В.І.ЖАБІН, І.А.ЖУКОВ,
І.А.КЛИМЕНКО, С.Г.СТІРЕНКО

АРИФМЕТИЧНІ ТА УПРАВЛЯЮЧІ ПРИСТРОЇ ЦИФРОВИХ ЕОМ

НАВЧАЛЬНИЙ ПОСІБНИК

Рекомендовано
Міністерством освіти і науки України
як навчальний посібник для студентів
вищих технічних навчальних закладів,
які навчаються за напрямом
“Комп’ютерна інженерія”

ВЕК +
КИЇВ
2008

УДК 004.31
ББК *З* 973.20-047я7
А 817

Рецензенти:
Н.І.Алішов, д-р техн. наук, с.н.с.
(Інститут кібернетики ім. В.М. Глушкова НАН України)
І.А.Дичка, д-р техн. наук, проф.
(Національний технічний університет України "КПІ")
С.Ф.Теленик, д-р техн. наук, проф.
(Національний технічний університет України "КПІ")

Гриф надано Міністерством освіти і науки України
(Лист № 1.4/18-Г-2426 від 29.12.2007)

Видання друкується за рішенням Вченої ради
Інституту комп'ютерних технологій НАУ
(Протокол № 10 від 14.11.2007)

Жабін В.І., Жуков І.А., Клименко І.А., Стіренко С.Г. –
А 817 Арифметичні та управляючі пристрої цифрових ЕОМ: Навчальний посібник. – К.:ВЕК +, 2008. – 176 с.

ISBN 966–7140–11–3

Навчальний посібник присвячений питанням реалізації арифметичних операцій в цифрових ЕОМ. Розглянуті питання побудови арифметичних пристроїв різних типів та засобів управління виконанням операцій. Запропоновані завдання та надані рекомендації по організації практичних і лабораторних занять, подані приклади побудові функціональних та принципових електричних схем.

Посібник призначений для студентів навчального напряму "Комп'ютерна інженерія". Може бути корисним спеціалістам, що працюють в області проектування цифрових систем.

ISBN 966–7140–11–3

ЗМІСТ

Вступ 5

Перелік скорочень 7

Розділ 1. Основи цифрової обробки інформації в арифметичних пристроях ЕОМ 8
1.1. Кодування чисел у ЕОМ 8
1.2. Форми подання чисел у ЕОМ 10
1.3. Додавання чисел із знаками у машинних кодах 14
1.4. Додавання і віднімання чисел в зворотних кодах 15
1.5. Додавання і віднімання чисел в доповнювальних кодах 18
1.6. Зсуви машинних кодів 20
1.7. Операційні схеми та мікроалгоритми 24

Розділ 2. Арифметико-логічні пристрої 29
2.1. Арифметико-логічні пристрої з розподіленою логікою 30
2.2. Арифметико-логічні пристрої з зосередженою логікою 42

Розділ 3. Синтез блоків мікропрограмного управління 56
3.1. Призначення та класифікація блоків управління 56
3.2. Блоки мікропрограмного управління 58
3.3. БМУ з примусовою адресацією 68
3.4. БМУ з відносною адресацією 76

Розділ 4. Проектування пристроїв з мікропрограмним управлінням для виконання арифметичних операцій 87
4.1. Операція ділення 87
4.2. Обчислення квадратного кореня 94

Розділ 5. Проектування пристроїв для машинного перетворення чисел в різні системи числення 102
5.1. Перетворення чисел в ЕОМ 102
5.2. Перетворення чисел із десяткової системи числення в двійкову методом «зсуву-корекції» 102
5.3. Перетворення чисел з двійкової системи числення в десяткову методом «зсуву-корекції» 107

Розділ 6. Проектування арифметичних пристроїв для виконання операцій додавання та віднімання чисел із плаваючою комою 116
6.1. Додавання чисел із плаваючою комою 116

6.2. Множення чисел із плаваючою комою 126
6.3. Ділення чисел із плаваючою комою 126
6.4. Добування квадратного кореня з числа із плаваючою комою 127

Розділ 7. Завдання до виконання практичних робіт 131
7.1. Практична робота 1 131
7.2. Практична робота 2 132
7.3. Практична робота 3 133
7.4. Загальні вказівки до виконання практичних робіт 134

Розділ 8. Задачі для самостійного розв'язування 139

Розділ 9. Завдання до виконання розрахунково-графічної роботи 145

Розділ 10. Вимоги до оформлення конструкторської документації 148

Розділ 11. Вимоги до оформлення електричних схем 150

Список літератури 153

Додатки 153

ВСТУП

В навчальному посібнику узагальнені матеріали методичних розробок, які виконувалися авторами в процесі викладання курсів схемотехнічного напрямку на кафедрі комп'ютерних систем та мереж Інституту комп'ютерних технологій Національного авіаційного університету та кафедри обчислювальної техніки Національного технічного університету України "КПІ" для студентів навчального напряму "Комп'ютерна інженерія".

Матеріал навчального посібника присвячений вивченню принципів організації та дослідженню арифметико-логічних і управляючих пристроїв, а також побудові функціональних та принципових електричних схем цифрових ЕОМ.

Крім необхідного теоретичного матеріалу, в посібнику приведені завдання та рекомендації до виконання лабораторних робіт. До лабораторних робіт надані теоретичні відомості, необхідні для виконання кожної роботи, приклади проектування, рекомендації до виконання завдання та саме завдання. До кожної лабораторної роботи надаються контрольні питання що застосовуються для контролю знань за відповідною тематикою.

Виконання лабораторних робіт дозволяє розширити і закріпити теоретичні знання з дисципліни, опанувати навички проектування і дослідження арифметичних та управляючих пристроїв цифрових ЕОМ. Кожній лабораторній роботі повинна передувати самостійна підготовка студентів, в процесі якої вони докладно вивчають опис лабораторної роботи, відповідні розділи посібника, конспекту лекцій та літературні джерела. В процесі підготовки складається звіт про лабораторну роботу, в якому повинні бути відображені всі пункти теоретичного завдання, а також заготовлені для виконання експериментальної частини лабораторної роботи таблиці, осі для часових діаграм і таке інше. Перед початком лабораторної роботи результати підготовки перевіряються викладачем. За цим студент повинен представити заготовлений звіт і

відповісти на контрольні питання. Перед початком наступного заняття в лабораторії студент представляє викладачеві цілком оформлений звіт за попередньою роботою. Звіт повинен містити короткі теоретичні відомості, необхідні для виконання завдання, відповіді на контрольні питання, усі схеми, формули, таблиці, діаграми, графіки, отримані при виконанні завдання та в процесі експериментального дослідження схем, а також висновки за роботою. Залік за виконання лабораторної роботи студент одержує після співбесіди за тематикою виконаної роботи.

У посібнику надані завдання до виконання практичних та розрахунково-графічних робіт, а також задачі до самостійного розв'язування на базі яких складаються тести до модульних контролів.

Теоретичний матеріал, зміст лабораторних робіт, практичних завдань та розрахунково-графічної роботи безпосередньо відповідають навчальному плану дисципліни "Цифрові ЕОМ". Посібник може бути корисним при вивченні курсів "Комп'ютерна схемотехніка", "Архітектура комп'ютерів", "Проектування комп'ютерних систем" та інших курсів схемотехнічного напрямку.

Автори посібника вдячні рецензентам: провідному науковому співробітнику Інституту кібернетики ім. В.М. Глушкова НАН України, доктору технічних наук, старшому науковому співробітнику Алішову Н.І., професору кафедри спеціалізованих комп'ютерних систем Національного технічного університету України "Київський політехнічний інститут", доктору технічних наук, професору Дичці І.А., завідувачу кафедри автоматики та управління в технічних системах Національного технічного університету України "Київський політехнічний інститут", доктору технічних наук, професору Теленику С.Ф. за слушні зауваження.

ПЕРЕЛІК СКОРОЧЕНЬ

АЛБ – Арифметико-логічний блок
АЛП – Арифметико-логічний пристрій
БМУ – Блок мікропрограмного управління
БУ – Блок управління
ВМ – Вертикальне мікропрограмування
ГМ – Горизонтальне мікропрограмування
ДК – Доповнювальний код
ЕОМ – Електронно-обчислювальні машини
ЗК – Зворотний код
МА – Мікроалгоритм
МК – Мікрокоманда
МО – Мікрооперація
МП – Мікропрограма
НОЗП – Надоперативний запам’ятовуючий пристрій
ОПр – Операційний пристрій
ОП – Основна пам’ять
ОС – Операційна схема
ОТ – Обчислювальна техніка
ПЗП – Постійний запам’ятовуючий пристрій
ПК – Прямий код
ПМК – Пам’ять мікрокоманд
ПЛМ Програмовані логічні матриці
УГП – Умовне графічне позначення
УП – Управляючий пристрій
УС – Управляючий сигнал

РОЗДІЛ 1

Основи цифрової обробки інформації в аріфметичних пристроях ЕОМ

1.1. Кодування чисел у ЕОМ

В ЕОМ доцільно подавати знаки чисел за допомогою тих самих символів, що застосовуються для запису самого числа в *k*-й системі числення. Для цього використовують додатковий розряд, названий знаковим, який розташовується ліворуч від старшого розряду числа.

В ЕОМ для виконання операцій з числами, що мають знаки, використовують спеціальні коди:

- прямий код,
- зворотний код,
- доповнювальний код.

Розглянемо прямий, зворотний та доповнювальний коди для двійкової системи числення з цифрами {0,1}, саме яка має найбільше поширення в ОТ.

При запису числа вважаємо, що число має *n*-розрядну цілу частину і *k*-розрядну дробову частину. Крім цього, до числа додається ще знаковий розряд. Відповідна *розрядна сітка* зображена на рис. 1.1.

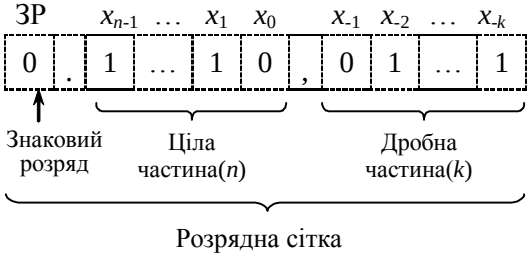


Рис. 1.1. Приклад запису двійкового числа зі знаком

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

Подання числа X у *прямому коді* визначається виразом:

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

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 А=10,101011; *В*= –10,0111010 у ПК.

Виконання завдання

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

Приклад 1.2. Записати числа *А*=0,101011; *В*= – 0,0111010 у ПК.

Виконання завдання

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

Прямий код застосовується для зберігання чисел в пам'яті комп'ютера. Для операцій додавання і віднімання ПК не використовується.

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

Формула перетворення чисел у зворотний код має вигляд:

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

Приклад 1.3. Записати числа *А*=10,1011; *В*= – 01,11010 в ЗК.

Виконання завдання

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

Приклад 1.4. Записати числа *А*=101011; *В*= – 0111010 в ЗК.

Виконання завдання

$$[A]_{зк} = 0.101011; [B]_{зк} = 1.1000101.$$

Приклад 1.5. Записати числа A=0,101011; B= – 0,0111010 в ЗК.

Виконання завдання

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

Під час перетворення від'ємного числа в *доповнювальний 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 ЗК, тобто в знаковий розряд записується 0, а основні розряди не змінюються.

Формула перетворення чисел у доповнювальний код має вигляд:

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

Приклад 1.6. Записати числа A= – 011,100; B= 010,010 у ПК, ЗК і ДК.

Виконання завдання

$$[A]_{ПК} = 1\ .\ 011\ ,\ 100 \qquad [B]_{ПК} = 0\ .\ 010\ ,\ 010$$

$$[A]_{зк} = 1\ .\ 100\ ,\ 011 \qquad [B]_{зк} = 0\ .\ 010\ ,\ 010$$

$$+ \qquad\qquad 1 \qquad\qquad [B]_{ДК} = 0\ .\ 010\ ,\ 010$$

$$[A]_{ДК} = 1\ .\ 100\ ,\ 100$$

Приклад 1.7. Записати числа A= – 0,011100; B= 0,010010 у ПК, ЗК і ДК.

Виконання завдання

$$[A]_{ПК} = 1\ ,\ 0111\ 0\ 0 \qquad [B]_{ПК} = 0\ ,\ 010010$$

$$[A]_{зк} = 1\ ,\ 1000\ 1\ 1 \qquad [B]_{зк} = 0\ ,\ 010010$$

$$+ \qquad\qquad 1 \qquad\qquad [B]_{ДК} = 0\ ,\ 010010$$

$$[A]_{ДК} = 1\ ,\ 1001\ 0\ 0$$

1.2. Форми подання чисел у ЕОМ

В ОТ переважно використовуються дві форми подання чисел:

- подання чисел із 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.

За подання 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.

За фіксації коми перед старшим розрядом мантиси ЕОМ оперує з 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, за 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 із фіксованою комою складається з 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 *з плаваючої комою*.

При поданні чисел у формі з 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

$$k^{-1} \le \mathrm{M} \le 1. \qquad (1.1)$$

Це означає, що старший розряд модуля мантиси завжди дорівнює 1 (мантиса нормалізована). Якщо в ЕОМ для запису порядку використається 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. 1.2).

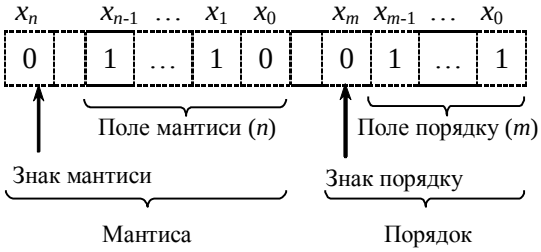


Рис. 1.2. Приклад запису числа у формі з плаваючою комою

Приклад 1.8. Записати у формі із плаваючою комою (m=n=4) двійкові доповнювальні коди чисел:

$$X = 9_{(10)},\ Y = -9_{(10)},\ Z = -9/256_{(10)}.$$

Виконання завдання

$$P_{(X)} = 0,0100 \qquad M_{(X)} = 0,1001$$

$$P_{(Y)} = 0,0100 \qquad M_{(Y)} = 1,0111$$

$$P_{(Z)} = 1,1100 \qquad M_{(Z)} = 1,0111$$

У ЕОМ, де використається подання чисел у формі з 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 (1.1). Тому машини з 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* 754-1985 на подання чисел із плаваючою комою. В цьому стандарті визначені два основні базові формати: короткий та довгий. У

кожному форматі використовують схований старший розряд мантиси з вагою 2^{-1}, який завжди дорівнює 1 для додатних чисел.

Короткий формат (рис. 1.3, *а*) має 32 розряди, з яких 8 розрядів призначені для подання порядку, 23 розряди – для подання мантис і один розряд – для знака мантиси. Спеціального розряду для знаку порядку немає. Вводиться *зміщений порядок:* $P_{зм}=P-128$. Таким чином, від'ємні порядки P будуть подані зміщеними порядками $P_{зм}$, меншими за 128, а додатні порядки – більшими за 128. Введення зміщеного порядку дозволяє звести операції над порядками до арифметичної дії над цілими додатними числами.

Отже, максимальний додатний порядок числа в короткому форматі дорівнює

$$11111111_{(2)} - 10000000_{(2)} = 01111111_{(2)} = 127_{(10)},$$

а максимальний від'ємний порядок

$$00000000_{(2)} - 1000000_{(2)} = -128_{(10)}.$$

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. 1.3. 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.

Крім розглянутих вище найбільш поширених форм подання чисел у спеціалізованих ЕОМ можуть використатися також форми, що одержані шляхом певного функціонального перетворення чисел. Прикладом такої форми є логарифмічна, коли числа представляються їхніми

логарифмами по деякій основі. Це дає можливість замінити операції множення й ділення чисел операціями додавання й вирахування їхніх логарифмів.

1.3. 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 порізному. Окрім того, операція віднімання повинна здійснюватися не на суматорі, а на спеціальній схемі.

З використанням зворотних та доповнювальних кодів операції додавання і віднімання можна виконувати за допомогою тільки суматорів, на яких оброблюються як основні, так й знакові розряди. В цьому випадку операція віднімання заміняється операцією додавання з числом, що має протилежний знак. Наприклад, операція $S = A – B$ виконується як $S = A +(– B)$.

Для виконання операції віднімання чисел в зворотних та доповнювальних кодах використовують багаторозрядні суматори. Схеми та принцип функціонування суматорів розглянуті у [6].

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

Ознакою переповнення є розбіжність значень вхідного переносу P_{in} у знаковий розряд і вихідного переносу P_{out} із знакового розряду. Отже, ознаку переповнення можна сформувати перемикальною функцією $OVR=P_{in}\oplus P_{out}$.

Інший підхід для виявлення переносу складається у використанні другого допоміжного знакового розряду ліворуч від основного (першого) знакового розряду. Таке подання чисел називають *модифікованим машинним кодом*. Старший знаковий розряд при цьому завжди зберігає знак результату.

Формули кодування для модифікованих кодів мають вигляд:

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

$$[A]_{ДК} = \begin{cases} A, якщо\ A \geq 0; \\ 2^{n+2} + A, якщо\ A < 0. \end{cases} \qquad (1.2)$$

1.4. Додавання і віднімання чисел в зворотних кодах

В ЗК операція віднімання замінюється операцією додавання, при цьому знаковий розряд 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 ЗК є циклічний перенос зі знакового розряду в молодший розряд суми, завдяки якому здійснюється корекція результату.

Розглянемо чотири можливих випадки додавання чисел у модифікованих кодах.

Випадок 1. $A>0$, $B>0$, $A+B>0$.

Під час підсумовування кодів відповідно до функції кодування (1.2) ми повинні одержати

$$[A]_{ЗК}+[B]_{ЗК} = [A+B]_{ЗК}=A+B,$$

тому що доданки додатні, тобто

$$[A]_{ЗК} =A,\ [B]_{ЗК}=B \text{ і } [A]_{ЗК}+[B]_{ЗК} = A+B.$$

У даному випадку шуканий результат збігається з отриманим. Отже, корекція не потрібна. Помітимо, що при підсумовуванні нульових знакових розрядів перенос не формується, тобто корекція за циклічним ланцюгом справді буде відсутня.

Випадок 2. $A>0$, $B<0$, $|A|>|B|$, $A+B>0$.

Сума додатна, отже, результат повинен мати вигляд

$$[A]_{ЗК}+[B]_{ЗК}= [A+B]_{ЗК}=A+B.$$

У підсумовуванні беруть участь коди $[A]_{ЗК}=A$ і $[B]_{ЗК}=2^{n+2}-2^k+B$. Сума буде мати вигляд

$$[A]_{ЗК}+[B]_{ЗК}=A+2^{n+2}-2^k+B= 2^{n+2}-2^k+(A+B).$$

У результаті операції додавання виникла помилка $(2^{n+2}-2^k)$. Тут 2^{n+2} – перенос зі знакового розряду за межі розрядної сітки, тобто на цю величину корекція не потрібна. Таким чином, результат отриманий з недоліком на величину 2^k. Корекція на $(+2^k)$ здійснюється додаванням у молодший розряд результату переносу за циклічним ланцюгом, що завжди виникає при зазначених значеннях операндів.

Приклад 1.9. Задано $A= 0{,}10101$; $B= -0{,}01001$. Знайти суму C.

Виконання завдання

$$\begin{array}{r} [A]_{\text{ЗК}} = 00,10101 \\ +\,[B]_{\text{ЗК}} = 11,10110 \\ \hline 00,01011 \\ +\qquad\quad 1 \text{ (перенос)} \\ \hline [C]_{\text{ЗК}} = 00,01100 \end{array}$$

Випадок 3. *A>0, B<0, |A|<|B|, A+B<0.*

Сума від'ємна, виходить, результат відповідно до виразу (1.2) повинен мати вигляд

$$[A]_{\text{ЗК}}+[B]_{\text{ЗК}}=[A+B]_{\text{ЗК}}= 2^{n+2}-2^{k}+(A+B).$$

Підсумовуються коди $[A]_{\text{ЗК}}$=A і $[B]_{\text{ЗК}}= 2^{n+2}-2^{k}+B$. В результаті одержимо

$$[A]_{\text{ЗК}}+[B]_{\text{ЗК}}=A+2^{n+2}-2^{k}+B= 2^{n+2}-2^{k}+(A+B).$$

Шуканий результат збігається з отриманим, тобто корекція не потрібна.

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

Виконання завдання

$$\begin{array}{r} [A]_{\text{ЗК}} = 00,01001 \\ +\,[B]_{\text{ЗК}} = 11,01010 \\ \hline [C]_{\text{ЗК}} = 11,10011 \\ [C]_{\text{ПК}} = 11,01100 \end{array}$$

Випадок 4. *A<0, B<0, A+B<0.*

Сума від'ємна, відповідно до цього, результат повинен мати вигляд

$$[A]_{\text{ЗК}}+[B]_{\text{ЗК}} = [A+B]_{\text{ЗК}}=2^{n+2}-2^{k}+(A+B).$$

Підсумовуються коди $[A]_{\text{ЗК}}= 2^{n+2}-2^{k}+A$ і $[B]_{\text{ЗК}}= 2^{n+2}-2^{k}+B$.

В результаті на суматорі одержимо

$$[A]_{\text{ЗК}}+[B]_{\text{ЗК}}= 2^{n+2}-2^{k}+A+ 2^{n+2}-2^{k}+B=2^{n+2}-2^{k}+(A+B)+2^{n+2}-2^{k}.$$

Потрібна корекція результату, що здійснюється аналогічно другому випадкові.

Приклад1.11. Задано зворотні коди чисел $A = -0.10101$ і $B = -0.01001$. Знайти зворотний код суми C.

Виконання завдання

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

Під час додавання чисел з однаковими знаками може виникнути переповнення розрядної сітки. Ознакою переповнення при використанні модифікованих кодів є різні цифри в знакових розрядах. Функцію переповнення можна записати у вигляді $OVR = ЗР_2 \oplus ЗР_1$. Старший знаковий розряд завжди зберігає правильний знак результату.

Приклад 1.12. Задано $A = 0{,}10101$ і $B = 0{,}01110$. Знайти суму C.

Виконання завдання

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

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

Виконання завдання

$$\begin{array}{r} [A]_{ЗК} = 11,01010 \\ +\underline{[B]_{ЗК} = 11,10001} \\ 10,11011 \\ + \qquad\quad 1 \text{ (перенос)} \\ \hline [C]_{ЗК} = 00,01100 \end{array}$$ – від'ємне переповнення

Функціональна схема пристрою, що реалізує операцію додавання і віднімання в зворотних кодах наведена на рис. 1.4.

Операнди надходять на вхід суматора *SM* з регістрів *RGx* і *RGy*. Операція віднімання виконується шляхом додавання зменшуваного 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 (ОР) 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 замкнутий, тому що перенос виникає тільки тоді,

коли потрібно реалізувати корекцію суми. Щоб знайти переповнення розрядної сітки використовується ознака переповнення *OVR* ($OVR = ЗР_2 \oplus ЗР_1$). При значенні *OVR* = 0, переповнення розрядної сітки відсутнє, у випадку, коли *OVR* = 1 наявне переповнення розрядної сітки.

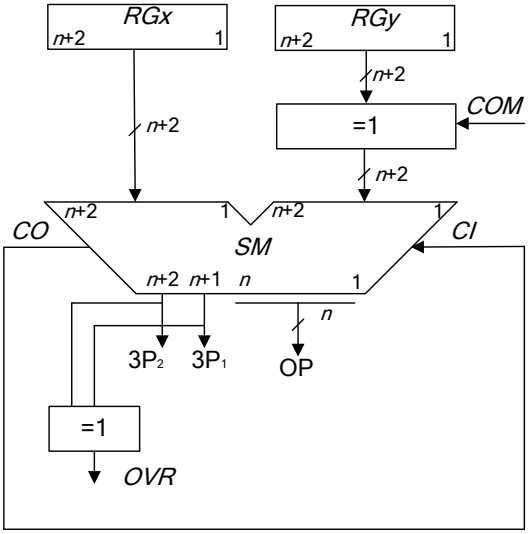


Рис. 1.4. Схема виконання операцій додавання і віднімання в зворотних кодах

1.5. Додавання і віднімання чисел в доповнювальних кодах

У ДК операція віднімання, як і в ЗК, замінюється операцією додавання зменшуваного до від'ємного від'ємника. За підсумовування операндів у ДК автоматично отримуємо суму в ДК з урахуванням знаку. За застосування модифікованого коду корекції робити 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

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

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

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

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

$$\begin{array}{r} [A]_{\text{ДК}} = 11,0\ 1011 \\ +\,[B]_{\text{ДК}} = 11,1\ 0010 \\ \hline [C]_{\text{ДК}} = 10,1\ 1101 \end{array}$$ – від'ємне переповнення

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. 1.4), але в цьому випадку відсутній циклічний ланцюг корекції результату *CO→ 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.

1.6. 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. 1.5.

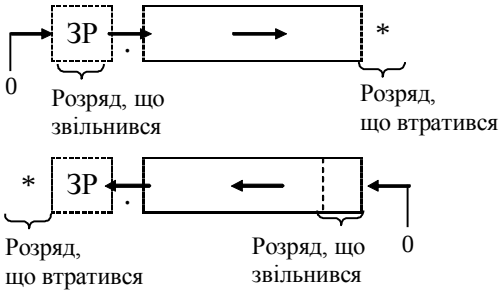


Рис. 1.5. Операційна схема логічного зсуву двійкових чисел

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

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

Виконання завдання

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

$$\begin{array}{ll} A & 1.0101,1101 \\ \leftarrow A & *0.1011\ 1010 \end{array}$$

Арифметичний зсув виконується з урахуванням знакового розряду. Правила зсуву чисел поданих у ПК, ЗК 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. 1.6

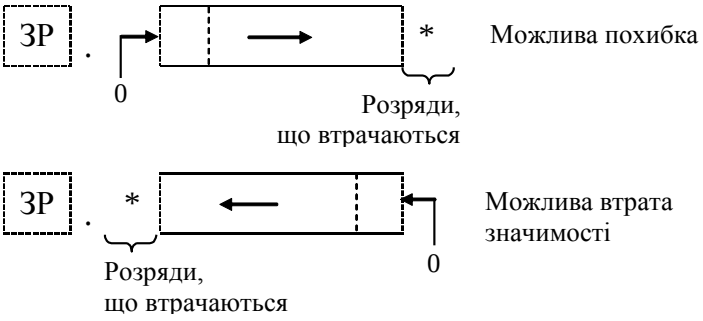


Рис. 1.6. Операційна схема арифметичного зсуву чисел у ПК

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

$$\begin{array}{lll} A & 1.\ 1\ 0\ 101 & \\ A\rightarrow & 1.\ 0\ 1\ 010* & \text{Похибка} \\ \leftarrow A & 1.*0\ 1\ 010 & \text{Втрата значимості} \end{array}$$

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 ЗК.

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. 1.7 Зображена операційна схема реалізації арифметичного зсуву ліворуч від'ємних чисел поданих у ЗК.

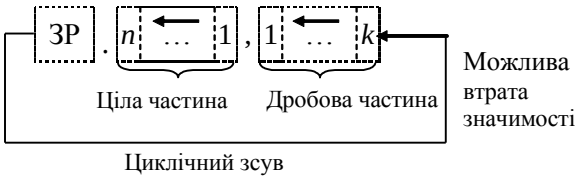


Рис. 1.7. Операційна схема арифметичного зсуву ліворуч від'ємних чисел у ЗК

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

$A_{(2)}$ = 1. 1101, 1010; $B_{(2)}$ = 1. 0011, 0011.

Виконання завдання

A	1.1101,0101	
$\leftarrow A$	1.1010,1011	
B	1.0011,0011	
$\leftarrow B$	0.0110,0111	Втрата значимості

Арифметичний зсув праворуч чисел, поданих у ЗК.

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

На рис. 1.8 зображена операційна схема реалізації арифметичного зсуву праворуч від'ємних чисел поданих у ЗК.

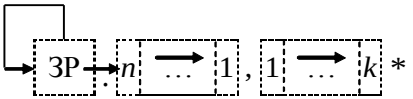


Рис. 1.8. Операційна схема арифметичного зсуву праворуч від'ємних чисел у ЗК

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

$A_{(2)} = 0.\ 1011,\ 1001;\ B_{(2)} = 1.\ 1010,\ 0011.$

Виконання завдання

$$\begin{array}{ll} A & 0.1011,1001 \\ \rightarrow A & 0.0101,1100^{*} \end{array}$$

$$\begin{array}{ll} B & 1.1010,0011 \\ \rightarrow B & 1.1101,0001^{*} \end{array}$$

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. 1.9 зображена операційна схема реалізації арифметичного зсуву ліворуч чисел поданих у ДК.

Рис. 1.9. 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. 1.10.

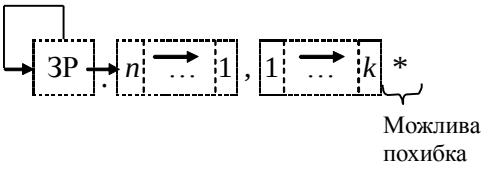


Рис. 1.10. Операційна схема арифметичного зсуву праворуч від'ємних чисел у ДК

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

$$\begin{array}{ll} A & 0.1011,0111 \\ \leftarrow A & *1.0110,1110 \\ A \rightarrow & 0.0101,1011* \end{array}$$

$$\begin{array}{ll} B & 1.1011,1001 \\ \leftarrow B & *1.0111,0010 \\ \rightarrow B & 1.1101,1100* \end{array}$$

1.7. 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 *мікрооперацією* 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 *RG*1(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 (: =).

Пересилання слів здійснюється, як правило, між двома регістрами. Результатом пересилання є запис слова в регістр, що є приймачем слова. Джерелом інформації, окрім регістрів, можуть бути зовнішні входи пристрою, на які надходять дані, наприклад, з пам'яті, загальної шини системи, тощо.

Приклади операційних схем для виконання МО пересилання *RG*1*:=DATA, RG*1*:=RG*2, *RG*1[31…24]:=*RG*2[7…0] відповідно показані на рис. 1.11.

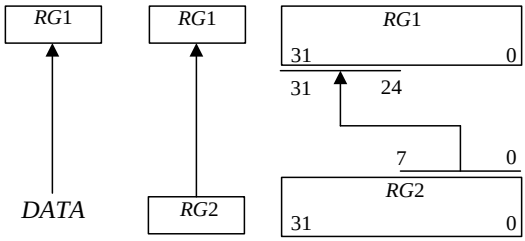
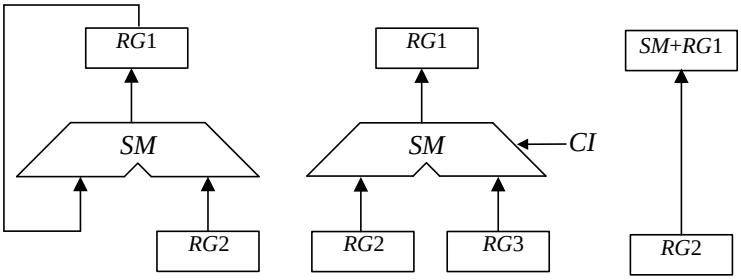


Рис. 1.11. Операційні схеми для виконання мікрооперацій пересилання даних

Для підсумовування слів в схемі використовують суматор. Приклади операційних схем, що відповідають мікрооперація *RG*1:=*RG*1+*RG*2 і *RG*1:=*RG*2+*RG*3+*CI* (де *CI* – перенос у молодший розряд суматора) відповідно показані на рис. 1.12, *а, б*. В схемах рис. 1.12, *а* і 1.12, *б* використовуються комбінаційні суматори, а в схемі рис. 1.12, *в* – накопичуючий суматор, який є композицією суматора і регістра.

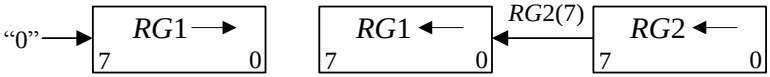
Мікрооперації інкременту ($CT := CT+1$) та декременту ($CT := CT-1$) виконуються на лічильнику.



а *б* *в*

Рис. 1.12. Операційні схеми для виконання мікрооперацій підсумовування чисел:
а, 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 *RG*1:=0.*r*[*RG*1] зображений на рис 1.13, *а*. Для реалізації зсуву, що відповідає операційній схемі на рис. 1.13, *б* необхідно виконати дві МО в одному такті *RG*2:=*l*[*RG*2].0 та *RG*1:=*l*[*RG*1].*RG*2(7).



а *б*

Рис. 1.13. 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. 1.14).

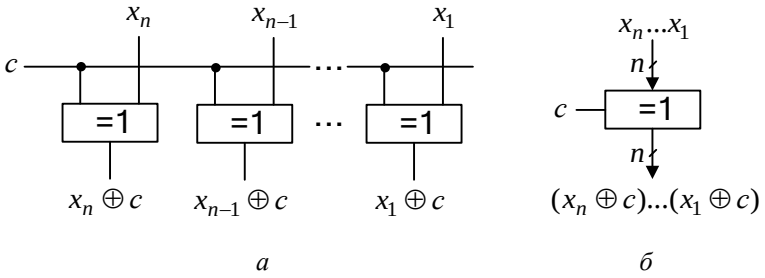


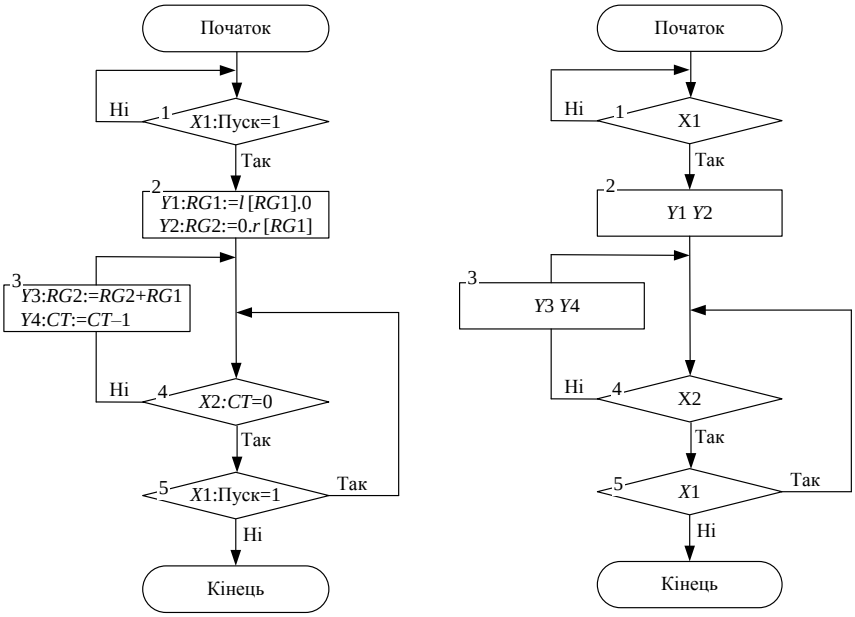
Рис. 1.14. Інвертування розрядів x_i двійкового числа:
а – логічна схема; *б* – умовне позначення (*с* – сигнал управління інвертуванням)

Послідовність мікрооперацій, що забезпечує задане перетворення інформації, називається мікроалгоритмом.

Для опису мікроалгоритмів використовують *графічні схеми мікроалгоритмів,* яки можуть бути описані мовами ГСА і ЛСА [6].

Мікроалгоритми можуть складатися як у змістовній, так і в закодованій формі. Для складання змістовного мікроалгоритму мікрооперації записують у змістовній формі, наприклад, через оператори присвоювання або їх ідентифікаторів (рис. 1.15). Для розробки такого мікроалгоритму достатньо скласти операційну схему пристрою, який виконує задані МО.

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.



а *б*

Рис. 1.15. Приклад запису мікроалгоритму:
а – з розширеним змістовним описом мікрооперацій; *б* – із скороченим описом мікрооперацій у вигляді ідентифікаторів

РОЗДІЛ 2

Арифметико-логічні пристрої

Арифметико-логічні пристрої призначені для виконання арифметичних та логічних операцій над машинними словами (числами, командами та машинними кодами інших об'єктів).

За структурою розрізняють АЛП з розподіленою та зосередженою логікою (інакше АЛП із закріпленими та загальними мікрооперація-ми). В АЛП першого типу апаратура для реалізації мікрооперацій розподілена між регістрами та закріплена за ними, тобто кожен регістр використовує власну логіку для виконання мікрооперацій. У пристроях другого типу всі логічні ланцюги об'єднані в арифметико-логічному блоці, а всі регістри реалізовані у вигляді надоперативного запам'ятовуючого пристрою.

За способом обміну між регістрами та арифметичним блоком АЛП із зосередженою логікою поділяються на послідовні, паралельні та послідовно-паралельні.

За формою подання чисел розрізняють АЛП із плаваючою комою, АЛП із фіксованою комою, та АЛП, що працюють як із плаваючою так і з фіксованою комою. В деяких ЕОМ передбачається режим цілих чисел, за яким кома фіксується після останнього розряду.

В залежності від основи системи числення розрізняють двійкові та десяткові АЛП. Відомі АЛП, що працюють в системах числення з основою 2^k, де k – додатне ціле число.

В залежності від часу, що відводиться на виконання окремих операцій, розрізняють АЛП синхронного, асинхронного та комбінованого типу. На виконання різних операцій в синхронних АЛП відводиться один і той самий час. В асинхронних АЛП на виконання кожної операції відводиться стільки тактів машинного часу, скільки потрібно, а виконання наступної операції розпочинається тільки за наявністю сигналу завершення поточної операції. Комбіновані АЛП поєднують простоту синхронних та швидкодію асинхронних АЛП. За цим всі операції поділяються на декілька групи, як правило на дві – однотактні та бага-

тотактні операції. Однотактні операції реалізуються за синхронним способом, багататактні – за асинхронним.

Набір операцій, що виконується в АЛП, є дуже важливою його характеристикою. Набір операцій повинен бути функціонально повним, для реалізації будь якого обчислювального алгоритму. З ціллю підвищення швидкодії і спрощення програмування указаний набір як правило має значну надлишковість. Кількість операцій коливається від декількох десятків до декількох сотень. За великим розмаїттям наборів операцій в їх складі звичайно є чотири основні арифметичні та найбільш важливі логічні операції, такі як порівняння, порозрядна кон'юнкція, тощо.

