Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра ЕОМ

АРИФМЕТИЧНІ ТА ЛОГІЧНІ ОСНОВИ КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

Методичні вказівки
до курсової роботи
з дисципліни «Комп'ютерна логіка»
для студентів спеціальності 123
«Комп'ютерна інженерія»

Затверджено
на засідання кафедри
електронних обчислювальних машин.
Протокол № 1 від 29.08.2021 р.

Львів – 2021

Методичні вказівки до курсової роботи «Арифметичні та логічні основи комп’ютерних технологій» з дисципліни «Комп’ютерна логіка» спеціальності 123 «Комп'ютерна інженерія» /Укл. В.С.Глухов, В.А.Голембо. Львів: НУ"ЛП", 2021-97 с.

Укладачі В.С.Глухов, д.т.н.
В.А.Голембо, к.т.н.

Відповідальний за випуск В.С.Глухов, д.т.н.

Рецензенти: Р.Б.Дунець, д.т.н.,
Л.О.Березко, к.т.н.

ЗМІСТ

ВСТУП

У процесі оволодіння студентами бакалаврату «Комп'ютерна інженерія» учбовим матеріалом із дисципліни «Комп'ютерна логіка» (КЛ) важливу роль відіграє виконання курсової роботи. Курсові роботи відносяться до самостіних робіт. Курсову роботу студент повинен виконати самостійно.

В запропонованих методичних указівках курсова робота складається з 29 окремих завдань, об'єднаних у чотири розділи: кодування інформації та перетворення кодів (6 завдань), використання функцій алгебри логіки та їхня мінімізація (4 завдання), синтез комбінаційних схем (15 завдань), виконання арифметико-логічних операцій (9 завдань).

Указівки складаються з теоретичної частини та прикладів розв'язку завдань, так само згрупованих у чотири розділи. Загальна кількість прикладів - 48 (розв'язки завдань першого розділу показані на 11 прикладах, другого - на 8, третього - на 11, четвертого – на 14).

Для поглибленого вивчення теоретичного та практичного матеріалу, у процесі виконання курсової роботи у списку літератури наведено основні навчальні підручники та посібники з КЛ [1 - 5] та суміжних дисциплін [6 - 8] і довідкова література [9 - 11].

Дане видання «Методичних вказівок...» базується на «Методичних вказівках…» [2]. У виданні [2] було змінено оформлення і виправленно помічені помилки.

Матеріали «Методичних указівок...» можуть бути також використані при проведенні практичних занять, для проведення розрахункових, контрольно-графічних і комплексних контрольних робіт, як з курсу КЛ, так і інших дисциплін.

Відгуки та зауваження до «Методичних вказівок...» укладачі просять надсилати відповідальному за випуск професору кафедри електронних обчислювальних машин Інституту комп'ютерних технологій, автоматики та метрології Національного університету «Львівська політехніка» д.т.н. Глухову В.С. glukhov@polynet.lviv.ua.

Завдання на роботу, вказівки щодо вибору варіанта роботи

Метою курсової роботи є закріплення у студентів основних теоретичних положень курсу «Комп'ютерна логіка», набуття практичних навичок побудови цифрових схем та самостійної роботи з учбовою літературою, яку рекомендовано при вивченні курсу.

Робота складається із завдань, які розподілено на чотири частини:

кодування інформації та перетворення кодів;

функції алгебри логіки та їх мінімізація;

синтез комбінаційних схем;

арифметико-логічні операції.

Таблиця TZ.1

Друга (молодша) цифра								Перша (старша) цифра					
							B8	1	2	4	7	9	3
							B7	3	6	2	4	7	5
							B6	2	4	7	5	3	9
							B5	5	3	6	2	4	7
							B4	9	7	5	3	1	2
							B3	7	5	3	6	2	4
							B2	4	9	6	3	5	7
B8	B7	B6	B5	B4	B3	B2	B1	3	5	7	9	1	6
9	3	7	4	5	1	6	2	А	Б	В	Г	Д	Е
6	5	9	6	7	3	2	4	Є	Ж	З	И	І	Ї
3	7	6	5	9	4	1	6	Й	К	Л	М	Н	О
1	6	9	1	6	5	3	7	П	Р	С	Т	У	Ф
3	9	4	3	9	7	5	1	Х	Ц	Ч	Ш	Щ	Ю
7	4	3	5	1	9	7	3	Я	Ь				

Для вибору варіанта роботи необхідно, користуючись заданим викладачем варіантом кодової таблиці (B1...B8 у табл.TZ.1), поставити у відповідність першим 8 різним літерам власного прізвища (при необхідності - ім'я та по-батькові) двозначне шістнадцяткове число.

Таким чином визначити 8 двозначних кодів, які відповідають 8 літерам. Далі в тексті (якщо не буде вказано інше) буде позначатися:

1ц1л - переведений у двійковий код шістнадцятковий код першої цифри (1ц) першої літери (1л);

2ц4л - переведений у двійковий код шістнадцятковий код другої цифри (2ц) четвертої літери (4л) і т.д.

Конкретна кодова таблиця задається викладачем окремо для кожної групи студентів.

1 У процесі виконання першої частини роботи необхідно виконати наведені нижче завдання.

1.1 Скласти шестизначне число, яке складається з отриманих за допомогою кодової таблиці кодів 1-ої, 2-ої та 8-ої літер прізвища. При цьому перші 3 цифри відповідають цілій частині числа, а останні - дробовій.

Наприклад, Біда Ян Андрійович, 8 перших різних літер для варіанту B1: Б, І, Д, А, Я, Н, Р, Й. З кодової таблиці маємо:

Б - 52, І - 14, Й - 36.

Число - 521,436.

Вважаючи це число десятковим, перевести його до шістнадцяткової, з шістнадцяткової до двійкової, з двійкової до вісімкової систем числення з точністю відповідно 3, 5 та 3 розрядів після коми.

1.2 Скласти шестизначне число, яке складається з отриманих за допомогою кодової таблиці кодів 1-ої, 2-ої та 8-ої літер прізвища. При цьому перші 3 цифри відповідають цілій частині числа, а останні - дробовій.

Таблиця TZ.2

abc	f
000	1ц4л
001	
010	
011	
100	2ц7л
101	
110	
111	

Вважаючи це число шістнадцятковим, перевести його до десяткової, з шістнадцяткової до двійкової, з двійкової до вісімкової систем числення з точністю відповідно 3, 5 та 3 розрядів після коми.

1.3 Скласти шестизначне число, яке складається з отриманих за допомогою кодової таблиці кодів 1-ої, 2-ої та 8-ої літер прізвища. Вважаючи це число десятковим, перевести його до системи числення залишкових класів із мінімально необхідною кількістю основ 2, 3, 5, 7, 11, ... Після цього зробити зворотне переведення отриманого результату до десяткової системи числення.

1.4 Виконати ефективне кодування визначених літер прізвища, при умові, що отримане за допомогою кодової таблиці число - десяткове і говорить про те, скільки разів у «повідомленні» зустрічається дана літера (при цьому, алфавіт «повідомлення» складається всього з 8 обраних літер).

Визначити ефективність проведеного кодування та порівняти її з ентропією джерела повідомлення і ефективністю рівномірного кодування, тобто, з випадком, коли довжина коду для кожної літери одна й та сама. За допомогою отриманих кодів скласти повідомлення, яке складається з визначених літер у тій послідовності, в якій вони зустрічаються у прізвищі. Визначити довжину (в бітах) повідомлення при ефективному і рівномірному кодуванні.

1.5 Для шістнадцяти розрядного двійкового коду (1ц1л)(2ц1л)(1ц8л)(2ц8л) сформувати код Геммінга (Hamming) і продемонструвати його реакцію на однократний збій. Результати подати у вигляді таблиці.

1.6 Для послідовності 16-кових цифр (1ц1л)(2ц1л)(1ц2л)(2ц2л)(1ц3л)(2ц3л)...(2ц7л)(1ц8л)(2ц8л), користуючись картами Карно, визначити всі можливі помилкові коди, які можуть виникати при переході від цифри до цифри.

Таблиця TZ.3

Зміщення першого невизначеного значення						-S	S	+S	-S	S
№ набору	a	b	c	d	e	f_0	f_1	f_2	f_3	f_4
0	0	0	0	0	0					
1	0	0	0	0	1	1ц	1ц	1ц	1ц	1ц
2	0	0	0	1	0	1л	1л	1л	5л	5л
3	0	0	0	1	1					
4	0	0	1	0	0					
5	0	0	1	0	1	2ц	2ц	2ц	2ц	2ц
6	0	0	1	1	0	1л	1л	1л	5л	5л
7	0	0	1	1	1					
8	0	1	0	0	0					
9	0	1	0	0	1	1ц	1ц	1ц	1ц	1ц
10	0	1	0	1	0	2л	2л	2л	6л	6л
11	0	1	0	1	1					
12	0	1	1	0	0					
13	0	1	1	0	1	2ц	2ц	2ц	2ц	2ц
14	0	1	1	1	0	2л	2л	2л	6л	6л
15	0	1	1	1	1					
16	1	0	0	0	0					
17	1	0	0	0	1	1ц	1ц	1ц	1ц	1ц
18	1	0	0	1	0	3л	3л	3л	7л	7л
19	1	0	0	1	1					
20	1	0	1	0	0					
21	1	0	1	0	1	2ц	2ц	2ц	2ц	2ц
22	1	0	1	1	0	3л	3л	3л	7л	7л
23	1	0	1	1	1					
24	1	1	0	0	0					
25	1	1	0	0	1	1ц	1ц	1ц	1ц	1ц
26	1	1	0	1	0	4л	4л	4л	8л	8л
27	1	1	0	1	1					
28	1	1	1	0	0					
29	1	1	1	0	1	2ц	2ц	2ц	2ц	2ц
30	1	1	1	1	0	4л	4л	4л	8л	8л
31	1	1	1	1	1					

2 У процесі виконання другої частини роботи необхідно виконати наведені нижче завдання.

2.1 Визначити класи функцій алгебри логіки, до яких належить задана за допомогою таблиці функція трьох змінних (табл. TZ.2), та її функціональну повноту.

Двійкові коди цифр у графі "f" табл. TZ.2 потрібно написати вертикально, старший розряд - наверху.

2.2 Мінімізувати за допомогою методу Квасна-Мак-Класкі-Петрика 5 функцій (f_0, f_1, f_2, f_3, f_4) 5-ти змінних (a, b, c, d, e). Функції задано за допомогою таблиці TZ.3. Побудувати таблицю, яка ілюструє процес знаходження простих імплікант,і таблицю покриття (імплікантну таблицю). За допомогою методу Петрика визначити всі мінімальні розв'язки. Кожний третій набір для кожної з функцій має невизначене значення. Відлік починається від першого згори одиничного значення функції з врахуванням зміщення, величина якого позначена у табл. TZ.3:

S - відлік починається безпосередньо від першого згори одиничного значення;

-S - відлік починається від попереднього набору відносно першого згори одиничного значення;

+S - відлік починається від наступного набору відносно першого згори одиничного значення.

Таблиця TZ.4

Зміщення першого невизначеного значення						-S	S	+S	-S	S
№ набору	a	b	c	d	e	f_0	f_1	f_2	f_3	f_4
0	0	0	0	0	0	(0)X	0	0	0	0
1	0	0	0	0	1	1	(1)X	1	(0)X	0
2	0	0	0	1	0	0	0	(0)X	1	(1)X
3	0	0	0	1	1	(1)X	1	1	1	1
4	0	0	1	0	0	0	X	0	(0)X	0
5	0	0	1	0	1	0	0	(0)X	0	(0)X
6	0	0	1	1	0	(1)X	1	1	1	1
7	0	0	1	1	1	0	X	0	(1)X	1
…										

Наприклад, Біда Ян Андрійович, із кодової таблиці (варіант B1) маємо:

1л) Б - $52_{16} = 0101\ 0010_2$,
2л) І - $14_{16} = 0001\ 0100_2$,
3л) Д - $12_{16} = 0001\ 0010_2$,
4л) А - $32_{16} = 0011\ 0010_2$,
5л) Я - $33_{16} = 0011\ 0011_2$,
6л) Н - $16_{16} = 0001\ 0110_2$,
7л) Р - $58_{16} = 0101\ 1000_2$,
8л) Й - $36_{16} = 0011\ 0110_2$.

Табл. TZ.3 буде мати вигляд табл. TZ.4 (X – невизначене значення, показані лише перші 8 рядків таблиці).

2.3 Мінімізувати за 1 за допомогою карт Карно функції, задані табл. TZ.3. Після мінімізації доповнити функції сполучними термами, підкреслити вирази для цих термів в аналітичному записі функції і позначити їх на картах Карно. Результат мінімізації повинен співпадати з одним із розв'язків, знайдених за допомогою методу Петрика (п. 2.2).

2.4 Мінімізувати за 0 за допомогою карт Карно функції, задані табл. TZ.3. Після мінімізації доповнити функції сполучними термами, підкреслити вирази для цих термів в аналітичному записі функції і позначити їх на картах Карно.

3 У процесі виконання третьої частини роботи необхідно виконати наведені нижче завдання.

3.1 Реалізувати функції, отримані в результаті виконання завдання 2.3, у базисі Буля. На виході кожного елемента написати формулу сигналу, який даним елементом

реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Елементи можуть мати довільну кількість входів. Навести таблиці істинності задіяних елементів. Навести матричну схему.

3.2 Реалізувати функції, отримані в результаті виконання завдання 2.3, у базисі Буля. На виході кожного елемента написати формулу сигналу, який даним елементом реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Усі елементи повинні мати не більше двох входів. Навести таблиці істинності задіяних елементів.

3.3 Реалізувати функції, отримані в результаті виконання завдання 2.3, у монобазисі І-НЕ. На виході кожного елемента І-НЕ написати формулу сигналу, який даним елементом реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Елементи можуть мати довільну кількість входів. Навести таблиці істинності задіяних елементів. Навести матричну схему.

3.4 Реалізувати функції, отримані в результаті виконання завдання 2.3, у монобазисі Шеффера. На виході кожного елемента Шеффера написати формулу сигналу, який даним елементом реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Усі елементи Шеффера повинні бути двовходовими. Навести таблицю істинності елемента Шеффера.

3.5 Реалізувати функції, отримані в результаті виконання завдання 2.4, у монобазисі АБО-НЕ. На виході кожного елемента АБО-НЕ написати формулу сигналу, який даним елементом реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Елементи можуть мати довільну кількість входів. Навести таблиці істинності задіяних елементів. Навести матричну схему.

3.6 Реалізувати функції, отримані в результаті виконання завдання 2.4, у монобазисі Пірса. На виході кожного елемента Пірса написати формулу сигналу, який даним елементом реалізується. Для 5 довільних вхідних наборів визначити рівні сигналів (0 або 1) на виході кожного елемента схеми. Усі елементи Пірса повинні бути двовходовими. Навести таблицю істинності елемента Пірса.

3.7 Функції, мінімізовані в завданні 2.3, реалізувати за допомогою дешифраторів. У кожного з задіяних дешифраторів кількість виходів не повинна перевищувати 16. Навести таблиці істинності, які пояснюють роботу задіяних елементів.

3.8 Функції, мінімізовані в завданні 2.3, реалізувати за допомогою мультиплексорів. У кожного з задіяних мультиплексорів кількість інформаційних входів не повинна перевищувати 16. Навести таблиці істинності, які пояснюють роботу задіяних мультиплексорів. Навести матричну схему.

3.9 Функції, мінімізовані в завданні 2.3, реалізувати за допомогою постійного запам'ятовуючого пристрою (ПЗП). Скласти таблиці прошиття ПЗП. Навести функціональну (матричну) схему запрограмованої ПЛМ.

3.10 Функції, мінімізовані в завданні 2.3, реалізувати за допомогою програмованої логічної матриці (ПЛМ) типу PLA. Скласти таблиці прошиття (програмування) ПЛМ. Навести функціональну (матричну) схему запрограмованої ПЛМ. Навести матричну схему.

3.11 Функції, мінімізовані в завданні 2.4, реалізувати за допомогою програмованої матриці логіки (ПМЛ) типу PAL. Скласти таблиці прошиття (програмування) ПМЛ. Навести функціональну (матричну) схему запрограмованої ПМЛ. Навести матричну схему.

3.12 Для схем, побудованих у завданнях 3.1 - 3.7, визначити їх «ціну», підрахувавши кількість корпусів задіяних елементів. Визначити оптимальний (найдешевший) варіант.

3.13 Для схем, побудованих у завданнях 3.1 - 3.7, визначити їх «ціну», підрахувавши кількість виводів задіяних елементів. Визначити оптимальний (найдешевший) варіант.

3.14 Для схем, побудованих у завданнях 3.1 - 3.7, визначити затримку проходження сигналів від входу до виходу за умови, що всі елементи мають однакову затримку. Визначити оптимальний (найшвидший) варіант.

3.15 На базі ПЛМ типу PLA з кількістю інформаційних входів не більше 16 і з входом вибору кристалу створити дешифратор діапазону адрес, який повинен формувати сигнали «більше», «дорівнює», «менше». Діапазон адрес задається 17-розрядним двійковим кодом, який формується з 17 молодших двійкових розрядів коду (2ц1л)(1ц2л)(2ц7л)(1ц8л)(2ц8л). Отриманий таким чином 17-розрядний двійковий код необхідно ще раз переписати, міняючи місцями старші й молодші двійкові розряди (переписати ззаду наперед). Менший з двох 17-розрядних кодів буде нижньою границею діапазону адрес, більший - верхньою. Сигнал «менше» повинен формуватися, коли на вході схеми присутні адреси, які менші за нижню границю діапазону. Сигнал «більше» повинен формуватись, коли на вході схеми присутні адреси, які більші за верхню границю діапазону. Сигнал «дорівнює» повинен формуватись, коли на вході схеми присутні адреси, які знаходяться посередині діапазону. У кожної з задіяних ПЛМ кількість входів не повинна перевищувати 16. Скласти таблиці прошиття ПЛМ, для кожного рядка таблиці прошиття визначити діапазон адрес, якому цей рядок відповідає. Намалювати числову вісь, на якій позначити:

мінімальне і максимальне значення 17-розрядного коду;

верхню і нижню границі;

діапазони кодів, які обробляються різними ПЛМ.

4 У процесі виконання четвертої частини роботи необхідно виконати наведені нижче завдання.

4.1 Виконати порозрядні операції над двома 16-розрядними

кодами:

(1ц1л)(2ц1л)(1ц2л)(2ц2л) and (1ц7л)(2ц7л)(1ц8л)(2ц8л) – операція І,

(1ц1л)(2ц1л)(1ц2л)(2ц2л) or (1ц7л)(2ц7л)(1ц8л)(2ц8л) – операція АБО,

(1ц1л)(2ц1л)(1ц2л)(2ц2л) xor (1ц7л)(2ц7л)(1ц8л)(2ц8л) – операція ВИКЛЮЧНЕ АБО.

Синтезувати в базисі Буля функціональні схеми пристроїв, які виконують дані операції, і навести значення сигналів на входах схеми і на виходах кожного елемента схеми.

4.2 Виконати операцію віднімання додатніх чисел у двійковому доповняльному коді:

(1ц3л)(1ц1л)(2ц1л)-(1ц8л)(2ц8л),

(1ц8л)(2ц8л)-(1ц3л)(1ц1л)(2ц1л).

Від'ємний результат подати у прямому коді.

Після виконання вказаних операцій навести у шістнадцятковому коді значення операндів і результату.

Синтезувати на базі повних однорозрядних суматорів функціональну схему багаторозрядного суматора, який виконує дані операції, і навести значення сигналів на входах схеми і на виходах кожного однорозрядного суматора.

Синтезувати в базисі Буля функціональну схему повного однорозрядного суматора, навести його таблицю істинності і значення сигналів на входах суматора і на виходах кожного його елемента для кожного розряду згаданого вище багаторозрядного суматора.

4.3 Виконати округлення 16-розрядних двійкових кодів із точністю до 1/2 одиниці молодшого розряду, який залишається. Коди:

1(1ц4л)(2ц4л)(1ц5л)(2ц5л) - від'ємне число в доповняльному коді,

0(1ц4л)(2ц4л)(1ц5л)(2ц5л) - додатне число в доповняльному коді.

При першому округленні відкинути два молодших розряди. Наступні округлення провести послідовно через кожних два двійкових розряди. Результат чергового округлення – це початкові дані для наступного округлення.

4.4 Виконати операцію віднімання чисел у двійково-десятковому коді (числа задані в шістнадцятковому коді):

(1ц1л)(2ц1л)-(1ц8л)(2ц8л),

(1ц8л)(2ц8л)-(1ц1л)(2ц1л).

Від'ємний результат подати у прямому двійково-десятковому коді.

4.5 Виконати операції множення в доповняльному коді двійкових чисел, поданих спочатку в прямому коді:

(+2ц1л) х (+2ц8л),

(-2ц1л) х (+2ц8л),

(+2ц1л) х (-2ц8л),

(-2ц1л) х (-2ц8л).

Попередньо всі числа перевести в доповняльний код. Навести алгоритм множення й таблицю, яка відображає зміни всіх операндів (множеного, множника, лічильника, проміжної суми, окремих розрядів та ознак), які беруть участь у множенні, після виконання кожного з операторів алгоритму.

Синтезувати на базі повних однорозрядних суматорів і з використанням елементів базиса Буля функціональну схему матричного помножувача, який виконує операцію множення додатніх чисел (+2ц1л) х (+2ц8л), і навести значення сигналів на входах схеми і на виходах кожного елемента схеми.

4.6 Виконати операцію множення в доповняльному коді методом Бута двійкових чисел, представлених спочатку в прямому коді:

(+2ц1л) х (+2ц8л),

(-2ц1л) х (+2ц8л),

(+2ц1л) х (-2ц8л),

(-2ц1л) х (-2ц8л).

Попередньо всі числа перевести в доповняльний код. Навести алгоритм множення й таблицю, яка відображає зміни всіх операндів (множеного, множника, лічильника, проміжної суми, окремих розрядів та ознак), які беруть участь у множенні, після виконання кожного з операторів алгоритму.

4.7 Виконати операцію ділення 10-розрядного двійкового коду (10)(1ц2л)(1ц8л) на 5-розрядний двійковий код (1)(1ц1л) методом із відновленням залишків. Навести алгоритм ділення й таблицю, яка відображає зміни всіх операндів (діленого, дільника, лічильника, частки, окремих розрядів та ознак), які беруть участь у множенні, після виконання кожного з операторів алгоритму.

4.8 Виконати операцію ділення 10-розрядного двійкового коду (10)(1ц2л)(1ц8л) на 5-розрядний двійковий код (1)(1ц1л) методом без відновлення залишків. Навести алгоритм ділення й таблицю, яка відображає зміни всіх операндів (діленого, дільника, лічильника, частки, окремих розрядів та ознак), які беруть участь у множенні, після виконання кожного з операторів алгоритму.

4.9 Виконати операції додавання A+B, віднімання A-B, множення AB і ділення A/B над числами, представленими у форматі з рухомою комою. Число A складається з кодів 2ц1л і 2ц8л, число B складається з кодів 2ц2л і 2ц7л. Формат операндів і результатів повинен задовільняти вимогам табл. TZ5.

Курсова робота оформляється у вигляді розрахунково-пояснювальної записки (обсягом 30-50 сторінок), до якої додаються функціональні схеми, виконані згідно з вимогами ЄСКД. Креслення в курсовій роботі не передбачаються.

Пояснювальна записка повинна починатися титульним аркушем. У записці повинен знаходитися Зміст з вказівкою назв і номерів розділів і задач, а також сторінок записки, з яких починається розв'язок кожної задачі і кожний розділ. Усі сторінки записки повинні бути пронумеровані. Перша сторінка (титульний аркуш) – не нумерується.

Пояснювальна записка повинна розкривати всі дії, які виконувалися студентом, по виконанню кожного з пунктів роботи. Перед розв'язком кожної задачі повинна наводитися

Таблиця TZ5

Номер розряду	Призначення	Примітки
7	знак числа	0 – число додатнє; 1 – число від'ємне
6..4	порядок, зміщений на 4	ознака рівності числа нулю – усі розряди порядка дорівнюють 0
3..0	мантиса числа	мантиса 5-розрядна, нормалізована, старший розряд мантиси завжди дорівнює 1 (1,0 – одна ціла, 0 десятих) і у форматі не відображається

її умова. Текстова інформація повинна бути зведена до мінімуму. У кінці записки повинен знаходитися список літератури, яка використовувалася.

Захист курсової роботи є формою перевірки якості її виконання й знань студента в галузі прикладної теорії цифрових автоматів. Захист роботи складається з короткої (10-15 хв.) доповіді студента по виконаній роботі і з письмової відповіді на запитання. У результаті захисту курсової роботи студенти отримують оцінку за чотирибальною системою (незадовільно, задовільно, добре, відмінно). Оцінки виставляються після виправлення студентом усіх помилок, які були виявлені викладачем у роботі під час її перевірки, а також після усунення його зауважень.

Студент, який не здав у встановлений термін курсової роботи і не захистив її без поважної причини, вважається таким, який має академічну заборгованість.

Для рівномірного навантаження студента під час виконання курсової роботи та для своєчасного її подання до захисту рекомендується дотримуватися послідовності виконання роботи згідно з табл. TZ.6. Таблиця TZ.6 може мінятися в залежності від програми курсу й уточнюється викладачем під час видачі завдання на курсову роботу.

Допускається захист роботи частинами (завданнями) протягом семестру.

Таблиця TZ.6

№	Характер роботи	Термін виконання (тижні)	
		початок	закінчення
1	Аналіз технічного завдання	1	2
2	Виконання завдань частини 1	2	4
3	Виконання завдань частини 2	5	8
4	Виконання завдань частини 3	9	11
5	Виконання завдань частини 4	12	14
6	Захист роботи	15	15

1 Методичні вказівки щодо кодування інформації та перетворення кодів

1.1 Переведення чисел до десяткової системи числення з іншої однорідної позиційної системи числення з основою k, коли дії виконуються в десятковій системі

В однорідних позиційних системах числення при безпосередньому представленні цифр число записується у вигляді

$X = x_s x_{s-1} ... x_1 x_0, x_{-1} ... x_{-m}$.

Кількісний еквівалент, який виражається цим записом, визначається так:

$$X = k^s x_s + k^{s-1} x_{s-1} + ... + k^1 x_1 + k^0 x_0 + k^{-1} x_{-1} + ... + k^{-m} x_{-m}, \quad (1.1.1)$$

де k - основа системи числення,

s+1 - розрядність цілої частини числа,

m - розрядність дробової частини числа,

x_i - цифри i-го розряду запису числа ($x_i = 0, 1, ..., k-1$),

k^i - вага i-го розряду. У даному випадку вага i-го розряду в k разів більша за вагу (i-1)-го розряду. Такі системи числення називають системами з природним порядком ваги, і до них належать двійкова, вісімкова, десяткова і шістнадцяткова системи числення.

Цифри цих систем та їхні еквіваленти наведено в табл. 1.1.1.

Таблиця 1.1.1

Основа системи числення			
10	2	8	16
0	0	0	0
1	1	1	1
2	(10)	2	2
3	(11)	3	3
4	(100)	4	4
5	(101)	5	5
6	(110)	6	6
7	(111)	7	7
8	(1000)	(10)	8
9	(1001)	(11)	9
(10)	(1010)	(12)	A
(11)	(1011)	(13)	B
(12)	(1100)	(14)	C
(13)	(1101)	(15)	D
(14)	(1110)	(16)	E
(15)	(1111)	(17)	F

Примітка: в дужках указані величини десяткових, двійкових та вісімкових еквівалентів.

На використанні формули (1.1.1) заснований один із методів переведення чисел із системи числення з основою k до десяткової системи. Для переведення необхідно записати число у формі (1.1.1), замінити цифри k-тої системи числення та основу k їхніми десятковими еквівалентами, а потім обчислити вираз за правилами десяткової арифметики.

Приклад 1.1.1. Перевести число X_2 = 1011,1001 до десяткової системи числення.

$1011{,}1001_2 =$

$= 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4} =$

$= 8 + 0 + 2 + 1 + 0{,}5 + 0 + 0 + 0{,}0625 = = 11{,}5625;$

$X_{10} = 11{,}5625.$

Приклад 1.1.2. Перевести вісімкове число 105,71 до десяткової системи числення.

$105{,}71_8 = 1 \cdot 8^2 + 0 \cdot 8^1 + 5 \cdot 8^0 + 7 \cdot 8^{-1} + 1 \cdot 8^{-2} =$

$= 64 + 0 + 5 + 0{,}875 + 0{,}015625 = 69{,}890625;$

$X_{10} = 69{,}890625.$

Приклад 1.1.3. Перевести шістнадцяткове число 2ED,0A до десяткової системи числення.

$2ED{,}0A_{16} = 2 \cdot 16^2 + 14 \cdot 16^1 + 13 \cdot 16^0 + 0 \cdot 16^{-1} + 10 \cdot 16^{-2} = 512 + 224 + 13 + 0 + 0{,}0390625 =$

$= 849{,}0390625;$

$X_{10} = 849{,}0390625.$

1.2 Переведення чисел із десяткової системи числення до іншої однорідної позиційної системи числення з основою k, коли дії виконуються в десятковій системі

При даному переведенні окремо виконується переведення цілої частини числа й окремо - дробової, результати потім додаються.

1.2.1. Переведення цілої частини числа

Цілу частину числа X_{10} ділять на основу системи числення k за правилами десяткової арифметики до отримання залишку, який буде десятковим еквівалентом молодшої цифри результату. Якщо частка від ділення не дорівнює 0, то вона стає діленим і процес ділення на k продовжується. Як тільки чергова частка стане рівною 0, процес ділення на k припиняється. Залишок, який отримали при першому діленні на k, є цифрою розряду результату з вагою k^0, залишок при другому діленні - цифрою з вагою k^1 і т.д. Останній залишок є старшою цифрою результату.

1.2.2. Переведення дробової частини числа

Дробова частина числа X_{10} помножується на k за правилами десяткової арифметики. В отриманому добутку від'єднується ціла частина, яка може дорівнювати 0, а дробова частина знову помножується на k із наступним від'єднанням цілої частини. Ця операція повторюється або до отримання нульової дробової частини добутку, або до отримання необхідної кількості розрядів числа X_k. Старша цифра результату переведення (тобто, перша після коми) збігається з першою від'єднаною цілою частиною, друга цифра результату - із другою від'єднаною цілою частиною і т.д. При цьому від’єднані цілі частини необхідно представити в системі числення з основою k.

Приклад 1.2.1. Перевести десяткове число 11,5625 до двійкової системи числення з точністю п’ять розрядів після коми.

Переведення цілої частини:

11 : 2 = 5, залишок 1 (молодший розряд результату),

5 : 2 = 2, залишок 1,

2 : 2 = 1, залишок 0,

1 : 2 = 0, залишок 1 (старший розряд результату).

Результат $X_{2ц}$ = 1011.

Таблиця 1.2.1

Крок	Дріб	Результат множення на k=2	Ціла частина результату множення, яка вилучається	Вага двійкового розряду
1	0,5625	1,125	1	Старший (перший після коми)
2	0,125	0,25	0	
3	0,25	0,5	0	
4	0,5	1,0	1	
5	0,0	0,0	0	Молодший

Хід переведення дробової частини наведений у табл. 1.2.1.

Результат $X_{2д}$ =0,10010.

Повний результат $X_2= X_{2ц}+X_{2д}$ = 1011 + 0,10010 = = 1011,10010.

1.3 Переведення чисел з шістнадцяткової й вісімкової систем до двійкової і зворотне переведення чисел

Для переведення чисел з 16-кової і 8-кової систем числення до двійкової необхідно кожну цифру числа, яке переводиться, замінити відповідно чотири- або трирозрядним двійковим еквівалентом (табл. 1.1.1) - тетрадою або тріадою, а отримані двійкові цифри розташувати на місцях 16-кових або 8-кових цифр.

При необхідності переведення чисел із 10-кової системи числення до 8-кової, 16-кової та двійкової переведення робиться тільки до однієї системи (8-кової або 16-кової), подальше переведення виконується через двійкову систему з використання тріад і тетрад.

Приклад 1.3.1. Перевести число 12345,67 з десяткової системи числення до двійкової, 8-кової, 16-кової.

1) переведення цілої частини числа до 8-кової системи:

12345 : 8 = 1543, залишок 1;

1543 : 8 = 192, залишок 7;

192 : 8 = 24, залишок 0;

24 : 8 = 3, залишок 0;

3 : 8 = 0, залишок 3;

результат 30071_8.

2) переведення дробової частини числа до 8-кової системи:

0,67 x 8 = 5,36;

0,36 x 8 = 2,88;

0,88 x 8 = 7,04;

0,04 x 8 = 0,32;

наближений результат 0,5270... .

Повний результат 30071,5270... .

3) переведення до двійкової та 16-кової систем числення наведене у табл. 1.3.1:

Таблиця 1.3.1

3			0			0			7			1			,	5			2			7			0			8-кові цифри
0	1	1	0	0	0	0	0	0	1	1	1	0	0	1	,	1	0	1	0	1	0	1	1	1	0	0	0	2-кові цифри
3			0			3			9						,	A			B			8						16-кові цифри

Розбиття двійкового числа на тріади і тетради починається від коми ліворуч і праворуч.

Результат $12345{,}67_{10}=30071{,}5270_8=11000000111001{,}101010111_2=3039{,}AB8_{16}$.

1.4 Ефективне кодування

1.4.1. Алгоритм ефективного кодування Шеннона – Фано

Алгоритм ефективного кодування Шеннона - Фано формує двійкові коди літер алфавіту за принципом «чим частіше зустрічається літера - тим коротшим є її код». Послідовність виконання ефективного кодування така:

1) літери алфавіту виписуються до таблиці в порядку зменшення ймовірності їхньої появи в тексті;

2) таблиця ділиться на дві групи так, щоб суми ймовірностей у кожній з груп були приблизно однаковими;

3) усім літерам верхньої групи як перший символ коду надається значення 1, а нижньої – 0;

4) для визначення наступних символів дії двох попередніх пунктів (пп. 2, 3) повторюються для кожної з груп, доти, поки в групах не залишиться по одній літері.

Якщо групи не вдається сформувати з приблизно рівними ймовірностями, то вводяться нові символи, які є комбінаціями літер (табл. 1.4.1).

1.4.2. Ентропія.

В інформатиці ентропія характеризує степінь нестачі інформації про значення фізичної величини.

Таблиця 1.4.1

Літера	Імовірність	Нова літера	Комбінація	Імовірність
a_0	0,8	b_0	$a_0 \cdot a_0$	0,64
		b_1	$a_0 \cdot a_1$	0,16
a_1	0,2	b_2	$a_1 \cdot a_0$	0,16
		b_4	$a_1 \cdot a_1$	0,04

Припустимо, що ми провели N дослідів, отримали k різних результатів, і-ий результат (i = 1,2,...,k) повторюється n_i разів і вносить кількість інформації I_i.

