1. **КОМП’ЮТЕРНА ЛОГІКА:**

**основні поняття та застосування**

* *Інформатика та обчислювальна техніка: базові поняття автомату, алгоритму, інформації (комп’ютер)*
* *Комп’ютерна логіка як фундаментальний розділ інформатики*
* *Основні поняття комп’ютерної логіки та теорії автоматів*
* *Застосування теорії цифрових автоматів*

**1.1. Основні поняття комп’ютерної логіки та теорії автоматів**

Комп’ютерна логіка – це навчальна дисципліна, що охоплює логічні, математичні та технічні основи, базові принципи, поняття та моделі обчислювальних та управляючих цифрових систем. Назва «комп’ютерна логіка» може вживатися у широкому та вузькому розумінні. У широкому розумінні комп’ютерна логіка охоплює собою всю множину логік, закономірностей, моделей та методів функціонування комп’ютерних систем та їх компонент. За такого трактування комп’ютерної логіки слушно пов’язати всі ці логіки із різними рівнями їєрархічної організації комп’ютерної системи, зокрема, це можна реалізувати прив’язавшись до багаторівневої моделі організації комп’ютерної системи Таненбаума [].

Комп’ютерна логіка досліджує ієрархію моделей, що описують фундаментальні закономірності функціонування комп’ютерних систем як програмно-апаратних засобів різної складності та призначення. Комп’терна логіка вивчає моделі, методи аналізу та синтезу логічних комбінаційних схем, цифрових автоматів, що є базовими логічними моделями сучасних комп’ютерних систем та систем контролю і управління.

Тому цілком слушно виділити такі розділи комп’ютерної логіки як математична логіка (алгебра логіки Буля, алгебра логіки Жегалкіна і т.п.), багатозначна логіка (нечітка логіка, континуальна логіка), теорія цифрових автоматів, теорія алгоритмів, теорія формальних систем, формальні граматики і т.п.

Комп’ютерну логіку можна викладати грунтуючись як мінімум на трьох підходах, а саме, абстрактно-теоретичному, прикладному та у певному поєднанні першого та другого підходів.

В результаті вивчення дисципліни студенти повинні:

знати:

- як працює комп’ютер на самому нижчому рівні його архітектури;

- з якими числами оперує ПК;

- які арифметичні дії виконує комп’ютер та як;

- за допомогою яких логічних елементів проходить виконання тієї чи іншої операції.

- як можна скоротити число логічних дій при виконанні конкретної задачі.

вміти:

- перетворювати числа з однієї системи числення в іншу;

- виконувати арифметичні операції з цими числами;

- представляти ці числа у різних форматах даних.

- використовувати необхідні логічні елементи для виконання певної;

- будувати схеми з набору логічних елементів,

- спрощувати виконання логічних операцый та кількість логічних елементів для поставленої задачі.

Дана дисципліна є базовою для студентів даної спеціальності, оскільки основним її завданням є розяснення того, як розуміє та обробляє інформацію . комп’ютер.

У вузькому, більш прикладному, розумінні поняття «комп’ютерна логіка» охоплює собою множину питань, що стосуються низькорівневих аспектів, логік, моделей функціонування обчислювальних систем та засобів автоматизації і управління. У цьому сенсі дисципліна «Комп’ютерна логіка» вбирає у себе питання, які раніше охоплював курс «Прикладна теорія цифрових автоматів». Власне у цьому контексті і побудовано навчальний посібник «Комп’ютерна логіка».

Фундаментальною логічною моделлю комп’ютерної системи, що вивчається у рамках предмету «Комп’ютерна логіка» є поняття автомату. Термін «автомат» має декілька смислових аспектів. По-перше, автомат – це пристрій (система), який без участі людини здійснює відбір, опрацювання, передавання та зберігання інформації (даних, повідомлень, сигналів) відповідно до закладеного у нього алгоритму. З іншого боку, автомат – це математична модель реальної технічної системи (наприклад, комп’ютера чи певного його вузла, елемента). У цьому випадку автомат розглядається як деяка «чорна скринька», що має скінченну кількість входів та виходів, а також деяку множину внутрішніх станів. У цьому контексті, *теорія цифрових автоматів* вивчає математичні моделі перетворювачів (автоматів) дискретної інформації, яка подана у цифровій формі. З певної точки зору такими перетворювачами є як реальні пристрої (обчислювальні машини, живі організми), так й абстрактні системи (наприклад, *формальна система* – це сукупність абстрактних об’єктів, не пов’язаних із зовнішнім світом, в яких подано правила оперування множиною символів у строго синтаксичному трактуванні без урахування смислового навантаження, тобто семантики).

З практичної точки зору, *автомат* – це деякий пристрій, призначений для перетворення інформації. У цьому сенсі автомат можна вважати інформаційною системою. Отже, поняття «автомат» суттєво пов’язане з поняттям «інформація».

Термін «інформація» має багато означень. У широкому сенсі *інформація* – відображення реального світу. Існує означення інформації й у вузькому сенсі: *інформація* – будь-які відомості, що є об’єктом відбору, передавання, перетворення та зберігання.

Важливе питання теорії інформації – встановлення міри, кількості та якості інформації. Інформаційні міри, переважно, розглядаються у трьох аспектах: структурному, статистичному та семантичному.

У *структурному аспекті* розглядається будова масивів інформації та їх зміна простим підрахунком інформаційних елементів або комбінаторним методом. Структурний підхід застосовується для оцінювання можливостей інформаційних систем незалежно від умов їх використання.

При *статистичному підході* використовується поняття ентропії як міри невизначеності, що враховує ймовірність появи того чи іншого повідомлення.

*Семантичний підхід* дає змогу виділити корисність або цінність інформаційного повідомлення.

Інформація надходить у систему передавання даних у формі повідомлень. Під *повідомленням* розуміють сукупність знаків або первинних сигналів, що містять інформацію. Джерело повідомлень, у загальному випадку, утворює сукупність джерела інформації (досліджуваного або спостережуваного об’єкта) й первинного перетворювача (давача, людини-оператора тощо), що сприймає інформацію про процес, що протікає в ньому.

Розрізняють дискретні та неперервні повідомлення.

*Дискретні повідомлення* формуються в результаті послідовного видавання джерелом повідомлень окремих елементів – *знаків, символів, букв*. Множину різних знаків називають *алфавітом* джерела повідомлення, а число знаків – *об’ємом алфавіту*. Якщо повідомлення є дискретними, то пристрій обробки такої інформації називають дискретним пристроєм або *дискретним автоматом*.

Неперервні повідомлення не поділені на елементи. Вони описуються функціями часу, що означені на континуальних множинах.

Для передавання повідомлення каналом зв’язку йому ставлять у відповідність певний сигнал. Під *сигналом* розуміють фізичний процес, що відображає, переносить повідомлення.

Перетворення повідомлення в сигнал, зручний для передавання каналом зв’язку, називають *кодуванням* у широкому сенсі. Операцію відновлення повідомлення по прийнятому сигналу називають *декодуванням*.

Закодовані таким чином сигнали називають *цифровими сигналами*. Подання сигналу в цифровій формі практично завжди дає суттєву перевагу при передаванні, зберіганні та опрацюванні інформації.

У сучасних дискретних автоматах прийнято ототожнювати букви довільних алфавітів з цифрами тієї або іншої системи числення. Тому дискретні автомати прийнято називати *цифровими автоматами*.

Як правило, на практиці вдаються до операції представлення початкових знаків в іншому алфавіті з меншим числом знаків. Пристрій, що виконує таку операцію, називають таким, що кодує, або *кодером*. У такому разі кожному знаку відповідає деяка послідовність символів, яку називають *кодовою комбінацією*.

Кількість символів у кодовій комбінації називають її *значущістю*, кількість ненульових символів – *вагою*.

Для операції зіставлення символів зі знаками початкового алфавіту використовують термін "декодування". Технічна реалізація цієї операції здійснюється декодуючим пристроєм, або *декодером*.

Найтісніше теорія автоматів пов’язана з *теорією алгоритмів*. Це пояснюється тим, що автомат перетворює дискретну інформацію по кроках у дискретні моменти часу та формує результуючу інформацію по кроках заданого алгоритму. Ці перетворення можливі за допомогою технічних та/або програмних засобів. При аналізі автоматів вивчають їх поведінку при різних збурюючих впливах та мінімізують число станів автомата для роботи згідно з заданим алгоритмом. Такий автомат називають *абстрактним*, оскільки абстрагуються від реальних фізичних вхідних та вихідних сигналів, розглядаючи їх просто як букви деякого алфавіту. При синтезі автоматів формують систему з елементарних автоматів, що є еквівалентною заданому абстрактному автомату. Такий автомат називають *структурним*.

Особливе місце в теорії автоматів займає поняття скінченного автомата. *Скінченний автомат* – математична абстракція, що дає змогу описувати шляхи зміни стану об’єкта залежно від його поточного стану та вхідних даних, за умови, що загальна можлива кількість станів та множина вхідних сигналів скінченні. Скінченний автомат є частковим випадком абстрактного автомата.

До складу цифрових автоматів обов’язково входять запам’ятовуючі елементи (елементи пам’яті). Вихідні сигнали в таких автоматах формуються залежно не тільки від вхідних сигналів, але й від деякої передісторії – *станів автомата*, тобто сигналів, які надійшли на входи автомата раніше.

Ще однією особливістю цифрових автоматів є властивість *стрибкоподібного переходу* автомата з одного стану в інший. Такий перехід можна вважати миттєвим, хоча для будь-якого реально існуючого автомата має місце скінченна тривалість перехідних процесів. Інше припущення, яке стосується роботи цифрового автомата, полягає в тому, що після переходу автомата в довільний стан перехід у наступний стан може бути здійсненим не раніше, ніж через деякий фіксований проміжок часу , який називають *інтервалом дискретності автомата*. Таке припущення дає змогу розглядати функціонування цифрового автомата в *дискретному часі*. При побудові автоматів з дискретним автоматним часом розрізняють синхронні та асинхронні автомати.

У *синхронних автоматах* моменти часу, в які необхідна зміна стану автомата, визначаються спеціальним пристроєм – генератором синхронізуючих імпульсів. Сусідні моменти часу при цьому поділені рівними часовими проміжками.

В *асинхронних автоматах* моменти переходів з одного стану в інший заздалегідь не визначаються та можуть здійснюватися через нерівні між собою проміжки часу.

Як вже зазначалося, зміни станів цифрового автомата здійснюються під впливом вхідних сигналів, які виникають поза автоматом і передаються в автомат скінченною кількістю вхідних каналів. Вхідний сигнал розглядається як миттєвий, хоча насправді він має скінченну тривалість. Зазначимо, що реальний фізичний вхідний сигнал, що викликає зміну стану автомата, у момент часу *t* може закінчитися до настання цього моменту, однак він відноситься саме до поточного моменту часу *t*, а не до попереднього .

Результатом роботи цифрового автомата є генерування вихідних сигналів, які генеруються по скінченній кількості вихідних каналів. Реальний фізичний вихідний сигнал , що належить до моменту часу , з’являється завжди після вхідного сигналу , що відповідає цьому ж моменту часу. Стосовно моменту часу  переходу автомата зі стану  у стан , то сигнал  може фактично з’являтися або раніше, або пізніше цього моменту.

У першому випадку приймається, що вихідний сигнал  однозначно визначається вхідним сигналом  та станом  автомата в попередній момент часу, в другому випадку сигнал  однозначно визначається парою . Вважають, що для будь-якого моменту часу завжди має місце лише одна із цих можливостей (одночасно для всіх переходів).

Цифрові автомати, в яких вихідний сигнал  визначається парою , називають *автоматами першого роду*, а автомати, в яких сигнал  визначається парою  – *автоматами другого роду*.

Цифровий автомат (першого або другого роду) називається *правильним*, якщо вихідний сигнал  визначається одним лише його станом ( або ) та не залежить явно від вхідного сигналу .

Автомати першого роду зазвичай називають *автоматами Мілі*, а автомати другого роду – *автоматами Мура*.

Загальна теорія цифрових автоматів за наведених вище припущень розбивається на дві великі частини: *абстрактну теорію цифрових автоматів* та *структурну теорію цифрових автоматів*. Відмінність між ними полягає в тому, що в абстрактній теорії не враховується структура як самого автомата, так і структури його вхідних та вихідних сигналів. Вхідні та вихідні сигнали розглядаються при цьому просто як букви двох фіксованих для цього автомата алфавітів: вхідного та вихідного. Не цікавлячись способом побудови автомата, абстрактна теорія вивчає лише ті переходи, які зазнає автомат під впливом вхідних сигналів, та ті вихідні сигнали, які він при цьому видає.

На противагу абстрактній теорії, структурна теорія автоматів враховує структуру автомата та його вхідних та вихідних сигналів. У структурній теорії вивчаються способи побудови автоматів з кількох елементарних автоматів, способи кодування вхідних та вихідних сигналів елементарними сигналами, що передаються по реальних вхідних та вихідних каналах. Структурна теорія автоматів є продовженням та подальшим розвитком абстрактної теорії.

**1.2. Застосування теорії цифрових автоматів**

Теорія цифрових автоматів займається вивченням процесів, що протікають у автоматах різного роду, та загальних закономірностей, яким вони підпорядковані, широко застосовуючи для цього апарат алгебри, математичну логіку, комбінаторний аналіз та теорію ймовірності.

Теорія автоматів знаходить застосування як в математиці, так і в розв’язанні практичних задач. Наприклад, засобами теорії автоматів доводиться розв’язок деяких формальних обчислень. Застосування методів та понять теорії автоматів до вивчення формальних та природних мов призвело до виникнення математичної лінгвістики (*математична лінгвістика* – математична дисципліна, предметом якої є розроблення формального апарата для описування будови природних та деяких штучних мов). Поняття автомата може служити модельним об’єктом в найрізноманітніших задачах, завдяки чому можливе застосування теорії автоматів у різних наукових і прикладних дослідженнях.

Існує багато об’єктів керування, що пов’язані з великою відповідальністю: ядерні та хімічні реактори, комплекси промислового, оборонного, космічного призначення, гірська справа. Успіх у роботі з ними прямо залежить від чіткості та злагодженості дій, від уміння приймати вивірені рішення та грамотно аналізувати ситуацію, від можливості однозначної інтерпретації інформації. Різна природа фізичних процесів, що протікають в об’єктах, складний характер взаємодії між ними та системами керування зумовлює труднощі розроблення, алгоритмізації та програмування задач керування. Виникають труднощі, які пов’язані з необхідністю досягнення наочності та структурованості. Для вирішення цих завдань використовується розвинений математичний апарат теорії автоматів.

Опис логіки поведінки (за яких умов необхідно виконати ті або інші дії) при автоматному підході структурований. Ця властивість робить автоматний опис складної поведінки наочним та зрозумілим. Коректність роботи при використанні автоматів закладається ще на етапі проектування завдяки графічному представленню, тобто:

* наочно подається поведінка керуючих автоматів (графічно, таблично) та композицій з них;
* відображаються бажані стани;
* відображається динаміка та умови переходів автомата зі стану в стан; можна легко побачити можливі помилки в проектуванні, такі, як відсутність деякого переходу, недоступність стану тощо.

Усе це призводить до чіткого розуміння роботи пристрою. Процеси керування, проектування можуть бути подані у вигляді елементів з передбачуваною поведінкою.

Сфери застосування теорії автоматів вражають своїм розмахом і не обмежуються вузькою спрямованістю та спеціалізацією. Розглянемо деякі з них.

**1.2.1. Виробництво сучасних цифрових систем**

Перші кроки в розвитку теорії автоматів були здійснені на початку минулого століття. У 1910 р. російський фізик П. Эренфест у передмові до книги Л. Кутюра "Алгебра логіки" вказав на можливість побудови технічних пристроїв на основі алгебри логіки. Математичне обґрунтування цього положення дав російський учений В.І. Шестаков (1935р.), пізніше цей результат був повторений в Японії – А. Накашима, М. Ханзава (1936–1938 рр.) та в США – К. Шеноном (1938 р.). У період 1942–1954 рр. інтенсивно проводилися роботи з теорії автоматів в Інституті автоматики та телемеханіки АН СРСР під керівництвом М.А. Гаврилова, який в 1950 р. опублікував першу монографію з цієї проблеми. Ці учені були засновниками теорії автоматів з боку техніки. Теорія автоматів стала основою створення обчислювальної техніки та кібернетики:

* кінець 30-х років минулого століття, інститут електротехніки АН Українскої РСР, науковий керівник С.А.Лебедєв, створення обчислювальної машини з двійковою системою числення;
* 1942 р., перший комп’ютер, розробники Атанасов, Баррі, США;
* 1948 р., проект 1-ї цифрової ЕОМ, наукові керівники Б.И. Рамєєв та І.С. Брук;
* 1951 р., Київ, Феофанія, 1-а в Європі Мала Електронна Обчислювальна Машина (МЕОМ), науковий керівник С.А. Лебедєв;
* 1952 р., ЕОМ М-1 та М-2, науковий керівник І.С. Брук;
* 1953 г., ЕОМ «Стріла», головний конструктор Б. И. Рамєєв.

Великий внесок у розвиток теорії цифрових автоматів внесли академіки B.М. Глушков, Є.А. Якубайтис та колективи очолюваних ними інститутів АН України та АН Латвії.

Теорія цифрових автоматів є основою виробництва сучасних цифрових систем: електронних обчислювальних машин, інформаційно-керуючих комплексів промислового, оборонного та космічного призначення. Пошук нових принципів побудови цифрових пристроїв, удосконалення алгоритмів виконання арифметичних задач та логічних операцій – головні завдання для спеціалістів у галузі проектування та створення комп’ютерних технологій.

**1.2.2. Програмування**

Виникає питання, чому ж скінченна автоматна модель теорії автоматів особливо актуальна зараз, коли існує величезна кількість як мов програмування, так і середовищ для розроблення програмного забезпечення?

Мають місце дві проблеми:

* непередбачувана поведінка коду програми, розробленої виключно засобами RAD (Rapid Application Development – засоби швидкого розроблення програмних засобів);
* «згасання» «культури програмування».

Приклади RAD: Borland Delphi та C, що забезпечують прискорене розроблення програм за рахунок використання об’єктно-орієнтованого та візуального програмування. Вони дають змогу не лише програмувати у звичному сенсі слова, але й фактично малювати програми (як інтерфейс, так і реалізацію), використовуючи візуальні компоненти VCL.

Будь-який візуальний об’єкт VCL характеризується рядом властивостей, методів та подій. Здавалося б, простою маніпуляцією перерахованими атрибутами можливо примушувати програму, що розробляється, робити те, що вимагає від неї програміст-розробник. Але це далеко не так.

Давно стало відомо, що VCL має тенденцію приховувати точну реалізацію певних об’єктів, тим самим не даючи стороннім особам змінювати замовчувану поведінку коду. Як показує практика, поведінка коду програми, створеної за допомогою засобів RAD, не завжди передбачувана навіть для дуже досвідченого програміста, не кажучи вже про початківця. Програма, незважаючи на "очевидність" авторського коду, завжди прагне піти своїм шляхом, потрапляючи в такі хитромудрі обробники подій, про існування яких можна навіть і не здогадуватися.

У сучасному світі об’єми та складність розроблюваних програм зростають із кожним днем, тому такий підхід різко збільшує час тестування та налагодження програмного забезпечення.

Керувати поведінкою коду дає можливість механізм теорії автоматів ще сорокарічної давності.

Стилі програмування розрізняються за базовими поняттями, використовуючи такі поняття, як "подія", "підпрограма", "функція", "клас" ("об’єкт") й т. д. Стиль програмування, заснований на явному виділенні станів та застосуванні автоматів для описування поведінки програм, носить назву "автоматне програмування", а відповідний стиль проектування програм – "автоматне проектування". Автоматне програмування можна розглядати не лише як самостійний стиль програмування, але й як доповнення до інших стилів, наприклад, до об’єктно-орієнтованого, оскільки йдеться не лише та не стільки про використання скінченних автоматів у програмуванні, скільки про метод створення програм у цілому, поведінка яких описується автоматами. Тобто як окремий компонент, так і програма в цілому може бути реалізована як автомат.

У автоматному програмуванні існує два напрями: Switch-технологія та скінченно-автоматна технологія. *Switch-технологія* – технологія розроблення систем логічного управління на базі скінченних автоматів, що охоплює процес проектування, реалізації, наладки, верифікації (перевірки), документування та супроводу.

