МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Миколаївський національний університет імені В.О. Сухомлинського
Коледж МНУ імені В.О. Сухомлинського
Циклова комісія з інформаційних технологій, математики та статистики

МЕТОДИЧНІ МАТЕРІАЛИ

для вивчення тем, запланованих

на самостійне опрацювання з навчальної дисципліни

«ПРИКЛАДНА ТЕОРІЯ ЦИФРОВИХ АВТОМАТІВ»

для студентів галузі знань 11 «Математика та статистика»

спеціальності 113 «Прикладна математика»

МИКОЛАЇВ – 2019

Методичні матеріали для вивчення тем, запланованих на самостійне опрацювання з навчальної дисципліни «Прикладна теорія цифрових автоматів» (для студентів ІІІ курсу спеціальності 113 «Прикладна математика» закладів вищої освіти І-ІІ рівнів акредитації, які здійснюють підготовку молодших спеціалістів на основі базової загальної середньої освіти) / Коледж МНУ імені В.О.Сухомлинського; уклад.: Божко Н.В. – М., 2019.
210 с.

Укладач: Божко Надія Валеріївна, викладач вищої категорії, старший викладач циклової комісії з інформаційних технологій, математики та статистики

Рецензенти:

Поздєєв В.О. – доктор фізико-математичних наук, професор, завідувач кафедри комп'ютерних наук та прикладної математики механіко-математичного факультету Миколаївського національного університету імені В.О. Сухомлинського;

Приходько С.Б – доктор технічних наук, професор, заступник директора Навчально-наукового інституту комп'ютерних наук та управління проектами, завідувач кафедри «Програмне забезпечення автоматизованих систем» Національного університету кораблебудування імені адмірала Макарова;

Шевчук Р.П. – кандидат технічних наук, доцент кафедри комп'ютерних наук Тернопільського національного економічного університету, директор Самбірської філії ТНЕУ.

Розглянуто та схвалено для використання в роботі на засіданні циклової комісії з інформаційних технологій, математики та статистики, протокол №6 від 15.01.2019р.

Рекомендовано до друку на засіданні педагогічної ради Коледжу МНУ імені В.О.Сухомлинського, протокол №7 від 24.01.2019р.

ЗМІСТ

ТЕМА 9. СХЕМОТЕХНІКА ЦИФРОВИХ АВТОМАТІВ

1 Елементарні цифрові автомати: тригери, регістри, лічильники.

2 Базові вузли ЦА на комбінаційних схемах: суматори, дешифратори, шифратори, селектори імпульсів, генератори синхросигналів.

ТЕМА 10. СПОСОБИ ПОДАННЯ ЦИФРОВОГО АВТОМАТА

1 Математична модель цифрового автомату. Автомати Мілі і Мура.

2 Способи подання цифрових автоматів: табличний і графічний (орграфом). Еквівалентність цифрових автоматів.

3 Структурна модель цифрового автомату.

4 Етапи структурного синтезу цифрового автомату.

ТЕМА 11. СИНХРОНІЗАЦІЯ І ЗАБЕЗПЕЧЕННЯ СТІЙКОСТІ ЦИФРОВОГО АВТОМАТА

1 Явище «гонок» та методи забезпечення стійкості цифрового автомату.

2 Синхронізація роботи цифрового автомату.

ТЕМА 12. КЕРУЮЧИЙ І ОПЕРАЦІЙНИЙ БЛОКИ АВТОМАТА

1 Принцип мікропрограмного керування.

2 Поняття операційних та керуючих автоматів (ОА і КА).

3 Способи опису алгоритмів і мікропрограм. Граф-схема алгоритму (ГСА).

4 Синтез мікропрограмних автоматів (Мілі та Мура) за граф-схемою алгоритму.

5 Структурний синтез мікропрограмних автоматів.

ТЕМА 13. СИНТЕЗ ОПЕРАЦІЙНИХ АВТОМАТІВ

1 Формалізований опис операційного автомату. Закодовані мікрооперацийна та мікрокомандна схеми алгоритму. Основна таблиця автомату.

2 Граф-схема переходів. Системи рівнянь переходів та виходів. Кодування внутрішніх станів автомату Схема операційного автомату.

ГЛОСАРІЙ ТЕРМІНІВ

ПЕРЕЛІК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

ПЕРЕДМОВА

Електронні цифрові обчислювальні машини (ЕОМ, комп'ютери) і інші засоби обчислювальної техніки (ОТ) мають особливе значення для систем управління і автоматики, де вони виконують основну частину роботи по збиранню, зберіганню та обробці інформації, необхідної для формування керівних дій в обмежений проміжок часу. В зв'язку з цим сучасні ЕОМ становлять собою базу не тільки для існуючих, але й для перспективних високоефективних автоматичних і автоматизованих систем керування технологічними процесами і виробництвом, систем автоматизації пошуку інформації, наукових досліджень. Цифрові електронні обчислювальні машини є окремим, але найбільш поширеним видом цифрових автоматів. Для успішного вивчення загальних принципів обробки цифрової інформації раціонально відсторонитися від реального апаратного забезпечення комп'ютера і розглядати його як деякий абстрактний цифровий автомат, призначений для обробки інформації, представленої в цифровій формі. Знання теорії таких автоматів необхідні для успішного пошуку нових принципів побудови комп'ютерів, вдосконалення вже відомих алгоритмів обробки цифрової інформації, грамотного застосування обчислювальної техніки в системах управління і автоматики і розробки різноманітного програмного забезпечення для таких систем.

Самостійна робота студента є невід’ємною складовою освітнього процесу у вищому навчальному закладі, в процесі якої заплановані завдання виконуються студентом під методичним керівництвом викладача, але без його безпосередньої участі. СРС є основним засобом засвоєння ним навчального матеріалу в час, вільний від обов’язкових навчальних занять.

Метою СРС є системне і послідовне засвоєння в повному обсязі навчальної програми та формування у студентів самостійності у здобутті і поглибленні знань як риси характеру, що сприятиме підвищенню конкурентоспроможності майбутніх фахівців на світовому ринку праці.

Самостійна робота є одним з найважливіших компонентів освітнього процесу, що передбачає інтеграцію різних видів індивідуальної та колективної навчальної діяльності, яка здійснюється як під час аудиторних, поза аудиторних занять, без участі викладача, так і під його безпосереднім керівництвом.

Зміст СРС визначається робочою програмою навчальної дисципліни, відповідним методичним матеріалом, завданнями та вказівками викладача.

Для успішного вивчення дисципліни «Прикладна теорія цифрових автоматів» необхідні знання з попередніх дисциплін «Вища математика», «Дискретна математика», «Програмування».

Мета викладання дисципліни «Прикладна теорія цифрових автоматів»: надати студентам інформацію про принципи виконання арифметичних і логічних операцій в ЦЕОМ, логічні основи цифрових автоматів (ЦА), методи синтезу схем комбінаційної дії та схем з пам'яттю.

Дисципліна вивчається протягом двох семестрів (п’ятого та шостого).

Вивчення дисципліни в 5 семестрі закінчується здачею заліку, в 6 семестрі іспитом.

У результаті вивчення курсу студент повинен: знати про представлення інформації в комп'ютері, виконання основних та неосновних арифметичних операцій в позиційних системах числення, методи логічного контролю комп'ютера, перемикальні функції, проектування комбінаційних схем, методи аналізу та синтезу логічних схем, основні поняття теорії інформації та кодування; вміти працювати з різними типами даних, що використовуються в цифрових автоматах, розробляти алгоритми функціонування арифметичних пристроїв, алгоритми виконання арифметичних операцій в різних системах числення, розробляти комбінаційні схеми для реалізації системи перемикальних функцій на заданому елементному базисі, виконувати мінімізацію функцій та отримувати необхідні операторні форми, виконувати структурний синтез синхронних та асинхронних автоматів, застосовуючи способи мінімізації функцій. Ці знання необхідні при проектуванні цифрових вузлів і блоків різних пристроїв ЕОМ.

На самостійну роботу з дисципліни відводиться 80 годин.

Самостійна робота студентів з дисципліни «Прикладна теорія цифрових автоматів» включає такі види робіт:

- самостійна робота студента під час аудиторних занять;
- робота над конспектами лекцій, підготовка до практичних та лабораторних робіт;
- вивчення навчального матеріалу за підручниками, навчальними посібниками, методичними вказівками, опрацювання матеріалу за першоджерелами, науковою і спеціальною літературою;
- робота з бібліотечними фондами та дистанційними джерелами з метою пошуку необхідної інформації;
- пошук інформації з використанням мережі Internet.
- виконання вправ в робочому зошиті для самостійної роботи з дисципліни.

В даних методичних матеріалах подано плани вивчення тем, ключові терміни та поняття, розглянуто схеми, їх принцип дії, характеристики.

Теоретичні положення підкріплено прикладами виконання практичних задач.

Для самоперевірки опрацьованого матеріалу треба скористатися питаннями та задачами, які вміщені в кінці кожної теми.

Для отримання більш широкої інформації щодо вище перелічених тем слід скористатися літературою, яка наведена в кінці даної розробки.

ТЕМАТИКА САМОСТІЙНОЇ РОБОТИ СТУДЕНТА

Перелік тем дисципліни «Прикладна теорія цифрових автоматів» для самостійної та індивідуальної роботи студентів денної форми навчання наведено в таблиці 1.

Таблиця 1

Перелік тем дисципліни, які виносяться на самостійну роботу для студентів денної форми навчання

№ з/п	Назва теми	Кількість годин
1	**ВСТУП. ВСТУП. ІСТОРІЯ РОЗВИТКУ ЦИФРОВИХ АВТОМАТІВ** Мета і завдання дисципліни. Коротка історія розвитку цифрових автоматів. Інформаційні основи цифрових автоматів. Інформація і загальні принципи її перетворення і опрацювання.	4
2	**ПОНЯТТЯ ПРО ЦИФРОВІ АВТОМАТИ** Роль і місце ЕОМ в системах управління. Апаратні засоби зберігання і обробки інформації. Поняття про цифровий автомат і алгоритм.	4
3	**ПРЕДСТАВЛЕННЯ ЧИСЛОВОЇ ІНФОРМАЦІЇ В ЦИФРОВИХ АВТОМАТАХ** Системи числення. Алгоритми переведення чисел з однієї системи в іншу. Форми і формати зображення чисел в цифрових автоматах. Зображення чисел в прямому, оберненому і доповнювальному кодах.	4
4	**АРИФМЕТИЧНІ ДІЇ З ДВІЙКОВИМИ ЧИСЛАМИ** Операції додавання і віднімання чисел з фіксованою комою. Операції додавання і віднімання чисел з плаваючою комою. Алгоритми множення і ділення чисел з фіксованою комою. Алгоритми множення і ділення чисел з плаваючою комою.	6
5	**КОДУВАННЯ ІНФОРМАЦІЇ. ПОНЯТТЯ ПРО КОНТРОЛЬ РОБОТИ ЦА** Поняття про кодування і коди. Послідовний і паралельний коди. Код Хеммінга. Контроль за парністю і за модулем.	6
6	**ЛОГІЧНІ ОСНОВИ ЦИФРОВИХ АВТОМАТІВ** Основні поняття теорії елементарних функцій алгебри логіки. Основні закони алгебри логіки та їх використання	6

	для подання одних функцій логіки через інші. Основні властивості функцій алгебри логіки. Поняття про логічний базис.	
7	**АНАЛІЗ І СИНТЕЗ КОМБІНАЦІЙНИХ СХЕМ** Способи реалізації булевих функцій. Аналогія між логічною функцією і комбінаційною схемою. Представлення функцій в ДДНФ і ДКНФ. Мінімізація логічних функцій.	6
8	**СИСТЕМИ (СЕРІЇ) ЛОГІЧНИХ ЕЛЕМЕНТІВ** Основні поняття. Класифікація логічних елементів. Базові логічні елементи різних серій, їх принципи побудови та технічні характеристики.	6
9	**СХЕМОТЕХНІКА ЦИФРОВИХ АВТОМАТІВ** Елементарні цифрові автомати: тригери, регістри, лічильники. Базові вузли ЦА на комбінаційних схемах: суматори, дешифратори, шифратори, селектори імпульсів, генератори синхросигналів.	8
10	**СПОСОБИ ПОДАННЯ ЦИФРОВОГО АВТОМАТА** Математична модель цифрового автомату. Автомати Мілі і Мура. Способи подання цифрових автоматів: табличний і графічний (орграфом). Еквівалентність цифрових автоматів. Структурна модель цифрового автомату. Етапи структурного синтезу цифрового автомату.	8
11	**СИНХРОНІЗАЦІЯ І ЗАБЕЗПЕЧЕННЯ СТІЙКОСТІ ЦИФРОВОГО АВТОМАТА** Явище «гонок» та методи забезпечення стійкості цифрового автомату. Синхронізація роботи цифрового автомату.	6
12	**КЕРУЮЧИЙ І ОПЕРАЦІЙНИЙ БЛОКИ АВТОМАТА** Принцип мікропрограмного керування. Поняття операційних та керуючих автоматів (ОА і КА). Способи опису алгоритмів і мікропрограм. Граф-схема алгоритму (ГСА). Синтез мікропрограмних автоматів (Мілі та Мура) за граф-схемою алгоритму. Структурний синтез мікропрограмних автоматів.	8
13	**СИНТЕЗ ОПЕРАЦІЙНИХ АВТОМАТІВ** Формалізований опис операційного автомату. Закодовані мікрооперацийна та мікрокомандна схеми алгоритму. Основна таблиця автомату. Граф-схема переходів. Системи рівнянь переходів та виходів. Кодування внутрішніх станів автомату Схема операційного автомату.	8
ВСЬОГО:		**80**

Поряд із основними темами для самостійної та індивідуальної роботи, студент також може отримати додаткові завдання для підвищення свого рейтингу, які включають:

- написання реферату, презентації, відеофільму по темі дисципліни;
- виступ з додатковою доповіддю по темі дисципліни на практичному занятті, конференції, науковому семінарі, олімпіаді;
- підготовка статті, проекту, творчої роботи на конкурс;
- активна робота під час лекцій, практичних, лабораторних занять, консультацій та ін.

ТЕМА 1. ВСТУП. ІСТОРІЯ РОЗВИТКУ ЦИФРОВИХ АВТОМАТІВ.

План

1. Мета і завдання дисципліни.
2. Коротка історія розвитку цифрових автоматів.
3. Інформаційні основи цифрових автоматів.
4. Інформація і загальні принципи її перетворення і опрацювання.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Мета і завдання дисципліни.

Попередні умови – вивчення курсу «Прикладна теорія цифрових автоматів» передбачає наявність систематичних та ґрунтовних знань із суміжних курсів (дискретна математика, комп'ютерна електроніка), цілеспрямованої роботи над вивченням спеціальної літератури, активної роботи на лекціях та практичних заняттях, виконання лабораторних робіт, самостійної роботи та виконання індивідуальних завдань.

Курс умовно роздільний на дві частини:

- частина 1 - «Комп'ютерна арифметика»;
- частина 2 - «Логічні основи і синтез цифрових автоматів».

У першій частині курсу вивчаються форми і методи представлення чисел в сучасних КС, алгоритми перетворення чисел різних систем числення, методи виконання основних арифметичних операцій в різних системах числення, методи виконання арифметичних операцій в бінарно- кодових і спеціальних системах численнях, структурні і операційні схеми основних арифметичних пристроїв.

У другій частині курсу вивчаються логічні основи цифрових автоматів, математичний апарат булевої алгебри, методи мінімізації булевих рівнянь, описи цифрових автоматів (ЦА), методи синтезу і аналізу ЦА (в т.ч. канонічні) на логічних елементах різних базисів, графи і граф схеми автоматів (ГСА). Контроль правильності виконання арифметичних і логічних операцій пристроями. Студенти повинні отримати практичні навики проектування елементарних, логічних і принципових електричних схем типових комбінаційних цифрових автоматів і автоматів з пам'яттю.

Мета і завдання вивчення дисципліни. Мета викладання дисципліни «Прикладна теорія цифрових автоматів» полягає у вивченні 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.

Завдання дисципліни визначаються тим, щоб дати студентам теоретичну та практичну підготовку використовувати методи синтезу операційних та керуючих цифрових автоматів при формалізації практичних задач.

В результаті вивчення дисципліни студент повинен знати:

- системи числення, форми та методи подання чисел в ПК;
- алгоритми виконання основних арифметичних та логічних операцій з числами в різних системах числення;
- основи математичної логіки;
- основи синтезу і аналізу цифрових операційних та керуючих автоматів.

В результаті вивчення дисципліни студент повинен вміти:

- розробляти алгоритми функціонування арифметичних пристроїв на підставі форми подання інформації, алгоритмів виконання арифметичних операцій в різних системах числення;
- розробляти комбінаційні схеми для реалізації системи перемикальних функцій у заданому елементному базисі;
- виконувати абстрактний синтез цифрових автоматів.

2. Коротка історія розвитку цифрових автоматів.

Здобутки у створенні і застосуванні засобів обчислювальної техніки визначають не тільки рівень розвитку промисловості та організацію різних процесів управління, а й можливості фундаментальних досліджень в різних галузях науки. Саме тому цифрова техніка стала найбільш важливим фактором технічного прогресу і істотною частиною продуктивних сил у провідних країнах світу.

Перед Другою Світовою Війною (1939-1945 рр.) і 10 років потому, цифрові пристрої були головним чином засновані на релейних схемах. Суворий доказ того, що Булева алгебра може використовуватися для аналізу релейних схем, була висунута радянським фізиком Шестаковим В. І. в 1935 р. Американський інженер Шеннон К. Е. і японський вчений Накашима привели аналогічні докази в 1936-1938 р. Шестаков В. І. показав, що релейні схеми здатні моделювати функції алгебри логіки. Причому істинність і хибність висловлювань моделюється замкнутими або відкритими контактами електричного кола.

Розвиток електроніки і мікроелектроніки стимулювали створення і розвиток теорії цифрових автоматів. Найбільш важливим досягненням 60-х років було створення ефективної моделі цифрового автомата з пам'яттю. Модель кінцевого автомата стала фундаментальною частиною методів синтезу в теорії автоматів з пам'яттю. Таким чином, теорія релейних пристроїв переросла в теорію дискретних (цифрових) автоматів.

Автомати, що широко застосовуються в практиці, поділяються на два класи: автомати Мілі та автомати Мура, названі так на честь американських вчених, які досліджували вперше ці типи автоматів.

Істотний вплив на теорію цифрових автоматів зробив винахід комп'ютера. Академік Глушков В. М. вніс великий вклад у розвиток теорії автоматів, особливо в застосуванні до ЕОМ. Численні теоретичні роботи в області автоматичного синтезу пристроїв ЕОМ пов'язані з його ім'ям.

Практика проектування поставила нові складні завдання не тільки в області теорії цифрових автоматів, а й у теорії алгоритмів, теорії інформації та теорії систем. Все це призвело до подальшого розвитку цифрових автоматів і переростанню її в розділ технічної кібернетики.

Слід зазначити, що радянські вчені займають провідні позиції в світі в галузі теорії кінцевих автоматів. Так, перша монографія по релейно-контактним схемам написана Гавриловим М. А., перша машина для автоматичного аналізу та синтезу схем розроблена Пархоменко П. П. і Рогинським В. Н., логічна мова для проектування алгоритмів синтезу вперше в світі розроблена Закревським А.Д. До подібних робіт відносяться дослідження з синтезу програмних автоматів Лазарева В. Г., дослідження по синтезу асинхронних автоматів Якубайтиса Е. А., методи синтезу мікропрограмних автоматів Баранова С. І .

3. Інформаційні основи цифрових автоматів.

Існують два різних підходи до вивчення явищ з інформаційної точки зору: безперервний і дискретний. При безперервному підході всі досліджувані явища розглядаються як змінні векторні поля. Конкретна фізична природа таких векторних полів, а також їх кількісні, просторові і тимчасові масштаби при цьому несуттєві. Завдання інформації складається у виборі якого-небудь певного (змінного) поля з фіксованої заздалегідь сукупності таких полів. Характерним для безперервного підходу є те, що всі величини, які описують явище (компоненти векторів, просторів і часові координати) є речовими числами і можуть змінюватися безперервно.

При дискретному підході також мають справу з змінними векторними полями. Але компоненти векторів, а також просторів і часові координати приймають дискретні ряди значень. Найбільш вживаними є випадки, коли число значень, прийнятих компонентами векторів і просторовими координатами звичайними (поле задано в кінцевому числі точок).

Завдання інформації при дискретному підході зводиться до завдання кінцевих послідовностей кінцевозначних (постійних) векторних полів. Вводячи для кожного такого поля спеціальне буквене позначення ми отримуємо можливість задавати інформацію кінцевими послідовностями букв. Подібний спосіб завдання дискретної інформації називається алфавітним, сукупність елементарних символів (букв), з яких складається інформація - алфавітом, а кінцеві послідовності букв алфавіту - словами в даному алфавіті.

Алфавітний спосіб вживається при завданні лексичної (мовної) і числової інформації і є досить універсальним. Що ж до інформації, що задається в безперервній формі, то на практиці, застосовуючи методи апроксимації її завжди вдається представити за будь-яким наперед заданим ступенем точності в дискретній формі.

Незважаючи на універсальність алфавітного способу завдання інформації, користування ним далеко не завжди є природним, так що специфічні методи для вивчення безперервної інформації повністю зберігають своє значення.

Процес завдання інформації складається у виборі певного слова в деякому фіксованому кінцевому алфавіті і сукупності всіх можливих слів у цьому алфавіті.

Подібно до того, як з усяким явищем в природі або суспільстві пов'язана несуча цим явищем інформація. Взаємозв'язок явищ призводить до поняття про перетворення інформації.

Припустимо, що будь-яке явище α з деякого класу А явищ тягне за собою деяке певне явище β, з того ж самого або будь-якого іншого класу В явищ. У такому випадку говорять, що нам задано перетворення інформації $\alpha \to \beta = f(\alpha)$. Відтак з абстрактної точки зору перетворення інформації є не що інше, як відображення одного класу явищ в інший клас явищ.

Об'єкт, який здійснює таке відображення, зазвичай називається перетворювачем інформації.

4. Інформація і загальні принципи її перетворення і опрацювання.

У загальному випадку, всі електронні пристрої, у тому числі і ЕОМ, ділять на аналогові, цифрові і змішані.

Під комп'ютером розуміють цифрову обчислювальну машину (ЦОМ), що представлена комплексом технічних і програмних засобів, об'єднаних загальним управлінням і призначених для введення інформації, її переробки за заданим алгоритмом програми і виведення результатів для користувача або інших пристроїв.

При роботі комп'ютер виконує перетворення початкової інформації, при цьому вигляд інформації (тобто як вона надана, закодована) визначає велику роль у формуванні алгоритмів арифметичних і логічних операцій.

Інформація є однією з основних філософських категорій і понять нашого миру (поруч з такими поняттями як матерія, енергія і ін.). Кібернетика, як наука про автоматизацію процесів управління базується на теоретичних основах інформації.

У словнику С.І. Ожегова наведено: інформація – відомості про навколишній світ і процеси, що протікають в ньому, сприймані людиною або спеціальними пристроями.

Комітет науково-технічної термінології Академії наук дає наступне визначення: інформація – це відомості, зберігання, що є об'єктом передачі і

перетворення [1-21]. Можна зупинитися на такому визначенні: інформація – це відомості про яку-небудь подію або предмет, що поступає до одержувача ззовні в результаті його взаємодії з навколишнім середовищем [1-21].

Як усяка категорія інформація характеризується наступними властивостями [1-21]:

- інформація приносить знання про навколишній світ, яких в даному місці не було до її отримання;
- інформація не матеріальна, але вона виявляється у формі матеріальних носіїв (дискретних знаків, символів, сигналів і ін.);
- інформація може бути поміщена як в знаках, як таких, так і в їх взаємному розташуванні (наприклад: знаки у вигляді літер «Т, Р, С, О» можуть принести інформацію у вигляді різних слів: сорт, торс, тор, трос, ін.);
- знаки і сигнали несуть інформацію тільки для одержувача, що здатен розпізнати їх.

Знаками назвемо реальні помітні одержувачем матеріальні об'єкти: літери, цифри, символи, предмети, умовні рухи, їх комбінації [1-21].**Сигналами** називатимемо динамічні процеси будь-якої природи, що змінюються в часі (зміна значення напруги, тиску, електромагнітного поля і ін.). Поняття «знак» і «сигнал» часто взаємозамінні, проте знаки частіше використовують для зберігання інформації, а сигнали для передачі повідомлень з однієї точки простору в іншу або перетворення інформації з однієї форми в іншу.

Із знаків і сигналів будуються **повідомлення**. Елементарним повідомленням є кожний із знаків або сигналів. Послідовність елементарних повідомлень несуть одержувачеві інформацію. Кінцеву кількість усіх знаків або сигналів називають ще **алфавітом.** При передачі повідомлень знаки перетворюють на сигнали, що однозначно зіставляються – **«коди».** Процес зіставлення називають – **кодуванням.** Коди бувають різними, але кожен відповідає своїм правилам (алгоритмам) перетворення. Обернений процес називають **декодуванням**.

Розрізняють інформацію безперервну і дискретну.

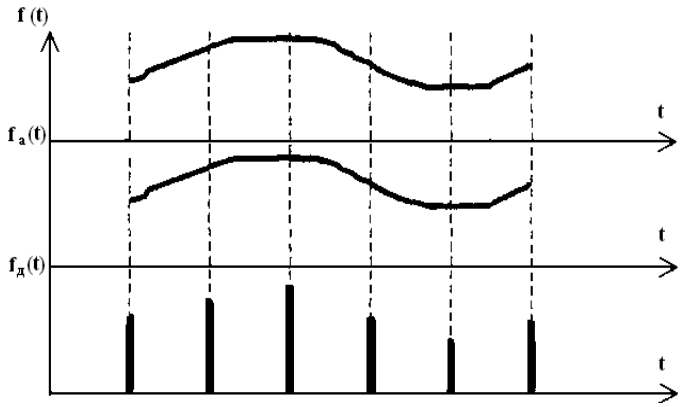


Рис. 1.1 – Інформація функції f(t) представлена в безперервному (аналоговому) і дискретному вигляді

У аналоговому уявленні функція має значення параметра в будь-якій точці аргументу *t*. У дискретному вигляді функція *f*(*t*) визначена лише на конкретних значеннях *t*.

Вид інформації, що переробляється, впливає на структуру ОМ і принцип їх дії. Як наголошувалося раніше, ОМ ділять на три основні класи: аналогові, цифрові, гібридні.

Аналогова ОМ – та, що оперує інформацією, представленою у вигляді безперервних змін деяких фізичних величин (струм, напруга, опір і ін.). Багато процесів в природі можна моделювати аналогічно при однакових математичних моделях їх опису.

Цифрова ОМ – та, що оперує інформацією, представленою в дискретному вигляді. Вона більш універсальна оскільки методи чисельного рішення дають можливість рішення будь-яких математичних і логічних задач.

Останніми роками при розробці деяких ОМ вчені суміщають позитивні сторони ЦОМ і АОМ. Такі машини отримали назву гібридних ОМ.

В літературі можна віднайти багато тлумачень терміну «Інформація», що відображує різні аспекти розуміння цього поняття. Під інформацією Закон України про інформацію N 2658-XII від 02.10.92 розуміє документовані або публічно оголошені відомості про події та явища, що відбуваються у суспільстві, державі та навколишньому природному середовищі.

Інформація та її властивості є об'єктом дослідження цілого ряду наукових дисциплін.

В інформатиці використовується таке визначення цього терміну:

Інформація – це усвідомленні відомості про оточуючий світ, які є об'єктом для збереження, перетворення, передачі та використання.

Відомості – це знання, виражені в сигналах, повідомленнях, вістях, звістках і так далі. Кожну людину в світі оточує велика кількість різновидів інформації.

Прагнення зафіксувати, зберегти надовго своє сприйняття інформації було завжди властиве людині. Мозок людини зберігає безліч інформації і використовує для зберігання її свої способи, основа яких – двійковий код, як і в комп'ютерів. Людина завжди прагнула мати можливість поділитися своєю інформацією з іншими людьми і знайти надійні засоби для її передачі і довготривалого зберігання. Для цього в даний час винайдена безліч способів зберігання інформації на зовнішніх (відносно мозку людини) носіях і її передачі на величезні відстані.

Основні види інформації по її формі вистави, способам її кодування і зберігання, що має найбільше значення для інформатики, це:

- графічна або образотворча – перший вигляд, для якого був реалізований спосіб зберігання інформації про навколишній світ у вигляді наскальних малюнків, а пізніше у вигляді картин, фотографій, схем, креслень на папері, полотні, мармурі і ін. матеріалах, що змальовують картини реального світу;
- звукова – світ довкола нас повний звуків і проблему їх зберігання і тиражування було вирішене з винахід звукозаписних пристроїв в 1877 р.;

її різновидом є музична інформація – для цього вигляду був винайдений спосіб кодування з використанням спеціальних символів, що зробило можливим зберігати її аналогічно графічній інформації;
- текстова – спосіб кодування мови людини спеціальними символами – буквами чи ієрогліфами, причому різні народи мають різні мови і використовують різні набори для відображення мови; особливо велике значення цей спосіб придбав після винаходу паперу і книгодрукування;
- числова – кількісна міра об'єктів і їх властивостей на навколишньому світі; особливо велике значення набула з розвитком торгівлі, економіки і грошового обміну; аналогічно текстовій інформації для її відображення використовується метод кодування спеціальними символами – цифрами, причому системи кодування (числення) можуть бути різними;
- відеоінформація – спосіб збереження «живих» картин навколишнього світу, що з'явився з винайденням кіно.

Існують також види інформації, для яких ще й донині не винайдено способів їх кодування і зберігання, – це тактильна інформація, передавана відчуттями, органолептична, передавана запахами і смаками і інші види, для яких сучасна наука навіть не знайшла визнаних всіма термінів визначення.

Для передачі інформації на великі відстані спочатку використовувалися кодовані світлові сигнали, з винаходом електрики – передача закодованого певним чином сигналу по дротах, а пізніше – з використанням радіохвиль.

Творцем загальної теорії інформації і основоположником цифрового зв'язку вважається Клод Шеннон (Claude Shannon). Всесвітню популярність йому принесла фундаментальна праця 1948 року – «Математична теорія зв'язку» (A Mathematical Theory of Communication), в якій вперше обгрунтовується можливість використовувати двійкові коди для передачі інформації.

З появою комп'ютерів спочатку з'явився засіб для обробки числової інформації. Проте надалі, особливо після широкого поширення персональних комп'ютерів, комп'ютери стали використовуватися для зберігання, обробки, передачі і пошуку текстової, числової, графічної, звукової і відеоінформації. З моменту появи перших персональних комп'ютерів – ПК (80-і роки 20 століття) до 80% їх робочого часу присвячувалось роботі з текстовою інформацією.

Зберігання інформації при використанні комп'ютерів здійснюється на магнітних дисках або стрічках, на лазерних дисках (CD і DVD), спеціальних пристроях незалежної пам'яті (флеш-пам'ять і ін.). Ці методи постійно удосконалюються, винаходяться нові пристрої і носії інформації. Обробку інформації (відтворення, перетворення, передачу, запис) виконує процесор комп'ютера. За допомогою комп'ютера можливе створення і зберігання нової інформації будь-яких видів, для чого використовують спеціальні програми.

До особливого вигляду інформації в даний час можна віднести інформацію, представлену в глобальній мережі Інтернет. Тут використовуються особливі прийоми зберігання, обробки, пошуку і передачі розподіленої інформації великих об'ємів і особливі способи роботи з різними видами інформації.

Постійно удосконалюється програмне забезпечення, що забезпечує колективну роботу з інформацією всіх видів.

Поняття «інформація» зазвичай передбачає наявність двох об'єктів – «джерела» інформації і «приймача» (споживача, адресата) інформації.

Інформація передається від джерела до приймача в матеріально-енергетичній формі у вигляді сигналів (наприклад, електричних, світлових, звукових і т. Д.), Що розповсюджуються в певному середовищі.

Сигнал (від лат. Signum - знак) – фізичний процес (явище), що несе повідомлення (інформацію) про подію або стан об'єкта спостереження.

Інформація може надходити в аналоговому (безперервному) вигляді або дискретно (у вигляді послідовності окремих сигналів). Відповідно розрізняють аналогову і дискретну інформацію.

Поняття інформації можна розглядати з двох позицій: у широкому сенсі слова – це навколишній світ, обмін відомостями між людьми, обмін сигналами між живою і неживою природою, людьми і пристроями; у вузькому сенсі слова інформація – це будь-які відомості, які можна зберегти, перетворити і передати.

Інформація – специфічний атрибут реального світу, що представляє собою його об'єктивне відображення у вигляді сукупності сигналів і що виявляється при взаємодії з «приймачем» інформації, що дозволяє виділяти, реєструвати ці сигнали з навколишнього світу і по тому чи іншому критерію їх ідентифікувати.

З цього визначення випливає, що:

- інформація об'єктивна, оскільки це властивість матерії - відображення;
- інформація проявляється у вигляді сигналів і лише при взаємодії об'єктів;
- одна і та ж інформація різними одержувачами може бути інтерпретована по-різному в залежності від «настройки» «приймача».

Людина сприймає сигнали за допомогою органів почуттів, які «ідентифікуються» мозком. Приймачі інформації в техніці сприймають сигнали за допомогою різної вимірювальної та реєструючої апаратури. При цьому приймач, що володіє більшою чутливістю при реєстрації сигналів і більш досконалими алгоритмами їх обробки, дозволяє отримати великі обсяги інформації.

Інформація має певні функції. Основними з них є:

• пізнавальна - отримання нової інформації. Функція реалізується в основному через такі етапи обігу інформації, як:

- її синтез (виробництво);
- представлення;
- зберігання (передача в часі);
- сприйняття (споживання).

• комунікативна - функція спілкування людей, що реалізується через такі етапи обігу інформації, як:

– передача (в просторі);
– розподіл.

• управлінська – формування доцільного поведінки керованої системи, яка одержує інформацію. Ця функція інформації нерозривно пов'язана з пізнавальною і комунікативною і реалізується через всі основні етапи обігу, включаючи обробку.

Без інформації не може існувати життя в будь-якій формі і не можуть функціонувати будь-які інформаційні системи, створені людиною. Без неї біологічні та технічні системи представляють купу хімічних елементів. Спілкування, комунікації, обмін інформацією властиві всім живим істотам, але в особливій мірі людині. Будучи акумульованої і обробленої з певних позицій, інформація дає нові відомості, приводить до нового знання. Отримання інформації з навколишнього світу, її аналіз і генерування складають одну з основних функцій людини, яка відрізняє його від решти живого світу.

У загальному випадку роль інформації може обмежуватися емоційним впливом на людину, однак найбільш часто вона використовується для вироблення керуючих впливів в автоматичних (чисто технічних) і автоматизованих (людино-машинних) системах. У подібних системах можна виділити окремі етапи (фази) обігу інформації, кожен з яких характеризується певними діями.

Послідовність дій, виконуваних з інформацією, називають інформаційним процесом.

Основними інформаційними процесами є:

– збір (сприйняття) інформації;
– підготовка (перетворення) інформації;
– передача інформації;
– обробка (перетворення) інформації;
– зберігання інформації;
– відображення (відтворення) інформації.

На етапі сприйняття інформації здійснюється цілеспрямоване вилучення та аналіз інформації про якомусь об'єкті (процесі), в результаті чого формується образ об'єкта, проводяться його впізнання і оцінка. Головна задача на цьому етапі – відокремити корисну інформацію від заважає (шумів), що в ряді випадків пов'язано зі значними труднощами.

На етапі підготовки інформації здійснюється її первинне перетворення. На цьому етапі проводяться такі операції, як нормалізація, аналого-цифрове перетворення, шифрування. Іноді етап підготовки розглядається як допоміжний на етапі сприйняття. В результаті сприйняття і підготовки виходить сигнал у формі, зручній для передачі, зберігання або обробки.

На етапі передачі інформація пересилається з одного місця в інше (від відправника одержувачу - адресату). Передача здійснюється по каналам різної фізичної природи, найпоширенішими з яких є електричні, електромагнітні та

оптичні. Витяг сигналу на виході каналу, схильності дії шумів, носить характер вторинного сприйняття.

На етапах обробки інформації виявляються її загальні та суттєві взаємозалежності, що представляють інтерес для системи. Перетворення інформації на етапі обробки (як і на інших етапах) здійснюється або засобами інформаційної техніки, або людиною.

Під обробкою інформації розуміється будь-яке її перетворення, що проводиться за законами логіки, математики, а також неформальним правилам, заснованим на «здоровому глузді», інтуїції, узагальненому досвіді, що склалися поглядах і нормах поведінки. Результатом обробки є теж інформація, але або представлена в інших формах (наприклад, упорядкована по якихось ознаках), або яка містить відповіді на поставлені запитання (наприклад, рішення деякої задачі). Якщо процес обробки формалізуємо, він може виконуватися технічними засобами. Кардинальні зрушення в цій області відбулися завдяки створенню ЕОМ як універсального перетворювача інформації, у зв'язку з чим з'явилися поняття даних та обробки даних.

Даними називають факти, відомості, представлені в формалізованому вигляді (закодовані), занесені на ті чи інші носії і допускають обробку за допомогою спеціальних технічних засобів (в першу чергу ЕОМ).

Обробка даних передбачає виробництво різних операцій над ними, в першу чергу арифметичних і логічних, для отримання нових даних, які об'єктивно необхідні (наприклад, при підготовці відповідальних рішень).

На етапі зберігання інформацію записують в запам'ятовуючий пристрій для подальшого використання. Для зберігання інформації використовуються в основному напівпровідникові і магнітні носії.

Етап відображення інформації повинен передувати етапам, пов'язаним з участю людини. Мета цього етапу – надати людині потрібну йому інформацію за допомогою пристроїв, здатних впливати на його органи чуття.

Будь-яка інформація має ряд властивостей, які в сукупності визначають ступінь її відповідності потребам користувача (якість інформації). Можна навести чимало різноманітних властивостей інформації, так як кожна наукова дисципліна розглядає ті властивості, які їй найбільш важливі. З точки зору інформатики найбільш важливими представляються наступні:

Актуальність інформації – властивість інформації зберігати цінність для споживача протягом часу, т. Е. Не наражатися «моральному» старінню.

Повнота інформації – властивість інформації, що характеризується мірою достатності для вирішення певних завдань. Повнота інформації означає, що вона забезпечує прийняття правильного (оптимального) рішення. Оцінюється щодо цілком певної задачі або групи завдань.

Адекватність інформації – властивість, що полягає у відповідності змістовної інформації станом об'єкта. Порушення ідентичності пов'язано з технічним старінням інформації, при якому відбувається розбіжність реальних ознак об'єктів і тих же ознак, відображених в інформації.

Збереження інформації – властивість інформації, що характеризується ступенем готовності визначених інформаційних масивів до цільовому застосуванню і яке визначається здатністю контролю та захисту інформації забезпечити постійну наявність і своєчасне надання інформаційного масиву, необхідних для автоматизованого вирішення цільових і функціональних завдань системи.

Достовірність інформації – властивість інформації, що характеризується ступенем відповідності реальних інформаційних одиниць їх істинного значення. Необхідний рівень достовірності інформації досягається шляхом впровадження методів контролю та захисту інформації на всіх стадіях її переробки, підвищення надійності комплексу технічних і програмних засобів інформаційної системи, а також адміністративно-організаційними заходами.

Контрольні питання:

1. Що вивчає дисципліна «ПТЦА»?
2. Як розвивалися способи збору, зберігання та передачі інформації?
3. Що таке інформація?
4. Які функції виконує інформація?
5. Дайте характеристику основним інформаційним процесам.
6. У чому основна відмінність даних від інформації?
7. Якими властивостями володіє інформація?
8. Що розуміється під інформатизацією суспільства?
9. Якими характерними рисами володіє інформаційне суспільство?
10. Що таке системи числення і які вони бувають? Наведіть приклади.
11. Дайте характеристику основним позиційним системам числення.
12. У яких двох видах може бути представлена інформація? Охарактеризуйте їх і наведіть приклади.
13. Що таке кодування? Наведіть приклади кодування з життя.
14. Що є основною одиницею подання інформації в ЕОМ?
15. Як кодуються різні види інформації в ЕОМ?
16. За допомогою яких одиниць вимірюють інформацію?

ТЕМА 2. ПОНЯТТЯ ПРО ЦИФРОВІ АВТОМАТИ

План

1. Роль і місце ЕОМ в системах управління.
2. Апаратні засоби зберігання і обробки інформації.
3. Поняття про цифровий автомат і алгоритм.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Роль і місце ЕОМ в системах управління

Застосування ЕОМ при моделюванні процесів мислення дало змогу використовувати її для рішення задач, котрі раніше вирішувала людина (переводи, ігри, доведення теорій тощо), тобто стало можливим відтворення певних сторін розумової діяльності людини. ЕОМ розширює можливості реалізації вищих форм розумової діяльності (перевірка гіпотез, розумове експериментування), що сприяє психічному розвиткові людини.

ЕОМ значно розширює можливості реалізації когнітивної функції, оскільки:

- полегшує інформаційно-довідкову діяльність користувача, накопичуючи, зберігаючи і відтворюючи значну кількість знань;
- сприяє інформаційній підготовці й оцінці проміжних рішень шляхом порівняння різних варіантів досягнення мети, проведення шаблонних перетворень просторових і часових ознак об’єкта, розглядання його у різних площинах;
- стимулює формулювання різних гіпотез, розгортаючи перед оператором цілий спектр методичних засобів, що забезпечує інформаційну підтримку і підказування різних шляхів пошуку остаточного рішення;
- вивільняє оператора від проведення трудомістких розрахунків у процесів знаходження нових закономірностей.

ЕОМ відіграє значну роль у підтримці регулятивної функції психіки людини, оскільки:

- сприяє більш раціональному плануванню діяльності , розкладаючи задачу на ряд підзадач за певними критеріями;
- контролює і коригує послідовність керуючих дій оператора відповідно до прийнятого плану;
- застосовує оперативний контроль за введенням та виведенням

інформації з ЕОМ;
- реалізує різні форми організації індивідуальної і групової діяльності;
- прискорює випуск нормативно-технічної документації об’єкта на різних носіях.

ЕОМ у реалізації комунікативної функції психіки забезпечує:
- реалізацію певних техніко-економічних відносин у системі;
- підтримку необхідного рівня і темпу взаємодії операторів;
- обмін інформацією про проміжні результати діяльності партнерів;
- збір інформації про оператора, формуючи його психологічний портрет з метою подальшого комплектування малих груп.

Негативні наслідки автоматизації діяльності оператора:
- трудомісткість розробки програмного забезпечення;
- вплив стандартного мислення на розвиток творчої компоненти в діяльності;
- анімізація (використання понять опису живого світу для опису роботи технічних засобів);
- персоніфікація (перенесення властивостей або рис людини на технічні об’єкти).

2. Апаратні засоби зберігання і обробки інформації

Система програмно-апаратних засобів обробки інформації ЕОМ (комп'ютери) в даний час є основними засобами реалізації інформаційних технологій. За допомогою ЕОМ здійснюють збір, обробку, зберігання та передачу інформації: представленої в дискретної формі.

Послідовності обробки даних на ЕОМ задаються користувача програмами. Програми цього типу були єдиними для перших ЕОМ. Але розвиток ЕОМ і досвід їх експлуатації виявили корисність застосування програм іншого типу, призначених для підвищення "інтелектуальних" здібностей ЕОМ і спрощують управління компонентами ЕОМ. Це системні програми, що поставляються виготовлювачами, як компоненти ЕОМ.

Структурно ЕОМ представляють собою взаємодіючу сукупність двох компонент: системи апаратних засобів (hardware) і системи програмного забезпечення (software). Зазначені системи, в певних рамках, незалежні, так що в ЕОМ можна окремо модернізувати (з деякими обмеженнями), як апаратні засоби, так і програмне забезпечення.

Для передбаченого функціонування ЕОМ в ній повинні протікати певні дії (процеси). Управління цими процесами може бути реалізовано або апаратними або програмними засобами.

Як правило, для управління ЕОМ використовуються ієрархічна система апаратно-програмних рішень. При цьому на самому нижньому рівні управління використовуються апаратні засоби, на верхніх рівнях - програмні.

Основою системи програмного забезпечення є Операційна система (ОС). ОС може нарощуватися програмними інструментальними засобами, сервісними програмами (утилітами) та іншими системними програмами.

Апаратні рішення, як правило, призводять до зменшення часових витрат, але програмні рішення більш дешеві. Тому в еволюції сучасних ЕОМ розвиток програмних засобів випереджає розвиток апаратних і сумарна частка вартості розробки програмних засобів значно переважає частку вартості розробки апаратних засобів.

Такі пристрої називаються накопичувачами. В основі їхньої роботи лежать різні принципи (в основному це магнітні або оптичні пристрої), але використовуються вони для однієї мети - для зберігання інформації для подальшого багаторазового використання. Цей вид пам'яті, на відміну від оперативної пам'яті, є енергонезалежною. Обсяг носіїв, використовуваних в цих пристроях, значно перевершує обсяг оперативної пам'яті. Вартість зберігання одиниці інформації істотно нижче. Накопичувачі бувають зовнішніми (власний корпус і джерело живлення) і внутрішніми (вбудовуються в корпус комп'ютера), зі змінними і незмінними носіями, з носіями різної форми (диски, стрічки). Накопичувачі мають різні характеристики: максимально можливий обсяг інформації, що зберігається, час доступу. Важливе значення має вартість зберігання інформації. Для інтеграції накопичувачів в комп'ютер розроблені спеціальні інтерфейси.

Інтерфейси накопичувачів

В даний час переважають два інтерфейсу: IDE і SCSI. Велике поширення отримав інтерфейс USB.

SCSI

Інтерфейс SCSI (Small Computer System Interface) був розроблений ще в 1970 році. До шині можна підключити до восьми пристроїв, включаючи основний контролер SCSI. Контролер SCSI має власну BIOS, яка управляє шиною SCSI, звільняючи центральний процесор. Шина SCSI - 8-розрядна, тактова частота - 5 МГц, має вісім ліній даних, лінію парності, дев'ять керуючих ліній. Максимальна швидкість передачі даних не перевищує 5 Мбіт / с. Інтерфейс SCSI має велику чутливість до якості виготовлення кабелів. Інтерфейс SCSI-2 (Fast SCSI) - розвиток стандарту, в якому тактова частота і швидкість передачі даних збільшені в два рази. Додані і стандартизовані команди доступу до периферійних пристроїв.

В стандарт SCSI-2 додана специфікація Wide (широка), в якій кількість інформаційних ліній збільшено до 24. До шини можна підключати як 8-розрядні, так і 16-розрядні пристрої.

У модифікації Ultra SCSI тактова частота - 20 МГц (інші назви SCS1-3, SCSI20). Додані команди для деяких графічних пристроїв. Ultra Wide SCSI забезпечує 16-розрядну передачу даних зі швидкістю до 40 Мбіт / с. В Ultra 2 Wide SCSI тактова частота збільшена до 40 МГц, швидкість передачі даних - 40 Мбіт / с. Існує Wide-специфікація цього стандарту, за якою підтримується до 15 пристроїв.

IDE

Інтерфейс IDE (Integrated Drive Electronics) був запропонований в 1988 році як недорога альтернатива використовуваним в той час інтерфейсам. Відмінною рисою IDE є реалізація функцій контролера в електронній частині пристрою. Друга назва інтерфейсу - ATA (AT Attachment). В якості пристроїв, що підключаються могли бути тільки накопичувачі на незнімних жорстких дисках. Інтерфейс підтримує PIO (Programmed Input / Output), відповідно до якого обмін даними здійснюється через центральний процесор. Другий спосіб обміну - режим DMA (Direct Memory Access), коли обмін даними з оперативною пам'яттю здійснюється безпосередньо, без участі центрального процесора. IDE одночасно може підтримувати до двох пристроїв, режими PIO Mode 1 і 2, DMA. Максимальна швидкість передачі даних 8,3 Мбіт / с.

Розвитком стандарту став інтерфейс EIDE (Enhanced IDE), який підтримує пристрої об'ємом понад 504 Мбайт, і не тільки пристрої з жорсткими дисками (така специфікація інтерфейсу називається ATAPI - ATA Packed Interface). Можливе підключення до чотирьох пристроїв. У системі можна встановлювати декілька контролерів EIDE. Інтерфейс підтримує режими PIO Mode 3 і 4, DMA Single Word Mode 2 і Multi Word Mode 1 і 2 Істотне збільшення пропускної здатності шини дає підтримка режиму Bus Mastering DMA. Максимальна швидкість передачі даних - до 16,7 Мбіт / с.

Інтерфейс ATA-3 розроблений для підвищення надійності передачі даних і підвищення її продуктивності. Стандарт підтримує технологію попередження відмов жорстких дисків SMART (Self-Monitoring Analysis and Reporting Technology).

Інтерфейс Ultra ATA підтримує протокол UltraDMA, за яким за один такт передається в два рази більше інформації, ніж у попередніх версіях. Випускаються накопичувачі, що використовують, крім розглянутих стандартів, стандарти USB, PCMCIA, FireWire і ін.

Накопичувачі на стрічках

Накопичувачі на магнітних стрічках називаються стримерами. Стримери використовуються, коли необхідно записати великі обсяги інформації при створенні архівних копій. Сучасні стримери використовують спеціальні касети (картриджі) з магнітною стрічкою. Стримери, як правило, мають власні засоби стиснення даних. Стримери мають рівні стандарти, що визначають інтерфейс з комп'ютером, формат магнітної стрічки, методи кодування і стиснення. QUC-стримери (Quarter Inch Cartridge) мають товщину картриджа 1/4 ". Ємність картриджів різна - від 250 Мбайт до 1,3 Гбайт. Travan-стримери використовують магнітні стрічки шириною 0,315". Ємність картриджів - від 400 Мбайт до 4 Гбайт. В DAT-стримерах (Digital Audio Tape) використовується технологія спірального сканування, що дозволяє збільшити щільність запису. Ємність картриджа досягає 8 Гбайт. DLT-стримери (Digital Linear Tape) мають високу надійність в експлуатації, дозволяють записувати інформацію з високою швидкістю. Ємність картриджа до 35 Гбайт.

Накопичувачі на дисках

Відмінною особливістю таких пристроїв є використання як носіїв інформації круглих дисків різного діаметру, що відрізняються форм-фактором. Випускаються носії з форм-фактором (розміром) 1,8 ", 2,5", 3,5 ", 5,25".

Накопичувачі на жорстких незнімних дисках

Ці накопичувачі називаються вінчестерами. Вони являють собою систему, що складається з механічного приводу, головок читання / запису, декількох носіїв і контролера, що забезпечує роботу всього пристрою і передачу даних. Магнітна головка (кілька магнітних головок в спеціальному позиціонері) є однією з найбільш важливих частин пристрою. Конструкція магнітних головок постійно вдосконалюється. Розрізняють такі типи: монолітні, виготовлені з феритів, композиційні, що складаються з декількох видів матеріалів (скло, сплави, кераміка), тонкоплівкові, що виготовляються методом фотолітографії, магніторезистивні, які з двох - для запису і для читання. Носій інформації складається з декількох дисків, кожен з яких має дві робочих поверхні. При запису інформації використовуються магнітні властивості шару, нанесеного на поверхню. Диски закріплені на шпинделі двигуна. Швидкість обертання дисків може бути 3600, 4500, 5400, 7200, 10000, 12000 об / хв. Із збільшенням швидкості обертання дисків збільшується продуктивність всієї системи. Геометричні розміри пристрою визначаються форм-фактором і розміром по висоті: Full Height (FH) - повна висота 3,25 ", Half-Height (HH) - половинна висота або Low-Profile (LP) - низький профіль 1".

Кожна поверхня будь-якого з дисків розбивається на окремі доріжки. Доріжки на одній вертикалі на всіх поверхнях утворюють циліндр. Доріжка розбивається на сектори. Доступ до необхідної інформації здійснюється за номером доріжки номеру циліндра, номеру сектора.

Щільність запису на зовнішніх секторах менше, ніж на внутрішніх секторах. У сучасних вінчестерах форм-фактора 3,5 ", 5,25" диск розбивають на зони, в межах яких кількість секторів постійно. Чим зона далі від центру, тим більше вона містить секторів.

Серед основних параметрів, що визначають продуктивність вінчестера, можна виділити наступні: середній час доступу, яке визначається часом позиціонування магнітних головок на доріжці і часом очікування сектора, і швидкість обміну даними, яка в основному залежить від використовуваного інтерфейсу.

Накопичувачі на змінних дисках

Приводи для floppy-дисків з'явилися вже на перших персональних комп'ютерах. Форм-фактор цих дисків 5,25 ". Ємність одного диска становила 160 Кбайт. Згодом ємність одного дискового носія збільшилася до 1,2 Мбайт. Наступний етап - форм-фактор 3,5". Ємність одного носія 720 Кбайт, 1,44 Кбайт. Операції читання / запису здійснюються контактним способом, коли магнітна головка пристрою стикається з поверхнею носія. У таких пристроїв невисока щільність запису, швидкість обміну, значний час доступу. Сучасні розробки floppy-пристроїв дозволяють поліпшити показники подібних

пристроїв, використовуючи гнучкі диски форм-фактора 3,5 ", на яких можна зберігати до 100-200 Мбайт інформації. Деякі накопичувачі можуть працювати з звичайними floppy-дисками, деякі - ні. Для збільшення щільності записи використовуються сучасні технології, такі як застосування лазерного променя для точного позиціонування магнітної головки пристрою, ефекту Бернуллі для безконтактного способу запису / читання.

MOD (Magneto-Optical Disk) - магнітооптичні диски мають різну ємність від 128 Мбайт до 640 Мбайт. Запис проводиться магнітним способом після нагрівання лазером магнітного шару до певної температурної точки (точка Кюрі). Надійність зберігання інформації забезпечується тим, що при звичайній температурі інформацію не схильна до дії зовнішніх магнітних полів. Пристрої CD-ROM використовують носії, ємністю до 650 Мбайт. Носій являє собою диск зі светоотражающим шаром на одній стороні, на якій зберігається інформація. На диск нанесена доріжка спіраль від центру до краю диска, що складається з відображають і не відображають світло точок. Зчитування проводиться лазерним променем. Швидкість зчитування інформації визначається в порівнянні зі стандартом Audio CD - 150 Кбайт / с.

Низькошвидкісні приводи використовували лінійну швидкість зчитування інформації. При цьому кутова швидкість збільшувалася при читанні на зовнішній стороні диска. В високошвидкісних приводах стали використовувати постійну кутову швидкість. В маркуваннях таких приводів вказується максимально досяжна швидкість зчитування інформації. Швидкість читання неповного диска ніколи не досягне максимального значення.

Накопичувачі CD-R з можливістю запису дозволяють одноразово записувати інформацію на диски діаметром 120 і 80 мм. Промінь лазера пропалює плівку на поверхні диска, змінюючи його відбивну здатність. Перезапис неможливий. Такі диски читаються на будь-якому приводі CD-ROM.

Накопичувачі CD-RW дозволяють робити багаторазову запис на диск. Тут використовується властивість робочого шару переходити під дією лазерного променя в кристалічний або аморфний стан, що мають різну відбивну здатність. Такі диски можуть не читатися на деяких, особливо застарілих приводах CD-ROM.

Накопичувачі DVD (Digital Versatile Disc) - цифровий універсальний диск. Призначений для зберігання відео, аудіо високої якості, комп'ютерної інформації великого обсягу. На рис. 3.14 зображено пристрій DVD. Односторонні одношарові DVD мають ємність 4,7 Гбайт інформації, двошарові - 8,5 Гбайт; двосторонні одношарові вміщають 9,4 Гбайт, двошарові - 17 Гбайт. Щільність запису вище, ніж у звичайних CD-ROM. Накопичувачі DVD можуть читати звичайні CD-ROM-диски. Двошвидкісні накопичувачі DVD можуть читати і CD-R, і CD-RW-диски.

Накопичувачі DVD-RAM дозволяють записувати і перезаписувати інформацію. На односторонньому одношаровому диску можна розмістити 2,58 Гбайт даних, на двосторонньому - 5,2 Гбайт. Конкуруючий стандарт DVD-R дозволяє зберігати 3,95 Гбайт інформації.

Накопичувачі па змінних жорстких дисках використовують технологію вінчестерів. Параметри таких пристроїв наближаються до параметрів пристроїв з жорсткими незнімними дисками. Як правило, використовується форм-фактор 3,5". Ємність одного носія варіюється від 230 Мбайт до 2 Гбайт.

RAID

RAID (Redundant Array of Inexpensive Disks) масив - пристрій, що складається з декількох вінчестерів і RAID-контролера. Такий пристрій має великий обсяг дискового простору, підвищеною швидкістю обміну даними, значною надійністю зберігання інформації. RAID level 0 забезпечує високу швидкість обміну даними за рахунок паралельного запису на декілька дисків. Швидкість обміну знаходиться в прямій пропорції від кількості використовуваних вінчестерів. Надійність такого RAID-масиву невелика. RAID level 1 забезпечує надійність зберігання інформації за рахунок дублювання інформації на декілька дисків. RAID level 3 і 5 є проміжними варіантами, де надійність забезпечується не простим дублюванням інформації, а додатковою надлишковою інформацією, необхідною для відновлення інформації при збоях системи.

RAID-масиви допускають заміну вінчестера без відключення живлення і зупину комп'ютерної системи, без втрат інформації.

3. Поняття про ЦА і алгоритм

Ми розглядатимемо арифметичні і логічні основи ЕОМ з погляду теорії **цифрових автоматів**. Відомий математик Джон фон Нейман у 1945р. у своїй доповіді описав, як повинен бути влаштований комп’ютер. У спрощеному вигляді, він повинен мати наступну структуру (рис 2.1).

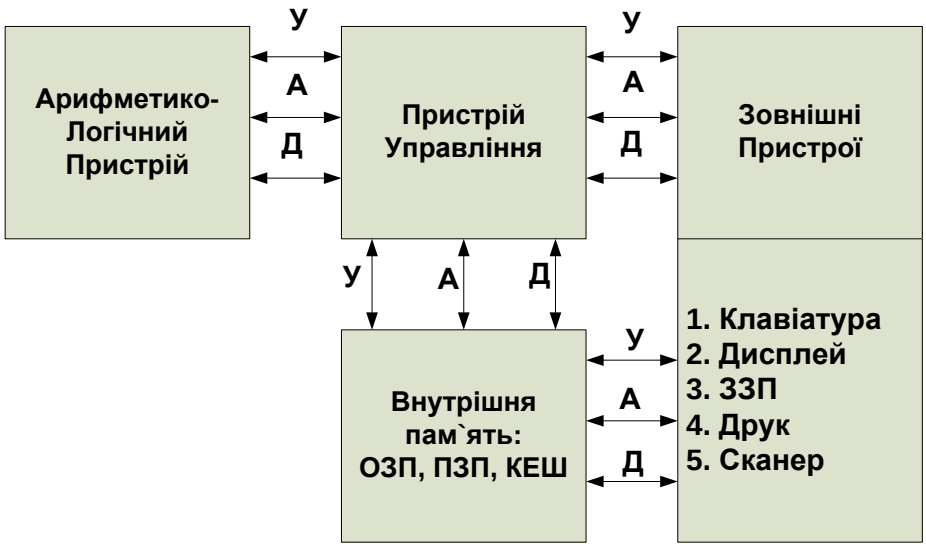


Рис. 2.1 – Структура ЕОМ

В неї входять: арифметичний логічний пристрій (АЛП); пристрій керування (ПК), який організує процес виконання програм; внутрішні пристрої, що запам'ятовують інформацію (оперативний запам'ятовуючий пристрій – ОЗП, постійний запам'ятовуючий пристрій – ПЗП), для зберігання даних, проміжних рішень, програм та інше; зовнішні пристрої для введення/виведення інформації і її тривалого зберігання, архівації – зовнішній запам'ятовуючий пристрій (ЗЗП).

Нас цікавлять теоретичні основи методів виконання арифметичних і логічних операцій АЛП. Загалом, будь-який пристрій ЕОМ можна представити як цифровий мікропрограмний автомат.

Під цифровим автоматом (ЦА) розуміють пристрій, що оперує з цифровою дискретною інформацією і характеризується кінцевою множиною внутрішніх станів $Z=\{z_0,z_1,z_2,\ldots,z_n\}$, в які він переходить під впливом множин сигналів: вхідних $X=\{x_1,x_2,\ldots,x_k\}$ і вихідних $Y=\{y_1,y_2,\ldots,y_m\}$, при цьому є кінцева множина правил переходу з одного стану в інший: $W=\{w_1,w_2,\ldots,w_L\}$ - функція переходів ЦА із стану Z_{i-1} у Z_i по правилам формування Z_i, станів ЦА; а також функція $\lambda=\{\lambda_1,\lambda_2\ldots,\lambda_j\}$ формування Y- виходів ЦА, що в загальному випадку може бути пов'язана з його станом Z_i і вхідними сигналами $Y=\lambda[Z_i, X_i]$). Всі множини є кінцеві, звідси кінцевий ЦА.

Математичною моделлю ЦА є деякий абстрактний автомат (рис.2.2), заданий таким чином: у початковий момент t_0 автомат знаходиться в стані Z_0. Під впливом вхідного коду X з множини вхідних слів, автомат переходить зі стану Z_0 в Z_1, при цьому може виникнути вихідне слово $Y=\lambda[Z_i,X_i]$, а перехід $Z_i=W[Z_0,X_i]$ здійснюється функцією переходів W, дії вхідного коду X і попереднього стану $Z_{0\ (\text{або}\ n)}$ і так далі (алгоритм перетворення).

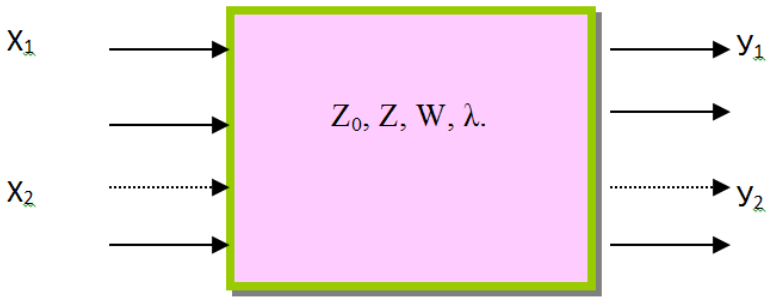


Рис. 2.2 – Модель абстрактного автомата

У загальному випадку математична модель абстрактного автомата записується як кортеж (сукупність): $A=\{X, Z_0, Z, W, \lambda, Y\}$.

Алгоритм – кінцева послідовність точно сформульованих правил рішення якоїсь задачі (узбецький математик Аль-Хорезмі в IX столітті сформулював правила арифметичних дій). Можуть бути словесними, математичними, заданий програмою, ін.. Бувають детерміновані алгоритми, випадкові алгоритми, чисельні алгоритми, логічні алгоритми.

Арифметико-логічний пристрій (АЛП) – функціональний пристрій ЕОМ, що виконує арифметичні і логічні дії, необхідні для обчислення або виконання логічних операцій над заданою в ОЗП інформацією відповідно до встановлених алгоритмів.

АЛП характеризується:

- часом виконання елементарних операцій або середньою швидкодією, тобто кількістю арифметичних або логічних операцій виконаних в одиницю часу (секунду);
- набором арифметичних дій і логічних команд, які він виконує;
- видом арифметичного базису;
- тактовою частотою (часова сітка) виконання операцій, її швидкодія.

Наприклад, частота 100 МГц позначає, що виконується 100 мл. тактів в секунду (якщо операція виконується за два такти, то швидкодія буде 50 мл. операцій в секунду).

Одиницею інформації в комп'ютері є один біт, тобто двійковий розряд, який може приймати значення 0 або 1. Вісім послідовних або паралельних бітів складають байт. У одному байті кодується значення одного повідомлення з 256 можливих. Продуктивність АЛП процесора додатково характеризується наступними параметрами: внутрішньою і зовнішньою розрядністю даних, що обробляються (чим вище розрядність, тим вище інформаційна продуктивність одночасно обробляємої інформації); об'ємами пам'яті, адресацією центрального процесорного пристрою CPU (central processor unit); ступенем інтеграції транзисторів, ін..

