

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ОПОРНИЙ КОНСПЕКТ ЛЕКЦІЙ

з курсу

“КОМП'ЮТЕРНА ЛОГІКА”

для студентів спеціальностей:

6.050102 *“Спеціалізовані комп'ютерні системи”*,

6.050102 *“Комп'ютерні системи та мережі”*

ТЕРНОПІЛЬ – 2011

МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ОПОРНИЙ КОНСПЕКТ ЛЕКЦІЙ

з курсу

“КОМП'ЮТЕРНА ЛОГІКА”

Для студентів спеціальностей:

6.050102 *“Спеціалізовані комп'ютерні системи”*,

6.050102 *“Комп'ютерні системи та мережі”*

ТЕРНОПІЛЬ – 2011

Опорний конспект лекцій з курсу “Комп’ютерна логіка” для студентів спеціальностей “Спеціалізовані комп’ютерні системи” та “Комп’ютерні системи та мережі” / Укл.: Яцків В. В. – Тернопіль: Економічна думка, 2011. – 90 с.

Відповідальний за випуск: д.т.н., професор Николайчук Я.М.

Рецензенти: к.т.н., доцент Кочан В.В.;

к.т.н. доцент Сегін А.І.

Методичні вказівки розглянуті та схвалені на засіданні кафедри Спеціалізованих комп’ютерних систем. Протокол № 2 від 16. 09. 11 р.

ЗМІСТ

| | |
|---|----|
| Вступ | 4 |
| 1. Арифметичні основи комп'ютерів | 5 |
| 2. Система залишкових класів | 17 |
| 3. Елементи математичної логіки | 26 |
| 4. Мінімізація перемикальних функцій | 33 |
| 5. Синтез комбінаційних схем | 36 |
| 6. Аналіз комбінаційних схем | 40 |
| 7. Синтез дешифраторів та шифраторів | 45 |
| 8. Синтез мультиплексорів та демультимплексорів | 48 |
| 9. Синтез суматорів | 52 |
| 10. Елементарні цифрові автомати | 56 |
| 11. Регістри | 64 |
| 12. Мікрооперації в регістрах | 68 |
| 13. Лічильники | 71 |
| 14. Проектування цифрових автоматів з пам'яттю | 76 |
| 15. Мікропрограмні автомати | 82 |
| 16. Програмовані логічні матриці | 85 |
| Література | 89 |

Вступ

Дисципліна “Комп’ютерна логіка” є однією з базових в системі знань і вмінь, що формують бакалавра та інженера - системотехніка за спеціальностями “Спеціалізовані комп’ютерні системи” та “Комп’ютерні системи та мережі”.

Метою викладення дисципліни “Комп’ютерна логіка” 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 та керуючих автоматів. Вивчення дисципліни “Комп’ютерна логіка” дає студентам необхідну теоретичну і практичну підготовку для того, щоб вміти розробляти і аналізувати алгоритми переробки дискретної інформації складних процесів, складати структурні схеми комбінаційних логічних схем та автоматів з пам’яттю, ефективно розв’язувати практичні задачі з прикладної теорії цифрових автоматів з використанням ЕОМ.

Дисципліна "Комп’ютерна логіка" відноситься до схемо - та системотехнічного напрямку підготовки бакалаврів. Вона є першою дисципліною цього напрямку. Дисципліна є базовою для курсів: "Комп’ютерна електроніка", "Архітектура ЕОМ", "Периферійні пристрої", "Основи автоматизації проектування засобів ОТ", "Архітектура обчислювальних систем".

1. АРИФМЕТИЧНІ ОСНОВИ КОМП'ЮТЕРІВ

Позиційні системи числення.

Способи переведення чисел з однієї системи числення в другу.

Арифметичні дії в різних системах числення.

Позиційні системи числення.

1. Принципи побудови систем числення.

Числова інформація в комп'ютерах характеризується:

- системою числення (двійкова, десяткова та інші);
- видом числа (числа дійсні, комплексні, масиви);
- типом числа (змішане, ціле, дробове);
- формою представлення 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 називається 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.

В 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 десяти використовують десяткові цифри, а для основи більшої десяти додають букви латинського алфавіту – А, В, С, D, E, F (табл. 1.1, табл.1.2).

Таблиця 1.1 – Алфавіт систем числення

| 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 записі числа називається його 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 використовуються сім знаків – I, V, X, L, C, D, M, таким значенням:

| | | | | | | |
|---|---|----|----|-----|-----|------|
| 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.

Таблиця 1.2 – Позиційні системи числення

| $q=10$ | $q=2$ | $q=16$ | $q=8$ | $q=5$ | $q=3$ |
|--------|---------|--------|-------|-------|-------|
| 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 з двома стійкими станами (тригерів), призначених для зберігання значень двійкового розряду – цифр 0 або 1.

Двійкові цифри називають також бітами. Назву БІТ у 1946 році запропонував видатний американський вчений статистик Джон Тьюкі.

Система числення повинна забезпечувати:

- можливість представлення 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 представити поліномом:

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

де q – основа системи числення;

q^i – вага позиції;

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

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

$-1, -2, \dots, -m$ – 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: $q = 8$; $a_i \in \{0, 1, \dots, 7\}$,

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

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

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

Способи перевodu чисел з однієї системи числення в другу

Існують два основних способи перевodu числа із однієї системи числення в другу: **табличний** і **розрахунковий**.

Табличний спосіб прямого перевodu оснований на співставленні таблиць відповідності чисел різних систем числення. Цей спосіб дуже громіздкий і вимагає великого об'єму пам'яті для зберігання таблиці, але його можна використати для будь-яких систем числення (не тільки для позиційних).

Перевід цілих чисел із однієї позиційної системи числення в іншу

Нехай задано число A в довільній позиційній системі числення з основою q і його необхідно перевести в нову систему з основою P .

тобто $A = Q_n q^n + Q_{n-1} q^{n-1} + \dots + Q_1 q^1 + Q_0 q^0$,
 $Q_1 = 0 \div (q - 1)$.

Необхідно перетворити до виду:

$$A = Q_n P^n + Q_{n-1} P^{n-1} + \dots + Q_1 P^1 + Q_0 P^0 \tag{1.1}$$

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

Вираз (1.1) можна записати:

$$A = A_1 P + Q_0,$$

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

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 до тих пір, поки не будуть виконуватися співвідношення:

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

Правило перевodu: щоб перевести ціле число із однієї позиційної системи числення в другу, необхідно задане число послідовно ділити на основу нової системи числення, записаної в числах старої (заданої) системи числення до одержання частки рівної 0.

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

Приклади перевodu.

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 десятичної системи числення в вісімкову.

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

Отже $92_{10} = 134_8$.

Переведемо число 168 з десятичної системи числення в шістнадцяткову.

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

Отже $168_{10} = A8_{16}$.

При переводі із двійкової системи числення в десятичну задане число необхідно ділити на основу нової системи числення тобто на 1010_2 .

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

Розрахунки проводяться в десятичній системі числення.

Приклад.

1) Перевести двійкове число 10010100 в десятичну систему $2 \rightarrow 10$:

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

2) $8 \rightarrow 10$:

$$A = 235_8;$$

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

3) $16 \rightarrow 10$:

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

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

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

Перевід правильних дробів.

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

Дріб в новій системі числення запишеться в виді цілих частин добутку, починаючи з першої частини.

Приклад: Перевести правильний дріб 0,456 із десяткової системи числення в двійкову і вісімкову.

1) При переводі із десяткової системи в двійкову множимо заданий дріб на 2, а при переводі в вісімкову – на 8.

| Ціла частина | Дробова частина | Ціла частина | Дробова частина |
|--------------|-----------------|--------------|-----------------|
| 0 | 456 | | 456 |
| 0 | 912 | 0 | x |
| 1 | 824 | | <u>8</u> |
| 1 | 648 | 3 | 648 |
| 1 | 296 | | <u>8</u> |
| 0 | 592 | 5 | 184 |
| 1 | 184 | | <u>8</u> |
| 0 | 368 | 1 | 472 |

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

2) При переводі із двійкової системи в десяткову множимо задане двійкове число на десять записане у двійковій системі числення (1010_2):

$$\begin{array}{r} \times 0,011101 \\ \quad 1010 \\ \hline 000000 \\ + 011101 \\ 000000 \\ \hline 011101 \\ \hline 100,100010 \end{array} \quad \begin{array}{r} \times 100,100010 \\ \quad 1010 \\ \hline 000000 \\ + 100010 \\ 000000 \\ \hline 100010 \\ \hline 101,010100 \end{array} \quad \begin{array}{r} \times 101,010100 \\ \quad 1010 \\ \hline 000000 \\ + 010100 \\ 000000 \\ \hline 010100 \\ \hline 011,100100 \end{array}$$

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

Перевід неправильних дробів.

При переводі неправильних дробів необхідно окремо перевести цілу і дробову частини числа по вище розглянутих правилах переводу і записати в новій системі числення, залишивши без зміни положення коми.

Перевід чисел із системи числення в систему з кратною основою.

Якщо основи систем числення кратні одна одній, тобто зв'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 того щоб перевести число із заданої системи числення в нову систему, основа якої кратна основі заданої системи, необхідно кожен цифру числа записати за допомогою m цифр в новій системі числення, якщо основа заданої системи більша за основу нової системи.

Наприклад, при переводі вісімкового числа 254_8 в двійкову систему числення достатньо кожен цифру вісімкового числа записати в виді двійкової тріади, так як $8 = 2^3$, $254_8 = 010101100_2$.

При переводі двійкового числа в шістнадцяткову систему достатньо кожен тетраду заданого числа записати в виді шістнадцяткової цифри $2^4 = 16$, $010101100 = AC$.

Вибір системи числення для використання в ЕОМ.

При виборі системи числення необхідно враховувати такі фактори:

1. Наявність фізичних елементів, здатних відтворити символи системи.
2. Економічність системи, тобто кількість елементів необхідних для представлення багаторозрядних чисел.
3. Трудоемність виконання операцій в ЕОМ.
4. Швидкодія 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.

Арифметичні дії в q-ричній системі числення

Розглянемо основні арифметичні операції: **додавання, віднімання**. Правила виконання цих операцій в десятковій системі добре відомі. Ці правила можна застосувати і до всіх інших позиційних систем числення. Тільки таблицями додавання і множення треба користуватися особливими для кожної системи.

Додавання.

Додавання в двійковій системі.

| | | |
|---|---|----|
| + | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 1 | 10 |

Додавання в вісімковій системі.

| | | | | | | | | |
|---|---|----|----|----|----|----|----|----|
| + | 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 він переноситься вліво (формується старший розряд).

Приклад.

Додамо числа в різних системах числення (СЧ).

Двійкова СЧ. 

$$\begin{array}{r}
 10101 \\
 + 0111 \\
 \hline
 11100 \\
 \begin{array}{l}
 \boxed{1+1=2=2+0} \\
 \boxed{0+1+1=2=2+0} \\
 \boxed{1+1+1=3=2+1} \\
 \boxed{0+0+1=1}
 \end{array}
 \end{array}$$

$$\begin{array}{r}
 10101 \\
 + 0111 \\
 \hline
 11100 \\
 \begin{array}{l}
 \boxed{1+1=2=10} \\
 \boxed{0+1+1=2=10} \\
 \boxed{1+1+1=3=11} \\
 \boxed{0+0+1=1}
 \end{array}
 \end{array}$$

Вісімкова СЧ. $16_8 + 7_8 = 25_8$

$$\begin{array}{r}
 + 16 \\
 \underline{7} \\
 25 \\
 \begin{array}{l}
 \boxed{7+6=13=8+5} \\
 \boxed{1+1=2}
 \end{array}
 \end{array}$$

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

$$\begin{array}{r}
 D \\
 \underline{5} \\
 12 \\
 \begin{array}{l}
 \boxed{13+5=18=16+2}
 \end{array}
 \end{array}$$

Віднімання.

Двійкова СЧ. Правила віднімання:

$$\begin{array}{r}
 \underline{1} \quad \underline{1} \quad \underline{0} \quad \underline{1} \\
 \underline{1} \quad \underline{0} \quad \underline{0} \quad \underline{1} \\
 \hline
 0 \quad 1 \quad 0 \quad 0
 \end{array}$$

Приклади:

$$\begin{array}{r}
 \underline{100} \quad \underline{1000} \quad \underline{100010} \quad \underline{10011010000} \\
 \underline{1} \quad \underline{0101} \quad \underline{011101} \quad \underline{01100011101} \\
 \hline
 011 \quad 0011 \quad 000101 \quad 00110110011
 \end{array}$$

Вісімкова СЧ.

$$\begin{array}{r}
 \underline{6} \quad \underline{10} \quad \underline{12} \\
 \underline{7} \quad \underline{1} \quad \underline{4} \\
 \hline
 7 \quad 7 \quad 5
 \end{array}$$

Шістнадцяткова СЧ.

$$\begin{array}{r}
 \underline{10} \quad \underline{1A} \quad \underline{2} \\
 \underline{1} \quad \underline{8} \quad \underline{C} \\
 \hline
 F \quad 12 \quad 6
 \end{array}$$

Кодовані позиційні системи числення

| 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 мають надлишковість, так як для кодування десятичних цифр використовуються тільки 10 комбінацій із 16.

Двійково-десятковий код

В двійково-десятковому (двійково-кодованому) представленні десятичного числа кожна десяткова цифра зображується тетрадою двійкових символів $a_i = a_4^i a_3^i a_2^i a_1^i$, a_i – десяткова цифра i – го розряду; a_i^i – двійкова цифра i – і тетради.

Одержаний таким чином десятиковий 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 в Д – кодах відрізняє їх від звичайних позиційних систем числення в яких всі комбінації – дозволені. Із всієї множини відомих Д – кодів найбільш поширені в обчислювальній техніці отримали код Д1 прямого заміщення (система 8421) і код Д2 з надлишком 3 (система 8421+3).

Із-за заборонених комбінацій, при додаванні чисел в Д – кодах виникає необхідність в корекції результату і труднощі в формуванні десятичного переносу в наступну тетраду.

Особливості додавання чисел в кожному із Д – кодів різні.

Задані числа

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

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

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

Необхідно отримати

$A + B = C = C_{n-1} C_n \cdots C_1 C_0$ при цьому $c_i = a_i + b_i + \Pi_{i-1} - \Pi_i \cdot p$; $c_{n+1} = \Pi_n$,
де $\Pi_i = \{0, 1\}$, $\Pi_{i-1} = \{0, 1\}$ - десяткові переноси;
 $p = 10$ - основа системи числення.

Так як найбільше десяткове однорозрядне число 9 то з врахуванням переносу в даний розряд, значення результату розрядного сумування лежить в межах від 0 до 19. При цьому одиниця в другому розряді представляє собою десятковий переніс в наступну тетраду, а суму одержуємо в двійковому коді, який відрізняється від потрібного двійково-десяткового представлення, тобто він потребує корекції.

При додаванні чисел в Д кодах можуть виникнути наступні випадки:

1) якщо $a_i + b_i + \Pi_{i-1} < 10$, то виконання дій над розрядами тетради по правилах двійкової арифметики зразу отримаємо правильний результат;

2) якщо $a_i + b_i + \Pi_{i-1} \geq 10$, то виникає десятковий переніс. Тому сума в даній тетраді повинна бути рівна:

$$a_i + b_i + \Pi_{i-1} - \Pi_i - 10,$$

де $\Pi_i = 1$.

При цьому ознакою неправильного результату є в одному випадку виникнення потетрадного переносу $\Pi_i = 16$, в другому поява забороненої комбінації, якщо $15 \geq a_i + b_i + \Pi_{i-1} \geq 10$.

В будь якому із цих випадків необхідно скоректувати результат в даній тетраді введенням поправки +0110, що приведе до виникнення потетрадного переносу і в другому випадку.

Корекція обумовлена тим, що кожний переніс забирає із собою із даної тетради 16 одиниць, а приносить в наступну тільки 10 одиниць.

Приклад. Додати тетради $a_i = 1000$; $b_i = 1001$ при $\Pi_{i-1} = 1$.

$$c_i' = a_i + b_i + \Pi_{i-1} = 10010.$$

Так як $\Pi_i = 1$, необхідна корекція результату $c_i = 0010 + 0110 = 1000$,

$$\Pi_i = \Pi_i' = 1$$

Приклад. Додати тетради $a_i = 1000$; $b_i = 0110$ при $\Pi_{i-1} = 1$.

$$c_i' = a_i + b_i + \Pi_{i-1} = 1111.$$

Так як величина $c_i' = 1111$ належить до заборонених комбінацій, то необхідно ввести поправку виду 0110.

Отже, якщо в i - й тетраді сума цифр з переносом із $(i-1)$ - і тетради менше 10, то додавання відбувається без поправок;

якщо сума цифр з переносом рівна або більша 10, то відбувається корекція результату тетради введенням поправки +0110, а переніс який при цьому виник додаємо до наступної тетради $(i+1) - i$.

При цьому, якщо в декількох тетрадах, починаючи з $(i+1) - i$, розрядна сума дорівнює 1001, то переніс приводить до формування забороненої

комбінації в $(i+1)$ – й тетраді. В результаті цього необхідна корекція, яка приведе до забороненої комбінації в $(i+2)$ – й тетраді і т.д.

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

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

Приклад: $A = 538$, $B = 465$.

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

проводимо корекцію в молодшій тетраді:

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

проводимо корекцію в другій тетраді:

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

проводимо корекцію в старшій тетраді:

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

Контрольні запитання

1. Які системи числення називаються позиційними, непозиційними ?
2. Переведіть задані числа із однієї системи числення в іншу.
3. Виконайте арифметичні операції в різних системах числення.
4. Які фактори необхідно враховувати при виборі СЧ.

2. СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ

Результати досліджень, що проводились різними групами вчених з метою пошуків шляхів підвищення продуктивності обчислювальних засобів, методів організації ефективної системи виявлення та виправлення помилок, а також побудови надійних обчислювальних комплексів, дають можливість стверджувати, що в межах позиційних систем числення не можна очікувати принципових зрушень в даних напрямках без суттєвого збільшення робочих частот і ускладнення апаратної частини. Причина полягає в тому, що позиційні системи числення, в яких представляється і обробляється інформація в сучасних ЕОМ, мають важливий недолік – наявність міжрозрядних зв'язків. Таким чином ефективним є використання непозиційних систем числення, які позбавлені даного недоліку.

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

Нехай задано набір із k взаємопростих натуральних чисел $p_i \in N$, $i = \overline{1, k}$, тоді під СЗК будемо розуміти таку систему, в якій ціле число представляється у вигляді невід'ємних залишків по вибраних модулях p_i .

$$b_i = \text{res } N \pmod{p_i}, \quad i = \overline{1, k}. \quad (1)$$

Даний вираз відповідає системі діофантових рівнянь:

$$N = c_i \cdot p_i + b_i, \quad i = \overline{1, k}, \quad (2)$$

де N – вихідна величина; p_i – набір модулів; b_i – набір залишків по відповідних модулях; c_i – ранг числа N по модулю p_i .

В теорії чисел доведено, що система рівнянь (2) має єдиний розв'язок при взаємопростих модулях. Діапазон чисел, що може бути представлений за допомогою набору модулів $(p_1, p_2, \dots, p_{k-1}, p_k)$ 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

$$\wp = \prod_{i=1}^3 p_i = 3 \cdot 5 \cdot 7 = 105.$$

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

Спосіб 1.

При невеликому діапазоні представлених даних найбільш ефективним є табличний метод кодування та перетворення даних в СЗК .

Таблиця 2.1 – Таблиця кодування даних в СЗК.

| 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 таблиці 2.1: $13_{10} = (1, 3, 6)_{(3,5,7)}$.