Кодування/програмування автоматів у рамках скінченно-автоматної технології засноване на таких принципах:

* введено поняття динамічного об’єкта, який може бути наділений алгоритмом поведінки в часі;
* алгоритм поведінки об’єкта задається моделлю скінченного автомата; мова описування автомата заснована на базі табличної форми подання автоматів; логіка поведінки об’єкта (таблиця переходів автомата) відокремлена від методів автоматного об’єкта (предикатів та дій), що пов’язані з реалізацією його поведінки в часі;
* будь-які динамічні об’єкти можуть проектуватися паралельно.

Основні положення Switch-технології: пропонується зробити поняття "стан" первинним, а алгоритми подавати у вигляді графів переходів (діаграм станів), тобто подавати програму як систему взаємодіючих скінченних автоматів, що описуються графами переходів.

Графи в наочній для людини формі відображають переходи між станами, а також "прив’язку" вихідних дій та інших автоматів до станів та/або переходів. На основі Switch-технології вже розроблені програми для облаштування продажу газованої води, банкомату, світлофора, системи керування пасажирським ліфтом, система керування автомобільної сигналізації, автоматизованої системи оплати мобільного телефону, пристрої обміни валюти, пристрої для продажу проїзних квитків й т.д.

*Скінченно-автоматна* *технологія* дає змогу реалізувати ідею паралелізму. На відміну від неї технології розроблення програм «зверху вниз», «від низу до верху», структурний підхід таких можливостей або не мають, або вони обмежені. Навіть у технології об’єктно-орієнтованого програмування питання паралельної роботи об’єктів винесені за її рамки. Використання інших технологій, що базуються на відомих паралельних моделях, пов’язане з труднощами, пов’язаними якщо не з обмеженнями сфери їх застосування, то з проблемами подальшої реалізації на програмному і/або апаратному рівнях.

Паралельні моделі – один із основних та перспективних напрямів в області розвитку програмування та апаратних засобів. Ідея паралелізму дуже приваблива. Але для її використання необхідно, по-перше, розв’язати проблему опису, тобто вибору формальної паралельної моделі, та, по-друге, проблему реалізації моделі. Скінченно-автоматна технологія використовує модель із засобами представлення та описування паралелізму, які за своїми можливостями не поступаються іншим паралельним моделям, а її реалізація багато в чому простіша. Крім того, застосовуючи стандартні прийоми, нескладно перейти від паралельного звичайного автоматного представлення до послідовного описування.

Приклади застосування скінченно-автоматної технології: бухгалтерські програми розрахунку заробітної плати або обліку квартирної плати, проект системи керування технологічним процесом вирощування кристалів з множиною динамічно породжуваних паралельно функціонуючих об’єктів, що реалізують процеси відбору даних з давачів, видачу керуючих дій на об’єкт, автоматні алгоритми роботи драйверів з різноманітною апаратурою, процеси відображення та розрахунку.

Отже, автоматне програмування використовується нині при проектуванні програмного забезпечення систем автоматизації відповідальних об’єктів управління (автомат управління кріогенно-вакуумною установкою, дизель-генератором). Скінченний автомат працює за принципом "крок убік – неприпустимо". Реалізувати непередбачені дії скінченний автомат не дасть ні користувачеві (початковий варіант коду), ні самій програмі (модифікований варіант коду).

**1.2.3. Побудова трансляторів**

Метою побудови трансляторів є практична побудова програми, яка перекладає вхідний текст (наприклад, програму, яка написана мовою програмування високого рівня) в машинну мову. При цьому досягнення цієї практичної мети ґрунтується на складному теоретичному фундаменті: теорії формальних мов та теорії автоматів, основними поняттями яких є алфавіт, ланцюжок, мова та граматика.

*Алфавітом* називають скінченну множину неподільних символів. Скінченні послідовності символів алфавіту називають *ланцюжками*, побудованими у цьому алфавіті. *Мова*  – це множина ланцюжків скінченної довжини у заданому скінченному алфавіті.

При описуванні мов виникає одне важливе питання: як описати мову ****** у тому випадку, коли вона нескінченна. Якщо довжини всіх ланцюжків обмежені, то ****** складається зі скінченного числа ланцюжків та можна скласти список усіх ланцюжків з ******.Однак для багатьох мов не можна або небажано встановити верхню границю припустимої дов­жини ланцюжка мови. Такі мови не можна визначити переліком усіх ланцюжків. Необхідний опис мови повинен бути скінченним, у той час, як мова може бути нескінченною. Відомо кілька таких способів описування мов. Один із них полягає у використанні системи, що називається *граматикою*.Залежно від способу визначення мови граматики поділяють на ті, що розпізнають, та на ті, що породжують.

*Граматика, що розпізнає,*– це граматика, яка для будь-якого рядка може визначити,є цей ланцюжок правиль­ним або ні. *Граматика, що породжує*– це граматика, яка може побудуватибудь-який правильний ланцюжок та не будує неправильних ланцюжків.

Серцевину граматики складає скінченна множина продукцій. *Продукція*– це впорядкована пара ланцюжків, першим еле­ментом якої є будь-який ланцюжок, що обов’язково містить хоча б один нетермінальний символ. Другим елементом пари є будь-який ланцюжок, що складений із символів ал­фавіту термальних та нетермальних символів.

Граматики можна класифікувати за виглядом їх продукцій. Така класифікація називається *ієрархією Хомського*. Граматики поділяють на:

### *Тип 0 – необмежені граматики* (*загального вигляду, з фразовою струк­турою*). Практичного застосування в міру своєї складності такі граматики не мають.

1. *Тип 1 – контекстно-залежні граматики.* До цього типу відносяться *контекстно-залежні* (*КЗ*) та *неукорочуючі* граматики. Ці класи граматик можуть використовуватися при аналізі текстів на природних мовах, але при побудові компіляторів практично не використовуються в міру своєї складності.

### *Тип 2 – контекстно-вільні (КВ) граматики.* КВ-граматики широко використовуються для описування синтаксису [комп’ютерних мов](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA).

1. *Тип 3 – регулярні граматики*(*автоматні*) *–* найпростіші з формальних граматик. Вони є контекстно-вільними, але з обмеженими можливостями. Регулярні граматики використовуються для описування найпростіших конструкцій: [ідентифікаторів](http://ru.wikipedia.org/wiki/%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80), [стрічок](http://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%BA%D0%B0), [констант](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D0%B0%D0%BD%D1%82%D0%B0), а також [мов асемблера](http://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0), [командних процесорів](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) тощо.

Кожному класу граматик відповідає клас мов. Мові приписується тип граматики, якою вона породжується. Для кожного класу граматик з ієрархії Хомського існує клас автоматів-розпізнавачів, що визначає той же клас мов. Цими розпізнавачами є скінченні автомати, автомати з магазинною пам’яттю, лінійно обмежені автомати та машини Тьюринга (див. рисунок 1.1). Точніше мови з ієрархії Хомського можна охарактеризувати так:

1. Мова регулярна (праволінійна або ліволінійна) тоді й тіль­ки тоді, коли вона визначається скінченним автоматом.

2. Мова контекстно-вільна тоді й тільки тоді, коли вона ви­значається автоматом з магазинною пам’яттю.

3. Мова контекстно-залежна тоді й тільки тоді, коли вона ви­значається лінійно обмеженим автоматом.

4. Мова визначається граматикою загального виду тоді й тіль­ки тоді, коли вона визначається машиною Тьюринга.

У традиційних областях застосування трансляторів прийнята своя термінологія. Основною такою областю є трансляція мов програмування високого рівня в еквівалентну програму іншою мовою. У цьому випадку початкова мова називається *вхідно*ю, а вхідна програма цією мовою називається *вхідним кодом*. Машинна мова, якою здійснюється трансляція, називається *об’єктною мовою*, результат трансляції називається *об’єктним кодом*. Комп’ютер, на якому відтрансльована програма буде запущена, називається *цільовою машиною*, він не обов’язково співпадає з типом машини, на якій працює транслятор.

При трансляції мов програмування від трансляторів може вимагатися виконання різних завдань. Якщо завданням є трансляція мовою низького рівня, наприклад, машинною мовою або мовою асемблера, то такий транслятор називається *компілятором*. Інше можливе завдання при трансляції програми алгоритмічною мовою – безпосереднє виконання транслятором вихідного коду. У цьому випадку транслятор називається *інтерпретатором*. У деяких випадках постає завдання трансляції, при якій виходом є програма мовою високого рівня, а входом – програма, що написана на деякому розширенні цієї мови. Такі транслятори називаються *препроцесорами*.

Рекурсивно-зліченні

мови

Машини Тьюринга

Контекстно-залежні мови

Лінійно-обмежені автомати

Контекстно-вільні мови

Автомати з магазинною пам’яттю

Регулярні мови

Скінченні автомати

Тип 0

Тип 1

Тип 2

Тип 3

**Граматики**

**Мови**

**Автомати**

Рисунок 1.1. Ієрархія граматик, мов та автоматів

У загальному випадку, завдання побудови трансляторів можуть бути дуже різноманітними. Вхідною мовою в трансляторі може бути не тільки мова програмування. Наприклад, транслятор може за словесним описом ситуації, що виражена на обмеженій підмножині природної мови, будувати анімаційну картинку цієї ситуації або формулювати обмірковану відповідь на вхідне питання з урахуванням стану діалогу. Такі програми-транслятори використовуються, наприклад, у штучному інтелекті.

**1.2.4. Реалізація візуалізації алгоритмів дискретної математики та програмування**

При вивченні алгоритмів опрацювання інформації, що подаються різними структурами даних, важливу роль відіграє візуалізація алгоритмів, що дає змогу в наочній формі динамічно відображати деталі їх роботи.

*Візуалізація* – це програма, в процесі роботи якої на екрані комп’ютера динамічно демонструється застосування алгоритму до вибраного набору даних. Візуалізація дає змогу вивчати роботу алгоритмів у покроковому режимі, аналогічному режиму трасування програм.

Для деяких алгоритмів динамічний варіант демонстрації його роботи є природнішим, ніж набір статичних ілюстрацій.

При вивченні більшості алгоритмів разом з режимом "крок вперед" дуже корисний також і режим "крок назад", що дає змогу швидше та повно зрозуміти алгоритм. Наприклад, в алгоритмах пошуку з поверненням буває необхідно зробити кілька кроків назад для того, щоб зрозуміти, чому та або інша гілка пошуку відкинута.

Приклади візуалізації: обхід двійкового дерева, алгоритми теорії розкладів, сортування тощо. Тобто, складні алгоритми з великою кількістю переходів, умов та галужень можна подати компактніше та зрозуміло у вигляді скінченного автомата з передбачуваною та наочною поведінкою.

**1.2.5. Штучний інтелект**

Штучні нейронні мережі (ШНМ) – математичні моделі, а також їх програмні або апаратні реалізації, побудовані за принципом організації та функціонування біологічних нейронних мереж – мереж нервових клітин живого організму. Це поняття виникло при вивченні процесів, що протікають у мозку, та при спробі змоделювати ці процеси.

Нейронні мережі – потужний метод моделювання, що дає змогу відтворювати надзвичайно складні залежності. ШНМ піддаються налаштуванню та навчанню. Використання автоматів при створенні штучних нейронних мереж дає змогу виключити появу непередбачених станів у їх роботі. Нейронні технології особливо інтенсивно застосовуються в експертних системах прогнозування родовищ та фінансовій справі при оцінюванні інвестицій.

**1.2.6. Створення прикладного програмного забезпечення для мобільних пристроїв та мікроконтролерів**

При побудові серверних програмних забезпечень, що відповідають на запити, велику роль відіграє "відсутність стану" – немає потреби зберігати стани між двома послідовними запитами.

При побудові вдалого інтерактивного програмного забезпечення, керованого подіями, багато що залежить від того, чи продумана модель управління станами. Скінченний автомат – дуже зручна концепція, яку доцільно використати для структуризації програмних продуктів.

Оскільки програмні забезпечення для мобільних систем повинні використати простір екрана та системні ресурси ефективно, скінченні автомати є особливо корисними при розробленні програмного забезпечення для таких застосувань.

Програма є сукупністю скінченних автоматів, що взаємодіють один з одним та із "зовнішнім світом". Діаграма переходів скінченного автомата описує переходи між екранними формами, дуги переходів зі стану в стан описують дії користувача. З кожною з конструйованих форм повинен зв’язуватися скінченний автомат, що керує візуальною поведінкою форми. Якщо в самій формі міститься кілька сторінок, наприклад, діалогові вікна з вкладками, то передбачається для кожної з підсторінок власний скінченний автомат.

Скінченні автомати значно розширюють можливості керування виконанням фонових завдань. Їх використання робить можливим надання фоновими потоками інформації про стан виконання, а також поводження інших потоків із запитами до фонового потоку на виконання певних дій, наприклад, із запитом на припинення виконання фонової роботи. При цьому в наочній графічній формі можуть бути виражені як зв’язки між автоматами, так і їхня внутрішня структура. Головна перевага – можливість повторного використання коду, швидка модифікація, наочність, що важливо у разі програмного забезпечення для мобільних пристроїв, які вимагають економного витрачання екранного простору, пам’яті, обчислювальної потужності та інших ресурсів.

**1.2.7. Побудова моделей документообігу на основі скінченно-автоматної моделі теорії автоматів**

У сучасному суспільстві йде процес інтенсифікації обчислювальних та інформаційних технологій в усіх галузях діяльності.

*Документообіг* – рух документів в організації з моменту їх створення або отримання до завершення використання. Впровадження електронного документообігу є актуальним завданням сучасного суспільства, оскільки він дає змогу зробити процес руху документів керованим та контрольованим, що забезпечує якісніші послуги управління. Підприємства та організації для вирішення цього завдання витрачають значні засоби та час. Водночас, кожна розробка системи документообігу є унікальною та можливість повторного використання отриманого досвіду в повному обсязі практично відсутня. Правильна організація цього процесу визначає якість та стабільність роботи будь-якого підприємства.

Використання автоматної моделі в розробці специфікацій документообігу та програмного продукту дає змогу створювати системи адекватніші вимогам користувачів та забезпечує можливість досягнення сумісності програмних продуктів.

Теорія автоматів дає змогу реалізувати логіку розгалуження руху документів між учасниками процесів документообігу. Автомат дає змогу встановити реакцію елементів системи документообігу на зміни в системі.

Як модельний об’єкт розглядається так званий композитний документообіг, тобто такий документообіг, в якому беруть участь як електронні, так і паперові форми подання документів. Композитний документообіг є трійкою: Дт ={У, Д, Ф}, де Дт – формальна модель документообігу; У – множина учасників; Д – множина дій; Ф – множина станів документів.

Множина станів документів S є результатом аналізу життєвого циклу документа. Ця множина станів, які можуть бути прийняті документом у межах модельованого документообігу, де кожне значення унікальне: {S}={Ф}.   
Під початковим станом мають на увазі первинний стан, в який потрапляє документ після ініціалізації процесу. При поданні документообігу у вигляді сукупності процесів, початковий стан є першим кроком, після якого можна казати про те, що документ існує та процес активізований. Таким чином, початкові стани – це об’єкти, елементи множини Ф, яка має один або кілька вихідних зв’язків та жодного вхідного.

Документообіг складається з сукупності процесів, кожен з яких обробляє один або кілька документів. Життєвий цикл процесу документообігу визначається рухом документів від початкових до завершальних станів. У даній моделі завершальні стани автомата визначаються як стани документа, після виникнення яких, робота автомата зупиняється, тобто процес документообігу перестає існувати. Таким чином, завершальні стани можна визначити як об’єкти множини Ф, яка має однин або кілька зв’язків, що входять, та жодного, що виходить.

У документообігу документ приймає наступний стан залежно від результату дії, яку над ним зробили. Функцію переходу автоматної моделі документообігу можна визначити як *i*-й елемент множини дій {Д} документообігу, після виконання якого відбувається зміна стану на стан {F}={Д} .

Алфавіт автомата – це множина символів, елементи якої надходять або можуть надійти до автомата. Як алфавіт автомата розглядають список учасників, між якими проходить документообіг.

Можна однозначно визначити автомат, який адекватно реалізовуватиме модель документообігу. Модель, побудована на детермінованих автоматах, дає змогу будувати моделі, які легше сприймаються візуально. Для них простіше побудувати програмну реалізацію. У той же час, при створенні моделей процесів, що мають складну розгалужену структуру, автоматна модель на детермінованих автоматах виходить великою та громіздкою.

Недетерміновані автомати дають змогу задавати складні процеси, використовуючи меншу кількість описового матеріалу. Проте для наочного сприйняття вони набагато складніші.

Отже, при невеликих слабо розгалужених процесах краще використовувати детерміновані автомати, в той час, як недетерміновані зручніші при зображенні процесів із великою кількістю кроків та розгалужень.

Після розроблення теоретичної бази реалізується програмне забезпечення, що втілює у практику автоматну та графову моделі документообігу. Кожен з учасників має можливість отримати доступ до конкретних видів документів та виконувати над ними певні дії.

Реалізація систем композитного документообігу дає змогу зробити діловодство прозорішим та прогнозованим, зменшує суб’єктивний вплив виконуючого персоналу на кінцевий результат.

Розглянута галузь нині швидко розвивається. Проводяться подальші дослідження в цьому напрямі, особливо це стосується застосування КС-граматик та створення програмного забезпечення, що реалізовує описану автоматну модель композитного документообігу в повному обсязі.

**1.2.8. Пошук ланцюжків у тексті**

Згідно з даними офіційної статистики близько 85% користувачів Інтернет постійно звертаються до пошукових систем (Google, Yandex, Rambler, Yahoo!, Апорт, [Пошук@Mail.ru](mailto:Пошук@Mail.ru), тощо) з метою пошуку необхідної їм інформації про товари або послуги.

Положення теорії автоматів підходять для описування таких реальних завдань, що виникають у програмних продуктах, як пошук у мережі Internet та видобування інформації з тексту. Багато сучасних пошукових систем Інтернету використовують спеціальну програму – пошуковий робот, який є автоматом.

У століття Інтернету та електронних бібліотек із безперервним доступом звичайною є проблема – задано деяку множини слів та необхідно знайти усі документи, в яких міститься одне чи усі з цих слів. Популярним прикладом такого процесу служить робота пошукової машини, яка використовує спеціальну технологію пошуку, що називається оберненими індексами (inverted indexes). Для кожного слова, що зустрічається в Internet (а їх близько 100000000), зберігається список адрес усіх місць, де воно зустрічається. Машини з дуже великим об’ємом оперативної пам’яті забезпечують постійний доступ до найбільш релевантних з цих списків, даючи змогу багатьом людям здійснювати пошук документів. У методі обернених індексів скінченні автомати не використовуються, але цей метод вимагає значних витрат часу для копіювання вмісту мережі та переписування індексів. Існує багато суміжних програмних продуктів, в яких застосовувати техніку обернених індексів не можна, однак можна з успіхом використовувати методи на основі автоматів. Ті застосування, для яких підходить технологія пошуку на основі автоматів, мають такі особливості:

1) вміст сховища тексту, в якому здійснюється пошук, швидко змінюється;

2) документи, пошук яких здійснюється, не можуть бути каталогізовані.

**Висновки**

1. Теорія цифрових автоматів – це розділ теорії систем керування, що вивчає математичні моделі перетворювачів дискретної інформації, які називаються автоматами. Такими перетворювачами є як реальні пристрої, так й абстрактні системи.
2. Теорія автоматів пов’язана з теорією інформації та теорією алгоритмів.
3. Теорія автоматів використовується у виробництві цифрових систем і програмуванні (побудові трансляторів, у штучному інтелекті тощо).

**Контрольні запитання**

1. Дайте означення автомату.
2. Наведіть задачі, які вивчає теорія цифрових автоматів. З яких основних частин вона складається?
3. Наведіть основні означення понять теорії інформації – «інформація», «повідомлення», «знак», «символ», «сигнал».
4. Яка операція опрацювання інформації називається кодуванням та декодуванням?
5. Наведіть класифікацію цифрових автоматів.
6. Охарактеризуйте застосування теорії автоматів для програмування та у виробництві цифрової техніки.

