МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

Опорний конспект лекцій

*(спеціальності **6.09150** – комп'ютерні системи та мережі,*

***6.091500** – спеціалізовані комп'ютерні системи)*

ТЕРНОПІЛЬ
Економічна думка
2006

Затверджено на засіданні кафедри спеціалізованих комп'ютерних систем
(протокол № 2 від 20. 09. 2006 р.).

Опорний конспект лекцій з курсу "Прикладна теорія цифрових автоматів" для студентів спеціальностей "Спеціалізовані комп'ютерні системи" та "Комп'ютерні системи та мережі" / Укл. В. В. Яцків, Н. Г. Яцків – Тернопіль: Економічна думка, 2006. – с.

Рецензенти: В. В. Кочан, к. т. н., доцент;
А. І. Сегін, к. т. н. доцент.

Відповідальний за випуск: Я. М. Николайчук, д. т. н., професор, завідувач кафедри спеціалізованих систем ТНЕУ.

ЗМІСТ

Вступ

Дисципліна “Прикладна теорія цифрових автоматів” є однією з базових у системі формування знань і вмінь інженер-системотехнік (освітньо-кваліфікаційний рівень “бакалавр” та “спеціаліст”) за спеціальностями “Спеціалізовані комп’ютерні системи” та “Комп’ютерні системи та мережі”.

Метою навчального курсу “Прикладна теорія цифрових автоматів” є вивчення методів подання чисел в ЕОМ, алгоритмів виконання основних арифметичних та логічних операцій з числами в різних системах числення, основ математичної логіки, аналізу та синтезу цифрових операційних і керуючих автоматів. Предмет “Прикладна теорія цифрових автоматів” дає змогу студентам отримати необхідні теоретичні та практичні знання і вміння для розроблення і аналізу алгоритмів переробки дискретної інформації складних процесів, складання структурних схем комбінаційних логічних схем та автоматів з пам’яттю, ефективного розв’язування практичних задач з прикладної теорії цифрових автоматів з використанням ЕОМ.

Дисципліна “Прикладна теорія цифрових автоматів” є основною в схемо- та системотехнічному напрямку підготовки бакалаврів, а також базовою для курсів “Комп’ютерна електроніка”, “Архітектура ЕОМ”, “Периферійні пристрої”, “Основи автоматизації проектування засобів ОТ”, “Архітектура обчислювальних систем”.

1. АРИФМЕТИЧНІ ОСНОВИ КОМП'ЮТЕРІВ

1.1. Принципи побудови систем числення. Позиційні системи числення.

Числова інформація наявна в комп'ютерах, характеризується:
– системою числення (двійкова, десяткова та ін.);
– видом числа (дійсні, комплексні, масиви чисел);
– типом числа (змішане, ціле, дробове);
– формою подання числа (місцем коми – з природною (змінною), фіксованою, плаваючою комами);
– розрядною сіткою і форматом числа;
– діапазоном і точністю подання чисел;
– способом кодування від'ємних чисел (прямим, оберненим та доповняльним кодами);
– алгоритмами виконання арифметичних операцій.

Системою числення називається сукупність цифр і правил для записування чисел.

Запис чисел у деякій системі числення є його кодом.

Усі системи числення поділяють на позиційні й непозиційні. Для запису чисел у позиційній системі числення використовують певну кількість графічних знаків (цифр і букв), які відрізняються один від одного. Число таких знаків q називається основою позиційної системи числення.

У комп'ютерах використовують позиційні системи з різною основою.

Система числення з основою два (цифри 0 і 1) називається двійковою, система числення з основою три (цифри 0, 1, 2) – трійковою і т. д.

У системах числення з основою меншою від десяти, використовують десяткові цифри, а для основи більши, ніж десять додають букви латинського алфавіту – A, B, C, D, E і F (табл. 1.1, 1.2).

Алфавіт систем числення

Таблиця 1.1

Основа q	Система числення	Знаки
2	Двійкова	0, 1
3	Трійкова	0, 1, 2
5	П'ятіркова	0, 1, 2, 3, 4
8	Вісімкова	0, 1, 2, 3, 4, 5, 6, 7
10	Десяткова	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
16	Шістнадцяткова	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

У позиційних системах числення значення кожної цифри визначається її зображенням і позицією в числі. Окремі позиції числа називають розрядами, а номер позиції – номером розряду.

Число розрядів у його записі є розрядністю і збігається з довжиною числа.

У непозиційних системах числення значення кожної цифри не залежить від її позиції.

Найвідомішою непозиційною системою є римська, в якій використовують сім знаків – I, V, X, L, C, D і M. Вони відповідають таким значенням (табл. 1.2).

Римська непозиційна система

Таблиця 1.2

I	V	X	L	C	D	M
1	5	10	50	100	500	1000

Наприклад: III – 3, LIX – 59, DLV – 555.

Недоліком непозиційної системи є відсутність нуля та формальних правил запису чисел і відповідно арифметичних дій з ними.

Позиційні системи числення

Таблиця 1.3

$q=10$	$q=2$	$q=16$	$q=8$	$q=5$	$q=3$
0	0 0 0 0	0	0	0	0
1	0 0 0 1	1	1	1	1
2	0 0 1 0	2	2	2	2
3	0 0 1 1	3	3	3	10
4	0 1 0 0	4	4	4	11
5	0 1 0 1	5	5	10	12
6	0 1 1 0	6	6	11	20
7	0 1 1 1	7	7	12	21
8	1 0 0 0	8	10	13	22
9	1 0 0 1	9	11	14	100
10	1 0 1 0	A	12	20	101
11	1 0 1 1	B	13	21	102
12	1 1 0 0	C	14	22	110
13	1 1 0 1	D	15	23	111
14	1 1 1 0	E	16	24	112
15	1 1 1 1	F	17	30	120

Перевагою двійкової системи є:

- простота виконання арифметичних операцій;
- наявність надійних мікроелектронних схем, призначених для зберігання значень двійкового розряду – цифр 0 або 1, з двома стійкими станами (тригери).

Двійкові цифри називають також бітами. Назву *біт* у 1946 р. запропонував видатний американський учений – статистик Дж. Тюкі.

Система числення має забезпечувати:

- можливість подання будь-якого числа в заданому діапазоні;
- однозначність, стислість запису числа і простоту виконання арифметичних операцій;
- досягнення високої швидкодії машини в процесі оброблення інформації.

Число в позиційній системі можна подати поліномом:

$$A_q = a_k \cdot q^k + a_{k-1} \cdot q^{k-1} + \cdots + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \cdots + a_{-m} \cdot q^{-m} = \sum_{i=-m}^{k} a_i \cdot q^i ,$$

де q – основа системи числення;

q^i – вага позиції;

$a_i \in \{0, 1, \ldots (q-1)\}$ – цифри в позиціях числа;

$0, 1, \ldots, k$ – номери розрядів цілої частини числа;

$-1, -2, \ldots, -m$ – номери розрядів дробової частини числа.

Позиційні системи з однаковою основою в кожному розряді називають однорідними.

Приклади запису чисел:

– двійкова система: $q = 2$; $a_i \in \{0, 1\}$,

$A_2 = 1011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 8 + 2 + 1 = 11_{10}$;

– вісімкова система: $q = 8$; $a_i \in \{0, 1, \ldots, 7\}$,

$A_8 = 425_8 = 4 \cdot 8^2 + 2 \cdot 8^1 + 5 \cdot 8^0 = 256 + 16 + 5 = 277_{10}$;

– шістнадцяткова система: $q = 16$; $a_i \in \{0, 1, \ldots, 9, A, B, C, D, E, F\}$,

$A_{16} = 4AC_{16} = 4 \cdot 16^2 + 10 \cdot 16^1 + 12 \cdot 16^0 = 1024 + 160 + 12 = 1196_{10}$.

1.2. Способи переведення чисел з однієї системи числення в іншу

Поширені два основних способи переведення числа з однієї системи числення в іншу: ***табличний*** і ***розрахунковий.***

Табличний спосіб прямого переведення базується на зіставленні таблиць відповідності чисел різних систем числення. Цей спосіб дуже громіздкий і потребує великого обсягу пам'яті для зберігання таблиці, але його можна використати для будь-яких систем числення (не тільки для позиційних).

Переведення цілих чисел з однієї позиційної системи числення в іншу

Нехай задано число A в довільній позиційній системі числення з основою ℓ і його необхідно перевести в нову систему з основою P, тобто

$$A = Q_n q^n + Q_{n-1} q^{n-1} + \ldots + Q_1 q^1 + Q_0 q^0,$$

$$Q_1 = 0 \div (q-1).$$

Здійснивши перетворення отримаємо:

$$A = Q_n P^n + Q_{n-1} P^{n-1} + \ldots + Q_1 P^1 + Q_0 P^0, \qquad (1.1)$$

де $Q_1 = 0 \div (P-1)$ – база нової системи числення.

Вираз (1.1) можна записати таким чином:

$$A = A_1 P + Q_0,$$

де $A_1 = (Q_n P^{n-1} + Q_{n-1} P^{n-2} + ... + Q_2 P + Q_1)$,

Q_0 – залишок від ділення A на P, який є цифрою молодшого розряду числа.

В результаті серії ділень вихідного числа на основу нової системи числення P знаходимо коефіцієнти:

$$A = A_1 P + Q_0;$$

$$A_1 = A_2 P + Q_1;$$

.....

$$A_{n-1} = A_n P + Q_{n-1};$$

$$A_n = 0 \cdot P + Q_n.$$

При цьому ділення продовжується доти, поки не виконуватимуться співвідношення:

$$A_n < P;\ A_{n+1} < 0.$$

Правило переведення*: щоб перевести ціле число із однієї позиційної системи числення в іншу, необхідно задане число послідовно ділити на число основи нової системи числення, записаної в числах старої (заданої) системи, до одержання частки, що дорівнює 0.*

Число в новій системі числення записується із залишків від ділення, починаючи з останнього.

Приклади переведення.

Переведемо число 25 з десяткової системи числення в двійкову.

25_{10}	2_{10}
12	1 – молодший розряд
6	0
3	0
1	1
0	1 – старший розряд

Отже, $25_{10} = 11001_2$.

Переведемо число 92 з десяткової системи числення у вісімкову.

92_{10}	8_{10}
11	4
1	3
0	1

Отже, $92_{10} = 134_8$.

Переведемо число 168 з десяткової системи числення в шістнадцяткову.

168_{10}	16_{10}
10	8
0	10 - A

Отже, $168_{10} = A8_{16}$.

При переведенні з двійкової системи числення в десяткову задане число необхідно ділити на число основи нової системи числення, тобто на 1010_2.

Оскільки виконувати ділення в двійковій системі складно, на практиці підраховують суму степенів основи 2, при яких коефіцієнти Q_i дорівнюють одиниці.

Розрахунки здійснюють в десятковій системі числення.

Наприклад:

1) перевести двійкове число 10010100 у десяткову систему $2 \to 10$:

$$A = 10010100_2;\ A = 1 \cdot 2^7 + 1 \cdot 2^4 + 1 \cdot 2^2 = 128 + 16 + 4 = 148_{10};$$

2) $8 \to 10$:

$$A = 235_8;$$

$$A = 2 \cdot 8^2 + 3 \cdot 8^1 + 5 \cdot 8^0 = 128 + 24 + 5 = 157_{10};$$

3) $16 \to 10$:

$$N = 12_{16};$$

$$A = 1 \cdot 16^1 + 6 \cdot 16^0 = 16 + 6 = 22_{10};$$

$$N = A1C_{16};\ N = A \cdot 16^2 + 1 \cdot 16 + C = 10 \cdot 256 + 16 + 12 = 2560 + 28 = 2588_{10}.$$

Переведення правильних дробів

Щоб перевести правильний дріб із одної позиційної системи в іншу, необхідно задане число послідовно множити на число основи нової системи числення, записаної в старій системі, до отримання заданої точності.

Дріб у новій системі числення записують у вигляді цілих частин добутку, починаючи з першої частини.

Наприклад: перевести правильний дріб 0,456 із десяткової системи числення в двійкову і вісімкову.

1) При переведенні з десяткової системи в двійкову слід помножити заданий дріб на 2, а у вісімкову – на 8.

Ціла частина	Дробова частина
0	456
0	912
1	824
1	648
1	296
0	592
1	184
0	368

Ціла частина	Дробова частина
	456
0	x
	8
3	648
	8
5	184
	8
1	472

Отже, одержали: $0{,}456_{10} = 0{,}0111010_2$; $0{,}456_{10} = 0{,}351_8$.

2) При переведенні з двійкової системи в десяткову слід помножити задане двійкове число, записане у двійковій системі числення (1010_2), на десять:

x 0,011101	x100,100010	x101,010100
1010	1010	1010
000000	000000	000000
011101	100010	010100
+ 000000	+ 000000	+ 000000
011101	100010	010100
100,100010	101,010100	011,100100

Одержані цілі частини переводимо у десяткову систему числення. Результат перетворення має вигляд: $0{,}0111010_2 = 0{,}453_{10}$.

Переведення неправильних дробів

При переведенні неправильних дробів необхідно окремо перевести цілу і дробову частини числа за вищеподаними правилами і записати в новій системі числення, не змінюючи розташування коми.

Переведення чисел із системи числення в систему з кратною основою

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

Відповідно для переведення числа із заданої системи числення в нову, число основи якої кратне числу основи заданої системи, необхідно кожну цифру

числа записати за допомогою m цифр у новій системі числення, якщо число основи заданої системи більше за число основи нової системи.

Наприклад, при переведенні вісімкового числа 254_8 у двійкову систему числення достатньо кожну цифру вісімкового числа записати у вигляді двійкової тріади: $8 = 2^3$, $254_8 = 010101100_2$.

При переведенні двійкового числа в шістнадцяткову систему слід кожну тетраду заданого числа записати у вигляді шістнадцяткової цифри: $2^4 = 16$, $010101100 = \text{AC}$.

Вибір системи числення для використання в ЕОМ

При виборі системи числення слід урахувати такі фактори:

1) наявність фізичних елементів, здатних відтворити символи системи;
2) економічність системи, тобто кількість елементів, необхідних для подання багаторозрядних чисел;
3) трудомісткість виконання операцій в ЕОМ;
4) швидкодію обчислювальних систем;
5) наявність формального математичного апарату для аналізу і синтезу обчислювальної системи;
6) зручність роботи людини з машиною;
7) завадостійкість кодування цифр на носіях інформації.

1.3. Арифметичні дії в q-ричній системі числення

Основними арифметичними операціями є **додавання і віднімання.** Правила виконання цих операцій у десятковій системі загальновідомі. Їх можна застосувати і до всіх інших позиційних систем числення. При цьому слід використовувати спеціальні таблиці додавання і множення для кожної системи.

Додавання

Додавання в двійковій системі

Таблиця 1.4

+	0	1
0	0	1
1	1	10

Додавання у вісімковій системі

Таблиця 1.5

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

При додаванні цифри додаються порозрядно, і якщо при цьому виникає надлишок, то його переносять уліво (формується старший розряд).

Приклад: додамо числа в різних системах числення (СЧ).

Двійкова СЧ. 10101_2+0111_2

```
  1 0 1 0 1
+   0 1 1 1
-----------
  1 1 1 0 0
      |   |   |   1+1=2=2+0
      |   |   0+1+1=2=2+0
      |   1+1+1=3=2+1
      0+0+1=1
```

```
  1 0 1 0 1
+   0 1 1 1
-----------
  1 1 1 0 0
      |   |   1+1=2=10
      |   0+1+1=2=10
      1+1+1=3=11
      0+0+1=1
```

Вісімкова СЧ. $16_8 + 7_8 = 25_8$

```
+ 16
   7
  --
  25
   |  7+6=13=8+5
   1+1=2
```

Шістнадцяткова СЧ. $D_{16} + 5_{16} = 12_{16}$

```
  D
  5
 --
 12
   |  13+5=18=16+2
```

Віднімання

Двійкова СЧ. Правила віднімання:

```
 1   1   0   10
-1  -0  -0  - 1
 0   1   0   01
```

Приклади:

```
 100    1000    100010    10011010000
-  1   - 0101  - 011101  - 01100011101
 011    0011    000101    00110110011
```

Вісімкова СЧ.

```
 16    100    125
- 7   -  1   - 46
 7     77     57
```

Шістнадцяткова СЧ.

```
 100    15A    235
-  1   - 38   - 1C9
 FF     122     6C
```

Кодовані позиційні системи числення

Таблиця 1.6

Десяткова цифра	Код 8421	Код 2421	Код 8421 + 3
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	0101	1000
6	0110	0110	1001
7	0111	0111	1010
8	1000	1110	1011
9	1001	1111	1100

Двійково-десяткові коди є надлишковими, оскільки для кодування десяткових цифр використовують тільки 10 комбінацій із 16.

Двійково-десятковий код

У двійково-десятковому (двійково-кодованому) поданні десяткового числа кожна десяткова цифра зображується тетрадою двійкових символів $a_l = a_4^i\, a_3^i\, a_2^i\, a_1^i$, де a_i – десяткова цифра i-го розряду; a_i^i – двійкова цифра i-ї тетради.

Одержаний таким чином десятковий код, кодований двійковими символами, називається Д-кодами.

Є деяка множина Д-кодів. Це зумовлено наявністю 10 дозволених із 16 можливих комбінацій, які допускає тетрада.

Наявність заборонених комбінацій у Д-кодах відрізняє їх від звичайних позиційних систем числення, в яких усі комбінації є дозволеними. З усієї множини відомих Д-кодів найпоширеніші в обчислювальній техніці отримали код Д1 прямого заміщення (система 8421) і код Д2 з надлишком 3 (система 8421+3).

Через заборону певних комбінацій при додаванні чисел у Д-кодах виникає потреба у коригуванні результату і складнощі у формуванні десяткового перенесення в наступну тетраду.

Особливості додавання чисел у кожному із Д-кодів різні.

Наприклад, задані числа:

$A = a_n\, a_{n-1}\, a_1\, a_0$;

$B = b_n\, b_{n-1} \ldots b_1\, b_0$,

де a_i, b_i – двійково-кодовані десяткові цифри (тетради).

Необхідно отримати

$A + B = C = C_{n-1}\, C_n \cdots C_1\, C_0$, при цьому $c_i = a_i + b_i + \prod_{i-1} - \prod_i \cdot p$; $c_{n+1} = \prod_n$,

де $\prod_i = \{0, 1\}$, $\prod_{i-1} = \{0,1\}$ – десяткові перенесення;

$p = 10$ – основа системи числення.

Оскільки найбільше десяткове однорозрядне число 9, то з урахуванням перенесення в даний розряд значення результату розрядного сумування перебуватиме в межах від 0 до 19. При цьому одиниця в другому розряді – це десяткове перенесення в наступну тетраду, а сума одержана в двійковому коді, який відрізняється від потрібного двійково-десяткового подання, тобто він потребує коригування.

При додаванні чисел у Д-кодах можуть виникнути такі випадки:

1) якщо $a_i + b_i + \prod_{i-1} < 10$, то виконання дій над розрядами тетради за правилами двійкової арифметики дасть змогу відразу отримати правильний результат;

2) якщо $a_i + b_i + \prod_{i-1} \geq 10$, то виникає десяткове перенесення, тому сума в даній тетраді має бути дорівнювати:

$$a_i + b_i + \prod_{i-1} - \prod_i - 10,$$

де $\prod_i = 1$.

При цьому ознакою неправильного результату в одному випадку є виникнення потетрадного перенесення $\prod_i' = 16$, в іншому – поява забороненої комбінації, якщо $15 \geq a_i + b_i + \prod_{i-1} \geq 10$.

У будь-якому із цих випадків необхідно скоригувати результат у даній тетраді введенням поправки +0110, що зумовить виникнення потетрадного перенесення і в іншому випадку.

Коригування зумовлене тим, що кожне перенесення передбачає вилучення із даної тетради 16 одиниць, а надходження в наступну тільки 10 одиниць.

Приклад: додати тетради $a_i = 1000$; $b_i = 1001$ при $\prod_{i-1} = 1$.

Так, $c_i^{'} = a_i + b_i + \prod_{i-1} = 10010$.

Отже, оскільки $\prod_i = 1$, необхідне коригування результату $c_i = 0010 + 0110 = 1000$.

$\prod_i = \prod_i^i = 1$

Приклад: додати тетради $a_i = 1000$; $b_i = 0110$ при $\prod_{i-1} = 1$.

Маємо: $c_i^{'} = a_i + b_i + \prod_{i-1} = 1111$.

Оскільки величина $c_i^{'} = 1111$ належить до заборонених комбінацій, то необхідно ввести таку поправку 0110.

Отже, а) якщо в i-й тетраді сума цифр з перенесенням із $(i-1)$-ї тетради менша, ніж 10, то додавання відбувається без поправок;

б) якщо сума цифр з перенесенням дорівнює або більша, ніж 10, то відбувається коригування результату тетради введенням поправки +0110, а перенесення яке при цьому виникло, слід додати до наступної тетради: $(i+1)-i$.

Разом з цим, якщо в кількох тетрадах, починаючи з $(i+1)$-ї, розрядна сума дорівнює 1001, то перенесення зумовлює формування забороненої комбінації в $(i+1)$-й тетраді. В результаті цього необхідне коригування, яке приведе до забороненої комбінації в $(i+2)$-й тетраді і т. д.

Приклад: додати два числа $A = 248_{10} = 0010\,0100\,1000$ і $B = 349_{10} = 0011\,0100\,1001$.

$$\begin{array}{r} +\ 0010\ 0100\ 1000 \\ 0011\ 0100\ 1001 \\ \hline 0101\ 1001\ 0001 \\ 0110 \\ \hline 0101\ 1001\ 0111 \end{array}: \; 597_{10} = 248_{10} + 348_{10}.$$

Приклад: $A = 538$, $B = 465$.

$$\begin{array}{r} +\ 0101\ 0011\ 1000 \\ 0100\ 0110\ 0101 \\ \hline 1001\ 1001\ 1101 \end{array};$$

здійснюємо коригування в молодшій тетраді:

$$\begin{array}{r} +\ 1001\ 1001\ 1101 \\ 0110 \\ \hline 1001\ 1010\ 0011 \end{array};$$

здійснюємо коригування в другій тетраді:

$$\begin{array}{r} 1001\ \ 1010\ \ 0011 \\ +\ \ \ \ \ 0110 \\ \hline 1010\ \ 0000\ \ 0011 \end{array};$$

здійснюємо коригування в старшій тетраді:

$$\begin{array}{r} 1010\ \ 0000\ \ 0011 \\ +\ \ 0110 \\ \hline 1\ 0000\ \ 0000\ \ 0011 \end{array}.$$

Контрольні запитання

1. Які системи числення називають позиційними, непозиційними?
2. Перевести задані числа з однієї системи числення в іншу.
3. Виконати арифметичні операції в різних системах числення.
4. Які фактори необхідно враховувати при виборі СЧ.

2. СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ

2.1. Подання чисел в системі залишкових класів

Результати досліджень, які проводили різні групи вчених з метою пошуку шляхів підвищення продуктивності обчислювальних засобів, методів організації ефективної системи виявлення та виправлення помилок, а також побудови надійних обчислювальних комплексів, дають змогу стверджувати, що в межах позиційних систем числення не можна очікувати принципових зрушень у цих напрямках без суттєвого збільшення робочих частот і ускладнення апаратної частини. Причина полягає в тому, що позиційні системи числення, в яких подають і обробляють інформацію в сучасних ЕОМ, мають значний недолік – наявність міжрозрядних зв'язків. Таким чином, ефективним є використання непозиційних систем числення, які позбавлені цього недоліку.

З огляду на сучасний рівень розвитку обчислювальних засобів використання непозиційних систем числення дає можливість збільшити надійність та швидкість цифрової обробки даних, увести методи контролю за правильністю виконання операцій без подальшого ускладнення апаратної частини та забезпечити необхідну точність обчислень без збільшення розрядності шини. Сучасні обчислювальні потужності дають змогу розв'язувати задачі оптимального вибору модулів системи та розрахунку відповідних вагових коефіцієнтів та базисних чисел, що відкриває нові можливості для застосування непозиційних систем числення.

Наприклад, нехай задано набір із k взаємопростих натуральних чисел $p_i \in N$, $i = \overline{1,k}$, тоді СЗК розумітимемо як таку систему, в якій ціле число подано у вигляді невід'ємних залишків за вибраними модулями p_i.

$$\text{Отже, } b_i = res\, N \,(\mathrm{mod}\, p_i),\ i = \overline{1,k}. \tag{2.1}$$

Даний вираз відповідає системі діофантових рівнянь:

$$N = c_i \cdot p_i + b_i,\ i = \overline{1,k}, \tag{2.2}$$

де N – вихідна величина; p_i – набір модулів; b_i – набір залишків за відповідними модулями; c_i – ранг числа N за модулем p_i.

У теорії чисел доведено, що система рівнянь (2.2) має єдиний розв'язок при взаємопростих модулях. Діапазон чисел, який можна подати за допомогою набору модулів (p_1, p_2, …, p_{k-1} p_k), становить $[0, \wp]$, $\wp = \prod_{i=1}^{k} p_i$.

Нехай у десятковій системі числення задано число N=13, вибираємо взаємно прості модулі: $p_1 = 3,\ p_2 = 5,\ p_3 = 7$, добуток яких дорівнює $\wp = \prod_{i=1}^{3} p_i = 3 \cdot 5 \cdot 7 = 105$.

Ураховуючи, що $N < \wp$, можна використовувати даний набір модулів для перетворення заданого числа.

Спосіб 1

За невеликого діапазону подання даних найефективнішим є табличний метод кодування та перетворення даних у СЗК .

Таблиця кодування даних у СЗК

Таблиця 2.1

Число в десятковій системі числення	$p_1 = 3$	$p_2 = 5$	$p_3 = 7$
0	0	0	0
1	1	1	1
2	2	2	2
3	0	3	3
4	1	4	4
5	2	0	5
6	0	1	6
7	1	2	0
8	2	3	1
9	0	4	2
10	1	0	3
11	2	1	4
12	0	2	5
13	1	3	6
14	2	4	0
15	0	0	1
…	…	….	….
…	…	….	….
100	1	0	2
101	2	1	3
102	0	2	4
103	1	3	5
104	2	4	6
105	0	0	0

Отже, згідно з даними таблиці 2.1: $13_{10} = (1, 3, 6)_{(3,5,7)}$.

