягається масштабуванням операндів перед початком операції. Змістовний мікроалгоритм зображен на рис. 8.4. В операторних вершинах розміщені мікрооперації Y_i . При цьому, в кожній операторній вершині розміщуються мікрооперації, що виконуються водночас, тобто в одному такті. У логічних вершинах розміщено логічні умови X_i .

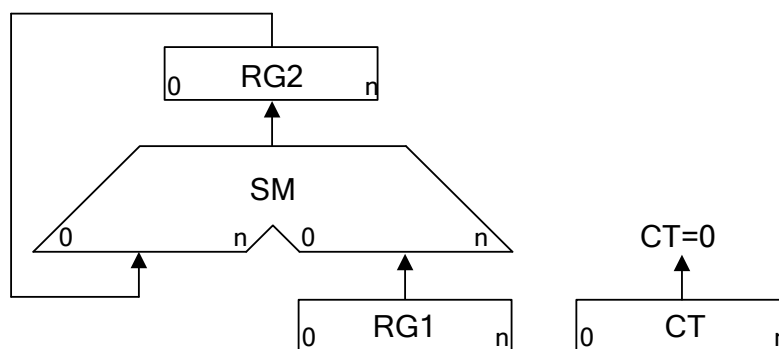


Рис. 8.3 - Операційна схема пристрою для обчислення функції

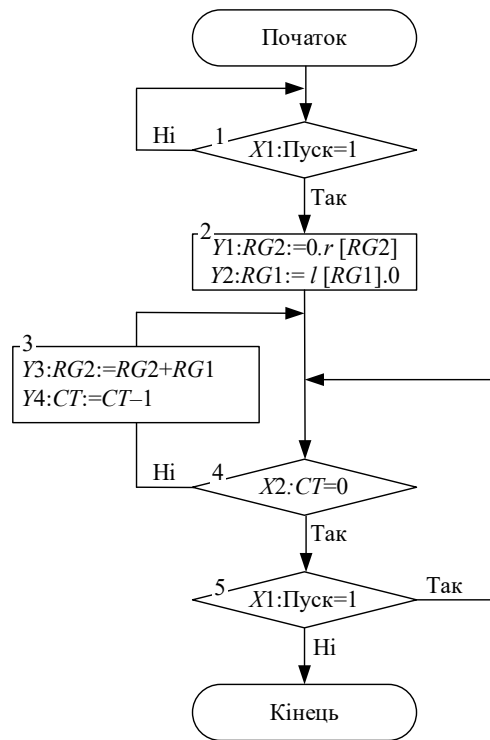


Рис. 8.4 - Змістовний мікроалгоритм обчислення функції

У вихідному стані операнд B записаний в регістр $RG2$, а операнд A – в регістр $RG1$ і в лічильник CT . В першому такті шляхом зсуву вліво операнду A в регістрі $RG1$ здійснюється подвоєння цього операнду і шляхом зсуву вправо операнду B в $RG2$ здійснюється ділення операнду B на 2. Далі до вмісту $RG2$ A раз додається слово, записане в $RG1$. Після кожного додатку вміст CT зменшується на 1. Обчислення закінчуються при виконанні умови $CT=0$. Відповідний цьому сигнал можна одержати, наприклад, дешифруванням нульового стану C . Результат операції формується в регістрі $RG2$.

На основі операційної схеми і змістовного мікроалгоритму розробляємо функціональну схему пристрою, яка зображена на рис. 8.5. На функціональній схемі указані способи формування умов і керуючі сигнали для усіх вузлів пристрою обчислення функції:

W – запис інформації в регістри;

SR – зсув вправо умісту регістрів;

- SL* – зсув вліво умісту регістрів;
- CLR* – обнулення;
- dec* – декримент лічильника;
- inc* – інкремент лічильника;
- d* – сигнал, що дозволяє одержати доповнювальний код числа при реалізації операції віднімання, подається на керуючий вхід пристрою інвертування та вхід *CI* суматора (рис. Б1-7.8).