Термін «автомат», як правило, використовується в двох аспектах. З одного боку, автомат – це пристрій, що виконує деякі функції без безпосередньої участі людини. У цьому сенсі ми говоримо, що ЕОМ - автомат, оскільки після завантаження програми і початкових даних ЕОМ вирішує задану задачу без участі людини. З іншого боку, термін «автомат» як математичне поняття позначає математичну модель реальних технічних автоматів. У цьому сенсі автомат представляється як «чорний ящик», що має кінцеве число входів і виходів і деяку безліч внутрішніх станів $Q = \{q_1(t), q_2(t) ..., q_n(t)\}$, у яких він під дією вхідних сигналів переходить стрибкоподібно, тобто практично миттєво, минувши проміжний стан. Звичайно, ця умова не виконується в реальності, оскільки будь-який перехідний процес триває кінцевий час.

Цифровий (дискретний) автомат (ЦА) – пристрій, який здійснює прийом, зберігання та / або перетворення дискретної інформації по деякому алгоритму. Прикладами ЦА можуть служити живі організми, процесори, побутова техніка калькулятори - це реальні пристрої, а також є абстрактні, наприклад, моделі алгоритмів. ЦА відносяться до послідовних пристроям. Цей клас пристроїв визначається тим, що значення виходів залежить не тільки від вхідних значень, але і від поточного стану пристрою. Т.е. вводиться поняття – стан. Для того щоб зберігати дані про стан, в якому знаходиться пристрій в ЦА використовуються запам'ятовуючі елементи – тригери.

Цифровий автомат (ЦА) – це умовна, формальна модель будь-якого інформаційного пристрою дискретної дії. Модель призначена для представлення пристрою на логічному рівні, тобто в ній не допускається використання будь-яких фізичних характеристик процесів, що відбуваються при роботі реального пристрою.

У зв'язку з таким спрощенням число помітних станів, в яких може опинитися ЦА, завжди обмежена. Перехід з одного стану в інший мислиться як миттєвий стрибок, без всяких проміжних станів, тоді як в реальній фізичній системі відповідний перехід повинен мати незліченну безліч перехідних фаз із зникаюче малими відмінностями подальшого стану від попереднього.

Нескінченне число станів для цифрового автомата означає його нескінченну складність, так як інакше не можна забезпечити явну відмінність одного стану від іншого.

В принципі можна уявити собі нескінченний ЦА, як автомат з необмежено нарощуваною пам'яттю станів, і такий автомат становить певний теоретичний інтерес, але в даному курсі ми будемо стосуватися тільки кінцевих автоматів. До речі, терміни «кінцевий автомат» і «цифровий автомат» часто використовуються як синоніми і по суті виражають одне і те ж поняття.

Отже, підсумовуючи сказане, ми визначили ЦА як умовну, спрощену модель технічного пристрою, що має обмежене дискретне безліч станів.

У той же час можна дати і зовсім інше визначення, що випливає з призначення описуваного пристрою.

Можна вважати ЦА моделлю алгоритму, що задає певний процес перетворення інформації. У цьому розумінні поняття ЦА аналогічно поняттю машини Тьюринга, хоча прямого зв'язку між цими поняттями немає. Машина Тьюринга це пристрій, який моделює дії людини, вирішального завдання, керуючись деяким алгоритмом. Тобто, машина Тьюринга являє умоглядну конструкцію, що дозволяє зручно описувати на інтуїтивному рівні спеціальні завдання математичної теорії виводу, а ЦА – це модель, що дозволяє займатися логічним синтезом реальних систем, відволікаючись від протікають у них фізичних процесів.

Робота ЦА протікає в часі, але це особливе, автоматне час, що представляє зміну послідовних дискретних моментів t = 0, 1, 2, 3, ... ЦА завжди починає свою роботу з моменту t = 0, а час закінчення може бути не визначеним або нескінченним.

Теорію автоматів прийнято ділити на дві основні частини: абстрактну і структурну.

Розглянемо деякі питання класифікації автоматів. Всі ЦА можна розділити на два основні класи: автомати без пам'яті, або примітивні автомати, і автомати з пам'яттю.

Автомати без пам'яті – це звичайні комбінаційні логічні мережі, які не містять елементів пам'яті, лінії зв'язку в них йдуть тільки від входів до виходів і не утворюють зворотних зв'язків. Внутрішній стан у них одне і воно не може змінюватися. Тому вихідна буква повністю визначається вхідний буквою, що діє

в даний момент автоматного часу. Робота автомата без пам'яті повністю описується функцією виходів

$y = 1(x)$.

У структурному алфавіті для автомата, що має кілька виходів, можна написати $\overline{y} = \overline{\lambda}(\overline{x})$, що слід розуміти як систему функцій, число яких дорівнює числу структурних виходів.

Типовими автоматами без пам'яті є цифрові комбінаційні елементи (КЕ): суматори, дешифратори, схеми порівняння, мультиплексори і т.п. При описі таких КЕ використовується структурний за своїм змістом алфавіт, але позначення (наприклад, вхідна змінна x) можуть збігатися з прийнятими в літературі буквами абстрактних алфавітів. Стосовно до автоматів без пам'яті це допустимо, зважаючи на простоту їх функцій, і не призводить до неправильного розуміння. При описі автоматів з пам'яттю доведеться більш строго розрізняти позначення змінних на абстрактному і структурному рівнях.

З точки зору сигналів цифрового автомат корисно визначити як систему, яка може приймати вхідні сигнали, і під їх впливом переходити з одного стану до іншого, зберігати його до приходу наступного вхідного сигналу, видавати вихідні сигнали (рис. 2.3 (а)).

Цифровий автомат вважається кінцевим, якщо кінцеві множини вхідних сигналів X, станів S і вихідних сигналів Y, тобто це автомат з обмеженою пам'яттю станів. Кінцевий автомат, в кожному стані якого існує функція переходу для всіх можливих вхідних символів, називається повністю визначеним кінцевим автоматом.

Робота ЦА здійснюється в автоматному часу, визначеному числом періодів надходження вхідних сигналів.

Будь цифрової автомат складається з двох частин: комбінаційної логічної схеми (КС) і пам'яті (П). З урахуванням цього ЦА може бути зображений так, як показано на рис. 2.3(б). В даному випадку в деякій мірі розкрита структура автомата. КС автомата формує вихідні сигнали, сигнали перекладу тригерів блоку пам'яті в нові стани. Наявність блоку пам'яті дозволяє пам'ятати передісторію роботи автомата під впливом вхідних сигналів.

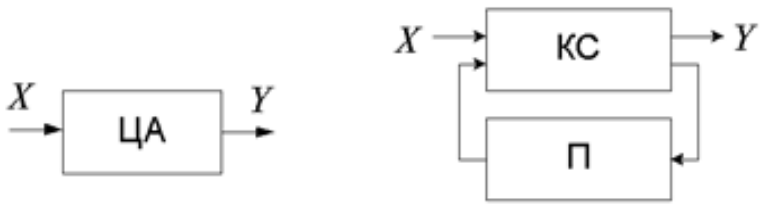


Рис. 2.3. а) Загальне зображення; б) структурне зображення

Основу будь КС складають булеві функції.

Контрольні питання:

1. Які функції виконує ЕОМ в системах управління?
2. Які інтерфейси користувачів ви знаєте?
3. Дайте визначення цифрового автомату.
4. Що входить до структури ЕОМ?
5. Що розуміють під математичною моделлю ЦА?
6. Що таке скінченний автомат з виходом і без, з чого він складається?
7. З яких частин складається цифрової автомат?
8. Дайте визначення поняттю «алгоритм».
9. На які два основні класи можна розділити цифрові автомати?

ТЕМА 3. ПРЕДСТАВЛЕННЯ ЧИСЛОВОЇ ІНФОРМАЦІЇ В ЦИФРОВИХ АВТОМАТАХ

План

1. Системи числення.
2. Алгоритми переведення чисел з однієї системи в іншу.
3. Форми і формати зображення чисел в цифрових автоматах.
4. Зображення чисел в прямому, оберненому і доповнювальному кодах.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Системи числення

Сам процес числення (нумерація) – сукупність певних прийомів (правил, алгоритмів) представлення натуральних чисел. Систем числення існує багато.

У будь-якій системі числення прийняті деякі символи (знаки, слова) для позначення певних чисел, ці знаки називають **базовими (вузловими).** Наприклад, в 10-ій системі базовими є знаки (зображення) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, що позначають цілі натуральні числа від 0 до 9 і дроби за допомогою коми (крапки).

Решта чисел уявного діапазону числової вісі мають назву алгоритмічні і створюються в результаті виконання дій за певними правилами над базовими числами (10, 123, 543). Тому системи числення відрізняються одна від одної вибором базових (вузлових) чисел і способами утворення алгоритмічних. При письмовому позначенні вони відрізняються характером використаних числових знаків і принципами їх запису. Наприклад, у стародавніх вавілонян базовими були числа 1,10,60; у племені маорі (Нова Зеландія) – 1, 11, 112, 113; у римській системі числення вузловими були числа 1, 5, 10, 50, 100, 500, 1000, які позначалися знаками I, V, X, L, C, D, M: приклад MCMXCVII - 1997.

Системи числення, в яких алгоритмічні числа утворюються складанням вузлових, називаються адитивними. Наприклад, в стародавньоєгипетській системі числення числа 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 19, 40 позначалися відповідно:

|, ||, |||, ||||, ||| ||| |||| |||| ||| |||

У римській системі числення числа виходять шляхом складання і віднімання вузлових.

У сучасних числах (в т.ч. європейських, українській і російській мовах, ін.)

застосований адитивно-мультиплікативний спосіб. В деяких країнах були прийняті алфавітні системи числення – наприклад, греки, стародавні слов'яни, ін. (цифри від 1 до 9 зображалися буквами алфавіту з межею зверху).

Будь-яка призначена для практичних розрахунків система числення повинна забезпечувати:

- однозначність відображення числа;
- представлення всіх чисел в діапазоні числової осі;
- простоту запису чисел і виконуваних операцій над ними;
- систему правил запису і читання чисел.

Системи числення діляться на позиційні і непозиційні.

Позиційна система числення – система, в якій значення базових знаків залежить від їх місцеположення в числі, розрядності. (Тобто 0…9 в розряді одиниць не дорівнює 0…9 в розряді десятків, сотень, тисяч, ін. – бо прийнято, що вага сусідніх розрядів таких чисел різниться на порядок).

Непозиційна система числення- система, в якій значення базових знаків не залежить від їх місцеположення в числі, розрядності.

Принцип побудови таких систем полягає у виборі базових знаків і при читанні (або запису) проводяться операції складання (іноді і віднімання). Наприклад: Робінзон Крузо застосував непозиційну систему числення для підраховування діб. Базовий знак – вертикальна риска, які він складав і отримував числа. Ця система неефективна з двох причин:

- запис числа може виявитися довгим і незручним;
- для виконання підрахунків необхідно використовувати іншу систему.

Проте вона дуже проста.

Римську систему числення можна назвати частково позиційною, хоча вона вважається непозиційною системою. Так, наприклад, в числах LX і XL знак X (10) приймає два значення ±10 і залежить від свого місцеположення відносно L (справа чи зліва).

Система залишкових класів

Однією з непозиційних систем числення, розробленою спеціально для застосування в спеціалізованих ЕОМ, є система в залишкових класах (СЗК).

Ідейне коріння СЗК пов'язане з працями Ейлера, Гауса і Чебишева з теорії порівнянь. Значний внесок в розвиток СЗК внесли чеські вчені математики М. Валах і А. Свобода, що працювали над представленням чисел у вигляді сукупності ненегативних вирахувань по групі взаємнопростих основ [1-21].

Хай необхідно записати число N, задане в 10-ій системі в системі СЗК.

Кінцеву множину позитивних цілих чисел, взаємно простих між со-бою, назвемо базисом основ системи числення в залишкових класах [14] $P = \{p_1, p_2, ..., p_n\}$.

Під системою числення в залишкових класах будемо розуміти таку систему, в якій ціле число представляється у вигляді набору залишків (вирахувань) по вибраних основах p_i.

Тоді число N в СЗК буде записано як: $N = \{\alpha_1, \alpha_2, ..., \alpha_n\}$,

(1)

де $\alpha_з = N - \left[\left[\frac{N}{p_i}\right]\right] p_i$, *(3.1)*

тобто a_i – є відображення числа N у вигляді позитивного залишку від ділення N націло на основу P_i.

При цьому буде завжди справедлива нерівність $a_i<p_i$. (де i=1,2…n).

У відомій з теорії чисел першій і другій функціональній теоремі Гауса, доведено, що якщо числа p_i взаємно прості між собою, то опис $\{a_1, a_2, a_n\}$, що представляє N, є єдиним.

Нагадаємо, що взаємно простими числами називають числа, що мають **найменший загальний дільник (НЗД)**, що дорівнює 1. Наприклад, 15 і 22 є взаємно прості числа (оскільки НЗД рівний 1).

Приклад. Нехай вибрані основи СЗК: p={3,5,7}. Представимо десяткове число N=67 у СЗК.

Рішення. Знайдемо залишки згідно виразу 3.1

$$\alpha_1 = 67 - \left[\frac{67}{3}\right] \cdot 3 = 67 - 66 = 1;$$

$$\alpha_2 = 67 - \left[\frac{67}{5}\right] \cdot 5 = 67 - 65 = 2;$$

$$\alpha_3 = 67 - \left[\frac{67}{7}\right] \cdot 7 = 67 - 63 = 4.$$

Отже, число 67 записується в СЗК як {1,2,4}, або в двійковій – {001,010,100}.

Діапазон чисел, що представляються, дорівнює:

$R = p_1 \cdot p_2 \cdot p_3 \cdot ... \cdot p_n$; $(R = 3 \cdot 5 \cdot 7 = 105)$.

Зворотний переклад числа з СЗК в десяткову систему проводиться за формулою:

$$N = (\alpha_1 \cdot B_1 + \alpha_2 \cdot B_2 + ... + \alpha_n \cdot B_n) - r \cdot R, \quad (3.2)$$

де r – ранг, що приймає значення 0, 1, 2, … так, щоб права частина виразу 3.2 була менше R;

B_i – ортогональний базис, що визначає при виборі базису СЗК.

$$B_i = k_i \cdot \frac{R}{p_i},$$

де k_i – ціле позитивне число (k_i=1, 2, …, p_i–1), при цьому k_i вибирається таким, щоб **залишок від ділення B_i на p_i дорівнював одиниці** (тобто вага базису k_i вибирається виходячи з рівності):

$$B_i = \frac{m_i \cdot R}{p_i} = 1 = k_i \cdot p_i + 1, \quad (3.3)$$

Приклад. Визначимо для СЗК з основами {3,5,7} ортогональні базиси B_i та переведемо число {1,2,4} в десяткову систему.

$B_1 = \frac{k_1 \cdot 105}{3} = k_1 \cdot 35 = (2 \cdot 35); \qquad B_1 = 70;$

(для p_1=3 умова 3.3 виконується при k=2),

$B_2 = \frac{k_2 \cdot 105}{5} = (1 \cdot 21); \qquad B_2 = 21;$

(для p_2 =5 умова 3.3 виконується при k=1),

$B_3 = \frac{k_3 \cdot 105}{7} = (1 \cdot 15); \qquad B_3 = 15;$

(для p_3 =7 умова 3.3 виконується при k=1), тоді $N = (a_1 \cdot B_1 + a_2 \cdot B_2 + a_3 \cdot B_3) - r \cdot R = (1 \cdot 70 + 2 \cdot 21 + 4 \cdot 15) - r \cdot 105 = 172 - 1 \cdot 105 = 67$.

Запишемо число 67 в двійковій системі 1000011 і в СЗК 001_ 010_ 100. Цифри СЗК в кожному розряді вирахувань по-перше, незалежні один від одного і не залежать від позиції, а лише залежать від базису; по-друге, кожен розряд вирахувань несе інформацію про число в цілому. Ця властивість СЗК використовується для відновлення спотвореної інформації при передачі по лінії зв'язку.

Система числення, в якій значення базисної цифри визначається її положенням в розрядах числа, називається позиційною. Наприклад, в десятковій системі числення в числі 222, перша цифра зліва від коми означає дві одиниці, друга – два десятки, третя – дві сотні.

Позиційна система числення визначається своєю основою.

Основа q позиційної системи числення – кількість знаків, що використовуються для відображення числа в даній системі.

Якщо q – ціле позитивне число, то позиційна система числення називається природна.

Позиційних систем числення може бути багато оскільки за основу можна прийняти будь-яке число і утворити нову систему числення.

Число в позиційній системі числення визначається рівністю:

$$A(q) = a_n q^n + a_{n-1} q^{n-1} + \ldots + a_0 q^0 + a_{-1} q^{-1} + \ldots + a_{-m} q^{-m},$$

або

$$A(q) = \sum_{i=-m}^{i=n} a_i q^i,$$

де a_i – коефіцієнти розрядів в знаках системи; q – основа системи числення; $A(q)$ – довільне число; m, n – кількість дробових і цілих розрядів.

При записі числа, запис основи і знак "+" опускають, а ступінь визначають по розряду від коми,

тобто $A(q) = a_n\, a_{n-1} \ldots a_1\, a_0,\, a_{-1} \ldots a_{-m}$.

Наприклад, число 934,125 записано в десятковій системі числення. Число $124{,}57_{(8)}$ записано у 8-й системі числення (беруться цифри 0, 1, ..., 7).

У двійковій системі числення використовують цифри 0, 1.

Наприклад, число $1001{,}1 = 1 \cdot 2^3 + 1 \cdot 2^0 + 1 \cdot 2^{-1} = 17{,}5$ (див. таблицю 3.1).

Вага розряду P_i числа в позиційній системі числення – це відношення $P_i = \frac{q^i}{q^0} = q^i$, де i – це номер розряду справа наліво. Вага розрядів росте справа

наліво. Якщо візьмемо розряд $P_i=10^K$, то наступний старший матиме вагу $P_{i+1}=10^{K+1}$, а молодший $P_{i-1}=10^{K-1}$. Такий взаємозв'язок розрядів припускає, при виконанні операцій, передачу інформації між ними.

Таблиця 3.1

Еквіваленти десяткових чисел в різних системах числення

Десяткова цифра	q=2	q=3	q=5	q=8	q=16
0	0000	000	00	00	0
1	0001	001	01	01	1
2	0010	002	02	02	2
3	0011	010	03	03	3
4	0100	011	04	04	4
5	0101	012	10	05	5
6	0110	020	11	06	6
7	0111	021	12	07	7
8	1000	022	13	10	8
9	1001	100	14	11	9
10	1010	101	20	12	A
11	1011	102	21	13	B
12	1100	110	22	14	C
13	1101	111	23	15	D
14	1110	112	24	16	E
15	1111	120	30	17	F

Якщо в даному розряді накопичилося значення одиниць (десятків, сотень, і т.д.) рівне або більше q, то повинна відбуватися передача одиниці в сусідній, старший розряд. При складанні, такі передачі назвемо ***переносом одиниці в старший розряд***, а при відніманні – ***позикою одиниці із старшого розряду***. Приклад: 9 + 2 = 11; 17 – 8 = 9.

Глибина числа – це кількість знаків (зображень), які використано при ***записі числа в одному розряді.*** Поняття «глибина числа» співпадає з поняттям «основа» і дорівнює значенню q – тобто для 10-ї, 8-ї, 2-ї систем числення, відповідно: 0, 1, 2, …, 9 (десять); 0, 1, 2, …, 7 (вісім); 0, 1 (два).

Довжина числа – це кількість розрядів в записі числа. Часто вжива-ється термін – довжина розрядної сітки пристрою (кількість розрядів n).

Для різних систем характерна своя довжина розрядної сітки для запису одного і того ж числа. Наприклад, $96_{(10)} = 140_{(8)} = 10120_{(3)} = 1100000_{(2)}$. Чим менша основа, тим більше потрібно розрядів для запису числа і навпаки.

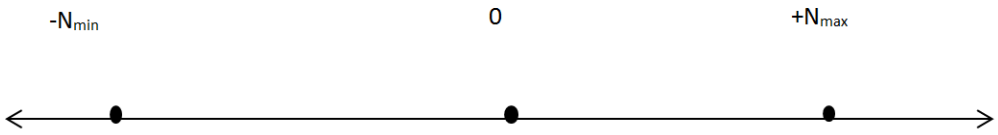


Рис.3.1 – Числова вісь діапазону чисел, що представляються

Величина розрядної сітки n, що прийнята в ЦА, у свою чергу накладає обмеження на уявний діапазон чисел (яким він може оперувати) – це інтервал числової вісі між максимальним і мінімальним числом (рис. 3.1). У загальному випадку, максимальне (мінімальне) число уявного діапазону, залежить від основи системи числення і кількості виділених розрядів n: $N(q)_{max} = (q^n-1)$, $N(q)_{min} = -(q^n-1)$. Наприклад, три розряди десяткової системи забезпечать діапазон чисел (з урахуванням знаку) $N_{min}= -999$, $N_{max}= 999$.

Вибір системи числення комп'ютера

Переваги двійкової системи

Розробку комп'ютера починають з вибору системи числення, методів виконання арифметичних і логічних операцій, елементної бази. Ці питання є важливими, оскільки повинні забезпечити задану точність і швидкість обчислень, надійність в роботі, діапазон чисел.

Вибір системи числення у великій мірі обумовлюється наступними основними причинами:

1. Основу системи визначає число знаків (їх зображень) в одному розряді. Тут перевагу має двійкова система, а не десяткова оскільки вона вимагає всього два знаки (стани), а десяткова – десять. Елементи, з десятьма станами (декатрони), що є в наш час, мають малу швидкість перемикання, громіздкі, мають невисоку надійність в роботі.

2. З одного боку система числень повинна забезпечити ***простоту арифметичних операцій***, а з іншого великий діапазон уявних чисел. Ці дві суперечності вирішуються на користь ***простоти арифметичних операцій*** при представленні чисел. Насправді, в двійковій системі не треба додаткових пристроїв представлення чисел. Сам сигнал несе інформацію про число і має двійкову природу: є сигнал або його немає - "1 " або "0". Крім того, майже всі елементи електронних схем (є струм/ напруга або ні) мають двійкову природу (реле, діод, транзистор, тригер, фотоприймач, стан оптоволокна, ін.).

Якщо прийняти показник ефективності системи з погляду витрат устаткування:

$$C=q\cdot n,$$

де q – основа (глибина) системи, n – довжина (розрядність) системи.

Розрядність n визначається з діапазону уявних чисел. Максимальне число з основою q: $N_{(q)max}=q^n-1$, звідси знайдемо n:

$$n=\log_q(N_{(q)\,max}+1),$$

тоді витрати на устаткування підраховують за формулою:

$$C = q \log_q (N(q)_{max} + 1).$$

Якщо за одиницю устаткування прийняти умовний елемент з одним стійким станом, то можна порівняти системи числення з погляду витрат.

Графік відносного показника приведений на рис. 3.2.

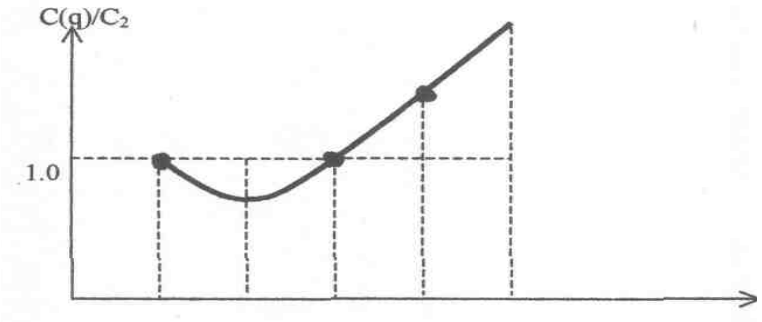


Рис.3.2 – Відносні витрати при різних значеннях основ

Наймінімальніші витрати будуть при основі e = 2,72, тобто, з погляду витрат, виходить найекономнішою буде система з q=3. Вона застосовується в ЕОМ «Сетунь». Проте, більшість ЕОМ використовують двійкову системи через простоту арифметичних операцій в ній.

Розглянемо це на прикладі складання в цій системі:

0+0=0, 1+0=1, 0+1=1, 1+1=10.

Складання цілих чисел без знаку представлено схемою (рис. 3).

Приклад. Знайти $C=A+B$. A=+1011 (+11); B=+1001 (+9); C=+10100 (+20).

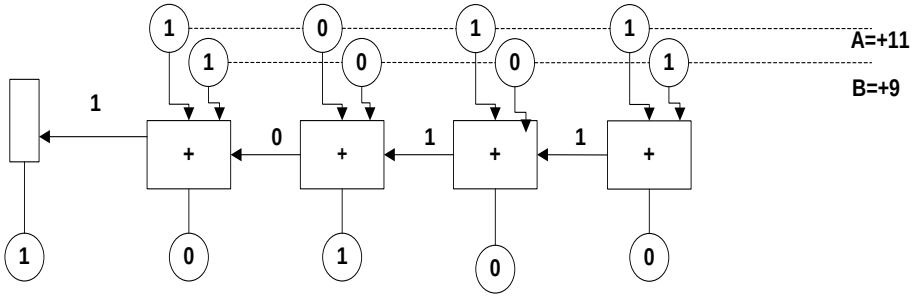


Рис.3.3 – Структурна схема бінарного (двійкового) суматора

Множення проводиться в двійковій системі ще простіше: 0·0=0, 0·1=0, 1·0=0, 1·1=1.

2. Алгоритми переведення чисел з однієї системи в іншу.

Раніше ми відзначали, що будь-яке число можна представити поліномом з основою q_1, але це ж число можна представити іншим поліномом з основою q_2, інакше: $N(q_1)= N(q_2)$. Представимо це для загального числа (що має цілу і дрібну частини – неправильний дріб) у наступному вигляді:

$$N(q_1)=a_n q_1^n+a_{n-1}q_1^{n-1}+\ldots+a_1 q_1^1+a_0 q_1^0+a_{-1}q_1^{-1}+\ldots+a_{-m}q_1^{-m}=$$

$$=b_k q_2^k+b_{k-1}q_2^{k-1}+\ldots+b_1 q_2^1+b_0 q_2^0+b_{-1}q_2^{-1}+\ldots+b_{-s}q_2^{-s}=N(q_2)$$

Методи перекладу цілих чисел

Метод підбору коефіцієнтів

Завдання перекладу числа з основою q_1 в число з основою q_2 зводиться до відшукування коефіцієнтів полінома нової основи. Цю задачу можна вирішити методом підбору коефіцієнтів полінома.

Правило. ***На початку беремо число з максимальною степінню основи, перевіряємо її вхід до даного числа так, щоб воно не перевищувало задане, потім підбираємо коефіцієнти менших степіней полінома, так щоб сума давала початкове число. Всі дії повинні виконуватися за правилами q_i-ї арифметики, тобто за правилами початкової системи числення.***

Наприклад. Перевести десяткове число 96 у трійкову систему.

Рішення. $96 = 0\cdot3^5+1\cdot3^4 + 0\cdot3^3+1\cdot3^2+2\cdot3^1 + 0\cdot3^0 = 010120_{(3)}$

Проведемо перевірку методом підстановки значень ваги розрядів, множення його на підібраний коефіцієнт і обчислення загальної суми.

$96=0\cdot243 + 1\cdot81 + 0\cdot27 + 1\cdot9 +2\cdot3 + 0\cdot1.$

Цей прийом застосовується тільки при «ручних» перекладах. Машинні алгоритми використовують метод ділення на основу нової системи числення.

Метод ділення на основу нової системи

Поліном цілого числа $N(q_1)$ можна записати за схемою Горнера в іншому вигляді:

$$N(q_2)=(\ldots((b_k q_2+b_{k-1})q_2+b_{k-2})q_2+\ldots+b_1)q_2+b_0$$

де $b_0 \ldots b_k$ -коефіцієнти молодшого b_0, старшого b_k розрядів числа $N(q_2)$.

Якщо праву частину розділити на q_2, то отримаємо цілу частину (у лапках) і залишок з b_i ($0<i<k$). Повторюючи ділення $k+1$ разів отримуємо кожного разу залишки b_1, b_2, ..., $b_к$ і цілі частини. Останній залишок $b_к$ є старшим розрядом залишку числа, представленого в основі q_2 (неподільний залишок).

Приклад. $12_{(10)}$ перевести в бінарну (двійкову) систему числення.

Рішення. Ділимо на q=2 (у дужки поміщений залишок):

12:2=6(0)
6 : 2=3(0)
3 : 2=1(1)

Частка (1) менше основи 2, тому це теж залишок, причому старший, тому припиняємо ділення. Відповідь: 1100 ($12_{(10)}$).

Метод ділення на основу в будь-якій позитивній степіні

Попередній метод має один недолік. При великих числах, операція ділення має багато ітерацій. Це знижує швидкодію. Метод ділення на основу нової

системи в будь-якому позитивному степіні аналогічний попередньому. Тут беруть для ділення число (з новою основою q_k) близьке до заданого числа $N(q_i)$, але що не перевищує його. Кожен залишок від ділення записують в двійкових розрядах, число яких рівне узятому степіні.

Наприклад. Перевести в бінарну систему числення число N=523 $_{(10)}$.

Рішення. Вибираємо, найближче до заданого, число $2^9 = 512$ і ділимо:

523 : 512=1(залишок 11).

Отримали два залишки: старший – 1, молодший – 11. Кожний із залишків розписуємо в дев'яти бінарних розрядах: 1000000000 і 000001011. Потім сполучаємо (додаємо) записи (старші нулі можна не записувати) і отримуємо результат- число 1000001011.

В двійковій системі (для цілого числа) b_0 завжди «0» або «1». В першому випадку $N(q_2)$ – це парне число, в другому – непарне.

Метод віднімання найближчих, менших степінних ваг

Метод полягає в наступному. Вибирають значення найближчої, розрядної ваги бінарного (двійкового) числа, менше заданого.

Віднімаючи його від заданого числа, отримують залишок. У розряді вибраної ваги, ставиться 1. Потім порівнюють залишок з новим меншим ваговим розрядом. Якщо залишок менший, то в цьому ваговому розряді ставиться 0, якщо залишок більший, то в цьому розряді ставиться 1, а із залишку віднімається вага цього розряду. Виходить новий залишок, який знову порівнюється з наступними меншими вагами. Так продовжується до останнього (молодшого) вагового бінарного розряду і отримають число.

Приклад. Перевести десяткове число 1125 в двійкове. Метод – віднімання менших найближчих розрядних ваг. Найближчою меншою розрядною вагою буде число $1024=2^{10}$. Ставимо в десятому розряді 1 (зліва від коми, де міститься 2^{10}).

Віднімаємо 1125–1024 =101, де 101 – десятковийзалишок, який порівнюємо з числом 2^9= 512. Залишок 101 менше 512, це означає, що на місці 2^9 ставимо 0. Порівнюємо наступну розрядну вагу 2^8= 256>101. Знову на місці розрядної ваги 2^8 ставимо 0. Аналогічно буде і для розряду 2^7 – ставимо 0. Порівнюємо наступну розрядну вагу 2^6=64<101. У цьому розряді ставимо 1 і віднімаючи 101–64=37 отримуємо новий залишок, який порівнюємо з розрядною вагою 2^5=32<37. В цьому розряді ставимо 1, і віднімаючи 37–32=5, отримуємо новий залишок, який легко розписати в чотирьох розрядах, що залишилися: 0101. Таким чином, отримуємо число $10001100101_{(2)}=1125_{(10)}$. Перевіримо: 1024+64+32+4+1=1125.

Метод виключає громіздку операцію ділення і при невеликому досвіді легко виконується користувачем. Метод придатний для перекладу як цілої, так і дробової частини числа.

Переклад правильних дробів множенням на основу системи числення

Дробову частину числа можна записати в новій системі:

$$N(q_2) = b_{-1}q_2^{-1} + b_{-2}q_2^{-2} + ... + b_{-k}q_2^{-k}, \qquad (3.4)$$

цей вираз можна переписати по схемі Горнера:

$$N(q_2) = q_2^{-1}\left(b_{-1} + q_2^{-1}\left(b_{-2} + \ldots + q_2^{-1}\left(b_{-(k-1)} + q_2^{-1} b_{-k}\right)\right)\ldots\right).$$

Якщо праву частину множити послідовно на q_2, то знаходитимемо новий неправильний дріб, в цілій частині якої будуть числа b_{-1}, b_{-2}, ..., b_{-k}, при цьому всі дії повинні виконуватися за правилами q_1-арифметики, і отже в цілій частині дробів, що виходять, з'являтимуться еквіваленти чисел нової системи числення, записані в початковій системі числення.

Приклад: Перевести десятковий дріб 0,625 в двійкову систему числення. Рішення:

0,	625·2
1	250·2
0	500·2
1	000·2
0	000

Відповідь: N= 0,1010. Перевірити за формулою.

Правило: ***для перекладу правильного дробу (без цілої частини) необхідно, діючи в арифметиці початкової системи числення, помножити число, що переводиться, на основу нової системи, у результату відокремити цілу частина, а дробову частину, що залишилася, знову помножити на цю основу і так до отримання потрібного числа значущих цифр. Результат записувати як 0, ... і дробову частину у порядку отримання.***

Приклад. Перевести двійковий дріб 0,1101 в десятковий.

Рішення:

	0,	1101·1010
b_{-1}	1000,	0010·1010
b_{-2}	0001,	0100·1010
b_{-3}	0010,	1000·1010
b_{-4}	0101,	0000

Відповідь: N=0,8125.

При перекладі правильних дробів з однієї системи числення в іншу, може вийде дріб у вигляді нескінченного ряду або ряду, що розходиться. Тому, процес перекладу необхідно закінчувати:

- при появі в дробовій частині по всіх розрядах нулів;
- якщо буде досягнута задана точність дробу (тобто необхідне число роз рядів).

Приклад. Перевести дріб 0,543 з шісткової системи в трійкову.

Рішення:

0,	$543_{(6)}$·3
2,	513·3
2,	343·3
1,	513

Відповідь: $N = 0{,}543_{(6)} \sim 0{,}221_{(3)} = 0{,}532_{(6)}$

Переклад неправильних дробів

Правило. ***Для перекладу неправильного дробу (тобто такого дробу, що містить цілу частину) з однієї системи числення в іншу необхідно здійснити роздільно переклад її цілої і дробової частин, а результати записати послідовно, відокремивши цілу частина від дробової комою.*** Наприклад: $98{,}625_{(10)}$ дорівнює $1100010{,}1010_{(2)}$.

Переклад з 16-ої і 8-ої систем в 2-ву і навпаки

При перекладі чисел з десяткової системи в 2-ву, часто використо-вують проміжну 8-ву систему. Це дає економію в числі операцій.

Таблиця 3.2

Переведення у бінарну систему

$q_2=8$		$q_2=2$		
Число	залишок	число	залишок	тріада
121	1	121	1	
15	7	60	0	1
1	1	30	0	
3 кроки		15	1	
		7	1	7
		3	1	
		1	1	1
		7 кроків		

Правило. ***Щоб перевести число з 16-ї і 8-ї системи в 2-у необхідно кожну цифру числа, що переводиться, представити відповідно чотирирозрядним або трирозрядним двійковим кодом (тетрадами або тріадами), розташував їх на місцях (розрядах) цих цифр. Нулі в старших і молодших розрядах, що не змінюють значення числа можна опускати.***

Приклади: $171_{(8)}$ в двійкове N=001 111 001;
$753{,}335_{(8)}$ в двійкове
N=111 101 011,011 011 101.

Правило. ***Для перекладу чисел з двійкової системи числення в шістнадцяткову (вісімкову) слід двійкові цифри числа, що переводиться, згрупувати по чотири (три) в обидві сторони від коми (при необхідності неповні групи доповнити нулями), кожну групу двійкових цифр замінити відповідною їй цифрою в новій системі числення. Нові цифри розташувати на місцях замінюваних кодів.***

Приклад: $111111010,1100001001_{(2)}$ перевести в 16-у і 8-у системи числення.

а) 0001 1111 1010 , 1100 0010 0100$_{(2)}$

1 F A , 3 2 4$_{(16)}$

Відповідь: $1FA,C24_{(16)}$

б) 111 111 010, 110 000 100 100$_{(2)}$

7 7 2 , 6 0 4 4 $_{(8)}$

Відповідь: $772,6044_{(8)}$

Узагальнюючи, робимо висновок: у якості проміжних систем числення, доцільно використати системи з основою $q=2^k$, для спрощення перетворення їх в двійкову систему і навпаки (де цифри замінюють їх еквівалентами).

Переклад двійково-десяткових систем числення

Двійкові розряди розбиваються на тетради з вагою розрядів наприклад, 8421, а в тетрадах записують цифри від 0 до 9 у двійково-десяткових кодах, тобто 1 треба ставити у тих розрядах, вага яких враховується в записуваній цифрі. Переклад з однієї системи до іншої проводять по тетрадах.

Приклад: $N=995_{(10)} \rightarrow$ 1001/1001/0101.

3. Форми і формати зображення чисел в цифрових автоматах

Форма представлення чисел в комп'ютері

У десятковій системі існують багато форм представлення чисел. Ми, вміло користуючись цим, самі вибираємо ту або іншу форму представлення чисел. Наприклад, число 0,25 можна уявити, як 1/4, коли виконуємо операції з такими ж дробами. Можна уявити, як $25\cdot10^{-2}$ або $2500\cdot10^{-4}$ або $0,0025\cdot10^{2}$ або 0,3 при округленні і т.д.

Всю різноманітність запису чисел розбивають на природні і нормалізовані (нормальні) форми.

Нормалізованою формою запису будь-якого числа N звичайно називають математичний вираз типу $N=\pm0,mxq^{(\pm n)}$, де $\pm m$ – мантиса (у вигляді правильного дробу зі значенням у старшому розряді, що не дорівнює «0»), q – основа, $\pm n$ – порядок числа.

При природній формі, число записують в натуральному вигляді, наприклад: для позитивних чисел: 125 (ціле число); 0,125 (правильний десятинний дріб); 125,125 (неправильний десятинний дріб). При норма-лізованій формі запису одне і те ж число може приймати різний вигляд і його форма залежить **від прийнятих правил (обмежень),** що діють при записі. Наприклад, 12500 може бути записано $12,5\cdot10^{3}=0,125\cdot10^{5}=125000\cdot10^{-1}$ і т.д.

Така різноманітність форм представлення чисел є причиною ускладнення пристроїв і алгоритмів функціонування ЦА обчислювачів.

Щоб уникнути цього, в ЦА прийняті ***свої нормалізовані форми запису і відображення чисел***, що дозволяють мінімізувати вартість апаратних і алгоритмічних засобів.

Формати представлення чисел в комп'ютері

Представлення чисел з фіксованою комою (крапкою)

Автоматне зображення числа – це представлення числа *N* в розрядній сітці ЦА, в наперед заданому форматі і правилами відображення. На рис. 3.4 представлений звичайний машинний формат числа з фіксованою комою на *n*-розрядів (для знакової частини числа надано один додатковий розряд *Sg*1).

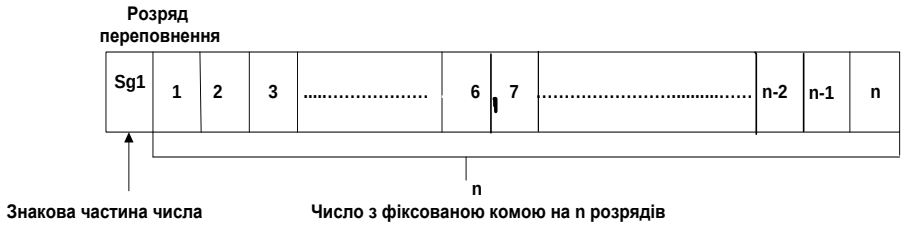


Рис. 3.4 – Формат машинного представлення числа з фіксованою комою (крапкою) – ФФК

При представленні числа у форматі з фіксованою комою (ФФК) в двійковій системі числення увесь формат *n*-розрядної сітки розбивається наперед на дві основні частини-знакову і числову:

- один (або два зліва) розряди для представлення знаку числа (0 або 00 плюс «+»; 1 або 11 мінус «–»);
- наступні k розрядів визначають для розміщення цілої частини числа;
- інші (*n*–2–*k*) розрядів відводять для розміщення дробової частини числа.

Дійсно, положення коми строго фіксоване в розрядній сітці ЦА.

Позначимо машинне зображення числа *N* через N_m, тоді число N_m дорівнює:

$$N_m = \frac{N}{K_\Phi},$$

де K_ϕ – масштабний коефіцієнт прийнятого формату, величина якого ***залежить від числа розрядів цілої частини***, тобто для рис 3.4 $K_\Phi=2^k=2^6$.

Приклад. Записати число N=+11,00111000111 (природний запис) у форматі машини з $K_\Phi=2^6$, кількість розрядів n=16. $N_m= N\cdot 2^{-6}$.

$[N]_m$=0/000011,001110001 (машинний запис числа *N*, останні два розряди 11 – втрачені із за відсутності знакомісць для їх розміщення).

Для сприйняття машинної форми числа N_m знакову його частину (крайні зліва один або два розряди) відокремлюють знаком «слеш» / чи \.

Навпаки, число *N* визначається як $N=N_m\cdot K_\Phi$=+11,001110001. Знак «+» для позитивного числа (природний запис) не записується.

Діапазон уявних чисел залежить від обраного формату цілої частини і складає:

$$R = \pm 2^{k-1} + 2^{k-2} + ... + 2^1 + 2^0,$$

де *k* – кількість розрядів числа.

При записі чисел у форматі з фіксованою комою, якщо число виходить за межі розрядної сітки числа, молодші розряди відкидаються. При цьому, можуть виникати значні погрішності, зокрема, ділення на «0», що приводить до невизначеності і необхідності втручання у розрахунки. Проте, при роботі ЕОМ у форматі з фіксованою комою швидкість виконання арифметичних і логічних операцій висока.

Помилка представлення чисел зменшується при правильному виборі (розрахунку) масштабних коефіцієнтів $K_Ф$. У деяких ЦА при переповненні розрядної сітки автомата, виводять один розряд «переповнення» із зупинкою розрахунків, що порушує його нормальне функціонування. (Наприклад, у ЄС ЕОМ ХХХХ АЛП виконуються операції окремо з цілими числами і дробами, а переповнення від складання дробів додається до цілої частини).

Представлення чисел у форматі з рухомою комою (ФРК)

Іншою найбільш поширеною нормальною формою є представлення чисел у формі з рухомою комою. В цьому випадку в нормальній формі число записується як

$$N = (\pm m)(\pm p),$$

N – нормалізоване число; m – мантисса, а p – характеристика (порядок) числа.

Таке уявлення, в загальному випадку, не є однозначним. Тому, для нормалізованого числа вводять, як правило, обмеження. Найбільш зручною і поширеною є вимога вигляду:

$$q^{-1} \le |m| < 1, \qquad (3.5)$$

де q – основа системи.

Правило. ***Нормалізованою формою представлення чисел є форма числа, для якої справедлива умова 3.5.***

В цьому випадку абсолютне значення мантиси може бути в межах від $0,q^{-1}$ до $(0,q^{-1}+0,q^{-2}+\ldots+0,q^{-n})$, тобто до 1 при $n \to \infty$, де n – кількість розрядів для запису мантиси без знаку.

Розташування коми у числі, мантису якого помножено на порядок, не є постійним. Тому і форма запису називається з рухомою комою (крапкою).

Формат машинного відображення чисел з рухомою комою представлений на рис. 3.5 для 16-и розрядної сітки.

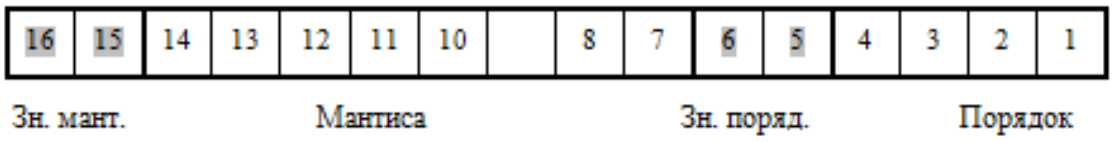


Рис.3.5 – Формат машинного представлення числа з рухомою комою (крапкою)

Формат уявлення цілої частини числа визначається знаком порядку і числом порядку і може змінюваватися. Але завжди формат числа повинен містити: ***знак числа, мантису, знак порядку, порядок*** (де порядок визначає скільки розрядів мантиси необхідно відлічити вправо або вліво, щоб поставити кому, яка відокремлює дробову частину у кінцевому вигляді числа).

Отже, перш ніж записати число в ЦА з його розрядною сіткою, це число ***нормалізують***, тобто приводять до вигляду, коли в старшому розряді мантиси немає «нуля», а є «одиниця» (для двійкової системи числення). Решта всіх форм називається ***ненормалізованими***.

Порядок – це показник степіні для обранної системи числення, на який потрібно помножити (для позитивного знаку порядку) або розділити (для негативного знаку порядку) мантису для перетворення, тобто, $P=q^{\pm p}$,

де P – число, яке записане в двійкових розрядах порядку. При нормалізації число зрушують порозрядно вправо (якщо є ціла частина числа) або вліво (до одиниці після коми). При зрушенні вправо порядок (тобто степінь основи) збільшується з кожним зрушенням на 1. При зрушенні вліво порядок зменшується кожного разу на 1.

Приклади. Записати двійкове число в нормальній формі (звичайний машинний код-один розряд під код знаку):

+11101,011 (природна форма); $0\backslash,11101011\cdot 2^{+5}$ (напівмашинна форма); 0\11101011\0\0101(звичайний машинний код).

Для прийнятого формату (рис.3.5) число запишеться у машинній формі (модифікований машинний код- два розряди під код знаку).

При записі мантиси зручно щоб старший розряд був зліва, а в полі порядку розташування старшого розряду обумовлено коментарем (визначено для окремого ЦА особливо). Приклад:

+0,00011111 (природна форма) = $0\backslash 11111\cdot 2^{-3}$ (напівмашинна форма) = 0\11111000\1\000011 (звичайний машинний код, наявність коми припускається, але в ЦА вона за «слешем» відсутня).

Оскільки основа $2_{(10)}=10_{(2)}$ завжди постійна, то її запис в характеристиці числа опускається, а діапазон уявних чисел визначається розрядною сіткою.

Максимальне уявне число буде при позитивних знаках мантиси і порядку, і при максимальних значеннях їх чисел, іншими словами, при запису числа – по всіх розрядах одиниці. Швидкість обчислень у ФРК менша ніж у ФФК, а точність і діапазон – вища.

Мінімальне уявне число має негативні знаки мантиси і порядку, при мінімальному значенні мантиси і максимальному (тобто одиниці по всіх розрядах) порядку.

$$N_{\max}=\left(1-2^{-n}\right)\cdot 2^{\left(2^{m}-1\right)},$$

$$N_{\min}=-2^{-1}\cdot 2^{\left(2^{m}-1\right)}.$$

Погрішності представлення чисел

Представлення чисел в ЦА, як ми бачимо, завжди спричиняє за собою появу погрішностей в розрахунках, величина яких залежить від обмежень, що накладаються на автомат по розрядній сітці та формі представлення чисел.

Абсолютна похибка представлення чисел ΔN

Абсолютна похибка представлення чисел ΔN називається різниця між дійсним значенням числа N_i та його значенням N_m, отриманим після машинного відображення, операцій АЛП.

Приклад: двійкове число +11,00111000111 при n=15 і $K_Ф=2^k$, k=6, тоді N_m = 0\000011,001110001 [останні два розряди 11 – втрачені]. Перевівши в десяткові числа, отримаємо:

N =11,00111000111 (+3,2221678);

N_m =0\000011,001110001 (+3,2207031);

$\Delta N = N - N_m$ = +3,2221678– (+3,2207031)= +0,0014647.

Правило. ***Максимальна похибка ΔN для чисел формату з фіксованою комою не перевищує одиниці молодшого розряду сітки:***

$$\Delta N = 2^{[n-(k+1)]},$$

де n – розрядність автомата;

k – степінь масштабного коефіцієнта $K_Ф$;

N_m – значення машинного числа в двійковій системі після його переводу (тобто з N в N_m).

Відносна похибка представлення числа δN

Відносна похибка представлення числа δN. Це відношення абсолютної погрішності ΔN до числа N в процентному відношенні, тобто:

$$\delta N = \frac{\Delta N}{N} \cdot 100\,\% \cdot$$

Зазвичай N визначається математично після виконання арифметичних операцій в десятковій системі.

Після визначення абсолютної погрішності визначаємо відносну похибку в %. Приклад: з попереднього рішення ΔN=0,0014647, тоді

$$\delta N = \frac{0{,}0014647}{3{,}2221678} \cdot 100\% = 0{,}04546\,\%.$$

4. Зображення чисел в прямому, оберненому і доповнювальному кодах

Проте, таке віднімання, ділення і множення для автоматів є складним оскільки при виконанні цих операцій в обчислювальних ЦА виникає проблема представлення негативних чисел. Для машинного представлення негативних чисел використовують коди**: прямий, обернений і доповняльний.**

Одним із способів виконання операцій за допомогою двійкового суматора, є заміна операції віднімання операцією суми із оберненим або доповняльним кодом негативного числа:

$$A - B = A + (-B) = A + B^{*}.$$

Заміна віднімання складанням ставить проблему представлення негативного числа в цифровому форматі. Це може бути вирішєно, наприклад, за допомогою використання властивості **циклічності** чисел при їх переліку, рахунку, зображенні (**два**, **три**, …, **два**надцять, **три**надцять…).

Прямий код числа

Один метод вже розглянуто. Він використовує **прямий код числа,** $N = -0, a_1 a_2 ... a_n$ (природний запис числа у вигляді правильного дробу), де для машинного його зображення $N_M = 1 \setminus a_1 a_2 .. a_n$ використовують 1 (або 11) для позначення знаку «–» негативного числа.

Для систем числення з іншою основою q знак «–» (мінус) машинного зображення числа кодується старшим зображенням (вісьмирічна 7, шістнадцятирічна F, т.д.).

Прямий код негативного числа є код, всі цифрові розряди якого залишаються незмінними, записаними в нормалізованій формі, а в знаковій частині числа записується «1» (або 11 для модифікованого коду). Наприклад: $N = -0{,}101110$ то $N_M = 1\setminus 101110$.

Позитивне число в прямому коді не міняє свого зображення.

Прямий код позитивного числа є код, всі цифрові розряди якого залишаються незмінними (наприклад, записаними в нормалізованій формі), а в знаковій частині числа записується 0 (або 00 для модифікованного коду).

Приклад: $N = +0{,}110101$, то $N_{M\,пр} = 0\setminus{,}110101$.

У прямому коді в числову сітку автомата можна записати

$$N_{M\max} = 0 \setminus 111...1.. = 1 - 2^{-n},$$

де n – кількість розрядів. Діапазон чисел прямого коду (для правильного дробу), лежить в межах, визначених виразом:

$$-(1 - 2^{-n}) \le N_M \le (1 - 2^{-n}), \quad (7.1)$$

Але в для операндів A і B в прямому коді виконуються тільки два випадки арифметичні операцій (з чотирьох наявних), тобто:

$C_{пр} = +A_{пр} + B_{пр}$ та $C_{пр} = -A_{пр} - B_{пр} = -(A_{пр} + B_{пр})$.

Іншим методом представлення негативних чисел є уявлення їх в оберненому або доповняльному коді.

Обернений код числа

Оберненим кодом числа $N_M = 1\setminus a_1 a_2 \ldots a_n$ називається таке машинне зображення числа для якого $a_i = 0$ якщо воно дорівнювало «1» і навпаки, $a_i = 1$, якщо воно дорівнювало «0».

Інакше, оберненим кодом **двійкового** числа є **інверсне** зображення цього числа, тобто всі розряди початкового бінарного числа, приймають обернене значення.

Інверсія, інвертування *– процедура переходу до протилежного значення розрядів числа в двійковій системі: інверсія «0» дає «1»; інверсія «1» дає «0». Позначається знаком риски ¯(або хвил ї˜, ін.) над числом, його розрядами.*

Приклад: $N = -101110$ (природний запис), то $N^{м}_{об} = 1\setminus 010001$ (машинний запис

в оберненому звичайному коді).

Узагальнюючи правила утворення оберненого коду на всі основи систем числення можна вважати, що **обернений код негативного числа виходить при відніманні з (*q*–1) цифр по кожному розряду числа за винятком знакових розрядів, які замінюються значенням (*q*–1) тобто:**

$N_{\text{їð}} = 0 \setminus a_1 a_2 .. a_n,$

$N_{\text{îá}} = (q-1) \setminus \overline{a_1}\,\overline{a_2} .. \overline{a_n},$

$\overline{a_i} = (q-1) - a_i.$

Приклад: а) $-0{,}286357_{(10)}$(природний) ↔ $1\setminus{,}286357_{(10)\text{пр}}$ (машинний прямий) ↔ $9\setminus{,}713642_{(10)\text{об}}$(машинний обернений).

б) $-0{,}1010111101_{(2)\text{пр}} \leftrightarrow 1\setminus{,}0101000010_{(2)\text{об}}$

Особливо звернути увагу: **якщо в знаковому розряді машинного запису знаходиться (*q*–1) – тобто число негативне, то всі цифри числа, виключаючи знаковий, замінюються вирахуванням з (*q*–1) значення розряду;**

якщо в знаковому розряді знаходиться нуль – тобто число позитивне, то перетворення не проводяться.

Приклади: (записи наведено без слеша «\», іноді його може заміняти крапка, кома або інший знак):

$$-0.243476_{(10)пп} \rightarrow 9.756523_{(10)оо}$$
$$1.0111000111_{(2)пр} \rightarrow 1.1000111000_{(2)об}$$
$$0.943890_{(10)пр} \rightarrow 0.943890_{(10)об}$$
$$0.1010110101_{(2)пр} \rightarrow 0.1010110101_{(2)об}$$
$$00.110000111_{(2)пр} \rightarrow 00.110000111_{(2)об}$$
$$11.110100000_{(2)пр} \rightarrow 11.001011111_{(2)об}$$

Перехід від оберненого коду до прямого проводиться за аналогічним правилом: із значення (*q*–1) віднімається значення по кожному розряду, окрім знакових (що остаються незмінними).

Для бінарної системи числення (просто щасливий випадок), можна перейти до прямого коду простим інвертуванням розрядів оберненого коду, окрім розрядів знаків.

Приклад: $A_{об}$= 11.1100110
$A_{пр}$= 11.0011001

Доповняльний код числа

Доповняльний код числа N=–0,$a_1a_2..a_n$ – таке машинне уявлення, в якому число $N_m = 1 \setminus a_1 a_2 .. a_n a_{n-1}$ записується оберненим кодом $N_m = 1 \setminus \overline{a_1}\,\overline{a_2} .. \overline{a_n}\,(\overline{a_{n-1}} + 1)$ із збільшенням в молодшому розряді на +1.

Правило перекладу з прямого коду в додатній код наступне:

- якщо в знаковому розряді знаходиться (*q*–1) – тобто число негативне, то всі цифри числа, окрім розрядів знаків, замінюються вирахуваннями з

(q–1) значення розряду, а потім до цифри останнього молодшого розряду додається одиниця;

- якщо в знаковому розряді знаходиться 0 (або 00) – тобто число позитивне, то перетворення цифр не відбувається.

Приклади:

а) -243476$_{(10)}$; 9\243476$_{(10)пр}$ = 9\756523$_{об}$ + 0000001 = 9\756524$_{(10)доп}$

б) -0111000111$_{(2)}$; 1\0111000111$_{(2)пр.}$; 1\1000111000$_{(2)об}$ + 0\0000000001$_{(2)}$= 1\1000111001$_{(2)доп}$

Перевірка:

1\1000111001$_{доп}$

+1\0111000111$_{пр}$

10.0000000000 ($2_{(10)}$)

в) –0,101110 ; 1\101110$_{пр}$; 1\010001$_{об}$ + 0\000001 = 1\010010$_{доп.}$

г) +425736$_{(10)}$; 0\425736$_{(10)пр}$ = 0\425736$_{(10)доп}$= 0\425736$_{(10)об}$

Таким чином, для позитивних чисел прямий, оберненій і доповняльний коди співпадають, для негативного числа вони різні. Для позитивного числа в розряді знаку завжди встановлюють 0 або 00, а для негативного 1 або 11 (для довільної основи q знак «-» має код (q – 1)).

Узагальнемо і отримуємо вираз перекладу чисел у доповняльний код:

$$N_{доп} = \begin{cases} N, N \geq 0; \\ q + (-N), \ N < 0. \end{cases}$$

Приклади:

а) +0,275936$_{(10)}$; 0\,275936$_{(10)\ пр}$= 0\,275936$_{(10)доп}$ (бо позитивне).

б) –0,275936$_{(10)}$; 10,000000$_{(10)}$+(–0,275936$_{(10)}$) = 9\,724064$_{доп}$

Звідси $N_{доп}$=q+(–N), тобто додатній код є математичним доповненням числа до основи системи числення.

Покажемо, що доповняльний код – це доповнення до q, тобто: $|A| + A_{äîï} = q$. Звідси, $A_{доп} = q - |A|$ враховуючи це, будемо віднімати.

Приклад:

A = –0,1011; тоді $A_{доп}$ = _ 10,0000

0,1011

1\,0101 Отримано доповняльний код.

Покажемо, що обернений код – це доповнення до q–q^{-n}, тобто:

$|A| + A_{îá} = q - q^{-n}$, якщо A негативне, то $A_{об}$=q–q^{-n}-|A|. Звідси, треба віднімати з q негативне число і молодшу 1 (тобто q^{-n}).

Для довільної основи q це означає, що обернений код числа $A_{об}$ порозрядно формується від`ємом значення $A_{пр}$ в і-му розряді від (q-1).

Максимальне додатне число, що представляється при цьому дорівнює (1–2^n), або N_{max} = – 0,111...11 тоді додатній код буде $N_{дод}$ =1\00... 01. Якщо в наймолодший розряд числа N_{max} додати 1 то отримаємо: –1,000...00. Перетворивши це число в додатній код, отримаємо, $N_{äîä\,min} = -1.000..0$ тобто: –1 ≤ $N_{дод} \leq (1)$.

Контрольні питання:

1. Що таке система числення?
2. Які типи систем числення ви знаєте?
3. Що таке основа позиційної системи числення?
4. У чому полягає проблема вибору системи числення для подання чисел у пам'яті комп'ютера?
5. Яка система числення використовується для подання чисел у пам'яті комп'ютера? Чому?
6. Яким чином здійснюється перевід чисел, якщо основа нової системи числення дорівнює деякому степеню старої системи числення?
7. За яким правилом переводяться числа з десяткової системи числення?
8. За яким правилом переводяться числа в десяткову систему числення?
9. Сформулюйте правило перетворення від'ємних і додатних чисел у прямий код?
10. Сформулюйте правило перетворення від'ємних і додатних чисел у зворотний код?
11. Сформулюйте правило перетворення від'ємних і додатних чисел у доповнювальний код?
12. Як реалізується операція віднімання в цифрових машинах.
13. У яких випадках може бути втрата значимості результату при додаванні і відніманні чисел у машинних кодах?
14. Як визначити переповнення розрядної сітки?

ТЕМА 4. АРИФМЕТИЧНІ ДІЇ З ДВІЙКОВИМИ ЧИСЛАМИ

План

1. Операції додавання і віднімання чисел з фіксованою комою.
2. Операції додавання і віднімання чисел з плаваючою комою.
3. Алгоритми множення і ділення чисел з фіксованою комою.
4. Алгоритми множення і ділення чисел з плаваючою комою.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Операції додавання і віднімання чисел з фіксованою комою

Складання чисел, представлених у формі з фіксованою комою, на двійковому суматорі прямого коду

Двійковим суматором прямого коду (ДСПК) є суматор, в якому відсутній ланцюг порозрядного перенесення між старшим цифровим і знаковим розрядами (рис. 4.1).

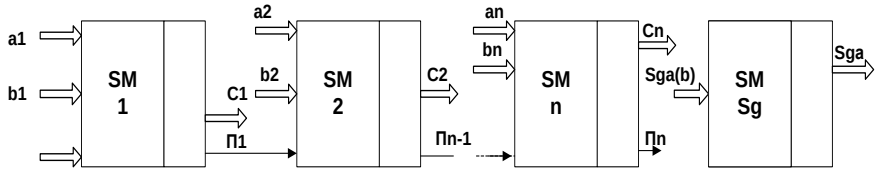


Рис. 4.1 – Двійковий суматор прямого коду (ДСПК)

На ДСПК можна складати тільки числа, що мають однакові знаки, а результат складання < 1 (для правильних дробів) або менше виразу 2^n-1 (для цілого числа). Аналіз показує, що такий суматор виконує не всі операції складання, а лише дві з чотирьох: $A+B$ і $-(A+B)$. Сигнал пере-повнення $П_n$ старшого n-розряду служить лише для індикації факту, що воно було, покладаючи подальші турботи на користувача або ЦА, а знак суми C визначається за знаком будь-якого операнда, наприклад Sg_A (Sg_B).

Нехай задані операнди:

$$A_{пр} = Sg_A \setminus a_1 a_2 \ldots a_n$$

$$B_{пр} = Sg_B \setminus b_1 b_2 \ldots b_n$$

де Sg_A і Sg_B – вміст знакових розрядів. Якщо $Sg_A=Sg_B$, то сума чисел матиме знак будь-якого з доданків, а цифрова її частина вийде порозрядним складанням операндів.

Приклади.

а) Скласти A = 0\1011 (+11); B = 0,0100 (+4). Тут Sg_A=0; Sg_B=0

+1011 (+11)

0100 (+4) C = 0\1111

0\1111 (+15)

б) Скласти A =–0\0101 (-5); B =–0\1001 (-9). Тут Sg_A=1; Sg_B=1

+0101 (–5)

1001 (–9) C = 1\1110

1\1110 (–14)

При складанні чисел на ДСПК можливі випадки, коли абсолютне значення суми операндів перевищує одиницю, тобто має місце переповнення розрядної сітки ЦА. Ознакою переповнення (γ=1) буде наявність одиниці перенесення $П_n$ із старшого розряду цифрової частини суматора. За цим сигналом повинні відбуватися автоматичний «останов» процесу рахування і коректування масштабних коефіцієнтів операндів A і B з розрахунком, щоб уникнути переповнення (наприклад зміна $K_ф$).

Складання чисел на двійковому суматорі доповняльного коду

Двійковим суматором доповняльного коду (ДСДК) називається суматор, що оперує числами, що представлені в доповняльному коді.

Основною особливістю ДСДК є наявність ланцюга перенесення ***одиниці переповнення*** $\boldsymbol{П_n}$ зі старшого розряду цифрової частини в знаковий розряд (рис. 4.2) і відсутність зворотнього зв'язку перенесення ***одиниці переповнення*** $\boldsymbol{П_{Sg}}$ із знакового розряду в наймолодший розряд числа.

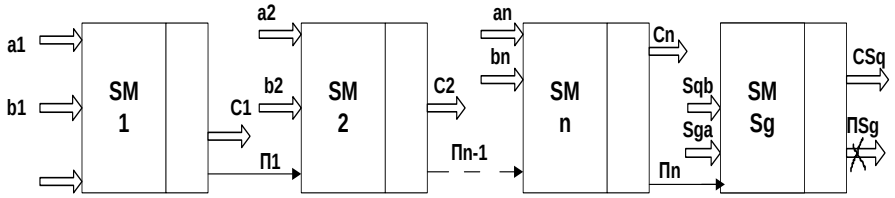


Рис. 4.2 –Двійковий суматор доповняльного коду (ДСДК)

Для визначення правил складання чисел в ДСДК розглянемо **наступну теорему**: **якщо результат суми доповняльних кодів чисел негативний, то він представлений у доповняльному коді.**

При доведенні теореми, припускаємо, що числа представлені у формі з фіксованою комою, що стоїть перед старшим розрядом. Розглянемо наступні випадки (всі числа є цілими):

Випадок 1. A>0, B>0, а A+B<1 числа позитивні і немає переповнення. Для цього випадку $A_{доп}$ =A, $B_{доп}$=B, то $A_{доп}$+$B_{доп}$=A+B=$[A+B]_{доп}$ – результат позитивний. Покажемо це на прикладі: A = +0,1010; B = +0,0100.

$$+\quad A_{доп} = 0\backslash,1010$$
$$B_{доп} = 0\backslash,0100 \quad C\text{=+0,1110.}$$
$$\overline{C = \ 0\backslash,1110}$$

Випадок 2. $A<0$, $B>0$, а $|A|>B$. Для цього випадку $A_{доп}=q+|-A|$, $B_{доп}=B$; тоді $A_{доп}+B_{доп}=q+|A|+B=[A+B]_{доп}$ – результат негативний.