Спосіб 2

Нехай у десятковій системі числення задано число N=103.

Використовуючи рівняння (2.1), маємо:

$$b_1 = res\,103 \pmod 3 = 1;$$
$$b_2 = res\,103 \pmod 5 = 3;$$
$$b_2 = res\,103 \pmod 5 = 5.$$

Отже, $103_{10} = (1, 3, 5)_{(3,5,7)}$.

Спосіб 3

Задано число N=103.

Число N подано в позиційній системі числення з основою $d = 10$. Подання степенів основи d у СЗК матиме такий вигляд:

$$d^0 = 1 = (1,1,1)_{(3,5,7)}\text{б}$$
$$d^1 = 10 = (1,0,3)_{(3,5,7)},$$
$$d^2 = 100 = (1,0,2)_{(3,5,7)}.$$

Таким чином, отримаємо представлення коефіцієнти полінома (2.4).

$$a_0 = 3 = (0,3,3)_{(3,5,7)},$$
$$a_1 = 0 = (0,0,0)_{(3,5,7)},$$
$$a_2 = 1 = (1,1,1)_{(3,5,7)}.$$

Згідно з формулою (2.5):

$$103_{10} = (0\cdot 1 + 0\cdot 1 + 1\cdot 1,\ 3\cdot 1 + 0\cdot 0 + 1\cdot 0,\ 3\cdot 1 + 0\cdot 3 + 1\cdot 2) = (1,3,5)_{(3,5,7)}.$$

Число $N=103_{10}$ отримано за допомогою різних методів. Вони є аналогічними, що підтверджує достовірність одержаних результатів.

Переведення числа з системи залишкових класів у десяткову систему числення

Переведення числа з системи залишкових класів у десяткову систему числення здійснюється за формулою:

$$\mathrm{N} = \sum_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{b_i} \cdot \beta_\mathrm{i} \pmod{\wp}. \qquad (2.3)$$

Згідно з визначенням ортогональних базисів, їх можна обчислити таким чином:

$$B_i = m_i \cdot \frac{\wp}{p_i},\ i = \overline{1, k}; \qquad (2.4)$$

де $1 \le m_i \le p_i - 1$ – вага ортогонального елементу.

При цьому слід зазначити, що

$$m_i \cdot \frac{\wp}{p_i} = 1 \pmod{p_i}. \tag{2.5}$$

Рівняння (5) еквівалентне такому діафантовому рівнянню:

$$m_i \cdot \frac{\wp}{p_i} = 1_i \cdot p_i + 1, \quad 1_i \in N. \tag{2.6}$$

Для обчислення m_i використовують формулу (2.5). Застосування операції визначення залишку за заданим модулем зумовлює обмежений діапазон можливих значень вагових коефіцієнтів: $m_i \in [1,\ p-1]$.

Позначимо $\wp_i = \frac{\wp}{p_i}$. Поділивши $\wp_i$ на p_i, згідно з рівнянням (2.5), отримаємо певний залишок δ_i:

$$m_i \cdot \delta_i = 1 \cdot (\mathrm{mod}\, \wp). \tag{7}$$

Зважаючи на порівняно невеликі значення величини p_i, можна скласти таблицю розв'язків рівняння (2.7). За допомогою цієї таблиці відповідно до величини δ_i знаходимо значення m_i. Припускаючи, що основи p_i вибирають з множини простих чисел, подамо таблицю розв'язків рівняння (2.7) для $p_i < 25$ (табл. 2.2).

Згідно з формулою (2.5):

$$\mathrm{B}_1 + \mathrm{B}_2 + \ldots + \mathrm{B}_\mathrm{k} = (1,0,\ldots,0) + (0,1,\ldots,0) + \ldots + (0,0,\ldots,1) = (1,1,\ldots,1). \tag{2.8}$$

Оскільки сумування здійснюють у СЗК, маємо:

$$\sum_{\mathrm{i}=1}^{\mathrm{k}} \mathrm{B_i} = 1 \pmod{\wp}. \tag{2.9}$$

Розв'язки рівняння $m \cdot \delta = 1 (\mathrm{mod}\, p)$ для множини простих чисел $p_i < 25$

Таблиця 2.2

δ	P								
	2	3	5	7	11	13	17	19	23
1	1	1	1	1	1	1	1	1	1
2		2	3	4	6	7	9	10	12
3			2	5	4	9	6	13	8
4			4	2	3	10	13	5	6
5				3	9	8	7	4	14
6				6	2	11	3	16	4
7					8	2	5	11	10
8					7	5	15	12	3
9					5	3	2	17	18
10					10	4г	12	2	7
11						6	14	7	21
12						12	10	8	2
13							4	3	16
14							11	15	5
15							8	14	20
16							16	6	13
17								9	19
18								18	9
19									17
20									15
21									11
22									22

Таким чином, рівняння (2.9) можна використати для перевірки достовірності знаходження базисів системи.

Наприклад, здійснимо зворотне перетворення для значень, отриманих вище. Маємо:

$$P_1 = 3,\ P_2 = 5,\ P_3 = 7,$$

$$a_1 = 1,\ a_2 = 3,\ a_3 = 5,$$

$$\delta_1 = 35(\mathrm{mod}\,3) = 2,$$

$$\delta_2 = 21(\mathrm{mod}\,5) = 1,$$

$$\delta_3 = 15(\mathrm{mod}\,7) = 1.$$

Використовуючи значення базисних чисел та дані табл. 2.2, отримаємо:

$$m_1 = 2;\ m_2 = 1;\ m_3 = 1;$$

$$B_1 = \frac{105}{3}\cdot 2 = 70;\ \ B_2 = \frac{105}{5}\cdot 1 = 21;\ B_3 = \frac{105}{7}\cdot 1 = 15.$$

Перевіримо достовірність обчислення базисних чисел згідно з формулою (2.9):

$$(70 + 21 + 15) = 106 = 1 \pmod{105}.$$

Відповідно до формули (2.3) маємо:

$$N_{10} = res\,(1\cdot 70 + 3\cdot 21 + 5\cdot 15) \pmod{105} = 103_{10}.$$

В результаті послідовного застосування прямого та зворотного перетворень для цілочисельної форми СЗК отримаємо вихідне число в позиційній системі числення.

Подання даних у системі залишкових класів дає змогу здійснювати розпаралелювання обробки інформації без значного ускладнення обчислювальних засобів. Використання СЗК спрощує побудову систем збору інформації, а також дає змогу розв’язувати задачі такого класу, які є невизначеними в позиційних системах числення. Особливістю СЗК вважають простоту реалізації прямого та зворотного перетворень.

2.2. Математичні операції в СЗК

Визначимо правила виконання операцій додавання і множення в СЗК за умови, що обидва числа і результат операції перебувають у діапазоні $[0, \wp]$.

Наприклад, нехай операнди A і B подано відповідно як залишки α_i і β_i за модулем P_i при $i = 1, 2, \ldots, n$. Результат операцій додавання і множення $A + B$ і

$A \cdot B$ поданий відповідними залишками γ_i і δ_i за тими самими модулями P_i, тобто:

$$A = (\alpha_1, \alpha_2, .., \alpha_n),$$
$$B = (\beta_1, \beta_2, .., \beta_n),$$
$$A + B = (\gamma_1, \gamma_2, .., \gamma_n),$$
$$A \cdot B = (\delta_1, \delta_2, .., \delta_n).$$

При цьому справджуються співвідношення:

$$A < \wp,\ B < \wp,\ A + B < \wp,\ A \cdot B < \wp.$$

Припустимо, що γ_i дорівнює $\alpha_i + \beta_i$ за модулем P_i, а $\delta_i = \alpha_i \cdot \beta_i$ також за модулем P_i.

$$\gamma_i \equiv \alpha_i + \beta_i (\mathrm{mod}\, P_i),$$
$$\delta_i \equiv \alpha_i \cdot \beta_i (\mathrm{mod}\, P_i).$$

Як цифри результату беремо відповідно:

$$\gamma_i = \alpha_i + \beta_i - [\frac{\alpha_i + \beta_i}{P_i}] \cdot P_i, \qquad (2.10)$$

$$\delta_i = \alpha_i \cdot \beta_i - [\frac{\alpha_i \cdot \beta_i}{P_i}] \cdot P_i. \qquad (2.11)$$

Отже, за умови, що $i = 1, 2, \ldots, n,$ для додавання матимемо такий вираз:

$$\gamma_i = A + B - [\frac{A + B}{P_i}] \cdot P_i,$$

$$A + B\ (\mathrm{mod}\, P) = \begin{cases} \alpha_i + \beta_i, & \text{якщо} \quad \alpha_i + \beta_i < P_i; \\ \alpha_i + \beta_i - P, & \text{якщо} \quad \alpha_i + \beta_i \geq P_i. \end{cases}$$

Для множення отримаємо:

$$\delta_i = A \cdot B - [\frac{A \cdot B}{P_i}] \cdot P_i.$$

Наприклад, основою системи є $P_1 = 3$, $P_2 = 5$ і $P_3 = 7$.

Діапазон подання чисел за допомогою обраних модулів визначемо таким чином: $\wp = P_1 \cdot P_2 \cdot P_3 = 105$.

Приклад: додати числа A=17 і B=63.

Переведемо числа A і B у систему залишкових класів за заданими модулями:

$$A = 17 = (2, 2, 3)_{(3,5,7)},$$
$$B = 63 = (0, 3, 0)_{(3,5,7)}.$$

Відповідно до формули (2.10) отримаємо:

$$A + B = (2, 0, 3)_{(3,5,7)}.$$

Можна перевірити, що число $(2, 0, 3)_{(3,5,7)}$ у десятковій системі числення є 80 і дорівнює сумі операндів.

Приклад: помножити число A=17 на B=6.

У СЗК числа A і B подано таким чином:

$$A = 17 = (2, 2, 3)_{(3,5,7)},$$
$$B = 6 = (0, 1, 6)_{(3,5,7)}.$$

Відповідно до формули (2.11) отримаємо: $A \cdot B = (0, 2, 4)_{(3,5,7)}$.

Перевіряємо, що число $(0, 2, 4)_{(3,5,7)}$ у СЗК дорівнює десятковому числу 102 у десятковій системі числення і добутку операндів.

Правила виконання операції віднімання, в СЗК, якщо два числа і результат операції перебувають у діапазоні $[0, \wp]$

Наприклад, нехай операнди A і B подані як відповідні залишки α_i і β_i за модулями P_i при $i = 1, 2, \ldots, n$.

Результат операції віднімання A–B подано як відповідні залишки γ_i за тими самими модулями P_i.

Тобто:

$$A = (\alpha_1, \alpha_2, .., \alpha_n),$$
$$B = (\beta_1, \beta_2, .., \beta_n),$$
$$A - B = (\gamma_1, \gamma_2, .., \gamma_n).$$

При цьому виконуються такі умови:

$$A < \wp,\ B < \wp,\ 0 \le A - B < \wp.$$

Аналогічно з формули (2.10) отримаємо вираз для віднімання:

$$\gamma_i = \alpha_i - \beta_i - [\frac{\alpha_i - \beta_i}{P_i}] \cdot P_i,$$
$$\gamma_i = \alpha_i - \beta_i(P_i),\ i = 1, 2, \ldots, n.$$

Операцію віднімання в тих випадках, коли її результат додатний, виконують відніманням відповідних цифр розрядів, при цьому завжди наводиться найменший додатний залишок, оскільки це випливає із визначення СЗК. Якщо різниця цифр від'ємна, то береться її доповнення до відповідного модуля.

Тобто:

$$A - B \,(\mathrm{mod}\, P) = \begin{cases} \alpha_i - \beta_i, & \textit{якщо} \quad \alpha_i - \beta_i \geq 0; \\ \alpha_i - \beta_i + P, & \textit{якщо} \quad \alpha_i - \beta_i < 0. \end{cases}$$

Приклад: виконати віднімання двох чисел у СЗК: C=A–B.

$\mathrm{A} = 17 = (2, 2, 3)_{(3,5,7)}$,

$\mathrm{B} = 6 = (0, 1, 6)_{(3,5,7)}$,

$\mathrm{C} = (2 - 0, 2 - 1, 3 - 6 + 7) = (2, 1, 4)_{(3,5,7)}$.

$\mathrm{C} = 11 = (2, 1, 4)_{(3,5,7)}$.

У результаті послідовного застосування прямого та зворотного перетворень для цілочисельної форми СЗК отримаємо вихідне число в позиційній системі числення.

Подання даних у системі залишкових класів дає змогу здійснювати розпаралелювання обробки інформації без значного ускладнення обчислювальних засобів. Використання СЗК спрощує побудову систем збору інформації, а також дає можливість розв'язувати задачі такого класу, які є невизначеними в позиційних системах числення. Особливістю СЗК вважають простоту реалізації прямого та зворотного перетворень.

Контрольні запитання

1. Назвати переваги та недоліки СЗК?
2. Перевести задані числа з десяткової СЧ у СЗК.
3. Перевести задані числа з СЗК у десяткову СЧ.
4. Виконати арифметичні операції в СЗК.

3. ЕЛЕМЕНТИ МАТЕМАТИЧНОЇ ЛОГІКИ

3.1. Перемикальні функції
3.2. Булева алгебра одного, двох аргументів
3.3. Закони алгебри-логіки

Теоретичною основою цифрових автоматів є алгебра логіки – наука, яка використовує математичні методи для розв'язування логічних задач. Алгебру логіки називають булевою на честь англійського математика Дж. Буля, який зробив значний вклад у розвиток цієї науки (1815–1864 рр.).

Основним предметом булевої алгебри служить – просте твердження: воно або істинне (позначають символом 1), або хибне (позначають символом 0).

Прості висловлювання позначають буквами, наприклад $X_1,\ X_2,\ \ldots X_m$, які у цифровій техніці називають змінними (аргументами).

За допомогою логічних зв'язок НЕ, АБО, І, ЯКЩО.., ТО будують складні висловлювання, які називають (логічними) функціями і позначають буквами F, L, K, M, P та ін.

Нині головне завдання алгебри-логіки – аналіз, синтез і структурне моделювання будь-яких дискретних скінченних систем.

Змінну із скінченним числом значень (станів) називають *перемикальною*, а з двома значеннями – *булевою.*

Функція, яка має, як і кожна її змінна, скінченне число значень, називається перемикальною (логічною).

Логічна функція, число можливих значень якої та кожної її незалежної змінної дорівнює двом, є булевою. Таким чином, булева функція – це окремий випадок перемикальної.

Операція – це чітко визначена дія над одним або кількома операндами, яка створює новий об'єкт (результат).

У булевій операції операнди і результат набувають "булевого значення 1" і "булевого значення 0".

Булеву операцію над одним операндом називають одномісною, над двома – двомісною тощо.

Булеві функції можуть залежати від однієї, двох і n-змінних.

Запис $F(X_1, X_2, \ldots X_n)$ означає, що деяка булева функція F залежить від змінних $X_1, X_2, \ldots X_n$.

Основними булевими операціями є ***заперечення*** (операція НЕ, інверсія), ***диз'юнкція*** (операція АБО, логічне додавання, об'єднання) і ***кон'юнкція*** (операція І, логічне множення).

Заперечення – це одномісна булева операція $F = \bar{x}$ (читається "не х"), результатом якої є значення, протилежне значенню операнда.

Диз'юнкція – це булева операція $F = x_1 \vee x_2$ (читається x_1 або x_2), результатом якої є значення нуль тоді і тільки тоді, коли обидва операнди мають значення нуль.

Кон'юнкція – це булева операція $F = x_1 \wedge x_2$ (читається x_1 і x_2), результатом якої є значення одиниці тоді і тільки тоді, коли значення кожного операнда дорівнює одиниці у виразі $x_1 \wedge x_2$.

Операції заперечення, диз'юнкції і кон'юнкції можна задати за допомогою таблиць істинності, в яких зліва подані значення операндів, а справа – значення булевої функції.

Таблиці істинності операцій заперечення, диз'юнкції і кон'юкції

Таблиця 3.1

x	$F = \bar{x}$
0	1
1	0

x_1	x_2	$F = x_1 \vee x_2$	$F = x_1 \wedge x_2$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

Для булевих операцій заперечення, диз'юнкції та кон'юнкції справджуються такі закони, властивості й тотожності:

1) комутативність:

$x \vee y = y \vee x$, $x \wedge y = y \wedge x$;

2) асоціативність:

$x \vee (y \vee z) = (x \vee y) \vee z$, $x \wedge (y \wedge z) = (x \wedge y) \wedge z$;

3) дистрибутивність:

$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$, $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$;

4) ідемпотентність:

$x \vee x \vee x \vee \ldots = x$, $x \wedge x \wedge x \wedge \ldots = x$;

5) закон поглинання:

$x \vee (x \wedge y) = x$, $x \wedge (x \vee y) = x$;

6) закон склеювання:

$(x \vee \bar{y}) \wedge (x \vee y) = x$, $(x \wedge \bar{y}) \vee (x \wedge y) = x$;

7) закон де Моргана:

$\overline{x \vee y} = \bar{x} \wedge \bar{y}$, $\overline{x \wedge y} = \bar{x} \vee \bar{y}$;

8) властивості заперечення і константи:

$$x \vee \bar{x} = 1, \qquad x \wedge \bar{x} = 0, \qquad x \wedge 0 = 0, \qquad \bar{1} = 0,\ \bar{0} = 1,$$
$$x \vee 0 = x, \qquad x \wedge 1 = x, \qquad x \vee 1 = 1, \qquad \bar{\bar{x}} = x.$$

Виконання зазначених законів булевої алгебри перевіряють шляхом підстановки в логічний вираз нуля і одиниці (див. табл. 3.2.).

Таблиця істинності логічних функцій

Таблиця 3.2

x	y	$x \wedge y$	$\overline{x \wedge y}$	$\bar{x}$	$\bar{y}$	$\bar{x} \vee \bar{y}$
0	0	0	1	1	1	0
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	1

Областю визначення булевої функції $F(x_1, x_2, \ldots, x_n)$ є скінченна множина різних двійкових наборів довжиною n, на кожному з яких указується значення функції – нуль або одиниця.

Кількість різноманітних двійкових наборів дорівнює множині n-розрядних двійкових чисел $m = 2^n$.

Наприклад, для функції двох змінних x і y є чотири двійкових набори: 00; 01; 10; 11.

Дві функції відрізняються одна від одної, якщо їхні значення будуть різними хоча б на одному наборі.

Число різноманітних булевих функцій від n-змінних дорівнює 2^m, де $m = 2^n$.

Довільну булеву функцію можна задати різними способами: часовими діаграмами, геометричними фігурами, графами, таблицями істинності та аналітичними виразами.

Словесний опис деякої булевої функції $F(x, y)$ можна подати так: $F = 1$ при $x \wedge y = 1$, і $F = 0$, якщо $x \wedge y = 0$. Таку функцію можна зобразити часовою діаграмою або геометрично за допомогою двовимірного куба, в якому точками визначені одиничні вершини, а також графом, де вершини відображають значення нуля і одиниці, а на орієнтованих дугах змінні вказують на умови переходу (рис. 3.1).

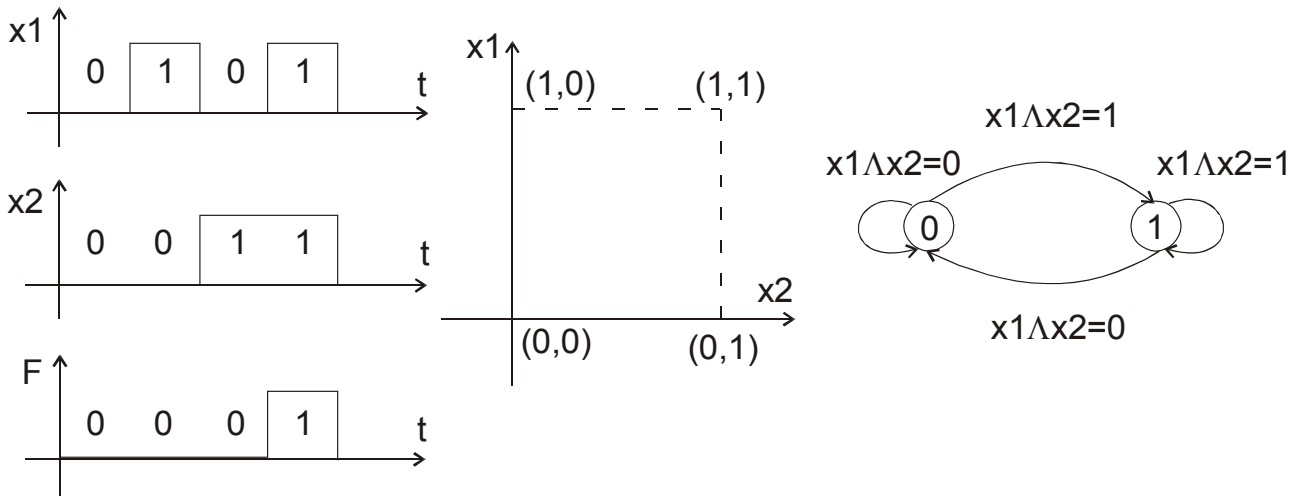


Рис. 3.1. Способи зображення булевої функції.

За допомогою таблиці істинності відображають усі можливі функції однієї (всього чотири функції) і двох змінних (усього 16 функцій).
Для $n=3$ число можливих булевих функцій дорівнює 256, для $n=4$ їхня кількість – $2^{16}=65536$.

Булеві функції однієї змінної

Таблиця 3.3

x		Вираз	Назва функції
0	1		
0	0	$F_0=0$	Константа –0
0	1	$F_1=x$	Повторення
1	0	$F_2=\overline{x}$	Заперечення
1	1	$F_3=1$	Константа –1

Еквівалентність (рівнозначність) – двомісна булева операція, результатом якої є одиниця тоді і тільки тоді, коли операнди набувають однакових значень.

Імплікація (включення) – двомісна булева операція, результатом якої є значення нуль тоді і тільки тоді, коли значення одного з операндів дорівнює нулю, а іншого – одиниці. Наприклад:

$$f_{11}=x_1\leftarrow x_2=x_1\vee\overline{x_2}\ ;$$
$$f_{13}=x_1\rightarrow x_2=\overline{x_1}\vee x_2.$$

Виключення (заборона) – двомісна булева операція, результатом якої є значення одиниця тоді і тільки тоді, коли значення одного операнда дорівнює одиниці, а іншого – нулю. Паприклад:

$$f_2 = x_1 \wedge \overline{x_2};$$
$$f_4 = \overline{x_1} \wedge x_2.$$

Булеві функції двох змінних подано в (табл. 3.2.)

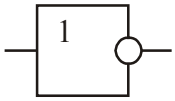
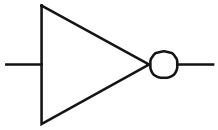
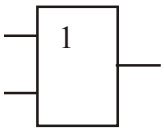
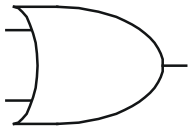
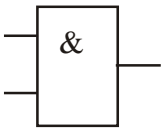
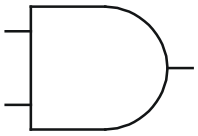
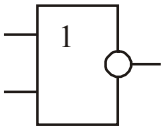
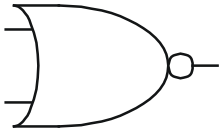
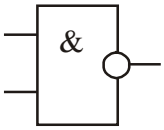
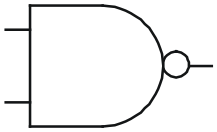
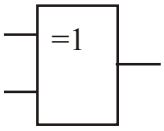
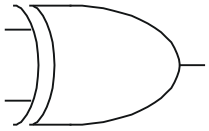
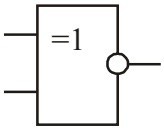
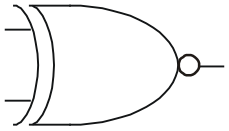
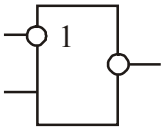
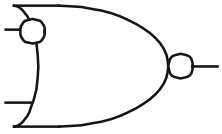
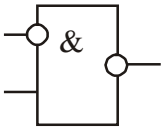
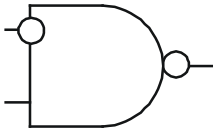
Структурні формули та назви логічних функцій

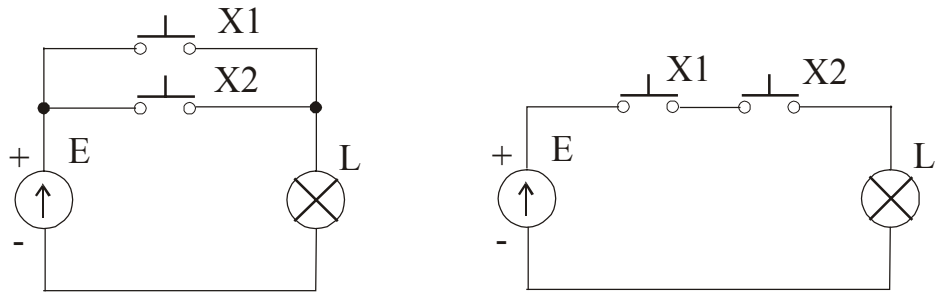
Таблиця 3.4

Аргументи				Функція	Назва логічної функції
x_1 0	0	1	1		
x_2 0	1	0	1		
0	0	0	0	$f_0 = 0$	Константа 0
0	0	0	1	$f_1 = x_1 \wedge x_2$	Кон'юнкція (операція І)
0	0	1	0	$f_2 = x_1 \wedge \overline{x_2}$	Заборона за x_2
0	0	1	1	$f_3 = x_1$	Повторення (тавтологія) x_1
0	1	0	0	$f_4 = \overline{x_1} \wedge x_2$	Заборона за x_1
0	1	0	1	$f_5 = x_2$	Повторення (тавтологія) x_2
0	1	1	0	$f_6 = x_1 \oplus x_2 =$ $= (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_2})$	Виключає АБО (додавання за модулем 2)
0	1	1	1	$f_7 = x_1 \vee x_2$	Диз'юнкція (операція АБО);
1	0	0	0	$f_8 = x_1 \downarrow x_2 =$ $= \overline{x_1 \vee x_2} = \overline{x_1} \wedge \overline{x_2}$	Стрілка Пірса (операція АБО-НЕ)
1	0	0	1	$f_9 = x_1 \sim x_2 =$ $= (x_1 \wedge x_2) \vee (\overline{x_1} \wedge \overline{x_2})$	Еквівалентність
1	0	1	0	$f_{10} = \overline{x_2}$	Заперечення (інверсія) x_2
1	0	1	1	$f_{11} = x_1 \leftarrow x_2 =$ $= x_1 \vee \overline{x_2}$	Імплікація від x_2 до x_1
1	1	0	0	$f_{12} = \overline{x_1}$	Заперечення (інверсія x_1)
1	1	0	1	$f_{13} = x_1 \rightarrow x_2 =$ $= \overline{x_1} \vee x_2$	Імплікація від x_1 до x_2
1	1	1	0	$f_{14} = x_1 \mid x_2 =$ $= \overline{x_1 \wedge x_2} = \overline{x_1} \vee \overline{x_2}$	Штрих Шеффера (операція І-НЕ)
1	1	1	1	$f_{15} = 1$	Константа 1

Графічні позначення логічних елементів

Таблиця 3.5

Назва операції	Назва елементу	Умовне графічне позначення	
Заперечення	НЕ		
Диз'юнкція	АБО		
Кон'юнкція	І		
Заперечення диз'юнкції	АБО-НЕ		
Заперечення кон'юнкції	І-НЕ		
Еквівалентність	Виключає АБО		
Заперечення еквівалентності	Еквівалентність		
Імплікація	ЯКЩО.., ТО		
Заборона	ЗАБОРОНА		



а) б)

Рис. 3.2. Схеми заміщення логічних елементів: а) логічний елемент **АБО**; б) логічний елемент **І**.