Спосіб 2. Нехай у десятковій системі числення задано число $N=103$. Використовуючи рівняння (1) маємо:

$$b_1 = \text{res } 103 \pmod{3} = 1;$$

$$b_2 = \text{res } 103 \pmod{5} = 3;$$

$$b_3 = \text{res } 103 \pmod{7} = 5.$$

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

Спосіб 3. Задано число $N=103$.

Число N представлено в позиційній системі числення з основою $d = 10$. Представлення степенів основи d в СЗК буде мати вигляд:

$$d^0 = 1 = (1, 1, 1)_{(3,5,7)} \bar{b}$$

$$d^1 = 10 = (1, 0, 3)_{(3,5,7)},$$

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

Отримаємо представлення коефіцієнтів 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 формули (5):

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

Представлення числа $N=103_{10}$ отримані за допомогою різних методів аналогічні, що підтверджує достовірність отриманих результатів.

Переведення числа з системи залишкових класів в десяткову систему числення

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

$$N = \sum_{i=1}^k b_i \cdot B_i \pmod{\varphi}. \quad (3)$$

Згідно визначення ортогональних базисів, вони можуть бути обчислені:

$$B_i = m_i \cdot \frac{\varphi}{p_i}, \quad i = \overline{1, k}; \quad (4)$$

де $1 \leq m_i \leq p_i - 1$ – вага ортогонального елемента.

При чому

$$m_i \cdot \frac{\wp}{p_i} = 1 \pmod{p_i}. \quad (5)$$

Рівняння (5) еквівалентне наступному діафантовому рівнянню:

$$m_i \cdot \frac{\wp}{p_i} = 1_i \cdot p_i + 1, \quad 1_i \in N. \quad (6)$$

Для обчислення m_i використовується формула (5). Застосування операції визначення залишку по заданому модулю обумовлює обмежений діапазон можливих значень вагових коефіцієнтів: $m_i \in [1, p - 1]$.

Позначимо $\wp_i = \frac{\wp}{p_i}$. В результаті ділення \wp_i на p_i отримаємо певний залишок δ_i , згідно рівняння (5):

$$m_i \cdot \delta_i = 1 \pmod{\wp}. \quad (7)$$

З огляду на порівняно невеликі значення величини p_i можемо скласти таблицю розв'язків рівняння (7), за допомогою якої згідно величини δ_i знаходиться відповідне значення m_i . Припускаючи, що основи p_i вибираються з множини простих чисел, приведемо таблицю розв'язків рівняння (7), для $p_i < 25$ (таблиця 2.2).

Згідно (5):

$$B_1 + B_2 + \dots + B_k = (1, 0, \dots, 0) + (0, 1, \dots, 0) + \dots + (0, 0, \dots, 1) = (1, 1, \dots, 1). \quad (8)$$

Оскільки сумування проводиться в СЗК:

$$\sum_{i=1}^k B_i = 1 \pmod{\wp}. \quad (9)$$

Таблиця 2.2 – Розв’язки рівняння $m \cdot \delta = 1 \pmod{p}$ для множини простих чисел $p_i < 25$

| 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 |

Рівняння (9) можна використати для перевірки достовірності знаходження базисів системи.

Розглянемо приклад зворотного перетворення для значень отриманих вище

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

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

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

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

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

Використовуючи означення базисних чисел та таблицю 2.2:

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

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

Перевіримо достовірність обчислення базисних чисел згідно формули (9):

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

Згідно формули (3):

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

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

Представлення даних в системі залишкових класів дає змогу здійснювати розпаралелювання обробки інформації без значного ускладнення обчислювальних засобів. Використання СЗК спрощує побудову систем збору інформації, а також дозволяє вирішувати клас задач, що є невизначеними в позиційних системах числення. Особливістю СЗК залишається простота реалізації прямого та зворотного перетворень.

2.2 Математичні операції в СЗК

Розглянемо правила виконання операцій додавання і множення в СЗК при умові, що обидва числа і результат операції знаходяться в діапазоні $[0, \varnothing]$.

Нехай операнди A і B представлені відповідно залишками α_i і β_i по модулю P_i при $i = 1, 2, \dots, n$.

Результат операцій додавання і множення $A + B$ і $A \cdot B$ представлені відповідними залишками γ_i і δ_i по тих же модулях P_i , тобто

$$A = (\alpha_1, \alpha_2, \dots, \alpha_n),$$

$$\begin{aligned} B &= (\beta_1, \beta_2, \dots, \beta_n), \\ A + B &= (\gamma_1, \gamma_2, \dots, \gamma_n), \\ A \cdot B &= (\delta_1, \delta_2, \dots, \delta_n), \end{aligned}$$

і при цьому мають місце співвідношення:

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

Припускається, що γ_i дорівнює $\alpha_i + \beta_i$ по модулю P_i , а δ_i дорівнює $\alpha_i \cdot \beta_i$ також по модулю P_i .

$$\begin{aligned} \gamma_i &\equiv \alpha_i + \beta_i \pmod{P_i}, \\ \delta_i &\equiv \alpha_i \cdot \beta_i \pmod{P_i}. \end{aligned}$$

При цьому в якості цифри результату береться відповідно

$$\gamma_i = \alpha_i + \beta_i - \left[\frac{\alpha_i + \beta_i}{P_i} \right] \cdot P_i \quad (10)$$

$$\delta_i = \alpha_i \cdot \beta_i - \left[\frac{\alpha_i \cdot \beta_i}{P_i} \right] \cdot P_i. \quad (11)$$

Отже, можна записати для додавання

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

для $i = 1, 2, \dots, n$.

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

Для множення

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

Приклад: нехай основою системи є $P_1 = 3, P_2 = 5, P_3 = 7$.

Діапазон представлення чисел за допомогою вибраних модулів визначається, як $\wp = P_1 \cdot P_2 \cdot P_3 = 105$.

Приклад. Додати числа $A=17$ і $B=63$. Переведемо числа A і B в систему залишкових класів по заданих модулях

$$\begin{aligned} A = 17 &= (2, 2, 3)_{(3,5,7)}, \\ B = 63 &= (0, 3, 0)_{(3,5,7)}. \end{aligned}$$

В відповідності з (2.10) отримаємо

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

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

Приклад. Помножити число $A=17$ на число $B=6$.
В СЗК числа A і B будуть представлені як

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

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

В відповідності з (11) отримаємо $A \cdot B = (0, 2, 4)_{(3,5,7)}$.

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

Правила виконання операції віднімання в СЗК в випадку, якщо два числа і результат операції знаходяться в діапазоні $[0, \wp]$.

Нехай операнди A і B представлені відповідними залишками α_i і β_i по модулях P_i при $i = 1, 2, \dots, n$.

Результат операції віднімання $A-B$ представлений відповідними залишками γ_i по тих же модулях P_i .

Тобто

$$A = (\alpha_1, \alpha_2, \dots, \alpha_n),$$

$$B = (\beta_1, \beta_2, \dots, \beta_n),$$

$$A - B = (\gamma_1, \gamma_2, \dots, \gamma_n),$$

і при цьому виконуються умови:

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

Аналогічно з (10) отримаємо вираз для віднімання

$$\gamma_i = \alpha_i - \beta_i - \left[\frac{\alpha_i - \beta_i}{P_i} \right] \cdot P_i,$$

$$\gamma_i = \alpha_i - \beta_i (P_i), \quad i = 1, 2, \dots, n.$$

Операція віднімання в тих випадках, коли її результат додатній, виконується відніманням відповідних цифр розрядів, при цьому завжди в результаті приводиться найменший додатній залишок, так як це впливає із визначення СЗК. Якщо різниця цифр від'ємна, то береться її доповнення до відповідного модуля.

Тобто

$$A - B \pmod{P} = \begin{cases} \alpha_i - \beta_i, & \text{якщо } \alpha_i - \beta_i \geq 0; \\ \alpha_i - \beta_i + P, & \text{якщо } \alpha_i - \beta_i < 0. \end{cases}$$

Приклад. Виконати віднімання двох чисел в СЗК. $C=A-B$.

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

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

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

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

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

Представлення даних в системі залишкових класів дає змогу здійснювати розпаралелювання обробки інформації без значного ускладнення обчислювальних засобів. Використання СЗК спрощує побудову систем збору інформації, а також дозволяє вирішувати клас задач, що є невизначеними в позиційних системах числення. Особливістю СЗК залишається простота реалізації прямого та зворотного перетворень.

Контрольні запитання

1. Назвіть переваги та недоліки СЗК ?
2. Переведіть задані числа з десяткової СЧ в СЗК.
3. Переведіть задані числа з СЗК в десяткову СЧ.
4. Виконайте арифметичні операції в СЗК.

3. ЕЛЕМЕНТИ МАТЕМАТИЧНОЇ ЛОГІКИ

Перемикальні функції.

Булева алгебра одного, двох аргументів.

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 вніс великий вклад в розвиток цієї науки (1815-1864).

Основним предметом булевої алгебри є висловлювання – просте твердження, про яке можна стверджувати: істинне воно (позначається символом 1) або хибне (позначають символом 0).

Прості висловлювання позначають буквами, наприклад X_1, X_2, \dots, 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.

У даний час головна задача алгебри логіки – аналіз, синтез і структурне моделювання будь-яких дискретних скінчених систем.

Змінну із скінченим 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 і кожної її незалежної змінної дорівнює двом є булевою. Таким чином, булева функція – це окремий випадок перемикальної.

Операція – це чітко визначена дія над одним або декількома 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 і т.д.

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

Запис $F(X_1, X_2, \dots, X_n)$ означає, що деяка булева функція F залежить від змінних X_1, X_2, \dots, 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 X”), 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, у яких зліва подані значення операндів, а справа значення булевої функції.

| | |
|-----|---------------|
| 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 і кон'юнкції справедливі 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

$$x \vee x \vee x \vee \dots = x$$

$$x \wedge x \wedge x \wedge \dots = x$$

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

$$\begin{array}{llll}
 x \vee \bar{x} = 1 & x \wedge \bar{x} = 0 & x \wedge 0 = 0 & \bar{1} = 0, \bar{0} = 1 \\
 x \vee 0 = x & x \wedge 1 = x & x \vee 1 = 1 & \bar{\bar{x}} = x
 \end{array}$$

Справедливість наведених законів булевої алгебри перевіряється підстановкою в логічний вираз нуля і одиниці, як показано в табл. 3.1. для різних логічних функцій.

Таблиця 3.1 – Таблиці істинності логічних функцій.

| 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 $F(x_1, x_2, \dots, x_n)$ є скінчена 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 хоч би 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 представити 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 у якому точками виділені одиничні вершини, а також графом, де вершини відображають значення нуля і одиниці, а на орієнтованих дугах змінні вказують на умови переходів (рис.3.1)

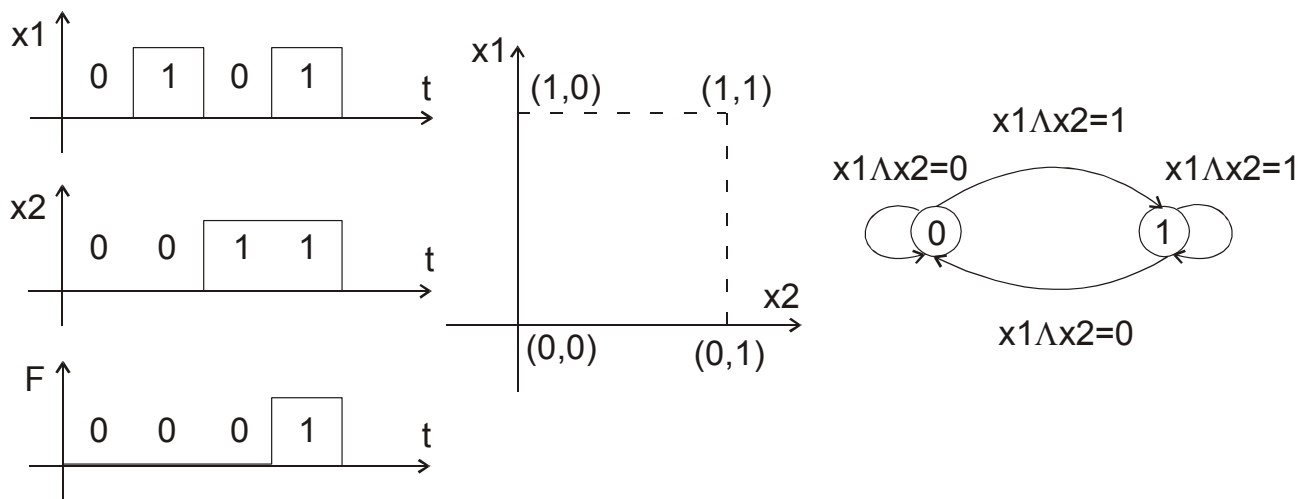


Рис. 3.1 – Способи зображення булевої функції

За допомогою таблиці істинності показують усі можливі функції однієї змінної (усього 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.

| x | | Вираз | Назва функції |
|-----|---|-----------------|---------------|
| 0 | 1 | | |
| 0 | 0 | $F_0 = 0$ | Константа –0 |
| 0 | 1 | $F_1 = x$ | Повторення |
| 1 | 0 | $F_2 = \bar{x}$ | 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.

$$f_{11} = x_1 \leftarrow x_2 = x_1 \vee \bar{x}_2 ;$$

$$f_{13} = x_1 \rightarrow x_2 = \bar{x}_1 \vee x_2 .$$

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

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

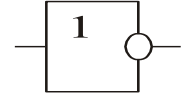
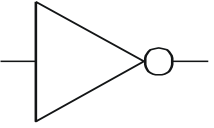
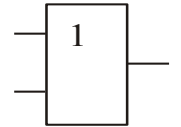
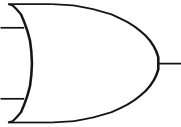
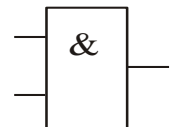

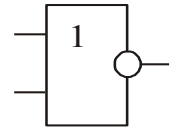
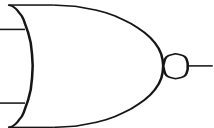
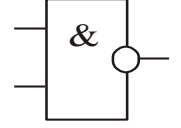
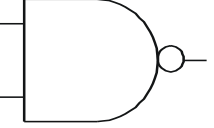
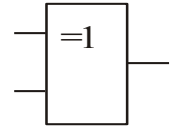
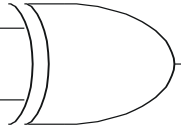
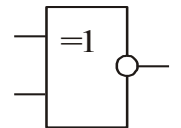
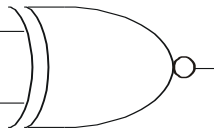
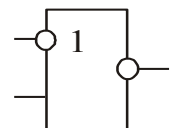
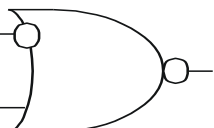
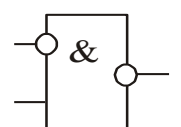
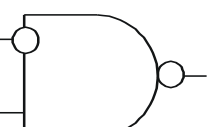
$$f_4 = \overline{x_1} \wedge x_2.$$

Булеві функції двох змінних (табл.3.1)

Таблиця 3.2 – Структурні формули та назви логічних функцій

| Аргументи | | | | Функція | Назва логічної функції |
|-----------|-------|---|---|--|---|
| x_1 | x_2 | | | | |
| 0 | 0 | 0 | 0 | $f_0 = 0$ | Константа 0 |
| 0 | 0 | 0 | 1 | $f_1 = x_1 \wedge x_2$ | Кон'юнкція, (операція I) |
| 0 | 0 | 1 | 0 | $f_2 = x_1 \wedge \overline{x_2}$ | Заборона по 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 по x_1 |
| 0 | 1 | 0 | 1 | $f_5 = x_2$ | Повторення (тавтологія) x_2 |
| 0 | 1 | 1 | 0 | $f_6 = x_1 \oplus x_2 =$ $= (\overline{x_1} \wedge x_2) \vee (x_1 \wedge \overline{x_2})$ | Виключаючи АБО (додавання по модулю 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 | $f_{14} = x_1 x_2 =$ $= \overline{x_1 \wedge x_2} = \overline{x_1} \vee \overline{x_2}$ | Штрих Шеффера (операція І-НЕ) |
| 1 | 1 | 1 | 1 | $f_{15} = 1$ | Константа 1 |

Графічні позначення логічних елементів

| 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 | Виключаючи АБО |  |  |
| Заперечення еквівалентності | Еквівалентність |  |  |
| Імплікація | ЯКЩО, ТО |  |  |
| Заборона | ЗАБОРОНА |  |  |

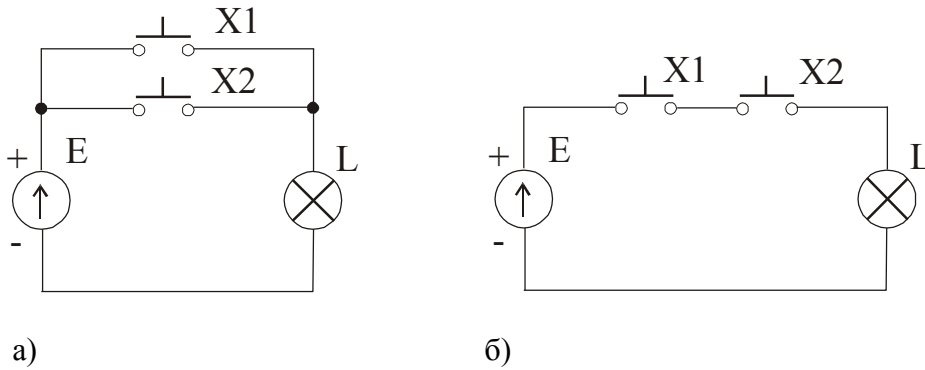


Рис.3.2 – Схеми заміщення логічних елементів: а) логічний елемент **АБО**; б) логічний елемент **І**.

Контрольні запитання

1. Дайте визначення логічних функцій: інверсії, диз'юнкції, кон'юнкції.
2. Зобразіть умовні графічні позначення логічних елементів.
3. Запишіть таблиці істинності заданих логічних елементів.
4. Визначте вихідний стан логічних елементів при заданих вхідних сигналах.
5. Визначте якому логічному елементу належить таблиця істинності.
6. Які логічні елементи можна використати в якості інвертора ?
7. Запишіть закони алгебри логіки.
8. Спростіть логічний вираз:

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

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

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

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

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

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

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

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

4. МІНІМІЗАЦІЯ ПЕРЕМИКАЛЬНИХ ФУНКЦІЙ

Діаграми Вейча та карти Карно для мінімізації перемикальних функцій.
Мінімізація неповністю визначених перемикальних функцій.

Мінімізація булевих функцій

Важливим етапом проектування цифрових пристроїв є мінімізація булевих функцій, тобто знаходження їхніх виразень з мінімальним числом букв.

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

Карти Карно для функцій трьох змінних $F(x_1, x_2, x_3)$.

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

| | | | | |
|--------------------------|--|-------------------------------------|--------------------------|-------------------------------------|
| $x_1 \backslash x_2 x_3$ | 00 | 01 | 11 | 10 |
| 0 | $\overline{x_1} \overline{x_2} \overline{x_3}$ | $\overline{x_1} \overline{x_2} x_3$ | $\overline{x_1} x_2 x_3$ | $\overline{x_1} x_2 \overline{x_3}$ |
| 1 | $x_1 \overline{x_2} \overline{x_3}$ | $x_1 \overline{x_2} x_3$ | $x_1 x_2 x_3$ | $x_1 x_2 \overline{x_3}$ |

Мінтерми в сусідніх клітинках карти Карно в рядку (з врахуванням верхніх і нижніх) або в стовпчику (з врахуванням крайніх) розрізняються значеннями однієї змінної, що дозволяє виконувати операцію склеювання по цій змінній.

Загальні правила мінімізації.

1. Зображають карту Карно для n змінних і роблять розмітку її рядків і стовпчиків. У клітинки таблиці, які відповідають мінтермам (одичним наборам) функції, яка мінімізується, записують одиницю.