Середня кількість інформації, яка надається одним дослідом, дорівнює $I_{сер}=(n_1I_1+ n_2I_2+...+n_kI_k)/N$,

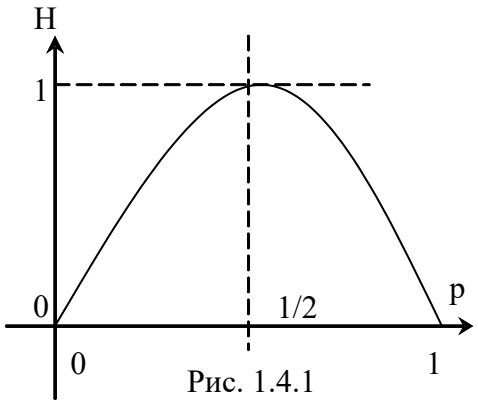


Рис. 1.4.1

$I_i= \log_2(1/p_i) = -\log_2 p_i$ (чим менша ймовірність події, тим більше інформації ми здобуваємо, коли подія відбувається).

$\log_2 x$ - позначення двійкового логарифму числа *x*, двійковий логарифм обраховується за допомогою десяткового або натурального логарифму

$\log_2 x = \log_{10} x/\log_{10} 2 = \lg x/\lg 2 = \ln x/\ln 2$;

$n_i/N = p_i$; $I_{сер}= -p_1\log_2 p_1 - p_2\log_2 p_2 - ... - p_i\log_2 p_i =$

$$= -\sum_{i=1}^{k} p_i \log_2 p_i = I_{сер} = H$$

(за Шенноном ентропія H дорівнює середній кількості інформації, вимірюється в бітах).

Таблиця 1.4.2

Літера	Імовірність	Ефективний код		Неефективний код	
a_i	p_i	код	Довжина	код	Довжина
a_0	1/2	1	1	000	3
a_1	1/4	01	2	001	3
a_2	1/8	001	3	010	3
a_3	1/16	0001	4	011	3
a_4	1/32	00001	5	100	3
a_5	1/64	000001	6	101	3
a_6	1/128	0000001	7	110	3
a_7	1/128	0000000	7	111	3

Властивості ентропії:

1) невід'ємна;

2) дорівнює 0, коли ймовірність однієї події = 1, а решти - 0;

3) максимальна, коли всі ймовірності однакові:

$p_1 = p_2 = \ldots = p_k = p = 1/k$,

$H_{max} = -kp\log_2 p = -\log_2 p$.

Залежність ентропії однократної події від її ймовірності наведено на рис. 1.4.1.

Приклад 1.4.1. Для табл. Шеннон довів: повідомлення, які складаються з літер певного алфавіту, можна закодувати так, що середнє число двійкових символів на літеру буде як завгодно близьке до ентропії джерела цих повідомлень, але не менше цієї величини. Таке кодування називається ефективним.

1.4.2 ентропія дорівнює

$$H = -\sum_{i=0}^{7} p_i \log_2 p_i = 1\cdot 1/2 + 2\cdot 1/4 + + 3\cdot 1/8 + 4\cdot 1/16 + 7\cdot 1/128 = = 127/64.$$

Середня довжина коду літери при ефективному кодуванні дорівнює

$$L_{сер.еф.} = \sum_{i=0}^{7} p_i n_i = 1\cdot 1/2 + 2\cdot 1/4 + ...$$

$$\ldots + 7\cdot 1/128 = 127/64.$$

Таблиця 1.4.3

Літера	Імовірність	Нефективний код		Ефективний код	
a_i	1/8	код	Довжина	код	Довжина
a_0	1/8	1	1	000	3
a_1	1/8	01	2	001	3
a_2	1/8	001	3	010	3
a_3	1/8	0001	4	011	3
a_4	1/8	00001	5	100	3
a_5	1/8	000001	6	101	3
a_6	1/8	0000001	7	110	3
a_7	1/8	0000000	7	111	3

$L_{сер.еф.} = H$.

Середня довжина коду літери при неефективному кодуванні дорівнює

$$L_{сер.нееф.} = \sum_{i=0}^{7} p_i n_i = 1/2 \cdot 3 + + 1/4 \cdot 3 + ... + 1/128 \cdot 3 = 3 > H.$$

Приклад 1.4.2. Для табл. 1.4.3 ентропія дорівнює

$$H = -\sum_{i=0}^{7} p_i \log_2 p_i = 8 \cdot 3 \cdot 1/8 = 3.$$

Середня довжина коду літери при ефективному кодуванні дорівнює

$$L_{сер.еф.} = \sum_{i=0}^{7} p_i n_i = 8 \cdot 3 \cdot 1/8 = 3 = H.$$

Таблиця 1.4.4

Літера	Кількість	Імовірність
Б	11	11/249=0.04
І	72	72/249=0.29
Д	71	11/249=0.28
А	01	01/249=0.01
Я	00	00/249=0.00
Н	74	74/249=0.30
Р	16	16/249=0.06
Й	04	11/249=0.02
Усього	249	249/249=1.00

Середня довжина коду літери при неефективному кодуванні дорівнює

$$L_{сер.нееф.} = \sum_{i=0}^{7} p_i n_i = 1/8 \cdot 1 + 1/8 \cdot 2 + ... + 1/8 \cdot 7 = 35/8 > H.$$

Приклад 1.4.3. Виконати ефективне кодування літер, для яких табл. 1.4.4 визначає, скільки разів вони зустрічаються в «тексті».

Процес кодування відображено у табл. 1.4.5.

Таблиця 1.4.5

i	Літера	Імовір-ність	Ефектив-ний код		Неефектив-ний код		Крок
	a_i	p_i		n_i		n_i	
0	Н	0.30	11	2	000	3	
1	І	0.29	10	2	001	3	<-2
2	Д	0.28	01	2	010	3	<-1
3	Р	0.06	001	3	011	3	<-3
4	Б	0.04	0001	4	100	3	<-4
5	Й	0.02	00001	5	101	3	<-5
6	А	0.01	00000	5	110	3	<-6

Послідовність поділу таблиці на групи літер показано в графі «Крок».

Ентропія:

$$H = -\sum_{i=0}^{6} p_i \log_2 p_i =$$

$$= 0{,}3\log_2 0{,}3 + 0{,}29\log_2 0{,}29 + 0{,}28\log_2 0{,}28 + + 0{,}06\log_2 0{,}06 + 0{,}04\log_2 0{,}04 + 0{,}02\log_2 0{,}02 + + 0{,}01\log_2 0{,}01 = 2{,}1.$$

Середня довжина коду літери при ефективному кодуванні дорівнює

$$L_{сер.еф.} = \sum_{i=0}^{6} p_i n_i = 0{,}30 \cdot 2 + 0{,}29 \cdot 2 + 0{,}28 \cdot 2 + 0{,}06 \cdot 3 + + 0{,}04 \cdot 4 + 0{,}02 \cdot 5 + 0{,}01 \cdot 6 + 0{,}00 \cdot 6 = 2{,}14 > H.$$

Середня довжина коду літери при неефективному кодуванні:

$$L_{сер.нееф.} = \sum_{i=0}^{6} p_i n_i = 3 \cdot \sum_{i=0}^{6} p_i = 3 > L_{сер.еф.} > H.$$

1.5 Система залишкових класів

У системі числення, яка називається системою залишкових класів (СЗК), будь- яке число X ($X_{min} \le X < X_{min} + P$, де $P = p_1 \cdot p_2 \cdot ... \cdot p_n$) можна зобразити однозначно як сукупність його залишків ($q_1, q_2, ..., q_n$), які виникають при діленні цього числа на вибрані основи. При цьому основи - це група взаємно простих чисел $p_1, p_2, ..., p_n$;

$0 \le q_i < p_i$; i = 1, 2, ..., n.

Таблиця 1.5.1

Число	Залишки від ділення на			Код у СЗК
	2	3	5	
0	0	0	0	0,0,0
1	1	1	1	1,1,1
2	0	2	2	0,2,2
3	1	0	3	1,0,3
4	0	1	4	0,1,4
5	1	2	0	1,2,0
6	0	0	1	0,0,1
7	1	1	2	1,1,2
8	0	2	3	0,2,3
9	1	0	4	1,0,4
10	0	1	0	0,1,0
11	1	2	1	1,2,1
12	0	0	2	0,0,2
13	1	1	3	1,1,3
14	0	2	4	0,2,4
15	1	0	0	1,0,0
16	0	1	1	0,1,1
17	1	2	2	1,2,2
18	0	0	3	0,0,3
19	1	1	4	1,1,4
20	0	2	0	0,2,0
21	1	0	1	1,0,1
22	0	1	2	0,1,2
23	1	2	3	1,2,3
24	0	0	4	0,0,4
25	1	1	0	1,1,0
26	0	2	1	0,2,1
27	1	0	2	1,0,2
28	0	1	3	0,1,3
29	1	2	4	1,2,4
30	0	0	0	0,0,0

Основна перевага СЗК - незалежність утворення розрядів числа, внаслідок чого кожний розряд містить у собі інформацію про все число X. Це створює можливість незалежної обробки розрядів, тобто, порозрядного виконання операцій.

До недоліків СЗК, які значно ускладнюють її практичне використання в обчислювальній техніці, можна віднести:

відсутність достатньо простих ознак виходу числа за межі діапазону $[X_{min}, P+X_{min}]$;

отримання завжди точного результату операції,

внаслідок чого виключається можливість безпосереднього округлення результату і взагалі наближеного виконання операцій;

неможливість прямого ділення двох довільних чисел;

неможливість візуального порівняння двох чисел.

Переведення чисел із позиційної десяткової системи числення у СЗК.

Переведення чисел із позиційної десяткової системи числення у СЗК виконується послідовним діленням числа на вибрані основи до отримання залишків. Набір цих залишків - це і є число в СЗК.

Приклад 1.5.1. Перевести в СЗК з основами (базисом) $p_1=2$, $p_2=3$, $p_3=5$ перші n чисел (n=3) натурального ряду.

Спочатку визначимо $P = p_1 \cdot p_2 \cdot p_3 = 2 \cdot 3 \cdot 5 = 30$.

Тобто, у вибраній системі можливо зобразити без повторень лише 30 різних чисел (у даному прикладі від 0 до 29, табл. 1.5.1).

З наведеного прикладу видно, що для обраного базису (2,3,5) у СЗК число 30 має такий самий код, як і число 0. Тобто, однозначне переведення можливе тільки в межах визначеного діапазону P.

Переведення чисел з СЗК до позиційної десяткової системи числення.

Переведення числа $(q_1,q_2,q_3,...q_n)$ з СЗК з базисом ($p_1, p_2, p_3, ...$ …, p_n) до позиційної десяткової системи числення виконується за формулою

$$A=(q_1B_1+q_2B_2+q_3B_3+q_nB_n)(\text{mod } P)$$

де $B_1 = (1,0,0,...,0)$,

$B_2 = (0,1,0,...,0)$,

$B_3 = (0,0,1,...,0)$,

... ,

$B_n = (0,0,0,...,1)$ - ортогональні базиси – вага n-го розряду числа, яку записано у СЗК. Вага B_n обчислюється за формулою

$B_n = m_n \cdot P/p_n$,

де m_n - ціле число, вага ортогонального базису, обирається з умови $B_n/p_n = 1(\text{mod } p_n)$.

Ортогональні базиси подаються числами десяткової системи числення.

Приклад 1.5.2. Перевести число (1,2,3,2) з СЗК із базисом (2,3,5,7) до позиційної десяткової системи числення.

Знаходимо $P = 2 \cdot 3 \cdot 5 \cdot 7 = 210$.

Знаходимо ортогональні базиси для різних m_n і починаємо з $m_n = 1$:

$B_1 = 1 \cdot n/2 = 105$; перевірка умови ортогональності: 105/2 = 52[1], залишок 1 - умова виконується, отже, $B_1 = 105$.

$B_2 = 1 \cdot n/3 = 70$; перевірка умови ортогональності: 70/3 = 23[1], залишок 1 - умова виконується, отже, $B_2 = 70$.

$B_3 = 1 \cdot n/5 = 42$; перевірка умови ортогональності: 42/5 = 8[2], залишок 2 - умова не виконується, пробуємо наступну вагу $m_3 = 2$:

$B_3 = 2 \cdot n/5 = 84$; перевірка умови ортогональності: 84/5 = 8[4], залишок 4 - умова не виконується, пробуємо наступну вагу $m_3 = 3$:

$B_3 = 3 \cdot n/5 = 126$; перевірка умови ортогональності: 126/5 = 25[1], залишок 1 - умова виконується, отже, $B_3 = 126$.

$B_4 = 1 \cdot n/7 = 30$; перевірка умови ортогональності: 30/7 = 4[2], залишок 2 - умова не виконується, пробуємо наступну вагу $m_4 = 2$:

$B_4 = 2 \cdot n/7 = 60$; перевірка умови ортогональності 60/7 = 8[4], залишок 4 - умова не виконується, пробуємо наступну вагу $m_4 = 3$:

$B_4 = 3 \cdot n/7 = 90$; перевірка умови ортогональності 90/7 = 12[6], залишок 6 - умова не виконується, пробуємо наступну вагу $m_4 = 4$:

$B_4 = 4 \cdot n/7 = 120$; перевірка умови ортогональності 120/7 = 17[1], залишок 1 - умова виконується, отже, $B_4 = 120$.

Власне переведення:

$(1,2,3,2) = (1 \cdot 105 + 2 \cdot 70 + 3 \cdot 126 + 2 \cdot 120)(\text{mod } 210) = 863(\text{mod } 210) = 23$.

Отже, (1,2,3,2) = 23.

1.6 Код Геммінга

Таблиця 1.6.1

n	i	k
7	4	3
15	11	4
31	26	5

Код Геммінга належить до кодів, які дозволяють виправляти помилки, що виникають при пересиланні інформації. Найпростіший код Геммінга дозволяє визначати та виправляти однократні помилки.

При пересиланні інформації до і інформаційних розрядів додається k перевірочних розрядів так, що загальна довжина n слова, яке пересилається, становить n = i + k розрядів. Співвідношення n, i, та k обирається з умови

$2^k \geq 1+n$, $2^{n-i} \geq 1+n$, $2^n \geq 2^i(1+n)$, $2^n/(1+n) \geq 2^i$.

Таблиця 1.6.2

n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
k_1	k_2	i_3	k_4	i_5	i_6	i_7	k_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}

Для різних значень n обчислені значення і та k наведено у табл. 1.6.1.

Формули, за допомогою яких формуються перевірочні розряди k, виводяться за допомогою перевірочної матриці (табл. 1.6.2).

У табл. 1.6.2 позначено: і - інформаційні розряди, які необхідно переслати лінією зв'язку;

k - перевірочні розряди, які додаються до інформаційних передавачем інформації;

$n_1...n_{15}$ - розряди слова, яке передається.

Номер розряду, який передається, записаний у таблиці в стовпчик у двійковому коді (старший розряд - верхній). Перевірочним розрядам відповідають ті графи таблиці, двійковий код яких має тільки одну 1. Перевірочні розряди знаходяться додаванням за модулем 2 тих інформаційних розрядів, які мають у своїй графі 1 на тому самому місці, що і відповідний перевірочний розряд:

$k_1 = i_3 \oplus i_5 \oplus i_7 \oplus i_9 \oplus i_{11} \oplus i_{13} \oplus i_{15}$;

$k_2 = i_3 \oplus i_6 \oplus i_7 \oplus i_{10} \oplus i_{11} \oplus i_{14} \oplus i_{15}$;

$k_4 = i_5 \oplus i_6 \oplus i_7 \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15}$;

$k_8 = i_9 \oplus i_{10} \oplus i_{11} \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15}$;

$\oplus$ - позначення операції додавання за модулем 2.

Для більшої кількості розрядів таблиці складаються аналогічно.

Схема пересилання інформації з використанням коду Геммінга наведено на рис. 1.6.1.

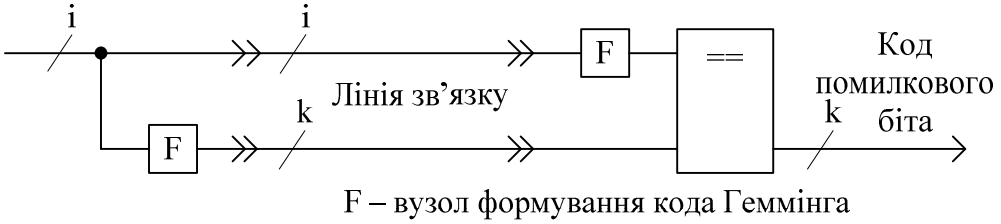


Рис. 1.6.1 Пересилання інформації з використанням коду Геммінга

На виході передавача і на вході приймача на інформаційних розрядах стоять однакові схеми формування коду Геммінга F. Їхні виходи порівнюються і при відсутності помилок повинні збігатися. Незбіжність - ознака помилки. Одночасно виходи схеми порівняння вказують, який інформаційний розряд помилковий, тобто, є можливість його виправити.

Приклад 1.6.1. Сформувати код Геммінга для двійкового коду

i = 010 0110 0011

і перевірити, як він розпізнає помилку в розряді i_{13}.

Передавач сформує перевірочні розряди (k) коду Геммінга згідно з табл. 1.6.3. Тобто, приймач повинен отримати код

n = 110 0100 0110 0011.

$k_1 = i_3 \oplus i_5 \oplus i_7 \oplus i_9 \oplus i_{11} \oplus i_{13} \oplus i_{15} = 0 \oplus 1 \oplus 0 \oplus 1 \oplus 0 \oplus 0 \oplus 1 = 1$;

$k_2 = i_3 \oplus i_6 \oplus i_7 \oplus i_{10} \oplus i_{11} \oplus i_{14} \oplus i_{15} = 0 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 1$;

$k_4 = i_5 \oplus i_6 \oplus i_7 \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15} = 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 1$;

$k_8 = i_9 \oplus i_{10} \oplus i_{11} \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15} = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 0$.

Припустимо, внаслідок помилки під час передачі інформації змінився розряд n_{13}, тобто, приймач отримав код

110 0100 0110 0(1)11

(розряд, який змінився, взятий у дужки).

Тоді приймач сформує перевірочні розряди (K) коду Геммінга згідно з табл. 1.6.4.

$K_1 = i_3 \oplus i_5 \oplus i_7 \oplus i_9 \oplus i_{11} \oplus i_{13} \oplus i_{15} = 0 \oplus 1 \oplus 0 \oplus 1 \oplus 0 \oplus (1) \oplus 1 = 0;$

$K_2 = i_3 \oplus i_6 \oplus i_7 \oplus i_{10} \oplus i_{11} \oplus i_{14} \oplus i_{15} = 0 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 1;$

$K_4 = i_5 \oplus i_6 \oplus i_7 \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15} = 1 \oplus 0 \oplus 0 \oplus 0 \oplus (1) \oplus 1 \oplus 1 = 0;$

$K_8 = i_9 \oplus i_{10} \oplus i_{11} \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15} = 1 \oplus 1 \oplus 0 \oplus 0 \oplus (1) \oplus 1 \oplus 1 = 1.$

Схема порівняння порозрядно порівнює коди K і k за допомогою операції додавання за модулем 2: $K \oplus k = <K_8 \oplus k_8><K_4 \oplus k_4><K_2 \oplus k_2><K_1 \oplus k_1> =$

$= <1 \oplus 0><0 \oplus 1><1 \oplus 1><0 \oplus 1> = 1101_2 = 13_{10}.$

Таблиця 1.6.3

n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}
1	1	0	0	1	0	0	0	1	1	0	0	0	1	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
k_1	k_2	i_3	k_4	i_5	i_6	i_7	k_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}
1	1	0	1	1	0	0	0	1	1	0	0	0	1	1

Таблиця 1.6.4

n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}
1	1	0	0	1	0	0	0	1	1	0	0	(1)	1	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
k_1	k_2	i_3	k_4	i_5	i_6	i_7	k_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}
0	1	0	1	1	0	0	1	1	1	0	0	(1)	1	1

Якщо отриманий чотирирозрядний код дорівнює 0000, то інформація передалася без помилок. Будь-який інший код указує на номер розряду, який передався з помилкою (у прикладі $1101_{(2)} = 13_{(10)}$).

1.7 Визначення помилкових станів при зміні двійкових кодів

Коли на вхід певного пристрою надходить послідовність багаторозрядних двійкових чисел, то, в ідеальному випадку, під час зміни коду числа всі його розряди повинні змінюватися одночасно. Насправді ж, завжди існує розбіжність у часі зміни окремих двійкових розрядів. Внаслідок цього можуть виникати паразитні проміжні коди на вході пристрою, який на ці коди може помилково реагувати. Тому завжди треба вміти визначати ці помилкові ситуації.

Таблиця 1.7.1

Початковий стан	Порядок зміни розрядів	Проміжні стани	Кінцевий стан
000	1-2-3	100 - 110	111
000	1-3-2	100 - 101	111
000	2-1-3	010 - 110	111
000	2-3-1	010 - 011	111
000	3-1-2	001 - 101	111
000	3-2-1	001 - 011	111

Приклад 1.7.1. На вхід пристрою подавався спочатку двійковий код 000, а потім він змінився на код 111. Визначити можливі помилкові коди.

Послідовність виникнення помилкових кодів залежать від того, який з двійкових розрядів змінюється першим. В залежності від цього можливі такі послідовності помилкових кодів (табл. 1.7.1).

0 X	1	3	2
4	5	7 X	6
C	D	F	E
8	9	B	A

Рис. 1.7.1

Якщо замінити двійкові коди вісімковими, то отримаємо такі послідовності помилкових кодів:

0 - 4 - 6 - 7; 0 - 4 - 5 - 7; 0 - 2 - 6 - 7;

0 - 2 - 3 - 7; 0 - 1 - 5 - 7; 0 - 1 - 3 - 7.

Усього серед помилкових кодів у даному прикладі будуть зустрічатися коди 1, 2, 3, 4, 5, 6.

Загалом, якщо два коди відрізняються n двійковими розрядами, то помилкових проміжних станів буде 2^n-2, а їхніх можливих послідовностей - n! (n факторіал). $n!=1\cdot2\cdot\ldots\cdot(n-1)\cdot n$.

Виявляти помилкові проміжні стани зручно за допомогою карт Карно. Для попереднього випадку карту Карно зображено на рис. 1.7.1, де хрестиками позначені два послідовних коди (0 і 7). Помилкові коди - це коди тих клітинок, якими пролягає шлях від клітинки з кодом 0 до клітинки з кодом 7. Оскільки коди 0 (0000) та 7 (0111) відрізняються на 3 двійкових розряди, то будь-який з потрібних нам шляхів від клітинки 0 до клітинки 7 буде пролягати через 2 проміжні клітинки. Загалом, якщо коди відрізняються на n розрядів, то потрібні нам шляхи будуть пролягати через n-1 клітинок карти Карно.

Старший розряд двійкового коду для клітинок 0 та 7 спільний і дорівнює 0. Тому клітинки, які мають у своєму двійковому коді старший розряд, що дорівнює 1 (8, 9, ..., E, F), не можуть утворювати помилкових кодів, і шлях від клітинки 0 до клітинки 7 через них пролягати не може.

2 Методичні вказівки щодо використання функцій алгебри логіки та мінімізації цих функцій у базисі Буля

2.1 Функціональна повнота системи функцій алгебри логіки і наборів логічних елементів

Основна вимога, яка ставиться до набору логічних елементів, полягає в тому, щоб за допомогою цього набору можна було побудувати будь-яку логічну схему. З огляду на те, що функціонування елементів однозначно описується функціями алгебри логіки (ФАЛ), застосовуючи операцію суперпозиції, можна з деякої системи ФАЛ отримати будь-яку, скільки завгодно складну ФАЛ. Тоді ця деяка система ФАЛ буде називатися функціонально повною (ФПС ФАЛ).

Функціонально повним є такий набір ФАЛ, який містить хоча б одну функцію, яка:

не зберігає константу 0;

не зберігає константу 1;

не є монотонною;

не є самодвоїстою;

не є лінійною.

Якщо функція f на нульовому наборі змінних дорівнює 0, тобто, f(0,0,...,0)=0, то ця функція зберігає константу 0.

Таблиця 2.1.1

x_0	0	0	1	1	Назва ФАЛ	Вираз ФАЛ	Клас				
x_1	0	1	0	1			0	1	Л	М	С
f_0	0	0	0	0	константа 0	0	*		*	*	
f_1	0	0	0	1	кон'юнкція, І	$x_0 \cdot x_1$	*	*		*	
f_2	0	0	1	0	заборона по x_1	$x_0 \overline{x_1}$	*				
f_3	0	0	1	1	x_0	x_0	*	*	*	*	*
f_4	0	1	0	0	заборона по x_0	$\overline{x_0} x_1$	*				
f_5	0	1	0	1	x_1	x_1	*	*	*	*	*
f_6	0	1	1	0	сума за mod 2	$x_0 \overline{x_1} \vee \overline{x_0} x_1$	*		*		
f_7	0	1	1	1	диз'юнкція, АБО	$x_0 \vee x_1$	*	*		*	
f_8	1	0	0	0	АБО-НЕ (Пірса)	$\overline{x_0 \vee x_1}$					
f_9	1	0	0	1	рівнозначність	$x_0 x_1 \vee \overline{x_0}\ \overline{x_1}$		*	*		
f_{10}	1	0	1	0	інверсія x_1	$\overline{x_1}$			*		*
f_{11}	1	0	1	1	імплікація звор.	$x_0 \vee \overline{x_1}$		*			
f_{12}	1	1	0	0	інверсія x_0	$\overline{x_0}$			*		*
f_{13}	1	1	0	1	імплікація пряма	$\overline{x_0} \vee x_1$		*			
f_{14}	1	1	1	0	І-НЕ (Шефера)	$\overline{x_0 x_1}$					
f_{15}	1	1	1	1	константа 1	1		*	*	*	

$f(x_0,x_1,...,x_n) = a_0 \cdot x_0 \oplus a_1 \cdot x_1 \oplus ... \oplus a_n \cdot x_n$,

де $a_i = (0, 1)$;

Якщо функція f на одиничному наборі змінних дорівнює 1, тобто, f(1,1,...,1)=1, то ця функція зберігає константу 1.

ФАЛ називається монотонною, якщо при будь-якому зростанні кількості 1 у послідовності сусідніх (тобто, таких, які відрізняються тільки в одному розряді) наборів змінних $(x_0,x_1,...,x_n)$ значення функції не зменшується.

ФАЛ називається самодвоїстою, якщо на кожній парі протилежних наборів $(x_0,x_1,...,x_n)$ та $(\overline{x_0},\overline{x_1},...,\overline{x_n})$ вона приймає протилежні значення, тобто, якщо виконується умова

$$f(x_0,x_1,...,x_n) = \overline{f(\overline{x_0},\overline{x_1},...,\overline{x_n})}.$$

ФАЛ називається лінійною, якщо її можна зобразити поліномом Жегалкіна без добутків змінних

· - позначення операції І;

⊕ - позначення операції «додавання за модулем 2».

Таблиця 2.1.2

№	a	b	c	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

Для того щоб записати функцію, яка задана таблично, у вигляді полінома Жегалкіна, досить записати цю функцію у вигляді суми за модулем 2 тих наборів аргументів, на яких функція дорівнює 1. Після цього потрібно всі змінні, які входять до отриманого виразу з інверсіями, замінити за допомогою співвідношення $\bar{x} = x \oplus 1$, розкрити дужки і звести подібні члени за допомогою тотожностей

x ⊕ x ⊕ ... ⊕ x = x, якщо кількість x непарна;

x ⊕ x ⊕ ... ⊕ x = 0, якщо кількість x парна;

x·x·...·x = x.

У табл. 2.1.1 наведено функції двох змінних і позначкою * вказана їх належність кожному з класів ФАЛ. У графі «Клас» позначено: 0 - зберігає константу 0; 1 - зберігає константу 1; Л - лінійна; М - монотонна; С - самодвоїста.

Приклад 2.1.1. Перевірити, чи створює функція трьох змінних, яка задана табл. 2.1.2, функціонально повну систему.

1) Оскільки на нульовому наборі f(0,0,0) = 1, то ця функція не зберігає константу 0.

2) Оскільки на одиничному наборі f(1,1,1) = 0, то ця функція не зберігає константу 1.

Таблиця 2.1.3

a	b	c	f
0	0	0	1
0	0	1	0
0	1	1	1
1	1	1	0

а

a	b	c	f
0	0	0	1
0	0	1	0
1	0	1	0
1	1	1	0

б

a	b	c	f
0	0	0	1
0	1	0	1
0	1	1	1
1	1	1	0

в

a	b	c	f
0	0	0	1
0	1	0	0
1	1	0	1
1	1	1	0

г

a	b	c	f
0	0	0	1
1	0	0	0
1	0	1	0
1	1	1	0

д

a	b	c	f
0	0	0	1
1	0	0	0
1	1	0	1
1	1	1	0

е

3) Послідовності сусідніх наборів подано в табл. 2.1.3 а, ..., е. Овалами відмічено значення функції f на сусідніх наборах, на яких вона зменшується при зростанні кількості 1 в наборах, тобто є немонотонною. Стрілочкою позначено напрям збільшення кількості 1 у сусідніх наборах змінних a, b, c.

Оскільки на всіх шести послідовностях сусідніх наборів функція немонотонна (а досить було б і на одному), то функція немонотонна взагалі.

Таблиця 2.1.4

	a	b	c	f
1	0	0	0	1
2	0	0	1	0
3	0	1	0	1
4	0	1	1	1

a	b	c	f
1	1	1	0
1	1	0	1
1	0	1	0
1	0	0	0

4) 4 пари протилежних наборів наведено в табл. 2.1.4 (номери пар проставлено збоку табл. 2.1.4).

Оскільки на кожній парі протилежних наборів функція f приймає протилежні значення, то функція самодвоїсна.

5) Для визначення лінійності функції подамо її у вигляді полінома Жегалкіна

$f = \bar{a}\bar{b}\bar{c} \oplus \bar{a}bc \oplus a\bar{b}c \oplus ab\bar{c} =$

=(1⊕a)(1⊕b)(1⊕c) ⊕ (1⊕a)b(1⊕c) ⊕ (1⊕a)bc ⊕ ab(1⊕c) =

=(1⊕a)(1⊕b⊕c⊕bc) ⊕ b(1⊕a⊕c⊕ac) ⊕ (bc ⊕ abc) ⊕ (ab ⊕ abc) =

=(1⊕b⊕c⊕bc⊕a⊕ab⊕ac⊕abc)⊕(b⊕ab⊕bc⊕abc)⊕(bc⊕abc)⊕(ab⊕abc) =

= 1 ⊕ (= 1)

⊕ a ⊕ (= a)

⊕ b ⊕ b ⊕ (= 0)

⊕ c ⊕ (= c)

⊕ ab ⊕ ab ⊕ ab ⊕ (= ab)
⊕ ac ⊕ (= ac)
⊕ bc ⊕ bc ⊕ bc ⊕ (= bc)
⊕ abc ⊕ abc ⊕ abc ⊕ abc = (= 0)
= 1 ⊕ a ⊕ c ⊕ ab ⊕ ac ⊕ bc.

Оскільки поліном містить добутки (ab, ac, bc) змінних, то функція нелінійна. Отже, із п'яти необхідних для створення ФПС властивостей відсутня одна - несамодвоїсність, тому дана функція не утворює ФПС.

2.2 Мінімізація функцій методом Квайна-МакКласкі-Петрика

Метод складається з двох частин.

1. Побудова простих імплікант.

Логічна функція g(a,b,...,x) називається імплікантою логічної функції f(a,b,...,x), якщо f(a,b,...,x) & g(a,b,...,x) = g(a,b,...,x).

Імпліканта називається простою, якщо вона є кон'юнкцією змінних, і будь-яка кон'юнкція, отримана з неї шляхом викреслювання будь-яких змінних, не є імплікантою. Побудова простих імплікант ілюструється прикладом 2.2.1.

Приклад 2.2.1. Нехай функція f(a,b,c,d) задана в табл. 2.2.1. Випишемо до графи I табл. 2.2.2 усі набори, на яких функція f обертається в 1. Для виконання алгоритму їх зручно виписати розбитими на групи у відповідності з кількістю одиничних компонент у наборах (колонка К у графі I табл. 2.2.2). Оскільки мінімізуються (склеюються) лише набори, які відрізняються в одній компоненті, то для того, щоб провести всі склеювання по одній змінній, досить продивитися всі можливі пари наборів, які входять до двох сусідніх груп. Результати склеювання наборів із графи I розмістимо у графі II. Набори із графи I, які приймали участь у склеюваннях, позначимо знаком +. У графі II набори вже автоматично розбиваються на групи за кількістю одиниць (при склеюванні наборів графи I з груп із j-1 одиницями і з j одиницями отримуються набори графи II із j-1 одиницями).

Таблиця 2.2.1

abcd	f
0000	1
0001	1
0010	0
0011	1
0100	1
0101	0
0110	1
0111	0
1000	1
1001	1
1010	0
1011	1
1100	1
1101	1
1110	0
1111	0

Таблиця 2.2.2

I			II				III	
К	П	У	С	К	П	У	С	К
0000	a_0	+	a_0b_0	000-	e_0	+	e_0f_3	-00-
0001	b_0	+	a_0b_1	0-00	e_1	+	e_1f_5	--00
0100	b_1	+	a_0b_2	-000	e_2	+	e_2f_4	--00
1000	b_2	+	b_0c_0	00-1	f_0	+	e_2f_2	-00-
0011	c_0	+	b_1c_1	01-0	f_1		f_2g_0	-0-1
0110	c_1	+	b_0c_2	-001	f_2	+	f_0g_1	-0-1
1001	c_2	+	b_2c_2	100-	f_3	+	f_3g_3	1-0-
1100	c_3	+	b_1c_3	-100	f_4	+	f_5g_2	1-0-
1011	d_0	+	b_2c_3	1-00	f_5	+		
1101	d_1	+	c_0d_0	-011	g_0	+		
			c_2d_0	10-1	g_1	+		
			c_2d_1	1-01	g_2	+		
			c_3d_1	110-	g_3	+		

До створених наборів знову застосовуємо операцію склеювання (клеяться пари наборів, які мають риску на однакових місцях і відрізняються однією компонентою). При цьому треба знову переглянути всі пари наборів із сусідніх груп. Набори, до яких застосована операція, позначені знаком +. Результати склеювання заносимо до графи III таблиці. У графі III знову намагаємося виконати склеювання, але цього зробити не вдається. На цьому процедура завершується.