Виконання будь якої операції в АЛП зводиться до виконання послідовності мікрооперацій на регістрах, суматорах і таке інше. Послідовність МО, виконання яких приводить до виконання операції називають мікроалгоритмом цієї операції. В залежності від способу реалізації мікроалгоритмів розрізняють АЛП із схемним та мікропрограмним способом управління операціями (дивись розділ 3).

Виконання любих операцій в АЛП розпочинається з підготовчих МО добування операндів із основної пам'яті та фіксації їх у визначеному порядку в регістрах АЛП. Ділі, з ціллю спрощення викладення матеріалу, робота схем, що виконують окремі операції, зазвичай, розглядається з того моменту, коли всі підготовчі дії стосовно добування операндів та фіксації їх у ОП завершені.

2.1. Арифметико-логічні пристрої з розподіленою логікою

2.1.1. Основні способи множення чисел у прямих кодах

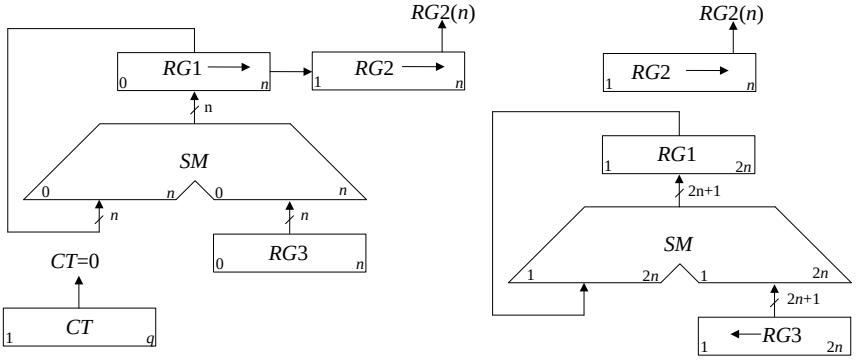
Принцип побудови пристроїв, що реалізують різні способи множення, показаний на рис. 2.1, де *RG*3 – регістр множеного, *RG*1 – регістр добутку, *RG*2 – регістр множника. Цифрами зазначені номери розрядів *SM* і регістрів, а стрілками показаний напрямок зсуву кодів у регістрах.

Цифри, що записані в молодших розрядах регістрів *RG*3 і *RG*1, при реалізації першого способу мають вагу 2^{-n}, а при реалізації інших способів – 2^{-2n}. Перед початком множення будь-яким способом регістр *RG1* встановлюється в нульовий стан. Підрахунок кількості циклів множення забезпечують лічильники *CT*, відповідно з чим обирається його розрядність *q*.

Під час множення *першим способом* (рис. 2.1, *а*) в першому такті *i*-го циклу аналізується значення *RG2(n)* – молодшого (*n*-го) розряду регістру *RG*2, в якому знаходиться чергова цифра множника. Вміст *RG*3 додається до суми часткових добутків, що знаходяться в регістрі RG1, якщо *RG2(n)*=1, або не додається, якщо *RG2(n)*=0. В другому такті здійснюється правий зсув в регістрах *RG*1 і *RG*2, що еквівалентно множенню їх вмісту на 2^{-1}. При зсуві цифра молодшого розряду регістру *RG*1 записується у вивільнюваний старший розряд регістру *RG*2. Після виконання *n* циклів молодші розряди 2*n*-розрядного добутку будуть записані в регістр *RG*2, а старші – у *RG*1.

Час множення, якщо не застосовуються методи прискорення операції, визначається виразом $t_м = n(t_п + t_з)$, де $t_п$ і $t_з$ – тривалості тактів підсумовування і зсуву відповідно.

Перед початком множення *другим способом* (рис. 2.1, *б*) множник *X* записують в регістр *RG*2, а множене *Y* – в молодші розряди регістру *RG*3 (тобто в регістрі *RG*3 установлюють $Y_0 = Y2^{-n}$). В кожному *i*-му циклі множення додаванням кодів *RG*3 і *RG*1 управляє цифра *RG*2(*n*), а в регістрі *RG*3 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 сполучити в часі з підсумовуванням (як правило, $t_п \geq t_з$). В цьому випадку $t_м = nt_п$. Завершення операції множення визначається за нульовим вмістом регістру *RG*2, що також приводить до збільшення швидкодії, якщо множник ненормалізований.



а *б*

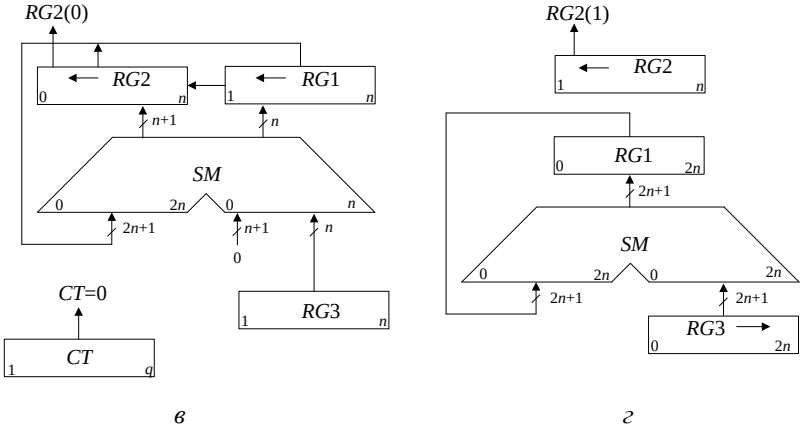


Рис. 2.1. Операційні схеми пристроїв для множення чисел: *а* – перший спосіб; *б* – другий спосіб; *в* – третій спосіб; *г* – четвертий спосіб

Під час множення *третім способом* (рис. 2.1, *в*) вага молодшого розряду *RG*3 дорівнює 2^{-2n}, тому код в регістрі *RG3* являє собою значення $Y2^{-n}$. На початку кожного циклу множення здійснюється лівий зсув в регістрах *RG*1 і *RG*2, а потім виконується додавання, яким управляє *RG*2(1). В результаті підсумовування вмісту *RG*3 і *RG*1 може виникнути перенос в молодший розряд регістру *RG*2. У старшій частині суматора, на якому здійснюється підсумовування коду *RG*2 з нулями, відбувається поширення переносу. Збільшення довжини *RG*2 на один розряд усуває можливість поширення переносу в розряди множника. Після виконання *n* циклів молодші розряди добутку будуть знаходитися в регістрі *RG*1, а старші – в регістрі *RG*2. Час множення третім способом визначається аналогічно першому способу і дорівнює $t_м = n(t_п + t_з)$.

Перед множенням *четвертим способом* (рис. 2.1, *г*) множник записують в регістр *RG*2, а множене – в старші розряди регістру *RG*3 (тобто в *RG*3 установлюють $Y_0=Y2^{-1}$). В кожнім циклі цифра *RG*2(1), що знаходиться в старшому розряді регістру *RG*2, управляє підсумовуванням, а в *RG*3 здійснюється правий зсув на один розряд, що еквівалентно множенню вмісту цього регістра на 2^{-1}. Час виконання множення четвертим способом складає $t_м = nt_п$, визначається аналогічно другому способу.

В ЕОМ при роботі з дробовими числами часто потрібно обчислювати не 2*n*, а тільки (*n*+1) цифр добутку й округляти його до *n* розрядів. В цьому випадку при реалізації другого способу можна зменшити довжину *SM* і *RG*1, а при реалізації четвертого – зменшити довжину *SM*, *RG*1 і *RG*3. 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 і відкидання всіх розрядів, розташованих правіше *n*-го. При цьому похибка стає знакозмінною, а максимальне абсолютне її значення не перевищує половини ваги молодшого розряду. Додаткового такту підсумовування для округлення не потрібно. Досить записати одиницю перед початком множення в той розряд регістру *RG*1, що після виконання множення залишається старшим розрядом, який відкидається.

У процесі формування суми часткових добутків код з регістру *RG*1 видається на суматор *SM*, а з виходів *SM* знову записується в регістр *RG*1. У зв'язку з цим при використанні потенційних елементів регістр *RG*1 будують на тригерах із внутрішньою затримкою. Характер управляючих сигналів і ланцюга, на який вони впливають, визначається конкретною теоретичною реалізацією вузлів і використовуваною елементною базою.

У операційних пристроях, що реалізують другий і четвертий способи множення, можна без пересилань кодів між регістрами обчислювати вирази вигляду $\sum X_i Y_i$, де $(i = \overline{1,n})$ для чого досить черговий результат операції залишати в регістрі *RG*2, який в цьому випадку повинен мати додаткові старші розряди.

У операційному пристрої, що реалізує третій спосіб, можна без пересилань обчислювати, наприклад, функції вигляду X_i. Для цього множник *X* перед початком обчислення записується в регістр *RG*3 і в молодші розряди регістру *RG*2, а потім (*i*–1) раз виконується операція множення з округленням проміжних результатів до *n* розрядів. Після кожної чергової операції регістр *RG*1 встановлюється в нульовий стан. Остаточний результат буде знаходитися в *n* молодших розрядах регістру *RG*2. 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 великих апаратурних витрат при реалізації.

2.1.2. Синтез арифметико-логічних пристроїв з розподіленою логікою

АЛП з розподіленою логікою застосовуються в спеціалізованих та проблемно-орієнтованих ЕОМ. Відрізняються від АЛП інших типів високою швидкодією, але мають досить обмежені функціональні можливості. Структура таких АЛП залежить від операцій, що вони виконують, причому для кожної системи операцій необхідно будувати окремий АЛП.

АЛП з розподіленою логікою складаються з двох функціональних частин (рис. 2.2):

- управляючий пристрій, що забезпечує формування всіх управляючих сигналів;
- операційний пристрій, забезпечує перетворення інформації та виконує мікрооперації над машинними словами.

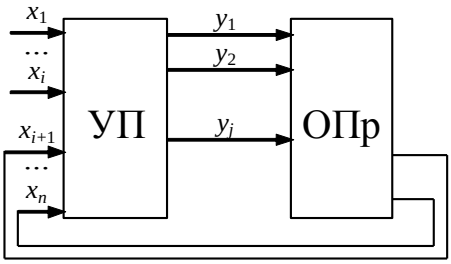


Рис. 2.2. Загальна структура АЛП

Побудова таких АЛП відбувається за наступними етапами:

1. Для кожної операції будується операційна схема та функціональний мікроалгоритм (Ф-микроалгоритм). Рекомендується обирати такі мікроалгоритми виконання операцій, що краще сполучаються, тобто вимагають однакового напрямку зсувів в регістрах, однакову розрядність регістрів, одні й ті самі джерела операндів суматорів і таке інше.

2. Обирається розрядність регістрів, лічильників. Виконується логічне моделювання роботи ОПр, наприклад, із застосуванням діаграми стану регістрів при виконанні МА з критичними значеннями операндів.

3. Розробляється функціональна та принципова схеми ОПр із зазначенням управляючих сигналів для кожного вузла пристрою.

4. Складається закодований структурний мікро алгоритм (С-мікроалгоритм) виконання заданих операцій.

5. Виконується синтез управляючого пристрою.

6. Складається функціональна та принципова схеми АЛП.

Приклад 2.1. Побудувати схему АЛП для реалізації операції множення чисел за першим способом.

Синтезувати схему, що дозволяє обчислити добуток $Z=Y\times X$ двох правильних дробів $Y= 0, y_1, y_2...y_n$ та $X= 0, x_1, x_2...x_n$. Вважати, що розрядність дробів $n = 16$.

Виконання завдання

Операційна схема, що реалізує перший спосіб множення, подана на рис. 2.3, де *RG*1 – регістр накопичення суми часткових добутків, *RG*2 – регістр множника, RG3 – регістр множеного, *RG*4 (*CT*) – лічильник циклів, *TC* – тригер переносу, *SM* – комбінаційний суматор. Регістри *RG*1 та *RG*2 реалізують мікрооперації зсуву, лічильник *RG*4 дозволяє формувати ознаку нуля – що визначає закінчення обчислення добутку. За нульовим вмістом регістру *RG*4 результат обчислення формується в регістрах *RG*1 та *RG*2.

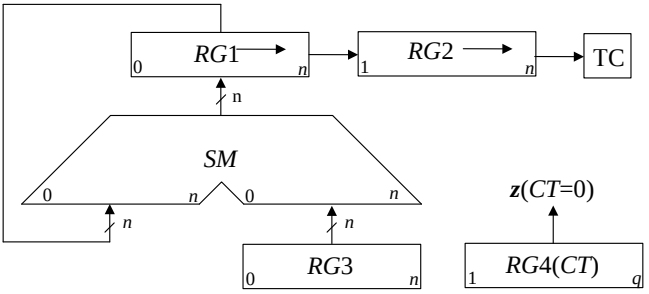


Рис. 2.3. Операційна схема множення

Зауваження. *Операційні схеми* застосовують для відображення апаратури, що застосовується для виконання послідовності заданих мікрооперацій. ОС містить всі функціональні частини операційного пристрою із зазначенням зв'язків між ними. За ОС виконання операції будують *структурну схему* ОПр.

Для розробленої операційної схеми побудуємо Ф-мікроалгоритм. Припустимо, що ОПр входить до складу АЛП із централізованим управлінням, отже робота цього блоку розпочинається із надходження сигналу “Пуск” від центрального блоку управління. Функціональний микроалгоритм зображений на рис. 2.4, де *TC* – стан тригера переносу, *z* – значення ознаки нуля в лічильнику циклів *RG*4.

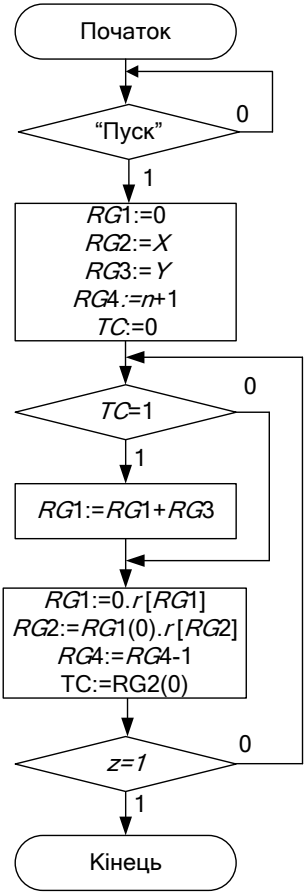


Рис. 2.4. Ф-мікроалгоритм множення чисел

Зауваження. Мікроалгоритми можна розглядати на функціональному та структурному рівнях. На *функціональному* рівні розглядають узагальнені МО, які не суперечать операційній схемі пристрою. При цьому можна не враховувати кількість тактів, необхід-

них для виконання МО. На *структурному* рівні операційна вершина відповідає одному такту перетворення інформації. С-мікроалгоритми повністю відповідають схемі пристрою з урахуванням елементної бази та тривалості управляючих сигналів. Для побудови С-мікроалгоритму необхідно отримати перелік МО в АЛП, що розробляється.

Логічне моделювання потактової роботи ОПр приведене в табл. 2.1

Значення операндів:

$Y = 5_{10} = 0101_2$;

$X = 7_{10} = 0111_2$;

$Z = 35_{10} = 00100011_2$.

Розрядність дробів $n = 4$.

Таблиця 2.1. Логічне моделювання роботи ОПр

№ такту	*RG*1	*RG*2	*TC*	*RG*3	*RG*4	*z*	МО
ПС	0000	0101	0	0111	0101	0	Початковий стан
1	0000	0010	1	0111	0100	0	*RG*1→, *RG*2→, *RG*4 – 1; *z* = 0
2	0000 +0111 0111 0011	0010 1001	1 0	0111 0111	0100 0011	0 0	*RG*1+ *RG*3 *RG*1→, *RG*2→, *RG*4 – 1; *z* = 0
3	0001	1100	1	0111	0010	0	*RG*1→, *RG*2→, *RG*4 – 1; *z* = 0
4	0001 +0111 1000 0100	1100 0110	1 0	0111 0111	0010 0001	0 0	*RG*1+ *RG*3 *RG*1→, *RG*2→, *RG*4 – 1; *z* = 0
5	0010	0011	0	0111	0000	1	*RG*1→, *RG*2→, *RG*4 – 1; *z* = 1

На підставі ОС множення та Ф-мікроалгоритму складемо перелік управляючих сигналів для всіх функціональних частин ОПр та побудуємо функціональну схему.

Перелік управляючих сигналів наведений в табл. 2.2, функціональна схема ОПр зображена на рис. 2.5.

Таблиця 2.2. Таблиця управляючих сигналів

Елемент	Мікрооперація	Управляючий сигнал
*RG*1	Скидання	*R*
	Запис	*W*
	Зсув вправо	*SR*
	Заповнення старшого розряду при зсуві вправо	*DR*
*RG*2	Запис	*W*
	Зсув вправо	*SR*
	Старший розряд при зсуві вправо	*DR*
*RG*3	Запис	*W*
*RG*4	Запис	*W*
	Декремент лічильника	*dec*
TC	Скидання	*R*
	Запис молодшого розряду множника у тригер переносу	*C*

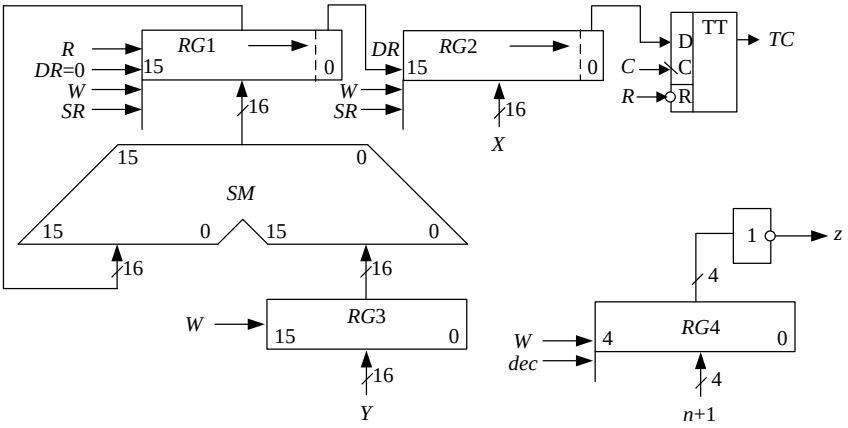


Рис. 2.5. Функціональна схема операційного пристрою

За побудованою функціональною схемою будуємо функціонально-структурний мікроалгоритм (ФС-мікроалгоритм), що зображений на

рис 2.6. Індекс указує до якої з функціональних частин пристрою множення належить управляючий сигнал.

Кодування сигналів управління та логічних умов наведене в табл. 2.3 – 2.4.

Для забезпечення перепаду сигналів управління SR_1, SR_2, dec, C_{TC} (вершину з цими сигналами охоплює петля рис. 2.6) необхідно ввести порожню додаткову вершину.

Закодований ФС-мікроалгоритм зображений на рис. 2.7, де управляючі сигнали та сигнали логічних умов відповідають рис. 2.6 та табл. 2.2 – 2.4.

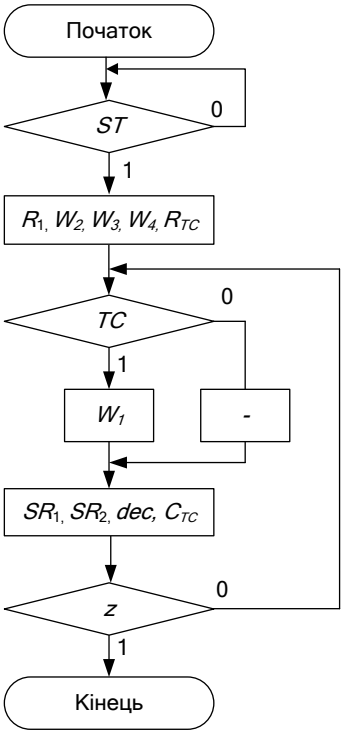


Рис. 2.6. Функціонально-структурний мікроалгоритм

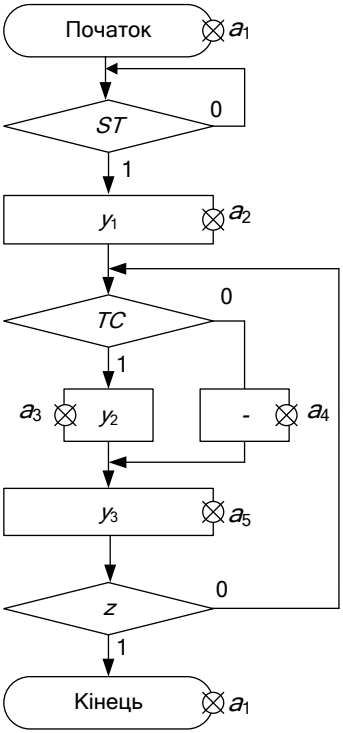


Рис. 2.7. Закодований функціонально-структурний мікроалгоритм

Таблиця 2.3. Кодування сигналів управління

Управляючі сигнали	Код
R_1	y_1
W_2	
W_3	
W_4	
R_{TC}	
W_1	y_2
SR_1	y_3
SR_2	
C_{TC}	
dec	

Отриманий закодований ФС-микроапгоритм є вихідним для здійснення синтезу управляючого пристрою.

Таблиця 2.4. Кодування логічних умов

Логічні умови	Код
Пуск	*ST*
Аналіз молодшого розряду множника	*TC*
Нульовий вміст лічильника	*z*

Для управління роботою ОПр застосуємо *пристрій управління з жорсткою логікою*, який реалізуємо у вигляді цифрового автомата Мура.

Розмітка ФС-микроалгоритма для автомата Мура наведена на рис. 2.7. Стани автомата позначені символами a_i. Часова діаграма роботи управляючого пристрою зображена на рис. 2.8. Часова діаграма відповідає потактовій роботі ОПр для прикладу, виконаного в табл. 2.1.

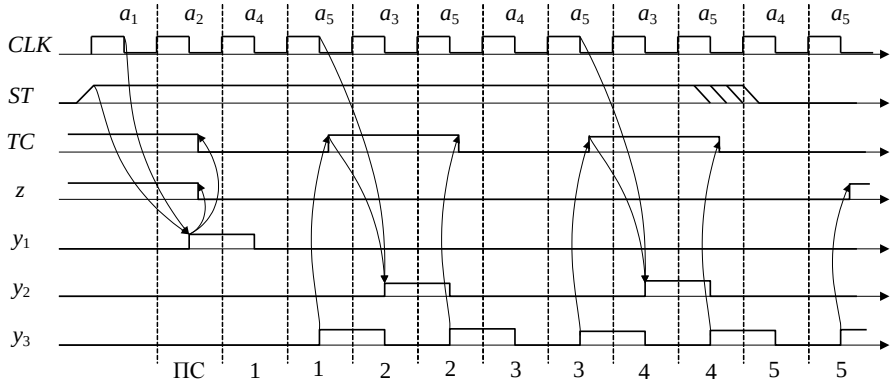


Рис. 2.8. Часова диаграма роботи пристрою управління

На рис. 2.9 зображена узагальнена структурна схема АЛП множення. Управляючі сигнали з виходів пристрою управління підключаються до входів відповідних функціональних частин ОПр.

Схема електрична функціональна АЛП для множення додатних чисел наведена у додатку А. Опис функціональної схеми наведений у прикладі 7.1.

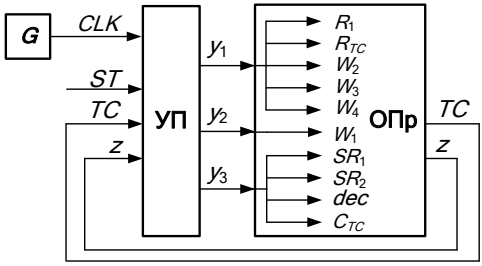


Рис. 2.9. Узагальнена структурна схема АЛП

2.2. Арифметико-логічні пристрої з зосередженою логікою

Можна визначити наступні різновиди АЛП із зосередженою логікою (з загальними мікрооперaціями):

- АЛП із двоспрямованою магістраллю та одноадресним НОЗП;
- АЛП із односпрямованими магістралями та одноадресним НОЗП;
- АЛП із односпрямованими магістралями та двоадресним НОЗП.

АЛП із зосередженою логікою мають широкі функціональні можливості і можуть виконувати велику кількість операцій. При цьому кількість операцій, що виконується, не впливає на складність пристрою. Зміна алгоритму роботи не потребує змін структури АЛП, а тягне за собою лише зміни МА, що зберігаються в пам'яті блока управління.

Усі регістри зосереджені у надоперативному запам'ятовуючому пристрої. Мікрооперації виконуються у процесі пересилки слів із одного регістру в інший через загальну логіку – арифметико-логічний блок.

Перетворення інформації в АЛП управляється словом, що називається мікрокомандой. Мікрокоманда має дві частини

F	AD

,

де *AD* – адресна частина, що визначає адресу регістру НОЗП, над яким необхідно виконати мікрооперацію;

F – функціональна частина, що задає тип МО в АЛП, спосіб модифікації регістру стану, напрямок зсуву акумулятора, комутацію зв'язків.

Структура АЛП із двоспрямованою магістраллю та одноадресним НОЗП зображена на рис. 2.10, де:

АЛБ – арифметико-логічний блок;
БРС – блок регістру стану;
РВ – регістр тимчасового зберігання;
РА – регістр акумулятор;
РС – регістр стану;
НОЗП – складається з регістрів (*R*0…*RN*);
ЛШ – локальна шина;
БД – буфер даних;
СМ – системна магістраль.

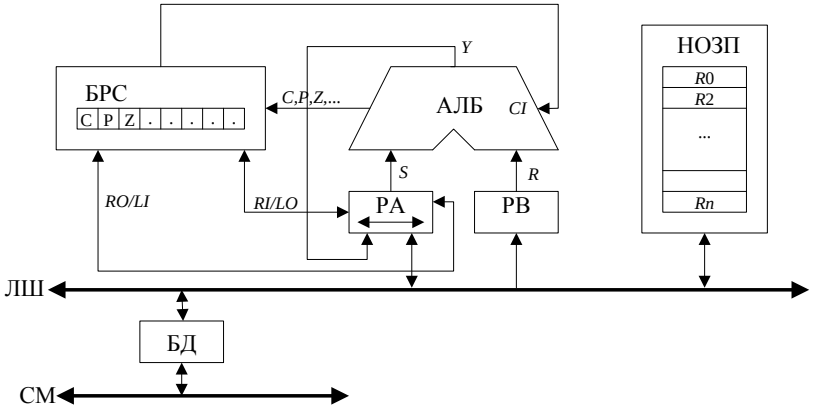


Рис. 2.10. Структура АЛП із двоспрямованою магістраллю та одноадресним НОЗП

АЛП є комбінаційною схемою призначеною для виконання арифметичних та логічних мікрооперацій над операндами *S* і *R*. Операнди *S* і *R* поступають на входи суматора з регістрів РА та РВ, результат виконання мікрооперації *Y* з виходу суматора записується у регістр РА. Регістри РА та РВ зв'язані з регістрами НОЗП по локальній шині. БРС застосовується для зберігання ознак. В якості вхідного переносу *CI* суматора можуть бути застосовані значення розрядів регістру стану (*C, Z* та інші). Вихідний перенос видається з АЛБ через вихід *CO* і може біти записаний в один із розрядів РС. Над вмістом акумулятором РА може бути виконана операція зсуву. При зсуві вправо (в сторону молодших розрядів) старший розряд заповнюється значенням, що надходить по ланцюгу *RI*/*LO*, а молодший (вихідний) розряд по ланцюгу *RO*/*LI* завантажується у регістр РС. При лівому зсуві вихідний молодший розряд

подається по ланцюгу *RO*/*LI*, а вихідний поступає в РС по ланцюгу *RI*/*LO*.

Локальна шина забезпечує обмін інформацією між вузлами НОЗП, причому, у кожному такті передача інформації по ЛШ може відбуватися лише в одному напрямку. Буфер даних БД застосовується для обміну інформацією між АЛП та системною магістраллю.

Мікрооперації, що виконуються в АЛП поділяються на арифметичні та логічні. Приклади МО, що реалізує АЛП даного типу, наведений у табл. 2.5.

Таблиця 2.5. Перелік мікрооперацій АЛП

Мнемоніка	**МО**
	Арифметичні операції
add	$Y := S + R + CI$
sub	$Y := S - R - 1 + CI$
sub	$Y := R - S - 1 + CI$
	Логічні операції
and	$Y := S \wedge R$
or	$Y := S \vee R$
xor	$Y := S \oplus R$

Структура мікрокоманди для даного типу АЛП має наступний вигляд:

F	*AD*

.

Для опису мікрокоманд можна застосовувати як змістовний запис МО так і запис у мнемонічному вигляді. Мнемонічна запис для арифметичної МК має наступний вигляд:

```
{<мнемоніка> [<оператор зсуву>], <приймач результату>,
<операнд 1>, <операнд 2>, CI}.
```

Наприклад, змістовний запис $R0 := r(R0 + R1)$ можна подати у мнемонічному вигляді `{add sr, r0, r0, r1, 0}`.

Формат логічної МК відрізняється від арифметичної відсутністю *CI*.

Приклад 2.2. Для АЛП із двоспрямованою магістраллю та одноадресним НОЗП побудувати мікроалгоритм виконання узагальненої дії $R0 := r(R1 + R2)$, де $R0$, $R1$, $R2$ – регістри НОЗП.

Виконання завдання

Мікроалгоритм виконання заданої операції зображений на рис. 2.11.

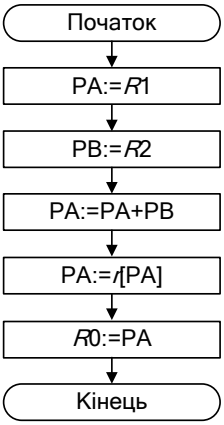


Рис. 2.11. Мікроалгоритм виконання операції в АЛП із двоспрямованою магістраллю

Перевагою АЛП розглянутого типу є простота реалізації, але такі АЛП характеризуються низькою швидкодією.

Для підвищення швидкодії застосовують декілька магістралей для обміну інформацію. До таких АЛП належать АЛП із односпрямованими магістралями із одноадренсним і двоадресним НОЗП.

Структура АЛП із односпрямованими магістралями та одноадресним НОЗП наведена на рис. 2.12, де:

S – зсувач даних;
MX – мультиплексор вибору другого операнду.

Зсувач *S* є комбінаційною схемою, що виконує модифікацію результату. Він дозволяє залишити результат виконання МО незмінним, або зсунутим вправо чи вліво. При виконанні зсувів із БРС може бути записана ознака у розряд, що звільняється при зсуві, а вихідний розряд може бути записаний в один із розрядів РС.

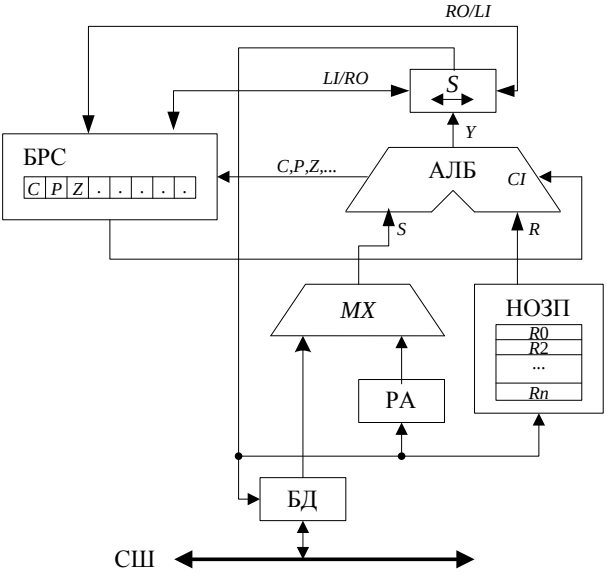


Рис. 2.12. Структура АЛП з односпрямованою внутрішньою магістраллю та одноадресним НОЗП

В АЛП із односпрямованою магістраллю та одноадресним НОЗП виконуються арифметичні та логічні МО (табл. 2.5). Одним з операндів є регістр НОЗП, другим операндом може бути або вміст акумулятору РА, або зовнішні по відношенню до АЛП дані, що подаються з системної магістралі через буфер даних БД.

Структура мікрокоманди має наступний вигляд

F	*AD*

.

Типи зсувів, що, зазвичай, реалізовані в АЛП наведені в табл. 2.6.

Таблиця 2.6. Типи зсувів в АЛП

Мнемоніка	**Зсув**	
sl	C ← S ←	Зсув вліво логічний
slc	C ← S ←	Зсув вліво циклічний
sr	→ S → C	Зсув вправо логічний
src	→ S → C	Зсув вправо циклічний

Приклад 2.3. Для АЛП із односпрямованою магістраллю та одноадресним НОЗП побудувати мікроалгоритм виконання узагальненої дії $R0 := r(R1 + R2)$, де $R0$, $R1$, $R2$ – регістри НОЗП.

Виконання завдання

Для виконання заданої операції можна застосувати мікроалгоритм зображений на рис. 2.13.

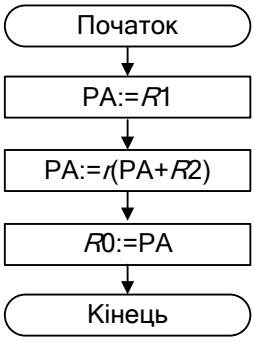


Рис. 2.13. Мікроалгоритм виконання операції в АЛП із односпрямованою магістраллю та одноадресним НОЗП

Як видно застосування даного типу АЛП дозволяє зменшити на два кількість кроків обробки інформації відносно АЛП, що розглядалося у прикладі 2.2.

Структура АЛП із односпрямованою магістраллю та двоадресним НОЗП приведена на рис. 2.14.

Всі регістри в НОЗП мають однакові можливості.

Для управління АЛП застосовуються мікрокоманда наступної структури:

F	$AD1$	$AD2$

,

де $AD1$ і $AD2$ – адреси регістрів НОЗП над вмістом яких виконується мікрооперація. Поле $AD1$ задає адресу першого операнда та адресу приймача результату, а $AD2$ – адресу другого операнда. Результат мікрооперації записується в НОЗП по першій або другій адресі.

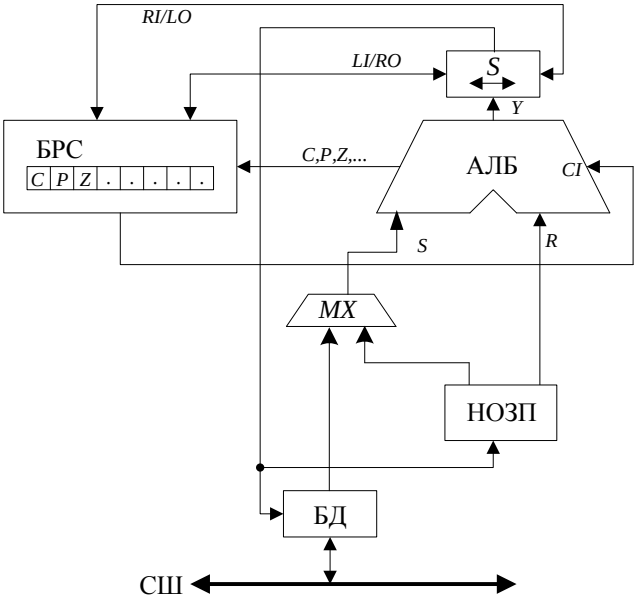


Рис. 2.14. Структура АЛП із однонаправленою магістраллю та двоадресним НОЗП

Приклад 2.4. Для АЛП із односпрямованою магістраллю та двоадресним НОЗП побудувати мікроалгоритм виконання узагальненої дії $R0 := r(R1 + R2)$, де $R0$, $R1$, $R2$ – регістри НОЗП.

Виконання завдання

Задану операцію можна виконати за допомогою мікроалгоритму зображеного на рис. 2.15.

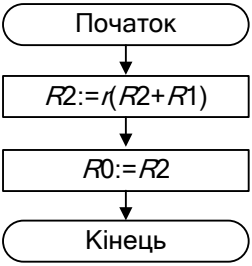


Рис. 2.15. Мікроалгоритм виконання операції в АЛП із односпрямованою магістраллю та двоадресним НОЗП

Таким чином, АЛП з двоадресним НОЗП та односпрямованою магістраллю дозволяє досягти більшої швидкодії по відношенню до АЛП з двоспрямованою магістраллю, але вони мають більш складну структуру.

Загальним недоліком АЛП із зосередженою логікою є неможливість суміщення мікрооперацій, бо для їх виконання застосовується загальний АЛБ. Тому основні напрями підвищення ефективності АЛП із зосередженою логікою направлені на вирішення проблеми суміщення мікрооперацій у часі та обробку слів подвійної довжини. Для цього у схему АЛП вводять додаткові вузли (регістри, лічильники, мультиплексори) на яких виконуються окремі МО. АЛП, що побудовані за такими принципами, називаються комбінованими, тобто вони поєднують у собі ознаки АЛП із розподіленою та зосередженою логікою.

Один із способів підвищення функціональних можливостей АЛП пов'язаний із введенням додаткового регістру розширювача *RQ* (рис. 2.16). Це дозволяє виконувати одночасно МО зсуву *RQ,* одночасно з будь-якою іншою МО в АЛБ, а також виконувати зсув слів подвійної довжини. Наприклад, можна виконати одночасно дві наступні МО: $R0:=l(R0+R1)$ та $RQ:=lRQ$.

За реалізації структур АЛП важливо, щоб включення додаткової апаратури не впливало на швидкодію системи. Схема, зображена на рис. 2.17 дозволяє виконувати зсув *RQ* на два розряди $RQ:=l2RQ$ за один такт, але при цьому тривалість такту АЛП, у порівнянні зі схемою на рис. 2.16, збільшується на час виконання МО у зсувачі *SQ*. Порівнювати швидкодію конкретних АЛП необхідно з врахуванням тривалості мікро операцій. На структурних схемах АЛП, зображених на рис. 2.16 – 2.17, умовно не показані БРС.

Щодо розробки мікроалгоритмів функціонування АЛП із зосередженою логікою можна визначити наступні етапи:

1. Відповідно до цільової функції проектування обрати структуру АЛП.

2. Для кожної операції, що виконується в АЛП розробляється операційна схема та структурний мікроалгоритм.

3. Виконується моделювання виконання операцій із застосуванням цифрових діаграм стану регістрів із критичними значеннями операндів.

4. Розробляється функціональний змістовний мікроалгоритм або мікропрограма у мнемонічних кодах, що є вихідними даними для побудови пристрою управління.