Контрольні запитання

1. Дати визначення логічних функцій: інверсії, диз'юнкції, кон'юнкції.
2. Зобразити умовні графічні позначення логічних елементів.
3. Записати таблиці істинності заданих логічних елементів.
4. Визначити вихідний стан логічних елементів при заданих вхідних сигналах.
5. Визначити якому логічному елементові належить таблиця істинності.
6. Які логічні елементи можна використати як інвертори?
7. Записати закони алгебри логіки.
8. Спростити логічний вираз:

8.1) $y = x_1 \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3 \vee \overline{x_1} \wedge x_2 \wedge \overline{x_3} \vee \overline{x_1} \wedge x_2 \wedge x_3$;.

8.2) $y = x_1 \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3 \vee x_1 \wedge \overline{x_2} \wedge x_3 \vee x_1 \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.3) $y = \overline{x_1} \wedge x_2 \wedge \overline{x_3} \vee \overline{x_1} \wedge x_2 \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.4) $y = x_1 \wedge \overline{x_2} \wedge x_3 \vee x_1 \wedge \overline{x_2} \wedge \overline{x_3} \vee \overline{x_1} \wedge \overline{x_2} \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.5) $y = x_1 \wedge x_2 \wedge \overline{x_3} \vee \overline{x_1} \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge \overline{x_2} \wedge \overline{x_3} \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.6) $y = x_1 \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.7) $y = \overline{x_1} \wedge x_2 \wedge \overline{x_3} \vee \overline{x_1} \wedge x_2 \wedge x_3 \vee x_1 \wedge \overline{x_2} \wedge x_3 \vee x_1 \wedge \overline{x_2} \wedge \overline{x_3}$;.

8.8) $y = x_1 \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3 \vee \overline{x_1} \wedge x_2 \wedge x_3 \vee \overline{x_1} \wedge \overline{x_2} \wedge x_3$.

4. МІНІМІЗАЦІЯ ПЕРЕМИКАЛЬНИХ ФУНКЦІЙ

4.1. Мінімізація булевих функцій

4.2. Діаграми Вейча та карти Карно для мінімізації перемикальних функцій

4.3. Мінімізація неповністю визначених перемикальних функцій

Важливим етапом проектування цифрових пристроїв є мінімізація булевих функцій, тобто знаходження їхніх виражень з мінімальною кількістю букв.

Мінімізація забезпечує побудову економічних схем цифрових автоматів. Для мінімізації функцій із кількістю букв $n \le 6$ застосовують карти Карно. Їх будують у вигляді таблиць з 2^n кліток із розмічуванням рядків і стовпчиків змінними.

Карти Карно для функцій з трьома змінними $F(x_1, x_2, x_3)$.

Таблиця 4.1

x_1 \ $x_2\,x_3$	00	01	11	10
0	000	001	011	010
1	100	101	111	110

x_1 \ $x_2\,x_3$	00	01	11	10
0	$\overline{x_1}\,\overline{x_2}\,\overline{x_3}$	$\overline{\mathbf{x_1}}\,\overline{\mathbf{x_2}}\,\mathbf{x_3}$	$\overline{\mathbf{x_1}}\,\mathbf{x_2}\,\mathbf{x_3}$	$\overline{\mathbf{x_1}}\,\mathbf{x_2}\,\overline{\mathbf{x_3}}$
1	$\mathbf{x_1}\,\overline{\mathbf{x_2}}\,\overline{\mathbf{x_3}}$	$\mathbf{x_1}\,\overline{\mathbf{x_2}}\,\mathbf{x_3}$	$\overline{\mathbf{x_1}}\,\overline{\mathbf{x_2}}\,\overline{\mathbf{x_3}}$	$\mathbf{x_1}\,\mathbf{x_2}\,\overline{\mathbf{x_3}}$

Мінтерми в сусідніх клітинках карти Карно в рядку (з урахуванням верхніх і нижніх) або стовпчику (з урахуванням крайніх) розрізняють за значеннями однієї змінної, що дає змогу виконувати операцію склеювання за цією змінною.

Загальні правила мінімізації:

1) зображають карту Карно для n змінних і розмічують її рядки і стовпчики. У клітинки таблиці, які відповідають мінтермам (одиничним наборам) функції, яка мінімізується, записують одиницю;

2) склеюванню підлягають прямокутні конфігурації, заповнені одиницями, які і містять 2, 4 або 8 клітинок. Верхні й нижні рядки, крайні ліві і праві стовпчики карти ніби склеюються, створюючи поверхню циліндра;

3) множину прямокутників, які покривають усі одиниці, називають покриттям. Чим менше прямокутників і чим більше клітинок у прямокутниках, тим краще покриття. З кількох варіантів обирають той, в якого менший

коефіцієнт покриття: $z = \frac{r}{s}$, де r – загальне число прямокутників, s – їхня сумарна площа в клітинках;

4) форми, отримані в результаті мінімізації, містять r елементарних кон'юнкцій (за кількістю прямокутників у покритті). Кожна кон'юнкція містить тільки ті змінні, які не змінюють свого значення в наборах, що склеюються у відповідному прямокутнику. Число змінних у кон'юнкції називають її рангом. При склеюванні двох сусідніх клітинок одержують ранг кон'юнкції $n-1$, чотирьох – $n-2$, восьми клітинок – $n-3$ і т. д.

Розмічування карт Карно для функцій чотирьох змінних

Таблиця 4.2

x_1x_2 \ x_3x_4	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

Для мінімізації булевих функцій використовують також діаграми Вейча, аналогічні до карт Карно, які відрізняються від них способом розмічування замість символів 0 і 1 використовують булеві аргументи x_1, $\overline{x_1}$, x_2 та ін.

Діаграми Вейча для 2, 3 та 4-х змінних мають такий вигляд (табл. 4.3, 4.4, 4.5).

Діаграма Вейча для 2-х змінних

Таблиця 4.3

	x2	$\overline{x2}$
x1	11	10
$\overline{x1}$	01	00

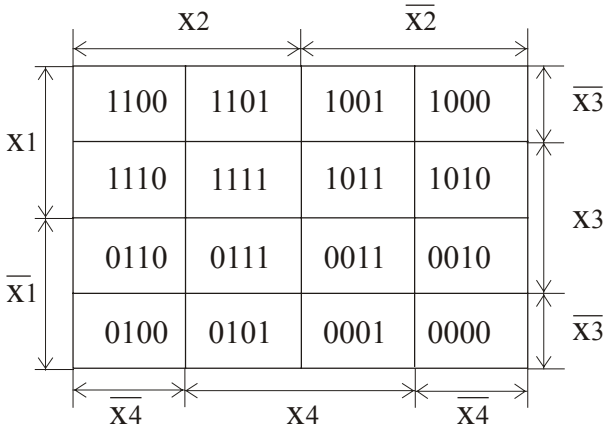
Діаграма Вейча для 3-х змінних

Таблиця 4.4

	x2		$\overline{x2}$	
x1	110	111	100	100
$\overline{x1}$	010	011	000	000
	$\overline{x3}$	x3		$\overline{x3}$

Діаграма Вейча для 4-х змінних

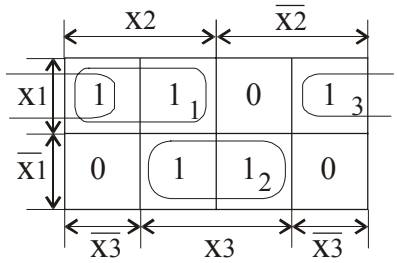
Таблиця 4.5



Приклад: спростити логічний вираз, використовуючи діаграми Вейча.

$$y = x_1 \wedge \overline{x_2} \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge \overline{x_3} \vee \overline{x_1} \wedge \overline{x_2} \wedge x_3 \vee \overline{x_1} \wedge x_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge x_3$$

Діаграма Вейча, згідно із заданим виразом, матиме такий вигляд:



Отже, отримаємо такий спрощений вираз:

$$y = x_1 \wedge x_2 \vee \overline{x_1} \wedge x_3 \vee x_1 \wedge \overline{x_3}\,.$$

Контрольні запитання

1. Для чого призначені методи мінімізації?
2. Назвати методи мінімізації логічних виразів.
3. Назвати графічні методи мінімізації.
4. Мінімізувати задані логічні вирази.

5. СИНТЕЗ КОМБІНАЦІЙНИХ СХЕМ

5.1. Аналітичне подання булевих функцій
5.2. Етапи синтезу логічних схем на логічних елементах

На сучасному етапі розроблені універсальні (канонічні) форми подання булевих функцій, які дають змогу одержати аналітичну форму довільної функції безпосередньо з таблиці істинності. Ця форма надалі може бути мінімізована або спрощена.

Оскільки між множиною аналітичного подання і множиною схем, які реалізують цю функцію, є взаємно однозначна відповідність, то пошук канонічної форми запису – це початковий етап синтезу логічних схем.

Найбільш поширеними є досконала диз'юнктивна нормальна форма (ДДНФ) і досконала кон'юнктивна нормальна форма (ДКНФ). Для одержання цих форм вводять поняття мінтермів (конституента 1) і макстермів (конституента 0).

Мінтерм – це функція n змінних, яка дорівнює одиниці тільки на одному наборі.

Мінтерм одержують як кон'юнкцію n змінних, що належать до нього безпосередньо, якщо значення даної змінної в наборі становить $x_i = 1$, та із запереченням, якщо $x_i = 0$. При n змінних є 2^n мінтермів $m_0, m_1, \ldots, m_R$, де $R = 2^n - 1$.

Усі мінтерми двох змінних подано в табл. 5.1.

Мінтерми двох змінних

Таблиця 5.1

x_1	x_2	F_9	f_i	Мінтерми	Макстерми
0	0	1	$f_0 = 1$	$m_0 = \overline{x_1} \wedge \overline{x_2}$	$M_0 = x_1 \vee x_2$
0	1	0	$f_1 = 0$	$m_1 = \overline{x_1} \wedge x_2$	$M_1 = x_1 \vee \overline{x_2}$
1	0	0	$f_2 = 0$	$m_2 = x_1 \wedge \overline{x_2}$	$M_2 = \overline{x_1} \vee x_2$
1	1	1	$f_3 = 1$	$m_3 = x_1 \wedge x_2$	$M_0 = \overline{x_1} \vee \overline{x_2}$

Значення функції F_9, які відповідають згідно з таблицею істинності кожному і-му наборові, позначені через f_0, f_1, f_2, і f_3.

Подання функції F_9 у ДДНФ є диз'юнктивною сумою мінтермів, які відповідають наборам змінних, для яких $f_i = 1$.

$$F_9 = f_0 \wedge m_0 \vee f_1 \wedge m_1 \vee f_2 \wedge m_2 \vee f_3 \wedge m_3 = 1 \wedge m_0 \vee 0 \wedge m_1 \vee 0 \wedge m_2 \vee 1 \wedge m_3 =$$
$$= \overline{x_1} \wedge \overline{x_2} \vee x_1 \wedge x_2$$

Макстерм – це функція n змінних, яка дорівнює нулю тільки на одному наборі.

Макстерм одержують як диз'юнкцію всіх змінних, що належать до нього безпосередньо, коли значення $x_i = 0$, або в інвертованому вигляді, якщо значення $x_i = 1$.

Кількість макстермів дорівнює 2^n, для функції двох змінних вони подані в табл. 5.2.

Подання функції у ДКНФ записується у вигляді:

$$F_9 = (f_0 \vee M_0) \wedge (f_1 \vee M_1) \wedge (f_2 \vee M_2) \wedge (f_3 \vee M_3) =$$
$$= (1_0 \vee M_0) \wedge (0 \vee M_1) \wedge (0_2 \vee M_2) \wedge (1_3 \vee M_3) = M_1 \wedge M_2 = (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2).$$

Приклад: (на прикладі табл. 5.2.) пояснити аналітичний запис функції з трьома змінними у ДДНФ і ДКНФ.

Таблиця істинності для функції з трьома змінними

Таблиця 5.2

x_1	x_2	x_3	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Для запису функції P у ДДНФ потрібно диз'юнктивно скласти ті мінтерми, для яких функція дорівнює одиниці. Отримаємо:

$$P = \overline{x_1} \wedge \overline{x_2} \wedge x_3 \vee \overline{x_1} \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge \overline{x_2} \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3.$$

Для запису функції P у ДКНФ необхідно подати кон'юнкцію макстермів, для яких функція дорівнює нулю. Матимемо:

$$P = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3).$$

За даним способом виконують запис у ДДНФ і ДКНФ функцій з довільною кількістю змінних.

Система функцій, за суперпозицією яких можна подати будь-яку булеву функцію, називається функціонально повною. Вона утворює базис у логічному просторі.

Систему функцій називають мінімально повним базисом, якщо усунення з неї будь-якої функції перетворює цю систему в неповну. В теорії алгебри-логіки доведено, що функціонально повні системи утворюють такі набори функцій:

1) НЕ, АБО, І;
2) НЕ, АБО;
3) НЕ, І;

4) І-НЕ;
5) АБО-НЕ.

Інша алгебра логіки будується на основі функції суми за модулем два і кон'юнкції (алгебра Жегалкіна).

Через операції алгебри Жегалкіна можна виразити всі інші булеві функції.

Функціональну схему логічного пристрою одержують у результаті абстрактного синтезу, який складається з таких етапів:

1) текстовий опис функцій логічного пристрою;
2) складання таблиці істинності за текстовим описом;
3) запис логічного рівняння пристрою у вигляді досконалої нормальної диз'юнктивної форми (ДДНФ) або досконалої нормальної кон'юнктивної форми (ДКНФ);
4) мінімізація логічного рівняння;
5) вибір одного із логічних базисів для реалізації функціональної схеми;
6) перетворення логічного рівняння з використанням правил де Моргана;
7) побудова функціональної схеми цифрового пристрою.

Приклад: синтезувати логічний пристрій з трьома вхідними змінними, який генерує сигнал "1" на виході, якщо хоча б дві змінні підряд набувають значення "1".

1. Складаємо таблицю істинності (табл. 5.3).

Таблиця істинності з трьома вхідними змінними

Таблиця 5.3

x_1	x_2	x_3	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

2. Логічне рівняння у вигляді ДДНФ є диз'юнкцією кон'юнкцій тих вхідних наборів, для яких $y = 1$:

$$y = \overline{x_1} \wedge x_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge \overline{x_3} \vee x_1 \wedge x_2 \wedge x_3 .$$

3. Мінімізація логічного рівняння здійснюється шляхом використання законів алгебри-логіки:

$$y = x_2 \wedge x_3 \wedge \left(\overline{x_1} \vee x_1\right) \vee \left(x_1 \wedge x_2\right) \wedge \left(\overline{x_3} \vee x_3\right) = x_2 \wedge x_3 \vee x_1 \wedge x_2 .$$

4. Функціональну схему реалізуємо в базисі І-НЕ. Для цього мінімізоване рівняння перетворимо за правилом де Моргана:

а) у базисі І-НЕ матимемо:

$$y = \overline{\overline{x_2 \wedge x_3} \vee \overline{x_1 \wedge x_2}} = \overline{\overline{x_2 \wedge x_3} \wedge \overline{x_1 \wedge x_2}},$$

б) у базисі АБО-НЕ отримаємо:

$$y = \overline{\overline{\overline{x_2 \wedge x_3}} \vee \overline{\overline{x_1 \wedge x_2}}} = \overline{\overline{(\overline{x_2} \vee \overline{x_3})} \vee \overline{(\overline{x_1} \vee \overline{x_2})}}$$

6) Функціональні схеми логічного пристрою, реалізовані у базисах І-НЕ, АБО-НЕ, подано на рис. 5.1 і 5.2.

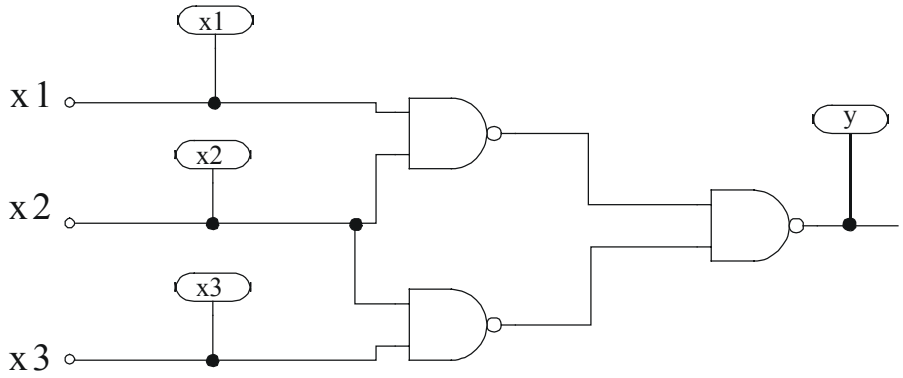


Рис. 5.1. Функціональна схема логічного пристрою у базисі І-НЕ.

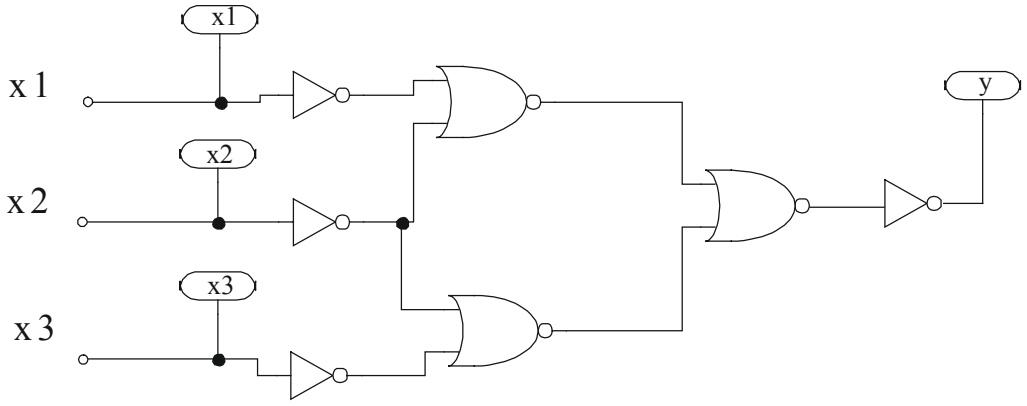


Рис. 5.2. Функціональна схема логічного пристрою у базисі АБО-НЕ.

Контрольні запитання

1. Дати визначення комбінаційного цифрового пристрою.
2. Назвати етапи синтезу цифрових комбінаційних пристроїв.
3. Як записується досконала нормальна диз'юнктивна форма?
4. Як записується досконала нормальна кон'юнктивна форма?
5. Перевести задане рівняння у базис І-НЕ, АБО-НЕ.
6. Синтезувати функціональну схему пристрою за заданим рівнянням.

6. АНАЛІЗ КОМБІНАЦІЙНИХ СХЕМ

6.1. Аналіз КС за методом синхронного моделювання
6.2. Аналіз КС за методом асинхронного моделювання

Завдання аналізу полягає у визначенні статичних і динамічних властивостей комбінаційної схеми (КС). У статиці визначають булеві функції (БФ), які реалізуються за відомою структурою комбінаційної схеми. В динаміці розглядається можливість надійного функціонування схеми в перехідних процесах при зміні значень змінних на вході схеми, тобто визначається наявність на входах схеми небажаних імпульсних сигналів, які не випливають безпосередньо із виразів для булевих функцій, що реалізує схема.

Завдання аналізу КС виникають за потреби перевірки правильності синтезу (на етапі проектування) або визначення булевої функції, яку реалізує КС (при аналізі або ремонті схеми).

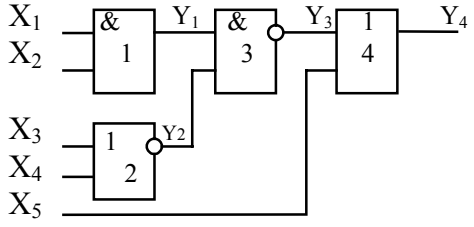
Усі методи аналізу поділяють на прямі і непрямі. В результаті аналізу КС за прямим методом одержуємо множину наборів вхідних змінних, які забезпечують задане значення на виході, що дає змогу записати математично БФ, які реалізує схема. До прямих належить метод π -алгоритму.

Використання непрямих методів дає можливість визначити реакцію схеми на заданий набір вхідних змінних у статиці або проаналізувати перехідний процес зміни одного вхідного набору на інший. Прикладами непрямих методів аналізу є методи синхронного й асинхронного моделювання. Зазначені методи аналізу машиноорієнтовані, що дає змогу виконати проаналізувати схеми на ЕОМ.

Аналіз КС за методом синхронного моделювання

Застосовуючи даний метод, слід припустити, що всі логічні елементи (ЛЕ) перемикаються одночасно без затримки. В результаті визначаємо значення сигналу на виході схеми.

Приклад: проаналізуємо метод синхронного моделювання за схемою (рис. 6.1).



N елем	Функція	Входи
1	2 І	X1 X2
2	2 АБО-НЕ	X3 X4
3	2 І-НЕ	Y1 Y2
4	2 АБО	Y3 X5

Рис. 6.1. Схема методу синхронного моделювання.

На першому етапі схему розбиваємо на рівні і записуємо в порядку зростання рівня рівняння, що описує функціонування ЛЕ (табл. 6.1).

Перший етап методу синхронного модулювання

Таблиця 6.1

№ рівня	№ елемента	Рівняння
1	1 2	$Y_1 = X_1 \wedge X_2$ $Y_2 = \overline{X_3 \vee X_4}$
2	3	$Y_3 = \overline{Y_1 \wedge Y_2}$
3	4	$Y_4 = Y_3 \vee X_5$

Для аналізу схеми при поданні на вхід набору $X_1 = 0$, $X_2 = 0$, $X_3 = 0$, $X_4 = 1$ і $X_5 = 1$

Для цього необхідно розв'язати рівняння в порядку зростання. Отримаємо:

$$Y_1 = X_1 \wedge X_2 = 0 \wedge 0 = 0,$$
$$Y_2 = \overline{X_3 \vee X_4} = \overline{0 \vee 1} = 0,$$
$$Y_3 = \overline{Y_1 \wedge Y_2} = \overline{0 \wedge 0} = 1,$$
$$Y = Y_4 = Y_3 \vee X_5 = 1 \vee 1 = 1.$$

Відповідно при поданні на вхід набору (0 0 0 1 1) на виході матимемо $Y = 1$. Аналогічно можна промоделювати роботу схеми при поданні на вхід будь-якого іншого набору.

Аналіз КС за методом асинхронного моделювання

Реальний ЛЕ перемикається за деякий кінцевий час, який залежить від технології виготовлення, умов експлуатації, місткості навантаження тощо. Проходження сигналу послідовно через кілька ЛЕ зумовлює сумування часу затримки і виникнення зсуву в часі вихідного сигналу щодо вхідного.

Наявність затримки і часового зсуву сигналів, який вона спричинює може зумовлювати появу на виході окремих ЛЕ і всієї схеми загалом короткочасних сигналів, не передбачених БФ, що реалізує схема (рис. 6.2).

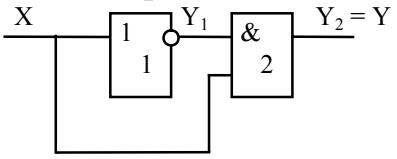


Рис. 6.2. Схема формування короткочасних сигналів.

Дана схема реалізує функцію $Y = X \wedge \overline{X} = 0$, тобто константу 0, незалежно від вхідного сигналу X. Проте в перехідному процесі внаслідок затримки спрацювання ЛЕ можлива ситуація, коли на обидвох входах елемента 2 І будуть логічні одиниці, що може привести до появи на виході схеми, логічної 1 (рис. 6.3).