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

3. Множина прямокутників, які покривають усі одиниці, називають покриттям. Чим менше прямокутників і чим більше клітинок у прямокутниках, тим краще покриття. З декількох варіантів вибирають той, у 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 числом 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 називається її рангом. При склеюванні двох сусідніх клітинок одержують ранг кон'юнкції $n-1$, чотирьох клітинок $n-2$, восьми клітинок $n-3$ і т. д.

Розмітка карт Карно для функцій чотирьох змінних.

| | | | | | |
|-----------|-----------|------|------|------|------|
| | $x_3 x_4$ | 00 | 01 | 11 | 10 |
| $x_1 x_2$ | 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, які аналогічні картам Карно і відрізняються від них способом розмітки замість символів 0 і 1 використовують булеві аргументи – x_1, \bar{x}_1, x_2 та інші.

Діаграми Вейча для 2, 3 та 4 - х змінних мають вигляд.

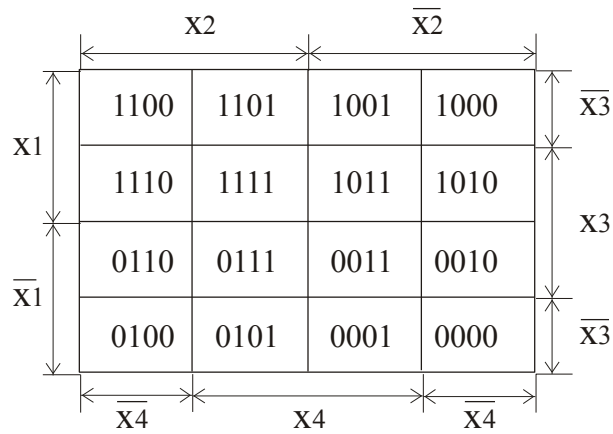
Діаграма Вейча для 2- х змінних.

| | | |
|-------------|-------|-------------|
| | x_2 | \bar{x}_2 |
| x_1 | 11 | 10 |
| \bar{x}_1 | 01 | 00 |

Діаграма Вейча для 3- х змінних.

| | | | | |
|-------------|-------------|-----|-------------|-----|
| | x_2 | | \bar{x}_2 | |
| x_1 | 110 | 111 | 100 | 100 |
| \bar{x}_1 | 010 | 011 | 000 | 000 |
| | \bar{x}_3 | | x_3 | |

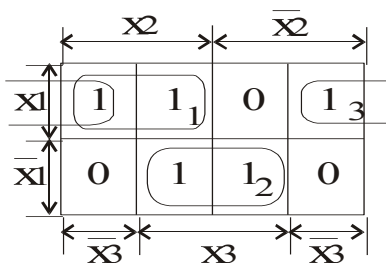
Діаграма Вейча для 4- х змінних.



Приклад. Спростити логічний вираз з використанням діаграм Вейча.

$$y = x_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \vee x_1 \wedge x_2 \wedge \bar{x}_3 \vee \bar{x}_1 \wedge \bar{x}_2 \wedge x_3 \vee \bar{x}_1 \wedge x_2 \wedge x_3 \vee x_1 \wedge x_2 \wedge x_3$$

Діаграма Вейча згідно заданого виразу буде мати вигляд:



Спрощений вираз має вигляд:

$$y = x_1 \wedge x_2 \vee \bar{x}_1 \wedge x_3 \vee x_1 \wedge \bar{x}_3.$$

Контрольні запитання

1. Для чого призначені методи мінімізації.
2. Назвіть методи мінімізації логічних виразів.
3. Назвіть графічні методи мінімізації.
4. Мінімізуйте задані логічні вирази.

5. СИНТЕЗ КОМБІНАЦІЙНИХ СХЕМ

Аналітичне представлення булевих функцій.

Етапи синтезу логічних схем на логічних елементах.

Розроблені універсальні (канонічні) форми представлення булевих функцій, які дають можливість одержати аналітичну форму довільної функції безпосередньо з таблиці істинності. Ця форма надалі може бути мінімізувати або спрощена.

Оскільки між множиною аналітичних представлень і множиною схем, які реалізують цю функцію, є взаємно однозначна відповідність, то пошук канонічної форми запису є початковим етапом синтезу логічних схем.

Найбільше поширення одержали досконала диз'юнктивна нормальна форма (ДДНФ) і досконала кон'юнктивна нормальна форма (ДКНФ). Для одержання цих форм вводяться 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 входять до нього у прямому виді, якщо значення даної змінної в наборі $x_i = 1$, і із запереченням, якщо $x_i = 0$. При n змінних є 2^n мінтермів m_0, m_1, \dots, m_R , де $R = 2^n - 1$.

Всі мінтерми двох змінних наведені в таблиці 5.1.

Таблиця 5.1. Мінтерми двох змінних

| 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 i -му наборові, позначені через f_0, f_1, f_2, f_3 .

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

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

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

Макстерм одержують як диз'юнкцію усіх змінних, що входять до у прямому вигляді, коли значення $x_i = 0$, або в інвертованому вигляді, якщо значення $x_i = 1$.

Число макстермів дорівнює 2^n , для функції двох змінних вони наведені в таблиці.

Представлення функції у ДКНФ записується у вигляді:

$$F_9 = (f_0 \vee M_0) \wedge (f_1 \vee M_1) \wedge (f_2 \vee M_2) \wedge (f_3 \vee M_3) = \\ = (1_0 \vee M_0) \wedge (0 \vee M_1) \wedge (0_2 \vee M_2) \wedge (1_3 \vee M_3) = M_1 \wedge M_2 = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2)$$

Приклад. На прикладі табл.5.2 пояснимо аналітичний запис функції трьох змінних у ДДНФ і ДКНФ..

Таблиця 5.2. Таблиця істинності

| 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

$$P = \bar{x}_1 \wedge \bar{x}_2 \wedge x_3 \vee \bar{x}_1 \wedge x_2 \wedge \bar{x}_3 \vee x_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \vee x_1 \wedge x_2 \wedge x_3.$$

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

$$P = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3).$$

За даним способом виконують запис у ДДНФ і ДКНФ функцій з довільним числом змінних.

Система функцій, суперпозицією яких може бути представлена будь-яка булева функція, називається функціонально повною і вона утворює базис у логічному просторі.

Систему функцій називають мінімально повним базисом, якщо видалення 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 усі інші булеві функції.

Функціональну схему логічного пристрою одержують в результаті абстрактного синтезу, який складається з наступних 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 би дві підряд змінні приймають значення "1".

1. Складаємо таблицю істинності.

| 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 в виді ДДНФ представляє собою диз'юнкцію 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:

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

4. Функціональну схему реалізуємо в базисі І-НЕ, для цього мінімізоване рівняння перетворимо по правилу де Моргана:

у базисі І-НЕ

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

в базисі АБО-НЕ

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

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

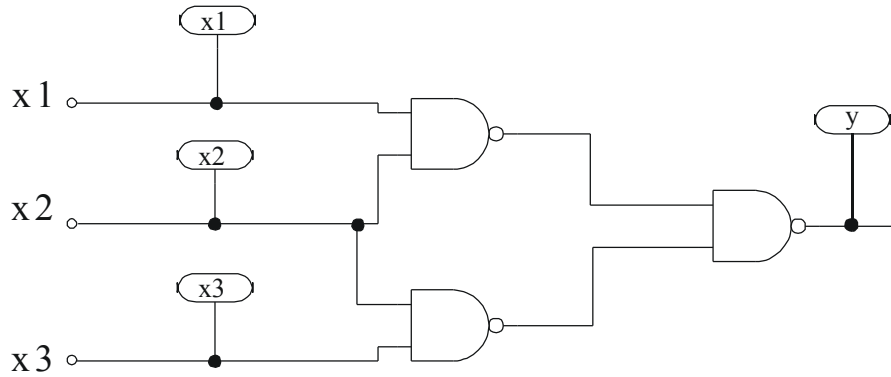


Рис. 5.1 – Функціональна схема логічного пристрою у базисі І-НЕ.

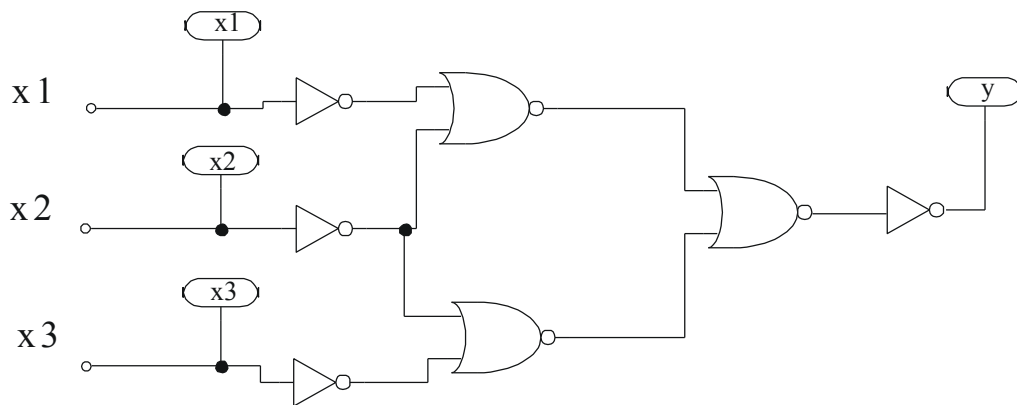


Рис. 5.2 – Функціональна схема логічного пристрою в базисі АБО-НЕ.

Контрольні запитання

1. Дайте визначення комбінаційного цифрового пристрою.
2. Назвіть етапи синтезу цифрових комбінаційних пристроїв.
3. Як записується досконала нормальна диз'юнктивна форма?
4. Як записується досконала нормальна кон'юнктивна форма?
5. Переведіть задане рівняння в базис І-НЕ, АБО-НЕ.
6. Синтезуйте функціональну схему пристрою за заданим рівнянням.

6. АНАЛІЗ КОМБІНАЦІЙНИХ СХЕМ

Аналіз КС методом синхронного моделювання

Аналіз КС методом асинхронного моделювання

Задача аналізу полягає в визначенні статичних і динамічних властивостей комбінаційної схеми (КС). В статичці визначаються булеві функції (БФ), які реалізуються 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 впливають безпосередньо із виразів для булевих функцій, які реалізує схема.

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

Всі існуючі методи аналізу діляться на прямі і непрямі. В результаті аналізу КС прямим методом одержуємо множину наборів вхідних змінних, які забезпечують задане значення на виході, що дозволяє записати в математичному виді БФ, яку реалізує схема. До прямих методів відноситься метод π – алгоритму.

Використання непрямих методів дає можливість визначити реакцію схеми на заданий набір вхідних змінних в статичці або проаналізувати перехідний процес зміни одного вхідного набору на інший. Прикладами непрямих методів аналізу є методи синхронного і асинхронного моделювання. Всі згадані методи аналізу є машиноорієнтованими, що дозволяє виконати аналіз схеми на ЕОМ.

Аналіз КС методом синхронного моделювання

При даному методі припускаємо, що всі логічні елементи (ЛЕ) переключаються одночасно, без затримки. В результаті використання методу визначаємо значення сигналу на виході схеми.

Розглянемо метод синхронного моделювання на прикладі схеми (рис.6.1).

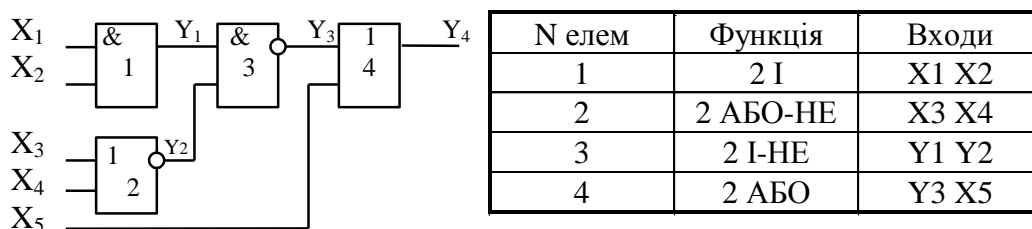


Рис. 6.1. Приклад схеми для методу синхронного моделювання

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

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

Проаналізуємо схему при подачі на вхід набору $X_1 = 0, X_2 = 0, X_3 = 0, X_4 = 1, X_5 = 1$.

Для цього необхідно розв'язати рівняння в порядку зростання.

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

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

Аналіз КС методом асинхронного моделювання.

Реальний ЛЕ переключається за деякий кінцевий час, який залежить від технології виготовлення, умов експлуатації, ємності навантаження і т.д. Проходження сигналу послідовно через декілька ЛЕ приведе до сумування часу затримки і виникнення зсуву в часі вихідного сигналу по відношенню до вхідного.

Наявність затримки і часового зсуву сигналів, який вона породжує може приводити до появи на виході окремих ЛЕ і всієї схеми в цілому короткочасних сигналів, які не передбачені БФ, що реалізує схема.

Розглянемо схему (рис.6.2).

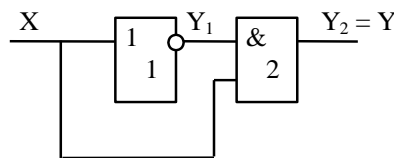


Рис. 6.2. Схема формування короткочасних сигналів

Дана схема реалізує функцію $Y = X \wedge \bar{X} = 0$, тобто константу 0 незалежно від вхідного сигналу X . Але в перехідному процесі в результаті 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).

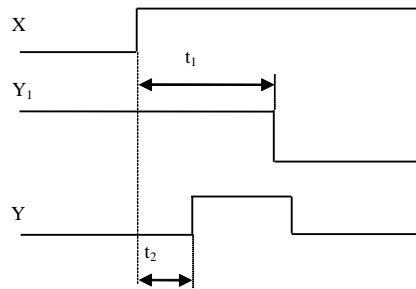


Рис. 6.3 – Статичний ризик збою: а) – часові діаграми; t_1 - час затримки інвертора; t_2 - час затримки елемента 2 І.

Даний випадок можливий при затримці спрацювання другого елемента більшій ніж у першого. Таке явище називається ризиком збою. Розрізняють статичні і динамічні ризики збою.

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

При **динамічному** ризику збою до і після перехідного процесу стану вихідного сигналу протилежний, але в перехідному процесі вихідний сигнал кілька разів міняє своє значення. Динамічний ризик збою можливий в схемі (рис.6.4 а) при зміні набору ($X_1=0, X_2=1, X_3=1$) на набір ($X_1=1, X_2=0, X_3=0$), що показано на діаграмі (рис.6.4 б).

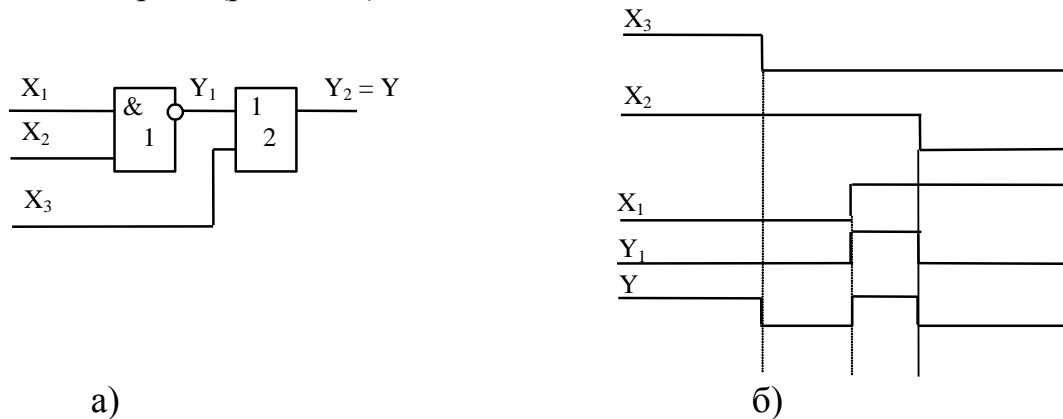


Рис. 6.4: а) – Схема; б)- Часові діаграми.

В даному прикладі динамічний ризик збою на виході КС супроводжується статичним на виході елемента 1. Як видно з часових діаграм ризик збою має місце за наявності певного часового зсуву між сигналами, що поступають на вхід ЛЕ. Небажані сигнали на виході можуть і не бути при іншому співвідношенні часових сигналів, проте принципова можливість їх появи є

чинником, що знижує надійність роботи схеми. Тому дуже важливо вміти знаходити і усувати такі явища.

Для аналізу процесу перемикання КС при зміні вхідних наборів і виявлення ризиків збою використовується метод *асинхронного моделювання*. При цьому методі вважається, що кожний елемент перемикається з однаковою затримкою. Аналіз включає 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 в порядку зменшення рівня.

3. Для початкового вхідного набору $A(X_1, X_2, \dots, X_n)$ визначаються значення сигналів на виходах всіх ЛЕ схеми. Нехай даний набір A замінюється набором $B(X_1, X_2, \dots, X_n)$.

4. Відмічаються ті рівняння, в правій частині яких хоча б одна із змінних змінила своє значення.

5. Розв'язуються відмічені рівняння в порядку їх запису в схемі. Після розв'язку рівняння вважається невідміченим.

6. Якщо після розв'язку всіх рівнянь системи змінні, які входять в ліві частини рівнянь, змінили свої значення, то знову відмічаються ті рівняння, в правій частині яких входять ці змінні. Потім здійснюється перехід до п.5. В іншому випадку моделювання даного вхідного набору вважається закінченим. Виконання п.5 називається *тактом моделювання*.

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

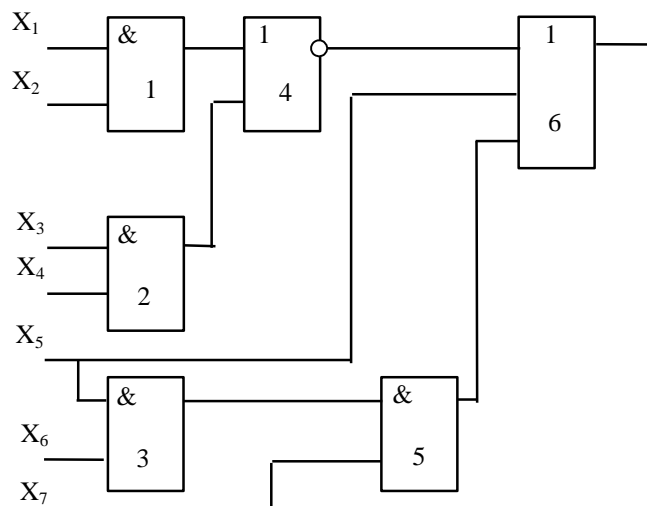


Рис.6.5 – Комбінаційна схема для методу асинхронного моделювання.

Рівняння, які описують ЛЕ.

| 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 | | | |
|--------|-------------------|----------|----------|----------|
| | <i>0</i> | <i>1</i> | <i>2</i> | <i>3</i> |
| 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 |

Як впливає з результатів моделювання, при зміні набору А набором В на виході елемента 4 має місце 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 закінчення перехідного процесу в КС (тобто коли на виході КС вже встановилося необхідне значення вихідного сигналу), що виключає вплив можливих збоїв на сигнал, що виробляється схемою.

Контрольні запитання

1. Призначення методів аналізу.
2. Назвіть етапи аналізу комбінаційних схем.
3. Причини статичних та динамічних ризиків в КС.
4. Провести аналіз заданої комбінаційної схеми.

7. СИНТЕЗ ДЕШИФРАТОРІВ ТА ШИФРАТОРІВ

Дешифратор – 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 називається неповним і у нього $N_{\text{вих}} < 2^n$.

Якщо вхідні змінні представити 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 в двійковому коді відповідає число 5 в десятковому коді, тобто при вказаній комбінації вхідних змінних $F_5 = 1$.

Дешифратори широко використовуються в якості перетворювачів 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 використовуються букви DC і CD (від слів decoder і coder відповідно).