A=–0,1011; B=+0,0100

$$+A_{доп}=1\backslash,0101$$

$$B_{доп}=0\backslash,0100 \qquad C\text{= - 0,0111.}$$

$$C_{доп}=\overline{1\backslash,1001} \qquad C_{пр}=1\backslash,0111$$

Отриманий результат негативний, це означає, що він представлений у доповняльному коді, і його необхідно перевести в прямий, щоб отримати дійсний результат. Переклад числа з доповняльного коду в прямий здійснюємо за наступним алгоритмом: усі розряди числа (окрім знакових) інвертуються (тобто береться обернений код результату) і в молодший розряд додається 1. Знак суми C зберігається.

Випадок 3. $A<0$, $B>0$, а $|A|<B$. для цього випадку $A_{доп}=q+|A|$, $B_{доп}=B$; тоді $A_{доп}+B_{доп}=q+|A|+B$. Оскільки значення цієї суми більше q, то з'являється одиниця перенесення в знаковий розряд, що дорівнює вилученню з суми q одиниці, тобто результат: $A_{доп}+B_{доп}=A+B=C_{пр}$.

Приклад. А =–0,0100; В =+0,1011.

$$+A_{доп}=1\backslash,1100$$

$$B_{доп}=0\backslash,1011 \quad C\text{= + 0,0111.}$$

$$\overline{C_{пр}=0\backslash,0111}$$

Випадок 4. $A<0$, $B<0$, а $|A+B|<1$. Для цього випадку $A_{доп}=q+(-A)$; $B_{доп}=q+(-B)$, тоді $A_{доп}+B_{доп}=q+(-A)+q+(-B)=[A+B]_{доп}$ – результат негативний і з'явиться одиниця перенесення із знакового розряду.

Приклад. А = –0,0100; В = –0,1011.

$$\begin{array}{l} A_{доп}=1\backslash,1100 \\ \underline{B_{доп}=1\backslash,0101} \\ C_{доп}=1\backslash,0001 \end{array} \quad C_{пр}=1\backslash,1111.\ \text{C= -0,1111}$$

Висновок. Теорема справедлива для всіх випадків, в яких не виникає переповнення розрядної сітки, що дозволяє складати в ДСДК машинні зображення чисел за правилами двійкової арифметики. Приклади:

Випадок 1. $A>0$, $B>0$, $C<1$.

а) $A_{(10)}$ = +0,372914$_{(10)пр}$; $B_{(10)}$ = +0,564019$_{пр}$

$$A_{доп} = 0\backslash,372914$$

$$+\underline{B_{доп} = 0\backslash,564019}$$

$$C_{доп} = 0\backslash,936933 = C$$

б) Для цілих чисел: $A_{пр}$= 0\1001110101 (+629) $B_{пр}$=0\0101110011(+371)

$$A_{доп} = 0\backslash1001110101$$

$$+\underline{B_{доп} = 0\backslash0101110011}$$

$$C_{доп} = 0\backslash1111101000 = C_{пр}\ 1000_{(10)}$$

Перевірка: 629+371=1000

Випадок 2. $A>0$, $B>0$, $C\geq 1$ (переповнення розрядової сітки):

$A_{пр}$=0\672914$_{(10)}$, $B_{пр}$=0\564019$_{(10)}$.

$A_{доп}$ = 0\672914

$+B_{доп}$= 0\564019

$C_{доп}$=1\236933→0\1236933 (проведено зсув вправо мантиси на один розряд та відновлено знак суми)

Поява в знаковому розряді одиниці позитивного переповнення говорить про переривання роботи для зміни $K_Ф$ або зсув мантиси вправо на один розряд, а до порядку результату додається одиниця (тобто проводиться нормалізація результату).

$A_{пр}$ = 0\1101110101 $B_{пр}$ = 0\0101110011

$A_{доп}$ = 0\1101110101

+ $B_{доп}$ = 0\0101110011

$C_{доп}$ = 1\0011101000 (далі зсув вправо на один розряд)

$C_{пр}$ = 0\10011101000.

Перевірка: +885 + 371 =1256

Відбулося переповнення (γ=1), тобто в знаковому розряді Sg_{C0} з'явилася ознака (1). Що це? ЦА не може розібрати ситуацію, це знак «мінус» суми C або ознака переповнення. Для вирішення цього треба в поле знаків додати ще один розряд (використовується ***модифікований*** доповняльний код, див. гл. 9).

Зсув вправо (вліво) числа зменшує (збільшує) його на порядок: тобто змінеться $K_ф$ (для ФФК) або порядок P (для ФРК) і цю зміну треба відкорегувати, щоб значення числа не змінилось.

Випадок 3. $A>0$, $B<0$, $|A|>|B|$: тобто $C=A+q+B$ оскільки $C>q$, то q повинне бути відібране, що робиться шляхом відкидання сигналу перенесення, що виникає в знаковому розряді.

$A_{(10)пр}$=+0.732904; $B_{(10)пр}$= -0.042703(зрівняно кількість розрядів A і B), $B_{(10)доп}$=9\,957297

$[A]_{доп}$ = 0\,732904

\+ $[B]_{доп}$ = 9\,957297

$[C]_{доп}$ =***1***0\,690201; $C_{пр}$ = +0,690201. При цьому в ДСДК переповнення ***1*** розряду SgC не використовується (відкидається).

При визначенні доповняльного коду $B_{(10)доп}$, спочатку провели вирівнювання

розрядності чисел, додавши 0 в старший розряд числа $B_{пр}$, а потім віднімали з $(q-1)=9$ значення по кожному розряду і отримали обернений код, потім до його молодшого розряду додали 1.

Розглянемо ще приклад:

$A_{(2)пр}$=0\1101110101 (+885); $B_{(2)пр}$ =1\0110110101 (–437).

$+ A_{доп}$ = 0\1101110101
$B_{доп}$ = 1\1001001011
$C_{доп}$=***1***0\0111000000
$C_{пр}$ = 0\0111000000 (+448)

Одиниця переповнення ***1*** відкидається (бо ДСДК), знак результату C позитивний і означає прямий код.

Випадок 4. $A>0$, $B<0$, $|A|<|B|$, $C=A+q+B$, оскільки $C<0$, то $C=q+(A+B)$.

а) $A_{(10)пр}$ = +0,324761 $B_{(10)пр}$ = -0,560129
$+ A_{доп}$ = 0\,324761
$B_{доп}$ = 9\,439871
$C_{доп}$ = 9\,764632 ; $C_{пр}$ = 9\,235368

б) $A_{(2)пр}$ = 0\0110011001 (+409); $B_{(2)пр}$ = 1\1010011101 (–669).
$+ A_{доп}$= 0\0110011001
$B_{доп}$ = 1\0101100011
$C_{доп}$ = 1\1011111100 – Необхідно перетворення в прямий код, тому що результат суми від'ємний.
Соб =1\0100000011 – Інверсний (обернений) код результату.
Далі в молодший розряд додаємо 1, тобто +0\0000000001. Тоді Спр=1\0100000100 = –260(10); (Формат числа надан в ФФЗ. При представленні його в ФРК необхідна буде нормалізація Спр: представлення мантиси у вигляді правильного дробу з зсувом вліво на 10розрядів і відповідним корегуванням значення порядку Р на +10, а потім зсув вліво на 1 розряд, бо маємо 0 після коми, а далі – корекція порядку на -1). Спр = 1\100000100. Перевірка: –669+409 =–260

Випадок 5. $A<0$, $B<0$, $|A|+|B|<1$, $C=q+A+q+B<0$. При цьому, одна основа повинна бути відкинута за рахунок втрати сигналу перенесення в знаковому розряді. Тоді $C=q+(A+B)$.

$A_{(10)пр}$ = -0,432096 $B_{(10)пр}$ = -0,392671
$+ A_{доп}$ = 9\,567904
$B_{доп}$ = 9\,607329
$C_{доп}$ =***1***9\,175233 $C_{пр}$ = 9\,824767

Переповнення ***1*** в знаковому розряді пропадає, оскільки використовується ДСДК.

Випадок 6. $A<0$, $B<0$, але $|A|+|B|>0$.

Це також випадок переповнення розрядної сітки, але на відміну від випадку 2 переповнення тут негативне.

$A_{(10)пр}$ = -0,610892$_{(10)}$; $B_{(10)пр}$ = -0,843507$_{(10)}$

$A_{доп}$ = 9\,389108
+ $B_{доп}$ = 9\,156493
$C_{доп}$ =18\,545601

Поява цифри 8 в знаковому розряді говорить про те, що відбулося переповнення Sg_{C0}, необхідний зсув вправо з відновленням знаку і перетворення результату в прямий код.

$C_{доп}$=18\,545601; після зсуву $C_{доп}$=9\,8545601 (проведено зсув на 1 розряд вправо і відновлення знаку), $C_{пр}$=9\,1454399.

Розглянемо приклад в двійковій системі числення.

$A_{пр}$ = -1101110100; $B_{пр}$ = -1110111010

$A_{доп}$ = 1\0010001100
+ $B_{доп}$ = 1\0001000110
$C_{доп}$ =10\0011010010

Було переповнення. Необхідний зсув вправо, для відновлення знаку з подальшим перетворенням результату в прямий код.

$C_{доп}$ =10\0011010010=1\00011010010;

$C_{пр}$=1\11100101110. Перевірка: –884–954= –1838.

У звичайному доповняльному коді є проблема із зображенням числа «–1». Тут розглядають два варіанти:

1. Виробляти сигнал переповнення, якщо в знаковому розряді (q–1), а в цифрових розрядах є нулі тобто $A_{доп}$= 1.0000…0.

2. Вважати –1 допустимим значенням, але при цьому її зображення в прямому коді співпадатиме із зображенням у доповняльному коді.

$-1_{(10)доп} = 9.00000…0 = 9.99999…9 + 0.0000…01 = 9.00000…0 = -1_{(10)пр}$

$-1_{(2)доп} = 1.00000…0 = 1.11111…1 + 0.0000…01 = -1_{(2)пр}$

Щоб відрізняти –1 від +1, використовують ***модифікований доповняльний код***

Складання чисел на суматорі оберненого коду

Двійковим суматором оберненого коду (ДСОК) є суматор, що оперує з числами в оберненому коді.

Структурна схема ДСОК приведена на рис. 4.3.

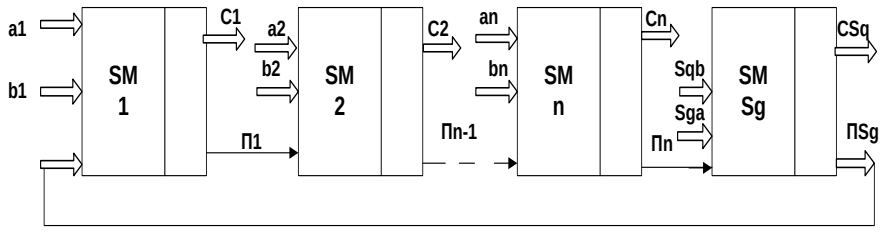


Рис.4.3. – Двійковий суматор оберненого коду (ДСОК)

Основною особливістю ДСОК є наявність ланцюга перенесення ***одиниці переповнення $П_n$*** із старшого розряду цифрової частини в знаковий розряд (рис. 4.3) і наявність зворотнього зв'язку перенесення ***одиниці пере-повнення $П_{Sg}$*** із

старшого знакового розряду Sg_1 в молодший розряд числа.

ДСОК має: n суматорів (по кількості розрядів мантиси); суматор знакових розрядів; a_i, b_i – цифрові розряди числа в машинному коді; перенесення із старших розрядів мантиси в знаковий розряд; нарешті, обернене перенесення із старшого знакового розряду в молодший розряд цифрової частини числа. Щоб сформувати правила складання чисел в обернених кодах на ДСОК розглянемо теорему.

Теорема: сума обернених кодів чисел є обернений код результату.

При доведенні цієї теореми припускаємо, що числа представлені у формі з фіксованою комою, що стоїть перед старшим цифровим розрядом.

Розглянемо наступні випадки складання на суматорі ДСОК.

Випадок 1. $A>0$, $B>0$, $A+B<1$ (тобто обидва числа позитивні).

Тоді $A_{об}+B_{об}=[A+B]_{об}=A+B$, тобто складання проводиться в прямих кодах.

Приклад: A = 0\0101 (+5)

$\underline{B = 0\backslash 0111}$ (+7)

C = 0\1100 (+12)

Випадок 2. $A<0$, $B>0$, $|A|>B$, $A_{об}=q-q^{-n}+A$, $B_{об}=B$, тоді $A_{об}+B_{об}=q-q^{-n}+A+B=[A+B]_{об}$, тобто результат негативний і в оберненому коді, треба повертатися до $C_{пр}$ шляхом інверсії розрядів числа.

Приклад: A = –0\1011 (–11), B = 0\0111 (+7).

$A_{об}$=1\0100

$+B_{об}=\underline{0\backslash 0111}$

$C_{об}$=1\1011 $C_{пр}$=1\0100 (–4)

Випадок 3. $A<0$, $B>0$, $A<B$, тут $A_{об}$ = q–q^{-n}+A. Тоді $A_{об}+B_{об}=q-q^{-n}+A+B$. Оскільки сума ($A+B$) позитивна, то права частина цього виразу стає більше q, що викликає появу одиниці перенесення із знакового розряду в молодший розряд числа (величина перенесення при цьому, рівна $q-q^{-n}$), тоді $A_{об}+B_{об}=A+B$. Результат позитивний, це і є $C_{пр}$.

Приклад: A = –0\0101 (–5); B = 0\0111 (+7).

$A_{об}$ = 1\1010

$+B_{об}$ = $\underline{0\backslash 0111}$

$C_{об}$ = 10\0001

$^{+}\underline{\mapsto\rightarrow\rightarrow 1}$

$C_{пр}$=0\0010(+2). Оскільки $B>|A|$, то їх алгебраїчна різниця позитивна.

Випадок 4. $A<0$, $B<$O, $|A+B|<1$, тут $A_{об}=q-q^{-n}+A$, $B_{об}=q-q^{-n}+B$. Тоді $A_{об}+B_{об}=q-q^{-n}+A+q-q^{-n}+B$. Тут з'являється одиниця перенесення, що дорівнює вилученню з суми $-q^{-n}$, тобто $A_{об}+B_{об}=q-q^{-n}+A+B=[A+B]_{об}$

Приклад: А = –0,0101 ; В = –0,1000.

$A_{об}$=1\,1010
+$B_{об}$=1\,0111
11\,0001
$^{+}\mapsto\rightarrow\rightarrow$1
Соб=1\,0010; Спр= 1\,1101 (-0,8125).

Модифіковані бінарні коди.

Переповнення розрядної сітки

Ми не раз спостерігали, як при складанні чисел з однаковими знаками, представлених у формі з фіксованою комою (ФФК), може виникнути переповнення розрядної сітки. При цьому, щоб уникнути спотворення результату, автомат повинен фіксувати переповнення і відповідно реагувати на це.

Переповнення при складанні прямих кодів

Ознакою переповнення розрядної сітки суматора прямого коду є поява одиниці перенесення із старшого розряду цифрової частини числа.

Приклади. Суматор ДСПК.

а) A=+1010, B=+1101.

$A_{пр}$= 0\1010 (+10)
+$B_{пр}$= 0\1101 (+13)

$C_{пр}$= 0\0111 (+7≠ +23). Отримано спотворений (невірний) результат із-за втрати перенесення П зі старшого цифрового розряду.

б) A=–1100, B= –1010.

$A_{пр}$= 1\1100
+$B_{пр}$= 1\1010

$C_{пр}$= 1\0110 (-6≠-22). Тут теж отриманий невірний результат із-за втрати одиниці перенесення.

Переповнення при складанні доповняльних кодів

Ознакою переповнення розрядної сітки суматора доповняльного коду при складанні позитивних чисел є негативний знак результату, а при складанні негативних чисел – позитивний знак результату.

Приклади:

а) А= +0\1011; В= +0\1010 (перевірку отриманого результату проведіть самостійно).

$A_{доп}$=0\1011
+$B_{доп}$=0\1010
$C_{доп}$≠1\0101

б) A= –0\1011; B= –0\1001 (перевірку отриманого результату проведіть самостійно).

$A_{доп}$= 1\0101
$+B_{доп}$= 1\0111
$C_{пр}$= 0\1100

Переповнення при складанні в обернених кодах

Ознакою переповнення розрядної сітки суматора оберненого коду є знак результату, протилежний знакам операндів.

Приклад:

1) A=0\0111; B=0\1101.

$A_{об}$=0\0111
$+B_{об}$=0\1101
$C_{об}$=1\0100

Знак числа спотворений (невірний):

2) A= -0\0110; B= -0\1101.

$A_{об}$=1\1001
$B_{об}$=1\0010
$C_{пр}$≠0\1011

Знак числа спотворений (невірний)

Для усунення втрат від неврахування перенесення одиниці та формування стану переповнення, у складі ЦА повинні бути перед-бачені додаткові засоби, що дозволяють фіксувати і аналізувати пере-повнення за їх ознаками з метою усунення спотворення результату.

Щоб фіксувати переповнення розрядної сітки в ДСПК, ДСДК, ДСОК, вводять допоміжний розряд в знакову частину зображення числа, який називають розрядом переповнення числа. Таке представлення числа називається модифікованим (див. рис. 4.4).

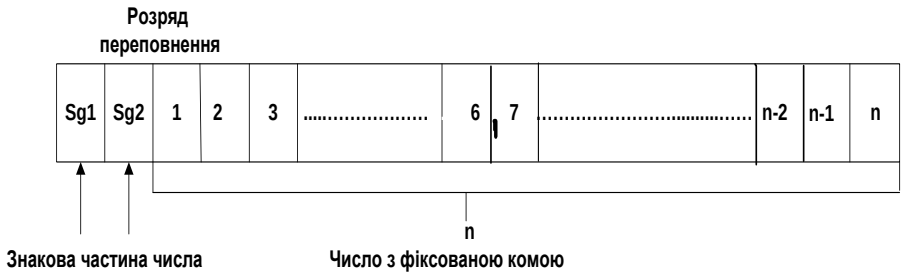
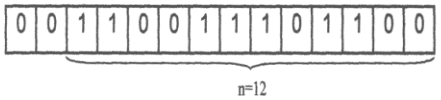


Рис. 4.4 – Модифікований код числа для ФФК

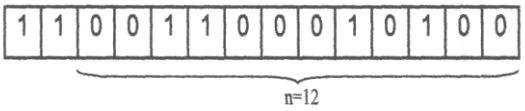
Приклади:

а) $[A]_{пр}$=00\1100111011, тоді, оскільки A>0 маємо: $[A]_{М.доп}$=00\1100111011



б) $A_{пр}$ = –0\1100111011, оскільки A<0, то $A_{м.доп}$ =100+(–$A_{пр}$)=q^2–A
_100\0000000000

0\1100111011
11\0011000101



ПРАВИЛО: для переводу бінарного числа з прямого коду в модифіковний доповняльний код, аналізують знак числа. Якщо в знаковому розряді знаходиться мінус (одиниця), то в два знакових розряди записують одиниці, а всі цифри числа інвертують (одиниці заміняють нулями, а нулі – одиницями), потім до останньої молодшій цифрі (молодшого розряду) числа додають одиницю. Якщо в знаковому розряді стоїть нуль, то додають ще один, а число не змінюють.

ПРАВИЛО: для переводу бінарного числа з прямого коду в модифіковний обернений код, аналізують знак числа. Якщо в знаковому розряді знаходиться мінус (одиниця), то в два знакових розряди записують одиниці, а всі цифри числа інвертують (одиниці заміняють нулями, а нулі – одиницями). Коли в знаковому розряді стоїть нуль, то додають ще один, а число не змінюють.

При виконанні операцій алгебраїчного додавання або вирахування два знакових розряди приймають участь в операції, як рівноправні цифрові розряди. Після виконання операцій, стан знакових розрядів (знак результату) встановлює **лівий знаковий розряд**, а наявність чи відсутність переповнення – **правий знаковий розряд**.

Знакові розряди можуть мати наступні комбінації станів при будь-яких значеннях операндів A і B:

00 – результат числа позитивний, переповнення немає;

01 – результат числа позитивний, було переповнення;

11 – результат числа негативний, переповнення немає;

10 – результат числа негативний , було переповнення.

У машинному представленні чисел, представлених в модифікованому коді, використовується (n+2) розрядів: з них два знакових, кома опускається, встановлюється постійний коефіцієнт формату K_ϕ. Сигнал переповнення γ виробляється по наступній математичній моделі (тут знак $\wedge$ позначає логічну операцію множення «і», а риска над символом Sg – логічну операцію інверсії

$\gamma = 1$, якщо $\begin{cases} Sg1 \wedge \bar{S}g2 = 1 \\ Sg1 \wedge Sg2 = 1, \end{cases}$ і $\gamma = 0$ в інших випадках.

Приклади: а) A= 0\1011 (+11), B= 0\1010 (+10).

$A^{м}_{д}$=00\1011

+$B^{м}_{д}$=00\1010

$C^{м}_{д}$= 0**1**\0101=00\10101 (+21)

Тут ознакою переповнення є **1** в знаковому розряді. За цією ознакою проводиться зрушення числа вправо з одночасним збільшенням коефіцієнта формату $K_ф$ на 1 і відновленням Sg_1 (по значенню Sg_2)обох розрядів Sg_1Sg_2\, тобто знаку 00 результату С числа, далі перевіряється нормалізація числа (наявність 1 після коми в числі).

б) A= –0\1011 (-11), B= –0\1001(-9). Перетворимо в машинну форму: $A^м_{пр}$=11\1011, $B^м_{пр}$=11\1001. Оскільки обидва числа негативні і операція складання виконується на ДСДК, то в суматор числа поступають в доповняльному коді.

$A^м_д$=11\0101
$B^м_д$=11\0111
$C^м_д$=10\1100 Було переповнення. Потрібен зсув вправо і відновлення знака.
$C^м_д$=11\01100 результату і перетворення його в прямий код.
$C^м_{пр}$=11\10011
\+ 1, тоді $C_{пр}$= 11\10100 (–20).

Аналіз розрядів знаку результату (10 – див. третю строку) показав, необхідність відновлення його (до 11), зрушуючи число вправо на один розряд, з одночасним збільшенням коефіцієнта формату $K_ф$ на 1. Результат негативний, і тому представлений у доповняльному коді. Тому результат перетвориться в істиний (прямий код) через інвертування (обернений код) і з додаванням +1 в молодший розряд.

Віднімання чисел з фіксованою комою.

Для апаратної реалізації віднімання чисел використовують сумування з додатковими кодами.

Додавання і віднімання чисел в зворотному і додатковому кодах виконується з використанням звичайного правила арифметичного додавання багаторозрядних чисел. Загальною для цих кодів особливістю (і дуже зручною особливістю) є лише те, що при порозрядному додаванні чисел розряди, що зображають знаки чисел розглядаються як рівноправні розряди двійкового числа, які додаваються один з одним і з одиницею перенесення з попереднього розряду числа за звичайними правилами арифметики.

Відмінності ж зворотного і додаткового кодів пов'язані з тим, що робиться з одиницею перенесення із старшого розряду (що зображає , як неодноразово мовилося, знак числа)

Операція віднімання в обчислювальних пристроях завжди заміняється операцією додавання з числом зворотного знаку (доповненням модулю від'ємника).

2. Операції додавання і віднімання чисел з плаваючою комою

Модифіковане складання чисел у форматі з рухомою крапкою (комою)

Числа, представлені у форматі з рухомою крапкою (комою) – ФРК, мають дві часті: мантису і порядок. Тому, операція додавання (складання) виконується окремо над мантисою і над порядком. Отже, в ЦА може бути два суматори: для мантиси і для порядку.

Для чисел з рухомою крапкою справедлива умова нормалізації:

$$q^{-1} \le |m_A| < 1, \qquad (4.1)$$

де q – основа системи числення; m_A – мантиси числа.

Це **нормалізоване представлення числа**, яке вимагає, щоб в старшому розряді мантиси 2-го числа була одиниця. Для двійкової системи це означатиме, що мантиса завжди знаходиться в межах:

$$0{,}5 \le |m_A| < 1, \qquad (4.2)$$

При виконанні автоматом операцій над числами, нормалізують як вхідні доданки A і B, так і вихідний результат C.

Операція нормалізації числа складається з умови нормалізації (4.1) і здійснюється методом зрушення мантиси числа в ту або іншу сторону.

Зрушення можуть проводитися вліво або управо в межах розрядної сітки ЦА за правилами представлених моделлю таблиці 4.1.

Таблиця 4.1

Зсув звичайних кодів числа (один розряд знаку)

Початкове число	Зсув вліво на один розряд (від порядку вираховується 1)	Зсув вправо на один розряд (до порядку додається 1)
$0/a_1\ a_2 \ldots\ a_n$ $1/a_1\ a_2 \ldots\ a_n$	$a_1/a_2 \ldots\ a_n\ 0$ $a_1/a_2 \ldots\ a_n \varepsilon$	$0/0\ a_1\ a_2 \ldots\ a_n$ $0/1\ a_1\ a_2 \ldots\ a_n$

Величина ε залежить від коду. Для доповняльного коду ε=0, для оберненого коду ε=1. При складанні чисел результат може вийти з нормалізації як зліва, так і справа (див. табл. 4.1, 4.2).

Ознакою порушення нормалізації числа справа γ=1 (коли результат має переповнення) є наявність різнойменних комбінацій в знакових розрядах сумматора:

$$\gamma = 1, \text{ якщо } \begin{cases} Sg1 \wedge \overline{Sg2} = 1 \\ \overline{Sg1} \wedge Sg2 = 1, \text{ і } \gamma = 0 \text{ у решті випадків.} \end{cases}$$

Таблиця 4.2

Зсув модифікованих кодів числа (два розряди знаку)

Початкове число	Зсув вліво на один розряд (від порядку вираховується 1)	Зсув вправо на один розряд (до порядку додається 1)
$00/\alpha_1,\alpha_2,...,\alpha_n$	$0\alpha_1/\alpha_2,...,\alpha_n\,0$	$00/0,\alpha_1,\alpha_2,...,\alpha_n$
$01/\alpha_1,\alpha_2,...,\alpha_n$	$1\alpha_1/\alpha_2,...,\alpha_n\,0$	$00/1,\alpha_1,\alpha_2,...,\alpha_n$
$10/\alpha_1,\alpha_2,...,\alpha_n$	$0\alpha_1/\alpha_2,...,\alpha_n\,\varepsilon$	$11/0,\alpha_1,\alpha_2,...,\alpha_n$
$11/\alpha_1,\alpha_2,...,\alpha_n$	$1\alpha_1/\alpha_2,...,\alpha_n\varepsilon$	$11/1,\alpha_1,\alpha_2,...,\alpha_n$

(де **γ** – ознака порушення нормалізації числа справа, вказує на необхідність зсуву числа вправо на один розряд для відновлення знаку числа).

Ознакою порушення нормалізації числа зліва δ=1 (коли результат по абсолютній величині виявляється менше 1/q) є наявність однакових комбінацій в розряді переповнення і старшому розряді (R1) цифрової частини суматора:

$$\delta=1,\ \text{якщо}\ \begin{cases} Sg2 \wedge R1 = 1 \\ \overline{Sg}2 \wedge \overline{R}1 = 1, \end{cases}$$ і δ=0 у решті випадків,

(де δ – ознака порушення нормалізації, що вказує на необхідність зсуву числа вліво на один розряд). Зазвичай $Sg1$ (або $Sg0$)– старший, а $Sg2$ (або $Sg1$)– молодший знакові розряди.

Таким чином, операція нормалізації отриманого числа (суми двох операндів) складатиметься з сукупності перевірки наявності ознак порушення γ і δ та зсувів (вправо, вліво) числа.

Отже, розглянемо складання чисел $A=m_Ap_A$ и $B=m_Bp_B$, що мають однаковий порядок $p_A=p_B$. Обидві мантиси задовольняють умові нормалізації. Складання мантиси здійснюють на суматорі ДСДК або ДСОК за правилом складання чисел представлених у формі з фіксованою комою. Якщо після складання мантиса результату задовольняє умові нормалізації (тобто δ=0, γ=0), то до цього результату приписується порядок будь-якого з операндів. Інакше відбувається нормалізація числа.

Приклад 1. Знайти суму чисел: $A=0{,}1000\cdot2^{-3}$ та $B=-0{,}1011\cdot2^{-3}$.

Мантиси і порядок обробляються на ДСДК у ФРК.

$[m_A]_\partial$=00/,1000 $[P_A]_\partial$=11/101

$+[m_B]_\partial$=11/,0101 $[Pв]_\partial$=11/101

$[m_c]_\partial$=11/,1101

Тут $Sg2\&R1=1$, тобто δ=1, γ=0. Це означає, що необхідний зсув мантиси $[m_c]_\partial$ вліво на 1 розряд.

$[m'_c]$=11/,1010 Перевіряємо. Знову δ=1, γ=0 – необхідний ще зсув вліво на 1 розряд.

$[m''_c]_\partial$= 11/,0100. Перевіряємо, все гаразд δ=0, γ=0.

Одночасно зі зрушенням потрібна корекція порядку на мінус $2_{(10)}=11/010_{(2)}$. (що рівнозначно збільшенню порядку на дві одиниці – у доповняльному коді 11/110).

$_{+}[P_c]_д$=11/101

$\underline{[\Delta P_c]_д=11/110}$ $[\Delta P_c]_{пр}$=11/010

$[P''_c]_д$=11/011 $P_{C\,пр}$=–11/101 (тобто P= 2^{-5})

тоді результат (число) дорівнює:

$[m''_c]_д$= 11/,0100

$[m''_c]_{об}$=11/,1011

\+ 1

11/,1100.

Відповідь: $m_{C\ пр}$=11/,1100. З урахуванням порядку запис для 16-ти розрядної ЕОМ $C_{пр}$= 11/,11000000/11/0101/.

Приклад 2. Знайти суму чисел A=–0,1100·2^4 та B=–0,1000·2^4. Мантиси і порядок обробляються на ДСОК у ФРК (шість розрядів мантиси і чотири для порядку). Порядки $[P_A]=[P_B]=[P_C]_{об}$= 0,100 (оскільки позитивні і рівні обидва порядки операндів *A* і *B*).

$_{+}[m_A]_{об}$=11/,0011

$\underline{[m_B]_{об}=11/,0111}$

$[m_C]_{об}$=110/,1010

\+ 1

10/,1011 (δ=0, γ=1)

Порушення нормалізації справа (тобто γ=1 означає, що було переповнення). Число зсуваємо вправо на один розряд, відновлюючи знак 11 і одночасно збільшуючи порядок *P* на 1.

Тобто $[m_C]_{об}$=11/,0101, тепер δ=0, γ=0. Одночасно коректуємо і порядок:

$_{+}[Pc]$= 00/100

$\underline{[\Delta Pc]=00/001}$

00/101 (+5), тоді відповідь: –0,1010·2^{+5} (Самостійно зробіть запис $C_{пр}$ для 16-ти розрядної ЕОМ).

Складання чисел при різних значеннях порядків

Для операції складання чисел необхідною умовою є зіставлення вагів розрядів операндів один одному. Тому спочатку потрібно вирівняти порядки, що спричинить за собою тимчасове порушення нормалізації одного з доданків (*A* або *B*).

Вирівнювання порядків означає, що порядок меншого числа треба збільшити на величину $\Delta P=P_A-P_B$, що означає зсув мантиси меншого числа вправо на кількість розрядів, рівне $P_A \geq P_B$. Тому, цифровий автомат повинен спочатку розпізнати, який порядок з двох чисел є меншим. На це вкаже знак ΔP. Якщо $P_A \geq P_B$ – знак позитивний і навпаки, якщо знак негативний, то $P_A < P_B$. Цю операцію іноді виконують шляхом порівняння чисел.

Алгоритм операції складання у форматі з рухомою крапкою (комою)

Отже, операція складання (віднімання) виконується в наступній послідовності.

1. Перевести операнди в двійкові доповняльні (або обернені) модифіковані коди, перевіривши перед цим нормалізацію початкових чисел.

2. Визначити різницю порядків $\Delta P = P_A - P_B$

3. Якщо $\Delta P>0$, зсунути мантису числа B на ΔP розрядів вправо; якщо $\Delta P<0$, зсунути мантису числа A на ΔP розрядів вправо; якщо $\Delta P=0$, мантиси не зсуваються (розряди виходять за межі розрядної сітки мантиси втрачаються).

4. Виконати операцію алгебраїчного складання (віднімання) над мантисами. Алгебраїчне складання виконується за наступним правилом. Аналізується знак числа:

- якщо знак позитивний (00), то в суматор мантиси число поступає в прямому коді;

- якщо знак негативний (11), то в суматор мантиси число поступає в доповняльному (оберненому) коді;

Складання проводиться порозрядно, за правилами бінарної арифметики, з перенесенням одиниці переповнення в старший розряд. Знакові розряди також беруть участь в складанні. Одиниця переповнення в старшому знаковому розряді для суматорів доповняльного коду пропадає, для суматорів оберненого коду по оберненому зв'язку додається до молодшого розряду мантиси.

До представлення результату в прямому коді, після складання, аналізується знак результату:

- якщо знак 01(або 10), то знак спочатку відновлюється шляхом зсуву мантиси вправо на один розряд з одночасним збільшенням порядку на 1;

- якщо знак 00, то результат- число позитивне, представлено в прямому коді;

- якщо знак 11, то результат - число негативне і представлено в оберненому (доповняльному) коді.

В цьому випадку, потрібне перетворення числа із оберненого (доповняльного) в прямий код, шляхом інвертування числа (окрім знаку) для ДСОК і інвертування числа (окрім знаку) з додаванням 1 до молодшого розряду мантиси для ДСДК. Результату встановлюється порядок більшого за модулем числа.

5. Проводиться перевірка числа на нормалізацію (для нормалізованих чисел, після знаку мантиси повинна стояти одиниця). Якщо після знаку мантиси стоїть нуль, то число зсувається вліво на один розряд (з одночасним зменшенням порядку на 1). Перевіряється нормалізація (перевіряємо **ознаки порушення γ і δ)**. Цикл може повторюватися до досягнення умов нормалізації **γ=0 і δ=0)**. За цим алгоритмом виконують операції складання і віднімання АЛП.

Приклад. Скласти A=+0,1011·2^{-2} и B = –0,1001·2^{-3}

Числа задані в природному вигляді. В АЛП використовується два суматори оберненого коду: суматор мантис (шість розрядів, включаючи знак); суматор порядків (разом із знаком – чотири розряди).

РІШЕННЯ.

1. Враховуючи, що числа задані вже в нормалізованому вигляді (після коми стоїть 1), представимо їх в машинному вигляді у форматі з рухомою крапкою (модифікований код): $A^{м}_{пр}$=00.1011.11.10; $B^{м}_{пр}$=11.1001.11.11.

2. Переводимо мантиси чисел в обернені коди: $m_{Аоб}$=00.1011;

$m_{Воб}$=11.0110.

3. Для вирівнювання порядків чисел, а значить, і їх вагових розрядів, необхідно з'ясувати, яке із заданих чисел підлягає тимчасовій денормалізації. Для цього автоматом визначається різниця порядків чисел: $\Delta P=P_A-P_B$. Переведемо порядки чисел в обернені коди.

$P_{Аоб}$=11.01 ; $P_{Воб}$=11.00. Тоді, $\Delta P=P_{Аоб}-P_{Воб}$ =11.01–11.00

Замінимо операцію віднімання на операцію складання.

$\Delta P=P_{Аоб}-P_{Воб} = P_{Аоб} + \overline{P}_{Впр}$ =11.01 + 00.11 (де $P_{Впр}$ – інверсний код $P_{Воб.}$, включаючи і знакову частину).

Виконаємо додавання.

11.01
+<u>00.11</u>
100.00
<u>+⊢→⟶1</u>
00.01

Величина ΔP позитивна, тому $P_A > P_B$. Треба зсувати мантису числа вправо на ΔP розрядів, тобто на один розряд, збільшивши одночасно порядок на 1.

4. Зсунемо мантису числа B вправо на один розряд і складемо їх значення на ДСОК.

5. Перевіримо умову нормалізації мантиси результату справа, зліва.

$m_{Аоб}$=00.1011 (тут δ=0, γ=0)
$m_{Воб}$=11.0110 (тут δ=0, γ=1, необхідно зробити зсув $m_{Воб}$ вправо)
$m_{Воб}$=11.1011
+$m_{Аоб}$=<u>00.1011</u>
$m_{Соб}$=100.0110
<u>⊢→⟶⟶ 1</u>
00.0111 (тут δ=1, γ=0; необхідно зробити зсув вліво)
$m_{Соб}$=00.1110 і тому робимо корекцію порядку на ΔP=–1
$P_{Аоб}$=11.01
+<u>$\Delta P_{об}$= 11.10</u> (ΔP представлено в оберненому коді)
110.11
<u>+⊢→⟶1</u>
$P_{Соб}$=11.00
Результат $P_{Соб}$ отриманий в оберненому коді, тобто знак негативний (11). Визначаємо прямий код порядку і записуємо результат $C_{пр}$=00.1110·2^{-3}.
Відповідь: $C_{пр}$=00.1110.11.11

Приклад. Скласти на ДСДК числа $A_{пр}$= +5, $B_{пр}$= –3, представлених в десятковій системі числення. $A_д$=00/5; $B_д$=99/7. Проведемо складання $A_д+B_д$=00/5+99/7=*1*00/2=00/2 ($+2_{(10)}$). Одиниця (*1*) пропадає, оскільки складання йде на ДСДК.

Віднімання чисел у форматі з рухомою крапкою (комою).

Додавання і віднімання чисел з рухомою комою виконується у декілька етапів.

1. Вирівнювання порядків;

Як відомо, реальна величина (вага) Ni одиниці і-го розряду мантиси визначається не тільки позицією даного розряду, але й порядком р числа, тобто

Ni = d p-i,

де і – номер позиції, рахуючи вправо від коми.

При додаванні (відніманні) необхідно, щоб ваги однойменних розрядів мантис чисел були однаковими. Для цього мантиси зсувають одна щодо одної так, щоб їх порядки вирівнялися. Щоб при вирівнюванні порядків не отримати мантиси більшої за одиницю, їх потрібно вирівнювати від меншого до більшого порядку. Мантиса з меншим порядком зсувається вправо (у бік молодших розрядів) на кількість розрядів, що дорівнює різниці порядків і одночасно коректується порядок (збільшується до значення спільного порядку).

1. Додавання мантис;

Додавання мантис із вирівненими порядками виконується згідно правил додавання чисел з фіксованою комою. Зазвичай використовується модифікований доповнювальний код. (Пригадайте правила подання від'ємних чисел з використанням доповнювального або оберненого модифікованого коду).

Переведення результату додавання мантис у прямий код.

У результаті додавання мантис може виникнути один з трьох випадків:

а). Виникає переповнення розрядної сітки комп'ютера (порушення нормалізації вправо). Ознакою переповнення є невизначенність знакових розрядів, коли результат неможливо віднести ні до додатних, ні до від'ємних чисел (код 01 або 10). У цьому випадку необхідно виконати корекцію результату додавання мантис шляхом його зсуву вправо на один розряд і одночасного збільшення порядку на 1.

Наприклад, після додавання мантис отримуємо число 01,00001. Нормалізуємо його зсувом вправо на один розряд і одночасно збільшуємо порядок на 1. В результаті отримуємо:

(А + В)доп = 00,100001 * 10к+1

Переводимо у прямий код: (А + В)пр = 00,100001 * 10к+1.

Внаслідок корекції результату шляхом зсуву вліво точність його погіршилася.

б). Результат від'ємний. Переведення результату додавання мантис у прямий код віконується шляхом виконання другого доповнення (тобто, виконується інверсія результату і до отриманого коду додається 1 у молодший розряд) або другого обертання (тобто, виконується інверсія результату);

в). Результат додатний. У прямому коді додатний результат залишається без змін;

Нормалізація результату.

Виконується тоді, коли після переведення у прямий код у отриманому результаті відбувається порушення нормалізації вправо, тобто з правого боку від коми є один або декілька нулів. Це порушення коректується зсувом мантиси результату вліво і одночасне зменшення порядку на кількість розрядів зсуву мантиси до повної нормалізації мантиси.

1. Остаточний результат додавання двох чисел з рухомою комою: сума мантис – це мантиса результату, порядок результату – вирівняний порядок доданків, із врахуванням всіх проведених корекцій. .

Розглянемо приклади виконання операції додавання двійкових чисел з рухомою комою. Всі арифметичні дії будемо виконувати у модифікованому доповнювальному коді.

Приклад 1: Додати А = - 0,1101 * 10101 і В = + 0,1100 * 10011.

а) Вирівнюємо порядки.. Спочатку, визначаємо арифметичну різницю між порядками, віднімаючи від значення порядку числа А значення порядку числа В. При цьому операцію віднімання замінюємо операцією додавання у доповнювальному модифікованому коді.

рА – рВ = рА + (– рВ) = (рА доп(м) = 00,101) + (рВ доп(м) = 11,101) = 00,010 – різниця порядків

Аналіз отриманої різниці:

- знак різниці показує, що порядок числа А більше порядку числа В;
- порядки відрізняються на дві одиниці.

Через те, що рА > рВ, зсуваємо мантису числа В вправо на два розряди, тобто

МВ пр= 00,001100 і Впр= 00,001100 * 10101.

Розряди мантиси числа В, які вийшли за межі розрядної сітки процесора, будуть втрачені, що погіршить точність обчислень.

б) Переводимо мантиси обох чисел у модифікований доповнювальний код (в межах 4-х розрядів):

МА пр = 11,1101 МА доп(М) =.11,0011

МВ пр = 00,0011 МВ доп(М) =.00,0011

в) Додаємо модифіковані доповнювальні коди мантис чисел А та В і отримуємо результат у модифікованому доповнювальному коді:

(А + В)доп (М) = 11,0011 + 00,0011 = 11,0110

г) Аналіз результату додавання мантис починається із знакових розрядів. Знакові розряди показують, що переповнення розрядної сітки у нас не виникло. Результат – від'ємний, тому переводимо мантису результату у прямий код шляхом виконання другого доповнення і дописуємо спільний порядок:

(А + В)пр (М) = (11,1001 + 00,0001) * 10 101 = 11,1010 * 10101 - остаточний результат.

При переведенні у прямий код знак результату не міняється.

Подальший аналіз показує, що порушення нормалізації результату вправо немає.

Приклад 2. : Додати двійкові числа: А = + 0,10100 * 10101 та В = - 0,10110 * 10100 .

а) Для вирівнювання порядків доданків необхідно із порядку числа А відняти порядок числа В. Віднімання замінимо додаванням у модифікованому доповняльному коді.

(рА доп(м)= 00,101) + (рВ доп(м) = 11,100) = ,101 – 00,100 = 00,101 + (-00,100) = 00,101 + (11,011 + 00,001) = 00,101 + 1,100 = 00,001

Через те, що pA > pB на +1, виконуємо зсув мантиси числа B вправо на 1 розряд.

MB пр = 11,010110 і B = 11,010110 * 10101.

б) Переведемо мантиси доданків у модифікований доповняльний код.

MA пр = 00,10100 MA доп(М) = 00,10100.

MB пр = 11,010110 MB доп(М) = 11,101010.

в) Додаємо мантиси чисел A і B у модифікованих доповняльних кодах:

(MA доп(М) = 00,10100) + (MB доп(М) = 11,10101) = 00,01001

г) Переводимо результат у прямий код (виконуємо друге доповнення). У нашому випадку прямий код суми мантис збігається з доповнювальним кодом суми мантис, тому що результат є число додатне.

(MA + MB) пр(м) = 00,01001

Як видно, виникло порушення нормалізації вправо, тому що вправо від коми розряд дорівнює 0.

д) Виконуємо нормалізацію результату, тобто зсув мантиси ліворуч на 1 розряд з одночасним відповідним зменшенням порядку:

(A + B)пр = 0,1001 * 10100 – остаточний результат.

3. Алгоритми множення і ділення чисел з фіксованою комою

Розглянемо основні способи виконання операції множення для різних систем. Найпоширенішим способом множення чисел є спосіб порозрядного множення множеного на множник, починаючи з молодшого розряду (1-й спосіб), починаючи зі старшого розряду (2-й спосіб). Розглянемо приклад:

```
347
532
694
  +1041
 +1735
  184604
```

```
347      множене (A)
532      множник (B)
1735 частковий (розрядний) добуток
+1041
  +694
184604   повний добуток (C)
```

Аналіз способів множення чисел в десятковій системі показує, що операція множення складається з порозрядного множення множеного на множник з перенесенням переповнення в старший розряд, зрушення часткових добутків на один розряд вліво (вправо), підсумовування часткових добутків.

У двійковому численні це завдання значно спрощується, оскільки множити порозрядно немає необхідності. Насправді, якщо множити множене на "1", то це повторення множеного із зсувом на один розряд вправо (вліво), а на "0" – записуються одні нулі із зрушенням.

```
 1101
*1101
 1101
  +0000
      +1101
    + 1101
     10101001
```

```
1101
    *1101
1101
+ 1101
   + 0000
     +1101
  10101001
```

В обох випадках операція множення складається з ряду послідовних операцій зсуву і складання часткових добутків. Таким чином, операція множення зводиться до складання часткових добутків, які виходять з множеного або нулів, якщо в розряді множника «нуль», або множеного, якщо в розряді множника «одиниця» і відповідним або зсувом.

Окрім операції складання, при виконанні множення, з'являється операція зсуву чисел. При цьому, можливі два варіанти:

- зсувати множене відповідно до вказівок множника;
- зсувати суму часткових добутків.

Розглянемо основні методи виконання операції множення ЦА.

МЕТОД 1. Нехай A – множене > 0, B – множник > 0, а C – добуток.

Запишемо числа у ФФК: $A=0,\alpha_1\alpha_2,...,\alpha_n$; $B=0,b_1,b_2,...,b_n$ (A і B правильні дроби). B – множник і він визначає розрядність C (чим він більший, тим більша розрядність C). Тоді, $B=b_1\cdot 2^{-1}+b_2\cdot 2^{-2}+...+b_n\cdot 2^{-n}$.

Звідси: $C=A\cdot B = 0,\alpha_1\alpha_{2...}\ \alpha_n(b_1\cdot 2^{-1}+b_2\cdot 2^{-2}+..\ .+b_n\cdot 2^{-n})=$

$=(2^{-1}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_1+(2^{-2}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_2+...+(2^{-n}\cdot 0,\alpha_1\alpha_{2...}\ \alpha_n)b_n..$ *(4.1)*

Множник 2^{-n} означає зсув на n розрядів вправо числа, що укладено в дужки, тобто зсуваэться вправо множене і множення на множник починається зі старших розрядів. Структурно це зображено на рис. 4.5.

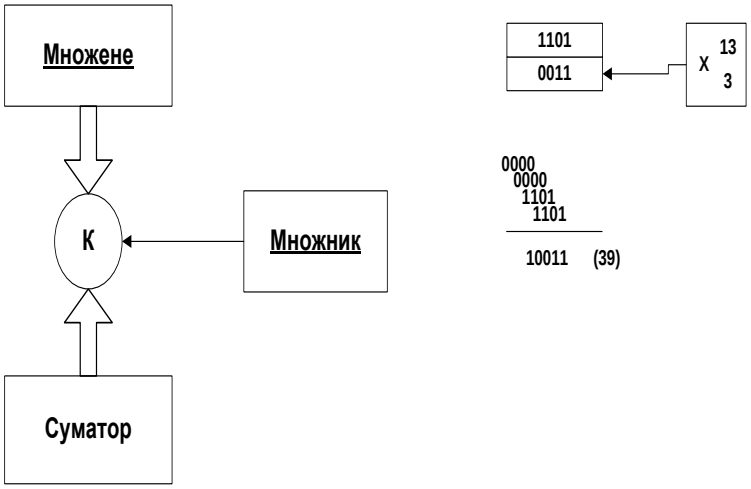


Рис.4.5 – Метод множення зі зрушенням вправо множеного

МЕТОД 2. Нехай $A=0,\alpha_1\alpha_{2...}\ \alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Перетворимо множник по методу Горнера:

$B=(...((b_n\cdot 2^{-1}+b_{n-1})\cdot 2^{-1}+b_{n-2})\cdot 2^{-1}+...+b_2)\cdot 2^{-1}+b_1)\cdot 2^{-1}$.

Тоді,

$C=A\cdot B=(...(b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot 2^{-1}+b_{n-1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot 2^1+...+b_1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^{-1}$, *(4.2)*

За цією моделлю, множення починається з молодших розрядів і вправо зрушується. Структура наведена на рис. 4.6

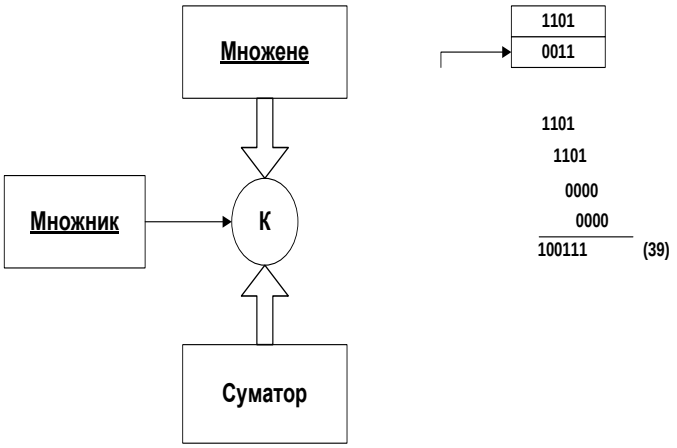


Рис.4.6 – Метод множення зі зрушенням вправо суми часткових добутків

МЕТОД 3. Нехай $A=0,\alpha_1\alpha_{2...}\alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Використовуючи метод Горнера, запишемо множник:

$B=2^{-n}(b_1\cdot 2n^{-1}+b_2\cdot 2^{n-2}+...+b_{n-1}\cdot 2^1+b_n\cdot 2^0)$. Тоді,$C=A\cdot B=$

$=2^{-n}(b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n+(2^1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot b_{n-1}+...+(2^{n-1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)\cdot b_1$ (4.3)

Це означає: множення починається з молодших розрядів і множене зсувається вліво на один розряд у циклі. Структура представлена на рис.4.7.

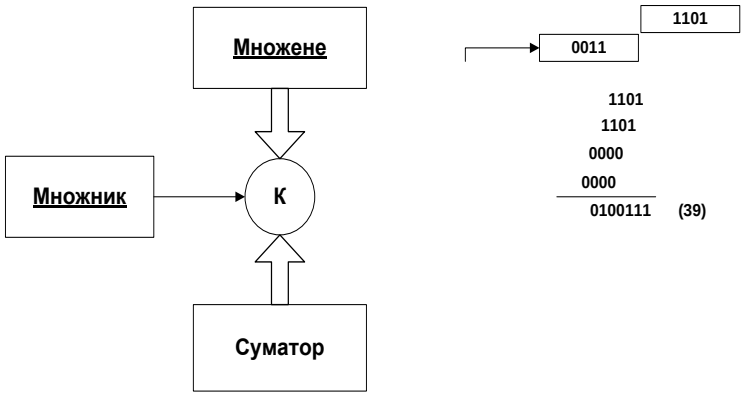


Рис.4.7 – Метод множення зі зрушенням вліво множеного на один розряд у циклі

МЕТОД 4. Нехай $A=0,\alpha_1\alpha_{2...}\alpha_n$ – множене; $B=0,b_1b_2...b_n$ – множник. Запишемо B – множник по методу Горнера, як у попередньому методі. Тоді, $C=A\cdot B=$

$=2^{-n}(...(2^1(b_1\cdot 0,\alpha_1\alpha_{2...}\alpha_n)+b_2\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^1+\ldots$

$+b_{n1}\cdot 0,\alpha_1\alpha_{2...}\alpha_n)2^1+ +b_n\cdot 0,\alpha_1\alpha_{2...}\alpha_n)$. (4.4)

За цією моделлю множення починається зі старшого розряду і в кожному циклі сума часткових добутків зрушується вліво. Схема такого множного пристрою наведена на рис.4.8.

Рис.4.8 – Метод множення зі зрушенням вліво суми часткових добутків

Таким чином, для реалізації операції множення необхідно мати, як мінімум: суматор, регістри для зберігання множеного й множника, схему аналізу розрядів множника. Суматор і регістри повинні мати ланцюги зсуву вмісту в ту або іншу сторону відповідно до прийнятого методу множення.

Множення чисел з фіксованою крапкою (комою) на ДСПК

Запишемо машинне зображення множеного і множника у формі з фіксованою комою в прямому коді. $A_{пр}=Sg,\alpha_1\alpha_{2...}\alpha_n$; $B_{пр}=Sg,b_1b_2...b_n$. Тоді, їхній добуток запишеться як $C_{пр}=Sg,c_1c_2...c_n$, де $Sg_C=Sg_A \square Sg_B$, де $\square$ – знак додавання функції «Σ по mod2».

Таким чином, при використанні ДСПК, знак добутку визначається окремо від цифрової частини, потім виконується операція множення. Вона виконується відповідно до заданої структури множного пристрою (див. наприклад, рис. 4.9) і методу множення (див. наприклад, метод 2).

За методом 2 множення починається з молодшого розряду і зсувається вправо сума (Σ) часткових добутків.

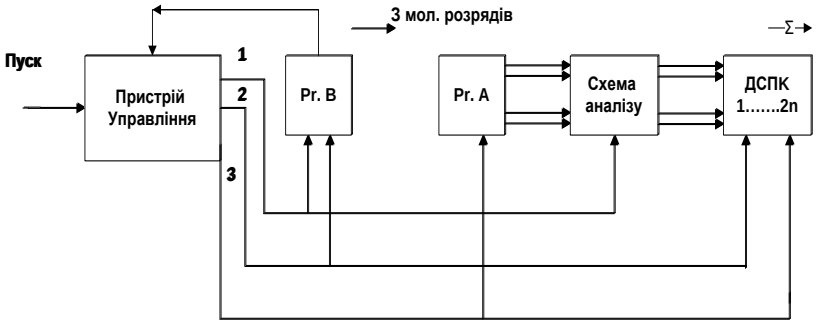


Рис.4.9 – Структура ЦА для пристрою множення чисел з фіксованою комою

Приклад. Помножити $A_{пр}$=11/11010 (–26); $B_{пр}$ =00/11001 (25), С= –650.
РІШЕННЯ: Визначається знак добутку С: 1*0 =1.
Приймемо:
1) суматор має 10 розрядів (без знака).

2) регістри мають по 5 розрядів (без знака).

Послідовність дій представимо таблицею 4.3

Для спрощення запису таблиць, приймемо наступні умовні позначки:

- оператор := привласнення значення (блоку ліворуч привласнюється значення, що є праворуч);
- позначення, наприклад, [См] – вміст суматора;
- оператор $[\overrightarrow{\text{РгА}}]$зсуву вмісту, наприклад, регістра А вправо наодин розряд;
- позначення В. П. – вхідне положення;
- позначення $A_{пр}$, $B_{пр}$- цифрова частина множенного, множника (прямий код).

Відповідь: $C_{пр}$= 11/1010001010.

Таблиця 4.3

Приклад рішення

Суматор [см]	Регістр [РгВ]	Коментар
+0000000000 11010 1101000000 0110100000 0011010000 +0001101000 11010 1110101000 +0111010100 11010 10100010100 1010001010	11001 -1100 --110 ---11 ----1 -----	і.П.[см]:=0; РгА:= А; РгВ:= В b_5=1; [см]+[РгА]; $[\overrightarrow{\text{РгВ}}]$; $[\overrightarrow{\text{см}}]$; b4=0; $[\overrightarrow{\text{РгВ}}]$; $[\overrightarrow{\text{см}}]$; b3=0; $[\overrightarrow{\text{РгВ}}]$; $[\overrightarrow{\text{см}}]$; b2=1; [см]:=[см]+ [РгА]; $[\overrightarrow{\text{см}}]$; $[\overrightarrow{\text{РгВ}}]$; b1=1; [см]:=[см]+ [РгА]; $[\overrightarrow{\text{РгВ}}]$; $[\overrightarrow{\text{см}}]$; Кінець

Якщо при множенні виникає одиниця переносу зі старшого розряду, то її зберігають шляхом зсуву суматора (тобто необхідно передбачати в ЦА стробування (фіксування) сигналу переповнення для виробу зсуву на один розряд вправо. Цей спосіб одержав найбільше поширення в практиці ЦА.

4. Алгоритми множення і ділення чисел з плаваючою комою

Особливості множення чисел представлених у формі з плаваючою комою.

Для чисел, представлених у формі з плаваючою комою, обов'язковим є представлення у вигляді мантиси і порядку (характеристики). При операції множення дії, які виконуються над мантисами і порядками, різні: мантиси перемножуються, порядки складаються. Очевидно, що результат множення може вийти ненормалізованим, тоді буде потрібно нормалізація з відповідною корекцією порядку результату. Отже, структурна схема розмножувального пристрою повинна змінитися (рис.4.10).

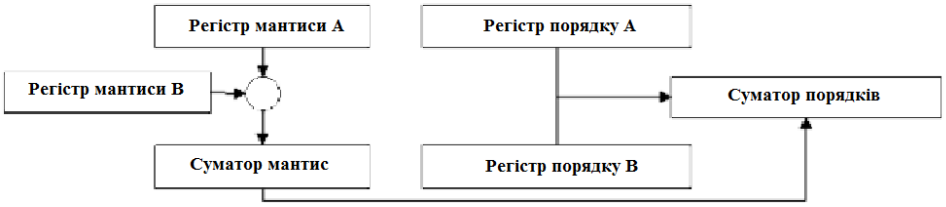


Рис. 4.10 – Структурна схема пристрою множення з плаваючою комою

Розглянемо приклад виконання операцій множення чисел, заданих в прямому коді.

Приклад. Перемножити числа А = - 0,11001·2-3 і В = 0,10011·2 +1.

Як пристрій множення використовується схема, показана на рис. 1, де РгА і РгВ - відповідно регістри для порядків рА і рВ.

Розв’язок: Мантиси перемножуються за правилами, розглянутим для чисел, представлених у формі з фіксованою комою. Для перемножування мантис використовується суматор прямого коду, а для складання порядків - суматор зворотного коду.

Спочатку записуються машинні зображення чисел:

[mA]пр=1.11001; [pA]об=1.100;

[mB]пр=0.10011; [pB] об= 0.001;

Послідовність дій у процесі виконання операції множення мантис представимо в табл.4.4.

Після виконання зазначених дій знаходиться мантиса добутку [mC] пр = 1,0111011011.

Одночасно з цим над порядками проводиться операція додавання:

[pC] об = [pA] об+ [pB] об = 1,100 + 0,001 = 1,101.

Так як мантиса результату не задовольняє умові нормалізації (порушена ліва межа: $\delta = 1$, $\gamma = 0$), то виконується зсув мантиси ліворуч на один розряд: [mC] пр = 1,1110110110, і корекція порядку [p'C]об = [pC]об+ 1,110 = 1,101+1,110 = 1,110.

Таблиця 4.4

Знак результату	Суматор	Регістр В	Примітка
$Sg_c = Sg_1 \oplus Sg_B=1\oplus0=1$	00000 + 11001	10011	Prm_B: = $[m_B]$; $Prm_A = [m_A]$ $Prp_A = [p_A]$;Prp_B: = $[p_B]$; CM_M: = 0;
	11001 011001 + 11001	01001	B_5=1;.CMm:= [CMm] +[Prm_A]; [PrB]; [CM]; B_4=1; CMm:= [CMm] + [Prm_A];
	1001011 1001011 01001011 001001011 + 11001	00100 00010 00001	 [Prm_B]; [CMm]; b_3 = 0; [Prm_B]; [CMm]; b_2 = 0; [Prm_B]; [CMm];
	111011011 0111011011	00000	b_1 = 1; CMm:=[CMm]+ [Prm_A]; [Prm_B]; [CMm]; Кінець

Якщо суматор мантис містить тільки n розрядів, то після округлення виходить вихідний результат.

Відповідь: С = - 0,11110·2-3.

При виконанні операції множення може мати місце ряд особливих випадків. Наприклад:

- якщо один із співмножників дорівнює нулю, то добуток також дорівнює нулю. Отже, необхідно передбачити блокування виконання алгоритму множення і формувати результат, рівний нулю;
- якщо порядок результату дорівнює найбільшій негативній величині, то необхідно формувати машинний нуль.

Ці особливі випадки можна передбачити в алгоритмі операції введенням аналізатора співмножників на нуль на початку операції (перший випадок), або корекцією добутку на підставі ознак результату (другий і третій випадки).

Ділення чисел представлених у формі з плаваючою комою.

В принципі, операція ділення для числа з плаваючою точкою, також як і всі інші, описані до цього операції, зводиться до знаходження порядку числа і його мантиси. Порядок знаходиться шляхом різниці між [Ап] - [Вп], а що стосується мантиси, то її результат виходить шляхом проведення операції ділення [Ам] - [Вм]. Ну і природно, заключним етапом буде процедура приведення результату до нормалізованого виду (порушення може бути зліва від точки).

Ділення: 1. Порядки віднімаються

2. Мантиси діляться
3. При необхідності результат нормалізується

Приклад:

$0{,}0101*10^{100}/0{,}1010*10^{101}=(0{,}0101/0{,}1010)*10^{-001}=0{,}1011*10^{-001}$

00,0101	00,1010	11,1010 МПК	4 - 5 = -1		
11,0110	0,1011	11,0101 МОК	0000 0100 ПК	1000 0101 ПК	
11,1011		11,0110 МДК	+1111 1011 ДК	1111 1010 ОК	
11,0110			1111 1111 ДК	1111 1011 ДК	
00,1010			1000 0000 ОК		
00,0000			+1		
00,0000			1000 0001 ПК		
11,0110					
11,0110					
11,1100					
00,1010					
00,0110					
00,1100					
11,0110					
00,0010					

Контрольні питання:

1. Що таке машинне слово?
2. Як в пам’яті ЕОМ представляються числа з фіксованою та плаваючою комою?
3. Що таке мантиса? Що таке порядок числа?
4. Скільки розрядів відводиться під мантису і порядок?
5. Як визначити знак числа?
6. Як отримати прямий, зворотний та додатковий коди числа?
7. Як здійснюється додавання і віднімання двійкових чисел?
8. Як здійснюється множення і ділення двійкових чисел?
9. Що таке переповнення розрядної сітки? Коли воно виникає?
10. Для чого застосовується модифікований код?
11. При якому сполученні знаків операндів модифікований код не потрібен для правильного формування знака результату.
12. Побудуйте схему пристрою тільки для додавання чисел в ОК (ДК).

ТЕМА 5. КОДУВАННЯ ІНФОРМАЦІЇ. ПОНЯТТЯ ПРО КОНТРОЛЬ РОБОТИ ЦИФРОВОГО АВТОМАТА.

План

1. Поняття про кодування і коди.
2. Послідовний і паралельний коди.
3. Код Хеммінга.
4. Контроль за парністю і за модулем.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Поняття про кодування і коди

Алгоритми виконання арифметичних операцій забезпечать правильний результат тільки у випадку, якщо машина працює без порушень. При виникненні будь-якого порушення нормального функціонування результат буде невірним, однак користувач про це не дізнається, якщо не будуть передбачені позначки для створення системи виявлення можливої помилки, а з іншого боку, повинні бути опрацьовані заходи, що дозволяють виправити помилки. Ці функції слід покласти на систему контролю роботи цифрового автомата.

Система контролю – сукупність методів і засобів, що забезпечують визначення правильності роботи автомата в цілому або його окремих вузлів, а також автоматичне виправлення помилки.

Помилки в роботі цифрового автомата можуть бути викликані або виходом з ладу якоїсь деталі, або відхиленням від норми параметрів (наприклад, зміна напруги живлення) або впливом зовнішніх перешкод. Викликані цими порушеннями помилки можуть прийняти постійний або випадковий характер. Постійні помилки легше виявити і виявити. Випадкові помилки, обумовлені короткочасними змінами параметрів, найбільш небезпечні і їх важче виявити.

Тому система контролю повинна будується з таким розрахунком, щоб вона дозволяла виявити і по можливості виправити будь-які порушення. При цьому треба розрізняти наступні види помилок результату:

1. виникають через похибки у вихідних даних;
2. обумовлені методичними похибками;
3. з'являються через виникнення несправностей у роботі машини.

Перші два види помилок не є об'єктом для роботи системи контролю. Звичайно, похибки перекладу або подання числової інформації в розрядної сітки автомата призведуть до виникнення похибки в результаті рішення задачі. Цю

похибку можна заздалегідь розрахувати і, знаючи її максимальну величину, правильно вибрати довжину розрядної сітки машини. Методичні похибки також враховуються попередньо.