У табл. 2.2.2 позначено графи:

К - код набору;

С - склеюванням яких наборів цей код утворився;

П - умовне позначення набору;

У - позначка про участь набору в склеюванні.

В отриманій таблиці знаходяться всі імпліканти функції f, які мають вигляд кон'юнкцій. Простими будуть лише ті з них, які не мають позначки + (із яких не можна викреслити жодної змінної, інакше може бути застосована операція склеювання). В розглянутому прикладі простими імплікантами є кон'юнкції

$\bar{a}b\bar{d}, \bar{b}\bar{c}, \bar{c}\bar{d}, \bar{b}d, a\bar{c}$,

які відповідають наборам

01-0, -00-, --00, -0-1, 1-0-.

2. Формування мінімальної диз'юнктивної нормальної форми.

Позначимо через I_1, I_2, ..., I_s всі прості імпліканти функції f. Будемо говорити, що кон'юнкція К покриває набір n, якщо на наборі n вона дорівнює 1. Побудуємо імплікантну таблицю (таблицю покриття) по функції f. Її рядки відповідають одиничним наборам функції f, а графи - простим імплікантам. На схрещенні рядка n і графи I_j проставляється *, якщо імпліканта I_j покриває набір n (у протилежному випадку не ставиться нічого).

Таблиця 2.2.3

Одиничні набори	I_1 $\bar{a}b\bar{d}$	I_2 $\bar{b}\bar{c}$	I_3 $\bar{c}\bar{d}$	I_4 $\bar{b}d$	I_5 $a\bar{c}$
abcd	01-0	-00-	--00	-0-1	1-0-
0000		*	*		
0001		*		*	
0011				*	
0100	*		*		
0110	*				
1000		*	*		*
1001		*		*	*
1011				*	
1100			*		*
1101					*
Літер в імплікації	3	2	2	2	2

Імплікантна табл. 2.2.3 відповідає функції, заданій у табл. 2.2.1, і імплікантам, знайденим за допомогою табл. 2.2.2.

З імплікантою I_j будемо пов'язувати логічну змінну i_j. Кожній множині імплікант приписується набір значень змінних i_j: якщо I_j входить у множину, то i_j=1, якщо ні, то i_j=0. Розглянемо рядок таблиці покриття, який відповідає якомусь набору n. Нехай у цьому рядку символи * знаходяться у графах I_1, I_2, ..., I_w. Рядок n буде покритий тоді і лише тоді, коли до множини буде введена хоча б одна з імплікант I_1, I_2, ..., I_w, тобто, коли диз'юнкція $i_1 \vee i_2 \vee \ldots \vee i_w$ дорівнює 1. Складемо таку диз'юнкцію для кожного рядка імплікантної таблиці і візьмемо їхній добуток по всіх рядках. Отримаємо функцію F, яка для табл. 2.2.3 має вигляд

$F=(I_2 \vee I_3)(I_2 \vee I_4)I_4(I_1 \vee I_3)I_1(I_2 \vee I_3 \vee I_5)(I_2 \vee I_4 \vee I_5)I_4(I_3 \vee I_5)I_5$.

Після спрощення $F = I_1I_2I_4I_5 \vee I_1I_3I_4I_5$. Тепер функція F вказує на 2 ненадмірних покриття: $\{I_1, I_2, I_4, I_5\}$ та $\{I_1, I_3, I_4, I_5\}$. Їм відповідають дві ДНФ, кожна з яких складаються з 9 літер, тому кожна з них є мінімальною. Таким чином, дана функція f має дві мінімальні ДНФ: $f = I_1 \vee I_2 \vee I_4 \vee I_5 = \bar{a}b\bar{d} \vee \bar{b}\bar{c} \vee \bar{b}d \vee a\bar{c}$,

$f = I_1 \vee I_3 \vee I_4 \vee I_5 = \bar{a}b\bar{d} \vee \bar{c}\bar{d} \vee \bar{b}d \vee a\bar{c}$.

При спрощенні слід брати до уваги:

в першу чергу, вирази вигляду $...(i_1 \vee i_3)i_1...$ спрощуються до $... i_1...$, оскільки $a(a \vee b) = a$;

аналогічно, вирази вигляду ...(i_1 v i_3)(i_1 v i_2)... спрощуються до ... (i_1 v i_2i_3) ..., і далі до … i_1 …, оскільки кількість літер у двох імплікантах i_2 та i_3 менша, як правило, за кількість літер в одній імпліканті i_1;

по-друге, якщо у дужках є декілька імплікант з різною кількістю літер, то в результаті спрощення у дужці залищаться лише імпліканти з мінімальною кількістю літер. Наприклад, у наведеному вище прикладі дужка (i_1 v i_3), до якої входять імпліканти з кількістю літер, відповідно, 3 і 2, може спроститися до виразу i_3 (кількість літер 2).

Мінімізація не повністю визначених функцій методом Квайна-МакКласкі-Петрика.

Таблиця 2.2.4

abcd	f
0000	0
0001	0
0010	0
0011	0
0100	0
0101	1
0110	0
0111	X
1000	0
1001	0
1010	X
1011	X
1100	0
1101	X
1110	X
1111	1

Таблиця 2.2.5

I			II				III	
К	П	У	С	К	П	У	С	К
0101	a_0	+	a_0b_0	01-1	d_0	+	d_0e_2	-1-1
1010	a_1	+	a_0b_2	-101	d_1	+	d_1e_0	-1-1
0111	b_0	+	a_1b_1	101-	d_2	+	d_2e_3	1-1-
1011	b_1	+	a_1b_3	1-10	d_3	+	d_3e_1	1-1-
1101	b_2	+	b_0c_0	-111	e_0	+		
1110	b_3	+	b_1c_0	1-11	e_1	+		
1111	c_0	+	b_2c_0	11-1	e_2	+		
			b_3c_0	111-	e_3	+		

Всі дії відбуваються, як описано вище. Лише на першому етапі до табл. 2.2.2 як початкові набори вводяться додатково і всі набори, на яких функція не визначена, а на другому – до табл. 2.2.3, як і раніше, тільки набори, на яких функція приймає значення 1.

Приклад 2.2.2. Функція задана табл.2.2.4, X – невизначене значення.

Знаходження простих імплікант - табл. 2.2.5.

Знаходження мінімального покриття - табл. 2.2.6.

Результат: F=I_1(I_1 v I_2) =I_1;

f = I_1 = *bd.*

2.3 Мінімізація функцій за допомогою карт Карно

Карти Карно є одним з найефективніших засобів знаходження мінімальних диз'юнктивних нормальних форм (МДНФ) для функцій невеликої кількості змінних (4...6).

Таблиця 2.2.6

Одиничні набори	I_1	I_2
	bd	ac
abcd	-1-1	1-1-
0101	*	
1111	*	*
Літер в імплікації	2	2

На картах Карно кожному з 2^n наборів відповідає одна клітинка. Якщо на даному наборі аргументів функція дорівнює 1, то в клітинці, яка відповідає даному набору, записується 1. Клітинки, які відповідають наборам, де функція дорівнює 0, або заповнюють 0, або залишають незаповненими. Клітинки, які відповідають наборам, де функція недовизначена, заповнюють X (рис. 2.3.1).

Кожна із змінних x розбиває карту Карно на дві половини - половину x і половину (-x). Половини карт, які відповідають неінверсним аргументам, позначені на рис. 2.3.1, де зображено карту Карно на 5 змінних - e, a, b, c, d, які утворюють набори {eabcd}.

Номери наборів проставлені у верхніх лівих кутках клітинок карти (у шістнадцятковому коді).

Правила мінімізації такі:

1) сусідніми вважаються клітинки, які відрізняються значенням лише однієї змінної, тобто:

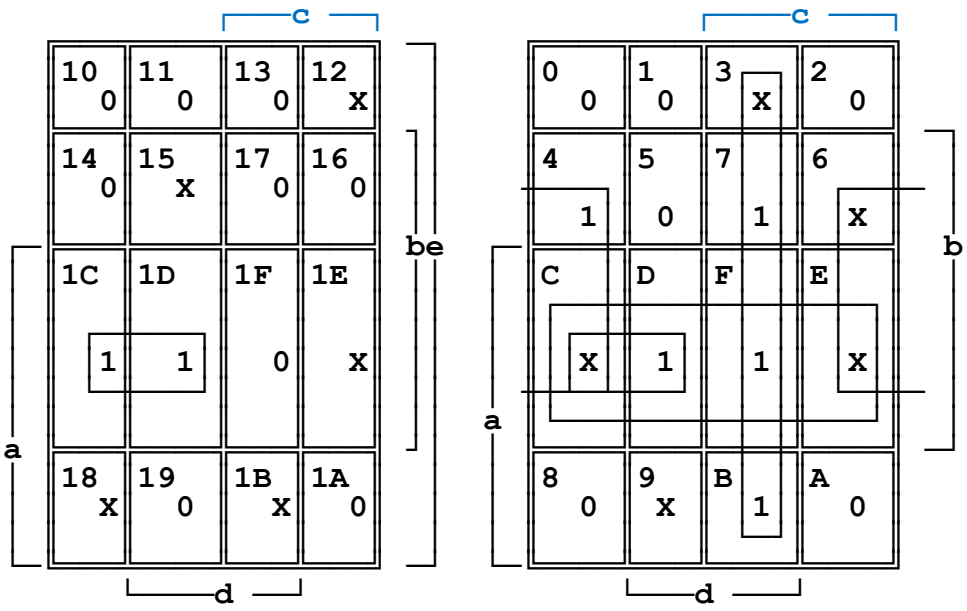


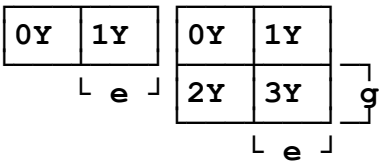
Рис. 2.3.1

а) дві клітинки, які мають спільну грань;

б) клітинки, які знаходяться в крайньому правому і крайньому лівому стовпцях однієї карти в одному рядку;

в) клітинки, які знаходяться в крайньому верхньому і крайньому нижньому рядках однієї карти в одному стовпчику;

г) клітинки, які займають однакове положення в сусідніх картах.



Примітка: Y = 0,...,F.

Рис. 2.3.2.

Для випадку 5 змінних (e, a, b, c, d) мінімізація проводиться на двох картах, для 6 (g, e, a, b, c, d) змінних - на 4-х картах (рис. 2.3.2). Кожна карта розбивається змінними a, b, c, d на 16 клітинок однаково, як це показано на рис. 2.3.1.

2) Дві сусідні клітинки об'єднуються (склеюються), при цьому зникає змінна, якою ці клітинки відрізняються.

3) 4 сусідні клітинки, які утворюють прямокутник (або у просторі прямокутний паралелепіпед, якщо накласти карти одна на одну), об'єднуються (склеюються), при цьому зникають 2 змінні, якими ці клітинки відрізняються.

4) 8 сусідніх клітинок, які утворюють прямокутник (прямокутний паралелепіпед), об'єднуються (склеюються), при цьому зникають 3 змінні, якими ці клітинки відрізняються.

5) 16 сусідніх клітинок, які утворюють прямокутник (прямокутний паралелепіпед), об'єднуються (склеюються), при цьому зникають 4 змінні, якими ці клітинки відрізняються.

6) Загалом: 2^n сусідніх клітинок, які утворюють прямокутник (прямокутний паралелепіпед), об'єднуються (склеюються), при цьому зникають n змінних, якими ці клітинки відрізняються.

7) Кожну клітинку можна склеювати безліч разів.

8) Невизначене значення функції (позначається на карті X) можна вважати як 0, так і 1 в залежності від зручності мінімізації.

9) Так само, як і за 1, можна робити склеювання і за 0. В результаті буде отриманий вираз для інверсної функції $\overline{f}$.

10) Одночасно можна клеїти або тільки за 1, або тільки за 0.

Приклад 2.3.1. Мінімізувати функцію, задану на рис. 2.3.1, за 1.

Результати склеювання позначені на рис. 2.3.1. Всі 4 склеювання - по 4 клітинки:

клітинки 4, C, 6, E - результат $\bar{e}\bar{b}\bar{d}$, зникли 2 змінні a і c. Це склеювання потрібне, щоб мінімізувати набір 4, на якому функція визначена і дорівнює 1;

клітинки C, D, F, E - результат $\bar{e}ab$, зникли 2 змінні c і d. Це склеювання потрібне, щоб мінімізувати набори D і F, на яких функція визначена і дорівнює 1;

клітинки 3, 7, F, B - результат $\bar{e}cd$, зникли 2 змінні a і b. Це склеювання потрібне, щоб мінімізувати набори 7, F і B, на яких функція визначена і дорівнює 1;

клітинки C, D, 1C, 1D - результат $ab\bar{c}$, зникли 2 змінні e і d. Це склеювання потрібне, щоб мінімізувати набір 1C і 1D, на яких функція визначена і дорівнює 1.

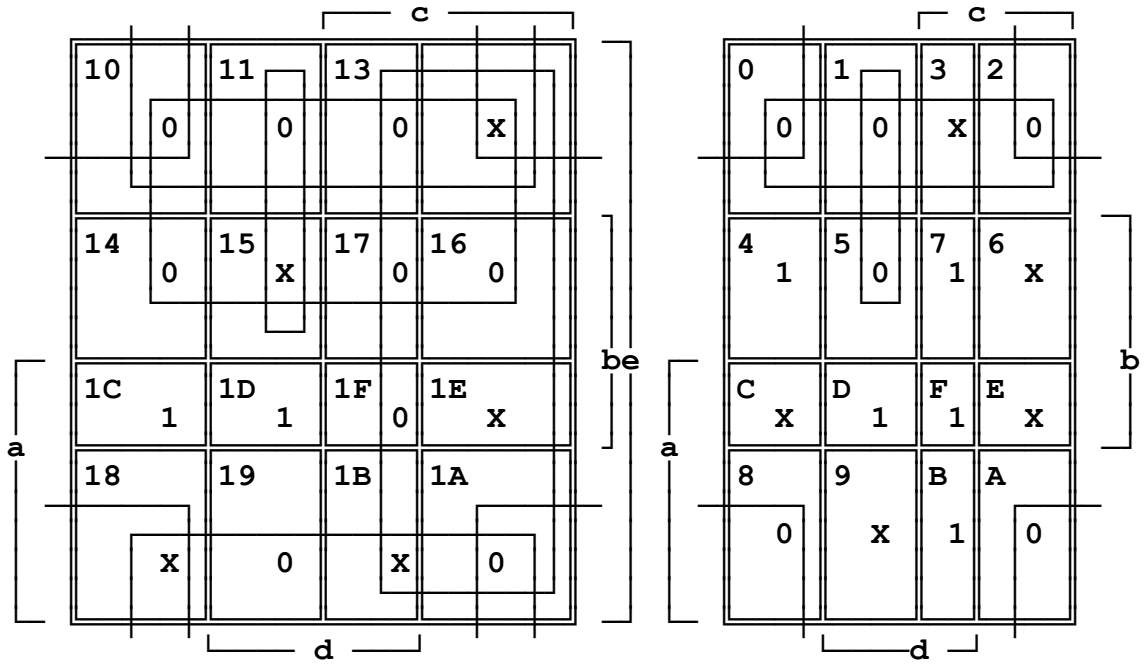


Рис. 2.3.3.

Невизначені значення функції в клітинках (наборах) 9, 12, 15, 1E, 18, 1B довизначаємо як 0, оскільки вони не допомагають при склеюванні за 1.

Невизначені значення функції в клітинках (наборах) 3, 6, C, E довизначаємо як 1, оскільки вони допомагають при склеюванні за 1.

Набори 3, 4, 6, 7, D, B, 1C, 1D - кожен бере участь в одному склеюванні.

Набори D, F, E - беруть участь у двох склеюваннях кожний.

Набір C - бере участь у трьох склеюваннях.

Остаточний результат:

$f = \bar{e}\bar{b}\bar{d} \vee \bar{e}ab \vee \bar{e}cd \vee \bar{e}cd \vee ab\bar{c}$.

Приклад 2.3.2. Мінімізувати функцію, задану на рис. 2.3.1, за 0.

Результати склеювання позначені на рис. 2.3.3.

5 склеювань по 8 клітинки:

клітинки 0, 1, 2, 3, 10, 11, 12, 13 - результат $\bar{a}\bar{b}$, зникли 3 змінні - c, d і e. Це склеювання потрібне, щоб мінімізувати набори 0, 1, 2, 10, 11, 12, на яких функція визначена і дорівнює 0;

клітинки 0, 2, 8, A, 10, 12, 18, 1A - результат $\bar{b}\bar{d}$, зникли 3 змінні - a, c і e. Це склеювання потрібне, щоб мінімізувати набори 8, A, 1A, на яких функція визначена і дорівнює 0 (набори 0, 2, 10, 12 вже були мінімізовані на попередньому етапі);

клітинки 10, 11, 12, 13, 14, 15, 16, 17 - результат $\bar{a}e$, зникли 3 змінні - b, c і d. Це склеювання потрібне, щоб мінімізувати набори 14, 16, 17, на яких функція визначена і дорівнює 0 (набори 10, 11, 12 вже були мінімізовані на попередніх етапах);

клітинки 10, 11, 12, 13, 18, 19, 1A, 1B - результат $\bar{b}e$, зникли 3 змінні - a, c і d. Це склеювання потрібне, щоб мінімізувати набори 19, 1A, на яких функція визначена і дорівнює 0 (набори 10, 11, 12 вже були мінімізовані на попередніх етапах);

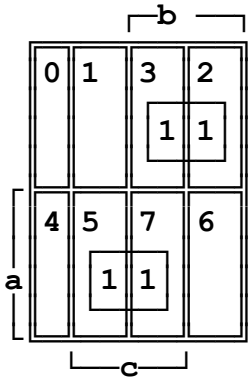


Рис. 2.4.1

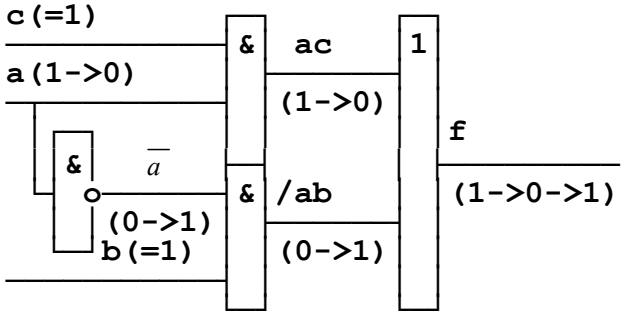


Рис. 2.4.2

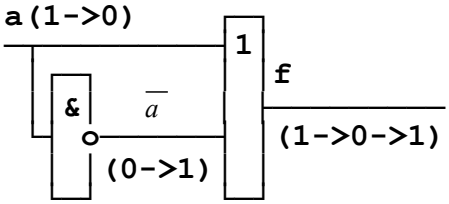


Рис. 2.4.3

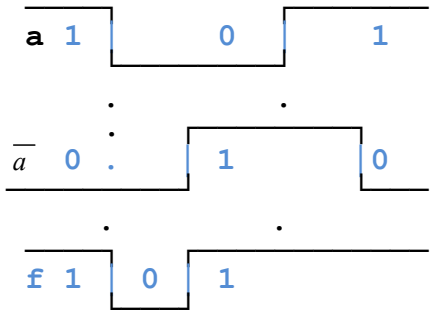


Рис. 2.4.4

клітинки 12, 13, 16, 17, 1A, 1B, 1E, 1F - результат ce, зникли 3 змінні - a, b і d. Це склеювання потрібне, щоб мінімізувати набір 1F, на якому функція визначена і дорівнює 0 (набори 12, 16, 17, 1A вже були мінімізовані на попередніх етапах).

Одне склеювання по 4 клітинки:

клітинки 1, 5, 11, 15 - результат $\bar{a}\bar{c}d$, зникли 2 змінні - b і e. Це склеювання потрібне, щоб мінімізувати набір 5, на якому функція визначена і дорівнює 0 (набори 1 і 11 вже були мінімізовані на попередніх етапах).

Невизначені значення функції в клітинках (наборах) 6, 9, C, F довизначаємо як 1, оскільки вони не допомагають при склеюванні за 0.

Невизначені значення функції в клітинках (наборах) 3, 13, 15, 18, 1B довизначаємо як 0, оскільки вони допомагають при склеюванні за 0.

Набори 3, 5, 8, A, 14, 19, 1E, 1F - беруть участь в одному склеюванні кожний.

Набори 0, 1, 2, 16, 17, 18, 1B - беруть участь у двох склеюваннях кожний.

Набори 10, 11, 1A - беруть участь у трьох склеюваннях кожний.

Набір 12 - бере участь у чотирьох склеюваннях.

Оскільки склеювання проводиться за 0, то в результаті мінімізації отримаємо диз'юнктивну нормальну форму для інверсного значення функції $\overline{f}$:

ДНФ: $\overline{f} = \bar{a}\bar{b} \vee \bar{b}\bar{d} \vee \bar{a}e \vee \bar{b}e \vee ce \vee \bar{a}\bar{c}d$.

Пряме значення функції f отримується згідно з правилами Моргана у вигляді кон'юнктивної нормальної форми:

КНФ: $f = (a \vee b)(b \vee d)(a \vee \bar{e})(b \vee \bar{e})(\bar{c} \vee \bar{e})(a \vee c \vee \bar{d})$.

2.4 Визначення сполучного терма

Розглянемо функцію f(a, b, c), яка задана картою Карно (рис. 2.4.1).

Результати мінімізації також зображені на рис. 2.4.1. Результат мінімізації $f(a,b,c) = ac \vee \bar{a}b$. Схему, яка реалізує цю функцію, наведено на рис. 2.4.2.

Розглянемо, що відбувається на виході схеми, коли інформація на входах міняється, а власне комбінація *abc* (7-ий набір) міняється на комбінацію $\bar{a}bc$ (3-ій набір). При такому переході змінні b і c залишаються сталими (=1), а змінна a міняє свій стан від 1 до 0, що зображено на рис. 2.4.2. Згідно з рис. 2.4.1, значення f не повинно при цьому змінюватися. Еквівалентну схему для цього переходу зображено на рис. 2.4.3, а часова діаграма - на рис. 2.4.4 (сигнал f на рис. 2.4.4. показано без врахування внутрішньої затримки елемента АБО).

Через затримку на елементі НЕ сигнал $\bar{a}$ змінюється з запізненням відносно сигналу a. Це призводить до одночасної появи на входах елемента АБО 2-х сигналів низького рівня, наслідком чого буде поява короткочасного сигналу 0 (просічки) на виході елемента АБО. Виникає так званий ефект перегонів сигналів. Щоб позбутися цього негативного ефекту, необхідно об'єднати всі сусідні набори, на яких функція приймає значення 1 і які не об'єднані в результаті мінімізації спільним склеюванням (спільним термом), за допомогою так званого сполучного терма. Для наведеного прикладу це показано на рис. 2.4.5. Остаточний результат мінімізації із врахуванням сполучного терма:

$f(a,b,c) = ac \vee \bar{a}b \vee bc$. Тут bc - сполучний терм.

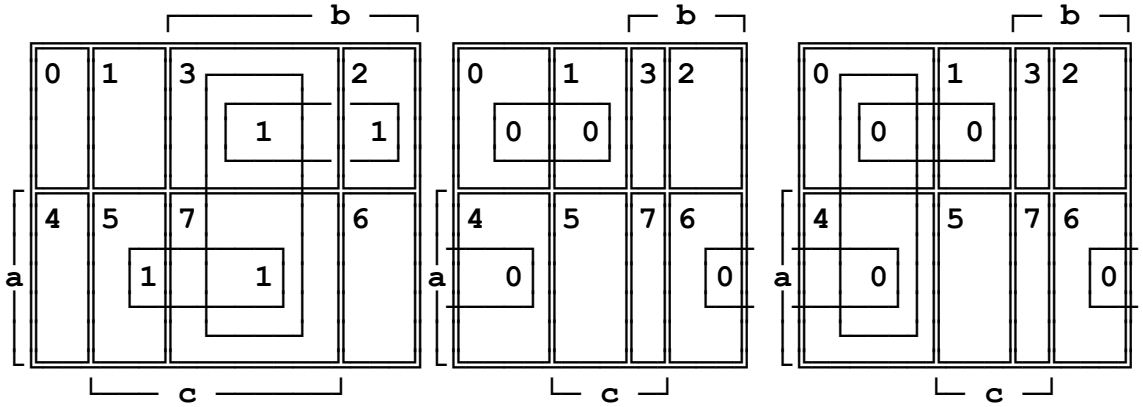


Рис. 2.4.5

Рис. 2.4.6

Рис. 2.4.7

Аналогічна ситуація може виникати при склеюванні за 0. Наприклад, рис. 2.4.6, де $\overline{f} = a\bar{c} \vee \bar{a}\bar{b}$. Перегони можуть виникати при здійсненні переходів між 0 і 4 наборами. Для їх усунення вводиться сполучний терм, як показано на рис. 2.4.7. Внаслідок цього остаточний результат буде $\overline{f} = a\bar{c} \vee \bar{a}\bar{b} \vee \bar{b}\bar{c}$, де $\bar{b}\bar{c}$ - сполучний терм.

Таблиця 2.4.1

Можливі комбінації розрядів пар простих імплікант

Перший елемент пари	Другий елемент пари	Результат
-	-	-
-	0	0
-	1	1
0	-	0
0	0	0
0	1	x(-)
1	-	1
1	0	x(-)
1	1	1

Визначення сполучного терма при записі методом Мак-Класкі.

Попарне порозрядне порівняння (табл. 2.4.1) простих імплікант здійснюється після виконання мінімізації методом Квайна-Мак-Класкі для визначення сполучного терма. Якщо при порівнянні будь-якої пари імплікант символ «х» зустрічається тільки один раз, то це говорить про необхідність введення сполучного терму. Розряди такого терма визначаються згідно з графою «Результат» табл. 2.4.1 і при цьому символ «х» заміняється міткою «-».

Приклад 2.4.1. Визначити сполучні терми для функції $f = \bar{b}\bar{c} \vee \bar{a}b \vee acd$.

При записі методом Мак-Класкі функція буде мати вигляд табл. 2.4.2. Результати попарного порівняння наведено в табл. 2.4.3. Оскільки при кожному порівнянні сформувався один символ «х», то кожний терм є сполучним і тому необхідним. Після введення сполучних термів, функція буде мати вигляд

$$f = \bar{b}\bar{c} \vee \bar{a}b \vee acd \vee \bar{a}\bar{c} \vee a\bar{b}d \vee bcd .$$

Наведений приклад ілюструється картою Карно (рис. 2.4.8).

Таблиця 2.4.2

Номер імпліканти	abcd
i_0	-00-
i_1	01--
i_2	1-11

Таблиця 2.4.3

Порівнюються		Порівнюються		Порівнюються	
i_0	-00-	i_0	-00-	i_1	01--
i_1	01--	i_2	1-11	i_2	1-11
Результат	0x0-	Результат	10x1	Результат	x111
	0-0-		10-1		-111
Терм	/a/c	Терм	a/bd	Терм	bcd

Таблиця 2.4.4

Номер імпліканти	abcd
i_0	0-0-
i_1	01--
i_2	1010

Приклад 2.4.2. Визначити сполучні терми, для функції $f = \bar{a}\bar{c} \vee \bar{a}b \vee a\bar{b}\bar{c}\bar{d}$.

При записі методом Мак-Класкі функція буде мати вигляд табл. 2.4.4. Результати попарного порівняння наведено в табл. 2.4.5. Оскільки при попарному порівнянні або не формувалося жодного символу «х», або їх формувалося більше ніж один, то в даному

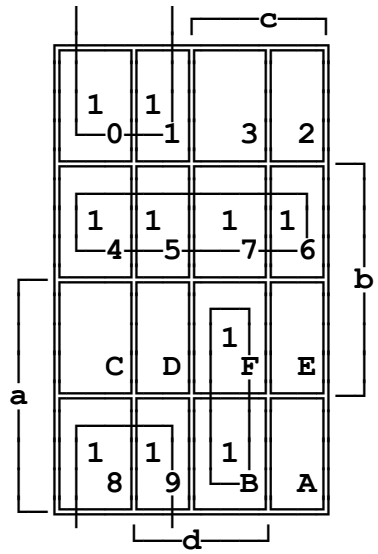
прикладі сполучних термів не буде. Наведений приклад ілюструється картою Карно (рис. 2.4.9).

Таблиця 2.4.5

Порівнюються	
i_0 i_1	0-0- 01--
Результат	010 - немає
Терм	немає

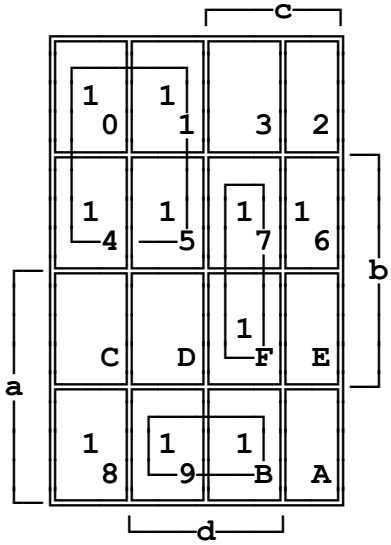
Порівнюються	
i_0 i_2	0-0- 1010
Результат	x0x0 немає
Терм	немає

Порівнюються	
i_1 i_2	01-- 1010
Результат	xx10 немає
Терм	немає

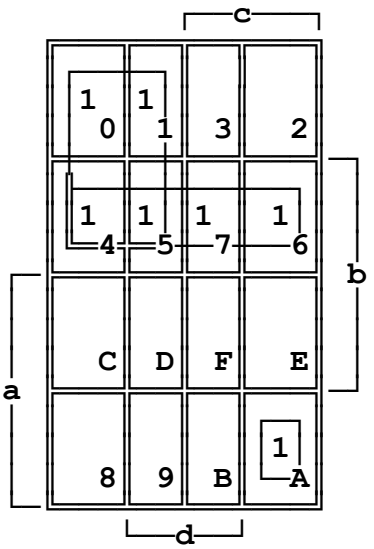


Задана функція

Рис. 2.4.8

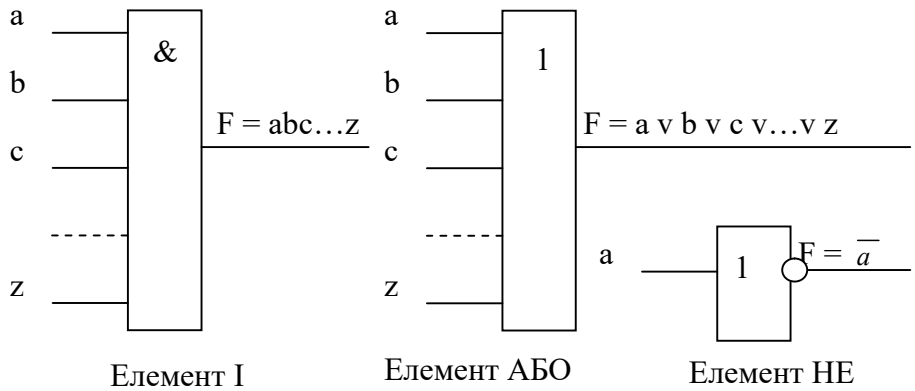


Сполучні терми



Задана функція

Рис. 2.4.9



Елемент І Елемент АБО Елемент НЕ

Рис. 3.1.1

3 Методичні вказівки щодо синтезу комбінаційних схем

3.1 Синтез функцій у базисі Буля на елементах з довільною кількістю входів

Базис Буля (базис І, АБО, НЕ) складається з трьох функцій алгебри логіки (ФАЛ):

функція І (кон'юнкція, логічне множення, AND, в аналітичному запису - &, *),

кількість входів – більше 1;

функція АБО (диз'юнкція, логічне додавання, OR, в аналітичному запису – «v», «+», «|»), кількість входів – більше 1;

функція НЕ (інверсія, в аналітичному запису – риска над символом, або «/» перед символом, або «–» перед символом) , кількість входів – 1.

Умовні графічні позначення елементів І, АБО, НЕ наведено на рис. 3.1.1.

Таблиця 3.1.1

Номер набору	Аргументи			Функція	Терм
	a	b	c	f	
0	0	0	0	0	
1	0	0	1	0	
2	0	1	0	1	$\bar{a}b\bar{c}$
3	0	1	1	1	$\bar{a}bc$
4	1	0	0	1	$a\bar{b}\bar{c}$
5	1	0	1	0	
6	1	1	0	0	
7	1	1	1	1	abc

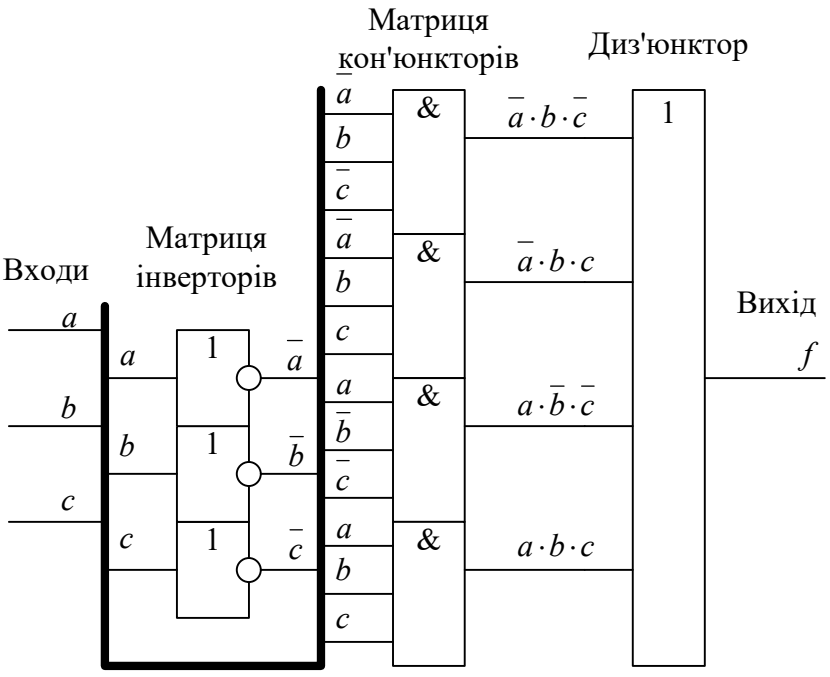


Рис. 3.1.2

На виході F елемента І буде одиниця тільки тоді, коли на всіх його входах a, b, c, …, z є одиниця.

На виході F елемента АБО буде одиниця тоді, коли хоча б на одному з його входів a, b, c, …, z є одиниця.

На виході F елемента НЕ буде одиниця тоді, коли на його вході a є нуль.

Таблиця істиності задає значення ФАЛ при всіх можливих станах аргументів (змінних) цієї функції.