**2. ІНФОРМАЦІЙНІ ОСНОВИ КОМП’ЮТЕРІВ**

* *Комп’ютер як інформаційна система*
* *Поняття інформації, даних та сигналу*
* *Міри інформації*
* *Основи кодування інформації*
* *Дискретизація та квантування*
* *Кодування*

**2.1. Комп’ютер як інформаційна система (система відбору, опрацювання та збереження інформації)**

З практичної точки зору, *комп’ютер* – це деякий пристрій, призначений для відбору, перетворення (опрацювання) та збереження інформації. У цьому сенсі комп’ютер можна вважати інформаційною системою. Отже, поняття «комп’ютер» суттєво пов’язане з поняттям «інформація».

**2.2. Поняття інформації, даних та сигналу**

*Інформація* - надзвичайно ємнісне поняття, оскіліки містить, відображає значну множину можливих смислів. На побутовому рівні інформація - це різного роду відомості про щось, які можуть бути передані за допомогою писемної чи усної мови, радіо, телебачення, комп’ютерних систем. Інформація - це весь спектр відомостей, що отримує людина при взаємодії із усім світом та сама з собою за допомогою технічних засобів, п’яти органів відчуття, логічних розмірковувань, інтуіції.

Проте вище наведені трактування інформації носять чисто побутовий характер, а не наукові, оскільки є нечіткими, розмитими. Існує багато спроб в науці та філософії означити поняття інформації, наведемо для прикладу лише деякі.

1.Н.Вінер. Інформація- не енергія і не матерія.

.

.

.

N

На думку автора одним із кращих означень поняття інформації є таке, яке пов’язане з універсальною властивістю реальності (матерії) – відображенням. Так, під інформацією в загальному розуміють зміст відображення. В даному означенні необхідно розкрити смисл понять “відображення” та “зміст відображення”.

Під *відображенням* розуміють атрибутивну властивість матеріальних тіл, систем, об’єктів при їх взаємодії змінювати свої властивості, характеристики, структуру узгоджено, скорельовано одинм із одним. Тобто, структура однієї системи змінюється відповідно до, і в залежності від особливостей структури іншої матеріальної системи, що діє на неї. Тим самим виникає свого роду “відбиток”, “образ” структури другої системи в в структурі першої системи, і навпаки. Виникнення таких “відбитків”, “образів”, “копій” у структурі взаємодіючих матеріальних систем і називають відображенням.

Наведемо приклади відображень в реальних системах, розглядаючи їх в порядку зростання складності процесу відображення.

1. Відбиток печатки на папері.

2. Зміна опору терморезистивного давача в залежності від температури середовища, в яке він поміщений.

3. Відображення на моніторі ЕОМ, яке залежить від стану відеопам’яті.

4. Картина художника, що відображає природу.

5. Математична модель електричного коливного контура, механічного коливного маятника у вигляді диференціального рівняння другого порядку.

6. Внутрішній зміст свідомості людини, що має властивість інтенції – направленості на зовнішній світ з метою його діяльного відображення.

7. Рефлексія – відображення, усвідомлення свого власного внутрішнього світу, вмісту свідомості.

Поняття системи, фізичної величини, фізичного процесу, математичної моделі та їх взаємозв’яок.

Реальні системи завжди вивчаються через їх характеристики, властивості, які, власне, і визначають у своїй сукупності саму систему. Так, наприклад, досліджуючи електричний пристрій завжди говорять про його електричні властивоті, масогабаритні характеристики, надійність.

В науці та техніці цікавляться не будь-якими властивостями об’єктів та систем, а тими, які можна виразити числом, тобто кількісно охарактеризувати ступінь наявності тієї чи іншої властивості. Числові значення властивостей отримують шляхом вимірювання, тобто процедури порівняння, співставлення властивості досліджуваної системи із однорідною їй за природою властивістю іншої системи, що прийнята за еталон, міру і яка є носієм одиниці вимірювання даної властивості.

Тобто, результатом вимірювання є число та числові похибки вимірювання. Властивість, що може бути виражена в числовій формі шляхом проведення процедури вимірювання називають *фізичною величиною*.

Отже, розглядаючи системи різної фізичної природи, насправді вивчають сукупність фізичних величин, що характеризують систему та взаємозв’язок між ними. Ілюстрацією такого підходу є рисунок1.

Рис.1

Більшість фізичних величин змінюються в часі та в просторі, тому їх подають та вивчають як математичні функції часу та просторових координат. Зміну фізичної величини в просторі і/або часі називають *фізичним процесом*. Фізичний процес, що у своїй просторово-часовій структурі несе відомості, інформацію про щось називають *сигналом*.

Власне все, що ми знаємо про деяку систему отримане нами через сукупність сигналів, що випромінює, генерує система. Самі ж сигнали завжди досліджують в рамках їх математичних моделей, тобто таких математичних об’єктів (функцій, векторів, інтегралів, диференціальних рівнянь), що адекватно відображають просторово-часову структуру сигналів і по своїй суті є носіями інформації про досліджуваноу систему чи процес.

Носіями інформації є сигнали. Під сигналами будемо розуміти будь-який фізичний процес (зміна фізичної величини в просторі і/або часі), що несе відомості про щось. Відомості, інформація міститься в просторово-часовій структурі сигналу, яку в науці представляють у вигляді математичної моделі сигналу. Власне математична модель сигналу відображає характерні особливості просторово-часової структури сигналу, тому природно, що інформаційні властивості сигналу визначаються в теоретичних дослідженнях за його математичною моделлю, а в експериментальних дослідженнях – за зареєстрованою шляхом вимірювання реалізацією чи ансамлем реалізацій сигналу.

Отже, враховучи вище сказане, перед вивченням інформаційних властивостей сигналів неохідно ознайомитися із множиною їх можливих математичних моделей.

В теорії моделювання сигналів розрізняють:

1. Детерміновані та стохастичні
2. Дискретні та неперервні моделі сигналів.

Дуже часто слово “модель” упускають і відповідно говорять про детерміновані та стохастичні, дискретні та неперервні сигнали.

Детерміновані моделі сигналів не підходять для дослідження інформаційних властивостей сигналів, оскільки за означенням їх просторово-часова структура повністю відома, тому вони не можуть переносити інформацію, якій завжди притаманний елемент новизни.

Тому, природно, як моделі інформаційних сигналів використовувати стохастичні, імовірнісні процеси та поля, які можуть бути як дискретними, квантованими, цифровими, так і неперервними. Отже, вивчаючи курс теорії інформації будемо спочатку чітко вказувати на імовірнісну модель інформаційного сигналу, а вже потім вивчати його інформаційні характеристики, які визначаються моделлю.

При розгляді тем та питань курсу будемо дотримуватися схеми викладу матеріалу – від простішого до складнішого, що полягатиме в зростанні складності імовірнісних моделей сигналів.

Так будемо розглядати таку послідовність імовірнісних математичних моделей інформаційних сигналів:

1) Імовірнісний простір;

2) Випадкова величина;

3) Вектор випадкових величин;

4) Випадковий процес;

5) Вектор випадкових процесів;

6) Випадкове поле.

Кожен об’єкт будемо розглядати як для дискретного, так і для неперервного випадків.

Слід зауважити про зв’язок між поняттями система та сигнал, модель системи та модель сигналу.

ПОНЯТТЯ ЯКІСНОЇ ТА КІЛЬКІСНОЇ ІНФОРМАЦІЇ (вимірювальна інформація).

Основою будь-якої є різноманіття. В математиці аналогом, відповідником різноманіття є поняття множини. І поняття “різноманіття” і поняття “множина” є фундаментальними, основними, тому природно, що математичні структури, базуючись на понятті множина, можуть бути і носіями інформації, тобто моделями, які несуть інформацію, відомості.

Якщо множини числові, то і інформація, що в них закладена є числового стану. Якщо ж множина нечислова, то і інформація є якісного типу.

Частковим випадком інформації числового типу є вимірювання є вимірювальна інформація.

Перш ніж аналізувати таке поняття як інформація, необхідно проаналізувати поняття “різноманітність”, “структура”, “відображення” та відповідні їм математичні поняття: “множина”, “простір”, “функція”. Але перед тим слід поставити питання про еквівалентність між вище названими парами понять.

А це в свою чергу зачіпає проблему степені сепантичної еквівалентності (чи нееквівалентності) математичних понять та загально-людських філософських понять взагалі.

Проведемо компоративний аналіз понять “різноманітність” та “множина”. Під різноманітністю ми розуміємо наявність неоднорідності, відмінності, множинності, неподібності чогось.

Це досить ємнісне поняття, яке містить і поняття множинність. Тобто, поняття множинність завжди слідує разом із поняттям різноманітність, хоча це нееквівалентні поняття.

Без множинності не може бути різноманітності. Отже, “множинність”(диференційованість) є більш первинним поняттям по відношенню до поняття “різноманітність”, оскільки “множинність” відображає факт наявності декількох, а не одного (чогось), а “різноманітність” окрім факту множинності відображає, вказує на те, що ці декілька (чогось) є різними, а не однаковими. Хоча при більш глибокому аналізі це не зовсім так, оскільки “побачити” множинність можна лише тоді, коли виділяються, розрізняються між собою елементи деякої множини, а таке розрізнення вказує на неодинаковість, різність елементів будь-якої множини з потужністю більше ніж один елемент.

Термін «інформація» має багато означень. У широкому сенсі *інформація* пов’язана із такою – зміст відображення реального світу. Існує означення інформації й у вузькому сенсі: *інформація* – будь-які відомості, що є об’єктом відбору, передавання, перетворення та зберігання.

Інформація надходить в систему у вигляді повідомлення. Під *повідомленням* розуміють сукупність знаків або первинних сигналів, що містять інформацію.

*Джерело повідомлень* в загальному випадку утворює сукупність джерела інформації (ДІ) об’єкту дослідження або спостереження та первинного перетворювача (ПП) (датчика, людини-оператора, тощо), що відтворює інформацію про процес, який протікає в ньому.

Розрізняють дискретні та неперервні повідомлення.

*Дискретні повідомлення* формуються в результаті послідовного видавання джерелом повідомлень окремих елементів – *знаків, символів, букв*. Множину різних знаків називають *алфавітом* джерела повідомлення, а число знаків – *об’ємом алфавіту*. Якщо повідомлення є дискретними, то пристрій обробки такої інформації називають дискретним пристроєм.

*Неперервні повідомлення* не поділені на елементи. Вони описуються неперервними функціями часу, що приймають неперервну множину значень (мова, телевізійне зображення). Неперервні повідомлення можна передавати дискретними способами. В цьому разі неперервні сигнали, якими передаються ці повідомлення, перетворюються на дискретні за допомогою операцій квантування за рівнем та дискретизації в часі. На приймальному боці виконується обернене перетворення: за прийнятими дискретними сигналами відновлюються передані неперервні сигнали.

Дискретні сигнали як засіб передачі повідомлень більш поширені, ніж неперервні, завдяки тому що вони меншою мірою зазнають впливу завад та спотворень в каналах зв'язку, а в разі спотворення їх легше регенерувати (відновити) та, окрім того, вони досить легко обробляються в комп’ютері.

Для передавання повідомлення каналом зв’язку йому ставлять у відповідність певний сигнал. Під *сигналом* розуміють фізичний процес, що відображає, переносить повідомлення.

Будь-який сигнал характеризується такими основними параметрами:

* + тривалістю,
  + шириною частотного спектра
  + динамічним діапазоном.

Під *тривалістю*  сигналу розуміють час, протягом якого сигнал знаходиться в каналі зв'язку.

*Частотний* *спектр*  сигналу визначає смугу частот, яку сигнал охоплював під час передачі по каналу зв'язку.

*Середньою* *потужністю*  сигналу є потужність, яка забезпечується апаратурою під час його надходження до каналу зв'язку.

На практиці частіше замість  користуються поняттям *динамічного* *діапазону* , що визначається логарифмом відношення найбільшої (максимальної) миттєвої потужності сигналу () до найменшої (мінімальної) , дозволене значення якої дорівнює потужності завад ():

**.

Усі ці параметри сигналу є його *обсягом*:

.

Аналогічними параметрами характеризується також канал зв'язку. Ними є:

* + *тривалість* *використання*  каналу – час використання каналу для передачі сигналів,
  + *смуга* *частот*  – смуга частот, яка забезпечується каналом,
  + *динамічний* *діапазон*  – динамічний діапазон рівнів сигналів, які можуть бути передані каналом.

Добуток цих трьох параметрів визначає *ємність* каналу зв'язку:

 .

Для забезпечення передачі сигналів по каналу зв'язку необхідно, щоб , окрім того, мають виконуватися такі умови: , , .

Якщо деякі з цих умов не виконуються, треба досягти їх за рахунок інших. Так, якщо , , , але , то, збільшуючи , можна зменшити частотний спектр  сигналу та виконати умову .

Перетворення повідомлення в сигнал, зручний для передавання каналом зв’язку, називають *кодуванням* *у широкому сенсі*. Операцію відновлення повідомлення по прийнятому сигналу називають *декодуванням*.

Закодовані таким чином сигнали називають *цифровими сигналами*. Подання сигналу в цифровій формі практично завжди дає суттєву перевагу при передаванні, зберіганні та опрацюванні інформації.

Як правило, на практиці вдаються до операції представлення початкових знаків в іншому алфавіті з меншим числом знаків. Пристрій, що виконує таку операцію, називають таким, що кодує, або *кодером*. У такому разі кожному знаку відповідає деяка послідовність символів (*кодова комбінація*) з деякої множини, що називається *ансамблем повідомлень*. Множина повідомлень називається *алфавітом повідомлень*, або *первинним алфавітом*, а множина символів (елементів, знаків) називається *алфавітом джерела*,або*вторинним алфавітом*.

Для операції зіставлення символів зі знаками початкового алфавіту використовують термін "декодування". Технічна реалізація цієї операції здійснюється декодуючим пристроєм, або *декодером*.

*Пристрій передавання* здійснює перетворення неперервних повідомлень або знаків в сигнали, що є зручними для проходження по лінії зв’язку. При цьому один або декілька параметрів вибраного сигналу змінюють у відповідності до інформації передавання. Такий процес називається *модуляцією* та здійснюється *модулятором*. Зворотне перетворення сигналів у символи проводиться *демодулятором*.

Під *лінією зв’язку* розуміють середовище, яке забезпечує надходження сигналів від пристрою передавання до пристрою приймання. Сигнали на виході лінії зв’язку можуть відрізнятися від сигналів на її вході (переданих) внаслідок затухання, викривлення та дії завад.

*Завадами* називають будь-які заважаючі збурення, як зовнішні, так і внутрішні, що викликають відхилення прийнятих сигналів від переданих сигналів.

З суміші сигналу із завадами *пристрій приймання* виділяє сигнал та за допомогою декодера відновлює повідомлення, яке у загальному випадку може відрізнятися від надісланого. Міру відповідності прийнятого повідомлення надісланому повідомленню називають *правильністю передавання*.

Прийняте повідомлення з виходу системи зв’язку надходить до абонента-отримувача, якому була адресована початкова інформація.

Сукупність засобів, призначених для передавання інформації, називають *каналом зв’язку*.

Як приклад, на рисунку 2.1. зображено структурну схему одноканальної системи передачі інформації, що містить всі розглянуті поняття сигналу, повідомлення, джерела повідомлень.

ДІ

ПП

джерело

повідомлень

К

М

пристрій

передавання

повідомлення

Лінія зв’язку

Джерело завад

ДК

ДМ

пристрій

приймання

Одержувач

повідомлення

сигнал

сигнал+завади

Рисунок 2.1. Структурна схема одноканальної системи передачі інформації

**2.3. Міри інформації**

Важливе питання теорії інформації – встановлення міри, кількості та якості інформації. Інформаційні міри, переважно, розглядаються у трьох аспектах: структурному, статистичному та семантичному. Ці три напрямки мають свої визначені області застосування.

У *структурному аспекті* розглядається будова масивів інформації та їх зміна простим підрахунком інформаційних елементів або комбінаторним методом. Структурний підхід застосовується для оцінювання можливостей інформаційних систем незалежно від умов їх використання.

При *статистичному підході* використовується поняття ентропії як міри невизначеності, що враховує ймовірність появи того чи іншого повідомлення. Статистичні оцінки використовуються при розгляді питань передачі даних, визначенні пропускної здатності каналів зв’язку.

*Семантичний підхід* дає змогу виділити корисність або цінність інформаційного повідомлення. Семантичні оцінки використовуються при розв’язанні задач побудови систем передавання інформації, розробки кодуючи пристроїв та при оцінюванні ефективності різних пристроїв.

Коротко розглянемо ці міри інформації.

### **2.3.1. Структурна міра інформації**

Структурна міра інформації враховує тільки дискретну будову інформації. Елементами інформаційного комплексу є кванти – неподільні частини інформації. Розрізняють *геометричну*, *комбінаторну* та *адитивну* міри.

Визначення інформації *геометричним* методом являє собою вимірювання довжини лінії, площі або об’єму геометричної моделі інформаційного комплексу в кількості квантів. Максимально можлива кількість квантів в заданих структурних габаритах визначає *інформаційну ємкість системи*. Інформаційна ємкість є число, що вказує кількість квантів в повному масиві інформації.

В *комбінаторній* мірі кількість інформації обчислюється як кількість комбінацій елементів, при чому враховуються можливі або реалізовані комбінації.

В багатьох випадках дискретне повідомлення можна розглядати як слово, що складається з деякої кількості елементів , якізадані алфавітом, що складається з  елементів-букв. Визначимо кількість різних повідомлень, які можна утворити із даного алфавіту. Якщо повідомлення складається з двох елементів (), то всього може бути  різних повідомлень. Наприклад, з десяти цифр (0, 1, 2,..., 9) може бути утворено сто різних чисел від 0 до 99. Якщо кількість елементів дорівнює трьом, то кількість різних повідомлень дорівнює , тощо.

Таким чином, кількість можливих повідомлень визначається згідно формули:



де  – кількість повідомлень;  – кількість елементів в слові;  – алфавіт.

Чим більше , тим сильніше може відрізнятися кожне повідомлення від інших. Величина  може бути прийнята в якості міри кількості інформації. Але вибір в якості міри кількості інформації пов'язаний із незручностями: по-перше, при *L* =1 інформація дорівнює нулю, оскільки завчасно відомий характер повідомлення (тобто повідомлення є, а інформація дорівнює нулеві); по-друге, не виконується умова лінійного додавання кількості інформації, тобто умова адитивності. Якщо, наприклад, перше джерело характеризується  різними повідомленнями, а другий – , то загальна кількість різних повідомлень для двох джерел визначається добутком

.

Для  джерел загальна кількість можливих різних повідомлень дорівнює

.

Тому Хартлі ввів логарифмічну (адитивну) міру кількості інформації, яка дозволяю оцінити кількість інформації, що міститься в повідомленні, логарифмом кількості можливих повідомлень:

.

Тоді при *L=*1 *I=*0, тобто інформація відсутня.

Для джерел інформації

,

тобто

.

### **2.3.2. Статистична міра інформації**

При статичному ймовірнісному підході отримання конкретної кількості інформації розглядається як результат визначеного вибору серед можливих повідомлень. Отримувач інформації може завчасно знати або вгадати її частину. Коли надходить повідомлення про події, що часто відбуваються, ймовірність появи яких  прямує до одиниці, то таке повідомлення малоінформативне. Так саме малоінформативне в середньому повідомлення про події, ймовірності яких прямують до нуля, тобто про майже неможливі події, оскільки повідомлення про такі події надходять вкрай рідко.

Події можна розглядати як можливі результати деякого досвіду. Всі результати складають повну групу подій або ансамбль.

Ансамбль характеризується тим, що сума ймовірностей всіх повідомлень в ньому дорівнює одиниці, тобто:

.

Розглянемо складні повідомлення, що складаються з  елементів, кожен з яких є незалежним та вибирається з алфавіту, що містить  букв, з ймовірностями вибору елементів  відповідно. Припустимо, що в деяке повідомлення увійшло  елементів  алфавіту,  елементів , тощо. Таке повідомлення характеризується таблицею 2.1.