Робота дешифратора описується системою логічних рівнянь складених на основі таблиці істинності:

| 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 складається функціональна схема дешифратора.

Шифратор перетворює код “1 із N” в двійковий код, тобто виконує операцію, обернену до дешифратора.

При подачі 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:

| 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:

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

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

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 адреси в вісімковій системі числення: 126, 127.

Переведемо адреси в двійкову систему числення.

001 010 110, 001 010 111

Запишемо логічні рівняння:

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

$$y_2 = x_0 \wedge x_1 \wedge x_2 \wedge \overline{x_3} \wedge x_4 \wedge \overline{x_5} \wedge x_6 \wedge \overline{x_7} \wedge \overline{x_8}.$$

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).

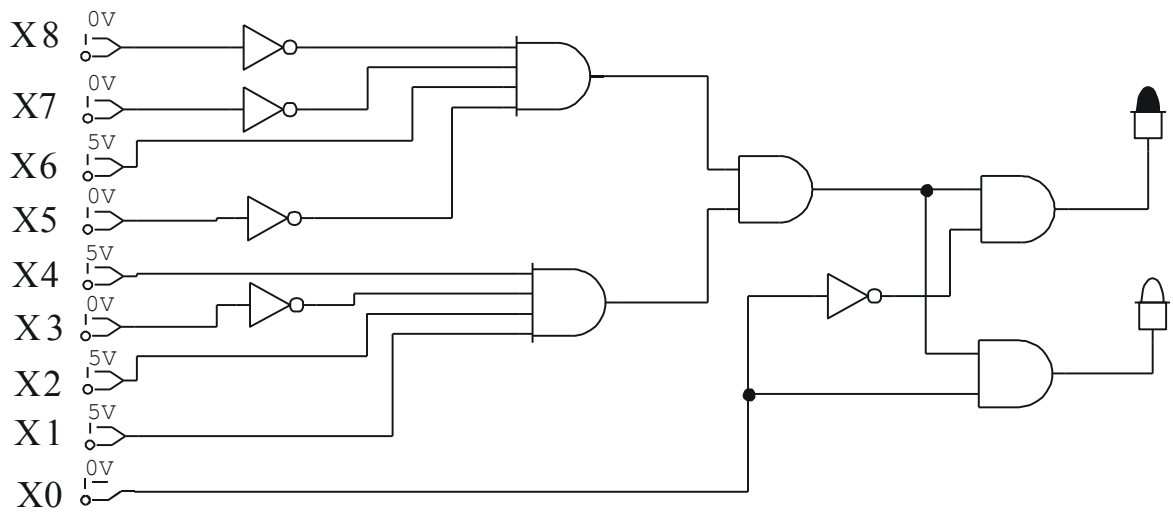


Рис. 7.1 – Функціональна схема шифратора.

Контрольні запитання

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

8. СИНТЕЗ МУЛЬТИПЛЕКСОРІВ ТА ДЕМУЛЬТИПЛЕКСОРІВ

Мультиплексор призначений для комутації в певному порядку на один вихід одного з декількох вхідних сигналів, в залежності від стану адресних входів. За допомогою мультиплексора здійснюється часове розділення інформації, що надходить по різних каналах. Мультиплексор можна порівняти з безконтактним багатопозиційним перемикачем.

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

- інформаційні;
- керуючі (адресні і дозволяючі).

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

Набір сигналів на адресних входах визначає конкретний інформаційний вхід, який буде з'єднаний 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, дозволяє синхронізувати його роботу з роботою інших вузлів. Дозволяючий вхід використовується також для нарощування розрядності мультиплексора. Адресні входи мультиплексора позначимо літерами $A_i \in \{0, 1\}$, а інформаційні входи $D_i \in \{0, 1\}$ (рис. 8.1).

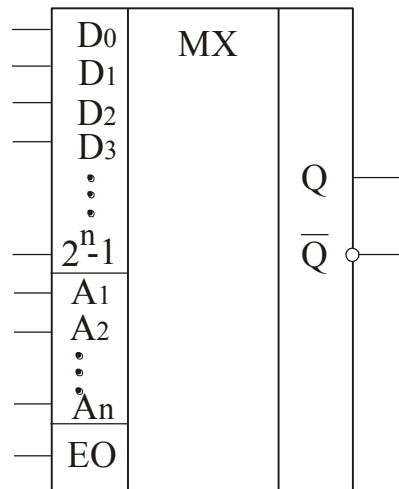


Рис. 8.1 – Графічне позначення мультиплексора.

Приклад: синтезувати мультиплексор 4:1.

Складаємо таблицю істинності мультиплектора 4:1

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

Робота мультиплектора 4:1 описується логічним рівнянням, складеним на основі таблиці істинності.

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

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

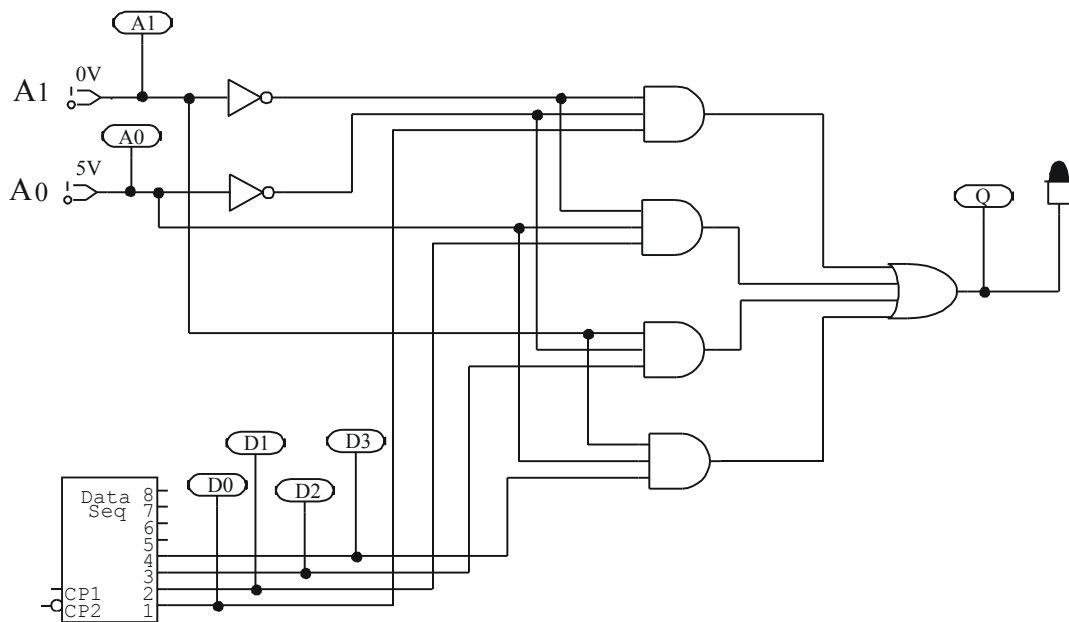


Рис. 8.2 – Функціональна схема мультиплектора 4:1.

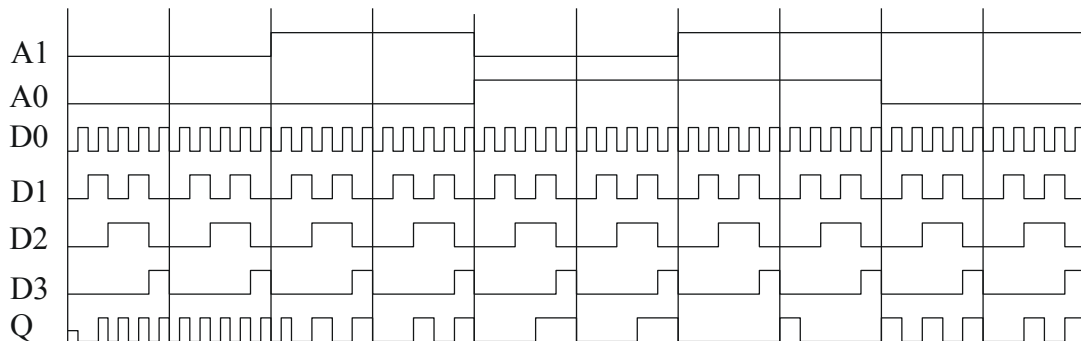


Рис. 8.3– Часова діаграма роботи мультиплектора

Рівняння мультиплексора з дозволяючим входом має вигляд:

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

Демультимплексор – комбінаційна схема, яка комутує сигнал з одного інформаційного входу на один з декількох виходів в залежності від стану адресних входів. При n адресних входах демультимплексор може мати 2^n виходів.

Приклад: синтезуємо демультимплексор 1:4 з дозволяючим входом EO (рис. 8.4).

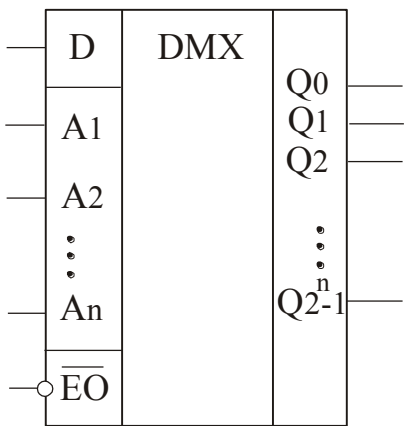


Рис. 8.4. Графічне позначення демультимплексора.

Таблиця істинності демультимплексора 1:4 має вигляд:

| 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 демультимплексора 1:4 з дозволяючим входом \overline{EO} описується логічними рівняннями складеними на основі таблиці істинності.

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

$$Q_2 = \overline{A_1} \wedge A_0 \wedge D \wedge \overline{E_0},$$

$$Q_3 = A_1 \wedge \overline{A_0} \wedge D \wedge \overline{E_0},$$

$$Q_4 = A_1 \wedge A_0 \wedge D \wedge \overline{E_0}.$$

Одержані рівняння переводять в заданий базис і складають функціональну схему.

5. Контрольні запитання

1. Дайте визначення мультиплексора.
2. Дайте визначення демультимплексора.
3. Які назви і призначення мають входи мультиплексора ?
4. Як мультиплексор і демультимплексор позначаються 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 виконуються в арифметико-логічних пристроях (АЛП) чи процесорних елементах, ядром яких є суматори.

Суматори класифікують по різних ознаках.

У залежності від системи числення розрізняють:

- двійкові;
- двійково-десяткові (у загальному випадку двійково-кодовані);
- десяткові.

По кількості одночасно оброблюваних розрядів чисел, що додаються:

- однорозрядні;
- багаторозрядні.

По числу входів і виходів однорозрядних двійкових суматорів:

- чвертьсуматори (елементи “сума по модулю 2”; елементи “виключаюче АБО”), що характеризуються наявністю двох входів, на які подаються два однорозрядних 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 переніс у наступний (старший) розряд;
- повні однорозрядні двійкові суматори, що характеризуються наявністю трьох входів, на які подаються однойменні розряди двох чисел, що складаються, і переніс з попереднього (молодшого) розряду, і двома виходами: на одному реалізується арифметична сума в даному розряді, а на іншому переніс у наступний (старший) розряд.

По способу представлення й обробки чисел, що додаються, багаторозрядні суматори поділяються на:

- послідовні, у яких обробка чисел ведеться по черзі, розряд за розрядом на тій самій елементній базі;
- паралельні, у яких доданки додаються одночасно по всіх розрядах, і для кожного розряду є своя елементна база.

Паралельний суматор у найпростішому випадку являє собою **n** однорозрядних суматорів, послідовно (від молодших розрядів до старших) з'єднаних ланцюгами переносу. Однак така схема суматора характеризується порівняно невисокою швидкістю тому, що формування сигналів суми і переносу в кожному *i*-му розряді виробляється лише після того, як надійде сигнал переносу з (*i*-1) – го розряду.

Чвертьсуматор

Найпростішим двійковим сумуючим елементом є чвертьсуматор. Походження назви цього елемента впливає з того, що він має в два рази менше виходів і в два рази менше рядків у таблиці істинності в порівнянні з повним двійковим однорозрядним суматором. Найбільш вживані назви: елемент “сума по модулю 2” і елемент “виключаюче АБО”. Схема (рис. 1) має два входи a і b для двох доданків, що додаються, й один вихід S для суми. Роботу її відображає таблиця істинності (табл. 9.1), а відповідне рівняння має вигляд:

$$S = \bar{a} \wedge b \vee a \wedge \bar{b} = a \oplus b \quad (1)$$

Таблиця 9.1. –Таблиця істинності чвертьсуматора

| a | b | S |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

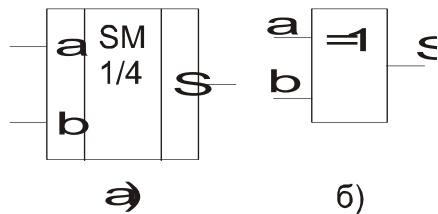


Рис. 9. 1. Графічне позначення чверть суматора

$$S = \bar{a}b + a\bar{b} = \bar{a}a + \bar{a}\bar{b} + \bar{b}b + a\bar{b} =$$

$$= a(\bar{a} + \bar{b}) + b(\bar{a} + \bar{b}) = a\bar{a}\bar{b} + b\bar{a}\bar{b} = \overline{\overline{a\bar{a}\bar{b}} \cdot \overline{b\bar{a}\bar{b}}} \quad (2)$$

$$S = \bar{a}b + a\bar{b} = \bar{a}a + \bar{a}\bar{b} + \bar{b}b + a\bar{b} =$$

$$= \overline{\overline{a(a+b)}} + \overline{\overline{b(a+b)}} = \overline{a + a + b + b + a + b} \quad (3)$$

$$S = \bar{a}b + a\bar{b} = \bar{a}a + \bar{a}\bar{b} + \bar{b}b + a\bar{b} =$$

$$= \bar{a}(a+b) + \bar{b}(a+b) = (a+b)(\bar{a} + \bar{b}) = (a+b)\bar{a}\bar{b} \quad (4)$$

Схеми, отримані за рівняннями (2-4), приведені на рис. 9.2.

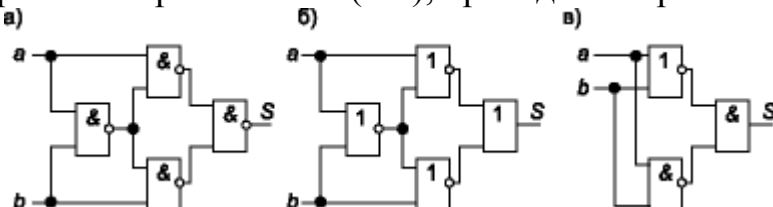


Рис. 9.2. Схеми чверть суматора

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

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

$$P = a \wedge b$$

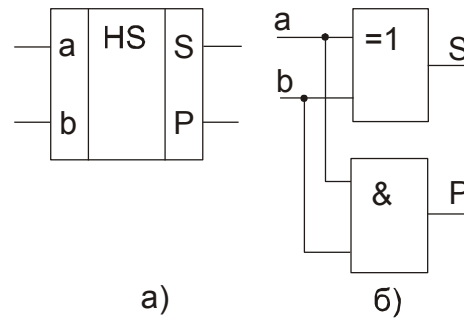


Рис. 9.3. Графічне позначення напівсуматора

Таблиця 9.2– Таблиця істинності напівсуматора

| a | b | P | S |
|----------|----------|----------|----------|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

З рівнянь випливає, що для реалізації напівсуматора потрібно один елемент “виключаюче АБО” і один двовходовий 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 переносу з попереднього (молодшого) розряду і два виходи: S – сума, P – переніс у наступний (старший) розряд. Позначають повний двійковий суматор буквами SM. Його роботу відображає таблиця істинності (табл. 9.3).

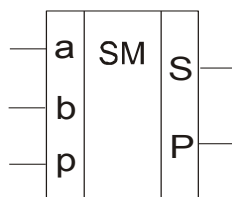


Рис. 9.4. Графічне позначення повного однорозрядного двійкового суматора.

Таблиця 9.3 – Таблиця істинності однорозрядного двійкового суматора

| № | 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 |

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

Контрольні запитання

1. Дайте визначення суматора.
2. Дайте визначення чвертьсуматора, напівсуматора, повного суматора.
3. Синтезуйте чвертьсуматор в базисі І-НЕ, АБО-НЕ.
4. Синтезуйте напівсуматор в базисі І-НЕ.
5. Запишіть 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 визначається сигналами на його входах. Під впливом вхідного сигналу тригер стрибкоподібно переходить з одного стійкого стану в інший. Тригери мають два виходи: прямий Q і інвертований \bar{Q} . Стан тригера визначається по прямому виходу.

За логічним функціонуванням розрізняють типи тригерів: RS, D, T, JK.

За способом запису інформації розрізняють асинхронні і синхронні тригери. Синхронні тригери мають спеціальний вхід синхронізації (тактовий) – С (від слова Clock).

За способом сприйняття тактових сигналів тригери діляться на статичні (керовані рівнем 0 або 1) і динамічні (керовані фронтом наростання або фронтом спаду).

RS-тригери.

Асинхронний RS – тригер з прямими входами.

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

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

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

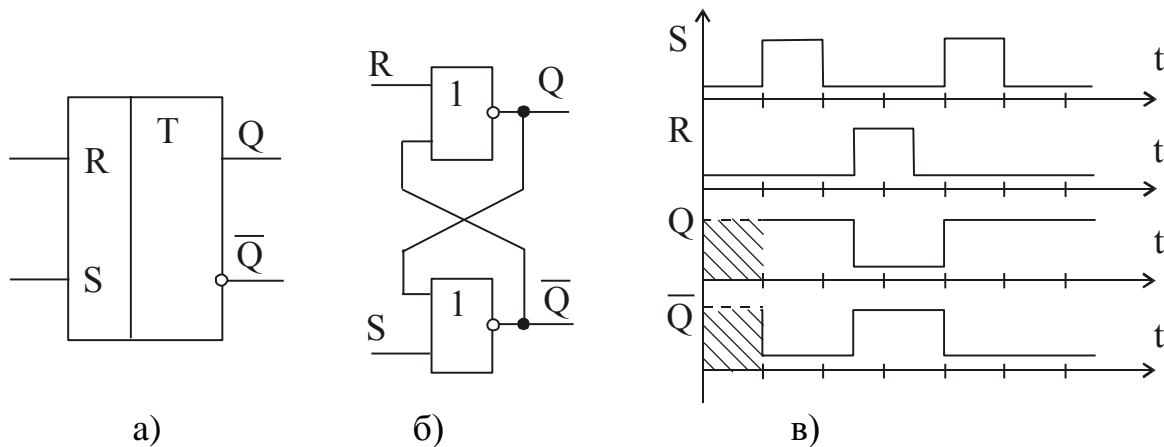


Рис. 10.1 – Асинхронний RS – тригер: а – графічне позначення; б – реалізація на елементах АБО-НЕ; в – часові діаграми роботи.

Таблиця переходів RS – тригера

| <i>Вхід – S</i> | <i>Вхід – R</i> | <i>Вихід – Q_{i+1}</i> | <i>Режим роботи</i> |
|-----------------|-----------------|-------------------------------------|-------------------------|
| 0 | 0 | Q_i | <i>Зберігання</i> |
| 0 | 1 | 0 | <i>Скидання в 0</i> |
| 1 | 0 | 1 | <i>Встановлення в 1</i> |
| 1 | 1 | – | <i>Заборонений</i> |

Асинхронний RS-тригер с інверсними входами.