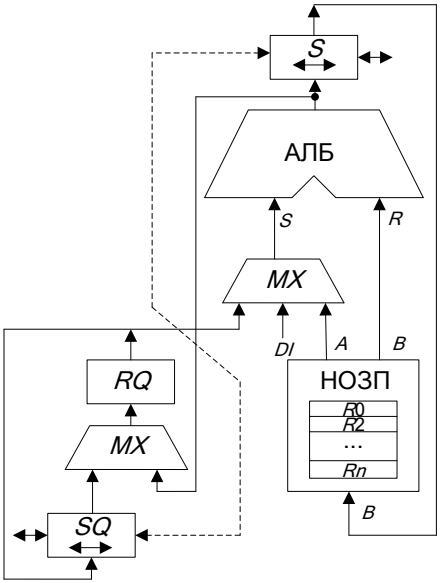


Рис. 2.16. Структура АЛП з регістром розширювачем

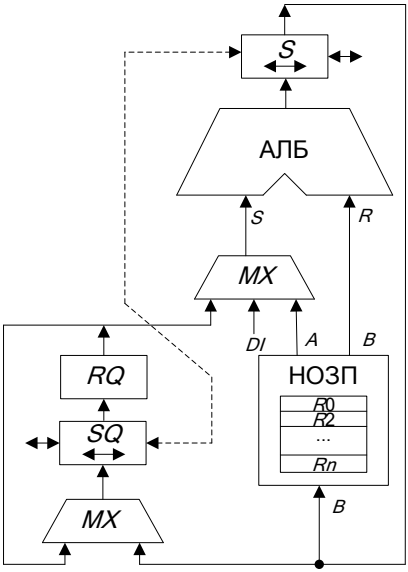


Рис. 2.17. Структура АЛП із зсувом на два розряди

Приклад 2.5. Розробити мікроалгоритм управління АЛП із зосередженою логікою для виконання операції множення за третім способом.

Виконання завдання

Для виконання задачі обираємо АЛП з односпрямованою магістраллю та двоадресним НОЗП, який дозволяє зменшити кількість мікрооперацій для перетворення даних.

Операційна схема множення за третім способом зображена на рис. 2.18. Операція множення виконуються із старших розрядів множника, за рахунок зсуву його вліво, сума часткових добутків також зсувається вліво, множене – нерухоме. Множник зберігається у регістрі $R1$, множене – у регістрі $R3$. У регістрі $R2$ накопичується сума часткових добутків. Регістр $R4$ застосовується як лічильник циклів. Розряд C регістру стану застосовується для збереження ознак при зсуві регістрів $R1$ та $R2$. Формування чергової суми часткових добутків у регістрі $R2$ відбувається із поширенням переносу у регістр $R1$. При цьому на сума-

торі *SM*1 підсумовується вміст регістру *R*1 з нулями, на вхід суматора *CI* подається розряд *C* регістру стану, де збережений вихідний перенос суматора *SM*2. Детальний опис принципу виконання операції множення за третім способом викладений у розділі 2.1.1.

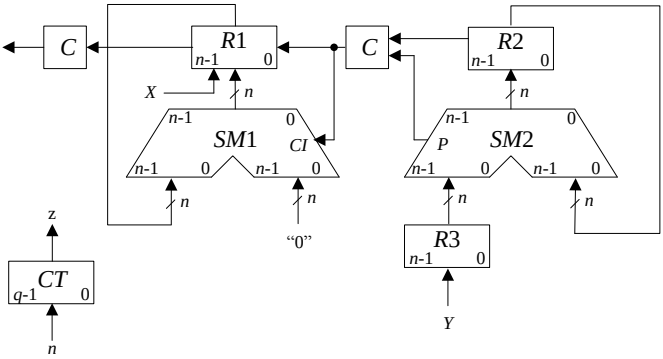


Рис. 2.18. Операційна схема операції множення

Цифрова діаграма операції множення для виконання операції $Z:=Y*X$, де X = 0,0101, Y = 0,0111, $0 < Y, X < 1$ наведена на рис. 2.19. Мікроалгоритм управління АЛП, розроблений на підставі операційного пристрою та цифрової діаграми операції множення, зображений на рис. 2.20.

№ такту	*C*	*RG*1 (*X*)	*RG*2 (*Z*)	*RG*3 (*Y*)	*RG*4	*z*	МО
ПС	0	0101	0000	0111	100	0	Початковий стан
1	0 0	 1010	0000	0111	 011	 0	←*RG*2 ←*RG*1, C=0 *RG*4 – 1; *z* = 0
2	0 1 0	 0100 0100 +0000 0100	0000 0000 +0111 0111	0111	 010	 0	←*RG*2 ←*RG*1, C=1 *RG*2+ *RG*3 *RG*4 – 1; *z* = 0
3	0 0	 1000	1110	0111	 001	 0	←*RG*2 ←*RG*1, C=0 *RG*4 – 1; *z* = 0
4	1 1	 0001	1100	0111			←*RG*2 ←*RG*1, C=1

	1	0001 +0001 **0010**	1100 +0111 **0011**		000	1	*RG*2+ *RG*3 Поширення переносу *RG*4 – 1; *z* = 0

Рис. 2.19. Логічне моделювання роботи АЛП

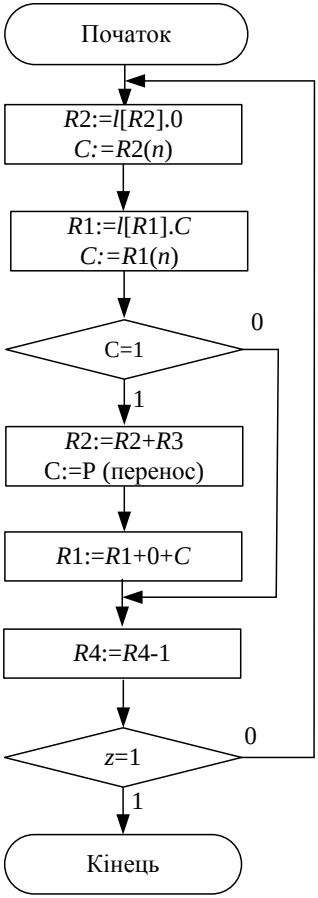


Рис. 2.20. Управляючий мікроалгоритм виконання операції множення

2.3. Лабораторна робота 1

Ціль роботи: Вивчити основні методи множення чисел у прямих кодах і способи їх апаратної реалізації, одержати навички в проектуванні й налагодженні схем управління операційними пристроями з розподіленою логікою.

Підготовка до лабораторного заняття

1. Розробити структурну схему операційного пристрою та змістовний мікроалгоритм множення додатних чисел відповідно до завдання наведеного у табл. 2.7), де $a_6,\ldots,a_1$ – молодші розряди двійкового номера залікової книжки. Для побудови схеми використати комбінаційний суматор, регістр-лічильник циклів та асинхронні регістри, що мають входи управління зсувами і занесенням інформації. На схемі повинні бути зазначені розрядність регістрів та шин.

2. Розробити функціональну схему операційного пристрою.

3. Виконати логічне моделювання роботи операційного пристрою за допомогою цифрової діаграми із зазначеними викладачем значеннями операндів.

4. Здійснити синтез пристрою управління, тип управляючого автомату обрати із табл.2.9. Пам’ять автомата реалізувати на тригерах, тип яких обрати з табл. 2.8. Ураховувати, що мікрооперації на регістрах виконуються за зворотним перепадом управляючих сигналів.

5. Побудувати часові діаграми роботи автомата для кожної комбінацій значень логічних умов.

Таблиця 2.7. Варіанти завдання

a_6	a_5	a_4	Спосіб множення	Розрядність операндів
0	0	0	1	16
0	0	1	2	8
0	1	0	3	16
0	1	1	4	8
1	0	0	1	8
1	0	1	2	16
1	1	0	3	8
1	1	1	4	16

Таблиця 2.8. Варіанти завдання

a_3	a_2	Тип тригера
0	0	JK
0	1	T
1	0	RS
1	1	D

Таблиця 2.9. Варіанти завдання

a_1	Тип автомата
0	Мили
1	Мура

Порядок виконання роботи

1. В моделюючій програмі ПРОГМОЛС 2.0 побудувати схему операційного пристрою для множення чисел та доповнити її схемою управляючого автомата. На першому етапі виходи автомата до входів операційного пристрою не підключати. Налагодити окремо схему операційного пристрою та схему управляючого автомата в синхронному режимі. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.

2. Підключити до управляючих входів операційного пристрою виходи автомата. Зробити комплексне налагодження схеми в синхронному режимі й переконатися в правильності одержання результату.

3. Перейти до асинхронного моделювання. Дослідити зазначені викладачем часові параметри схеми.

Зміст звіту

Звіт з лабораторної роботи повинен включати короткі теоретичні відомості, необхідні для виконання лабораторної роботи; структурні та функціональні схеми; таблиці та діаграми, отримані при виконанні теоретичного завдання, а також у процесі моделювання схем; висновки за роботою.

Контрольні питання

1. Охарактеризуйте чотири основних методи множення чисел.
2. Як розрахувати розрядність вузлів операційного пристрою?
3. Визначить поняття: операція, мікроалгоритм, мікрооперація.
4. Що таке мікроалгоритм операції?
5. Визначте основне призначення арифметико-логічного пристрою в ЕОМ.

6. Наведіть типи арифметико-логічних пристроїв, та їх основні відмінності.

7. Охарактеризуйте основні етапи проектування арифметико-логічного пристрою з розподіленою логікою.

8. Що відображує операційна схема виконання операції?

9. Що відображує функціональна схема пристрою?

10. В чому відмінність функціонального та структурного мікроалгоритмів?

11. Напишіть вирази, що визначають закони функціонування автоматів Милі та Мура.

12. У чому відмінність автоматів Милі та Мура?

13. Намалюйте узагальнену структурну схему управляючого автомата.

14. Охарактеризуйте основні етапи проектування управляючого автомата.

15. Як перейти від змістовного мікроалгоритму до закодованого мікроалгоритму?

16. Як побудувати граф автомата?

17. Як здійснюється оцінка станів автомата?

18. Як визначити необхідну тривалість управляючих сигналів?

19. Від чого залежить кількість тригерів, необхідних для побудови пам'яті автомата?

20. Як скласти структурну таблицю автомата?

21. Складіть таблицю переходів для *JK*-, *RS*-, *T*- і *D*-тригерів. Наведіть їх умовне графічне позначення.

22. Чи можливий перехід автомата в стан, що непередбачений графом, при використанні тригерів із внутрішньою затримкою (тригерів, керованих рівнем сигналів)?

23. Коли можливе виникнення помилкових управляючих сигналів (що непередбачені графом автомата) і чим визначається їх тривалість?

24. Наведіть способи усунення короткочасних помилкових управляючих сигналів.

25. У чому суть «протигоночного» кодування станів автомата?

26. Як забезпечити перепад управляючого сигналу у випадку, коли операторну вершину з цим сигналом охоплює «петля»?

27. Як визначити час переходу автомата з одного стану в інший?

РОЗДІЛ 3

Синтез блоків мікропрограмного управління

3.1. Призначення та класифікація блоків управління

Блоки управління є складовою частиною пристрою управління, що входить у склад процесору та забезпечує виконання програм у ЕОМ.

Основне призначення БУ – формування всіх управляючих сигналів, які забезпечують виконання кожної команди в ЕОМ.

Можна проводити класифікацію БУ за різними ознаками.

За функціональними можливостями:

- БУ із *жорсткою логікою*, призначені для реалізації певного набору мікроалгоритмів; такі БУ реалізують у вигляді управляючих автоматів;
- БУ із *гнучкою логікою*, як правило, *з мікропрограмним управлінням*; такі БУ дозволяють забезпечити модифікацію та запис нових мікропрограм, змінювати логіку управління в залежності від записаної у пам'яті мікропрограми.

За способами управління розрізняють централізовані та децентралізовані БУ:

- *централізовані* – мікропрограми формуються в одному пристрої для всіх пристроїв у системі. Такі БУ забезпечують виконання всіх МО послідовно у часі, що приводить до зменшення швидкодії системи.
- *децентралізовані* – кожен пристрій у системі має свій БУ, роботу яких синхронізує централізований ПУ. Такий спосіб забезпечує виконання мікрооперацій водночас в різних складових частинах системи, що приводить до збільшення швидкодії, але й збільшує апаратурні витрати. У сучасних ЕОМ більш поширені децентралізовані БУ.

Розділяють синхронні та асинхронні БУ.

- *синхронні* – для виконання кожної МО виділяються однакові проміжки часу, що дорівнюють максимальній тривалості МО;

- *асинхронні* – для виконання кожної МО виділяється необхідний для виконання цієї МО проміжок часу.

Слід зазначити, що з точки зору швидкодії асинхронні БУ є більш швидкодіючими, але потребують збільшення апаратної складності.

На практиці застосовуються *комбіновані* БУ – МО поєднуються в групи за часом виконання. В одну групу уключають МО, що найбільш близькі за часом виконання і для всіх МО в цій групі виділяють час що дорівнює максимальній тривалості МО у межах групи.

Приклад 3.1. Для заданого МА тривалість t_i кожної МО y_i задана графічно (рис.3.1), де τ – такт машинного часу. На рис.3.2 наведені часові діаграми виконання заданого МА для різних способів управління.

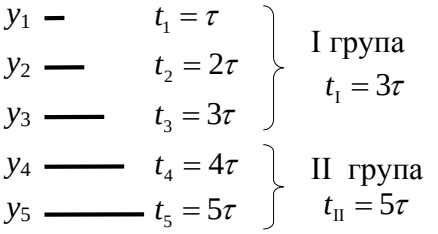


Рис. 3.1. Задана тривалість мікрооперацій

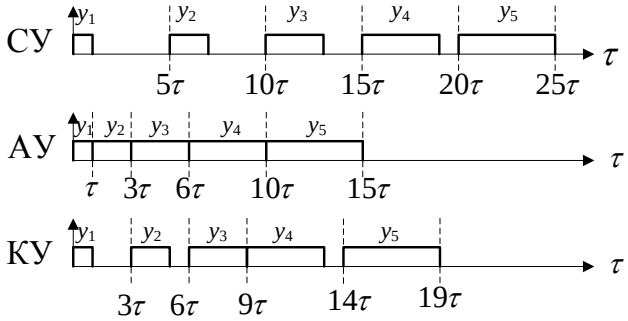


Рис. 3.2. Виконання МО для різних способів управління:
СУ – синхронного; АУ – асинхронного; КУ – комбінованого.

3.2. Блоки мікропрограмного управління

Кожній команді, яка записана у основній пам’яті ЕОМ, відповідає мікропрограма, що зберігається в пам'яті блоку мікропрограмного управління.

Мікропрограма – це зв’язаний список мікрокоманд, виконання яких забезпечує виконання заданої команди.

Мікрокоманда це інформаційне слово, що містить наступну інформацію:

- управляючі сигнали;
- тривалість управляючих сигналів;
- інформацію щодо формування адреси наступної МК.

БМУ функціонує у відповідності з *принципом мікропрограмного управління*, що полягає в наступному.

Під час виконання мікропрограми в кожному такті із постійної пам'яті БМУ зчитується та розшифровується чергова мікрокоманда. В результаті виконання мікрокоманди формуються управляючі сигнали необхідної тривалості, що поступають на всі функціональні частини обчислювальної системи, а також формується адреса наступної мікрокоманди.

Можна виділити наступні етапи виконання команди в обчислювальній системі.

1. *Вибірка команди.* З ОП зчитується команда в регістр команд процесора, для чого виконується відповідна МП, що записана у пам’ять БМУ.

2. *Розпакування команди*. Команда розшифровується (аналізуються поля слова команди, визначаються операнди), що забезпечується виконанням відповідної МП.

3. *Виконання операції.* Виконується МП виконання заданої операції над визначеними операндами.

4. *Формування адреси наступної команди.* Відповідна МП формує адресу наступної команди у лічильнику команд.

Спрощена структурна схема БМУ наведена на рис. 3.3.

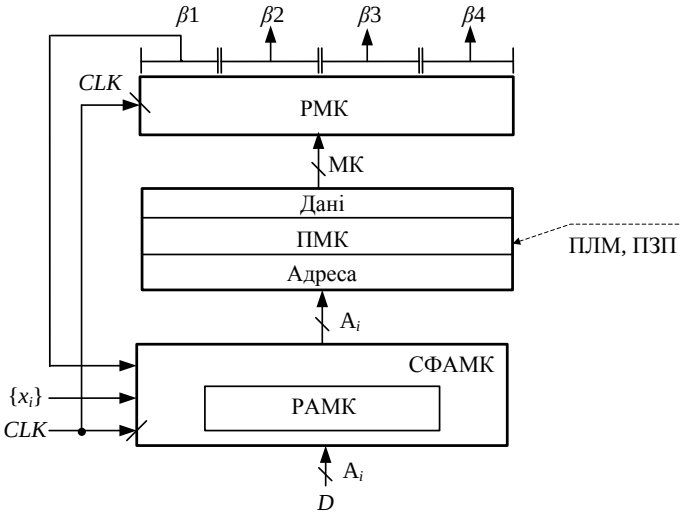


Рис. 3.3. Структурна схема БМУ

Основні функціональні частини БМУ:

РАМК – регістр адреси МК;
СФАМК – схема формування адреси МК;
ПМК – пам'ять МК;
РМК – регістр МК;
A_i – адреса МК;
CLK – синхросигнал;
$\{x_i\}$ – логічні умови;
D – вхід завдання початкової адреси мікропрограми.

МК розміщуються у пам’яті мікрокоманд. На рис. 3.4 наведений формат мікрокоманди.

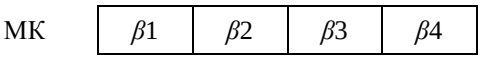


Рис. 3.4. Формат мікрокоманди

Слово МК складається з чотирьох зон:

$\beta 1$ – зона формування адреси наступної МК;
$\beta 2$ – зона управляючих сигналів;
$\beta 3$ – зона визначення тривалості управляючих сигналів;
$\beta 4$ – зона службових розрядів.

У кожному такті за синхросигналом *CLK* адреса мікрокоманди поновлюється у РАМК і надходить на адресний вхід ПМК. За адресою, що надійшла в ПМК, обирається відповідна мікрокоманда і видається на вихід даних ПМК. Слово мікрокоманди записуються у РМК за зворотним перепадом синхросигналу *CLK*.

Сигнали зони $\beta 2$ управляють вузлами комп'ютера, зони $\beta 3$ – визначають тривалість цих сигналів, сигнали зони $\beta 1$ разом із логічними умовами $\{x_i\}$ поступають на вхід СФАМК і формують адресу наступної МК. За черговим сигналом *CLK* адреса наступної МК буде сформована у РАМК. Зона $\beta 4$ використовується для виконання допоміжних функції, наприклад контролю апаратури.

3.2.1. Структура зони управляючих сигналів $\beta 2$

Зона $\beta 2$ застосовується для кодування управляючих сигналів. Існують два основні способи кодування управляючих сигналів у зоні $\beta 2$:

- *горизонтальне мікропрограмування*, яке також називають мінімальним кодуванням управляючих сигналів;
- *вертикальне мікропрограмування*, яке також називають максимальним кодуванням управляючих сигналів.

Під час мінімального кодування кожен управляючий сигнал відображується одним розрядом слова мікрокоманди (рис. 3.5), де *R, W, I, O* – відповідно управляючі сигнали зчитування, запису, вводу та виводу інформації, що показані у якості прикладу.

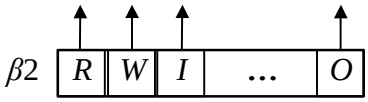


Рис. 3.5. Структура зони $\beta 2$ при мінімальному кодуванні (горизонтальне мікропрограмування)

За мінімального способу кодування довжина зони $\beta 2$ дорівнює

$$n_{\beta 2} = N_{УС}, \qquad (3.1)$$

Де $N_{УС}$ – кількість управляючих сигналів.

Під час максимального кодування розряди зони $\beta 2$ формуються за допомогою коду $\alpha_i ... \alpha_1$ на дешифраторі (рис. 3.6).

Довжина зони $\beta 2$ в цьому випадку дорівнює

$$n_{\beta 2} =]\log_2 N_{УС}[. \qquad (3.2)$$

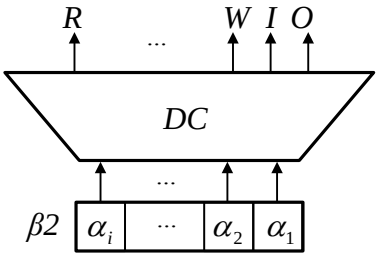


Рис. 3.6. Структура зона $\beta 2$ при максимальному кодуванні (вертикальне мікропрограмування)

Недоліки горизонтального мікропрограмування полягають в тому, що кодування великої кількості УС потребує більшої довжини слова МК. До переваги слід віднести можливість формувати водночас будь яку кількість УС, що приводить до сумісного виконання декількох МО в одному такті.

До переваг вертикального мікропрограмування відносять значне скорочення довжини зони $\beta 2$. Недоліки – в кожному такті можна сформувати тільки один УС, що ускладнює сумісне виконання МО.

На практиці зазвичай використовують *комбінований спосіб кодування* УС – сигнали поєднують у групи, таким чином, що сигнали, які мають бути виконані водночас розміщуються в різних групах. Усередині групи використають ВМ, а між групами ГМ.

На рис. 3.7 наведена можлива реалізація зони $\beta 2$ при комбінованому способі кодування.

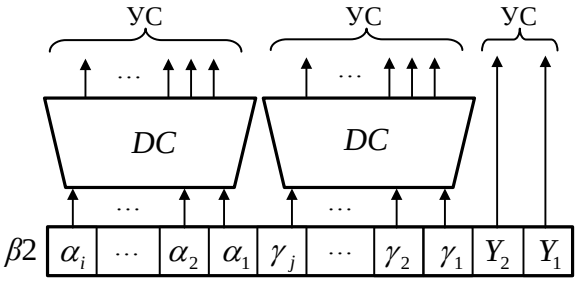


Рис. 3.7. Структура зони $\beta 2$ при комбінованому способі кодування

Приклад 3.2. Побудувати структуру та карту програмування зони управляючих сигналів $\beta 2$ для ефективної реалізації МА, що заданий у вигляді лінійної схеми МА:

$$\Pi\, y_1 y_2 (\, y_1 y_2 y_3) \, y_4 y_5 (\, y_2 y_3 y_5 \,) \, y_6 y_7 \,\text{К}$$

Виконання завдання

Для формування зони $\beta 2$ застосуємо комбіноване мікропрограмування. Розподілимо управляючі сигнали на групи так, щоб сигнали які формуються водночас розміщувались у різних групах, отримаємо:

I	II	III
y_1	y_2	y_3
y_5		
y_4		
y_6		
y_7		

Розрахуємо довжину зони $\beta 2$.

Для кодування сигналів першої групи використаємо дешифратор. За виразом (2.2) розрахуємо довжину зони:

$$n_{DC} =]\log_2 6[= 3.$$

Враховуючи, що для кодування сигналів груп II та III необхідно ще два розряди зони $\beta 2$, отримаємо:

$$n_{\beta 1} = 5.$$

Кодування входів дешифратора наведене в табл. 3.1. Карта програмування зони $\beta 2$ – в табл. 3.2.

Таблиця 3.1. Таблиця кодування УС

α_3	α_2	α_1	УС
0	0	0	немає сигналів
0	0	1	y_1
0	1	0	y_5
0	1	1	y_4
1	0	0	y_6
1	0	1	y_7

Таблиця 3.2. Карта програмування

№ такту	УС	$\beta 2$				
		α_3	α_2	α_1	y_2	y_3
1	y_1	0	0	1	0	0
2	y_2	0	0	0	1	0
3	y_1, y_2, y_3	0	0	1	1	1
4	y_4	0	1	1	0	0
5	y_5	0	1	0	0	0
6	y_2, y_3, y_4	0	1	0	1	1
7	y_6	1	0	0	0	0
8	y_7	1	0	1	0	0

Структурна схема зони $\beta 2$ зображена на рис. 3.8.

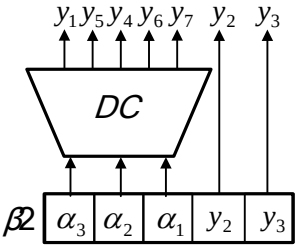


Рис. 3.8. Структурна схема зони $\beta 2$

3.2.2. Структура зони визначення тривалості управляючих сигналів $\beta 3$

Зона $\beta 3$ відповідає за тривалість виконання мікрооперацій. Під час асинхронного та комбінованого способу управління мікрооперація виконується за один або декілька тактів, тобто необхідно забезпечити необхідну затримку управляючого сигналу при виконанні МО.

Найбільш розповсюдженим способом є використання лічильника тактів, у який заноситися константа, що визначає час затримки УС.

У кожному такті виконується декремент лічильника (або інкремент, якщо у лічильнику записане від'ємне число), за нульовим вмістом якого дозволяється зміна інформації в РАМК і відбувається формування наступних УС.

З точки зору апаратної реалізації – частина РМК (зона $\beta 3$) виконується у вигляді лічильника тактів *CT* (рис. 3.9).

$\beta 1$	$\beta 2$	$\beta 3$	$\beta 4$
RG	*RG*	*CT*	*RG*

Рис. 3.9. Апаратна реалізація регістру мікрокоманди

Довжина зони $\beta 3$ визначається за формулою

$$n_{\beta 3} =]\log_2 k[+ 1, \qquad (3.3)$$

де k – максимальна затримка управляючих сигналів у тактах, один додатковий розряд (+1) застосовується для урахування знакового розряду (ЗР).

Структурна схема БМУ з урахуванням зони затримки управляючих сигналів зображена на рис. 3.10.

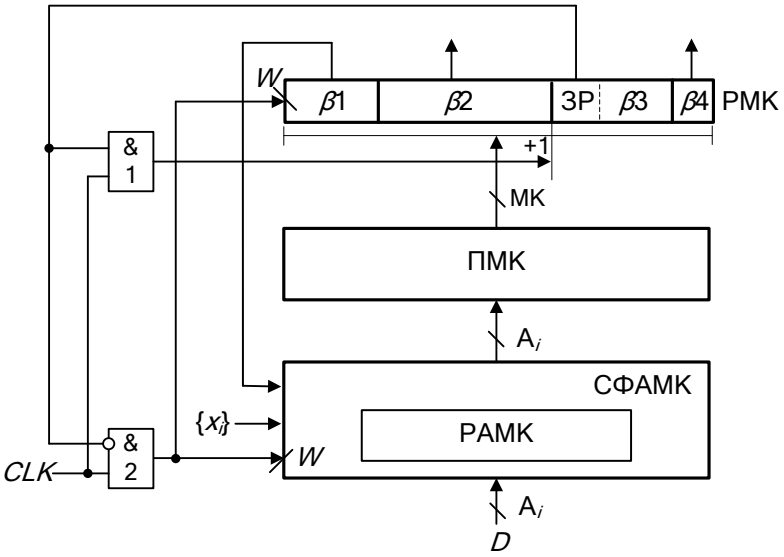


Рис. 3.10. Структурна схема БМУ з урахуванням зони затримки УС

Принцип функціонування. Кількість тактів, на які необхідно затримати МО, записують у двійковому доповнювальному коді від'ємного числа в лічильник *СТ*. Це означає, що в ЗР буде записана одиниця, якщо затримка не дорівнює нулю. У кожному такті до вмісту лічильника *СТ* додається одиниця, поки вміст лічильника не дорівнюватиме нулю. За нульовим кодом відбувається завантаження адреси наступної МК у РАМК. До схеми додаються два логічні елементи І (ЛЕ1 та ЛЕ2). За наявністю одиниці у ЗР лічильника ЛЕ2 блокує сигнал *CLK*, що має поступити на вхід *W* СФАМК. З виходу ЛЕ1 сигнал поступає на вхід (+1) лічильника. За нульовим вмістом лічильника, тобто коли ЗР дорівнює нулю, навпаки є заблокованим ЛЕ1 і сигнал *CLK* поступає на вхід *W* СФАМК, що дозволяє формування адреси наступної МК.

Приклад 3.3. Для БМУ з асинхронним принципом формування управляючих сигналів розробити структуру й карту програмування зони $\beta 3$.

Вихідні дані

- максимальна тривалість МО – 25 тактів,

- карту програмування побудувати для МО тривалістю 5 тактів.

Виконання завдання

Визначимо довжину зони $\beta 3$ за виразом (3.3):

$$n_{\beta 3} =]\log_2 24[+1=5+1=6,$$

де $\Delta = 24\tau$ – максимальна затримка МО.

Якщо тривалість чергової МО $t_i = 5\tau$, то час затримки дорівнюватиме $\Delta = 4\tau$.

Подамо знайдену затримку у двійковому доповнювальному коді від'ємного числа з урахуванням знакового розряду:

$$-4_{10} = 1.00100_2;$$
$$1.00100_{[ПК]} = 1.11100_{[ДК]}.$$

Тоді інформація в зоні $\beta 3$ для карти програмування ПМК має вигляд:

$\beta 3$	1	1 1 1 0 0

Зміна станів лічильника зони $\beta 3$ наведена у табл. 3.3.

Таблиця 3.3. Стани лічильника

№ такту	СТ		Примітки
	ЗР		
ПС	1	11100	ЗР =1 (ЛЕ2 заблокований)
1	1	+1 11101	ЗР =1 (ЛЕ2 заблокований)
2	1	+1 11110	ЗР =1 (ЛЕ2 заблокований)
3	1	+1 11111	ЗР =1 (ЛЕ2 заблокований)
4	0	+1 00000	ЗР =0 (ЛЕ1 заблокований, формування наступної адреси)

3.2.3. Призначення зони службових розрядів $\beta 4$

У обчислювальних системах зона $\beta 4$ може складатися із сотні розрядів. Найчастіше цю зону використають для контролю апаратури.

Як приклад можна навести використання зони для контролю слова мікрокоманди на парність або непарність.

Схема контролю має вигляд зображений на рис. 3.11. Для контролю використають операцію згортки (суму за модулем 2). У цьому випадку зона $\beta 4$ має довжину 1 розряд, вміст цього розряду доповнює кількість 1 у слові мікрокоманді до парної (або непарної, при контролі слова МК на непарність).

Під час контролю на парність сигнал «помилка»=1 визначає, що слово МК вміщує непарну кількість 1, тобто наявна помилка.

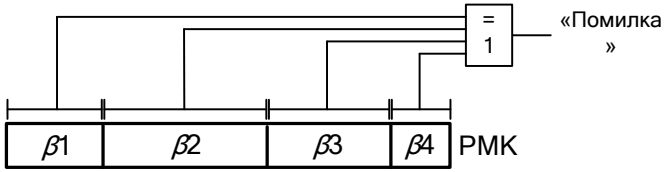


Рис. 3.11. Схема контролю слова МК на парність

Приклад 3.4. Сформувати зону $\beta 4$ для контролю заданих слів МК в зонах $\beta 1$, $\beta 2$ і $\beta 3$ на парність (табл. 3.4).

Виконання завдання

Таблиця 3.4. Кодування зони $\beta 4$

$\beta 1$	$\beta 2$	$\beta 3$	$\beta 4$
01010	1110	1010	1
11001	1101	1110	1
10100	0100	1000	0

3.2.4. Способи формування адреси МК. Структура зони $\beta 1$

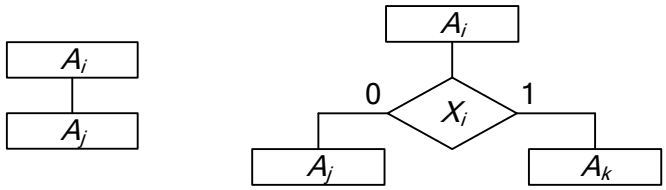
Адресація мікрокоманд у БМУ забезпечується зоною $\beta 1$. Для переходу на наступну МК у зоні $\beta 1$ поточної МК розміщується адреса переходу, або інформація для обчислення адреси переходу. Для виконання розгалуження мікроалгоритмів застосовуються наступні основні конструкції мікроалгоритмів:

- безумовний перехід;

- умовний перехід;
- цикли;
- мікроподпрограми.

Для реалізації безумовного переходу, зона $\beta 1$ мікрокоманди, що розміщується в БМУ за адресою A_i, містить частину або всю адресу переходу A_j. Тобто адреса переходу визначається як $[A_i] \rightarrow [A_j]$. За цим способом формуються адреси на лінійних ділянках МА (рис. 3.12, *а*).

За умовного переходу у зоні $\beta 1$ мікрокоманди A_i вказується інформація щодо адреси A_j або адреси A_k, куди здійснюється перехід в залежності від умови X_i, де X_i – будь-яка умова, що формується поза БМУ (рис. 3.12, *б*).



а *б*

Рис. 3.12. Типові конструкції МА:
а – безумовний перехід; *б* – умовний перехід

Циклічні мікроалгоритми можуть бути організовані двома способами:

- за умовою X_i, що формується поза БМУ;
- за кількістю повторень N, що формується за лічильником циклів усередині БМУ.

Цикли у свою чергу поділяються на:

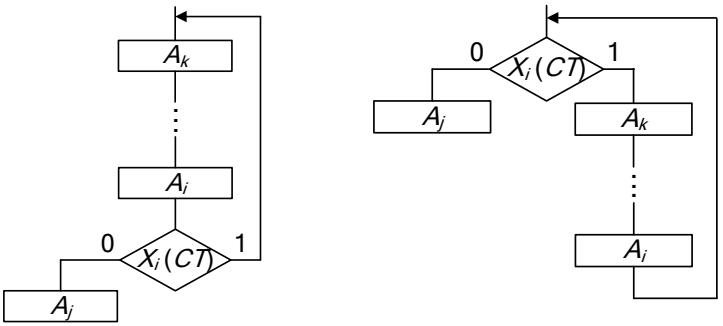
- цикли з перевіркою на вході (рис. 3.13, *а*);
- цикли з перевіркою на виході (рис. 3.13, *б*).

В залежності від умови, що перевіряється на вході/виході циклу, або стану лічильника CT, відбувається вихід з циклу – зона $\beta 1$ містить адресу A_j ($[A_i] \rightarrow [A_j]$), або перехід на початок циклу – зона $\beta 1$ містить адресу A_k ($[A_i] \rightarrow [A_k]$) (рис. 3.13, *а*, *б*).

За способом формування зони $\beta 1$ розрізняють наступні способи адресації мікрокоманд:

- примусова адресація МК;
- відносна адресація МК;
- природна адресація МК (інкрементна).

Для кожного способу адресації мікрокоманд використається відповідний формат зони $\beta 1$.



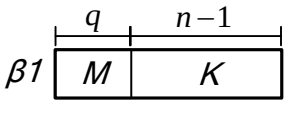
а *б*

Рис. 3.13. Циклічні МА:
а – цикл із перевіркою на виході;б – цикл із перевіркою на вході

3.3. БМУ з примусовою адресацією

3.3.1. Синтез БМУ з примусовою адресацією

За примусової адресації зона $\beta 1$ має наступний формат:



де M – поле управління мультиплексором;
q – довжина поля управління мультиплексором;
K – константа, що визначає адресу наступної мікрокоманди;
n – розрядність адреси мікрокоманди.

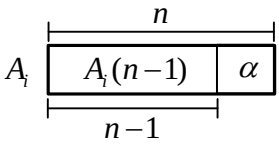
Довжина поля управління мультіплексором визначається за формулою:

$$q =]\log_2(k+2)[, \qquad (3.4)$$

де k – кількість зовнішніх умов.

Поле константи K являє собою $(n–1)$ старших розрядів адреси мікрокоманди.

Формат адреси мікрокоманди має наступний вигляд:



де α – визначає умову переходу, яка формується на виході мультиплексора в залежності від логічних умов X_i.

Спрощена структурна схема БМУ з примусовою адресацією зображена на рис. 3.14. На цій та подальших схемах БМУ входи для занесення початкової адреси D в РАМК умовно не показані.

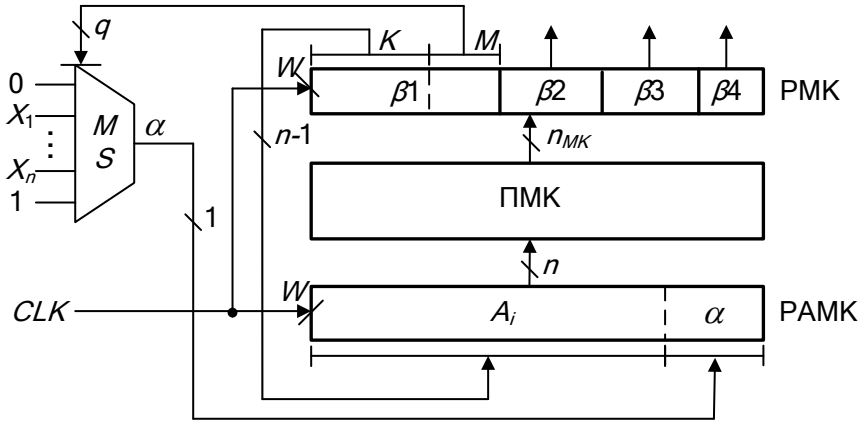


Рис. 3.14. Структурна схема БМУ з примусовою адресацією

Приклад 3.5. Розробити схему БМУ для реалізації заданого мікроалгоритму (рис. 3.15). Розробити карту настроювання БМУ.

Вихідні дані:

- спосіб управління асинхронний;
- ПМК ємністю 32 слова;
- комбіноване кодування зони $\beta 2$;
- забезпечити контроль слова МК на парність;
- час формування управляючих сигналів у тактах: $t(y_1)=1$, $t(y_2)=2$, $t(y_3)=5$, $t(y_4)=1$, $t(y_5)=12$, $t(y_6)=7$, $t(y_7)=9$.