Таблица 2.1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Тип елементу |  |  |  | … |  | … |  |
| Кількість елементів |  |  |  | … |  | … |  |
| Ймовірності вибору елементів |  |  |  | … |  | … |  |

Ймовірність того, що в повідомлення увійдуть  елементів  дорівнює , а ймовірність утворення повідомлення з , , ,…, ,…,  елементів буде дорівнювати

. (2.1)

При великій довжині  джерелом будуть формуватися типові повідомлення, в яких відносна частота появи окремих елементів  прямує до ймовірності появи цих елементів, тобто

, (2.2)

а ймовірності появи типових повідомлень  будуть однакові та можуть бути знайдені з (2.1), (2.2):

. (2.3)

Визначимо кількість типових повідомлень:

, (2.4)

оскільки сумарна ймовірність всіх типових повідомлень прямує до одиниці при збільшенні довжини повідомлень.

Хоча кількість можливих повідомлень , джерело практично буде генерувати тільки  типових повідомлень, а ймовірність появи інших повідомлень прямує до нуля.

Знайдемо кількість інформації , що міститься в одному повідомленні:

. (2.5)

Даний вираз називається *формулою Шеннона* та дає більш повне уявлення про джерело інформації ніж адитивна міра (міра Хартлі). Пояснимо це на простому прикладі. Якщо спостерігач підкидає монету, то отримає повідомлення з двох можливих станів (орел або ріжка), тобто алфавіт повідомлень з двох букв. Якщо спостерігач підкидає кубик, одна грань якого біла, а інші грані чорного кольору, то тут також маємо алфавіт з двох букв (білий або чорний). Щоб записати отриманий текст (повідомлення), в обох випадках достатньо одної двійкової цифри на букву (*п=*1*, т=*2).

За формулою Хартлі обох випадках отримаємо:

.

Але в першому випадку ймовірність кожного результату досліду дорівнює 0,5 (), а в другому випадку  та  відповідно. Отже, міра Хартлі не враховує цього.

В частинному випадку, при рівноймовірності символів, формула Шеннона співпадає з формулою Хартлі:

.

Для випадку с монетою:

.

Для випадку з кубиком:

.

Кількість інформації, що припадає на один елемент повідомлення, називається *питомою інформативністю* або *ентропією*.

. (2.6)

Фізичний зміст ентропії – це середньостатистична міра невизначеності знань одержувача інформації щодо стану спостережуваного об'єкта.

Кількість інформації та ентропія є логарифмічними мірами та вимірюються в одних й тих самих одиницях. Основа логарифма визначає одиницю виміру кількості інформації та ентропії. Двійкова одиниця відповідає основі логарифма, що дорівнює двом, та називається *бітом*. Один біт – це кількість інформації у повідомленні в одному з двох рівноймовірнісних результатів деякого досліду.

Ентропія як кількісна міра інформаційності джерела має такі *властивості*:

1) ентропія дорівнює нулю, якщо хоча б одне з повідомлень достовірне;

2) ентропія завжди більша або дорівнює нулю, є величиною дійсною та обмеженою;

3) ентропія джерела з двома альтернативними подіями може змінюватися від 0 до 1;

4) ентропія – величина адитивна: ентропія джерела, повідомлення якого складаються з повідомлень декількох статистично незалежних джерел, дорівнює сумі ентропій цих джерел;

5) ентропія максимальна, якщо всі повідомлення мають однакову імовірність.

При нерівноймовірних елементарних повідомленнях *xi* ентропія зменшується. У зв'язку з цим вводиться така міра джерела, як *статистична надмірність*

, (2.7)

де *H*(*X*) – ентропія джерела повідомлень; *H*(*X*)*max*=*log*2*k* – максимально досяжна ентропія даного джерела.

Надмірність джерела (2.7) свідчить про інформаційний резерв повідомлень, елементи яких нерівноймовірні. Вона показує, яка частка максимально можливої при заданому об'ємі алфавіту невизначеності (ентропії) не використовується джерелом.

Синтаксичні міри інформації

* + Розглянемо базові поняття теорії інформації— кількість (міру) інформації та ентропію.

Будь–яке повідомлення описує стан певної системи (фізичної, біологічної, соціальної, ідеальної) за допомогою сигналу чи сукупності символів. Причому стан системи описується шляхом задання сукупності інформативних ознак (як правило, за допомогою фізичних величин), моделлю яких можуть бути тільки випадкові події, випадкові величини чи випадкові функції, оскільки детерміновані математичні моделі не можуть описувати невизначеність стану системи. А повідомлення є зміст передавати лише тоді, коли стан системи є невизначеним (невідомим), випадковим.

Тому в якості об’єкта, про який передається інформація, ми будемо розглядати деяку фізичну систему , яка випадковим чином може бути в тому чи іншому стані, тобто систему, якій властива певна степінь невизначеності.

Мірою невизначеності в теорії ніформації є ентропія .

Ентропія системи  визначається як:

, (1)

де — множина ймовірностей, з якими дискретна система  перебуває у відповідних -тих станах чи приймає відповідні -ті значення фізична величина, що характеризує систему.

У випадку, якщо система може перебувати в станах, що належать континуальній множині, її ентропія рівна:

, (2)

де – щільність розподілу неперервної випадкової величини, що характеризує стани системи .

Ентропія має такі властивості, що оправдовує її вибір в ролі характеристики ступеня невизначеності. По-перше, вона стає рівною нулю, коли один із станів системи є достовірним, а інші неможливі. По-друге, при заданому числі станів вона буде максимальною, коли ці стани є рівноймовірними, а при збільшенні числа станів – збільшуватися. Ентропія має властивість адитивності, тобто коли декілька незалежних систем об’єднуються в одну, їх ентропії додаються.

Відзначимо, що логарифм в формулах (1) та (2) може бути взятий при будь-якій основі . Зміна основи логарифму рівносильна простому множенню ентропії на постійне число, а вибір основи рівносильний вибору одиниці вимірювання ентропії. На практиці використовують логарифм при основі 2 і вимірюють ентропію в двійкових одиницях. Такий вибір зумовленний використанням двійкової системи числення в ЕОМ. У такому випадку ентропія системи (комірки памяті), що може мати лише два рівноімовірних стани рівна 1, оскільки:

 (3)

Означена таким чином одиниця ентропії називається “двійковою одиницею” або біт. Ентропія системи, що може перебувати в  станах з рівною імовірністю рівна

 (4)

Тобто ентропія системи з рівноможливими станами рівна логарифму числа станів. Цікавим є інтерпритація ентропії, як математичного сподівання випадкової величини :

 (5)

де  приймає значення , коли система  приймає значення .

### **2.3.3. Семантична міра інформації**

Семантичні міри інформації оцінюють змістовність, логічну кількість, доцільність та істотність інформації.

*Змістовність* події  виражається через функцію міри  – змістовності її заперечення. Оцінка змістовності заснована на математичній логіці, в якій логічні функції істинності  та хибності  мають формальну подібність з функціями ймовірності події  та антиподії  в теорії ймовірності. Подібно до ймовірності, змістовність події змінюється в межах .

*Логічна кількість* інформації , що подібна до статистичної кількості інформації, розраховується згідно формули:

.

Відмінність статистичної оцінки від логічної пролягає в тому, що в першому випадку враховуються ймовірності тих чи інших подій, що наближує до оцінки змісту інформації.

*Доцільність*, *корисність* інформації для розв’язання будь-якої задачі можна оцінити за ефектом, який виявляє отримана інформація на розв’язок задачі. Якщо ймовірність досягнення мети збільшується, то інформацію необхідно вважати корисною. Отримана інформація може бути порожньою, тобто не змінювати ймовірність досягнення мети, то в такому випадку її міра дорівнює нулеві. Якщо отримана інформація змінює положення діла в гіршу сторону, тобто зменшувати ймовірність досягнення мети, то тоді вона буде дезінформацією.

Міра доцільності у загальному вигляді може бути аналітично виражена у вигляді співвідношення:

,

де  та  – початкова (до отримання інформації) та кінцева (після отримання інформації) ймовірності досягнення мети.

# Семантичні міри інформації

Під семантикою розуміють зміст інформації.

Знаком називається умовне зображення елементів повідомлення, словом — сукупність знаків, які мають змістове(предметне) значення, мовою — словник і правила користування ним.

Відповідно наведеній вище структурі в семіотиці розрізніють синтаксичний, семантичний, сигмантичний і прагматичний аспекти теорії інформації.

Розглянуті в попередній параграфах структурна і статистична оцінки інформації відносяться до синтаксичного аспекту.

Сигмантичний аспект відображається теорією сигналів та кодів, яка розгядає умовні позначення елементів інформації. Сигнали являються фізичними носіями позначених елеметнів, а коди — позначеннями цих елементів. Сигматичні оцінки не мають прямого відношення до мір інформації. Тому залишається розлянути семантичні і прагматичні оцінки інформації.

В цьому параграфі розгядається оцінки, які відовідають як семантичному, так і прагматичному аспектам теорії інформації: в інженерній практиці прагматичні оцінки зливаються з семантичними, оскільки поняття які не мають змісту є безкорисні, а безкорисна інформація не має змісту.

Оцінка ефективності логічного виводу, степені наближення до істини потребує деякої формалізації, в даному випадку — формалізації змісту. Один з шляхів такої формалізації був запропонований семантичною теорією інформації.

Карнап і Бар-Хілел запропонували використовувати для вимірювння змісту функції істиності логічних висловлювань.

За основу дискретного опису обєкту береться атомарне представлення, схоже елементарній дії теорії відносності і яке відповідає неподільному кванту інформації.

Отримана таким чином оцінка одержала назву змістовності інформації.

## Змістовність інформації

Міра змістовності інформаціїь позначається cont (від англійського content – зміст).

Змістовніст події *і* виржається через функцію міри *M(i)* –змістовність його заперечення ­­— як

,

де *і -* подія яку розглядають; *m-*функція міри;  - знак заперечення.

Оцінка змістовності основана на математичній логіці, в якій логічні функції істиності *m(i)* інеправдивості *m(i )* мають формальну схожість з функціями ймовірносних подій *p(i)* і антиподій *q(i)* в теорії відносності.

В обох випадках мають місце схожі умови

;

як і відносність змістовність змінюється в межах

.

Відповідно схожі статистична та логічна кількість інформації.

Статистична оцінка кількості інформації (ентропія) згідно параграфу 1.2

.

Логічна оцінка кількості інформації, яка позначається inf , має схожий вираз

.

Відмінність статистичної оцінки від логічної полягає в тому. Що в першому випадку враховуються відносні реалізації тих чи інших подій, а в другому — міри ітиності чи неправдивості подій, що наближує до оцінки змісту інформації.

## Доцільніть інформації

Якщо інформація використовується в системах управління, то її користь розумно оцінювати по тому, як вона впливає на результат управління.

**2.4. Основи кодування інформації**

*Кодування* – це процес перетворення повідомлення на впорядкований набір символів, елементів, знаків. При кодуванні кожному повідомленню ставиться у відповідність зумовлена кодова комбінація – набір символів (елементів, знаків) з деякої скінченної кількості їх, яка називається *алфавітом*. Кількість символів у кодовій комбінації називають її *значущістю*, кількість ненульових символів – *вагою*.

Побудована відповідно до певної схеми кодування множина кодових комбінацій називається *кодом*. Залежно від алфавіту, що використовується для побудови кодових комбінацій, розрізняють *двійкові* (*бінарні*) *коди*, алфавіт яких складається з двох символів (0 та 1) та *недвійкові* (*багатопозиційні, q-коди*), алфавіт яких містить більшу кількість символів.

За функціональним призначенням коди поділяють на*безнадмірні (некоригувальні*, *первинні*, *прості)* та *надмірні (коригувальні, завадостійкі)*. Перша група кодів призначена для *економного кодування інформації – стиснення*. Друга використовується для виявлення та/або виправлення помилок, що виникають у процесі передачі даних *каналом зв'язку із завадами.*

Обидві групи кодів поділяються на *рівномірні* та *нерівномірні*, тобто коди зі сталою та змінною кількістю розрядів.

Надмірні коди також бувають:

* + *неперервними* (рекурентними) – процес кодування та декодування має неперервний характер;
  + *блоковими* –кожному повідомленню відповідає кодова комбінація (блок) зі скінченної кількості елементів. Блоки кодуються та декодуються окремо.

Блокові коди поділяються на:

* + *подільними* – коди, що будуються доповненням інформаційних елементів перевірними;
  + *неподільними* – коди, в яких немає чітко зумовлених інформаційних та перевірних елементів.

Подільні блокові коди бувають:

* + *систематичними* – код, у комбінаціях якого перші  позицій (розрядів) зайнято інформаційними елементами, а решту  позицій, де  – загальна кількість позицій в кодовій комбінації, – перевірними. Різновидом подільних систематичних блокових кодів є *циклічні* коди.
  + *несистематичними* –коди, в яких інформаційними елементами не зайнято всі  перших позицій.

Код кожного виду має свій найраціональніший спосіб подання, що випливає з його властивостей. Проте відомо також кілька загальних способів подання кодів, які є досить універсальними і можуть застосовуватися для опису широких класів кодів.

До цих способів належать подання кодів у вигляді:

* + таблиць кодових комбінацій;
  + кодового дерева;
  + геометричної моделі;
  + матриці.

Подання кодів у вигляді *таблиць кодових комбінацій* полягає в поданні коду у вигляді таблиці всіх його комбінацій та застосовується для подання будь-яких блокових кодів, але не може бути використаний для неперервних кодів.

*Приклад.* Триелементний двійковий блоковий код зі сталою вагою, в кожній комбінації якого містяться дві одиниці, задається так:

|  |  |  |  |
| --- | --- | --- | --- |
| Номер кодової комбінації | 1 | 2 | 3 |
| Комбінація двійкового блокового коду з вагою 2 | 011 | 101 | 110 |

Подання кодів у вигляді *кодового дерева* полягає в зображенні комбінацій коду у вигляді графа спеціального вигляду (дерева), в якому кодові комбінації розміщуються у вузлах. Застосовується для зображення як блокових, так й неперервних кодів.Перший вузол, від якого починається розходження ребер, називається *коренем* *дерева*, а кількість ребер, які треба пройти від кореня до будь-якого вузла – *рівнем*, або *порядком*, цього вузла.Максимальна кількість вузлів, які зустрічаються під час руху вздовж кодового дерева в напрямку від кореня до вершини, визначає *висоту*  кодового дерева. Вона дорівнює максимальній довжині комбінації коду, побудованому за допомогою цього дерева.

Вузли кодового дерева розташовуються на різних рівнях. Кожний рівень дерева рівномірного коду може мати  вузлів, де ** – основа коду,  – номер рівня (,  – довжина коду). Для рівномірного двійкового простого коду кількість вузлів на останньому рівні  дорівнює кількості  комбінацій коду, тобто . Вузли, що не з'єднуються з наступними рівнями, називаються *кінцевими* та відповідають комбінаціям коду.

Основою коду обмежується максимальна кількість ребер, яка може виходити з кожного вузла дерева, а максимальною довжиною кодової комбінації – максимальна кількість рівнів кодового дерева. Кожному вузлу приписується значення розрядів комбінації, що відповідає напрямкам руху вздовж ребер від кореня дерева до вузла. Ребра, що йдуть від кореня до вузлів першого рівня, визначають значення першого зліва розряду кодової комбінації, а ті, що з'єднують вузли першого та другого рівнів, – значення другого зліва розряду й т.д. На рисунку 2.2 подано приклади кодових дерев: рівномірного двоелементного тріскового (рисунок 2.2.а) та нерівромірного двійкового (рисунок 2.2.б).

0

1

2

00

01

02

10

11

12

20

21

22

0

1

00

01

010

011

Рисунок 2.2. Приклади кодових дерев

а)

б)

Подання кодів у *вигляді геометричної моделі* полягає в зображенні комбінацій коду точками дискретного -вимірного векторного простору.

Кожну комбінацію рівномірного блокового коду (з основою ** та довжиною ) *V*=(*Vn*, *Vn*-1,..., *V*2, *V*1) можна розглядати як вектор або точку деякого *n*-вимірного векторного простору з координатами *Vn*, *Vn*-1,..., *V*2, *V*1. Якщо значення ** скінченне, а будь-яка координата вектора є цілим додатним числом від 0 до *q*-1, то зазначений код можна розглядати як дискретний *n*-вимірний простір, що складається з  точок, які відповідають кінцям усіх можливих векторів. Цей простір називається *кодовим простором*.

Кількість просторових вимірювань кодового простору для коду з будь-якою основою дорівнює довжині *n* коду, а кількість градацій по кожній з осей (напрямків вимірювання) визначається основою коду і становить *q*-1.

Для дискретного *n*-вимірного простору вводиться поняття *кодової відстані*  між точками  та , яка дорівнює:

.

Одним з основних параметрів коду з довільною основою , що визначають його завадостійкість, є *мінімальна кодова відстань* , яка визначається мінімальною кількістю якісних ознак, за якими відрізняються одна від одної будь-яка пара комбінацій цього коду (на відміну від кодової відстані , що визначає кількість станів, які мають пройти якісні ознаки кодової комбінації, щоб опинитися в стані, який відповідає порівнюваній кодовій комбінації). Мінімальна кодова відстань характеризує не дві окремо взяті комбінації, а код у цілому.

Для визначення кодової відстані між комбінаціями коду з основою  треба виконати їх порозрядне віднімання за модулем . Кодова відстань дорівнює вазі комбінації, що складається з різниці значень комбінацій, між якими визначається ця відстань.

З'єднавши кожну точку простору, що розглядається, прямими лініями з усіма точками, віддаленими на відстань , отримують геометричну фігуру сіткової структури, що називається *геометричною моделлю* *n*-елементного -коду.

Точки дискретного простору, які містить ця геометрична фігура, називаються її *вершинами*, а лінії, що їх з'єднують, – *ребрами*.

*Приклад*. Моделлю будь-якого двозначного набору якісних ознак (двоелементного коду) є фігура двовимірного простору – квадрат або фігура, що складається з квадратів (рисунок 2). Моделлю будь-якого тризначного набору якісних ознак (триелементного коду) – фігура тривимірного простору – куб або фігура, що складається з кубів (рисунок 2.).

01

11

10

00





01

11

10

00





20

12

21

22

02

Рисунок 2. Геометричні моделі двоелементних кодів

001

101

100

000





001

212

121

000





200

102

201

202

002

Рисунок 2. Геометричні моделі триелементних кодів

110

111

010

011



011

111

110

010

012

021

120

020

022



220

210

112

211

122

221

222

100

101

Недоліком геометричної моделі є те, що при довжині коду  зобразити її у звичайному тривимірному просторі неможливо. Тому вона застосовується лише для рівномірних блокових кодів з метою наочного зображення та полегшення аналізу їхніх властивостей.

Спосіб подання кодів у *вигляді матриці* з  рядками та *n* стовпцями можливий тільки для рівномірних *n*-елементних двійкових блокових кодів. Якщо матрицею подається сукупність ненульових комбінацій коду, то кількість рядків дорівнюватиме .

З урахуванням того, що матриця *n*-елементного коду складається з  комбінацій, записаних у вигляді рядків, особливість такого запису полягає в тому, що додавання за модулем 2 будь-якої кількості рядків цієї матриці приводить до появи дозволеної комбінації коду, в тому числі й нульової. Якщо останню відкинути, то дістанемо нову матрицю коду, але вже з меншою кількістю рядків. Повторивши аналогічну операцію додавання рядків матриці за модулем 2, можна знову дістати нульову комбінацію коду. Ця операція повторюється доти, поки не буде отримана матриця з лінійно незалежними рядками, додавання яких за модулем 2 не приведе до утворення нульової комбінації коду.

*Приклад*. Побудувати матрицю триелементного двійкового простого коду.

Для цього треба, записавши у вигляді матриці всі  комбінацій простого коду, крім нульової, послідовно додати їх за модулем 2, виключаючи кожного разу ті комбінації, які в сумі з попередніми утворюють нульову комбінацію:

1) 2) 3) 4) 5) 6) 7)

001 001 001 001 001 001 100

010 010 010 010 010 010 010

011 000 100 100 100 100 001

100 100 000 000 000

101 101 110 111

110 110 111

111 111

Квадратна матриця, діагональ якої складається з одиниць, а решта її елементів – нулі, називається *одиничною*. Якщо рядки такої *n*-елементної матриці додавати за модулем 2, то підбором відповідної комбінації їх можна дістати всі комбінації *n*-елементного коду. Тому такі матриці ще називаються *визначальними*.