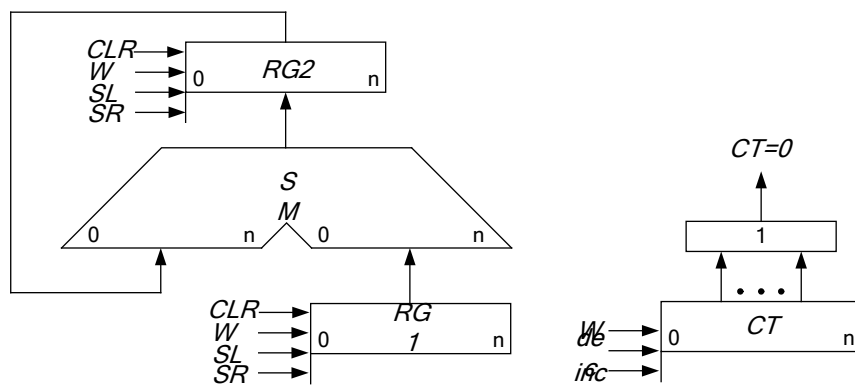


Рис. 8.5 – Функціональна схема операційного автомата

Для виконання мікрооперації на регістрі необхідно подати одиничний сигнал на відповідний керуючий вхід. На всі інші керуючі входи цього регістру повинний подаватися нульовий сигнал.

Визначимо сукупність керуючих сигналів необхідних для виконання кожної мікрооперації. Мікрооперації Y_1 , Y_2 , Y_3 та Y_4 кодуються відповідно керуючими сигналами SL_1 , SR_2 , W_2 , та dec , що подаються на керуючі входи вузлів пристрою відповідно до функціональної схеми.

Необхідна тривалість керуючих сигналів визначається з урахуванням затримок в елементах операційного пристрою. Період t тактуючих сигналів

звичайно обирається або рівним максимальній тривалості керуючих сигналів, або мінімальним. При цьому величина t повинна бути не менше часу переключення автомата з одного стану в інший. У першому випадку всі мікрооперації виконуються в синхронному режимі (за однаковий проміжок часу), а в другому – в асинхронному, причому тривалість керуючих сигналів кратна величині t .

Асинхронний режим можна забезпечити, наприклад, введенням у мікроалгоритм додаткових операторних вершин з керуючими сигналами, тривалості яких перевищують t . Будемо вважати, що з урахуванням швидкодії елементів для розглянутого приклада керуючі сигнали повинні мати тривалість t .

Для одержання закодованого мікроалгоритму узагальнимо результати перших двох етапів в табл. Б1-7.1, де описи мікрооперацій подамо відповідними керуючими сигналами із указанням їх тривалості, описи логічних умов в змістовному мікроалгоритмі подамо їх позначеннями (табл. Б1-7.2). Відповідно до отриманих таблиць виконаємо заміну мікрооперацій і логічних умов у змістовному мікроалгоритмі. Отримаємо закодований мікроалгоритм, що приведений на рис. Б1.7-4.

Таблиця 8.2 – Таблиця кодування мікрооперацій

Мікрооперації	Керуючі сигнали	Тривалість керуючих сигналів
$Y1 :RG1 := \lceil RG1 \rceil . 0$	SL1	t
$Y2 :RG2 := 0 . \lceil RG2 \rceil$	SR2	t
$Y3 :RG2 := RG2 + RG1$	W1	t
$Y4 :CT := CT - 1$	dec	t

Таблиця 8.3 – Таблиця кодування логічних умов

Логічні умови	Позначення логічних умов
$X1 : \text{Пуск} = 1$	x_1
$X2 : CT = 1$	x_2

Керуючі сигнали $SL1$ і $SR2$ генеруються управляючим автоматом в один і той самий такт автоматного часу, отже, можуть бути замінені одним керуючим сигналом y_1 , аналогічним чином сигнали $W1, inc$ замінимо сигналом y_2 . Отримаємо спрощений закодований мікроалгоритм, що зображений на рис. 8.6.