Досконалі диз'юнктивні нормальні форми (ДДНФ) це аналітичний вираз ФАЛ у вигляді диз'юнкції кон'юнкцій. При цьому кожна кон'юнкція містить усі змінні (аргументи) ФАЛ, а кількість термів дорівнює кількості одиничних значень функції у її таблиці істиності. ДДНФ визначає, коли дана ФАЛ набуває значення одиниці.

Для того, щоб утворити з таблиці істиності ДДНФ необхідно:

визначити з таблиці істиності, скільки разів (N) функція набуває значення одиниці;

написати рівняння

F = abc…z V abc…z V abc…z V … V abc…z,

де добуток (терм) abc…z містить усі аргументи функції і повторюється N разів. При цьому перший терм рівняння відповідає першому набору таблиці істиності, на якому ФАЛ набуває значення одиниці, другий – другому, і так далі, а останній, відповідно, останньому;

розставити позначки інверсії змінних в кожному термі згідно із значенням цих змінних у відповідних наборах таблиці істиності: якщо змінна у відповідному наборі таблиці істиності має значення 0, то у термі вона повинна мати позначку інверсії (інверсне значення), якщо у набрі змінна має значення 1, то у термі змінна повинна бути без інверсії (пряме значення).

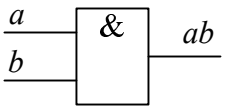
Таблиця 3.2.1

Номер набору	Аргументи		Функція	
	a	b	І	АБО
0	0	0	0	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	1

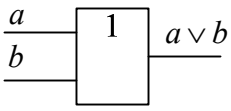
Таблиця 3.2.2

Номер набору	Аргументи	Функція
	a	НЕ
0	0	1
1	1	0

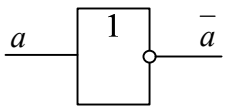
Приклад 3.1.1. ФАЛ задана таблицею істиності (таблиця 3.1.1). Скласти ДДНФ для даної функції.



елемент 2І (2AND), кон'юнктор



елемент 2АБО (2OR), диз'юнктор



елемент НЕ (NOT), інвертор

Рис. 3.2.1

Рішення: $f = \bar{a}b\bar{c} \vee \bar{a}bc \vee a\bar{b}\bar{c} \vee abc$.

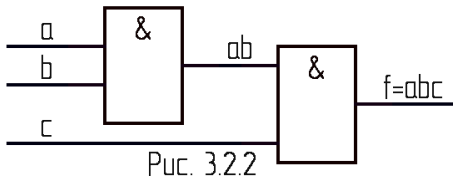


Рис. 3.2.2

Функція набуває значення одниниці чотири рази, тому ДДНФ містить чотири терми.

Функціональна схема ФАЛ, заданої у вигляді ДДНФ, у базисі Буля складається з трьох послідовно з'єднаних частин:

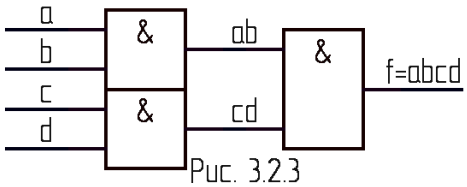


Рис. 3.2.3

матриці інверторів, яка реалізує інверсні значення змінних;

матриці елементів І, яка реалізує окремі терми ФАЛ;

елемента АБО, на виході якого власне і

реалізується ФАЛ.

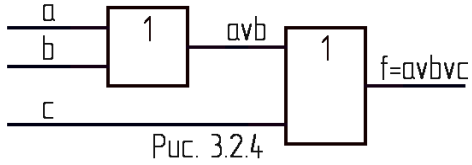


Рис. 3.2.4

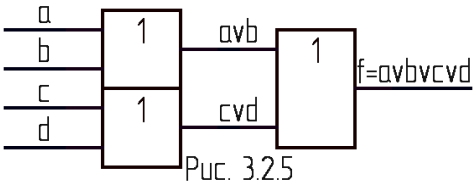


Рис. 3.2.5

Приклад 3.1.2. Синтезувати функціональну схему ФАЛ для ДДНФ Прикладу 3.1.1.

Результат синтезу наведений на рис. 3.1.2.

3.2 Синтез функцій у базисі Буля на елементах з обмеженою кількістю входів

У ході виконання курсової роботи необхідно синтезувати функціональну схему, яка реалізує задану функцію, з використанням елементів І та ІБО, які мають по 2 входи, і елементаів НЕ, які мають 1 вхід. Таблиці істинності цих елементів наведено у табл. 3.2.1 та табл. 3.2.2, а умовні графічні позначення відповідних елементів – на рис. 3.2.1. Реалізація функцій 3І $f = abc$, 4І $f = abcd$, 3АБО $f = a \vee b \vee c$ та 4АБО $f = a \vee b \vee c \vee d$ на елементах І та АБО з двома входами каскадна, її зображено на рис. 3.2.2 – рис. 3.2.5.

Приклад 3.2.1. Реалізувати функцію $f = abc \vee deg \vee hij$. Результат реалізації зображено на рис. 3.2.6.

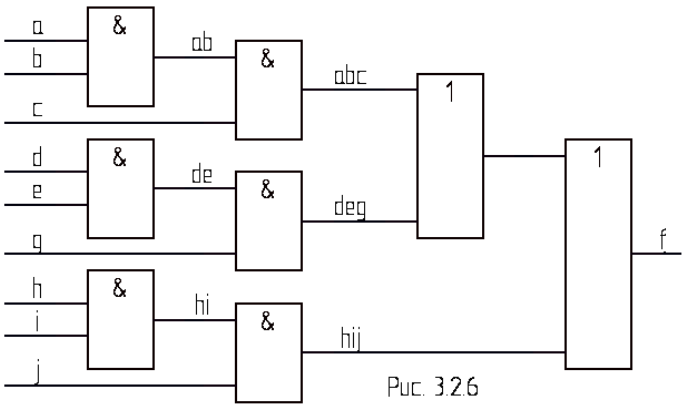
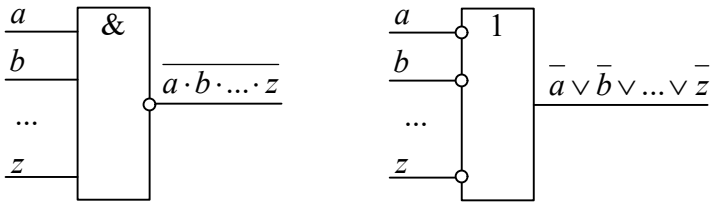


Рис. 3.2.6

3.3 Синтез функцій у монобазисі І-НЕ

Елементи монобазиса І-НЕ повинні мати кількість входів не менше 2. При одному вході елемент І-НЕ перетворюється на інвертор. Відома інша назва цієї функції - заперечення кон'юнкції. Багатовходовий елемент nІ-НЕ (символ n у назві вказує на кількість входів елемента) реалізує функцію $f = \overline{abc...z}$ - функцію І-НЕ n змінних. Згідно з правилом Моргана $f = \overline{abc...z} = \bar{a} \vee \bar{b} \vee \bar{c} \vee ... \vee \bar{z}$. Тобто, можуть існувати 2 абсолютно рівноправні умовні графічні позначення (УГП) цього елемента: перше - як елемента nІ-НЕ, друге - як елемента НЕ-nАБО (рис. 3.3.1).



елемент І-НЕ (NAND) елемент НЕ-АБО

Рис. 3.3.1

На виході f елемента І-НЕ буде ноль тільки тоді, коли на всіх його входах a, b, c, …, z є одиниця.

Реалізація функції інверсії у монобазисі І-НЕ відбувається відповідно до формули: f = $\bar{a}$ на елементі, який має 1 вхід (на інверторі).

Реалізація функції І n змінних у монобазисі nІ─НЕ відбувається відповідно до формули $f = \overline{\overline{abc...z}}$, як послідовність операцій nІ-НЕ та інверсії. Схемну реалізацію наведено на рис. 3.3.2.

Реалізація функції АБО n змінних у монобазисі nI-НЕ відбувається відповідно до формули $f = \overline{a} \vee \overline{b} \vee \overline{c} \ldots \vee \overline{z}$, як послідовність операцій інверсії кожної змінної та НЕ-nАБО. Схемну реалізацію наведено на рис. 3.3.3.

Реалізація функції І-АБО в монобазисі І-НЕ відбувається як послідовність операцій І-НЕ та НЕ-АБО з потрібною кількістю входів.

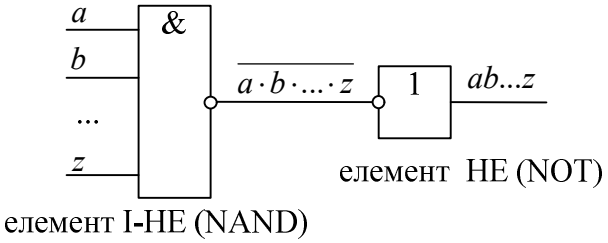


Рис. 3.3.2

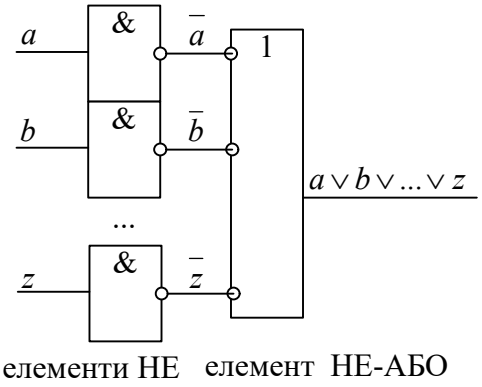


Рис. 3.3.2

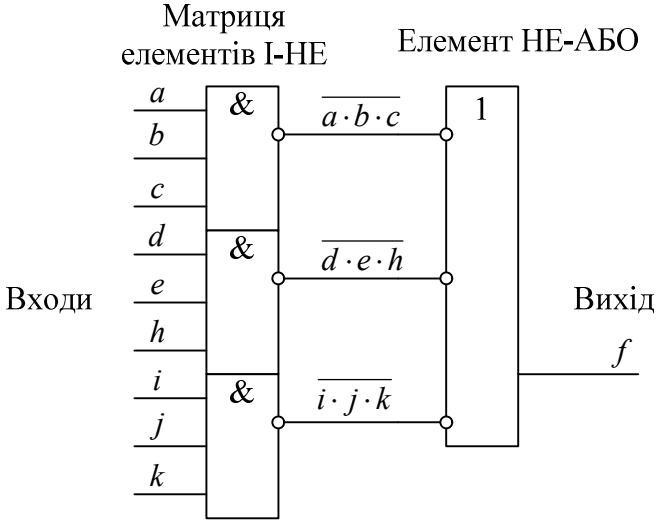


Рис. 3.3.4

Приклад 3.3.1. Реалізувати задану у вигляді диз'юнктивної нормальної форми (ДНФ) функцію f=abc v deg v hij. Результат реалізації зображено на рис. 3.3.4. Доречно звернути увагу, на те, що на всіх наведених рисунках прямі виходи з'єднуються із прямими входами, а інверсні ─ з інверсними.

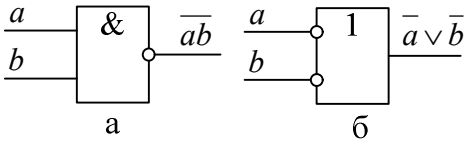
Таблиця 3.4.1

Номер	Аргументи	Функція

набору	a	b	І-НЕ
0	0	0	1
1	0	1	1
2	1	0	1
3	1	1	0

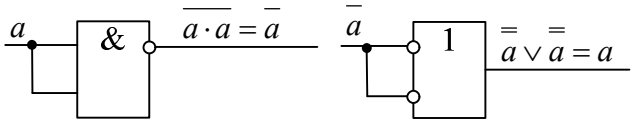
3.4 Синтез функцій у монобазисі 2І-НЕ (Шеффера)

Елементи монобазиса 2І-НЕ мають два входи. Відомі інші назви цієї функції: заперечення кон'юнкції, елемент Шеффера, штрих Шеффера. Двовходовий елемент 2І-НЕ (символ 2 у назві вказує на кількість входів елемента) реалізує функцію $f = \overline{ab}$ - функцію І-НЕ двох змінних. Згідно з правилом Моргана $f = \overline{ab} = \bar{a} \vee \bar{b}$. Тобто, можуть існувати 2 абсолютно рівноправні умовні графічні позначення (УГП) цього елемента: перше, як елемента 2І-НЕ (рис. 3.4.1, а), друге - як елемента НЕ-2АБО (рис. 3.4.1, б).



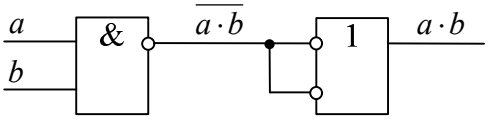
а б

Рис. 3.4.1



елемент 2І-НЕ елемент НЕ-2АБО

Рис. 3.4.2



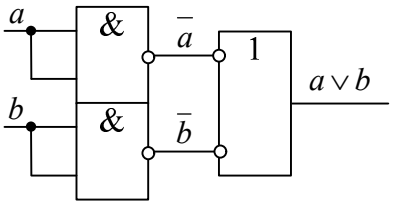
елемент 2І-НЕ елемент НЕ-2АБО

Рис. 3.4.3

Таблицю істинності елемента 2І-НЕ наведено у табл. 3.4.1.

Реалізація функції інверсії у монобазисі 2І-НЕ відбувається відповідно до формули $f = \overline{aa} = \bar{a}$. Схемну реалізацію інверсії наведено на рис. 3.4.2: на обидва входи елемента треба подати сигнал, який потрібно інвертувати.

Реалізація функції І в монобазисі 2І–НЕ відбувається відповідно до формули $f = \overline{\overline{ab}}$, як послідовність операцій 2І-НЕ та інверсії. Схемну реалізацію наведено на рис. 3.4.3.

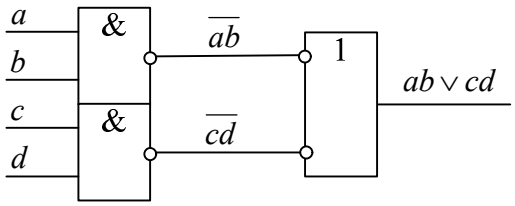


елементи 2І-НЕ елемент НЕ-2АБО

Рис. 3.4.4

Реалізація функції АБО в монобазисі 2І-НЕ відбувається відповідно до формули $f = \bar{\bar{a}} \vee \bar{\bar{b}}$, як послідовність операцій інверсії кожної змінної та НЕ-2АБО. Схемну реалізацію наведено на рис. 3.4.4.

Реалізація функції І-АБО в монобазисі 2І-НЕ відбувається відповідно до формули $f = \overline{\overline{ab}} \vee \overline{\overline{cd}}$ як послідовність операцій 2І-НЕ та НЕ-2АБО. Схемну реалізацію наведено на рис. 3.4.5.



елементи 2І-НЕ елемент НЕ-2АБО

Рис. 3.4.5

Реалізація функції 3І-НЕ $f = \overline{abc}$ каскадна, її зображено на рис. 3.4.6. Реалізація функції 4І─НЕ $f = \overline{abcd}$ каскадна, її зображено на рис. 3.4.7. Реалізація функції 3АБО $f = a \vee b \vee c$ каскадна, її зображено на рис. 3.4.8.

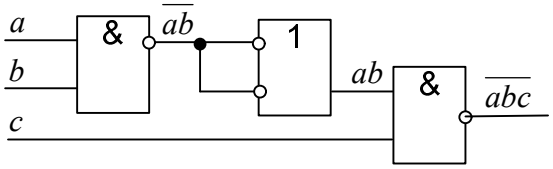


Рис. 3.4.6

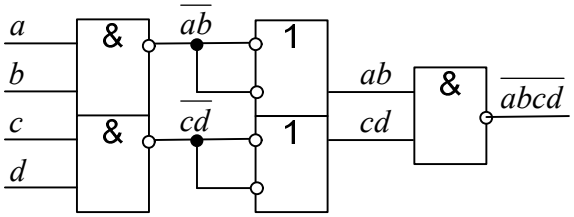


Рис. 3.4.7

Реалізація функції 4АБО $f = a \vee b \vee c \vee d$ каскадна, її зображено на рис. 3.4.9.

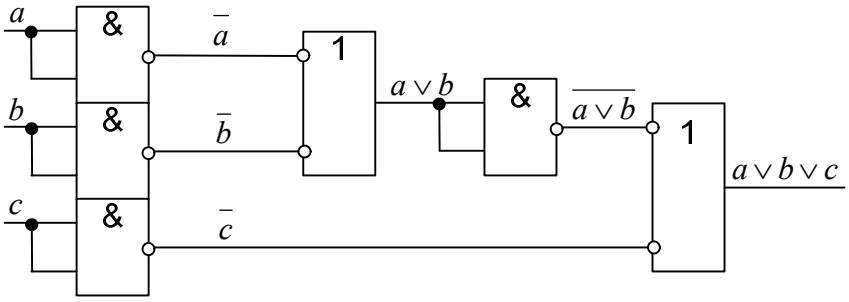


Рис. 3.4.8

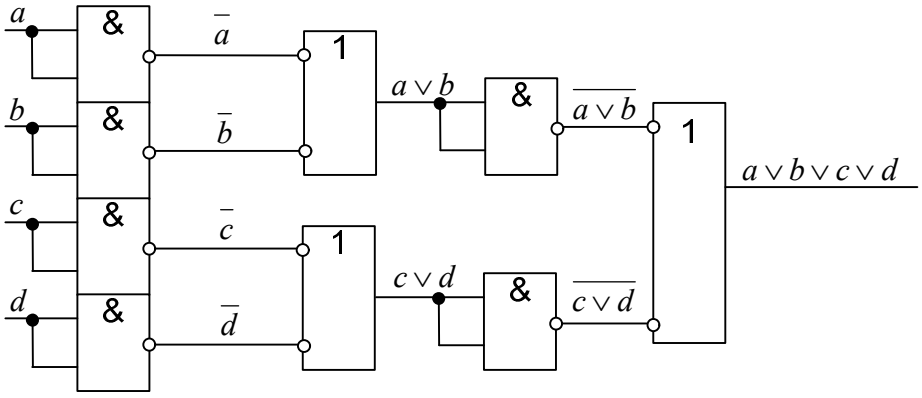


Рис. 3.4.9

Приклад 3.4.1. Реалізувати функцію *f=abc v deg v hij*. Результат реалізації зображено на рис. 3.4.10. Доречно звернути увагу, на те, що на всіх наведених рисунках прямі виходи з'єднуються із прямими входами, а інверсні ─ з інверсними.

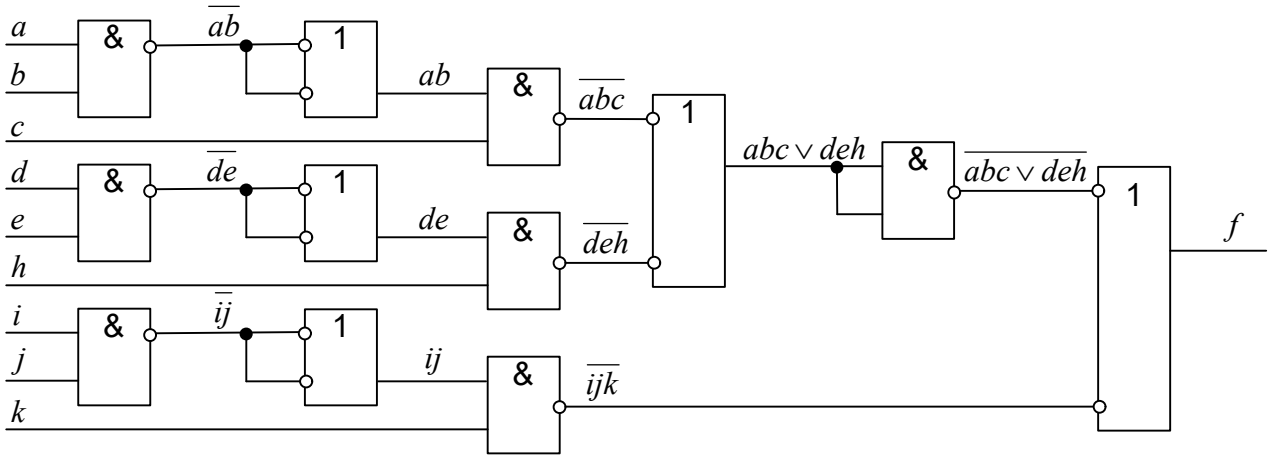
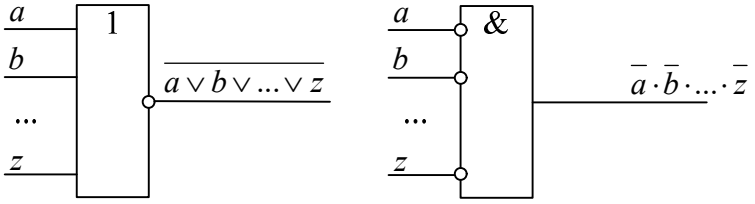


Рис. 3.4.10

3.5 Синтез функцій у монобазисі АБО-НЕ

Елементи монобазиса АБО-НЕ повинні мати кількість входів не менше 2. При одному вході елемент АБО-НЕ перетворюється на інвертор. Відома інша назва цієї функції - заперечення кон'юнкції. Багатовходовий елемент nАБО-НЕ (символ n у назві вказує на кількість входів елемента) реалізує функцію $f = \overline{a \vee b \vee c \vee ... \vee z}$ - функцію АБО-НЕ n змінних. Згідно з правилом Моргана $f = \overline{a \vee b \vee c \vee ... \vee z} = \bar{a}\bar{b}\bar{c}...\bar{z}$. Тобто, можуть існувати 2 абсолютно рівноправні умовні графічні позначення (УГП) цього елемента: перше - як елемента nАБО-НЕ, друге - як елемента НЕ-nІ (рис. 3.5.1).



елемент АБО-НЕ (NOR) елемент НЕ-І

Рис. 3.5.1

На виході f елемента АБО-НЕ буде одиниця тільки тоді, коли на всіх його входах a, b, c, …, z є нулі.

Реалізація функції інверсії у монобазисі АБО-НЕ відбувається відповідно до формули: f = $\overline{a}$ на елементі, який має 1 вхід (на інверторі).

Реалізація функції АБО n змінних у монобазисі nАБО─НЕ відбувається відповідно до формули $f = \overline{\overline{a \vee b \vee c \vee ... \vee z}}$, як послідовність операцій nАБО-НЕ та інверсії. Схемну реалізацію наведено на рис. 3.5.2.

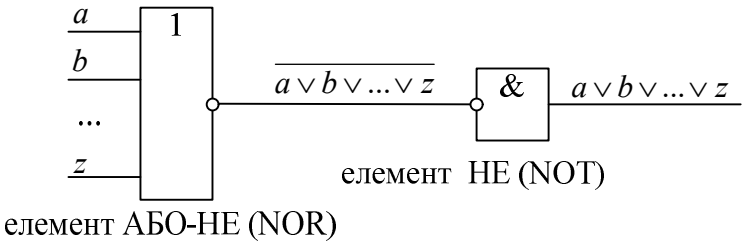


Рис. 3.5.2

Реалізація функції І n змінних у монобазисі nАБО-НЕ відбувається відповідно до формули $f = \overline{a}\,\overline{b}\,\overline{c}...\overline{z}$, як послідовність операцій інверсії кожної змінної та НЕ-nІ. Схемну реалізацію наведено на рис. 3.5.3.

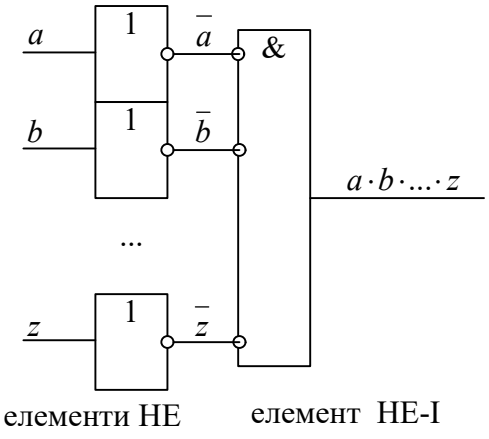


Рис. 3.5.3

Реалізація функції АБО-І в монобазисі АБО-НЕ відбувається як послідовність операцій АБО-НЕ та НЕ-АБО з потрібною кількістю входів.

Приклад 3.5.1. Реалізувати задану у вигляді кон'юнктивної нормальної форми (КНФ) функцію $f = (a \vee b \vee c)(d \vee e \vee h)(i \vee j \vee k)$. Результат реалізації зображено на рис. 3.5.4. Доречно звернути увагу, на те, що на всіх наведених рисунках прямі виходи з'єднуються із прямими входами, а інверсні ─ з інверсними.

3.6 Синтез функцій у монобазисі 2АБО-НЕ (Пірса)

Елементи монобазиса 2АБО-НЕ мають 2 входи. Відомі інші назви функції 2АБО-НЕ: заперечення диз'юнкції, елемент Пірса, стрілка Пірса. Двовходовий елемент 2АБО-НЕ (символ 2 у назві вказує на кількість входів елемента) реалізує функцію $f = \overline{a \vee b}$ - функцію АБО-НЕ двох змінних. Згідно з правилом Моргана $f = \overline{a \vee b} = \overline{a}\,\overline{b}$. Тобто, можуть існувати 2 абсолютно рівноправні умовні графічні позначення (УГП) цього елемента: перше, як елемента 2АБО-НЕ (рис. 3.6.1, а), друге - як елемента НЕ-2І (рис. 3.6.1, б).

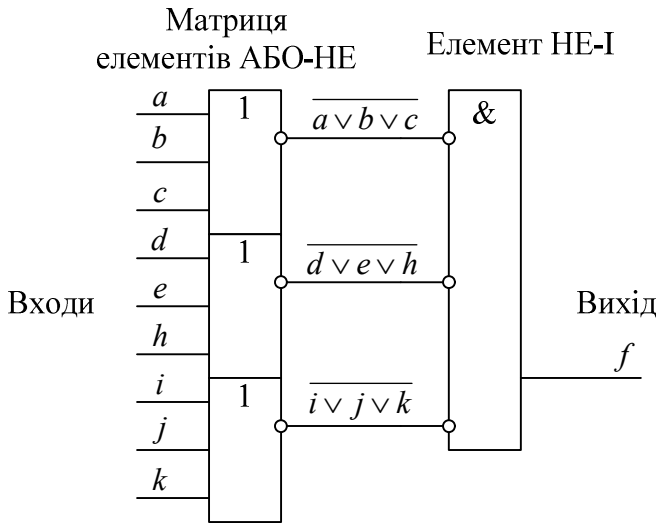


Рис. 3.5.4

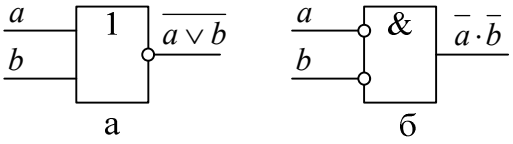


Рис. 3.6.1

Таблиця 3.6.1

Номер набору	Аргументи		Функція
	a	b	АБО-НЕ
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	0

Таблицю істинності елемента 2АБО-НЕ наведено у табл. 3.6.1.

Реалізація функції інверсії в монобазисі 2АБО-НЕ відбувається відповідно до формули: $f = \overline{a \vee a} = \bar{a}$. Схемну реалізацію інверсії наведено на рис. 3.6.2: на обидва входи елемента треба подати сигнал, який потрібно інвертувати.

Реалізація функції АБО в монобазисі 2АБО-НЕ відбувається відповідно до формули $f = \overline{\overline{a \vee b}}$ як послідовність операцій 2АБО-НЕ та інверсії. Схемну реалізацію наведено на рис. 3.6.3.

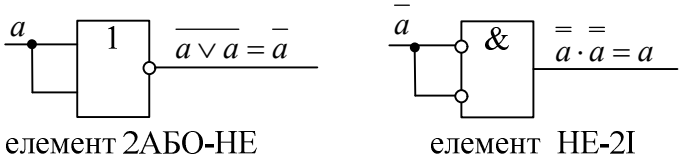
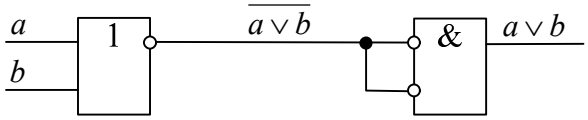


Рис. 3.6.2

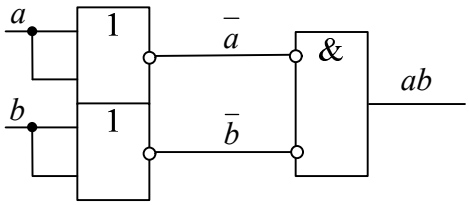
Реалізація функції І в монобазисі 2АБО-НЕ відбувається відповідно до формули $f = \bar{\bar{a}}\,\bar{\bar{b}}$, як послідовність операцій інверсії кожної змінної та НЕ-2І. Схемну реалізацію наведено на рис. 3.6.4.

Реалізація функції АБО-І в монобазисі 2АБО-НЕ відбувається відповідно до формули $f = \overline{\overline{a \vee b}}\,\overline{\overline{c \vee d}}$, як послідовність операцій 2АБО-НЕ та НЕ-2І. Схемну реалізацію наведено на рис. 3.6.5.



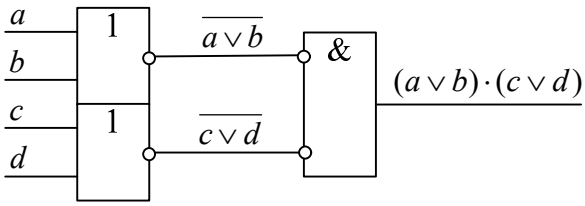
елемент 2АБО-НЕ (NOR) елемент НЕ-2І

Рис. 3.6.3



елементи 2АБО-НЕ елемент НЕ-2І

Рис. 3.6.4



елементи 2АБО-НЕ елемент НЕ-2І

Рис. 3.6.5

Реалізація функції 3АБО-НЕ $f = \overline{a \vee b \vee c}$ каскадна, її зображено на рис. 3.6.6.

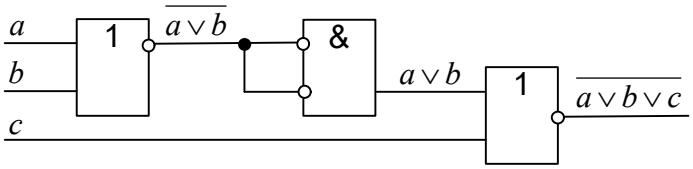


Рис. 3.6.6

Реалізація функції 4АБО-НЕ $f = \overline{a \vee b \vee c \vee d}$ каскадна, її зображено на рис. 3.6.7.

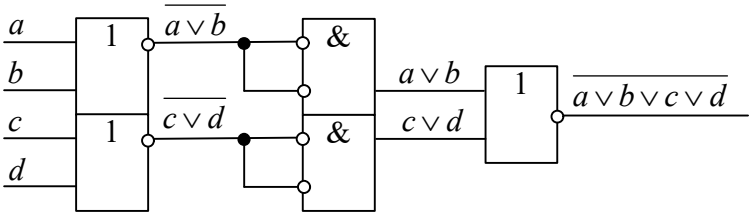


Рис. 3.6.7

Реалізація функції 3І $f = abc$ каскадна, її зображено на рис. 3.6.8.

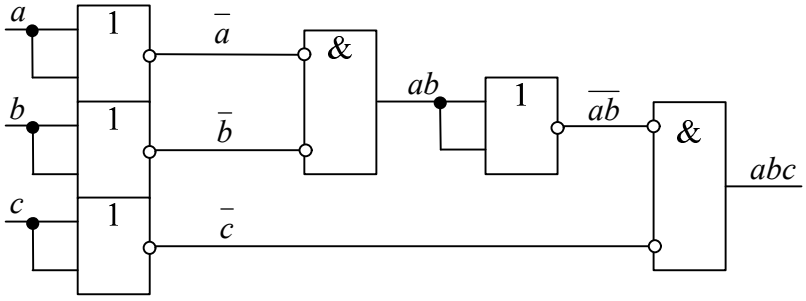


Рис. 3.6.8

Реалізація функції 4І $f = abcd$ каскадна, її зображено на рис. 3.6.9.

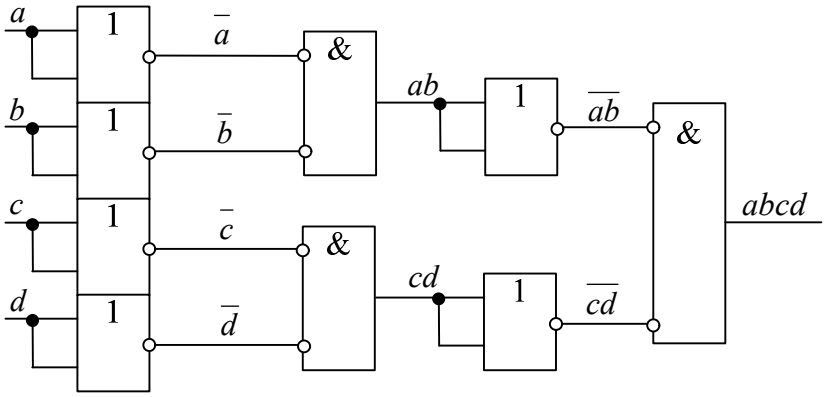


Рис. 3.6.9

Приклад 3.6.1. Реалізувати функцію f = (avbvc)(dvevh)(ivjvk). Результат - на рис. 3.6.10. Доречно звернути увагу, що на всіх наведених рисунках прямі виходи з'єднуються із прямими входами, а інверсні - з інверсними.