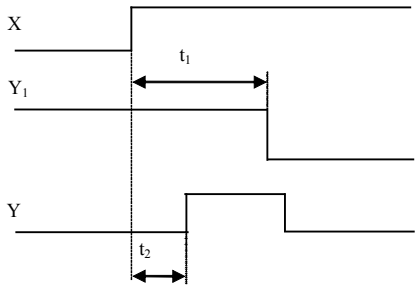
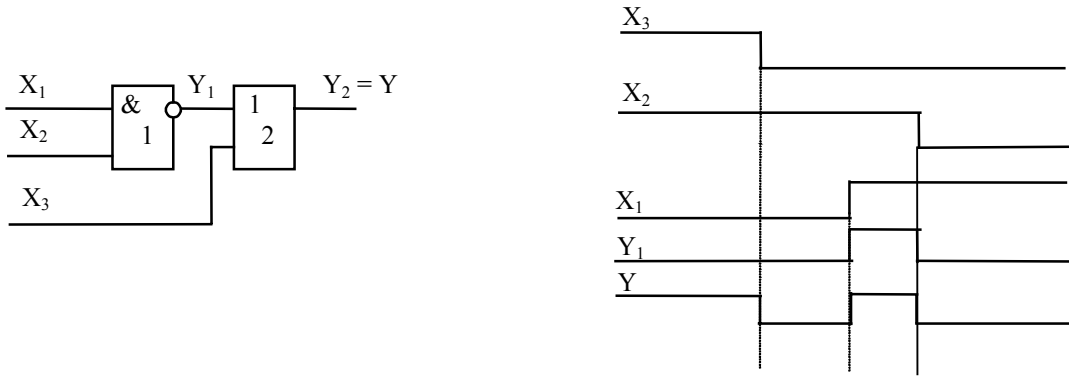


Рис. 6.3. Статичний ризик збою. Часові діаграми:
t_1 – час затримки інвертора; t_2 – час затримки елементу 2 І.

Даний випадок можливий при затримці спрацювання другого елементу, що є більшою ніж у першого. Таке явище називають ризиком збою. Розрізняють статичні і динамічні ризики збою.

За ***статичного*** ризику збою до і після перехідного процесу стан вихідного сигналу однаковий, а під час перехідного процесу можлива короткочасна поява протилежного сигналу.

За ***динамічного*** ризику збою до і після перехідного процесу стан вихідного сигналу протилежний, але в перехідному процесі вихідний сигнал кілька разів змінює значення. Динамічний ризик збою можливий у схемі (рис. 6.4, а) при зміні одного набору (X1=0, X2=1, X3=1) на інший (X1=1, X2=0, X3=0), що відображено за допомогою діаграми (рис. 6.4 б).



а) б)

Рис. 6.4: а) схема; б) часові діаграми.

У даному прикладі динамічний ризик збою на виході КС супроводжується статичним на виході елементу 1. Відповідно до зображення з часових діаграм, ризик збою є за наявності певного часового зсуву між сигналами, що надходять на вхід ЛЕ. Небажані сигнали на виході можуть і не бути за іншого співвідношення часових сигналів, проте можливість їх появи є фактором, що

знижує надійність роботи схеми. Таким чином, важливо вміти знаходити і усувати такі явища.

Для аналізу процесу перемикання КС при зміні вхідних наборів і виявлення ризиків збою використовують метод ***асинхронного моделювання***. При цьому методі вважається, що кожний елемент перемикається з однаковою затримкою. Аналіз має такі етапи:

1) кожному елементу схеми присвоюється рівень, причому рівень 1 мають елементи, всі входи яких є незалежними входами схеми;

2) записуються рівняння, що описують кожний ЛЕ за порядком зменшення рівня;

3) для початкового вхідного набору A(X1, X2, ... , Xn) визначають значення сигналів на виходах всіх ЛЕ схеми. Нехай даний набір А замінюється набором B(X1, X2,.., Xn);

4) визначають ті рівняння, в правій частині яких хоча б одна із змінних змінила значення;

5) розв'язують визначені рівняння за порядком їх запису в схемі. Після розв'язання рівняння його вважають невизначеним;

6) якщо після розв'язання всіх рівнянь системи змінні, які належать до їх лівої частини, змінили значення, то знову визначають ті рівняння, в правій частині яких є ці змінні. Потім здійснюється перехід до п. 5. В іншому разі моделювання даного вхідного набору вважається завершеним. Виконання п. 5 називають ***тактом моделювання.***

Аналіз схеми за методом асинхронного моделювання подано нижче (рис. 6.5). Для даної схеми вхідний набір A(1011110) замінюється набором B(1101011).

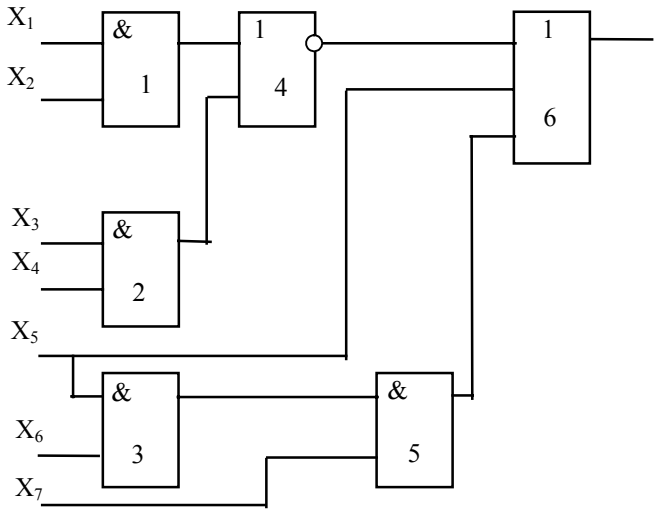


Рис. 6.5. Комбінаційна схема для методу асинхронного моделювання.

Рівняння, що описують ЛЕ

Таблиця 6.2

Рівняння	1-й такт	2-й такт	3-й такт
$Y = Y_6 = Y_4 \vee Y_5 \vee X_5$	*	*	*
$Y_5 = Y_3 \wedge X_7$	*	*	–
$Y_4 = \overline{Y_1 \vee Y_2}$	–	*	–
$Y_3 = \overline{X_5 \wedge X_6}$	*	–	–
$Y_2 = X_3 \wedge X_4$	*	–	–
$Y_1 = X_1 \wedge X_2$	*	–	–

Виходи	Такти моделювання			
	0	***1***	***2***	***3***
Y_6	1	0	1	0
Y_5	0	1	0	0
Y_4	0	0	0	0
Y_3	1	0	0	0
Y_2	1	0	0	0
Y_1	0	1	0	1

Згідно з результатами моделювання, при заміні набору А набором В на виході елементу 4 наявний статичний ризик збою, а на виході схеми – динамічний ризик збою.

Радикальним способом усунення ризиків збою є введення стробування для зняття вихідного сигналу КС. Стробуючий імпульс подається після завершення перехідного процесу в КС (коли на виході КС уже встановлено необхідне значення вихідного сигналу), що виключає вплив можливих збоїв на сигнал, які виробляє схема.

Контрольні запитання

1. Для чого призначені методи аналізу?
2. Назвати етапи аналізу комбінаційних схем.
3. Перелічити причини статичних та динамічних ризиків у КС.
4. Проаналізувати задану комбінаційну схему.

7. СИНТЕЗ ДЕШИФРАТОРІВ ТА ШИФРАТОРІВ

Дешифратор – це комбінаційний пристрій, який перетворює комбінацію вхідних змінних в активний сигнал “лог. 1” або “лог. 0” тільки на одному із виходів.

Дешифратори і шифратори належать до перетворювачів кодів.

Максимальна кількість виходів дешифратора дорівнює 2^n, де n – кількість входів.

Якщо частина вхідних наборів не використовується, то дешифратор називають неповним і в нього $N_{вих} < 2^n$.

Якщо вхідні змінні подати як двійкову систему запису чисел, то логічна одиниця формується на тому виході, номер якого відповідає десятковому запису числа.

Наприклад, $A=0$, $B=1$, $C=0$, $D=1$. Числу 0101 у двійковому коді відповідає число 5 в десятковому коді, тобто при вказаній комбінації вхідних змінних $F_5=1$.

Дешифратори широко використовують як перетворювачі двійкового коду в десятковий.

В ЕОМ з допомогою дешифраторів здійснюється вибірка необхідних комірок запам’ятовуючих пристроїв, розшифровка кодів операцій з видачею відповідних керуючих сигналів.

В умовних позначеннях дешифраторів і шифраторів використовують букви DC і CD (від decoder і coder відповідно).

Робота дешифратора описується системою логічних рівнянь, складених на основі таблиці істинності (табл.. 7.1).

Таблиця істинності для дешифратора

Таблиця 7.1

Входи		Виходи			
x_2	x_1	y_0	y_1	y_2	y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Складаємо рівняння виходів:

$$y_0 = \overline{x_1} \wedge \overline{x_2};\ \ y_1 = x_1 \wedge \overline{x_2};\ \ y_2 = \overline{x_1} \wedge x_2;\ \ y_3 = x_1 \wedge x_2.$$

За заданими рівняннями складаємо функціональну схему дешифратора.

Шифратор перетворює код “1 із N” у двійковий, тобто виконує операцію, протилежну до тієї, що здійснює дешифратор.

При поданні на один із входів шифратора сигналу “лог. 1” на виході формується двійковий код, що відповідає номеру входу, на який подано сигнал “лог. 1”.

Робота шифратора описується системою логічних рівнянь, складених на основі таблиці істинності (табл. 7.2).

Таблиця істинності для шифратора

Таблиця 7.2

Входи				Виходи	
x_4	x_3	x_2	x_1	y_2	y_1
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Складаємо рівняння виходів:

$$y_1 = \overline{x_1} \wedge x_2 \wedge \overline{x_3} \wedge \overline{x_4} \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3} \wedge x_4 ;$$

$$y_2 = \overline{x_1} \wedge \overline{x_2} \wedge x_3 \wedge \overline{x_4} \vee \overline{x_1} \wedge \overline{x_2} \wedge \overline{x_3} \wedge x_4 .$$

Рівняння переводимо до заданого базису і складаємо функціональну схему шифратора.

Приклад: записати логічне рівняння дешифратора, на виході якого буде “лог. 1” тільки при заданих вхідних адресах 126, 127.

Задані такі адреси у вісімковій системі числення: 126, 127.

Переведемо адреси в двійкову систему числення:

001 010 110, 001 010 111

Складемо логічні рівняння:

$$y_1 = \overline{x_0} \wedge x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4 \wedge \overline{x_5} \wedge x_6 \wedge \overline{x_7} \wedge \overline{x_8} ,$$

$$y_1 = x_0 \wedge x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4 \wedge \overline{x_5} \wedge x_6 \wedge \overline{x_7} \wedge \overline{x_8} .$$

Виразимо через z спільну частину:

$$z = x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4 \wedge \overline{x_5} \wedge x_6 \wedge \overline{x_7} \wedge \overline{x_8} .$$

Запишемо y_1 і y_2 через z:

$$y_1 = \overline{x_0} \wedge z ;$$

$$y_2 = x_0 \wedge z .$$

Одержані рівняння необхідно перевести в заданий базис і побудувати функціональну схему (рис. 7.1).

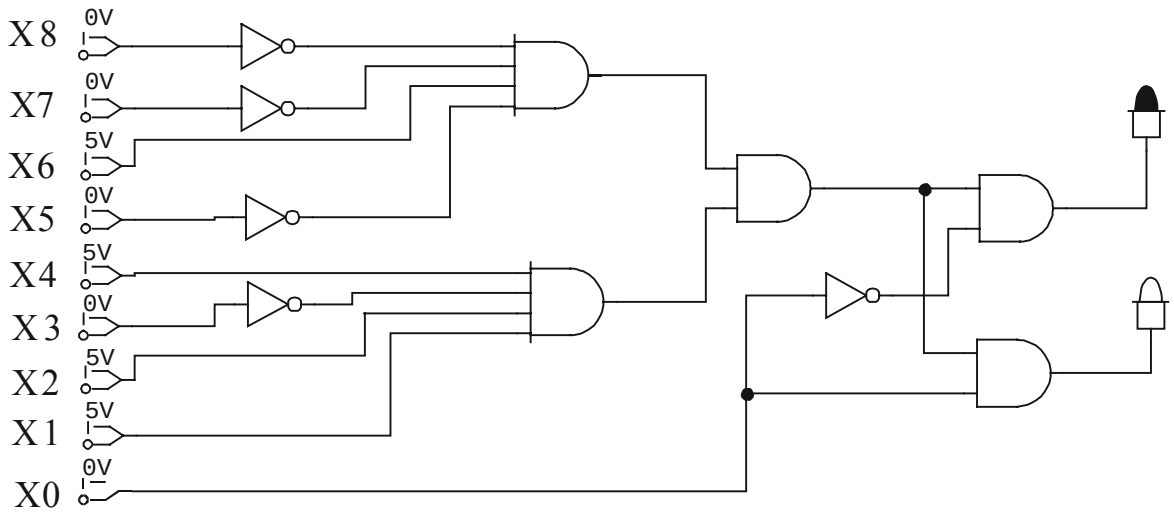


Рис. 7.1. Функціональна схема шифратора.

Контрольні запитання

1. Дати визначення дешифратора.
2. Дати визначення шифратора.
3. Скільки в дешифратора входів, якщо він має 10 виходів?
4. Якщо дешифратор має 3 входи, скільки в нього виходів?
5. Як дешифратори і шифратори позначають на схемах?

8. СИНТЕЗ МУЛЬТИПЛЕКСОРІВ ТА ДЕМУЛЬТИПЛЕКСОРІВ

Мультиплексор призначений для комутації в певному порядку на один вихід одного з кількох вхідних сигналів залежно від стану адресних входів. За допомогою мультиплексора здійснюють часовий розподіл інформації, що надходить за різними каналами. Мультиплексор можна порівняти з безконтактним багатопозиційним перемикачем.

Мультиплексор має один або два взаємодоповнюючі виходи (прямий і інвертований) і дві групи входів:

– інформаційні;
– керуючі (адресні та дозволяючі).

Якщо мультиплексор має n адресних входів, то кількість інформаційних входів дорівнюватиме 2^n.

Набір сигналів на адресних входах дає змогу визначити інформаційний вхід, який потрібно з'єднати з виходом мультиплексора. Дозволяючий (стробуючий) вхід керує одночасно всіма інформаційними входами, незалежно від стану адресних входів. Заборонений сигнал на цьому вході блокує роботу всього пристрою. Наявність дозволяючого входу розширює функціональні можливості мультиплексора, дає змогу синхронізувати його роботу з роботою інших вузлів. Дозволяючий вхід використовують також для нарощування розрядності мультиплексора. Адресні входи мультиплексора позначимо буквами $A_i \in \{0, 1\}$, а інформаційні – $D_i \in \{0, 1\}$ (рис. 8.1).

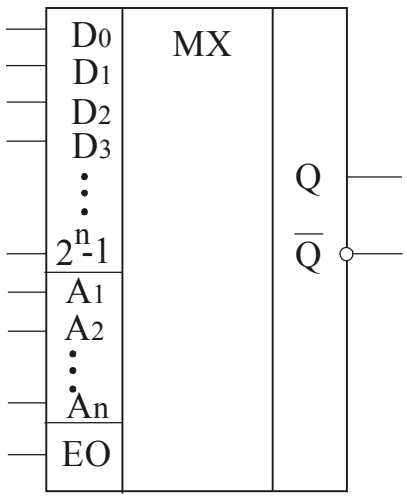


Рис. 8.1. Графічне позначення мультиплексора.

Приклад: синтезувати мультиплексор 4:1.

Складаємо таблицю істинності мультиплексора 4:1 (табл. 8.1).

Таблиця істинності мультиплексора 4:1

Таблиця 8.1

A1	A0	Вихід Q
0	0	D0
0	1	D1
1	0	D2
1	1	D3

Робота мультиплексора 4:1 описується логічним рівнянням, складеним на основі таблиці істинності. Таким чином, отримаємо:

$$Q = \overline{A_1} \wedge \overline{A_0} \wedge D_0 \vee \overline{A_1} \wedge A_0 \wedge D_1 \vee A_1 \wedge \overline{A_0} \wedge D_2 \vee A_1 \wedge A_0 \wedge D_3 .$$

За одержаним рівнянням розробляємо функціональну схему мультиплексора (рис. 8.2).

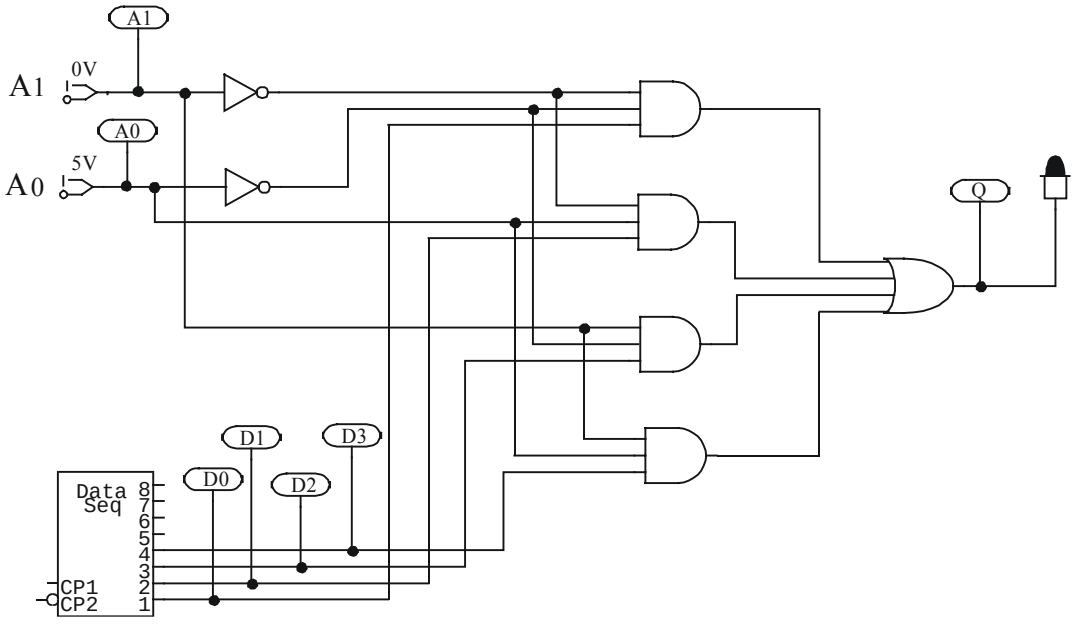


Рис. 8.2. Функціональна схема мультиплексора 4:1.

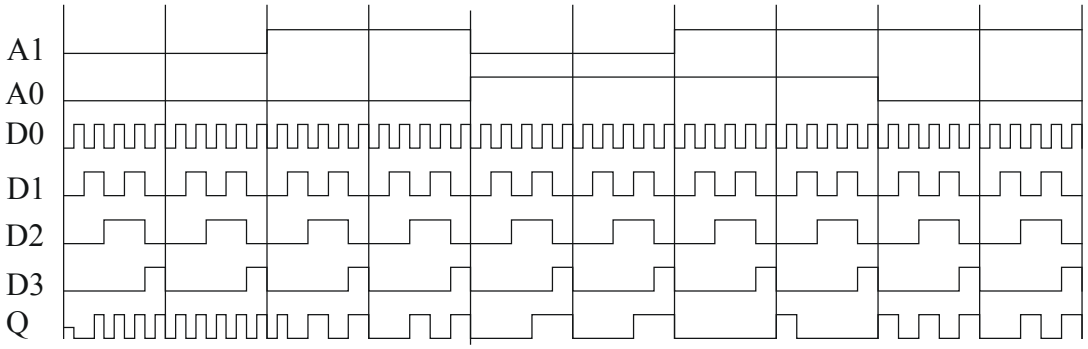


Рис. 8.3. Часова діаграма роботи мультиплексора.

Рівняння мультиплексора з дозволяючим входом має такий вигляд:

$$Q = \overline{A_1} \wedge \overline{A_0} \wedge D_0 \wedge EO \vee \overline{A_1} \wedge A_0 \wedge D_1 \wedge EO \vee A_1 \wedge \overline{A_0} \wedge D_2 \wedge EO \vee$$
$$\vee A_1 \wedge A_0 \wedge D_3 \wedge EO.$$

Демультиплексор – це комбінаційна схема, яка комутує сигнал з інформаційного входу на один з кількох виходів залежно від стану адресних входів. При n адресних входах демультиплексор може мати 2^n виходів.

Приклад: синтезувати демультиплексор 1:4 з дозволяючим входом ЕО (рис. 8.4).

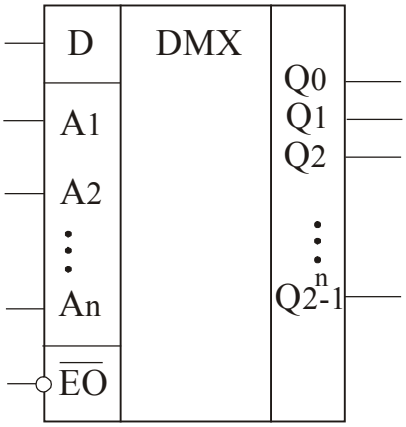


Рис. 8.4. Графічне позначення демультиплексора.

Таблицю істинності демультиплексора 1:4 подано нижче (табл. 8.2).

Таблиця істинності демультиплексора 1:4

Таблиця 8.2

Входи				Виходи			
A1	A0	D	$\overline{EO}$	Q1	Q2	Q3	Q4
0	0	0/1	0	D	0	0	0
0	1	0/1	0	0	D	0	0
1	0	0/1	0	0	0	D	0
1	1	0/1	0	0	0	0	D
0	0	x	1	0	0	0	0
0	1	x	1	0	0	0	0
1	0	x	1	0	0	0	0
1	1	x	1	0	0	0	0

Робота демультиплексора 1:4 з дозволяючим входом $\overline{EO}$ описується логічними рівняннями, складеними на основі таблиці істинності. Рівняння матимуть такий вигляд:

$$Q_1 = \overline{A_1} \wedge \overline{A_0} \wedge D \wedge \overline{EO},$$

$$Q_2 = \overline{A_1} \wedge A_0 \wedge D \wedge \overline{EO},$$

$$Q_3 = A_1 \wedge \overline{A_0} \wedge D \wedge \overline{EO},$$
$$Q_4 = A_1 \wedge A_0 \wedge D \wedge \overline{EO}.$$

Одержані рівняння переводять у заданий базис і складають функціональну схему.

5. Контрольні запитання

1. Дати визначення мультиплексора.
2. Дати визначення демультиплексора.
3. Як називають входи мультиплексора? Їх призначення.
4. Як мультиплексор і демультиплексор позначають на схемах?
5. Яка максимальна кількість інформаційних входів у мультиплексора з трьома адресними входами?

9. СИНТЕЗ СУМАТОРІВ

Суматор – логічний комбінаційний пристрій, що виконує ***арифметичне*** додавання кодів двох чисел. При арифметичному додаванні виконуються й інші додаткові операції: врахування знаків чисел, вирівнювання порядків. Зазначені операції виконують в арифметико-логічних пристроях (АЛП) чи процесорних елементах, ядром яких є суматори.

Суматори класифікують за різними ознаками.

Залежно від системи числення*:*

– двійкові;

– двійково-десяткові (у загальному випадку двійково-кодовані);

– десяткові.

За кількістю одночасно оброблюваних розрядів чисел, що додаються:

– одноразрядні;

– багаторозрядні.

За кількістю входів і виходів одноразрядних двійкових суматорів:

- чвертьсуматори (елементи “сума за модулем 2”; елементи “виключає АБО”), що характеризуються наявністю двох входів, на які подаються два одноразрядних числа, і одним виходом, на якому реалізується їхня арифметична сума;
- напівсуматори, що характеризуються наявністю двох входів, на які подаються однойменні розряди двох чисел, і двох виходів: на одному реалізується арифметична сума в даному розряді, а на іншому – перенесення в наступний (старший) розряд;
- повні одноразрядні двійкові суматори, що характеризуються наявністю трьох входів, на які подаються однойменні розряди двох чисел, що складаються, і перенесення з попереднього (молодшого) розряду, і двох виходів: на одному реалізується арифметична сума в даному розряді, а на іншому – перенесення у наступний (старший) розряд.

За способом подання й обробки чисел, що додаються, багаторозрядні суматори поділяються на*:*

- послідовні, в яких обробку чисел здійснюють за чергою: розряд за розрядом на тій самій елементній базі;
- паралельні, у яких доданки додаються одночасно за всіма розрядами, і для кожного розряду є своя елементна база.

Паралельний суматор у найпростішому випадку є **n** одноразрядними суматорами, послідовно (від молодших розрядів до старших) з’єднаними ланцюгами перенесення. Однак така схема суматора характеризується порівняно невисокою швидкодією, тому що формування сигналів суми і перенесення в кожному і-му розряді виробляється лише після того, як надійде сигнал перенесення з (і-1)-го розряду.

Чвертьсуматор

Найпростішим двійковим сумуючим елементом є чвертьсуматор. Походження назви цього елементу випливає з того, що він має в два рази менше

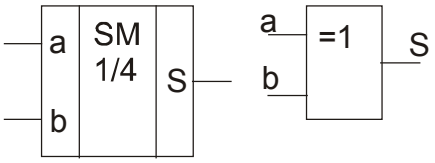
виходів і рядків у таблиці істинності порівняно з повним двійковим однорозрядним суматором. Найвживаніші назви: елемент “сума за модулем 2” і елемент “виключає АБО”. Схема (рис. 9.1) має два входи a і b для двох доданків, що додаються, й один вихід S для суми. Її роботу відображає таблиця істинності (табл. 9.1). Відповідне рівняння має такий вигляд:

$$S = \bar{a} \wedge b \vee a \wedge \bar{b} = a \oplus b \ . \qquad (9.1)$$

Таблиця істинності чвертьсуматора

Таблиця 9.1

a	b	S
0	0	0
0	1	1
1	0	1
1	1	0



а) б)

Рис. 9. 1. Графічне позначення чвертьсуматора.

$$S = a \wedge \overline{a \wedge b} \wedge b \wedge \overline{a \wedge b} \qquad (9.2)$$

$$S = a \vee \overline{a \vee b} \vee b \vee \overline{a \vee b} \qquad (9.3)$$

$$S = (a \vee b) \wedge \overline{a \wedge b} \qquad (9.4)$$

Схеми, отримані за рівняннями (9.2–9.4), подано на рис. 9.2.