Якщо напрямок головної діагоналі матриці проходить з права на ліво, то така матриця називається *транспонованою*. Для розглянутого прикладу це буде матриця вигляду (шоста колонка):

.

Загалом визначальна матриця *n*-елементного коду має вигляд:



Розглянутий приклад утворення визначальної матриці й здобута одинична матриця можуть бути використані тільки для побудови всіх комбінацій двійкового простого коду з мінімальною кодовою відстанню .

**Вибір коду для передачі інформації**

При виборі кодів для передачі інформації керуються вимогами до вірогідності інформації, що передається, та швидкості передачі, які визначаються такими характеристиками кодів:

* кількістю  інформаційних елементів;
* кількістю  перевірних елементів (для коректувальних кодів);
* довжиною (розрядністю)  – кількістю елементів (символів), які входять до складу кодової комбінації ();
* основою (алфавітом) ;
* потужністю  – кількістю дозволених кодових комбінацій, що використовуються для передачі повідомлень;
* повною кількістю  кодових комбінацій, тобто кількістю всіх можливих комбінацій, яка дорівнює  (для двійкових кодів );
* надмірністю для неподільних кодів:

,

* або для подільних кодів при  та :

;

* відносною швидкістю , яка характеризує ступінь використання в надмірному коді інформаційних можливостей його потужності, причому

або;

* вагою кодової комбінації (для двійкового коду визначається кількістю одиниць у ній);
* мінімальною кодовою відстанню , тобто мінімальною відстанню між парами кодових комбінацій

,

* де ,  – елементи, що знаходяться в *l*-му місці в *і*- та *j*-й комбінаціях. Це значить, що  визначається кількістю однойменних розрядів з різними значеннями;
* імовірністю  невиявленої помилки, тобто ймовірністю такої події, за якої прийнята кодова комбінація відрізняється від переданої, а властивості коду не дають змоги визначити факт наявності помилки;
* імовірністю  виявленої помилки, тобто ймовірністю, за якої прийнята кодова комбінація відрізняється від переданої і завдяки властивостям коду встановлюється факт наявності помилки в кодовій комбінації;
* імовірністю виправленої помилки, тобто ймовірністю такої події, за якої прийнята кодова комбінація відрізняється від переданої і завдяки властивостям коду виправляється помилка в кодовій комбінації;
* імовірністю  виникнення помилки, тобто ймовірністю такої події, за якої прийнята кодова комбінація відрізняється від переданої ( – для кодів, які виявляють помилки, та  – для кодів, які виправляють помилки, де  – імовірність невиправленої виявленої помилки);
* кратністю *v* помилки, що визначається кратністю виявлених *v*в та *v*вп виправлених помилок;
* ефективністю:

,

де  – ймовірність виявленої або виправленої помилки залежно від властивостей коду.

Ступінь захисту інформації від помилок відповідним способом кодування залежить головним чином від мінімальної кодової відстані  даного коду.

Розрізняють три види кодової відстані: Хеммінга, Лі та матричну.

Кодова відстань Хеммінга *d* між двома комбінаціями однієї довжини *n* знайшла найбільше поширення в теорії кодування та визначається як кількість однойменних розрядів (позицій), які мають неоднакові елементи:

,

де  – кількість позицій в кодовій комбінації,



Згідно метрики Лі кодова відстань визначається згідно виразу:

,

де .

У модульній метриці кодова відстань дорівнює , тобто слід виконувати віднімання за модулем .

**Основні теореми кодування для каналів**

Теорія завадостійкого кодування базується на результатах досліджень, проведених Шенноном та сформульованих ним у вигляді *теорем*:

1. При будь-якій продуктивності джерела повідомлень, що є меншою за пропускну здатність канала, існує такий метод кодування, який дозволяє забезпечити передачу всієї інформації, створеної джерелом повідомлень, із як завгодно малою ймовірністю похибки.

2. Не існує методу, що дозволяє вести передавання інформації із як завгодно малою ймовірністю похибки, якщо продуктивність джерела повідомлень більша за пропускну здатність каналу.

Доведення цієї теореми можна знйти в [].

*Висновок з теореми Шеннона*: вірогідність передачі інформації по дискретному каналу із завадами тим вища, чим більша довжина кодової комбінації та менша продуктивність джерела відносно пропускної здатності каналу.

Для однозначного декодування прийнятих повідомлень, а також для передачі великих обсягів інформації з якомога мінімальнішими матеріальними та часовими витратами, коди мають задовольняти деяким вимогам, які сформулюємо у вигляді теорем. Доведення цих теорем можна знайти в [].

*Теорема 2.1* (*необхідна умова однозначного декодування коду*). Нехай код, що однозначно декодується, складається з  комбінацій завдовжки , a його алфавіт містить  символів. Тоді

.

Ця нерівність називається *нерівністю Крафта* для префіксних кодів, який довів, що для того щоб існував префіксний код в алфавіті обсягом  з комбінаціями завдовжки , необхідно й достатньо, щоб вона виконувалась.

Ця нерівність є необхідною та достатньою умовою існування кодового дерева, вершини якого мають порядки .

*Теорема 2.2.* Для будь-якого коду з властивістю однозначного декодування виконується умова

,

де  – середня довжина -коду при  кодових комбінаціях:

;

 – ентропія ансамблю повідомлень.

*Теорема 2.3.* Існує -код з властивістю однозначного декодування, для якого виконується нерівність:

.

**Оптимальне кодування**

Оптимальний з усіх точок зору код знайти практично неможливо. Тому кодування може бути оптимальним тільки за певних умов (з точок зору швидкості передачі інформації, здатності виправляти помилки тощо).

У теорії інформації існує кілька методик побудови оптимальних з точки зору швидкості передачі інформації безнадмірних кодів.

До *оптимальних безнадмірних кодів* (з точки зору довжини їх, тобто швидкості передачі інформації) належать нерівномірні коди, які передають повідомлення комбінаціями мінімальної середньої довжини.

Це зовсім не означає, що вони дійсно є абсолютно безнадмірними, оскільки такими вважаються коди, які задовольняють умову рівності обсягу та кількості інформації. Ці коди все ж таки мають потенціальну надмірність через заборонені кодові комбінації, до яких належать комбінації, що доповнюють вершини неповного кодового дерева, яке відповідає оптимальному нерівномірному коду (ОНК), до повного утворення рівномірного коду.

*Оптимальним кодуванням* називається процедура перетворення символів первинного алфавіту  на кодові комбінації вторинного алфавіту , при якій середня довжина повідомлення у вторинному алфавіті мінімальна.

Таким чином, основним завданням оптимального кодування є досягнення рівності між кількістю інформації , що виробляється джерелом повідомлень, та обсягом інформації  на вході приймача повідомлень. Якщо , то збільшення швидкості передачі інформації завдяки поліпшенню процедури кодування стає неможливим.

Існує дві методики побудови ОНК для дискретних ансамблів повідомлень  із середньою довжиною кодових комбінацій.

*Перша універсальна методика* побудови ОНК ґрунтується на методиці Шеннона-Фано та передбачає цю побудову в кодовому алфавіті з кількістю якісних значень . Згідно з цією методикою виконують такі процедури:

* 1. множину з  повідомлень, які кодуються, розташовують у порядку спадання ймовірностей;
  2. впорядковані за ймовірностями повідомлення розбивають, по можливості, на  рівноймовірних груп;
  3. кожній з груп завжди в одній й тій самій послідовності присвоюють символи алфавіту  (всім повідомленням першої групи – першу якісну ознаку цього алфавіту, всім повідомленням другої групи – другу якісну його ознаку тощо);
  4. створені групи розбивають, по можливості, на рівноймовірні підгрупи, кількість яких дорівнює або менша ніж  (якщо після розбивання в групі залишається одне повідомлення, то подальший поділ стає неможливим);
  5. кожній з утворених підгруп присвоюють якісні ознаки з алфавіту  за процедурою п.3;
  6. розбивання та присвоєння ознак алфавіту  повторюють доти, поки після чергового поділу в утворених підгрупах залишиться не більш як одне повідомлення.

Для побудови ОНК за викладеною методикою слід ураховувати також відхилення від рівноймовірних значень, що утворюються при поділі на підгрупи. Вони враховуються згідно з правилами заліку остач ділення та середнього відхилення:

* 1. для того щоб повідомлення первинного джерела можна було поділити по можливості на якомога рівноймовірні підгрупи при побудові ОНК з алфавітом , остача попереднього ділення додається за абсолютним значенням сумарної ймовірності чергового ділення (остачею ділення називається різниця між квантом ділення та реальним значенням сумарної ймовірності в групі (підгрупі), де квант ділення дорівнює );
  2. середнє відхилення має бути меншим або дорівнювати значенню ймовірності першого символу чергового ділення. Якщо середнє відхилення не дорівнює нулю, то середнє значення сумарної ймовірності в групі (підгрупі) при черговому діленні підраховується з додаванням значення середнього відхилення (середнім відхиленням називається абсолютне значення суми остач ділень на проміжних етапах побудови коду).

*Друга універсальна методика* побудови ОНК ґрунтується на відомій методиці Хаффмена. Вона, як і методика Шеннона-Фано, передбачає побудову ОНК у кодовому алфавіті з кількістю якісних значень . Згідно з цією методикою виконують такі процедури:

* 1. множину з  повідомлень, які кодуються, розташовують у порядку спадання ймовірностей;
  2. останні  повідомлень () об'єднують у нове повідомлення з імовірністю, що дорівнює сумі ймовірностей об'єднуваних повідомлень;
  3. утворену множину () повідомлень розташовують у порядку спадання ймовірностей;
  4. об'єднують останні  повідомлень та впорядковують множину повідомлень у порядку спадання ймовірностей. Так діють доти, доки ймовірність чергового об'єднаного повідомлення не дорівнюватиме одиниці;
  5. будують кодове дерево, починаючи з кореня, та гілкам цього дерева присвоюють якісні ознаки кодового алфавіту .

Обидві методики мають неоднозначність, але перша з них дає змогу точніше будувати ОНК. До недоліків другої універсальної методики побудови ОНК слід віднести громіздкість (особливо зі збільшенням кількості повідомлень  та алфавіту -коду), що пояснюється необхідністю побудови кодового дерева. Відзначимо, що переваги другої універсальної методики побудови ОНК з  при  будуть вагоміші при більш ретельному виборі кількості найменш імовірних повідомлень, що об'єднуються на першому етапі (). На всіх наступних етапах ця кількість має дорівнювати .

**Завадостійке кодування**

Кодування повинно здійснюватися так, щоб сигнал, який відповідає прийнятій послідовності символів, після впливу на нього наперед передбаченої в каналі завади лишався блище до сигналу, що відповідає конкретній переданій послідовності символів, ніж до сигналів, що відповідають іншим можливим послідовностям. (Степінь близькості зазвичай визначається за кількістю разрядів, в яких послідовності відрізняються одна від іншої.)

Це досягається шляхом введення при кодуванні надмірності, яка дозволяє так вибрати передані послідовності символів, щоб вони задовільняли додатковим умовам, перевірка яких на стороні приймання дає можливість виявити та виправити помилки.

Коди, що володіють такою властивітю, називають *завадостійкими*. Вони використовуються як для виправлення помилок (коректуючі коди), так й для їх знаходження.

Здатність коду виявляти та виправляти помилки обумовлена наявністю надлишкових символів. На вхід кодуючого пристрою надходить послідовність з  інформаційних двійкових символів. На виході їй відповідає послідовність з  двійкових символів, причому .

Всього може бути  різних вхідних та  різних вихідних послідовногстей. Із загальної кількості  вихідних послідовностей тільки  послідовностей відповідають вхідним. Вони називаються *дозволеними кодовими комбінаціями*. Інші  можливих вихідних послідовностей для передачі не використовуються. Вони називаються *забороненими комбінаціями*.

Викривлення інформації в процесі передачі зводиться до того, що деякі з переданих символів замінюються іншими – хибними. Так як кожна з  дозволених комбінацій в результаті дії завад може трансформуватися в будь-яку іншу, то всього існує  можливих випадків передачі. В цю кількість входять:  випадків безпомилкової передачі;  випадків переходу в інші дозволені комбінації, що відповідає помилкам, які не були виявлені;  випадків переходу в заборонені комбінації, які можуть бути виявлені.

Отже, частина помилкових кодових комбінацій, що можуть бути виявлені, від загальної кількості можливих випадків передачі дорівнює:

.

* + Розглянемо випадок виправлення помилок.

Будь-який метод декодування можна розглядати як правило разбиття всієї множини заборонених кодових комбінацій на  непересічних підмножини , кажна з яких ставиться у відповідність одній з дозволених комбінацій. При отриманні дозволеної комбинації, що належить підмножині , приймають рішення, що передавалась дозволена комбінація . Похибка буде виправлена у тих випадках, коли отримана комбінація дійсно утворилась з , тобто  випадках.

Всього випадків переходу в недозволені комбинації . Отже, за наявності надлишковості будь-який код може виправляти помилки. Відношення кількості виправлених кодом помилкових кодових комбінацій до кількості виявлених помилкових комбінацій дорівнює:

.

Метод розбиття на підмножини завлежить від того, які помилки повинні виправлятися даним конкретним кодом.

Завадостійкі коди поділяються на два великі класи: блокові та згорткові коди. Визначальна різниця між цими кодами – у відсутності або наявності пам'яті кодера.

Кодер для *блокових кодів* поділяє неперервну інформаційну послідовність на блоки завдовжки  символів. Кодер каналу перетворює блоки повідомлення  у більш довгі послідовності , що складаються з  символів й називаються *кодовими словами*.  символів, що додаються кодером до кожного блоку повідомлення, називають надлишковими, або перевірними, або контрольними. Вони не несуть додаткової інформації, їх функції полягають у забезпеченні можливостей виявлення й виправлення помилок, спричинених наявністю завад у каналі зв'язку.

-розрядним двійковим словам можна поставити у відповідність  різних значень з алфавіту джерела, яким відповідатиме  кодових слів на виході кодера. Така множина  кодових слів називається *блоковим кодом*. Термін «*без пам'яті»* означає, що кожний блок з  символів залежить лише від відповідного -символьного інформаційного блоку і не залежить від інших блоків повідомлення.

Кодердля *згорткових кодів* опрацьовує інформаційні послідовності без розбиття їх на незалежні блоки. У кожний момент часу кодер перетворює невелику послідовність з  інформаційних символів у блок з  кодових символів, де . При цьому кодовий -символьний блок залежить не тільки від -символьного інформаційного блоку, наявного на вході у поточний момент часу, але й від попередніх  блоків повідомлення.У цьому й полягає наявність пам'яті кодера.

Блокові коди доцільно використовувати у тих випадках, якщо первинні дані згруповані у блоки або масиви.

Згорткові коди найчастіше використовуються для передачі по радіоканалах та краще підходять для побітової передачі даних. Крім того, при однаковій надмірності згорткові коди, як правило, мають кращу виправну здатність.

Однією з умов практичної реалізації блокового завадостійкого коду є умова його *лінійності*.

Блоковий код завдовжки n символів, що складається з  кодових слів, називається *лінійним *-*кодом* за умови, що усі його  кодових слів утворюють -вимірний підпростір векторного простору -послідовностей двійкового поля GF(2). Іншими словами, двійковий код є лінійним, якщо порозрядна сума за модулем 2 (mod 2) двох кодових слів також є кодовим словом даного коду.

Бажаною якістю лінійних блокових кодів є систематичність. *Систематичний* код має інформаційну частину з  символів та надмірну (перевірну) частину з  символів постійної довжини.

Блочний код, що володіє властивістю лінійності та систематичності, називається *лінійним блочним систематичним* -кодом.

Способи задання лінійних кодів

Найпростіший спосіб задання лінійних кодів – *табличний*, при якому кожній інформаційній послідовності ставиться у відповідність кодове слово з таблиці кодів. Приклад такого задання коду для послідовностей довжиною  символи поданий у таблиці 2.2.

Таблиця 2.2.

|  |  |
| --- | --- |
|  |  |
| 000 | 0000 |
| 001 | 0011 |
| 010 | 0101 |
| 011 | 0110 |
| 100 | 1001 |
| 101 | 1010 |
| 110 | 1100 |
| 111 | 1111 |

Недоліком такого способу подання кодів є те, що при великих  розмір кодової таблиці виявляється дуже великим.

Іншим способом подання лінійних блокових кодів є *система перевірних рівнянь*, що визначає правило, за яким символи інформаційної послідовності перетворюються у кодові символи. Для прикладу з таблиці 2.2 система перевірних рівнянь буде мати вигляд:



Проте найзручнішим та наочним способом подання лінійного блокового коду є його запис за допомогою *породжуючої матриці*, яку записують у вигляді:

,

де – інформаційна підматриця;

 – перевірна підматриця.

*Лінійний блоковий систематичний* -код повністю визначається матрицею  розміром  з двійковими матричними елементами. При цьому кожне кодове слово є лінійною комбінацією рядків матриці , а кожна лінійна комбінація рядків  – кодовим словом.

Нехай  – блок-повідомлення, яке необхідно закодувати за допомогою даного коду. Тоді відповідним до нього кодовим словом  буде послідовність:

.

З врахуванням матриці  символи кодового слова  будуть такими:

1.  для ;

2.  для .

Іншими словами,  крайніх лівих символів кодового слова співпадаює з символами інформаційної послідовності, яку треба закодувати, а ніші  символів є лінійними комбінаціями символів інформаційної послідовності.

Отриманий таким чином код називається *лінійним блочним систематичним* *-кодом* з узагальненими перевірками на парність, а матриця , що його задає, *породжуючою матрицею коду*.

Сформулюємо правила побудови породжуючої матриці лінійного блокового - коду:

1) кількість початкових кодових комбінацій (число рядків) твірної матриці дорівнює , тобто кількості інформаційних елементів;

2) усі кодові комбінації твірної матриці повинні відрізнятися, причому нульова комбінація до їх складу не входить;

3) усі кодові комбінації твірної матриці лінійно незалежні;

4) кількість одиниць в кожній кодовій комбінації твірної матриці повинна бути ;

5) кодова відстань між будь-якою парою кодових слів повинна бути .

Лінійний блоковий код може бути заданий *перевірною матрицею* , що має таку властивість: якщо деяка послідовність  є кодовим, то

,

тобто перевірна матриця  ортогональна будь-якій кодовій послідовності даного коду.

Перевірна матриця має розмірність  та таку структуру:

,

де  – транспонована підматриця  з породжуючої матриці ;

 – одинична інформаціна матриця.

З властивості перевірної матриці лінійного блокового коду випливає, що за її допомогою можна визначити, чи є прийнята послідовність кодовим словом даного коду чи ні.

Виявлення та виправлення помилок лінійним блоковим кодом

Нехай  – кодове слово, передане по каналу із завадами, а  – прийнята послідовність, що через вплив завад може відрізнятися від переданого кодового слова .

Для опису виникаючих у каналі помилок використовується *вектор помилок*, що зазвичай позначають як  та являючий собою двійкову послідовність довжиною  з одиницями у тих позиціях, де виникли помилки.

Наприклад, вектор помилок означає однократну помилку у п’ятому біті,  – двократну помилку у першому та другому бітах.

Тоді, при передачі кодового слова  по каналу з помилками прийнята послідовність  матиме вигляд:

.

Наприклад: , , тоді .

Щоб перевірити наявність помилок у прийнятій послідовності , декодер обчислює -послідовність, що визначається наступним виразом:

,

де ** – транспонована перевірна матриця коду.

При цьомує кодовим словом тоді, та тільки тоді, коли , та не є кодовим словом даного коду, якщо ****.

Послідовність є ознакою наявності помилок у прийнятій послідовності та називається*кодовим синдромом****.***

Деякі комбінації помилок, використовуючи кодовий синдром, виявити неможливо. Наприклад, якщо передане кодове слово під впливом завад перетвориться на інше кодове слово цього самого коду, то синдром , та декодер не виявить помилок.

Покажемо, як можна використати синдром прийнятого вектора не тільки для виявлення, але й для виправлення помилок.

Нехай ,  та  є переданим кодовим словом, вектором-помилкою та прийнятою послідовністю відповідно. Тоді



та синдром

,

оскільки для будь-якого кодового слова .

Таким чином, кодовий синдром прийнятої послідовності  залежить лише від помилки, що має місце в цій послідовності, та зовсім не залежить від переданого кодового слова. Задача декодера, використовуючи цю залежність, полягає у визначенні елементів (координат) вектора помилок. Визначивши координати вектора помилок, можна відновити кодове слово як:

.