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

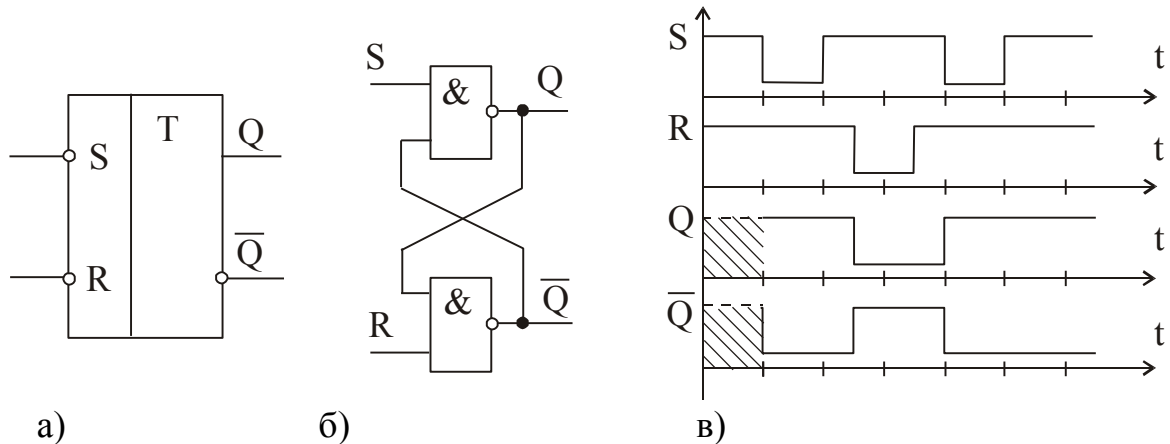


Рис. 10.2 – Асинхронний RS – тригер: а – графічне позначення; б – реалізація на елементах І-НЕ; в – часові діаграми роботи.

Робота тригера визначається таблицею переходів.

Таблиця переходів RS-тригер с інверсними входами

| <i>Вхід – \bar{S}</i> | <i>Вхід – \bar{R}</i> | <i>Вихід – Q_{i+1}</i> | <i>Режим</i> |
|------------------------------------|------------------------------------|-------------------------------------|-------------------------|
| 0 | 0 | - | <i>Заборонений</i> |
| 0 | 1 | 1 | <i>Встановлення в 1</i> |
| 1 | 0 | 0 | <i>Скидання в 0</i> |
| 1 | 1 | Q_i | <i>Зберігання</i> |

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 свій стан при логічній 1 на вході C (якщо C вхід прямий) або при логічному 0 на вході C (якщо C вхід інвертований).

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

Таблиця переходів синхронного RS-тригера

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

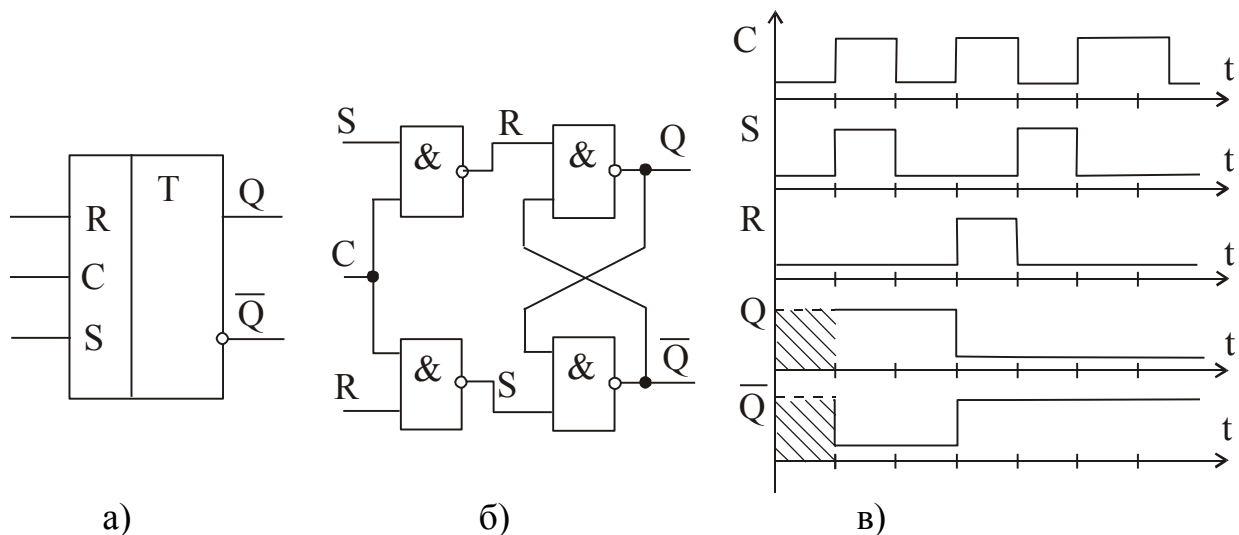


Рис. 10.3 – Синхронний RS – тригер: а – графічне позначення; б – реалізація на елементах І-НЕ; в – часові діаграми роботи.

T-тригери.

Асинхронний T-тригер.

T – тригер має один інформаційний T – вхід (від англ. toggle) (рис. 10.4). Зміна стану тригера відбувається кожний раз при зміні вхідного сигналу в визначеному напрямку. Стан T – тригера визначається його станом в попередньому такті.

Таблиця переходів асинхронного T-тригера.

| T | Q_{i+1} | Режим роботи |
|----------|-------------|---------------------|
| 0 | Q_i | <i>Зберігання</i> |
| 1 | \bar{Q}_i | <i>Інверсія</i> |

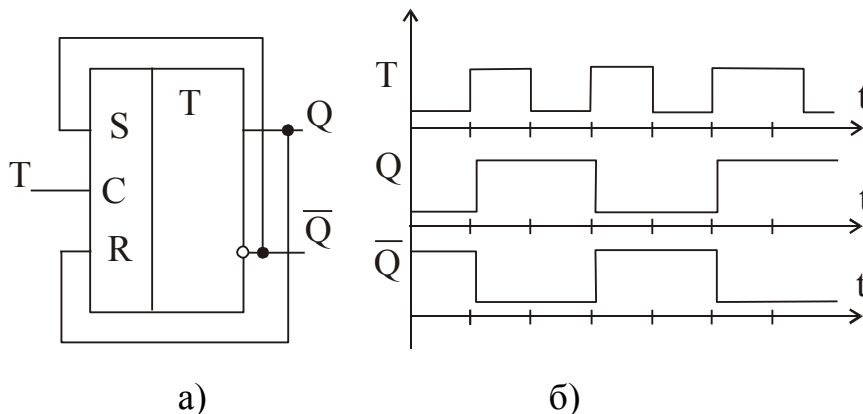


Рис. 10.4 – Асинхронний T – тригер: а – графічне позначення; б – часові діаграми роботи.

Синхронний Т - тригер.

Синхронний Т – тригер спрацьовує по фронту наростання або спаду 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.

| C | T | Q_{i+1} | Режим роботи |
|----------|----------|------------------|---------------------|
| 0 | * | Q_i | <i>Зберігання</i> |
| 1 | 0 | Q_i | <i>Зберігання</i> |
| 1 | 1 | \overline{Q}_i | <i>Інверсія</i> |

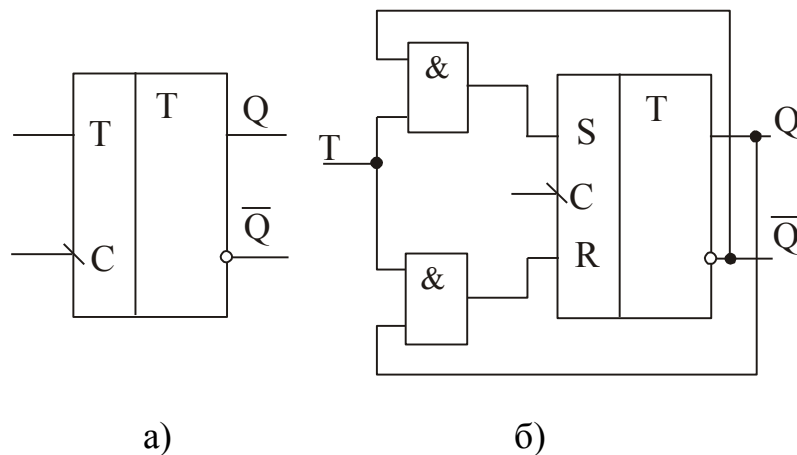


Рис. 10.5 – Синхронний Т – тригер: а – графічне позначення; б – реалізація Т – тригера на базі RS – тригера;

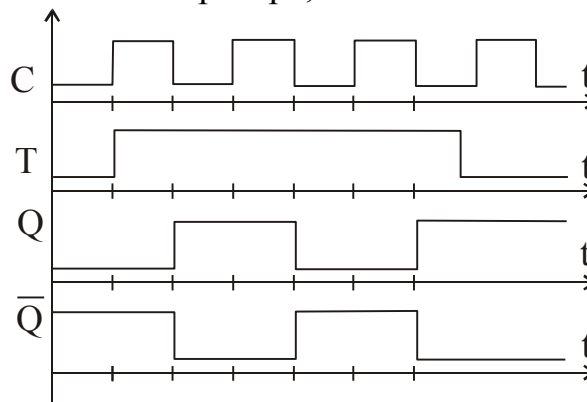


Рис. 10.6 – Часові діаграми роботи синхронного Т – тригера.

D-тригери.

D – тригери (тригери затримки) на відміну від розглянутих раніше мають один інформаційний вхід D для встановлення в стан 1 або 0. Позначення **D** – це перша буква англ. слова delay – затримка. Функціональна особливість **D** – тригерів в тому, що сигнал на виході Q в такті **n+1** повторює вхідний сигнал **Dⁿ** в 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.

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

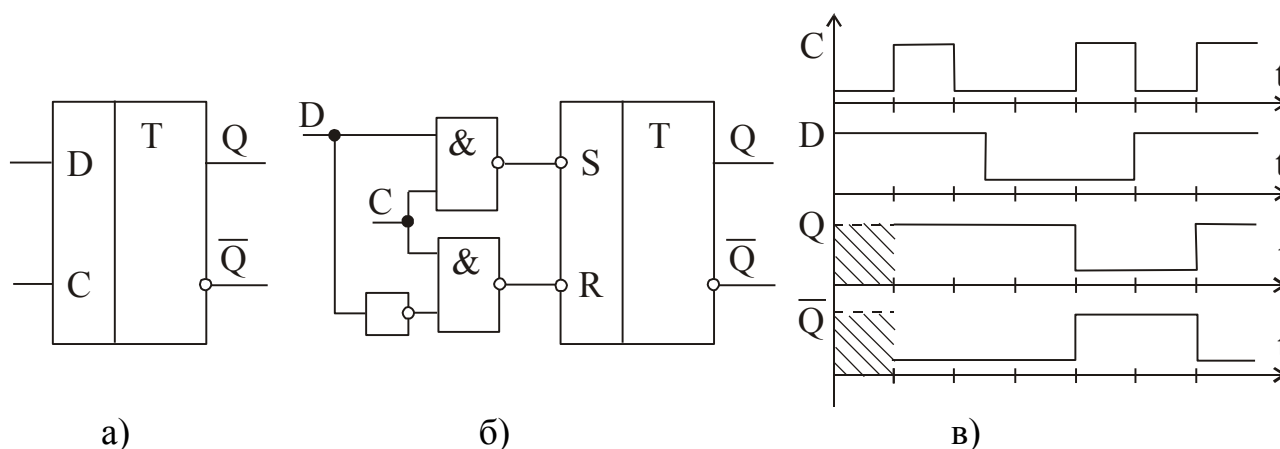


Рис. 10.7 – Синхронний D – тригер: а – графічне позначення; б – реалізація D – тригера на базі RS – тригера; в – часові діаграми роботи.

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 буде працювати як синхронний D-тригер) (рис. 10.8).

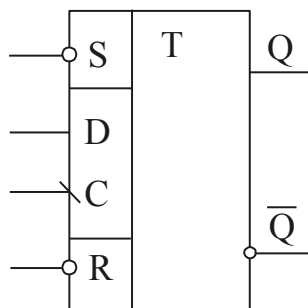


Рис. 10.8 – Синхронний D – тригер з асинхронними інвертованими входами RS.

Таблиця переходів синхронного D - тригера

| C | D | S | R | Q_{i+1} | <i>Режим роботи</i> |
|----------|----------|----------|----------|-----------|-------------------------|
| 0 | * | 0 | 0 | - | <i>Заборонений</i> |
| 0 | * | 0 | 1 | 1 | <i>Встановлення в 1</i> |
| 0 | * | 1 | 0 | 0 | <i>Скидання в 0</i> |
| 0 | * | 1 | 1 | Q_i | <i>Зберігання</i> |
| 1 | 0 | 1 | 1 | 0 | <i>Скидання в 0</i> |
| 1 | 1 | 1 | 1 | 1 | <i>Встановлення в 1</i> |

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 відповідає 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:

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

При $J^n = K^n = 0$ на виходах елементів 1 і 2 буде $q_1 = q_2 = 1$ (незалежно від значень сигналів Q і \bar{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 визначатись логічним елементом 1 або 2, на всіх входах якого логічна 1. Вхідна комбінація $J^n = K^n = 1$ при будь якому стані тригера викликає зміну (інвертування) вихідного сигналу.

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

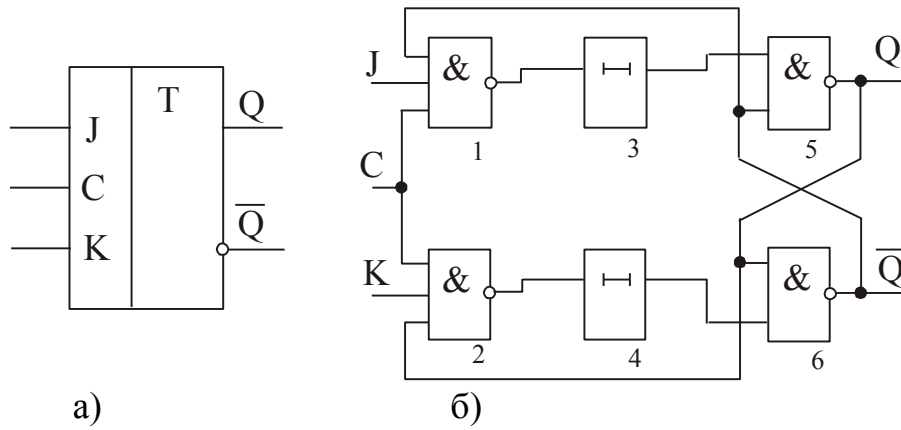


Рис. 10.9 – Синхронний JK – тригер: а – графічне позначення; б – логічна структура JK – тригера.

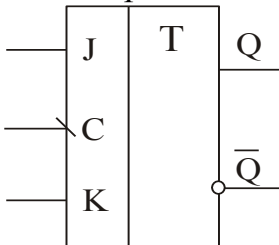
Таблиця переходів синхронного JK – тригера

| 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 | \bar{Q}_i | Інверсія |

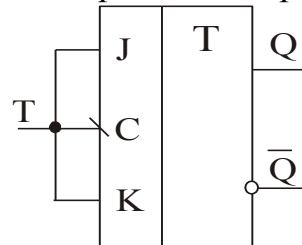
JK – тригери відносяться до універсальних пристроїв, шляхом відповідного з'єднання виводів вони перетворюються в тригери інших типів.

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

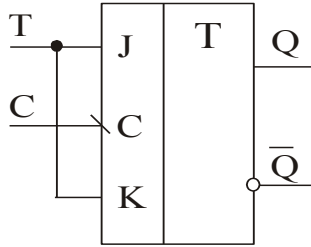
Синхронний RS-тригер



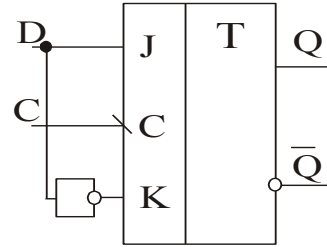
Асинхронний T-тригер



Синхронний Т-тригер



Синхронний D-тригер



Контрольні запитання

1. Призначення тригера.
2. Назвіть типи тригерів?
3. Чим відрізняються синхронні тригери від асинхронних?
4. Напишіть таблицю переходів RS-, T-, D-, JK – тригерів.
5. Нарисуйте умовне графічне зображення тригерів.
6. Нарисуйте схему RS – тригера на логічних елементах.
7. Як синхронний RS – тригер включити в режимі T-тригера?
8. Як синхронний JK – тригер включити в режимі D-тригера?

11. РЕГІСТРИ

Регістри призначені для зберігання проміжних результатів обчислень.

Всі регістри, в залежності від функціональних можливостей, поділяються на два типи: регістри зберігання (пам'яті) (рис. 11. 1) і регістри зсуву (рис. 11.2).

В свою чергу регістри зсуву поділяються на:

- за способом вводу і виводу інформації на паралельні, послідовні і комбіновані (паралельно-послідовні, послідовно-паралельні);
- за напрямком передачі (зсуву) інформації на однонаправлені і реверсивні.

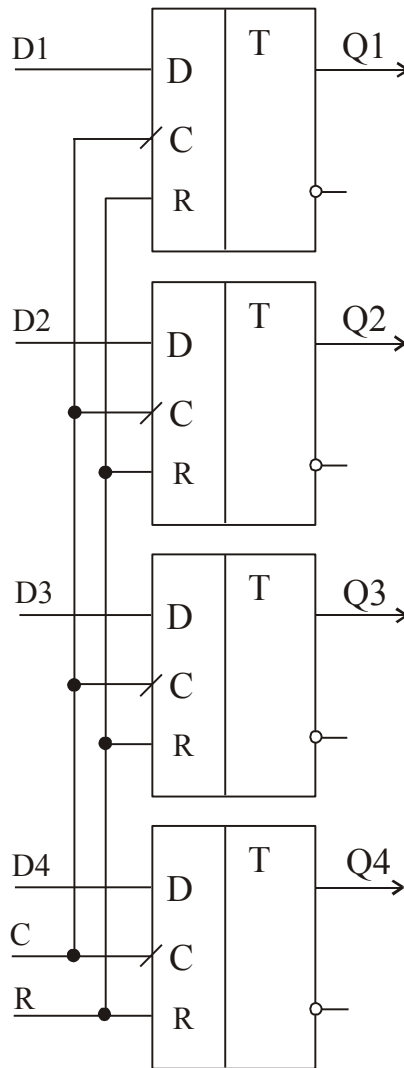


Рис. 11.1 – Регістр зберігання: D1 – D4 – паралельні інформаційні входи; C – тактовий вхід; R – вхід скидання регістра в “0”; Q1 – Q4 – виходи регістра.

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

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

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

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

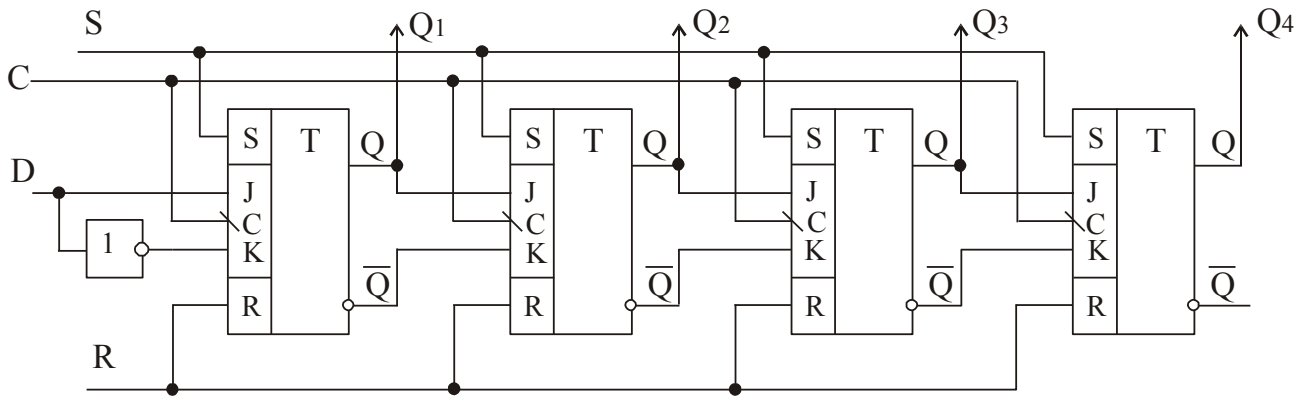


Рис. 11.2 – Чотирирозрядний регістр зсуву вправо на JK – тригерах: S – вхід встановлення регістра в “1”; C – тактовий вхід; D – інформаційний вхід; R – вхід скидання регістра в “0”; Q1 – Q4 – паралельні виходи регістра.