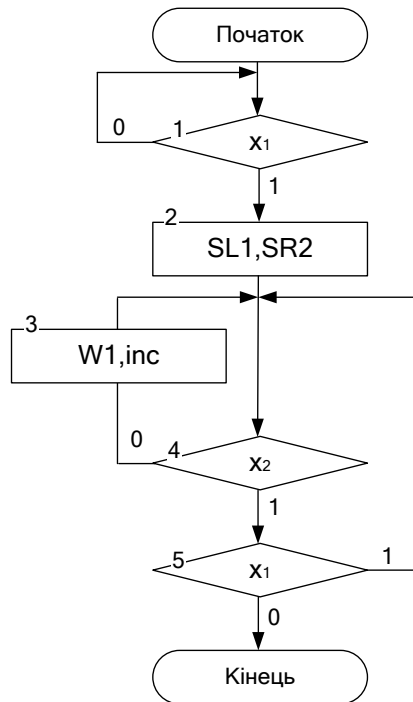


Рис. 8.6 - Закодований мікроалгоритм автомата

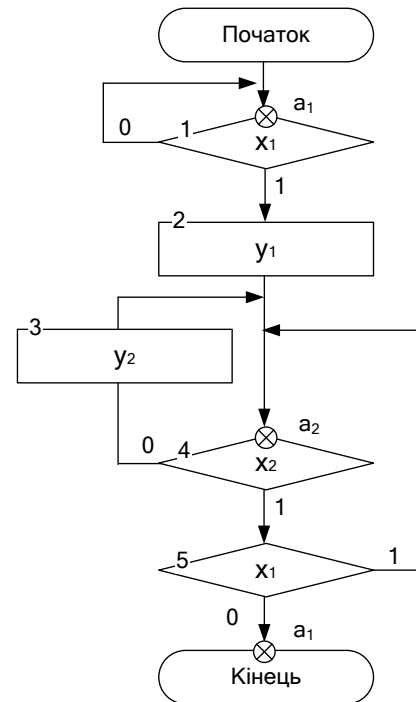


Рис. 8.7 - Розмічений мікроалгоритм автомата

Закодований мікроалгоритм (рис. 8.6) є вихідним для побудови управляючого автомата.

Будемо вважати, що управляючий автомат необхідно побудувати у вигляді автомата Мілі. Виконаємо розмітку закодованого мікроалгоритму, що наведений на рис. 8.6. Далі необхідно виконати синтез автомата Мілі. В результаті синтезу буде отримана функціональна схема управляючого автомату, побудова якої у даному прикладі не розглядається. Керуючі сигнали з виходів управляючого автомату подаються до відповідних входів

операційного автомату. Загальний вигляд ОП для обчислення заданої функції поданий на рис. 8.7.

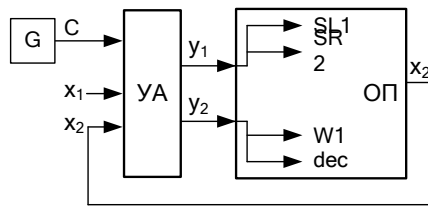


Рис. 8.7 - Структурна схема ОП для обчислення функції

Якщо при виконанні обчислень необхідно виконувати мікрооперацію віднімання, то до складу операційної схеми, зображеної на рис. 8.3 необхідно ввести вузол інвертування для отримання доповнювального коду операнду.

Розробка операційного пристрою з розподіленою логікою для обчислення функції $D=0,5C-2AB$.

На підставі операційної схеми (рис. 8.3) та змістовного мікроалгоритму обчислення заданої функції, що зображений на рис. 8.8, розробляємо функціональну схему операційного автомату для обчислення функції (рис. 8.9). Будемо вважати, що операнди і результат операції мають тільки дробову частину, що досягається масштабуванням операндів перед початком операції.