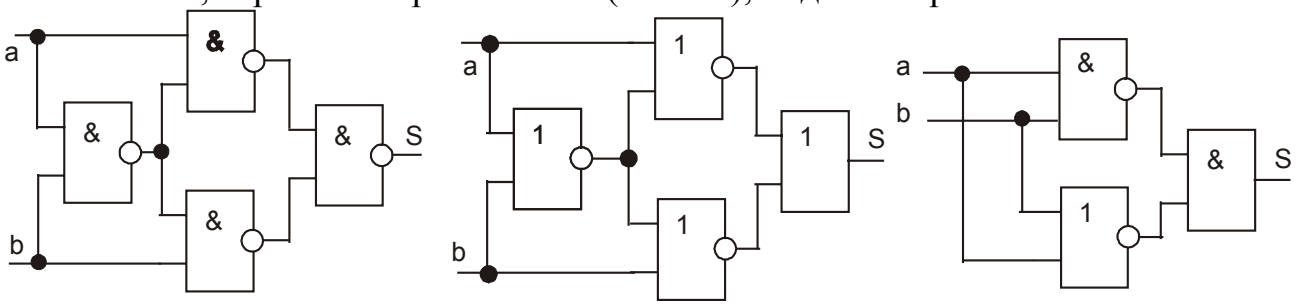


Рис. 9.2. Схеми чвертьсуматора.

Напівсуматор (рис. 9.3) має два входи a і b для двох чисел, що сумуються, і два виходи: S – сума, P – перенесення. Позначають напівсуматор буквами HS (half sum – напівсума). Його роботу відображає таблиця істинності (табл. 9.2), а відповідні рівняння мають такий вигляд:

$$S = \bar{a} \wedge b \vee a \wedge \bar{b} = a \oplus b,$$

$$P = a \wedge b.$$

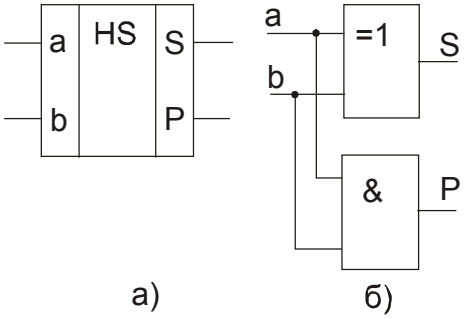


Рис. 9.3. Графічне позначення напівсуматора.

Таблиця істинності напівсуматора

Таблиця 9.2

a	**b**	**P**	**S**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Таким чином, з вищеподаних рівнянь випливає, що для реалізації напівсуматора потрібно один елемент “виключає АБО” і один двоходовий елемент І (рис. 9.3, б).

Повний однорозрядний двійковий суматор

Повний однорозрядний двійковий суматор (рис. 9.4) має три входи: **a**, b для двох доданків і **p** для перенесення з попереднього (молодшого) розряду і два виходи: **S** – сума, **P** – перенесення у наступний (старший) розряд. Позначають повний двійковий суматор буквами SM. Його роботу відображає таблиця істинності (табл. 9.3).

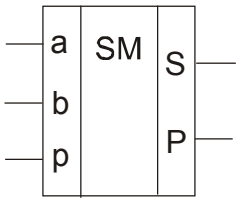


Рис. 9.4. Графічне позначення повного однорозрядного двійкового суматора.

Таблиця істинності однорозрядного двійкового суматора

Таблиця 9.3

№	a	b	p	S	P
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

Отже, записуємо рівняння виходів для S і для P, після чого складаємо принципову схему.

Контрольні запитання

1. Дати визначення суматора.
2. Дати визначення чвертьсуматора, напівсуматора, повного суматора.
3. Синтезувати чвертьсуматор у базисі І-НЕ, АБО-НЕ.
4. Синтезувати напівсуматор у базисі І-НЕ.
5. Записати таблицю істинності повного суматора.

10. ЕЛЕМЕНТАРНІ ЦИФРОВІ АВТОМАТИ

Тригер – цифровий автомат, який має два стійких стани 0 або 1 і призначений для зберігання одного біту даних. Стан тригера визначають за сигналами на його входах. Під впливом вхідного сигналу тригер стрибкоподібно переходить з одного стійкого стану в інший. Тригери мають два виходи: прямий Q та інвертований $\overline{Q}$. Стан тригера визначають за прямим виходом.

За логічним функціонуванням визначають такі типи тригерів: RS, D, T і JK.

За способом запису інформації є асинхронні і синхронні тригери. Синхронні тригери мають спеціальний вхід синхронізації (тактовий) – C (від Clok).

За способом сприйняття тактових сигналів тригери поділяють на статичні (керовані рівнем 0 або 1) і динамічні (керовані фронтом зростання або спадання).

RS-тригери

Асинхронний RS-тригер з прямими входами

Вхід R – це вхід скидання тригера в 0 (**Reset** – скидання).

Вхід S – це вхід установлення тригера в 1 (**S**et – встановлення).

Асинхронним називають такий тригер, який змінює свій стан у момент подання вхідного сигналу на входи R і S (рис. 10.1).

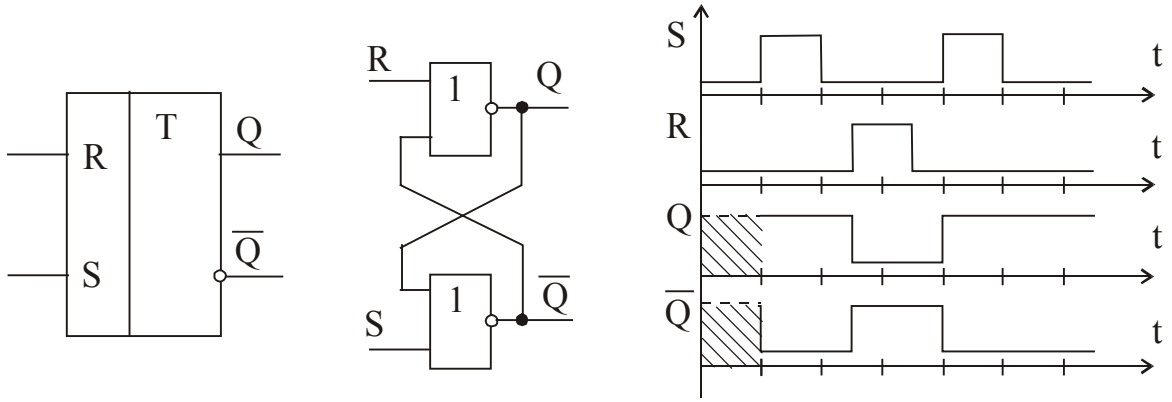


Рис. 10.1. Асинхронний RS-тригер: а) графічне позначення;
б) реалізація на елементах АБО-НЕ; в) часові діаграми роботи.

Таблиця переходів RS-тригера

Таблиця 10.1

***Вхід* S**	***Вхід* R**	***Вихід*** Q_{i+1}	***Режим роботи***
0	0	Q_i	*Зберігання*
0	1	0	*Скидання в* **0**
1	0	1	*Встановлення в* **1**
1	1	–	*Заборонений*

Асинхронний RS-тригер с інверсними входами

Активним сигналом для такої схеми є ***логічний 0*** (рис. 10.2).

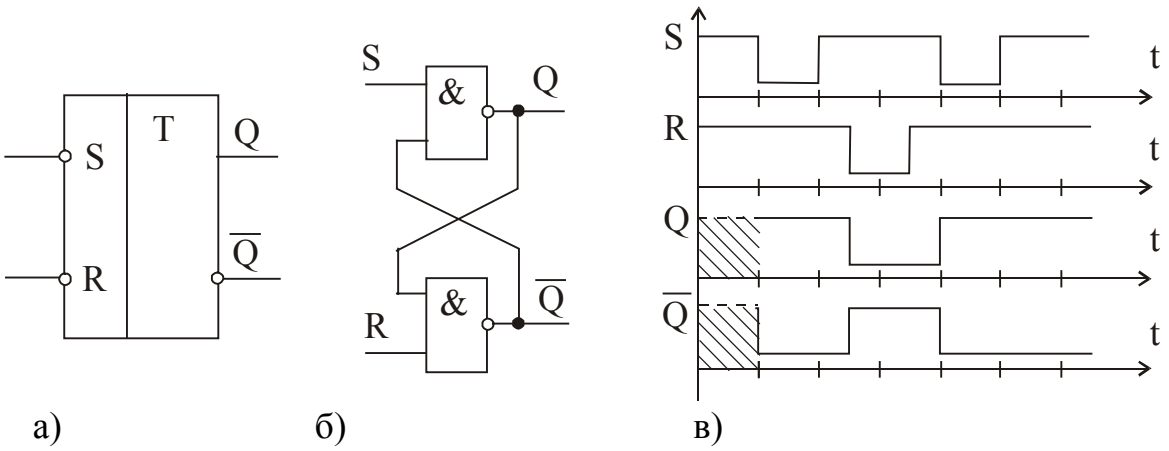


Рис. 10.2. Асинхронний RS-тригер: а) графічне позначення; б) реалізація на елементах І-НЕ; в) часові діаграми роботи.

Робота тригера визначається за таблицею переходів.

Таблиця переходів RS-тригера з інверсними входами

Таблиця 10.2

Вхід $\overline{S}$	***Вхід*** $\overline{R}$	***Вихід*** Q_{i+1}	***Режим***
0	0	–	*Заборонений*
0	1	1	*Встановлення в* **1**
1	0	0	*Скидання в* **0**
1	1	Q_i	*Зберігання*

Синхронний RS-тригер

Тригер називається ***синхронним,*** якщо в нього, крім інформаційних входів S і R, є ***тактовий вхід C***.

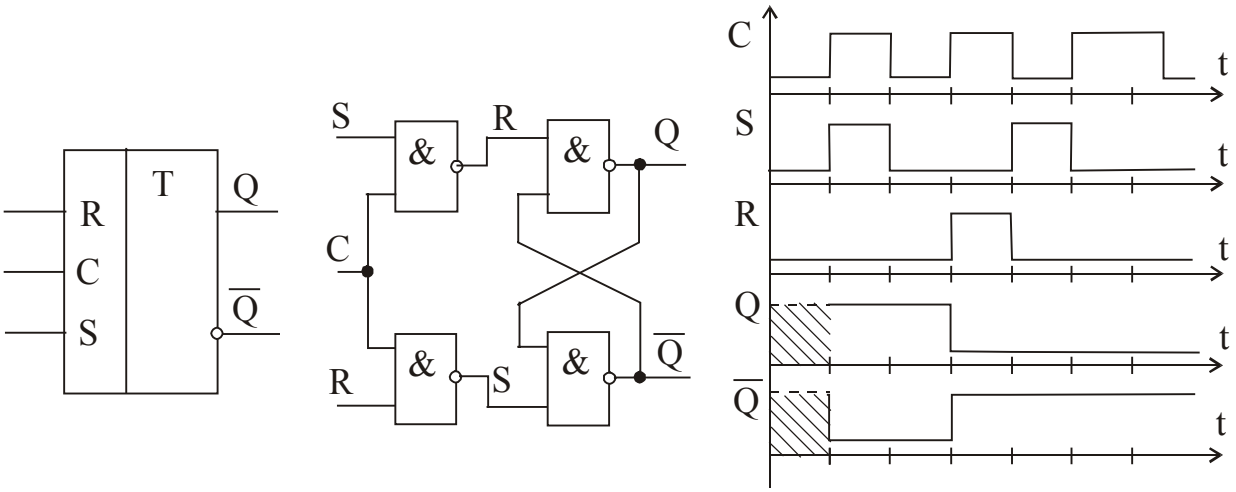
Синхронний тригер зі статичним керуванням змінює стан при логічній 1 на вході С (якщо С – вхід прямий) або при логічному 0 на вході С (якщо С – вхід інвертований).

Активним сигналом для поданої схеми (рис. 10.3) є ***логічна 1.***

Таблиця переходів синхронного RS-тригера

Таблиця 10.3

C	**S**	**R**	Q_{i+1}	***Режим роботи***
0	*	*	Q_i	*Зберігання*
1	**0**	**0**	Q_i	*Зберігання*
1	**0**	**1**	**0**	*Скидання в* **0**
1	**1**	**0**	**1**	*Встановлення в* **1**
1	**1**	**1**	**–**	*Заборонений*



а) б) в)

Рис. 10.3. Синхронний RS-тригер: а) графічне позначення; б) реалізація на елементах І-НЕ; в) часові діаграми роботи.

Т-тригери.

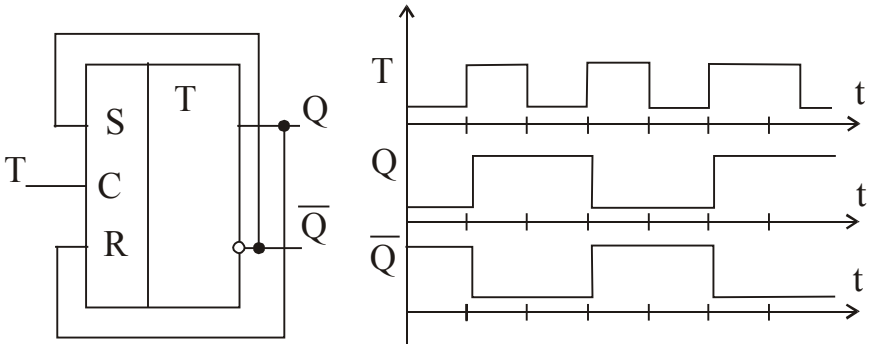
Асинхронний Т-тригер

Т-тригер має один інформаційний Т-вхід (від toggle) (рис. 10.4). Зміна стану тригера відбувається щоразу за зміни вхідного сигналу в визначеному напрямку. Стан Т-тригера визначають за його станом у попередньому такті.

Таблиця переходів асинхронного Т-тригера

Таблиця 10.4

T	Q_{i+1}	***Режим роботи***
0	Q_i	*Зберігання*
1	$\overline{Q_i}$	*Інверсія*



а) б)

Рис. 10.4. Асинхронний Т-тригер: а) графічне позначення; б) часові діаграми роботи.

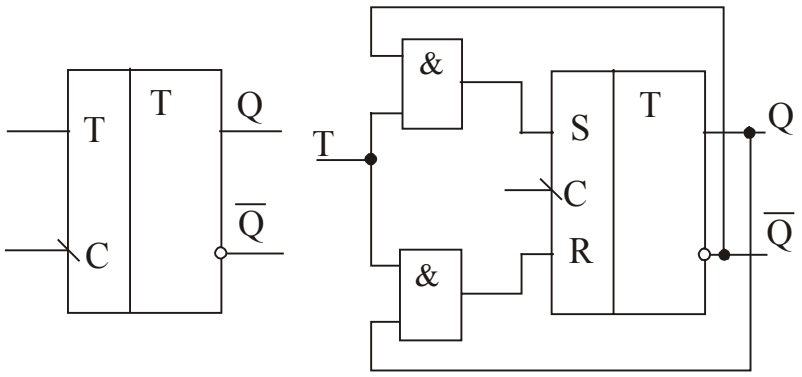
Синхронний Т-тригер.

Синхронний Т-тригер спрацьовує за фронтом зростання або спадання інформаційного сигналу (рис. 10.5). При 1 на вході Т-тригер змінює свій стан на протилежний під дією інформаційного сигналу.

Таблиця переходів синхронного Т-тригера

Таблиця 10.5

C	**T**	Q_{i+1}	**Режим роботи**
0	*	Q_i	*Зберігання*
1	**0**	Q_i	*Зберігання*
1	**1**	$\overline{Q_i}$	*Інверсія*



а) б)

Рис. 10.5. Синхронний Т-тригер: а) графічне позначення; б) реалізація Т-тригера на базі RS-тригера.

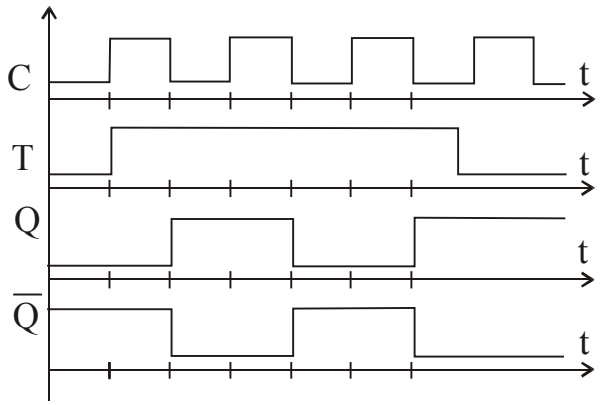


Рис. 10.6. Часові діаграми роботи синхронного Т-тригера.

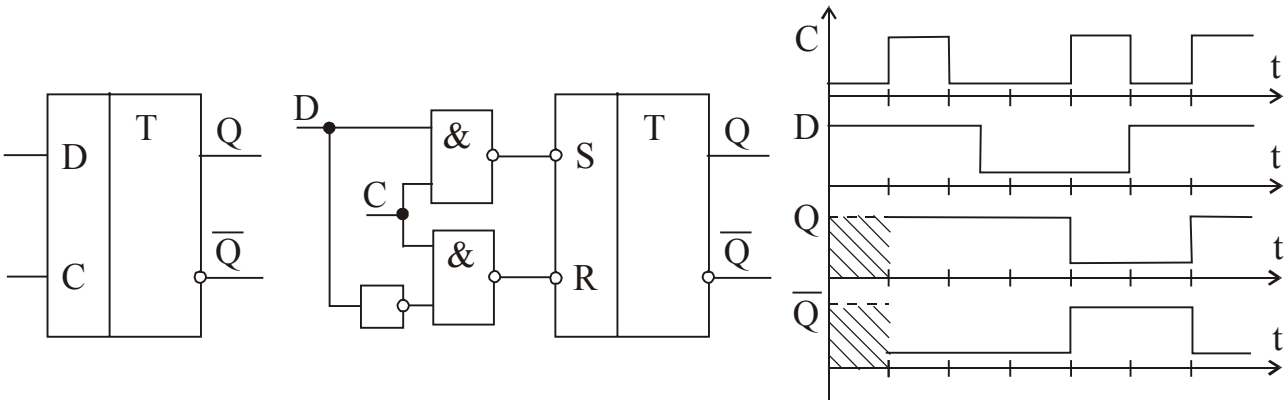
D-тригери

D-тригери (тригери затримки), на відміну від RS- і T-тригерів, мають один інформаційний вхід D для встановлення у стан **1** або 0. Позначення **D** – це перша буква англ. слова delay – затримка. Функціональна особливість D-тригерів у тому, що сигнал на виході Q у такті **n+1** повторює вхідний сигнал D^n у попередньому такті **n** і зберігає цей стан до приходу наступного тактового імпульсу. D-тригер затримує на один такт інформацію на вході **D** (рис. 10.7).

Таблиця переходів D-тригера

Таблиця 10.6

C	**D**	Q_{i+1}	***Режим роботи***
0	*	Q_i	*Зберігання*
1	**0**	**0**	*Скидання в 0*
1	**1**	**1**	*Встановлення в 1*



а) б) в)

Рис. 10.7. Синхронний D-тригер: а) графічне позначення; б) реалізація D-тригера на базі RS-тригера; в) часові діаграми роботи.

Синхронний D-тригер з асинхронними входами R і S

Асинхронні входи **R** і **S** мають пріоритет (тільки при R=1 і S=1 даний тригер працюватиме як синхронний D-тригер) (рис. 10.8).

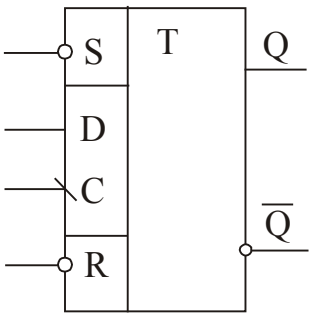


Рис. 10.8. Синхронний D-тригер з асинхронними інвертованими входами RS.

Таблиця переходів синхронного D – тригера

Таблиця 10.7

C	D	S	R	Q_{i+1}	***Режим роботи***
0	*	**0**	**0**	**–**	*Заборонений*
0	*	**0**	**1**	**1**	*Встановлення в 1*
0	*	**1**	**0**	**0**	*Скидання в 0*
0	*	**1**	**1**	Q_i	*Зберігання*
1	**0**	**1**	**1**	**0**	*Скидання в 0*
1	**1**	**1**	**1**	**1**	*Встановлення в 1*

JK-тригер

JK-тригери не мають невизначеного стану. При всіх вхідних комбінаціях, крім однієї $J^n = 1, K^n = 1$, вони працюють як RS-тригери, при цьому вхід J відповідає входу S, а вхід K – входу R. При вхідній комбінації $J^n = 1, K^n = 1$ з кожним тактом відбувається зміна вихідного сигналу (режим лічильника).

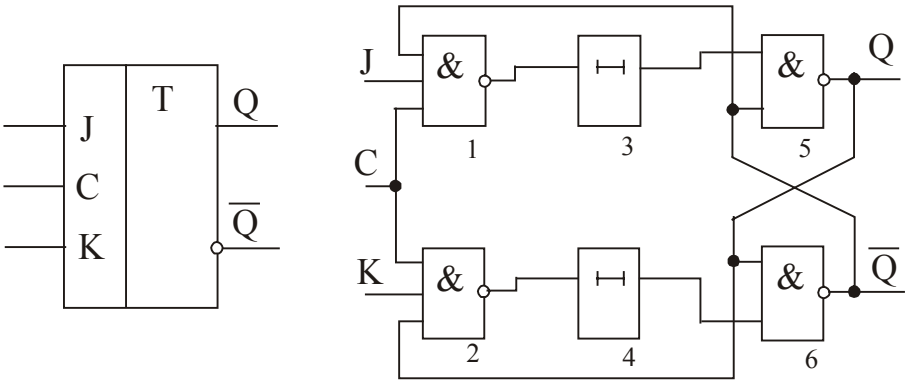
Логічна структура синхронного JK-тригера подана на рис. 10.9, б.

Робота JK-тригера описується характеристичним рівнянням:

$$Q^{n+1} = J^n \wedge \overline{Q}^n \vee \overline{K}^n \wedge Q^n .$$

При $J^n = K^n = 0$ на виходах елементів 1 і 2 буде $q_1 = q_2 = 1$ (незалежно від значень сигналів Q і $\overline{Q}$). Це нейтральна комбінація для тригера (елементи 5 і 6), який зберігає записану раніше інформацію. Якщо $J^n \neq K^n$, вихідний стан тригера буде визначатиме логічний елемент 1 або 2, на всіх входах якого є логічна 1. Вхідна комбінація $J^n = K^n = 1$ за будь-якого стану тригера зумовлює зміну (інвертування) вихідного сигналу.

Елементи затримки (3, 4) створюють часовий зсув між моментом введення вхідної інформації $J^n \overline{Q}^n$ або $\overline{K}^n Q^n$ і початком формування вихідної (Q^{n+1} і $\overline{Q}^{n+1}$). Без цих елементів під час дії вхідної комбінації $J^n = K^n = 1$ почнеться генерація, оскільки з кожною зміною вихідних сигналів на входах присутня комбінація, яка спричиняє зміну вихідного сигналу.



а) б)

Рис. 10.9. Синхронний JK-тригер: а) графічне позначення; б) логічна структура JK-тригера.

Таблиця переходів синхронного JK-тригера

Таблиця 10.8

C	**J**	**K**	Q_{i+1}	***Режим роботи***
0	*	*	Q_i	*Зберігання*
1	**0**	**0**	Q_i	*Зберігання*
1	**1**	**0**	**1**	*Встановлення в **1***
1	**0**	**1**	**0**	*Скидання в* **0**
1	**1**	**1**	$\overline{Q_i}$	*Інверсія*

JK-тригери належать до універсальних пристроїв, які шляхом відповідного з'єднання виводів перетворюються в тригери інших типів.

Універсальний JK-тригер можна використовувати як D-, T- і RS-тригер.

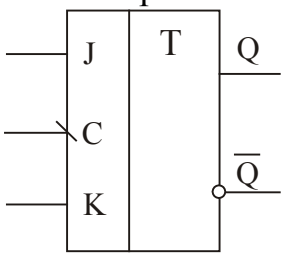


Рис. 10.10. Синхронний RS-тригер.

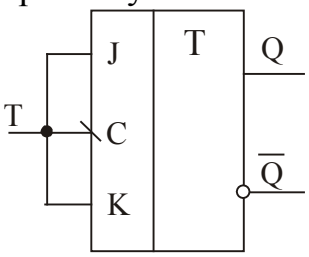


Рис. 10.11. Асинхронний T-тригер.

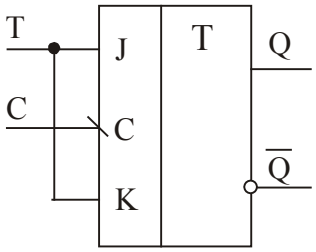


Рис. 10.12. Синхронний T-тригер.

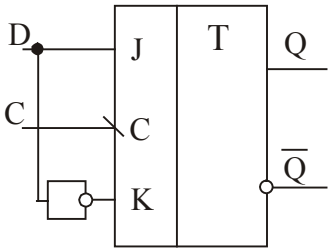


Рис. 10.13. Синхронний D-тригер.

Контрольні запитання

1. Для чого призначені тригери?
2. Назвати типи тригерів.
3. Чим відрізняються синхронні тригери від асинхронних?
4. Написати таблицю переходів RS-, T-, D- і JK-тригерів.
5. Подати умовне графічне зображення тригерів.
6. Зобразити схему RS-тригера на логічних елементах.
7. Як синхронний RS-тригер увімкнути в режимі T-тригера?
8. Як синхронний JK-тригер увімкнути в режимі D-тригера?

11. РЕГІСТРИ

Регістри призначені для зберігання проміжних результатів обчислень.

Усі регістри, залежно від функціональних можливостей, поділяють на два типи: зберігання (пам'яті) (рис. 11.1) і зсуву (рис. 11.2).

Регістри зсуву поділяють:

– за способом введення і виведення інформації на паралельні, послідовні і комбіновані (паралельно-послідовні, послідовно-паралельні);

– за напрямком передання (зсуву) інформації на односпрямовані і реверсивні.