Перевірка правильності функціонування окремих пристроїв машини і виявлення несправностей може здійснюватися за двома напрямами:

- профілактичний контроль, завдання якого – попередження появи помилок в роботі;
- оперативний контроль, завдання якого – перевірка правильності виконання машиною усіх операцій.

Рішення всіх завдань контролю стає можливим тільки при наявності певної **надмірності.** Надмірність може бути або апаратними (схемними) засобами, або логічними або інформаційними засобами. До методів логічного контролю можна віднести наступні прийоми. У ЕОМ першого і другого поколінь відсутність системи оперативного контролю призводило до необхідності здійснення «подвійного рахунку», коли кожна задача вирішувалася двічі, і в разі збігу відповідей приймалося рішення про правильність функціонування ЕОМ.

Якщо в процесі вирішення якоїсь задачі обчислюються тригонометричні функції, то для контролю можна використовувати відомі співвідношення між цими функціями, наприклад, $\sin^2 ④+\cos^2 ④=1.$ Якщо це співвідношення виконується заданою точністю на кожному кроці обчислень, то можна з упевненістю читати, що ЕОМ працює правильно.

Обчислення визначеного інтеграла із заданим кроком інтегрування можна контролювати порівнянням отриманих при цьому результатів з тими результатами, які відповідають більш крупному кроці. Такий «скорочений» алгоритм дасть, мабуть, більш грубі оцінки і по суті вимагає додаткових витрат машинного часу.

Всі розглянуті приклади свідчать про те, що такі методи контролю дозволяють лише зафіксувати факт появи помилки, але не визначають місце, де сталася ця помилка. Для оперативного контролю роботи ЕОМ визначення місця, де сталася помилка, тобто рішення задачі пошуку несправності, є досить суттєвим питанням.

У цифровому автоматі можуть відбутися ті чи інші збої, що призводять до спотворення інформації. Тому, при проектуванні цифрових автоматів повинні бути передбачені засоби, що дозволяють контролювати, виявляти і виправляти виникаючі помилки. Вирішення всіх задач контролю стає можливим тільки при наявності певної надмірності інформації, яка супроводжує основну інформацію. Інакше кажучи, при поданні числа в будь-якому коді, тобто під час кодування інформації, необхідно передбачити в цьому коді додаткові, так звані, контрольні розряди.

Систематичний код – код, який містить у собі, крім інформаційних, контрольні розряди.

У контрольні розряди записується деяка інформація про вихідний числі. Тому можна говорити, що систематичний код володіє надмірністю.

2. Послідовний і паралельний коди

Кодування – процес присвоєння умовного позначення різним позиціям номенклатури.

Код – це знак чи сукупність знаків, прийнятих для позначення класифікаційного угруповання чи об’єкта класифікації.

Для кодування інформації в інформаційних системах застосовують порядковий, серійно-порядковий, послідовний та паралельний методи кодування.

Порядковий метод кодування – найпростіший і найпоширеніший. Побудова кодів виконується в міру зростання або спадання ознак без пропуску номерів.

Серійно-порядковий метод кодування на кожну групу ознак має серію порядкових номерів із резервом номерів.

Послідовний метод кодування передбачає виокремлення певних розрядів коду під певні ознаки.

Паралельний метод кодування теж передбачає виокремлення розрядів, але значення ознаки, записаної на будь-якому розряді коду, не залежить від значення ознак, записаних на інших розрядах.

Вибір методів класифікації та кодування об’єктів передбачає:

- можливість розширення кодової множини об’єктів і внесення відповідних змін;
- однозначність ідентифікованих об’єктів;
- мінімальну довжину коду;
- можливість оброблення інформації за допомогою ЕОМ;
- простоту методу кодування;
- застосування загальноприйнятих позначень.

Послідовне кодування використовується для кодування об'єктів, утворених ієрархічною класифікацією. Суть методу полягає ось у чому: спочатку записують код угруповання 1-го рівня, потім код угруповання 2-го рівня, далі код угрупування 3-го рівня і т. д. У результаті утворюється кодова комбінація, кожний розряд якої містить інформацію про специфіку виділеної групи на кожному рівні ієрархічної структури. Послідовній системі кодування притаманні ті самі переваги і недоліки, що й ієрархічній системі класифікації. Послідовне кодування широко використовується у практичній роботі.

Паралельне кодування використовується для фасетної системи класифікації. Суть методу полягає в такому: всі фасети кодуються незалежно одна від одної; для значень кожної фасети виділяється певна кількість розрядів коду (у цьому разі для всіх фасет виділено один розряд, тому що кількість значень у фасетах не перевищує числа десять). Паралельній системі кодування притаманні ті ж переваги і недоліки, що її фасетній системі класифікації.

3. Код Хеммінга

До найбільш поширених кодів з виявленням та виправленням помилок відносять код Хемінга, циклічні та рекурентні коди.

Коди, запропоновані американським вченим Р. Хемінгом, дозволяють не тільки виявити але і виправити поодинокі помилки. Ці коди - систематичні.

Код Хеммінга – це алгоритм самоконтролюючого і самокорегуючого коду, який дозволяє закодувати будь-яке інформаційне повідомлення певним чином і після передачі (наприклад, по мережі) визначити чи з'явилася якась помилка в цьому повідомленні і, при можливості, відновити це повідомлення.

Хемінг вперше ввів поняття кодового відстані.

Кодовою відстанню між двома словами називається число розрядів, в яких символи слів не збігаються.

Мінімальним кодовою відстанню (dmin) даного коду називається мінімальна відстань між двома будь-якими словами в цьому коді. Якщо довжина слова n, то кодова відстань може приймати значення від 1 до n. Якщо є хоч одна пара слів, що відрізняються в одному розряді, то мінімальна кодова відстань дорівнює 1. Для систематичних кодів dmin> 1. У загальному випадку, щоб код дозволяв виявляти помилки кратністю r, має виконуватися умова:

$$\text{dmin} \quad r + 1.$$

Код Хемінга відноситься до систематичних кодів, в яких з n символів, які утворюють комбінацію, n_0символів є інформаційними, а останні $k = n - n_0$ с надлишковими (контрольними), призначеними для перевірки (контрольні символи у всіх комбінаціях займають однакові позиції). Коди Хемінга дозволяють виправити всі одиничні помилки (при кодовій відстані d=3) і визначити всі подвійні помилки (при $d=4$), але не виправляти їх.

Зв'язок між кількістю інформаційних та контрольних символів в коді Хемінга знаходять на основі таких міркувань. При передачі комбінації по каналу з шумами може бути спотворений довільний з n символів коду, або комбінація може бути передана без спотворень. Таким чином може бути $n + 1$ варіантів спотворення (включаючи передачу без спотворення). Використовуючи контрольні символи, необхідно перевірити всі $n+1$варіантів. За допомогою контрольних символів k можна описати $2k$ подій. Для цього повинна бути використана умова:

$$2^k \geq n+1 = n_0 + k + 1.$$

В таблиці 3.1 подана залежність між k і n_0, яка отримана з цієї невірності, де k - число контрольних символів в коді Хемінга, n_0 - інформаційних символів.

Таблиця 5.1

Розміщення контрольних символів в комбінаціях коду Хемінга

n_0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
k	2	3	3	3	4	4	4	4	4	4	4	5	5	5	5

В коді Хемінга контрольні символи розташовують на місцях, кратних степеню числа 2, тобто на позиціях 1, 2, 4, 8 і т. п. Інформаційні символи розташовують на місцях, що залишилися. Наприклад, для семиелементної закодованої комбінації можна записати

k_1	k_2	a_{04}	k_3	a_{03}	a_{02}	a_{01}
a_1	a_2	a_3	a_4	a_5	a_6	a_7

.

Символи коду Хемінга, які обведені прямокутниками, є *контрольними*, останні – *інформаційні*, де a_3 – старший (четвертий) розряд вихідної кодової комбінації двійкового коду, який необхідно кодувати, a_7 – молодший (перший) розряд. Після розташування на відповідних місцях кодової комбінації контрольних і інформаційних символів в коді Хемінга складають спеціальні перевірні рівняння, які використовують для визначення наявності спотворень і їх виправлення. 3 перевірних рівнянь і отримують контрольні символи при кодуванні вихідної кодової комбінації двійкового коду. Для визначення контрольних символів необхідно використати такий алгоритм.

1. Всі символи коду Хемінга з номерами розрядів розташовують в порядку збільшення номерів і під ними записують номери розрядів в двійковому коді

a_1 a_2 a_3 a_4 a_5 a_6 a_7

0001 0010 0011 0100 0101 0110 0111

2. Перше перевірне рівняння складають як суму за mod 2 всіх розрядів, в номерах яких в молодшому розряді 2^0 стоїть одиниця: $S_1 = a_1 \oplus a_2 \oplus a_5 \oplus a_7$.

Друге перевірне рівняння складають як суму за mod 2 всіх розрядів, в номерах яких стоїть одиниця на другому місці відповідного двійкового еквівалента (2^1): $S_2 = a_2 \oplus a_3 \oplus a_6 \oplus a_7$.

Третє перевірне рівняння складають як суму за mod 2 всіх розрядів, в номерах яких стоїть одиниця на третьому місці (2^2): $S_3 = a_4 \oplus a_5 \oplus a_6 \oplus a_7$.

Аналогічно утворюються і інші перевірні суми (при більшій кількості інформаційних і контрольних символів, відповідно).

Як видно з наведених рівнянь, в кожну перевірну суму входить тільки один невизначений контрольний символ k_i (a_1, a_2, a_4, відповідно), а всі інші інформаційні символи відомі

Всі перевірні рівняння за умовою Хемінга повинні дорівнювати 0 при підсумовуванні за mod 2. 3 цієї умови і знаходять контрольні символи.

Наприклад, необхідно передати інформаційну кодову комбінацію:

a_1 a_2 a_3 a_4

1 1 0 0

з числом розрядів $n_0 = 4$.

З формули $2^k n + 1 = n_0 + k + 1$ визначимо, що $k = 3$. Запишемо перевірні суми:

$S_1 = a_1 + a_3 + a_5 + a_7$;

$S_2 = a_2 + a_3 + a_6 + a_7$;

$S_3 = a_4 + a_5 + a_6 + a_7$.

Підставимо у рівняння значення відомих інформаційних символів. З умови рівності нулю всіх перевірних сум визначаємо відповідно контрольні символи. З першого рівняння $a_0 = 0$, з другого $a_1 = 1$, з третього $a_4 = 1$.

Відповідно буде передана така комбінація:

a_1	a_2	a_3	a_4	a_5	a_6	a_7
0	1	1	1	1	0	0,

яка є комбінацією коду Хемінга.

4. Контроль за парністю і за модулем

Кодування за методом парності-непарності

Якщо в математичному коді виділений один контрольний розряд (k = 1), то до кожного бінарного числа додається один надлишковий розряд і в нього записується 1 або 0 з такою умовою, щоб сума цифр в кожному числі була за модулем 2 дорівнює 0 для випадку непарності. Поява помилки в кодуванні виявиться по порушенню парності (непарності). При цьому допускається, що може виникнути тільки одна помилка. У самому справі, для випадку парності правильним буде тільки половина можливих комбінацій. Щоб одна допустима комбінація перетворилася в іншу, має виникнути, принаймні, два порушення або парне число порушень. Приклад реалізації методу парності представлений в табл. 5.2.

Таблиця 5.2

Число	Контрольний розряд	Перевірка
10101011	1	0
11001010	0	0
10010001	1	0
11001011	0	1-порушення

Таке кодування має мінімальна кодова відстань, рівна 2.

Можна уявити і дещо видозмінений спосіб контролю за методом парності - непарності. Довге число розбивається на групи. Контрольні розряди виділяються всім групам по рядках і по стовпцях згідно з наступною схемою:

a_1	a_2	a_3	a_4	a_5	k_1
a_6	a_7	a_8	a_9	a_{10}	k_2
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	K_3
a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	k_4
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	k_5
k_6	k_7	k_8	k_9	k_{10}	

Збільшення надмірності інформації призводить до того, що з'являється можливість не тільки виявити помилку, але і виправити її. У самому справі, нехай відбулася несправність у якомусь із розрядів цього числа (уявімо, що розряд a_{18} змінив стан, тобто $a_{18 = 1)}$. Це призводить до того, що при перевірці на парність сума $\sum_i a_i + k_i$ по відповідним рядок зміниться для значень, які містять елемент $a_{18,}$ тобто це буде четверта зверху рядок і третій зліва стовпець. Отже, порушення парності по цьому рядку і стовпцю можна зафіксувати, що в кінцевому рахунку означає виявлення не тільки самої помилки, але і місця, де виникла помилка. Змінивши вміст зазначеного розряду (в даному випадку $a_{18)}$ на протилежне, можна виправити помилку.

Контроль за методом парності-непарності широко використовують в ЕОМ для контролю запису, зчитування інформації в запам'ятовуючих пристроях на магнітних носіях.

Контроль за модулем

Контроль виконання арифметичних і логічних операцій можна здійснювати за допомогою контрольних кодів, що представляють собою залишки від ділення чисел на деякий модуль. Такий контроль називається контролем по модулю. Для двійкових чисел цей модуль звичайно дорівнює або більше 3. Розрізняють числовий і цифровий контроль за модулем.

При числовому методі код заданого числа визначається як найменший позитивний залишок від ділення числа на обраний модуль.

Наприклад: визначити контрольний код чисел A = 125 і B = 89 по модулю 11. 1) 125: 11 = 12 (4), тобто контрольний код дорівнює 4. 2) 89: 11 = 8 (1), тобто контрольний код дорівнює 1.

При цифровому методі контролю контрольний код числа утворюється діленням суми цифр числа на обраний модуль.

У даному варіанті можливі два шляхи отримання контрольного коду:

1) безпосереднє ділення суми цифр на модуль;

2) просто підсумовування цифр по вибраному модулю.

Наприклад: визначити контрольний код для чисел 153 і 41 по модулю 3.

Сума цифр 153 дорівнює 9, а сума цифру 41 дорівнює 5. Розділивши ці суми на 3 отримаємо відповідно контрольні коди 0 і 2.

Контрольні питання:

1. Що таке довжина коду?
2. Які є системи кодування?
3. Що являє собою система контролю цифрового автомата?
4. У чому полягає метод паралельного кодування?
5. У чому полягає метод послідовного кодування?
6. Розкрийте суть методу Хеммінга.
7. Які види контролю за модулем ви знаєте? Поясніть їхню суть.

ТЕМА 6. ЛОГІЧНІ ОСНОВИ ЦИФРОВИХ АВТОМАТІВ

План:

1. Основні поняття теорії елементарних функцій алгебри логіки.
2. Основні закони алгебри логіки та їх використання для подання одних функцій логіки через інші.
3. Основні властивості функцій алгебри логіки. Поняття про логічний базис.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.

ІІ. Дайте відповідь на запитання.

ІІІ. Виконайте вправи для самостійної роботи з використанням середовища Microsoft Excel відповідно свого варіанту.

Матеріали для самостійного опрацювання

1. Основні поняття теорії елементарних функцій алгебри логіки.

При проектуванні обчислювальної техніки для формального опису логічних схем використовують математичний апарат алгебри логіки, об'єктом дослідження якого є функції, які набувають, як і їх аргументи, тільки два значення – 0 та 1. Вивчення властивостей таких функцій є дуже важливим для успішного розв'язання задач, які виникають перед фахівцями з проектування засобів обчислювальної техніки.

Алгебра логіки (алгебра тверджень, висловлювань) – це розділ математичної логіки, в якому вивчаються логічні операції над твердженнями (висловлюваннями). Часто передбачається, що твердження можуть бути лише істинними чи хибними (двійкова логіка). Теорія логічних основ комп'ютерної техніки надзвичайно насичена досить специфічними термінами, поняттями.

Функцію $f(x_1, x_2, ...x_n)$, яка набуває тільки значення 0 або 1, як і її аргументи, прийнято називати логічною функцією або булевою функцією. Аргументи булевої функції також називають булевими.

Довільна булева функція задається одним із трьох способів: табличним, геометричним та аналітичним.

Оскільки аргументи логічних функцій можуть набувати лише двох значень, область визначення будь-якої логічної функції скінченна. Тому будь-яка функція алгебри логіки може бути задана таблицею її значень залежно від значень аргументів.

В табл. 6.1 задані дві логічні функції трьох аргументів $f(x_1, x_2, x_3)$ і $\phi(x_1, x_2, x_3)$.

Таблиця 6.1

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
$f(x_1, x_2, x_3)$	0	1	0	1	0	0	0	1
$\phi(x_1, x_2, x_3)$	1	0	0	1	0	1	1	0

Сукупність значень аргументів називається набором функції. Функції f і ϕ, задані в табл. 6.1, визначені на восьми наборах. Функція $f(x_1, x_2, x_3)$ набуває значення, що дорівнюють одиниці на наборах (0, 0, 1), (0, 1, 1) і (1, 1, 1), а на решті дорівнює нулю. Функція $\phi(x_1, x_2, x_3)$ дорівнює одиниці на чотирьох наборах (0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0), а на решті – дорівнює нулю. До основних властивостей логічних функцій відносяться такі:

а) будь-яка логічна функція n аргументів визначена на 2^n наборах.

Відомо, що кількість різних n-розрядних чисел дорівнює 2^n, якщо кожному набору аргументів можна поставити у відповідність двійкове n-розрядне число. Так, наприклад, подані в табл. 1.1 логічні функції визначені на 8 наборах: (0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1);

б) число різних логічних функцій n аргументів скінченне і дорівнює 2^{2n}.

Оскільки логічна функція n аргументів визначена на 2^n наборах, то можна поставити у відповідність кожній логічній функції двійкове число, що містить 2^n розрядів. При цьому кількість різних двійкових 2^n - розрядних чисел дорівнює 2^{2n}, таким чином і кількість різних логічних функцій дорівнює 2^{2n}.

в) кількість логічних функцій n аргументів різко зростає зі збільшенням n (табл. 6.2).

Таблиця 6.2

Кількість аргументів	1	2	3	4	5
Число перемикаючих функцій	4	16	256	65536	$4{,}3 \cdot 10^9$

Логічна функція одного аргументу подана в табл. 6.3.

Таблиця 6.3

x	0	1	Умовне позначення	Найменування функції
$f_0(x)$	0	0	0	Константа нуля
$f_1(x)$	0	1	x	Змінна
$f_2(x)$	1	0	$\overline{x}$	Інверсія
$f_3(x)$	1	1	1	Константа одиниці

Логічні функції двох аргументів подані в табл. 6.4. Наведені булеві функції дозволяють будувати нові булеві функції за допомогою узагальненої операції, яка називається операцією суперпозиції. Операція суперпозиції полягає в підстановці замість аргументів інших булевих функцій (зокрема аргументів).

Таблиця 6.4

Функція	0 0	0 1	1 0	1 1	Назва функції	Позначення	Функція, що виконується
$f_0(x,y)$	0	0	0	0	константа нуль	0	
$f_1(x,y)$	0	0	0	1	добуток (кон'юнкція)	$x \wedge y$	$x \cdot y$
$f_2(x,y)$	0	0	1	0	f заборона по у	$x \Delta y$	$x\overline{y}$
$f_3(x,y)$	0	0	1	1	змінна х	x	X
$f_4(x,y)$	0	1	0	0	f заборона по х	$y \Delta x$	$\overline{x}y$
$f_5(x,y)$	0	1	0	1	змінна у	y	Y
$f_6(x,y)$	0	1	1	0	сума за модулем 2	$x \oplus y$	$\overline{x}y \vee \overline{y}x$
$f_7(x,y)$	0	1	1	1	диз'юнкція	$x \vee y$	$x \vee y$
$f_8(x,y)$	1	0	0	0	операція Пірса (функція Вебба)	$x \downarrow y$	$\overline{x \vee y}$
$f_9(x,y)$	1	0	0	1	логічна рівнозначність	$x \sim y$	$x \equiv y$
$f_{10}(x,y)$	1	0	1	0	інверсія у	$\overline{y}$	$\overline{y}$
$f_{11}(x,y)$	1	0	1	1	імплікація від у до х	$y \to x$	$x \vee \overline{y}$
$f_{12}(x,y)$	1	1	0	0	інверсія х	$\overline{x}$	$\overline{x}$
$f_{13}(x,y)$	1	1	0	1	імплікація від х до у	$x \to y$	$\overline{x} \vee y$
$f_{14}(x,y)$	1	1	1	0	операція Шеффера (штрих Шеффера)	x/y	$\overline{x} \vee \overline{y} = \overline{xy}$
$f_{15}(x,y)$	1	1	1	1	константа одиниці	1	

Існує 16 різних логічних функцій двох аргументів (x і y), кожна з яких визначена на 4-х наборах.

З 16-ти функцій, які подані в табл. 6.4, 6 функцій

$$f_0(x,y)=0;\ \ f_3(x,y)=x;\ \ f_5(x,y)=y;\ \ f_{10}(x,y)=\overline{y};\ \ f_{12}(x,y)=\overline{x};\ \ f_{15}(x,y)=1.$$

є константами або функціями одного аргументу. Решта десять функцій залежать від двох аргументів і мають свої загальноприйняті позначення і назви.

Функція $f_1(x,y)$ називається кон'юнкцією, логічним множенням або логічним І. Для її позначення використовуються: знак множення $x \cdot y$; знак кон'юнкції $x \wedge y$; знак логічного І - $x \& y$.

Функція $f_7(x,y)$ носить назву диз'юнкції, логічного додавання, логічного АБО. Для позначення використовується знак $f_7(x,y)=x\vee y$, іноді для зручності $x+y$.

Функція $f_6(x,y)$ називається функцією нерівнозначності або сумою за модулем 2:

$$f_6(x,y)=x\oplus y\ .$$

Функція $f_9(x,y)$ називається функцією рівнозначності або еквівалентності:

$$f_9(x,y)=x\equiv y$$

Функція $f_{14}(x,y)$ називається штрихом Шеффера або запереченням кон'юнкції:

$$f_{14}(x,y)=x/y \text{ або } f_{14}(x,y)=\overline{xy}\ .$$

Останнє позначення показує, що функція може бути отримана шляхом суперпозиції, кон'юнкції та інверсії.

Функція $f_8(x,y)$ називається запереченням диз'юнкції, функцією Пірса або стрілкою Пірса:

$$f_8(x,y)=x\downarrow y \text{ або } f_8(x,y)=\overline{x\vee y}\ .$$

Функції $f_{11}(x,y)$ і $f_{13}(x,y)$ називаються імплікацією:

$$f_{11}(x,y)=y\to x \text{ і } f_{13}(x,y)=x\to y \text{ або } f_{11}(x,y)=x\vee\overline{y} \text{ і } .\ f_{13}(x,y)=\overline{x}\vee y$$

Функція $f_2(x,y)$ і $f_4(x,y)$ називається функцією заборони або заперечення імплікації:

$$f_2(x,y)=x\overline{y} \text{ і } f_4(x,y)=\overline{x}\,y\ .$$

2. Основні закони алгебри логіки та їх використання для подання одних функцій логіки через інші.

Вперше логічні функції були використані в алгебрі логіки, початок якій покладено працями англійського математика Дж. Буля, її також називають булевою алгеброю або алгеброю висловлень.

Під висловленням розуміється будь-яке твердження, яке може бути істинним або хибним.

Істинному висловленню приписується 1, хибному – 0. Висловлення можуть бути простими і складними. Складні висловлення складаються з простих.

Для об'єднання простих висловлень в складні використовуються логічні зв'язки, що відповідають логічним функціям, аргументами яких є прості висловлення.

Логічний зв'язок «І» (кон'юнкція). Кон'юнкцією називають складне висловлення, що містить 2 або більше простих висловлень і яке є істинним тоді і лише тоді, коли істинними є прості висловлення, і хибним, якщо хоч одне з простих висловлень хибне.

Кон'юнкція являє собою логічний зв'язок «І» (див. табл. 6.5).

З’єднання двох висловлень читається як x і y. Позначається $x \wedge y$ або $x \cdot y$.

Таблиця 6.5

x	0	0	1	1
y	0	1	0	1
$x \cdot y = x \wedge y$	0	0	0	1

Логічний зв’язок «АБО» (диз’юнкція). Диз’юнкцією називають складне висловлення, що містить декілька простих висловлень і яке є істинним тоді, коли істинним буде хоч одне з простих висловлень, які входять в це складне висловлення, і хибним, якщо всі прості висловлення хибні.

Диз’юнкція являє собою логічний зв’язок «АБО» (табл. 6.6) і позначається $x \vee y$. Читається x або y .

Таблиця 6.6

x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

Логічний зв’язок «НЕ» (заперечення). Логічний зв’язок «НЕ» означає заперечення висловлення і читається «НЕ x», позначається $\bar{\bar{x}}$ (табл. 6.7)

Таблиця 6.7

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

Запереченням висловлення x називають складне висловлення «НЕ x», яке є істинним, коли x хибне, і хибним, коли x істинне.

Для зручності подальших викладок використаємо позначення:

«$\cdot$» – кон’юнкція, «$\vee$» – диз’юнкція і «$\bar{\bar{\ }}$» – заперечення.

Булевою алгеброю називається множина M, що складається не менше ніж з двох елементів, на якій визначені три операції – диз’юнкції ($x \vee y$), кон’юнкції ($x \cdot y$), заперечення ($\bar{\bar{x}}$). Для будь-яких елементів $x.y.z \in M$ виділяємо набір незалежних властивостей, які вважають аксіомами булевої алгебри, а саме:

– закон комутативності:

$$\left.\begin{array}{l} x \vee y = y \vee x \\ x \cdot y = y \cdot x \end{array}\right\} \quad (6.1)$$

– закон асоціативності:

$$\left.\begin{array}{l}(x \vee y) \vee z = x \vee (y \vee z) \\ (x \cdot y)z = x(y \cdot z)\end{array}\right\} \quad (6.2)$$

– закон дистрибутивності:

$$\left.\begin{array}{l}x(y \vee z) = xy \vee xz) \\ x \vee (y \cdot z) = (x \vee y)(x \cdot z)\end{array}\right\} \quad (6.3)$$

для спрощення формул крім аксіом використовують такі співвідношення або закони алгебри логіки:

– логічне додавання до нуля:

$$x \vee 0 = x \ ; \quad (6.4)$$

– логічне додавання до одиниці:

$$x \vee 1 = 1 \ ; \quad (6.5)$$

– логічне множення на 0:

$$x \cdot 0 = 0 \ ; \quad (6.6)$$

– логічне множення на 1:

$$x \cdot 1 = x \ ; \quad (6.7)$$

– закон протиріччя:

$$x \cdot \bar{x} = 0 \ ; \quad (6.8)$$

– закон виключеного третього:

$$x \vee \bar{x} = 1 \ . \quad (6.9)$$

Всі інші закони є наслідком зазначених вище:

– закон ідемпотентності:

$$\left.\begin{array}{l}x \vee x \vee x = x \\ x \cdot x \cdot x = x\end{array}\right\} \ ; \quad (6.10)$$

– закон подвійного заперечення:

$$\bar{\bar{x}} = x \ ; \quad (6.11)$$

– закон поглинання (x поглинає y):

$$\left.\begin{array}{l}x \vee xy = x \\ (x \vee y)x = x\end{array}\right\} \ ; \quad (6.12)$$

– закон де Моргана:

$$\overline{x \vee y} = \bar{x}\,\bar{y} \quad (6.13)$$

$$\overline{xy} = \bar{x} \vee \bar{y} \quad (6.14)$$

– наслідки законів де Моргана:

$$x \vee y = \overline{\bar{x} \wedge \bar{y}} \ ; \quad (6.15)$$

$$xy = \overline{\bar{x} \vee \bar{y}} \ . \quad (6.16)$$

За допомогою розглянутих співвідношень можна виконувати різні тотожні перетворення булевих виразів.

При цьому порядок виконання дій такий:

При відсутності дужок виконуються операції заперечення, потім кон’юнкції, останніми – диз’юнкції.

Подання одних функцій алгебри логіки через інші

1. Операція заборони:

$$x_1 \Delta x_2 = x_1 \cdot \overline{\overline{x_2}} \ . \quad (6.17)$$

Для доведення цього і наступних співвідношень будемо підставляти в ліву і праву частини виразу окремі значення аргументів і перевіряти правильність рівності.

Таблиця 6.8

x_1	x_2	$x_1 \Delta x_2$
0	0	0
0	1	0
1	0	1
1	1	0

Таблиця 6.9

x_1	x_2	$\overline{x_2}$	$x_1 \cdot \overline{x_2}$
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

2. Сума за модулем 2:

$$x_1 \oplus x_2 = x_1 \cdot \overline{x_2} \vee \overline{x_1} \cdot x_2 = (x_1 \vee x_2) \cdot (\overline{x_1} \vee \overline{x_2}) \ . \quad (6.18)$$

Таблиця 6.10

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

Таблиця 6.11

x_1	x_2	$x_1 \cdot \overline{x_2}$	$\overline{x_1} \cdot x_2$	$x_1 \cdot \overline{x_2} \vee \overline{x_1} \cdot x_2$
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

3. Операція Пірса:

$$x_1 \downarrow x_2 = \overline{x_1 \vee x_2} \text{ (операція АБО-НЕ).} \quad (6.19)$$

Таблиця 6.12

x_1	x_2	$x_1 \vee x_2$	$\overline{x_1} \vee \overline{x_2}$	$(x_1 \vee x_2) \cdot (\overline{x_1} \vee \overline{x_2})$
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	0	0

Таблиця 6.13

x_1	x_2	$x_1 \downarrow x_2$	$\overline{x_1 \vee x_2}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

4. Логічна рівнозначність:

$$x_1 \sim x_2 = \overline{\overline{x_1 \oplus x_2}} = x_1 \cdot x_2 \vee \overline{x_1} \cdot \overline{x_2} = (\overline{x_1} \vee x_2) \cdot (x_1 \vee \overline{x_2}). \quad (6.20)$$

Справедливість першої рівності може бути встановлена безпосередньо по таблицях істинності функції логічної рівнозначності і суми по модулю 2; наступних рівностей - шляхом інвертування лівої і правої частин виразу і перетворення за формулами де Моргана.

5. Імплікація:

$x_1 \to x_2 = \overline{x_1} \vee x_2$. (6.21)

x_1	x_2	$x_1 \to x_2$
0	0	1
0	1	1
1	0	0
1	1	1

x_1	x_2	$\overline{x_1}$	$\overline{x_1} \vee x_2$
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

6. Функція Шеффера:

$x_1 / x_2 = \overline{x_1 \cdot x_2}$ (операція І-НЕ). (6.22)

x_1	x_2	x_1 / x_2
0	0	1
0	1	1
1	0	1
1	1	0

x_1	x_2	$x_1 \cdot x_2$	$\overline{x_1 \cdot x_2}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

3. Основні властивості функцій алгебри логіки. Поняття про логічний базис.

Функції кон'юнкції та диз'юнкції мають ряд властивостей, аналогічних властивостям звичайних операцій множення та додавання. Легко переконатися в тому, що для цих функцій мають місце: комутативний закон для кон'юнкції та диз'юнкції:

$$x_1 x_2 = x_2 x_1 ;$$
$$x_1 \vee x_2 = x_2 \vee x_1 .$$

Для доведення цього закону необхідно замість аргументів підставляти відповідно значення 0 або 1, а потім робити порівняння стовпців в таблицях для функцій, що знаходяться в лівій та правій частинах розглядуваного співвідношення.

Функції кон'юнкції та диз'юнкції підпорядковуються асоціативному закону :

$$x_1 (x_2 x_3) = (x_1 x_2) x_3 \qquad x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3 .$$

Дистрибутивний закон:

$$x_1 (x_2 \vee x_3) = (x_1 x_2) \vee (x_1 x_3) ;$$
$$x_1 \vee (x_2 x_3) = (x_1 \vee x_2)(x_1 \vee x_3) .$$

Перевіримо справедливість цього закону для диз'юнкції відносно кон'юнкції шляхом порівняння стовпців в таблицях для функцій, що знаходяться в лівій та правій частинах розглядуваного співвідношення:

x_1	x_2	x_3	$x_2 x_3$	$x_1 \vee (x_2 x_3)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

x_1	x_2	x_3	$x_1 \vee x_2$	$x_1 \vee x_3$	$(x_1 \vee x_2)(x_1 \vee x_2)$
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

Збіг крайніх стовпців в побудованих таблицях доводить наше твердження.

Розглянемо тепер ряд простих, але вельми важливих для функцій кон'юнкції та диз'юнкції співвідношень, які в подальшому будемо використовувати в задачах мінімізації логічних функцій:

– закони логічного додавання та множення з константою одиниці:

$$\left.\begin{array}{l} x \vee 1 = 1; \\ x1 = x. \end{array}\right\}$$

x	1	$x \vee 1$
0	1	1
1	1	1

x	1	$x1$
0	1	0
1	1	1

– закони логічного додавання та множення з константою нуля:

$$\left.\begin{array}{l} x \vee 0 = x; \\ x0 = 0. \end{array}\right\}$$

x	0	$x \vee 0$
0	0	0
1	0	1

x	0	$x0$
0	0	0
1	0	0

– закон ідемпотентності:

$$\left.\begin{array}{l} x \vee x = x; \\ xx = x. \end{array}\right\}$$

x	x	$x \vee x$
0	0	0
1	1	1

x	x	xx
0	0	0
1	1	1

– закон виключеного третього та закон протиріччя:

$$\left.\begin{array}{l} x \vee \bar{x} = 1; \\ x\bar{x} = 0. \end{array}\right\}$$

x	$\bar{x}$	$x \vee \bar{x}$
0	1	1
1	0	1

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

– закон подвійного заперечення:

$$\bar{\bar{x}} = x.$$

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

– формули де Моргана:

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

$$\overline{xy} = \bar{x} \vee \bar{y}.$$

Доведення правила де Моргана:

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

З таблиці видно, що при різних значеннях x і y права і ліва частини однакові. Формули де Моргана можна використовувати і для випадку, коли функція має більше двох аргументів, нижче наведені відповідні співвідношення:

$$x_1 \vee x_2 \vee \ldots \vee x_n = \overline{\bar{x}_1 \bar{x}_2 \ldots \bar{x}_n};$$

$$x_1 x_2 \ldots x_n = \overline{\bar{x}_1 \vee \bar{x}_2 \vee \ldots \vee \bar{x}_n}$$

Аналогічним чином доводяться інші тотожності.

Властивості функції додавання за модулем 2 та функції імплікації часто бувають корисними при аналізі та синтезі різних дискретних приладів.

Для функції додавання за модулем 2 мають місце переставний та сполучний закони, а також розподільний закон відносно кон'юнкції:

$$x_1 \oplus x_2 = x_2 \oplus x_1;$$
$$x_1 \oplus (x_2 \oplus x_3) = (x_1 \oplus x_2) \oplus x_3;$$
$$x_1 \,\&\, (x_2 \oplus x_3) = (x_1 \,\&\, x_2) \oplus (x_1 \,\&\, x_3).$$

Мають місце також очевидні співвідношення:

$$\left.\begin{array}{l} x \oplus x = 0; \\ x \oplus 0 = x; \\ x \oplus 1 = \bar{x}; \\ x \oplus \bar{x} = 1. \end{array}\right\}$$

Крім того, має місце формула

$$x_1 \vee x_2 = x_1 \oplus x_2 \oplus x_1 x_2.$$

На відміну від усіх розглянутих раніше функцій для імплікації не мають місця переставний та сполучний закони:

$$\left.\begin{array}{l} x \to \bar{x} = 1; \\ x \to \bar{\bar{x}} = x; \\ x \to 1 = 1; \\ x \to 0 = \bar{x}; \\ 0 \to x = 1; \\ 1 \to x = \bar{\bar{x}}; \\ x_1 \to x_2 = \bar{x}_2 \to \bar{x}_1; \\ x_1 \to x_2 \to x_1 = x_1. \end{array}\right\}$$

Функції диз'юнкції та кон'юнкції можуть бути виражені через імплікацію таким чином:

$$x_1 \vee x_2 = \overline{\bar{x}_1 \to x_2};$$
$$x_1 x_2 = x_1 \to \overline{\bar{x}_2}.$$

Доведення співвідношень:

x_1	x_2	$x_1 \vee x_2$	$\bar{x}_1$	$\bar{x}_1 \to x_2$
0	0	0	1	0
0	1	1	1	1
1	0	1	0	1
1	1	1	0	1

x_1	x_2	$x_1 x_2$	$\bar{x}_2$	$x_1 \to \bar{x}_2$	$\overline{x_1 \to \bar{x}_2}$
0	0	0	1	1	0
0	1	0	0	1	0
1	0	0	1	1	0
1	1	1	0	0	1

З таблиць видно, що при різних значеннях x та y права і ліва частина формул однакові.

Для функцій штрих Шеффера і операція Пірса має місце переставний закон

$$x_1 / x_2 = x_2 / x_1;$$
$$x_1 \downarrow x_2 = x_2 \downarrow x_1.$$

Сполучний закон для них не виконується:

$$x_1 / (x_2 / x_3) \neq (x_1 / x_2) / x_3;$$
$$x_1 \downarrow (x_2 \downarrow x_3) \neq (x_1 \downarrow x_2) \downarrow x_3.$$

Мають місце такі очевидні співвідношення:

$$\left.\begin{array}{ll} x/x=\bar{x}; & x\downarrow x=\bar{x};\\ x/\bar{x}=1; & x\downarrow \bar{x}=0;\\ x/1=\bar{x}; & x\downarrow 1=0;\\ x/0=1; & x\downarrow 0=\bar{x};\end{array}\right\}$$

$$x_1/x_2=\overline{x_1x_2}=\bar{x}_1\vee\bar{x}_2;$$

$$x_1\downarrow x_2=\overline{x_1\vee x_2}=\bar{x}_1\bar{x}_2.$$

В силу відсутності сполучного закону дії розкриття дужок та винесення за дужки для функцій штрих Шеффера і операція Пірса специфічні та виконуються за такими правилами:

$$\left.\begin{array}{l}(x_1/x_2)/(x_1/x_3)=\overline{x_1/(\bar{x}_2/\bar{x}_3)}=\bar{x}_1\downarrow(x_2\downarrow x_3);\\ (x_1\downarrow x_2)\downarrow(x_1\downarrow x_3)=\overline{x_1\downarrow(\bar{x}_2\downarrow\bar{x}_3)}=\bar{x}_1/(x_2/x_3);\\ (x_1/x_2)\downarrow(x_1/x_3)=\overline{x_1/(\bar{x}_2\downarrow\bar{x}_3)}=\bar{x}_1\downarrow(x_2/x_3);\\ (x_1\downarrow x_2)/(x_1\downarrow x_3)=\overline{x_1\downarrow(\bar{x}_2/\bar{x}_3)}=\bar{x}_1/(x_2\downarrow x_3);\\ (x_1/x_2)\downarrow(x_3/x_4)=\bar{x}_1\downarrow\overline{\bar{x}_2\downarrow\overline{\bar{x}_3\downarrow\bar{x}_4}};\\ (x_1\downarrow x_2)/(x_3\downarrow x_4)=\bar{x}_1/\overline{\bar{x}_2/\overline{\bar{x}_3/\bar{x}_4}}.\end{array}\right\}$$

Доведення справедливості цих співвідношень аналогічне. Доведемо, наприклад, справедливість рівності

$$(x_1\downarrow x_2)/(x_1\downarrow x_3)=\overline{x_1\downarrow(\bar{x}_2/\bar{x}_3)}.$$

Використовуючи два останніх співвідношення з (1.30), перетворимо обидві частини цього співвідношення таким чином:

$$(x_1\downarrow x_2)/(x_1\downarrow x_3)=\overline{x_1\vee x_2}/\overline{x_1\vee x_3}=$$
$$=(x_1\vee x_2)\vee(x_1\vee x_3)=x_1\vee x_2\vee x_3\quad;$$
$$\overline{x_1\downarrow(\bar{x}_2/\bar{x}_3)}=x_1\vee(\bar{x}_2/\bar{x}_3)=x_1\vee(x_2\vee x_3)=x_1\vee x_2\vee x_3.$$

Збіг лівої та правої частин після проведення еквівалентних перетворень доводить рівність.

Функції штрих Шеффера і операція Пірса пов’язані між собою співвідношеннями, аналогічними до формул де Моргана для функцій кон’юнкції та диз’юнкції:

$$\left.\begin{array}{l}x_1/x_2=\overline{\bar{x}_1\downarrow\bar{x}_2};\\ x_1\downarrow x_2=\overline{\bar{x}_1/\bar{x}_2}.\end{array}\right\}$$

$$\overline{x_1x_2}=\bar{x}_1\vee\bar{x}_2;$$

$$x_1x_2=\overline{\bar{x}_1\vee\bar{x}_2}.$$

Оскільки отримане співвідношення є формулою де Моргана, то перше зі співвідношень справедливе. Для другого співвідношення доведення аналогічне.

Поняття про логічний базис

Набір логічних операцій, що дозволяє аналітично описати будь-яку логічну функцію, називається ***функціонально повним набором*** або ***логічним базисом***.

Такий набір складають основні логічні операції АБО, І, НЕ, тому він є одним з логічних базисів. Логічний базис називається мінімальним, якщо видалення з набору хоча б однієї операції перетворює його в функціонально неповний.

Логічний базис НЕ, АБО, І не є мінімальним, так як на підставі законів подвійності можна виключити з логічних виразів операцію АБО або І, отже, він є надлишковим базисом.

Мінімальний базис складають дві операції НЕ, АБО і НЕ, І. Практичного уваги заслуговують мінімальні базиси, що представляють собою тільки одну операцію. До них відносяться операції логічного множення з запереченням (І-НЕ, штрих Шеффера) і логічного додавання з запереченням (АБО-НЕ, стрілка Пірса).

Контрольні питання:

1. Яку функцію називають функцією алгебри логіки?
2. Які значення можуть мати аргументи функцій алгебри логіки?
3. Що називається набором логічної функції?
4. Скільки наборів має будь-яка логічна функція?
5. Яка існує кількість логічних функцій, які мають 2 аргументи?
6. Яка існує кількість логічних функцій, які мають n аргументів?
7. Які є логічні функції від одного аргументу?
8. Які є логічні функції від двох аргументів?
9. Навести таблицю істинності логічної функції «кон'юнкція».
10. Навести таблицю істинності логічної функції «диз'юнкція».
11. Навести таблицю істинності логічної функції «сума за модулем 2».
12. Навести таблицю істинності логічної функції «операція Пірса».
13. Навести таблицю істинності логічної функції «операція Шеффера».
14. Які основні властивості логічних функцій?
15. Як виконується закон протиріччя для кон'юнкції та закон виключеного третього для диз'юнкції?
16. Як виконується закон ідемпотентності для кон'юнкції та для диз'юнкції?
17. Як виконується закон подвійного заперечення?
18. Як виконуються закони з константами для кон'юнкції (логічне множення на одиницю, логічне множення на нуль)?
19. Як виконуються закони з константами для диз'юнкції (логічне додавання до одиниці, логічне додавання з нулем)?
20. Як перевірити справедливість формул де Моргана?
21. Які існують способи подання одних логічних функцій через інші?
22. Як подати логічну функцію «сума за модулем 22 за допомогою функцій «кон'юнкція», «диз'юнкція» та «заперечення»?

ТЕМА 7. АНАЛІЗ І СИНТЕЗ КОМБІНАЦІЙНИХ СХЕМ

План:

1. Способи реалізації булевих функцій.
2. Аналогія між логічною функцією і комбінаційною схемою.
3. Представлення функцій в ДДНФ і ДКНФ.
4. Мінімізація логічних функцій.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Способи реалізації булевих функцій.

Булеві функції належать до класу двозначних однорідних функцій. Це найпростіший і водночас найважливіший клас однорідних функцій, що використовуються для опису скінченних автоматів та ЕОМ. Останні, у свою чергу, призначаються для перероблення дискретної інформації. Як модель засобів перероблення застосовується поняття автомата. Для формального опису цифрового автомата слугує апарат алгебри логіки (булевої алгебри). Останню утворюють множини всіх булевих функцій разом з операціями заперечення, кон'юнкції, диз'юнкції, імплікації тощо.

Існує декілька способів подання булевих функцій.

1. Вербальний (словесний) спосіб - при цьому способі словесний опис однозначно визначає всі випадки, при яких функція приймає значення 0 або 1. Наприклад, багатовходова функція АБО може мати такий словесний опис: функція приймає значення 1, якщо хоч би один з аргументів приймає значення 1, інакше - 0.

2. Числовий - функція задається у вигляді десяткових (або вісімкових, або шістнадцяткових) еквівалентів номерів тих наборів аргументів, на яких функція приймає значення 1. Умова, що функція f (x1, x2, x3) = 1 на наборах 1,3,5,6,7 записується f (1, 3, 5, 6, 7) = 1. Аналогічним чином булева функція може бути задана по нульових значеннях. При нумерації наборів змінним x1, x2, x3 ставиться у відповідність ваги 2^2, 2^1, 2^0 тобто 6 набору відповідає двійковий еквівалент 110, а 1 набору - 001.

3. Табличний - Функція задається у вигляді *таблиці істинності* (відповідності), яка містить 2^n рядків (по числу наборів аргументів), n стовпців по числу змінних і один стовпець значень функції. У такій таблиці кожному набору аргументів відповідає значення функції. n = 3, число рядків $2^3 = 8$, число можливих функцій три змінних 2 $2^n = 2^8 = 256$.

4. Аналітичне задання функції – опис її аналітичним виразом (формулою). Наприклад,

$$f_1(x_1, x_2, x_3) = x_1, x_2 + x_2(\bar{x}_3 + x_1); \;\; f_2(abc) = abc + \bar{a}\bar{b}\bar{c}.$$

Функція задається у вигляді виразу алгебри, що отримується шляхом застосування яких-небудь логічних операцій до змінних алгебри логіки. застосовуючи операції кон'юнкції і диз'юнкції можна задати функцію виразом f (x_1, x_2, x_3) = x_1x_2 v x_3. Спосіб отримання такого аналітичного опису булевої функції буде розглянутий в подальших розділах.

5. Координатний - при цьому способі завдання таблиця істинності функції представляється у вигляді координатної карти станів, яка часто називається *картою Карно*. Така карта містить 2n кліток по числу наборів всіляких значень n змінних функції. Змінні функції розбиваються на дві групи так, що одна група визначає координати стовпця, а інша - координати рядка. При такому способі побудови клітка визначається координатами змінних, відповідних певному двійковому набору. Усередині клітки карти Карно ставиться значення функції на даному наборі. Змінні в рядках і стовпцях розташовуються так, щоб сусідні клітки карти Карно розрізнялися тільки в одному розряді змінних, тобто були сусідніми. Такий спосіб уявлення дуже зручний для наочності при мінімізації булевих функцій.

x_1 \ x_2x_3	00	01	11	10
0	0	1	1	0
1	0	1	1	1

6. Діаграмний - є способом представлення функціонування схеми, що реалізовує булеву функцію, в часі. Зображається у вигляді системи графіків, у яких вісь X відповідає автоматному часу (моментам часу), а вісь Y відповідає напрузі дискретних рівнів сигналів «логічний 0» (0,4 В) і «логічна 1» (2,4 В).

7. Графічний - Функція задається у вигляді n-мірного одиничного куба, *вершинам* якого відповідають набори значень аргументів і приписані значення функції на цих наборах. Куб названий одиничним, оскільки кожне ребро сполучає вершини, набори яких розрізняються тільки по одній змінній, тобто є *сусідніми*.

Такий спосіб завдання булевих функцій іноді називають геометричним, але найчастіше *кубічним*. Кубічне уявлення найпридатніше для машинних методів аналізу булевих функцій, оскільки дозволяє компактно представляти булеві функції від великої кількості змінних.

Будь-який набір змінних в кубічному представленні булевих функцій прийнято називати *кубом* або *вектором*. Змінні куба називають *координатами*. Кількість змінних в кубі визначає його *міру* (3-мірний...n-мірний). Кількість символів X в кубі визначає його *ранг*. Куб нульового рангу називають 0-куб, першого рангу 1-куб і так далі 1-куб(ребро) покриває 2 набори, 2-куб(грань) покриває 4 набори і так далі Набір кубів, що покривають всі набори функції, називається *покриттям*. Куби, на яких функція рівна 0, називають 0-покриттям,

рівна 1 - 1-покриттям. Кубічне представлення булевих функцій і операції над кубами називається *кубічним численням*.

В обчислювальній техніці булеві функції застосовуються для:

- опису алгоритмів;
- засобів цієї техніки – дискретних пристроїв, які призначаються для перетворення дискретної інформації, що розкладається на елементарні одиниці – біти, які в пристроях реалізуються сигналами, що описуються двійковими змінними – булевими;
- для вирішення деяких економічних задач;
- для вирішення задач цілочисельного програмування.

2. Аналогія між логічною функцією і комбінаційною схемою.

Логічний елемент – це електронний пристрій, який реалізує певну *логічну (перемикальну) функцію*. Сукупність логічних елементів і зв'язків між ними, призначену для перетворення двійкових змінних, називають *логічною схемою*. Логічні схеми поділяють на послідовнісні і комбінаційні.

Комбінаційною називають схему, ***m*** вихідних сигналів якої в кожний момент часу повністю визначаються сукупністю ***n*** її вхідних сигналів в цей самий момент часу. Тобто вихідні сигнали комбінаційної схеми в даний момент часу не залежать від вхідних сигналів, які діяли в попередні моменти часу (схема не має пам'яті). Кажуть, що така схема має один стан.

Початковими даними (технічним завданням) для проектування комбінаційного вузла є його функціональний опис та вимоги до основних електричних параметрів. Функціональний опис комбінаційного вузла зазвичай задається у вигляді таблиці істинності або алгебраїчного виразу.

Процес проектування розбивається на декілька послідовних етапів:

1) вибір елементної бази та способу реалізації;
2) мінімізація заданої логічної функції;
3) перетворення мінімізованої логічної функції та синтез логічної схеми;
4) синтез електричної схеми;
5) аналіз та оптимізація електричної схеми.

Вибір елементної бази визначається вимогами, пропонованими до електричних параметрів комбінаційного вузла: швидкодією, потужністю, що споживається, завадостійкістю та ін.

Графічне зображення комбінаційної схеми, при якому показані зв'язки між різними елементами, а самі елементи представлені умовними позначеннями, називається функціональною схемою.

Поведінка комбінаційної схеми описується системою логічних функцій. Виділяють задачі аналізу та синтезу комбінаційних схем.

Задача аналізу комбінаційної схеми полягає в знаходженні системи логічних функцій, що відображають логіку роботи такої схеми. В процесі аналізу з схеми вилучають елементи, що не впливають на логіку її роботи (формувачі,

елементи узгодження і т.д.), після чого визначають згадану систему логічних функцій.

Задача синтезу є оберненою до задачі аналізу.

Задачу синтезу комбінаційної схеми можна поставити так. Описати аналітично закон функціонування КС, за визначеним законом функціонування КС необхідно побудувати схему, яка реалізує цей закон, при мінімальних витратах на обладнання.

Закон функціонування комбінаційної схеми можна задати:

- за допомогою словесного опису через прості та складні висловлювання;
- за допомогою таблиць істинності;
- у вигляді рівнянь або систем рівнянь аналітичних функцій алгебри логіки;
- за допомогою рівнянь ненульових коефіцієнтів ФАЛ;
- за допомогою карт - Карно;
- за допомогою часової діаграми.

Словесний опис функціонування КС простими висловлюваннями, об'єднаних потім за допомогою сполучників НІ, АБО, І в складні, є основою для побудови таблиці істинності, а інколи й логічних функцій. Пошук рівнянь з ненульовими коефіцієнтами, об'єднання сусідніх мінтермів карт Карно використовується при мінімізації (спрощенні) логічних виразів. Часові діаграми використовують при аналізі явища «запізнень» при «гонках» в елементах схеми.

Процес синтезу комбінаційних схем містить в собі такі етапи та операції:

Опис закону функціонування – постановка задачі, виявлення умов функціонування та їх залежність від вхідних сигналів, зв'язок з іншими пристроями, побудова словесного опису роботи КС за допомогою сполучників НІ, І, АБО, побудова опису функціонування за допомогою таблиць істинності, визначення та позначення вхідних та вихідних змінних.

Аналітичний аналіз – побудова логічних функцій булєвої алгебри, а якщо вихідних функцій більше однієї, то побудова системи логічних функцій булєвої алгебри, використовуючи таблиці істинності, в формі досконалих ДНФ (КНФ); знаходження мінімізованих ДНФ (КНФ), простих імплікант, тупикових форм за допомогою засобів мінімізації, наприклад, карт Карно, ненульових коефіцієнтів, Квайна та інших; перетворення мінімізованих булєвих функцій, якщо потрібно, до вибраного або заданого базису логічних функцій (Пірса, Шеффера).

Побудова логічних схем – визначення рівнів логіки КС за результатом аналізу послідовності виконання логічних операцій, розподілення змінних за кількістю входів вибраних логічних елементів, побудова логічної схеми, перевірка адекватності виконання функцій схемою та аналітичними ФАЛ.

В теорії цифрових пристроїв комбінаційною логікою (комбінаційною схемою) називають логіку функціонування пристроїв комбінаційного типу. У комбінаційних пристроїв стан виходу однозначно визначається набором вхідних сигналів. Це відрізняє комбінаційну логіку від секвенційної логіки, в рамках якої

вихідне значення залежить не тільки від поточного вхідного впливу, але й від передісторії функціонування цифрового пристрою. Іншими словами, секвенційна логіка припускає наявність пам'яті, яку комбінаційна логіка не передбачає.

3. Представлення функцій в ДДНФ і ДКНФ.

Нормальна форма заданої функції називається мінімальною, якщо кількість букв (літералів), яку вона містить, буде не більше, ніж у будь-який інший НФ тієї ж функції. Саме букв, а не змінних!

Мінімальна форма представлення ФАЛ - це така форма, що містить мінімальну кількість термів, які мають мінімальні ранги.

Методи, які дають змогу для будь-якої логічної функції записати булевий вираз, ґрунтуються на тому, що вводяться вирази певного типу - канонічні форми, а потім формуються досить прості правила запису будь-якої функції у цих формах.

Як канонічні звичайно використовуються досконалі диз'юнктивна та кон'юнктивна нормальні форми (ДДНФ і ДКНФ).

При описі функції алгебри логіки алгебраїчним виразом використовують дві стандартні форми її подання:

1) ***диз'юнктивною нормальною формою (ДНФ)*** називається логічна сума елементарних логічних добутків, в кожному з яких аргумент або його інверсія входить один раз. Отримана ДНФ може бути з таблиці істинності з використанням наступного алгоритму:

а) для кожного набору змінних, на якому функція алгебри логіки (ФАЛ) дорівнює одиниці, записують елементарні логічні добутки вхідних змінних. Причому змінні, рівні нулю, записують з інверсією. Отримані добутки називають конституентами одиниці.

б) логічно підсумовують всі конституенти одиниці. ДНФ, отриману внаслідок підсумовування конституентів одиниці, називають досконалою (ДДНФ).

2) ***кон'юнктивною нормальною формою (КНФ)*** називається логічний добуток елементарних логічних сум, в кожному з яких аргумент або його інверсія входять один раз. КНФ може бути знайдена з таблиці істинності з використанням наступного алгоритму:

а) для кожного набору змінних, на якому ФАЛ дорівнює нулю, записують елементарні логічні суми вхідних змінних. Причому змінні, значення яких дорівнюють 1, записують з інверсією. Отримані суми називають конституєнтамі нуля.

б) логічно перемножують всі отримані конституєнти нуля. КНФ, отриману перемноженням конституєнтів нуля, називають досконалою (ДКНФ).

Процес спрощення зводиться до використання загальних властивостей для зменшення загальної кількості входжень до кожної формули змінних і символів логічних операцій. Зазвичай перетворення виконують у досконалій диз'юнктивний нормальній формі (ДДНФ) (сумі мінтермів), а використовують

властивості склеювання, поглинання і виявлення. Склеювання дозволяє замінити два мінтерма, які відрізняються однією змінною (з інверсією і без) одним мінтермом більш низького рівня і рангу. Поглинання дозволяє виключити всі співмножники, в які в якості співмножника входить інший мінтерм більш низького рівня. Процедура склеювання не забезпечує отримування мінімальних форм, а приводить до скороченої форми, мінтерми якої називаються простими імплікантами. Серед простих імплікант можуть бути такі імпліканти, які покриваються іншими імплікантами, тобто є надлишковими. Після вилучення надлишкових імплікант отримують тупикові форми, серед яких знаходяться і мінімальні форми. Пошук мінімальних форм серед нормальних форм конкретної функції є головною задачею синтезу логічних схем.

Запис булевих функцій (ФАЛ) у вигляді ДДНФ і ДКНФ часто виявляється не економічним, що відчувається на етапі структурного синтезу схем. Покажемо це на прикладі.

Приклад 7.1. Нехай задано ДДНФ

$$f(x,y,z) = x \cdot y \cdot z \vee x \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot z \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot y \cdot z$$

Перетворимо цю ДДНФ шляхом додавання ще одного кон'юнктивного члена x * y *z. Це додавання не змінює даної функції, тому що $A \vee A = A$.Тоді:

$$f(x,y,z) = x \cdot y \cdot z \vee x \cdot y \cdot \bar{z} \vee x \cdot \bar{y} \cdot z \vee x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot y \cdot z \vee x \cdot y \cdot z$$

Тепер перетворимо цей вираз, використовуючи сполучний і розподільний закони для кон'юнкції і диз'юнкції:

$$f(x,y,z) = x \cdot y \cdot (z \vee \bar{z}) \vee x \cdot \bar{y} \cdot (z \vee \bar{z}) \vee y \cdot z \cdot (x \vee \bar{x})$$

Оскільки $A \vee \overline{A} = 1$, одержимо (склеювання по A)

$$f(x,y,z) = x \cdot y \vee x \cdot \bar{y} \vee y \cdot z$$

Аналогічно попередньому, робимо подальші перетворення

$$f(x,y,z) = x \cdot (y \vee \bar{y}) \vee y \cdot z = x \vee yz$$

З прикладу видно неекономність нормальних форм для представлення ФАЛ. Так виникає проблема найпростішого представлення функцій, а вона складається у виборі базису і проблемі найбільш ощадливого представлення функцій у цьому базисі.

В даний час істотні результати отримані лише в базисі ТА, АБО, НЕ.

Для уточнення постановки задачі про мінімізацію нагадаємо та додамо ряд визначень:

Кон'юнкція $x_1\ x_2\ \ldots\ x_n$ називається елементарною, якщо в цій кон'юнкції кожна змінна зустрічається не більше одного разу.

Рангом елементарної кон'юнкції називається число букв, що утворюють цю кон'юнкцію.

Диз'юнкція елементарних кон'юнкцій називається **диз'юнктивною нормальною формою (ДНФ)**

ДНФ $f(x_1,x_2,..,x_n)$ елементарних кон'юнкцій, що складається з елементарних кон'юнкцій рангу n, називається **досконалою ДНФ**.

Довжиною ДНФ назвемо число елементарних кон'юнкцій, що утворюють цю ДНФ.

Функції f і φ **називаються еквівалентними**, якщо вони приймають однакові значення на всіх наборах аргументів. Еквівалентні функції можуть відрізнятися формами представлення та ціною.

Під **ціною логічної функції** розуміється кількість букв, які входять в її запис.

ДНФ, що містить найменше число букв x_t у порівнянні з всіма іншими ДНФ, еквівалентними даній функції, називається **мінімальною ДНФ (МДНФ)**.

Аналогічні визначення можна зробити і для КНФ.

Логічну схему, яка реалізує заданий алгоритм перетворення сигналів, можна синтезувати безпосередньо за виразом, представленим у вигляді ДДНФ або ДКНФ. Однак отримана при цьому схема, як правило, не оптимальна з точки зору її практичної реалізації. Тому вихідні ФАЛ зазвичай мінімізують.

Мінімізацією – називається пошук коротких форм представлення, перемикаючих функцій для скорочення числа фізичних елементів призначених для реалізації цих функцій.

Мінімізація досягається за допомогою законів булевої алгебри.

Проблема мінімізації зводиться до відшукання форми представлення логічної функції з мінімальною ціною. Мінімізація дозволяє спростити схеми, які реалізують логічні (перемикальні) функції.

Метою мінімізації логічної функції зазвичай є зменшення вартості її технічної реалізації.

Критеріями мінімізації можуть бути:

– зменшення числа логічних елементів;

– збільшення регулярності внутрішньої структури логічних пристроїв;

– зменшення числа зовнішніх з'єднань в інтегральних схемах.

Метою мінімізації ФАЛ у класі ДДНФ є відшукання МДНФ.

Ця задача виконується в *два* етапи. На *першому* етапі за таблицею істиності будується ДДНФ, на *другому* етапі знаходиться МДНФ. Другий етап неоднозначний, тому що для даної функції може існувати декілька МДНФ і власне одержання МДНФ робиться на підставі перебору.

Існує досить багато методів мінімізації ФАЛ у класі ДДНФ:

1. Аналітичний (розрахунковий) метод, або метод безпосередніх перетворень;
2. Метод Квайна;
3. Метод Квайна-Мак-Класкі;
4. Метод мінімізуючих карт, так звані діаграми Вейча або карти Карно;
5. Метод Петрика;
6. Метод невизначених коефіцієнтів;
7. Метод гіперкубів;
8. Метод функціональної декомпозиції

Ми зосередимо нашу увагу на перших п'яти методах.

4. Мінімізація логічних функцій.

Метод Квайна. Імплікантна матриця

Метод Квайна ґрунтується на застосуванні двох основних співвідношень.

1. Співвідношення склеювання (по змінній **x**): $Ax \vee A\bar{x} = Ax \vee A\bar{x} \vee A$, де A — будь- який елементарний добуток.

2. Співвідношення поглинання: $A\tilde{x} \vee A = A, \tilde{x} \in \{x, \bar{x}\}$.

Справедливість обох співвідношень легко перевіряється. Суть методу полягає в послідовному виконанні всіх можливих склеювань і потім усіх поглинань, що призводить до скороченої ДНФ. Метод застосовується до досконалих ДНФ. Зі співвідношення поглинання випливає, що довільний елементарний добуток поглинається будь-якою його частиною.

Для доказу достатньо показати, що довільна проста імпліканта $p = \tilde{x}_{i_1} \tilde{x}_{i_2} \ldots \tilde{x}_{i_k}$ може бути отримана. Справді, застосовуючи до p операцію розгортання (зворотню операції склеювання): $A = A(x \vee \bar{x}) = Ax \vee A\bar{x}$ по усім відсутнім змінним $x_{i_{k+1}}, \ldots, x_{i_n}$ вихідної функції f, одержуємо сукупність S конституент одиниці. При склеюванні всіх конституент із S одержимо імпліканту p. Останнє очевидно, оскільки операція склеювання зворотна операції розгортання. Множина S конституент обов'язково є присутною в досконалій ДНФ функції f оскільки p — її імпліканта.

Приклад 7.2. Нехай є булева функція, задана таблицею істинності (табл. 5.2). Її ДДНФ має вигляд:

$$f = \bar{x}_1\bar{x}_2\bar{x}_3x_4 \vee \bar{x}_1\bar{x}_2x_3x_4 \vee \bar{x}_1x_2\bar{x}_3x_4 \vee \bar{x}_1x_2x_3x_4 \vee x_1x_2x_3\bar{x}_4 \vee x_1x_2x_3x_4$$

Для зручності позначимо кожну конституенту одиниці з ДДНФ функції f яким-небудь десятковим номером (довільно). Виконуємо склеювання. Конституента 1 склеюється тільки з конституентою 2 (по змінній x_3) і з конституентою 3 (по змінній x_2), конституента 2 - з конституентою 4 і т.д.

x_1	x_2	x_3	x_4	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

В результаті одержуємо:

1—2: $\bar{x}_1\bar{x}_2x_4$

1—3: $\bar{x}_1\bar{x}_3x_4$

2—4: $\bar{x}_1x_3x_4$

3—4: $\bar{x}_1x_2x_4$

4—6: $x_2x_3x_4$

5—6: $x_1x_2x_3$

Помітимо, що результатом склеювання є завжди елементарний добуток, що представляє собою спільну частину, що склеює конституенти.

Далі робимо склеювання одержаних елементарних добутків. Склеюються тільки ті добутки, що містять однакові змінні.

Має місце два випадки склеювання:

$\bar{x}_1\bar{x}_2x_4 \vee \bar{x}_1x_2x_4 = \bar{x}_1x_4(1-2-3-4)$;

$\bar{x}_1\bar{x}_3x_4 \vee \bar{x}_1x_3x_4 = \bar{x}_1x_4(1-3-2-4)$, з появою того самого елементарного добутку $\bar{x}_1x_4$.