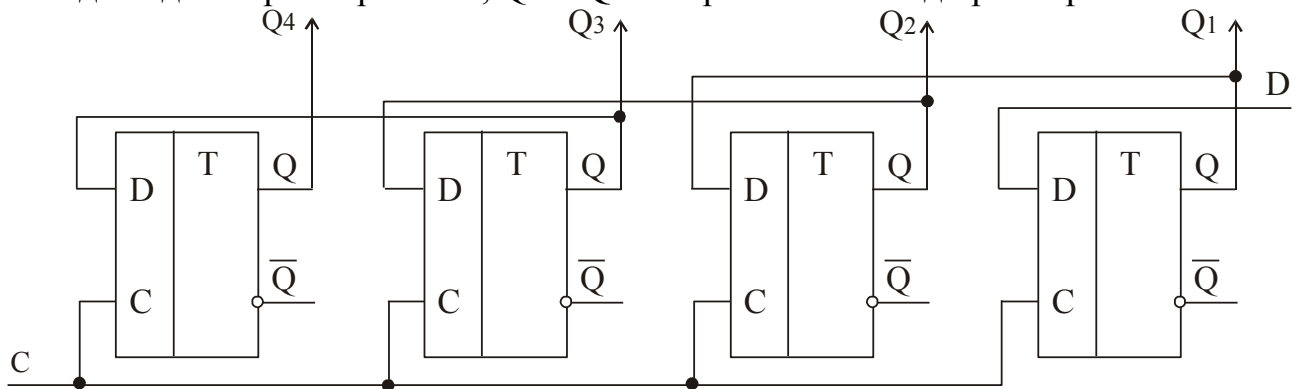


Рис. 11.3 – Чотирирозрядний регістр зсуву вліво на D-тригерах.

Універсальний регістр на JK (RS) – тригерах.

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

При цьому входи D1 – D4 – для паралельного запису інформації, блоковані. Тактові імпульси на вході C забезпечують синхронний ввід інформації в послідовному коді (з входу DS), а також зсув її вправо. За рахунок

інверсії тактових імпульсів елементом DD7.1 спрацювання тригерів відбувається по фронту наростання тактових імпульсів.

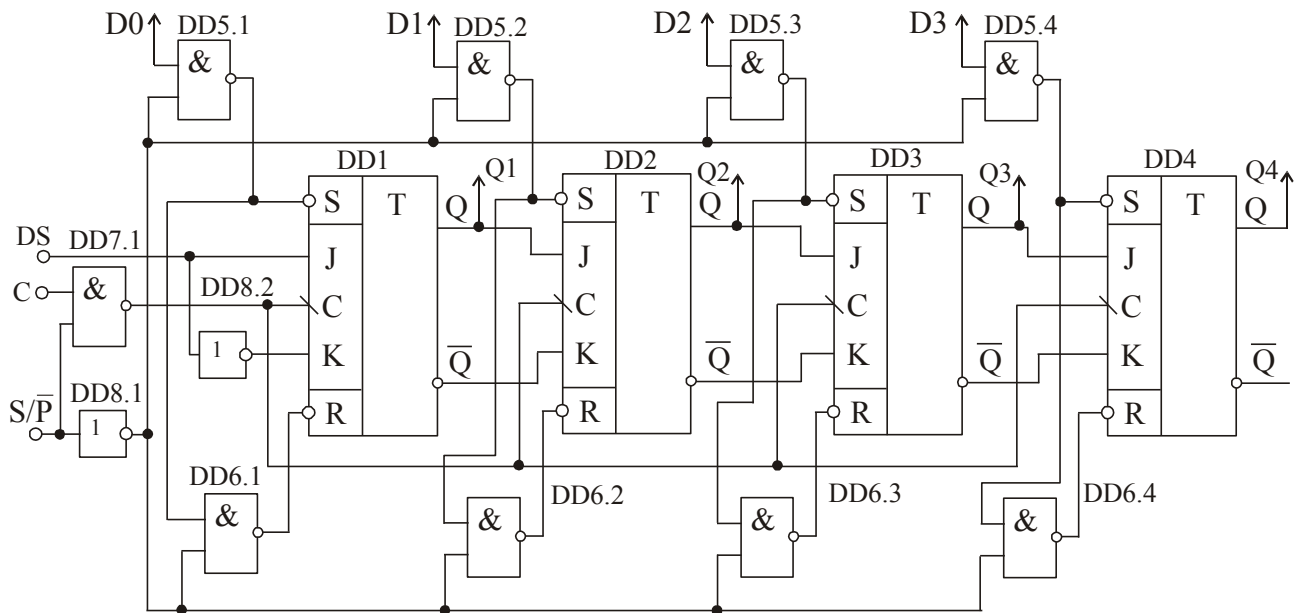


Рис. 11.4 – Універсальний регістр: S/\bar{P} – вибір режиму роботи (послідовний/паралельний); $D_0 - D_1$ – паралельні інформаційні входи; DS – послідовний інформаційний вхід; C – тактовий вхід; $Q_1 - Q_4$ – паралельні виходи.

Якщо на вході S/\bar{P} буде лог. “0” логічний елемент DD7.1 закритий і тактові імпульси не проходять на C входи тригерів. Сигнал на загальних входах елементів DD5.1 – DD5.4 і DD6.1 – DD6.4 дорівнює лог. “1”, внаслідок чого кожний із цих елементів для сигналів на паралельних входах $D_1 - D_4$ служить інвертором. Під дією вхідних сигналів паралельного запису виходи відповідних тригерів приймають той же стан $Q_i = D_i$. З появою на вході S/\bar{P} сигналу лог. “1” інформація, введена в паралельному коді, з кожним тактовим імпульсом буде зсуватися на один розряд і видаватись в послідовній формі.

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

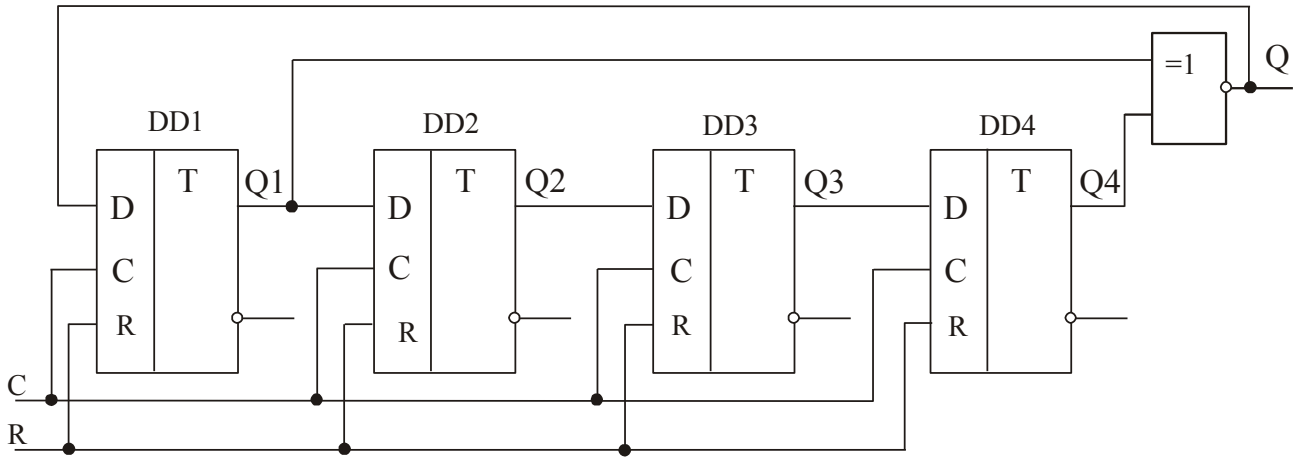


Рис. 11.5 – Генератор М – послідовності.

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.

Контрольні запитання

1. Призначення регістрів?
2. На яких елементах побудовані регістри?
3. Нарисуйте схему 3-х розрядного регістру пам'яті (зберігання).
4. Нарисуйте схему 3-х розрядного регістру зсуву вправо.
5. Нарисуйте схему 3-х розрядного регістру зсуву вліво.
6. Нарисуйте схему універсального регістру.
7. Назвіть призначення входів універсального регістру.
8. Нарисуйте схему генератора М-послідовності.

12. МІКРООПЕРАЦІЇ В РЕГІСТРАХ

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

Для запису інформації від декількох 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:

$$D_i = Y_1 \wedge A_i \vee Y_2 \wedge B_i, \quad (1)$$

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 i -го розряду регістра на основі рівняння (1) приведена на рисунку 12.1.

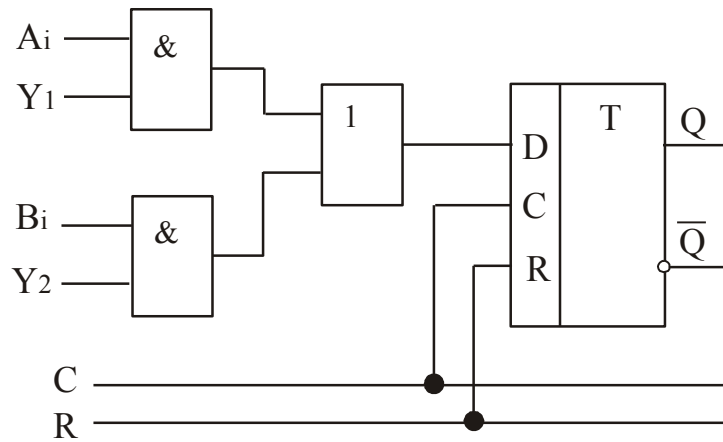


Рис. 12.1 – Схема вхідної логіки i -го розряду регістра

$Y_1 = 1, Y_2 = 0$ запис коду A_i ;

$Y_1 = 0, Y_2 = 1$ запис коду B_i .

Зчитування інформації

Для реалізації мікрооперації зчитування до виходів кожного тригера підключаються 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 i -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 повинні збігатися в часі.

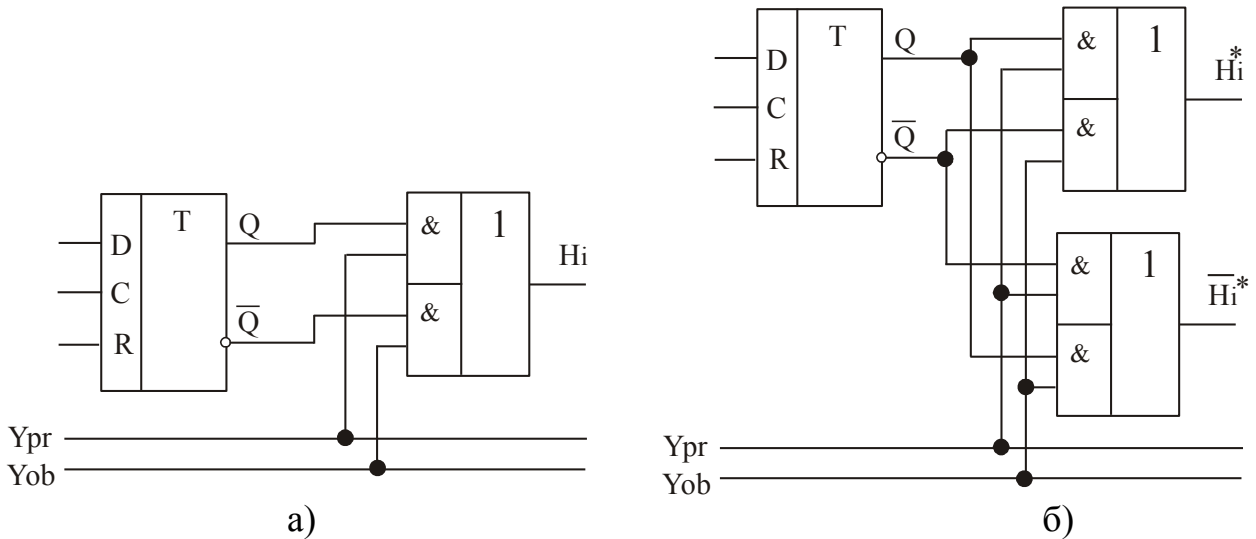


Рис. 12. 2 – Схеми вихідної логіки i – го розряду регістра для зчитування інформації: а) однофазним кодом; б) парафазним кодом.

Логічні операції в регістрах

В регістрах можуть виконуватись такі порозрядні (без перенесень) логічні мікрооперації над словами A_i і B_i : логічного додавання і множення:

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

– додавання за модулем два і його заперечення:

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

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

Логічні мікрооперації передбачають наявність першого слова A в регістрі. З врахуванням 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 падаванням інверсних значень розрядів слова B на входи R (або K) тригерів 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 по логічному входу вводиться слово V . Після цього на прямих виходах тригерів фіксується результат операції $Q = A \oplus V$ а на інверсних виходах $\overline{Q} = \overline{A \oplus V}$.

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

Мікрооперація зсуву.

Зсув – це одночасне просторове переміщення двійкового слова в розрядній сітці із збереженням порядку слідування нулів і одиниць. 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 може виконуватись 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 в регістрі A записано слова $A_n A_{n-1} \dots A_2 A_1$, де A_1 – молодший розряд; A_n – старший розряд.

Символічно мікрооперації зсуву записуються таким чином:

– арифметичні зсуви (знаковий розряд не зсувається:

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

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

– логічні зсуви (одночасно зсуваються всі розряди);

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

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

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

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

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

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

13. ЛІЧИЛЬНИКИ

Лічильники – пристрої, які під дією вхідних імпульсів переходять із одного стану в інший і при цьому відображають в певному коді число імпульсів, що поступило 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 включений 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, Джонсона.

За напрямом підрахунку лічильники поділяються на сумуючі, віднімаючі, реверсивні.

За способом організації внутрішніх зв'язків: з послідовним переносом, з паралельним переносом, з комбінованим переносом.

В **лічильниках з послідовним переносом** – імпульси, які підлягають підрахунку, поступають на вхід першого тригера, а сигнал переносу передається послідовно від одного розряду до другого (рис. 13.1, рис. 13.2). Такі лічильники складаються з асинхронних Т – тригерів з прямим або інверсним керуванням або JK – і D – тригерів, включених в режимі Т – тригера.

Основна перевага лічильників з послідовним переносом – проста схема.

Недолік – порівняно низька швидкодія, оскільки тригери спрацьовують послідовно один за другим.

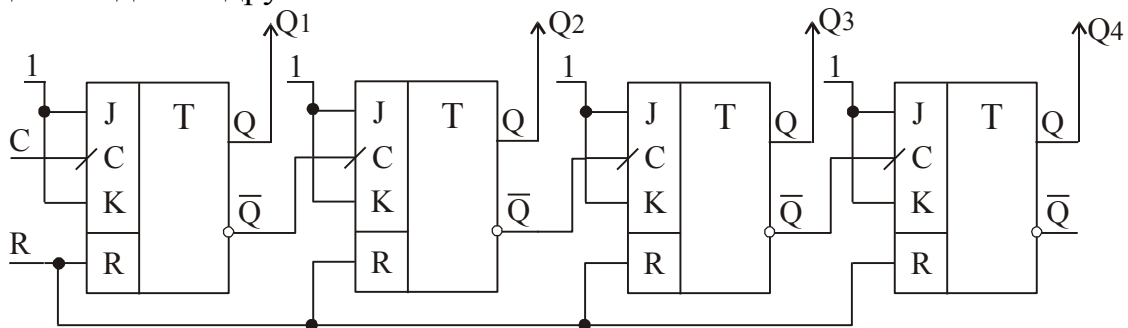


Рис. 13.1 – Функціональна схема асинхронного лічильника з послідовним переносом.

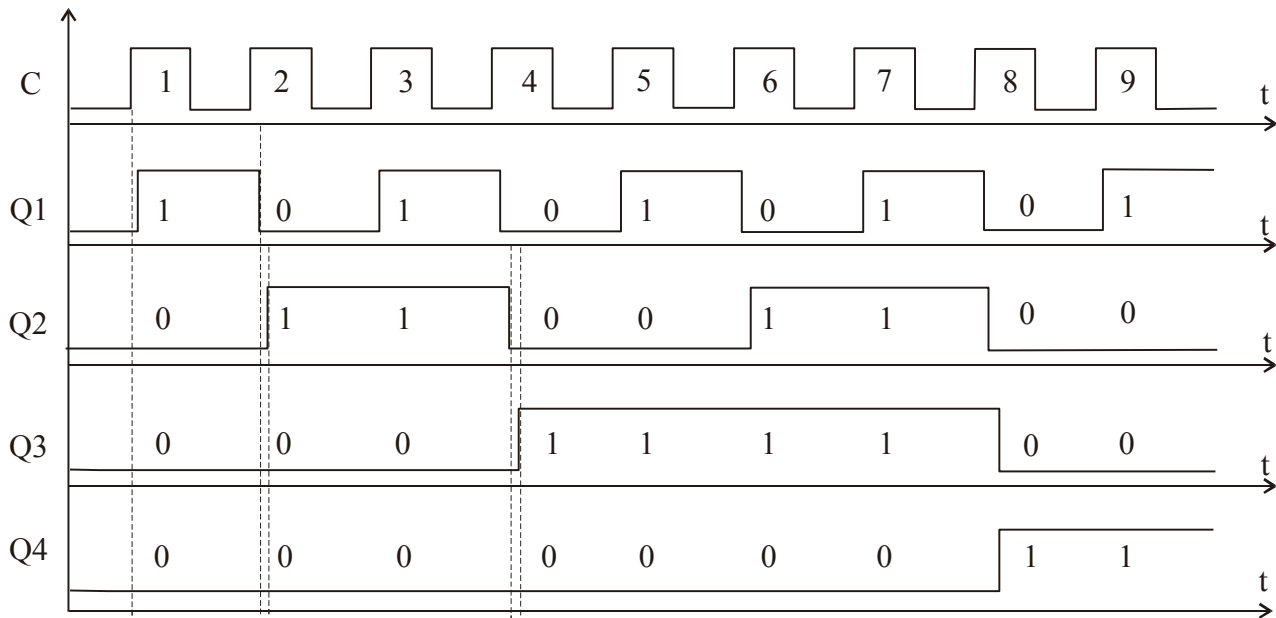


Рис. 13. 2 – Часові діаграми

Лічильники з паралельним переносом складаються із синхронних JK – і D – тригерів.

Вхідні імпульси поступають одночасно на всі тактові входи, а кожний з тригерів служить по відношенню до наступного тільки джерелом інформаційних сигналів (рис. 13. 3).

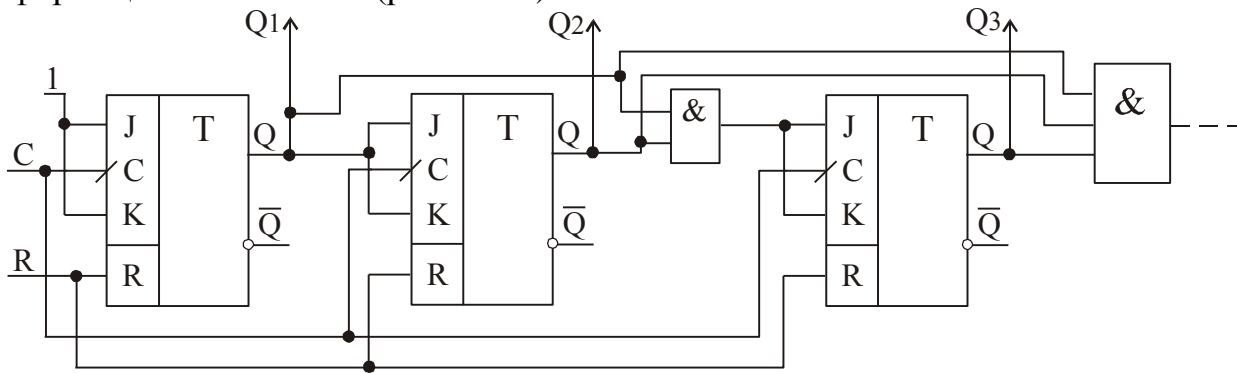


Рис. 13.3 – Функціональна схема синхронного лічильника з паралельним переносом.