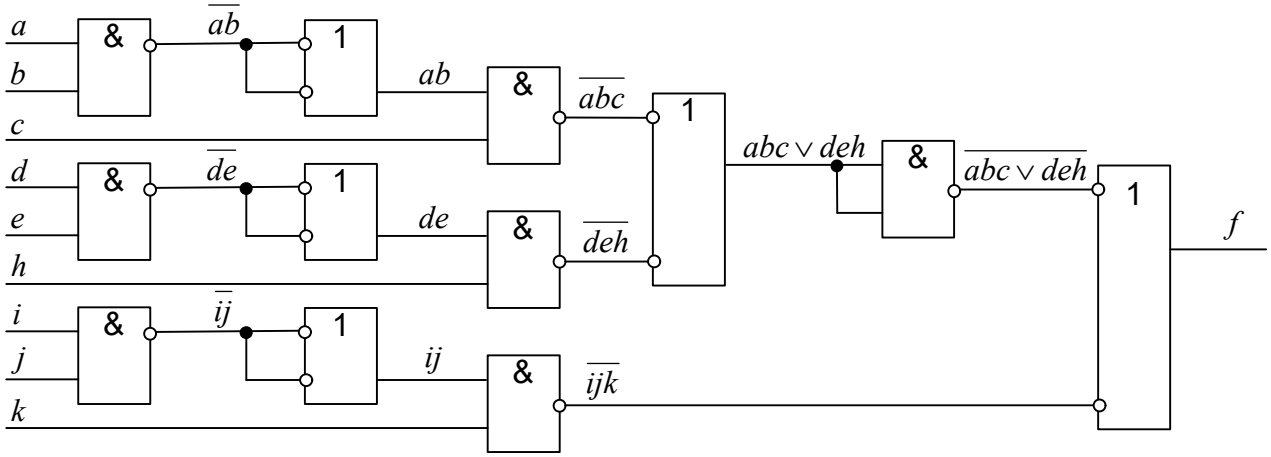


Рис. 3.6.10

Таблиця 3.7.1

N	Входи			ВК	Виходи							
4	4	2	1		7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	0	0	0	0	1
1	0	0	1	1	0	0	0	0	0	0	1	0
2	0	1	0	1	0	0	0	0	0	1	0	0
3	0	1	1	1	0	0	0	0	1	0	0	0
4	1	0	0	1	0	0	0	1	0	0	0	0
5	1	0	1	1	0	0	1	0	0	0	0	0
6	1	1	0	1	0	1	0	0	0	0	0	0
7	1	1	1	1	1	0	0	0	0	0	0	0
X	X	X	X	0	0	0	0	0	0	0	0	0

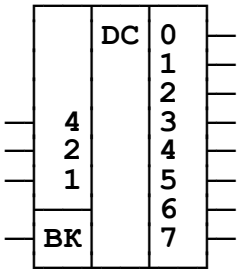


Рис.

3.7 Реалізація функцій алгебри логіки на дешифраторах

Дешифратор - це схема, яка має n інформаційних входів і до 2^n виходів і використовується для перетворення n-розрядного двійкового коду на вході в унітарний код на виході. Унітарний код складається з набору нулів і лише однієї одиниці (або навпаки). В ЕОМ дешифратори найчастіше використовуються для перетворення n-розрядного двійкового коду в унітарний код на 2^n виходах (кожному n-розрядному двійковому коду на вході відповідає сигнал 1 тільки на одному з 2^n виходів). Для прикладу наводиться таблиця істинності (табл. 3.7.1) дешифратора на 3 входи і його умовне графічне позначення (рис. 3.7.1). Інформаційні входи дешифратора позначаються вагою розрядів двійкового коду. Виходи дешифратора позначаються порядковими номерами, які відповідають номеру

набору на інформаційних входах (N у табл. 3.7.1). Виходи дешифраторів можуть бути інверсними. Додатковий вхід управління ВК (вибір кристалу) дозволяє або забороняє формування сигналів на виходах дешифратора.

Таблиця 3.7.2

Входи	Вихід
abc	f
000	1
001	0
010	1
011	1
100	0
101	0
110	1
111	1

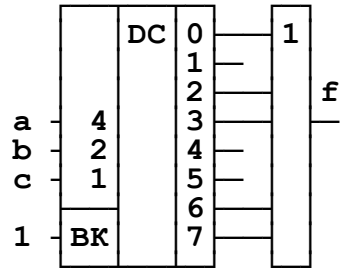


Рис. 3.7.2

Описані вище дешифратори називаються повними, оскільки кількість їхніх виходів дорівнює 2^n. Дешифратори, які мають меншу кількість виходів, називаються неповними.

На дешифраторах зручно реалізовувати ДДНФ, оскільки ДДНФ безпосередньо вказує на ті набори з 2^n, на яких функція приймає значення 1. Для схемної реалізації такої функції необхідно на інформаційні входи дешифратора подати вхідні змінні і з'єднати виходи дешифратора, які відповідають номерам набору, на яких функція приймає значення 1, з входами додаткового елемента АБО. На виході елемента АБО буде сформована потрібна функція.

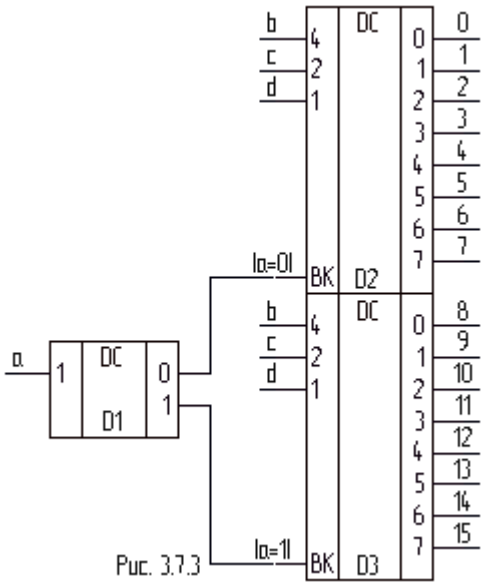


Рис. 3.7.3

Приклад 3.7.1. Реалізувати функцію, яка задана табл. 3.7.2.

Схему наведено на рис. 3.7.2.

Розширення по входах схем, які реалізовані на дешифраторах.

Якщо потрібно дешифрувати більше сигналів, ніж є інформаційних входів у дешифратора, то необхідно використовувати вхід управління ВК і заводити на нього результат дешифрації тих розрядів, які безпосередньо не подані на інформапційні входи дешифратора. Приклад дешифрації 4 розрядів на дешифраторах з трьома інформаційними входами наведений на рис. 3.7.3.

Коли сигнал a=0 - працює верхній дешифратор D2, на його виходах формуються сигнали які відповідають наборам 0..7 вхідних змінних a, b, c, d. Коли a=1 - працює нижній дешифратор D3, на його виходах формуються сигнали які відповідають шістнадцятковим наборам 8..f вхідних змінних a, b, c, d. Таким чином, вхід ВК має найбільшу вагу, в наведеному прикладі вага входа ВК дорівнює 8. Окремо взяті дешифратори D2 і D3 в наведеному прикладі неповні.

Таблиця 3.8.1

Входи	На виході f сигнал з інформаційного
4 2 1	
0 0 0	входу 0
0 0 1	входу 1
0 1 0	входу 2
0 1 1	входу 3
1 0 0	входу 4
1 0 1	входу 5
1 1 0	входу 6
1 1 1	входу 7

3.8 Синтез комбінаційних схем на базі мультиплексорів

Мультиплексор - це комбінаційна багатовходова схема з одним виходом f. Входи мультиплексора поділяються на інформаційні (k входів) та управління

(n входів). Загалом, $k \le 2^n$. Код, який надходить на входи управління, визначає один з інформаційних входів, сигнал з якого передається на вихід f. У табл. 3.8.1 показано таблицю істинності мультиплексора з трьома входами управління і 8 інформаційними входами, на рис. 3.8.1 показане умовне графічне позначення такого мультиплексора. Входи управління мультиплексора позначаються вагою розрядів двійкового коду. Інформаційні входи мультиплексора позначаються порядковими номерами, які відповідають номеру набору на вхід управління.

Таблиця 3.8.2

Сигнали (входи)	Вихід
a b c (4 2 1)	f
0 0 0	1
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	0
1 1 0	1
1 1 1	1

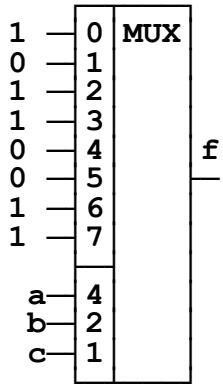


Рис. 3.8.1

На мультиплексорах зручно реалізовувати ДДНФ, оскільки ДДНФ безпосередньо вказує на ті набори з 2^n, на яких функція приймає значення 1. Для схемної реалізації такої функції необхідно подати на входи управління мультиплексора змінні-аргументи у відповідності з їхнею вагою і з'єднати входи мультиплексора, номери яких відповідають наборам, на яких функція приймає значення 1, з логічною 1. На решту входів треба подати логічний 0. На виході мультиплексора буде сформована потрібна функція.

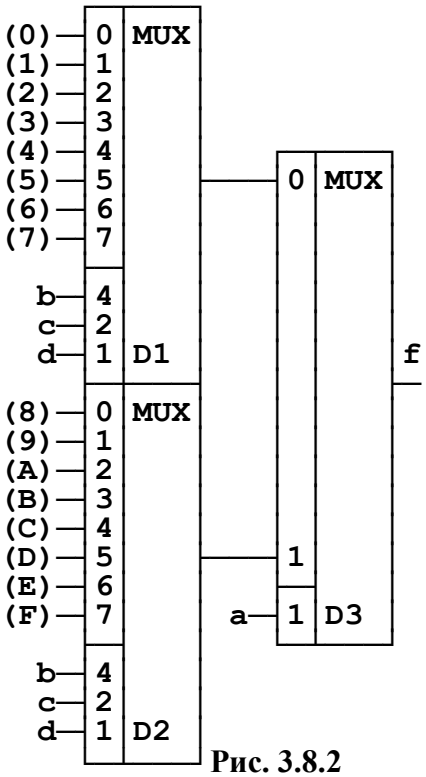


Рис. 3.8.2

Приклад 3.8.1. Реалізувати функцію, яка задана табл. 3.8.2.

Схему наведено на рис. 3.8.1.

Розширення по входах схем, які реалізовані на мультиплексорах.

Коли кількість змінних перевищує кількість входів управління мультиплексора, необхідно використовувати каскадне з'єднання мультиплексорів.

Приклад мультиплексора на 16 входів, який побудований на базі 8-входового і 2-входового мультиплексорів наведений на рис. 3.8.2.

Коли a=0 на вихід 2-входового мультиплексора D3 проходить сигнал з верхнього мультиплексора D1, тобто, з входів 0...7, які подані в дужках.

Коли a=1 на вихід 2-входового мультиплексора D3 проходить сигнал з нижнього мультиплексора D2, тобто, з входів 8...F, які подані в дужках.

3.9 Синтез комбінаційних схем базі постійних запам'ятовуючих пристроїв

Постійний запам'ятовуючий пристрій (ПЗП) - це комбінаційна багатовходова схема з одним або кількома виходами f_0, ..., f_{k-1}. На входи подаються набори, які називаються адресами і позначаються A_0, A_1, …, A_{n-1}, а з виходів знімаються набори, які називаються даними і позначаються D_0, D_1, ..., D_{k-1}. Кожній адресі відповідають свої дані, які записані в ПЗП або в процесі виготовлення, або користувачем перед встановленням на плату (комірку) чи вже на самій платі. Для занесення інформації в ПЗП необхідно скласти таблицю

Таблиця 3.9.1

Вхідні сигнали (Входи ПЗП)	Вихідні сигнали (Виходи ПЗП)
a b c (A2 A1 A0)	f_0 f_1 f_2 (D0 D1 D2)
0 0 0	1 0 0
0 0 1	0 1 0
0 1 0	1 1 1
0 1 1	1 0 0
1 0 0	0 0 1
1 0 1	0 1 1
1 1 0	1 0 1
1 1 1	1 0 1

прошиття, яка встановлює відповідність між адресами і даними. Занесення інформації в ПЗП здійснюється користувачем за допомогою пристрою, який називається програматором.

Якщо кількість розрядів адреси дорівнює n, то в ПЗП зберігається 2^n наборів даних (слів) розрядністю k, де k - кількість виходів ПЗП, тобто, об'єм ПЗП дорівнює $2^n \cdot k$ біт.

На ПЗП зручно реалізовувати ДДНФ набору функцій, оскільки ДДНФ безпосередньо вказує на ті набори з 2^n, на яких функція приймає значення 1. Для реалізації таких функцій необхідно завести на входи ПЗП усі змінні, з яких формуються функції, кожній з функцій поставити у відповідність один з виходів ПЗП і скласти таблицю прошиття.

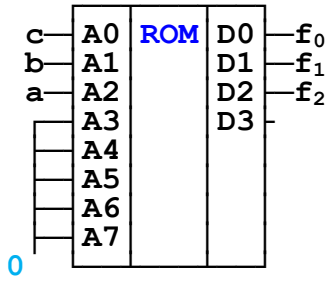


Рис. 3.9.1

Приклад 3.9.1. Реалізувати за допомогою ПЗП об'ємом 256x4 функції, які задані табл. 3.9.1.

Схему на базі ПЗП наведено на рис. 3.9.1. Кількість входів адреси ПЗП дорівнює 8 (оскільки $256 = 2^8$). Кількість виходів ПЗП - 4.

Таблиця 3.9.2

Адреси в кодах				Дані в кодах			
двійковому			16-ковому	двійковому			16-ковому
A2	A1	A0		D0	D1	D2	
a	b	c		f_0	f_1	f_2	
0	0	0	0	1	0	0	04
0	0	1	1	0	1	0	02
0	1	0	2	1	1	1	07
0	1	1	3	1	0	0	04
1	0	0	4	0	0	1	01
1	0	1	5	0	1	1	03
1	1	0	6	1	0	1	05
1	1	1	7	1	0	1	05

Розряди адрес A3...A7 і даних D3 - не використовуються. Таблиця прошиття має вигляд табл. 3.9.2.

У даному випадку використовується тільки 8 (0...7) з 256 можливих адрес (3 розряди адресних входів з 7) і тільки 3 розряди даних з 4.

Розширення по входах схем, реалізованих на ПЗП, здійснюється, коли кількість вхідних змінних перевищує кількість адресних входів ПЗП. Розширення виконується з використанням входів вибору кристалу (ВК) так само, як розширення по входах дешифраторів. При цьому об'єднання виходів ПЗП здійснюється за допомогою монтажного АБО.

3.10 Синтез комбінаційних схем на базі програмованих логічних матриць

Програмована логічна матриця (ПЛМ, PLA - Programmable Logic Array) - це комбінаційна багатовходова схема з одним або кількома виходами. На входи подаються набори, які називаються адресами, а з виходів знімаються набори, які називаються даними. Кожній адресі відповідають свої дані, які записані в ПЛМ або в процесі виготовлення, або користувачем перед встановленням на плату (комірку) чи вже на самій платі. Для занесення інформації в ПЛМ необхідно скласти таблицю прошиття, яка встановлює відповідність між

адресами і даними. Занесення користувачем інформації в ПЛМ здійснюється за допомогою пристрою, який називається програматором.

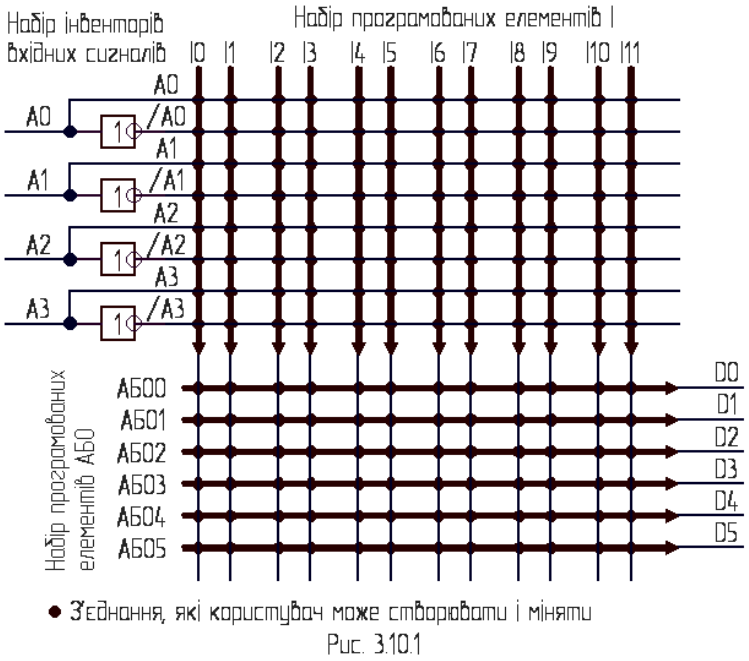


Рис. 3.10.1

На відміну від ПЗП, у ПЛМ зберігається тільки частина з 2^n наборів даних (слів) розрядністю k, де n - кількість адресних входів, k - кількість виходів ПЛМ. Яка саме частина - визначається з таблиці прошиття.

ПЛМ типу PLA (Programmable Logic Array) має структуру (матричну схему), подібну до показаної на рис. 3.10.1. ПЛМ складається з трьох основних частин:

набору (матриці) інверторів вхідних сигналів;

набору (матриці) програмованих елементів І (на рис. 3.10.1 12 елементів І0...І11 умовно показані вертикальними стрілками, місця перетину стрілок і ліній вхідних сигналів є входами елементів І, продовженням стрілки є зв'язок, на якому формується вихідний сигнал даного елемента І);

набору (матриці) програмованих елементів АБО (на рис. 3.10.1 6 елементів АБО0…АБО5 умовно показані горизонтальними стрілками, місця перетину стрілок і ліній вихідних сигналів елементів І є входами елементів АБО, продовженням стрілки є зв'язок, на якому формується вихідний сигнал даного елемента АБО).

Умовне графічне позначення ПЛМ (рис. 3.10.1) з вхідними і вихідними сигналами прикладу 3.10.1 наведене на рис. 3.10.2.

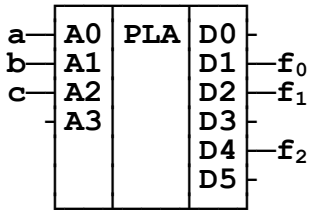


Рис. 3.10.2

Кількість елементів у матриці І залежить від типу мікросхеми (може бути 48, 64 і т.д.). На кожний елемент І заведені сигнали з кожного адресного входу в прямому і інверсному коді, тобто, входів у елемента І вдвічі більше, ніж входів у ПЛМ.

Кількість елементів у матриці АБО залежить від кількості виходів ПЛМ (може бути 8, 14, 16 і т.д.). У ПЛМ типу PLA на кожний елемент АБО заведені виходи всіх елементів І. ПЛМ (рис. 3.10.1) має 4 адресних входи, 12 елементів І, 6 елементи АБО, 6 виходів.

Під час програмування ПЛМ користувач має можливість забирати зайві сигнали з входів елементів І та АБО.

На ПЛМ зручно реалізовувати мінімізовані ДНФ набору функцій. Для їхньої реалізації необхідно завести на входи ПЛМ всі змінні, з яких формуються функції, кожній з функцій поставити у відповідність один з виходів ПЛМ і скласти таблицю прошиття.

Приклад 3.10.1. На ПЛМ із структурою рис. 3.10.1 реалізувати функції

$$f_0 = bc \vee \bar{a}\bar{c} \vee a\bar{b};$$

$f_1 = a\bar{b} \vee \bar{a}b\bar{c}$;

$f_2 = c \vee \bar{a}b \vee a\bar{b}$.

Схему використання запрограмованої ПЛМ наведено на рис. 3.10.2, а її внутрішня структура (матрична схема) - на рис. 3.10.3. Які саме елементи І та АБО будуть використовуватися визначає користувач. Під'єднання незадіяних елементів не показані.

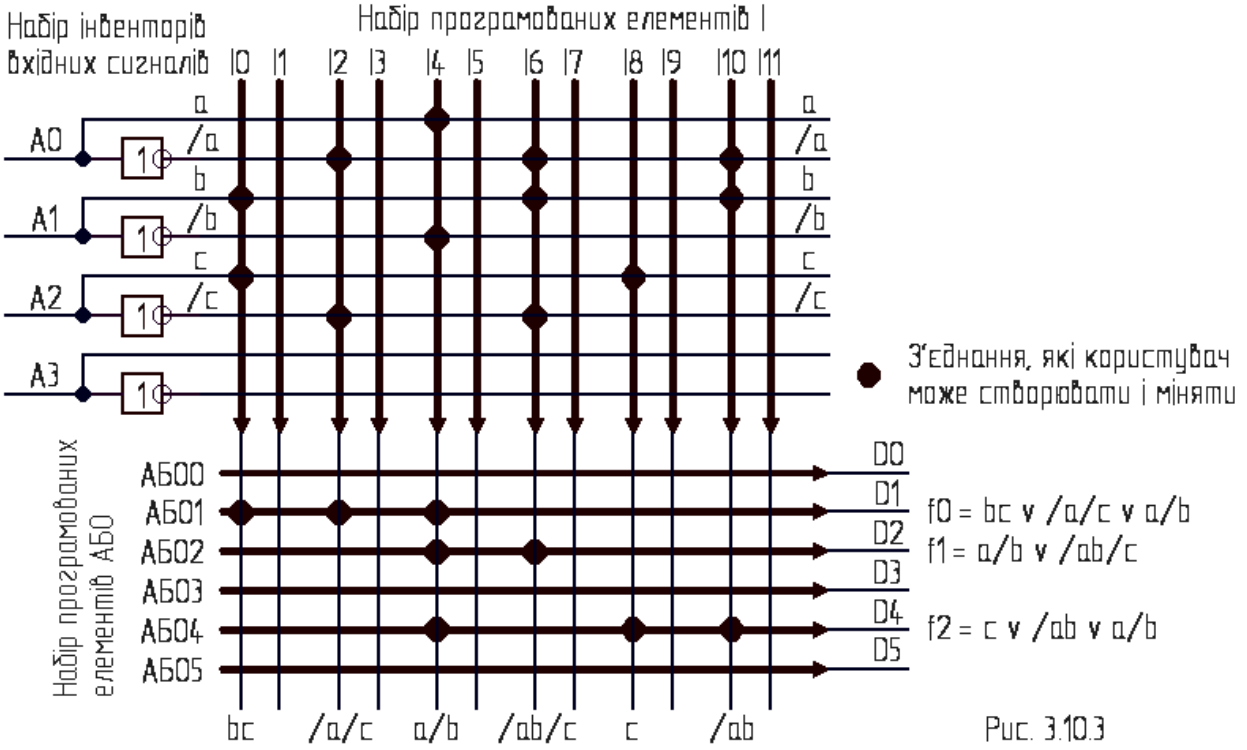


Рис. 3.10.3

Таблиця 3.10.2

N	Входи A3 A1 A2 A0	Виходи D0 D2 D4 D1 D3 D5
	c b a	f_0 f_1 f_2
I0	- H H -	- A - - - -
I2	- L - L	- A - - - -
I4	- - L H	- A A - A -
I6	- L H L	- - A - - -
I8	- H - -	- - - - A -
I10	- - H L	- - - - A -
I1	0 0 0 0	A A A A A A
I3	0 0 0 0	A A A A A A
I5	0 0 0 0	A A A A A A
I7	0 0 0 0	A A A A A A
I9	0 0 0 0	A A A A A A
I11	0 0 0 0	A A A A A A

Терм bc реалізовано на елементі I0;

терм $\bar{a}\bar{c}$ реалізовано на елементі I2;

терм $a\bar{b}$ реалізовано на елементі I4 один раз для всіх функцій;

терм $\bar{a}b\bar{c}$ реалізовано на елементі I6;

терм c реалізовано на елементі I8;

терм $\bar{a}b$ реалізовано на елементі I10;

функція f_0 реалізована на елементі АБО1 (формується на виході D1);

функція f_1 реалізована на елементі АБО2 (формується на виході D2);;

функція f_2 реалізована на елементі АБО4 (формується на виході D4).

Розряд адреси A3 і даних D0, D3, D5 - не використовуються. Таблиця прошиття має вигляд табл. 3.10.2. У даному випадку використовуються 6 (I0, I2, I4, I6, I8, I10) елементів І (кон'юнкторів) з 12, 3 розряди адресних входів з 4 і 3 розряди даних з 6.

У табл. 3.10.2 прийняті такі позначення:

N - позначення елемента І.

На перетині рядка з позначенням елемента І та графи з номером входу можуть стояти такі символи:

«Н» - сигнал з даного входу заведений на відповідний елемент І в прямому коді;

«L» - сигнал з даного входу заведений на відповідний елемент І в інверсному коді;

«-» - сигнал з даного входу від'єднаний від відповідного елемента І;

0 - сигнал з даного входу заведений на відповідний елемент І одночасно в прямому та інверсному кодах (це початковий стан ПЛМ, даний елемент І не використовується).

На перетині рядка з номером елемента І та графи з номером виходу (тобто, номером елемента АБО) можуть стояти такі символи:

«А» - сигнал з даного елемента І заведений на вхід відповідного елемента АБО і проходить через елемент АБО на відповідний вихід ПЛМ (це початковий стан незапрограмованої ПЛМ);

«-» - сигнал з даного елемента І від'єднаний від входу відповідного елемента АБО і не проходить на відповідний вихід ПЛМ.

Літери А, які стоять в одній графі, вказують на елементи І, які об'єднуються по АБО.

Розширення по входах схем, які реалізовані на ПЛМ, здійснюється, коли кількість вхідних змінних перевищує кількість адресних входів ПЛМ. Розширення виконується так само, як розширення по входах ПЗП з використанням сигналу «Вибір кристалу» (ВК) ПЛМ. ПЛМ у цьому випадку повинна мати додатковий вхід ВК.

3.11 Синтез комбінаційних схем на базі програмованих матриць логіки

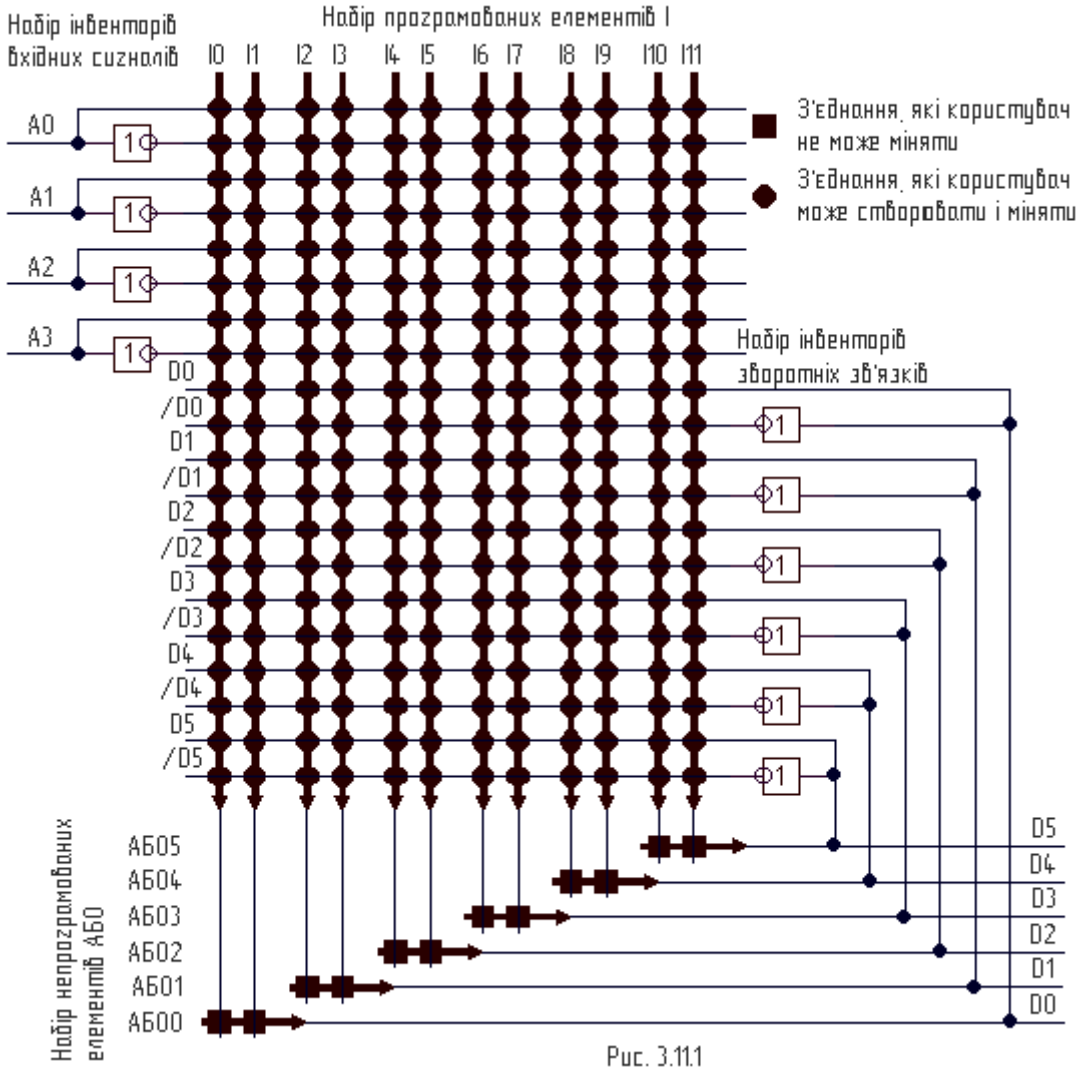


Рис. 3.11.1

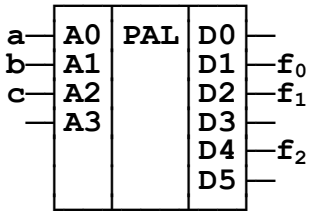


Рис. 3.11.2

На відміну від програмованих логічних матриць (ПЛМ, PLA) у програмованих матриць логіки (ПМЛ, PAL - Programmable Array Logic) на кожний елемент АБО заведено виходи не всіх елементів І. Найчастіше елементи І розділені нарівно між елементами АБО, входи елементів АБО запрограмовані в процесі виробництва. ПМЛ типу PAL має структуру (матричну схему), подібну до поданої на рис. 3.11.1.

ПМЛ має 4 адресних входи (A0...A3), 12 елементів І на 8 входів кожний, 6 елементів АБО на 2 входи кожний, 6 зворотних зв'язків (D0...D5). Як видно, програмується тільки матриця елементів І.

ПМЛ складається з чотирьох основних частин:

набору (матриці) інверторів вхідних сигналів;

набору (матриці) програмованих елементів І (на рис. 3.11.1 12 елементів I0...I11 умовно показані вертикальними стрілками, місця перетину стрілок і ліній вхідних сигналів є входами елементів І, продовженням стрілки є зв’язок, на якому формується вихідний сигнал даного елемента І);

набору (матриці) непрограмованих елементів АБО (на рис. 3.11.1 6 елементів АБО0…АБО5 умовно показані горизонтальними стрілками, місця перетину стрілок і ліній вихідних сигналів елементів І є входами елементів АБО, продовженням стрілки є зв'язок, на якому формується вихідний сигнал даного елемента АБО);

набору (матриці) інверторів зворотніх зв'язків.

Умовне графічне позначення ПМЛ (рис. 3.11.1) з вхідними і вихідними сигналами прикладу 3.11.1 наведене на рис. 3.11.2.

Кількість елементів у матриці І залежить від типу мікросхеми (може бути 48, 64 і т.д.). На кожний елемент І заведені сигнали з кожного адресного входу в прямому і інверсному коді, тобто, входів у елемента І вдвічі більше, ніж входів у ПЛМ.

Кількість елементів у матриці АБО залежить від кількості виходів ПЛМ (може бути 8, 14, 16 і т.д.). У ПМЛ типу PAL на кожний елемент АБО заведені виходи тільки деяких елементів І. ПМЛ (рис. 3.11.1) має 4 адресних входи, 12 елементів І, 6 елементи АБО, 6 виходів. Елементи І розподілені по 2 на кожний елемент АБО.

Під час програмування ПЛМ користувач має можливість забирати зайві сигнали тільки з входів елементів І.

На ПМЛ зручно реалізовувати мінімізовані за 1 ДНФ набору функцій. Для їхньої реалізації необхідно завести на входи ПЛМ усі змінні, з яких формуються функції, кожній з функцій поставити у відповідність один з виходів ПЛМ і скласти таблицю прошиття.

Приклад 3.11.1. На ПМЛ із структурою рис. 3.11.1 реалізувати функції

$f_0 = bc \vee \bar{a}\bar{c} \vee a\bar{b}$;

$f_1 = a\bar{b} \vee \bar{a}b\bar{c}$;

$f_2 = c \vee \bar{a}b \vee a\bar{b}$.

Схему використання запрограмованої ПЛМ наведено на рис. 3.11.2, а її внутрішня структура (матрична схема) - на рис. 3.11.3. Які саме елементи І та АБО будуть використовуватися визначає користувач. Під'єднання незадіяних елементів не показані.

Розряд адреси А3 і даних D5 - не задіяні.

Таблиця прошиття має вигляд табл. 3.11.1.

Терм bc реалізовано на елементі І0;

терм $\bar{a}\bar{c}$ реалізовано на елементі І1;

терм $a\bar{b}$ реалізовано три рази для кожної з функцій окремо:

на елементі І2 – для функції f_0;

Таблиця 3.11.1

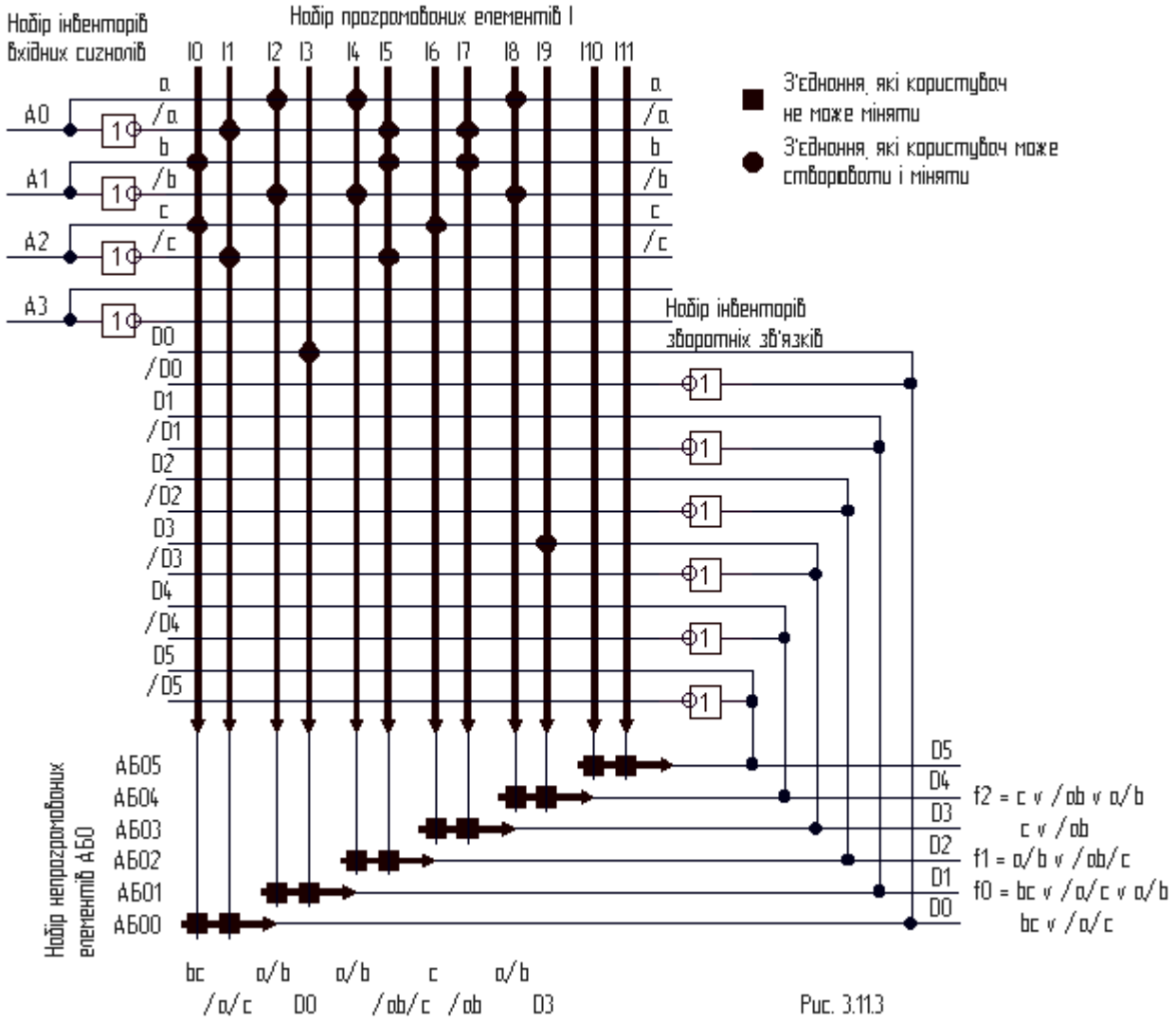


Рис. 3.11.3

на елементі І4 – для функції f_1;

на елементі І8 – для функції f_2;

терм $\bar{a}bc$ реалізовано на елементі І5;

терм c реалізовано на елементі І6;

терм $\bar{a}b$ реалізовано на елементі І7;

функція f_0 реалізована на елементах АБО0 і АБО1 (формується на виході D1);

функція f_1 реалізована на елементі АБО2 (формується на виході D2);

функція f_2 реалізована на елементах АБО3 і АБО4 (формується на виході D4).