Подальші склеювання неможливі.

Зробивши поглинання (з отриманої ДНФ викреслюємо всі елементарні добутки, що поглинаються), одержимо скорочену ДНФ:

$f_{скор} = x_2x_3x_4 \vee x_1x_2x_3 \vee \bar{x}_1x_4$.

Переходимо до **другого етапу**. Для одержання мінімальної ДНФ необхідно забрати зі скороченої ДНФ усі зайві прості імпліканти. Це робиться за допомогою спеціальної **імплікантної матриці Квайна**. Рядки такої матриці відзначаються простими імплікантами булевої функції, тобто членами скороченої ДНФ, а стовпці — конституентами одиниці, тобто членами ДДНФ булевої функції.

Імплікантна матриця для прикладу 7.2 має вигляд, поданий у таблиці 7.1.

Таблиця 7.1

Прості імпліканти	Конституенти одиниці					
	$\bar{x}_1\bar{x}_2\bar{x}_3x_4$	$\bar{x}_1\bar{x}_2x_3x_4$	$\bar{x}_1x_2\bar{x}_3x_4$	$\bar{x}_1x_2x_3x_4$	$x_1x_2x_3\bar{x}_4$	$x_1x_2x_3x_4$
$\bar{x}_1x_4$	x	x	x	x		
$x_2x_3x_4$				x		x
$x_1x_2x_3$					x	x

Як уже відзначалося, проста імпліканта поглинає деяку конституенту одиниці, якщо є її власною частиною. Відповідна клітинка імплікантної матриці на перетині рядка (з розглянутою простою імплікантою) і стовпця (з конституентою одиниці) відзначається хрестиком (табл. 7.1). Мінімальні ДНФ будуються за імплікантною матрицею у такий спосіб:

1) шукаються стовпці імплікантної матриці, що мають тільки один

хрестик. Відповідні цим хрестикам прості імпліканти називаються базисними і складають так зване **ядро булевої функції**. Ядро обов'язково входить у мінімальну ДНФ.

2) розглядаються різні варіанти вибору сукупності простих імплікант, що накриють хрестиками інші стовпці імплікантної матриці, і обираються варіанти з мінімальним сумарни числом букв у такій сукупності імплікант.

Ядром функції з прикладу 7.2 є імпліканти $x_1x_2x_3$; $\bar{x}_1x_4$. Імпліканта $x_2x_3x_4$— зайва, тому що ядро накриває всі стовпці імплікантної матриці. Тому функція має єдину тупикову і мінімальну ДНФ:

$$f = \bar{x}_1x_4 \vee x_1x_2x_3.$$

Слід зазначити, що число N хрестиків в одному рядку завжди є степенем 2. Більш того, можна легко переконатися в тому, що $N = 2^{n-k}$, де k — число букв, що містяться в простій імпліканті.

Відзначимо також, що використовуючи різні співвідношення, можна розширити область застосування методу Квайна за межі досконалої ДНФ.

Приклад 7.3. Мінімізувати булеву функцію f методом Квайна:

$$f(x_1x_2x_3) = \overline{(\bar{x}_1 \vee \bar{x}_3)(x_2 \vee \bar{x}_3)} \vee \bar{x}_1\bar{x}_2 \vee x_1(x_2 \vee \bar{x}_2x_3) \quad .$$

1. Уникаємо заперечень і дужок, використовуючи наведені раніше співвідношення

$$f = \overline{(\bar{x}_1 \vee \bar{x}_3)} \vee \overline{(\bar{x}_2 \vee \bar{x}_3)} \vee \bar{x}_1\bar{x}_2 \vee x_1x_2 \vee x_1\bar{x}_2x_3 = x_1x_3 \vee \bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2 \vee x_1x_2$$

2. Відновлюємо ДДНФ функції f, застосовуючи операцію розгортання

$$f = x_1x_3(x_2 \vee \bar{x}_2) \vee \bar{x}_2x_3(x_1 \vee \bar{x}_1) \vee \bar{x}_1\bar{x}_2(x_3 \vee \bar{x}_3) \vee x_1x_2(x_3 \vee \bar{x}_3) =$$

$$= x_1x_2x_3 \vee x_1\bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2x_3 \vee x_1x_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2\bar{x}_3$$

3. Знайдемо скорочену ДНФ функції f, роблячи в ДДНФ усі можливі склеювання:

$$f = x_1x_3 \vee x_1x_2 \vee \bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2 \quad .$$

4. Шукаємо мінімальну ДНФ функції f за імплікантною матрицею (табл. 7.2)

Таблиця 7.2

Прості імпліканти системи функцій	Конституенти одиниці функції φ					
	$x_1x_2x_3$	$x_1\bar{x}_2x_3$	$\bar{x}_1\bar{x}_2x_3$	$\bar{x}_1\bar{x}_2\bar{x}_3$	$x_1x_2\bar{x}_3$	
x_1x_3	X	X				
x_1x_2	X				X	
$\bar{x}_2x_3$		X	X			
$\bar{x}_1\bar{x}_2$			X	X		

Ядро булевої функції: x_1x_2, $\bar{x}_1\bar{x}_2$.

Мінімальні ДНФ:

$f = x_1x_2 \vee \bar{x}_1\bar{x}_2 \vee x_1x_3$;

$f = x_1x_2 \vee \bar{x}_1\bar{x}_2 \vee \bar{x}_2x_3$.

Метод Квайна-Мак-Класкі

Метод являє собою формалізований на етапі відшукання простих імплікант метод Квайна. Формалізація виконується у такий спосіб:

1. Всі конституенти одиниці з ДДНФ булевої функції ***f*** записуються їхніми двійковими номерами.
2. Всі номери розбиваються на групи, що не перетинаються. Ознакою утворення ***i***-*ї* групи є наявність ***i*** одиниць у кожному двійковому номері конституенти одиниці.
3. Склеювання роблять тільки між номерами сусідніх груп. Номери, що склеюються, позначаються яким-небудь знаком (закреслюванням або *).
4. Склеювання роблять всі можливі, як і в методі Квайна. Непозначені після склеювання номери є простими імплікантами.
5. Будується імплікантна матриця Квайна.
6. За імплікантною матрицею Квайна відбувається пошук мінімальної ДНФ.

Більш докладно розглянемо метод на наступному прикладі.

Приклад 7.5. Мінімізувати методом Квайна—Мак-Класкі булеву функцію ***f***, задану таблицею істинності (табл. 5.2).

1. У ДДНФ функції ***f*** замінимо всі конституенти одиниці їхніми двійковими номерами:

$$f = 0001 \vee 0011 \vee 0101 \vee 0111 \vee 1110 \vee 1111 .$$

2. Утворимо групи двійкових номерів. Ознакою утворення ***i***-*ї* групи є ***i*** одиниць у двійковому номері конституенти одиниці (табл. 7.5).

Таблиця 7.5

Номер групи	Двійкові номери конституент одиниці
0	–
1	**0001 ***
2	**0011 *** **0101 ***
3	**0111 *** **1110 ***
4	**1111 ***

1. Склеїмо номери із сусідніх груп табл. 7.5. Номери, що склеюються, викреслимо. Результати склеювання занесемо в табл. 7.6. Склеїмо номери із сусідніх груп табл. 7.6. Склеюватися можуть тільки ті номери, що мають знаки « - » в однакових позиціях. Номери, що склеюються, викреслимо. Результати склеювання занесемо в табл. 7.7.

Таблиця 7.6

Номер групи	Двійкові номери імплікант
1	**00-1 *** **0-01 ***
2	**0-11 *** **01-1 ***
3	**-111** **111-**

Таблиця 7.7

Номер групи	Двійкові номери імплікант
1	**0--1** **0--1**

4. Маємо три прості імпліканти: **-111, 111-, 0--1.**
5. Будуємо імплікантну матрицю (табл. 7.8).

Таблиця 7.8

Двійкові номери простих імплікант				Двійкові номери конституент					
				0001	**0011**	**0101**	**0111**	**1110**	**1111**
0	**-**	**-**	**1**	**x**	**x**	**x**	**x**		
-	**1**	**1**	**1**				**x**		**x**
1	**1**	**1**	**-**					**x**	**x**

За таблицею визначаємо сукупність простих імплікант: **0--1** і **111-**, що відповідає мінімальній ДНФ. Для відновлення буквеного вигляду простої імпліканти досить виписати добутки тих змінних, котрі відповідають збереженим двійковим цифрам:

$0--1 \to x_1x_4$; $111- \to x_1x_2x_3$.

Зазначимо, що розбиття конституент на групи дозволяє зменшити число попарних порівнянь при склеюванні.

Метод діаграм Вейча

Метод дозволяє швидко одержувати мінімальні ДНФ булевої функції ***f*** невеликої кількості змінних. В основі методу лежить представлення булевих функцій діаграмами деякого спеціального виду, що одержали назву діаграм Вейча.

Для булевої функції двох змінних діаграма Вейча має вигляд (табл. 7.9). Кожна клітинка (комірка) діаграми відповідає набору змінних булевої функції в її таблиці істиності. Ця відповідність показана в таблицях 7.9 – 7.11. У клітинці діаграми Вейча ставиться одиниця, якщо булева функція приймає одиничне значення на відповідному наборі. Нульові значення булевої функції в діаграмі Вейча не проставляються.

Для булевої функції трьох змінних діаграма Вейча має наступний вигляд (табл. 7.10).

Таблиця 7.9

	x_1	$\overline{x_1}$
x_2	11	01
$\overline{x_2}$	10	00

Таблиця 7.10

	x_1		$\overline{x_1}$	
x_2	110	111	011	010
$\overline{x_2}$	100	101	001	000
	$\overline{x_3}$	x_3		$\overline{x_3}$

Додавання до неї ще такої ж таблиці дає діаграму для функції 4-х змінних (табл. 7.11). У такий же спосіб, тобто приписуванням ще однієї діаграми 4-х змінних до щойно розглянутого, можна одержати діаграму для функції 5-ти змінних і т.д., однак діаграми для функцій з числом змінних більше 4-х використовуються рідко.

Для наведених діаграм характерно наступне:

1) кожній клітинці діаграми відповідає свій набір;
2) сусідні набори розташовані поруч у рядку або в стовпці.

Сусідніми наборами називаються набори, що відрізняються одним компонентом і підлягають склеюванню. Нагадаємо, що конституенти, які відповідають таким наборам, склеюються (див. метод Квайна-Мак-Класкі).

Приклад 7.6. Для функції, заданої табл. 7.12, конституенти, які відповідають парі одиниць у лівій частині таблиці, склеюються і породжують елементарний добуток з 2-х букв:

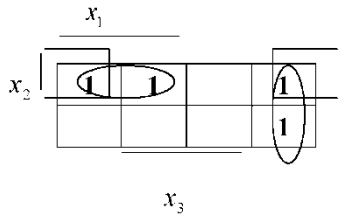
$x_1x_2\bar{x}_3 \vee x_1x_2x_3 = x_1x_2$. Про пару одиниць у правій частині діаграми можна сказати те ж саме:

$\bar{x}_1x_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2\bar{x}_3 = \bar{x}_1\bar{x}_3$.

Таблиця 7.11

		x_2		$\overline{x_2}$		
x_1		1100	1101	1001	1000	x_3
		1110	1111	1011	1010	x_3
$\overline{x_1}$		0110	0111	0011	0010	
		0100	0101	0001	0000	$\overline{x_3}$
		$\overline{x_4}$	x_4		$\overline{x_4}$	

Таблиця 7.12



Відзначимо, що отриманий елементарний добуток легко визначити відразу на діаграмі: це добуток змінних, які приймають однакові значення в обох клітинках.

Ще одне важливе зауваження: стовпці, розташовані по краях діаграми, теж вважаються сусідніми!

Для прикладу 7.6 це означає, що має місце ще одне склеювання, в результаті якого, задовольняючи зазначеному правилу, одержуємо елементарний добуток x_2x_3 .

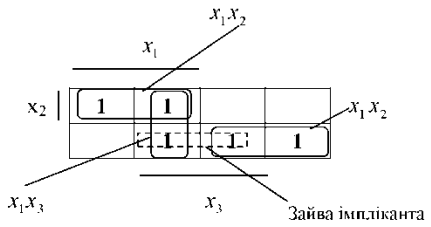
З розглянутих раніше методів нам відомо, що можливо подальше склеювання одержуваних елементарних добутків. На діаграмах Вейча вони теж розташовуються поруч.

Загальне правило склеювання на діаграмах Вейча можна сформулювати у такий спосіб: склеюванню підлягають прямокутні конфігурації, заповнені одиницями, які містять число клітинок, що є степенем числа 2. Новий отриманий елементарний добуток визначається як добуток змінних, що не змінюють свого значення на всіх наборах, які склеюються. Число m змінних, що залишились в елементарному добутку, визначається легко:

$$m = n - log_2 M ,$$

де n — число змінних функції; M — число наборів, що склеюються. Метод широко використовується на практиці, завдяки простоті і зручності. Після невеликого тренування досягається елементарна навичка визначення мінімальної ДНФ за діаграмою «з першого погляду».

Таблиця 7.13



Мінімізація булевої функції полягає у відшуканні мінімального покриття всіх одиниць діаграми Вейча блоками з одиниць (зазначеної конфігурації), розташованих у сусідніх клітинках діаграми. При цьому завжди вважається, що лівий край діаграми Вейча 4-х змінних примикає до її правого краю, а верхній край діаграми примикає до нижнього її краю. Після одержання мінімального покриття всіх одиниць діаграми Вейча, мінімальна ДНФ булевої функції записується як диз'юнкція елементарних кон’юнкцій, що відповідають виділеним блокам одиниць у діаграмі. **Конституента, яка не**

ввійшла в жодне покриття, є простою імплікантою і записується без змін в МДНФ. Розглянемо кілька прикладів.

Приклад 7.7. Булева функція f має наступну ДДНФ:

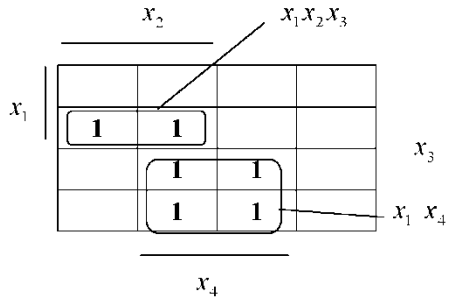
$$f = x_1x_2x_3 \vee x_1\bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2\bar{x}_3 \vee x_1x_2\bar{x}_3 .$$

Знайти мінімальну ДНФ за допомогою діаграми Вейча.

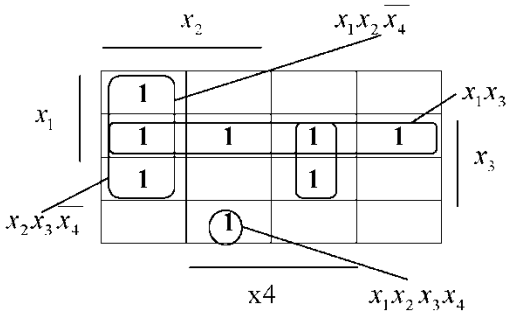
Діаграма Вейча, що відповідає функції f, представлена в табл. 7.13. Мінімальне покриття всіх одиниць діаграми можливо тільки блоками по дві одиниці. Кожному такому блоку відповідає своя кон'юнкція, як показано в табл. 7.13. Отже, мінімальна ДНФ функції має вигляд: $f = x_1x_2 \vee x_1x_3 \vee \bar{x}_1\bar{x}_2$

Очевидно, проста імпліканта $\cancel{x_2}\ x_3$ не потрібна для одержання мінімальної ДНФ.

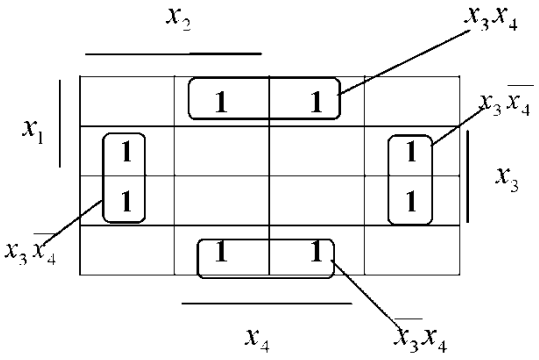
Таблиця 7.14



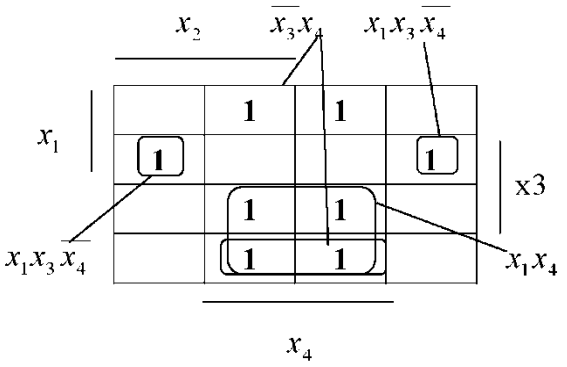
Таблиця 7.15



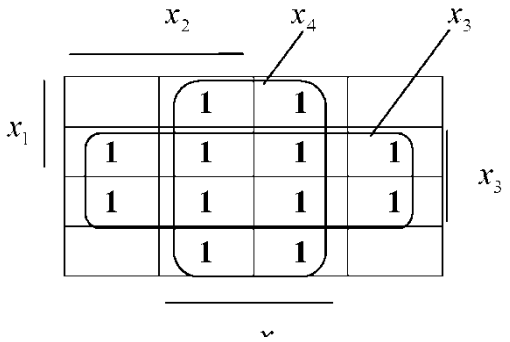
Таблиця 7.16



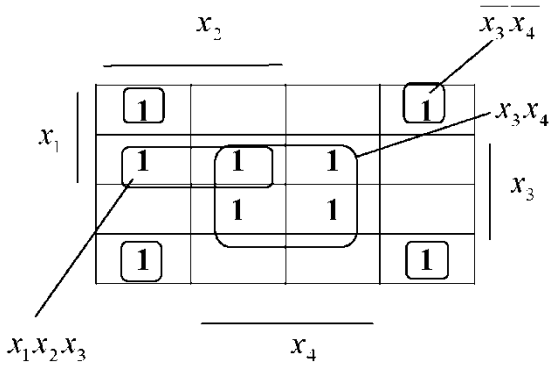
Таблиця 7.17



Таблиця 7.18



Таблиця 7.19



Приклад 7.8. Булеві функції $f_1, f_2, f_3, f_4, f_5, f_6$ задані діаграмами Вейча (табл. 7.14 - 7.19). Знайти їхні мінімальні ДНФ.

Мінімальні покриття одиниць показані в таблицях. Там же виписані кон'юнкції, щ відповідають виділеним блокам одиниць. Мінімальні ДНФ функцій мають наступний вигляд:

$$f_1 = x_1 x_2 x_3 \vee \bar{x}_1 x_4;$$

$$f_2 = x_1 x_2 \bar{x}_4 \vee x_2 x_3 \bar{x}_4 \vee x_1 x_3 \vee \bar{x}_2 x_3 x_4 \vee \bar{x}_1 x_2 \bar{x}_3 x_4;$$

$$f_3 = x_3 \bar{x}_4 \vee \bar{x}_3 x_4;$$

$$f_4 = \bar{x}_3 x_4 \vee \bar{x}_1 x_4 \vee x_1 x_3 \bar{x}_4;$$

$$f_5 = x_3 \vee x_4;$$

$$f_6 = x_3 x_4 \vee \bar{x}_3 \bar{x}_4 \vee x_1 x_2 x_3.$$

Метод карт Карно

У 1953 р. Моріс Карно опублікував статті про розроблену їм систему графічного представлення і спрощення логічних виразів.

Таблиця істинності та карта Карно для двох змінних має вигляд:

y	x	$f(x,y)$
0	**0**	**0**
0	**1**	**1**
1	**0**	**2**
1	**1**	**3**

x \ $\bar{y}$	$\bar{y}$	y
$\bar{x}$	**0**	**1**
x	**2**	**3**

x \ y	0	1
0	**0**	**1**
1	**2**	**3**

Чотири комірки **(0,1,2,3)** відповідають чотирьом можливим комбінаціям ***x*** і ***y*** у таблиці істинності з двома змінними.

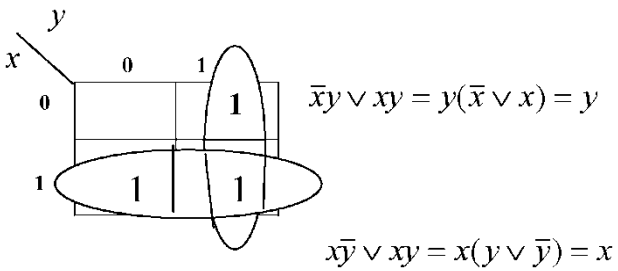
Приклад 7.9. За допомогою карт Карно потрібно представити функцію $f(x,y)=\bar{x}y\vee x\bar{y}\vee xy$.

Розмістимо одиниці у всіх комірках, яким відповідають добутки (кон'юнкції) у початковій ФАЛ $\bar{x}y, x\bar{y}, xy$:

	$\bar{y}$	y
		1
$\bar{x}$	**1**	**1**
x		

x \ y	0	1
0		**1**
1	**1**	**1**

Заповнена карта Карно готова для побудови МДНФ і ця процедура демонструється рисунком:



$\bar{x}y\vee xy=y(\bar{x}\vee x)=y$

$x\bar{y}\vee xy=x(y\vee\bar{y})=x$

Це рівносильно перетворенню:

$\bar{x}y\vee xy\vee x\bar{y}\vee xy=(x\bar{y}\vee xy)\vee(\bar{x}y\vee xy)=x\vee y$

Суть побудови - сусідні одиниці поєднуються в один контур по **2, 4**, **8, 16** одиниць. Побудова контурів продовжується доти, доки всі одиниці не виявляться всередині контурів.

Кожен контур являє собою новий член спрощеної ФАЛ. Тепер спростимо цей вираз, приймаючи до уваги два контури, у кожному з яких застосовуємо операцію склеювання.

$f=x\vee y$

Взагалі, процедура спрощення ФАЛ здійснюється шляхом послідовних шести кроків, а саме:

1. Одержати ДДНФ заданої функції.
2. Нанести одиниці на карту Карно.

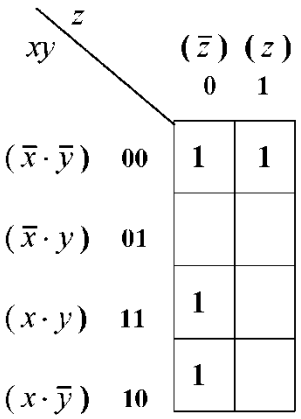
3. Об'єднати сусідні одиниці контурами, що охоплюють **2,4**, **8** чи **16** квадратів.
4. Зробити спрощення, крім членів, що доповнюють один одного всередині контуру (зробити операцію склеювання).
5. Об'єднати члени, що залишилися, (по одному в кожному контурі) операцією АБО.
6. Записати отриману спрощену ФАЛ у ДНФ.

Карти Карно з трьома змінними

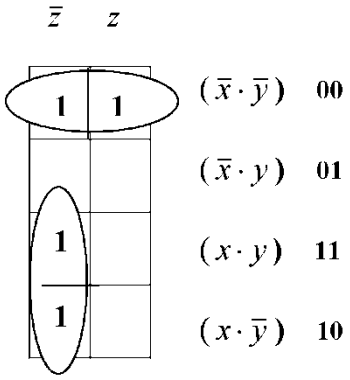
Приклад 7.10. Розглянемо ФАЛ, задану в ДДНФ.

а) $f(x,y,z) = x \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot \bar{y} \cdot \bar{z} \vee \bar{x} \cdot \bar{y} \cdot z \vee x \cdot y \cdot \bar{z}$

б) нанесемо на карту логічні одиниці.



в) поєднуємо кожну групу одиниць контурами і виконуємо операцію склеювання



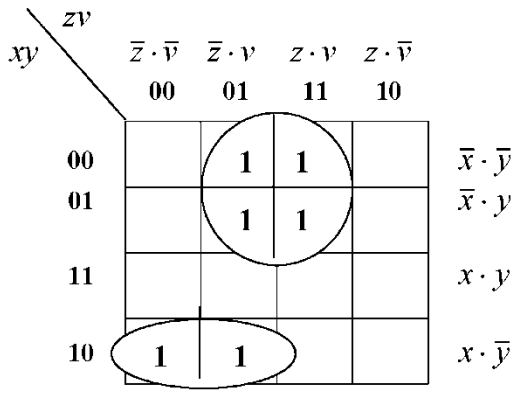
$f(x,y,z) = x \cdot \bar{z} \vee \bar{x} \cdot \bar{y}$

Карти Карно з чотирма змінними

Приклад 7.11. Розглянемо ФАЛ:

$f(x,y,z,v) = x \cdot \bar{y} \cdot \bar{z} \cdot \bar{v} \vee \bar{x} \cdot y \cdot \bar{z} \cdot v \vee \bar{x} \cdot \bar{y} \cdot \bar{z} \cdot v \vee \bar{x} \cdot \bar{y} \cdot z \cdot v \vee \bar{x} \cdot y \cdot z \cdot v \vee x \cdot \bar{y} \cdot \bar{z} \cdot v$

Таблиця істиності для чотирьох змінних включає **16** можливих комбінацій. Нанесемо на карту **6** одиниць, що відповідають конституентам функції.



Видно, що в нижньому контурі відбувається склеювання по *v*, а у верхньому контурі з чотирьох одиниць попарно опускаються *z* і $\bar{z}$, ***y*** і ***y*** , так що одержуємо член $\bar{x} \cdot v$.

Всі принципи по укладанню прямокутних покриттів у діаграмах Вейча поширюються і на карти Карно. Карти Карно зазвичай використовують для ручної мінімізації ФАЛ.У загальному випадку карта Карно функції ***N*** змінних складається з $\mathbf{2}^{N}$ клітинок за числом можливих вхідних комбінацій. Кожна клітинка карти має свою унікальну адресу у вигляді номера вхідного стану, як показано на рис. 3.1 для функцій трьох і чотирьох змінних.

а)

X_3 \ X_2X_1	00	01	11	10
0	0	1	3	2
1	4	5	7	6

б)

X_4X_3 \ X_2X_1	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Рис.7.1 – Карти Карно для логічних іункцій трьох (а) і чотирьох (б) змінних

Карти Карно функцій більшого числа аргументів мають розмірність $\mathbf{2}^{L}\mathbf{*2}^{H}$,де ***L*** і ***H*** - число рядків і стовпців у карті, і ***L+H=N***, або розбиваються на субкарти меншої розмірності, як зображено на рис. 7.2. В останньому випадку отримана за картою форма вимагає додаткової мінімізації за допомогою відомих законів і тотожностей.

X_5=0

X_4X_3 \ X_2X_1	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

X_5=1

X_4X_3 \ X_2X_1	00	01	11	10
00	16	17	19	18
01	20	21	23	22
11	28	29	31	30
10	24	25	27	26

Рис.7.2 – Карти Карно для логічної функції п'яти змінних

Найбільш важливе значення при мінімізації функцій за допомогою карт Карно мають способи об'єднання комірок у групи. На рис. 7.3 представлені основні способи формування й опису груп для функцій 4 змінних. (Вираз під картою відповідає тільки виділеній групі.)

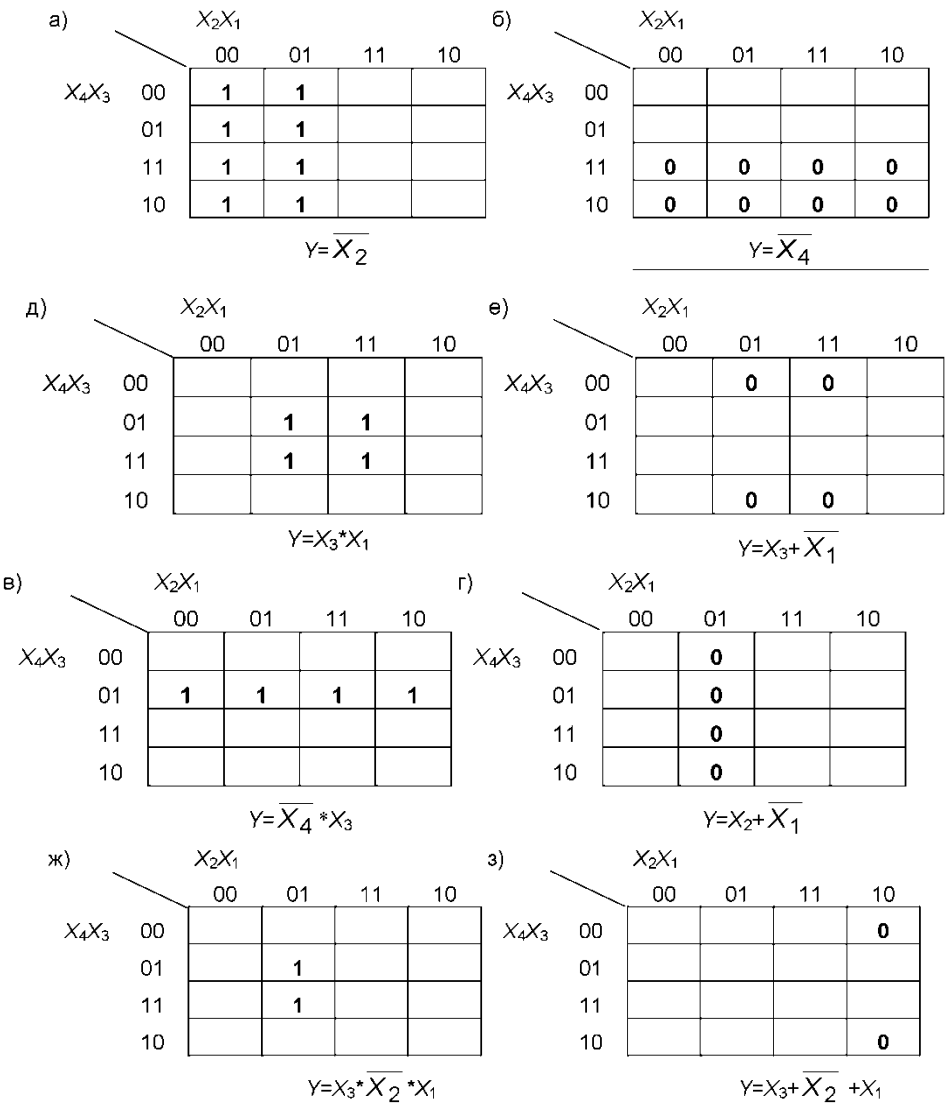


Рис.7.3 – Приклади формування та опису груп комірок карт Карно

Мінімізація кон'юнктивних нормальних форм

Мінімізація КНФ виконується аналогічно розглянутим методам мінімізації ДНФ булевих функцій, тому зупинимося лише на основних положеннях.

Нагадаємо, що **конституентою нуля** називається функція, що приймає значення **0** тільки на одному наборі. Вона виражається диз'юнкцією всіх змінних набору. Наприклад, набору **0110** відповідає конституента нуля $x_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4$.

При склеюванні конституент нуля утворюються імпліценти. **Імпліцентою *g* булевої функції *f*** називається функція, що приймає значення **0** на підмножині нульових наборів функції ***f***.

Простою імпліцентою функції *f* називається елементарна диз'юнкція, що є імпліцентою функції ***f***, причому жодна її власна частина не є імпліцентою функції ***f*** .

Задачею мінімізації КНФ є визначення мінімальної КНФ. Ця задача також вирішується в два етапи — пошук скороченої КНФ (кон'юнкція всіх простих імпліцент) і потім знаходження мінімальної КНФ. Другий етап мінімізації виконується за допомогою матриці Квайна точно так, як і при пошуку

мінімальної ДНФ, оскільки можливі тільки два варіанти: або дана проста імпліцента поглинає дану конституенту нуля, або ні - відповідно до співвідношення поглинання

$(A \vee x)A = A$.

Що стосується першого етапу – пошуку всіх простих імпліцент, то практично всі методи мінімізації ДНФ мають свої аналоги для КНФ. Розглянемо це докладніше.

Співвідношення склеювання за Квайном

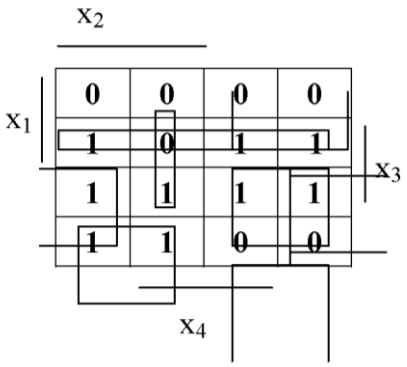
$(A \vee x)(A \vee \bar{x}) = (A \vee x)(A \vee \bar{x})A$.

Співвідношення склеювання за Блейком

$(A \vee x)(B \vee \bar{x}) = (A \vee x)(B \vee \bar{x})(A \vee B)$.

За діаграмою Вейча пошук мінімальної КНФ здійснюється так само просто, як у випадку ДНФ (для карт Карно приклади наведені на рис. 7.3). Відмінність складається лише в тому, що аналізуються набори, на яких функція дорівнює 0, і змінні виписуються з інверсіями у відповідні імпліценти.

Таблиця 7.20



Наприклад, для функції, заданій діаграмою (табл. 7.20), мінімальною КНФ є:

$(\bar{x}_1 \vee x_3)(x_2 \vee x_3)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_4)$. Для порівняння знайдемо мінімальну ДНФ: $x_1\bar{x}_2 \vee \bar{x}_2 x_3 \vee x_3\bar{x}_4$.У даному випадку ДНФ виявилася простіше. У загальному випадку про порівняльну складність мінімальних ДНФ і КНФ не можна говорити заздалегідь, але можна відзначити наступне: **кількості букв мінімальної ДНФ довільної функції *f* і мінімальної КНФ функції *f* однакові.**

Метод Петрика. Мінімізація частково визначених булевих функцій.

Метод Петрика використовується для знаходження всіх мінімальних покриттів конституент одиниці і дозволяє одержати всі тупикові ДНФ по імплікантній матриці. Суть методу полягає в наступному. За імплікантною матрицею будується так зване кон'юнктивне представлення імплікантної матриці. Для цього всі прості імпліканти позначаються різними буквами (звичайно прописними латинськими). Після цього, для кожного ***i***-го стовпця імплікантної матриці будується диз'юнкція всіх букв, що позначають рядки матриці, перетин яких з ***i***-м стовпцем відзначено хрестиком. Кон’юнктивне представлення імплікантної матриці утвориться як кон’юнкция побудованих диз'юнкцій для всіх стовпців матриці. До кон’юнктивного представлення

матриці можуть бути застосовані всі співвідношення булевої алгебри з метою його спрощення. Після розкриття дужок і виконання всіх можливих поглинань виходить диз'юнкція кон'юнкций, кожна з яких містить всі імпліканти тупикової ДНФ.

Приклад 7.12 Задана імплікантна матриця (табл.7.21)

Таблиця 7.21

Прості імпліканти	Конституенти одиниці					
	$\bar{x}_1\bar{x}_2\bar{x}_3x_4$	$\bar{x}_1\bar{x}_2x_3x_4$	$\bar{x}_1x_2\bar{x}_3x_4$	$\bar{x}_1x_2x_3x_4$	$x_1x_2x_3\bar{x}_4$	$x_1x_2x_3x_4$
$\bar{x}_1x_4$	x	x	x	x		
$x_2x_3x_4$				x		x
$x_1x_2x_3$					x	x

Знайти методом Петрика всі тупикові ДНФ булевої функції f, описуваної даною матрицею.

Наявні прості імпліканти позначимо буквами:

$\bar{x}_1x_4 = A$; $x_2x_3x_4 = B$; $x_1x_2x_3 = C$

Тоді кон'юнктивне представлення φ матриці має вигляд

$\varphi = A * A * A * (A \vee B) * C * (B \vee C)$.

Спростимо його

$\varphi = A(A \vee B)C(B \vee C) = (A \vee AB)(BC \vee C) = ABC \vee AC \vee ABC \vee ABC = AC \vee ABC = AC$.

Єдина тупикова ДНФ містить дві прості імпліканти $A = \bar{x}_1x_4$ і $C = x_1x_2x_3$ і має вигляд

$f = \bar{x}_1x_4 \vee x_1x_2x_3$.

Приклад 7.13. Задано імплікантну матрицю (табл. 7.22).

Таблиця 7.22

Прості імпліканти системи функцій	Конституенти одиниці функції φ					
	$x_1x_2x_3$	$x_1\bar{x}_2x_3$	$\bar{x}_1\bar{x}_2x_3$	$\bar{x}_1\bar{x}_2\bar{x}_3$	$x_1x_2\bar{x}_3$	
x_1x_3	x	x				
x_1x_2	x				x	
$\bar{x}_2x_3$		x	x			
$\bar{x}_1\bar{x}_2$			x	x		

Знайти методом Петрика всі тупикові ДНФ булевої функції / описуваної даною матрицею.

Наявні прості імпліканти позначимо буквами:

$x_1x_3 = A$; $x_1x_2 = B$; $\bar{x}_3x_3 = C$; $\bar{x}_1\bar{x}_2 = D$.

Тоді кон'юнктивне представлення **q>** матриці має вигляд

$\varphi = (A \vee B)(A \vee C)(C \vee D)DB$.

Після спрощення одержимо $\varphi = ABD \vee BCD$.Дві тупикові ДНФ є мінімальними:

$f = x_1x_3 \vee x_1x_2 \vee \bar{x}_1\bar{x}_2$;

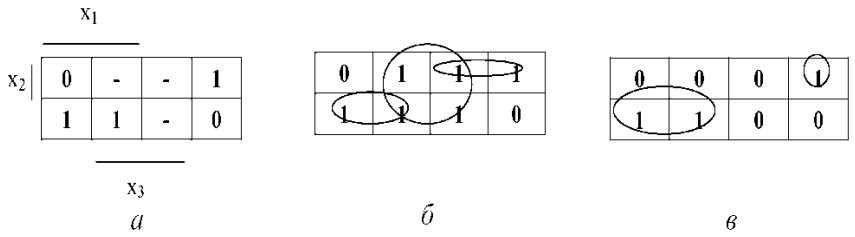
$f = x_1x_2 \vee \bar{x}_2x_3 \vee \bar{x}_1\bar{x}_2$.

У реальних задачах дуже часто трапляється, що значення булевої функції на деяких наборах не визначені і можуть довизначатися довільно. У цьому випадку

довизначення функції доцільно робити так, щоб її мінімальна форма мала найменше число букв із усіх можливих варіантів довизначення.

Приклад 7.14. Розглянемо простий приклад (табл. 7.23). Функція задана діаграмою Вейча, представленою таблицею 7.23, *а.* Довизначення функції на невизначених наборах одиницями (табл.7.23, *б)* чи нулями (табл.7.23, *в)* призводить до різних мінімальних ДНФ.

Таблиця 7.23



а *б* *в*

$$f_{МДНФ_а} = x_3 \vee x_1\bar{x}_2 \vee \bar{x}_1 x_2$$

$$f_{МДНФ_в} = x_1\bar{x}_2 \vee \bar{x}_1 x_2\bar{x}_3$$

Однак більш проста мінімальна ДНФ виходить, якщо зробити довизначення так, як це зроблено на діаграмі Вейча (табл. 7.24).

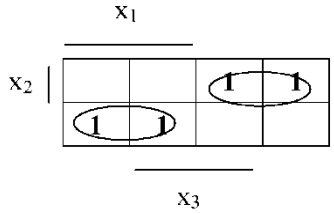
$$f_{МДНФ} = x_1\bar{x}_2 \vee \bar{x}_1 x_2$$

Алгоритм пошуку мінімальної ДНФ частково визначеної функції можна представити у такий спосіб:

1. Знайти будь-яким відомим способом скорочен у ДНФфункції, що виходить довизначенням одиницями вихідної функції ***f*** на всіх невизначених наборах.
2. Вибрати мінімальну ДНФ по імплікантній матриці, де в стовпцях виписані лише ті конституенти одиниці функції ***f***, що відповідають цілком визначеним одиничним наборам.

Аналогічний алгоритм (з довизначенням нульовими наборами) може бути запропонований для пошуку КНФ. При цьому довизначення таблиці істинності функції ***f*** може бути зроблене по різному (окремо) для КНФ і ДНФ.

Таблиця 7.24



Зазначимо, що для розв'язку розглянутої задачі практично вистачить тих навичок, що були отримані при мінімізації цілком визначених булевих функцій безпосередньо по діаграмі Вейча. У випадках, коли мінімальних форм декілька, приводиться одна з них.

Контрольні питання:

1. Якими способами можна задати ФАЛ? Поясніть кожній із них.
2. Що таке ДНФ? КНФ?
3. Яка функція алгебри логіки вважається повністю визначеною?
4. Які диз'юнкції та кон'юнкції називаються елементарними?
5. Які диз'юнкції та кон'юнкції називаються неелементарними?
6. Які мінімальні нормальні форми булевих функцій ви можете назвати?
7. Що таке ранг правильної конфігурації?
8. Як обчислюється кількість змінних, які описують ПК?
9. Як описуються правильні конфігурації?
10. Які дії треба виконати, щоб отримати мінімальні нормальні форми булевих функцій: диз'юнктивну, кон'юнктивну, Шефферівську і Пірсівську?
11. Назвіть мету мінімізації логічних функцій.
12. Які способи мінімізації Ви знаєте?
13. Чому карти Вейча знайшли широке практичне застосування при мінімізації ФАЛ ?
14. Що собою представляють карти Вейча?
15. З яких основних кроків складається процес мінімізації логічної функції за методом Квайна?
16. Як ви розумієте нульові куби в методі Квайна-Мак-Класкі?
17. Назвіть основні закони булевої алгебри, які використовуються в методах Квайна та Квайна-Мак-Класкі?
18. Які є правила створення контурів при використанні діаграми Вейча?
19. Що називають картою Карно?
20. Які особливості використання карт Карно для отримання мінімальних нормальних форм булевих функцій?
21. Як отримуються десяткові порядкові номери клітин карти Карно?
22. Які клітини карти Карно є сусідніми?
23. Які є правила створення контурів при використанні карти Карно?
24. Як використати карту Карно для переходу до нормальної форми логічної функції?

ТЕМА 8. СИСТЕМИ (СЕРІЇ) ЛОГІЧНИХ ЕЛЕМЕНТІВ

План:

1. Основні поняття. Класифікація логічних елементів.
2. Базові логічні елементи різних серій, їх принципи побудови та технічні характеристики.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Основні поняття. Класифікація логічних елементів.

Логічний елемент – це електронний пристрій, який реалізує певну логічну (перемикальну) функцію.

Логічні елементи являють собою електронні пристрої, у яких оброблювана інформація закодована у вигляді двійкових чисел, відображуваних напругою (сигналом) високого і низького рівня. Термін «логічні» прийшов в електроніку з алгебри логіки, що оперує зі змінними величинами і їхніми функціями, що можуть приймати тільки два значення: «істинно» чи «хибно». Для позначення чи істинності хибності висловлень використовують відповідно символи 1 чи 0. Кожна логічна перемінна може приймати тільки одне значення: 1 чи 0. Ці двійкові змінні і функції від них називаються логічними змінними і логічними функціями. Пристрої, що реалізують логічні функції, називаються логічними чи цифровими пристроями.

Класифікація логічних елементів:

За видом вхідного сигналу:

- потенціальні елементи;
- імпульсні логічні елементи;
- імпульсно потенціальні.

За елементною базою:

- діодна логіка;
- діодно – транзисторна логіка;
- резисторно – транзисторна логіка;
- транзисторно – транзисторна логіка на діодах Шоткі;
- транзисторна логіка;
- транзисторно – транзисторна логіка;
- логічні елементи на польових транзисторах;
- оптоелектронні логічні елементи.

За видом зв’язку між елементами:

- безпосередній зв’язок;
- резистивний зв’язок;
- діодний зв’язок;
- ємнісний зв’язок.

За способом з’єднання транзисторів:

- колекторним зв’язком;
- емітерним зв’язком.

За видом функції перетворення:

- одноступеневі логічні елементи;
- дво- або більше ступеневі логічні елементи.

За видом фізичних ефектів що застосовуються в фізичних елементах:

- електронні;
- оптоелектронні;
- магнітокеровані;
- пневмоавтоматичні та інші.

Сукупність логічних елементів і зв’язків між ними, призначену для перетворення двійкових змінних, називають логічною схемою. Логічні схеми поділяють на послідовнісні і комбінаційні.

Комбінаційною називають схему, ***m*** вихідних сигналів якої в кожний момент часу повністю визначаються сукупністю ***n*** її вхідних сигналів в цей самий момент часу. Тобто вихідні сигнали комбінаційної схеми в даний момент часу не залежать від вхідних сигналів, які діяли в попередні моменти часу (схема не має пам’яті). Кажуть, що така схема має один стан.

Початковими даними (технічним завданням) для проектування комбінаційного вузла є його функціональний опис та вимоги до основних електричних параметрів. Функціональний опис комбінаційного вузла зазвичай задається у вигляді таблиці істинності або алгебраїчного виразу.

Процес проектування розбивається на декілька послідовних етапів:

1) вибір елементної бази та способу реалізації;
2) мінімізація заданої логічної функції;
3) перетворення мінімізованої логічної функції та синтез логічної схеми;
4) синтез електричної схеми;
5) аналіз та оптимізація електричної схеми.

Вибір елементної бази визначається вимогами, пропонованими до електричних параметрів комбінаційного вузла: швидкодією, потужністю, що споживається, завадостійкістю та ін.

Для реалізації комбінаційних схем використовують логічні елементи, що випускаються у вигляді інтегральних схем. До цього класу відносяться інтегральні схеми дешифраторів, шифраторів, мультиплексорів, демультиплексорів та суматорів.

До типових вузлів належать логічні схеми, які найчастіше використовуються в цифрових ЕОМ. Серед них існують як комбінаційні так і послідовнісні схеми з пам’яттю.

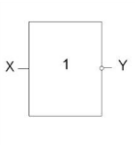
Графічне зображення комбінаційної схеми, при якому показані зв'язки між різними елементами, а самі елементи представлені умовними позначеннями, називається функціональною схемою.

Поведінка комбінаційної схеми описується системою логічних функцій.

2. Базові логічні елементи різних серій, їх принципи побудови та технічні характеристики.

На рис. 8.1 – 8.10 представлені логічні елементи, що реалізують логічні функції. Там же представлені так називані таблиці станів чи таблиці істинності, що описують відповідні логічні функції в двійковому коді у виді станів вхідних і вихідних перемінних. Таблиця істинності є також табличним способом завдання ФАЛ.

На рис.8.1 представлений елемент «НЕ», що реалізує функцію логічного заперечення $y = \overline{x}$.



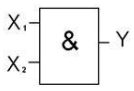
X	Y
0	1
1	0

Рис. 8.1 – Елемент «НЕ»

Елемент «АБО» (рис.8..2) і елемент «І» (рис.8.3) реалізують функції логічного додавання і логічного множення відповідно.

X_1	X_2	Y
0	0	0
1	0	1
0	1	1
1	1	1

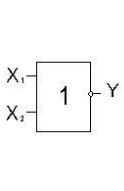
Рис. 8.2 – Елемент «АБО»



X_1	X_2	Y
0	0	0
0	1	0
1	0	0
1	1	1

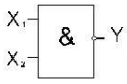
Рис. 8.3 – Елемент «І»

Функції Пірса і функції Шеффера реалізуються за допомогою елементів «АБО-НЕ» і «І-НЕ», представлених на рис.8.4 і рис. 8.5 відповідно.



X_1	X_2	Y
0	0	1
1	0	0
0	1	0
1	1	0

Рис. 8.4 – Елемент «АБО-НЕ»



X_1	X_2	Y
0	0	1
0	1	1
1	0	1
1	1	0

Рис. 8.5 – Елемент «І-НЕ»

Елемент Пірса можна представити у виді послідовного з'єднання елемента «АБО» і елемента «НЕ» (рис.8.6), а елемент Шеффера – у виді послідовного з'єднання елемента «І» і елемента «НІ» (рис.8.7).

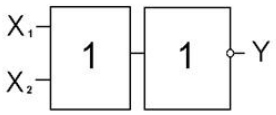


Рис. 8.6 – Елемент Пірса

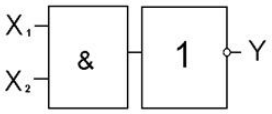
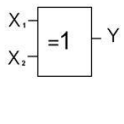


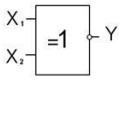
Рис. 8.7 – Елемент Шеффера

На рисунку 8.8 і 8.9 представлені елементи «Виключаюче АБО» і «Виключаюче АБО-НЕ», що реалізують функції нерівнозначності і нерівнозначності з запереченням відповідно.



X_1	X_2	Y
0	0	0
1	0	1
0	1	1
1	1	0

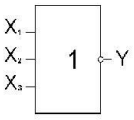
Рис. 8.8 – Елемент «Виключаюче АБО»



X_1	X_2	Y
0	0	1
1	0	0
0	1	0
1	1	1

Рис. 8.9 – Елемент «Виключаюче АБО-НЕ»

Логічні елементи, що реалізують операції кон'юнкції, диз'юнкції, функції Пірса і Шеффера, можуть бути, у загальному випадку, n - входові. Так, наприклад, логічний елемент із трьома входами, що реалізує функцію Пірса, має вид, представлений на рис. 8.10.



X_1	X_2	X_3	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	0

Рис. 8.10 – Логічний елемент, що реалізує функцію Пірса

У таблиці істинності (рис.8.10) є вісім значень вихідних змінних Y. Ця кількість визначається числом можливих комбінацій вхідних змінних N, що, у загальному випадку, дорівнює: $N = 2^n$, де n - число вхідних змінних.

Логічні елементи по режиму роботи підрозділяються на статичні і динамічні. Статичні ЛЕ можуть працювати як у статичному, так і динамічному (імпульсному) режимах. Статичні елементи найбільше широко використовуються в сучасних мікросхемах. Динамічні ЛЕ можуть працювати тільки в імпульсному режимі.

Логічні елементи класифікують також за типом транзисторів, які застосовуються. Для кожного з перерахованих типів ЛЕ існує число схемотехнічних і конструктивно – технологічних різновидів.

Логічне проектування

Зазвичай, логічне проектування виконується в наступній послідовності:

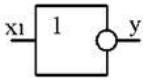
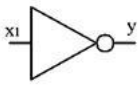
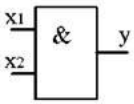
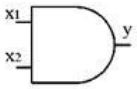
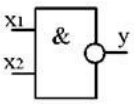
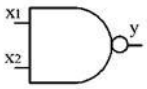
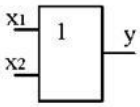
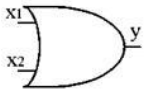
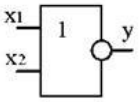
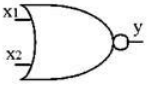
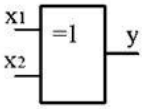
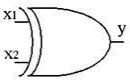
1) складання таблиці істинності синтезованого вузла відповідно до його означення, призначення і (словесного) опису принципу роботи;

2) складання математичної формули для логічної функції, що описує роботу синтезуючого вузла, відповідно до наявної таблиці істинності;

3) аналіз отриманої функції з метою побудови різних варіантів її математичного виразу (на підставі законів булевої алгебри) і знаходження найкращого з них відповідно до того чи іншого критерію;

4) складання функціональної (логічної) схеми вузла із заздалегідь заданим набором логічних елементів.

Графічні позначення логічних елементів

Назва елемента	Вітчизняні позначення	Міжнародні позначення	Таблиці істинності
Інвертор			x | y 0 | 1 1 | 0
І			x_1 | x_2 | y 0 0 0 0 1 0 1 0 0 1 1 1
І-НЕ			x_1 | x_2 | y 0 0 1 0 1 1 1 0 1 1 1 0
АБО			x_1 | x_2 | y 0 0 0 0 1 1 1 0 1 1 1 1
АБО-НЕ			x_1 | x_2 | y 0 0 1 0 1 0 1 0 0 1 1 0
Виключаюче АБО			x_1 | x_2 | y 0 0 0 0 1 1 1 0 1 1 1 0

Серією (системою, комплексом) логічних елементів ЕОМ називається призначений для побудови цифрових пристроїв функціонально повний набір логічних елементів, поєднаний спільними електричними, конструктивними і технологічними параметрами, що використовує однаковий спосіб представлення інформації, однаковий тип міжелементних зв'язків. Система елементів найчастіше надлишкова за своїм функціональним складом, що дозволяє будувати схеми більш економічні за кількістю використаних елементів.

До складу серії входять елементи для виконання логічних операцій, елементи пам'яті, елементи, що реалізують функції вузлів ЕОМ, а також спеціальні елементи для підсилення, відновлення і формування сигналів стандартної форми.

У більшості сучасних серій елементів маються мікросхеми малого ступеня інтеграції (ІС до 100 елементів на кристал), середнього ступеня (СІС – до 1000 елементів на кристал), великого ступеня інтеграції (ВІС – до 10000 елементів на кристал) і надвеликого ступеня інтеграції (НВІС – більш 10000 елементів на кристал).

Основними ***параметрами*** серії логічних елементів є:

– напруги живлення і сигнали для представлення логічного 0 і логічної 1;

– коефіцієнти об'єднання по входу;
– навантажувальна здатність (коефіцієнт розгалуження по виходу);
– завадостійкість;
– розсіювана потужність;
– швидкодія.

Серія елементів характеризується кількістю використовуваних напруг живлення та їхніми номінальними значеннями. Звичайно логічному 0 відповідає низький рівень напруги, а логічній 1 – високий.

Для найбільш часто використовуваних серій напруга живлення складає +5В, рівень логічної одиниці – (2,4 … 5)В, рівень логічного 0 – (0 … 0,4)В.

Коефіцієнт об'єднання по входу (Коб) визначає максимально можливе число входів логічного елемента, іншими словами, функцію скількох змінних може реалізувати цей елемент. Звичайно Коб приймає значення від 2 до 4, рідше Коб=8. Збільшення числа входів зв'язано з ускладненням схеми елементів і приводить до погіршення інших параметрів – завадостійкості, швидкодії і т.д.

Коефіцієнт розгалуження по виходу (Кроз) показує, на скільки логічних входів може бути одночасно навантажений вихід даного логічного елемента. Звичайно Кроз для найбільш часто використовуваних серій дорівнює 10. Іноді замість Кроз задається гранично припустиме значення вихідного струму логічного елемента в стані 0 або 1.

Завадостійкість (перешкодостійкість) – це здатність елемента правильно функціонувати при наявності перешкод. Вона визначається максимально припустимою напругою перешкоди, при якій не відбувається збою в його роботі. Звичайно це напруга порядку 0,6-0,9 В.

Швидкодія логічних елементів є одним з найважливіших параметрів і характеризується часом затримки поширення сигналу. Цей параметр істотно залежить від технології виготовлення мікросхем і лежить у діапазоні від одиниць до сотень наносекунд.

При синтезі КС на реальних логічних елементах необхідно обов'язково враховувати обмеження на Коб і Кр.

Всі характеристики логічних елементів поділяються на:
– статичні
– динамічні
– функціональні
– загальні

Статичні характеристики логічних елементів розглянемо на прикладі інвертора рис. 8.11.

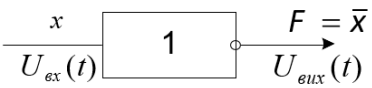


Рис. 8.11 – Логічний елемент типу «Інвертор»

При застосуванні позитивної логіки рівень логічної «1»

характеризується високою напругою, наприклад U_1(4,5...5,5)В, а рівень логічного «0» характеризується низькою напругою, наприклад U_0 (0,1...0,5)В (рис. 8.12)

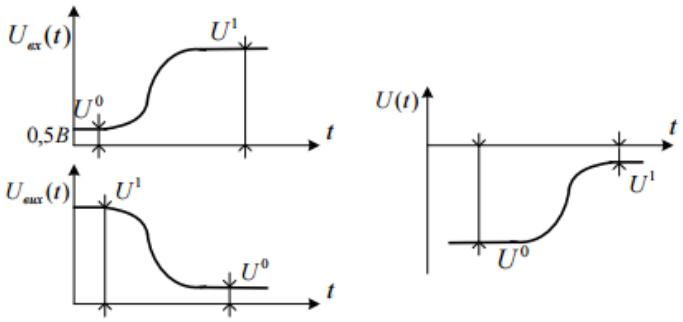


Рис. 8.12 – Логічний елемент в статичному режимі (позитивна логіка)

Якщо живлення ЛЕ негативне, то рівень логічної «1» повинен бути більшим від рівня напруги логічного «0» (рис. 8.12).

При застосуванні негативної логіки все навпаки (рис. 8.13).

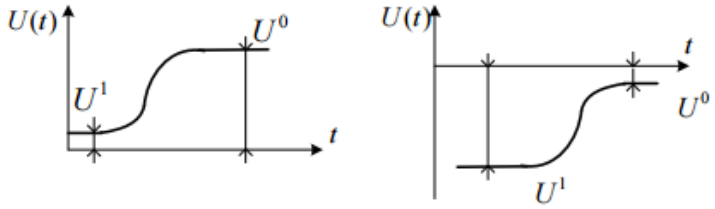


Рис. 8.13 – Логічний елемент в статичному режимі (негативна логіка)

Так само для цих логік вводиться поняття струму логічної «1» і логічного «0», тобто $I^0_{вх}$, $I^1_{вх}$, $I^0_{вих}$, $I^1_{вих}$.

Потужність споживання логічних елементів і багатьох мікросхем характеризується середньою потужністю споживання при станах коли ЛЕ знаходиться в «одиничному» стані і «нульовому» стані.

$$P_{ср} = \frac{P_{(1)} + P_{(1)}}{2}$$

Динамічні характеристики ЛЕ характеризують часом затримки вихідного сигналу відносно вхідного (рис. 8.14):

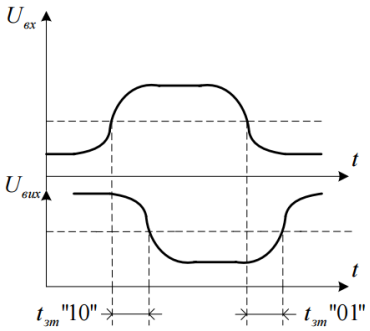


Рис. 8.14 – Логічний елемент в динамічному режимі

$t_{зт}^{10}$ - час затримки при переході елемента із одничного стану в нульовий;

$t_{зт}^{01}$ - час затримки при переході елемента із нульового стану в одиничний;

$t_{зтср}$ - середній час, як $\frac{(t_{зт}^{10}+t_{зт}^{01})}{2}$;

f_{max} - максимальна частота перемикання логічного елемента без збоїв.

На практиці використовують, як правило, не всі названі параметри.

Функціональні характеристики логічних елементів характеризуються такими параметрами:

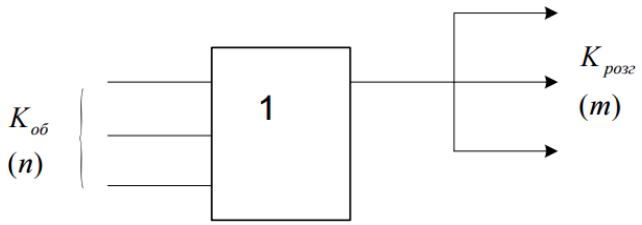


Рис. 8.15 – Функціональна характеристика логічного елемента

n – коефіцієнт об'єднання по входу, це кількість однотипних ходів логічного елемента;

m – коефіцієнт розгалуження, який показує кількість логічних елементів тієї ж самої серії яку можна під'єднати до виходу даного логічного елемента

Завадостійкість характеризується максимальною амплітудою завади, яка не викликає помилкового спрацювання схеми.

Загальні характеристики логічних елементів характеризують умови їх експлуатації: температурний діапазон, тиск, прискорення, а також габарити, вагу, коштовність і т.п.

Контрольні питання:

1. Поясніть, що таке логічний елемент.
2. Охарактеризуйте класифікацію логічних елементів.
3. Які основні параметри та характеристики ЛЕ ви знаєте?
4. Що таке швидкодія?
5. Дайте визначення «серії»?
6. Чим характеризується завадостійкість?
7. Що характеризує параметр «складність»?
8. Які серії логічних елементів ви знаєте?
9. З якими основними характеристиками логічних елементів ви знайомі?
10. Які логічні функції виконують елементи Пірса та Шеффера?
11. Чим визначена кількість можливих комбінацій вхідних змінних для будь-якого логічного елемента?

ТЕМА 9. СХЕМОТЕХНІКА ЦИФРОВИХ АВТОМАТІВ

План:

1. Елементарні цифрові автомати: тригери, регістри, лічильники.
2. Базові вузли ЦА на комбінаційних схемах: суматори, дешифратори, шифратори, селектори імпульсів, генератори синхросигналів.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Елементарні цифрові автомати: тригери, регістри, лічильники.

Елементарним цифровим автоматом з пам'яттю є тригерний пристрій (тригер). Тригер – найпростіша цифрова схема послідовнісного типу. Тригер – це клас електронних пристроїв, які мають здатність довго знаходитись в одному з двох стійких станів та чергувати їх під впливом зовнішніх сигналів. Кожен стан тригеру легко розпізнається за значенням вихідної напруги. За характером дії тригери належать до імпульсних пристроїв – їх активні елементи працюють в активному режимі, а зміна станів триває дуже короткий час.

Особливістю тригера як функціонального пристрою є властивість запам'ятовування двійкової інформації. Під пам'яттю тригера розуміють здатність залишатись в одному з двох станів після припинення дії сигналу перемикання. Таким чином, тригер зберігає (пам'ятає) один розряд числа в двійковому коді.

При виготовленні тригерів застосовуються переважно напівпровідникові прилади. В наш час логічні схеми, в тому числі з використанням тригерів, створюють в інтегрованих середовищах розроблення під різні програмовані логічні інтегральні схеми. Тригери використовуються, в основному, в обчислювальній техніці для організації компонентів обчислювальних систем: регістрів, лічильників, процесорів, ОЗП.

В якості елементів пам'яті структурного автомата звичайно використовуються тригери.

Тригер – це пристрій, що має два стійких стани, у які він переходить під дією визначених вхідних сигналів.

Звичайно в тригерах виділяють два види вхідних сигналів (і відповідно входів): інформаційні і синхросигнали.

Інформаційні сигнали визначають новий стан тригера і присутні в будь-яких тригерах. За типом інформаційних сигналів здійснюється класифікація тригерів: D T, RS, J, і т.д.

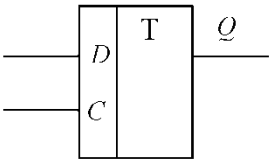
Синхросигнал не є обов'язковим і вводиться в тригерах з метою фіксації моменту переходу тригера в новий стан, що задається інформаційними входами. Звичайно, при синтезі ЦА використовуються тригери із синхровходом, тому надалі будемо розглядати тільки такі тригери.

На синхровход тригера надходять імпульси тактового генератора, що синхронізує роботу ЦА. Період проходження імпульсів відповідає одному тактові автоматного часу ЦА.

Розглянемо основні типи тригерів, використовувані для синтезу ЦА: D T, RS, J.

D-тригер - елемент затримки - має один інформаційний вхід *D* і один вихід *Q* і здійснює затримку на один такт сигналу, що надійшов на його вхід.

Умовне позначення і таблиця переходів *D-тртера* представлена на рис. 9.1.



***D*-тригер**

D	Q^t	Q^{t+1}
0	0	0
0	1	0
1	0	1
1	1	1

Таблиця переходів *D*-тригера

Рис.9.1 – Умовне позначення і таблиця переходів *D-тригера*

3 приведеної таблиці переходів для даного тригера Qt+1 = AQ\Dx) можна одержати таблицю функцій його входів Dx = y(Q\ Qt+l).

Q^t	Q^{t+1}	D^t
0	0	0
0	1	1
1	0	0
1	1	1

Таблиця функцій входів *D*-тригера

Як видно з таблиці, стан, у який переходить тригер (середній стовпець), збігається з сигналом *D(t)* (правий стовпець), що надійшов на його вхід. У зв'язку з цим таблиця функцій збудження пам'яті синтезованого автомата з використанням *D-тригерів* буде цілком збігатися з кодованою таблицею переходів цього автомата.