Спрацювання тригерів паралельного лічильника відбувається синхронно і затримка переключення лічильника дорівнює затримці одного тригера (рис. 13.4).

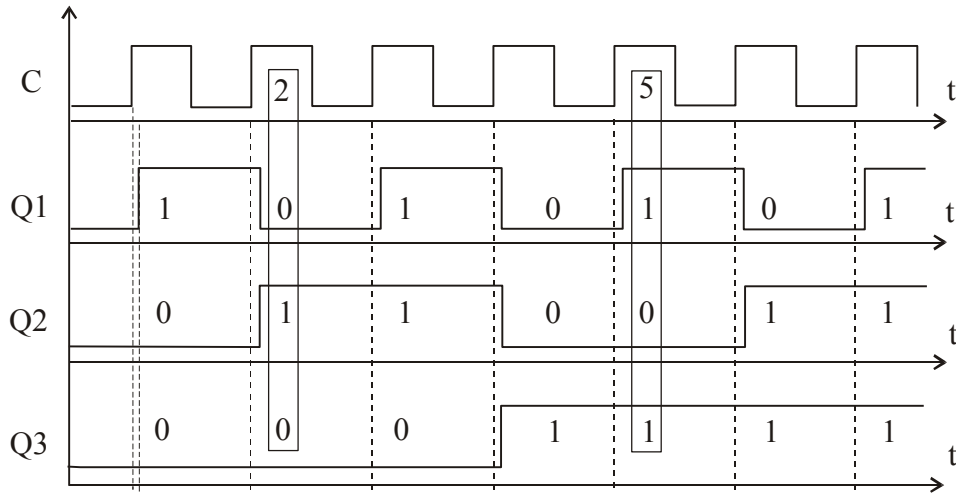


Рис. 13.4 – Часові діаграми роботи лічильника.

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

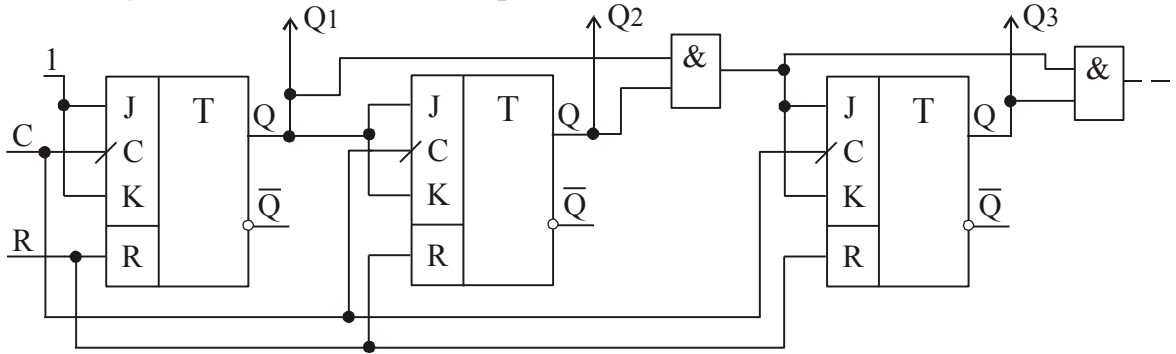


Рис. 13.5 – Функціональна схема синхронного лічильника з послідовним переносом.

Функціональна схема синхронного реверсивного лічильника приведена на рис. 13.6.

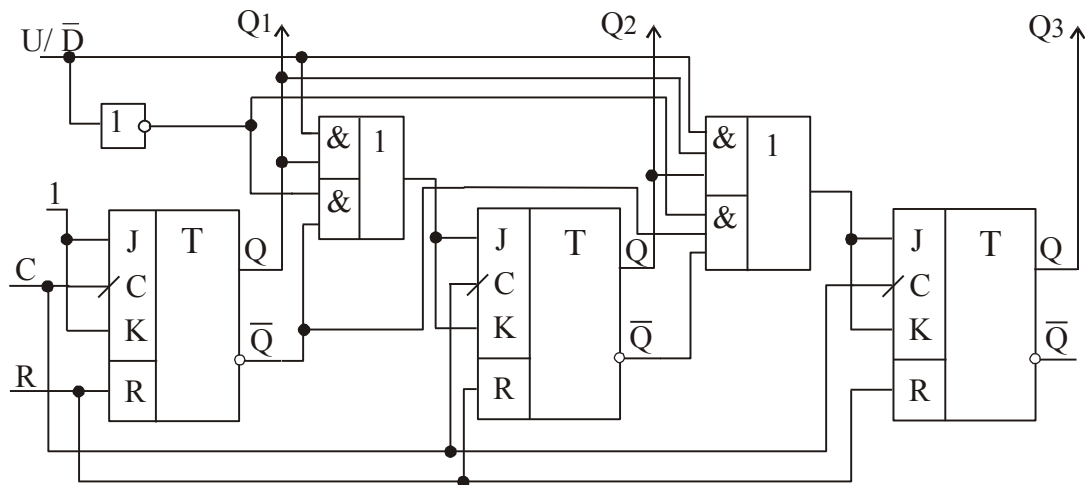


Рис. 13.6 – Функціональна схема синхронного реверсивного лічильника: U/\bar{D} – вхід вибору режиму роботи Up/Down (1 – додавання, 0 – віднімання).

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

За допомогою додаткового логічного елемента можна змінювати коефіцієнт ділення в межах $2^{m-1} < K < 2^m$. Для цього входи логічного елемента з'єднуються з виходами відповідних тригерів, а вихід логічного елемента – до входів R – (скидання тригерів в нуль) (рис.13.7). 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 тригер – від кожного восьмого імпульсу ($8 = 2^3$). Коефіцієнт ділення визначається $K = 11 = 8 + 2 + 1 = 2^3 + 2^1 + 2^0$.

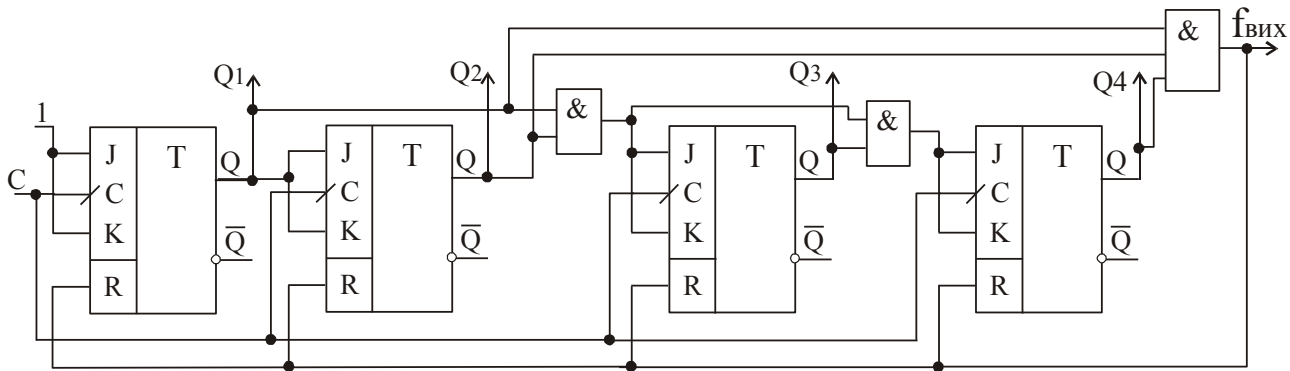


Рис. 13.7 – Функціональна схема лічильника з коефіцієнтом ділення $K=11$.

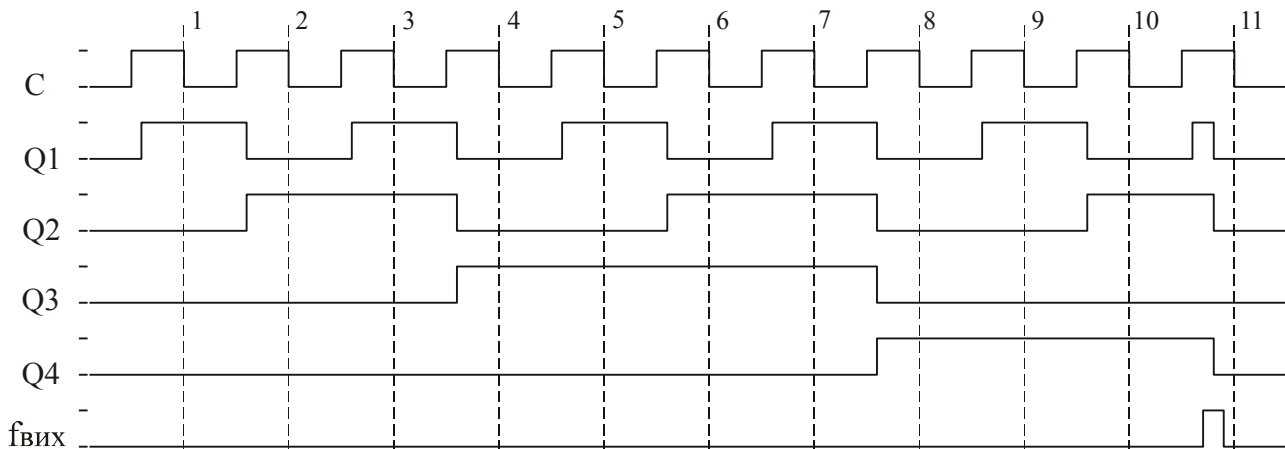


Рис. 13.8 – Часова діаграма роботи лічильника з коефіцієнтом ділення $K=11$.

Контрольні запитання

1. Призначення лічильників.
2. На яких елементах побудовані лічильники?
3. Нарисуйте схему асинхронного лічильника з послідовним переносом.
4. Нарисуйте схему синхронного лічильника з паралельним переносом.
5. Синтезуйте лічильник з заданим коефіцієнтом ділення.

14. ПРОЕКТУВАННЯ ЦИФРОВИХ АВТОМАТІВ З ПАМ'ЯТТЮ

Вузли і пристрої, які містять елементи пам'яті відносяться 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 A (вхідний алфавіт автомата A);

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

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

φ – функція переходів автомата A , яка відображає $(X \times S) \rightarrow S$, тобто ставить у відповідність будь-якій парі елементів добутку множин $(X \times S)$ елемент множини S ;

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

За способом формування функції виходів розрізняють наступні типи автоматів: автомат Мілі, автомат Мура (рис.1).

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

Автомат Мілі характеризується системою рівнянь:

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

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

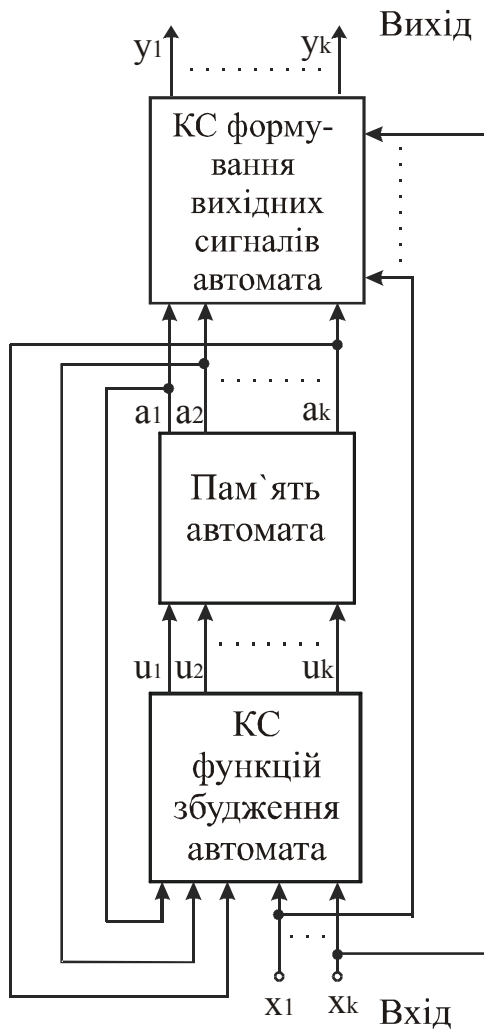
Автомат Мура – системою рівнянь:

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

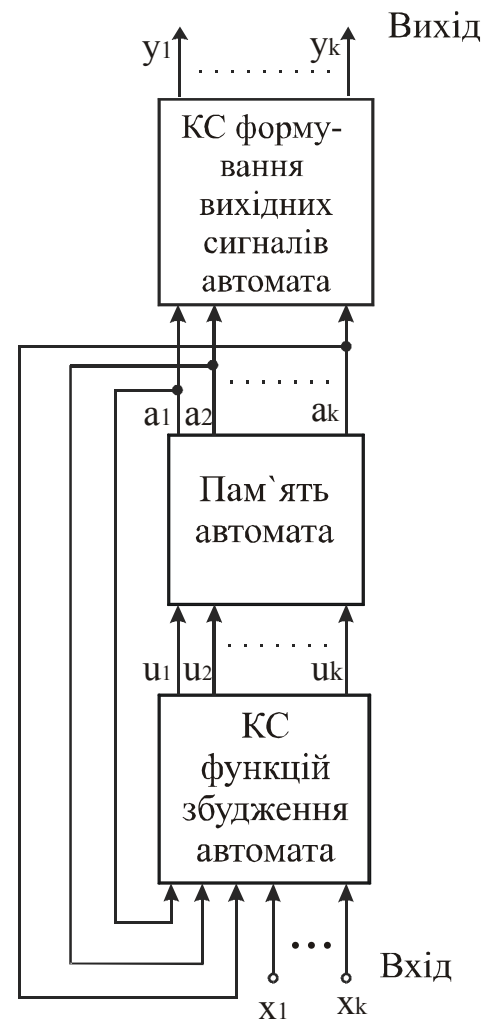
$$s(t+1) = \varphi[s(t), x(t)].$$

Синтез цифрових автоматів з пам'яттю можна розділити на наступні етапи:

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



Автомат Мілі



Автомат Мура

Рис.14. 1. – Структурні схеми автоматів з пам'яттю
Розглянемо кожний із етапів детально.

1. Кодування.

Процес заміни букв алфавітів X, Y, S цифрового автомата двійковими векторами називається кодуванням і може бути описаний таблицею (табл. 14.3, табл. 14.4, табл. 14.5). В лівій частині таблиці перераховуються всі букви (наприклад вхідного алфавіту), а в правій – двійкові вектори, які ставляться у відповідність цим буквам.

Таблиця 14.1 – Таблиця переходів

| 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 14.2 – Таблиця виходів

| 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 – $s_k = \varphi(s_i, x_j)$;

Функція виходів $y_k = \lambda(s_i, x_j)$.

Розглянемо кодування букв алфавітів S, X, Y.

Таблиця 14.3.

| Вхідні сигнали | Код вхідних сигналів |
|----------------|----------------------|
| x ₁ | 0 |
| x ₂ | 1 |

Таблиця 14.4.

| Стан | Код стану |
|----------------|-----------|
| s ₁ | 00 |
| s ₂ | 01 |
| s ₃ | 10 |

Таблиця 14.5

| Вихідні сигнали | Код вихідних сигналів |
|-----------------|-----------------------|
| y ₁ | 00 |
| y ₂ | 01 |
| y ₃ | 10 |
| y ₄ | 11 |

Таблиця переходів і виходів після кодування має вигляд:

Таблиця 14.6 – Таблиця переходів

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

Таблиця 14.7 – Таблиця виходів

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

2. Вибір елементів пам'яті автомата.

В якості елементів пам'яті структурного автомата використовують D – тригери, T – тригери, RS – тригери, JK – тригери.

Таблиці переходів тригерів.

| Стан D-тригера | Вхідний сигнал (D) | |
|----------------|--------------------|---|
| | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 0 | 1 |

| Стан T-тригера | Вхідний сигнал (T) | |
|----------------|--------------------|---|
| | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 1 | 0 |

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

| Стан JK-тригера | Вхідні сигнали (J, K) | | | |
|-----------------|-----------------------|----|----|----|
| | 00 | 01 | 10 | 11 |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |

Виберемо в якості елемента пам'яті T – тригер. Складаємо таблицю функцій збудження автомата.

Таблиця збудження елементів пам'яті будується на основі таблиці переходів (табл. 14. 6).

Таблиця переходів.

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

Складаємо рівняння.

Символами u_1 і u_2 в таблиці позначають функції збудження елементів пам'яті α_1 і α_2 .

Складаємо рівняння для побудови комбінаційної схеми збудження цифрового автомата.

$$u_1 = \alpha_1 \wedge \bar{\alpha}_2 \wedge x;$$

$$u_1 = \bar{\alpha}_1 \wedge \bar{\alpha}_2 \wedge \bar{x} \vee (\bar{\alpha}_1 \wedge \alpha_2 \vee \alpha_1 \wedge \bar{\alpha}_2) \wedge x.$$

Таблиця виходів складається на основі таблиці 14.7.

Таблиця виходів

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

Складаємо рівняння для побудови КС формування вихідних сигналів автомата.

$$y_1 = (\bar{\alpha}_1 \wedge \bar{\alpha}_2 \vee \bar{\alpha}_1 \wedge \alpha_2) \wedge x = \bar{\alpha}_1 \wedge x;$$

$$y_2 = \bar{\alpha}_1 \wedge \alpha_2 \wedge \bar{x} \vee (\bar{\alpha}_1 \wedge \alpha_2 \vee \alpha_1 \wedge \bar{\alpha}_2) \wedge x.$$

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

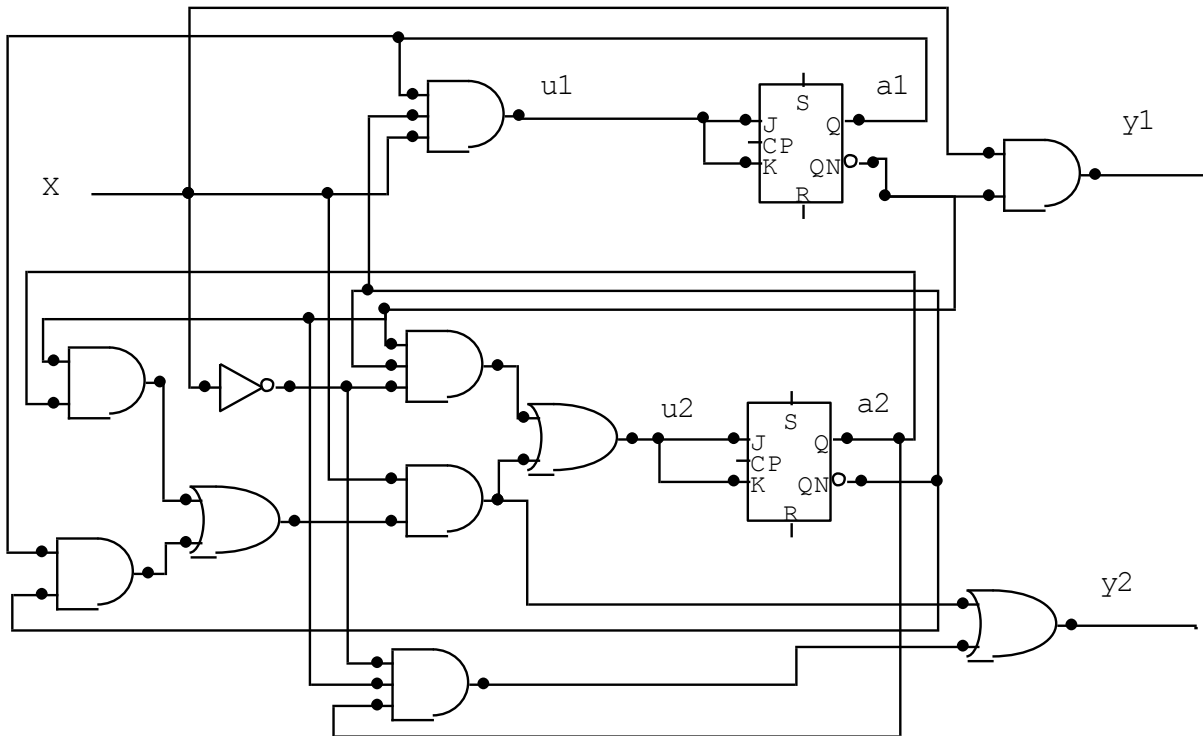


Рис. 14.2. Функціональна схема цифрового автомата Мілі.