У даному випадку використовується тільки 10 елементів І0…І9 з 12, 5 елементів АБО0...АБО4 з 6, 3 розряди адресних входів А0…А3 з 4, 5 розрядів даних D0…D4 з 6 і два зворотніх зв'язки:

D0 – використовуэться при формуванні функції f_0;

D3 – використовуэться при формуванні функції f_2.

Табл. 3.11.1 має такі позначення:

N – позначення елемента І;

на перетині рядка з позначенням елемента І та граф з номером входу і номеру зворотнього зв'язку можуть стояти такі символи:

«Н» - сигнал з даного входу заведений на відповідний елемент І в прямому коді;

«L» - сигнал з даного входу заведений на відповідний елемент І в інверсному коді;

«-» - сигнал з даного входу від'єднаний від відповідного елемента І;

0 - сигнал з даного входу заведений на відповідний елемент І одночасно в прямому та інверсному кодах (початковий стан ПЛМ, даний елемент І не використовується).

Розширення по входах схем, які реалізовані на ПМЛ, здійснюється, коли кількість вхідних змінних перевищує кількість адресних входів ПЛМ. Розширення виконується так само, як розширення по входах ПЗП з використанням сигналу «Вибір кристалу» (ВК) ПМЛ. ПМЛ у цьому випадку повинна мати додатковий вхід ВК.

Таблиця 3.12.1

Адреса у 16-ковому коді	Розряди адреси							
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
b8	1	0	1	1	1	0	0	0
b9	1	0	1	1	1	0	0	1
ba	1	0	1	1	1	0	1	0
bb	1	0	1	1	1	0	1	1
bc	1	0	1	1	1	1	0	0
bd	1	0	1	1	1	1	0	1
be	1	0	1	1	1	1	1	0
bf	1	0	1	1	1	1	1	1
b8…bf	1	0	1	1	1	-	-	-

3.12 Синтез дешифратора діапазону адрес на ПЛМ

Задача синтезу дешифратора діапазону адреси є окремим випадком задачі мінімізації функції багатьох змінних. Розв'язок полегшується тим, що набори, які підлягають мінімізації, розташовані послідовно і їхні шістнадцяткові коди відрізняються один від одного на +1. Мінімізація в цьому випадку ґрунтується на поділі розрядів групи наборів двійкових кодів (адрес) на 2 частини, так щоб старші розряди кожного набору з групи залишалися незмінними, а молодші - пробігали всі значення від 0...0 до 1...1, тоді до мінімізованого виразу групи наборів увійдуть тільки незмінні старші розряди.

Приклад 3.12.1. Знайти вираз для групи адрес b8…bf (табл. 3.12.1).

У загальному вигляді - це задача мінімізації функції 8 змінних A_0, A_1,..., A_7.

Але розряди адрес заданих наборів можна розбити на дві групи: незмінна частина - розряди A_3...A_7; змінна частина - розряди A_0, A_1, A_2.

Розряди змінної групи перебігають всі можливі значення від 000 до 111. Тому їх можна спростити і записати загальний вираз для усіх заданих у табл. 3.12.1 наборів, тобто, для наведеного діапазону адрес Diap:

$$Diap = A_7 \overline{A_6} A_5 A_4 A_3 .$$

У більш складних випадках описаний принцип використовується послідовно.

Приклад 3.12.2. Скласти таблицю прошиття ПЛМ з 16 входами адреси і входом вибору кристалу для діапазону адрес 04B8F...1E3A4, адреси наведено в 16-ковому коді (1E3A4 - прочитана справа наліво в двійковому коді адреса 04B8F). При цьому 1 на вході вибору кристалу примушує ПЛМ працювати згідно з таблицею прошиття, а 0 – примушує видавати на всі виходи 0.

Таблиця 3.12.2

N	Входи А ПЛМ 15 13 11 09 07 05 03 01 14 12 10 08 06 04 02 00	Виходи 0 1 2 3	Диапазон кодів	
	A_{15} A_{13} A_{11} A_9 A_7 A_5 A_3 A_1 A_{14} A_{12} A_{10} A_8 A_6 A_4 A_2 A_0	M	від	до
I0	L H L L H L H H H L L L H H H H	A - - -	4B8F	4B8F
I1	L H L L H L H H H L L H - - - -	A - - -	4B90	4B9F
I2	L H L L H L H H H L H - - - - -	A - - -	4BA0	4BBF
I3	L H L L H L H H H H - - - - - -	A - - -	4BC0	4BFF
I4	L H L L H H - - - - - - - - - -	A - - -	4C00	4FFF
I5	L H L H - - - - - - - - - - - -	A - - -	5000	5FFF
I6	L H H - - - - - - - - - - - - -	A - - -	6000	7FFF
I7	H - - - - - - - - - - - - - - -	A - - -	8000	FFFF

Оскільки дані адреси для свого представлення вимагають 17 біт, а ПЛМ має всього 16 входів, то доведеться робити розширення ПЛМ по входах і використати для синтезу дві мікросхеми ПЛМ D1 і D2 (рис. 3.12.1): на D2 буде створений дешифратор адрес, у яких старший двійковий розряд дорівнює 0, а на D1 - адрес, у яких він дорівнює 1. Тобто, на мікросхемі ПЛМ D2 створюємо дешифратор для діапазону адрес (0)4B8F...(0)FFFF (сигнал M). Таблицю прошиття цієї мікросхеми наведено в табл. 3.12.2. Старший 17-ий розряд A16 (у дужках) на адресні входи ПЛМ не подається і в табл. 3.12.2 не відображається.

Кон'юнктор I0 кодує початкову адресу діапазону - 4B8F. Оскільки вона непарна, то в даному випадку мінімізації не підлягає.

Додаємо до неї 1 і отримуємо наступну адресу - 4B90. Оскільки вона закінчується чотирма двійковими нулями, то можна ці чотири розряди виділити в змінну групу і мінімізувати, що і зроблено в рядку I1.

Остання адреса, яку кодує рядок I1, дорівнює 4B9F. Вона менша за верхню границю діапазону (FFFF), і тому додаємо до неї 1 і отримуємо наступну адресу - 4BA0. Так як вона закінчується п'ятьма двійковими 0, то можна ці п'ять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I2.

Остання адреса, яку кодує рядок I2, дорівнює 4BBF. Вона менша за верхню границю діапазону, тому додаємо до неї 1 і отримуємо наступну адресу - 4BC0. Оскільки вона закінчується шістьма двійковими нулями, то можна ці шість розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I3.

Остання адреса, яку кодує рядок I3, дорівнює 4BFF. Вона менша за верхню границю діапазону, тому додаємо до неї 1 і отримуємо наступну адресу - 4C00. Оскільки вона закінчується десятьма двійковими нулями, то можна ці десять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I4.

Остання адреса, яку кодує рядок I4, дорівнює 4FFF. Вона менша за верхню границю діапазону, тому додаємо до неї 1 і отримуємо наступну адресу - 5000. Оскільки вона закінчується дванадцятьма двійковими нулями, то можна ці дванадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено у рядку I5.

Остання адреса, яку кодує рядок I5, дорівнює 5FFF. Вона менша за верхню границю діапазону, тому додаємо до неї 1 і отримуємо наступну адресу - 6000. Оскільки вона закінчується тринадцятьма двійковими нулями, то можна ці тринадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено у рядку I6.

Остання адреса, яку кодує рядок I6, дорівнює 7FFF. Вона менша за верхню границю діапазону, тому додаємо до неї 1 і отримуємо наступну адресу - 8000. Оскільки вона закінчується п'ятнадцятьма двійковими нулями, то можна ці п'ятнадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I7.

Остання адреса, яку кодує рядок I7, дорівнює FFFF. Вона дорівнює верхній границі діапазону, тому мінімізацію на цьому закінчуємо.

Подібним чином створюється таблиця прошиття для мікросхеми ПЛМ D1. Для неї створюємо дешифратор для діапазону адрес (1)0000...(1)E3A4 (сигнал S). Таблицю прошиття цієї мікросхеми наведено в табл. 3.12.3. Старший 17-ий розряд A16 (у дужках) на адресні входи ПЛМ не подається і у табл. 3.12.3 не відображається.

Таблиця 3.12.3

N	Входи А ПЛМ 15 13 11 09 07 05 03 01 14 12 10 08 06 04 02 00	Виходи 0 1 2 3	Діапазон кодів	
	A_{15} A_{13} A_{11} A_9 A_7 A_5 A_3 A_1 A_{14} A_{12} A_{10} A_8 A_6 A_4 A_2 A_0	S	від	до
I0	H H H L L L H H H L H L L H L L	A - - -	E3A4	E3A4
I1	H H H L L L H H H L H L L L - -	A - - -	E3A0	E3A3
I2	H H H L L L H H H L L - - - - -	A - - -	E380	E39F
I3	H H H L L L H H L - - - - - - -	A - - -	E300	E37F
I4	H H H L L L H L - - - - - - - -	A - - -	E200	E2FF
I5	H H H L L L L - - - - - - - - -	A - - -	E000	E1FF
I6	H H L - - - - - - - - - - - - -	A - - -	C000	DFFF
I7	H L - - - - - - - - - - - - - -	A - - -	8000	BFFF
I8	L - - - - - - - - - - - - - - -	A - - -	0000	7FFF

На відміну від попередньої таблиці рухатися будемо в зворотному напрямі: від верхньої границі до середньої точки.

Кон'юнктор I0 кодує кінцеву адресу діапазону - E3A4. Оскільки вона парна, то в даному випадку мінімізації не підлягає. Віднімаємо від неї 1 і отримуємо попередню адресу - E3A3. Оскільки вона закінчується двома двійковими одиницями, то можна ці два розряди виділити у змінну групу і мінімізувати, як це зроблено в рядку I1.

Перша адреса, яку кодує рядок I1, дорівнює E3A0. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - E39F. Оскільки вона закінчується п'ятьма двійковими 1, то можна ці п'ять розрядів виділити у змінну групу і мінімізувати, як це зроблено у рядку I2.

Перша адреса, яку кодує рядок I2, дорівнює E380. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - E37F. Оскільки вона закінчується сімома двійковими 1, то можна ці сім розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I3.

Перша адреса, яку кодує рядок I3, дорівнює E300. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - E2FF. Оскільки вона закінчується вісьмома двійковими 1, то можна ці вісім розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I4.

Перша адреса, яку кодує рядок I4, дорівнює E200. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - E1FF.

Оскільки вона закінчується дев'ятьма двійковими 1, то можна ці дев'ять розрядів виділити у змінну групу і мінімізувати, як це зроблено у рядку I5.

Перша адреса, яку кодує рядок I5, дорівнює E000. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - DFFF. Оскільки вона закінчується тринадцятьма двійковими 1, то можна ці тринадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I6.

Перша адреса, яку кодує рядок I6, дорівнює C000. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - BFFF. Оскільки вона закінчується чотирнадцятьма двійковими 1, то можна ці чотирнадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I7.

Перша адреса, яку кодує рядок I7, дорівнює 8000. Вона більша за нижню границю діапазону (0000), і тому віднімаємо від неї 1 і отримуємо попередню адресу - 7FFF. Оскільки вона закінчується п'ятнадцятьма двійковими 1, то можна ці п'ятнадцять розрядів виділити у змінну групу і мінімізувати, як це зроблено в рядку I8.

Перша адреса, яку кодує рядок I8, дорівнює 0000. Вона дорівнює нижній границі, тому мінімізацію на цьому закінчуємо.

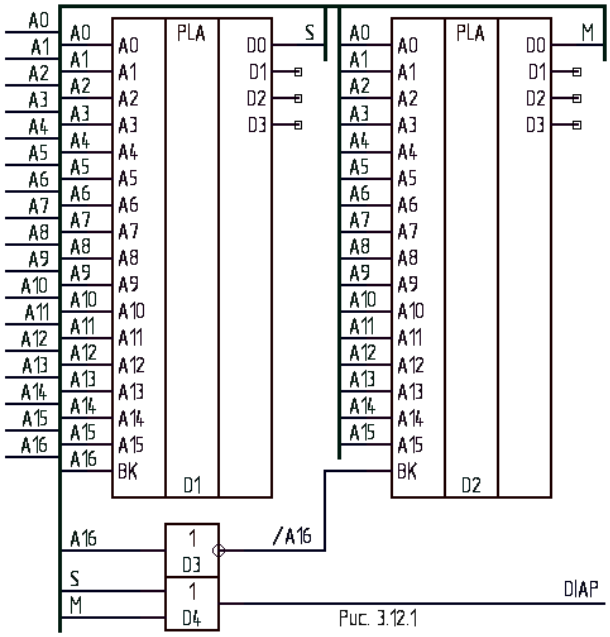


Рис. 3.12.1

Схему з'єднання ПЛМ наведено на рис. 3.12.1.

Входи обох ПЛМ запаралелені, за винятком сигналу ВК: на ПЛМ D1 на ВК подано сімнадцятий розряд адреси (A_{16}), а на D2 - його інверсія ($\overline{A_{16}}$). Тому ПЛМ D1 працює, коли адреси на вході менші за 10000 (згідно з табл. 3.12.2), а D2 - коли адреси більше або дорівнюють 10000 (згідно з табл. 3.8.3). Виходи ПЛМ об'єднані за допомогою елемента АБО D4, на його виході формується потрібний сигнал діапазону адрес DIAP = M v S.

Розташування на числовій осі мінімального і максимального значень 17-бітної адреси, границь діапазона, а також адрес, які дешифрує кожна з ПЛМ, показані на рис. 3.12.2.

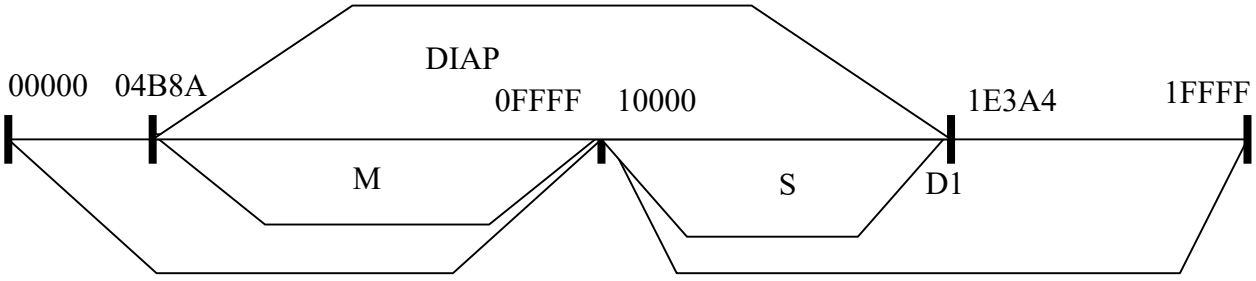


Рис. 3.12.2

Можна записати ДНФ результатів мінімізації:

$$M = I_0 \vee I_1 \vee \ldots \vee I_7 = \overline{A_{16}}\ \overline{A_{15}} A_{14} \overline{A_{13}}\ \overline{A_{12}} A_{11} \overline{A_{10}} A_9 A_8 A_7 \overline{A_6}\ \overline{A_5}\ \overline{A_4} A_3 A_2 A_1 A_0 \vee \ldots \vee \overline{A_{16}} A_{15}.$$

Ця формула припускає незначних подальших скорочень (використовується формула поглинання $a\bar{b} \vee b = a \vee b$ - зникають закреслені 0 у табл. 3.12.4):

$$M = I_0 \vee I_1 \vee \ldots \vee I_7 = \overline{A_{16}} A_{14} A_{11} A_9 A_8 A_7 A_3 A_2 A_1 A_0 \vee \overline{A_{16}} A_{14} A_{11} A_9 A_8 A_7 A_4 \vee \ldots \vee \overline{A_{16}} A_{15}.$$

Аналогічно, виконується додаткове спрощення сигналу S (використовується формула поглинання $a\bar{b} \vee b = a \vee b$ - зникають закреслені 1 у табл. 3.12.5). Тоді

$$S = I_0 \vee I_1 \vee \ldots \vee I_8 = A_{16} \overline{A_{12}}\ \overline{A_{11}}\ \overline{A_{10}}\ \overline{A_6}\ \overline{A_4}\ \overline{A_3}\ \overline{A_1}\ \overline{A_0} \vee A_{16} \overline{A_{12}}\ \overline{A_{11}}\ \overline{A_{10}}\ \overline{A_6}\ \overline{A_4}\ \overline{A_3}\ \overline{A_2} \vee \ldots \vee A_{16} \overline{A_{15}}.$$

Кінцеві результати мінімізації наведено в табл. 3.12.6 та табл. 3.12.7.

Таблиця 3.12.4

Мінімізація діапазону кодів з довільними границями в напрямку зростання кодів

N	Розряди коду A															
	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
I0	0	~~0~~	1	~~0~~	~~0~~	1	~~0~~	1	1	1	~~0~~	~~0~~	~~0~~	1	1	1	1
I1	0	~~0~~	1	~~0~~	~~0~~	1	~~0~~	1	1	1	~~0~~	~~0~~	1	–	–	–	–
I2	0	~~0~~	1	~~0~~	~~0~~	1	~~0~~	1	1	1	~~0~~	1	–	–	–	–	–
I3	0	~~0~~	1	~~0~~	~~0~~	1	~~0~~	1	1	1	1	–	–	–	–	–	–
I4	0	~~0~~	1	~~0~~	~~0~~	1	1	–	–	–	–	–	–	–	–	–	–
I5	0	~~0~~	1	~~0~~	1	–	–	–	–	–	–	–	–	–	–	–	–
I6	0	~~0~~	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–
I7	0	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I8	H	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Таблиця 3.12.5

Мінімізація діапазону кодів з довільними границями в напрямку зменшення кодів

N	Входи A ПЛМ															
	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
I0	1	~~1~~	~~1~~	~~1~~	0	0	0	~~1~~	~~1~~	~~1~~	0	~~1~~	0	0	~~1~~	0	0
I1	1	~~1~~	~~1~~	~~1~~	0	0	0	~~1~~	~~1~~	~~1~~	0	~~1~~	0	0	0	–	–
I2	1	~~1~~	~~1~~	~~1~~	0	0	0	~~1~~	~~1~~	~~1~~	0	0	–	–	–	–	–
I3	1	~~1~~	~~1~~	~~1~~	0	0	0	~~1~~	~~1~~	0	–	–	–	–	–	–	–
I4	1	~~1~~	~~1~~	~~1~~	0	0	0	~~1~~	0	–	–	–	–	–	–	–	–
I5	1	~~1~~	~~1~~	~~1~~	0	0	0	0	–	–	–	–	–	–	–	–	–
I6	1	~~1~~	~~1~~	0	–	–	–	–	–	–	–	–	–	–	–	–	–
I7	1	~~1~~	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I8	1	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I9	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Таблиця 3.12.6

Кінцевий результа мінімізації діапазону кодів з довільними границями в напрямку зростання кодів

N	Розряди коду A															
	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
I0	0	–	1	–	–	1	–	1	1	1	–	–	–	1	1	1	1
I1	0	–	1	–	–	1	–	1	1	1	–	–	1	–	–	–	–
I2	0	–	1	–	–	1	–	1	1	1	–	1	–	–	–	–	–
I3	0	–	1	–	–	1	–	1	1	1	1	–	–	–	–	–	–
I4	0	–	1	–	–	1	1	–	–	–	–	–	–	–	–	–	–
I5	0	–	1	–	1	–	–	–	–	–	–	–	–	–	–	–	–
I6	0	–	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–
I7	0	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I8	H	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Таблиця 3.12.7

Кінцевий результа мінімізації діапазону кодів з довільними границями в напрямку зменшення кодів

N	Входи A ПЛМ															
	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
I0	1	–	–	–	0	0	0	–	–	–	0	–	0	0	–	0	0
I1	1	–	–	–	0	0	0	–	–	–	0	–	0	0	0	–	–
I2	1	–	–	–	0	0	0	–	–	–	0	0	–	–	–	–	–
I3	1	–	–	–	0	0	0	–	–	0	–	–	–	–	–	–	–
I4	1	–	–	–	0	0	0	–	0	–	–	–	–	–	–	–	–
I5	1	–	–	–	0	0	0	0	–	–	–	–	–	–	–	–	–
I6	1	–	–	0	–	–	–	–	–	–	–	–	–	–	–	–	–
I7	1	–	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I8	1	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
I9	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

4 Методичні вказівки щодо виконання арифметико-логічних операцій

4.1 Виконання логічних операцій над двійковими числами

Логічні операції належать до порозрядних операцій, тобто, до таких, де результат дії над будь-яким розрядом не залежить від результатів дії над рештою розрядів.

Таблиця 4.1.1

Аргументи		a & b	a v b	a ⊕b
a	b	(and)	(or)	(xor)
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Таблиця 4.1.2

a=	1	1	0	0	0	1	1	1	C7
b=	0	0	1	1	0	0	1	1	33
a&b=	0	0	0	0	0	0	1	1	03
avb=	1	1	1	1	0	1	1	1	F7
aÅb=	1	1	1	1	0	1	0	0	F4

Результати виконання логічних операцій І (&, and, кон'юнкція, логічне множення), АБО (v, +, or, диз'юнкція, логічне додавання), виключне АБО (⊕, xor, додавання за модулем 2) над однобітними числами a і b ілюструються табл. 4.1.1.

Приклад 4.1.1. Виконати порозрядні операції І, АБО, виключне АБО над багаторозрядними двійковими числами, поданими у шістнадцятковому коді: a=c7 і b=33. Результати виконання вказаних операцій наведено у табл.4.1.2.

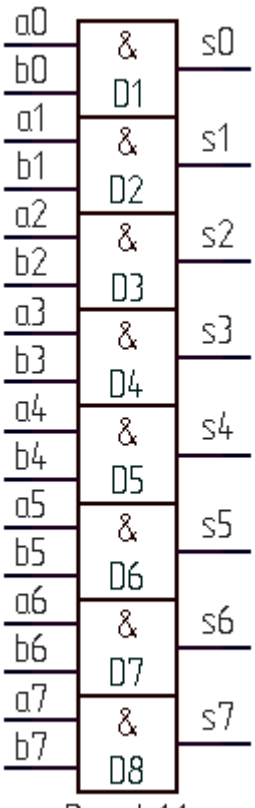


Рис. 4.1.1

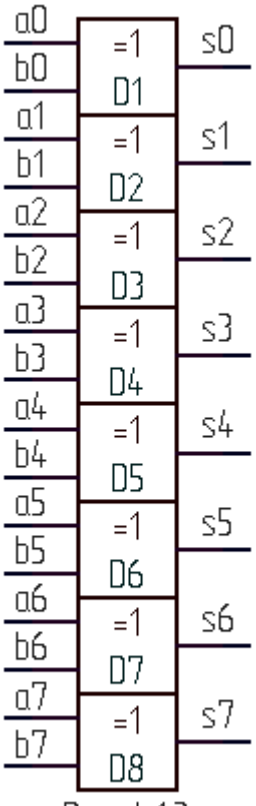


Рис. 4.1.3

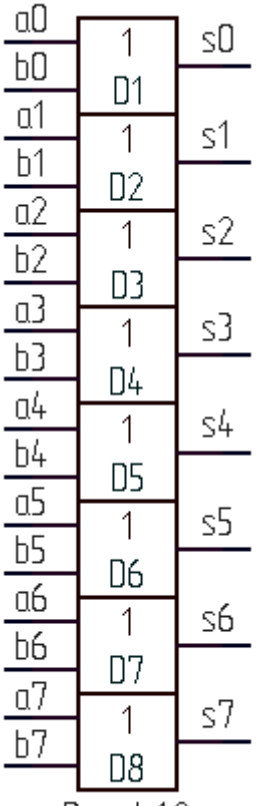


Рис. 4.1.2

Функціональні схеми пристроїв, які виконують описані вище операції наведено на рис. 4.1.1...4.1.3. Елементи додавання за модулем 2 (виключне АБО) на рис.4.1.3 у відповідності з їхньою досконалою диз'юнктивною нормальною формою (ДДНФ) $a \oplus b = \bar{a}b \vee a\bar{b}$ мають внутрішню функціональну схему, наведену на рис. 4.1.4. Така схема знаходиться всередині кожного з елементів D1…D8 на рис. 4.1.3. Рис. 4.1.1 і 4.1.2 є схемами в базисі Буля, для представлення схеми рис. 4.1.3 у базисі Буля необхідно кожний її елемент D1…D8 замінити не його функціональну схему рис. 4.1.4.

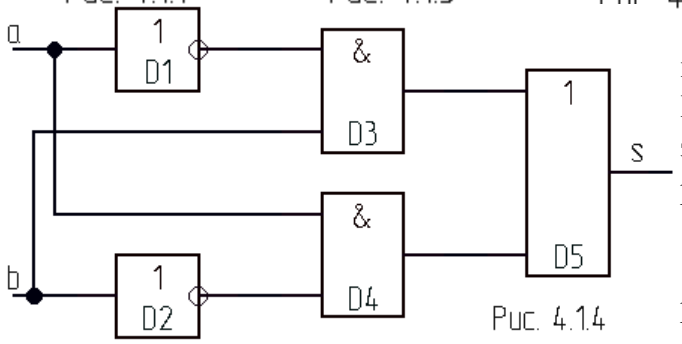


Рис. 4.1.4

4.2 Віднімання двійкових чисел

Віднімання двійкових чисел можна робити кількома способами:

відніманням прямих кодів цих чисел;

додаванням обернених кодів цих чисел;

додаванням доповняльних кодів цих чисел.

Останній варіант найбільш поширений.

Перед виконанням віднімання чисел у доповняльних кодах треба:

1) замінити операцію віднімання на операцію додавання і одночасно змінити знак числа, яке віднімається, на протилежний;

2) для вирівнювання довжини чисел дозволяється дописувати:

двійкові 0 перед старшим значущим розрядом у додатних числах і у числах, представлених у прямому коді;

двійкові 0 після молодшого розряду дробу - незалежно від того, додатне це число, чи від'ємне, прямий код чи доповняльний;

двійкові 1 - перед старшим значущим розрядом від'ємних чисел у доповняльному коді;

3) перевести обидва числа в доповняльний код;

4) виконати додавання;

5) при потребі результат перевести в прямий код.

Приклад 4.2.1. Виконати операцію +1234 - 56 (числа шістнадцяткові).

1) +1234 + (-56);

2) 0.0001 0010 0011 0100 + 1.0000 0000 0101 0110 - числа у прямому двійковому коді, крапка відділяє в даному випадку знак числа від значущих розрядів числа;

3) 0.0001 0010 0011 0100 + 1.1111 1111 1010 1010 - числа у доповняльному двійковому коді;

4) 0.0001 0010 0011 0100
+ 1.1111 1111 1010 1010

0.0001 0001 1101 1110 - результат додатній у двійковому коді.

0.11DE - результат у прямому шістнадцятковому коді.

Результат: +11DE.

Приклад 4.2.2. Виконати операцію +56 - 1234 (числа шістнадцяткові).

1) +56 + (-1234);

2) 0.0000 0000 0101 0110 + 1.0001 0010 0011 0100 - числа у прямому двійковому коді, крапка відділяє в даному випадку знак числа від розрядів числа;

3) 0.0000 0000 0101 0110 + 1.1110 1101 1100 1100 - числа у доповняльному двійковому коді;

4) 0.0000 0000 0101 0110
+ 1.1110 1101 1100 1100

1.1110 1110 0010 0010 - результат від'ємний у двійковому доповняльному коді.

5) 1.0001 0001 1101 1110 - результат від'ємний у прямому двійковому коді.

Результат: -11DE (у шістнадцятковому коді).

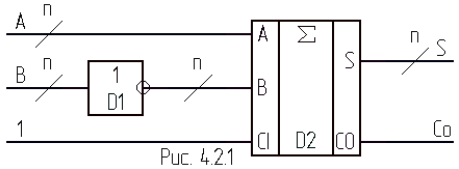


Рис. 4.2.1

Функціональну схему пристрою, який виконує віднімання n-розрядних двійкових чисел, наведено на рис. 4.2.1. Пристрій складається з n-розрядного інвертора і n-розрядного суматора. Інвертор утворює обернений код числа B, який після додавання одиниці молодшого розряду (на вході переносу CI – Carry In - суматора) перетворюється у доповняльний код. Суматор здійснює додавання доповняльних кодів. Результат формується на n-розрядному виході суми S і виході переносу CO (Carry Out). Внутрішню структуру n-розрядного суматора (для n=4 - для чотирирозрядного суматора) наведено на рис. 4.2.2. N-розрядний суматор складається з n послідовно з'єднаних однорозрядних повних суматорів.

Таблицю істинності повного однорозрядного суматора наведено в табл. 4.2.1. З табл. 4.2.1 видно, що

$$C_o = \overline{A}BC_i \vee A\overline{B}C_i \vee AB\overline{C_i} \vee ABC_i = BC_i \vee AC_i \vee AB;$$

$$S = \overline{A}\,\overline{B}C_i \vee \overline{A}B\overline{C_i} \vee A\overline{B}\,\overline{C_i} \vee ABC_i.$$

Таблиця 4.2.1

Входи			Виходи	
A	B	Ci	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

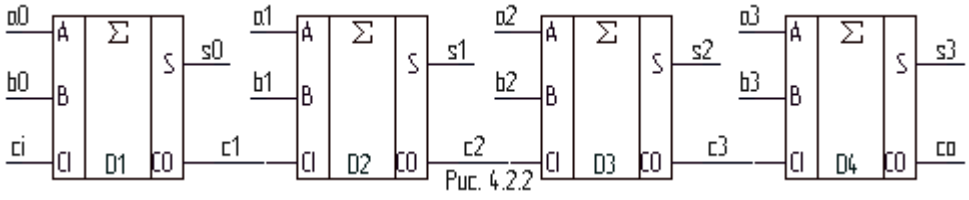


Рис. 4.2.2

Функціональну схему повного однорозрядного суматора наведено на рис. 4.2.3.

4.3 Округлення двійкових чисел

Найчастіше округлення двійкових чисел відбувається шляхом відкидання неточних (непотрібних) розрядів і заміни їх нулями. При цьому розряди, які залишаються, лишаються або незмінним (якщо сумарна вага розрядів, які відкидаються, менша за 1/2 ваги наймолодшого розряду, який залишається), або число, яке вони зображають, збільшується на 1 молодшого розряду (у протилежному випадку). Якщо число додатне, або воно представлено в прямому коді, то сумарна вага розрядів, які відкидаються, буде менша за 1/2 ваги наймолодшого розряду з тих, які залишаються, тоді, коли старший з розрядів, які відкидаються, дорівнює 0.

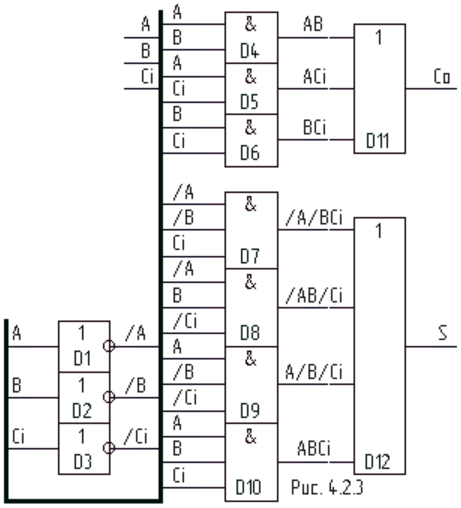


Рис. 4.2.3

Якщо число від'ємне і представлено в доповняльному коді, то сумарна вага розрядів, які відкидаються, буде менша за 1/2 ваги наймолодшого розряду з тих, які залишаються, тоді, коли старший з розрядів, які відкидаються, дорівнює 0 або тоді, коли одночасно старший розряд дорівнює 1, а всі молодші за нього - 0.

Щоб не аналізувати розряди, на практиці до чисел у прямому коді, які підлягають округленню, додають константу типу 00...00'100...0, де апострофом позначено місце округлення. Після цього відкидають розряди, які розташовані правіше від апострофа.

До від'ємного числа в доповняльному коді, яке підлягає округленню, додають константу типу 00...00'011....1, де апострофом позначене місце округлення, після цього відкидають розряди, які розташовані правіше від апострофа.

Приклад 4.3.1. Послідовно округлити додатне число в місцях, позначених крапками: 01.001.100.011.100.001.111.

1) 01 001 100 011 100 001(111)
+00 000 000 000 000 000(100)
———————————————————
01 001 100 011 100 010(011)

Розряди, які відкидаються, взято в дужки.

Результат першого округлення 01 001 100 011 100 010 (000).

2) 01 001 100 011 100(010 000)
+00 000 000 000 000(100 000)
———————————————————
01 001 100 011 100(110 000).

Результат другого округлення 01 001 100 011 100 (000 000).

3) 01 001 100 011(100 000 000)
+00 000 000 000(100 000 000)
———————————————————
01 001 100 100(000 000 000).

Результат третього округлення 01 001 100 100 (000 000 000).

4) 01 001 100(100 000 000 000)
+00 000 000(100 000 000 000)
———————————————————
01 001 101(000 000 000 000).