**Коди, що виявляють помилки**

Коди, що *виявляють помилки* поділяються на дві групи:

1. Двійкові коди, що виявляють помилки:
   * код із перевіркою на парність,
   * код із перевіркою на непарність,
   * код із простим повторенням,
   * інверсний код,
   * кореляційний код,
   * код зі сталою вагою,
   * код із кількістю одиниць у комбінації, кратною трьом.
2. Недвійкові коди, що виявляють помилки
   * код із перевіркою за модулем ,
   * код із повторенням,
   * незвідні змінно-позиційні коди.

Розглянемо лише двійкові коди. До найбільш поширених двійкових кодів з виявленням помилок відносять код з перевіркою на парність, коди з прямим та інверсним повторенням та кореляційний код. Розглянемо їх детальніше.

*Код із перевіркою на парність* містить лише один надлишковий символ. Вибирається надлишковий символ таким чином, щоб загальна кількість одиниць у кодовій комбінації була парною. Перевірка кодової комбінації робиться шляхом підсумовування за модулем два всіх його символів. Надмірність коду дорівнює:

,

де  – довжина кодової комбінації;

 – число перевірних символів.

Код дозволяє виявляти однократні помилки та всі помилки непарної кратності, тому що тільки в цих випадках кількість одиниць у комбінації стане непарною. Не виявляються помилки парної кратності.

Кодером коду з перевіркою на парність є електронний пристрій, який при надходженні на його входи  інформаційних символів двійкового коду автоматично добавляє до iнфopмaцiйниx символів один контрольний символ «0» чи «1», значення якого залежить від клькості одиниць в двійковій кодовій комбінації. Якщо кількість одиниць в двійковому коді парна, то контрольний символ дорівнює «0», якщо непарна – «1».

Таким чином у всіх кодових комбінаціях коду з перевіркою на парність кількість одиниць завжди повинна бути парною. Виявити спотворену внаслідок завад комбінацію можна на основі аналізу парності кількості одиниць в комбінації. Якщо парність порушена, кодова комбінація не приймається.

*Код із перевіркою на непарність* відрізняється від попереднього тим, що кожна його комбінація має непарну кількість одиниць, тобто додатковий перевірний елемент формують, виходячи з кількості одиниць у початковій кодовій комбінації: при парній кількості перевірний елемент дорівнює одиниці, а при непарній – нулеві.

Для виявлення помилки в кодовій комбінації на приймальному боці її перевіряють на непарність. Код є подільним завдовжки  інформаційних й один перевірний елементи, він може так само виявляти помилки та має надмірність, як й код із перевіркою на парність.

*Кодами з подвоєнням числа елементів* називаються коди, в яких для збільшення завадозахищенності до інформаційної кодової комбінації, яка складається з  інформаційних символів двійкового безнадлишкового коду, додається  контрольних символів, причому ; таким чином загальна кількість елементів в кодовій комбінації дорівнює .

В залежності від способу утворення контрольних символів такі коди класифікують на:

а) коди з прямим подвоєнням;

б) коди з інверсним подвоєнням;

в) кореляційні коди.

*Кодер коду з прямим повторенням* – це пристрій, який автоматично добавляє до інформаційної кодової комбінації двійкового коду таку ж саму кодову комбінацію (повторює її), збільшуючи загальну кількість двійкових символів вдвічі. *Декодер коду з прямим повторенням* порівнює інформаційні та контрольні символи комбінації, що надійшли на приймач (вони повинні повторювати одна одну) та дає команду на її подальше перетворення, якщо дія завади не виявлена.

*Коди з інверсним повторенням інформаційної двійкової кодової комбінації* утворюються за таким правилом: якщо кількість одиниць в інформаційній комбінації парна, то до неї автоматично додається така сама комбінація (пряме повторення) та якщо кількість одиниць в інформаційній комбінації непарна, до неї додається інвертована інформаційна комбінація. Наприклад, якщо інформаційна кодова комбінація має вигляд 1100, то до неї додається така сама комбінація 1100 та в цілому для подальшого перетворення надійде комбінація 11001100. Якщо інформаційна комбінація має вигляд 1110, то до неї буде додана комбінація 0001 та комбінація інверсного коду буде мати вигляд 11100001.

*Кодером коду з інверсним повторенням* називається електронний пристрій, який до інформаційної -розрядної комбінації двійкового коду автоматично додає контрольну комбінацію з  символів за таким алгоритмом: якщо кількість одиниць інформаційного коду парна, то до неї додається така ж сама комбінація, якщо непарна, до неї додається інвертована інформаційна комбінація.

*Декодер коду з інверсним повторенням* – пристрій, який за відомим алгоритмом утворення порівнює інформаційні та контрольні символи, що надійшли на приймальний пристрій, та дає команду на її подальше перетворення, якщо дія завади не виявлена.

У *кореляційному коді* кожний розряд двійкового початкового коду записується у вигляді двох елементів: 0 – як 01, а 1 – як 10. Наприклад, початковій кодовій комбінації 010011 відповідатиме комбінація 011001011010 кореляційного коду. В технічній літературі такий двійковий запис дуже часто називається *Манчестер*-*кодом*.

*Кодером кореляційного коду* є електронний пристрій, який при надходженні на його входи  інформаційних символів двійкового коду автоматично додає до кожного інформаційного символу по одному контрольному "0" чи "1" та значення якого утворюється за правилом: інформаційний нуль перетворюється в 01, інформаційна одиниця перетворюється в 10. Таким чином загальна кількість символів кодової комбінації кореляційного коду . Тому цей код відносять до коду з подвоєнням числа елементів.

*Декодером кореляційного коду* є пристрій, який аналізує прийняту кодову комбінацію. На прийомі помилка визначається в тому випадку, коли в парних елементах містяться однакові символи, тобто 11 чи 00 (заміcть 10 та 01). При правильному прийомі парні елементи відкидаються та залишається початкова комбінація.

**Коди, що виправляють помилки**

* Двійкові групові коди
  + Лінійний систематичний груповий (блоковий) код
  + Коди Хеммінга
  + Циклічні коди
  + Коди Боуза — Чоудхурі — Хоквінгема
  + Код Файра
  + Код із багатократним повторенням
  + Ітеративні коди
  + Каскадні коди
* Рекурентні коди
* Недвійкові коди
  + Код із багатократним повторенням
  + Узагальнений код Хеммінга
  + Коди Боуза — Чоудхурі — Хоквінгема
  + Коди Ріда — Соломона
  + Багатовимірні ітеративні коди
  + Недвійковий ланцюговий код

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

*Код Хемінга* відноситься до систематичних кодів, в яких з  символів, які утворюють комбінацію,  символів є інформаційними, а інші  є надлишковими (контрольними), призначеними для перевірки (контрольні символи у вcix комбінаціях займають однакові позиції). Коди Xeмінга дозволяють виправити вci одиничні помилки (при кодовій відстані ) та визначити вci подвійні помилки (при ), 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  символів коду, або комбінація може бути передана без спотворень. Таким чином може бути вapiaнтів спотворення (включаючи передачу без спотворення). Використовуючи контрольні символи, необхідно перевірити вci  варіантів. За допомогою контрольних символів  можна описати  подій. Для цього повинна бути використана умова:

.

В таблиці 2.3 подана залежність між  та , яка отримана з цієї умови, де  – число контрольних символів в коді Хемінга,  – інформаційних символів.

Таблиця 2.3. Розміщення контрольних символів в комбінаціях коду Хемінга

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|  | 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 i т.д. Інформаційні символи розташовують на місцях, що залишилися. Наприклад, для семиелементної закодованої комбінації можна записати

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| |  | | --- | |  | | |  | | --- | |  | |  | |  | | --- | |  | |  |  |  |

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

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |

1. Перше перевірне рівняння складають як суму за mod 2 вcix розрядів, в номерах яких в молодшому розряді () стоїть одиниця:

.

Друге перевірне рівняння складають як суму за mod 2 вcix розрядів, в номерах яких стоїть одиниця на другому місці відповідного двійкового еквівалента ():

.

Третє перевірне рівняння складають як суму за mod 2 всіх розрядів, в номерах яких стоїть одиниця на третьому місці ():

.

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

Як видно з наведених рівнянь, в кожну перевірну суму входить тільки один невизначений контрольний символ  (, , відповідно), а вci інші інформаційні символи відомі.

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

Наприклад, необхідно передати інформаційну кодову комбінацію: 1 1 0 0.

З формули  визначимо, що . Запишемо перевірні суми та прирівняємо їх до нуля:

,

,

.

Підставимо у рівняння значення відомих інформаційних символів та визначаємо відповідно контрольні символи. З першого рівняння , з другого , з третього .

Відповідно буде передана кодова комбінація (комбінація коду Хемінга):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | |  | | |  | | --- | |  | |  | |  | | --- | |  | |  |  |  |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 |

На приймальному пристрої комбінація коду Хемінга декодується –визначається наявність спотворення прийнятої кодової комбінації та, якщо один символ комбінації спотворений, він автоматично спочатку визначається, а потім виправляється. Визначення та виправлення спотвореного символу здійснюється на основі перевірних рівнянь. При правильному прийомі всі суми повинні дорівнювати нулю. В разі спотворення двійкове число, яке є результатом цих сум (синдром помилки), переводять в десяткове число, яке вказує номер спотвореного символу, який в подальшому виправляється шляхом інвертування.

Наприклад, при прийомі закодованої вище неспотвореної кодової комбінації Хемінга значення вcix символів комбінації підставляють на відповідні місця у перевірних сумах та виявляють ознаку правильно прийнятої комбінації – рівність нулю вcix сум.

Припустимо, що шостий символ кодової комбінації спотворений, тобто замість комбінації 0111100 буде прийнята кодова комбінація 0111110.

Підставимо значення прийнятих символів у перевірні рівняння та отримаємо:

,

,

.

Переведемо двійкове число (синдром) в десяткове. Одержимо десяткове число 6, яке й вказує на номер спотвореного символу.



*Циклічний код* – це лінійний -код, в якого частина або всі дозволені кодові комбінації утворюються шляхом циклічної перестановки розрядів (нульового розряду в останній) одної або кількох дозволених кодових комбінацій. Це значить, що якщо, наприклад, комбінація  є дозволеною комбінацією циклічного коду, то комбінація  також належить цьому кодові.

Основний пристрій, на базі якого здійснюється формування кодових комбінацій циклічного коду це регістр зсуву. Він являє собою послідовність тригерів (що буде розглянуто в 7 розділі даного посібника). Якщо зняти стан кожного тригера, то отримаємо деяку кодову комбінацію.

Основні властивості циклічних кодів:

1. В циклічному -коді кожен ненульовий поліном повинний мати степінь, в крайньому разі , але й не більше .
2. Існує тільки один кодовий поліном  степені  виду

,

що називається *породжуючим поліномом коду*.

1. Кожен кодовий поліном  є кратний , тобто

.

Кодування з використанням циклічних кодів полягає в наступному.

Нехай необхідно закодувати деяку інформаційну послідовність . Відповідний до неї поліном має такий вигляд:

.

Помноживши  на , отримаємо поліном степені  або меншої:

.

Скориставшись теоремою про ділення поліномів (ділення поліномів відбувається за правилами ділення степеневих функцій, при цьому операція віднімання замінюється додаванням за mod 2), можна записати:

,

де  та  – частка та залишок від ділення полінома  на породжуючий поліном .

Оскільки степень  дорівнює , то степень  має бути  або менше, а сам поліном  буде мати вигляд:

.

З врахуванням правил арифметики у  даний вираз можна переписати наступним чином:

,

звідки видно, що поліном  є кратний  та має степень  або меншу. Отже, поліном  – це кодовий поліном, що відповідає кодованій інформаційній послідовності .

Розкривши останній вираз, отримаємо:



що відповідає кодовому слову

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | перевірні символи | інформаційні символи |

Таким чином, кодове слово складається з незмінної інформаційної частини , перед якою розташовано  перевірних символів. Перевірні символи є коефіцієнтами поліному , тобто залишку від ділення  на породжуючий поліном .

*Приклад.* З використанням коду, що задається породжуючим поліномом , закодувати послідовність .

Послідовності  відповідає поліном . Помножимо  на  та отримаємо:

.

Поділимо  на породжуючий поліном :















Таким чином, кодовий поліном, що відповідає інформаційній послідовності , буде мати такий вигляд:

,

а відповідне кодове слово .

Отже, циклічний -код -розрядної інформаційної послідовності  отримується таким чином:

1. інформаційну послідовність  множать на , тобто зсувають вправо на  розрядів;
2. поліном отриманої послідовності ділять на породжуючий поліном коду ;
3. отриманий залишок від ділення  на  додають до , тобто записують в молодших  розрядах коду.

Обчислення синдрому для циклічних кодів є доволі простою процедурою.

Нехай  та  – поліноми, що відповідають переданому кодовому слову та прийнятій послідовності.

Поділивши  на , отримаємо:

,

де  – частка від ділення,

 – залишок від ділення.

Якщо  є кодовим поліномом, то він ділиться на  без залишку, тобто .

Отже,  є умовою наявності помилки в прийнятій послідовності, тобто є синдромом прийнятої послідовності.

Синдромний поліном  має в загальному випадку вигляд:

.

Можна довести, що синдромний поліном  є залишком від ділення поліному помилки  на породжуючий поліном , а отже, по синдрому  можна однозначно визначити вектор помилки  та виправити помилку.

Циклічні коди дозволяють виявляти та виправляти як одиночні та подвійні помилки, так й пачки помилок. Однак практичне застосування цих кодів для виправлення пачок помилок ускладнюється тим, що при не дуже великій надмірності довжина кодових комбінацій значно перевищує довжину пачок. У цьому відношенні більш зручними є рекурентні коди.

*Рекурентні* (*неперервні*) *коди* відрізняються тим, що в них операції кодування та декодування робляться неперервно над послідовністю символів без розподілу їх на блоки.

Ці коди призначені в основному для виправлення пачок помилок. Формування перевірних символів здійснюється шляхом додавання двох або більше інформаційних символів, зміщених один відносно одного на визначену відстань , що називається *кроком додавання*.

Довжина, що виправляється рекурентним кодом пачки помилок  залежить від кроку додавання  та визначається з умови:

.

Мінімальна необхідна відстань між пачками помилок, при якій забезпечується виправлення всіх помилок у пачці довжиною , дорівнює:

.

З усіх рекурентних кодів найбільше поширення одержав так званий *ланцюговий код*, що відрізняється гранично простими методами кодування та декодування. У цьому коді кожний перевірний символ формується шляхом додавання за mod 2 двох інформаційних символів, що віддалені один від одного на крок додавання .

Позначаючи послідовність інформаційних символів через , отримаємо таку послідовність перевірних символів для ланцюгового коду: ; ; …; ;  . У загальному потоці символів ланцюгового коду між кожними двома інформаційними символами міститься один перевірний:



Оскільки число перевірних символів, сформованих за деякий час, дорівнює числу інформаційних символів, що надійшли за той же час, то надмірність ланцюгового коду дорівнює 0,5.

На прийомі інформаційні перевірні символи розділяються та реєструються незалежно один від одного. З прийнятої послідовності інформаційних символів формуються контрольні символи так само, як формувалися перевірні символи при передачі. Після затримки на величину () кожний сформований контрольний символ звіряється з відповідним прийнятим перевірним символом. У випадку перекручування одного з перевірних символів буде розбіжність відповідних контрольних та перевірних символів. Якщо ж перекручений один з інформаційних символів, то буде розбіжність двох контрольних та відповідних перевірних символів, у формуванні яких бере участь даний інформаційний символ.

З розглянутого принципу виправлення помилок у коді випливає, що правильне виправлення помилок можливо тільки в тому випадку, якщо два з трьох символів, охоплених перевіркою, прийняті правильно. Це виконується за умови, що при кроці додавання , довжина пачки помилок не більше  та перевірні символи передаються в канал із затримкою на ().

Таким чином, код виявляє та виправляє пачки помилок порівняно просто, але ціною великої надмірності.

**2.5. Представлення інформації в комп’ютерах**

**2.5.1. Представлення символьної інформації в комп’ютерах**

Окрім чисел комп’ютери повинні представляти букви алфавіту, спеціальні знаки (пробіл, дужки, знаки пунктуації та інші символи) та знаки операцій, які можуть вводитися з клавіатури або виводитися на монітор чи принтер. Для опису таких символів часто використовується термін буквено-цифрові символи, оскільки вони включають в себе й десяткові цифри від 0 до 9. для їх кодування спеціалістами був розроблений стандартний код, що отримав назву ASCII-код (від англ. American Standard Code for Information Interchange – Американський стандартний код для обміну інформацією).

Стандартний набір символів ASCII використовує тільки 7 бітів для кожного символу. Додавання 8-го розряду дозволяє збільшити кількість кодів таблиці ASCII до 255.

Вісьмибітове кодування символів складається з двох таблиць кодування: базової та розширена. Базова таблиця побудована за стандартом ASCII та однакова для всіх комп’ютерів. Перші 32 коди (з 0 по 31) містять коди керування (таблиця 3.7), з коду 32 по код 127 розташовані коди символів англійського алфавіту, знаків пунктуації, цифр, арифметичних операцій та деяких допоміжних символів (таблиця 3.8).

Розширена таблиця відноситься до символів з номерами від 128 до 255 та може відрізнятися на комп’ютерах різного типу. В ній містяться символи псевдографіки, символи національного алфавіту, спеціальні знаки (таблиця 3.9).

*Юнікод* (Unicode – УНІфіковане КОДування) – це стандарт кодування символів, розроблений для забезпечення цифрового представлення символів усіх писемностей світу та спеціальних символів. Удосконалений сумісно з стандартом Універсальний Набір Символів (Universal Character Set – UCS) та опублікований у формі книги Стандарт Юнікод, Юнікод складається з асортименту символів, методології кодування та комплекту (набору) стандартів [кодування символів](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D1%96%D0%B2), комплекту кодових таблиць для посилань на зображення символів, списку властивостей символів таких, наприклад, як верхній та нижній [регістр](http://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%96%D1%81%D1%82%D1%80), комплект довідкових даних комп’ютерних файлів, правил нормалізації, декомпозиції, зіставлення т а зображення.

Стандарт запропонувала в [1991](http://uk.wikipedia.org/wiki/1991) році організація Консорціум Юнікоду з метою замінити в існуючі системи кодування символів Юнікодом та його системою стандартів Формат Перетворень Юнікоду (UTF, Unicode Transformation Format), тому що багато існуючих систем кодування є обмеженими в розмірі й можливостях та несумісними з багатомовними середовищами. Успіхи Юнікоду в уніфікації наборів символів призвели до його розповсюдження та домінуючого використання в [інтернаціоналізації](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F) та [локалізації](http://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F) [програмного забезпечення](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) комп’ютерів. Стандарт був використаний у багатьох новітніх технологіях, наприклад, у [XML](http://uk.wikipedia.org/wiki/XML), [мові програмування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [JavaScript](http://uk.wikipedia.org/wiki/JavaScript) та сучасних [операційних системах](http://uk.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0).

Юнікод знімає старе обмеження на кодування символів лише одним байтом. Натомість використовується 17 просторів, кожен з яких визначає 65,536 кодів та дає можливість описати максимум 1 114 112 (17\*216) різних символів.