Завдання.

Спроекувати цифровий автомат Мілі згідно заданих таблиць переходів і виходів. Комбінаційну схему збудження і комбінаційну схему формування вихідних сигналів реалізувати в заданому базисі.

Варіант №1

Таблиця переходів.

| 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 |

Тип тригера – J-K. Базис –I-HE.

Таблиця виходів.

| 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 |

Варіант №2

Таблиця переходів.

| 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 |

Тип тригера – J-K. Базис –АБО-HE.

Таблиця виходів.

| 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 |

Контрольні запитання

1. Дайте визначення цифрового автомату.
2. Нарисуйте структурну схему автомата Мілі.
3. Нарисуйте структурну схему автомата Мура.
4. Чим відрізняється цифровий автомат Мілі від цифрового автомата Мура.
5. З яких етапів складається структурний синтез цифрових автоматів.

15. МІКРОПРОГРАМНІ АВТОМАТИ

При описі функціонування різних засобів обчислювальної техніки досить часто використовується їх представлення в виді сукупності керуючого і операційного автоматів (рис.15.1).

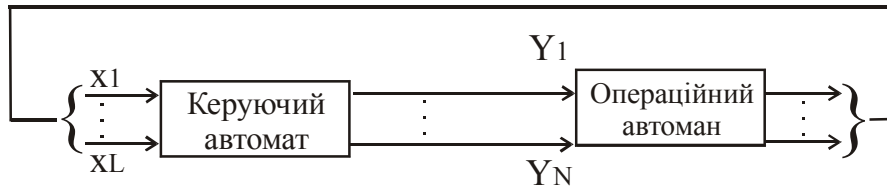


Рисунок 15.1 – Структурна схема мікропрограмного автомату

Так, в ЕОМ до операційного автомату відносяться блоки пам'яті, регістри, суматори, канали передачі інформації, шифратори і т.д. тобто всі пристрої, які виконують деякі операції, а до керуючого автомату – ту частину ЕОМ, яка координує дію перерахованих пристроїв, визначаючи послідовність оброблення в них інформації.

Задачею керуючого автомату є вироблення розподіленої в часі послідовності вихідних (керуючих) сигналів під дією яких в операційному автоматі здійснюється деяка операція.

Елементарний неподільний акт оброблення інформації в операційному автоматі, який відбувається на протязі одного моменту автоматного часу (одного такту роботи автомату), називається мікрооперацією. Прикладом мікрооперацій можуть служити "Зсув інформації", "+1", "Інверсія змінної" та інші.

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

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

Для опису мікропрограм зручно використовувати язык 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 вершини і умовної вершини ГСА представлено на рис. 15.2.

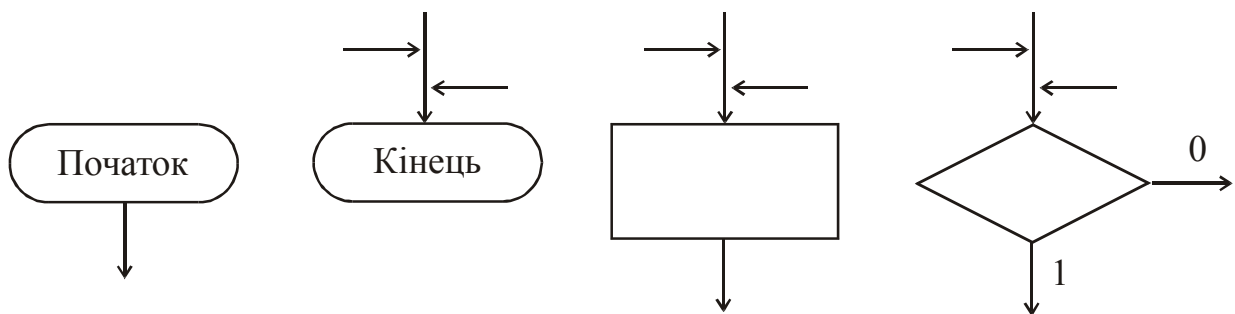


Рис. 15.2 – Умовні позначення на ГСА

ГСА повинен задовольняти наступним умовам.

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, по крайній мірі з одним входом.
 4. Будь-яка вершина ГСА лежить, по крайній мірі на одному шляху із вершини "Початок" в вершину "Кінець".
 5. Один із виходів умовної вершини може з'єднуватися з її входом, що недопустимо для операторної вершини. Такі умовні вершини інколи називають поворотними.
 6. В 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 представляє собою вихідний сигнал або сукупність вихідних сигналів керуючого автомату. Дозволяється в різних операторних вершинах записувати однакові оператори.
- Приклад ГСА, який містить три операторні і дві умовні вершини приведений на рис.15.3.

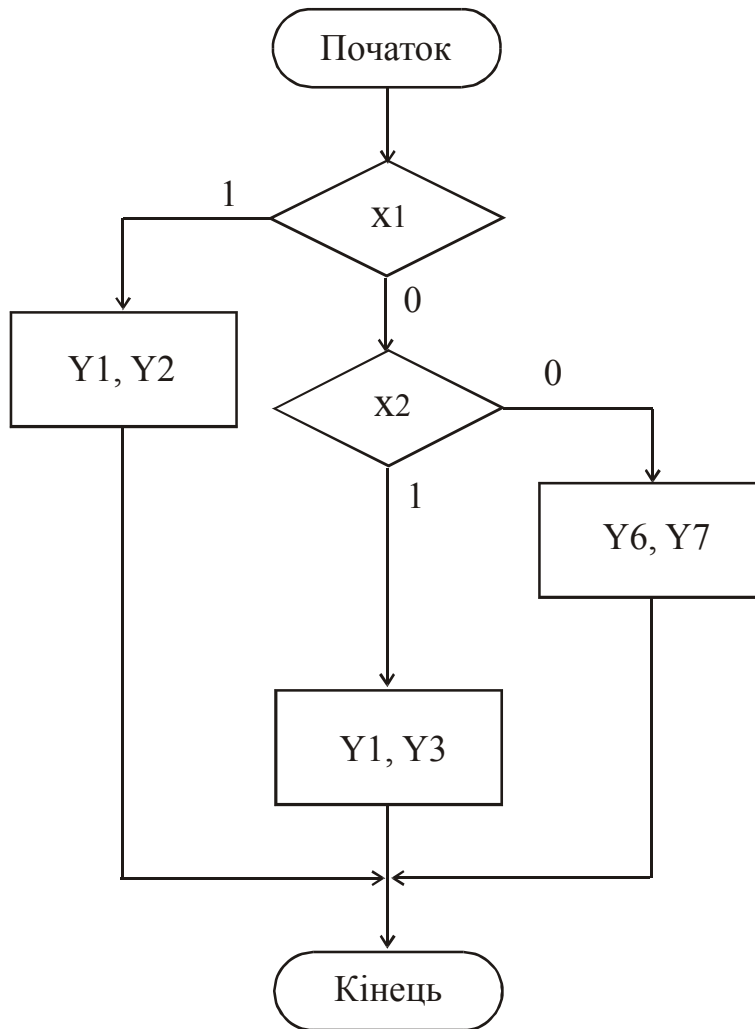


Рис.15.3– Приклад ГСА

Змістовні граф - схеми алгоритмів

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

16. ПРОГРАМОВАНІ ЛОГІЧНІ МАТРИЦІ (ПЛМ)

Синтез комбінаційних схем та цифрових автоматів з пам'яттю на ПЛМ.

Основою програмованих логічних матриць (ПЛМ) є послідовність програмованих матриць елементів І і АБО. В структуру входять 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.16.1).

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

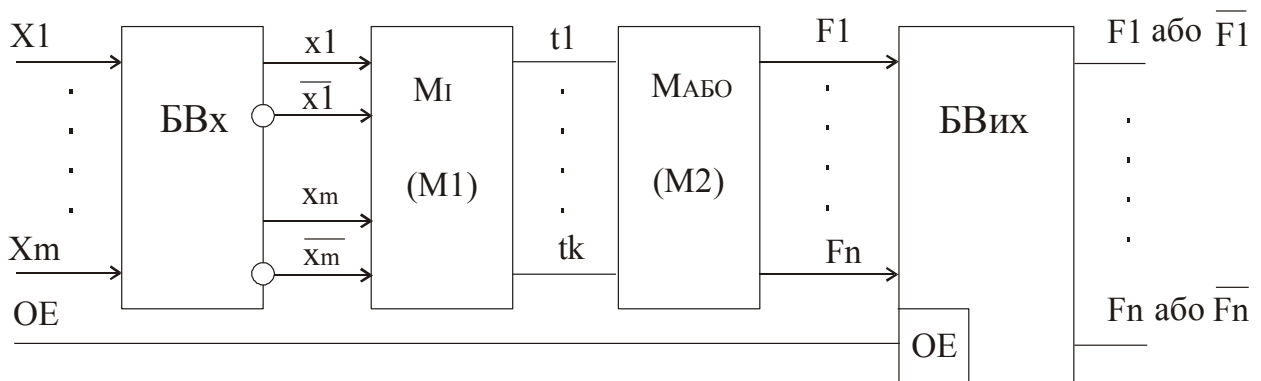


Рис.16.1 – Базова структура ПЛМ

Змінні $x_1 \dots x_m$ подаються через БВх на входи елементів І і в матриці І утворюють k термів. Терми – це кон'юнкція, яка зв'язує вхідні змінні представлені в прямій або інверсній формі. Число термів дорівнює числу кон'юнкторів, або числу виходів матриці І.

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

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

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

В матрицях є горизонтальні і вертикальні зв'язки в вузлах перетину яких при програмуванні створюються або ліквідуються елементи зв'язку (рис.16.2).

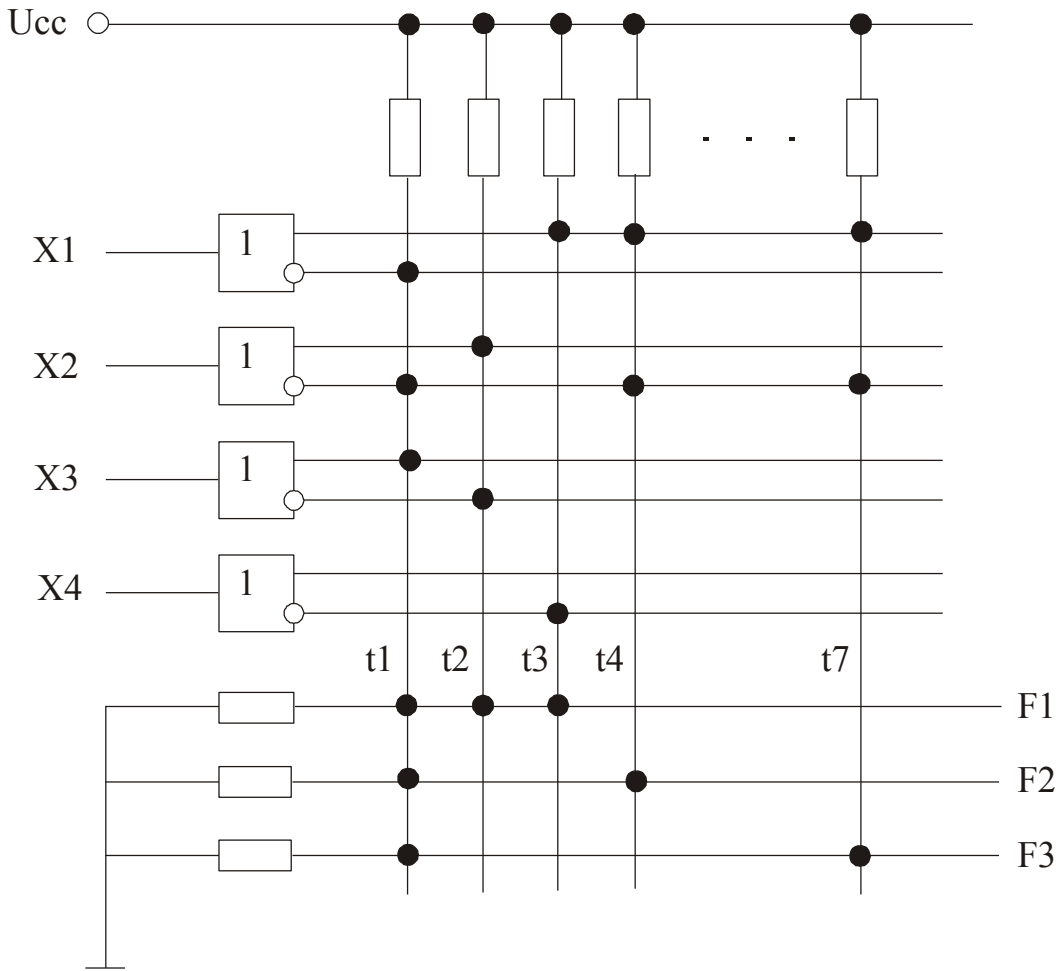


Рис. 16.2 – Матриці кон'юнкції і диз'юнкції

З допомогою ПЛМ можна реалізувати не тільки диз'юнктивні нормальні форми булевих функцій але й функції з дужками.

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

В схемі появляються зворотні зв'язки – проміжні результати з виходу знову подаються на вхід, логічна глибина схеми збільшується, збільшується затримка вироблення результату.

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

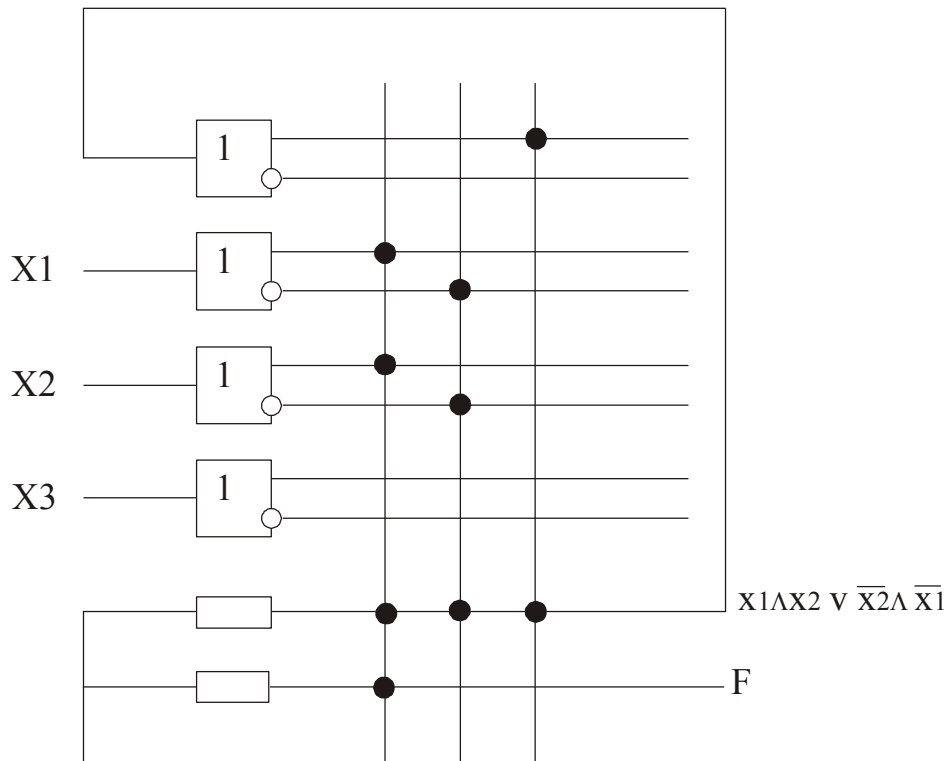


Рис 16.3 – Реалізація булевої функції F.

ПЛМ з пам'яттю.

Ці схеми дозволяють будувати автомати найбільш зручним способом, тобто крім комбінаційної частини містять на кристалі тригери (D – тригери) (рис.16.4).

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

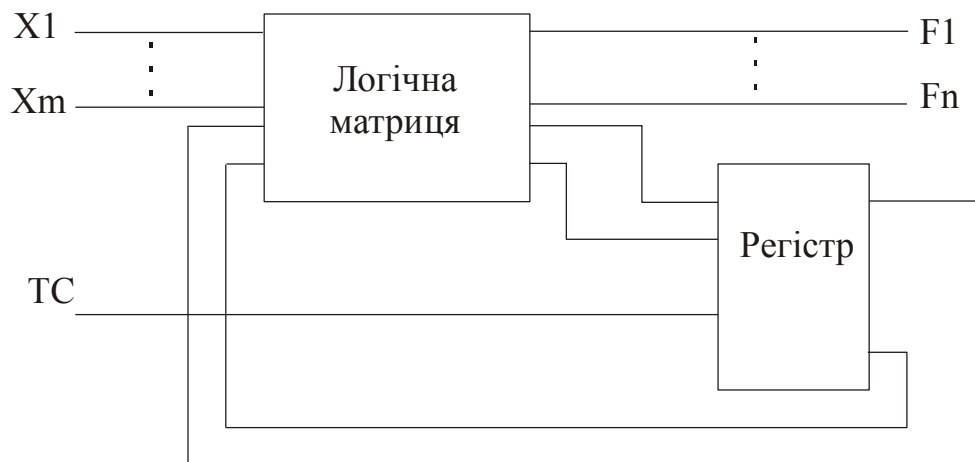


Рис. 16.4– Структура ПЛМ з пам'яттю

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

Результати обробки інформації залежить від результатів попередніх кроків, що забезпечують зворотним зв'язком з регістра на вхід ПЛМ. Максимальне число внутрішніх станів автомата 2^n . Автомат розглядається, як синхронний зворотний зв'язок активізується тільки по тактових сигналах (ТС).

ЛІТЕРАТУРА

1. Прикладная теория цифровых автоматов / Самофалов К.Г., Романкевич А.М. и др.. - Київ: Вища школа, 1987. – 369 с.
2. Савельев А.Я. Прикладная теория цифровых автоматов: Учеб. для вузов. – М.: Высш. Шк., 1987. – 272с.
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: Учебное пособие для вузов. –СПб.: Политехника, 1996. – 885 с.
6. Бабич М. П., Жуков І.А. Комп'ютерна схемотехніка: Навчальний посібник. – К.: “МП – Прес”, 2004. – 412 с., іл.
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 Вл. В. Методы синтеза надежных автоматов . – Энергия. Ленингр. Отд-ние, 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. – 224с.
15. Зельдин Е. А. Цифровые интегральные микросхемы в информационно-измерительной аппаратуре. - Л.: Энергоатомиздат. Ленингр. отд-ние, 1986.–280 с.

Підписано до друку 20.09.2011 р.
Формат 84x108/32. Папір офсетний. Друк на різнографі.
Умов. – друк. арк. 0,95. Обл. – вид. арк.0,90. Зам. №
Тираж 100 прим.

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