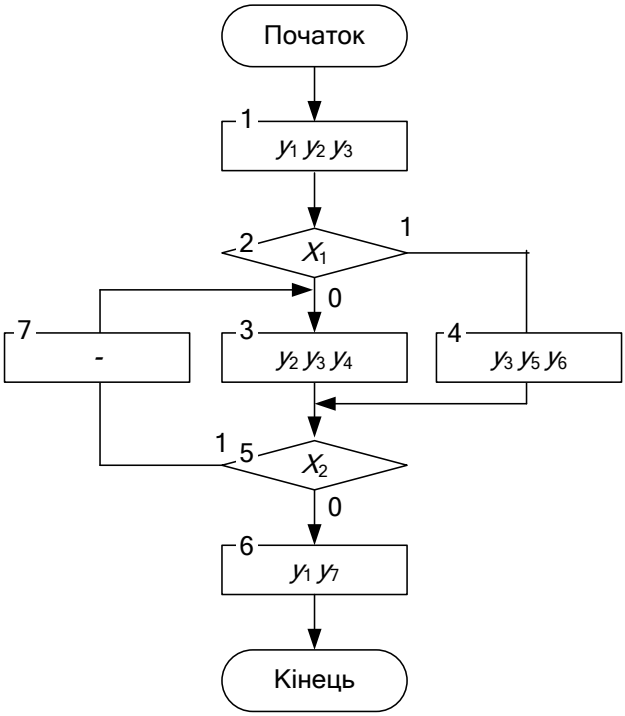


Рис. 3.15. Вихідний мікроалгоритм

Виконання завдання

1. Формат зони $\beta 1$.

Враховуючи, що ємність ПМК дорівнює 32 слова, розрахуємо розрядність адреси:

$$n =]\log_2 32[= 5.$$

Виходячи з розрядності адреси, отримаємо довжину поля константи:

$$K = 5 - 1 = 4.$$

Кількість управляючих входів мультиплексора, що визначає розрядність поля M розрахуємо за виразом (3.4), враховуючи, що кількість управляючих сигналів дорівнює двом (X_1, X_2):

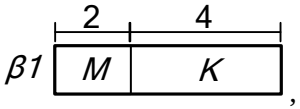
$$q =]\log_2(2+2)[= 2.$$

Складемо таблицю кодування розрядів поля управління мультиплексором (табл. 3.5).

Таблиця 3.5. Кодування поля M

$m_2\ m_1$	УС
00	0
01	X_1
10	X_2
11	1

Враховуючи наступний формат зони $\beta 1$



,

отримаємо:

$$n_{\beta 1} = 6.$$

2. Формат зони $\beta 2$:

Розподілимо управляючи сигнали за групами, так, що сигнали, які виробляються в одному такті будуть знаходитися у різних групах:

I	II	III
y_1	y_2	y_3
y_5	y_6	
y_4	y_7	

Для формування сигналів першої і другої групи будемо застосовувати дешифратори. Розрахуємо кількість розрядів кодів дешифраторів за виразом (2.2):

$$n_{\text{I}} = n_{\text{I}} =]\log_2 3[= 2.$$

Наведемо таблиці кодування сигналів у зоні $\beta 2$ (табл. 3.6 і 3.7).

Таблиця 3.6. Кодування сигналів

$\alpha_2\ \alpha_1$	УС
00	–
01	y_1
10	y_4
11	y_5

Таблиця 3.7. Кодування сигналів

$\gamma_2\ \gamma_1$	УС
00	–
01	y_2
10	y_6
11	y_7

В результаті отримали наступну структуру зони $\beta 2$:

$\beta 2$	α_1	α_2	γ_1	γ_2	y_3

.

Тоді довжина зони $\beta 2$:

$$n_{\beta 2} = 5.$$

3. Формат зони $\beta 3$.

Максимальна тривалість МО дорівнює: $t_{\max}=12$. Тоді максимальна затримка дорівнює: $\Delta t_{\max} = 11$.

За виразом (3.3) розрахуємо довжину зони $\beta 3$:

$$n_{\beta 3} =]\log_2 11[+ 1 = 5.$$

Враховуючи попередні обчислення, а тож те, що для перевірки на парність у зоні $\beta 3$ необхідно виділити один розряд, довжина слова МК дорівнює:

$$n_{МК} = 17.$$

4. Розміщення мікрокоманди у ПМК.

Правило. Мікрокоманди в альтернативних вершинах МА розміщуються у ПМК таким чином, щоб їх адреси відрізнялися лише одним молодшим розрядом (рис.3.16).

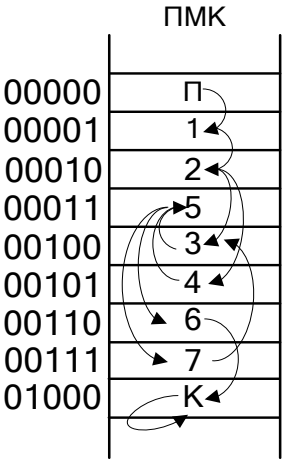


Рис. 3.16. Розміщення мікрокоманд в ПМК

5. Карта програмування наведена у табл. 3.8.

Таблиця 3.8. Карта програмування БМУ

№ МК	Адреса	$\beta1$		$\beta2$			$\beta3$		$\beta4$
		K	M	$\alpha_2\ \alpha_1$	$\gamma_2\ \gamma_1$	y_3	ЗР		
П	00000	0000	11	00	00	0	0	0000	0
1	00001	0001	00	01	01	1	1	1100	0
2	00010	0010	01	00	00	0	0	0000	1
3	00100	0001	11	10	01	1	1	1100	1
4	00101	0001	11	11	10	1	1	0101	0
5	00011	0011	10	00	00	0	0	0000	1
6	00110	0100	00	01	11	0	1	1000	0
7	00111	0010	00	00	00	0	0	0000	1
К	01000	0100	00	00	00	0	0	0000	1

6. Структурна схема БМУ з примусовою адресацією мікрокоманд, розробленого для реалізації заданого мікроалгоритму, зображена на рис. 3.17.

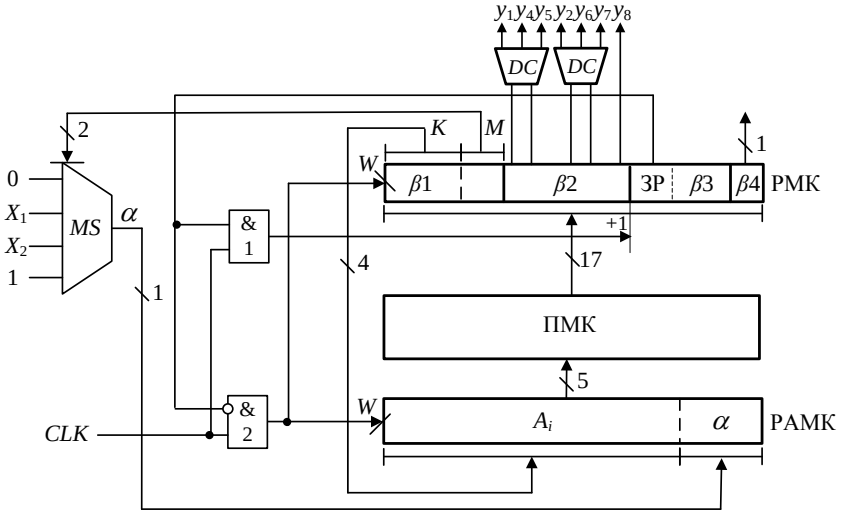


Рис. 3.17. Схема БМУ з примусовою адресацією

3.3.2. Скорочення зони $\beta1$ для примусової адресації МК

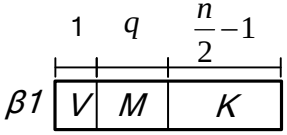
Для скорочення довжини зони $\beta1$ ПМК розглядають як двомірну матрицю. У цьому випадку розрядність адреси n поділяють на дві частини, одна з яких визначає номер рядку, а друга – номер стовпця, при-

чому, розбіжність у розрядності номерів стовбців й рядків має бути не більш ніж один розряд.

Формат адреси в цьому випадку має такий вигляд:

.

Формат зони $\beta 1$ для двовимірної ПМК має наступний вигляд:



де V – напрямок переходу: $V = 0$ (за рядком „→”), $V = 1$ (за стовпцем „↓”);

M – поле управління мультиплексором, довжиною q розрядів;

K – номер стовпця або рядку;

n – розрядність адреси мікрокоманди.

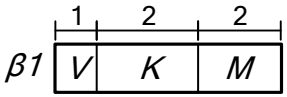
Приклад 3.6. Для БМУ із двовимірної ПМК розробити структуру зони $\beta 1$ і карту програмування для заданого МА (рис. 3.18), якщо ємність ПМК – 64 слова.

Виконання завдання

1. Формат зони $\beta 1$:

$$n_a = \lceil \log_2 64 \rceil = 6;$$
$$n_K = 6 : 2 - 1 = 2;$$
$$n_M = \lceil \log_2 3 \rceil = 2;$$
$$n_{\beta 1} = 5.$$

Отримаємо:



2. Розміщення МК у ПМК (рис. 3.19).

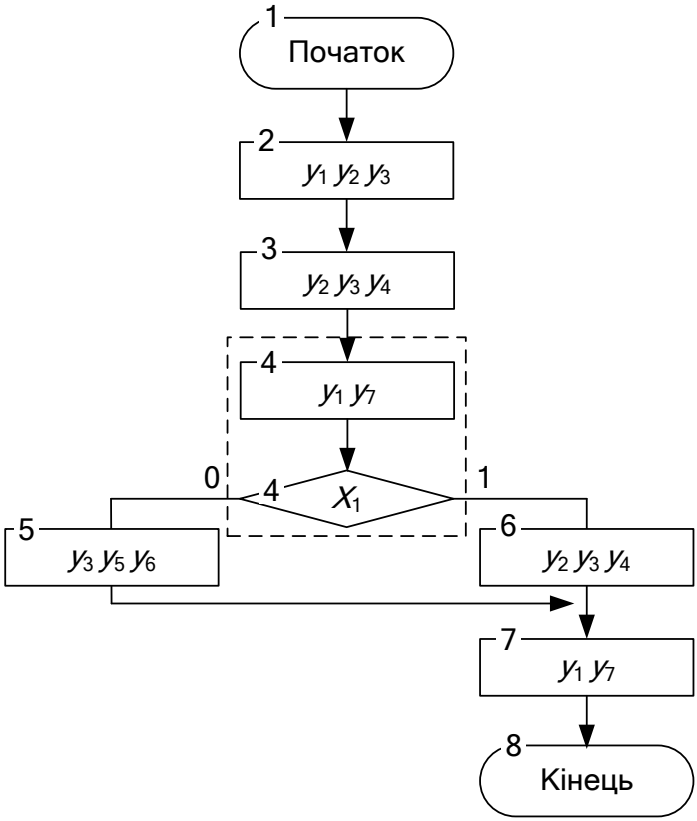


Рис. 3.18. Вихідний мікроалгоритм

ПМК

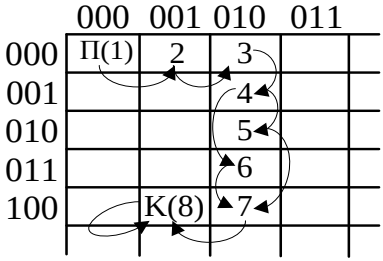


Рис. 3.19. Розміщення мікрокоманд в ПМК

3. Карта програмування зони $\beta 1$ (табл. 3.9).

Таблиця 3.9. Карта програмування БМУ

МК	Адреса МК		$\beta 1$		
	Номер рядку	Номер стовпця	V	K	M
1	000	000	0	00	11
2	000	001	0	01	00
3	000	010	1	00	11
4	000	010	1	01	01
5	010	010	1	10	00
6	011	010	1	10	00
7	100	010	0	00	11
8	100	001	0	00	11

4. Структурна схему БМУ з двовимірною організацією ПМК наведена на рис. 3.20.

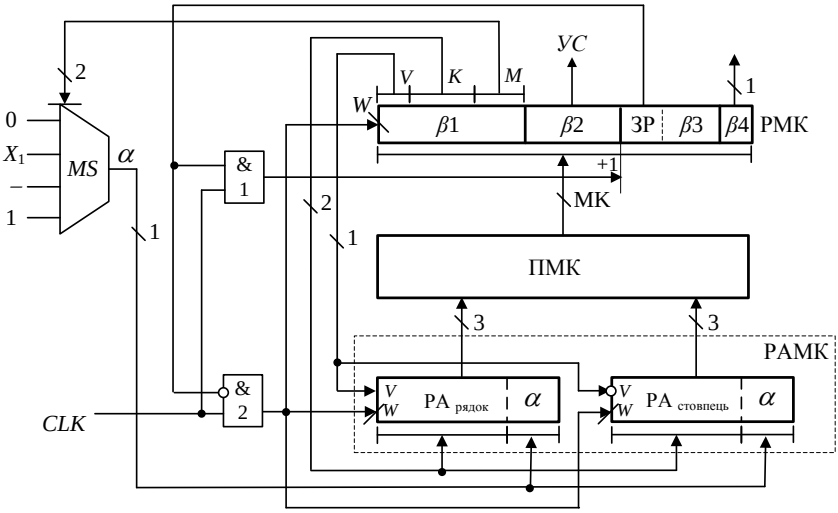


Рис. 3.20. Структурна схему БМУ з матричною ПМК

3.4. БМУ з відносною адресацією

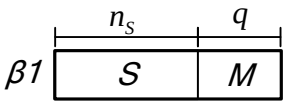
За відносної адресації адреса наступної МК визначається за формулою:

$$A_{i+1} = A_i + S + \alpha, \tag{3.5}$$

де S – приріст адреси МК;

α – сигнал на виході мультиплексора, що залежить від логічних умов X_i.

Формат зони $\beta 1$ у загальному вигляді:



Довжину поля S визначають за виразом:

$$n_S =]\log_2 N[+ 1, \qquad (3.6)$$

де N – максимальний приріст, додатковий знаковий розряд додається для визначення напрямку переходу (зменшення або збільшення адреси).

Приклад 3.7. Побудувати ПМК із відносною адресацією мікрокоманд для мікроалгоритму заданого на рис. 3.21.

Вихідні дані:

- ємність ПМК – 64 слова;
- максимальний перехід – 8 рядків;
- мінімальне кодування управляючих сигналів;
- синхронний спосіб управління.

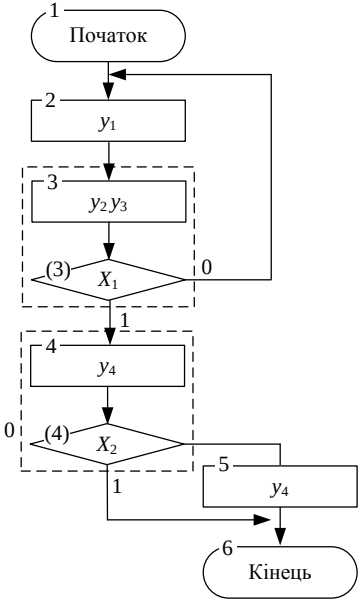


Рис. 3.21. Вихідний мікроалгоритм

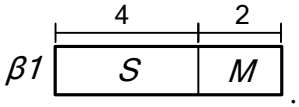
Виконання завдання

1. Формат зони $\beta 1$.

Виходячи з кількості логічних умов та заданого максимального збільшення, за виразами (3.6) та (3.4) відповідно, визначимо:

$$n_M = \lceil \log_2 4 \rceil = 2;$$
$$n_S = \lceil \log_2 8 \rceil + 1 = 4;$$
$$n_{\beta 1} = 6.$$

Отримаємо формат зони:



2. Розрядність адреси ПМК:

$$n_a = \lceil \log_2 64 \rceil = 6.$$

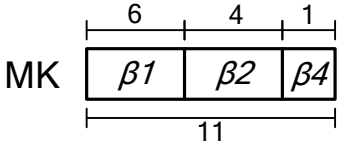
3. При синхронному способі управління для виконання кожної мікрооперації виділяється однакові проміжки часу, тому зону $\beta 3$ не використовуємо.

4. Формат зони $\beta 2$.

При мінімальному кодуванні управляючих сигналів довжина зони $\beta 2$ дорівнює кількості управляючих сигналів:

$$n_{\beta 2} = 4.$$

Враховуючи попередні обчислення отримаємо формат мікрокоманди ($n_{\text{МК}} = 11$):



5. Розміщення мікрокоманди у ПМК (рис. 3.22).

Правило. Мікрокоманди, що відповідають альтернативним вершинам мікроалгоритму, розміщуються у ПМК таким чином, щоб їх адреси відрізнялися на одиницю.

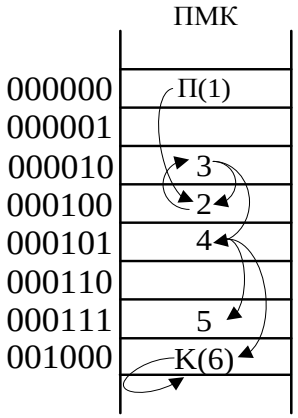


Рис. 3.22. Розміщення мікрокоманд в ПМК

6. Карта програмування зображена у табл. 3.10.

Таблиця 3.10. Карта програмування БМУ

№ МК	Адреса МК	$\beta 1$		$\beta 2$	$\beta 4$
		S	*M*	$y_1y_2y_3y_4$	
1	000000	0011	00	0000	0
2	000011	1111	00	1000	1
3	000010	0001	01	1010	1
4	000100	0010	10	1000	0
5	000110	0001	00	1000	0
6	000111	0000	00	0000	1

7. Структурна схема БМУ наведена на рис. 3.23.

Очевидно, що тривалість такту в даному випадку відносно БМУ з примусовою адресацією збільшується за рахунок затримки сигналів у суматорі. Таким чином, БМУ із примусовою адресацією мають вищу швидкодію, але більшу розрядність зони $\beta 1$ ніж у БМУ з відносною адресацією. Складність БМУ з відносною адресацією зменшується за рахунок скорочення зони $\beta 1$, але це приводить до втрати швидкодії пристрою.

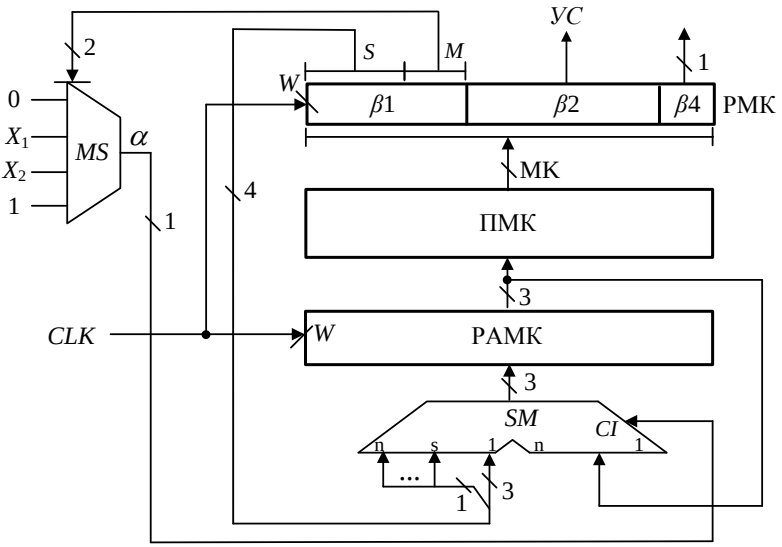


Рис. 3.23. Структурна схема БМУ з відносною адресацією

Приклад 3.8. Побудувати структурну схему БМУ і карту пам'яті мікропрограм для мікроалгоритму виконання операції множення. Мікроалгоритм повинен забезпечувати управління арифметико-логічним пристроєм із розподіленою логікою.

Вихідні дані:

- Спосіб адресації мікрокоманд – примусовий;
- Структура ПМК – лінійна;
- Ємність ПМК – 16 слів;
- Тривалість мікрооперації підсумовування – 4 такти;
- Початкова адреса мікропрограми – 0007*h*;
- Виконати перевірку слова МК на непарність;
- Розрядність операндів – 16 розрядів;
- Розрядність регістрів та суматорів – 8 розрядів.

Виконання завдання

Структурна схема пристрою для виконання операції множення першим способом з урахуванням елементної бази наведена на рис.

3.24. Мікроалгоритм управління роботою пристрою наведений на рис. 3.25. Змістовний МА наведений на рис. 3.26.

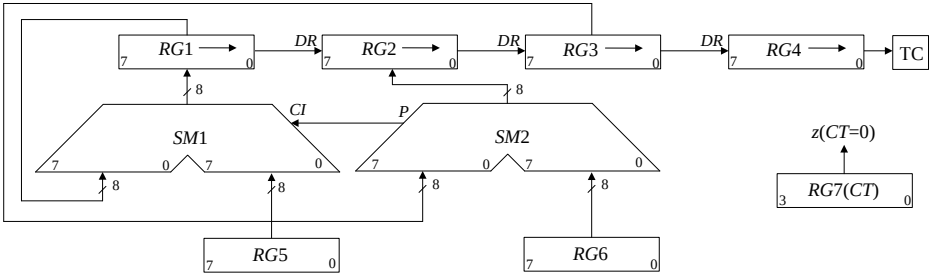


Рис. 3.24. Структурна схема пристрою множення

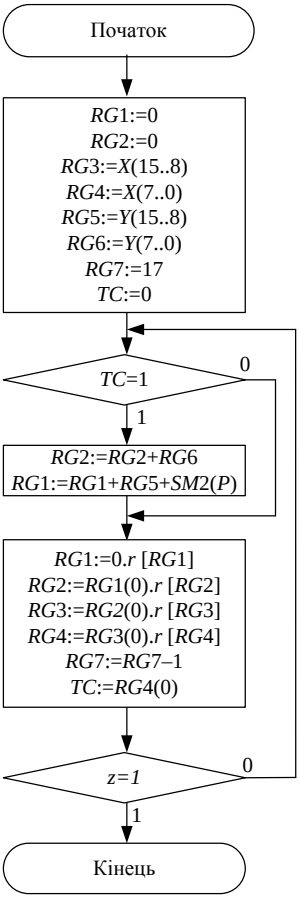


Рис. 3.25. Змістовний мікроалгоритм

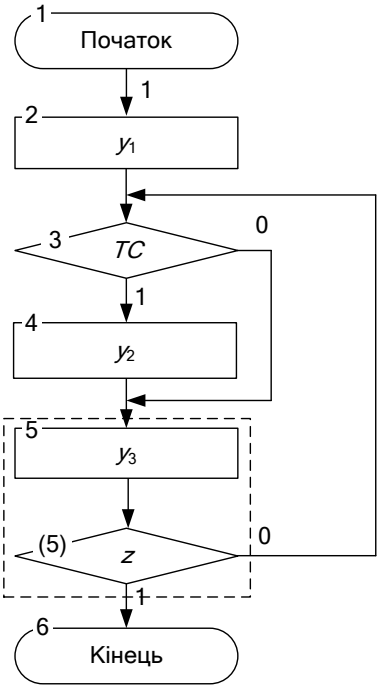


Рис. 3.26. Закодований алгоритм управління пристроєм множення

Визначимо формат зони $\beta 1$:

$$n_a = \lceil \log_2 16 \rceil = 4;$$
$$n_K = 3;$$
$$n_M = \lceil \log_2 4 \rceil = 2;$$
$$n_{\beta 1} = 5.$$

Визначимо спосіб управління мультиплексором (табл. 3.11).

Таблиця 3.11. Кодування поля *M*

$m_2\, m_1$	УС
00	0
01	*TC*
10	*z*
11	1

Визначимо формат зони $\beta 2$. Для максимального способу кодування управляючих сигналів розрахуємо розрядність коду дешифратора за виразом (3.2):

$$n_{\beta 2} = \left\lceil \log_2 4 \right\rceil = 2.$$

Наведемо кодування сигналів у зоні $\beta 2$ (табл. 3.12).

Таблиця 3.12. Кодування сигналів

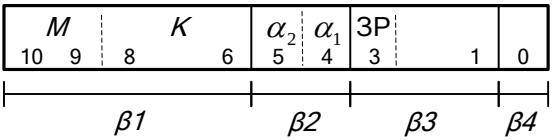
$\alpha_2\ \alpha_1$	УС
00	–
01	y_1
10	y_2
11	y_3

За виразом (3.3) розрахуємо довжину зони $\beta 3$:

$$\Delta t_{\max} = 3;$$

$$n_{\beta 3} = \left\lceil \log_2 3 \right\rceil + 1 = 3.$$

Для перевірки на парність у зоні $\beta 4$ необхідно виділити один розряд. Отримаємо наступний формат мікрокоманди ($n_{\text{МК}} = 10$):



Розміщуємо мікрокоманди в пам'яті мікрокоманд (рис. 3.27).

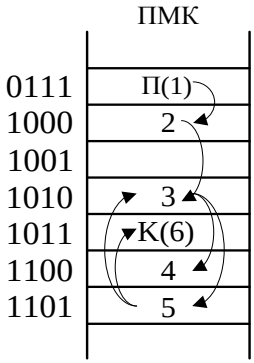


Рис. 3.27. Розміщення мікрокоманд в ПМК

Карта програмування БМУ наведена у табл. 3.13.

Таблиця 3.13. Карта програмування БМУ

№ МК	Адреса	$\beta 1$		$\beta 2$	$\beta 3$		$\beta 4$
		K	M	$\alpha_2\ \alpha_1$	ЗР		
П(1)	0111	100	00	00	0	00	0
2	1000	101	00	01	0	00	0
3	1010	110	01	00	0	00	1
4	1100	110	11	10	1	01	1
5	1101	101	10	11	0	00	0
К(6)	1011	101	11	00	0	00	1

Структурна схема БМУ із лінійною ПМК та примусовим способом адресації зображена на рис. 3.17.

3.5. Лабораторна робота 2

Ціль роботи: Дослідити засоби побудови блоків мікропрограмного управління. Одержати навички в проектуванні й налагодженні схем пристроїв управління з мікропрограмним управлінням.

Підготовка до роботи

1. Побудувати структурну схему БМУ і карту пам'яті мікропрограм для мікроалгоритму виконання операції множення. Мікроалгоритм повинен забезпечувати управління арифметико-логічним пристроєм із розподіленою логікою відповідно до варіанту лабораторної роботи 1 (БМУ повинен замінити управляючий пристрій із жорсткою логікою).

2. Під час виконання завдання необхідно враховувати дані наведені у табл. 3.14 – 3.15.

Таблиця 3.14. Вихідні дані до проектування

a_4	a_2	Спосіб адресації мікрокоманд	Структура ПМК	Ємність ПМК (слів)	Використати зону $\exists_4$ для перевірки слова МК
0	0	примусовий	лінійна	64	на непарність
0	1	примусовий	матрична		на парність
1	0	відносна	лінійна		на непарність
1	1				на парність
Спосіб мікропрограмування – горизонтальний;					
Забезпечити занесення початкової адреси мікроалгоритму в регістр адреси мікрокоманд.					

Таблиця 3.15. Вихідні дані до проектування

a_6	a_5	a_4	Тривалість мікрооперації підсумовування	Початкова адреса мікропрограми
0	0	0	7	18*h*
0	0	1	4	0*ah*
0	1	0	3	06*h*
0	1	1	6	0*eh*
1	0	0	11	13*h*
1	0	1	4	07*h*
1	1	0	5	11*h*
1	1	1	2	0*bh*

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити БМУ. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.

2. На спроектованому пристрої виконати числовий приклад із заданими викладачем значеннями операндів.

3. У протоколі навести функціональну схему пристрою для виконання операції множення.

Зміст звіту

Звіт з лабораторної роботи повинен включати короткі теоретичні відомості, необхідні для виконання лабораторної роботи; структурні та функціональні схеми; таблиці та діаграми, отримані при виконанні тео-

ретичного завдання, а також у процесі моделювання схем; висновки за роботою.

Контрольні питання

1. Наведіть загальну конструктивно-функціональну структуру ЕОМ, пояснити загальне призначення БМУ та АЛП.
2. Наведіть порівняльну характеристику АЛП з розподіленою та зосередженою логікою.
3. Приведіть етапи побудови АЛП із розподіленою логікою.
4. Визначіть призначення АЛП у ЕОМ. Наведіть класифікацію АЛП.
5. Визначіть призначення БМУ у ЕОМ, наведіть класифікації БМУ.
6. Поясніть, що розуміють під принципом мікропрограмного управління?
7. Наведіть класифікацію БМУ з точки зору забезпечення тривалості виконання мікрооперацій. Наведіть недоліки і переваги кожного із способів.
8. Як забезпечується тривалість виконання мікрооперацій при асинхронному способі управління виконанням МК у БМУ?
9. Наведіть формат слова мікрокоманди і поясніть призначення кожної із зон.
10. Як визначити довжину зони $\beta 2$ при горизонтальному способі мікропрограмування?
11. Назвіть способи формування структури зони $\beta 2$, переваги та недоліки кожного із способів.
12. Як визначити довжину зони $\beta 3$ формування управляючих сигналів БМУ при асинхронному способі управління виконанням МК?
13. Назвіть способи формування структури зони $\beta 1$, переваги та недоліки кожного із способів.
14. Як скоротити довжину зони $\beta 1$ при застосуванні примусової адресації МК?
15. Наведіть приклад застосування зони $\beta 4$.

РОЗДІЛ 4

Проектування пристроїв з мікропрограмним управлінням для виконання арифметичних операцій

4.1. Операція ділення

Існують два основних методи ділення чисел:

- з відновленням від'ємного залишку;
- без відновлення від'ємного залишку.

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 більш ефективним, тому надалі будемо розглядати саме цей метод.

Нехай ділене X і дільник Y є n-розрядними правильними дробами, поданими в прямому коді. В цьому випадку знакові й основні розряди операндів обробляються окремо. Знак результату визначається шляхом підсумовування за модулем два 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 2^0, дорівнює 1, а при $R_0 < 0$ – дорівнює 0. Різниця R_0 є залишком.
2. Подвоїти залишок (тобто одержати значення $2R_i$).
3. При $2R_i < 0$, додати Y, в зворотному випадку, якщо $2R_0 \geq 0$, відняти Y. Якщо знову отриманий залишок $R_{i+1} \geq 0$, то $Z_{i+1} = 1$, інакше $Z_{i+1} = 0$.
4. Повторити дії описані в пунктах 2 та 3 $(n - 1)$ раз.

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

Під час реалізації ділення *за першим способом* здійснюється зсув вліво залишку при нерухомому дільнику, такий спосіб називається *діленням із зсувом залишку*. На рис. 4.1 показаний принцип побудови пристрою для ділення чисел. Черговий залишок формується в регістрі

*RG*2 (у вихідному стані в цьому регістрі записане ділене *X*). Дільник *Y* знаходиться в регістрі *RG*1. Максимальний час одержання цифри результату визначається виразом $t = t_Д + t_З$, де $t_Д$ – тривалість виконання мікрооперації додавання-віднімання; $t_З$ – тривалість виконання мікрооперації зсуву. Результат формується в регістрі *RG*3 за (n + 1) циклів.

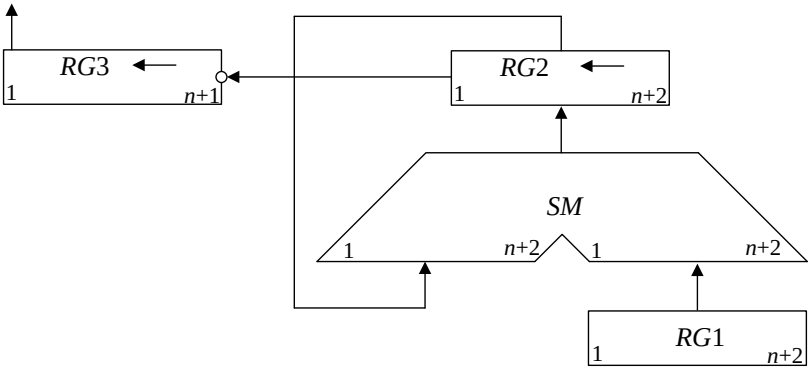


Рис. 4.1. Операційна схема пристрою ділення із зсувом залишку

Під час реалізації ділення *за другим способом*, який називається *діленням із зсувом дільника*, збільшується розрядність регістрів *RG*1, *RG*3 і суматора *SM*. Принцип побудови пристрою для ділення чисел за другим способом показаний на рис. 4.2. В даному випадку процеси додавання-віднімання і зсуву можуть бути сполучені в часі. Отже, для ділення за другим способом час одержання цифри результату дорівнює $t = t_Д$. Цифра результату формується на виході переносу суматора *SM*(*p*).

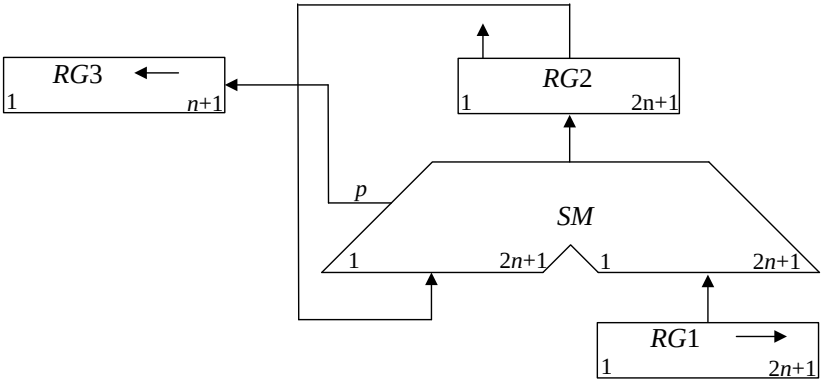


Рис. 4.2. Операційна схема пристрою ділення із зсувом дільника

Під час ділення чисел з фіксованою комою повинна бути передбачена можливість фіксації переповнення розрядної сітки. Ознака переповнення виробляється, якщо в першому циклі обчислення отримана цифра результату із значенням 1.

Приклад 4.1. Розробити операційний пристрій для виконання операції ділення за *способом ділення із зсувом дільника*. Виконати ділення мантис Z= Y / X двох додатних чисел: Y = 0,1001; X = 0,1100; 0 < Y, X < 1; Y < X. Виконати моделювання роботи пристрою за допомогою цифрової діаграми. Побудувати функціональну схему пристрою ділення.

Виконання завдання

Операційна схема пристрою для реалізації операції ділення із зсувом дільника наведена на рис. 4.2. Цифрова діаграма стану вузлів для заданих значень мантис наведена на рис. 4.3.

№ такту	*RG*3		*RG*2		*RG*1		Логічна умова
	1	1111	0	10010000	0	11000000	
1			0	10010000			*RG*1(0)=0
			1	01000000			
			1	11010000			
	1	1110			0	01100000	*RG*2(0)=1
2	1	1110	1	11010000	0	01100000	*RG*1(0)=1
			0	01100000			
			0	00110000			
	1	1101			0	00110000	*RG*2(0)=1
3	1	1101	0	00110000	0	00110000	*RG*1(0)=0
			1	11010000			
			0	00000000			
	1	1011			0	00011000	*RG*2(0)=1
4	1	1011	0	00000000	0	00011000	*RG*1(0)=0
			1	11101000			
			1	11101000			
	1	0110			0	00001100	*RG*2(0)=1
5	1	0110	1	11101000	0	00001100	*RG*1(0)=1
			0	00001100			
			1	11110100			
	0	1100			0	00000110	*RG*2(0)=0

Рис. 4.3. Цифрова діаграма виконання операції ділення із зсувом дільника

Для підрахунку кількості циклів застосовані маркерні одиниці у регістрі *RG*2, що дозволяє усунути лічильник, чим зменшити кількість

устаткування у пристрої. Змістовний алгоритм виконання операції ділення наведений на рис. 4.4.

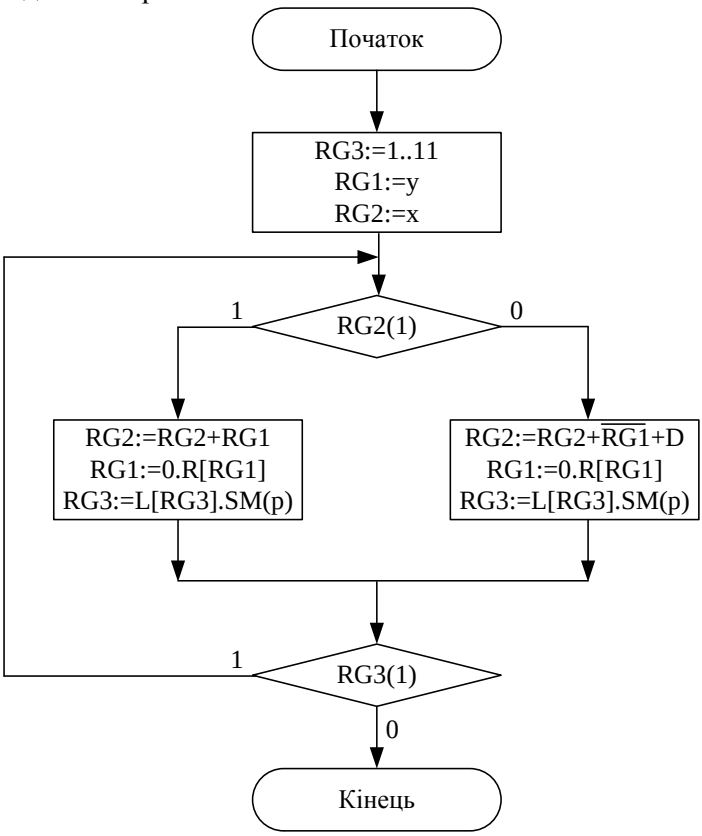


Рис. 4.4. Змістовний мікроалгоритм виконання операції ділення із зсувом дільника

На рис. 4.5 наведена функціональна схема пристрою для виконання операції ділення із зсувом дільника. На схемі зображені сигнали управління:

W – запис інформації в регістри;
SR – зсув вправо вмісту регістрів;
SL – зсув вліво вмісту регістрів;
CLR – обнуління вмісту регістру;
dec – декримент лічильника (у разі його використання);
d – сигнал, що дозволяє одержати доповнювальний код числа при ре-

алізації операції віднімання, подається на управляючий вхід пристрою інвертування та вхід *CI* суматора;

V – сигнал управління мультиплексором;

та логічні умови:

M – маркер кінця операції;

N – знак залишку.