***D*-тригер К155ТМ2**

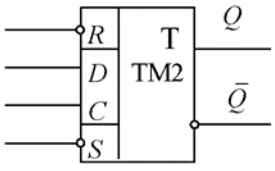
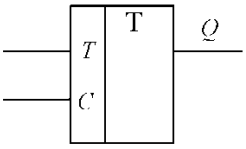


Рис.9.2 – Умовне позначення D-тригера К155ТМ2

Промисловість випускає *D-тригери* в інтегральному виконанні. Наприклад, К155ТМ2 (рис. 9.2). Таких тригерів два в одному корпусі. Вхід *C* - вхід

синхронізації, *Q, Q* -виходи, *Q* - прямій, *Q* - інверсний. *R, S* - входи установки в 0 і 1 відповідно. При подачі на вхід *R* і *S* логічного нуля тригер встановлюється у відповідні стани незалежно від сигналу на входах *D* і *C.*

Т-тригер - тригер з лічильним входом - має один інформаційний вхід *T і* один вихід *Q* і здійснює додавання за модулем два значення сигналу *T* і стану *Q* у заданий момент часу. Умовне позначення і таблиця переходів *Т-тригера* представлені на рис. 9.3.



T-триггер

T	Q^t	Q^{t+1}
0	0	0
0	1	1
1	0	1
1	1	0

Рис.9.3 – Умовне позначення і таблиця переходів *Т-тригера*

Таблиця функцій входів тригера $T^t = f(Q^t, Q^{t+1})$ представлена нижче.

Q^t	Q^{t+1}	T^t
0	0	0
0	1	1
1	0	1
1	1	0

Таблиця функцій входів *Т*-тригера

На основі цієї таблиці можна одержувати функцію збудження елементів пам'яті при синтезі автомата на базі *Т-тригера*. Наприклад, якщо автомат перейшов зі стану a_i = 010 у стан a_j = 110, то для забезпечення цього переходу функції збудження повинні бути:

для першого тригера при переході з **0** в **1** $T_1 = \mathbf{1}$,
для другого тригера при переході з **1** в **1** $T_2 = \mathbf{0}$,
для третього тригера при переході з **0** в **0** $T_3 = \mathbf{0}$
і т.д.

У безпосередньому вигляді промисловість не випускає *Т-тригери*. **RS-тригер** – тригер з роздільними входами.

Даний тригер має два вхідних канали *R* і *S* та один вихідний *Q*. Вхід *S* (set) називається входом установки в одиницю, вхід *R* (reset) – входом установки в нуль. Умовна позначка і таблиця переходів *RS-тригера* представлена на рис. 9.4.

В таблиці переходів при подачі комбінації *S* = *R* = 1 стан переходу Q^{t+1} не визначений і ця комбінація сигналів є забороненою для *RS-тригера*.

Таблицю переходів можна більш компактно зобразити у вигляді (див. табл. б)) Аналізуючи табл. б), в), відзначаємо що, наприклад, перехід тригера з 0 у 0 вимагає подачі комбінації *R*=0, *S*=0 або *R*=1,*S*=0, тобто можна сказати що цей перехід буде при *R*=X (байдужний стан), *S*=0.

Аналогічно міркуючи стосовно інших переходів, одержимо наступну таблицю функцій входів.

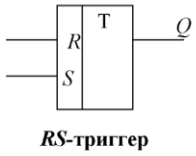
На основі таблиці можна одержати функцію збудження пам'яті автомата при синтезі на базі *RS-тригерів*. Наприклад, якщо автомат переходить зі стану a_i= 010 у стан a_j=110, то для забезпечення такого переходу функції збудження повинні бути:

для першого тригера при переході з **0** у **1** $R_1 = 0, S_1 = 1;$

для другого тригера при переході з **1** у **1** $R_2 = 0, S_2 = X;$

для третього тригера при переході з **0** у **0** $R_3 = X, S_3 = 0.$

Аналогічно для будь-якого іншого переходу автомата.



***RS*-триггер**

R	S	Q^t	Q^{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	–
1	1	1	–

а)

R	S	Q^{t+1}
0	0	0
0	1	1
1	0	0
1	1	–

б)

Таблиця переходів *RS*-тригера

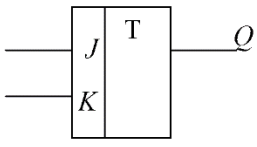
Q^t	Q^{t+1}	R^t	S
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

в)

Таблиця функцій входів *RS*-тригера

Рис.9.4 – Умовне позначення, таблиця переходів та таблиця функцій входів *RS-тригера*

JK- тригер - має два інформаційних входи J, K *та* один вихід *Q*. Вхід *J* вхід установки в 1, вхід *K*- вхід установки в 0, тобто ці входи аналогічні відповідним входам RS-тригера: *J* -відповідає *S, K* - відповідає *R*. Однак, на відміну від RS-тригера, вхідна комбінація *J* = 1, *K*= 1 не є забороненою. Умовне позначення і таблиця переходів JK-тригера представлені на рис. 9.5.



***JK*-триггер**

J	K	Q^t	Q^{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

а)

J	K	Q^{t+1}
0	0	Q^t
0	1	0
1	0	1
1	1	$\overline{Q}^t$

б)

Таблиця переходів *JK*-тригера

Рис. 9.5 – Умовне позначення і таблиця переходів *JK-тригера*

Як випливає з таблиці переходів, для комбінацій вхідних сигналів ***J*** = 00^10 тригер поводиться як RS-тригер, а при комбінації ***J*** = 11 - як Т-тригер.

Аналізуючи таблицю переходів (табл. а), відзначаємо, що перехід тригера, наприклад, з **0** у **1** вимагає подачі вхідних сигналів *J=1, K=0* або *J=1, **K***=\, тобто *J=1, K=X* (байдуже яке значення). Аналогічно міркуючи стосовно інших переходів, одержимо наступну таблицю функцій входів J-тригера.

Q^t	Q^{t+1}	J	K
0	0	X	0
0	1	1	X
1	0	X	1
1	1	0	X

Таблиця функцій виходів *JK*-тригера

На підставі останньої таблиці можна одержати функцію збудження елементів пам'яті при синтезі автомата на JK-тригерах. Наприклад, при переході автомата зі стану a_i=010 у стан a_j=110, функції збудження повинні бути:

для першого тригера при переході з **0** у **1** $J_1 = 1, \quad K_1 = X$;

для другого тригера при переході з **1** у **1** $J_2 = X, K_2 = 0$;

для третього тригера при переході з **0** у **0** $J_3 = 0, \quad K_3 = X$.

Припустимо, що конкретна схема описується наступною таблицею станів (таблиця 9.1).

Таблиця 9.1

$x_{1(t+1)}$	$x_{2(t+1)}$	y_t	y_{t+1}	$\bar{y}_{t+1}$
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

Вихідна функція для цієї схеми описується рівнянням:

$$y_{t+1} = \bar{x}_{1(t+1)}\bar{x}_{2(t+1)}y_t \vee x_{1(t+1)}\bar{x}_{2(t+1)}\bar{y}_t \vee x_{1(t+1)}\bar{x}_{2(t+1)}y_t \vee x_{1(t+1)}x_{2(t+1)}\bar{y}_t ,$$

яке після перетворення та мінімізації має вигляд:

$$y_{t+1} = \bar{x}_{2(t+1)}y_t \vee x_{1(t+1)}\bar{y}_t \qquad (9.1)$$

Тригери з двома входами — схеми, вихідна функція для яких має вигляд (9.1). Вони мають два стійких внутрішніх стани. Перехід тригера з одного стану в інший здійснюється при виконанні наступних умов:

якщо $y_t = 0$ при $x_{1(t+1)} = 1$, то $x_{2(t+1)}$ не впливає;

якщо $y_t = 1$ при $x_{2(t+1)} = 1$, то $x_{1(t+1)}$ не впливає (на стан тригера).

Аналізуючи табл.9.1, можна зробити висновок, що при $x_{1(t+1)} = x_{2(t+1)} = 1$ перехід тригера з одного стану в інший залежить від значення y_t. Якщо $y_t = 0$, то $y_{t+1} = 1$, якщо $y_t = 1$, то $y_{t+1} = 0$. Звідси випливає, що не обов'язково подавати на входи

тригера дві незалежних змінних, а достатньо подавати один сигнал на два входи одночасно.

Вихідна функція для цієї схеми описується рівнянням:

$$y_{t+1} = \bar{x}_{t+1} y_t \vee x_{t-1} \bar{y}_t \qquad (9.2)$$

Тригери з лічильним входом — схеми, вихідна функція яких має вигляд (9.2). Таблиця станів цієї схеми має наступний вигляд (табл. 9.2).

Таблиця 9.2

x_{t+1}	y_t	y_{t+1}	$\bar{y}_{t+1}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

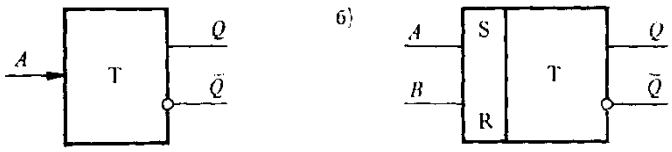


Рис. 9.6 - а, б відповідно представлені умовні позначення тригера з лічильним входом та тригера з двома входами

На рис. 9.6, *а, б* відповідно представлені умовні позначення тригера з лічильним входом та тригера з двома входами.

Регістри

Регістр – це послідовністний цифровий пристрій (цифровий автомат), який призначено для тимчасового зберігання інформації, яка подана у вигляді двійкових слів та виконання деяких операцій над ними. Кожен розряд двійкового слова зберігається в окремій комірці, яка являє собою тригер, тип котрого визначається вибором схеми керування процесами запису/зчитування інформації в регістрі. Двійкове слово, що зберігається в регістрі може відповідати будь-якій інформації. Це може бути число, код команди, логічна функція тощо.

Як усі цифрові автомати регістри можуть бути синхронним або асинхронними.

Регістри використовуються для виконання мікрооперацій запису, зберігання та читання інформації, а також для:

- оперативного зберігання інформації;
- затримки інформації на певний час;
- перетворення послідовного коду представлення інформації в паралельний та навпаки;
- зсуву коду, що зберігається на один чи декілька розрядів вліво або вправо.

Регістри також можуть використовуватися для побудування лічильників імпульсів та генераторів послідовностей імпульсів.

Головною класифікаційною ознакою регістрів є спосіб запису інформації до

нього. Відповідно до цієї ознаки регістри можливо розподілити на три групи:

– паралельні (регістри зберігання);

– послідовні (зсувні регістри);

– послідовно-паралельні (універсальні регістри).

У паралельні регістри запис коду відбувається одночасно в усі розряди регістра. Паралельні регістри можуть бути побудовані з будь-яких типів тригерів.

У послідовні регістри запис інформації відбувається у результаті зсуву двійкової інформації вправо або вліво. Для побудування послідовних регістрів можливо використовувати лише двохступеневі синхронні тригери.

Послідовно-паралельні забезпечують можливість організації схеми регістра будь-якого типу, для цього вони доповнюються логічними схемами, які забезпечують необхідну організацію подавання сигналів на входи тригерів регістра, відповідно до сигналів керування цими логічними схемами.

Регістри є важливими елементами цифрової та мікропроцесорної техніки.

Регістр – це операційний елемент, що служить для запам'ятовування слів і забезпечує в загальному випадку виконання наступних мікрооперацій:

- установка регістра в 0 (скидання);
- прийом слова з іншого регістра, шини і т.п.;
- передача слова на інший регістр, шину і т.п.;
- перетворення кодів збережених слів в інверсні коди;
- зсув збереженого слова ліворуч або праворуч на необхідну кількість розрядів.

Регістр, що виконує такі мікрооперації, називається **багатофункціональним**. Оскільки регістр призначений для збереження інформації, він містить елементи пам'яті, у якості яких використовуються тригери. Кількість тригерів визначає розрядність регістра. Будемо позначати регістр у вигляді прямокутника з вказівкою розрядності (рис. 9.7).

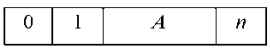


Рис.9.7 – Регістр *A(0;n)*

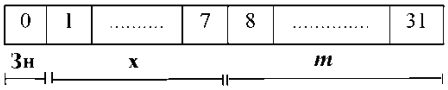


Рис.9.8 – Регістр для зберігання числа у формі з плаваючою комою

Регістр може складатися з окремих підрегістрів, що мають самостійне значення (рис. 9.7). На рис. 9.8 представлений регістр, що зберігає число у формі з плаваючою комою. У цьому регістрі три підрегістри: для збереження знака Рг(0), характеристики Рг(1:7), мантиси Рг(8:31) числа.

Регістр може виконувати ряд мікрооперацій, наприклад:

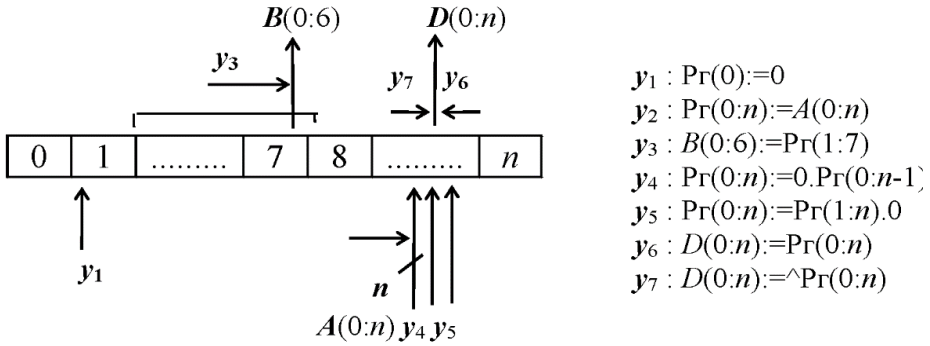


Рис.9.9 – Багатофункціональний регістр

Регістр, що виконує мікрооперацію y_4 (зсуву праворуч) або y_5 (зсуву ліворуч) називається **регістром зсуву**.

Лічильники

Функціональний типовий вузол комп'ютера, призначений для підрахунку вхідних імпульсів називається лічильником.

Лічильник – це послідовнісний функціональний вузол, створений із послідовно з'єднаних Т-тригерів і призначений для лічби вхідних імпульсів. Вхідні імпульси можуть надходити на лічильник як періодично, так і довільно розподіленими у часі. Амплітуда і тривалість лічильних імпульсів мають задовольняти технічні вимоги для серій мікросхем, які використовуються. Розрядність лічильника *n* дорівнює числу Т-тригерів. Кожний вхідний імпульс змінює стан лічильника, який зберігається до надходження наступного сигналу. Значення виходів тригерів лічильника відображають результат лічби. По мірі надходження вхідних імпульсів лічильник послідовно перебирає свої стани у визначеному для даної схеми порядку. Довжина списку станів лічильника, що використовуються, називається коефіцієнтом (модулем) лічби. Один із станів обирають за початковий. Після підрахунку імпульсів лічильник повертається у початковий стан.

Послідовність станів лічильника можна кодувати різними способами. Найчастіше використовують двійкові лічильники, у яких порядок зміни станів тригерів відповідає послідовності двійкових кодів, та побудовані на їх основі лічильники, що працюють у двійково-десятковій системі. Застосовуються й інші види кодування, наприклад, одинарне («один із n»), коли стан лічильника визначається розміщенням однієї-єдиної одиниці (або нуля), та лічильники з унітарним кодуванням, коли стан визначається кількістю одиниць і нулів (лічильник Джонсона).

До основних параметрів лічильників, окрім модуля лічби, відносять швидкодію. Швидкодію лічильників оцінюють трьома параметрами. Це:

- роздільна здатність $t_{рз}$ – мінімальний час між двома вхідними імпульсами, при якому ще зберігається працездатність лічильника;
- максимальна частота лічби $F_{\max} = \frac{1}{t_{рз}}$ – величина, обернена до

роздільної здатності, що визначає кількість імпульсів, які може підрахувати лічильник за 1 секунду;

– час встановлення вихідного коду $t_{вст}$ – часовий інтервал між моментом приходу вхідного сигналу і переходом лічильника в новий стійкий стан.

Лічильники можуть бути з попереднім встановленням або без нього. Для попереднього встановлення початкового стану лічильника використовують спеціальні входи, а сама процедура встановлення здійснюється лише за наявності спеціального сигналу дозволу. Під час роботи лічильника входи попереднього встановлення блокуються і не впливають на лічбу. Лічильники з попереднім встановленням називають також програмованими.

Усе різноманіття *n*-розрядних лічильників можна класифікувати за такими ознаками:

- за способом кодування станів лічильника розрізняють двійкові лічильники, лічильники з кодом «один із *n*», лічильники з унітарним кодуванням та інші;
- за модулем лічби лічильники поділяють на
  - двійкові;
  - двійкові з довільним постійним модулем лічби. Найбільшого поширення набули десяткові (декадні) лічильники;
  - двійкові зі змінним модулем лічби;
  - недвійкові. Лічильники з одинарним кодуванням мають модуль лічби , з унітарним кодуванням;
- за напрямом лічби розрізняють прості (підсумовуючі, віднімальні) і реверсивні лічильники. Якщо коди станів лічильника змінюються під час лічби у зростаючому порядку, то лічильник називають підсумовуючим. Лічильники, коди станів яких чергуються у порядку спадання, називають віднімальними. У реверсивних лічильників напрям перебору кодів може змінюватися залежно від керуючого сигналу;
- за структурою організації міжрозрядних зв’язків розрізняють
  - лічильники з послідовним перенесеннями (позикою), в яких перемикання розрядних тригерів відбувається по черзі один за одним;
  - лічильники з паралельним перенесеннями (позикою), в яких усі розрядні тригери перемикаються одночасно за сигналом синхронізації;
  - лічильники з комбінованим перенесеннями (позикою), в яких використовують різні комбінації послідовних і паралельних перенесень;
- за способом перемикання тригерів під час підрахунку імпульсів (за приналежністю до певного класу автоматів) лічильники поділяють на асинхронні та синхронні; Лічильники з послідовним перенесенням

як правило асинхронні, з паралельним перенесенням – синхронні;

- за типом тригерів, використаних для побудови лічильника, розрізняють лічильники на Т-тригерах, D-тригерах у лічильному режимі, JK-тригерах у лічильному режимі.

Класифікаційні ознаки незалежні і можуть зустрічатися у різних поєднаннях.

Лічильник – операційний елемент, що реалізує мікрооперацію лічби, яка складається в зміні стану лічильника (значення збереженого слова) на **1**. Крім того, лічильник може виконати і такі мікрооперації: установка в **0** і прийом довільного числа.

Тобто повний набір можливих мікрооперацій для лічильника (Лч):

Лічильник, що виконує мікрооперацію y_1 називається **підсумовуючим**, мікрооперацію y_2 – **віднімаючим**, обидві мікро- операції – **реверсивним**.

y_1 : Лч(0:n):=Лч(0:n)+1
y_2 : Лч(0:n):=Лч(0:n)-1
y_3 : Лч(0:n):=0
y_4 : Лч(0:n):=A(0:n)

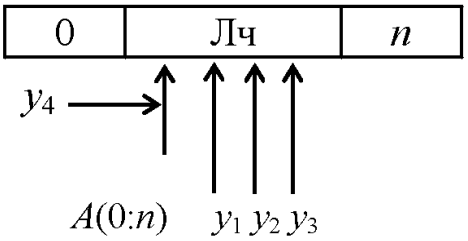


Рис. 9.10 – Лічильник

2. Базові вузли ЦА на комбінаційних схемах: суматори, дешифратори, шифратори, селектори імпульсів, генератори синхросигналів.

Суматор – операційний елемент, що виконує додавання кодів чисел. У залежності від кодів чисел розрізняють суматори прямого, оберненого, додаткового кодів.

Суматором називається функціональний вузол комп'ютера, призначений для додавання двох n-розрядних слів (чисел). Операція віднімання заміняється додаванням слів в оберненому або додатковому коді. Операції множення та ділення зводяться до реалізації багаторазового додавання та зсуву Тому суматор є важливою частиною арифметико-логічного пристрою. Функція суматора позначається буквами SM.

Суматор складається з окремих схем, які називаються однорозрядними суматорами; вони виконують усі дії з додавання значень однойменних розрядів двох чисел (операндів). Суматори класифікуються за такими ознаками:

1. способом додавання - паралельні, послідовні та паралельно-послідовні;
2. числом входів - напівсуматори, однорозрядні та багаторозрядні суматори;
3. організацією зберігання результату додавання – комбінаційні, накопичувальні, комбіновані;
4. організацією перенесення між розрядами – з послідовним, наскрізним,

паралельним або комбінованим перенесеннями (з груповою структурою);
5. системою числення – позиційні (двійкові, двійково-десяткові, трійкові) та непозиційні, наприклад, у системі залишкових класів;
6. розрядністю (довжиною) операндів – 8-, 16-, 32-, 64-розрядні;
7. способом представлення від'ємних чисел – в оберненому або додатковому кодах, а також в їхніх модифікаціях;
8. часом додавання – синхронні, асинхронні.

У паралельних *n*-розрядних суматорах значення всіх розрядів операндів надходять одночасно на відповідні входи однорозрядних підсумовуючих схем. У послідовних суматорах значення розрядів операндів та перенесення, що запам'ятовувалися в минулому такті, поступають послідовно в напрямку від молодших розрядів до старших на входи одного однорозрядного суматора. В паралельно-послідовних суматорах числа розбиваються на частини, наприклад, байти, розряди байтів поступають на входи восьмирозрядного суматора паралельно (одночасно), а самі байти – послідовно, в напрямку від молодших до старших байтів з врахуванням запам'ятованого перенесення.

У комбінаційних суматорах результат операції додавання запам'ятовується в регістрі результату. В накопичувальних суматорах процес додавання поєднується із зберіганням результату. Це пояснюється використанням Т-тригерів як однорозрядних схем додавання.

Організація перенесення практично визначає час виконання операції додавання. Послідовні перенесення схемно створюються просто, але є повільнодіючими. Паралельні перенесення схемно організуються значно складніше, але дають високу швидкодію.

Розрядність суматорів знаходиться в широких границях: 4–16 – для мікро- та міні-комп'ютерів та 32–64 і більше – для універсальних машин.

Суматори з постійним інтервалом часу для додавання називаються синхронними. Суматори, в яких інтервал часу для додавання визначається моментом фактичного закінчення операції, називаються асинхронними. В асинхронних суматорах є спеціальні схеми, які визначають фактичний момент закінчення додавання і повідомляють про це в пристрій керування. На практиці переважно використовуються синхронні суматори.

Суматори характеризуються такими параметрами:

1) швидкодією – часом виконання операції додавання ta, який відраховується від початку подачі операндів до одержання результату; часто швидкодія характеризується кількістю додавання в секунду Fa=1/ta, тут маються на увазі операції типу регістр– регістр (тобто числа зберігаються в регістрах АЛП);

2) апаратурними витратами: вартість однорозрядної схеми додавання визначається загальним числом логічнихвходів використаних елементів; вартість багаторозрядного суматора визначається загальною кількістю використаних мікросхем;

3) споживаною потужністю суматора.

Повний однорозрядний суматор - це пристрій, що здійснює додавання за *mod 2* відповідних розрядів *(X1,X2)* двійкових чисел з урахуванням переносу (P_m) у даний розряд із сусіднього молодшого розряду суми. Суматор виробляє цифру результату (*S*) у даному розряді і перенос (P_c) у сусідній старший розряд суми. Таблиця істинності (табл. 9.11) такого суматора представлена нижче.

Умовне графічне позначення однорозрядного суматора наведене на рисунку 9.11.

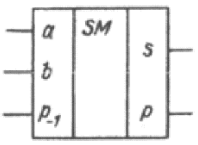


Рис. 9.11 – Умовне позначення однорозрядного суматора

Таблиця 9.3

Таблиця істинності повного однорозрядного двійкового суматора

X_1	0	0	0	0	1	1	1	1
X_2	0	0	1	1	0	0	1	1
P_m	0	1	0	1	0	1	0	1
S	0	1	1	0	1	0	0	1
P_c	0	0	0	1	0	1	1	1

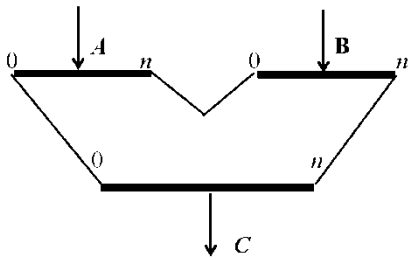


Рис.9.12 – Комбінаційний суматор, A,B- операнди, C- результат

Багаторозрядні суматори виконують операцію арифметичного додавання двох багаторозрядних чисел. Кількість входів та виходів суматора визначається розрядністю доданків.

За організацією переносу розрізняють суматори з послідовним та паралельним переносом. За першим способом побудовані, наприклад, чотирьохрозрядні суматори К155ИМ3, К555ИМ7. Швидкодія такого суматора визначається часом розповсюдження сигналу через усі його елементи, і тому вона значно нижча за швидкодію елементів.

Суматори бувають комбінаційними і накопичуючими.

Комбінаційний суматор виробляє вихідні сигнали суми і переносу, обумовлені комбінацією одночасно поданих на входи суматора цифр, що додаються. Даний суматор не має пам'яті і після зняття сигналів зі входів вихідні сигнали також зникають.

Умовне позначення комбінаційного суматора представлене на рис. 9.12.

Накопичуючим суматором називається суматор, що здійснює додавання слів *A* і *B* при подачі їх на суматор одного за іншим. У накопичуючому суматорі мається додатковий регістр для збереження результату.

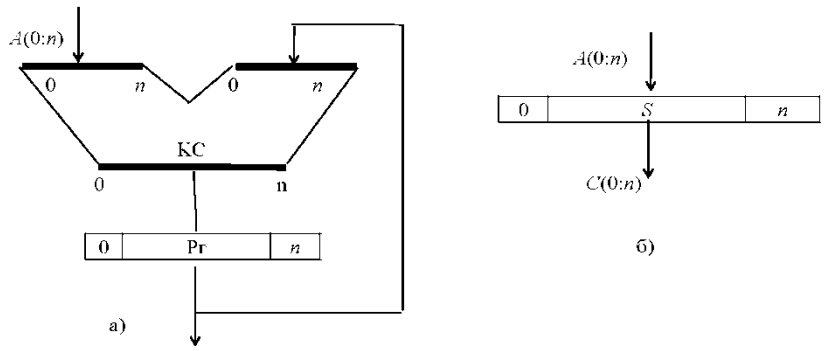


Рис. 9.13 – Накопичуючий суматор (а) структура, (б) умовне позначення

Структура та умовне позначення накопичуючого суматора представлені на рис. 9.13.

Напівсуматор — логічна схема з двома входами та двома виходами, який виконує операцію арифметичного додавання двох однорозрядних чисел A та B у відповідності до наступного правила: при будь-яких наборах сигналу A та B на виході сигналу суми S' формується результат додавання по модулю два, на виході сигналу переносу P' у всіх випадках буде 0, крім A=B=1, тоді P'=1. Таким чином, для реалізації напівсуматора необхідні суматор по модулю два та логічний елемент ТА.

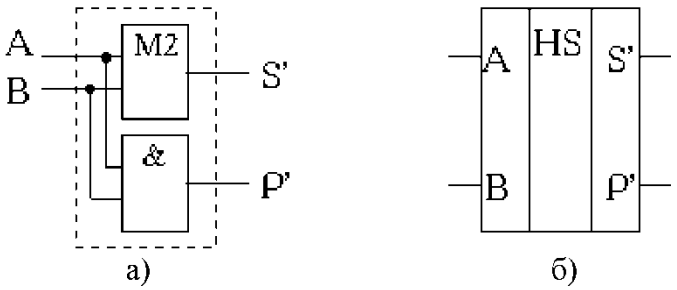


Рис.9.14 – Напівсуматори: а) функціональна схема; б) умовне позначення

Перетворювачем коду називається комбінаційний пристрій, призначений для зміни виду кодування інформації.

Як і будь-який комбінаційний пристрій, перетворювач коду характеризується таблицею істинності, яка ставить у відповідність кодам, що подаються на вхід, коди, що знімаються з виходу комбінаційного пристрою. Слід зазначити, що в цій таблиці в загальному випадку число розрядів вхідного і вихідного кодів може не співпадати. Головне – вона повинна давати однозначну відповідність різних кодів. Дана таблиця є основою для синтезу логічної структури конкретного перетворювача коду. Умовне графічне позначення перетворювача коду на схемах наведено на рисунку 9.15.

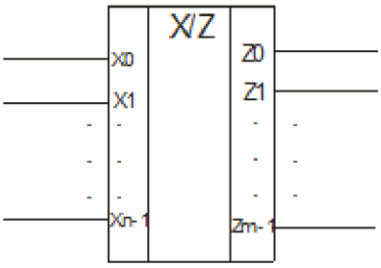


Рис.9.15 – Умовне графічне позначення перетворювача коду

В якості прикладу перетворювача кодів, що випускається у вигляді інтегральної схеми, можна навести схеми, що забезпечують перетворення інформації з двійкового в двійково-десятковий код. Частим випадком перетворювача кодів є шифратори та дешифратори.

Шифратором називається пристрій, який перетворює вхідний сигнал одного із його входів у кодову комбінацію на його виходах. Функціонує зворотно до функціонування дешифратора. Повний шифратор має 2^n входів і n виходів. На рисунку 9.16 для прикладу показаний шифратор, який має 8 входів і 3 виходи.

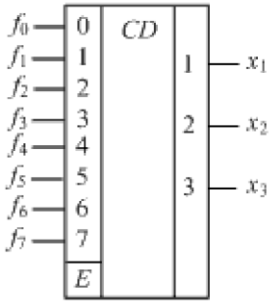


Рис. 9.16 – Шифратор

Призначення: Такий шифратор, наприклад, можна використати для відображення натиснутої кнопки генератора символів. В таблиці 9.4 показана логіка функціонування цього шифратора, а на рисунку 9.17 - функціональна схема, яка його реалізує.

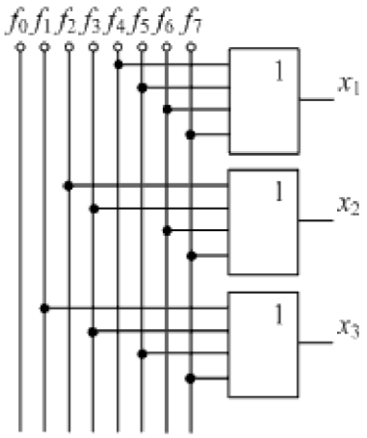


Рис. 9.17 – Схема шифратора

Таблиця 9.4

Функціонування шифратора

f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	x_1	x_2	x_3
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Дешифратор - операційний елемент, що виконує функцію перетворення деякого n-розрядного двійкового коду в унітарний код «один з N». Якщо $N=2^n$, то такий дешифратор називається повним, якщо $N<2^n$, то частковим. Таблиця істинності найпростішого повного дешифратора (n=2) і його умовні позначення на функціональних та принципових схемах наведені в табл. 9.5 і на рис. 9.18.

Таблиця 9.5

Таблиця істинності дешифратора

e_1	e_0	f_0	f_1	f_2	f_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

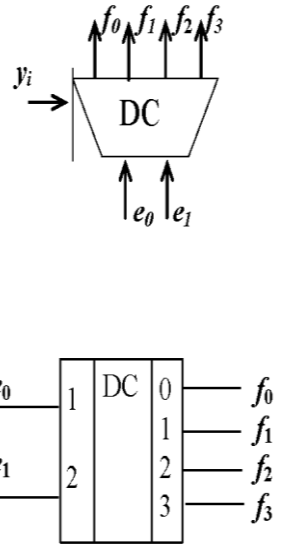


Рис. 9.18 – Дешифратор

Промисловість може випускати дешифратори з інверсними виходами. Для такого дешифратора таблиця істинності та умовне позначення на принципових схемах мають наступний вигляд (табл. 9.6., рис. 9.19)

Таблиця 9.6

Таблиця істинності дешифратора

x_1	x_0	y_0	y_1	y_2	y_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

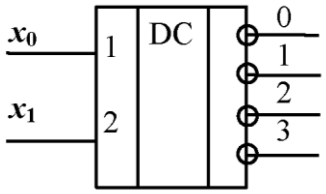


Рис. 9.19 – Дешифратор з інверсними входами

Селектори імпульсів

Селекторами імпульсів називають такі пристрої, які виділяють із деякої сукупності різноманітних сигналів імпульсні сигнали із заданими параметрами.

Селектори широко застосовуються в автоматиці, радіотехніці, телебаченні, радіолокації.

Технічні умови та роботи, які ставляться до них, досить різноманітні. Інколи потрібно передати на вихід пристрою імпульси з заданими параметрами і забезпечити при цьому найменше його спотворення. Найчастіше потрібно отримати тільки відгук пристрою на імпульси з заданими параметрами, причому характер відгуку може бути довільним.

У цьому випадку пристрій селекції називають у відповідності до переліку основних параметрів імпульсних сигналів поділяють на:

- частотні;
- амплітудні;
- тривалості;
- часові.

Селектори, які виділяють імпульси з заданим часовим положенням називають часовими. Кожний із селекторів може вирішувати різноманітні функціональні задачі: виділяти імпульси, у яких значення параметру, який селектується менше або більше за задане значення; лежить в заданому діапазоні значень або відповідає з визначеною ступіню точності фіксованому значенню параметра.

Амплітудними – називають такі селектори імпульсів, які вибирають із деякої сукупності різноманітних сигналів імпульсні сигнали із заданою амплітудою.

В амплітудних селекторах параметром селекції є амплітуда імпульсних сигналів, які подаються на вхід пристрою. За їх допомогою із імпульсної

послідовності виділяють ті імпульси, амплітуда яких більша або менша за амплітуду визначеного рівня, який називається - *рівнем (порогом) селекції*. Можна також виділити ті імпульси, амплітуда яких більша за амплітуду одних із заданих рівнів але менша за інші.

Селектори, які реагують на тривалість імпульсів називають - *квазіселекторами*. За правило ці селектори тільки регіструють наявність імпульсів тої чи іншої тривалості на вході пристрою

Для виготовлення селекторів імпульсів, які виконують селекцію за тривалістю імпульсів використовують різні технічні методи, наприклад:

- перетворення тривалості імпульсів в амплітуду пилкоподібної напруги з послідуючим застосуванням методів селекції за амплітудою;
- підрахунок тактових імпульсів, які подаються на пристрій за час дії вхідного імпульсу;
- порівняння тривалості імпульсів, які подаються на пристрій з тривалістю відомого імпульсу;
- порівняння тривалості імпульсів, які подаються на пристрій з відомого величиною часової затримки.

Часовими – називають такі селектори імпульсів, які вибирають тільки ті імпульси, які надходять на їх вхід в задані інтервали часу.

Селекція за часом виконується шляхом порівняння заданої послідовності імпульсів із спеціальною, яка складається із вирішувальних (стробуючих або селекторних) імпульсів, часове положення яких однозначно визначене.

Генератори синхросигналів

Генерування періодичних синусоїдних та несинусоїдних напруг здійснюється за допомогою генераторів – пристроїв, в яких виникають та автоматично підтримуються незатухаючі електричні коливання.

Такі пристрої перетворюють енергію джерела живлення в енергію незатухаючих коливань. Генератори містять в собі: активний елемент та частотно–вибірковий чотириполюсник Активний елемент це, як правило, транзистори (біполярні або польові), операційні підсилювачі (ОП). Як чотириполюсники в звуковому діапазоні (на низьких частотах) використовують диференціювальні, або інтегрувальні RC–ланцюги.

Генератор – це певна схема, яка самовільно (чи за зовнішньою командою) формує послідовність сигналів, яка характеризується періодом повторення і формою.

Класифікація генераторів:

I. За функціональними можливостями генератори поділяють на:

- автогенератори;
- генератори затримки (генератори, які запускаються зовнішнім сигналом).

II. За формою генерованого сигналу розрізняють:

- генератори гармонійних сигналів;
- генератори прямокутних імпульсів;

– генератори сигналів спеціальної форми (пилкоподібної, прямокутної тощо).

До генераторів можна також віднести і тригери, тобто схеми електронної пам’яті, які мають два постійних стани рівноваги і також спрацьовують від пускових імпульсів.

Контрольні питання:

1. Які типи тригерів ви знаєте? В чому їх відмінності?
2. Які тригери використовуються в якості елементів пам’яті структурного автомата?
3. Які особливості побудови таблиць переходів для різних тригерів?
4. Як будується таблиця збудження тригера?
5. Як отримати системи рівнянь функцій збудження тригера і виходів автомата?
6. Яку класифікацію регістрів ви знаєте? Перелічіть їх основні параметри.
7. Призначення шифратора?
8. Пояснити властивість пріоритетності, яке реалізовано в шифраторах.
9. Пояснити різницю між шифратором та мультиплексором.
10. Призначення дешифратора?
11. Як за допомогою дешифратора реалізувати функцію, задану, наприклад, її таблицею істинності?
12. Привести визначення суматора.
13. По яких ознаках класифікують суматори?
14. Поясніть відмінності в побудові двійкового й двійково-десяткового суматорів.
15. Назвіть способи збільшення швидкодії суматорів
16. Які входи має суматор?
17. Які виходи має напівсуматор?
18. Чому до складу пристрою узгодження можуть входити тільки реактивні елементи і які вимоги до них?
19. Поясніть призначення селекторів і особливості їх побудови.

ТЕМА 10. СПОСОБИ ПОДАННЯ ЦИФРОВОГО АВТОМАТА

План:

1. Математична модель цифрового автомату. Автомати Мілі і Мура.
2. Способи подання цифрових автоматів: табличний і графічний (орграфом). Еквівалентність цифрових автоматів.
3. Структурна модель цифрового автомату.
4. Етапи структурного синтезу цифрового автомату.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Математична модель цифрового автомату. Автомати Мілі і Мура.

На практиці обчислювальні пристрої зазвичай сполучають комбінаційну та секвенціальну логіку.

Секвенціальна логіка - це логіка пам'яті цифрових пристроїв. Така логіка може називатись також послідовнісною. У секвенціальній логіці вихідне значення залежить не лише від поточних вхідних впливів, але й від передісторії функціонування цифрового пристрою. Іншими словами, секвенціальна логіка передбачає наявність пам'яті, яка в комбінаційній логіці не передбачена.

Цифровий автомат - це приклад пристрою, реакція якого залежить не лише від значень входів, але й від стану в попередній момент часу. Синонімом терміну «цифровий автомат» є термін «кінцевий автомат». Перший термін підкреслює, що автомат працює з цифровою (дискретною) інформацією, другий - що його пам'ять кінцева. Кінцевий автомат - абстракція, яка дозволяє не розглядати динамічні стани, які виникають під час перехідних процесів. Кінцевий автомат розглядає стани перед початком і після завершення переходу, в проміжні моменти часу кінцево-автоматний опис неможливий. Методи конструювання реальних пристроїв дозволяють розглядати кінцево-автоматний опис як опис функції автомату, а перехідні процеси врахувати і приховати від зовнішнього світу. Будь-який кінцевий автомат може бути представлений програмою.

Відмінності цифрового автомату (ЦА) від кінцевого автомату:

1. ЦА призначений для представлення чисел та виконання операцій над ними;
2. ЦА має похибку представлення - похибку, яка виникає при представленні чисел через обмежену кількість розрядів та пам'яті;
3. ЦА надто складний для автоматичного синтезу.

Цифровий автомат є більш складним пристроєм, ніж логічна схема, оскільки включає в себе декілька логічних схем. Його вихідні слова є складними логічними залежностями, які можуть бути одержані за допомогою найпростіших логічних операцій над вхідними словами.

ЦА обов'язково містить пам'ять, яка складається із запам'ятовуючих елементів, фіксуючих стан, в якому він знаходиться.

Цифрові автомати, у яких вихідні сигнали визначаються вхідними сигналами та станом автомату в попередній момент часу називають цифровими **автоматами Мілі**. На відміну від них є автомати, для яких вихідні сигнали залежать лише від стану автомату та не залежать від значень вхідних сигналів, - **автомати Мура**.

Структурний цифровий автомат, на відміну від абстрактного, є його подальшою деталізацією, коли розглядається як його внутрішня структура, так і структура вхідних та вихідних сигналів.

Ознайомившись із принципами роботи цифрової ЕОМ, можна вказати на дві основні особливості таких обчислювальних машин: оперування даними, представленими в цифровій формі, й автоматична робота за заздалегідь складеною програмою. Ці особливості показують, що будь-яка цифрова ЕОМ є цифровим автоматом (ЦА). Поняття ЦА служить узагальненням для усіх видів пристроїв обробки цифрової інформації, що мають програмне керування.

Цифровий автомат – це пристрій, що характеризується набором внутрішніх станів, у які він потрапить під впливом команд закладеної в нього програми. Перехід автомата з одного стану в інший здійснюється у визначений момент часу.

Математичною моделлю ЦА(а в загальному випадку будь-якого дискретного пристрою) є так званий абстрактний автомат, визначений як 6-компонентний корте ж $\mathbf{S=(A,Z,W,\delta,\lambda,a_1)}$ у якого:

1. $\mathbf{A=\{a_1, a_2, \dots, a_m\}}$ - множина станів (внутрішній алфавіт)
2. $\mathbf{Z=\{z_1, z_2, \dots, z_f\}}$ - множина вхідних сигналів (вхідний алфавіт)
3. $\mathbf{W=\{w_1, w_2, \dots, w_g\}}$ - множина вихідних сигналів (вихідний алфавіт)
4. $\mathbf{\delta : A\bullet Z \to A}$ - функція переходів, що реалізує відображення $\mathbf{D\delta \subseteq A\bullet Z}$ в **A**. Іншими словами, деяким парам «стан - вхідний сигнал» $\mathbf{(a_m, z_f)}$ функція δ ставить у відповідність стани автомата $\mathbf{a_s = \delta (a_m, z_f),\ a_s \in A.}$
5. $\mathbf{\lambda : A\bullet Z \to W}$ - функція виходів, що реалізує відображення $\mathbf{D\lambda \subseteq A\bullet Z}$ на **W**. Функція λ деяким парам “стан - вхідний сигнал” $\mathbf{(a_m, z_f)}$ ставить у відповідність вихідні сигнали автомата $\mathbf{Wg=\lambda(a_m, z_f),\ Wg \in W.}$
6. $\mathbf{a_1 \in A}$ - початковий стан автомата.

Під **алфавітом** тут розуміється непорожня множина попарно різних символів. Елементи алфавіту називаються **буквами**, а кінцева упорядкована послідовність букв - **словом** у даному алфавіті.

Абстрактний автомат (рис. 10.1) має один вхід і один вихід. Автомат працює в дискретному часі, що приймає цілі невід’ємні значення $\mathbf{t = 0,1,2,\dots}$ У кожен момент **t** дискретного часу автомат знаходиться в деякому стані **a(t)** з

множини станів автомата, причому в початковий момент **t = 0** він завжди знаходиться в початковому стані **a(0)=a$_1$**..

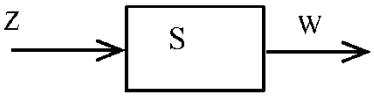


Рис. 10.1 – Абстрактний автомат

В момент **t**, перебуваючи в стані **a(t)**, автомат здатний сприйняти на вході букву вхідного алфавіту **z(t) ∈ Z**. Відповідно до функції виходів він видасть у той же момент часу **t** букву вихідного алфавіту **w(t)=λ(a(t), z(t))** і відповідно до функції переходів перейде в наступний стан **a(t+1)=δ[a(t), z(t)], a(t) ∈A, w(t) ∈W.**

Зміст поняття абстрактного автомата полягає в тому, що він реалізує деяке перетворення множини слів вхідного алфавіту **Z** у множину слів вихідного алфавіту **W**. Якщо на вхід автомата, встановленого в початковий стан **a$_1$**, подавати буква за буквою деяку послідовність букв вхідного алфавіту **z(0), z(1),...** - вхідне слово, то на виході автомата будуть послідовно з'являтися букви вихідного алфавіту **w(0), w(1),...** , тобто вихідне слово. Таким чином вихідне слово = φ(вхідне слово), де φ - відображення, здійснюване абстрактним автоматом.

На рівні абстрактної теорії поняття «робота автомата» розуміється як перетворення вхідних слів у вихідні. Можна сказати, що в абстрактному автоматі ми відволікаємося від його структури - вмісту прямокутника (рис. 10.1), розглядаючи його як «чорну скриньку», тобто основну увагу приділяємо поводженню автомата щодо зовнішнього середовища.

Поняття стану у визначенні автомата введено в зв'язку з тим, що часто виникає необхідність в описі поводження систем, виходи яких залежать не тільки від значень входів у даний момент часу, але і від деякої передісторії, тобто від сигналів, що надходили на входи системи раніше. Саме стани і відповідають деякій пам'яті про минуле, дозволяючи усунути час як явну змінну і виразити вихідний сигнал як функцію стану і входу в даний момент часу.

На практиці найбільшепоширення одержали два класи автоматів - автомати **Мілі** (Mealy) і **Мура** (Moore).

Закон функціонування автомата Мілі задається рівняннями:

a(t+1) =δ((a(t), z(t)); w(t) = λ((a(t), z(t)), t = 0,1,2,...

Закон функціонування автомата Мура задається рівняннями:

a(t+1)= δ((a(t), z(t)); w(t) =λ((a(t)), t = 0,1,2,...

З порівняння законів функціонування видно, що, на відміну від автомата Мілі, вихідний сигнал в автоматі Мура залежить тільки від поточного стану автомата й у явному вигляді не залежить від вхідного сигналу. Для повного задання автоматів Мілі або Мура необхідно вказати, додатково до законів

функціонування, початковий стан і визначити внутрішній, вхідний і вихідний алфавіти.

Крім автоматів Мілі і Мура, іноді виявляється зручним користуватися сполученою моделлю автомата, так званим **С-автоматом**.

Під абстрактним **С**-автоматом будемо розуміти математичну модель дискретного пристрою, обумовлену восьмикомпонентним вектором **S=(A, Z, W, U, δ, λ_1, λ_2, a_1)**, у якого:

1. **$A=\{a_1, a_2, \ldots, a_m\}$** - множина станів;
2. **$Z=\{z_1, z_2, \ldots, z_f\}$** - вхідний алфавіт;
3. **$W=\{w_1, w_2, \ldots, w_g\}$** - вихідний алфавіт типу 1;
4. **$U=\{u_1, u_2, \ldots, u_h\}$** - вихідний алфавіт типу 2;
5. **$\delta : A \bullet Z \rightarrow A$** - функція переходів, що реалізує відображення (вхідний сигнал, стан) **$D\delta \subseteq A \bullet Z$** в **А**;
6. **$\lambda_1 : A \bullet Z \rightarrow W$** - функція виходів, що реалізує відображення **$D\lambda_1 \subseteq A \bullet Z$** у **W**;
7. **$\lambda_2 : A \rightarrow U$** - функція виходів, що реалізує відображення **$D\lambda_2 \subseteq A$** в **U**;
8. **$a_1 \in A$** - початковий стан автомата.

Абстрактний **С**- автомат можна представити у вигляді пристрою з одним входом, на який надходять сигнали з вхідного алфавіту **Z**, і двома виходами, на яких з'являються сигнали з алфавітів **W** і **U**. Відмінність **С** - автомата від моделей Мілі і Мура полягає в тому, що він одночасно реалізує дві функції виходів λ_1 і λ_2, кожна з яких характерна для цих моделей окремо. Закон функціонування **С**- автомата можна описати наступними рівняннями:

$a(t+1) = \delta(a(t), z(t)); w(t) = \lambda_1(a(t), z(t)); u(t) = \lambda_2(a(t)); t = 0, 1, 2, \ldots$

Вихідний сигнал **$u_h=\lambda_2(a_m)$** видається весь час, поки автомат знаходиться в стані **a_m**. Вихідний сигнал **$w_g=\lambda_1(a_m, z_f)$** видається під час дії вхідного сигналу **z_f** при перебуванні автомата в стані **a_m.**

Розглянуті вище абстрактні автомати можна розділити на:

1. цілком визначені і часткові;
2. детерміновані і ймовірнісні;
3. синхронні й асинхронні;

Цілком визначеним називається абстрактний цифровий автомат, у якого функція переходів і функція виходів визначені для всіх пар **(a_i, z_j).**

Частковим називається абстрактний автомат, у якого або функція переходів, або функція виходів, або обидві ці функції визначені не для всіх пар **(a_i, z_j).**

До **детермінованих** належать автомати, у яких виконана умова однозначності переходів: автомат, що знаходиться в деякому стані **a_i**, під дією будь-якого вхідного сигналу **z_j** не може перейти більш, ніж в один стан.

Інакше це буде **імовірнісний автомат**, у якому при заданому стані **a_i** та заданому вхідному сигналі **z_j** можливий перехід із заданою ймовірністю в різні стани.

Для визначення синхронних і асинхронних автоматів вводиться поняття стійкого стану. **Стан a_s** автомата **називається стійким**, якщо для будь-якого стану a_i і вхідного сигналу z_j таких, що $\delta(a_i,z_j)=a_s$, має місце $\delta(a_s,z_j)=a_s$, тобто стан стійкий, якщо потрапивши в цей стан під дією деякого сигналу z_j, автомат вийде з нього тільки під дією іншого сигналу z_k, відмінного від z_j.

Автомат, у якого всі стани стійкі - **асинхронний**, інакше автомат є **синхронним**. Абстрактний автомат називається **кінцевим**, якщо кінцевими є множини

$A = \{a_1, a_2, ..., a_m\}$, $Z = \{z_1, z_2, ..., z_f\}$, $W = \{w_1, w_2, ..., w_g\}$.

Автомат зветься **ініціальним**, якщо в ньому виділено початковий стан a_1.

2. Способи подання цифрових автоматів: табличний і графічний (орграфом). Еквівалентність цифрових автоматів.

Для того, щоб задати автомат, необхідно описати усі компоненти кортежу $S=(A, Z, W, \delta, \lambda, a_1)$. Множини **A, Z, W** описуються і задаються простим перерахунком своїх елементів. Серед різноманіття різних способів завдань функцій δ і λ (отже і всього автомату в цілому) найбільше поширення одержали табличний і графічний.

При табличному способі завдання автомат Мілі описується за допомогою двох таблиць. Одна з них (таблиця переходів) задає функцію δ, тобто $a(t+1)=((a(t), z(t))$ (табл. 10.1), друга (таблиця виходів) - функцію λ , тобто $W(t)=((a(t), z(t))$ (табл. 10.2).

Табл. 10.1. Таблиця переходів автомата Мілі

	a_1	a_2	a_3	a_4
z_1	a_2	a_1	a_2	a_3
z_2	a_3	a_4	a_1	a_1

Табл. 10.2. Таблиця виходів автомата Мілі

	a_1	a_2	a_3	a_4
z_1	w_1	w_2	w_2	w_2
z_2	w_2	w_3	w_4	w_5

Кожному стовпцеві з приведених таблиць поставлений у відповідність один стан з множини **A**, кожному рядкові - один вхідний сигнал з множини **Z**. На перетинанні стовпця a_m і рядка z_f у табл. 10.1 записується стан a_s, в який повинен перейти автомат зі стану a_m під дією вхідного сигналу z_f, тобто $a_s = \delta(a_m, z_f)$. На перетинанні стовпця a_m і рядка z_f у табл. 10.2 записується вихідний сигнал w_g, виданий автоматом у стані a_m при надходженні на вхід сигналу z_f, тобто $w_g = \lambda(a_m, z_f)$.

Для приведених таблиць множинами, що утворюють автомат, є $A=\{a_1, a_2, a_3, a_4\}$, $Z=\{z_1, z_2\}$, $W=\{w_1, w_2, w_3, w_4, w_5\}$. Автомат Мілі може бути заданий одною сполученою таблицею переходів і виходів (табл. 10.3), у якій кожен елемент a_s / w_g записаний на перетинанні стовпця a_m і рядка z_f, визначається в такий спосіб:

$$a_s=\delta(a_m, z_f); \quad w_f=\lambda(a_m, z_f).$$

Табл. 10.3. Сполучена таблиця переходів та виходів для автомата Мілі

	a_1	a_2	a_3	a_4
z_1	a_2/w_1	a_1/w_2	a_2/w_2	a_3/w_2
z_2	a_3/w_2	a_4/w_3	a_1/w_4	a_1/w_5

Автомат Мура задається одною відміченою таблицею переходів (табл. 10.4), у якій кожному стовпцеві приписаний не тільки стан $\mathbf{a_m}$, але й вихідний сигнал $\mathbf{w_g}$, що відповідає цьому станові, де $\mathbf{w_g=\lambda(a_m)}$.

Табл. 10.4. Відмічена таблиця переходів автомата Мура

	w_1	w_2	w_3	w_4
	a_1	a_2	a_3	a_4
z_1	a_1	a_2	a_2	a_3
z_2	a_2	a_3	a_4	a_1

Для часткових автоматів Мілі і Мура в розглянутих таблицях на місці невизначених станів і вихідних сигналів ставиться прочерк. У таких автоматах вихідний сигнал на якому-небудь переході завжди невизначений, якщо невизначеним є стан переходу. Крім того, вихідний сигнал може бути невизначеним і для деяких існуючих переходів.

Для задання **C** - автоматів також використовується табличний метод. У цьому випадку таблиця переходів (табл. 10.5) аналогічна таблиці переходів автомата Мілі, а в таблиці виходів кожен стан відмічений відповідним вихідним сигналом $\mathbf{u_i}$ вихідного алфавіту типу 2 (табл. 10.6).

Табл. 10.5. Таблиця переходів С – автомата

	a_1	a_2	a_3	a_4
z_1	a_1	a_2	a_2	a_3
z_2	a_3	a_4	a_1	a_2

Табл. 10.6. Таблиця виходів С – автомата

	u_1	u_2	u_3	u_4
	a_1	a_2	a_3	a_4
z_1	w_1	w_4	w_1	w_2
z_2	w_3	w_2	w_1	w_3

При графічному способі автомат задається у вигляді орієнтованого графа, вершини якого відповідають станам, а дуги - переходам між ними. Дуга, спрямована з вершини $\mathbf{a_m}$ у вершину $\mathbf{a_s}$, задає перехід в автоматі зі стану $\mathbf{a_m}$ у стан $\mathbf{a_s}$. На початку цієї дуги записується вхідний сигнал $\mathbf{z_f \in Z}$, що викликає даний перехід $\mathbf{a_s=\delta(a_m, z_f)}$. Для графу автомата Мілі вихідний сигнал $\mathbf{w_g \in W}$, який формується на переході, записується наприкінці дуги, а для автомата Мура - поруч з вершиною $\mathbf{a_m}$, відзначеною станом $\mathbf{a_m}$, у якому він формується. Якщо

перехід в автоматі зі стану $\mathbf{a_m}$ у стан $\mathbf{a_s}$ виробляється під дією декількох вхідних сигналів, то дузі графа, спрямованої з $\mathbf{a_m}$ у $\mathbf{a_s}$, приписуються всі ці вхідні і відповідні вихідні сигнали. Граф **С**-автомата містить вихідні сигнали двох типів, які позначаються на графі як на графах відповідних автоматів. Графи автоматів, заданих своїми таблицями переходів і виходів представлені на рисунках 10.2, 10.3, 10.4.

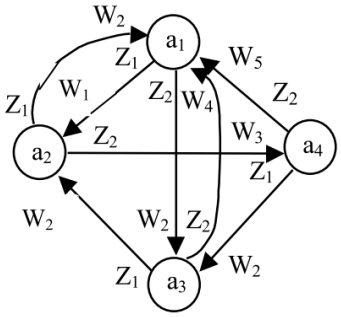


Рис.10.2 – Граф автомата Мілі

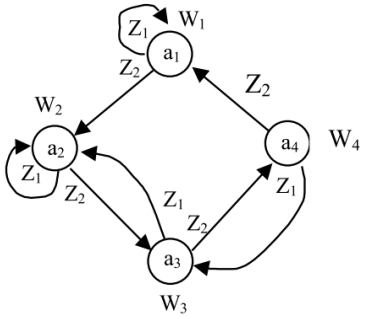


Рис.10.3 – Граф автомата Мура

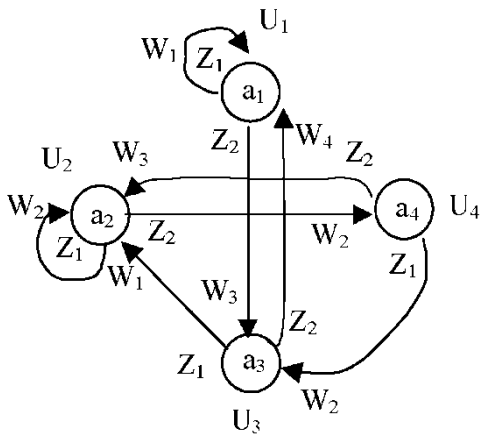


Рис. 10.4 Граф С-автомата

Два автомати з однаковими вхідними і вихідними алфавітами називаються ***еквівалентними***, якщо після установки їх у початковий стан їхні реакції на будь-яке вхідне слово збігаються.

Для кожного автомата Мілі може бути побудований еквівалентний йому автомат Мура і навпаки.

3. Структурна модель цифрового автомату

За етапом абстрактного синтезу автоматів випливає етап **структурного синтезу**, метою якого є побудова схеми, що реалізує автомат з елементів заданого типу. Якщо **абстрактний автомат** був лише математичною моделлю проектованого пристрою, то в **структурному автоматі** враховується структура вхідних і вихідних сигналів автомата, а також його внутрішні пристрої на рівні логічних схем. Основною задачею структурної теорії автоматів є розробка загальних методів побудови структурних схем автоматів.

На відміну від абстрактного автомата, що має один вхід і один вихід, на які надходять сигнали у вхідному $Z=\{Z_1,..,Z_F\}$ і виходять у вихідному $W=\{W_1,..,W_G\}$ алфавітах, структурний автомат має L вхідних каналів $x_1,x_2,..,x_L$ і N вихідних $y_1,y_2,\ldots,y_N$, на кожнім з яких є присутнім сигнал структурного алфавіту.

Звичайно в якості структурного використовується двійковий алфавіт.

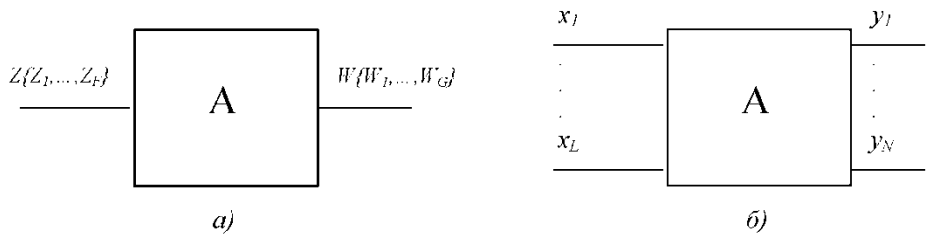


Рис. 10.5. Абстрактний *(а)* та структурний *(б)* автомати.

У цьому випадку кожному вхідному сигналові Z_F абстрактного автомата відповідає деякий двійковий вектор $(l_{f1},l_{f2},..,l_{fL})$, де $l_{fL}\in\{0,1\}$.

Очевидно, що для представлення (кодування) вхідних сигналів $Z_1,..,Z_F$ абстрактного автомата різними двійковими векторами повинна виконуватись умова $L \geq] \log_2 F [$, аналогічно $N \geq] \log_2 G [$.

Наприклад , $Z=\{Z_1,Z_2,Z_3,Z_4\}$, $W=\{W_1,W_2,W_3\}$.

Тоді $L \geq \log_2 4=2$, $N \geq \log_2 3=2$.

Закодувати вхідні і вихідні сигнали можна, скажімо, так:

$Z_1 = 00$ $W_1 = 00$
$Z_2 = 01$ $W_2 = 01$
$Z_3 = 11$ $W_3 = 11$
$Z_4 = 10$

Тобто структурний автомат з двома входами x_1 та x_2 і двома виходами y_1 та y_2 може бути представленим у вигляді:

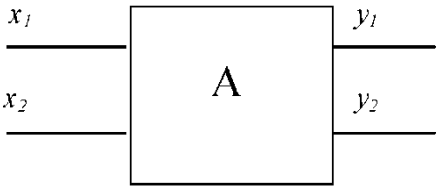


Рис. 10.6 – Структурний автомат з двома входами

4. Етапи структурного синтезу цифрового автомату

На етапі структурного синтезу попередньо обираються елементарні автомати, шляхом композиції яких будують логічні схеми отриманих на етапі абстрактного синтезу автоматів Мілі та Мура. Якщо розв'язок задачі структурного синтезу існує, кажуть, що задана **система автоматів структурно повна**.

Розглянемо **канонічний метод структурного синтезу** при якому використовуються елементарні автомати деякого спеціального виду – **автомати з пам'яттю**, що мають більше одного стану, і **автомати без пам'яті** – з одним станом. Перші автомати називаються **елементами пам'яті**, другі являють собою комбінаційні (логічні) елементи.

Теоретичним обґрунтуванням канонічного методу структурного синтезу автоматів служить **теорема про структурну повноту:** Будь-яка система елементарних автоматів, яка містить автомат Мура з нетривіальною пам’яттю, з повною системою переходів та повною системою виходів, та яку-небудь функціонально-повну систему логічних елементів, є структурно повною.

В цьому випадку задача структурного синтезу довільних автоматів зводиться до задачі структурного синтезу комбінаційних схем.

Результатом канонічного методу структурного синтезу є система логічних рівнянь, які задають залежність вихідних сигналів автомата і сигналів на входах запам’ятовуючих елементів від сигналів, які діють на вході усього автомату в цілому, і сигналів на виходах елементів пам’яті. Ці рівняння мають назву **канонічних рівнянь**.

Етапи канонічного методу структурного синтезу автоматів

Синтез комбінаційних схем зводиться до реалізації аналітичних виразів булевих функцій за допомогою логічних елементів. Один з методів синтезу цифрових автоматів з пам’яттю дозволяє звести задачу структурного синтезу довільного автомата з пам’яттю до задачі синтезу комбінаційних схем. Метод синтезу, в основу якого покладений вказаний принцип, отримав назву канонічного методу структурного синтезу автоматів з пам’яттю. Канонічний метод структурного синтезу оперує з елементарними автоматами, які поділяються на два класи. Перший клас складають елементарні автомати, які називають елементами пам’яті. Другий клас складають елементарні комбінаційні автомати – логічні елементи. Для зведення задачі структурного синтезу довільного автомата з пам’яттю до задачі синтезу комбінаційних схем накладають обмеження на тип елементів пам’яті. Результатом роботи методу являються рівняння булевих функцій автомата в канонічній формі подання. Необхідними даними для початку роботи методу служить абстрактний цифровий автомат з пам’яттю.

Канонічний метод структурного синтезу автомата умовно можна розділити на такі етапи:

- кодування множин вхідних сигналів, вихідних сигналів та станів абстрактного автомата;

- побудова канонічної таблиці структурного автомата;
- вибір елементів пам’яті автомата;
- отримання значень вхідних сигналів тригерів;
- побудова рівнянь булевих функцій для виходів автомата і входів тригерів;
- побудова функціональної схеми автомата.

Для правильної роботи схем сигнали на виходах запам'ятовуючих елементів не повинні безпосередньо брати участь в утворенні вихідних сигналів, що по ланцюгах зворотного зв'язку подавалися б у той же самий момент часу на ці входи. Тому запам'ятовуючими елементами повинні бути не автомати Милі, а автомати Мура. Таким чином, структурно повна система елементарних автоматів повинна містити хоча б один автомат Мура. У той же час, для синтезу автоматів з мінімальним числом елементів пам'яті, необхідно в якості таких елементів вибирати автомати Мура, що мають повну систему переходів і повну систему виходів – повні автомати.

Повнота системи переходів означає, що для будь-якої упорядкованої пари станів автомату знайдеться вхідний сигнал, що переводить перший елемент цієї пари в другий, тобто в такому автоматі в кожнім стовпці таблиці переходів повинні зустрічатися всі стани автомата.

Повнота системи виходів автомату Мура полягає в тому, що кожному станові автомата поставлений у відповідність свій особливий вихідний сигнал, відмінний від вихідних сигналів інших станів. Таким чином, у такому автоматі число вихідних сигналів дорівнює числу станів автомата. У зв'язку з цим, в автоматах пам'яті будемо використовувати ті самі позначення і для станів, і для вихідних сигналів.

Канонічний метод структурного синтезу припускає представлення структурної схеми автомата у вигляді двох частин - пам'яті та комбінаційної схеми (рис. 10.7).