Результат четвертого округлення 01 001 101 (000 000 000 000).

5) 01 001(101 000 000 000 000)
+00 000(100 000 000 000 000)
———————————————————
01 010(001 000 000 000 000).

Результат п'ятого округлення 01 010 (000 000 000 000 000).

6) 01(010 000 000 000 000 000)
+00(100 000 000 000 000 000)
———————————————————
01(110 000 000 000 000 000).

Результат шостого округлення 01 (000 000 000 000 000 000).

Приклад 4.3.2. Послідовно округлити від'ємне число в доповняльному коді у місцях, позначених крапками: 11.001.100.011.100.001.111.

1) 11 001 100 011 100 001(111)
+00 000 000 000 000 000(011)
———————————————————
11 001 100 011 100 010(010).

Розряди, які відкидаються, взято в дужки.

Результат першого округлення 11 001 100 011 100 010 (000).

2) 11 001 100 011 100(010 000)
+00 000 000 000 000(011 111)
———————————————————
11 001 100 011 100(101 111).

Результат другого округлення 11 001 100 011 100 (000 000).

3) 11 001 100 011(100 000 000)
+00 000 000 000(011 111 111)
———————————————————————
11 001 100 011(111 111 111).

Результат третього округлення 11 001 100 011 (000 000 000).

4) 11 001 100(011 000 000 000)
+00 000 000(011 111 111 111)
———————————————————————
11 001 100(110 111 111 111).

Результат четвертого округлення 11 001 100 (000 000 000 000).

5) 11 001(100 000 000 000 000)
+00 000(011 111 111 111 111)
———————————————————————
11 001(111 111 111 111 111).

Результат п'ятого округлення 11 001 (000 000 000 000 000).

6) 11(001 000 000 000 000 000)
+00(011 111 111 111 111 111)
———————————————————————
11(100 111 111 111 111 111).

Результат п'ятого округлення 11 (000 000 000 000 000 000).

4.4 Арифметика двійково-десяткових чисел

Двійково-десяткові числа утворюються, коли в двійковий код переводиться не все десяткове число, а окремо його цифри. Існує багато двійково-десяткових кодів, один з поширених серед них - код «8-4-2-1». При використанні цього коду кожна десяткова цифра від 0 до 9 кодується чотирма двійковими розрядами (тетрадою), які можуть приймати значення, відповідно, від 0000 до 1001 (табл. 4.4.1). Значення від 1010 до 1111 є забороненими. Поява таких значень свідчить про переповнення при роботі з двійково-десятковими числами. Зрозуміло, що арифметичне додавання двійково-десяткових чисел

Таблиця 4.4.1

Десятковий код	Двійково-десятковий код	Двійковий код
1239	0001 0010 0011 1001	0100 1101 0111

повинно відбуватися за іншими правилами, ніж додавання звичайних двійкових чисел, якщо ми хочемо отримати у результаті двійково-десяткові числа.

Послідовність дій при двійково-десятковому додаванні така:

1) виконати двійкове додавання двох чисел;

2) проаналізувати молодшу тетраду результату, для визначення необхідності її корекції. Ознакою необхідності корекції э значення тетради в діапазоні від 1010 до 1111 або наявність переносу з цієї тетради до наступної;

3) якщо потрібно, виконати корекцію проаналізованої тетради результату: додати до нього число, всі розряди якого дорівнюють 0, за винятком розрядів, які відповідають даній тетраді, вони повинні дорівнювати 0110;

4) після цього повторити пп. 2, 3 для всіх наступних тетрад.

Приклад 4.4.1. Додати у двійково-десятковому коді 45+55.

45 = 0100 0101;

55 = 0101 0101.

```
 0100 0101
+0101 0101
----------
 1001 1010
```

Молодша тетрада приймає заборонене значення (1010), тому її потрібно відкоректувати:

```
 1001 1010
+0000 0110
----------
 1010 0000
```

Тепер і наступна тетрада приймає заборонене значення, її також треба відкоректувати:

```
  1010 0000
 +0110 0000
-----------
1 0000 0000.
```

Результат дорівнює 100 у десятковому коді.

Приклад 4.4.2. Додати у двійково-десятковому коді 86+87.

86 = 1000 0110;

87 = 1000 0111.

```
  1000 0110
 +1000 0111
----------
1 0000 1101.
```

Молодша тетрада приймає заборонене значення (1101), тому її потрібно відкоректувати:

```
1 0000 1101
 +0000 0110
----------
1 0001 0011.
```

З наступної тетради був перенос при першому додаванні, її також треба відкоректувати:

```
1 0001 0011
+ 0110 0000
-----------
1 0111 0011.
```

Результат дорівнює 173 у десятковому коді.

Віднімання двійково-десяткових чисел (А - В) так само, як і двійкових чисел, виконується методом заміни операції віднімання числа В на додавання доповняльного двійково-десяткового коду В(дв.-дес.д.к.).

Правила отримання доповняльного двійково-десяткового коду числа В:

1) отримати доповняльний двійковий код числа В(дв.д.к.);

2) отримати обернений двійково-десятковий код, виконавши додавання двійкових чисел (без корекції): В(дв.-дес.о.к.) = 99...9(шістнадцятковий код) + В(дв.д.к.);

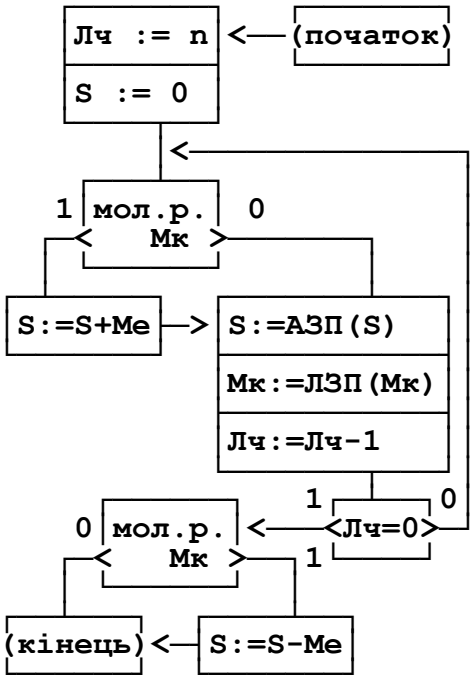


Рис. 4.5.1

- оператор дії;

- умовний оператор;

- оператори початку та кінця.

3) отримати доповняльний двійково-десятковий код:

В(дв.-дес.д.к.) = В(дв.-дес.о.к.) + 1 молодшого розряду.

Приклад 4.4.3. Виконати віднімання 55 - 13.

55 = 0101 0101;

13 = 0001 0011 = В,

Двійковий обернений код В(дв.о.к.)= 1110 1100,

Двійковий доповняльний код В(дв.д.к.)= 1110 1101,

1001 1001 (99)
+1110 1101
———————

1000 0110 – двійково-десятковий обернений код В(дв.-дес.о.к.)

1000 0111 - двійково-десятковий доповняльний код В(дв.-дес.д.к.).

Додавання доповняльного коду за правилами двійково-десяткової арифметики:

0101 0101 (55)
+ 1000 0111
———————

1101 1100
+ 0000 0110 (корекція першої тетради)
———————

1110 0010
+ 0110 0000 (корекція другої тетради)
———————

1 0100 0000

Результат 0100 0010 = 42, перенос за межі розрядної сітки вказує на те, що результат додатній, тобто, 42 - це його абсолютна величина.

Приклад 4.4.4. Виконати віднімання 13 - 55.

55 = 0101 0101 = В;

13 = 0001 0011.

Вдв.о.к = 1010 1010,

Вдв.д.к = 1010 1011,

1001 1001 (99)
+1010 1011
———————

0100 0100 В(дв.-дес.о.к);

0100 0101 В(дв.-дес.д.к).

Додавання доповняльного коду за правилами двійково-десяткової арифметики:

0001 0011 (55)
+0100 0101

0101 1000.

Результат 0101 1000, відсутність переносу за межі розрядної сітки вказує на те, що результат у доповняльному двійково-десятковому коді від'ємний, щоб отримати його абсолютну величину необхідно перевести результат з доповняльного коду в прямий. Алгоритм переведення такий же, як і для переведення з прямого коду в доповняльний:

інверсія двійкових розрядів 1010 0111;
додавання 1 молодшого розряду 1010 1000
доповнення до 99 +1001 1001

0100 0001;
прямий двійково-десятковий код 0100 0010.

Результат = -0100 0010 = -42.

Таблиця 4.5.1

Лч	S	Мк	мол.р. Мк	Наступна операція
4	00.00000000	0.1001	1	S:=S+Me
	00.00000000 +00.0101 00.01010000			S:=АЗП(S)
	->00.00101000			Мк:=ЛЗП(Мк)
	00.00101000	0.0100		Лч:=Лч-1
3	00.00101000		0	S:=АЗП(S)
	->00.00010100			Мк:=ЛЗП(Мк)
	00.00010100	0.0010		Лч:=Лч-1
2	00.00010100		0	S:=АЗП(S)
	->00.00001010			Мк:=ЛЗП(Мк)
	00.00001010	0.0001		Лч:=Лч-1
1	00.00001010 +00.0101 00.01011010		1	S:=S+Me S:=АЗП(S)
	->00.00101101			Мк:=ЛЗП(Мк)
	00.00101101	0.0000		Лч:=Лч-1
0	00.00101101		0	кінець

4.5 Множення двійкових чисел у доповняльному коді

Множення двійкових чисел може здійснюватися:

послідовним способом – при цьому одночасно аналізується тільки один розряд множника і формуються часткові добутки множеного на один розряд множника, ці добутки послідовно сумуються;

паралельним способом - при цьому одночасно аналізуються всі розряди множника і одразу формується кінцевий результат - добуток множеного на усі розряди множника;

проміжним способом – при цьому одночасно аналізуються декілька розрядів множника і формуються часткові добутки множеного на декілька розрядів множника, ці добутки послідовно сумуються.Один з методів послідовного множення двійкових чисел у доповняльному коді починається з аналізу молодших розрядів множника і виконується з

зсувом суматора праворуч. Якщо множник від'ємний, то наприкінці множення необхідно робити корекцію результату. Оскільки для n-розрядного від'ємного числа сума його доповняльного і прямого кодів дорівнює 2^{n+1}, а при множенні доповняльний код буде сприйматися саме як прямий, то, насправді, при множенні буде одержаний результат, більший від правильного на $X \cdot 2^{n+1}$(де X - множене). На цю величину і потрібно буде відкоректувати результат. Тобто,

$X \cdot Y_{пк} = X \cdot Y_{дк} - X \cdot 2^{n+1}$.

На алгоритмі множення (рис. 4.5.1) введені такі позначеня:

Лч - лічильник;

n - кількість двійкових розрядів множника без знаку;

Мк - множник;

Ме - множене;

S - суматор, після виконання алгоритму в ньому буде знаходитися результат. Розрядність суматора без врахування знаку дорівнює n+m, де m - кількість розрядів множеного без врахування знаку (у прикладі n=m);

мол.р. - молодший розряд;

АЗП - арифметичний зсув праворуч;

ЛЗП - логічний зсув праворуч.

Приклад 4.5.1. Помножити (+5)x(+9)=(+45), всі числа - десяткові.

Перебіг множення поданий у табл. 4.5.1.

Ме = 0.0101;

Мк = 0.1001 (крапка відділяє знак від розрядів ваги, а не цілу частину числа від дробової, у цьому прикладі всі числа цілі).

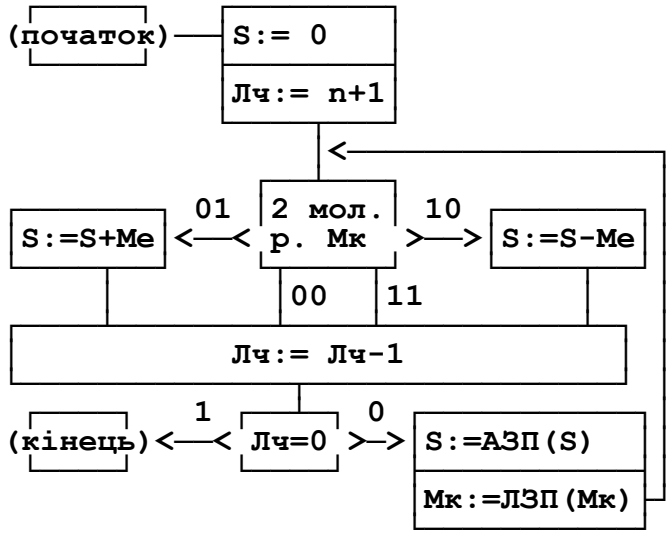


Рис. 4.6.1

Щоб полегшити виявлення переповнення, додавання в суматорі відбувається в модифікованому доповняльному коді.

Результат у двійковому коді: 00.00101101 (перші два 0 кодують знак результату «+»), у шістнадцятковому коді:

+2D=2·16+13=45 (у десятковому коді).

Приклад 4.5.2. Помножити (+5)x(-9)=(-45), всі числа - десяткові.

Перебіг множення поданий у табл. 4.5.2.

Ме = 0.0101;

-Ме = 1.1011;

Таблиця 4.5.2

Лч	S	Мк	мол. р. Мк	Наступна Наступна операція
4	00.00000000	1.0111	1	S:=S+Me
	00.00000000 +00.0101 00.01010000			S:=АЗП(S)
	>00.00101000			Мк:=ЛЗП(Мк)
	00.00101000	0.1011		Лч:=Лч-1
3	00.00101000		1	S:=S+Me
	00.00101000 +00.0101 00.01111000			S:=АЗП(S)
	>00.00111100			Мк:=ЛЗП(Мк)
	00.00111100	0.0101		Лч:=Лч-1
2	00.00111100		1	S:=S+Me
	00.00111100 +00.0101 00.10001100			S:=АЗП(S)
	>00.01000110			Мк:=ЛЗП(Мк)
	00.01000110	0.0010		Лч:=Лч-1
1	00.01000110		0	S:=АЗП(S)
	>00.00100011			Мк:=ЛЗП(Мк)
	00.00100011	0.0001		Лч:=Лч-1
0	00.00100011		1	S:=S-Me
	00.00100011 +11.1011 11.11010011			кінець

Мк = 1.0111.

Результат у двійковому доповняльному коді 11.11010011 (перші дві 1 кодують знак результату «-»), у двійковому прямому коді -00101101, у шістнадцятковому коді –2D, у десятковому коді -45.

4.6 Множення двійкових чисел у доповняльному коді методом Бута

У загальному випадку метод Бута полягає в одночасному аналізі відразу декількох розрядів множника. У цьому розділі розглядається множення на два розряди. Цей метод зменшує кількість операцій додавання в алгоритмі. Алгоритм наведений далі (рис. 4.6.1).

На рис. 4.6.1 введені такі позначеня:

Лч - лічильник;

n - кількість двійкових розрядів множника без знаку;

Мк - множник; Ме - множене; S - суматор, після виконання алгоритму в ньому буде знаходитися результат. Розрядність суматора без врахування знаку дорівнює n+m, де m - кількість розрядів множеного без врахування знаку (у прикладі n=m).

2 мол.р. - два молодших розряди;

АЗП - арифметичний зсув праворуч;

ЛЗП - логічний зсув праворуч.

Приклад 4.6.1. Помножити (+5)х(+9)=(+45), всі числа - десяткові.

Перебіг множення поданий у табл. 4.6.1.

Ме = 0.0101;

-Ме = 1.1011;

Мк = 0.1001(0).

Таблиця 4.6.1

Лч	S	Мк	мл. р. Мк	Наступна операція
5	00.00000000	0.1001(0)	10	S:=S-Me
	00.00000000 +11.1011 11.10110000			Лч:=Лч-1
4				S:=АЗП(S)
	11.11011000			Мк:=ЛЗП(Мк)
	11.11011000	0.0100(1)	01	S:=S+Me
	11.11011000 +00.0101 00.00101000			Лч:=Лч-1
3				S:=АЗП(S)
	00.00010100			Мк:=ЛЗП(Мк)
	00.00010100	0.0010(0)	00	Лч:=Лч-1
2				S:=АЗП(S)
	00.00001010			Мк:=ЛЗП(Мк)
	00.00001010	0.0001(0)	10	S:=S-Me
	00.00001010 +11.1011 11.10111010			Лч:=Лч-1
1				S:=АЗП(S)
	11.11011101			Мк:=ЛЗП(Мк)
	11.11011101	0.0000(1)	01	S:=S+Me
	11.11011101 +00.0101 00.00101101			Лч:=Лч-1
0				Кінець

У дужках вказаний додатковий розряд множника, який спочатку повинен дорівнювати 0. Крапка відділяє знак від розрядів ваги, а не цілу частину числа від дробової, у цьому прикладі всі числа цілі.

Щоб полегшити виявлення переповнення, додавання в суматорі відбувається в модифікованому доповняльному коді.

Результат у двійковому доповняльному коді: 00.00101101 (перші два 0 кодують знак результату «+»), у шістнадцятковому коді: +2D=2·16+13=45 (у десятковому коді).

Приклад 4.6.2. Помножити (+5)x(-9)=(-45), всі числа - десяткові.

Перебіг множення поданий у табл. 4.6.2.

Me = 0.0101;

-Me = 1.1011;

Мк = 1.0111.

Результат у двійковому доповняльному коді: 11.11010011 (перші дві 1 кодують знак результату «-»), у двійковому прямому коді: -00101101, у шістнадцятковому коді: -2D = -(2·16 + 13) = -45 (у десятковому коді).

4.7 Паралельний (матричний) помножувач.

Таблиця 4.6.2

Лч	S	Мк	мл. р. Мк	Наступна операція операція
5	00.00000000	1.0111(0)	10	S:=S-Me
	00.00000000 +11.1011 11.10110000			Лч:=Лч-1
4				S:=АЗП(S)
	11.11011000			Мк:=ЛЗП(Мк)
	11.11011000	0.1011(1)	11	Лч:=Лч-1
3				S:=АЗП(S)
	11.11101100			Мк:=ЛЗП(Мк)
	11.11101100	0.0101(1)	11	Лч:= Лч-1
2				S:=АЗП(S)
	11.11110110			Мк:=ЛЗП(Мк)
	11.11110110	0.0010(1)	01	S:=S+Me
	11.11110110 +00.0101 00.01000110			Лч:=Лч-1
1				S:=АЗП(S)
	00.00100011			Мк:=ЛЗП(Мк)
	00.00100011	0.0001(0)	10	S:=S-Me
	00.00100011 +11.1011 11.11010011			Лч:=Лч-1
0				Кінець

Функціональна схема паралельного помножувача ілюструє множення чисел у стовпчик на папері. На рис. 4.7.1 наведений приклад виконання множення у стовпчик АхВ = 7х7 (десятковий код) = 111х111 (двійковий код) = $=49_{10}$ = 110001_2. Функціональну схему паралельного помножувача для наведеного випадку множення трирозрядних двійкових чисел без знаку наведено на рис. 4.7.2. Схема призначена для множення трирозрядного числа А з розрядами а2 (старший), а1, а0 (молодший) на трирозрядне число В з розрядами b2 (старший), b1, b0 (молодший). На рис. 4.7.2 цифрами 0 та 1 позначені значення двійкових розрядів операндів і стани усіх елементів схеми.

```
  111
 x111
 ----
  111
 +111
 -----
 10101
+111
------
110001
```

Рис

4.8 Ділення двійкових чисел методом без відновлення залишків

Алгоритм виконання операції ділення методом без відновлення залишків наведений на рис. 4.8.1.

В алгоритмі використані такі позначення:

Лч - лічильник;

Ч - частка;

n - кількість двійкових розрядів частки без знаку. Для визначення кількості двійкових розрядів частки при діленні цілого числа на ціле необхідно в прямому двійковому коді написати одне під одним ділене і дільник, вирівнявши їх за одиницею в старшому розряді. Тоді кількість розрядів цілої частини частки n буде дорівнювати різниці довжин вирівняних дільника і діленого, збільшеній на 1;

Дк - дільник;

Де - ділене;

Зн - знак;

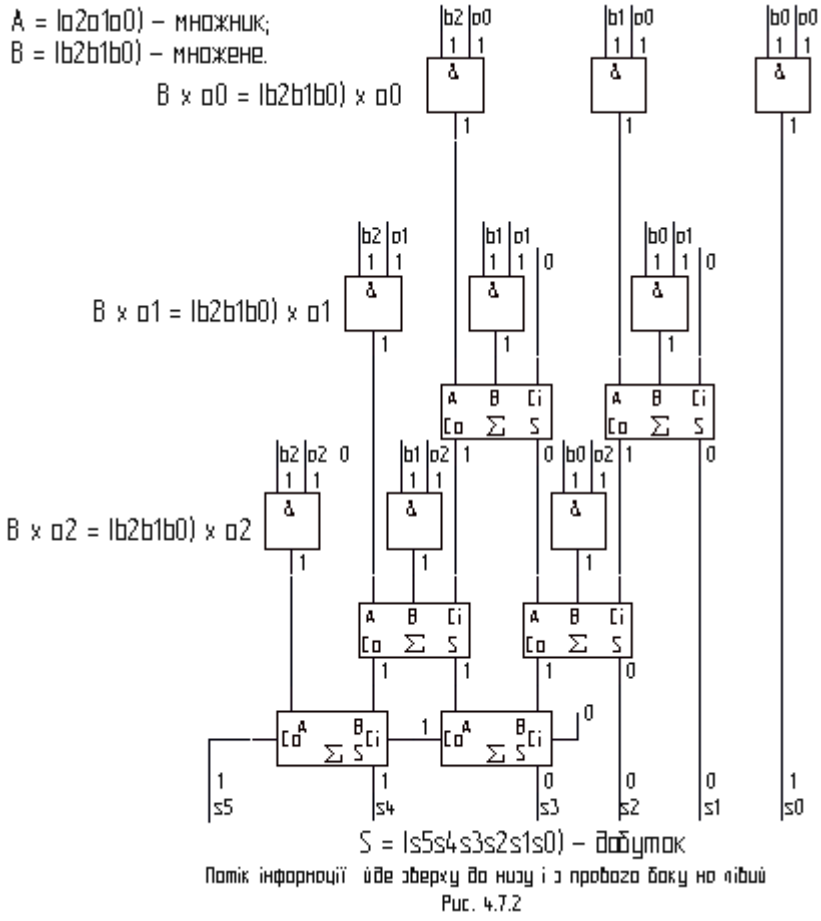


Рис. 4.7.2

⊕ - позначення операції додавання за модулем 2;

S - суматор, на початку виконання алгоритму в ньому буде знаходитися ділене;

ЛЗЛ(X),(Y) - логічний зсув ліворуч числа X, на звільнене місце наймолодшого розряду праворуч записується двійковий розряд Y.

Приклад 4.8.1. Поділити (-2D):5.

Де=-101101(прямий код);

Де = 11.010011 (доповняльний код);

Дк = 00.101 (доповняльний код), ЗнДк = 0;

-Дк = 11.011 (доповняльний код).

Визначення n:

Де = -101101,

Дк = +101. Де довше за Дк на 3 біти, тобто, n = 3+1 =4.

Процес ділення відображений в табл. 4.8.1.

Результат Ч = 10111, перший розряд треба розуміти як знаковий, тобто, результат у доповняльному коді дорівнює 1.0111 (крапка відділяє розряд знаку від вагових розрядів).

Результат від'ємний, у прямому двійковому коді його абсолютна величина дорівнює 1001, тобто, частка Ч=-1001=-9; (-2D):5 = (-9).

Приклад 4.8.2. Поділити 32:(-5) (числа - у шістнадцятковому коді), тобто, 50:(-5) (у десятковому коді).

Де = 00.110010 (доповняльний код);

Де = 00.110010 (прямий код);

Дк = 11.011 (доповняльний код), ЗнДк=1.

-Дк = 00.101 (доповняльний код).

Визначення n:

Де = -110010,

Дк = +101. Де довше за Дк на 3 біти, тобто, n = 3+1 = 4.

Процес ділення відображений в табл. 4.8.2.

Результат Ч = 10110, перший розряд треба розуміти як знаковий, тобто, результат у доповняльному коді дорівнює 1.0110 (крапка відділяє розряд знаку від вагових розрядів).

Результат від'ємний, у прямому коді його абсолютна величина дорівнює 1010, тобто, частка Ч = -1010 = -A. (-2D):5 = (-A).

Приклад 4.8.3. Поділити (32):5.

Де = 00.110010 (доповняльний код);

Де = + 110010 (прямий код);

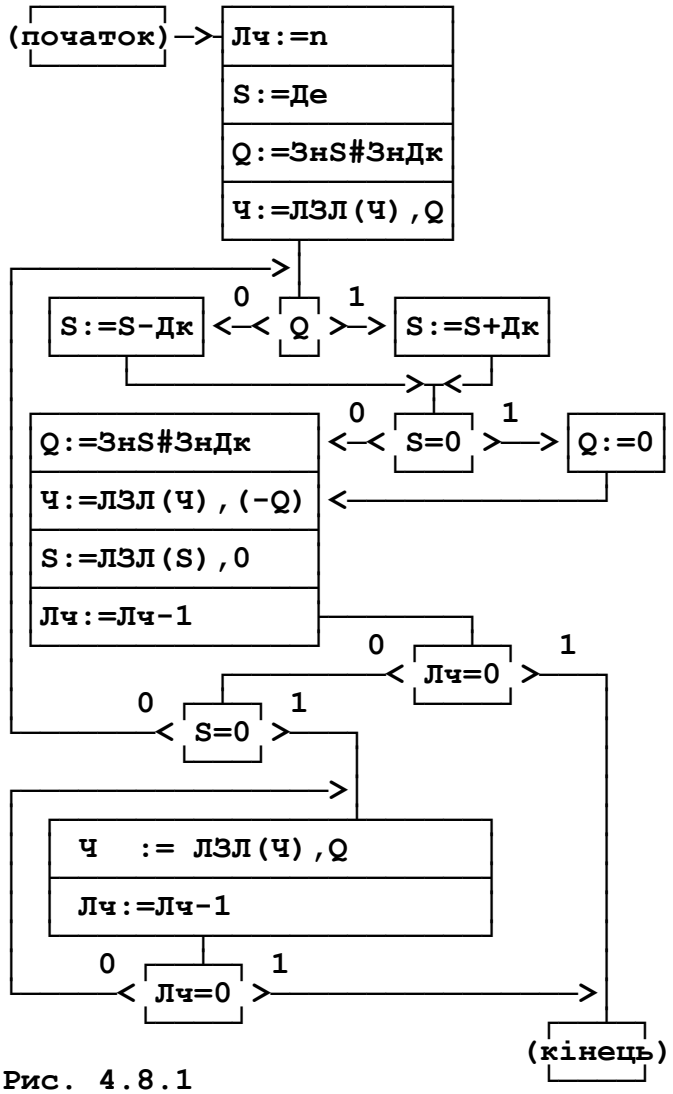


Рис. 4.8.1

Дк = 00.101 (доповняльний код);

-Дк = 11.011 (доповняльний код).

Визначення n:

Де = +101101, Дк = +101. Де довше за Дк на 3 біти, тобто, n = 3+1 = 4.

Процес ділення відображений в табл. 4.8.3.

Результат Ч = 01010, перший розряд треба розуміти як знаковий, тобто, результат у доповняльному коді дорівнює 0.1010 (крапка відділяє розряд знаку від вагових розрядів).

Результат додатний, тобто, частка Ч=+1010=+А. 32:5 = А.

4.9 Ділення чисел методом з відновленням залишків

Алгоритм виконання операції ділення методом з відновленням залишків наведений на рис. 4.9.1.

Метод розрахований на ділення абсолютних величин діленого і дільника, тому вхідні дані повинні бути у прямому коді без знаку, у середині алгоритму дії виконуються в доповняльному коді, вихідний код - прямий із знаком, знак обраховується попередньо із знаків діленого і дільника.

В алгоритмі використані такі позначення:

Лч - лічильник;

Ч - частка;

n - кількість двійкових розрядів частки без знаку. Для визначення кількості двійкових розрядів частки при діленні цілого числа на ціле необхідно в прямому двійковому коді написати одне під одним ділене і дільник, вирівнявши їх за одиницею у старшому розряді. Тоді кількість розрядів частки n буде дорівнювати різниці довжин вирівняних дільника і діленого, збільшеній на 1;

Дк - дільник; Де - ділене;

Зн - знак;

$\oplus$ - позначення операції додавання за модулем 2;

S - суматор, на початку виконання алгоритму в ньому буде знаходитися ділене;

ЛЗЛ(X),(Y) - логічний зсув ліворуч числа X, на звільнене місце наймолодшого розряду праворуч записується двійковий розряд Y.

Приклад 4.9.1. Поділити (-2D):5.

Де = 1.101101 (прямий код із знаком); ЗнДе = 1 (-);

Де = 0.101101 (прямий код без знаку);

Де = 1.010011 (доповняльний код);

Дк = 0.101 (прямий код, доповняльний код), ЗнДк = 0 (+);

-Дк = 1.011 (доповняльний код).

Визначення n:

Де = -101101, Дк = +101. Де довше за Дк на 3 біти, тобто, n= 3+1 = 4.

Процес ділення показаний у табл. 4.9.1.

Результат Ч = 11001, перший розряд треба розуміти як знаковий, тобто, результат у прямому коді дорівнює 1.1001 (крапка відділяє розряд знаку від вагових розрядів).

Результат від'ємний: (-2D):5 = (-9).

Приклад 4.9.2. Поділити 32:(-5) (числа - у шістнадцятковому коді), тобто, 50:(-5) (у десятковому коді).

Де = 0.110010 (прямий код);

ЗнДе = 0 (+);

Де = 0.110010 (доповняльний код);

Дк = 1.101 (прямий код), ЗнДк=1 (-);

Дк = 0.101 (прямий код, без знаку);

-Дк = 1.011 (доповняльний код). Визначення n:

Де = -110010,

Дк = +101. Де довше за Дк на 3 біти, тобто, n = 3+1 =4.

Процес ділення показаний в табл. 4.9.2.

Результат Ч = 11010, перший розряд треба розуміти як знаковий, тобто, результат у прямому коді дорівнює 1.1010 (крапка відділяє розряд знаку від вагових розрядів).

Результат від'ємний.

(-2D):5 = (-A).

Таблиця 4.8.1

Лч	S	ЗнДк # ЗнS	Q	Ч	Наступна операція
4	11.010011	0#1			Q:=ЗнS#ЗнДк
			1		Ч:=ЛЗЛ(Ч),Q
				1	S:=S+Дк
	11.010011 +00.101 ——————— 11.111011	0 # 1			Q:=ЗнS#ЗнДк
			1	1	Ч:=ЛЗЛ(Ч),/Q
			1	10	S:=ЛЗЛ(S),0
	11.110110				Лч:=Лч-1
3			1		S:=S+Дк
	11.110110 +00.101 ——————— 00.011110	0 # 0			Q:=ЗнS#ЗнДк
			0	10	Ч:=ЛЗЛ(Ч),/Q
			0	101	S:=ЛЗЛ(S),0
	00.111100				Лч:=Лч-1
2			0		S:=S-Дк
	00.111100 +11.011 ——————— 00.010100	0 # 0			Q:=ЗнS#ЗнДк
			0	101	Ч:=ЛЗЛ(Ч),/Q
			0	1011	S:=ЛЗЛ(S),0
	00.101000				Лч:=Лч-1
1			0		S:=S-Дк
	00.101000 +11.011 ——————— 00.000000	0 # 0			Q:=0
			0	1011	Ч:=ЛЗЛ(Ч),/Q
			0	10111	S:=ЛЗЛ(S),Q
	00.000000				Лч:=Лч-1
0					Кінець

Таблиця 4.8.2

Лч	S	ЗнS	Q	Ч	Наступна операція
4	00.110010	0			Q:=ЗнS#ЗнДк
		0	1		Ч:=ЛЗЛ(Ч),Q
				1	S:=S+Дк
	00.110010 +11.011 ——— 00.001010	0			Q:=ЗнS#ЗнДк
		0	1	1	Ч:=ЛЗЛ(Ч),/Q
			1	10	S:=ЛЗЛ(S),0
	00.010100	0			Лч:=Лч-1
3			1		S:=S+Дк
	00.010100 +11.011 ——— 11.101100	1			Q:=ЗнS#ЗнДк
		1	0	10	Ч:=ЛЗЛ(Ч),/Q
			0	101	S:=ЛЗЛ(S),0
	11.011000	1			Лч:=Лч-1
2			0		S:=S-Дк
	11.011000 +00.101 ——— 00.000000	0			Q:=0
		0	0	101	Ч:=ЛЗЛ(Ч),/Q
			0	1011	S:=ЛЗЛ(S),0
	00.000000	0			Лч:=Лч-1
1			1		Ч:=ЛЗЛ(Ч),Q
				10110	Лч:=Лч-1
0					Кінець

Таблиця 4.8.3

Лч	S	Зн S	Q	Ч	Наступна операція
4	00.110010	0			Q:=ЗнS#ЗнДк
		0	0		Ч:=ЛЗЛ(Ч),Q
				0	S:=S-Дк
	00.110010 +11.011 ——— 00.001010	 0			 Q:=ЗнS#ЗнДк
		0	0	0	Ч:=ЛЗЛ(Ч),/Q
			0	01	S:=ЛЗЛ(S),0
	00.010100	0			Лч:=Лч-1
3			0		S:=S-Дк
	00.010100 +11.011 ——— 11.101100	 1			 Q:=ЗнS#ЗнДк
		1	1	01	Ч:=ЛЗЛ(Ч),/Q
			1	010	S:=ЛЗЛ(S),0
	11.011000	1			Лч:=Лч-1
2			1		S:=S+Дк
	11.011000 +00.101 ——— 00.000000	 0			 Q:=0
		0	0	010	Ч:=ЛЗЛ(Ч),/Q
			0	0101	S:=ЛЗЛ(S),0
	00.000000	0			Лч:=Лч-1
1			0	0101	Ч:=ЛЗЛ(Ч),Q
			0	01010	Лч:=Лч-1
0					Кінець

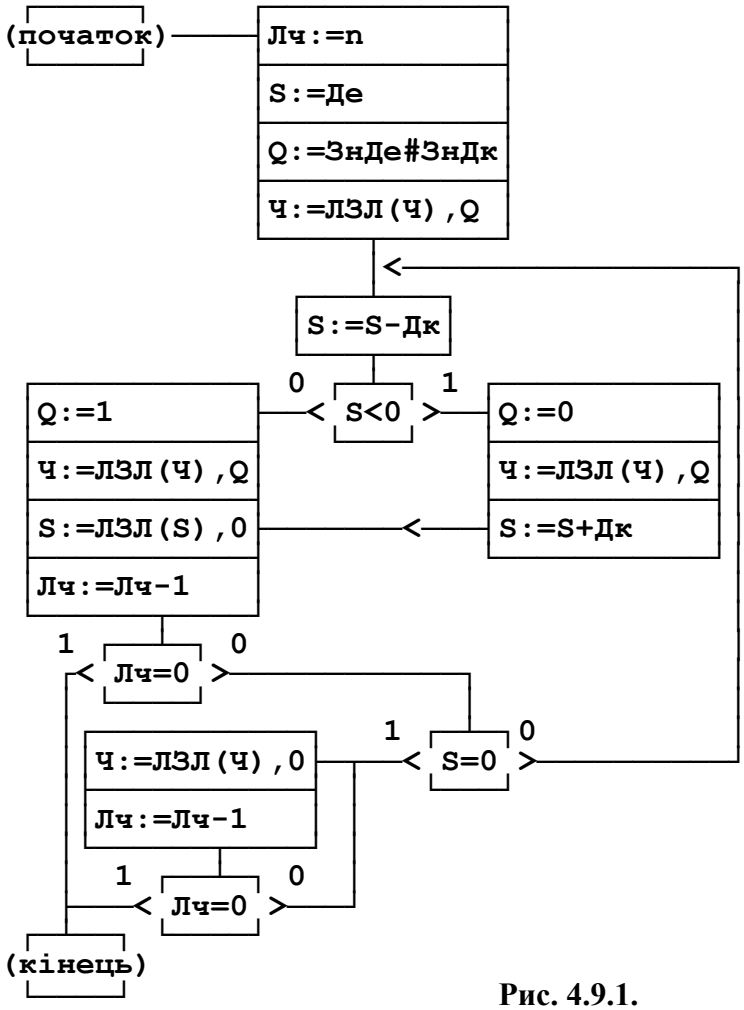


Рис. 4.9.1.