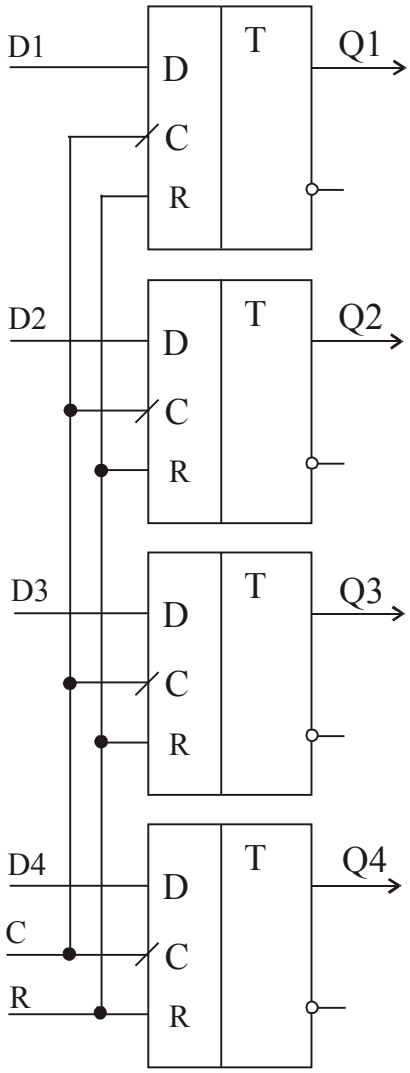


Рис. 11.1. Регістр зберігання: D1 – D4 – паралельні інформаційні входи; C – тактовий вхід; R – вхід скидання регістра в "0"; Q1 – Q4 – виходи регістра.

Регістри зсуву, крім операції зберігання, здійснюють перетворення послідовного двійкового коду в паралельний, а паралельного – в послідовний.

Операція зсуву полягає в тому, що з надходженням кожного тактового імпульсу здійснюється перезапис (зсув) вмісту тригера кожного розряду в сусідній розряд без зміни порядку проходження одиниць і нулів.

При зсуві інформації вправо після кожного тактового імпульсу біт із старшого розряду зсувається в молодший, а при зсуві уліво – навпаки.

Регістри зсуву можуть бути реалізовані на JK- та D-тригерах (рис. 11.2, 11.3).

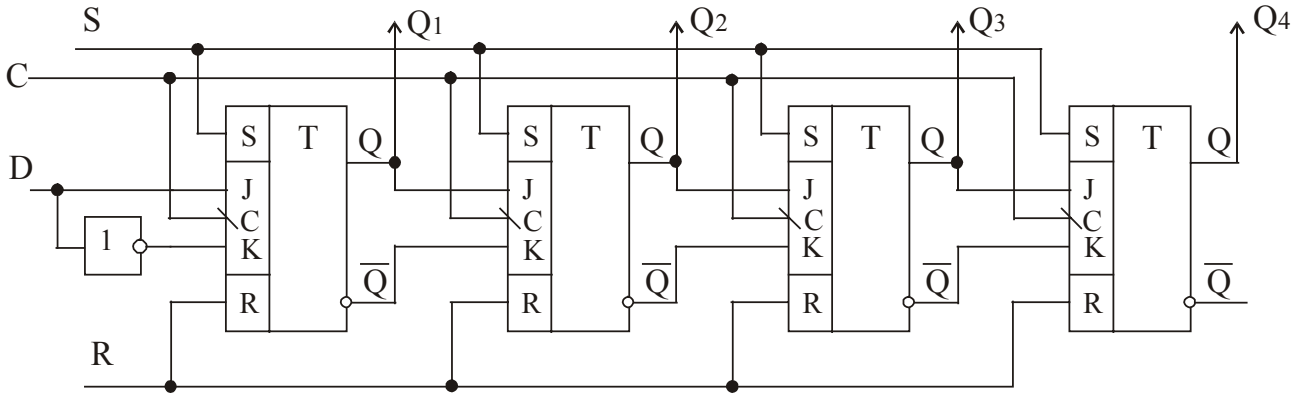


Рис. 11.2. Чотирирозрядний регістр зсуву вправо на JK-тригерах: S – вхід установлення регістра в “1”; C – тактовий вхід; D – інформаційний вхід; R – вхід скидання регістра в “0”; Q1 – Q4 – паралельні виходи регістра.

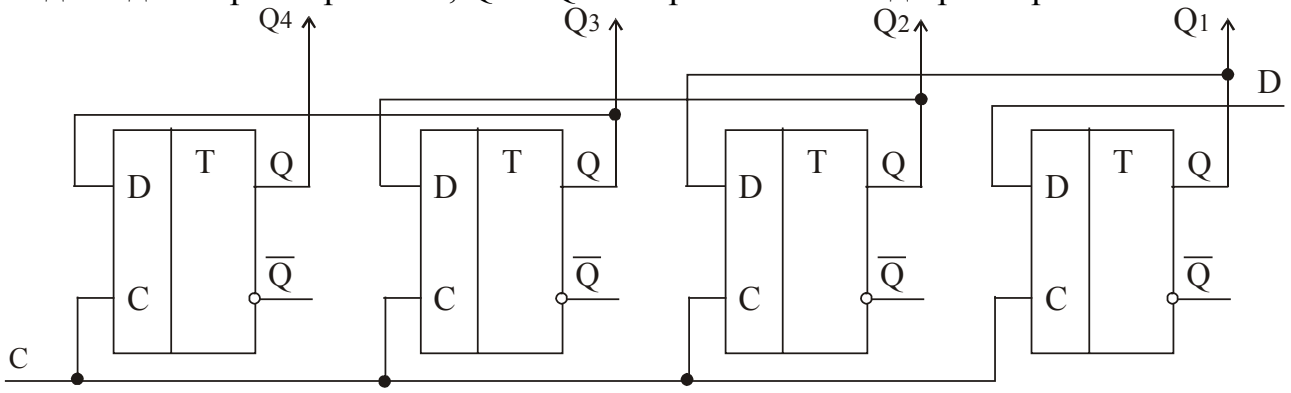


Рис. 11.3. Чотирирозрядний регістр зсуву вліво на D-тригерах.

Універсальний регістр на JK(RS)-тригерах

Режим роботи регістра (рис. 11.4) визначається за сигналом на вході $S/\overline{P}$. Припустимо, що на вході $S/\overline{P}$ сигнал лог. “1” на виході інвертора буде лог. “0”, який закриє логічні елементи DD5.1 – DD5.4 і DD6.1 – DD6.4 та встановить на асинхронних входах тригерів $\overline{S}$ і $\overline{R}$ лог. “1”, що дозволяє синхронну роботу тригерів.

При цьому входи D1 – D4 (для паралельного запису інформації) заблоковані. Тактові імпульси на вході C забезпечують синхронне введення інформації в послідовному коді (з входу DS), а також зсув її вправо. За рахунок інверсії тактових імпульсів елементом DD7.1 спрацювання тригерів відбувається за фронтом зростання тактових імпульсів.

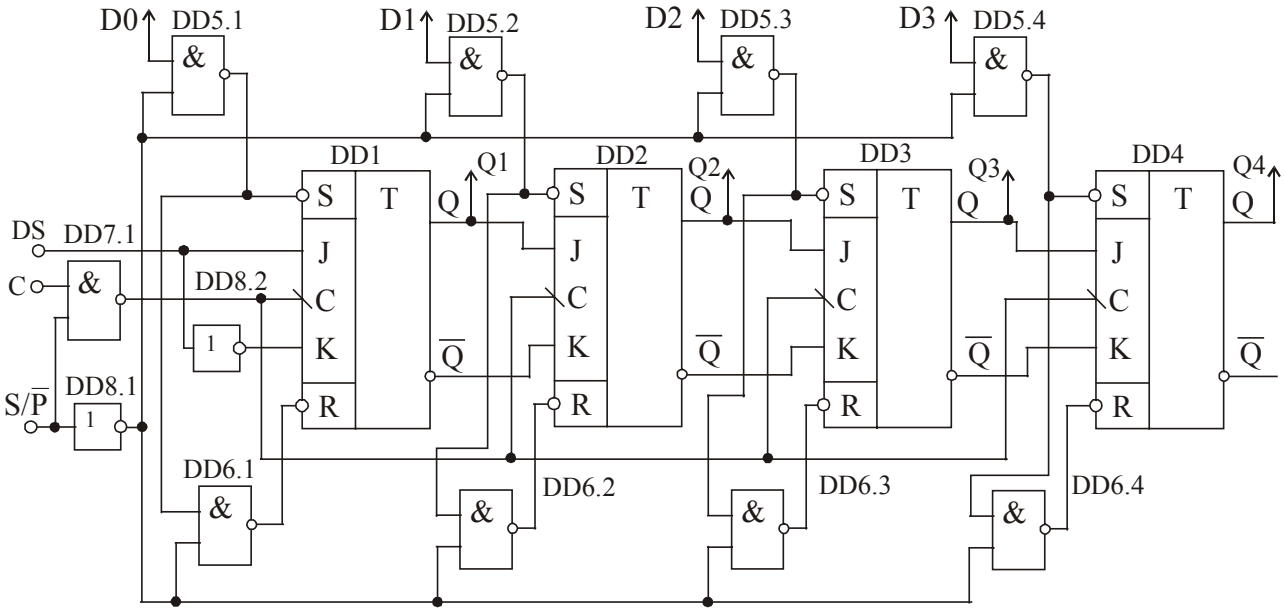


Рис. 11.4. Універсальний регістр: $S/\overline{P}$ – вибір режиму роботи (послідовний/паралельний); D0 – D1 – паралельні інформаційні входи; DS – послідовний інформаційний вхід; C – тактовий вхід; Q1 – Q4 – паралельні виходи.

Якщо на вході $S/\overline{P}$ буде лог. “0”, логічний елемент DD7.1 закритий і тактові імпульси не проходять на С-входи тригерів. Сигнал на загальних входах елементів DD5.1 – DD5.4 і DD6.1 – DD6.4 дорівнює лог. “1”, у результаті цього кожний із цих елементів для сигналів на паралельних входах D1 – D4 служить інвертором. Під дією вхідних сигналів паралельного запису виходи відповідних тригерів набувають того самого стану $Q_i = D_i$. З появою на вході $S/\overline{P}$ сигналу лог. “1” – інформація, введена в паралельному коді, з кожним тактовим імпульсом зсуватиметься на один розряд і видаватиметься у послідовній формі.

Регістри зсуву використовують для реалізації генераторів М-послідовностей (послідовності максимальної довжини, псевдовипадкові послідовності) (рис. 11.5).

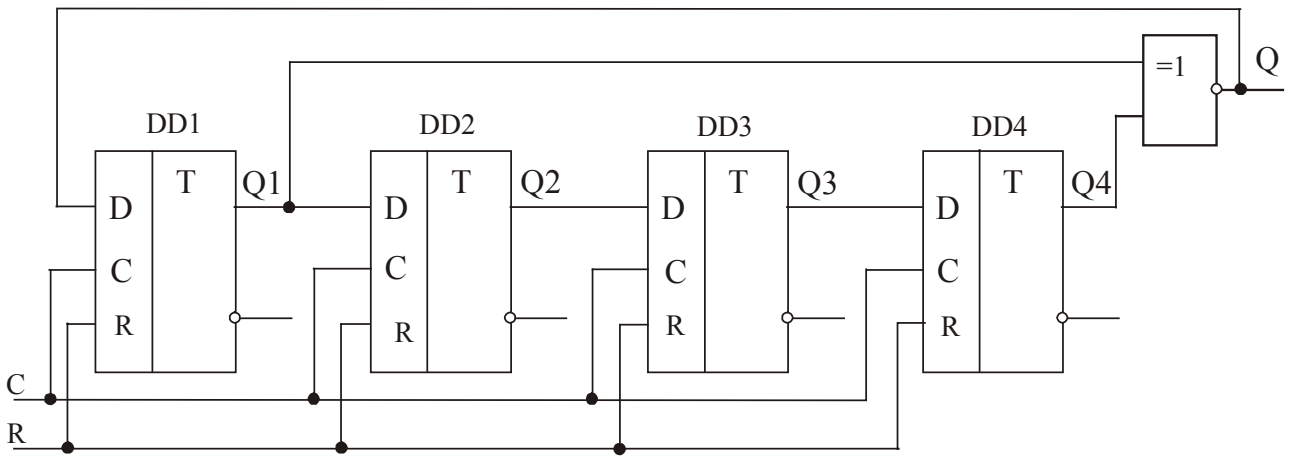


Рис. 11.5. Генератор М-послідовності.

Якщо символи зчитувати з виходу Q, то отримаємо періодичну послідовність: 000010100110111000010100110... з періодом $N = 2^4 - 1 = 15$.

Символи М-послідовності можна зчитувати з будь-якого виходу тригера, при цьому отримаємо послідовність, зсунуту в часі.

Мікрооперації в регістрах

Мікрооперація – це елементарна дія, яка виконується в комп'ютерах в одному машинному такті.

Для запису інформації від кількох джерел на вході кожного тригера ставлять додаткові комбінаційні схеми, які створюють вхідну логіку регістра. Запис кожного слова ініціюється відповідним керуючим сигналом Y_1, Y_2 та ін.

Для запису в регістр кодів A, B потрібно реалізувати таку функцію:

$$D_i = Y_1 \wedge A_i \vee Y_2 \wedge B_i, \qquad (12.1)$$

де A_i і B_i – двійкові розряди слів A і B; Y_1, Y_2 – сигнали керування приймання слів A і B відповідно.

Схема вхідної логіки і-го розряду регістра на основі рівняння (12.1) подано на рис. 12.1.

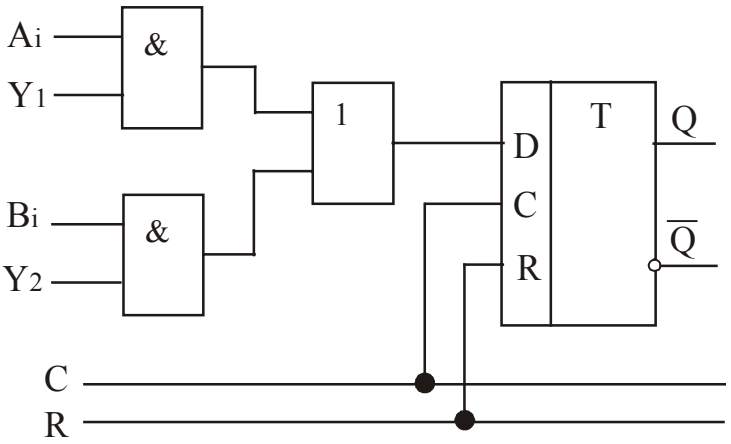


Рис. 12.1. Схема вхідної логіки і-го розряду регістра:

$Y_1 = 1$, $Y_2 = 0$ запис коду A_i;

$Y_1 = 0$, $Y_2 = 1$ запис коду B_i.

Зчитування інформації

Для реалізації мікрооперації зчитування до виходів кожного тригера під'єднують комбінаційні схеми, які створюють вихідну логіку регістра.

Схеми вихідної логіки будуються на основі таких порозрядних логічних рівнянь:

– для зчитування однофазним прямим або оберненим кодом:

$H_i = Y_{pr} \wedge Q_i \vee Y_{ob} \wedge \overline{Q_i}$;

– для зчитування парафазним прямим або оберненим кодом:

$H_i^* = Y_{pr} \wedge Q_i \vee Y_{pr} \wedge \overline{Q_i}$;

$\overline{H}_i^* = Y_{ob} \wedge \overline{Q}_i \vee Y_{ob} \wedge Q_i$;

де Y_{pr} і Y_{ob} – керуючі сигнали відповідно прямого або оберненого коду;

Q_i, $\overline{Q_i}$ – пряме та інверсне значення виходу і-го розряду регістра;

H_i – розряд однофазної шини даних;

H_i^*, $\overline{H}_i^*$ – розряди парафазної шини даних.

Керуючі сигнали Y_{pr} і Y_{ob} не мають збігатися в часі.

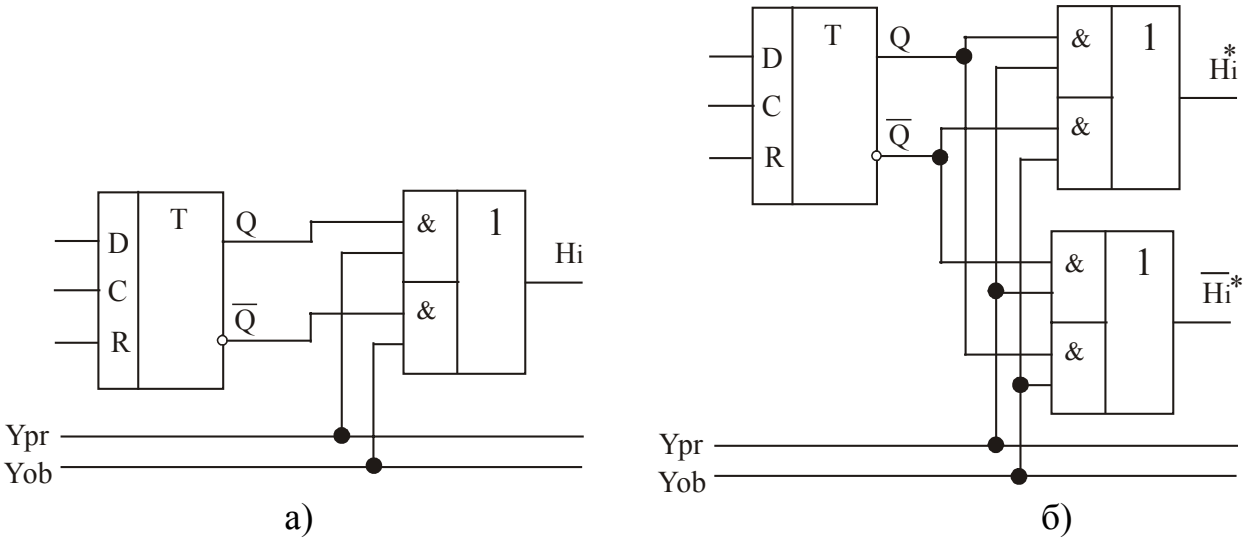


Рис. 12. 2. Схеми вихідної логіки і-го розряду регістра для зчитування інформації: а) однофазним кодом; б) парафазним кодом.

Логічні операції в регістрах

У регістрах можна виконувати такі порозрядні (без перенесень) логічні мікрооперації над словами A_i і B_i:

– логічного додавання і множення:

$RG1 := A \vee B$, $RG1 := A \wedge B$;

– додавання за модулем два і його заперечення:

$RG1 := A \oplus B$, $RG1 := \overline{A \oplus B}$;

– інверсія слова $RG1 := \overline{A}$.

Логічні мікрооперації передбачають наявність першого слова A в регістрі. З урахуванням цього логічне додавання слів A і B в регістрі на RS- або JK-тригерах з однофазним записом виконується введенням слів B без попереднього скидання.

Логічне множення реалізується подаванням інверсних значень розрядів слова B на входи R-тригерів(або K-) регістра. Якщо значення $B_i = 0$, то $\overline{B_i} = 1$ і відповідно тригери обнуляються, що і потрібно для порозрядного логічного множення.

Мікрооперації за модулем два і його заперечення реалізуються в регістрах на T-тригерах. Спочатку записується слово A, а потім без попереднього скидання за логічним входженням вводиться слово B. Після цього на прямих виходах тригерів фіксується результат операції $Q = A \oplus B$, а на інверсних виходах $\overline{Q} = \overline{A \oplus B}$.

Мікрооперація інвертування складається з подавання імпульсу на всі T-входи тригерів регістра, в яких зберігається слово A. В результаті на прямих виходах тригерів установлюється результат згідно із співвідношенням $Q_i = A_i \oplus 1 = \overline{A}_i$.

Мікрооперація зсуву

Зсув – це одночасне просторове переміщення двійкового слова в розрядній сітці із збереженням порядку проходження нулів і одиниці. Мікрооперації зсуву використовують у процесі виконання команд множення, ділення і нормалізації. Крім того, з допомогою зсуву здійснюється перетворення паралельного коду в послідовний або навпаки. Зсув слова можна виконувати вправо (у бік молодших розрядів) або вліво (у бік старших розрядів). Позначимо однорозрядні мікрооперації зсуву вправо і вліво символами R і L відповідно. Розрізняють правий і лівий арифметичний (R_a, L_a), логічний (R_L, L_L) і циклічний (R_Z, L_Z) зсуви слова.

Нехай у регістрі A записано слова $A_n\ A_{n-1} \ldots A_2\ A_1$,

де A_1 – молодший розряд; A_n – старший розряд.

Символічно мікрооперації зсуву записуються таким чином:

– арифметичні зсуви (знаковий розряд не зсувається):

$RGA := R_a(A) = A_n 0 A_{n-1} \ldots A_2$;

$RGA := L_a(A) = A_n A_{n-2} \ldots A_1 0$;

– логічні зсуви (одночасно зсуваються всі розряди):

$RGA := R_L(A) = 0 A_n A_{n-1} \ldots A_2$;

$RGA := L_L(A) = A_{n-1} A_{n-2} \ldots A_1 A_0$;

– циклічні зсуви (між старшим і молодшим розрядами є кільцевий зв'язок):

$RGA := R_Z(A) = A_1 A_n A_{n-1} \ldots A_2$;

$RGA := L_Z(A) = A_{n-1} A_{n-2} \ldots A_1 A_n$.

Арифметичні та циклічні зсуви переважно використовуються при виконанні команд у процесорах, а логічні зсуви забезпечують перетворення послідовного коду в паралельний і навпаки.

Контрольні запитання

1. Для чого призначені регістри?
2. На яких елементах побудовані регістри?
3. Подати схему 3-х розрядного регістра пам’яті (зберігання).
4. Подати схему 3-х розрядного регістра зсуву вправо.
5. Подати схему 3-х розрядного регістра зсуву вліво.
6. Подати схему універсального регістра.
7. Для чого призначені входи універсального регістра?
8. Подати схему генератора М-послідовності.
9. Дати визначення мікрооперації.

12. ЛІЧИЛЬНИКИ

Лічильники – пристрої, які під дією вхідних імпульсів переходять із одного стану в інший і при цьому відображають у певному коді кількість імпульсів, що надійшли на вхід.

Лічильник, який складається із m-тригерів, може порахувати в двійковому коді 2^m імпульсів. Число m визначає кількість розрядів двійкового числа, яке може бути записане в лічильник. Якщо лічильник працює на додавання, то кожний вхідний імпульс збільшує число, записане в лічильник, на одиницю. Якщо лічильник увімкнено на віднімання, то число, що зберігається в лічильнику, з кожним вхідним імпульсом зменшується на одиницю.

Реверсивний лічильник може працювати як на додавання, так і на віднімання.

Лічильники характеризуються модулем (коефіцієнтом) підрахунку. Модуль визначає кількість можливих станів лічильника. Після приходу на лічильник M вхідних сигналів починається новий цикл, який повторює попередній.

За способом кодування внутрішніх станів (за модулем підрахунку) лічильники поділяються на двійкові, двійково-десяткові, із визначеним модулем, із змінним модулем, лічильники Джонсона.

За напрямом підрахунку лічильники є сумуючі, віднімаючі, реверсивні.

За способом організації внутрішніх зв'язків визначають лічильники з послідовним, з паралельним і з комбінованим перенесенням.

У **лічильниках з послідовним перенесенням** імпульси, які підлягають підрахунку, надходять на вхід першого тригера, а сигнал перенесення передається послідовно від одного розряду до іншого (рис. 12.1, 12.2). Такі лічильники складаються з асинхронних Т-тригерів з прямим або інверсним керуванням або JK- і D-тригерів, увімкнених у режимі Т-тригера.

Основна перевага лічильників з послідовним перенесенням – проста схема.

Недоліком є порівняно низька швидкодія, оскільки тригери спрацьовують послідовно один за одним.

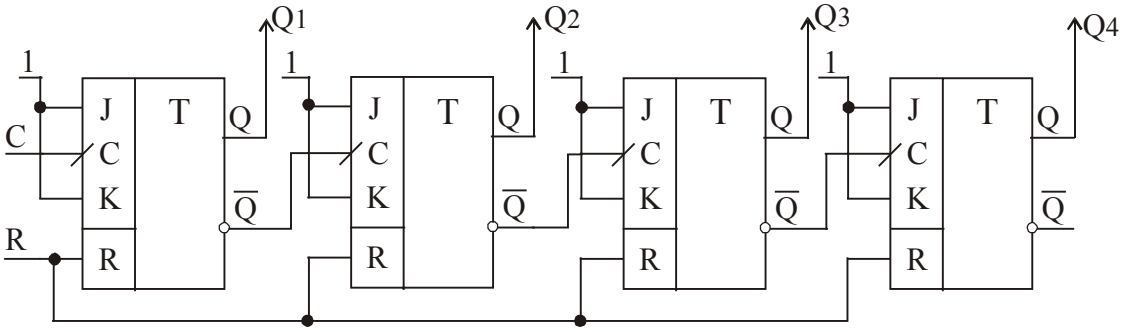


Рис. 12.1. Функціональна схема асинхронного лічильника з послідовним перенесенням.