Пам'ять складається з елементарних автоматів Мура $П_1$,....,$П_2$,....,$П_R$. Після вибору елементів пам'яті кожен стан синтезованого автомата A кодується набором їхніх станів. Якщо всі автомати $П_1$...,$П_R$ однакові, що в загальному випадку необов'язково, то їхнє число

$$R \geq \log_b M$$

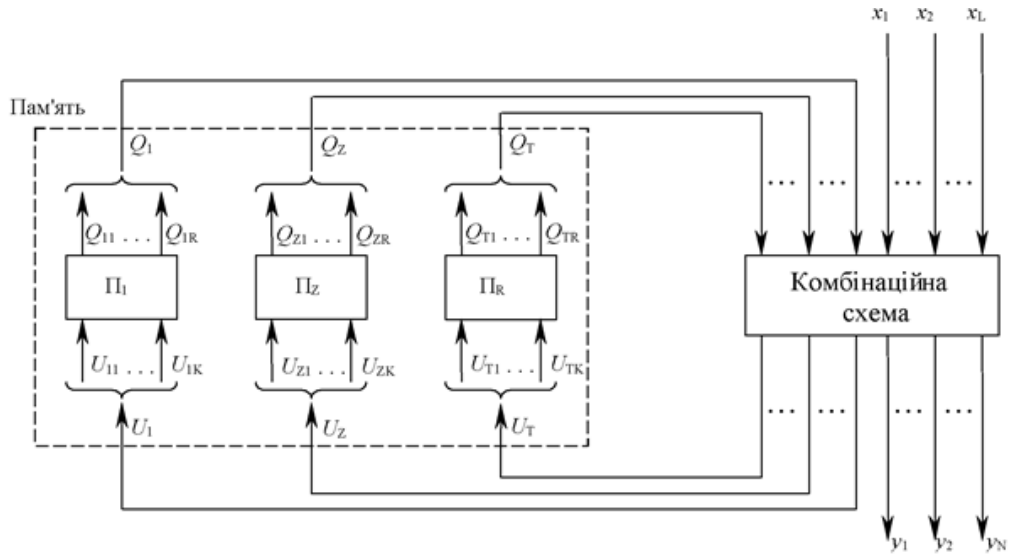


Рис. 10.7 – Структурна схема ЦА

Кодування внутрішніх станів ЦА полягає в співставленні кожному станові автомата набору (коду) станів елементів пам'яті. При цьому набори для всіх станів повинні мати однакову довжину, а різним станам автомата повинні відповідати різні набори.

Функціонування ЦА однозначно визначається, якщо встановлені зв'язки в часі між трьома його алфавітами (внутрішнім, вхідним та вихідним). Зазвичай значення цих алфавітів розділяють за часовими інтервалами, які називаються тактами. Протягом такту стан всіх трьох алфавітів автомату зберігається незмінним.

У цифрових автоматах реалізується накопичуючий спосіб обробки інформації, при якому інформація обробляється за декілька тактів. В кожному окремому такті генерується і запам'ятовується у формі відповідного стану автомату проміжний і, нарешті, в останньому такті кінцевий результат обробки інформації.

Зміни станів ЦА викликаються вхідними сигналами, які виникають поза автоматом та передаються у автомат по кінцевій кількості вхідних каналів. Щодо вхідних сигналів ЦА приймаються два припущення:

1. для будь-якого ЦА кількість різних вхідних сигналів обов'язково кінцева;
2. вхідні сигнали розглядаються як причина переходу автомата з одного стану в інший та належать до моментів часу, які визначаються відповідними їм переходами. Результатом роботи ЦА є видача вихідних сигналів, які передаються з автомату назовні по вихідних каналах.

Щодо вихідних сигналів ЦА вводяться припущення, аналогічні припущенням щодо вхідних сигналів:

1. кількість різних вихідних сигналів для будь-якого ЦА завжди кінцева;
2. кожному відмінному від нуля моменту автоматного часу належить відповідний йому вихідний сигнал.

Логічні схеми ЦА представляються у вигляді схеми з'єднання логічних елементів, які виконують певні операції. При цьому для полегшення роботи зі схемами в них нехтують ланцюгами живлення та іншими характерними з'єднаннями, відсутність яких не призводить до зміни логіки роботи схеми. Структурна схема ЦА містить комбінаційні схеми та запам'ятовуючі елементи.

Структура цифрового автомата (рис. 10.7) може бути використана для створення алгоритму проектування цифрового автомата, суть якого полягає у виконанні такої послідовності дій:

1. Вибираються необхідні для реалізації елементи пам’яті.
2. Визначаються режими роботи елементів пам'яті й необхідні для цього входи.
3. Складаються функції збудження (логічні вирази) для входів елементів пам’яті, що використовуються.
4. Вибирається елементна база для комбінаційної схеми елементів пам’яті.
5. Мінімізуються логічні вирази.
6. Здійснюється перехід на елементну базу й реалізується комбінаційна схема для елементів пам’яті.
7. Складаються функції збудження для вихідних сигналів цифрового автомата.
8. Вибирається елементна база для комбінаційної схеми вихідних сигналів.
9. Мінімізуються логічні вирази.
10. Здійснюється перехід на елементну базу й реалізується комбінаційна схема для вихідних сигналів.

Контрольні питання:

1. Дайте визначення цифрового автомату. Що входить до структури ЕОМ?
2. Що розуміють під математичною моделлю ЦА?
3. Що таке скінченний автомат з виходом і без, з чого він кладається?
4. З яких частин складається цифрової автомат?
5. Дайте визначення поняттю «алгоритм».
6. На які два основні класи можна розділити цифрові автомати?
7. У чому відмінність абстрактної та структурної теорії автоматів?
8. У чому особливість абстрактної теорії автоматів?
9. Якими системами рівнянь описуються автомати Мілі та Мура?
10. Назвіть і опишіть способи подання абстрактних автоматів.
11. Що показує функція переходів автомата?
12. Назвіть два класи елементарних автоматів.
13. Які етапи канонічного методу структурного синтезу автомата ви знаєте?
14. Як будується основна таблиця абстрактного автомата?
15. Як кодуються вхідні і вихідні сигнали та внутрішні стани автомата?
16. Як будується закодована таблиця структурного автомата?
17. Як будується канонічна таблиця структурного автомата?

ТЕМА 11. СИНХРОНІЗАЦІЯ І ЗАБЕЗПЕЧЕННЯ СТІЙКОСТІ ЦИФРОВОГО АВТОМАТА

План:

1. Явище «гонок» та методи забезпечення стійкості цифрового автомату.
2. Синхронізація роботи цифрового автомату.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Явище «гонок» та методи забезпечення стійкості цифрового автомату.

Завдання кодування станів є однією з основних задач канонічного методу структурного синтезу автоматів. Перехід автомата зі стану a_m в стан a_s супроводжується перемиканням деяких (можливо, всіх) елементів пам'яті.

Якщо при переході автомата з будь-якого стану в суміжне має відбутися перемикання більш ніж одного елемента пам'яті, то виникає ситуація, названа змаганнями елементів пам'яті. При цьому через технологічну неідентичність тригерів час перемикання їх виявляється різним. Під впливом зовнішнього сигналу першим переключиться тригер, час затримки якого виявиться мінімальним. Потім відбудеться перемикання наступного тригера і т.п. Крім того, різні часи поширення сигналів збудження по ланцюгах різної довжини. Той елемент, який виграє змагання, тобто змінить свій стан раніше інших елементів, може через ланцюг зворотного зв'язку змінювати сигнали на входах деяких запам'ятовуючих елементів до того, як інші елементи пам'яті, що беруть участь в змаганнях, змінять свій стан. Розглянуте явище називається «гонками» або «змаганнями».

Це може привести автомат в стан, не передбачений його графом. Тому в процесі переходу автомат може виявитися в деякому проміжному стані a_k або a_j в залежності від того, який елемент пам'яті виграє змагання. Якщо потім при тому ж вхідному сигналі автомат перейде в стан a_s, то такі змагання називаються некритичними. Якщо ж автомат перейде в стан, не передбачене його ДСА, то такі змагання називаються критичними гонками. Якщо при незмінному вхідному сигналі з отриманих перехідних нестійких станів можливий перехід в інші стани, що не відповідають бажаному переключенню, виникає неправильна робота автомата, тобто збій. Така ситуація називається критичними змаганнями елементів пам’яті. В будь-якому випадку гонки - явище небажане і їх слід уникати. Приклад критичних і некритичних змагань елементів пам'яті наведено на рис.11.1.

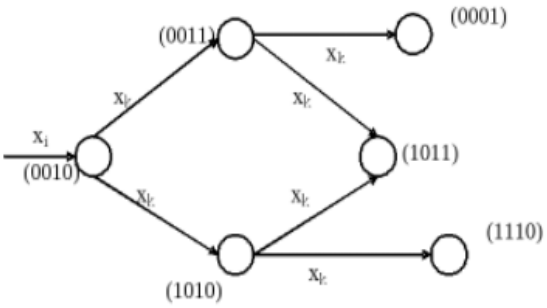


Рис.11.1 – Гонки в цифровому автоматі

Якщо після впливу на вхід автомата сигналу x_i автомат потрапляє в стан, який має код (0010), а потім на вхід його надходить сигнал x_k, який повинен перевести автомат в стан з кодом (1011), то під час цього переходу повинні переключитися два елементи пам'яті. При цьому можливі проміжні перемикання в стан з кодами (1010) або (0011), що не входять в програму роботи автомата.

Уникнути критичних змагань елементів пам'яті можна, кодуючи стани таким чином, щоб всі переходи відбувалися між станами, закодованими сусідніми кодами.

Кодування суміжних станів сусідніми кодами називається протигоночним кодуванням. Такі коди, як правило, надлишкові. Крім того, для таких кодів існують обмеження:

- неможливе використання протигоночного кодування при наявності в графі автомата циклів непарної довжини;
- якщо пара сусідніх вершин другого порядку має більше двох проміжних станів, сусіднє кодування також неможливо.
- існують такі випадки, коли таке кодування неможливо без введення додаткових позначок в ДСА.

Тому такий метод боротьби з гонками практично не застосовується.

Головна проблема, яка пов’язана з гонками, полягає у тому, що проектант не має реальних можливостей визначити момент появи можливої завади та її тривалість.

Гонки називаються критичними або недопустимими, якщо хоча б один вихідний сигнал під час перехідного процесу змінюється більш ніж один раз.

Критичні змагання суттєво впливають на роботу цифрових пристроїв та їх проектування.

На практиці використовуються три наступні способи боротьби з гонками:

- синхронізація;
- побудова протигоночних систем;
- врахування мінімального часу затримки.

З підвищенням складності цифрових схем поява місцевих та загальних зв’язків у них призводить до того, що аналізувати і враховувати гонки в таких схемах стає практично неможливо. Радикальним вирішенням проблеми гонок є синхронізація.

Протигоночні системи – це другий практичний спосіб боротьби з гонками.

Вони будуються таким чином, що в них відсутній ризик появи на виході сигналів, не передбачених логікою роботи схеми. Прикладом подвійної схеми може бути два паралельних канали з однаковою кількістю елементів, об’єднаних елементом АБО; який би канал не виграв гонку, результат буде однаковим, зміниться лише момент його появи.

Третій спосіб – урахування мінімального часу затримок – знаходить використання при проектуванні таких цифрових схем, які потім виготовлятимуться на одному кристалі. Це пов’язано з тим, що проектант паралельно зі схемотехнікою закладає й технологічні особливості виготовлення схеми з необхідними параметрами. В результаті у проект можна закласти необхідні часові затримки в окремих каналах або схемах. Використання таких прийомів у дискретній цифровій схемотехніці вимагає великого досвіду і уваги.

Один із способів ліквідації гонок є тактування вхідних сигналів автомата імпульсами певної тривалості. Якщо тривалість тактового сигналу менше найкоротшого шляху проходження тактованого сигналу зворотного зв'язку по комбінаційній схемі, то до моменту переходу в проміжний стан тактуючий сигнал дорівнює 0 і, отже, не впливає на формування функцій збудження. Це виключає гонки.

Інший спосіб ліквідації гонок полягає у використанні двотактної пам'яті. У цьому випадку кожен елемент пам'яті дублюється, причому перемикання другого шару тригерів відбувається по відсутності тактуючого сигналу. Сигнали зворотного зв'язку знімаються з другого шару тригерів, а надходження сигналів збудження на входи тригерів і сигналів зворотного зв'язку на входи схеми виявляються рознесеними в часі, що і усуває причину гонок.

У будь-якому випадку, після проектування цифрових схем їх аналізують на предмет можливої появи завад в результаті гонок. На практиці з цією метою використовуються спеціальні комп’ютерні програми.

Ще одним прикладом появи завад завдяки гонкам є прийом сигналів з недостатньо крутими фронтами. Така ситуація має місце тоді, коли подвійний сигнал подається на паралельні канали з різними типами мікросхем. У такому випадку в деякому інтервалі рівнів вхідного сигналу для мікросхеми одного типу сигнал відповідатиме рівню логічного нуля, а для іншого – логічної одиниці. Це приведе до появи хибних сигналів. Це явище називається «гонками по виходу». Для боротьби з ним використовуються спеціальні прийоми попередньої обробки вхідних сигналів.

2. Синхронізація роботи цифрового автомату.

Синхронізація є найбільш універсальним засобом боротьби з гонками. Її суть полягає у наступному: по всьому цифровому пристрою створюється єдина система синхронізуючих сигналів. У практиці побудови систем синхронізації використовуються У залежності від типу елементної бази: однофазна і багатофазна синхронізація, одночастотна і багаточастотна.

Слід зазначити, що синхронізація дає можливість суттєво спростити процес проектування цифрових схем, адже значно спрощує вирішення проблеми гонок.

Розгляд систем синхронізації почнемо з двофазної системи, коли всі схеми синхронізуються двома послідовностями імпульсів C_1 та C_2 однієї частоти $f_T = T_T^{-1}$ та одного фазового зсуву $T_Ф$. Тривалість імпульсів двох послідовностей однакова і дорівнює T_i. Для симетричної двофазної синхронізації $T_T = 2T_\phi$. Для несиметричної $T_{\phi 1} \neq T_{\phi 2}$.

При побудові синхронних цифрових схем їх розподіляють на дві групи.

В одну групу входять комбінаційні схеми з визначеною кількістю входів та виходів. До іншої групи входять схеми D-тригерів, які мають особливість зберігати записану інформацію протягом одного такту.

Узагальнена цифрова схема може бути приведена до структури, що зображена на рис. 11.2 (аналог конвеєрної обробки інформації в мікропроцесорах і мікропроцесорних системах).

Схема включає в себе послідовно об'єднані групи D-тригерів, позначені на рис. 11.2 як DI, DII, …, DN, і комбінаційних схем, позначені КСІ, КСІІ, …, КС(N–1), КСN. Кожна з груп тригерів об'єднується за принципом синхронізації від одного синхроімпульсу і в загальному плані представляє собою паралельний регістр, виконаний на D-тригерах.

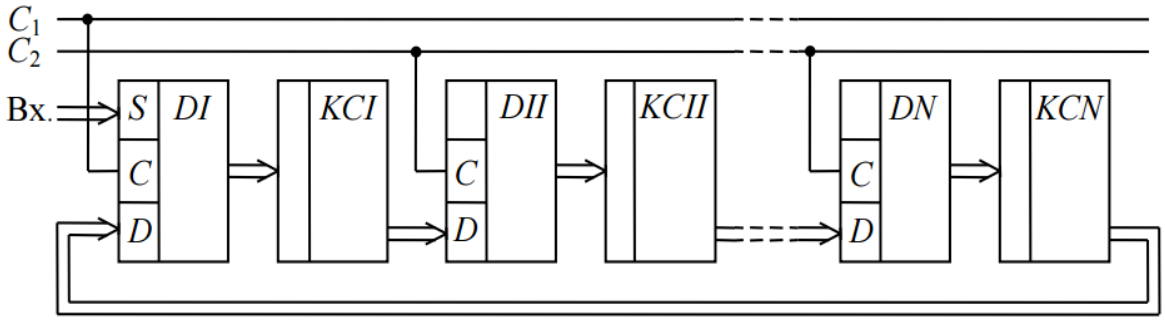


Рис.11.2 – Структура узагальненої цифрової схеми

Кожна група КС включає в себе чисто комбінаційну схемотехніку, яка виконує одночасно ряд логічних функцій, приймаючи інформацію з виходів попереднього регістра пам'яті і передаючи на наступний, який синхронізується другим синхроімпульсом. Внутрішні зворотні зв'язки в групі комбінаційних схем відсутні.

Фізичну суть процесів у схемі та ідеологію проектування цифрових пристроїв з двофазною синхронізацією пояснює рис. 11.3.

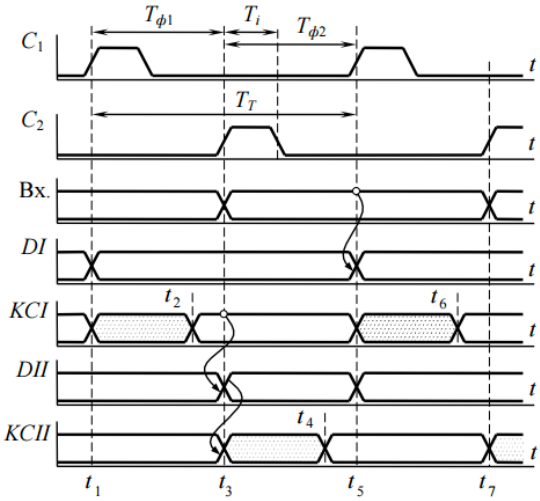


Рис.11.3 – Часові діаграми

За синхросигналом C_2 чергова група інформаційних сигналів записується в регістрову групу D-тригерів *N* і передається для обробки на вхід комбінаційної схеми KCN. Через деякий інтервал часу ця інформація з'явиться на вхідних шинах D-тригерів DI. У момент часу t_1 з'являється фронт синхросигналу C_1, за яким як вхідна інформація, що подається на входи S, так і інформація з виходів KCN записуються в тригери DI. По закінченні перехідних процесів в тригерах DI на їх виходах з'являються сигнали, які починають опрацьовуватись комбінаційною схемою KCI. Як правило, в схемі мають місце паралельні шляхи розповсюдження сигналів, тому вихідні сигнали KCI спочатку є невизначеними, адже вони спотворюються перехідними процесами.

На рис. 11.3 картина гонок в KCI відображена на інтервалі часу $t_1 – t_2$. Інтервал часу $t_1 – t_2$ перехідних процесів для DII не є безпечним через те, що всі тригери в цьому інтервалі часу закриті нульовим рівнем сигналу C_2. До моменту t_2 всі перехідні процеси закінчуються, сигнали на виході KCI фіксуються, і в інтервалі часу $t_2 – t_3$ ніякі стани в схемі не змінюються.

При подачі сигналу C_2 у момент t_3 встановлені значення виходів KCI записуються в DII і по завершенні в них перехідних процесів подаються на входи наступної комбінаційної схеми KCII. Процеси гонок в KCII проходять в інтервалі часу $t_3 … t_4$ і до моменту t_5 появи фронту синхроімпульсу C_1 встановлюються незмінними. При появі C_1 результати обробки сигналів в KCII перезаписуються в наступні регістрові схеми. Як результат, у синхронному пристрої йде циклічна багатоступенева обробка інформації в комбінаційних схемах, при якій комбінаційні схеми працюють по черзі. Завдяки цьому ніякі гоночні процеси в комбінаційних схемах не можуть внести похибку в обробку вхідних сигналів. Для цього необхідно лише, щоб інтервал часу $T_Ф$ перевершував максимальну тривалість перехідних процесів. Проектант завжди в змозі забезпечити таке співвідношення на основі паспортних значень максимальних затримок мікросхем.

Величина $T_Ф$ залежить від величини затримки t_3 комбінаційних схем, яка може змінюватись у широких межах. Якщо t_3 менша вибраної величини $T_Ф$, то

таке співвідношення не має негативних наслідків, виключаючи лише зниження швидкості обробки інформації. Але якщо затримка деяких комбінаційних схем перевищує величину робочого інтервалу $T_Ф$, відповідно до рис. 11.3, схема стає непрацездатною. У подібних ситуаціях можуть використовуватись різні шляхи вирішення проблеми.

Найпростішим з них є збільшення тривалості $T_Ф$ і, відповідно, періоду синхроімпульсів. Як результат, це може суттєво знизити швидкодію розробленої схеми. Для того, щоб залишити частоту синхронізації незмінною, використовують несиметричну двофазну синхронізацію, при якій $T_{ф1} \neq T_{ф2}$.

У цьому випадку, якщо можливо, комбінаційні схеми з більшим часом затримки розміщуються в більшому робочому інтервалі. Якщо подібна організація схемотехніки неможлива, то комбінаційну схему з великою тривалістю $t_З$ розбивають на дві схеми і між ними встановлюють проміжний запам'ятовуючий вузол. Такий спосіб приводить до необхідної наступної перефазовки схеми. Широко використовується спосіб, при якому комбінаційні вузли з низькою швидкодією виділяють окремо і для них знижують частоту синхронізації до необхідної.

Найбільш гнучкий спосіб забезпечення високої швидкодії при наявності комбінаційних схем з великою затримкою – це використання багатофазних схем синхронізації, які використовуються у швидкодіючих пристроях. Переваги таких схем ілюструє рис. 11.4.

У залежності від величини конкретної затримки кожної комбінаційної схеми, на С-входи пристроїв пам'яті можливо заводити різні фази синхронізації і, відповідно, відкривати тригери-приймачі з затримкою на інтервали часу, кратні $T_Ф$ ($T_Ф$, $2T_Ф$, $3T_Ф$, …) відносно тієї фази, яка синхронізує передавач інформації. Недопустимо тільки синхронізувати тригери-приймачі синхросигналом тієї фази, якою синхронізувалися тригери-передавачі даної комбінаційної схеми. Розглянутий спосіб широко використовується на практиці, адже він дає також можливість зменшити неробочі інтервали комбінаційних схем, що мають місце при очікуванні синхросигналу.

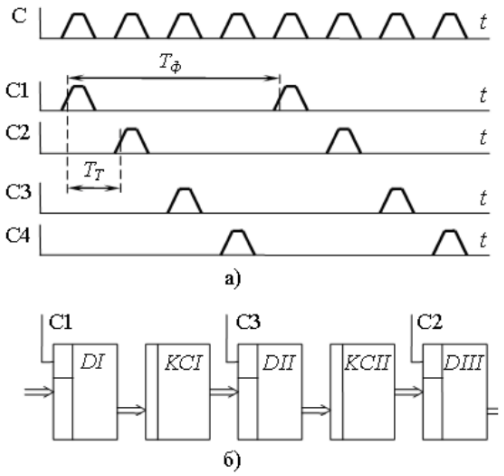


Рис.11.4 – Багатофазна схема синхронізації та часові діаграми

На вибір тактової частоти генератора синхросигналів впливають також типи тригерів, розгалуженість схеми розподілення синхросигналів. Вказані особливості використовуються тільки досвідченими конструкторами при проектуванні складних цифрових схем автоматики. З розвитком потужних мікропроцесорів та машинних методів проектування ці особливості стають неактуальними.

У процесі проектування часто виникає необхідність у створенні зворотних зв'язків у синхронних схемах. У таких випадках необхідно дотримуватись наступних правил:

в усіх схемах з двофазною синхронізацією петля зворотного зв'язку як з логічними елементами, так і без них повинна починатись з виходів тригерів, що синхронізуються однією фазою і закінчуватись на вході тригерів, що синхронізуються іншою фазою;

недопустимі зв'язки, які передають сигнали з виходу однієї групи тригерів на вхід іншої, що синхронізується однією і тією ж фазою.

З вказаних правил витікає, що відсутність у схемах з двофазною синхронізацією замкнутих кіл забезпечується тим, що у будь-який момент часу хоча б один з запам'ятовуючих пристроїв є відключеним і не передає інформацію з входу на вихід. Ці умови можуть бути забезпечені і при використанні однофазної синхронізації, якщо використовувати тригери, які не є – прозорими для інформаційного сигналу, – наприклад, динамічні тригери.

Особливістю однофазної синхронізації є складність її використання при розгалуженій системі синхронізації. Пояснюється це тим, що на окремих ділянках схеми синхронізації можуть виникати суттєві затримки. Інформаційні сигнали для ланки схеми, що розглядається, можуть не мати затримок.

Як результат цього явища, неспівпадіння інформаційних тактів з фронтами синхросигналу – наприклад, N-го такту синхросигналу з (N+1) тактом інформаційного сигналу. Проконтролювати подібну ситуацію не завжди можливо, тому однофазна синхронізація находить обмежене використання.

Двофазна і багатофазна синхронізація не має вказаного недоліку через те, що вона має можливість попередньо врахувати будь-які затримки як в передачі синхронізуючих, так і інформаційних сигналів.

Важливо звернути увагу і на інші переваги багатофазної синхронізації.

Перш за все, модулі пам'яті в багатофазних схемах синхронізації можуть бути побудовані на найпростіших синхронних тригерах, а принципових обмежень на типи тригерів практично немає. Немає обмежень також на часові співвідношення в імпульсних послідовностях синхросигналу або крутизну фронтів, що є обов'язковими для динамічних тригерів. Вказані переваги багатофазних схем синхронізації, незважаючи на складність побудови розгалуженого дерева синхронізуючих сигналів, приводять до того, що в складних цифрових схемах використовуються переважно вони. Однофазні схеми знаходять використання лише в окремих вузлах або нескладних схемах – регістрах, лічильниках і т. п. Часто однофазна синхронізація використовується в

мікроконтролерах, в яких немає необхідності багатоступінчатого розмноження сигналів.

Прив’язка зовнішніх сигналів до синхроімпульсів необхідна тому, що синхронні цифрові схеми приймають вхідні сигнали без похибок лише в визначені інтервали часу. Якщо вхідний сигнал подається на комбінаційну схему безпосередньо перед синхроімпульсом, то перехідні процеси в ній можуть не завершитись до появи синхроімпульсу, і в тригери буде записана хибна інформація. Інша причина пов’язана з реакцією на одиночні сигнали.

Наприклад, сигнал від натискання кнопки може тривати багато періодів синхросигналу. У той же час, для цифрових синхронних схем його тривалість не повинна перевищувати один період синхрочастоти. Вказані задачі вирішуються за допомогою тригерних схем, які називаються синхронізаторами.

Асинхронний обмін інформацією має місце між цифровими пристроями, кожен з яких має свою власну схему синхронізації. В такому випадку сигнали, що поступають з іншого пристрою, сприймаються приймачем як асинхронні.

При інтенсивному обміні інформацією постає питання максимально можливої частоти передачі при асинхронному зв’язку.

Окрім синхросигналів, у цифрових системах існує ще ряд сигналів, на які слід звертати особливу увагу при проектуванні. Це можуть бути сигнали дозволу, які необхідно подавати до подачі синхросигналу, а також інші керуючі сигнали. Слід також звертати особливу увагу на введення у цифрову систему асинхронних сигналів. Асинхронними є сигнали вводу інформації (наприклад, з клавіатури), сигнали переривань, а також ряд внутрішніх сигналів, що з’являються в результаті виконання обчислень (ознаки). Зрозуміло, що для вводу цих сигналів у синхронну цифрову систему використовуються синхронізатори, які забезпечують вибір асинхронного сигналу в тактовий момент часу. Проблема синхронізації обумовлена розглянутими вище затримками синхросигналу. Якщо, наприклад, асинхронний сигнал одночасно подається на декілька тригерів, то можлива ситуація, коли в один з тригерів інформація буде записана, а в інший, внаслідок затримки синхросигналу, – ні. В результаті в роботі системи виникне помилка.

Контрольні питання:

1. Наведіть приклади метастабільності, які мають місце у повсякденному житті.
2. У чому проявляється необхідність врахування затримок і перехідних процесів при проектуванні цифрових пристроїв?
3. Що називається «змаганнями» («гонками»)?
4. Які «гонки» називаються критичними?
5. Які засоби боротьби з гонками використовуються в цифровій схемотехніці?
6. Сформулюйте правило мінімізації логічних функцій для забезпечення мінімальної диз'юнктивної форми, вільної від гонок.

7. Поясніть призначення синхронізації у цифрових системах високого рівня складності.
8. Які види синхронізації використовуються у цифрових системах?
9. Які шляхи і засоби використовуються для вирішення проблеми затримок у складних цифрових системах?
10. У чому полягають переваги багатофазних способів синхронізації перед однофазним?
11. Які вимоги до вхідних сигналів пред'являються цифровими системами?
12. У чому проявляється необхідність врахування затримок і перехідних процесів при проектуванні цифрових пристроїв?

ТЕМА 12. КЕРУЮЧИЙ І ОПЕРАЦІЙНИЙ БЛОКИ АВТОМАТА

План:

1. Принцип мікропрограмного керування.
2. Поняття операційних та керуючих автоматів (ОА і КА).
3. Способи опису алгоритмів і мікропрограм. Граф-схема алгоритму (ГСА).
4. Синтез мікропрограмних автоматів (Мілі та Мура) за граф-схемою алгоритму.
5. Структурний синтез мікропрограмних автоматів.

Завдання для самостійної роботи

І. Ознайомтесь з навчальними матеріалами.
ІІ. Дайте відповідь на запитання.
ІІІ. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Принцип мікропрограмного керування

Головною метою синтезу цифрового автомата з пам’яттю є визначення всіх його можливих станів та переходів, відповідно заданому алгоритму функціонування, та отримання функцій збудження всіх входів тригерів, з яких складається автомат. Цього достатньо для складання логічної схеми цифрового автомата.

Поряд з теорією формальних мов і заснованих на ній методів побудови компіляторів кінцеві автомати активно використовуються в комп'ютерних іграх, у реалізації мережних протоколів, системах стиску інформації. Іншими словами, там, де потрібна велика надійність і де логіка поводження надто складна, щоб програміст зміг реалізувати її на одному лише рівні здорового глузду.

З появою структурного програмування стало очевидним, що з трьох головних конструкторів керування (проходження, циклу і розгалуження) останній є найважчим у сприйнятті програміста, оскільки при безлічі альтернатив перетворює лінійну структуру алгоритму в деревоподібну. При цьому складність навіть послідовних програм росте стрімко і часом може перевершувати дерево варіантів у настільки непросту для автоматичного аналізу модель, як традиційні шахи. Розбивка програми на процеси й об'єкти з заміною багатоступінчастого розгалуження засобами обробки повідомлень (подій) заміняє одну проблему на іншу: вкладеність зменшується, зате кількість взаємодіючих компонентів помітно зростає. Логіка «розмивається» і в підсумку ми одержуємо погано контрольовану ситуацію, коли через хаотичність «ручного» синтезу і неможливості побудувати вичерпний набір тестів немає ніякої впевненості в коректності побудованої системи. Ключ до рішення

знаходиться взастосуванні формальних методів, у створенні зручної абстракції, здатної «вичавити» з алгоритму квінтесенцію логіки його роботи і дати можливість проводити весь необхідний аналіз. Однією з таких зручних абстракцій можуть служити кінцеві автомати, серед різновидів яких варто виділити автомати Мілі і Мура. Близькість до булевої алгебри і теорії графів, наочність графічного представлення і детермінованість поводження є помітними перевагами цієї абстракції.

ЕОМ переробляє інформацію, виконуючи над нею якісь операції. Для виконання операцій над інформацією використовуються операційні пристрої – процесори, канали вводу-виводу, вузли керування зовнішніми пристроями і т.д. Функцією операційного пристрою є виконання заданої множини операцій F={f1,...,fG} над вхідними словами D={d1,...,dH} з метою обчислення слів R={r1,...,rQ}, що представляють результати операцій R=fg(D), де g=1,2,...,G.

Функціональна і структурна організація операційних пристроїв базується на принципі мікропрограмного керування, що полягає в наступному:

1. Будь-яка операція fg(g=1,...,G), реалізована пристроєм, розглядається як складна дія, що розділяється на послідовність елементарних дій над словами інформації. Ці елементарні дії називаються мікрооперaціями (передача інформації з одного регістра в інший, взяття оберненого коду, зсув і т.і).

2. Для керування порядком проходження мікрооперацій використовуються логічні умови, що у залежності від значень слів, перетворених мікрооперaціями, приймають значення «неправда» або «істина» (0 або1).

3. Процес виконання операцій у пристрої описується у формі алгоритму, що представляється в термінах мікрооперацій і логічних умов і називається мікропрограмою. Мікропрограма визначає порядок перевірки значень логічних умов і проходження мікрооперацій, необхідний для одержання результатів, які вимагаються.

4. Мікропрограма використовується як форма представлення функції пристрою, на основі якої визначається структура і порядок функціонування пристрою в часі..

Таким чином, із принципу мікропрограмного керування випливає, що структура і порядок функціонування операційних пристроїв визначається алгоритмом виконання операції

F={f1,...,fG}.

2. Поняття операційних та керуючих автоматів (ОА і КА).

Як показав академік В.М. Глушков, у будь-якому пристрої обробки цифрової інформації можна виділити два основних блоки – операційний автомат (ОА) і керуючий автомат (КА).

Операційний автомат (ОА) служить для збереження слів інформації, виконання набору мікрооперацій і обчислення значень логічних умов, тобто ОА є структурою, організованою для виконання дій над інформацією. Мікрооперації, виконувані ОА, задаються множиною керуючих сигналів

Y={y1,....,yM}, з кожним з яких ототожнюється визначена мікрооперація (рис. 12.1).

Значення логічних умов, що обчислюються в операційному автоматі, відображаються множиною інформаційних сигналів X={x1,...,xL}, кожний з яких ототожнюється з визначеною логічною умовою.

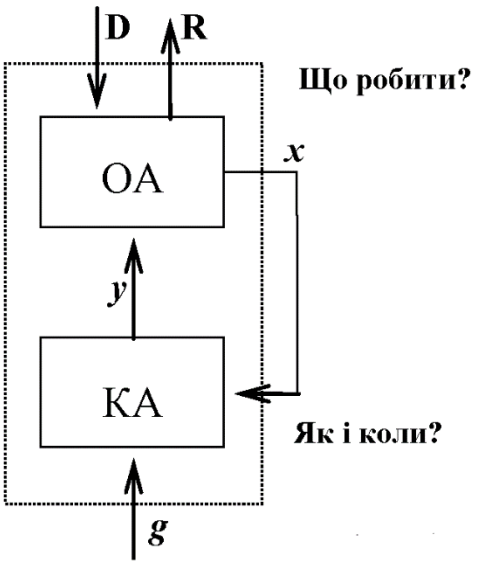


Рис.12.1 – Операційний автомат

Керуючий автомат (КА) генерує запропоновану мікропрограмою послідовність керуючих сигналів, відповідно до значень логічних умов. Інакше кажучи, КА задає порядок виконання дій в ОА згідно алгоритму виконання операцій. Найменування операції, яку необхідно виконати в пристрої, визначається кодом g операції, що надходить в КА ззовні.

Стосовно КА сигнали gi,...,gh, за допомогою яких кодується найменування операції, та повідомляючі сигнали хь...,хь, формовані в ОА, відіграють однакову роль: вони впливають на порядок вироблення керуючих сигналів Y. Тому сигнали gi,...,gh і x\,...jcl належать до одного класу - до класу повідомляючих сигналів, що надходять на вхід КА.

Таким чином, будь-який операційний пристрій - процесор, канал вводу-виводу і т.д. - є композицією операційного і керуючого автоматів. ОА, реалізуючи дії над словами інформації, є виконавчою частиною пристрою, роботою якого керує КА, що генерує необхідні послідовності керуючих сигналів.

ОА і КА можуть бути визначені своїми функціями - переліком виконуваних ними дій.

Функція ОА визначається наступною сукупністю відомостей:

1) множиною вхідних слів D={db...,dH}, що вводяться в автомат у якості операндів;

2) множиною вихідних слів R={n,...,rQ}, що представляють результати операцій;

3) множиною внутрішніх слів S={si,...,sn}, використовуваних для

представлення інформації в процесі виконання операцій. Можна вважати, що вхідні і вихідні слова збігаються з визначеними внутрішніми DeS, ReS.

4) множиною мікрооперацій Y={ym}, що реалізують перетворення S=qm(s) над словами інформації, де фm - функція, що обчислюється;

5) множиною логічних умов X={pa}, де xi=#si) і булева функція;

Таким чином, функція ОА задана, якщо задані (визначені) множини D R, S, Y, X Час не є аргументом функції ОА! Функція встановлює список дій - мікрооперацій і логічних умов, що може виконувати автомат, але ніяк не визначає порядок проходження цих дій у часі. Тобто, функція ОА характеризує засоби, що можуть бути використані для обчислень, але не сам обчислювальний процес.

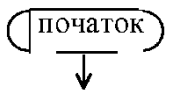
Порядок виконання дій у часі визначається у формі функцій керуючого автомата.

Функція керуючого автомата - це операторна схема алгоритму (мікропрограми), функціональними операторами якої є символи уі,...ут що ототожнюються з мікрооперaціями, і логічними умовами, в якості яких використовуються булеві змінні x_1...,x_n.

Операторна схема алгоритму найбільш часто представляється у вигляді граф-схеми алгоритму (ГСА). ГСА визначає обчислювальний процес послідовно у часі, встановлюючи порядок перевірки логічних умов хх– xL і порядок проходження мікрооперацій ух–ут.

3. Способи опису алгоритмів і мікропрограм. Граф-схема алгоритму (ГСА).

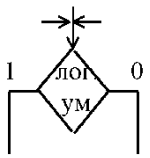
Найбільш наочно зображувати мікропрограми та алгоритми у вигляді орієнтованого графа, так званої **граф-схеми алгоритму (ГСА)**. Крім наочності, це дає можливість використовувати для аналізу і перетворення мікропрограм ефективні методи теорії графів. При графічному описі окремі функції алгоритмів (мікрооперації) відображаються у вигляді умовних графічних зображень, так званих вершин. У ГСА зазвичай використовують **вершини наступних типів**:



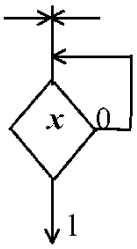
- вершина *«початок»* має один вихід, входів не має. Позначає початок мікропрограми;

- вершина *«кінець»* має будь-яке число входів, виходів не має. Позначає кінець мікропрограми;

- *операторна вершина* має будь-яке число входів, один вихід. Всередині операторної вершини записується одна мікрокоманда – сукупніст мікрооперацій з одночасним виконанням;



- *умовна вершина* має будь-яку кількість входів і два виходи. Всередині умовної вершини записується булевий вираз, в залежності від значення якого здійснюється вибір напрямку подальшого виконання мікропрограми;



- особливий вид умовної вершини, *вершина очікування,* має множину входів, два виходи, один з яких заведений на вхід. При влученні у вершину очікування, вихід з неї можливий тільки при виконанні умови ***X***.

Граф мікропрограми складається із сукупності перерахованих вершин і дуг, що з'єднують виходи одних вершин зі входами інших. З'єднання вершин і напрямки дуг графа визначають, виходячи з алгоритму операції, описуваного графом, і структури ОА.

Сама мікропрограма та її граф повинні бути коректними, тобто відповідати наступним умовам:

- У графі повинна бути тільки одна початкова і одна кінцева вершини.
- У будь-яку вершину графа повинен вести принаймні один шлях з початкової вершини.
- З кожного виходу будь-якої вершини графа повинен існувати принаймні один шлях у кінцеву вершину.
- При всіх можливих значеннях логічних умов і використовуваних слів повинен

  існувати шлях з початкової вершини в кінцеву.

Приклад ГСА представлений на рисунку 12.2.

ГСА на рис. 12.2 називається **змістовною**, тому що всередині вершин записані в явному вигляді мікрооперації та логічні умови. Якщо ж кожну мікрооперацію позначити символами ***y***, а логічні умови через ***x***, то вийде так звана **кодована ГСА** (рис. 12.3).

Для правильного сприйняття мікропрограми, заданої у вигляді кодованої ГСА, необхідно знати відповідності між $y_h x_t$ та змістом відповідних мікрооперацій і логічних умов.

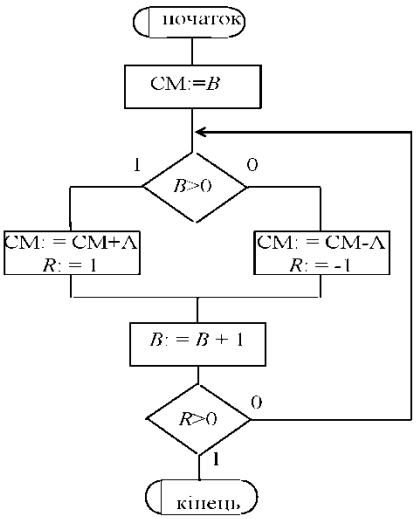


Рис.12.2 – Змістовна ГСА

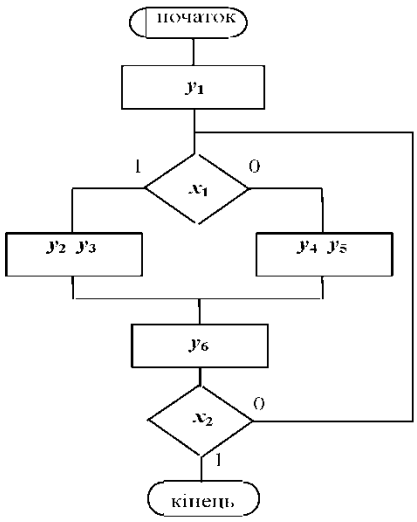


Рис.12.3 – Кодована ГСА

Для запису мікрооперацій всередині вершин використовується так звана Ф-мова (або **мова функціонального мікропрограмування**). Докладно з цією мовою знайомляться в курсах «Комп'ютерна схемотехніка», «Теорія і проектування ЕОМ». У дисципліні «Прикладна теорія цифрових автоматів» ми розглядаємо тільки основні положення цієї мови.

У цій мові існують двійкові константи і змінні: **0010** - константа, ***A(1:4)*** - чотирьохрозрядне слово - чотирьохрозрядна двійкова змінна. Наприклад, **Л(1:4)=1010** означає, що в першому розряді слова ***A*** буде **1**, у другому - **0** і т.д. ***A(2:3)*** - частина слова ***A***, розміщена в другому і третьому розрядах.

Найбільш вживані операції Ф-мови:

присвоювання - ***A(*** **0:3): = 1000,** ***B(*** **1:4): =** ***A(*** **5:8)** і т.д.;

інвертування - ***A(*** **0:3): = ^** ***B(*** **1:4);**

конкатенації - ***C*****(0:6): =** ***A(*** **0:3) . Б(1:3).**

Приклад 12.1.

1. *A*(0:3): = 1100; *B*(1:4): = *A*(0:3)->*B*(1:4): = 1100;
2. *B*(1:4): = 0101; *A*(0:3): = ^ *B*(1:4)->*A*(0:3): = 1010;
3. *A*(0:3): = 1101; *B*(1:3): = 110; *C*(0:6): = *A*(0:3). *B*(1:3); *C*(0:6):=1101110

Запис виду ***A(2)*** означає, що тут розглядається другий розряд слова ***A***, тобто це біт, записаний у другому розряді слова ***A***

При виконанні операцій присвоювання необхідно дотримуватись рівності розрядів у словах ліворуч і праворуч від знака присвоювання. Операції додавання, логічного додавання і т.д. у Ф-мові записуються звичайним способом через оператор присвоювання:

C(0:n):=A(0:n)+B(0:n)

D(0:n):=A(0:n)vB(0:n) і т.д.

4. Синтез мікропрограмних автоматів за граф-схемою алгоритму

Граф-схема алгоритму є формою представлення мікропрограми, яку повинен виконати **операційний пристрій** (ОП). При побудові операційного пристрою, що складається з операційного (ОА) і керуючого (КА) автоматів, необхідно уміти виділити функції ОА і КА з ГСА. Зазвичай мікропрограма представляється у вигляді змістовної ГСА. У цьому випадку для задання функцій ОА необхідно перерахувати усі виконувані мікрооперації і всі логічні умови даної мікропрограми, а також описати розрядність слів, оброблюваних ОП. На підставі цих даних можна побудувати ОА методами, що будуть викладені в наступному курсі «Комп'ютерна схемотехніка». Для ініціалізації виконання тієї або іншої мікрооперації на ОА повинні надходити в потрібний згідно ГСА момент часу керуючі сигнали *Y*. Звичайно при проектуванні ОП приймають визначений спосіб кодування мікрооперацій (найчастіше кодом, що містить стільки розрядів, скільки усього різних мікрооперацій) і для розробки ОА вважають, що КА видає коди мікрооперацій, які виконуються в даний момент часу.

Для КА важлива послідовність видачі відповідних кодів мікрооперацій у залежності від логічних умов, вироблюваних ОА й аналізованих КА в потрібні моменти часу. Якщо прийнято спосіб кодування мікрооперацій, то функції КА задаються кодованою ГСА. Тому для різних змістовних ГСА, що мають однакову кодовану ГСА, ОА будуть різні але КА буде тим самим.

Надалі будемо розглядати синтез тільки КА і тільки кодованої ГСА.

Кінцевий автомат, що інтерпретує мікропрограму роботи дискретного пристрою, називається **мікропрограмним автоматом**. Ту саму ГСА можна інтерпретувати як автоматом Мілі, так і автоматом Мура.

Абстрактний синтез мікропрограмного автомата за ГСА здійснюється в два етапи:

1. Одержання позначеної ГСА.
2. Побудова графа автомата або таблиць переходів і виходів.

Абстрактний синтез автомату Мілі. *На першому етапі* (одержання позначеної ГСА) входи вершин, слідуючих за операторними, позначають символами $\mathbf{a_1, a_2, \ldots}$ за наступними правилами:

1) символом a_1 позначають вхід вершини, наступної за початковою, а також вхід кінцевої вершини;
2) входи усіх вершин, наступних за операторними, повинні бути позначені;
3) входи різних вершин, за винятком кінцевої, позначаються різними символами;
4) якщо вхід вершини позначається, то тільки одним символом.

Ясно, що для проведення оцінок буде потрібно кінцеве число символів $a_1,\ldots,a_m$. Результатом першого етапу є позначена ГСА, що є основою для другого етапу - переходу до графа або таблиць переходів-виходів. Приклад **ГСА, позначеної для автомату Мілі,** представлений на рис. 12.4.

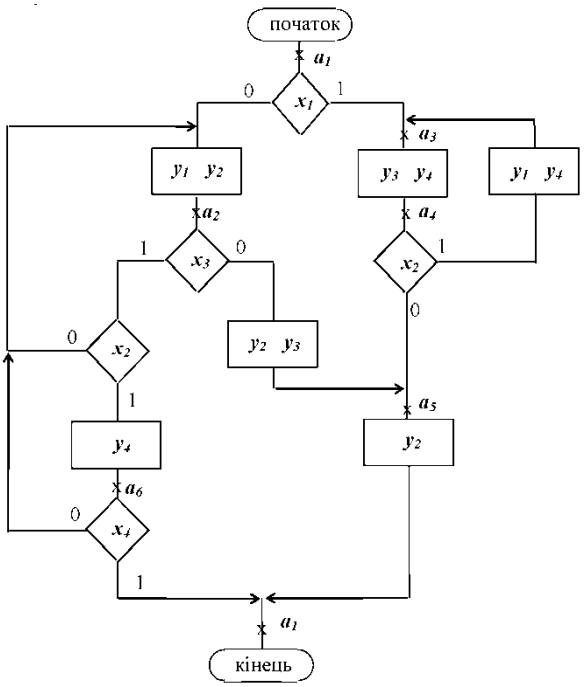


Рис. 12.4 – ГСА, позначена для автомату Мілі

На другому етапі з позначеної ГСА будують граф автомата або таблиці переходів-виходів. Для цього вважають, що в автоматі буде стільки станів, скільки символів a_i знадобилося при оцінці ГСА.

Значення умов x_2, x_3, x_4 на цьому переході не робить впливу на автомат.

Виключення складає тільки шлях, який веде в кінцеву вершину, він може не містити жодної операторної вершини (наприклад, перехід з a_6 в a_2 тобто не супроводжується виробленням вихідних сигналів.

Відмічаємо на графі всі зазначені шляхи для всіх станів у вигляді дуг, яким приписуємо умови переходу і вихідний сигнал цього переходу. Одержимо граф автомату (рис. 12.5).

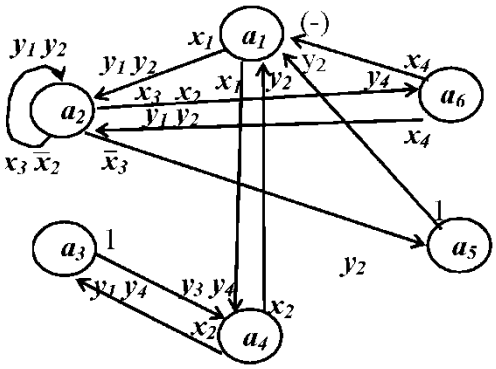


Рис.12.5 – Граф автомату Мілі

На цьому графі переходам типу a_3 —*XI*$_4$, a_S —> a_j приписується умова переходу **1**, тому що ці переходи є безумовними і виконуються завжди, коли автомат попадає в стан a_S (або a_S).

На підставі позначеної ГСА або графу автомата можна побудувати таблицю

переходів-виходів. Для мікропрограмних автоматів таблиця переходів-виходів будується у вигляді списку і розрізняються пряма і зворотня таблиці. Для даного автомату пряма таблиця представлена в табл. 12.1, зворотня - у табл. 12.2.

Таблиця 12.1 Пряма таблиця переходів-виходів автомату Мілі

a_m	a_s	X	Y
a_1	a_2	x_1	$y_1 y_2$
	a_4	$\bar{x}_1$	$y_3 y_4$
a_2	a_2	$x_3 \bar{x}_2$	$y_1 y_2$
	a_5	x_3	$y_2 y_3$
	a_6	$x_3 x_2$	y_4
a_3	a_4	1	$y_3 y_4$
a_4	a_1	$\bar{x}_2$	y_2
	a_3	x_2	$y_1 y_4$
a_5	a_1	1	y_2
a_6	a_1	x_4	-
	a_2	$\bar{x}_4$	$y_1 y_2$

Таблиця 12.2 Зворотна таблиця переходів-виходів автомату Мілі

a_m	a_s	X	Y
a_4	a_1	x_2	y_2
a_5		1	y_2
a_6		x_4	-
a_1	a_2	$\bar{x}_1$	$y_1 y_2$
a_2		$x_3 x_2$	$y_1 y_2$
a_6		x_4	$y_1 y_2$
a_4	a_3	x_2	$y_1 y_4$
a_1	a_4	x_1	$y_3 y_4$
a_3		1	$y_3 y_4$
a_2	a_5	$\bar{x}_3$	$y_2 y_3$
a_2	a_6	$x_3 x_2$	y_4

У наведених таблицях a_m - вихідний стан, a_s - стан переходу, X - умова (вхідний сигнал), що забезпечує перехід зі стану a_m у стан a_s, Y - вихідний сигнал, вироблюваний автоматом при переході з a_m у a_s.

Абстрактний синтез автомату Мура. Для автомату Мура на етапі одержання позначеної ГСА розмітка виконується згідно наступних правил:

1) символом ai позначаються початкова і кінцева вершини;
2) всі операторні вершини повинні бути позначені;
3) різні операторні вершини позначаються різними символами.

Приклад ГСА, позначеної для автомату Мура, представлений на рис. 12.6.

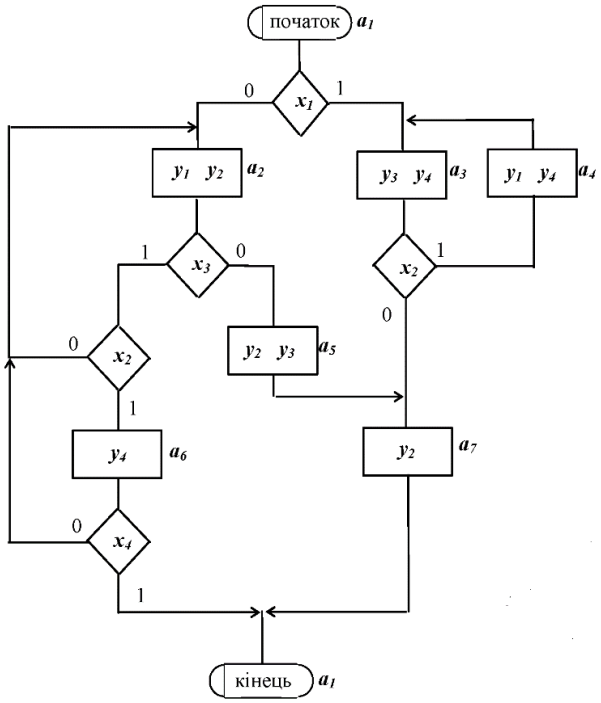


Рис.12.6 – ГСА, позначена для автомату Мура

Граф автомату Мура, що відповідає позначеній ГСА, представлений на рис. 12.7. Побудова його аналогічна побудові графу автомату Мілі.

Таблиці переходів виходів автомату Мура представлені в табл. 12.3 (пряма) і табл. 12.4 (зворотня).

Звичайно для автомату Мура в таблиці переходів-виходів додатковий стовпець для вихідних сигналів не використовується, і вихідний сигнал записується в стовпці, де вказується вихідний стан am або стан переходу as.

Одержанням графа або таблиць переходів-виходів закінчується етап абстрактного синтезу мікропрограмного автомата. Як і для кінцевих автоматів, на етапі абстрактного синтезу можна виконати мінімізацію кількості внутрішніх станів автомата.

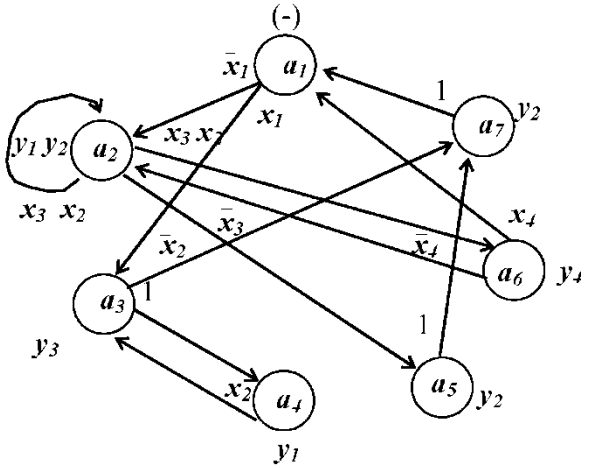


Рис.12.7 – Граф автомату Мура

Таблиця 12.3 Пряма таблиця переходів автомату Мура

$a_m(Y)$	a_s	X
$a_1(\ )$	a_2 a_3	$\bar{x}_1$ x_1
$a_2(y_1y_2)$	a_2 a_5 a_6	$x_3\,\bar{x}_2$ $\bar{x}_3$ $x_3\,x_2$
$a_3(y_3y_4)$	a_4 a_7	x_2 x_2
$a_4(y_1y_4)$	a_3	1
$a_5(y_2y_3)$	a_7	1
$a_6(y_4)$	a_1 a_2	x_4 $\bar{x}_4$
$a_7(y_2)$	a_1	1

Таблиця 12.4 Зворотна таблиця переходів- автомату Мура

a_m	$a_s(Y)$	X
a_6 a_7	$a_1(-)$	x_4 1
a_1 a_2 a_6	$a_2(y_1y_2)$	x_1 $x_3\,\bar{x}_2$ $\bar{x}_4$
a_1 a_4	$a_3(y_3y_4)$	x_1 1
a_3	$a_4(y_1y_4)$	x_2
a_2	$a_5(y_2y_3)$	x_3
a_2	$a_6(y_4)$	$x_3\,x_2$
a_3 a_5	$a_7(y_2)$	$\bar{x}_2$ 1

5. Структурний синтез мікропрограмних автоматів

Структурний синтез мікропрограмних автоматів після одержання графу автомата або таблиці переходів-виходів в принципі аналогічний канонічному методові синтезу цифрових автоматів, розглянутому раніше. Однак існують і певні особливості, в першу чергу зв'язані з тим, що для реальних автоматів кількість елементів пам'яті і вхідних сигналів може досягати десяти і більше. Функції збудження і вихідних сигналів важко піддаються мінімізації та й практично мінімізація не дає істотного спрощення цих функцій при великій кількості змінних. Тому мінімізація практично не використовується при синтезі мікропрограмних автоматів.

При виконанні структурного синтезу будують так звані структурні таблиці переходів і виходів, що також можуть бути як прямими так і зворотними.

Розглянемо етапи структурного синтезу на конкретних прикладах.

Структурний синтез автомату Мілі. Виконаємо структурний синтез мікропрограмного автомату Мілі, заданого своєю таблицею переходів-виходів (табл. 12.1 або табл. 12.2). Як приклад, синтез будемо виконувати за прямою таблицею (табл. 12.5).

Таблиця 12.5

Пряма таблиця переходів – виходів автомату Мілі

a_m	a_s	X	Y
a_1	a_2 a_4	$\bar{x}_1$ x_1	y_1y_2 y_3y_4
a_2	a_2 a_5 a_6	$x_3\bar{x}_2$ x_3 x_3x_2	y_1y_2 y_2y_3 y_4
a_3	a_4	1	y_3y_4
a_4	a_1 a_3	$\bar{x}_2$ x_2	y_2 y_1y_4
a_5	a_1	1	y_2
a_6	a_1 a_2	x_4 $\bar{x}_4$	- y_1y_2

1. У вихідному автоматі кількість станів M**=6**, отже, кількість елементів пам'яті:

$$m = |\log_2 M| = |\log_2 6| = 3.$$

Нехай для синтезу використовуються *JK*- тригери.

2. Кодуємо внутрішні стани автомата, використовуючи для цього карту Карно (рис.12.8) і по можливості метод сусіднього кодування.

Q_2Q_3 / Q_1	00	01	11	10
0	a_1	a_4	a_6	a_2
1		a_3		a_5

a_1 - 000 a_4 - 001
a_2 - 010 a_5 - 110
a_3 - 101 a_6 - 011

Рис. 12.8 – Карта Карно для кодування станів автомата

3. Будуємо пряму структурну таблицю переходів-виходів автомата Мілі (табл. 12.6). У даній таблиці в стовпцях $k(a_m)$ і $k(a_s)$ вказується код вихідного стану і стану переходу відповідно. У стовпці функцій збудження (ФЗ) вказуються ті значення функцій збудження, які на даному переході обов'язково дорівнюють **1**. Інші (тобто рівні **0** або які приймать невизначені значення) не вказуються. Це еквівалентно тому, що всім невизначеним значенням функцій збудження приписується значення **0**, що в загальному випадку не дає мінімальної функції, однак у реальних автоматах мінімізація звичайно не проводиться з погляду її неефективності. Пропонується самостійно побудувати структурну таблицю переходів із вказівкою всіх значень функцій збудження (у тому числі і невизначених), виконати мінімізацію і порівняти результати з приведеними нижче.

Таблиця 12.6

Структурна таблиця переходів – виходів автомата Мілі

A_m - початковий стан	$K(a_m)$ - кодований початковий стан	a_s - стан переходу	$K(a_s)$ - кодований стан переходу	X - умова (вхідний сигнал, що забезпечує перехід a_m–a_s)	Y - вихідний сигнал при переході a_m–a_s	ФЗ - функція збудження тригерів
a_1	000	a_2 a_4	010 001	$\bar{x}_1$ x_1	y_1y_2 y_3y_4	J_2 J_3
a_2	010	a_2 a_5 a_6	010 110 011	$x_3\bar{x}_2$ $\bar{x}_3$ x_3x_2	y_1y_2 y_2y_3 y_4	- J_1 J_3
a_3	101	a_4	001	1	y_3y_4	K_1
a_4	001	a_1 a_3	000 101	$\bar{x}_2$ x_2	y_2 y_1y_4	K_3 J_1
a_5	110	a_1	000	1	y_2	K_1K_2
a_6	011	a_1 a_2	000 010	x_4 $\bar{x}_4$	- y_1y_2	K_2K_3 K_3

4. Для одержання функцій збудження діємо у такий спосіб. Вираз для кожної функції збудження виходить у вигляді логічної суми добутків виду a_iX, де a_i - початковий стан, X - умова переходу. Для спрощення отриманих виразів

$$J_1 = a_2\bar{x}_3 + a_4x_2 \qquad K_1 = a_3 + a_5$$
$$J_2 = a_1\bar{x}_1 \qquad K_2 = a_5 + a_6x_4$$
$$J_3 = a_1x_1 + a_2x_3x_2 \qquad K_3 = a_4\bar{x}_2 + a_6x_4 + a_6\bar{x}_4 = a_6 + a_4\bar{x}_2$$

виконуємо всі можливі операції склеювання .

5. Аналогічно одержуємо функції виходів:

$$y_1 = a_1\bar{x}_1 + a_2x_3\bar{x}_2 + a_4x_2 + a_6\bar{x}_4$$
$$y_2 = a_1\bar{x}_1 + a_2x_3\bar{x}_2 + a_2\bar{x}_3 + a_4\bar{x}_2 + a_5 + a_6\bar{x}_4$$
$$y_3 = a_2\bar{x}_3 + a_3 + a_1x_1$$
$$y_4 = a_1x_1 + a_2x_3x_2 + a_3 + a_4x_2$$

6. На базі одержаних функцій збудження та функцій виходів можна побудувати функціональну схему автомату Мілі. Для побудови функціональної схеми автомату за отриманими виразами необхідно або замінити a_i його

значеннями через $Q_1Q_2Q_3$, або одержати сигнал, що відповідає a_i. Зазвичай використовують другий спосіб і для одержання сигналу a_i застосовують так званий дешифратор станів, на вхід якого надходять сигнали з виходів елементів пам'яті $Q_1Q_2Q_3$. Крім того, при побудові схеми намагаються виділити спільні частини, що зустрічаються у функціях збудження або вихідних сигналах. У цьому випадку остаточна система рівнянь, за якими будується схема, матиме вигляд:

$A = a_2x_3\bar{x}_2 + J_2$; $J_1 = D + B$; $y_1 = A + B + E$;
$B = a_4x_2$; $K_1 = a_3 + a_5$; $y_2 = A + D + C + a_5 + E$;
$C = a_4\bar{x}_2$; $J_2 = a_1\bar{x}_1$; $y_3 = D + F + a_3$;
$D = a_2\bar{x}_3$; $K_2 = a_5 + a_6x_4$; $y_4 = a_3 + B + J_3$;
$E = a_6\bar{x}_4$; $K_3 = a_6 + C$;
$F = a_1x_1$ $J_3 = F + a_2x_3x_2$

Функціональна схема автомата, побудована на підставі отриманих рівнянь, представлена на рис. 12.9.

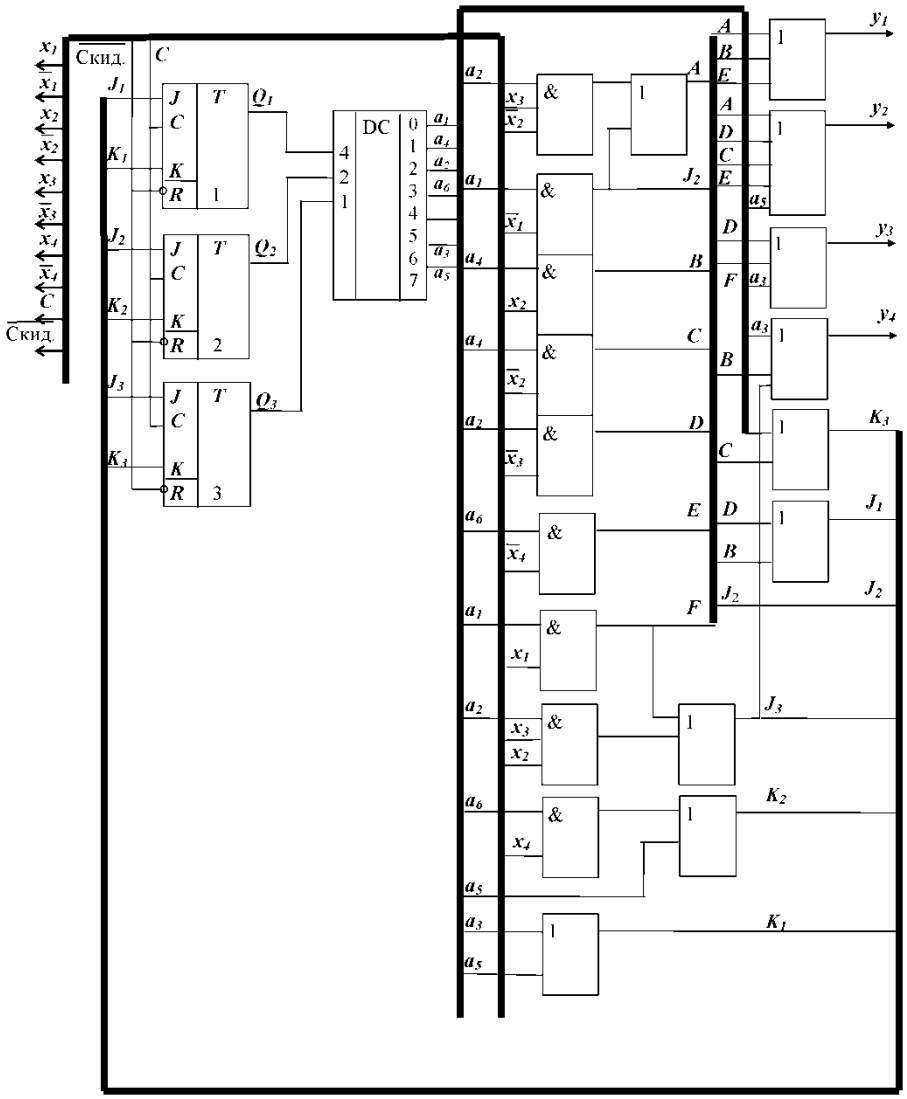


Рис.12.9 – Функціональна схема мікропрограмного автомату Мілі

Структурний синтез автомату Мура.