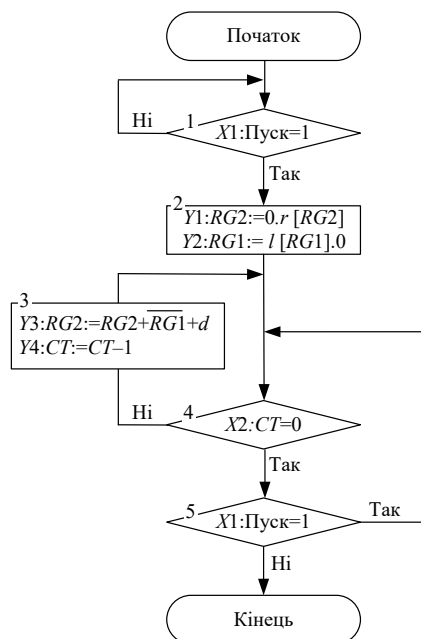


Рис. 8.8 - Змістовний мікроалгоритм обчислення функції

У вихідному стані операнд C записаний в регістр $RG2$, операнд A – в регістр $RG1$, операнд B – в лічильник CT . В першому такті шляхом зсуву вправо операнду C в $RG2$ здійснюється ділення операнду C на 2 і шляхом зсуву вліво операнду A в регістрі $RG1$ здійснюється подвоєння цього операнду. Далі із вмісту регістра $RG2$ B раз віднімається слово, записане в регістрі $RG1$. Операція віднімання реалізується в доповнювальних кодах. Операнд A є додатнім числом, тому реалізація доповнювального коду не потребує вводу додаткових вузлів до операційного пристрою. Перетворення вмісту регістру $RG2$ у доповнювальний код, так як операнд C , записаний у цьому регістрі є оберненим числом, відбувається за допомогою вузла інвертування і додавання одиниці на вхід CI суматора. Для реалізації операції віднімання на вхід d операційного пристрою необхідно подати одиничний сигнал. Після кожного віднімання вміст CT зменшується на 1. Обчислення закінчуються при виконанні умови $CT=0$. Результат обчислення функції формується в регістрі $RG2$.

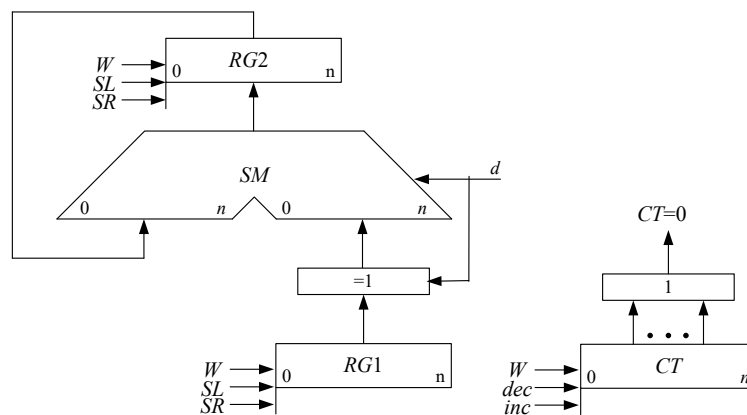


Рис. 8.9 - Функціональна схема операційного автомата

Подальші етапи розробки операційного пристрою виконуються аналогічно попередньому прикладу.

Література

1. Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів: Навч. посібник. – К.: Вид-во НАУ, 2009. – 364 с.
2. Жабін В.І., Жуков І.А., Клименко І.А., Стіренко С.Г. Арифметичні та управляючі пристрої цифрових ЕОМ. – К.: ВЕК+, 2008. – 176 с.
3. Матвієнко М. П. Комп'ютерна логіка : підручник / М. П. Матвієнко. – К.: Видавництво «Ліра-К», 2017. – 320 с.

ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ МОДЕЛЮВАННЯ ЛОГІЧНИХ СХЕМ

Програмний комплекс ПРОГМОЛС 2.0 (ПРОГрама МОделювання Логічних Схем) призначений для моделювання процесів у комбінаційних і послідовнісних схемах. Він дозволяє створювати і редагувати логічні схеми, здійснювати моделювання у синхронному (без урахування затримок сигналів в елементах схеми) і в асинхронному (з урахуванням затримок) режимах, а також зберігати отримані моделі.

Вигляд моделі проілюстровано на рис. 1.1.