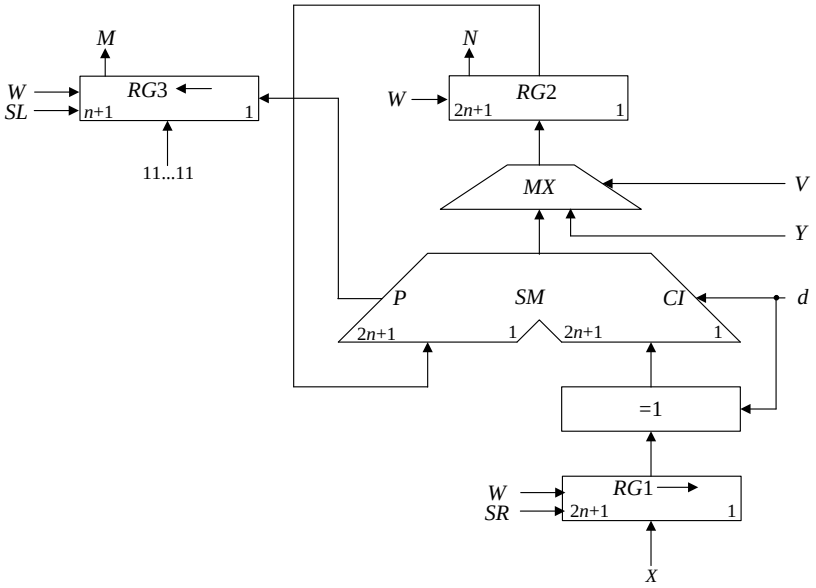


Рис. 4.5. Функціональна схема пристрою ділення за способом ділення із зсувом дільника

Приклад 4.2. Розробити арифметичний пристрій для ділення чисел за *способом ділення із зсувом залишку*. Виконати моделювання роботи пристрою за допомогою цифрової діаграми для наступних значень аргументів: *Y* = 0,1101, *X* = 0,1000, якщо 0 < Y, X < 1; Y < X.

Виконання завдання

Під час реалізації способу, що розглядається, здійснюється зсув вліво залишку при нерухомому дільнику. Операційна схема пристрою для виконання операції ділення із зсувом залишку наведена на рис. 3.1.

У початковому стані в регістрі *RG*2 розміщується ділене *X*. Дільник *Y* знаходиться в регістрі *RG*1. Черговий залишок формується в регістрі *RG*2. Розрядність регістрів *RG*1 і *RG*2 дорівнює (*n* + 2), де два розряди відводяться для подання знаку операндів. В лічильник *CT* записується кількість циклів обчислень, що необхідна для отримання ре-

зультату певної розрядності. За нульовим вмістом лічильника циклів *CT* визначається закінчення операції множення.

Змістовний мікроалгоритм виконання операції наведений на рис. 4.6. Діаграма стану регістрів при виконанні операції зображена на рис. 4.7.

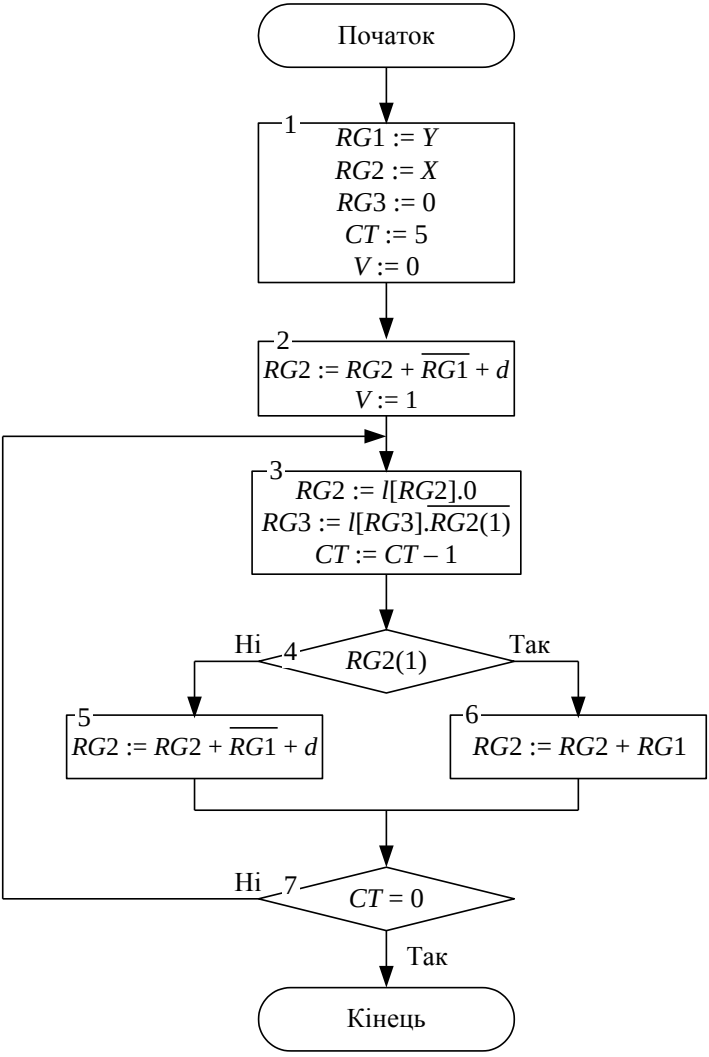


Рис. 4.6. Змістовний мікроалгоритм виконання операції ділення із зсувом залишку

	←RG3(Z) [1...(n+1)]	←RG2(X) [1...(n+2)]	RG1(Y) [1..(n+2)]	CT(n) [1...q]	Мікрооперації
ПС	00000	00,1000	$00{,}1101_{\text{пк}}$ $11{,}0011_{\text{дк}}$	101	
1		00,1000 +11,0011 11,1011			$RG2 := RG2 - RG1$
2	$\bar{1}$ 00000	11,0110 11,0110 +00,1101 00,0011		100	$\leftarrow RG2$ $\leftarrow RG3$ $CT := CT - 1$ $RG2(1) = 1$ $RG2 := RG2 + RG1$ $CT \neq 0$
3	$\bar{0}$ 00001	00,0110 00,0110 +11,0011 11,1001		011	$\leftarrow RG2$ $\leftarrow RG3$ $CT := CT - 1$ $RG2(1) = 0$ $RG2 := RG2 - RG1$ $CT \neq 0$
4	$\bar{1}$ 00010	11,0010 11,0010 +00,1101 11,1111		010	$\leftarrow RG2$ $\leftarrow RG3$ $CT := CT - 1$ $RG2(1) = 1$ $RG2 := RG2 + RG1$ $CT \neq 0$
5	$\bar{1}$ 00100	11,1110 11,1110 +00,1101 00,1011		001	$\leftarrow RG2$ $\leftarrow RG3$ $CT := CT - 1$ $RG2(1) = 1$ $RG2 := RG2 + RG1$ $CT \neq 0$
6	$\bar{0}$ 01001	01,0110 01,0110 +11,0011 00,1001		000	$\leftarrow RG2$ $\leftarrow RG3$ $CT := CT - 1$ $RG2(1) = 0$ $RG2 := RG2 - RG1$ $CT \neq 0$

Рис. 4.7. Цифрова діаграма стану регістрів при виконанні операції ділення зі зсувом залишку

Схема пристрою для виконання операції ділення, що змодельована за допомогою програми ПРОГМОЛС 2.0 наведена у додатку Г.

4.2. Обчислення квадратного кореня

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 різність $(B - A_i^2)=R_i$. Якщо $R_i \geq 0$, то A_i і є число, де цифри всіх розрядів співпадають 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. Так як в цьому випадку обчислення знову розпочинається з підстановки пробної одиниці, то замість заміни одиниці на нуль в i -му розряді віднімається одиниця з (i+1)-го розряду. Аналогічним чином обчислюють значення і $\sqrt[3]{B}$, $\sqrt[4]{B}$ і таке інше.

Обчислення різності $(B - A_i^2)=R_i$ є достатньо складною процедурою, тому обчислення чергової цифри результату виконують за наступним способом, що краще піддається автоматизації і може бути реалізований у вигляді самостійної операції.

Для отримання чергового залишку R_i до (i–1)-го результату дописують справа пару цифр 01 ($0{,}1^2 = 0{,}01$), зсувають його на (i–1) розряд вправо і віднімають з попереднього залишку R_{i-1}. За цього якщо $R_i \geq 0$, то цифра результату $a_i = 1$, інакше $a_i = 0$. В останньому випадку перед обчисленням наступної цифри результату необхідно відновити залишок. Але можливо замість відновлення залишку в i -му розряді результату записати $a_i = 0$, а в наступному циклі замість цифр 01 дописати цифри 11 і віднімання замінити додаванням. Таким чином ми відразу отримаємо правильний залишок R_{i+1}.

Приклад 4.3. Виконати обчислення $A = \sqrt{B} = \sqrt{10010001}$ з точністю до шостого розряду. Побудувати операційний пристрій для обчислення квадратного кореню. Виконати моделювання роботи операційного пристрою для заданого значення аргументу. Скласти змістовний мікроалгоритм роботи пристрою для обчислення квадратного кореню.

Виконання завдання

Виконання обчислення значення $A=\sqrt{10010001}$ з точністю до шостого розряду наведене на рис. 4.8, *а*. Для досягнення регулярності обчислень у випадку отримання додатної різності операцію віднімання доцільно замінити додаванням зворотного коду чергового результату R_{i-1} із дописаними цифрами 11 при цьому отримаємо підсумовування у доповнювальному коді, наприклад, у другому такті: $-00{,}10\underline{1}_{[\text{ПК}]}=+11{,}01\underline{1}_{[\text{ДК}]}$ (рис. 4.8, *б*).

0,		00	10010001		0,	00	10010001
	–	00	01			+ 11	11
1		00	01010001		1	00	01010001
		00	10100010	зсув		00	10100010 зсув
	–	00	101			+ 11	011
1		00	00000010		1	00	00000010
		00	00000100	зсув		00	00000100 зсув
	–	00	1101			+ 11	0011
0		11	00110100		0	11	00110110
		10	01101000	зсув		10	01101100 зсув
	+	00	11011			+ 00	11011
0		11	01000000		0	11	01000100
		10	10000000	зсув		10	10001000 зсув
	+	00	110011			+ 00	110011
0		11	01001100		0	11	01010100
		10	10011000	зсув		10	10101000 зсув
	+	00	1100011			+ 00	1100011
0		11	01011110		0	11	01011110
$A=\sqrt{B}=0{,}110000$					$A=\sqrt{B}=0{,}110000$		
а					*б*		

Рис. 4.8. Виконання обчислення квадратного кореню:
а – у прямому коді; *б* – у доповнювальному коді

На рис. 4.9 зображений операційний пристрій для виконання операції обчислення квадратного кореня. На рис. 4.10 виконано моделювання роботи пристрою за допомогою цифрової діаграми. На рис. 4.11 наведений змістовний мікроалгоритм роботи пристрою для обчислення квадратного кореню. Схема пристрою для виконання операції ділення,

що змодельована за допомогою програми ПРОГМОЛС 2.0 наведена у додатку Д.

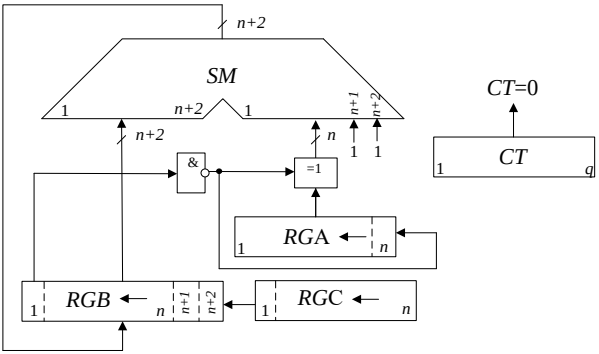


Рис. 4.9. Операційний пристрій для обчислення квадратного кореня

№ т-ту	← RGA (1 … n)	← RGB (1 … n)	← RGB (n+1 n+2)	← RGC (1 … n)	CT	Мікрооперації
ПС	00000000	00000000	00	10010001	110	
1	00000000	00000000	10	01000100		$2(RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0))$
		+11111111	11			$RGB := RGB + \overline{RGA}.11$
		00000000	01			
2	00000001	00000000	10	10001000		$RGC := l[RGC].0,$ $RGB := l[RGB].\overline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		00000001	01	00010000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+11111110	11			$RGB(0) = 0 \Rightarrow$ $RGB := RGB + \overline{RGA}.11$
		00000000	00		101	$CT := CT - 1; CT \neq 0$
3	00000011	00000000	00	00100000		$RGC := l[RGC].0,$ $RGB := l[RGB].\overline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		00000000	00	01000000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+11111100	11			$RGB(0) = 0 \Rightarrow$ $RGB := RGB + \overline{RGA}.11$

		11111100	11		100	$CT := CT - 1; CT \neq 0$
4	00000110	11111001	10	10000000		$RGC := l[RGC].0,$ $RGB := l[RGB].\underline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		11110011	01	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+00000110	11			$RGB(0) = 1 \Rightarrow$ $RGB := RGB + RGA.11$
		11111010	00		011	$CT := CT - 1; CT \neq 0$
5	00001100	11110100	00	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].\underline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		11101000	00	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+00001100	11			$RGB(0) = 1 \Rightarrow$ $RGB := RGB + RGA.11$
		11110100	11		010	$CT := CT - 1; CT \neq 0$
6	00011000	11101001	10	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].\underline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		11010011	00	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+00011000	11			$RGB(0) = 1 \Rightarrow$ $RGB := RGB + RGA.11$
		11101011	11		001	$CT := CT - 1; CT \neq 0$
7	00110000	11010111	10	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].\underline{RGC}(0),$ $RGA := l[RGA].RGB(0)$
		10101111	00	00000000		$RGC := l[RGC].0,$ $RGB := l[RGB].RGC(0)$
		+00110000	11			$RGB(0) = 1 \Rightarrow$ $RGB := RGB + RGA.11$
		11011111	11		000	$CT := CT - 1; CT = 0$

* – виділенням відмічені знакові розряди, що розмножуються на ширину розрядної сітки

Рис. 4.10. Цифрова діаграма обчислення квадратного кореня

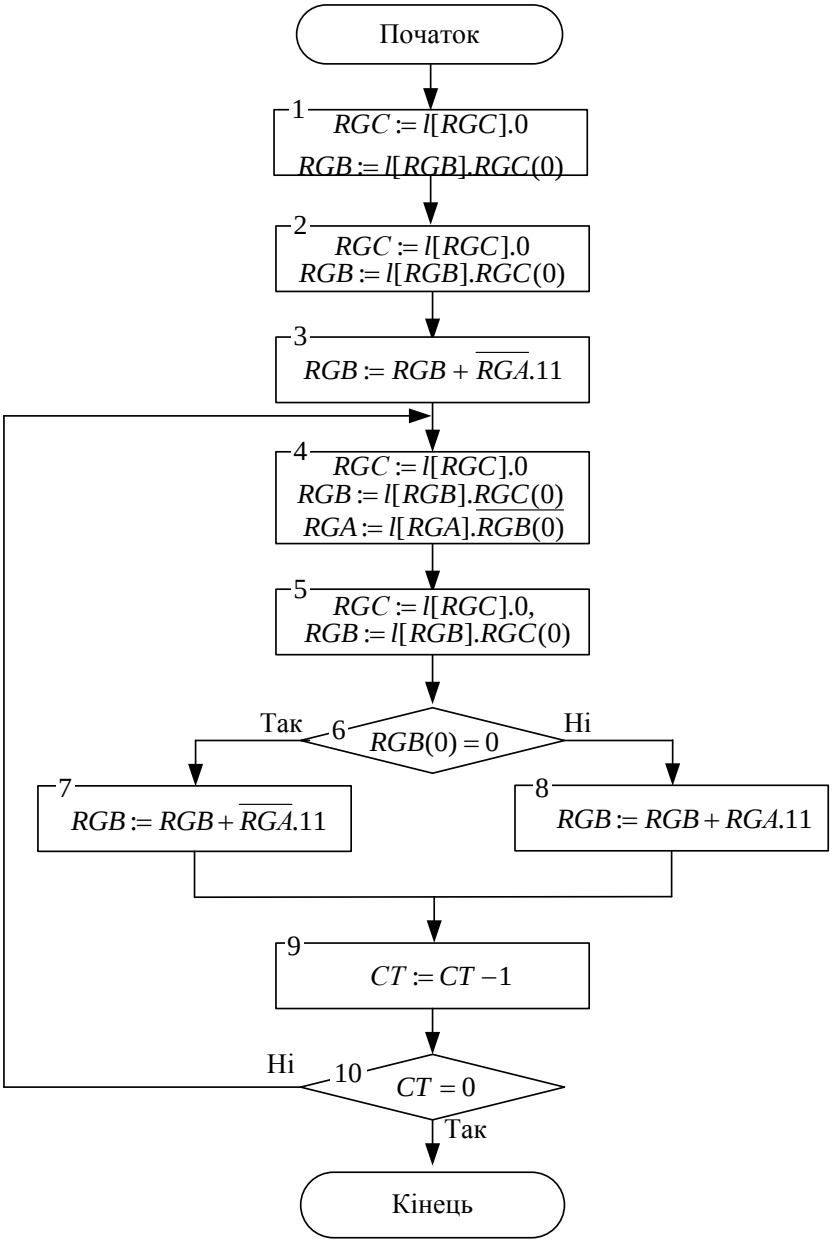


Рис. 4.11. Змістовний мікроалгоритм обчислення квадратного кореня

4.3. Лабораторна робота 3

Ціль роботи: Дослідити способи побудови пристроїв для машинного виконання різноманітних арифметичних операцій. Закріпити навички в проектуванні й налагодженні схем пристроїв з мікропрограмним управлінням.

Підготовка до роботи

1. Синтезувати операційну схему для виконання заданої арифметичної операції. Задану операцію обрати відповідно до варіанту з табл. 4.1.

2. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів. Розрядність та значення операндів обрати з табл. 4.1.

3. Скласти змістовний мікроалгоритм виконання заданої операції.

4. Побудувати структурну схему БМУ і карту пам'яті мікропрограм для мікроалгоритму виконання заданої операції.

5. Під час виконання завдання необхідно враховувати дані наведені у табл. 4.2 – 4.3.

Таблиця 4.1. Варіанти завдання

a_5	a_4	Операція	a_6	Розрядність операндів
0	0	$Z = \frac{Y}{X}$, способом ділення із зсувом дільника	0	8
			1	6
0	1	$Z = \frac{Y}{X}$, способом ділення із зсувом залишку	0	8
			1	6
1	0	$Z = \sqrt{X}$	0	8
			1	6
1	1	$Z = X^2$	0	8
			1	6

* – під час виконання операцій всі аргументи є правильними додатними дробами $Y = 0, y_1y_2\ldots y_n$ та $X = 0, x_1x_2\ldots x_n$, причому $Y < X$.

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити пристрій з мікропрограмним управлінням. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.

2. На спроектованому пристрої виконати числовий приклад із заданими викладачем значеннями операндів.

3. У протоколі навести функціональну схему пристрою для виконання заданої операції.

Таблиця 4.2. Вихідні дані до проектування

a_4	a_2	Спосіб адресації мікрокоманд	Структура ПМК	Ємність ПМК (слів)	Використати зону $\beta 4$ для перевірки слова МК
0	0	відносна	лінійна	64	на непарність
0	1				на парність
1	0	примусовий	лінійна		на непарність
1	1	примусовий	матрична		на парність
Спосіб мікропрограмування – горизонтальний;					
Забезпечити занесення початкової адреси мікроалгоритму в регістр адреси мікрокоманд.					

Таблиця 4.3. Вихідні дані до проектування

a_6	a_5	a_4	Тривалість мікрооперації підсумовування	Початкова адреса мікропрограми
0	0	0	7	18*h*
0	0	1	4	0*ah*
0	1	0	3	06*h*
0	1	1	6	0*eh*
1	0	0	11	13*h*
1	0	1	4	07*h*
1	1	0	5	11*h*
1	1	1	2	0*bh*

Зміст звіту

Звіт з лабораторної роботи повинен включати короткі теоретичні відомості, необхідні для виконання лабораторної роботи; структурні та

функціональні схеми; таблиці та діаграми, отримані при виконанні теоретичного завдання, а також у процесі моделювання схем; висновки за роботою.

Контрольні питання

1. Наведіть основні методи реалізації операції машинного ділення (обчислення кореню).
2. Наведіть склад устаткування необхідний для реалізації АЛП з розподіленою логікою для виконання операції машинного ділення (обчислення кореню).
3. Наведіть порівняльну характеристику АЛП з розподіленою та зосередженою логікою.
4. Наведіть загальну конструктивно-функціональну структуру пристрою з мікропрограмним управлінням для виконання арифметичних операцій, поясніть загальне призначення БМУ та АЛП.
5. Приведіть етапи побудови АЛП із розподіленою логікою.
6. Наведіть класифікації БМУ.
7. Поясніть, що розуміють під принципом мікропрограмного управління.
8. Як забезпечується тривалість виконання мікрооперацій при асинхронному способі управління виконанням МК у БМУ?
9. Наведіть отриманий при виконанні лабораторної роботи формат слова мікрокоманди у БМУ і поясніть призначення кожної із зон.

РОЗДІЛ 5

Проектування пристроїв для машинного перетворення чисел в різні системи числення

5.1. Перетворення чисел в ЕОМ

В ЕОМ обробка даних, як правило, виконується у двійковій системі числення, а вводу та виводу інформації – у двійково-десятковій системі числення, наприклад, в коді з вагами розрядів “8421”. Перетворення чисел з двійково-десяткової системи числення в двійкову і навпаки може бути виконано множенням та діленням чисел на два шляхом зсуву і корекції кодів.

Перехід від двійково-десяткової 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-десятковій системі, окрім зсуву, може бути необхідна корекція.

5.2. Перетворення чисел із десяткової системи числення в двійкову методом «зсуву-корекції»

Алгоритм перетворення n-розрядних двійково-десяткових чисел в коді “8421” у двійкову систему числення з природнім порядком вагів 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-десяткового числа будуть отримані нулі.

Значення корекції (–3) пояснюється наступним. Одиниця в тетраді двійково-десяткового числа дорівнює 10 одиницям сусідньої молодшої тетради. Коли при зсуві вправо (діленні на два) із старшої тетради в молодшу тетраду переходить одиниця, то вона повинна перенести половину ваги свого десяткового розряду, що складає п’ять одиниць для молодшої тетради. Фактично одиниця потрапляє в двійковий розряд молодшої тетради, який має вагу 8. Звідси корекція дорівнює (–3).

Віднімання замінюють додаванням доповнювального коду числа три (1101) без розповсюдження переносу в старшу тетраду.

Приклад 5.1. Розробити арифметичний пристрій для переводу чисел *із десяткової системи числення в двійкову* методом «зсуву-корекції». Виконати моделювання роботи пристрою за допомогою цифрової діаграми для наступного значення аргументу:

$$A=125_{10}=0001\ 0010\ 0101_{2\text{-}10}.$$

Скласти змістовний мікроалгоритм роботи пристрою та функціональну схему.

Виконання завдання

Операційна схема пристрою для переводу чисел із десяткової системи числення в двійкову методом «зсуву-корекції» зображена на рис. 5.1. У вихідному стані число подане у двійково-десятковій системі числення і складається з *i* тетрад. Пристрій містить *i* регістрів (*RG*1 – *RGi*) для зберігання тетрад двійково-десяткового числа, (*i*–1) суматорів (*SM*1 – *SM*(*i*–1)) для виконання корекції результату, регістр *RGR* для накопичування чергових розрядів результату, регістр *RGC* для зберігання значення на яке здійснюється корекція та лічильник *CT*.

Моделювання операції перетворення в двійкову систему числення для заданого аргументу проілюстроване цифровою діаграмою на рис. 5.2. Для зберігання трьох тетрад вихідного числа застосовується три чотирьох-розрядні регістри – *RG*1, *RG*2, *RG*3. У регістрі *RG*4 накопичуються чергові розряди результату. У регістрі *RG*5 розміщується значення корекції результату.

Змістовний мікроалгоритм виконання операції перетворення чисел із десяткової системи числення в двійкову методом «зсуву-корекції» наведений на рис. 5.3.

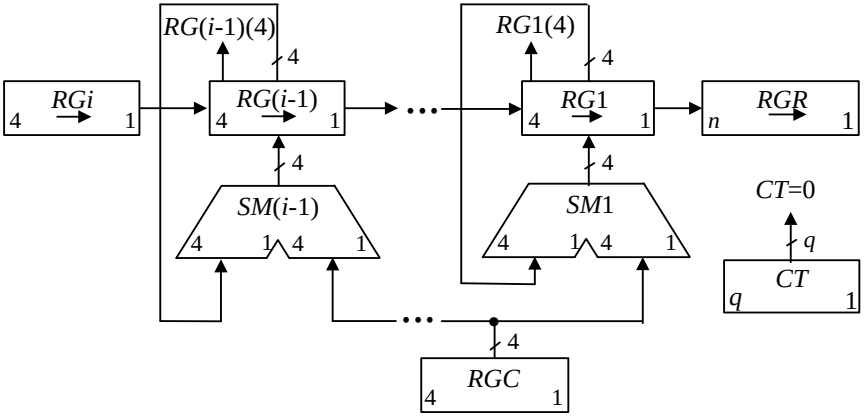


Рис. 5.1. Операційна схема пристрою для переводу чисел із десяткової системи числення в двійкову методом «зсуву-корекції».

№ такт	Двійково-десяткове число			Двійкове число		Мікрооперація
	*RG*1 (4 1)	*RG*2 (4 1)	*RG*3 (4 1)	*RG*4 (12 1)	*CT* (4 1)	
ПС	0001	0010	0101		1100	
1	0000	1001 +1101 0110	0010	1	1011	(*RG*1, *RG*2, *RG*3, *RG*4) → *RG*2:= *RG*2+ *RG*5 Корекція (–3) *CT*:=*CT* – 1
2	0000	0011	0001	01	1010	(*RG*1, *RG*2, *RG*3, *RG*4) → *CT*:=*CT* – 1
3	0000	0001	1000 +1101 0101	101	1001	(*RG*1, *RG*2, *RG*3, *RG*4) → *RG*3:= *RG*3+ *RG*5 Корекція (–3) *CT*:=*CT* – 1
4	0000	0000	1010 +1101 0111	1101	1000	(*RG*1, *RG*2, *RG*3, *RG*4) → *RG*3:= *RG*3+ *RG*5 Корекція (–3) *CT*:=*CT* – 1
5	0000	0000	0011	11101	0111	(*RG*1, *RG*2, *RG*3, *RG*4) → *CT*:=*CT* – 1
6	0000	0000	0001	111101	0110	(*RG*1, *RG*2, *RG*3, *RG*4) → *CT*:=*CT* – 1
7	0000	0000	0000	1111101	0101	(*RG*1, *RG*2, *RG*3, *RG*4) → *CT*:=*CT* – 1

8	0000	0000	0000	01111101	0100	$(RG1, RG2, RG3, RG4) \rightarrow CT:=CT-1$
9	0000	0000	0000	001111101	0011	$(RG1, RG2, RG3, RG4) \rightarrow CT:=CT-1$
10	0000	0000	0000	0001111101	0010	$(RG1, RG2, RG3, RG4) \rightarrow CT:=CT-1$
11	0000	0000	0000	00001111101	0001	$(RG1, RG2, RG3, RG4) \rightarrow CT:=CT-1$
12	0000	0000	0000	000001111101	0000	$(RG1, RG2, RG3, RG4) \rightarrow CT:=CT-1$

* – *Відповідь:* $A=1111101_2$.

Рис. 5.2. Цифрова діаграма перетворення десяткового числа в двійкову систему числення

Функціональна схема пристрою для виконання операції перетворення чисел із десяткової системи числення в двійкову методом «зсув-корекції» наведена на рис. 5.4.

Пристрій складається з чотирьохрозрядних регістрів *RG*1, *RG*2 та *RG*3, де у початковому стані розміщуються тетради двійково-десяткового вихідного числа. Регістр *RG*4 застосовується для накопичування розрядів двійкового результату, у вихідному стані в цьому регістрі розміщуються нулі. В регістрі *RG*5 зберігається двійкове число 1101 ($-3_{ДК}$), для здійснення корекції результату. Суматори *SM*1 та *SM*2 застосовується для корекції результату в тетрадах в регістрах *RG*1 та *RG*2, відповідно. Розрядність регістру результату *RG*4 визначається за виразом $n=2^2\times m$, де m – кількість двійкових тетрад вихідного двійково-десяткового числа. Лічильник *CT* застосовується для підрахунку кількості зсувів. Розрядність лічильника дорівнює $q =]\log_2 n[$. У вихідному стані у лічильник заноситься значення *n*. Закінчення операції перетворення визначається за нульовим вмістом лічильника *CT*.

В першому такті роботи пристрою виконується зсув вправо регістрів двійкових тетрад і регістру результату. Під час зсуву здійснюється запис молодших розрядів кожної старшої тетради у відповідні розряди кожної молодшої тетради та у розряд регістру результату які звільнилися при зсуві. У другому такті виконується перевірка старшого розряду регістру *RG*2, що перейшов із старшої тетради при зсуві, якщо значення цього розряду дорівнює одиниці у третьому такті виконуються корекція результату, тобто до вмісту регістру *RG*2 додається значення корекції з регістру *RG*5. Аналогічним чином у четвертому і п’ятому та-

ктах виконуються перевірка старшого розряду регістру *RG*1, та за необхідністю корекція тетеради у цьому регістрі. У шостому такті за умови, що вміст лічильника *CT* дорівнює нулю перетворення закінчуються, інакше виконуються чергові зсуви. Результат перетворення формується в регістрі *RG*4.

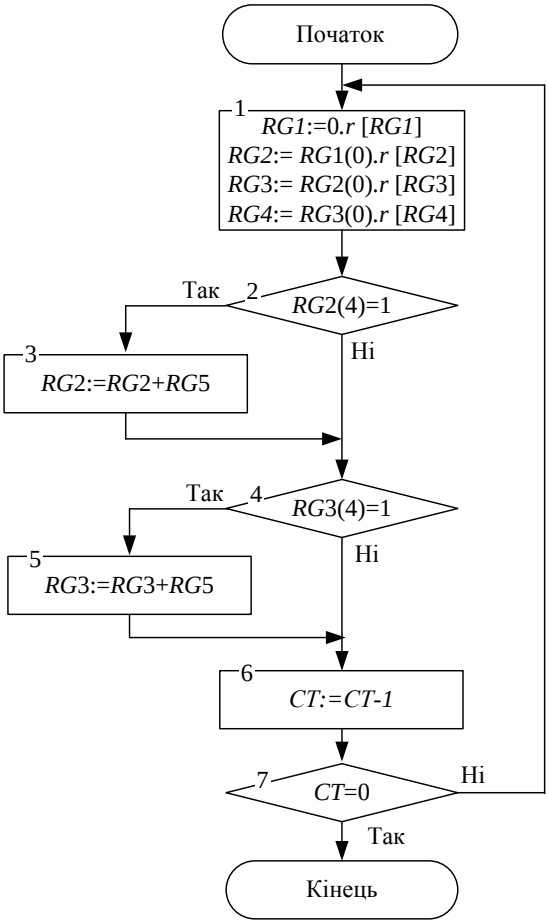


Рис. 5.3. Змістовний мікроалгоритм виконання операції перетворення чисел із десяткової системи числення в двійкову методом «зсуву-корекції»

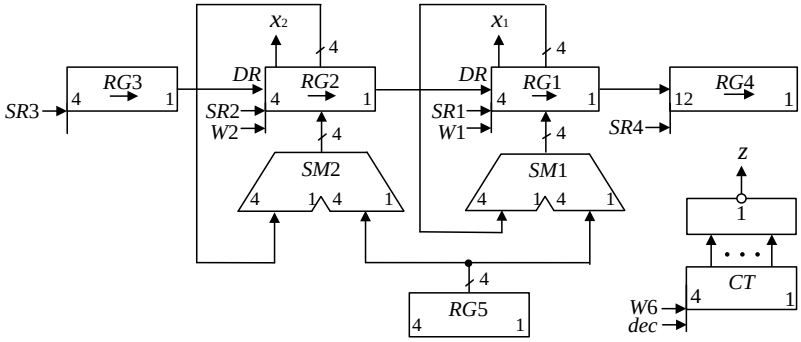


Рис. 5.4. Функціональна схема пристрою перетворення десяткового числа в двійкову систему числення

5.3. Перетворення чисел з двійкової системи числення в десяткову методом «зсуву-корекції»

Алгоритм перетворення двійкових чисел в двійково-десяткові числа в коді “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, то потрібна корекція, яка полягає у додаванні до коду тетради числа 6 з розповсюдженням переносу.

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

Необхідність корекції (+6) в тетрадах пояснюється наступним. Одиниця, що залишає тетраду, повинна забрати з неї 10 одиниць, а фактично забирає 16 (подвійну вагу старшого двійкового розряду тетради). Тому для компенсації до тетради додається число 6.

Приклад 5.2. Розробити арифметичний пристрій для переводу чисел *з двійкової системи числення в десяткову* методом «зсуву-корекції». Виконати моделювання роботи пристрою за допомогою цифрової діаграми для наступного значення аргументу:

$$A=1111101_2 .$$

Побудувати функціональну схему та скласти функціонально-структурний алгоритм роботи пристрою.

Виконання завдання

Операційна схема пристрою для переводу чисел з двійкової системи числення в десяткову методом «зсуву-корекції» зображена на рис. 5.5. Пристрій містить регістр *RGR* для зберігання вихідного двійкового числа, *i* регістрів (*RGT*1 – *RGTi*) для формування тетрад двійково-десяткового результату, де *i* – кількість тетрад, *i* суматорів (*SM*1 – *SMi*), регістр *RGC* для зберігання значення корекції, мультиплексори *MX*, логічні елементи та комбінаційна схема *КС*.

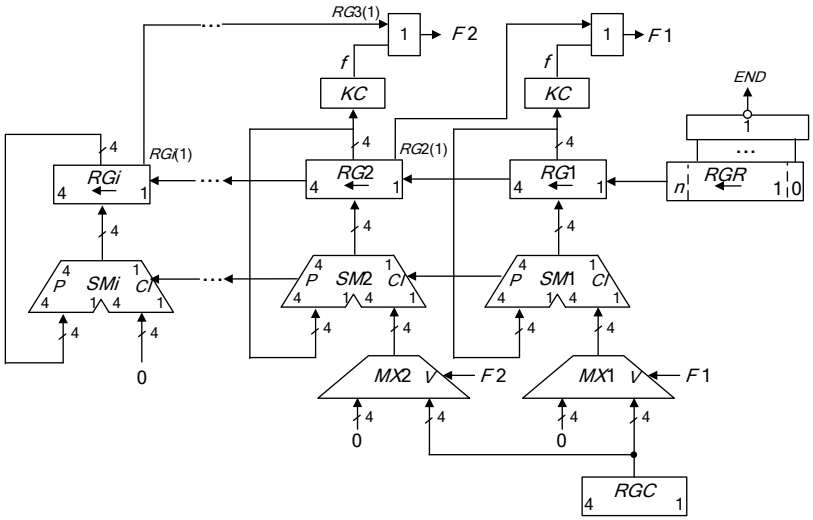


Рис. 5.5. Операційна схема пристрою для переводу чисел із десяткової системи числення в двійкову методом «зсуву-корекції»

До управляючих входів мультиплексорів підключені виходи *Fi* пристрою, що управляють здійсненням корекції в регістрах тетрад *RGT*1 – *RGT*(*i*-1). Управляючі сигнали *Fi* формуються на виходах логічних елементів АБО, на вхід кожного з яких поступає значення розряду, що вийшов з відповідної тетради під час зсуву та сигнал з виходу комбінаційної схеми *КС*.

Комбінаційні схеми підключені до виходів регістрів тетрад, окрім самої старшої тетради, вони реалізують перемикальну функцію порів-

няння вмісту регістрів з числом 9. Вихідний сигнал на виході комбінаційної схеми дорівнює одиниці, якщо вміст регістру тетради більше за дев'ять.

Перемикальна функція порівняння може бути описана за допомогою таблиці істинності наведеної у табл. 5.1. Виконаємо синтез комбінаційної схеми. Діаграма Вейча зображена на рис. 5.6.

Після виконання мінімізації отримуємо логічний вираз:

$$f1 = Q_4Q_2 \vee Q_4Q_3$$

Комбінаційна схема для порівняння вмісту регістру тетради з числом 9 зображена на рис. 5.7.

Таблиця 5.1. Таблиця істинності *КС*

Розряди регістру *RGT*				Ознака
Q_4	Q_3	Q_2	Q_1	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Моделювання операції перетворення в десяткову систему числення для заданого аргументу проілюстроване цифровою діаграмою на рис. 5.8. Для зберігання трьох тетрад результату застосовується три регістри – *RG*1, *RG*2, *RG*3, у регістрі *RG*4 зберігається вихідне двійкове число. У регістрі *RG*5 розміщується значення корекції результату.

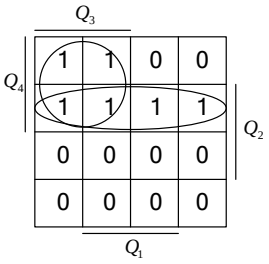


Рис. 5.6. Діаграма Вейча

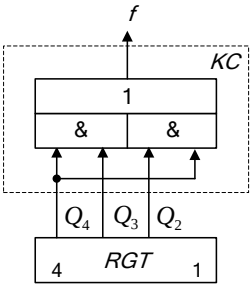


Рис. 5.7. Комбінаційна схема

№ такт	Двійково-десяткове число			Двійкове число	Маркер	Мікрооперація
	*RG*1	*RG*2	*RG*3	*RG*4	*END*	
ПС	0000	0000	0000	1111010010	0	
1	0000	0000	0001	111010010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
2	0000	0000	0011	11010010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
3	0000	0000	0111	1010010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
4	0000	0000 +0000 0001	1111 +0110 0101	010010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ← *RG*3:= *RG*3+ *RG*5 Корекція (+6) Поширення переносу
5	0000	0010 +0000 0001	1010 +0110 0000	10010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ← *RG*3:= *RG*3+ *RG*5 Корекція (+6) Поширення переносу
6	0000	0110	0001	0010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
7	0000 +0000 0001	1100 +0110 0010	0010	010	0	(*RG*1, *RG*2, *RG*3, *RG*4) ← *RG*2:= *RG*2+ *RG*5 Корекція (+6) Поширення переносу
	0010	0100	0100	10	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
	0100	1000	1000	0	0	(*RG*1, *RG*2, *RG*3, *RG*4) ←
	1001	0001 +0110 0111	0000 +0110 1000		1	*RG*2:= *RG*2+ *RG*5 *RG*2:= *RG*2+ *RG*5 Корекція (+6)

* – *Відповідь:* A=1001 0111 $0010_{2\text{-}10}$ = 972_{10}.

Рис. 5.8. Цифрова діаграма перетворення двійкового числа в десяткове

Змістовний мікроалгоритм виконання операції перетворення чисел із двійкової системи числення в десяткову методом «зсуву-корекції» наведений на рис. 5.9.