Виконаємо структурний синтез мікропрограмного автомату Мура, заданого своєю таблицею переходів (табл. 12.3 або табл. 12.4). Як приклад, синтез будемо виконувати за зворотньою таблицею (табл. 12.7).

Таблиця 12.7

Зворотна таблиця переходів автомату Мура

a_m	$a_s(Y)$	X
a_6 a_7	a_1 (-)	x_4 1
a_1 a_2 a_6	a_2 $(y_1 y_2)$	x_1 $x_3 x_2$ x_4
a_1 a_4	a_3 $(y_3 y_4)$	x_1 1
a_3	a_4 $(y_1 y_4)$	x_2
a_2	a_5 $(y_2 y_3)$	x_3
a_2	a_6 (y_4)	$x_3 x_2$
a_3 a_5	a_7 (y_2)	x_2 1

1. У вихідному автоматі кількість станів $M = 7$, отже кількість елементів пам'яті

$$m = \lceil \log_2 M \rceil = \lceil \log_2 7 \rceil = 3$$

Нехай для синтезу використовуються ***D***-тригери.

2. Кодуємо внутрішні стани автомата, використовуючи алгоритм кодування для ***D***- тригерів. Кількість переходів у даний стан легко визначається зі зворотної таблиці: $a_1 \sim 2$, $a_2 \sim 3$, $a_3 \sim 2$, $a_4 \sim 1$, $a_5 \sim 1$, $a_6 \sim 1$, $a_7 \sim 2$. Тому коди станів наступні: a_2 - 000, a_1 - 001, a_3 - 010, a_7 - 100, a_4 - 011, a_5 - 101, a_6 - 110.

3. Будуємо структурну таблицю переходів - виходів автомату Мура. Побудова таблиці виконується аналогічно автомату Мілі.

Таблиця 12.8

Структурна таблиця переходів – виходів автомата Мура

a_m - початковий стан	$K(a_m)$ - кодований початковий стан	$a_s(Y)$ - стан переходу сумісно з вихідним сигналом	$K(a_s)$ - кодований стан переходу	X - умова (вхідний сигнал, що забезпечує перехід a_m–a_s)	ФЗ - функція збудження
a_6 a_7	110 100	a_1 (–)	001	x_4 1	D_3 D_3
a_1 a_2 a_6	001 000 110	$a_2(y_1 y_2)$	000	$\overline{x}_1$ $x_3 \overline{x}_2$ $\overline{x}_4$	–
a_1 a_4	001 011	$a_3(y_3 y_4)$	010	x_1 1	D_2 D_2
a_3	010	$a_4(y_1 y_4)$	011	x_2	$D_2 D_3$
a_2	000	$a_5(y_2 y_3)$	101	$\overline{x}_3$	$D_1 D_3$
a_2	000	a_6 (y_4)	110	$x_3 x_2$	$D_1 D_2$
a_3 a_5	010 101	a_7 (y_2)	100	$\overline{x}_2$ 1	D_1 D_1

4. Вирази для функцій збудження виходять у вигляді суми добутків $a_i X$, де a_i-вихідний стан, X - умова переходу.

5. Вирази для вихідних сигналів автомата Мура одержуємо, виходячи з того, що ці сигнали визначаються тільки внутрішнім станом автомата.

$$D_1 = a_2\bar{x}_3 + a_2x_3x_2 + a_3\bar{x}_2 + a_5$$
$$D_2 = a_1x_1 + a_4 + a_3x_2 + a_2x_3x_2$$
$$D_3 = a_6x_4 + a_7 + a_3x_2 + a_2\bar{x}_3$$

або

$$A = a_3x_2 \qquad B = a_2x_3x_2$$
$$D_1 = a_2\bar{x}_3 + B + a_3\bar{x}_2 + a_5$$
$$D_2 = a_1x_1 + a_4 + A + B$$
$$D_3 = a_6x_4 + a_7 + A + a_2\bar{x}_3$$

$$y_1 = a_2 + a_4 \qquad y_2 = a_2 + a_5 + a_7$$
$$y_3 = a_3 + a_5 \qquad y_4 = a_3 + a_4 + a_6$$

6. На основі одержаних функцій збудження та вихідних функцій можна побудувати функціональну схему автомату Мура. Для побудови функціональної схеми автомату як і в попередньому випадку використовуємо дешифратор станів. Схема представлена на рис. 12.10.

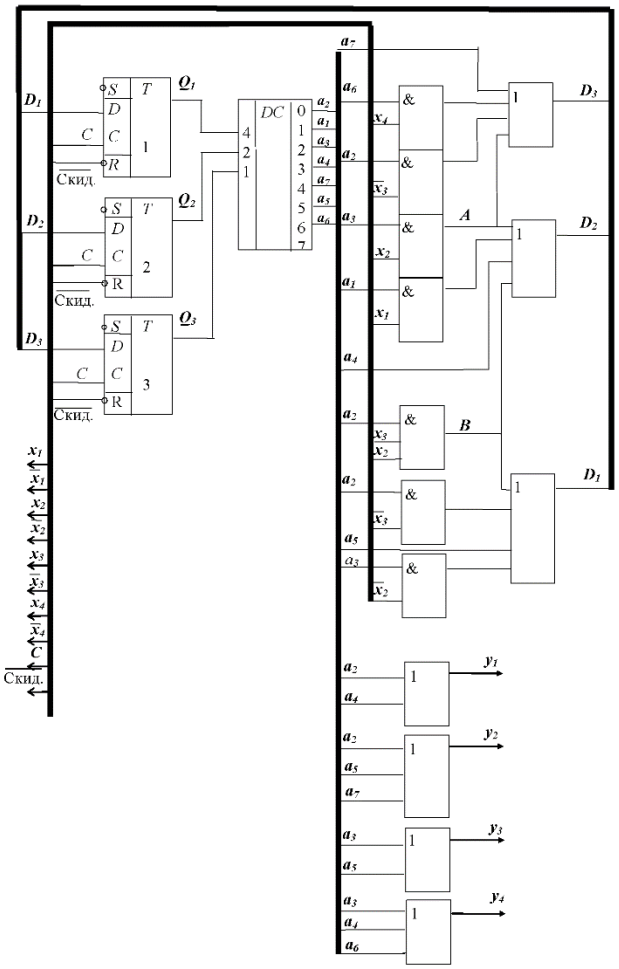


Рис.12.10 – Функціональна схема автомату Мура

При синтезі мікропрограмних автоматів викладеним методом одержувані вирази для функцій збудження не завжди є мінімальними і у деяких випадках можуть бути спрощені. Зокрема, можна довизначити функції збудження на деяких наборах одиничним значенням (а не нульовим, як було раніше) і

виконати всі операції склеювання. Наприклад, у синтезованому раніше автоматі Мілі у такий спосіб можна одержати значення $K_1=1$.

Автомат Мілі на протязі такту зберігає свій стан і лише наприкінці його переходить у новий. Поки автомат знаходиться в даному стані A_i , він виробляє вихідні сигнали $y=f(A_i,x)$, де x - сигнали, що подаються на вхід автомата протягом даного такту. У зв'язку з цим автомат Мілі може інтерпретувати мікропрограму коректно тільки в тому випадку, якщо для будь-якого

Переходу виконується умова незалежності логічних умов від результатів виконання мікрооперацій на даному переході.

Умова незалежності порушується, якщо на деякому переході з a_m в a_s під дією сигналу x з виробленням y_i спостерігається функціональна залежність $x = f(y_i)$. У такому випадку виконання мікрооперації y_i може привести до зміни значення x і автомат, знаходячись у стані a_m, і, реагуючи на набір вхідних сигналів, сформує набір вихідних сигналів, не відповідний мікропрограмі. Щоб уникнути цього, можна використовувати наступні способи:

1) запам'ятати значення сигналів x на проміжок часу, рівний тривалості такту;
2) ввести в автомат додаткові стани;
3) реалізувати автомат за схемою Мура.

Запам'ятовування значень сигналів x протягом такту здійснюється операційним автоматом за допомогою додаткових елементів пам'яті – тригерів. Інтерпретація мікропрограми автоматом Мура розглядалася раніше. Введення додаткових станів ілюструється рис.12.11.

У початковому автоматі (рис. 12.11.а) є переходи з a_i в a_j під дією сигналів x и $\bar{x}$ з виробленням y_1 і y_2 відповідно і має місце $x = f(y_1, y_2)$. Дійсно, введення допоміжних станів a_k і a_l дозволяє усунути можливу помилку у видачі вихідних сигналів. На переходах a_i a_k або $a_i a_l$ вихідні сигнали не виробляються. Переходи a_k a_j або a_l a_j є безумовними з виробленням y_1 або y_2 відповідно. У такому випадку зміна значення x, у результаті виконання мікрооперацій y_1 або y_2, не вплине на вихідний сигнал, вироблюваний автоматом, тому що на переходах a_i a_k або $a_i a_l$ вихідний сигнал вже не залежить від x.

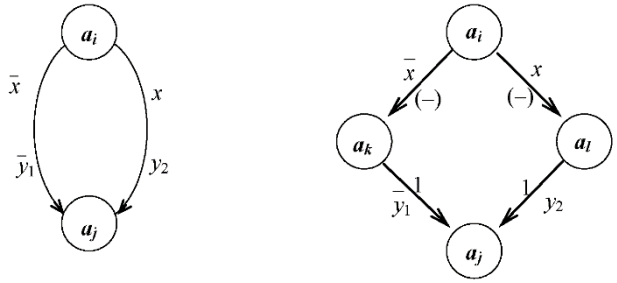


Рис.12.11 - Введення додаткових станів в автомат Мілі:
а)початковий автомат; б) автомат з додатковими станами

Контрольні питання:

1. Як взаємодіють управляюча і операційна частини пристрою?
2. Чим відрізняються автомат Мілі та автомат Мура?
3. Яка послідовність дій необхідна для побудови управляючого пристрою для автомата Мілі чи автомата Мура?
4. Як виконується етап кодування операційних та умовних вершин схеми алгоритму кодами мікрооперацій і кодами вхідних сигналів?
5. Як перейти від схеми алгоритму до закодованої мікрооперaційної схеми автомата?
6. Як виконується етап кодування мікрокоманд з використанням кодів мікрооперацій?
7. Як отримати закодовану мікрокомандну схему автомата?
8. Який вміст основної таблиці абстрактного автомата?
9. Як побудувати граф-схему переходів автомата із одного внутрішнього стану в інший?
10. Як побудувати систему рівнянь для функцій переходів і для функцій виходів автомата?
11. Як виконується кодування внутрішніх станів автомата?
12. Що входить до складу схеми операційного автомата?
13. Опишіть особливості функціонування схеми операційного автомата.

ТЕМА 13. СИНТЕЗ ОПЕРАЦІЙНИХ АВТОМАТІВ

План:

1. Формалізований опис операційного автомату. Закодовані мікрооперaційна та мікрокомандна схеми алгоритму. Основна таблиця автомату.
2. Граф-схема переходів. Системи рівнянь переходів та виходів. Кодування внутрішніх станів автомату Схема операційного автомату.

Завдання для самостійної роботи

I. Ознайомтесь з навчальними матеріалами.
II. Дайте відповідь на запитання.
III. Виконайте вправи для самостійної роботи в робочому зошиті.

Матеріали для самостійного опрацювання

1. Формалізований опис операційного автомату. Закодовані мікрооперaційна та мікрокомандна схеми алгоритму. Основна таблиця автомату.

Як показав академік В.М. Глушков, у будь-якому пристрої обробки цифрової інформації можна виділити два основних блоки –операційний автомат (ОА) і керуючий автомат (КА).

Операційний автомат (ОА) служить для збереження слів інформації, виконання набору мікрооперацій і обчислення значень логічних умов, тобто ОА є структурою, організованою для виконання дій над інформацією.

Мікрооперації, виконувані ОА, задаються *множиною керуючих сигналів* $Y=\{y_1,....,y_M\}$, з кожним з яких ототожнюється визначена мікрооперація.

Значення логічних умов, що обчислюються в операційному автоматі, відображаються *множиною інформаційних сигналів* $X=\{x_1,...,x_L\}$, кожний з яких ототожнюється з визначеною логічною умовою.

Таким чином, будь-який **операційний пристрій**–процесор, канал вводу-виводу і т.п. – є композицією операційного і керуючого автоматів. ОА, реалізуючи дії над словами інформації, є виконавчою частиною пристрою, роботою якого керує КА, що генерує необхідні послідовності керуючих сигналів.

В основу проектування операційних пристроїв різного призначення покладено принцип функціонального мікропрограмування і концепцію операційних і керуючих автоматів. При цьому **мікропрограмування** – це спосіб опису функцій операційних пристроїв безвідносно до технічних засобів, які використовуються для їх реалізації. Таке тлумачення мікропрограмування дозволяє формалізувати синтез структур будь-яких операційних пристроїв незалежно від способу керування роботою пристрою. Необхідно відзначити, що принципи побудови і методи проектування операційних і керуючих автоматів є

тою основою, на якій базується теорія і практика проектування більшої частини пристроїв ЕОМ.

Складність і відповідальність задач, що вирішуються сучасними ЕОМ та системами, потребують від них високої надійності та продуктивності. Тому, однією з основних проблем, які стоять перед розробниками сучасної обчислювальної техніки, є підвищення продуктивності, відмовостійкості та життєздатності.

В наш час основним напрямком вирішення цих проблем є створення обчислювальних машин, які побудовані з великої кількості однорідних модулів, що утворюють єдину систему шляхом встановлення логічних зв`язків між ними. В цьому суть концепції мультипроцесорних ЕОМ, частковими випадками яких є матричні, конвеєрні, з програмованою архітектурою і т.д. При цьому висовуються вимоги простоти контрольного обладнання і високої достовірності обробки інформації.

Операційний автомат визначається наступною сукупністю множин:

1. множиною *вхідних слів* $\boldsymbol{D=\{d_1,...,d_н\}}$, що вводяться в автомат у якості операндів;
2. множиною *вихідних слів* $\boldsymbol{R=\{r_1,...,r_Q\}}$, що представляють результати операцій;
3. множиною *внутрішніх слів* $\boldsymbol{S=\{s_1,...,s_N\}}$, використовуваних для представлення інформації в процесі виконання операцій. Можна вважати, що вхідні і вихідні слова збігаються з визначеними внутрішніми $D \subseteq S$, $R \subseteq S$.
4. множиною *мікрооперацій* $\boldsymbol{Y=\{y_m\}}$, що реалізують перетворення $S=\phi m(s)$ над словами інформації, де ϕm –функція, що обчислюється;
5. множиною *логічних умов* $\boldsymbol{X=\{xi\}}$, де $xi=\omega i(si)$ і ωi – булева функція;

Таким чином, функція ОА задана, якщо задані (визначені) множини ***D, R, S, Y, X***. **Час не є аргументом функції ОА!**

Функція встановлює список дій – мікрооперацій і логічних умов, що може виконувати автомат, але ніяк не визначає порядок проходження цих дій у часі. Тобто функція ОА характеризує засоби, що можуть бути використані для обчислень, але не сам обчислювальний процес.

Складемо закодовану мікрооперацийну схему алгоритму мікропрограмного автомата за схемою Уілкса - Стрінжера у вигляді автомата Мілі.

Приклад 14.1. Побудувати операційний автомат, що обчислює кількість парних елементів у двох одномірних масивах (A [n], B [m]). Мікропроцесорний автомат необхідно реалізувати за схемою Уілкса-Стрінжера у вигляді автомата Мілі. Оптимальну функціональну схему керуючих частин автомата синтезувати на елементах системи ТА, АБО, НІ, RS-, D- тригерах, доповнюючи її необхідними за алгоритмом функціональними автоматами.

До складу змістовної схеми алгоритму входять операційні та умовні верхівки. Наш алгоритм виконує знаходження кількості парних елементів у двох одномірних масивах розмірністю [n] та [m], використовуючи при цьому чотири (4) умовні верхівки і десять (10) операційних верхівок. Позначення операційних

верхівок показано на рис.13.1.

Позначення умовних верхівок на рис.13.2.

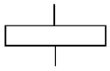


Рис.13.1 – Позначення операційних верхівок

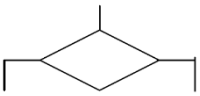


Рис.13.2 – Позначення умовних верхівок

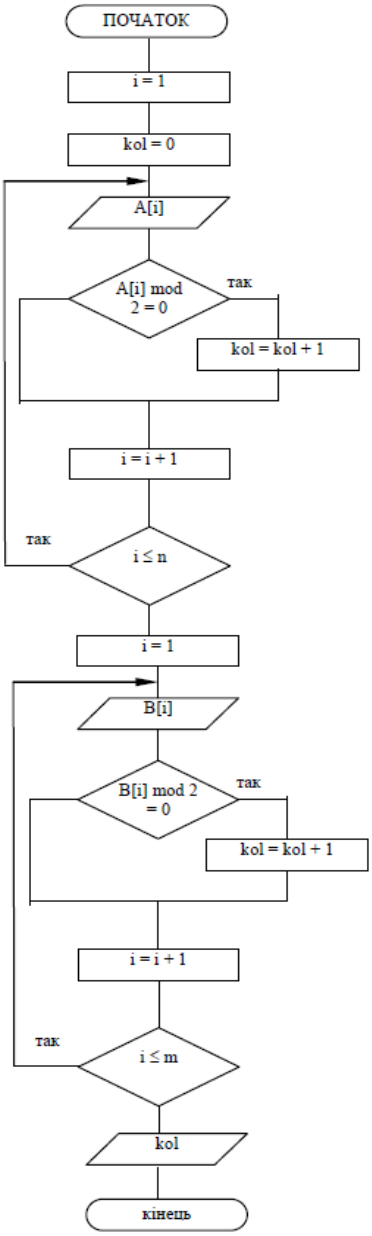


Рис.13.3 – Змістовна схема алгоритму

Кожна верхівка, чи то операційна чи умовна, кодується. Причому, якщо мікрооперації повторюються і умовні верхівки повторюються, вони кодуються однаково. У даному прикладі мікрооперації повторюються двічі, тому, однакові верхівки ми можемо кодувати одним кодом. Таблиця кодування верхівок зображена у таблиці 13.1.

Таблиця 13.1.

Таблиця кодування операційних та умовних верхівок

Код	Зміст	Примітка
mY_1	i = 1	
mY_2	kol = 0	
mY_3	A [i]	Ввід A [i]
mY_4	kol = kol + 1	
mY_5	i = i + 1	
mY_6	B [i]	
mY_7	kol	Вивід kol
X_1	A [i] mod 2 = 0	так - 1, ні - 0
X_2	i ≤ n	так - 1, ні - 0
X_3	B [i] mod 2 = 0	так - 1, ні - 0
X_4	i ≤ m	так - 1, ні - 0

Закодована мікрооперaційна схема алгоритму будується на основі змістовної схеми алгоритму (рис.13.3) і таблиці кодування операційних та умовних верхівок (таблиця 13.1), шляхом заміни відповідних блоків. Закодована мікрооперaційна схема алгоритму зображена на рис.13.4.

Складаємо таблицю кодування мікрокоманд. Кожна мікрооперація кодується своєю мікрокомандою. Мікрооперації, які виконуються одна за одною послідовно на протязі одного такту часу, поєднуються до однієї мікрокоманди. У даному прикладі дві мікрооперації (mY1, mY2) виконуються одна за одною послідовно. Тому ми поєднуємо їх в одну мікрокоманду. Таблиця кодування зображена в таблиці 13.2.

Таблиця 13.2.

Таблиця кодування

Мікрокоманда	Мікрооперація
Y_1	mY_1, mY_2
Y_2	mY_3
Y_3	mY_4
Y_4	mY_5
Y_5	mY_1
Y_6	mY_6
Y_7	mY_7

Складаємо закодовану мікрокомандну схему алгоритму (рис.13.5). Проставляємо мітки внутрішніх станів автомата Мілі таким чином:

- мітки ставляться після кожної мікрокоманди (перед блоком після операційного);
- початок та кінець мікрокомандної схеми алгоритму відмічається міткою a0;
- мітки проставляються згідно з порядковим номером.

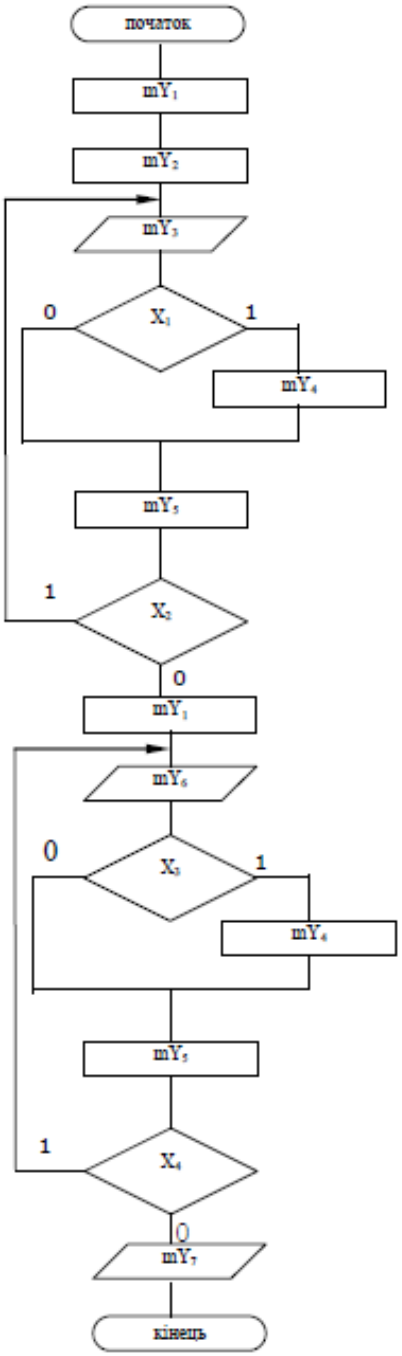


Рис.13.4 – Закодована мікрооперацiйна схема

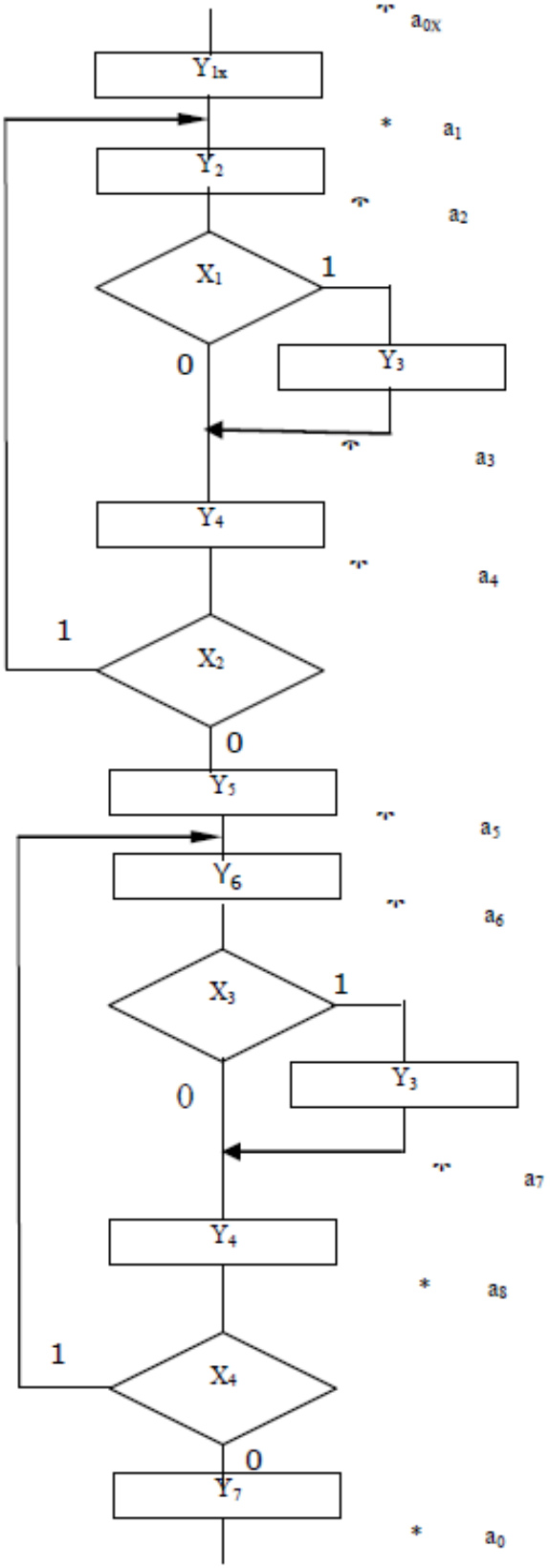


Рис.13.5 – Закодована мікрокомандна схема алгоритму

2. Структурна схема операційного автомату. Побудова схеми операційного автомату.

Продовжимо синтез операційного автомату з прикладу 13.1 на основі одержаних закодованих мікрооперацийної (рис.13.4) та мікрокомандної (рис.13.5) схеми алгоритму.

Будуємо основну таблицю автомату (таблиця 13.3*)*. Ця таблиця складається на основі закодованої мікрокомандної схеми алгоритму. В першому стовпчику таблиці записуються усі стани, в яких може знаходитися наш автомат. В першому рядку таблиці записуються способи переходу автомата з одного стану в інший (вхідні сигнали), тобто, чи то буде CI (той перехід, в процесі якого на шляху не зустрілась жодна верхівка), чи то при переході автомату буде поставлена умова. В клітинках таблиці фіксується, перехід до якого стану здійснюється, і що буде на виході. Наприклад, із стану a_0 автомат може здійснити перехід до стану a_1 і в результаті цього переходу на виході автомата буде Y1, тобто, автомат виконає ті мікрооперації, які виконуються на протязі одного такту часу (mY1 і mY2 закодовані мікрокомандою Y1), при чому, цей перехід станеться під синхроімпульсним сигналом.

Таблиця 13.3.

Основна таблиця автомату

	CI=1	X_1	$\overline{X_1}$	X_2	$\overline{X_2}$	X_3	$\overline{X_3}$	X_4	$\overline{X_4}$
a_0	a_1/Y_1								
a_1	a_2/Y_2								
a_2		a_3/Y_3	$a_3/\_\_$						
a_3	a_4/Y_4								
a_4				$a_1/\_\_$	a_5/Y_5				
a_5	a_6/Y_6								
a_6						a_7/Y_3	$a_7/\_\_$		
a_7	a_8/Y_4								
a_8								$a_5/\_\_$	a_0/Y_7

Будуємо граф-схему переходів (рис.13.6*)*. Граф-схема будується на основі рис.13.5. і таблиці 13.3. Кружечками позначаються можливі стани автомата. Стрілки вказують на перехід із стану *i* до стану *j*. Над стрілкою вказується, під яким вхідним сигналом станеться перехід, і що при цьому буде на виході автомата

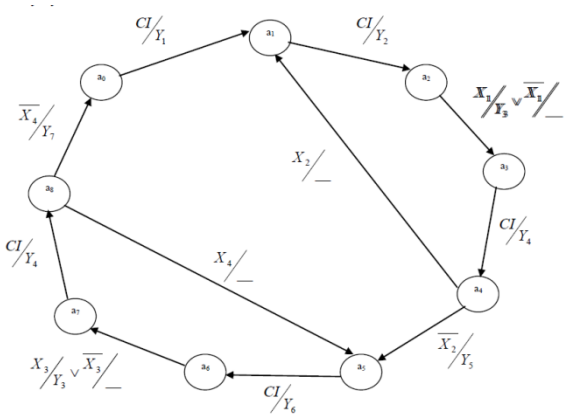


Рис.13.6 – Граф-схема переходів

Складаємо систему рівнянь переходів. Ця система складається на основі граф-схеми переходів (рис.13.6) або основної таблиці абстрактного автомата (таблиця 13.3). Сигнал СІ опущений.

$$\begin{cases} a_{0t} = a_{8t-1}\overline{X_4} \\ a_{1t} = a_{0t-1} \vee a_{4t-1}X_2 \\ a_{2t} = a_{1t-1} \\ a_{3t} = a_{2t-1}X_1 \vee a_{2t-1}\overline{X_1} = a_{2t-1} \\ a_{4t} = a_{3t-1} \\ a_{5t} = a_{4t-1}\overline{X_2} \vee a_{8t-1}X_4 \\ a_{6t} = a_{5t-1} \\ a_{7t} = a_{6t-1} \\ a_{8t} = a_{7t-1} \end{cases}$$

Складаємо систему рівнянь виходів. Ця система складається на основі закодованої мікрокомандної схеми алгоритму (рис.13.5), де X - вхід, Y - вихід.

$$\begin{cases} Y_{1t} = a_{0t-1} \\ Y_{2t} = a_{1t-1} \\ Y_{3t} = a_{2t-1}X_1 \vee a_{6t-1}X_3 \\ Y_{4t} = a_{3t-1} \vee a_{7t-1} \\ Y_{5t} = a_{4t-1}\overline{X_2} \\ Y_{6t} = a_{5t-1} \\ Y_{7t} = a_{8t-1}\overline{X_4} \end{cases}$$

Для того, щоб закодувати внутрішні стани автомата, визначаємо кількість необхідних для цього тригерів (n). Кількість тригерів розраховується із співвідношення: log2 A ≤ n, де n - кількість необхідних тригерів; A - кількість станів ai (a0 - a8)

A = 9log2 9 ≤ n□n = 4.

Нам необхідні 4 тригери, значить внутрішні стани автомата будемо кодувати чотирьох- розрядним двійковим кодом. Процес кодування зображений у таблиці 13.4.

Таблиця 13.4

Таблиця процесу кодування

	S_1	S_2	S_3	S_4
a_0	0	0	0	0
a_1	0	0	0	1
a_2	0	0	1	0
a_3	0	0	1	1
a_4	0	1	0	0
a_5	0	1	0	1
a_6	0	1	1	0
a_7	0	1	1	1
a_8	1	0	0	0

Операційний автомат складається з наступних частин. У *вхідній частині* розташовані чотири (4) RS-тригери, чотири (4) логічних елементи АБО, на які подається вхідний сигнал, декодер та дві шини, одна з яких необхідна для передачі сигналів, які надходять з декодера, а інша - для сигналів з виходів компаратора. У *перехідній частині* автомата виконується перетворення сигналу на протязі одного такту часу. Пройшовши через логічні елементи ТА і або) АБО, чи того не роблячи, сигнал змінюється і результат надходить на шину (at), відкіля продовжує передаватися до *програмованої логічної матриці* (ПЛМ). Перехідна частина будується на основі системи рівнянь переходів. *Вихідна частина*. Ця частина будується на основі системи рівнянь виходів. Тут виконується той самий процес, що й у перехідній частині, тільки сигнали подаються на вихідну шину Yt, з якої сигнал надходить до вихідної матриці.

Схема операційного автомату зображена на рис.13.7

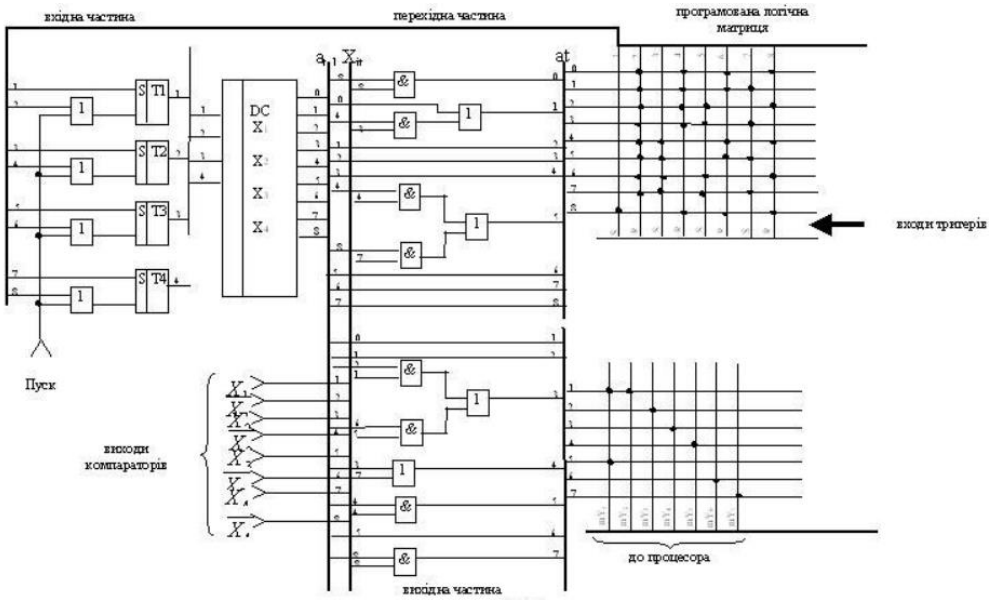


Рис.13.7 – Схема операційного автомату

Операційний автомат (ОА) - це сукупність комбінаційних схем, виконуючих мікрооперації, регістрів/ОЗП для зберігання даних та зовнішніх ліній даних.

Алгоритм роботи операційного автомату:

1) в залежності від одержаних від керуючого автомату сигналів, операційний автомат виконує мікрооперації - різні дії над регістрами, введення і виведення на зовнішні контакти;

2) після виконання мікрооперацій в кожному стані ОА формує мікроумови, за якими керуючий автомат приймає свій наступний стан.

На сьогодні, проектування операційного автомату автоматизоване. Спеціалізованою мовою MPDL (MicroProgram Description Language), розробленою у лабораторії «Нові інформаційні технології» Гомельського університету розробники описують алгоритми. Далі у системі Winter MPDL-програми редагуються, аналізуються та відлагоджуються. За коректною

відлагодженою програмою спеціальним генератором GenMPA генерується схема мікропрограмного автомату, яка реалізує відлагоджений алгоритм. При цьому до операційного автомату включаються лише ті інструкції, які були реально задіяні у мікропрограмі, що забезпечує скорочення апаратних витрат у порівнянні з використанням стандартного процесорного ядра. Системи HLCCAD/IEESD забезпечують візуалізацію, симуляцію, аналіз та при необхідності оптимізацію сгенерованої схеми, а також її конвертування до синтезованого VHDL-опису.

Контрольні питання:

1. Що означає формалізований опис операційного автомату?
2. Дайте визначення таким поняттям: операція, мікрооперація, мікроалгоритм, змістовний мікроалгоритм, закодований мікроалгоритм, операційна схема, структурна схема, функціональна схема, принципова схема.
3. Як перейти від змістовного мікроалгоритму до закодованого мікроалгоритму?
4. Як побудувати граф управляючого автомата за закодованим алгоритмом?
5. Як розрахувати розрядність регістрів для операційного пристрою?
6. Як визначити необхідну тривалість управляючих сигналів?
7. Як визначити час переходу автомата з одного стану в інший?
8. Поясніть необхідність введення додаткових станів автомата.
9. Чи можливий перехід автомата в стан, непередбачений графом, за використання тригерів із внутрішньою затримкою (тригерів, керованих рівнем сигналів)?
10. Коли можливе виникнення помилкових управляючих сигналів (непередбачених графом автомата) і чим визначається їх тривалість?
11. Як визначити час виконання мікроопераций?

ГЛОСАРІЙ ТЕРМІНІВ

D-тригер – елемент затримки – має один інформаційний вхід *D* і один вихід *Q* і здійснює затримку на один такт сигналу, що надійшов на його вхід.

JK- тригер – має два інформаційних входи *J* і *K* та один вихід *Q*. Вхід *J* – вхід установки в 1, вхід *K* – вхід установки в 0, тобто ці входи аналогічні відповідним входам RS-тригера.

RS-тригер – тригер з роздільними входами.Даний тригер має два вхідних канали *R* і *S* та один вихідний *Q*. Вхід *S* (set) називається входом установки в одиницю, вхід *R* (reset) – входом установки в нуль.

T-тригер – тригер з лічильним входом – має один інформаційний вхід *T* і один вихід *Q* і здійснює додавання за модулем два значення сигналу *T* і стану *Q* у заданий момент часу.

Абстрактний автомат називається *кінцевим*, якщо кінцевими є множини: A = {a1, a2, ..., am}, Z = {z1, z2, ..., zf}, W = {w1, w2, ..., wg}.

Автомат зветься *ініціальним*, якщо в ньому виділено початковий стан a1.

Автомат, у якого всі стани стійкі - *асинхронний*, інакше автомат є *синхронним*.

Автоматне (машинне) зображення числа – це представлення числа А в розрядній сітці цифрового автомата.

Адекватність - ступінь відповідності змісту реально одержаної інформації та її очікуваного вмісту.

Актуальність інформації - це ступінь відповідності інформації поточному моменту часу.

Алгебра логіки (алгебра тверджень, висловлювань) - це розділ математичної логіки, в якому вивчаються логічні операції над твердженнями (висловлюваннями).

Алгоритмом називається кінцева сукупність точно сформульованих правил розв'язку певної задачі.

Аналогічно, вхідні набори, що забезпечують на виході ЛС логічний 0, утворять *нульове покриття* . 0 С

Арифметико-логічний пристрій (АЛП) - функціональна частина мікропроцесора, яка виконує логічні та арифметичні дії, необхідні для обробки інформації, яка зберігається в пам'яті комп'ютера.

Арифметико-логічний пристрій (АЛП) - функціональна частина мікропроцесора, яка виконує логічні та арифметичні дії, необхідні для обробки інформації, яка зберігається в пам'яті комп'ютера.

Арифметико-логічний пристрій виконує арифметичні та логічні операції над даними: додавання, віднімання, множення, ділення, порівняння і т.і.

Архітектура ЕОМ - це загальний опис структури та фнкцій ЕОМ на рівні, достатньому для розуміння принципів роботи та системи команд ЕОМ, який не включає деталі технічного та фізичного устрою комп'ютера.

*Багаторозрядні суматори*виконують операцію арифметичного додавання двох багаторозрядних чисел.

Будь-яка булева функція *f* еквівалентна диз'юнкції усіх своїх простих імплікант. Така форма представлення булевої функції називається *скороченою ДНФ.*

Булева функція називається *імплікантою булевої функції ,* якщо для будь-якого набору змінних, на якому *g* = 1, справедливо *f* =1. n x,...,xg 1 n x,...,xf 1

В *комбінаційних схемах* логічний стан виходів елементів залежить лише від комбінації вхідних сигналів в даний момент часу.

Вага розряду Pi числа в позиційній системі числення - це відношення Pi = *qi* / *q*0 = *qi*, де *i* - номер розряду справа наліво, а *q*0 - перший розряд: його номер дорівнює 0, а значення дорівнює 1.

Виключаюче АБО (XOR) - це функція f6(x1,x2), що позначається знаком . Ця операція, як видно з таблиці, реалізує функцію нерівнозначності, тобто фактично реалізується процедура додавання за модулем 2.

Висловлюванням називається твердження, про яке можна чітко сказати, істинне *воно чи* хибне.

Внутрішня пам'ять призначена для зберігання відносно невеликих обсягів інформації при її обробці мікропроцесором.

Генератор тактової частоти генерує електричні імпульси, які синхронізують роботу всіх вузлів комп'ютера.

Графічне зображення комбінаційної схеми, при якому показані зв'язки між різними елементами, а самі елементи представлені умовними позначеннями, називається *функціональноюсхемою.*

*Граф-схема алгоритму (ГСА)*визначає обчислювальний процес послідовно учасі, встановлюючи порядок перевірки логічних умов *x*1–*xL*i порядок проходження мікрооперацій *y*1–*yt*.

ГСА називається *змістовною*, якщо всередині вершин записані в явному вигляді мікрооперації та логічні умови. Якщо ж кожну мікрооперацію позначити символами *yi*, а логічні умови через *xi*, то вийде так звана *кодована ГСА.*

Двійковий суматор додаткового коду (ДСДК) оперує числами, представленими в додатковому коді. Особливість ДСДК - наявність ланцюга порозрядного переносу зі старшого розряду цифрової частини в знаковий розряд.

Двійковий суматор оберненого коду (ДСОК) оперує зображеннями чисел у оберненому коді. Особливість ДСОК - наявність ланцюга циклічного переносу зі старшого цифрового розряду в знаковий розряд та із знакового розряду в молодший цифровий розряд.

Двійковий суматор прямого коду (ДСПК) - це суматор, у якому відсутній ланцюг порозрядного перенесення між старшим цифровим і знаковим розрядами.

Два автомати з однаковими вхідними і вихідними алфавітами називаються *еквівалентними*, якщо після установки їх у початковий стан їхні реакції на будь-яке вхідне слово збігаються.

Два висловлювання називаються *еквівалентними*, якщо істинності їх однакові.

Демультиплексором називається комбінаційний цифровий пристрій, призначений для керування передачею даних від одного джерела інформації до декількох вихідних каналів.

Дешифратор - операційний елемент, що виконує функцію перетворення деякого *n*- розрядного двійкового коду в унітарний код «один з *N*».

Диз'юнктивна нормальна форма (ДНФ) - це диз'юнктивне об'єднання мінтермів різних рангів, включаючи ранг, рівний одиниці.

Диз'юнктивний терм (макстерм) - це логічна функція, що зв'язує знаком диз'юнкції всі змінні в прямій чи інверсній формі, тобто літерали.

Диз'юнкція (логічне додавання) або функція АБО (ИЛИ, OR) - це функція f7(x1,x2), що істинна тоді, коли істинна хоча б одна з її змінних.

Діаграма переходів представляє собою коло, розділене на *k* рівних частин, з яких кожна частина представляє стан входів автомату (відповідно виходів).

Діапазон представлення чисел в заданій системі числення - це інтервал числової вісі між максимальним та мінімальним числами, значення яких, як бачимо, залежить від довжини розрядної сітки.

До *детермінованих* належать автомати, у яких виконана умова однозначності переходів: автомат, що знаходиться в деякому стані ai, під дією будь-якого вхідного сигналу zj не може перейти більш, ніж в один стан.

Довжиною ДНФ назвемо число елементарних кон'юнкцій, що утворюють цю ДНФ.

Досконала нормальна форма відрізняється від нормальної форми тим, що завжди містить терми тільки максимального рангу і дає однозначне представлення функції.

Достовірність - вірність інформації, яка не викликає сумнівів.

Доступність інформації - це міра можливості одержати ту чи іншу інформацію.

Дуже малі числа представляються в машині зображенням, названим *машинним нулем.*

Елементами в комп'ютерній схемотехніці називаються найменші неподільні мікроелектронні схеми (вироби), призначені для виконання логічних операцій або зберігання біту інформації.

Елементарні дії, виконувані в комп'ютерах за один машинний такт, називаються *мікрооперaціями*.

Елементи (сивмоли) алфавіту, які використовуються для запису чисел в деякій системі числення, називають *цифрами*. Кожній цифрі даного числа однозначно співставляється її *кількісний (числовий) еквівалент*.

Емоційність - властивість інформації викликати різні емоції в людей.

Енергозалежною називають пам'ять, вміст якої втрачається при вимкненні комп'ютера.

Енергонезалежною називається пам'ять, вміст якої не втрачається при вимкненні комп'ютера.

З точки зору кібернетики, *інформація* - це розпізнаний кібернетичною системою сигнал або комплекс сигналів (образ), який зменшує кількість варіантів вибору нею чергової дії (команди).

Завадостійкість(перешкодостійкість) –це здатність елемента правильно функціонувати при наявності перешкод.

*Задача аналізу*полягає у визначенні статичних і динамічних властивостей комбінаційної схеми.

*Задача синтезу*полягає у побудові з заданого набору логічних елементів комбінаційної схеми, що реалізує задану систему булевих функцій.

Запам'ятовуючий пристрій - це внутрішня пам'ять процесора.

Запис числа в деякій системі числення часто називають *кодом числа.*

Згідно Закону України «Про інформацію», *інформація* - це документовані або публічно оголошені відомості про події та явища, що відбуваються у суспільстві, державі та навколишньому природному середовищі.

Згідно концепції Шеннона, *інформація* - це знята невизначеність, тобто відомості, які повинні зняти в тій чи іншій мірі існуючу до їх одержання невизначеність, розширити розуміння об'єкту корисними відомостями.

Зовнішня пам'ять призначена для тривалого зберігання великих обсягів інформації незалежно від того ввімкнено чи вимкнено комп'ютер.

Імпліканта g булевої функції f, що є елементарною кон'юнкцією, *називається простою*, якщо жодна частина імпліканти g не є імплікантою функції f.

Інакше це буде *імовірнісний автомат*, у якому при заданому стані аі та заданому вхідному сигналі zj можливий перехід із заданою ймовірністю в різні стани.

Іноді для зображення знаків чисел відводять два розряди, тоді такі коди називають *модифікованими кодами*.

Іноді зі скороченої ДНФ можна забрати одну чи декілька простих імплікант, не порушуючи еквівалентності вихідної функції. *Такі прості імпліканти назвемо зайвими.*

Інформаційні сигнали визначають новий стан тригера і присутні в будь-яких тригерах.

Інформацію можна вважати *повною*, якщо вона містить мінімальний, але достатній для прйиняття вірного рішення набір показників.

Інформацію можна розглядати також як результат інтелектуальної (аналітико- синтетичної чи еврістичної) діяльності певної людини щодо подання відомостей, повідомлень, сигналів, кодів, образів тощо.

*Керуючий автомат (КА)*генерує запропоновану мікропрограмою послідовність керуючих сигналів, відповідно до значень логічних умов. Інакше кажучи, КА задає порядок виконання дій в ОА згідно алгоритму виконання операцій.

Керуючий пристрій організовує процес виконання програм та координує взаємодію всіх пристроїв ЕОМ під час її роботи.

Кількість розрядів у запису числа називається *розрядністю числа* і співпадає з його *довжиною*.

Кількість тригерів, тобто двійкових розрядів у регістрі або комірці пам'яті визначає *довжину слова*, характерну для даного комп'ютера, а сукупність цих двійкових розрядів називається *розрядною сіткою*.

Клавіатура - стандартний пристрій, призначений для ручного введення інформації.

*Коефіцієнт об'єднання по входу(Коб)*визначає максимально можливе число входів логічного елемента, іншими словами, функцію скількох змінних може реалізувати цей елемент.

*Коефіцієнт розгалуження по виходу(Кроз)*показує, на скільки логічних входів може бути одночасно навантажений вихід даного логічного елемента.

Комбінаційний суматор виробляє вихідні сигнали суми і переносу, обумовлені комбінацією одночасно поданих на входи суматора цифр, що додаються.

Комбінаційними називаються функціональні вузли (блоки), логічний стан виходів яких залежить тільки від комбінації логічних сигналів на входах в певний момент часу.

Кон'юнкція (логічне множення) або функція ТА (И, AND) - це функція f1(x1,x2), яка істинна тоді, коли *всі її змінні одночасно істинні*.

Кон'юнкція x1 x2 ... xn називається елементарною, якщо в цій кон'юнкції кожна змінна зустрічається не більше одного разу.

Конфігурацією комп'ютера називають фактичний набір компонентів ЕОМ, які складають комп'ютер.

Кон'юнктивна нормальна форма (КНФ) - це кон'юнктивне об'єднання макстермів, що включає в себе макстерми різних рангів.

Кон'юнктивний терм (мінтерм) - це логічна функція, що зв'язує знаком кон'юнкції змінні в прямій чи інверсній формі, тобто літерали.

Крім автоматів Мілі і Мура, іноді виявляється зручним користуватися сполученою моделлю автомата, так званим *С-автоматом.*

Лічильник - операційний елемент, що реалізує мікрооперацію лічби, яка складається в зміні стану лічильника (значення збереженого слова) на 1. Крім того, лічильник може виконати і такі мікрооперації: установка в 0 і прийом довільного числа.

Логіка — це наука про форми і закони правильного мислення.

Логічна (булева) змінна – це така величина x, що може приймати тільки два значення: 0 *чи* 1.

Логічний оператор схеми – це елементарна логічна функція, за допомогою якої описується робота схеми.

Логічними елементами - елементи з двома станами називають.

Мінтерм називають також *конституентою одиниці*, тому що *ця функція дорівнює 1 тільки тоді, коли всі її аргументи одночасно дорівнюють одиниці*.

Максимальний рівень елементів *r*визначає кількість рівнів КС, яке називається *рангом схеми.*

Мікроконтролер (англ. microcontroller), або *однокристальна мікроЕОМ* — виконана у вигляді мікросхеми спеціалізована мікропроцесорна система, що включає мікропроцесор, блоки пам'яті для збереження коду програм і даних, порти вводу-виводу і блоки зі спеціальними функціями (лічильники, компаратори, АЦП та інші).

Мікропрограмування-це спосіб опису функційопераційних пристроїв безвідносно до технічних засобів, які використовуються для їх реалізації.

Мікропроцесор (англ. microprocessor) — інтегральна схема, яка виконує функції центрального процесора (ЦП) або спеціалізованого процесора. Мікропроцесором називається програмований електронний пристрій для обробки інформації, виконаний у вигляді однієї чи кількох мікросхем високого ступеня інтеграції.

Мінімальна форма представлення ФАЛ - це така форма, що містить мінімальну кількість термів, які мають мінімальні ранги.

Мінімальна форма представлення ФАЛ - це така форма, що містить мінімальну кількість термів, які мають мінімальні ранги.

Мультиплексор - операційний елемент, що виконує функцію почергової комутації (переключення) інформації від одного з *n* входів на спільний вихід.

На відміну від них є автомати, для яких вихідні сигнали залежать лише від стану автомату та не залежать від значень вхідних сигналів, - *автомати Мура.*

Набори, що забезпечують на виході ЛС логічну 1, утворять так зване *одиничне покриття* . 1 С

Найпростішими називають перетворення, що полягають у заміні кожної букви повідомлення, представленого у почактовому алфавіті певною комбінацією букв кінцевого алфавіту . X Y

Накопичуючим суматором називається суматор, що здійснює додавання слів *A* і *B* при подачі їх на суматор одного за іншим.

Напівсуматор — логічна схема з двома входами та двома виходами, який виконує операцію арифметичного додавання двох однорозрядних чисел A та B у відповідності до наступного правила: при будь-яких наборах сигналу A та B на виході сигналу суми S'

Неповністю визначена логічна функція *n* змінних – це функція, задана на числі наборів меншому 2*n*.

Непозиційна система числення - це система, для якої значення символу, тобто цифри, не залежить від його положення в числі.

Нормалізованим вважається число з ПК, мантиса якого є правильним дробом і перша цифра після коми відмінна від 0.

Об'єктивність інформації характеризує її незалежність від чиєїсь думки або свідомості, а також від методів одержання.

Однокристальний мікрокомп'ютер є різновидом універсальних комп‘ютерів. Він містить усі стандартні пристрої, необхідні для реалізації цифрової системи мінімальної конфігурації, а саме - процесор, пам‘ять команд, пам‘ять даних, внутрішній тактовий генератор.

*Операційний автомат (ОА)*служить для збереження слів інформації, виконання набору мікрооперацій і обчислення значень логічних умов, тобто ОА є структурою, організованою для виконання дій над інформацією.

*Операційний автомат (ОА)*служить для збереження слів інформації, виконання набору мікрооперацій і обчислення значень логічних умов, тобто ОА є структурою, організованою для виконання дій над інформацією.

Операція приведення числа до нормалізованого вигляду називається *нормалізацією*.

Основа або базис q природньої позиційної системи числення - це кількість знаків або сивмолів, використовуваних для зображення числа в даній системі.

Пам'яттю комп'ютера називають сукупність пристроїв для зберігання програм, введеної інформації, проміжних результатів та вихідних даних.

Перетворення називають еквівалентним, якщо кінцева інформація повністю та однозначно визначає початкову.

Перетворювачем коду називається комбінаційний пристрій, призначений для зміни виду кодування інформації.

Перехід елементу з одного стану в інший називають його *перемиканням.*

Під *ціною логічної функції* розуміється кількість букв, які входять в її запис.

Повний однорозрядний суматор-це пристрій, що здійснює додавання за *mod 2*відповідних розрядів *(X1,X2)*двійкових чисел з урахуванням переносу *(Pm)*у даний розряд із сусіднього молодшого розряду суми.

Погляд на інформацію з точки зору її споживачів окреслює таке поняття: «*інформація* — це нові відомості, які прийняті, зрозумілі і оцінені її користувачем як корисні; це нові знання, які отримує споживач (суб'єкт) у результаті сприйняття і переробки певних відомостей».

Позиційна система числення - це система, в якій значення кожної цифри залежить від її числового еквіваленту та від її місця (позиції) в числі, тобто один символ (цифра) може приймати різні значення.

Поняття інформації є найбільш важливим, загальним і суперечливим поняттям для сучасної науки. Інформація є інтуїтивно зрозумілим абстрактним поняттям, що може набувати різний зміст у різних галузях знань.

При *формі з плаваючою комою* запис одного числа може приймати різний вигляд в залежності від обмежень, що накладаються на форму.

Пристрій виведення виводить інформацію з комп'ютера і перетворює її з внутрішнього представлення у зовнішнє.

Пристрої введення перетворюють інформацію у зрозумілу машині форму, після чого комп'ютер може її обробляти та запам'ятовувати.

Пристрої виведення перекладають інформацію з машинного представлення в образи, зрозумілі людині.

Пристрої, призначені для обробки та перетворення цифрової інформації, називаються *цифровими автоматами.*

Програмована логічна інтегральна схема (ПЛІС) — електронний компонент, що використовується для створення цифрових інтегральних схем. На

відміну від звичайних цифрових мікросхем, логіка роботи ПЛІС не визначається при виготовленні, а задається за допомогою програмування.

Процес виконання операцій у пристрої описується у формі алгоритму, що представляється в термінах мікрооперацій і логічних умов і називається *мікропрограмою*.

Процесор - це центральний пристрій комп'ютера, який виконує операції по обробці даних та керує периферійними пристроями комп'ютера.

Ранг терма - *r*, визначається кількістю літералів, що входять у даний терм.

Рангом елементарної кон'юнкції називається число букв, що утворюють цю кон'юнкцію.

Регістр - це ОЕ, що служить для запам'ятовування слів і забезпечує в загальному випадку виконання наступних мікрооперацій: установка регістра в 0 (скидання); прийом слова з іншого регістра, шини і т.д.; передача слова на інший регістр, шину і т.д.; перетворення кодів збережених слів в інверсні коди; зсув збереженого слова ліворуч або праворуч на необхідну кількість розрядів.

Рекурентна булева функція (РБФ) — логічна функція, яка залежить як від поточних значень вхідних змінних, так і від попередніх значень власне функції *yt-i*. і tx

Різні мікрооперації виконуються елементарними ОА - так званими *операційними елементами (ОЕ),* що є складовими частинами основного ОА.

Розрядність процесора - це максимальна кількість бітів інформації, які можуть оброблятись та передаватись процесором одночасно.

Секвенціальна логіка - це логіка пам'яті цифрових пристроїв. Така логіка може називатись також послідовнісною. У секвенціальній логіці вихідне значення залежить не лише від поточних вхідних впливів, але й від передісторії функціонування цифрового пристрою. Іншими словами, секвенціальна логіка передбачає наявність пам'яті, яка в комбінаційній логіці не передбачена.

*Серією (системою, комплексом)логічних елементів ЕОМ*називається призначений для побудови цифрових пристроїв функціонально повний набір логічних елементів, поєднаний спільними електричними, конструктивними і технологічними параметрами, що використовує однаковий спосіб представлення інформації, однаковий тип міжелементних зв'язків.

Синхросигнал не є обов'язковим і вводиться в тригерах з метою фіксації моменту переходу тригера в новий стан, що задається інформаційними входами.

Система команд процесора - це набір окремих операцій, які може виконати процесор даного типу.

Система логічних функцій називається *функціонально повною*, якщо за допомогою функцій, що входять у цю систему, застосовуючи операції суперпозиції і підстановки, можна одержати будь-яку як завгодно складну логічну функцію.

Система числення - це сукупність прийомів та правил для запису чисел цифровими знаками.

*Складність (ціна) за Квайном*визначається сумарним числом входів логічних елементів у складі схеми.

*Складність схеми*оцінюється кількістю елементів, які складають схему.

Скорочена ДНФ булевої функції називається *тупиковою формою*, якщо в ній відсутні зайві прості імпліканти.

Стан as автомата *називається стійким*, якщо для будь-якого стану ai і вхідного сигналу zj таких, що □(ai,zj)=as, має місце □(as,zj)=as, тобто стан стійкий, якщо потрапивши в цей стан під дією деякого сигналу zj, автомат вийде з нього тільки під дією іншого сигналу zk, відмінного від zj.

Стандарт ISO/IEC 2382: 2000. Information Technology Vocabulary надає наступне визначення: «*Інформація* - будь-який факт, поняття або значення, отримані з даних, а також контекст, обраний зі знань, або контекст, асоційований із знаннями».

Сукупність значень аргументів логічної функції називається набором (або *точкою*) і може позначатися зокрема, як *x*1, *x*2,..., *x*n, де *xi* дорівнює нулю чи одиниці.

Сукупність правил переходу автомату з одного стану в інший в залежності від вхідної інформації та внутрішніх станів автомату називається *алгоритмом перетворення (обробки) інформації.*

Суматор - ОЕ, що виконує додавання кодів чисел.

Суперпозицією називається підстановка в логічну функцію замість її аргументів інших логічних функцій.

Схема Sназивається *комбінаційною*, якщо кожну з *n*функцій її виходів *Y1,Y2, ..., Yn*можна представити як булеву функцію вхідних змінних *X1, X2, ..., Xm.*

Тактова частота у МГц - рівна кількості тактів на секунду. Такт - це проміжок часу між початком подачі поточного імпульсу ГТЧ і початком подачі наступного.

Терм - це група логічних змінних у прямій чи інверсній формі, тобто група літералів, об'єднаних однаковим знаком логічного зв'язку: логічного додавання або ж логічного множення.

Типовими *функціональними вузлами* комп'ютерів називаються мікроелектронні схеми, призначені для виконання однієї або декількох мікроопераций.

Тригер - найпростіша цифрова схема послідовнісного типу. Тригер - це клас електронних пристроїв, які мають здатність довго знаходитись в одному з двох стійких станів та чергувати їх під впливом зовнішніх сигналів. Тригер – це пристрій, що має два стійких стани, у які він переходить під дією визначених вхідних сигналів.

У *послідовнісних схемах* логічне значення виходів визначається як комбінацією вхідних сигналів, так і чтаном пам'яті схеми в даний момент часу.

формується результат додавання по модулю два, на виході сигналу переносу P' у всіх випадках буде 0, крім A=B=1, тоді P'=1.

Функціонально повною системою булевих функцій (ФПСБФ) називається сукупність таких булевих функцій {*f1, f2, ..., fk*}, що довільна булева функція *f* може бути записана у вигляді формули через функції цієї сукупності.

Функція (стрілка) Пірса (Вебба) або функція АБО-НЕ - це функція f8(x1,x2), яка істинна тільки тоді, коли всі змінні хибні.

Функція (штрих) Шеффера або функція ТА-НЕ - це функція f14(x1,x2), яка хибна тоді, коли всі змінні істинні.

Функція f(x1, x2, ..., x*n*) *називається логічною (перемикальною, булевою), якщо вона, як і її аргументи xi, може приймати тільки два значення:* 0 *чи* 1 (хибне чи істинне).

Функція імплікації або функція ЯКЩО-ТО (IF-THEN) це функція f13(x1,x2), яка помилкова тоді і тільки тоді, коли x1 істинне та x2 помилкове.

Функція керуючого автомата–цеоператорна схема алгоритму (мікропрограми), функціональними операторами якої є символи *y*1,...,*yt*, що ототожнюються з мікрооперaціями, і логічними умовами, в якості яких використовуються булеві змінні x1,...,xL.

Цифровий автомат - це приклад пристрою, реакція якого залежить не лише від значень входів, але й від стану в попередній момент часу.

Цифровий компаратор — цифровий пристрій комібнаційного типу, що призначений для порівняння двох чисел у двійковому або двійково-десятковому коді.

Цифрові автомати, у яких вихідні сигнали визначаються вхідними сигналами та станом автомату в попередній момент часу називають цифровими *автоматами Мілі.*

Цілком визначеним називається абстрактний цифровий автомат, у якого функція переходів і функція виходів визначені для всіх пар (ai, zj).

Часова булева функція (ЧБФ) - це логічна функція *y*= φ(*x*1 *x*2*, ... , xn, t),* яка приймає значення {0, 1} при *0* ≤ *t* ≤ *s*-1, де *s* — кількість інтервалів автоматного часу.

Частковим називається абстрактний автомат, у якого або функція переходів, або функція виходів, або обидві ці функції визначені не для всіх пар (ai, zj).

Числа, представлені в ЦА, називають *операндами*.

Швидкодія (обчислювальна потужність) - це середня кількість операцій процесора на секунду.

*Швидкодія комбінаційної схеми*оцінюється величиною, оберненою до максимальної затримки сигналу при проходженні його від входу схеми до виходу, тобто визначається проміжком часу від моменту надходження вхідних сигналів до моменту установлення відповідних значень вихідних.

*Швидкодія*логічних елементів є одним з найважливіших параметрів і характеризується часом затримки поширення сигналу.

Шина - це сукупність ланцюгів, призначених для передачі слова інформації.

Шифратором називається пристрій, який перетворює вхідний сигнал одного із його входів у кодову комбінацію на його виходах.

Як філософську категорію, *інформацію* розглядають як один з атрибутів матерії, що відбиває її структуру.

Якість інформації - це ступінь її відповідності потребам споживачів.

Якщо місце коми в розрядній сітці машини заздалегідь фіксовано, то така форма представлення чисел називається представленням *з фіксованою комою*.

Якщо число, отримане в результаті обчислень, перевищує за абсолютним значенням максимальне машинне слово, яке можна представити розрядною стікою використовуваної ширини, то відбувається *переповнення розрядної сітки* комп'ютера (цифрового автомату).

ПЕРЕЛІК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

Базова

1. Бабич М.П., Жуков І.А. Комп’ютерна схемотехніка. Навчальний посібник. – К.: НАУ, 2002. – 508 с.
2. Жабін В.І., Жуков І.А., Климено І.А., Ткаченко В.В. Прикладна теорія цифрових автоматів. Навчальний посібник. Київ, Національний авіаційний університет, 2007р., 363с.
3. Лупенко С.А., Тиш Є.В. Прикладна теорія цифрових автоматів. Навчальний посібник. - Тернопіль: ТНТУ ім. І. Пулюя, 2017. – 247 с.
4. Матвієнко М.П. Комп'ютерна схемотехніка. Навчальний посібник - Київ: ТОВ «Центр навчальної літератури», 2012. - 190 с.
5. Самофалов К.Г. и др. Прикладная теория цифровых автоматов.- Киев: Высш. школа, 1987.

Допоміжна

1. Бойко В. І., Багрій В. В. Цифрова схемотехніка. – К: ІЗМН, 2001.- 228 с
2. Глушков В.М. Синтез цифровых автоматов. – М.: Государственное издательство физико-математической литературы, 1962.
3. Захаров Н.Г., В.Н. Рогов Синтез цифровых автоматов: учебное пособие. – Ульяновск: УлГТУ, 2003.
4. Кочубей О.О., Сопільник О.В. Прикладна теорія цифрових автоматів: Логічні основи: навчальний посібник – Д.: РВВ ДНУ; вид-во ДНУ, 2009. – 264 с.
5. Рябенький В.М., Жуйков В.Я., Ямненко Ю.С., Заграничний А.В. «Схемотехніка: Пристрої цифрової електроніки», - Київ, 2016.
6. Савельев, А.Я. Прикладная теория цифровых автоматов: учебник для вузов – М.: Высшая школа, 1987. – 272с.