Таблиця 4.9.1

Лч	S	S<0 ?	Q	Ч	Наступна операція
4	0.101101				Q:=ЗнДе#ЗнДк
			1		Ч:=ЛЗЛ(Ч),Q
				1	S:=S-Дк
	0.101101 +1.011 0.000101	S>0			Q:=1
			1	1	Ч:=ЛЗЛ(Ч),Q
			1	11	S:=ЛЗЛ(S),0
	0.001010				Лч:=Лч-1

Продовження табл. 4.9.1

Лч	S	S<0 ?	Q	Ч	Наступна операція
3			1		S:=S-Дк
	0.001010 +1.011 1.100010	S<0			Q:=0
			0	11	Ч:=ЛЗЛ(Ч),Q
			0	110	S:=S+Дк
	1.100010 +0.101 0.001010				S:=ЛЗЛ(S),0
	0.010100				Лч:=Лч-1
2					S:=S-Дк
	0.010100 +1.011 1.101100	S<0			Q:=0
			0	110	Ч:=ЛЗЛ(Ч),Q
			0	1100	S:=S+Дк
	1.101100 +0.101 0.010100				S:=ЛЗЛ(S),0
	0.101000				Лч:=Лч-1
1					S:=S-Дк
	0.101000 +1.011 0.000000	S=0			Q:=1
			1	1100	Ч:=ЛЗЛ(Ч),Q
				11001	S:=ЛЗЛ(S),0
	0.000000				Лч:=Лч-1
0					Кінець

Таблиця 4.9.2

Лч	S	S<0	Q	Ч	Наступна операція
4	0.110010				Q:=ЗнS#ЗнДк
			1		Ч:=ЛЗЛ(Ч),Q
				1	S:=S-Дк
	0.110010 +1.011 0.001010	S>0			Q:=1
			1	1	Ч:=ЛЗЛ(Ч),Q
			1	11	S:=ЛЗЛ(S),0
	0.010100				Лч:=Лч-1
3			1		S:=S-Дк
	0.010100 +1.011 1.101100	S<0			Q:=0
			1	11	Ч:=ЛЗЛ(Ч),Q
			1	110	S:=S+Дк
	1.101100 +0.101 0.010100				S:=ЛЗЛ(S),0
	0.101000				Лч:=Лч-1
2					S:=S-Дк
	0.101000 +1.011 0.000000	S=0		110	Ч:=ЛЗЛ(Ч),Q
			1	1101	S:=ЛЗЛ(S),0
	0.000000	S=0			Лч:=Лч-1
1				1101	Ч:=ЛЗЛ(Ч),0
				11010	Лч:=Лч-1
0					Кінець

4.10 Операції над числами з рухомою комою

У загальному випадку число у формі з рухомою комою (інша назва – з комою, що плаває) представляється у вигляді $A = \pm m \cdot q^{\pm p}$, де *m* - мантиса числа, *q* - основа системи числення, $q^{\pm p}$ - порядок числа, що для спрощення в прикладах будемо іноді зображати як ±P. Тоді очевидно, що *p* - це показник степеня порядку, який звичайно називають просто порядком числа, в основному завжди *q* = 2. Отже, попередній вираз можна записати як $A = \pm m_A \cdot \pm P_A$, маючи на увазі, що в комп'ютерах звичайно *q* = 2.

Для забезпечення однозначного і максимально точного представлення чисел прийнято представляти число з рухомою комою в так званому нормалізованому вигляді. Якщо виконується нерівність $q^{-1} \le |m| < 1$, а у випадку двійкової системи числення $0.5 \le |m| < 1$ (старший двійковий розряд мантиси дорівнює 1), то вважається, що число представлене в нормалізованому вигляді.

Таким чином, у двійкового нормалізованого числа у формі з рухомою комою мантиса - правильний дріб і в старшому розряді мантиси завжди стоїть 1. Операція приведення числа до нормалізованого вигляду називається нормалізацією. Нормалізація чисел у комп'ютері виконується або автоматично або ж спеціальною програмою.

Оскільки система числення для заданого цифрового автомата (комп'ютера) є постійною, то при представленні числа у форматі з рухомою комою немає необхідності

S_p	p	S_m	m
S_m	r	m	

вказувати її основу, досить лише представити показник степеня порядку числа.

Для представлення двійкового числа у формі з рухомою комою у розрядній сітці, виділеній для цієї мети, приділяється по одному розряду для представлення знака числа S_m (поле знаку числа) і знака показника степеня порядку S_P (поле знаку порядку); певне число розрядів для представлення значення самого *показника* p (поле показника порядку), а також розряди для розміщення значення модуля *мантиси* m (поле мантиси). Наприклад, можливий наступний варіант, коли формат числа складається з чотирьох полів:

тобто, $[A] = S_p p_A S_m m_A$.

Звичайно у форматі з рухомою комою замість *показника* p використовують так звану характеристику («зміщений порядок») $r = \pm p + l$, де *l* - надлишок (зсув), значення якого підбирається таким чином, щоб при зміні значення показника від деякого мінімального значення $-|p_{min}|$ до максимального $+|p_{max}|$, характеристика *r* мінялася від 1 до r_{max}. Отже, характеристика не змінює свого знака. У такому випадку відпадає необхідність у відображенні знака порядку S_p. Для цього приймається, що $l = 2^{k-1}$, де *k* - число розрядів, виділених для представлення порядку числа у форматі з рухомою комою.

Тоді формат числа з рухомою комою можна представити у такий спосіб (з використанням трьох полів):

тобто, $[A] = S_m r m_A$.

Одиниця самого старшого розряду нормалізованої мантиси звичайно є «схованою», тобто, є уявною і не відображається у форматі числа. Розряд слова, у якому повинна була бути відображена ця одиниця, використовується як молодший розряд характеристики, що дозволяє збільшити діапазон представлення чисел у форматі з рухомою комою.

Таким чином, мантиса в такому варіанті відображається, починаючи з розряду, що йде після самого старшого. При всіх операціях з мантисою числа цю обставину треба враховувати і перед початком операцій відновлювати старший розряд мантиси.

Після завершення операцій під час формування відображення нормалізованого результату у відведеній для нього розрядній сітці машинних слів, старша одиниця мантиси знову відкидається.

n-розрядне поле порядку дозволяє змінювати показник порядку в межах від $-2^{n-1}+1$ до $+2^{n-1}-1$ (табл. 4.10.1, n=3).

Як уже відзначалося, характеристика *r* - це показник порядку *p* з надлишком $l = 2^{n-1}$, вона не змінює свого знака і змінюється від 1 (при $p = -2^{n-1}$) до 2^n-1 (при $p = +2^{n-1}-1$).

Таблиця 4.10.1

Показник порядку	Прямий код показника	Характеристика (покзник+4)	Примітки
+3	011	111	3+4=7
+2	010	110	2+4=6
+1	001	101	1+4=5
0	000	100	0+4=4
-1	101	011	-1+4=3
-2	110	010	-2+4=2
-3	111	001	-3+4=1
		000	Ознака нуля

Виключення складає число 0: нуль виражається нульовою характеристикою і (не обов'язково) нульовою мантисою.

Основною перевагою представлення чисел у формі з рухомою комою є великий діапазон машинних чисел і висока точність їхнього представлення. Діапазон визначається довжиною розрядної сітки, виділеної під характеристику, а точність - визначається довжиною розрядної сітки, виділеної під мантису.

Для формату чисел з рухомою комою, наведеного у табл. 4.10.2, значення деяких кодів наведено у табл. 4.10.3.

Перед описом алгоритмів виконання арифметичних операцій над числами з рухомою комою треба попередньо описати деякі особливості цих чисел:

збільшення мантиси в 2 рази здійснюється зсувом двійкового значення мантиси ліворуч (у бік старших розрядів);

зменшення мантиси в 2 рази здійснюється зсувом двійкового значення мантиси праворуч (у бік молодших розрядів);

величина числа не зміниться, якщо збільшити мантису в 2 рази і одночасно зменшити показник порядку на 1;

величина числа не зміниться, якщо зменшити мантису в 2 рази і одночасно збільшити показник порядку на 1.

При виконанні арифметичних операцій над числами з рухомою комою виконуються дії як над порядком, так і над мантисою.

Номер розряду	Призначення поля	Примітки
7	знак числа	0 – число додатнє; 1 – число від'ємне
6..4	порядок, зміщений на 4	ознака рівності числа нулю – усі розряди порядка дорівнюють 0
3..0	мантиса числа	мантиса 5-розрядна, нормалізована, старший розряд мантиси завжди дорівнює 1 й у форматі не відображається

Далі під додаванням буде матися на увазі операція додавання операндів з однаковими знаками, а під відніманням - операція додавання операндів з різними знаками.

При виконанні додавання порядки операндів вирівнюються, а мантиси додаються. Порядки вирівнюються збільшенням порядку меншого операнда до значення порядку більшого операнду. Цей порядок є порядком результату. Щоб при вирівнюванні порядків величина операнда не мінялася, його мантиса одночасно зменшується. Після виконання вирівнювання порядків виникають додаткові молодші розряди мантиси, які після додавання відкидаються з використанням операції заокруглення результату.

Таблиця 4.10.2

Таблиця 4.10.3

Число (16-кова система числення)	7f	70	10	0X	90	ff
Число (2-ва система числення)	01111111	01110000	00010000	0000xxxx	10010000	11111111
Число у форматі з рухомою комою розбите на поля	0.111.1111	0.111.0000	0.001.0000	0.000.xxxx	1.001.0000	1.111.1111
Знак	+ (0)	+ (0)	+ (0)	+ (0)	- (1)	- (1)
Характеристика	7=111	7=111	1=001	0=000	1=001	7=111
Показник порядку	3=7-4	3=7-4	-3=1-4	не задіяний	-3=1-4	3=7-4
Порядок	$8=2^3$	$8=2^3$	$1/8=2^{-3}$	не задіяний	$1/8=2^{-3}$	$8=2^3$
Мантиса	31/32= =0,11111	1/2= =0,10000	1/2= =0,10000	не задіяна	1/2= =0,10000	31/32= =0,11111
Величина числа	$+8*31/32= =+7^3/_4$	+8*1/2=+4	+1/8*1/2= =+1/16	0	-1/8*1/2= =-1/16	$-8*31/32= =-7^3/_4$
Примітка	Найбільше додатнє число		Найменше додатнє число	Число 0	Найменше за модулем від'ємне число	Найбільше за модулем від'ємне число

Приклад 4.10.1. Додати два числа у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=50, шістнадцятковий код числа B=60.

Процес додавання ілюструється табл. 4.10.4. У таблиці формат результату такий самий, як і формат операндів. Він відтворюється після виконання додавання і заокруглення. Результат додавання A+B=1+2=3. В обраному форматі шістнадцятковий код результату записується як 68.

Після виконання додавання може відбутися переповнення мантиси, яке усувається зменшенням мантиси в 2 рази з відповідним збільшенням порядку результату.

Приклад 4.10.2. Додати два числа у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=50, шістнадцятковий код числа B=68.

Процес додавання ілюструється табл. 4.10.5. У таблиці формат результату відтворюється після виконання додавання, усунення переповнення мантиси і заокруглення. Результат додавання A+B=1+3=4. В обраному форматі шістнадцятковий код результату записується як 70.

Після виконання додавання може відбутися переповнення порядку, яке є фатальною помилкою.

Приклад 4.10.3. Додати два числа у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=74, шістнадцятковий код числа B=68.

Процес додавання ілюструється табл. 4.10.6. У таблиці формат результату відтворюється після виконання додавання, усунення переповнення мантиси і заокруглення. Результат додавання – фатальна помилка. В обраному форматі результат записати неможливо – для запису порядку не вистачає розрядів обраного формату.

При виконанні віднімання порядки операндів вирівнюються, а мантиси віднімаються. Порядки вирівнюються збільшенням порядку меншого операнда до значення порядку більшого операнду. Цей порядок є порядком результату. Щоб при цьому величина операнда не мінялася, його мантиса одночасно зменшується. Після виконання вирівнювання порядків виникають додаткові молодші розряди мантиси, які після віднімання відкидаються з використанням операції заокруглення результату.

Після виконання віднімання може відбутися денормалізація мантиси (старший двійковий розряд мантиси не дорівнює 1), яке усувається збільшенням мантиси в 2 рази з відповідним зменшенням порядку результату (процес нормалізації результату).

Приклад 4.10.4. Виконати віднімання A-B двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=68, шістнадцятковий код числа B=60.

Процес віднімання ілюструється табл. 4.10.7. У таблиці формат результату відтворюється після виконання віднімання і заокруглення. Результат віднімання A-B=3-2=1. В обраному форматі результат записується як 50 (у 16-ковій системі числення).

При виконанні віднімання чисел з рухомою комою може виникати так зване антипереповнення, коли результат за своїм абсолютним значенням менший від найменшого числа, яке можна представити у форматі з рухомою комою, але не рівний 0. Як результат в цьому випадку повинен видаватися 0.

Таблиця 4.10.4

Аналіз і підготовка операндів	Операнд A	Операнд B
Операнди у форматі з рухомою комою (шістнадцятковий код)	50	60

Операнди у форматі з рухомою комою (двійковий код)	01010000	01100000
Розбиті на поля операнди у форматі з рухомою комою	0.101.0000	0.110.0000
Знак	+ (0)	+ (0)
Характеристика	5=101	6=110
Показник порядку	1=5-4	2=6-4
Порядок	$2=2^1$	$4=2^2$
Мантиса	½=0,10000	½=0,10000
Величина числа	+2*½=+1	+4*½=+2
Примітка	A=+1	B=+2
Вирівняний порядок – порядок результату	$4=2^2$	
Мантиса, яка відповідає вирівняному порядку	¼=0,010000	
Визначення результату:	Результат:	
Знак результату	+ (0)	
Мантиса результату визначається додаванням мантис операндів у модифікованому доповняльному коді	00.10000 +00.010000 00.110000	
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.11000(0)	
Мантиса результату	¾=0,11000	
Показник порядку результату	2	
Порядок результату	$4=2^2$	
Характеристика результату	6=2+4=110	
Величина результату	+3=4*¾	
Розбитий на поля результат у форматі з рухомою комою	0.110.1000	
Результату у форматі з рухомою комою (двійковий код)	01101000	
Результату у форматі з рухомою комою (шістнадцятковий код)	68	
Примітка	A+B=1+2=3	

Таблиця 4.10.5

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	50	68
Операнди у форматі з рухомою комою (двійковий код)	01010000	01101000
Розбиті на поля операнди у форматі з рухомою комою	0.101.0000	0.110.1000
Знак	+ (0)	+ (0)
Характеристика	5=101	6=110
Показник порядку	1=5-4	2=6-4
Порядок	$2=2^1$	$4=2^2$
Мантиса	½=0,10000	¾=0,11000
Величина числа	+2*½=+1	+4*¾ =+3
Вирівняний порядок – порядок результату	$4=2^2$	
Мантиса, яка відповідає вирівняному порядку	¼=0,010000	
Примітка	A=+1	B=+3
Визначення результату:	Результат:	
Знак результату	+ (0)	

Мантиса результату визначається додаванням мантис операндів у модифікованому доповняльному коді	00.11000 +00.010000 01.000000
Усунення переповнення мантиси результату діленням мантиси на 2 (зсувом праворуч на 1 розряд)	00.1000000
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.10000(00)
Мантиса результату	½=0,10000
Корекція показника порядку результату збільшенням на 1 після усунення переповнення мантиси	2+1=3
Порядок результату	$8=2^3$
Величина результату	+4=+8*1/2
Показник порядку результату	3
Характеристика результату	7=3+4
Розбитий на поля результат у форматі з рухомою комою	0.111.0000
Результату у форматі з рухомою комою (двійковий код)	01110000
Результату у форматі з рухомою комою (шістнадцятковий код)	70
Примітка	A+B=1+3=4

Таблиця 4.10.6

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	74	68
Операнди у форматі з рухомою комою (двійковий код)	01110100	01101000
Розбиті на поля операнди у форматі з рухомою комою	0.111.0100	0.110.1000
Знак	+ (0)	+ (0)
Характеристика	7=111	6=110
Показник порядку	3=7-4	2=6-4
Порядок	$8=2^3$	$4=2^2$
Мантиса	5/8=0,10100	¾=0,11000
Величина числа	+8*5/8=+5	+4*¾ =+3
Вирівняний порядок – порядок результату		$2^{2+1}=2^3=8$
Мантиса, яка відповідає вирівняному порядку		3/8=0,011000
Примітка	А=+5	В=+3
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається додаванням мантис операндів у модифікованому доповняльному коді	00.011000 +00.10100 01.000000	
Усунення переповнення мантиси результату діленням мантиси на 2 (зсувом праворуч на 1 розряд)	00.1000000	
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.10000(00)	
Мантиса результату	½=0,10000	
Корекція показника порядку результату збільшенням на 1 після усунення переповнення мантиси	3+1=4	
Порядок результату	$16=2^4$	
Величина результату	+8=+16*1/2	
Характеристика результату	$4+4=8=1000_2$ (Переповнення порядку, для запису порядку потрібно 4 біти, а у форматі є тільки 3)	
Результат	Фатальна помилка	
Примітка	А+В=5+3=8>7¾ (результат більлший за найбільше число, яке можна представити у даному форматі)	

Приклад 4.10.5. Виконати віднімання А-В двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа А=21, шістнадцятковий код числа В=20.

Процес віднімання ілюструється табл. 4.10.8. У таблиці формат результату відтворюється після виконання віднімання і заокруглення. Результат віднімання А-В=3-1=2. В обраному форматі результат записується як 00 (у 16-ковій системі числення).

При виконанні множення порядки операндів додаються, а мантиси перемножуються. Після перемножування мантис виникають додаткові молодші розряди мантиси, які відкидаються з використанням операції заокруглення результату. Також може відбутися денормалізація мантиси (старший двійковий розряд мантиси не дорівнює 1), яке усувається збільшенням мантиси в 2 рази з відповідним зменшенням порядку результату (процес нормалізації результату).

Приклад 4.10.6. Виконати множення A*B двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=51, шістнадцятковий код числа B=68.

Процес множення ілюструється табл. 4.10.9. У таблиці формат результату відтворюється після виконання множення і заокруглення. Результат множення A*B=17/16*3=13/4=3,25 (точний результат A*B=3,1875, помилка винакає з-за обмеженості розрядної сітки). В обраному форматі результат записується як 6a (у 16-ковій системі числення).

Після виконання додавання порядків може відбутися переповнення порядку, яке може усунутися нормалізацією мантиси (приклад 4.10.7) або стане фатальною помилкою (приклад 4.10.8).

Приклад 4.10.7. Виконати множення A*B двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=51, шістнадцятковий код числа B=78.

Процес множення ілюструється табл. 4.10.10. У таблиці формат результату відтворюється після виконання множення і заокруглення. Результат множення A*B=17/16*6=13/2=6,5 (точний результат A*B=6,375, помилка винакає з-за обмеженості розрядної сітки). В обраному форматі результат записується як 7a (у 16-ковій системі числення).

Приклад 4.10.8. Виконати множення A*B двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, шістнадцятковий код числа A=61, шістнадцятковий код числа B=78.

Процес множення ілюструється табл. 4.10.11. У таблиці формат результату відтворюється після виконання множення і заокруглення. Результат множення A*B=17/16*6=13/2=6,5 (точний результат A*B=6,375, помилка винакає з-за обмеженості розрядної сітки). В обраному форматі результат записується як 7a (у 16-ковій системі числення).

При виконанні ділення порядки операндів віднімаються, а мантиси діляться. Після ділення мантис може відбутися переповнення мантиси, яке усувається зменшенням мантиси в 2 рази з відповідним збільшенням порядку результату. Також може відбутися денормалізація мантиси (старший двійковий розряд мантиси не дорівнює 0), яке усувається збільшенням мантиси в 2 рази з відповідним зменшенням порядку результату (процес нормалізації результату).

Після віднімання порядків може відбутися переповнення порядку, яке може усунутися нормалізацією результату або стане фатальною помилкою.

Таблиця 4.10.7

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	68	60
Операнди у форматі з рухомою комою (двійковий код)	01101000	01100000
Розбиті на поля операнди у форматі з рухомою комою	0.110.1000	0.110.0000
Знак	+ (0)	+ (0)
Характеристика	6=110	6=110
Показник порядку	2=6-4	2=6-4
Порядок	$4=2^2$	$4=2^2$
Мантиса	¾=0,11000	½=0,10000
Величина числа	+3=4*¾	+4*½=+2
Примітка	А=+3	В=+2
Визначення результату:	Результат:	
Знак результату	+ (0)	
Мантиса результату визначається відніманням мантис операндів, результат ненормалізований	00.11000 -00.100000 00.010000	
Нормалізація мантиси результату множенням на 2 (зсувом ліворуч на 1 розряд)	0.10000	
Мантиса результату	½=0,10000	
Зменшення показника порядку результату на 1 для компенсації нормалізації мантиси результату	2-1=1	
Порядок результату	$2=2^1$	
Величина результату	+2*½=+1	
Характеристика результату	5=1+4=101	
Розбитий на поля результат у форматі з рухомою комою	0.101.0000	
Результату у форматі з рухомою комою (двійковий код)	01010000	
Результату у форматі з рухомою комою (шістнадцятковий код)	50	
Примітка	А-В=3-2=+1	

Приклад 4.10.9. Виконати ділення А/В двох чисел, представлених у форматі з рухомою комою, який відповідає табл. 4.10.2, число А=61, число В=78 (у 16-ковій системі числення).

Процес ділення ілюструється табл. 4.10.12. У таблиці формат результату відтворюється після виконання ділення. Результат ділення А*В=17/16*6=13/2=6,5 (точний результат А*В=6,375, помилка винакає з-за обмеженості розрядної сітки). В обраному форматі результат записується як 7а (у 16-ковій системі числення).

При виконанні множення і ділення чисел з рухомою комою слід пам’ятати, що їхні порядки зміщені і зміщення зникає після додавання і віднімання порядків. Для відновлення формату результату треба додатково вводити змішення порядку результату після додавання і віднімання порядків.

При виконанні множення і ділення чисел з рухомою комою так само, як і при відніманні, може виникати так зване антипереповнення, коли результат за своїм

абсолютним значенням менший від найменшого числа, яке можна представити у форматі з рухомою комою, але не рівний 0. Як результат в цьому випадку повинен видаватися 0.

Таблиця 4.10.8

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	21	20
Операнди у форматі з рухомою комою (двійковий код)	00100001	00100000
Розбиті на поля операнди у форматі з рухомою комою	0.010.0001	0.010.0000
Знак	+ (0)	+ (0)
Характеристика	2=010	2=010
Показник порядку	-2=2-4	-2=2-4
Порядок	$1/4=2^{-2}$	$1/4=2^{-2}$
Мантиса	17/32= =0,10001	½=0,10000
Величина числа	+17/128=1/4*17/32	+1/4*½=+1/8
Примітка	A=17/128	A=+1/8=16/128
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається відніманням мантис операндів, результат ненормалізований	00.10001 -00.10000 00.00001	
Нормалізація мантиси результату (4 зсуви ліворуч)	0.10000	
Мантиса результату	½=0,10000	
Зменшення показника порядку результату на 4 для компенсації нормалізації мантиси результату	-2-4=-6	
Показник порядку результату	-6	
Порядок результату	$1/64=2^{-6}$	
Величина результату	+1/64*½=+1/128	
Характеристика результату	-6+4=-2 – недопустиме значення характеристики, характеристика від'ємна, антипереповнення	
Розбитий на поля результат у форматі з рухомою комою	0.000.0000 – як результат видається 0	
Результату у форматі з рухомою комою (двійковий код)	00000000	
Результату у форматі з рухомою комою (шістнадцятковий код)	00	
Примітка	A-B=17/128-16/128= =1/128<1/16 (найменше число, яке можна представити у даному форматі)	

Таблиця 4.10.9

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	51	68

Операнди у форматі з рухомою комою (двійковий код)	01010001	01101000
Розбиті на поля операнди у форматі з рухомою комою	0.101.0001	0.110.1000
Знак	+ (0)	+ (0)
Характеристика	5=101	6=110
Показник порядку	1=5-4	2=6-4
Порядок	$2=2^1$	$4=2^2$
Мантиса	17/32=0,10001	3/4=0,11000
Величина числа	+2*17/32=+17/16	+4*3/4=+3
Примітка	A=+17/16	A=+3
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається перемноженням мантис операндів, результат ненормалізований	0.10001 x 0.11000 00000 + 00000 + 00000 + 10001 +10001 0.0110011000	
Нормалізація мантиси результату (1 зсув ліворуч)	0.1100110000	
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.11001(10000) +0.00000 10000 0.11010	
Мантиса результату	13/16=0,11010	
Порядок результату визначається додаванням показників порядків операндів	1+2=3 (найбільше допустиме значення порядку в даному форматі)	
Зменшення показника порядку результату на 1 для компенсації нормалізації мантиси результату	3-1=2	
Показник порядку результату	2	
Порядок результату	$4=2^2$	
Величина результату	+4*13/16=+13/4	
Характеристика результату	2+4=6=110	
Розбитий на поля результат у форматі з рухомою комою	0.110.1010	
Результату у форматі з рухомою комою (двійковий код)	01101010	
Результату у форматі з рухомою комою (шістнадцятковий код)	6a	
Примітка	A*B=17/16*3=13/4=3,25 (точний результат A*B=3,1875, помилка винакає з-за обмеженості розрядної сітки)	

Таблиця 4.10.10

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	51	78
Операнди у форматі з рухомою комою (двійковий код)	01010001	01111000
Розбиті на поля операнди у форматі з рухомою комою	0.101.0001	0.111.1000
Знак	+ (0)	+ (0)
Характеристика	5=101	7=111
Показник порядку	1=5-4	3=7-4
Порядок	$2=2^1$	$8=2^3$
Мантиса	17/32=0,10001	3/4=0,11000
Величина числа	+2*17/32=+17/16	+8*3/4=+6
Примітка	А=+17/16	А=+6
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається перемноженням мантис операндів, результат ненормалізований	0.10001 x 0.11000 00000 + 00000 + 00000 + 10001 +10001 0.0110011000	
Нормалізація мантиси результату (1 зсув ліворуч)	0.1100110000	
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.11001(10000) +0.00000 10000 0.11010	
Мантиса результату	13/16=0,11010	
Порядок результату визначається додаванням показників порядків операндів	1+3=4 (переповнення порядку)	
Зменшення показника порядку результату на 1 для компенсації нормалізації мантиси результату	4-1=3 (відновлення допустимого значення порядку)	
Показник порядку результату	3	
Порядок результату	$8=2^3$	
Величина результату	+8*13/16=+13/2	
Характеристика результату	3+4=7=111	
Розбитий на поля результат у форматі з рухомою комою	0.111.1010	
Результату у форматі з рухомою комою (двійковий код)	01111010	
Результату у форматі з рухомою комою (шістнадцятковий код)	7a	
Примітка: А*В=17/16*6=13/2=6,5 (точний результат А*В=6,375, помилка винакає з-за обмеженості розрядної сітки)		

Таблиця 4.10.11

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	61	78
Операнди у форматі з рухомою комою (двійковий код)	01100001	01111000
Розбиті на поля операнди у форматі з рухомою комою	0.110.0001	0.111.1000
Знак	+ (0)	+ (0)
Характеристика	6=110	7=111
Показник порядку	2=6-4	3=7-4
Порядок	$4=2^2$	$8=2^3$
Мантиса	17/32=0,10001	3/4=0,11000
Величина числа	+4*17/32=+17/8	+8*3/4=+6
Примітка	A=+17/8	A=+6
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається перемноженням мантис операндів, результат ненормалізований	0.10001 x 0.11000 00000 + 00000 + 00000 + 10001 +10001 0.0110011000	
Нормалізація мантиси результату (1 зсув ліворуч)	0.1100110000	
Заокруглення мантиси результату (в дужках – розряди, які відкидаються)	0.11001(10000) +0.00000 10000 0.11010	
Мантиса результату	13/16=0,11010	
Порядок результату визначається додаванням показників порядків операндів	2+3=5 (переповнення порядку)	
Зменшення показника порядку результату на 1 для компенсації нормалізації мантиси результату	5-1=4 (допустиме значення порядку не відновлюється, фатальна помилка)	
Примітка	A*B=17/8*6 – виникає фатальна помилка, точний результат A*B=12,75 > 7¾ (результат більший за найбільше число, яке можна представити у даному форматі), помилка винакає з-за обмеженості розрядної сітки.	

Таблиця 4.10.12

Аналіз і підготовка операндів	Операнд А	Операнд В
Операнди у форматі з рухомою комою (шістнадцятковий код)	72	78
Операнди у форматі з рухомою комою (двійковий код)	01110010	01111000
Розбиті на поля операнди у форматі з рухомою комою	0.111.0010	0.111.1000
Знак	+ (0)	+ (0)
Характеристика	7=111	7=111
Показник порядку	3=7-4	3=7-4
Порядок	$8=2^3$	$8=2^3$
Мантиса	9/16=0,10010	3/4=0,11000
Величина числа	+8*9/16=+9/2	+8*3/4=+6
Примітка	A=+9/2	A=+6
Визначення результату	Результат	
Знак	+ (0)	
Мантиса результату визначається діленням мантис операндів, результат нормалізований	0,10010:0,11000=0,11 -0,1100 0,00110 -0,110 0,00000	
Мантиса результату	¾=0,11000	
Порядок результату визначається відніманням показників порядків операндів	3-3=0	
Порядок результату	$1=2^0$	
Величина результату	+1*3/4=+3/4	
Характеристика результату	0+4=4=100	
Розбитий на поля результат у форматі з рухомою комою	0.100.1000	
Результату у форматі з рухомою комою (двійковий код)	01001000	
Результату у форматі з рухомою комою (шістнадцятковий код)	48	
Примітка	A/B=3/4	

Література

1 «Логіка роботи комбінаційних цифрових вузлів». Навчальний посібник з дисципліни «Комп’ютерна логіка» для студентів усіх форм навчання напряму 6.050102 «Комп’ютерна інженерія», обсяг 102 стор, укладач — д.т.н. Глухов В.С., доц. каф. ЕОМ. Львів: НУЛП, 2014.

2 Глухов В.С., Голембо В.А. «Проектування цифрових структур». Методичні вказівки до курсової роботи з дисципліни «Прикладна теорія цифрових автоматів» для студентів базового напряму 0915 «Комп'ютерна інженерія». Львів. Інститут піддприємництва та перспективних технологій при НУ «ЛП». 2003

3 Дунець Р.Б., Кудрявцев О.Т. Арифметичні основи комп’ютерної техніки. Навчальний посібник. Інститут підприємництва та перспективних технологій при Національному університеті «Львівська політехніка». Львів. 2006. – 142 с.

4 Лукащук Л.О. Схемотехніка логічних та послідовнісних схем. – Львів. Видавництво Національного унівкерситету «Львівська політехніка», 2004. – 116 с.

5 Матвієнко М. П. Комп'ютерна логіка. Навчальний посібник. — К.: Видавництво Ліра-К, 2012. — 288 с.

6 Рицар Б.Є. Цифрова техніка : Учбовий посібник для студентів радіотехнічних спеціальностей - К.: НМК ВО, 1991.

7 Мельник А.О. Архітектура комп’ютера. Підручник. – Луцьк: Волинська обласна друкарня, 2008. – 470 с.

8 Вищенчук И.М., Черкасский Н.В. Алгоритмические операционные устройства и суперЭВМ. – К.:Тэхника, 1990. – 197 с.

9 ГОСТ 2.743-91. Обозначения условные графические в схемах. - М.: 1991.

10 Справочник по Единой системе конструкторской документации. Под редакцией Ю.И.Степанова. Издание третье, переработанное и дополненное. Харьков, «Прапор» 1981.

11 Графическое изображение электрорадиосхем: Справочник / Усатенко С.Т., Каченюк Т.К., Терехова М.В. – К.: Техніка, 1986. – 120 с., ил.

НАВЧАЛЬНЕ ВИДАННЯ

МЕТОДИЧНІ ВКАЗІВКИ
до курсової роботи
«Арифметичні та логічні основи комп’ютерних технологій»

з дисципліни

«Комп’ютерна логіка»

для студентів спеціальності 123
«Комп'ютерна інженерія»

Укладачі: Глухов Валерій Сергійович
Голембо Вадим Адольфович

Редактор

Комп’ютерне складання

Підписано до друку __.__.2021 р.
Формат 70 х 100 $^{1}/_{16}$. Папір офсетний.
Друк на різографі. Умовн. друк. арк. __ Обл.-вид. арк. ___.
Наклад ____ прим. Зам. ____.

Поліграфічний центр
Видавництва Національного університету «Львівська політехніка»
Вул. Колесси, 2, 79000, Львів

Арифметичні та логічні основи комп’ютерних технологій- Назва
Комп’ютерна логіка – дисципліна
спеціальності – Напрям або спеціальність
123 – номер напряму або спеціальності
НАВЧАЛЬНЕ ВИДАННЯ
2021 рік