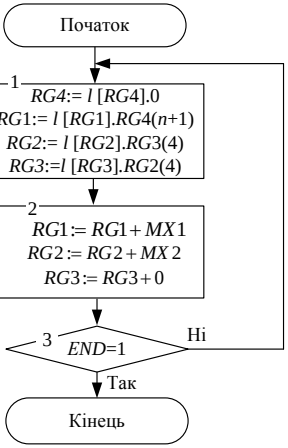


Рис. 5.9. Змістовний мікроалгоритм виконання операції перетворення чисел з двійкової системи числення в десяткову методом «зсуву-корекції»

Функціональна схема пристрою для виконання операції перетворення чисел із двійкової системи числення в десяткову методом «зсуву-корекції» наведений на рис. 5.10.

Пристрій складається з регістру *RG*4, що застосовується для зберігання вихідного десяткового числа, регістрів *RG*1, *RG*2, *RG*3, де формуються двійкові тетради результату, регістру *RG*5, де зберігається двійкове значення корекції, суматорів *SM*1, *SM*2, *SM*3, мультиплексорів *MX*1, *MX*2 та логічних елементів.

Розрядність регістру *RG*4 визначається за розрядністю *n* вихідного двійкового числа і дорівнює (*n* +1) розряд. За цього у молодший нульовий розряд записують одиницю, яка виконує роль маркерної одиниці. Одиниця на виході пристрою *END* сигналізує про закінчення перетворення.

Перетворення здійснюється шляхом зсуву регістрів вліво, та корекції кожної двійкової тетради у випадку, якщо отримане значення в тетраді більше за число 9 або з цієї тетради вийшла одиниця.

Корекція результату здійснюється у кожній тетраді окрім старшої, шляхом додавання до вмісту регістрів *RG*1, *RG*2 та *RG*3 двійкового числа 0110, що зберігається у регістрі *RG*5. Додавання виконуються з поширенням переносу у старшу тетроду, для чого виходи переповнення

розрядної сітки P суматорів кожної молодшій тетраді підключені до входів CI відповідних старших суматорів.

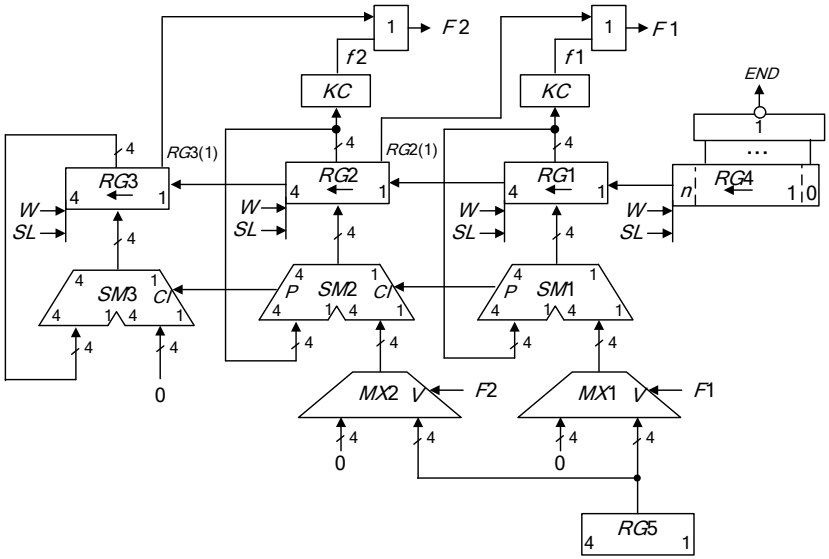


Рис. 5.10. Функціональна схема пристрою для виконання операції перетворення чисел з двійкової системи числення в десяткову методом «зсуву-корекції»

На першому такті роботи пристрою виконується лівий зсув регістрів *RG*1, *RG*2, *RG*3, *RG*4, при зсуві реалізовані ланцюги переносу старших розрядів молодшої тетради у молодші розряди відповідних старших тетрад. Розряди, що вийшли під час зсуву за межі розрядної сітки, поступають на вхід логічних елементів АБО і разом із ознакою *f*, що виконує порівняння значення тетради із числом 9, формують вихідний сигнал *F*1. Таким чином, якщо управляючий сигнал *F* дорівнює одиниці, то у тетраді необхідно виконувати корекцію.

У другому такті, в залежності від сигналу *F*, що поступає на управляючий вхід мультиплексора, на суматорах відбувається або додавання вмісту регістру *RG*5 до регістрів кожної тетради, або додавання нуля із ознакою переносу з молодшої тетради.

На наступних етапах, поки не встановився сигнал завершення перетворення *END*, виконуються чергові зсуви у регістрах.

Двійкові тетради результату формуються у регістрах *RG*1, *RG*2, *RG*3.

5.4. Лабораторна робота 4

Ціль роботи: Дослідити способи побудови пристроїв для машинного перетворення чисел в різні системи числення. Закріпити навички в проектуванні й налагодженні схем пристроїв з мікропрограмним управлінням.

Підготовка до роботи

1. Синтезувати операційну схему для виконання переводу чисел із однієї системи числення в іншу. Задану операцію обрати відповідно до варіанту з табл. 5.2.

Таблиця 5.2. Варіанти завдання

a_5	a_4	Операція	a_6	Кількість двійкових тетрад	a_3	Розрядність двійкового числа
0	0	перетворення з десяткової системи числення в двійкову	0	2		
			1	3		
0	1	перетворення з двійкової системи числення в десяткову	0	2	0	7
					1	8
			1	3	0	7
					1	8

2. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів. Розрядність операндів обрати з табл. 5.2.

3. Скласти змістовний мікроалгоритм виконання заданої операції.

4. Побудувати пристрій управління для виконання заданої операції. Тип пристрою обрати з табл. 5.3.

5. Під час виконання завдання необхідно враховувати дані наведені у табл. 5.4, для розробки БМУ, та у табл. 2.8 – 2.9 для розробки управляючого автомату. Тривалість виконання операції підсумовування прийняти три такти. Початкова адреса мікропрограми дорівнює двом молодшим розрядам номеру залікової книжки поданого у шістнадцятирічній системі числення.

Таблиця 5.3. Варіанти завдання

a_4	a_1	Тип пристрою управління
0	0	БМУ з відносною адресацією МК
0	1	БМУ з примусовою адресацією МК
1	0	Управляючий автомат Мілі
1	1	Управляючий автомат Мура

Таблиця 5.4. Вихідні дані до проектування

a_4	a_2	Спосіб адресації мікрокоманд	Структура ПМК	Ємність ПМК (слів)	Використати зону β_4 для перевірки слова МК
0	0	відносна	лінійна	64	на непарність
0	1				на парність
1	0	примусовий	лінійна		на непарність
1	1	примусовий	матрична		на парність
Спосіб мікропрограмування – горизонтальний;					
Забезпечити занесення початкової адреси мікроалгоритму в регістр адреси мікрокоманд.					

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити пристрій з мікропрограмним управлінням. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.

2. На спроектованому пристрої виконати числовий приклад із заданими викладачем значеннями операндів.

3. У протоколі навести функціональну схему пристрою для виконання заданої операції.

Зміст звіту

Звіт з лабораторної роботи повинен включати короткі теоретичні відомості, необхідні для виконання лабораторної роботи; структурні та функціональні схеми; таблиці та діаграми, отримані при виконанні теоретичного завдання, а також у процесі моделювання схем; висновки за роботою.

Контрольні питання

1. Як у ЕОМ подаються десяткові числа?
2. Від чого залежить кількість тетрод двійково-десяткового числа?
3. Поясніть, коли при переводі чисел необхідна корекція результату.
4. Як забезпечується занесення початкової адреси мікрокоманди а регістр адреси БМУ?
5. Наведіть склад устаткування для реалізації АЛП для переводу чисел з двійкової у десяткову систему числення та навпаки.
6. Наведіть склад устаткування для реалізації БМУ у даній роботі.
7. Наведіть формат слова мікрокоманди БМУ отриманий при виконанні лабораторної роботи і поясніть призначення кожної із зон.
8. Поясніть, як забезпечити необхідну тривалість виконання мікрооперації в БМУ.

РОЗДІЛ 6

Проектування арифметичних пристроїв для виконання операцій додавання та віднімання чисел із плаваючою комою

6.1. 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:

$$2^{P_X} M_X + 2^{P_Y} M_Y = 2^{P_Z} M_Z .$$

Для додавання чисел із плаваючою комою необхідно привести їх до загального порядку $P_{заг}$, в якості якого зручно обрати більший порядок з двох доданків $P_{заг} = \max(P_X, P_Y)$.

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

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

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

Виконання операцій додавання та віднімання чисел із плаваючою комою у загальному вигляді складається з наступних етапів: вирівнювання порядків; підсумовування мантис; визначення порядку результату; нормалізація результату; округлення результату; кінцева нормалізація результату.

6.1.1. Пристрій для вирівнювання порядків

Функціональна схема пристрою для виконання операції вирівнювання порядків під час підсумовування чисел із плаваючою комою зображена на рис. 6.1.

Пристрій складається з регістрів *RGPX*, *RGPY* для зберігання порядків та *RGX*, *RGY* для зберігання мантис чисел із плаваючою комою. Регістри порядків та мантис мають розрядність (n+2) та (m+2), де – n та m розрядність порядку та мантиси відповідно, по два розряди додають-

ся для зберігання знакових розрядів порядків та мантис. У регістрі *RGΔ* запам'ятовується різність порядків *Δ*. Також у склад пристрою входять суматор порядків *SMP*, що застосовується для визначення числа з меншим порядком, тригер *TT*, який застосовується для запам'ятовування того числа, порядок якого більший, інвертор та логічний елемент АБО.

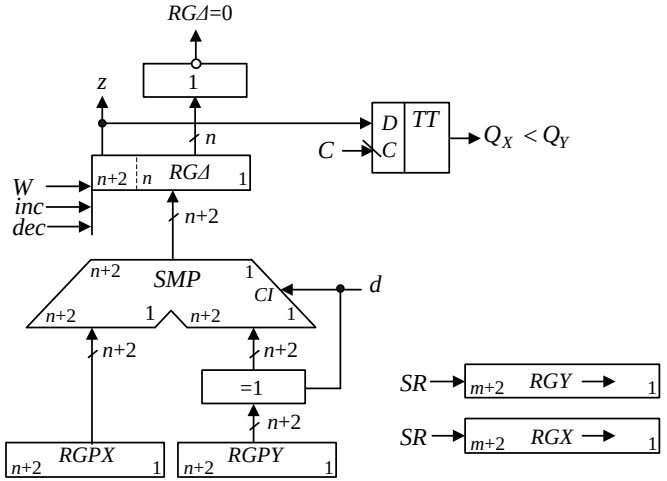


Рис. 6.1. Операційний пристрій для виконання операції вирівнювання порядків чисел із плаваючою комою

Моделювання операції вирівнювання порядків для заданих значень операндів проілюстроване цифровою діаграмою на рис. 6.2 – 6.3.

Змістовний мікроалгоритм роботи пристрою для виконання операції вирівнювання порядків зображений на рис. 6.4.

Для вирівнювання порядків на першому кроці мікроалгоритму визначають, порядок якого числа менший. Для цього з порядку одного числа віднімають порядок другого числа. За знаком різності визначають більший порядок, а за абсолютною величиною різності визначають необхідне число зсувів.

Порядки чисел *PX* та *PY* поступають на суматор *SMP* та за сигналом *W* різність *Δ* запам'ятовується у регістрі *RGΔ*. Знак різності *Δ* аналізується і запам'ятовується у тригері *T*1. За знаком різності формується сигнал правого зсуву *SR* мантиси у регістрі *RGX* або у регістрі *RGY* та відповідно сигнал *inc* або *dec* регістру *RGΔ*. Операція вирівнювання порядків закінчується за нульовим вмістом регістру *RGΔ*.

	PX	*MX*
X =	00 110	11 00110
Y =	00 011	11 11011
	PY	*MY*

№ такту	*RGPX*		*RGPY*		*RGΔ*		*RGX*		*RGY*		z	*RGΔ*	Мікрооперація
	n+2 *n*+1	*n* 1	*n*+2 *n*+1	*n* 1	*n*+2 *n*+1	*n* 1	*m*+2 *m*+1	*m* 1	*m*+2 *m*+1	*m* 1			
ПС	00	110	00	$011_{[ПК]}$	00	000	11	$00110_{[ПК]}$	11	$11011_{[ПК]}$			
1			11	$101_{[ДК]}$	00 +11 00	110 101 011	11	$11010_{[ДК]}$	11	$00101_{[ДК]}$	0	0	$RGΔ := RGPX - RGPY$ $PY < PX \Rightarrow \begin{cases} MY \rightarrow \\ PZ := PX \end{cases}$
2					00	010			11	100101	0	0	$RGY := RGY(m+2).r[RGY]$ $RGΔ := RGΔ - 1$
3					00	001			11	1100101	0	0	$RGY := RGY(m+2).r[RGY]$ $RGΔ := RGΔ - 1$
4					00	000			11	11100101	0	1	$RGY := RGY(m+2).r[RGY]$ $RGΔ := RGΔ - 1$

PZ = 00, 110

$MX_{[ДК]}$ = 11, 11010
$MY_{[ДК]}$ = + 11, 11100101
$MZ_{[ДК]}$ = 11, 10110101

Рис. 6.2. Цифрова діаграма операції вирівнювання порядків

	PX	*MX*
X =	00 011	11 10101
Y =	00 101	11 11001
	PY	*MY*

№ такту	*RGPX*			*RGPY*			*RGΔ*			*RGX*			*RGY*			z	*RGΔ*	Мікрооперація
	n+2	*n*+1	*n* 1	*n*+2	*n*+1	*n* 1	*n*+2	*n*+1	*n* 1	*m*+2	*m*+1	*m* 1	*m*+2	*m*+1	*m* 1			
ПС	00		011	00		$101_{[ПК]}$	00		000	11		$10101_{[ПК]}$	11		$11001_{[ПК]}$			
1				11		$011_{[ДК]}$	00 +11 11		011 011 110	11		$01011_{[ДК]}$	11		$00111_{[ДК]}$	1	0	*RGΔ*:= *RGPX* – *RGPY* $PX < PY \Rightarrow \begin{cases} MX \rightarrow \\ PZ := PY \end{cases}$
2							11		111	11		101011				1	0	*RGX:=RGX*(*m*+2).*r[RGX]* *RGΔ:= RGΔ* + 1
3							00		000	11		1101011				0	1	*RGX:=RGX*(*m*+2).*r[RGX]* *RGΔ:= RGΔ* + 1

PZ = 00, 110

$MX_{[ДК]}$ =		11, 1101011
$MY_{[ДК]}$ =	+	11, 00111
$MZ_{[ДК]}$ =		11, 0000111

Рис. 6.3. Цифрова діаграма операції вирівнювання порядків

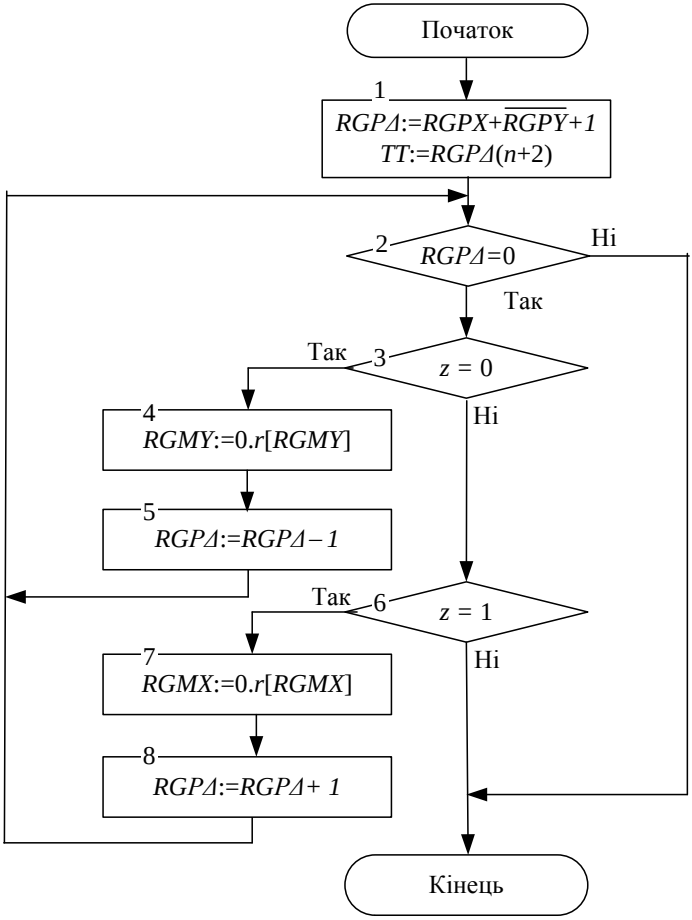


Рис. 6.4. Змістовний мікроалгоритм вирівнювання порядків чисел із плаваючою комою

Після вирівнювання порядків здійснюється підсумовування мантис за способом, який зазвичай застосовується для підсумовування чисел з фіксованою комою. Результат отримує порядок більшого за абсолютним розміром операнда, тобто того операнда мантиса якого залишалась нерухомою під час вирівнювання порядків. Встановити цей порядок можливо за допомогою тригеру *TT*.

6.1.2. Пристрій для виконання нормалізації модифікованого додатного коду

Підсумовування мантис порушує нормалізацію результату. Додатні нормалізовані числа завжди мають одиницю в розряді k^{-1}, а від'ємні числа, подані інверсними кодами, у розряді k^{-1}мають нуль. Тому у нормалізованих чисел розряди k та k^{-1} на співпадають. Внаслідок чого незбіжність цифр у знакових розрядах свідотствує про порушення нормалізації вліво (ЛПН) – переповнення, а збіг цифр знакових розрядів і старшого розряду мантиси – про порушення нормалізації вправо (ППН). Таким чином, якщо для підсумовування мантис застосовувати модифіковані інверсні коди, то ліве порушення нормалізації визначається за комбінацією цифр 01 та 10 у знакових розрядах суматора мантис суматора мантис, який використовується для підсумовування мантис. Тоді праве порушення нормалізації визначається за комбінаціями цифр 00,0 та 11,1 у старшому та знаковому розрядах суматора.

Для приведення числа до нормалізованого вигляду при ЛПН мантиса результату зсувається на один розряд вправо, а порядок збільшується на одиницю. Під час ППН мантиса зсувається вліво до появи одиниці в старшому розряді, за додатною мантисою, або нуля – за від'ємною мантисою, а з порядку віднімається така кількість одиниць, що дорівнює кількості зсувів мантиси.

Слід відмітити особливості виконання арифметичних зсувів інверсних машинних кодів, які виконуються з урахуванням знакових розрядів.

Під час арифметичного зсуву додатних чисел (відображення інверсних кодів співпадає з прямим кодом) знаковий розряд не зсувається, а основні розряди зсуваються ліворуч або праворуч із заповненням нулями розрядів, що звільнилися при зсуві.

Під час арифметичного зсуву праворуч від'ємних чисел поданих у інверсних кодах знаковий розряд переходить у поле основних розрядів і знов заповнюється тим самим значенням, тобто відбувається поширення знакового розряду.

Під час арифметичного зсуву ліворуч від'ємних чисел поданих у зворотному коді відбувається циклічний зсув, тобто розряд що звільнився при зсуві заповнюється одиницею, що вийшла із знакового розряду.

Під час арифметичного зсуву ліворуч від'ємних чисел поданих у додатному коді розряд що звільнився при зсуві заповнюється нулем.

Детально особливості виконання зсувів машинних кодів наведені у розділі 1.6.

Операційна схема пристрою для виконання операції нормалізації модифікованого додатного коду при підсумовуванні чисел із плаваючою комою зображена на рис. 6.5.

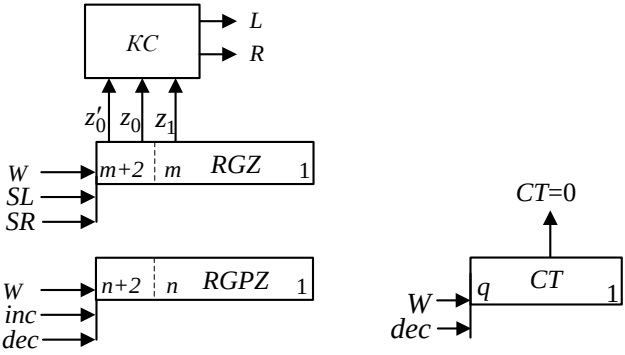


Рис. 6.5. Операційний пристрій для виконання операції нормалізації модифікованого додатного коду

Операційна схема пристрою складається з регістрів *RGPZ, RGZ* – відповідно регістрів порядку та мантиси результату, записаної у модифікованому додатному коді. Комбінаційна схема *KC* застосовуються для виявлення порушення нормалізації мантиси результату. За цього перевіряються умови порушення нормалізації та генеруються управляючи сигнали *L* та *R* зсуву регістру *RGZ* вліво або вправо та додавання або віднімання одиниці в регістрі *RGPZ.*

Лічильник *CT* застосовується для обмеження кількості зсувів числом розрядів мантиси результату, бо процес зсувів під час ППН може виявитись нескінченним, якщо в результаті додавання мантис отриманий нуль. Розрядність лічильника дорівнює $q =]\log_2 m[$, де m – розрядність мантиси. У вихідному стані у лічильник заноситься значення m.

Виконаємо синтез комбінаційної схеми. Таблиця істинності перемикальної функції, що визначає порушення нормалізації зображена у табл. 6.1. Аргументами перемикальної функції є значення z'_0, z_0, z_1 – цифр старшого та молодшого знакових розрядів результату і старшого розряду мантиси відповідно. Функції *L* та *R* є функціями порушення нормалізації вліво та вправо відповідно.

Таблиця 6.1. Таблиця істинності

Розряди регістру *RGZ*			Значення функцій	
z_0'	z_0	z_1	L	R
0	0	0	1	0
0	0	1	0	0
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	0

Виходячи з діаграм Вейча (рис. 6.6) отримаємо логічні вирази для побудови комбінаційної схеми:

$$L = z_0' z_0 z_1 \vee \overline{z_0'}\,\overline{z_0}\,\overline{z_1},$$

$$R = z_0' \overline{z_0} \vee \overline{z_0'} z_0 = z_0' \oplus z_0.$$

Комбінаційна схема зображена на рис. 6.7.

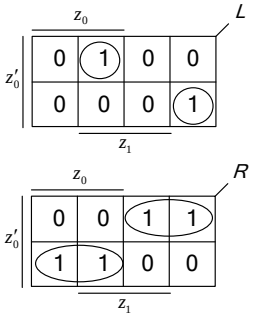


Рис. 6.6. Діаграма Вейча

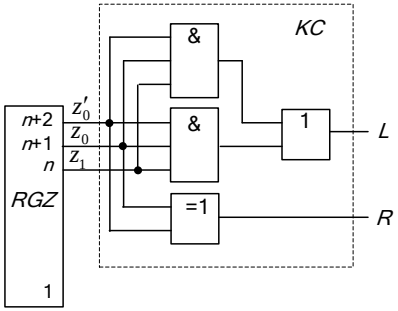


Рис. 6.7. Комбінаційна схема

Моделювання операції нормалізації для заданих значень аргументів з різними випадками порушення нормалізації проілюстроване цифровою діаграмою на рис. 6.8 – 6.9.

Змістовний мікроалгоритм роботи пристрою для виконання операції нормалізації зображений на рис. 6.10.

№ такту	*RGPZ*			*RGZ*			ЛПН	ППН	Мікрооперація
	n+2 *n*+1	*n*	1	*m*+2 *m*+1	*m*	1	*L*	*R*	
ПС	00	110		11	11100101		0	1	
1	00	101		11	11001010		0	1	*RGZ*:=*l*[*RGZ*].0 *RGPZ*:=*RGPZ* – 1
2	00	100		11	10010100		0	1	*RGZ*:=*l*[*RGZ*].0 *RGPZ*:=*RGPZ* – 1
3	00	011		11	00101000		0	0	*RGZ*:=*l*[*RGZ*].0 *RGPZ*:=*RGPZ* – 1

Рис. 6.8. Цифрова діаграма нормалізації мантиси результату

№ такту	*RGPZ*			*RGZ*			ЛПН	ППН	Мікрооперація
	n+2 *n*+1	*n*	1	*m*+2 *m*+1	*m*	1	*L*	*R*	
ПС	00	101		10	11100101		1	0	
1	00	110		11	011100101		0	0	*RGZ*:=RGZ(*m*+2).*r*[*RGZ*] *RGPZ*:=*RGPZ* + 1

Рис. 6.9. Цифрова діаграма нормалізації мантиси результату

Виконання нормалізації результату виконується шляхом зсуву регістру *RGZ*, що зберігає мантису результату, вліво (або вправо) і віднімання (або додавання) одиниці від регістру порядку результату *RGPZ* при кожному зсуві до тих пір, поки будуть зберігатися вказані комбінації цифр.

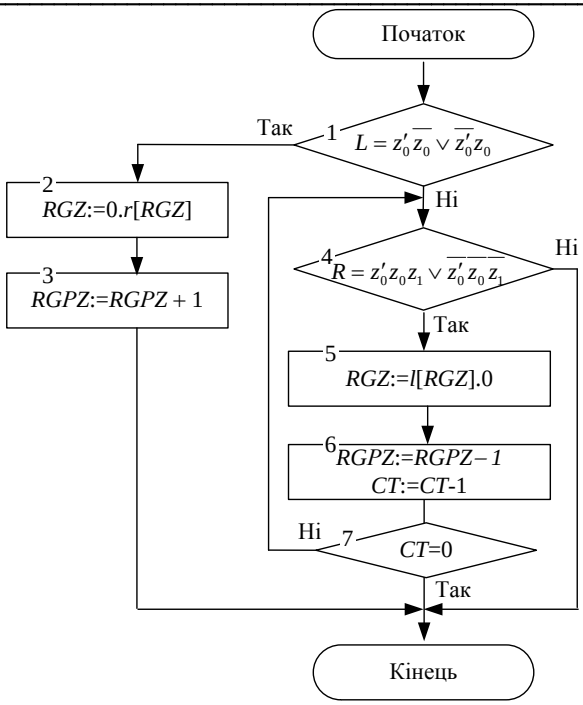


Рис. 6.10. Змістовний мікроалгоритм виконання операції нормалізації модифікованого зворотного коду

6.1.3. Округлення результату

Під час нормалізації результату шляхом зсуву вправо відкидання молодшого розряду може привести до накопичування великої додатної похибки. Для виключення цього застосовується операція округлення Суматор *SM* та регістр *RGZ* мають додатковий розряд зі сторони молодших розрядів. Округлення здійснюється додаванням у цей розряд одиниці і наступним відкиданням його вмісту. Таким чином, якщо в додатковому розряді була одиниця, то виникає одиниця переносу у молодший основний розряд, і похибка дорівнює ½ ваги молодшого розряду.

6.1.4. Операційний пристрій для додавання чисел із плаваючою комою

У загальному випадку операційний пристрій, що працює у режимі із плаваючою комою складається з двох функціональних частин: опе-

раційного пристрою порядків (ОПП), що виконує етап вирівнювання порядків і отримання порядку результату, та операційного пристрою для підсумовування мантис (*SMM*) та операційного пристрою нормалізації мантиси (ОПН) результату з подальшим його округленням (рис. 6.11).

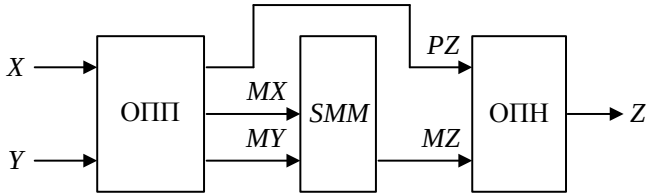


Рис. 6.11. Операційний пристрій для додавання чисел із плаваючою комою

6.2. 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

$$2^{P_Z} M_Z = 2^{P_X} M_X \cdot 2^{P_Y} M_Y = 2^{(P_X + P_Y)} \cdot (M_X \cdot M_Y).$$

Під час виконання операції множення мантис можна отримати результат із порушенням нормалізації вправо.

Можна виділити наступні *етапи множення чисел із плаваючою комою*: одержання порядку результату у вигляді суми порядків множників; знаходження мантиси результату, у вигляді добутку мантис множників; нормалізація результату, тобто приведення мантиси результату до вигляду $2^{-1} \le M_Z < 1$.

6.3. 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 $M_X < M_Y$, яка не завжди виконується при поданні мантис у нормалізованій формі. Тому перед началом ділення мантису діленого завжди зсу-

вають вправо, чим забезпечують зменшення її у два рази. Відповідно до порядку діленого додається одиниця, тобто

$$2^{P_X} M_X = 2^{P_X+1} \cdot M_X \cdot 2^{-1}.$$

Етапи ділення чисел із плаваючою комою наступні: одержання порядку результату із урахуванням виконання умови $M_X < M_Y$ шляхом віднімання порядку дільника від порядку діленого; одержання мантиси результату, на цьому етапі виконується ділення мантис; нормалізація результату. Нормалізація результату виконується аналогічно нормалізації під час множення чисел із плаваючою комою.

6.4. Добування квадратного кореня з числа із плаваючою комою

Добування квадратного кореня з числа $X = 2^{P_X} M_X$, що задане у форматі із плаваючою комою, можна подати у вигляді

$$Y = \sqrt{X} = \sqrt{2^{P_X} M_X} = 2^{P_X/2} \cdot \sqrt{M_X}\ .$$

Таким чином, для добування квадратного кореня з числа із плаваючою комою необхідно порядок числа поділити на два, а з мантиси добути квадратний корінь за правилами для чисел з фіксованою комою.

Ділення порядку на два відбувається шляхом зсуву його на один розряд вправо, якщо порядок парний. Якщо порядок непарний, то до нього необхідно додати одиницю, та зсунути мантису на один розряд вправо, після чого порядок зсувають на один розряд вправо – ділення на два. Тобто, якщо порядок $P_X = 2k–1$, то

$$X = 2^{2k-1} M_X = 2^{2k}(M_X \cdot 2^{-1}),\ Y = \sqrt{X} = 2^k \sqrt{M_X \cdot 2^{-1}}\ .$$

Мантиси чисел із плаваючою комою завжди нормалізовані і, коли у першому циклі з мантиси відбувається віднімання числа 0,01, перший залишок буде завжди додатнім, таким чином перша цифра результату завжди буде дорівнювати одиниці. Відповідно то цього при виконанні операції добування квадратного кореня у пристрої із плаваючою комою не може відбутися порушення нормалізації.

Етапи добування квадратного кореню з числа із плаваючою комою наступні: одержання порядку результату; одержання мантиси результату, яка є завжди нормалізованою, тому етап нормалізації не виконується.

Перед початком операції знак операнда та сам операнд аналізується на рівність нулю. За нульовою мантисою добування квадратного корені не відбувається, а результату привласнюється значення нуля. Якщо знак операнда зворотній, то пристрій генерує сигнал помилки.

6.5. Лабораторна робота 5

Ціль роботи: Дослідити способи побудови пристроїв для машинного виконання арифметичних операцій з плаваючою комою. Закріпити навички в проектуванні й налагодженні схем пристроїв з мікропрограмним управлінням.

Підготовка до роботи

1. Синтезувати операційну схему для виконання арифметичної операції із плаваючою комою. Задану операцію обрати відповідно до варіанту з табл. 5.1. Операцію додавання та нормалізації результату виконати у заданому машинному коді (табл. 6.2).

Таблиця 6.2. Варіанти завдання

a_5	a_4	Операція	a_6	a_0	Операнди	a_3	Код
0	0	C=A+B	0	0	A > 0; B > 0	0	ДК
			0	1	A < 0; B > 0		
0	1	C=A-B	1	0	A < 0; B < 0	1	ЗК
			1	1	A > 0; B < 0		
1	0	C=A∗B	A > 0; B > 0				
1	1	C=A/B					

2. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів. Розрядність операндів обрати з табл. 6.3.

3. Скласти змістовний мікроалгоритм виконання заданої операції.

4. Побудувати пристрій управління для виконання заданої операції. Тип пристрою обрати з табл. 6.4.

5. Під час виконання завдання необхідно враховувати дані наведені у табл. 4.2, для розробки БМУ, та у табл. 2.8 – 2.9 для розробки управляючого автомату. Тривалість виконання операції підсумовування прийняти два такти. Початкова адреса мікропрограми дорівнює

двом молодшим розрядам номеру залікової книжки поданого у шістнадцятирічній системі числення.

Таблиця 6.3. Варіанти завдання

a_6	Розрядність порядку	a_3	a_2	Розрядність мантиси
0	4	0	0	6
		0	1	5
1	3	1	0	8
		1	1	7

Таблиця 6.4. Варіанти завдання

a_4	a_1	Тип пристрою управління
0	0	Управляючий автомат Мілі
0	1	Управляючий автомат Мура
1	0	БМУ з відносною адресацією МК
1	1	БМУ з примусовою адресацією МК

Виконання роботи

1. Використовуючи моделюючу систему ПРОГМОЛС 2.0 побудувати і налагодити пристрій для виконання заданої операції. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.
2. На спроектованому пристрої виконати числовий приклад із заданими викладачем значеннями операндів.
3. У протоколі навести функціональну схему пристрою для виконання заданої операції.

Зміст звіту

Звіт з лабораторної роботи повинен включати короткі теоретичні відомості, необхідні для виконання лабораторної роботи; структурні та функціональні схеми; таблиці та діаграми, отримані при виконанні теоретичного завдання, а також у процесі моделювання схем; висновки за роботою.

Контрольні питання

1. Як у ЕОМ подаються числа із плаваючою комою?
2. Як виконуються операції додавання та віднімання чисел із плаваючою комою у ЕОМ?

3. Як виконується операція множення чисел із плаваючою комою у ЕОМ?

4. Як виконується операція ділення чисел із плаваючою комою у ЕОМ?

5. Як виконується етап зрівняння порядків у розробленому пристрої?

6. Наведіть склад устаткування необхідного для реалізації розглянутих операцій з числами з плаваючою комою.

7. Наведіть склад устаткування необхідного для побудови реалізації пристрою управління у даній роботі.

8. Наведіть ознаки порушення нормалізації. Як виконується нормалізація результату у даній роботі?

9. Поясніть, як забезпечити необхідну тривалість виконання мікрооперації в БМУ.

РОЗДІЛ 7

Завдання до виконання практичних робіт

7.1. Практична робота 1

РОЗРОБКА ФУНКЦІОНАЛЬНОЇ СХЕМИ АРИФМЕТИКО-ЛОГІЧНОГО ПРИСТРОЮ ІЗ РОЗПОДІЛЕНОЮ ЛОГІКОЮ

1. Розробити операційну схему і змістовний мікроалгоритм виконання операції множення додатних чисел (табл. 7.1), де a_6, …, a_1 – молодші розряди двійкового номера залікової книжки. При розробці урахувати що розрядність регістрів, суматорів та лічильників дорівнює восьми.

2. Розробити структурну та функціональну схему операційного пристрою із розподіленою логікою для виконання операції множення.

3. Виконати моделювання роботи пристрою за допомогою діаграми стану регістрів із заданими значеннями значенням операндів. Розрядність операндів обрати у табл. 7.2

4. Виконати синтез управляючого пристрою із жорсткою логікою. Тип управляючого автомата і тригерів для реалізації пам'яті автомата обрати з табл. 7.3 – 7.4.

5. Побудувати структурну та функціональну схеми арифметико-логічного пристрою для виконання операції множення.

Таблиця 7.1. Варіанти завдання

a_6	a_5	a_4	Спосіб множення	Розрядність операндів
0	0	0	2	16
0	0	1	3	8
0	1	0	1	16
0	1	1	4	8
1	0	0	1	8
1	0	1	2	16
1	1	0	3	8
1	1	1	4	16

Таблиця 7.2. Варіанти завдання

a_6	a_3	a_2	Значення операндів	
0	0	0	34*h*	17*h*
0	0	1	29*h*	1*eh*
0	1	0	3*fh*	05*h*
0	1	1	1*dh*	0*ch*
1	0	0	19*h*	0*fh*
1	0	1	29*h*	15*h*
1	1	0	4*ah*	07*h*
1	1	1	4*ch*	48*h*

Таблиця 7.3. Варіанти завдання

a_3	a_2	Тип тригера
0	0	JK
0	1	T
1	0	RS
1	1	D

Таблиця 7.4. Варіанти завдання

a_1	Тип автомата
0	Мили
1	Мура

7.2. Практична робота 2

СИНТЕЗ БЛОКІВ МІКРОПРОГРАМНОГО УПРАВЛІННЯ

1. Обчислити варіант для виконання практичної роботи, для чого необхідно номер залікової книжки подати у двійковій системі числення.

2. За табл. 7.5, де a_6, …, a_1 відповідають молодшим розрядам двійкового номера залікової книжки, обрати номери завдань, що надані у розділі 7 “Задачі для самостійного розв’язування”.

3. За отриманими завданнями виконати синтез блоків мікропрограмного управління.

Таблиця 7.5. Варіанти завдання

a_6	a_5	a_4	Номери задач			
			7.1	7.2	7.3	7.4
0	0	0	a	a, f	c, e	c, f
0	0	1	b	b, e	a, f	a, e
0	1	0	c	c, f	b, f	b, e
0	1	1	a	a, e	d, f	d, e
1	0	0	d	d, e	b, e	b, f
1	0	1	b	b, f	d, e	d, f
1	1	0	c	c, e	c, f	c, e
1	1	1	d	d, f	a, e	a, f

7.3. Практична робота 3

РОЗРОБКА ЕЛЕКТРИЧНИХ СХЕМ ПРИСТРОЮ З МІКРОПРОГРАМНИМ УПРАВЛІННЯМ

1. Побудувати структурну схему БМУ і карту пам'яті мікропрограм для мікроалгоритму виконання операції множення. БМУ повинен забезпечити управління операційним пристроєм із розподіленою логікою відповідно до варіанту практичної роботи 1.

2. Під час виконання завдання необхідно врахувати вихідні дані наведені у табл. 7.6 – 7.7.

3. Розробити структурну, функціональну та принципову схеми обчислювального пристрою, що складається з операційного пристрою (розробленого у практичній роботі 1) та блоку мікропрограмного управління.