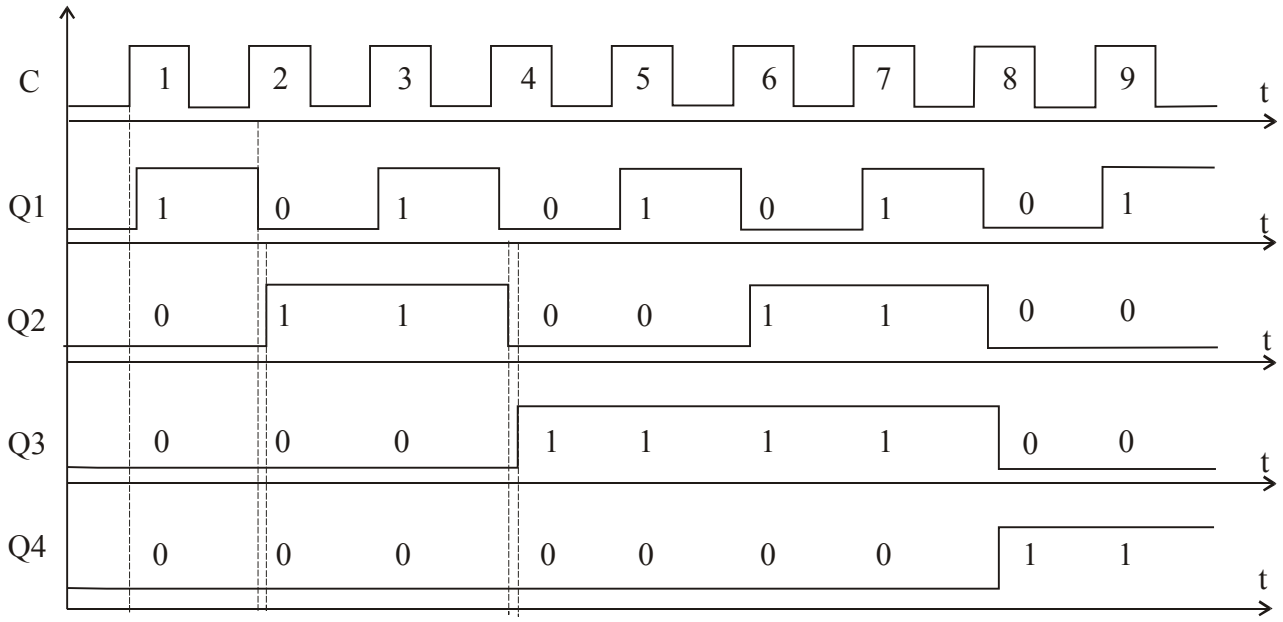


Рис. 12. 2. Часові діаграми

Лічильники з паралельним перенесенням складаються із синхронних JK- і D-тригерів.

Вхідні імпульси надходять одночасно на всі тактові входи, а кожний з тригерів щодо наступного є тільки джерелом інформаційних сигналів (рис. 12. 3).

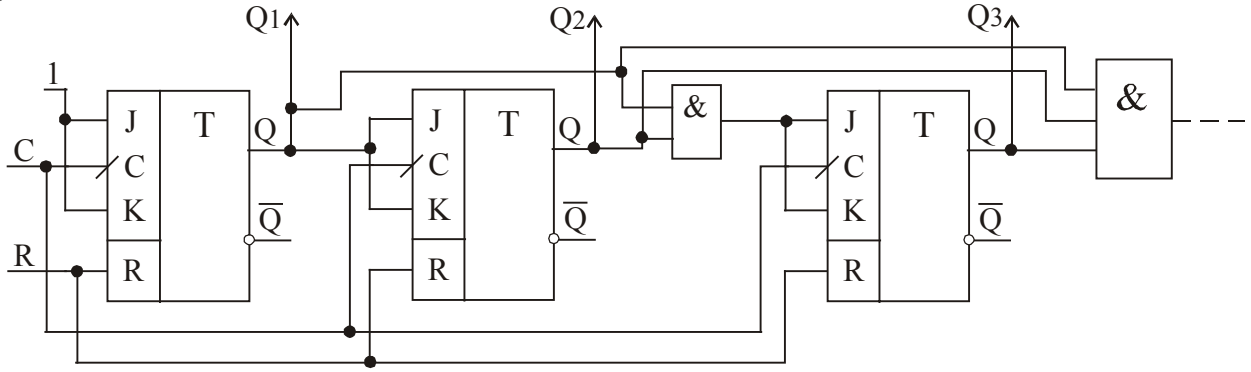


Рис. 12.3. Функціональна схема синхронного лічильника з паралельним перенесенням.

Спрацювання тригерів паралельного лічильника відбувається синхронно і затримка перемикання лічильника дорівнює затримці одного тригера (рис. 12.4).

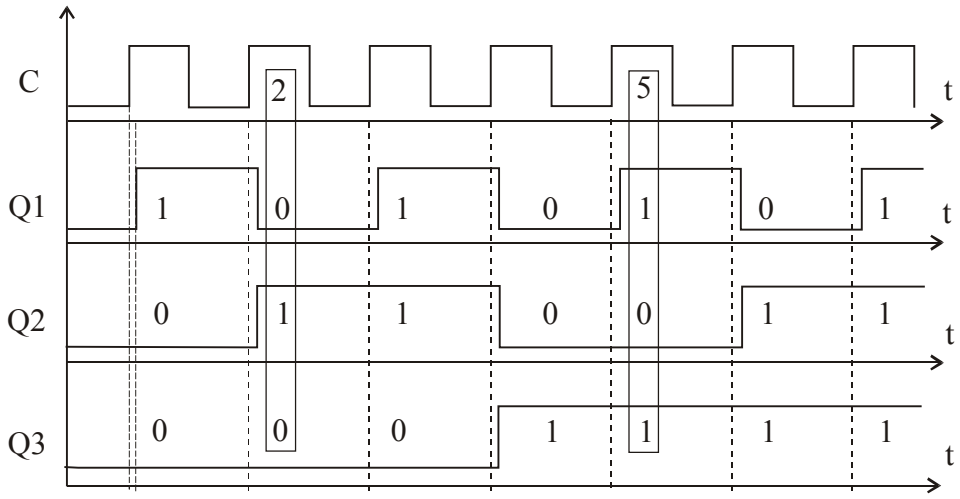


Рис. 12.4. Часові діаграми роботи лічильника.

У лічильниках з паралельно-послідовним перенесенням тригери об'єднані в групи так, що окремі групи утворюють лічильник з паралельним перенесенням, а групи з'єднуються послідовним перенесенням.

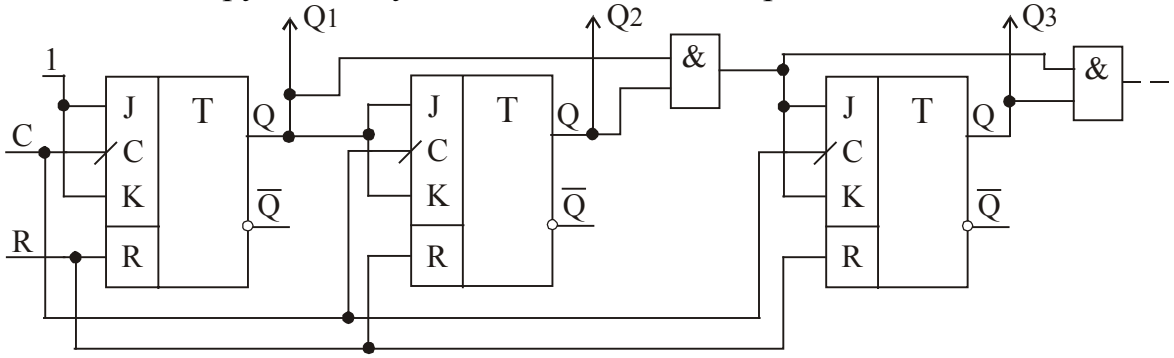


Рис. 12.5. Функціональна схема синхронного лічильника з послідовним перенесенням.

Функціональна схема синхронного реверсивного лічильника подана на рис. 12. 6.

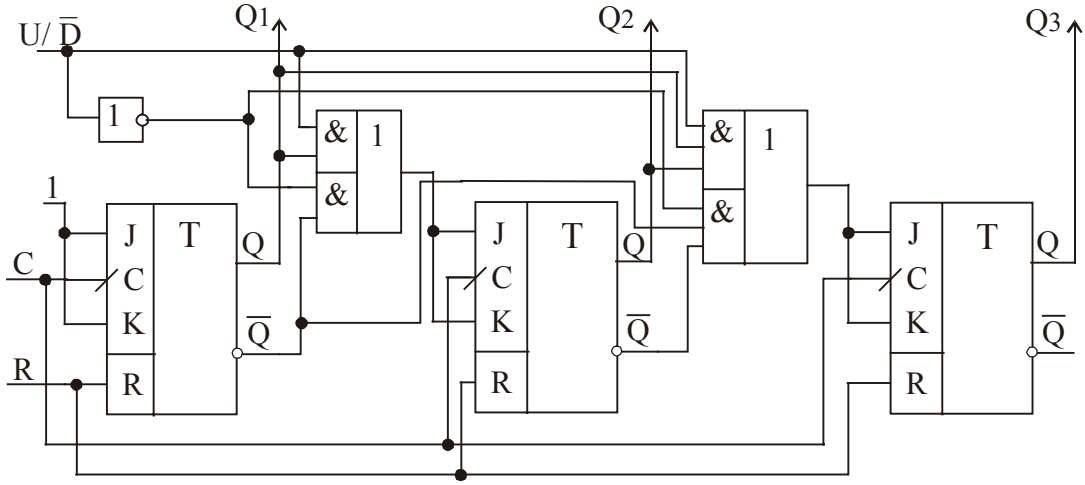


Рис. 12.6. Функціональна схема синхронного реверсивного лічильника: $U/\overline{D}$ – вхід вибору режиму роботи Up/Down (1 – додавання, 0 – віднімання).

Лічильник – це дільник (сумуючий, віднімаючий) з послідовним перенесенням у коді 8421 з коефіцієнтом ділення $K = 2^m$, що складається з послідовно з'єднаних m-тригерів з прямим або інверсним керуванням.

За допомогою додаткового логічного елементу можна змінювати коефіцієнт ділення в межах $2^{m-1} < K < 2^m$. Для цього входи логічного елементу з'єднуються з виходами відповідних тригерів, а вихід логічного елементу – до входів R (скидання тригерів у нуль) (рис. 12.7). Перший тригер спрацьовує від кожного вхідного імпульсу, тобто $1 = 2^0$, другий – від кожного другого імпульсу ($2 = 2^1$), третій тригер спрацьовує від кожного четвертого імпульсу ($4 = 2^2$), а четвертий – від кожного восьмого імпульсу ($8 = 2^3$). Коефіцієнт ділення визначають таким чином: $K = 11 = 8 + 2 + 1 = 2^3 + 2^1 + 2^0$.

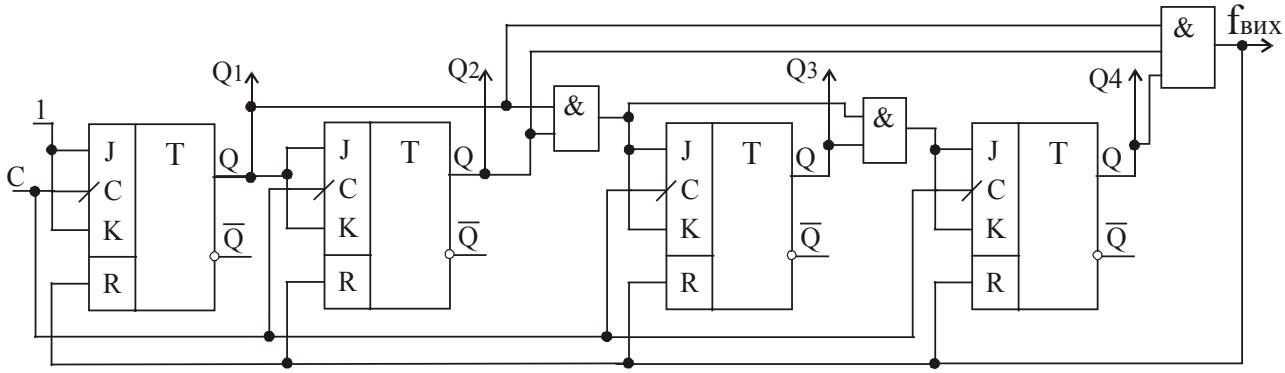


Рис. 12.7. Функціональна схема лічильника з коефіцієнтом ділення К=11.

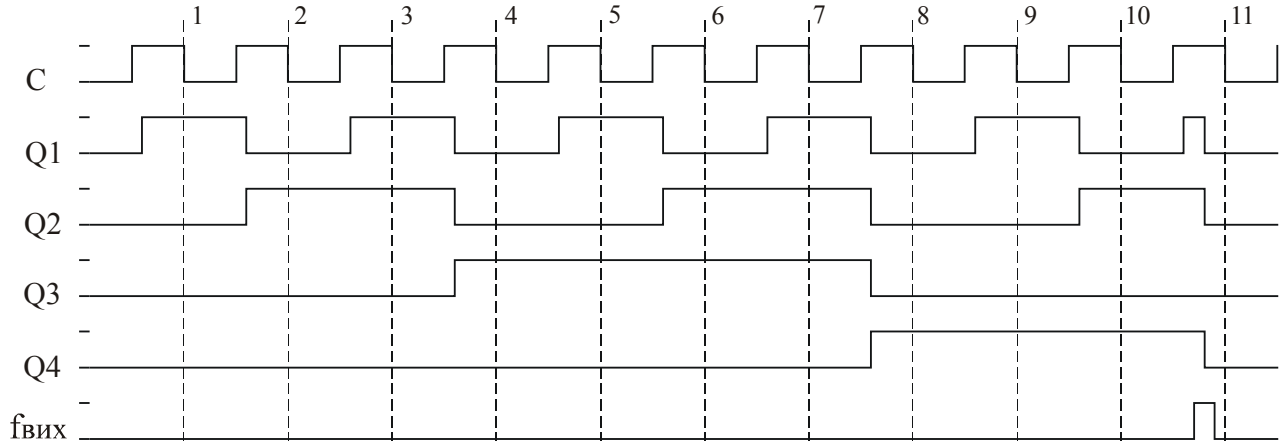


Рис. 12.8. Часова діаграма роботи лічильника з коефіцієнтом ділення К=11.

Контрольні запитання

1. Яке призначення лічильників?
2. На яких елементах побудовані лічильники?
3. Подати схему асинхронного лічильника з послідовним перенесенням.
4. Подати схему синхронного лічильника з паралельним перенесенням.
5. Синтезувати лічильник із заданим коефіцієнтом ділення.

13. ПРОЕКТУВАННЯ ЦИФРОВИХ АВТОМАТІВ З ПАМ'ЯТТЮ

Вузли і пристрої, які містять елементи пам'яті, належать до класу автоматів з пам'яттю.

Цифровий автомат – це пристрій, який здійснює приймання, зберігання і перетворення дискретної інформації за деяким алгоритмом.

Абстрактний цифровий автомат A визначається сукупністю п'яти об'єктів $\{X, S, Y, \varphi, \lambda\}$,

де $X = \{x_i\}$, $i \in \overline{1,m}$ – множина вхідних сигналів автомата А (вхідний алфавіт автомата А);

$S = \{s_j\}$, $j \in \overline{1,n}$ – множина станів автомата А (алфавіт станів автомата А);

$Y = \{y_k\}$, $k \in \overline{1,\ell}$ – множина вихідних сигналів автомата А (вихідний алфавіт автомата А);

φ – функція переходів автомата А, яка відображає $(X \times S) \to S$, тобто визначає відповідність будь-якої пари елементів добутку множин $(X \times S)$ елементам множини S;

λ – функція виходів автомата А, яка задає відображення $(X \times S) \to Y$ або $S \to Y$.

За способом формування функції виходів розрізняють такі типи автоматів: автомат Мілі, автомат Мура (рис. 13.1).

В абстрактному автоматі Мілі функція виходів λ задає відображення $(X \times S) \to Y$.

Автомат Мілі характеризується системою рівнянь:

$$y(t) = \lambda[s(t), x(t)];$$

$$s(t+1) = \varphi[s(t), x(t)].$$

Автомат Мура можна визначити за такою системою рівнянь:

$$y(t) = \lambda[s(t)];$$

$$s(t+1) = \varphi[s(t), x(t)].$$

Синтез цифрових автоматів з пам'яттю має такі етапи:

1) кодування;
2) вибір елементів пам'яті автомата;
3) вибір структурно повної системи елементів (типу автомата);
4) побудова рівнянь булевих функцій виходів і збурення автомата;
5) побудова функціональної схеми автомата.

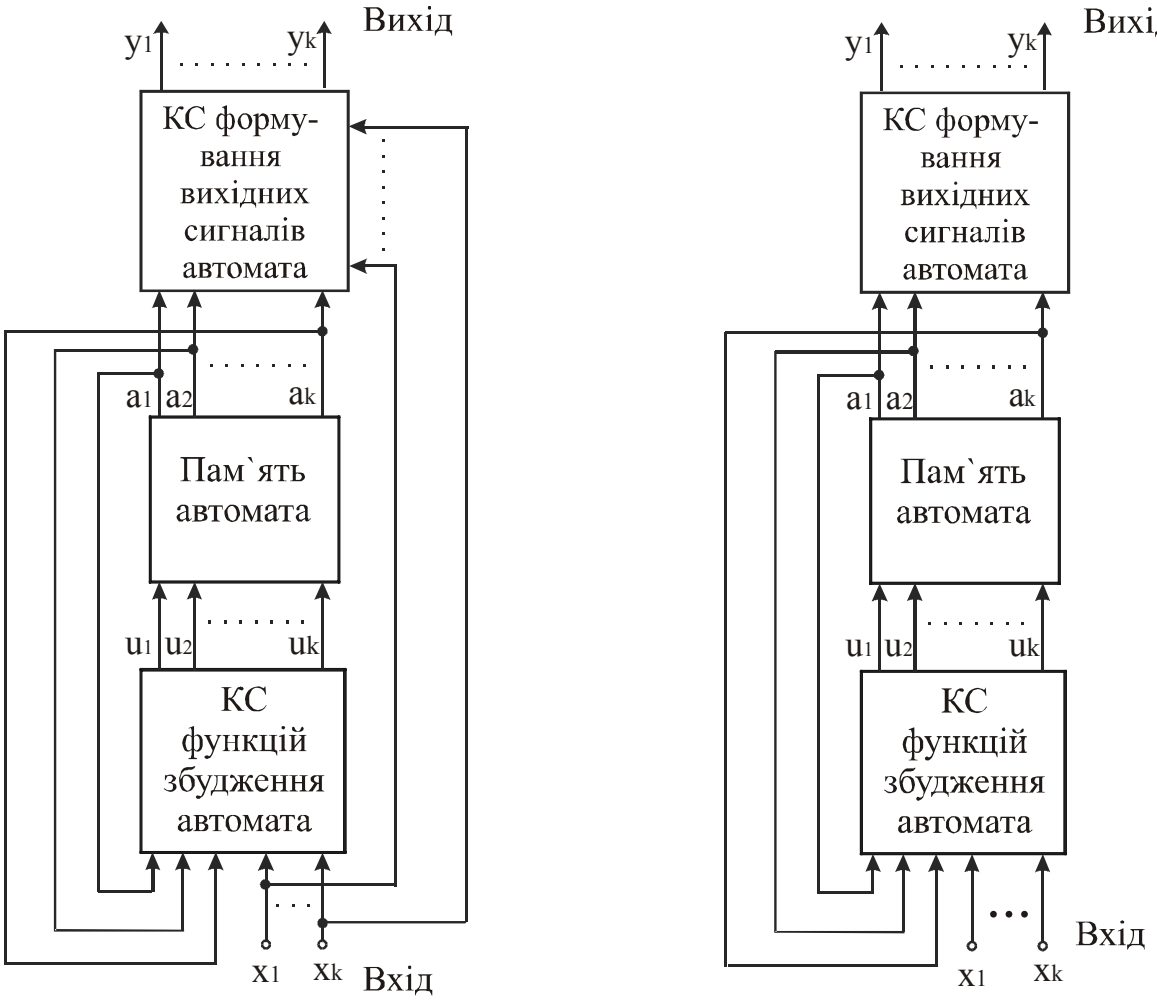


Рис. 13.1. Структурні схеми автоматів з пам’яттю:

1. Кодування

Процес заміни букв алфавітів X, Y і S цифрового автомата двійковими векторами називається кодуванням і може бути описаний таблицею (табл. 13.3, 13.4, 13.5). У лівій частині таблиці перераховуються всі букви (наприклад, вхідного алфавіту), а в правій – двійкові вектори, які ставляться у відповідність цим буквам.

Таблиці переходів і виходів перед кодуванням має такий вигляд (табл. 13.1, 13.2).

Таблиця переходів

Таблиця 13.1

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	s_2	s_1
s_2	s_2	s_1
s_3	s_3	s_2

Таблиця виходів

Таблиця 13.2

Стан автомата	*Вхідні сигнали*	
	x_1	x_2
s_1	y_1	y_3
s_2	y_2	y_4
s_3	y_1	Y_2

Функція переходів має такий вигляд: $s_k = \varphi(s_i, x_j)$; функцію виходів відображено таким чином: $y_k = \lambda(s_i, x_j)$.

Кодування букви X

Таблиця 13.3.

Вхідні сигнали	Код вхідних сигналів
x_1	0
x_2	1

Кодування букви S

Таблиця 13.4.

Стан	Код стану
s_1	00
s_2	01
s_3	10

Кодування букви Y

Таблиця 13.5

Вихідні сигнали	Код вихідних сигналів
y_1	00
y_2	01
y_3	10
y_4	11

Таблиця переходів і виходів після кодування має такий вигляд (табл. 13.6, 13.7).

Таблиця переходів

Таблиця 13.6

Стан автомата	Вхідні сигнали	
	0	1
00	01	00
01	01	00
10	10	01

Таблиця виходів

Таблиця 13.7

Стан автомата	Вхідні сигнали	
	0	1
00	00	10
01	01	11
10	00	01

2. Вибір елементів пам'яті автомата.

Як елементи пам'яті структурного автомата використовують D-тригери, Т-тригери, RS-тригери, JK-тригери.

Таблиці переходів D-тригерів.

Таблиця 13.8

Стан D –тригера	Вхідний сигнал (D)	
	0	1
0	0	1
1	0	1

Таблиця переходів Т-тригерів

Таблиця 13.9

Стан Т –тригера	Вхідний сигнал (Т)	
	0	1
0	0	1
1	1	0

Таблиця переходів PS-тригера

Таблиця 13.10

Стан RS – тригера	Вхідні сигнали (R, S)		
	00	01	10
0	0	1	0
1	1	1	0

Таблиця переходів JK-тригерів

Таблиця 13.11

Стан JK – тригера	Вхідні сигнали (I, K)			
	00	01	10	11
0	0	0	1	1
1	1	0	1	0

Оберемо як елемент пам'яті Т-тригер. Складаємо таблицю функцій збурення автомата.

Таблиця збурення елементів пам'яті будується на основі таблиці переходів (табл.13.6). Таким чином, таблиця набуде такого вигляду (табл. 13.12).

Таблиця переходів.

Таблиця 13.12

Стан автомата	*Вхідні сигнали*	
$a_1\ a_2$	$x=0$	$x=1$
00 01 10	01 00 00	00 01 11
	$u_1\ u_2$	$u_1\ u_2$

На основі даних табл. 13.2 складаємо рівняння. Символами u_1 і u_2 в таблиці позначають функції збурення елементів пам'яті α_1 і α_2.

Складаємо рівняння для побудови комбінаційної схеми збурення цифрового автомата. Отже отримаємо:

$$u_1 = \alpha_1 \wedge \overline{\alpha}_2 \wedge x;$$
$$u_1 = \overline{\alpha}_1 \wedge \overline{\alpha}_2 \wedge \overline{x} \vee \left(\overline{\alpha_1} \wedge \alpha_2 \vee \alpha_1 \wedge \overline{\alpha_2}\right) \wedge x.$$

Таблиця виходів складається на основі табл. 13.7.

Таблиця виходів

Таблиця 13.13

Стан автомата	Вхідні сигнали	
	$x=0$	$x=1$
00 01 10	00 01 00	10 11 01
$\alpha_1\ \alpha_2$	$y_1\ y_2$	$y_1\ y_2$

Складаємо рівняння для побудови КС формування вихідних сигналів автомата. Матимемо:

$$y_1 = \left(\overline{\alpha}_1 \wedge \overline{\alpha}_2 \vee \overline{\alpha}_1 \wedge \alpha_2\right) \wedge x = \overline{\alpha}_1 \wedge x;$$
$$y_2 = \overline{\alpha}_1 \wedge \alpha_2 \wedge \overline{x} \vee \left(\overline{\alpha}_1 \wedge \alpha_2 \vee \alpha_1 \wedge \overline{\alpha}_2\right) \wedge x,$$

де α_1, $\overline{\alpha}_1$ – відповідно прямий і інвертований вихід тригера.

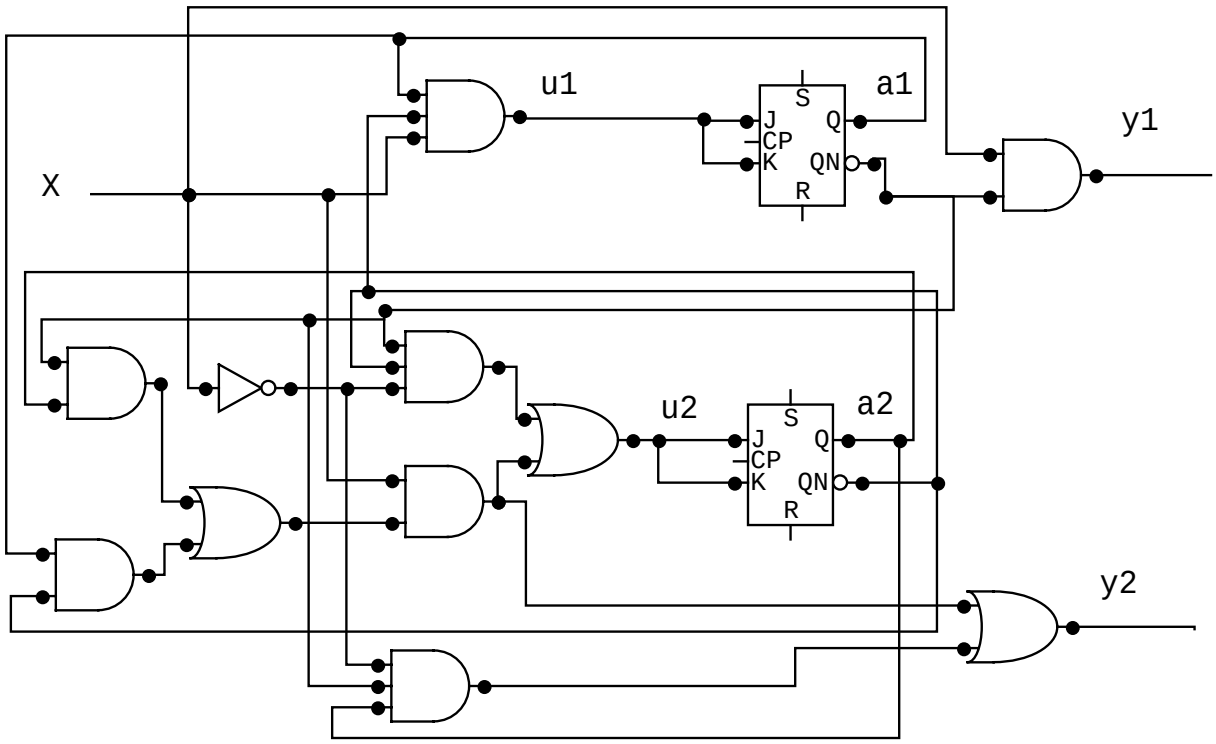


Рис. 13.2. Функціональна схема цифрового автомата Мілі.