Таблиця 3.7. Коди керуючих символів ASCII (0-31)

|  |  |  |  |
| --- | --- | --- | --- |
| Коди | Позначення | Клавіша | Значення |
| 0 | nul | <Ctrl>@ | Нуль |
| 1 | soh | <Ctrl>A | Початок заголовка |
| 2 | stx | <Ctrl>B | Початок тексту |
| 3 | etx | <Ctrl>C | Кінець тексту |
| 4 | eot | <Ctrl>D | Кінець передачі |
| 5 | enq | <Ctrl>E | Запит |
| 6 | ack | <Ctrl>F | Підтвердження |
| 7 | bel | <Ctrl>G | Сигнал (дзвінок) |
| 8 | bs | <Ctrl>H | Крок назад |
| 9 | ht | <Ctrl>I | Горизонтальна табуляція |
| 10 | lf | <Ctrl>J | Переведення рядка |
| 11 | vt | <Ctrl>K | Вертикальна табуляція |
| 12 | ff | <Ctrl>L | Нова сторінка |
| 13 | cr | <Ctrl>M | Повернення каретки |
| 14 | so | <Ctrl>N | Виключити зсув |
| 15 | si | <Ctrl>O | Включити зсув |
| 16 | dle | <Ctrl>P | Ключ зв’язку даних |
| 17 | dc1 | <Ctrl>Q | Керування пристроєм 1 |
| 18 | dc2 | <Ctrl>R | Керування пристроєм 2 |
| 19 | dc3 | <Ctrl>S | Керування пристроєм 3 |
| 20 | dc4 | <Ctrl>T | Керування пристроєм 4 |
| 21 | nak | <Ctrl>U | Заперечне підтвердження |
| 22 | syn | <Ctrl>V | Синхронізація |
| 23 | etb | <Ctrl>W | Кінець переданого блоку |
| 24 | can | <Ctrl>X | Відмова |
| 25 | em | <Ctrl>Y | Кінець середовища |
| 26 | sub | <Ctrl>Z | Заміна |
| 27 | esc | <Ctrl>[ | Ключ |
| 28 | fs | <Ctrl>\ | Подільник файлів |
| 29 | gs | <Ctrl>] | Подільник групи |
| 30 | rs | <Ctrl><Ctrl> | Подільник записів |
| 31 | us | <Ctrl>\_ | Подільник модулів |

Юнікод має декілька реалізацій, але найпоширенішими є дві: UTF (Unicode Transformation Format) – Формат Перетворення Юнікоду та UCS (Universal Character Set) – Універсальна Таблиця Символів. Число після UTF визначає кількість біт виділених під один юніт, а число після UCS визначає кількість байтів. Універсальний набір символів задає однозначну відповідність символів [кодам](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4) – елементам кодового простору, тобто невід’ємним цілим числам. [UTF-8](http://uk.wikipedia.org/wiki/UTF-8) став найпоширенішим для інтернаціональних кодувань.

Таблиця 3.8. Коди з символами ASCII (32-127)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Код | Символ | Код | Символ | Код | Символ | Код | Символ |
| 32 | пробіл | 56 | 8 | 80 | P | 104 | h |
| 33 | ! | 57 | 9 | 81 | Q | 105 | i |
| 34 | “ | 58 | : | 82 | R | 106 | j |
| 35 | # | 59 | ; | 83 | S | 107 | k |
| 36 | $ | 60 | < | 84 | T | 108 | l |
| 37 | % | 61 | = | 85 | U | 109 | m |
| 38 | & | 62 | > | 86 | V | 110 | n |
| 39 | ‘ | 63 | ? | 87 | W | 111 | o |
| 40 | ( | 64 | @ | 88 | X | 112 | p |
| 41 | ) | 65 | A | 89 | Y | 113 | q |
| 42 | \* | 66 | B | 90 | Z | 114 | r |
| 43 | + | 67 | C | 91 | [ | 115 | s |
| 44 | , | 68 | D | 92 | \ | 116 | t |
| 45 | - | 69 | E | 93 | ] | 117 | u |
| 46 | . | 70 | F | 94 | ^ | 118 | v |
| 47 | / | 71 | G | 95 | \_ | 119 | w |
| 48 | 0 | 72 | H | 96 | ` | 120 | x |
| 49 | 1 | 73 | I | 97 | a | 121 | y |
| 50 | 2 | 74 | J | 98 | b | 122 | z |
| 51 | 3 | 75 | K | 99 | c | 123 | { |
| 52 | 4 | 76 | L | 100 | d | 124 | | |
| 53 | 5 | 77 | M | 101 | e | 125 | } |
| 54 | 6 | 78 | N | 102 | f | 126 | ~ |
| 55 | 7 | 79 | O | 103 | g | 127 | del |

[UTF-8](http://uk.wikipedia.org/wiki/UTF-8) є системою кодування зі змінною довжиною кодування символів. Це означає, що для кодування символів він використовує від 1 до 4 байт на символ. Так, перший байт UTF-8 можна використовувати для кодування [ASCII](http://uk.wikipedia.org/wiki/ASCII), що дає повну сумісність з кодами ASCII. Перекодування кодів ASCII у кодах UTF-8 для латинських символів не збільшить розмір даних, бо для цього використовується тільки один байт на символ. Для символів інших мов, де, наприклад, для кодування треба використовувати два байти на символ, це кодування збільшує розмір даних на, приблизно, 50% або більше.

[UTF-8](http://uk.wikipedia.org/wiki/UTF-8) дозволяє працювати в стандартизованому міжнародно прийнятому багатомовному середовищі, з порівняно незначним збільшенням обсягу даних. UTF-8 являє собою ідеальний спосіб передачі символів через Інтернет, електронну пошту, чат тощо.

Коди в стандарті Юнікод поділені на декілька областей. Область з кодами від U+0000 до U+007F містить символи набору [ASCII](http://uk.wikipedia.org/wiki/ASCII). Далі розміщені області знаків різних писемностей, знаки пунктуації та технічні символи. Частина кодів зарезервована для використання в майбутньому. Для символів кирилиці виділені коди від U+0400 до U+052F.

Таблиця 3.9. Символи з кодами 128-255 (Кодова таблиця 1251 – MS Windows)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Код | Символ | Код | Символ | Код | Символ | Код | Символ |
| 128 | Ђ | 160 |  | 192 | A | 224 | а |
| 129 | Ѓ | 161 | Ў | 193 | Б | 225 | б |
| 130 | ‚ | 162 | ў | 194 | В | 226 | в |
| 131 | ѓ | 163 | Ј | 195 | Г | 227 | г |
| 132 | „ | 164 | ¤ | 196 | Д | 228 | д |
| 133 | … | 165 | Ґ | 197 | Е | 229 | е |
| 134 | † | 166 | ¦ | 198 | Ж | 230 | ж |
| 135 | ‡ | 167 | § | 199 | З | 231 | з |
| 136 | € | 168 | Ё | 200 | И | 232 | и |
| 137 | ‰ | 169 | © | 201 | Й | 233 | й |
| 138 | Љ | 170 | Є | 202 | К | 234 | к |
| 139 | ‹ | 171 | « | 203 | Л | 235 | л |
| 140 | Њ | 172 | ¬ | 204 | М | 236 | м |
| 141 | Ќ | 173 | ­ | 205 | Н | 237 | н |
| 142 | Ћ | 174 | ® | 206 | О | 238 | о |
| 143 | Џ | 175 | Ї | 207 | П | 239 | п |
| 144 | ђ | 176 | ° | 208 | Р | 240 | р |
| 145 | ‘ | 177 | ± | 209 | С | 241 | с |
| 146 | ’ | 178 | І | 210 | Т | 242 | т— |
| 147 | “ | 179 | і | 211 | У | 243 | у |
| 148 | ” | 180 | ґ | 212 | Ф | 244 | ф |
| 149 | • | 181 | µ | 213 | Х | 245 | х |
| 150 | – | 182 | ¶ | 214 | Ц | 246 | ц |
| 151 | — | 183 | . | 215 | Ч | 247 | ч |
| 152 |  | 184 | ё– | 216 | Ш | 248 | ш |
| 153 | ™ | 185 | № | 217 | Щ | 249 | щ |
| 154 | љ | 186 | є | 218 | Ъ | 250 | ъ |
| 155 | › | 187 | » | 219 | Ы | 251 | ы |
| 156 | њ | 188 | j | 220 | Ь | 252 | ь |
| 157 | ќ | 189 | S | 221 | Э | 253 | э |
| 158 | ћ | 190 | s | 222 | Ю | 254 | ю |
| 159 | џ | 191 | Ї | 223 | Я | 255 | я |

Стандарт Юнікод складається з двох основних розділів: універсальний набір символів та сімейство кодувань. Універсальний набір символів задає однозначну відповідність символів кодам – елементам кодового простору, що представляють ненегативні цілі числа. Сімейство кодувань визначає машинне представлення послідовності кодів універсального набору символів.

Стандарти наборів символів:

UCS-4 (англ. Universal Character Set) – 1 символ = 4 байти, всього можна закодувати 232 = 4 294 967 296 символів. Проте максимальна кількість Юнікод-символів на сьогодні – 220 + 216 = 1 114 112.

UCS-2 (англ. Universal Character Set) – 1 символ = 2 байти, всього можна закодувати 216 = 65 536 символів.

Стандарти кодувань:

UTF-32 (англ. Unicode Transformation Format – формат перетворення Юнікода) – один із способів кодування символів із Unicode у вигляді 32-бітових послідовностей. 1 символ = 32-біти.

UTF-16 – один із способів кодування символів із Unicode у вигляді 16-бітних послідовностей. Символи з кодами менше 0x10000 (216) представляються як є (одна 16-бітова послідовність), а символи з кодами 0x10000-0x10FFFE – у вигляді двох 16-бітових послідовностей (так звана «сурогатна» пара), перша з яких лежить в діапазоні 0xD800–0xDBFF, а друга – 0xDC00-0xDFFF. Легко побачити, що існує 210 \* 210 = 220 таких комбінацій. А загальна кількість можливих символів 220 + 216 = 1 114 112. Слід зазначити, що за стандартом ніякі символи не можуть мати кодів власне з діапазону 0xD800-0xDFFF, так що розшифровка кодування завжди однозначна. Втім, в переважній більшості випадків текст в UTF-16 є просто послідовністю символів з UCS-2, оскільки символи Юнікод після коду 0x10000 використовуються вкрай рідко. У потоці даних UTF-16 старший байт може записуватися або перед молодшим (UTF-16 Big Endian або UTF-16BE), або після молодшого (UTF-16 Little Endian або UTF-16LE). Іноді кодування Юнікода Big Endian (UTF-16BE) називають Юнікодом із зворотним порядком байтів. Аналогічно існує два варіанти 32-бітового кодування: UTF-32LE та UTF-32BE.

UTF-8 – поширене сьогодні кодування, що реалізовує представлення Юнікоду, сумісне з 8-бітовим кодуванням тексту. Текст, що складається тільки з символів з номером менше 128, при записі в UTF-8 перетворюється на звичайний текст ASCII. І навпаки, в тексті UTF-8 будь-який байт із значенням менше 128 зображає символ ASCII з тим же кодом. Решта символів Юнікоду зображається послідовностями завдовжки від 2 до 6 байтів (реально тільки до 4 байт, оскільки використання кодів більше 221 не планується), в яких перший байт завжди має вид 11xxxxxx, а інші – 10xxxxxx.

Простіше кажучи, у форматі UTF-8 символи латинського алфавіту, розділові знаки і керуючі символи ASCII, записуються кодами ASCII-кодами, а решта всіх символів кодується за допомогою октетів (послідовності довжиною 8 біт) із старшим бітом 1. В результаті, навіть якщо програма не розпізнає Юнікод, то латинські букви, арабські цифри та розділові знаки зображатимуться правильно.

Хоча форми запису UTF-8 та UTF-32 (8 та 32 десяткові числа, які вказують кількість двійкових розрядів) дозволяють кодувати до 231 (2 147 483 648) кодових позицій, було прийнято рішення використовувати лише 220+216 (1 114 112) для сумісності з UTF-16. Втім, навіть й цього більш ніж достатньо – сьогодні (в версії Unicode 5.0) використовується трохи більше 99 000 кодових позицій.

**2.5.2. Представлення сигналів в комп’ютері. Цифровий сигнал.**

Інформацію, що надходить від джерела інформації, можна зберігати, обробляти та передавати як у вигляді неперервних сигналів, так й у вигляді дискретних. Насьогодні в інформаційній техніці перевага віддається дискретним сигналам, а тому неперервні сигнали перетворюються в дискретні.

Сприйняття інформації про об’єкти чи процеси здійснюється за допомогою пристроїв, які називаються первинними перетворювачами чи сенсорами. В більшості випадків сенсори відображають вхідну інформацію у вигляді еквівалентного електричного параметра. Тобто, сенсором називається елемент, який приймає контрольований параметр та перетворює його до вигляду, зручного для подальшої обробки (вимірювання, передачі, контролю).

Відповідно до схеми вмикання сенсорів можна визначити дві групи узгоджувально-нормувальних пристроїв. До першої групи належать пристрої, в яких сенсори є елементами подільників напруги, до другої – пристрої, в яких сенсори є елементами коливальних систем ВЧ генераторів.

В пристроях першої групи сенсори найчастіше вмикаються за диференціальною чи мостовою схемою.

Диференціальні схеми відрізняються високою стабільністю, оскільки дестабілізуючі фактори одночасно діють на обидва елементи диференціального сенсора, що компенсує цей вплив.

Сенсори в мостовій схемі входять до складу моста, який врівноважений при деякому (звичайно нульовому чи початковому) значенні контрольованого параметра.

При вимірюваннях деяких неелектричних величин не завжди вдається перетворити їх безпосередньо в електричну величину. В цих випадках здійснюють подвійне перетворення первинної вимірюваної величини в проміжну неелектричну величину, яку перетворюють потім у вихідну електричну величину.

Сукупність двох відповідних вимірювальних перетворювачів утворюють комбінований сенсор (рисунок 2.18).

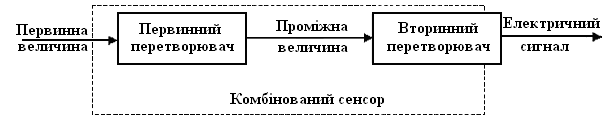


Рисунок 2.18. Блок-схема комбінованого сенсора

Подібні перетворювачі зручні для вимірювання неелектричних (механічних) величин, які викликають в первинному перетворювачі деформацію або переміщення вихідного елемента, до яких чутливий вторинний перетворювач.

Процес перетворення неперервних сигналів на дискретні має кілька різновидів. При цьому розрізняють:

* + дискретизацію (квантування за часом),
  + квантування (квантування за рівнем),
  + квантизацію (квантування за часом та рівнем, або комбіноване квантування).

**2.3.1. Дискретизація**

*Дискретизацією* називається перетворення функції неперервного часу в функцію дискретного часу, що представлена сукупністю величин, що називається координатами, по значенням яких початкова неперервна функція може бути відновлена із заданою точністю. Роль координат часто виконують миттєві значення функції, зафіксовані у визначені моменти часу.

*Постановка задачі дискретизації* звучить наступним чином.

В самому загальному випадку представлення неперервного сигналу  на інтервалі  сукупністю координат  може бути записано в вигляді

,

де  – оператор дискретного представлення сигналу, що реалізується пристроєм, який називається дискретизатором.

Аналогічно можна записати й операцію відновлення по сукупності координат  неперервної функції  (функції відновлення), що відображає початковий сигнал з деякою похибкою наближення :

,

де  – оператор відновлення, що реалізується пристроєм відновлення сигналу.

Задача дискретизації в математичному сенсі зводиться до сумісного вибору пари операторів  та , які б забезпечували задану точність відновлення сигналу.

В основі процесу дискретизації неперервних сигналів лежить теорема відліків, яку сформулював В.О.Котельников. Теорема встановлює принципову можливість повного відновлення детермінованої функції з обмеженим спектром по її відлікам та вказує граничне значення інтервалу часу між відліками, при яких таке відновлення ще можливе.

*Теорема В.О. Котельникова* формулюється наступним чином: якщо неперервна в часі функція має обмежений частотний спектр, який не містить складових з частотами, що перевищують ,то вона повністю визначається сукупністю своїх миттєвих значень (дискрет), які відлічуються через інтервали часу , де – максимальна частота спектра неперервного сигналу.

Дискрети ще визначаються *відліками* (визначальними ординатами), а моменти відліку – *тактовими точками*. Інтервал між відліками (дискретами)  називається *інтервалом* (або *кроком*) *дискретизації* (кроком квантування за часом).

Таким чином, якщо є потреба передачі неперервного сигналу з обмеженим спектром, то досить передати його окремі дискретні значення, відлічені через проміжки часу  (рисунок 2.2). Чим менші інтервали , тим точніше буде передана функція . Тривалість  дискрет теоретично може бути дуже малою, але практично вона вибирається з урахуванням ширини смуги пропускання  каналу зв'язку, оскільки , де  – стала величина, що близька до 1.



0







…

Рисунок 2.2.

Методи дискретизації та відновлення інформації класифікуються залежно від регулярності інтервалів дискретизації, критерію оцінки точності дискретизації та відновлення, виду базисної функції, принципу наближення.

Якщо крок дискретизації є сталим на всьому інтервалі перетворення, то дискретизація вважається *рівномірною*. Методи рівномірної дискретизації отримали найбільш широке застосування. Вони характеризуються простим алгоритмом, що виключає необхідність фіксувати час відліків, що суттєво полегшує технічну реалізацію. Але, в цьому випадку невідповідність кроку дискретизації характеру поведінки конкретної реалізації сигналу на окремих ділянках часто призводить до значної надмірності відліків.

Якщо відрізки часу між вибірками змінюються, наприклад, в залежності від швидкості зміни сигналу або по заданій програмі, дискретизацію називають *нерівномірною*.

Тип базисних (наближуючих, відновлюючих) функцій визначається вимогами обмеження складності пристроїв (програм) дискретизації та відновлення сигналів. Найчастіше для дискретизації та відновлення використовуються ряди Фур’є та Котельникова, степеневі поліноми, поліноми Чебишева, Лежандра, Уолша та Хаара, а також гіпергеометричні функції.

Вимоги до точності відновлення диктуються отримувачем інформації. В залежності від цільового призначення отриманої інформації використовуються різні критерії точності наближення  до , а саме:

* критерій найбільшого відхилення, в якому встановлюється абсолютне значення допустимої похибки:

,

де  – максимальна похибка наближення;

 – ділянка апроксимації;

 – поточна похибка наближення.

* критерій середньоквадратичного наближення визначається згідно виразу:

,

де  – допустима середньоквадратична похибка наближення;

 – середньоквадратична похибка наближення.

* інтегральний критерій наближення визначається співвідношенням:

,

де  – допустима середня похибка наближення;

 – середня похибка наближення.

* імовірнісний критерій, відповідно до якого задається допустимий рівень  величини  – ймовірності того, що поточна похибка наближення  не перевище деякого визначеного значення :

.

**2.3.2. Квантування**

*Квантуванням* неперервного сигналу за рівнем називається представлення величини сигналу у вигляді скінченого числа *дозволених рівнів*, що знаходяться один від іншого на скінченний інтервал. Якщо істине миттєве значення рівня сигналу знаходиться всередині цього інтервалу, то замість нього передається найближчий дозволений рівень. Якщо кількість рівнів квантування дорівнює , то переданий при цьому сигнал будет містити не більше  різличних значень.

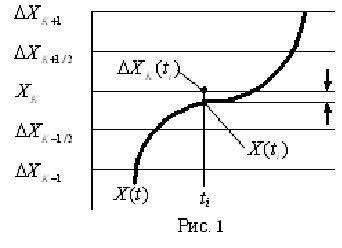
Розрізняють *рівномірне* () та *нерівномірне* () квантування за рівнем. На практиці найпоширенішим є рівномірне (лінійне) квантування завдяки більш простій його технічній реалізації.

При неоднаковій імовірності розподілу значень функції  за шкалою рівнів ефективніше застосовувати нерівномірне квантування, оскільки його основною метою є зменшення усередненої за параметром дисперсії похибки квантування. При такому квантуванні значення неперервної функції , які мають велику ймовірність виникнення, передаються з меншою похибкою квантування, а менш імовірні – з більшою.

Очевидно,  що при квантуванні сигналу виникає похибка в переданих значеннях, обумовлена заміною істиного значення сигналу дозволеним рівнем. Отже, можна вважати, що квантований сигнал  є сумою істиного сигналу  та похибою  (що називається *похибкою квантування* або *шумом квантування*) (рисунок 1):

.

Ймовірність появи рівня  визначається ймовірністю знаходження  сигналу  в інтервалі від   до . Згідно рисунку 1, похибка  може змінюватися в межах від  до  .



Оскільки значення  відоме, то ймовірність появи того або іншого значення похибки  визначається ймовірністю появи відповідного значення сигналу  у момент . Для рівномірного кроку квантування дисперсія похибки буде дорівнювати:

,

де  – крок квантування,

 – функція щільності ймовірності сигналу .

При рівномірному законі розподілу випадкового сигналу в діапазоні від 0 до  дисперсія при рівномірному кроці квантування рівна:

,

а середньоквадратичне відхилення похибки:

,

де ,  – кількість рівнів квантування.