Таблиця 7.6. Вихідні дані до проектування

a_4	a_2	Спосіб адресації мікрокоманд	Структура ПМК	Ємність ПМК (слів)	Використати зону β_4 для перевірки слова МК
0	0	примусовий	лінійна	64	на непарність
0	1	примусовий	матрична		на парність
1	0	відносна	лінійна		на непарність
1	1				на парність
Спосіб мікропрограмування – горизонтальний;					
Забезпечити занесення початкової адреси мікроалгоритму в регістр адреси мікрокоманд.					

Таблиця 7.7. Вихідні дані до проектування

a_6	a_5	a_4	Тривалість мікрооперації підсумовування	Початкова адреса мікропрограми
0	0	0	7	18*h*
0	0	1	4	0*ah*
0	1	0	3	06*h*
0	1	1	6	0*eh*
1	0	0	11	13*h*
1	0	1	4	07*h*
1	1	0	5	11*h*
1	1	1	2	0*bh*

7.4. Загальні вказівки до виконання практичних робіт

Звіти з практичних робіт повинні включати структурні схеми розроблених операційних та управляючих пристроїв; функціонально-структурні та закодовані мікроалгоритми їх функціонування, таблиці та діаграми, отримані при виконанні синтезу пристроїв, формати мікрокоманд та карти розміщення мікрокоманд у пам'яті для БМУ, структурні, функціональні та принципові (відповідно до завдання) електричні схеми обчислювальних пристроїв розроблені відповідно до вимог встановлених Єдиною системою конструкторської документації (ЄСКД), та описи функціональних схем.

Правила оформлення електричних схем наведені у розділі 10. У додатку Ж наведені умовні графічні позначення основних елементів цифрової техніки та типи мікросхем.

Наведемо приклади побудови функціональних та принципових схем обчислювальних пристроїв для множення чисел.

Приклад 7.1. Розробити функціональну схему арифметичного пристрою для виконання операції множення. Для управління обчисленням застосувати управляючий автомат.

Виконання завдання

Функціональна схема арифметичного пристрою для виконання операції множення за другим способом наведена у додатку А. Операндами є восьми-розрядні додатні правильні дроби подані у формі з фіксованою комою. Пристрій складається з операційного пристрою та управляючого пристрою (рис. 2.2). Структурна схема ОПр, що реалізує множення за другим способом зображена на рис. 2.1, *б*. Операційний пристрій реалізований у вигляді АЛП із розподіленою логікою. Для управління операційним пристроєм використовується управляючий автомат Мура. Синтез управляючого автомату виконаний за методом декомпозиції тригерів, описаним у [6].

Опис функціональної схеми

Пристрій множення складається з регістрів *RG* – елементи 1, 2, 3, 8, 9; суматорів *SM* – елементи 21, 22; тригерів *TT* – елементи 6, 23, 24, 25; лічильника тактів *CT* – елемент 4; логічних елементів І та АБО – елементи 5, 7, 10 – 20, 26 – 28.

Функціонально пристрій множення поділяється на дві частини – операційна частина та управляючий автомат.

Суматори, регістри 1, 2, 3, 8, 9, лічильник циклів 4, тригер 6 та логічний елемент 5 складають операційну частину пристрою множення.

Регістри 1, 2 призначенні для зберігання значення множеного, регістр 3 – для зберігання значення множника, регістри 8, 9 – для накопичування сум часткових добутків та отримання результату множення. Суматори 21, 22 застосовуються для обчислення сум часткових добутків. Лічильник 4 – для підрахунку кількості циклів множення, логічні елементи 5, 7 – для формування логічної умови на виході лічильника.

У склад управляючого автомату входять тригери 23, 24, 25 та логічні елементи 10 – 20, 26 – 28.

На входи пристрою 1 – 20 поступають вхідні дані – розряди множеного $X[0] – X[7]$, розряди множника $Y[0] – Y[7]$, кількість циклів $CT[0] – CT[3]$. На входи 21, 22 подаються тактовий сигнал *CLK* генератора та сигнал початкового встановлення тригерів *R*. На виходах 73 – 88 формується розряди результату $F[0] – F[15]$.

У кожному такті роботи пристрою відповідно до мікроалгоритму управління управляючі сигнали з виходів автомату 37 – 40 поступають на управляючі входи елементів операційного пристрою де відбуваються відповідні дії.

Приклад 7.2. Розробити функціональну схему обчислювального пристрою для виконання операції множення. Для управління обчисленням застосувати блок мікропрограмного управління.

Виконання завдання

У додатках Б та В наведені відповідно функціональна та принципова електричні схеми арифметичного пристрою для виконання операції множення за другим способом. Операційна частина реалізована аналогічно, як у прикладі 7.1. Для управління АЛП використовується блок мікропрограмного управління з примусовим способом адресації мікрокоманд та ПМК із двовимірною матричною структурою. Синтез БМУ для управління пристроєм множення виконуємо аналогічно, як показано у прикладі 3.8. Синтез БМУ з матричною ПМК приведений у прикладі 3.6. Структурна схема розробленого БМУ зображена на рис. 7.1.

Закодований структурний мікроалгоритм управління та схема розміщення мікрокоманд у ПМК наведені на рис. 7.2 – 7.3. відповідно.

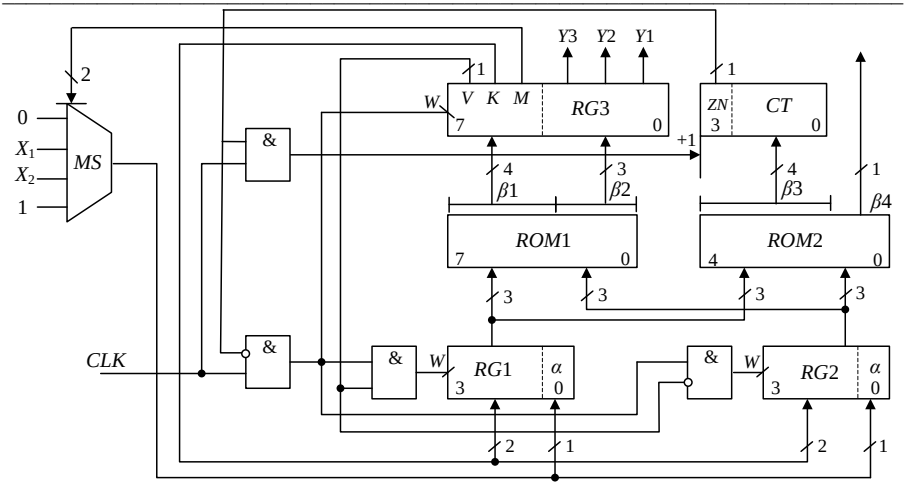


Рис. 7.1. Структурна схема БМУ

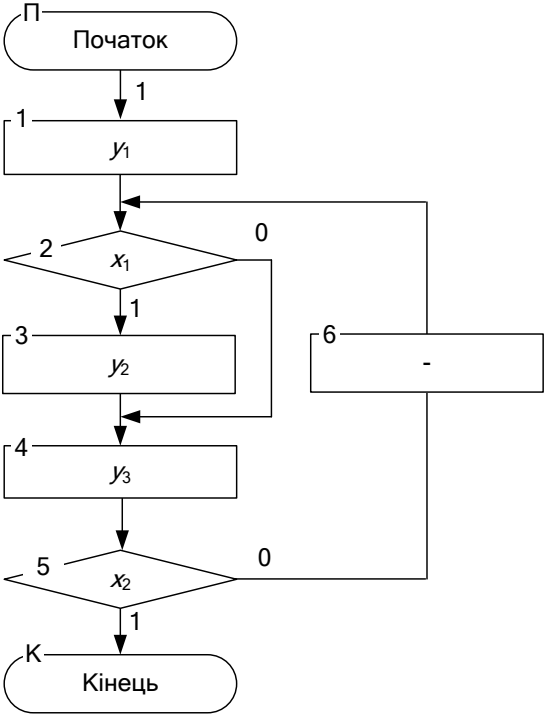


Рис. 7.2. Закодований алгоритм управління пристроєм множення

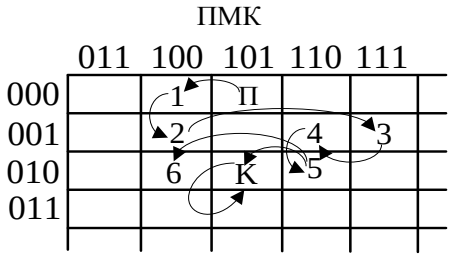


Рис. 7.3. Карта розміщення мікрокоманд у ПМК

Опис функціональної схеми

Пристрій множення складається з регістрів *RG* – елементи 1, 2, 3, 8, 9, 16, 17, 20; суматорів *SM* – елементи 21, 22; тригеру *TT* – елемент 6; мультиплексору *MS* – елемент 10; лічильників тактів *CT* – елементи 4, 5; постійних запам'ятовуючих пристроїв *ROM* – елементи 18, 19; логічних елементів І та АБО – елементи 7, 11, 12, 13, 14, 15.

Опис операційного пристрою множення наведений у прикладі 7.1.

Пристрій управління складається з наступних функціональних складових частин.

Пам'ять мікрокоманд блоку мікропрограмного управління реалізована за допомогою постійно запам'ятовуючих пристроїв 18, 19. У регістрах 16, 17 формується адреса наступної мікрокоманди. Регістр 20 призначений для реалізації регістру мікрокоманди. Мультиплексор 10 застосовується для формування умови переходу при визначенні адреси наступної мікрокоманди. Лічильник 5 застосовується для реалізації затримки управляючих сигналів. Логічні елементи призначенні для реалізації частини логіки роботи пристрою управління.

На початку роботи пристрою із ПМК БМУ зчитується перша мікрокоманда, частина якої за від'ємним перепадом *CLK* записується у регістр мікрокоманди 20, а зона формування затримки – у лічильник тактів затримки виконання мікрокоманди 5. З виходу 53 регістру 20 управляючий сигнал поступає на управляючі входи *W* запису інформації регістрів 2, 3, лічильника циклів 4 та через інвертор 15 на входи *R* скидання у нуль регістрів 1, 8, 9. У цьому такті відбувається занесення вихідних даних.

Сигнали з виходів 46 – 50 регістру 20, що відповідають сигналам зони формування адреси наступної мікрокоманди, поступають на схему формування адреси наступної мікрокоманди (СФАМК), що

складається з регістрів 16, 17 та логічних елементів 13, 14. Ураховуючи логічну умову, що формується на виході 56 мультиплексора 10, на виході регістрів 16, 17 формується адреса наступної мікрокоманди.

На наступному такті при відповідній логічній умові виконується мікрокоманда підсумовування на суматорах 21, 22, в результаті якої у регістрах 8, 9 формується черговий частковий добуток. Під час цього з виходу 52 регістру 20 управляючий сигнал поступає на управляючі входи *W* запису інформації регістрів 8, 9, куди записується результат підсумовування з інформаційних виходів суматорів 21, 22.

Операція підсумовування має тривалість декілька тактів, тому у лічильнику тактів 5 при виконанні мікрокоманди підсумовування буде записане значення відмінне від нуля. Тому одиниця з виходу 55 лічильника 5 поступає на вхід логічного елементу 12, за цього блокується сигнал *CLK*, що має поступити на вхід *W* СФАМК, але при цьому сигнал *CLK* поступає на вхід «+1» лічильника 5, де відбувається інкремент його вмісту. За нульовим вмістом лічильника, тобто коли лічильник відрахує записану у нього кількість тактів, нульовий сигнал з виходу 55 навпаки заблокує вхід «+1» лічильника 5 за допомогою логічного елементу 11 та дозволить надходження сигналу *CLK* на вхід *W* СФАМК, тобто сформується адреса наступної мікрокоманди.

Далі під управлінням наступної мікрокоманди виконується зсув регістрів 1, 2, 3, та декремент лічильника циклів 4. За цього управляючий сигнал надходить на відповідні управляючі входи регістрів з виходу 51 регістру мікрокоманди 20.

У цьому такті на виході 27 логічного елементу 7 формується логічна умова, що відображує вміст лічильника циклів. За вмістом лічильника відмінним від нуля – логічний сигнал дорівнює нулю, відбувається формування наступної мікрокоманди чергового циклу обчислення. За нульовим вмістом лічильника обчислення закінчуються і результат з інформаційних виходів регістрів 8, 9 поступає на виходи пристрою.

РОЗДІЛ 8

Задачі для самостійного розв'язування

8.1. Розробити структуру зони $\beta 2$ формування управляючих сигналів БМУ для реалізації заданого мікроалгоритму з максимальною швидкодією (у дужках указані управляючі сигнали, що формуються в одному такті):

a) П (Y1Y2Y3)(Y1Y2)Y1Y2(Y1Y3)Y5(Y1Y2Y6)Y5Y1Y2K;
b) П (Y1Y2Y3)Y1Y2(Y4Y2Y1)Y3(Y5Y1)(Y2Y1Y5)Y4(Y2Y6)K;
c) П (Y1Y2)Y3(Y1Y2)Y1Y2(Y1Y3)Y5(Y1Y2Y6)Y5Y1Y2K;
d) П (Y1Y3)Y2(Y1Y4)Y2(Y1Y6)Y5(Y1Y5Y6)Y3Y1Y2K.

8.2. Розробити структуру зони $\beta 3$ формування тривалості управляючих сигналів БМУ та карту програмування при асинхронному способі управління для реалізації заданого мікроалгоритму з максимальною швидкодією:

a) П (Y1Y2Y3)(Y1Y2)Y1Y2(Y1Y3)Y5(Y1Y2Y6)Y5Y1Y2K;
b) П (Y1Y2Y3)Y1Y2(Y4Y2Y1)Y3(Y5Y1)(Y2Y1Y5)Y4(Y2Y6)K;
c) П (Y1Y2)Y3(Y1Y2)Y1Y2(Y1Y3)Y5(Y1Y2Y6)Y5Y1Y2K;
d) П (Y1Y3)Y2(Y1Y4)Y2(Y1Y6)Y5(Y1Y5Y6)Y3Y1Y2K.

Вихідні дані:

Тривалість управляючих сигналів – $t(y2, y3,) = 4$ тактів,
$t(y4) = 1$ такт,
$t(y5, y6) = 21$ такт,
$t(y1, y2, y4,) = 4$ тактів,
$t(y3) = 1$ такт,
$t(y5, y6) = 27$ тактів.

8.3. Розробити структуру БМУ і карту програмування ПМК для заданого мікроалгоритму:

a) $\Pi x1 \overset{1}{\uparrow} y1 \overset{1}{\downarrow} x2 \overset{2}{\uparrow} y1 \overset{3}{\uparrow} \overset{2}{\downarrow} y2 \overset{3}{\downarrow} K$;

b) $\Pi \overset{2}{\downarrow} \overset{1}{\downarrow} y1 x1 \overset{1}{\uparrow} (y2, y3) x2 \overset{2}{\uparrow} y1 K$;

c) $П y1 \overset{3}{\downarrow} y2\, x1 \overset{1}{\uparrow} (y1, y2) \overset{2}{\uparrow} \overset{1}{\downarrow} y2 \overset{2}{\downarrow} x2 \overset{3}{\uparrow} К$;

d) $П(y1, y2) \overset{3}{\downarrow} x1 \overset{1}{\uparrow} y2 \overset{2}{\uparrow} \overset{1}{\downarrow} y3 \overset{2}{\downarrow} x2 \overset{3}{\uparrow} К$.

Вихідні дані:

Спосіб адресації мікрокоманд –	примусовий;
Структура ПМК –	e) двомірна;
	f) лінійна;
Ємність ПМК –	e) 64 слова;
	f) 32 слова;
Спосіб мікропрограмування –	e) горизонтальний;
	f) вертикальний;
Тривалість управляючих сигналів –	e) $t(y1)$ = 1 такт; $t(y2)$ = 4 такти;
	f) $t(y1)$ = 1 такт; $t(y2)$ = 5 тактів;
Початкова адреса мікропрограми –	e) $5_{(10)}$;
	f) $9_{(10)}$;

Забезпечити контроль слова МК на парність.

8.4. Розробити схему БМУ для реалізації заданого мікроалгоритму. Розробити карту настроювання БМУ.

a) $П x1 \overset{1}{\uparrow} (y1, y3, y5) \overset{1}{\downarrow} x2 \overset{2}{\uparrow} (y3, y6) \overset{3}{\uparrow} \overset{2}{\downarrow} (y1, y2, y7)(y2, y4) \overset{3}{\downarrow} К$;

b) $П(y1, y2, y3) x1 \overset{1}{\uparrow} (y3, y4, y5) \overset{2}{\uparrow} \overset{1}{\downarrow} \overset{3}{\downarrow} (y1, y4, y3) \overset{2}{\downarrow} x2 \overset{3}{\uparrow} (y2, y6) К$;

c) $П(y2, y3) x1 \overset{1}{\uparrow} (y1, y4, y5) x2 \overset{2}{\uparrow} (y4, y5, y7) \overset{3}{\uparrow} \overset{2}{\downarrow} \overset{1}{\downarrow} (y6, y7)(y1, y3) \overset{3}{\downarrow} К$;

d) $П x1 \overset{1}{\uparrow} (y2, y4, y6)(y1, y4) x2 \overset{2}{\uparrow} (y4, y5, y7) \overset{3}{\uparrow} \overset{2}{\downarrow} \overset{1}{\downarrow} (y1, y7) \overset{3}{\downarrow} (y3, y4, y6) К$.

Вихідні дані:

Спосіб адресації мікрокоманд –	відносний;	
Емність ПМК –	д) 64 слова;	
	e) 32 слова;	
Спосіб мікропрограмування –	комбінований;	
Тривалість управляючих сигналів –	e) $t(y_1)$=1, $t(y_2)$=2,	f) $t(y_1)$ = 1, $t(y_2)$=2,

	$t(y_3)=5$,	$t(y_3)=5$,
	$t(y_4)=1$,	$t(y_4)=1$,
	$t(y_5)=12$,	$t(y_5)=21$,
	$t(y_6)=7$,	$t(y_6)=6$,
	$t(y_7)=9$;	$t(y_7)=9$;

Початкова адреса мікропрограми – д) $11_{(10)}$;
е) $4_{(10)}$;

Забезпечити контроль слова МК на непарність.

8.5. Синтезувати операційну схему для обчислення добутку $Z=Y\times X$ двох правильних дробів $Y = 0,y_1y_2\ldots y_n$ та $X = 0,x_1x_2\ldots x_n$. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

Вихідні дані:

розрядність дробів (n) –
- e) 6;
- f) 8;
- g) 4;
- h) 3;

операцію множення реалізувати –
- i) першим способом;
- j) другим способом;
- k) третім способом;
- l) четвертим способом.

8.6. Синтезувати операційну схему для обчислення значення функції D, якщо її аргументами є правильні дроби $A = 0,a_1a_2\ldots a_n$, $B = 0,b_1b_2\ldots b_n$ та $C = 0,c_1c_2\ldots c_n$. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

Вихідні дані:

розрядність дробів (n) –
- a) 6;
- b) 8;
- c) 4;
- d) 3;

реалізувати функцію –
- e) $D=2C-4AB$;
- f) $D=A(B-1)+0,5C$;
- g) $D=2A(B+1)+0,5C$;
- h) $D=2C-A(B+1)$.

8.7.** Синтезувати операційну схему для машинного перетворення чисел методом «зсуву-корекції». Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

Вихідні дані:

Реалізувати перетворення чисел –	a) із двійкової системи числення у десяткову; b) із десяткової системи числення у двійкову;
Двійково-десяткове число містить –	c) дві тетради; d) три тетради.

8.8. Синтезувати операційну схему для обчислення значення $Z = \frac{Y}{X}$ двох правильних додатних дробів $Y = 0,y_1y_2\ldots y_n$ та $X = 0,x_1x_2\ldots x_n$, якщо $Y < X$. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

Вихідні дані:

розрядність дробів (n) –	a) 6; b) 4;
операцію ділення реалізувати –	c) способом ділення із зсувом дільника; d) способом ділення із зсувом залишку.

8.9.** Синтезувати операційну схему для обчислення значення $Y = \sqrt{X}$ додатного числа $X = 0,x_1x_2\ldots x_n$. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

Вихідні дані:

розрядність числа (n) –	a) 6; b) 4.

8.10.** Синтезувати операційну схему для виконання операцій додавання та віднімання чисел із плаваючою комою. Виконати логічне

моделювання роботи пристрою за допомогою цифрової діаграми із заданими викладачем значеннями операндів.

8.11. Побудувати пристрій управління для управління розробленим арифметико-логічним пристроєм.

Вихідні дані:

Арифметико-логічний пристрій –
a) завдання 10.5;
b) завдання 10.6;
c) завдання 10.7**;
d) завдання 10.8;
e) завдання 10.9**;
f) завдання 10.10**;

Тип пристрою управління –
g) управляючий автомат Мілі;
h) управляючий автомат Мура;
i) БМУ з відносною адресацією МК;
j) БМУ з примусовою адресацією МК.

8.12. Розробити змістовний мікроалгоритм та мікропрограму у мнемонічних кодах для виконання заданої операції у АЛП із зосередженою логікою. Побудувати структурно-функціональну схему АЛП із заданою розрядністю НОЗП. За потреби обрати спосіб множення або ділення у задачах 8.5, 8.8.

Вихідні дані:

Арифметична операція –
a) $RZ:=RX+2RY+1$;
b) $RZ:=4(RX+2RY)-1$;
c) $RZ:=0{,}5(RX+RY)$;
d) $RZ:=2RX-0{,}5RY$;
e) $RZ:=RX\times RY$;
f) $RZ:=RX\times RY+1$;
g) $RZ:=RX/RY$;
h) $RZ:=\sqrt{RX}$.

Номери регістрів (*RX*,*RY*,*RZ*)–
i) *R*1,*R*3,*R*4;
j) *R*0,*R*2,*R*5;
k) *R*2,*R*0,*R*4;
l) *R*5,*R*3,*R*1;

Результат подати у машинному
m) ПК;

коді –	n) ЗК; o) ДК;
Тип АЛП –	p) АЛП із двоспрямованою внутрішньою магістраллю та одноадресним НОЗП; q) АЛП із односпрямованою магістраллю та одноадресним НОЗП; r) АЛП із односпрямованою магістраллю та двоадресним НОЗП;
Розрядність НОЗП –	s) 8-ми розрядне; t) 16-ти розрядне.

****** – задачі підвищеної складності.

РОЗДІЛ 9

Завдання до виконання розрахунково-графічної роботи

Ціль виконання РГР. Розрахунково-графічна робота (РГР) з курсу «Цифрові ЕОМ» виконується за індивідуальним завданням і є самостійною роботою студента. Вона призначена для розширення, закріплення, узагальнення й практичного застосування знань, умінь і навичок, отриманих студентом при вивченні курсу. У процесі виконання роботи студент повинен також навчитися користуватися довідковою літературою й вивчити процес створення проектно-конструкторської документації відповідно до діючих стандартів.

РОЗРОБКА ОБЧИСЛЮВАЛЬНОГО ПРИСТРОЮ З МІКРОПРОГРАМНИМ УПРАВЛІННЯМ

1. Розробити операційний пристрій для обчислення добутку $Z=Y\times X$ двох правильних дробів $Y = 0,y_1y_2\ldots y_n$ та $X = 0,x_1x_2\ldots x_n$, спосіб множення обрати відповідно до варіанту з табл. 9.1. Виконати логічне моделювання роботи пристрою за допомогою цифрової діаграми із заданими в табл. 9.2 значеннями операндів.

Таблиця 9.1. Варіанти завдання

a_6	a_5	a_4	Спосіб множення	Розрядність операндів
0	0	0	1	16
0	0	1	2	8
0	1	0	3	16
0	1	1	4	8
1	0	0	1	8
1	0	1	2	16
1	1	0	3	8
1	1	1	4	16

Таблиця 9.2. Варіанти завдання

a_6	a_3	a_2	Значення операндів	
0	0	0	34*h*	17*h*
0	0	1	29*h*	1*eh*
0	1	0	3*fh*	05*h*
0	1	1	1*dh*	0*ch*
1	0	0	19*h*	0*fh*
1	0	1	29*h*	15*h*
1	1	0	4*ah*	07*h*
1	1	1	4*ch*	48*h*

2. Побудувати структурну схему БМУ і карту пам'яті мікропрограм для мікроалгоритму виконання операції множення. БМУ повинен забезпечити управління розробленим операційним пристроєм.

3. Під час виконання завдання необхідно врахувати вихідні дані наведені у табл. 9.3 – 9.4.

Таблиця 9.3. Вихідні дані до проектування

a_4	a_2	Спосіб адресації мікрокоманд	Структура ПМК	Ємність ПМК (слів)	Використати зону β_4 для перевірки слова МК
0	0	примусовий	лінійна	64	на непарність
0	1	примусовий	матрична		на парність
1	0	відносна	лінійна		на непарність
1	1				на парність
Спосіб мікропрограмування – горизонтальний;					
Забезпечити занесення початкової адреси мікроалгоритму в регістр адреси мікрокоманд.					

Таблиця 9.4. Вихідні дані до проектування

a_6	a_5	a_4	Тривалість мікрооперації підсумовування	Початкова адреса мікропрограми
0	0	0	7	18*h*
0	0	1	4	0*ah*
0	1	0	3	06*h*
0	1	1	6	0*eh*
1	0	0	11	13*h*
1	0	1	4	07*h*
1	1	0	5	11*h*
1	1	1	2	0*bh*

4. Налагодити розроблений пристрій за допомогою моделюючої програми ПРОГМОЛС 2.0. Опис програмного комплексу ПРОГМОЛС 2.0 наведений у додатку М.

5. Побудувати функціональну та принципову схеми розробленого пристрою, що складається з операційного пристрою та блоку мікропрограмного управління.

Зміст розрахунково-графічної роботи

РГР оформлюється у вигляді комплекту текстових і графічних документів.

РГР повинна містити наступні текстові і графічні документи (документи наведені у порядку їх комплектування).

- Титульний лист;
- Індивідуальне завдання на розробку, підписане викладачем;
- Опис альбому;
- Обчислювальний пристрій, схема електрична структурна;
- Обчислювальний пристрій, схема електрична функціональна;
- Пояснювальна записка і додатки до неї.

Зразок оформлення *титульного аркуша* наведений у додатку З.

Індивідуальне завдання розробляється на підставі вихідних даних до виконання РГР у відповідності до варіанту. Індивідуальне завдання до виконання РГР оформлюється у вигляді, наведеному у додатку І, підписується викладачем та студентом, та додається до альбому.

Опис альбому містить перелік усіх документів альбому. Приклад оформлення опису альбому наведений у додатку К.

Пояснювальна записка повинна містити наступні розділи.

ВСТУП
1.1. Розробка операційного пристрою виконання операції множення.
1.2. Розробка структури БМУ та її обґрунтування.
1.3. Формат мікрокоманди.
1.4. Опис функціональної схеми
ВИСНОВКИ
ПЕРЕЛІК ЛІТЕРАТУРИ
ДОДАТКИ

У додатках необхідно представити роздруківку функціональної схеми пристрою множення та вмісту ПМК, отриманих при моделюванні роботи обчислювального пристрою у програмі ПРОГМОЛС 2.0.

РОЗДІЛ 10

Вимоги до оформлення конструкторської документації

Пояснювальна записка та технічне завдання належать до текстової конструкторської документації.

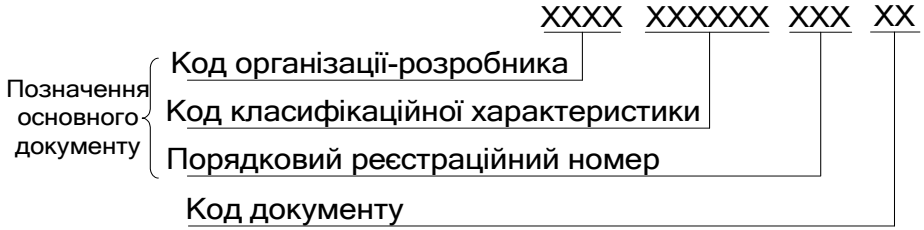
Порядок побудови розділів і підрозділів текстових документів, правила викладу тексту, розрахунків, а також побудови таблиць повинні цілком відповідати вимогам діючих стандартів щодо правил оформлення конструкторської документації.

Текст виконується авторським рукописом чорним чорнилом чи роздруковується на принтері.

Текстова документація супроводжується основним написом. Основні написи розміщують у правому нижньому куті документа.

Приклади основних написів для текстових документів, а також для графічних документів наведені у додатку Л.

Уся технічна документація повинна мати позначення. Структуру позначення для основних конструкторських документів встановлено відповідно до діючих стандартів. Під час виконання РГР прийнято наступну умовну структуру позначення основних конструкторських документів:



Код організації розробника визначається скороченою назвою університету.

Перші дві цифри коду класифікаційної характеристики відображують рік виконання РГР, наступні чотири – номер залікової книжки студента.

Порядковий реєстраційний номер присвоюють документам в межах одного альбому.

Позначення конструкторських документів, складається із позначення основного документу і коду документу, встановленого відповідними стандартами. Конструкторські документи, з яких складається РГР мають наступні коди:

- опис альбому – ОА;
- пояснювальна записка – ПЗ;
- схема електрична структурна – Е1;
- схема електрична функціональна – Е2;
- схема електрична принципова – Е3.

Приклади позначень, що прийняті для позначення конструкторських документів при виконанні курсових та розрахунково-графічних робіт:

- позначення технічного завдання:

НАУ 07 4512 002 ТЗ або НТУУ КПІ 07 4512 002 ТЗ,

- позначення схеми електричної функціональної:

НТУУ КПІ 07 4512 003 Е2 або НАУ 07 4512 003 Е2.

Перед комплектацією всі документи альбому повинні бути підписані виконавцем і керівником на титульний аркуші та в основних написах текстових і графічних документів.

Підпис керівника про допуск до захисту роботи ставиться після остаточного оформлення альбому.

Укомплектовані документи слід виконати на аркушах формату А4 і скріпити в один альбом.

РОЗДІЛ 11

Вимоги до оформлення електричних схем

Схема – це графічний конструкторський документ, на якому показані у вигляді умовних графічних зображень або позначень складові частини пристрою і зв'язки між ними.

Структурна схема визначає основні функціональні частини пристрою, їх призначення та взаємозв'язок. Структурні схеми розробляють на етапі проектування пристрою, на стадіях, що передують розробці схем інших типів.

Функціональна схема пояснює визначені процеси, що відбуваються в окремих функціональних ланцюгах виробу або у виробі в цілому. Функціональними схемами користуються для вивчення принципів роботи пристрою, а також при налагоджені та ремонті.

Принципова схема визначає повний склад елементів та зв'язків між ними і, зазвичай, дає детальне уявлення за принципів роботи виробу.

На структурних схемах, функціональні частини зображують у вигляді прямокутників або умовних графічних позначень, із зазначенням найменування кожної функціональної частини або її умовного позначення. Наприклад, схеми електричні структурні пристроїв множення зображені на рис. 2.1.

На функціональних схемах елементи дозволяється зображувати також у вигляді прямокутників та УГП. Лінії зв'язку підводять наступним чином: інформаційні лінії підводять до більшої сторони УГП, управляючі – до меншої сторони. Приклад функціональної схеми пристрою множення зображений на рис. 2.5.

На електричних функціональних та принципових схемах функціональні частини зазвичай зображують за допомогою УГП елементів цифрової техніки. Наведемо основні правила побудови таких УГП.

УГП елемента має форму прямокутника, до якого підводять лінії виводів та містить три поля: основне і два додаткові, які розміщують зліва і справа від основного (рис. 11.1).

В основному полі розміщують позначення функції, що виконує елемент. У додаткових полях розміщують інформацію за призначення

виводів (мітки, позначення виводів). УГП може складатись також тільки з основного поля, або з основного поля і одного додаткового, яке розміщують справа або зліва від основного.

Усім функціональним частинам і функціональним групам допускається присвоювати порядкові номери в послідовності зверху вниз у напрямі зліва направо. Елементи на функціональних схемах нумеруються натуральними числами, на функціональних – позначеннями *D*1, *D*2, *D*3, … .

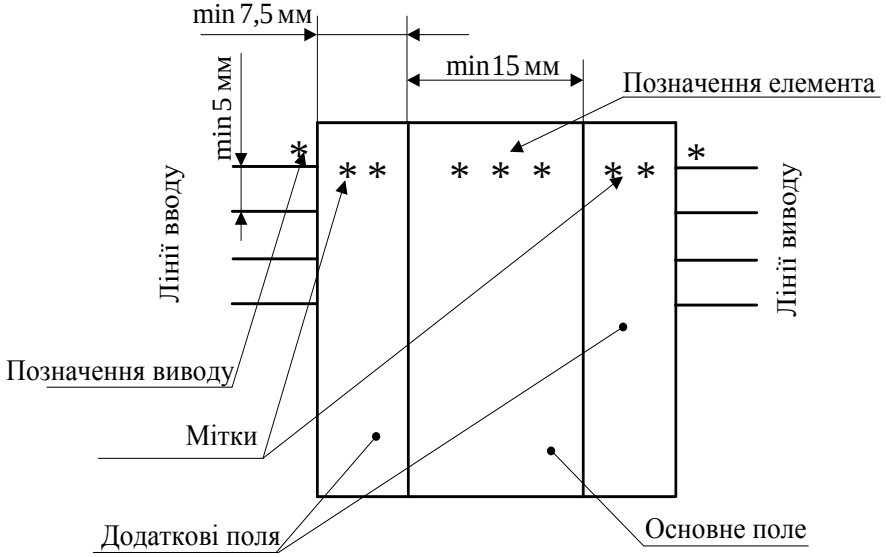


Рис. 11.1. УГП елемента цифрової техніки

Виводи елементів поділяють на входи, виходи. Входи елемента зображують із лівої сторони УГП, виходи – з правої. У разі підведення ліній виводів до контуру УГП **не допускається** проводити їх на рівні сторони прямокутника та проставляти на них стрілки, що вказують напрям інформації.

Розміри УГП визначають: по висоті – за кількістю ліній виводів, інтервалів та рядків в основному та додаткових полях, за розміром шрифту; по ширині – наявністю додаткових полів, кількістю знаків в одному рядку (з урахуванням пропусків) та розміром шрифту. Взагалі розмір основного та додаткових полів має бути *min* 15 мм та *min* 7,5 мм відповідно. Відстань між лініями виводів дорівнює *min* 5 мм.

Товщини ліній вибирають залежно від формату схеми і розмірів УГП. Оптимальна товщина лінії s= 0,3...0,4 мм. Лінії зв'язку, шини, лінії УГП виконують лініями однієї і тієї ж товщини – суцільною тонкою лінією товщиною s. Допускається для зображення шин застосовувати потовщені 2s і товсті 3s...4s лінії.

Приклади функціональної та принципової електричних схем наведені у додатках А, Б, В.

Список літератури

1. *Бабич М.П., Жуков І.А.* Атестаційні роботи магістрів і спеціалістів: Навчально-методичний посібник. – К. НАУ, 2004. – 216 с.

2. *Жабін В.І, Ткаченко В.В.* Однокристальные и микропрограммируемые ЭВМ. – К.: Діалектика, 1995 – 115 с.

3. *Каган Б.М.* Электронные вычислительные машины и системы. – М.: Энергоатомиздат, 1985. – 552 с.

4. *Карцев М.А.* Архитектура цифровых вычислительных машин.– М.: "Наука", 1978. – 295 с.

5. *Молчанов А.А., Корнейчук В.И., Тарасенко В.П.* Справочник по микропроцессорным устройствам. – К.: Техніка, 1987. – 288 с.

6. *Прикладана* теорія цифрових автоматів: Навчальний посібник / В.І.Жабін, І.А.Жуков, І.А.Клименко, В.В.Ткаченко. – К.: Книжкове видавництво НАУ, 2007. – 364 с.

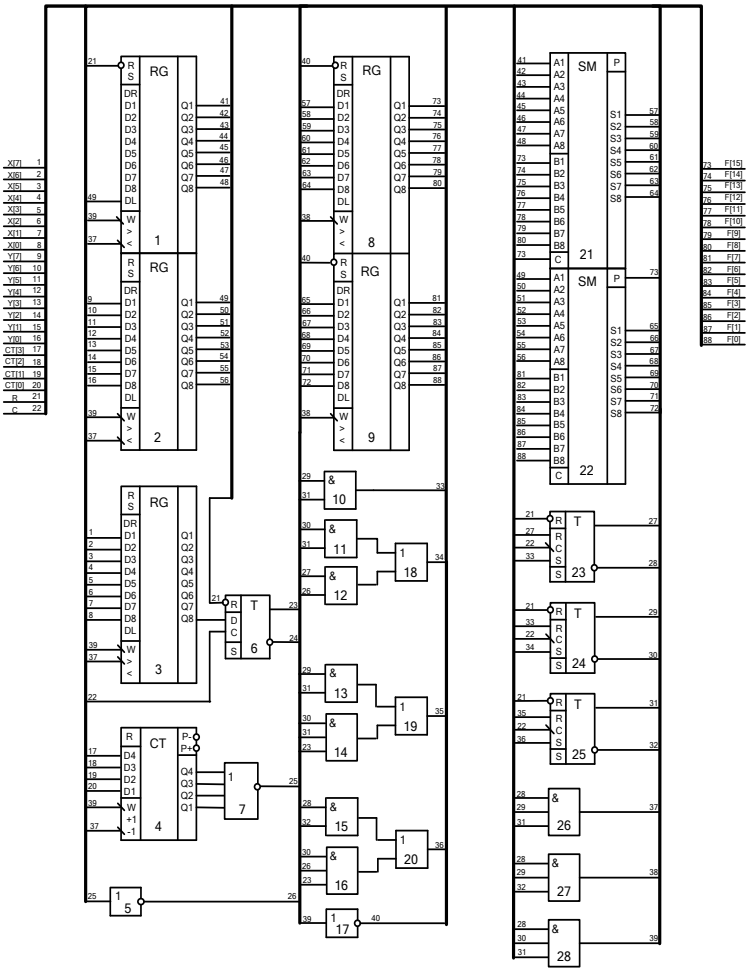
7. *Самофалов К.Г., Корнейчук В.И., Тарасенко В.П.* Цифровые ЭВМ. Теория и проектирование.– К.: Высш.шк. 1989. – 424 с.