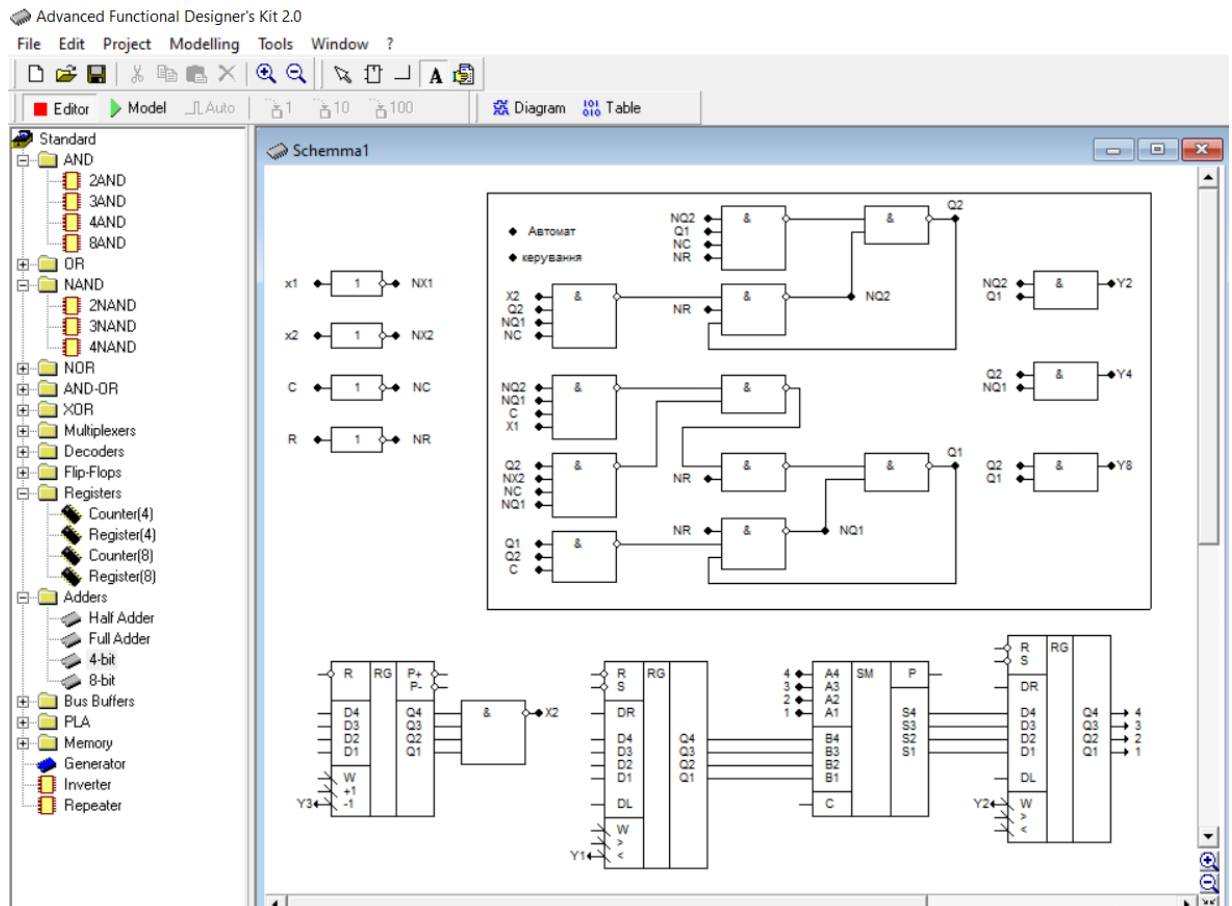


Рис. 1.1. Зображення моделі операційного пристрою з автоматом керування

Для роботи з програмою використовують систему ієрархічних меню, що містить наступні розділи:

- *файл;*
- *виправлення;*
- *проект;*
- *моделювання;*
- *інструменти;*
- *вікно;*
- *допомога.*

Комплекс включає систему підказок, що полегшує роботу в різних режимах моделювання. Побудова та елементи керування для кожного розділу пояснюються нижче.

Файл



Створити (сполучення клавіш **Ctrl+N).** Створює новий файл проекту.



Відкрити (Ctrl+O**).** Викликає діалогове вікно відкриття проекту.



Зберегти (Ctrl+S**).** Зберігає файл проекту на диск під поточним ім'ям.



Зберегти як (Ctrl+A**).** Зберігає файл проекту на диск і запитує ім'я та шлях файлу.



Закрити (Ctrl+F4**).** Закриває поточний файл проекту.



Вихід (Alt+F4**).** Закриває програму і всі вікна.

Виправлення



Вирізати (Ctrl+X**).** Вирізає фрагмент схеми у внутрішній буфер обміну програми.



Копіювати (Ctrl+C**).** Копіює фрагмент схеми у внутрішній буфер обміну програми.



Вставити (Ctrl+V**).** Вставляє фрагмент схеми з внутрішнього буфера обміну програми.



Видалити (Delete). Видаляє фрагмент схеми.

Проект



Корпус мікросхеми. Показати/сховати корпус мікросхеми поточного проекту.



Редактор корпусу. Викликає діалогове вікно редактора корпусу мікросхеми поточного проекту.



Компіляція (Alt+C). Компілює поточний проект. Дана команда використовується для відновлення списку змінних у діаграмі і таблиці проекту після зміни схеми без включення режиму моделювання. При включенні режиму моделювання компіляція робиться автоматично.