Завдання

Спроектувати цифровий автомат Мілі згідно з заданими таблицями переходів і виходів (табл. 13.14 і 13.15, 13.16 і 13.17). Комбінаційну схему збурення і комбінаційну схему формування вихідних сигналів реалізувати в заданому базисі.

Варіант № 1

Таблиця переходів

Таблиця 13.14

Стан автомата	Вхідні дані			
	X1	X2	X3	X4
S1	S3	S3	S1	S5
S2	S5	S2	–	S1
S3	S2	S4	S5	S3
S4	–	–	S5	S5
S5	S4	–	S2	S2

Таблиця виходів

Таблиця 13.15

Стан автомата	Вхідні дані			
	X1	X2	X3	X4
S1	Y6	Y3	Y2	Y4
S2	Y1	Y1	–	Y2
S3	Y2	Y1	Y6	Y6
S4	–	–	Y6	Y2
S5	Y6	–	Y3	Y5

Тип тригера: J-K; базис: І-НЕ.

Варіант № 2

Таблиця переходів

Таблиця 13.16

Стан автомата	Вхідні дані			
	X1	X2	X3	X4
S1	S4	–	S4	S3
S2	S5	S2	–	–
S3	S5	S5	S5	S3
S4	S1	S3	S2	S3
S5	S3	S3	–	S2

Таблиця виходів

Таблиця 13.17

Стан автомата	Вхідні дані			
	X1	X2	X3	X4
S1	Y3	–	Y6	Y2
S2	Y2	Y2	–	–
S3	Y5	Y6	Y6	Y4
S4	Y6	Y2	Y1	Y3
S5	Y2	Y3	–	Y5

Тип тригера: J-K; базис: АБО-НЕ.

Контрольні запитання

1. Дати визначення цифрового автомата.
2. Подати структурну схему автомата Мілі.
3. Подати структурну схему автомата Мура.
4. Чим відрізняється цифровий автомат Мілі від цифрового автомата Мура?
5. Перелічити етапи структурного синтезу цифрових автоматів.

14. МІКРОПРОГРАМНІ АВТОМАТИ

При описуванні функціонування різних засобів обчислювальної техніки доволі часто використовується їх подання у вигляді сукупності керуючого і операційного автоматів (рис.14.1).

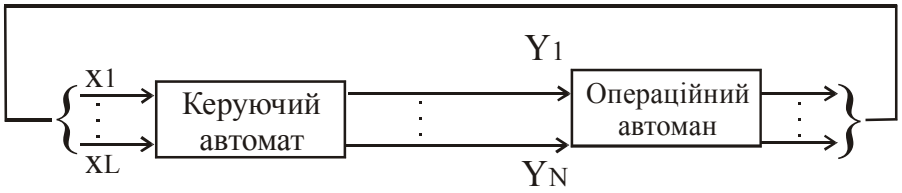


Рис. 14.1. Структурна схема мікропрограмного автомата.

Так, в ЕОМ до системи операційного автомата належать блоки пам'яті, регістри, суматори, канали передання інформації, шифратори тощо, тобто всі пристрої, які виконують певні операції. До системи керуючого автомата належить та частина ЕОМ, яка координує дію перелічених пристроїв, визначаючи послідовність оброблення в них інформації.

Завданням керуючого автомата є вироблення розподіленої в часі послідовності вихідних (керуючих) сигналів, під дією яких в операційному автоматі здійснюється певна операція.

Елементарний неподільний акт оброблення інформації в операційному автоматі, який відбувається протягом одного моменту автоматного часу (одного такту роботи автомата), називається мікрооперацією. Прикладом мікрооперацій є "Зсув інформації", "+1", "Інверсія змінної" та ін.

Якщо в операційному автоматі одночасно здійснюється кілька мікрооперацій, то така множина мікрооперацій називається мікрокомандою.

Не є винятком випадок, коли множина мікрооперацій утворюють мікрокоманду "Пусто". Реалізація такої мікрокоманди в операційному автоматі рівносильна відсутності виконання будь-яких елементарних операцій. У синхронних дискретних пристроях ця мікрокоманда інтерпретується як пропуск такту, коли ніяких сигналів від керуючого автомата на операційний автомат не надходить. Мікрооперації збуджуються вихідними сигналами керуючого автомата, а їх послідовність у часі визначається функціями переходу керуючого автомата. Сукупність мікрокоманд і функцій переходу утворює мікропрограму. Отже, для описування мікропрограми необхідно задати множину мікрокоманд і функцій переходу, які визначають порядок їх виконання.

Для описування мікропрограм зручно використовувати мову граф-схем алгоритмів (ГСА).

Граф-схеми алгоритмів

ГСА називають орієнтований зв'язний граф, який містить одну початкову вершину ("Початок"), одну кінцеву вершину ("Кінець") і довільну кінцеву множину умовних і операторних вершин.

Будь-яка вершина ГСА, крім вершини "Початок", має по одному входу. Вершина "Початок" входів не має. Вершина "Початок" і будь-яка операторна вершина має по одному виходу. Вершина "Кінець" виходів не має. Будь-яка умовна вершина має два виходи, які позначаються символами "Так" і "Ні". Замість цих символів можуть використовуватись цифри "1" і "0" відповідно.

Зображення вершин "Початок", "Кінець", операторної та умовної вершин ГСА подано на рис. 14.2.

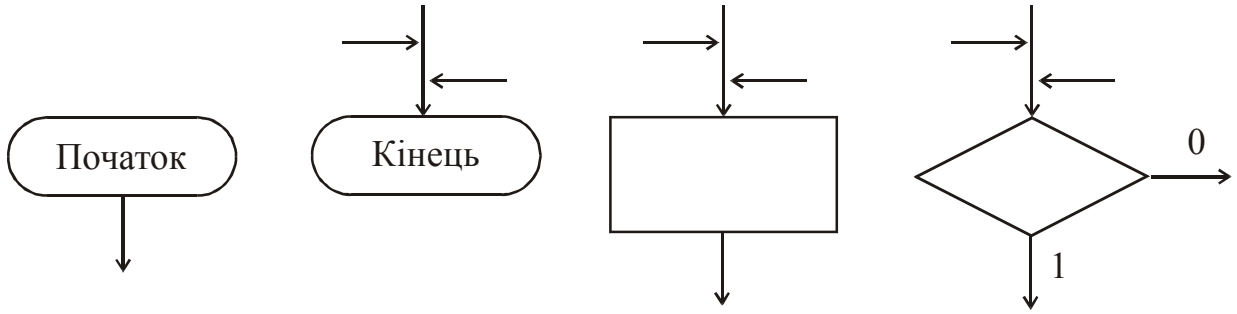


Рис. 14.2. Умовні позначення у ГСА.

ГСА має задовольняти такі умови:

1) входи і виходи вершин з'єднуються один з одним з допомогою дуг, направлених завжди від виходу до входу;

2) кожний вихід з'єднаний тільки з одним входом;

3) будь-який вхід з'єднується з одним виходом;

4) будь-яка вершина ГСА лежить на одному шляху із вершини "Початок" у вершину "Кінець";

5) один із виходів умовної вершини може з'єднуватися з її входом, що неприпустимо для операторної вершини. Такі умовні вершини інколи називають поворотними;

6) у кожній умовній вершині записується логічна умова із множини логічних умов. Дозволяється в різних умовних вершинах записувати однакові логічні умови;

7) в кожній операторній вершині записується оператор, який є вихідним сигналом або сукупністю вихідних сигналів керуючого автомата. Дозволяється в різних операторних вершинах записувати одинакові оператори.

Приклад ГСА, який містить три операторні і дві умовні вершини, подано на рис.14.3.

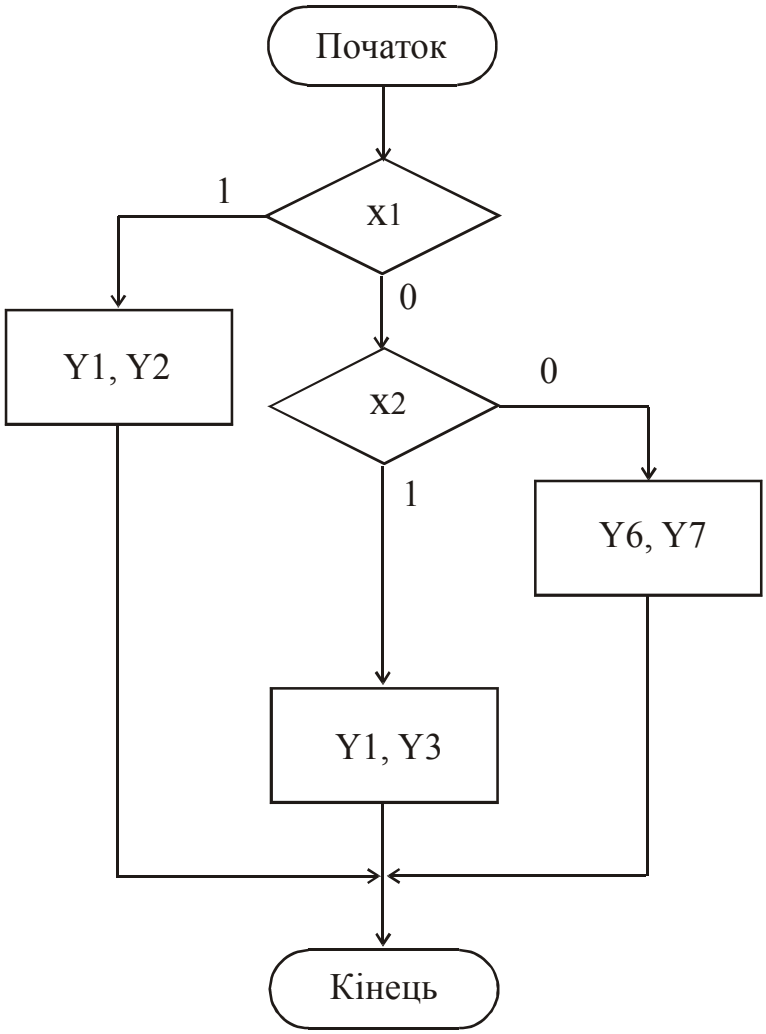


Рис.14.3. Приклад ГСА.

Змістовні граф-схеми алгоритмів

При проектуванні різних пристроїв попередньо складається так звана змістовна ГСА, в якій усередині умовних і операторних вершин записані логічні умови і мікрооперації в змістовних термінах.

Контрольні запитання

1. Дати визначення мікрокоманди.
2. Дати визначення мікропрограми.
3. Подати умовні зображення для побудови ГСА.
4. Подати приклад ГСА.

15. ПРОГРАМОВАНІ ЛОГІЧНІ МАТРИЦІ (ПЛМ)

15.1. Синтез комбінаційних схем і цифрових автоматів з пам'яттю у ПЛМ

Основою програмованих логічних матриць (ПЛМ) є послідовність програмованих матриць елементів І і АБО. До їх структури належать також блоки вхідних і вихідних буферних каскадів (БВх. і Бвих.).

Вхідні буфери формують сигнали необхідної потужності для живлення матриці елементів І. Вихідні буфери забезпечують необхідну потужність виходів, дозволяють або забороняють вихід ПЛМ на зовнішні шини з допомогою сигналу ОЕ (рис. 15.1).

Основними параметрами ПЛМ є число входів m, число термів k і число виходів n.

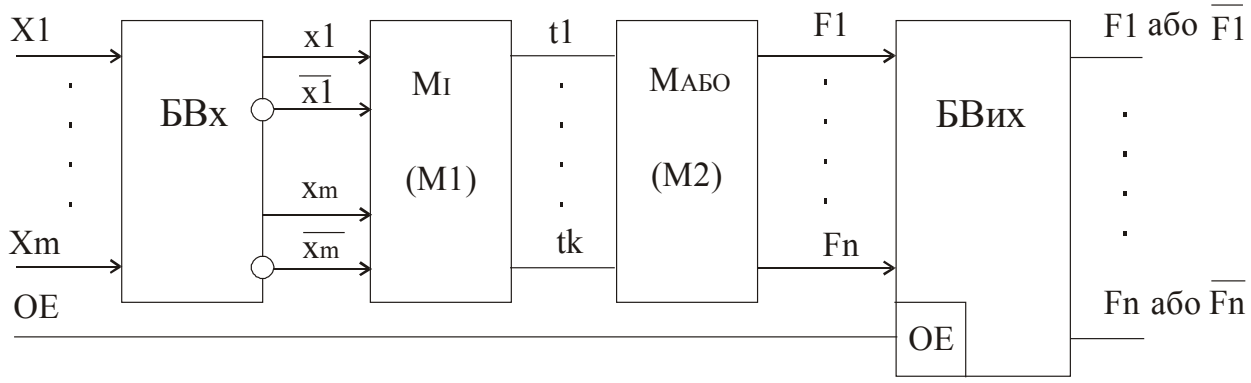


Рис. 15.1. Базова структура ПЛМ.

Змінні $x_1 \ldots x_m$ подаються через БВх на входи елементів І і в матриці І утворюють k термів. Терми – це кон'юнкція, яка позв'язує вхідні змінні, подані в прямій або інверсній формах. Число термів дорівнює числу кон'юкторів, або числу виходів матриці І.

Терми подаються на входи матриці АБО, тобто на входи диз'юнкторів, які формують вихідні функції. Число диз'юнкторів дорівнює функції n.

Таким чином, ПЛМ реалізують диз'юнктивну нормальну форму (ДНФ).

ПЛМ може реалізувати систему n логічних функцій від m аргументів, яка містить не більш ніж k термів.

У матрицях є горизонтальні та вертикальні зв'язки, у вузлах перетину яких при програмуванні створюються або ліквідуються елементи зв'язку (рис. 15.2).

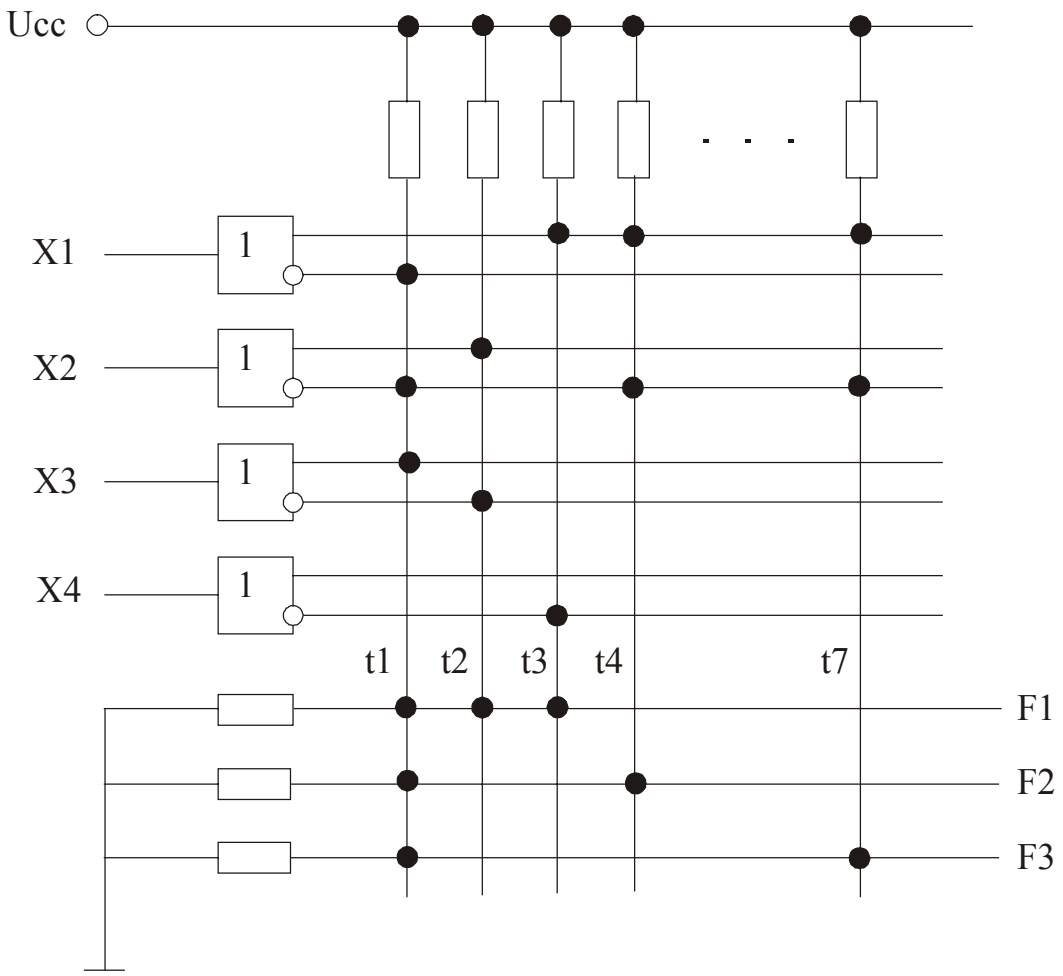


Рис. 15.2. Матриці кон’юнкції та диз’юнкції.

З допомогою ПЛМ можна реалізувати не тільки диз’юнктивні нормальні форми булевих функцій, а й функції з дужками.

У цьому разі спочатку реалізують вирази в дужках, а потім вони розглядаються як аргументи для отримання кінцевого результату.

В схемі виникають зворотні зв’язки – проміжні результати з виходу знову подаються на вхід, логічна глибина схеми збільшується, як і затримка вироблення результату.

Отримуємо таку функцію:

$$F = x_1 \wedge x_2 \vee \left(x_1 \wedge x_2 \vee \overline{x_2} \wedge \overline{x_1}\right) \wedge x_3 .$$

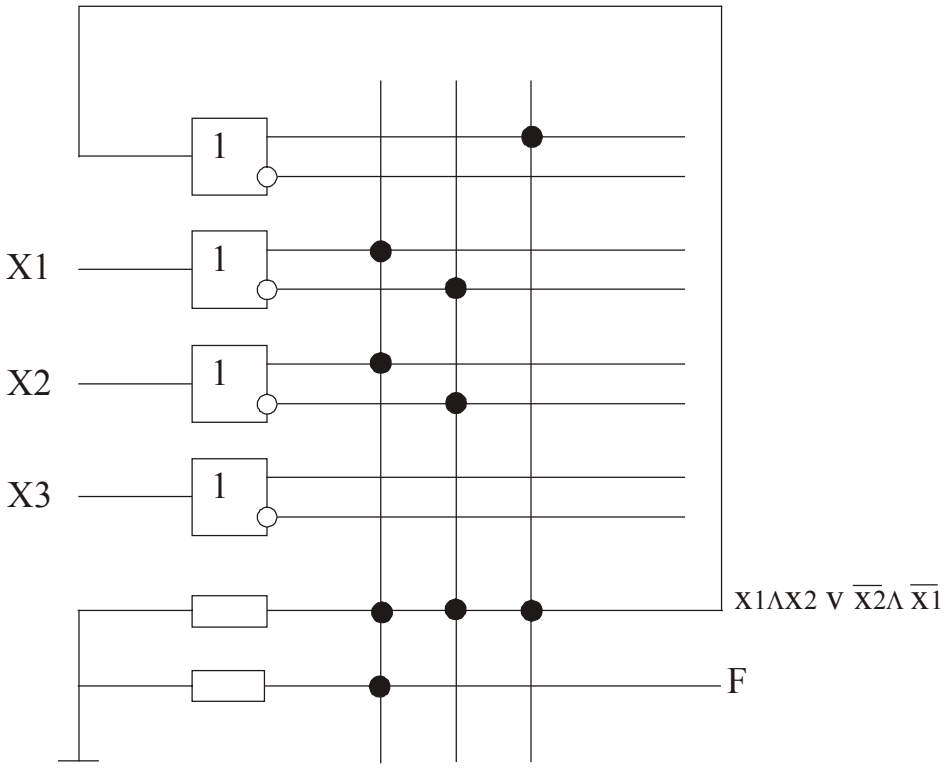


Рис. 15.3. Реалізація булевої функції F.

16.2. ПЛМ з пам'яттю.

Ці схеми дають змогу будувати автомати найзручнішим способом, тобто, крім комбінаційної частини, містять на кристалі тригери (D-тригери) (рис.15.4).

ПЛМ з пам'яттю характеризуються такими параметрами: крім трьох попередніх параметрів вона має і параметр r – число елементів пам'яті.

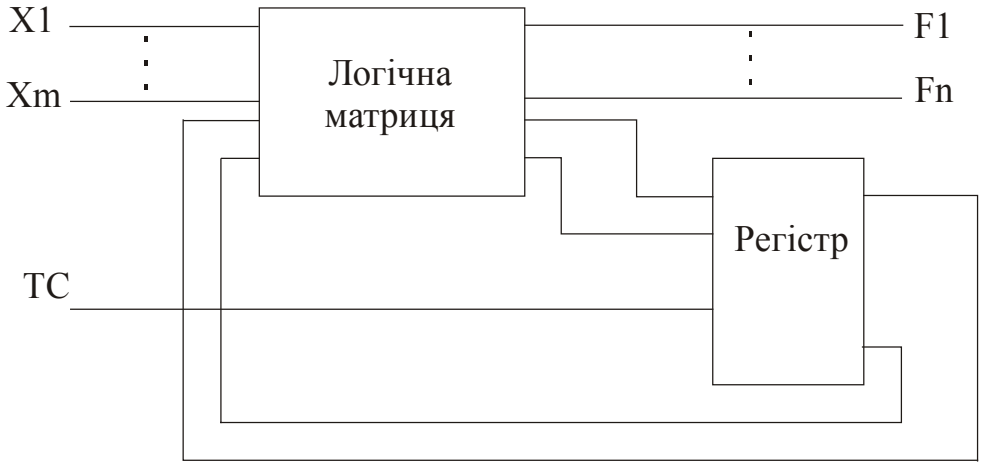


Рис. 15.4. Структура ПЛМ з пам'яттю.

Структура ПЛМ з пам'яттю збігається з канонічною схемою автомата.

Результати обробки інформації залежить від результатів попередніх кроків, що забезпечують зворотним зв'язком з регістра на вхід ПЛМ. Максимальна кількість внутрішніх станів автомата становить 2^n. Автомат вважають синхронним зворотним зв'язком, який активізується тільки за тактовими сигналами (ТС).

Контрольні запитання

1. Назвати переваги використання ПЛМ?
2. Подати базову структуру ПЛМ.
3. Реалізувати задану булеву функцію на базі ПЛМ.
4. Подати структуру ПЛМ з пам'яттю.

ЛІТЕРАТУРА

1. Прикладная теория цифровых автоматов / Самофалов К. Г., Романкевич А. М. и др. – К.: Вища школа, 1987. – 369 с.

2. Савельєв А. Я. Прикладная теория цифровых автоматов: Уч. для вузов. – М.: Высш. шк., 1987. – 272 с.

3. Майоров С. А.,Новиков Г. И. Принципы организации цифровых машин. – Л.: Машиностроение, 1974. – 431 с.

4. Баранов С. И. Синтез микропрограммных автоматов. – Л.: Энергия, 1979 – 232 с.

5. Пухальський Г. И., Новосельцева Т. Я. Цифровые устройства: Уч. пособие для вузов. – СПб.: Политехника, 1996. – 885 с.

6. Бабич М. П., Жуков І. А. Комп'ютерна схемотехніка: Навч. посібник. – К.: "МП-Прес", 2004. – 412 с.

7. Угрюмов Е. П. Цифровая схемотехника. – СПб.: БХВ – Петербург, 2001. – 528 с.

8. Новиков Ю. В. Основы цифровой схемотехники. Базовые элементы и схемы. Методы проектирования. – М.: Мир, 2001. –379 с.

9. Майоров С. А., Новиков Г.И. Принципы организации цифровых машин. – Л.: Машиностроение, 1974. – 431 с.

10. Сапожников В. В., Сапожников В. В. Методы синтеза надежных автоматов. – Л.: Энергия, 1980. – 96 с.

11. Карцев М. А., Брик В. А. Вычислительные системы и синхронная арифметика. – М.: Радио и связь, 1981. – 360 с.

12. Акушский И. Я., Юдицкий Д. И. Машинная арифметика в остаточных классах. – М.: Сов. радио, 1968. – 460 с.

13. Торгашев В. А. Система остаточных классов и надёжность ЦВМ. – М.: Сов. радио, 1973. – 274 с.

14. Стешенко В. ПЛИС фирмы ALTERA: проектирование устройств обработки сигналов – М.: "Додека", 2000. – 224 с.

15. Зельдин Е. А. Цифровые интегральные микросхемы в информационно-измерительной аппаратуре. – Л.: Энергоатомиздат, 1986. – 280 с.

Навчально-методичне видання

ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ

Опорний конспект лекцій
(спеціальності 6.09150 – спеціалізовані
комп'ютерні системи,
6.091500 – комп'ютерні системи та мережі)

Редактор	Оксана Бойчук
Технічний редактор	Лариса Щербак
Комп'ютерна верстка	Ігор Барський

Підписано до друку 20. 09. 2006 р.
Формат 60x84/16. Папір офсетний. Друк на різографі.
Умов.-друк. арк. 0,95. Обл.-вид. арк. 0,90. Зам. №
Тираж 100 прим.

Віддруковано у видавництві ТНЕУ
"Економічна думка"
46004, Тернопіль, вул. Львівська, 11,
тел. (0352) 43-22-18, факс (0352) 43-24-40
E-mail: edition@tane.edu.ua