8. *Тарабрин В.В., Лунин Л.Ф., Смирнов Ю.Н.* Интегральные микросхемы: Справочник. – М.: Радио и связь, 1990. – 528 с.

9. *Цифровые* ЭВМ. Практикум / К.Г.Самофалов, В.И. Корнейчук, В.П. Тарасенко, В.И.Жабин – К.: Высш.шк. 1989. – 124 с.

ДОДАТОК А

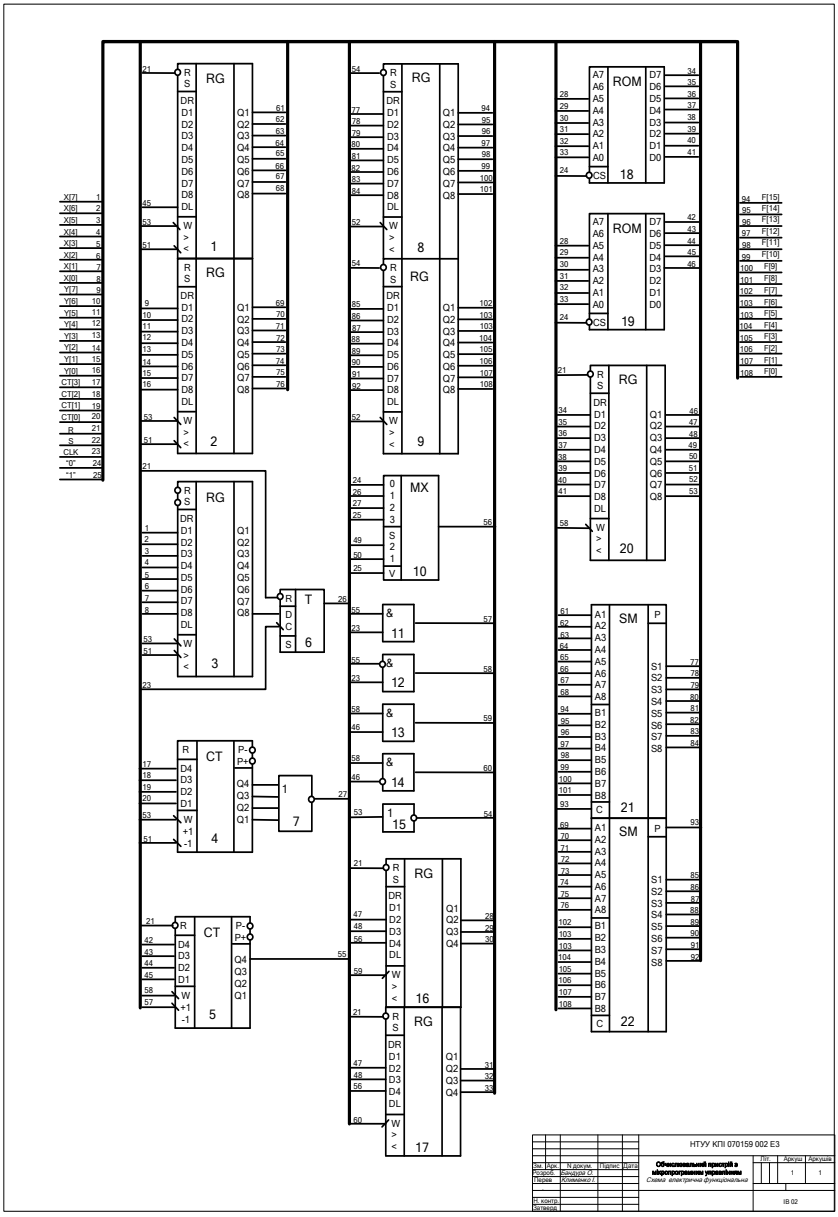
Схема електрична функціональна обчислювального пристрою



						НАУ 070159 001 Е2	
						Обчислювальний пристрій *Схема електрична функціональна*	Літ. / Аркуш / Аркушів: 1 / 1
Зм.	Арк.	N докум.	Підпис	Дата			
Розроб.		*Бандура О.*					
Перев.		*Клименко І. А.*					
Н. контр.							ФКС 402
Затверди							

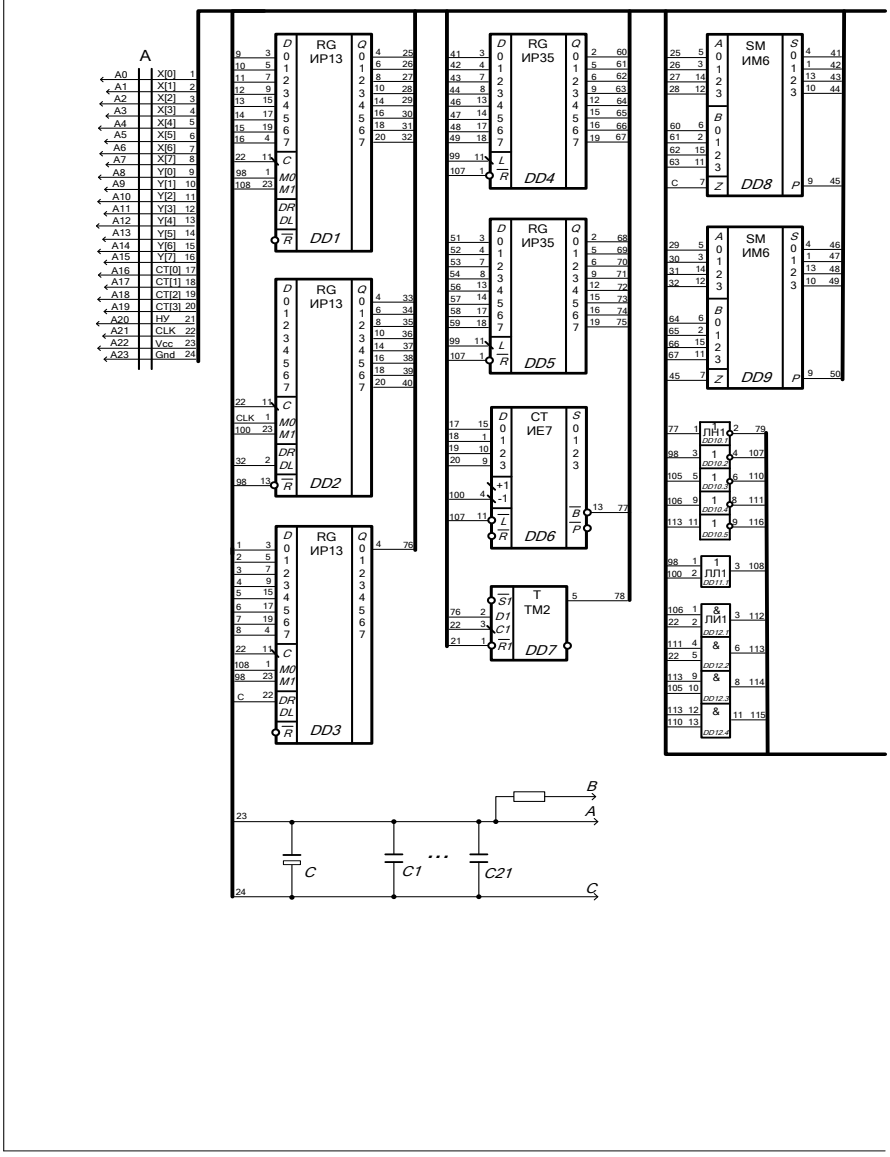
ДОДАТОК Б

Схема електрична функціональна обчислювального пристрою з мікропрограмним управлінням

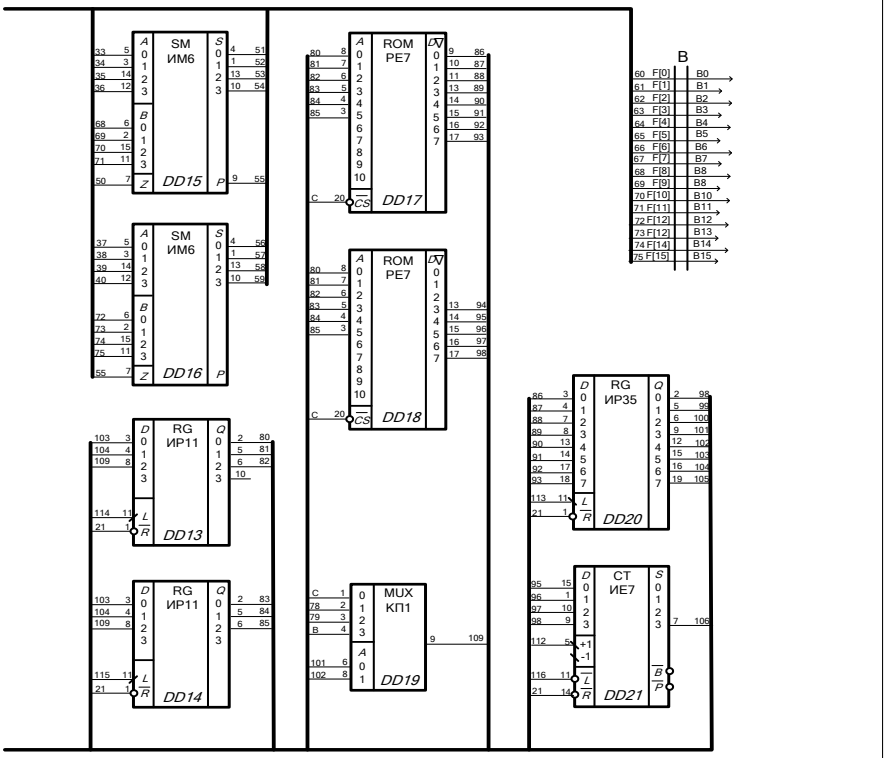


ДОДАТОК В

Схема електрична принципова обчислювального пристрою



з мікропрограмним управлінням



Вхід 24 мікросхем DD1 - DD3, DD17,DD18; вхід 20 мікросхем DD4,DD5,DD20; вхід 14 мікросхем DD7, DD10 - DD14,DD19; вхід 16 мікросхем DD6,DD8,DD9, DD15,DD16,DD21 підключений до ланцюгу A

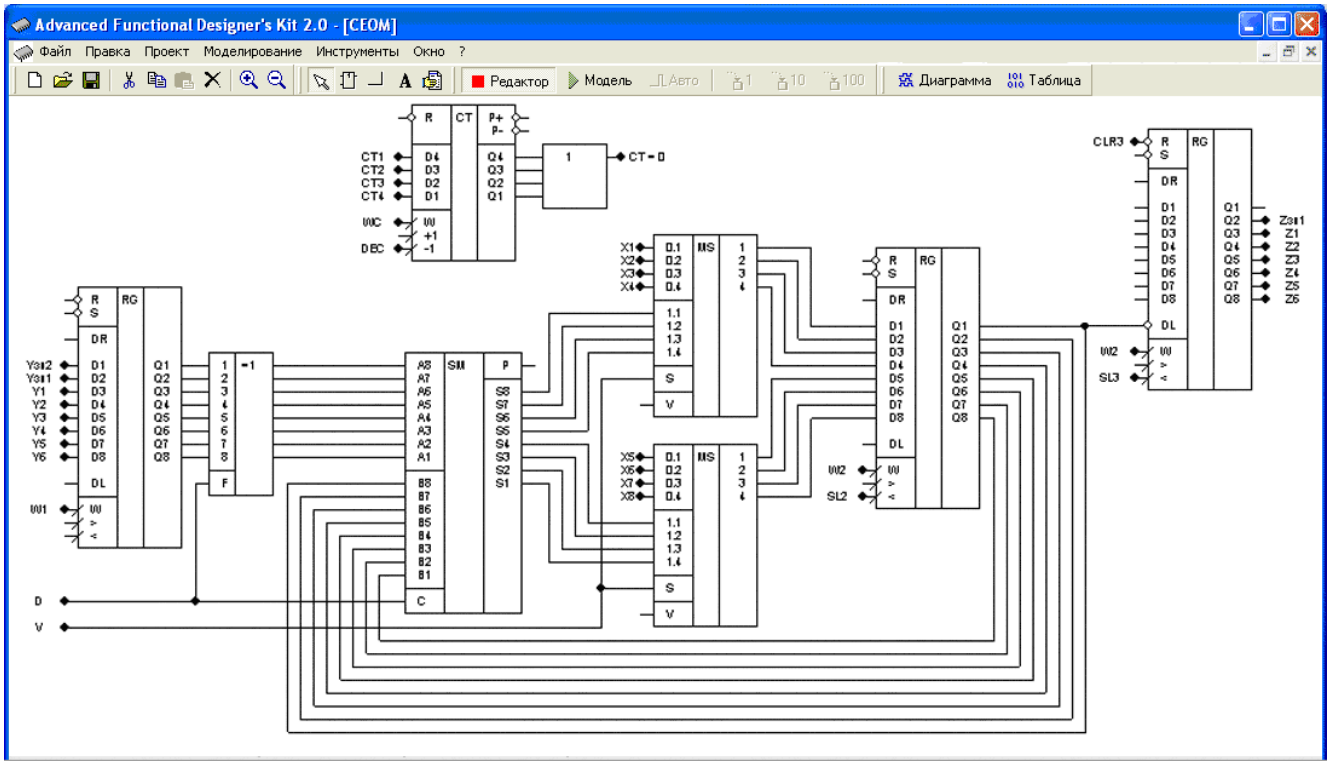
Вхід 12 мікросхем DD1 - DD3, DD17,DD18; вхід 10 мікросхем DD4,DD5,DD20; вхід 7 мікросхем DD7, DD10 - DD14,DD19; вхід 8 мікросхем DD6,DD8, DD9,DD15,DD16,DD21 підключений до ланцюгу C

B - рівень логічної одиниці

					НАУ 070159 003 Е3		
						Літ.	Аркуш	Аркушів
Зм.	Арк.	N докум.	Підпис	Дата	**Обчислювальний пристрій з мікропрограмним управлінням**		1	1
Розроб.		*Бандура О.*			*Схема електрична принципова*			
Перев.		*Клименко І.*						
Н. контр.						ФКС 402		
Затверд.								

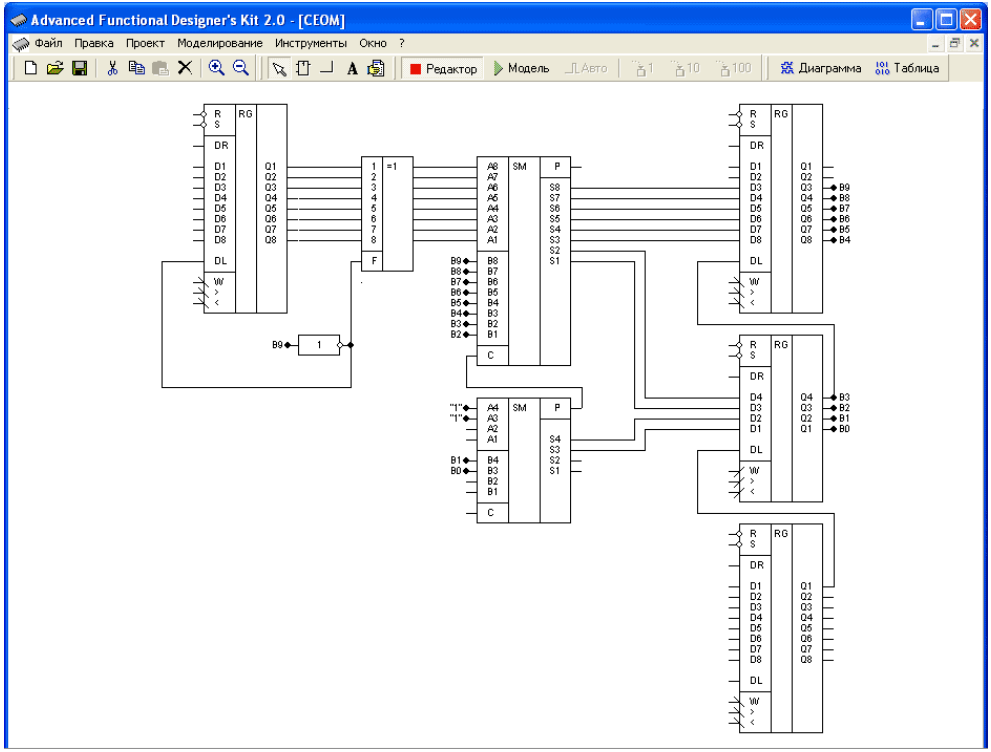
ДОДАТОК Г

Схема операційного пристрою для ділення чисел



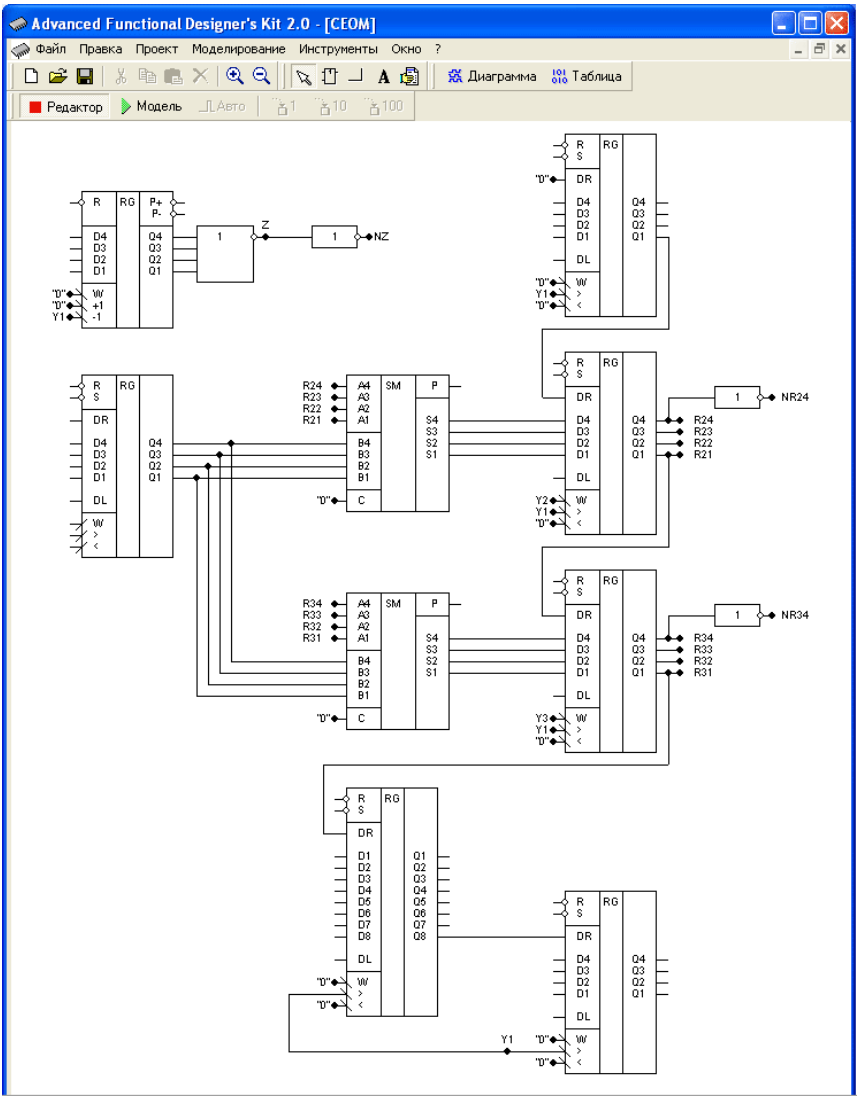
ДОДАТОК Д

Схема операційного пристрою для обчислення функції $A = \sqrt{B}$



ДОДАТОК Е

Схема операційного пристрою для переводу чисел із десяткової системи числення в двійкову методом «зсуву-корекції»

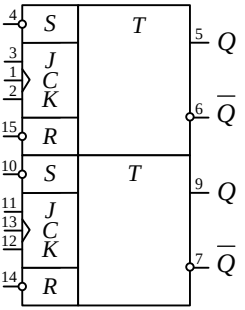


ДОДАТОК Ж

ОСНОВНІ ЕЛЕМЕНТИ ЦИФРОВОЇ ТЕХНІКИ

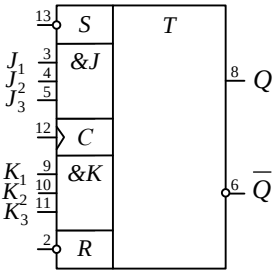
Тригери

555*ТВ*9



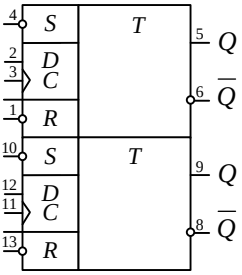
$8 - GND; 16 - V_{CC}$

155*ТВ*1



$7 - GND; 14 - V_{CC}$

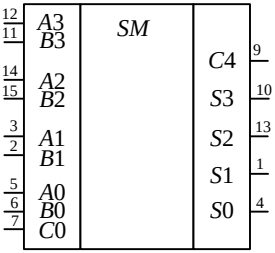
1533*ТМ*2



$7 - GND; 14 - V_{CC}$

Чотирьохрозрядний суматор

555ИМ6



$8 - GND; 16 - V_{CC}$

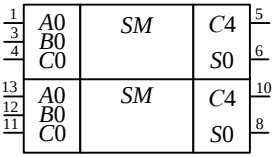
Дворозрядний суматор

155ИМ2

$11 - GND; 4 - V_{CC}; 6 - 9 - NC$

Два однорозрядних суматора

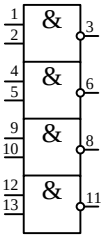
555ИМ5



$7 - GND; 14 - V_{CC}; 2, 9 - NC$

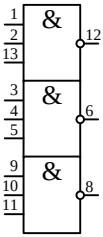
Логічні елементи

1533ЛА3
133ЛА15



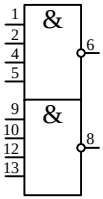
7 – *GND*;
14 – V_{CC}

1533ЛА4



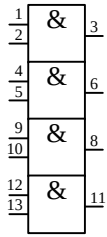
7 – *GND*;
14 – V_{CC}

1533ЛА22
555ЛА6



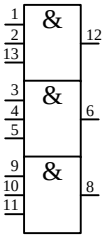
7 – *GND*;
14 – V_{CC}

1533ЛИ1



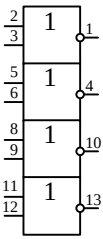
7 – *GND*;
14 – V_{CC}

1531ЛИ3



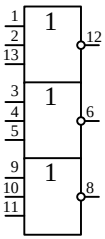
7 – *GND*;
14 – V_{CC}

1533ЛЕ1



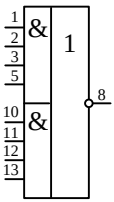
7 – *GND*;
14 – V_{CC}

555ЛЕ4



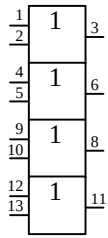
7 – *GND*;
14 – V_{CC}

555ЛР4



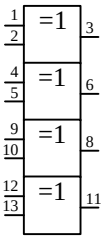
7 – *GND*;
14 – V_{CC}

1533ЛЛ1



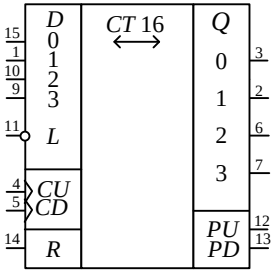
7 – *GND*;
14 – V_{CC}

1533ЛП5



7 – *GND*;
14 – V_{CC}

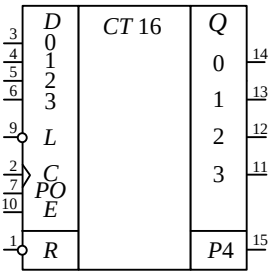
Реверсивний лічильник
555ИЕ7



8 – *GND*; 16 – V_{CC}

$\overline{L} = 1$ – підраховування, $\overline{L} = 1$ – завантаження

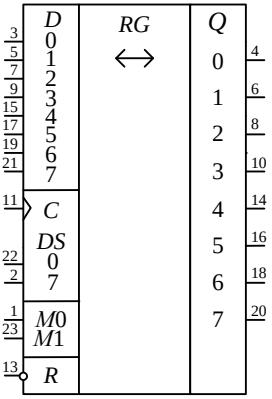
Синхронний лічильник за *mod* 16 з переносом
555ИЕ10, 1561ИЕ21



8 – *GND*; 16 – V_{CC}

Реверсивний зсуваючий регістр

155ИР13

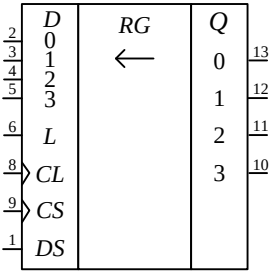


$12 - GND; 24 - V_{CC}$

*M*0=0, *M*1=1 – зсув в сторону старших розрядів; *M*0=1, *M*1=0 – зсув в сторону молодших розрядів; *M*0=0, *M*1=0 – режим зберігання даних.

Зсуваючий регістр

155ИР1

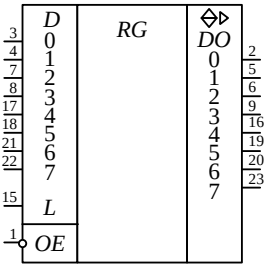


$7 - GND; 14 - V_{CC}$

L=0 – послідовний ввід та зсув даних; *L*=1 – синхронна паралельна загрузка даних; *CS* – тактовий сигнал зсуву даних; *CL* – тактовий сигнал завантаження даних.

Регістр пам'яті

533ИР22

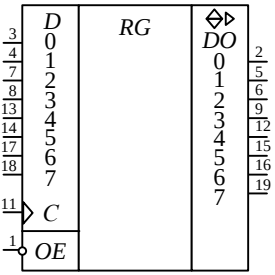


$10 - GND; 24 - V_{CC}$

При $L = 0$ та $OE = 1$ виходи $DO_n = D_n$ – режим прямої передачі значень вхідних сигналів на виходи регістру.

Регістр пам'яті

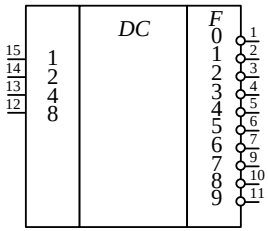
555ИР23



$10 - GND; 20 - V_{CC}$

Дешифратор

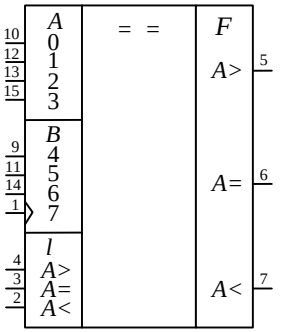
555 ИД6



$8 - GND; 16 - V_{CC}$

Схема порівняння
4-х розрядних двійкових чисел

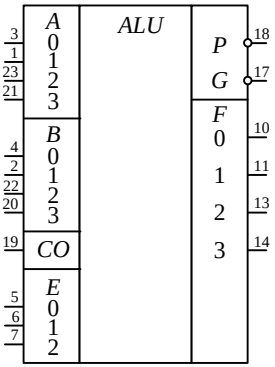
555 СП1



$8 - GND; 16 - V_{CC}$

Чотирьохрозрядний АЛП

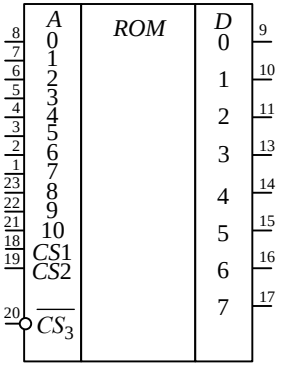
531 НК2



$12 - GND; 24 - V_{CC}$

Пам’ять ПЗП

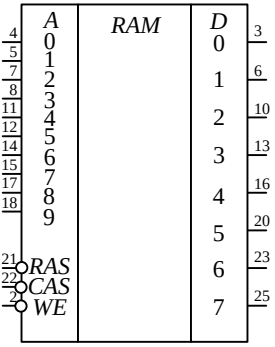
К555 РЕ7



$12 - GND; 24 - V_{CC}$

Пам’ять ОЗП

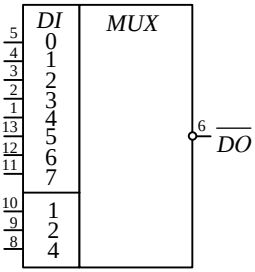
565 РУ7



$30 - GND; 1 - V_{CC}$

Мультиплексор

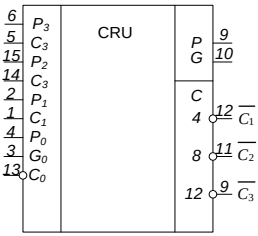
155КП5



$7 - GND; 14 - V_{CC}$

Пристрій переносу
для каскадування суматорів

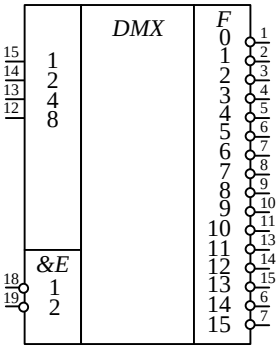
1533ИР4



$8 - GRD;\ 16 - V_{cc}$

Демультиплексор

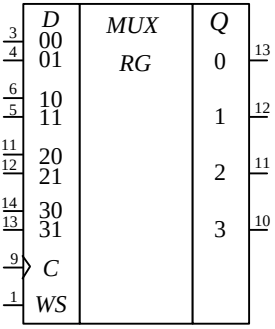
1533 ИД3



$12 - GND; 24 - V_{CC}$

Регістр з мультиплексними
входами

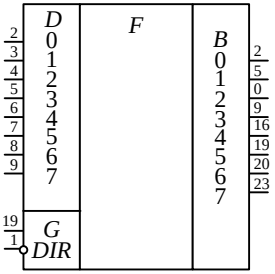
531ИР20



$8 - GND; 16 - V_{CC}$

Шинний формувач

ВА86



$10 - GND; 24 - V_{CC}$

ДОДАТОК З

Зразок оформлення титульного аркуша РГР

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЗВА НАВЧАЛЬНОГО ЗАКЛАДУ

Назва кафедри

РОЗРАХУНКОВО ГРАФІЧНА РОБОТА

з дисципліни

НАЗВА ДИСЦИПЛНИ

Виконав_____

Група _____ Спеціальність_____

Залікова книжка № _____

_____

(допущений до захисту)

_____

(підпис викладача)

_____

(захистив з оцінкою)

2008

ДОДАТОК І

Індивідуальне завдання
до виконання розрахунково-графічної роботи
з дисципліни “НАЗВА ДИСЦИПЛІНИ”

Студента_____
Групи_____

Вихідні дані до розробки	
1 частина	
Операція	
Розрядність операндів	
Значення операндів (X, Y)	
Спосіб адресації	
Структура ПМК	
Тривалість виконання МО підсумовування	
Ємність ПМК	
Призначення зони β_4	
Спосіб мікропрограмування	
Початкова адреса мікропрограми в ПМК	

Завдання видав “_____” _____
Завдання отримав “_____” _____

ДОДАТОК К

Зразок оформлення опису альбому

№ екз.	Формат	Позначення	Найменування	Кількість аркушів	№ примір.	Примітка
1			Документація загальна			
2						
3			Розроблена по новому			
4						
5	А1	НАУ 07 4502 001 ОА	Обчислювальний пристрій	1		
6			з мікропрограмним управлінням			
7			*Опис альбома*			
8						
9	А3	НАУ 07 4502 002 Е1	Обчислювальний пристрій	1		
10			з мікропрограмним управлінням			
11			*Схема електрична*			
12			*структурна*			
13						
14	А3	НАУ 07 4502 003 Е2	Обчислювальний пристрій	1		
15			з мікропрограмним управлінням			
16			*Схема електрична*			
17			*функціональна*			
18						
19	А4	НАУ 07 4502 004 ПЗ	Обчислювальний пристрій	35		
20			з мікропрограмним управлінням			
21			*Пояснювальна записка*			
22						
23						
24						
25						

20 — 8 — 70 — 64 — 8 — 8 — 20 — 5

8 min

Зм	Лит	№ докум.	Підпис	Дата	НТТУ КПІ 07 4502 001 ОА		
Виконав		*Петров П.П.*			**Обчислювальний пристрій з мікропрограмним управлінням** *Опис альбому*	Літера: К П	Лист: 1
Керівник		*Іванов І.І.*					Листів: 1
Перевірив						Група Спеціальність	
Н.контр.							
Зав.каф.		*Іванов І.І.*					

ДОДАТОК Л

Основні написи

Основний напис для перших аркушів креслень і схем

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="4" rowspan="3">НАУ 07 4602 003 Е2</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="2" rowspan="2">Обчислювальний пристрій з мікропрограмним управлінням
Схема електрична функціональна</td><td>Літера</td><td>Маса</td><td>Масштаб</td></tr>
<tr><td>Зм</td><td>Лит</td><td>№ докум.</td><td>Підпис</td><td>Дата</td><td></td><td></td><td></td></tr>
<tr><td>Виконав</td><td></td><td>Іванов І. І.</td><td></td><td></td><td colspan="2" rowspan="3"></td><td colspan="2">Аркуш 1</td><td>Аркушів 1</td></tr>
<tr><td>Керівник</td><td></td><td>Пєтров П. П.</td><td></td><td></td></tr>
<tr><td>Консульт.</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="2" rowspan="3"></td><td colspan="3" rowspan="3">ФКС 106
6.091501</td></tr>
<tr><td>Н. контр.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Зав.каф.</td><td></td><td>Пєтров П.П.</td><td></td><td></td></tr>
</table>

Основний напис для перших аркушів текстових документів

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="4" rowspan="2">НТУУ КПІ 07 4602 004 ПЗ</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Зм</td><td>Лит</td><td>№ докум.</td><td>Підпис</td><td>Дата</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Виконав</td><td></td><td>Іванов І.І.</td><td></td><td></td><td colspan="2" rowspan="4">Обчислювальний пристрій з мікропрограмним управлінням
Пояснювальна записка</td><td>Літера</td><td>Аркуш</td><td>Аркушів</td></tr>
<tr><td>Керівник</td><td></td><td>Пєтров П.П.</td><td></td><td></td><td></td><td>12</td><td>28</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="3" rowspan="2">ФКС 106
6.091501</td></tr>
<tr><td>Н.контр.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Зав.каф.</td><td></td><td>Пєтров П.П.</td><td></td><td></td><td colspan="2"></td><td colspan="3"></td></tr>
</table>

Основний напис для наступних аркушів всіх конструкторських документів

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="2" rowspan="2">НАУ 06 4602 003 ПЗ</td><td>Аркуш</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td>23</td></tr>
<tr><td>Зм</td><td>Лит</td><td>№ докум.</td><td>Підпис</td><td>Дата</td><td colspan="2"></td><td></td></tr>
</table>

ДОДАТОК М

Опис програмного комплексу для моделювання логічних схем

Для проведення лабораторних робіт використовуються програмні засоби моделювання цифрових схем.

Програмний комплекс ПРОГМОЛС 2.0 призначений для моделювання процесів у комбінаційних і послідовних 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 у вигляді файлів на дисках.

Вигляд моделі логічної схеми, побудованої під час застосування програмного комплексу ПРОГМОЛС 2.0 проілюстровано на рис. 1.

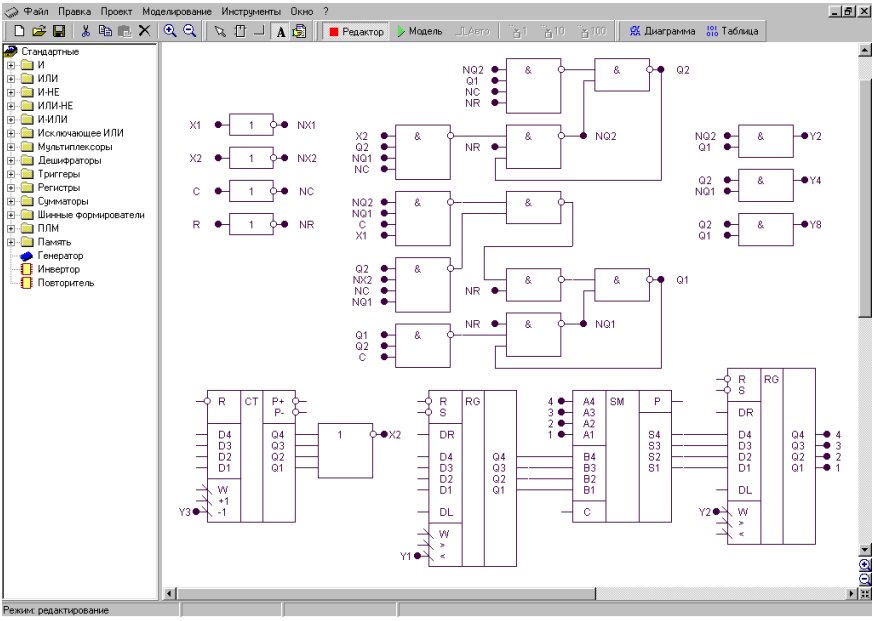


Рис.1. Зображення моделі операційного пристрою

Програмний комплекс включає систему підказок, що полегшує роботу в різних режимах моделювання. Для роботи з програмою використовують систему ієрархічних меню.

Меню містить наступні розділи:

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

Розглянемо детально розділи меню.

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 копію файлу проекту на диск, з вказанням нового ім'я та шляху збереження.

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

Вихід (*Alt+F4*). Закриває програму і всі відкриті 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 корпуса. Викликає діалогове вікно редактора корпуса 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 здійснюється 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 мо-

дельного 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.

На окремі панелі винесені елементи керування

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.

Змінна. Дозволяє визначити змінну в схемі.

Захоплення. Дозволяє переносити 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.

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

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

Панель компонентів

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

У загальному випадку для дослідження логічних схем необхідно виконати наступну послідовність дій:

1. Створити за допомогою редактора логічну схему на екрані дисплея.
2. Позначити на схемі вхідні та вихідні змінні.
3. Створити заголовок (перелічити вхідні і вихідні змінні) і сформувати 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.

Учбове видання

Жабін Валерій Іванович
Жуков Ігор Анатолійович
Клименко Ірина Анатоліївна
Стіренко Сергій Григорович

Арифметичні та управляючі пристрої цифрових ЕОМ

Підп. до друку 14.11.2007. Формат 60×84/16. Папір офсетний №1. Друк офсетний. Гарнітура Times. Ум. др.. арк. 10,95. Наклад 500 прим.

ТОВ «ВЕК+», 05056, м. Київ, пров. Ковальський, 22-а