Додати в бібліотеку. Копіює поточний проект у буфер обміну редактора бібліотек і викликає редактор бібліотек. Для вставки компонента в бібліотеку необхідно викликати команду **Вставити** редактора бібліотек.



Настроювання (Ctrl+F4). Викликає діалогове вікно настроювання проекту.

Моделювання



Відробити інтервал. Відпрацьовує заданий користувачем інтервал модельного часу.

Відробити до. Відпрацьовує до моменту модельного часу, заданого користувачем.

Генератор (G). У синхронному режимі викликає чергову зміну стану генераторів і відпрацьовує схему до закінчення перехідних процесів. В асинхронному режимі відпрацьовує 1 такт модельного часу.

Інструменти



Редактор бібліотек (Alt+L). Активізує редактор бібліотек.



Настроювання. Викликає діалогове вікно настроювань програми.



Діаграма (Alt+D). Виводить на екран діаграму станів змінних поточного проекту.



Таблиця (Alt+T). Виводить на екран таблицю станів змінних поточного проекту.

Вікно



Каскадом. Розташовує вікна відкритих проектів каскадом.



Розділити по вертикалі. Розташовує вікна відкритих проектів по вертикалі таким чином, щоб вони не перекривалися.



Розділити по горизонталі. Розташовує вікна відкритих проектів по горизонталі таким чином, щоб вони не перекривалися.



Збільшити. Збільшує масштаб у редакторі схеми поточного проекту.



Зменшити. Зменшує масштаб у редакторі схеми поточного проекту.

Допомога

Про програму. Виводить відомості про програму і розроблювачів.



Допомога. Викликає файл довідки.

Вказані нижче елементи керування винесені на окремі панелі.



Створити (Ctrl+N). Створює новий файл проекту.



Відкрити (Ctrl+O). Викликає діалогове вікно відкриття проекту.



Зберегти (Ctrl+S). Зберігає файл проекту на диск під поточним ім'ям.



Вирізати (Ctrl+X). Вирізає фрагмент схеми у внутрішній буфер обміну програми.



Копіювати (Ctrl+C). Копіює фрагмент схеми у внутрішній буфер обміну програми.



Вставити (Ctrl+V). Вставляє фрагмент схеми з внутрішнього буфера обміну програми.



Видалити (Delete). Видаляє фрагмент схеми.



Збільшити. Збільшує масштаб у редакторі схеми поточного проекту.



Зменшити. Зменшує масштаб у редакторі схеми поточного проекту.



Виділити. Дозволяє виділити і перетягнути лівою кнопкою миші фрагмент схеми, а також інвертувати виходи та входи елементу подвійним натисканням.



Елемент. Дозволяє вставити новий елемент у схему.



Лінія. Дозволяє провести зв'язок у схемі.



Перемінна. Дозволяє визначити змінну в схемі.



Захоплення. Дозволяє перетаскувати лівою кнопкою миші весь зміст вікна редактора логічних схем.



Редактор. Переводить редактор логічних схем у режим редагування.



Модель. Переводить редактор логічних схем у режим моделювання.



Авто. У натиснутому стані включає режим автоматичного моделювання, а у віджатому – режим покрокового моделювання.



1 такт. Відпрацьовує 1 такт модельного часу.



10 тактів. Відпрацьовує 10 тактів модельного часу.



100 тактів. Відпрацьовує 100 тактів модельного часу.

Панель компонентів

Панель компонентів розташована в лівій частині робочого стола моделі і являє собою ієрархічний список, у якому відображені підключені до програми бібліотеки і їх зміст. Кожна бібліотека містить ієрархічну структуру компонентів, подібну до файлової системи. Панель компонентів призначена для вибору компонента для наступної вставки його в схему.

Для дослідження схем у загальному випадку необхідно виконати послідовність дій.

1. Створити за допомогою редактора логічну схему на екрані дисплея.
2. Позначити на схемі вхідні і вихідні змінні.
3. Створити заголовок (перелічити вхідні і вихідні змінні) і сформулювати послідовність вхідних наборів в таблиці істинності.
4. Задати необхідні величини затримок сигналів для елементів схеми.
5. Встановити початковий стан схеми.
6. Перейти до режиму моделювання схеми.