Для нормального та експоненціального законів розподілу функції щільності ймовірності дисперсія похибки квантування визначається з виразу:

.

Передача такого множества различных по уровню импульсов даже на небольшие расстояния применяется крайне редко. Если провести нумерацию уровней, то их передача сведется к передаче чисел. Тогда, выразив эти числа в какой-либо системе счисления, можно обойтись меньшим множеством передаваемых сигналов. Как правило, дискретный сигнал преобразуется в последовательность чисел, выраженных в двоичном коде. Каждое дискретное значение сигнала представляется в этом случае последовательностью сигналов двух уровней. Наличие или отсутствие импульса на определенном месте интерпретируется единицей или нулем в соответствующем разряде двоичного числа.

**2.4. Технічні засоби представлення інформації в цифровій формі**

**2.4.1. Аналого-цифрові перетворювачі**

Пристрої, які приймають вхідні аналогові сигнали та генерують відповідні до них цифрові сигнали, що придатні для обробки мікропроцесорами та іншими цифровими пристроями, називаються *аналого-цифровими перетворювачами* (АЦП).

Принципово не виключена можливість безпосереднього перетворення різних фізичних величин в цифрову форму, однак це завдання вдається розв’язати тільки досить рідко через складність таких перетворювачів. Тому зараз найраціональнішим вважається спосіб перетворення різних за фізичною природою величин спочатку в функціонально пов'язані з ними електричні, а потім уже за допомогою перетворювачів напруга-код – в цифрові. Саме ці перетворювачі й мають на увазі, коли говорять про АЦП.

Зараз відома велика кількість методів перетворення напруга-код. Ці методи суттєво відрізняються один від одного потенційною точністю, швидкістю перетворення та складністю апаратної реалізації. На рисунку 2.5 наведена класифікація АЦП за методами перетворення.

В основу класифікації АЦП покладено ознаку, яка вказує на те, як в часі розгортається процес перетворення аналогової величини в цифрову. В основі перетворення вибіркових значень сигналу в цифрові еквіваленти лежать операції квантування та кодування. Вони можуть проводитись за допомогою або послідовної, або паралельної, або послідовно-паралельної процедур наближення цифрового еквівалента до перетворюваної величини.

Розглянемо детальніше найбільш поширені типи АЦП.

Аналогово-цифрові перетворювачі

Паралельні

Послідовні

Послідовно-паралельні

Послідовного наближення

Послідовного відліку

Інтегруючі

Багатотактні

Багатоступінчасті

Конвеєрні

Однотактні

Багатотактні

Сігма-дельта

Перетворювачі напруга-частота

Рисунок 2.5. Класифікація аналогово-цифрових перетворювачів

Функціонування аналого-цифрового перетворення за *методом послідовного підрахунку* можна проілюструвати за допомогою структурної схеми на рисунку 2.6.

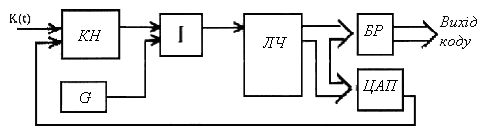


Рисунок 2.6. АЦП послідовного підрахунку

До складу схеми входять: генератор тактових сигналів (G), компаратор напруги (КН), схема І, лічильник (ЛЧ), буферний регістр (БР), цифро-аналоговий перетворювач (ЦАП). Схема працює наступним чином. На вхід перетворювача подається аналоговий сигнал , який підключається до одного з входів компаратора напруги КН. На другий вхід компаратора подається еталонна напруга (), яка формується на виході ЦАП під управлінням колового слова на виході ЛЧ. Компаратор формує на своєму виході сигнал або логічної одиниці, або логічного нуля в залежності від того, яке значення більше. Якщо , то на виході компаратора формується одиниця, яка дозволяє проходження імпульсів з тактового генератора через схему І на лічильний вхід лічильника ЛЧ. На виході лічильника йде процес підрахунку цих імпульсів в двійковому коді від 2 до 2. Двійковий код з ЛЧ подається на вхід ЦАП, на виході якого формується ступінчатий сигнал . Кожна сходинка цього сигналу відповідає за рівнем інтервалу дискретизації . Сигнал порівнюється із сигналом та в момент, коли стає меншим за , на виході компаратора формується сигнал логічного нуля. Схема І закривається, лічильник зупиняє підрахунок і набраний двійковий код переписується у вихідний буферний регістр БР для видачі користувачу.



*Метод безпосереднього зчитування* реалізовується за допомогою так званого АЦП паралельної дії. Такий перетворювач має лінійку 2 компараторів напруги, перші входи яких запаралелені і на них подається сигнал . На інші входи під’єднуються виходи подільника еталонної напруги. Виходи компараторів під’єднані до перетворювача одиничного коду в двійковий. Процес перетворення здійснюється за один такт, причому на виході лінійки компараторів до компаратора, який зафіксує буде хвиля одиниць, а далі хвиля нулів одиничного коду. Структурно-функціональна схема перетворення зображена на рисунку 2.7.

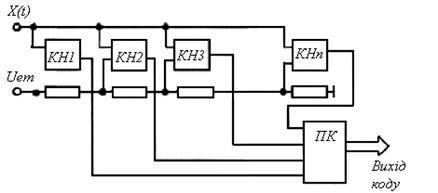


Рисунок 2.7. АЦП безпосереднього зчитування

Найбільше поширення отримав *метод порозрядного зрівноваження*, який забезпечує час перетворення від 1 мкс до 1 мс. Структурно-функціональна схема перетворення зображена на рисунку 2.8 та працює наступним чином. На вхід АЦП подається вхідний сигнал , який порівнюється з еталонним сигналом , що формується на виході ЦАП. ЦАП складається із сукупності еталонних джерел сигналів, які управляються за допомогою спеціального регістра порозрядного зрівноваження (РПУ). Перетворення проходить за часових тактових інтервалів. Причому на першому такті РПУ примусово вмикає в роботу перший розряд ЦАП. Значення першого розряду еталонних величин на виході ЦАП дорівнює половині діапазону перетворення сигналу. Потім в кінці першого тактового інтервалу компаратор проводить порівняння з . Якщо , то примусово увімкнений старший розряд ЦАП залишається ввімкненим до закінчення процесу перетворення. Це забезпечується під управлінням певного сигналу на виході компаратора (1чи 0). Якщо ж , то перший розряд вимикається на початку другого такту. На початку другого такту в роботу примусово вмикається другий розряд ЦАП і знову проводиться порівняння з . Процедура повторюється доти, поки всі розряди ЦАП не візьмуть участі у процесі зрівноваження. В результаті на виході АЦП формується код, що відповідає вхідному сигналу.

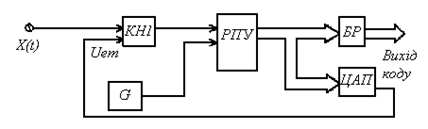


Рисунок 2.8. АЦП порозрядного зрівноваження

Відомо, що недоліком послідовних АЦП є низька завадостійкість результатів перетворення. Дійсно, вибірка миттєвого значення вхідної напруги, переважно включає доданок у вигляді миттєвого значення завади. Згодом при цифровій обробці послідовності вибірок ця складова може бути подавлена, однак на це потрібен час та обчислювальні ресурси. Переважно у АЦП вхідний сигнал інтегрується або неперервно, або у певному часовому діапазоні, тривалість якого зазвичай вибирається кратною періодові завади. Це дозволяє в багатьох випадках приглушити заваду ще на етапі перетворення. Платою за це є понижена швидкодія інтегруючих АЦП.

Спрощена схема АЦП, який працює в два основних такти (АЦП двотактного інтегрування), наведена на рисунку 2.10.

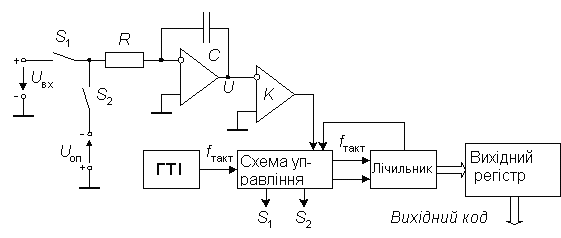
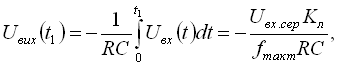


Рисунок 2.10. Спрощена схема АЦП двотактного інтегрування

Перетворення проходить протягом двох стадій: стадії інтегрування та стадії підрахунку. На початку першої стадії ключ замкнутий, а ключ розімкнутий. Інтегратор інтегрує вхідну напругу . Час інтегрування вхідної напруги постійний; як таймер використовується лічильник з коефіцієнтом підрахунку , так, що



До моменту закінчення інтегрування вихідна напруга інтегратора складає



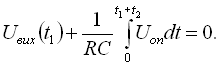
де – середнє за час значення вхідної напруги.



Після закінчення стадії інтегрування ключ розмикається, а ключ замикається та опорна напруга надходить на вхід інтегратора. При цьому вибирається опорна напруга, протилежна за знаком вхідній напрузі. На стадії підрахунку вихідна напруга інтегратора лінійно зменшується за абсолютною величиною.



Стадія підрахунку закінчується, коли вихідна напруга інтегратора переходить через нуль. При цьому компаратор К переключається та підрахунок зупиняється. Діапазон часу, у якому проходить стадія підрахунку, визначається рівнянням



Далі, виконавши прості математичні дії і врахувавши, що:



де – вміст лічильника після закінчення стадії підрахунку,



отримаємо результат



З цієї формули випливає, що відмітною рисою методу багатотактного інтегрування є те, що ні тактова частота, ні постійна інтегрування не впливають на результат. Необхідно тільки, щоб тактова частота протягом часу залишалася постійною. Це можна забезпечити при використанні простого тактового генератора, оскільки істотні часові чи температурні дрейфи частоти відбуваються за час який більший, ніж час перетворення.



*Багатоканальні* АЦП на сьогодні досить поширені, особливо там, де потрібно об’єднати інформацію, отриману від кількох її джерел, тобто, наприклад, від різних сенсорів. Такі АЦП можна застосовувати, наприклад, для моніторингу напруги на входах, контролю крайніх значень, реєстрації показів, управління виходами (навантаженням) тощо.

**2.4.2. Цифро-аналогові перетворювачі**

Необхідність здійснення операції відновлення вихідного сигналу з дискретних відліків, а також необхідність здійснення операцій формування еталонних сигналів при аналого-цифровому перетворенні висуває задачу цифро-аналогового перетворення. Суть операції цифро-аналогового перетворення полягає у формуванні аналогових сигналів, що відповідають кодовим словам дискретного сигналу. Технічно це формування здійснюється *цифро-аналоговим перетворювачем* (ЦАП).

Аналоговий сигнал на виході ЦАП може бути сформований шляхом множення опорної напруги на вагові розрядні коефіцієнти кодового слова , таким чином, що



Технічно найпростіше ЦАП реалізовується на принципі підсумовування розрядних струмів (рисунок  2.15).



Схема реалізації ЦАП для підсумовування струму містить джерело стабільної напруги , матрицю двійково-зважених резисторів , набір ключів , що реалізовують розрядні коефіцієнти та перетворювач струму в напругу на операційному підсилювачі ОП.

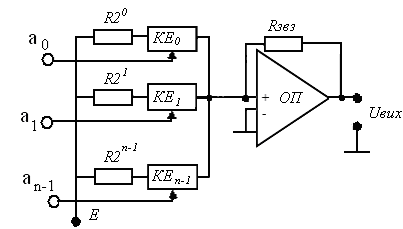


Рисунок 2.15. ЦАП для підсумовування струму

Основні типи електронних ЦАП:

1. [Широтно-імпульсний модулятор](http://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%80%D0%BE%D1%82%D0%BD%D0%BE-%D0%B8%D0%BC%D0%BF%D1%83%D0%BB%D1%8C%D1%81%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D1%8F) – найпростіший тип ЦАП. Стабільне джерело [струму](http://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%82%D0%BE%D0%BA) чи [напруги](http://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%BD%D0%B0%D0%BF%D1%80%D1%8F%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5) періодично вмикається на час, пропорційний перетворюваному цифровому коду, далі отримана імпульсна послідовність фільтрується аналоговим [фільтром низьких частот](http://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%BB%D1%8C%D1%82%D1%80_%D0%BD%D0%B8%D0%B7%D0%BA%D0%B8%D1%85_%D1%87%D0%B0%D1%81%D1%82%D0%BE%D1%82). Такий спосіб часто використовується для керування швидкістю електромоторів, а також стає популярним в Hi-Fi аудіотехніці.

2. ЦАП передискретизації, такі, як дельта-сигма ЦАП, основані на змінюваній густоті імпульсів. Передискретизація дозволяє використовувати ЦАП з меншою розрядністю для досягнення більшої розрядності кінцевого перетворення; часто дельта-сигма ЦАП будується на основі найпростішого однобітового ЦАП, який є практично лінійним. На ЦАП малої розрядності надходить імпульсний сигнал з модульованою густотою імпульсів (з постійною тривалістю імпульсу, але зі змінною шпаруватістю), створений з використанням негативного зворотного зв’язку. Негативний зворотний зв’язок виступає в ролі фільтра високих частот для шуму квантування. Більшість ЦАП більшої розрядності (більше 16 біт) побудовані на цьому принципі внаслідок його високої [лінійності](http://ru.wikipedia.org/w/index.php?title=%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%BE%D1%81%D1%82%D1%8C&action=edit&redlink=1) та низької вартості. Швидкодія дельта-сигма ЦАП сягає сотень тисяч відліків в секунду, розрядність – до 24 біт. Для генерації сигналу з модульованою густотою імпульсів можна використати простий дельта-сигма модулятор першого порядку чи більш високого порядку як MASH. Зі збільшенням частоти передискретизації знижуються вимоги до вихідного фільтра низьких частот та поліпшується приглушення шуму квантування.

3. ЦАП зважування, в якому кожному біту перетворюваного двійкового коду відповідає резистор чи джерело струму, підключене до спільної точки додавання. Сила струму джерела (провідність резистора) пропорційна вазі біта, якому він відповідає. Таким чином, всі ненульові біти коду додаються з вагою. Метод зважування – один з найшвидших, але йому властива низька точність через необхідність наявності набору множини різних прецизійних джерел чи резисторів. Через цю причину ЦАП зважування мають розрядність не більше восьми біт.

4. Ланцюгова R-2R схема є варіацією ЦАП зважування. В R-2R ЦАП зважені значення створюються в спеціальній схемі, яка складається з резисторів опорами R та 2R. Це дозволяє суттєво збільшити точність порівняно зі звичайним ЦАП зважування, оскільки порівняно просто виготовити набір прецизійних елементів з однаковими параметрами. Недоліком методу є більш низька швидкість внаслідок паразитної ємності.

5. Сегментний ЦАП містить по одному джерелу струму чи резистору на кожне можливе значення вихідного сигналу. Так, наприклад, восьмибітовий ЦАП цього типу містить 255 сегментів, а 16-бітовий – 65535. Теоретично, сегментні ЦАП мають найбільшу швидкодію, оскільки для перетворення достатньо замкнути один ключ, який відповідає вхідному коду.

6. Гібридні ЦАП використовують комбінацію перерахованих вище способів. Більшість мікросхем ЦАП належать до цього типу, вибір конкретного набору способів є компромісом між швидкодією, точністю і вартістю ЦАП.

**Висновки**

**Контрольні запитання**

1. Поняття інформації, інформаційного процесу, сигналу, даних, повідомлення, каналу та лінії зв’язку .

2. Що таке об’єм сигналу?

* + 3. Ансамбль і джерело повідомлення. Кількісні міри інформації за Хартлі та Шеноном.
  + 4. Безумовна ентропія та її властивості.
  + 5. Ентропія об’єднаних джерел.
  + 6. Часткова та повна умовні ентропії.
  + 7. Характеристики дискретних джерел інформації. Продуктивність дискретного джерела. Пропускна здатність дискретного каналу.
  + 8. Характеристики неперервних джерел інформації. Пропускна здатність неперервного каналу.
  + 9.Інформаційні втрати при кодуванні неперервних джерел.
  + 10. Кодування. Класифікація кодів та їх характеристики.
  + 11. Способи подання кодів.
  + 12. Основні теореми кодування для каналів передачі інформації.
  + 13. Оптимальне кодування. Код Шенона-Фено.
  + 14. Побудова кода Хафмана.
  + 15. Завадостійке кодування.
  + 16. Коди, що виявляють помилки.
  + 17. Коди, що виправляють помилки.
  + 18. Стиснення даних. Способи стиснення даних при їх передачі та архівації.
  + 19. Наведіть структурну схему системи передачі інформації та опишіть її основні ланки (підсистеми).
  + 20. Що таке швидкість передачі інформації? Яка міра інформації використовується при вимірюванні інформаційної ємності жорстких дисків (вінчестерів) ЕОМ?
  + 21. В чому полягає взаємозв’язок між мірою інформації Хартлі та Шенона?
  + 22. Що таке динамічний та частотний діапазони сигналу?
  + 23. Яку максимальну кількість інформації можна записати на 10 тристабільних елементах?
  + 24. Дайте означення сигналу та інформації.

25. Умовна ентропія та її властивості.

**3. ПОДАННЯ ІНФОРМАЦІЇ В КОМП’ЮТЕРНИХ СИСТЕМАХ. СИСТЕМИ ЧИСЛЕННЯ**

* *Основи систем числення*
* *Переведення чисел з однієї системи числення в іншу*
* *Двійкова, вісімкова та шістнадцяткова системи числення*
* *Форми подання чисел у комп’ютерних системах*

**3.1. Основи систем числення**

*Системою числення* називається метод подання (зображення, записування) чисел, який є спеціальною формальною мовою, алфавітом якої є множина символів, що називаються цифрами, а синтаксисом – правила, що дають змогу однозначно здійснити записування чисел. Подання числа в певній системі числення називають кодом числа у цій системі числення. Коротко число подається таким чином:

.

Окрему позицію в зображенні числа прийнято називати роз­рядом, а номер позиції – номером розряду. Число розрядів у записі числа називається *розрядністю числа*. У технічному аспекті розрядність інтерпретується як довжина розрядної сітки. Якщо алфавіт має  різних значень, то розряд  в числі розглядається як -та цифра, якій може бути при­своєно одне із значень.

Кожній цифрі  даного числа *А*однозначно відповідає її кількісний (числовий) еквівалент – . Кількісний еквівалент коду (запису) числа *А* у визначеній системі числення, є деякою функцією числових еквівалентів усіх його цифр, а саме:

.

Очевидно, що для будь-якої скінченної розрядної сітки кількісний еквівалент числа *А*буде набувати значення від мінімального  до максимального  значення.

Діапазон  подання чисел у даній системі числення – це інтервал числової вісі, що знаходиться між максимальним та мінімальним числами, при заданій розрядності (довжині розрядної сітки)

.

Існує багато методів зображення чисел циф­ровими знаками. Однак із практичної точки зору будь-яка система числення повинна забезпечувати:

* можливість подання будь-якого числа в заданому діапазо­ні чисел;
* однозначність подання;
* компактність та простоту записування чисел;
* легкість оволодіння системою, а також простоту й технічну зручність опе­рування нею.

Залежно від способу записування чисел та способу обчислення їх кількісного еквівалента, системи числення можна класифікувати згідно з рисунком 3.1 [15].

Системи числення

Непозиційні

Позиційні

Ієрогліфічні

Алфавітні

Спеціальні

Однорідні

Неоднорідні

З постійними вагами

Символічні

З природним порядком ваг

Зі штучним порядком ваг

Рисунок 3.1. Класифікація систем числення

Переважно системи числення будують за таким принципом:

,

де  запис числа в системі з базисом ;  – база або по­слідовність цифр системи числення з -тим алфавітом;  – базис системи числення (сукупність ваг окремих роз­рядів числа).

База системи числення може бути додатною, тоді в ній як значення цифр використовується набір 0, 1,…, . Вона може бути також змішаною й тоді в ній поруч з додатними цифрами знаходяться й від’ємні. Наприклад, для симетрич­ної бази з нулем кількість додатних значень цифр дорівнює кількості від’ємних. Значення цифр алфавіту в цьому випадку при  (тобто при непарній основі) формують таку множину:

.

Основою системи числення називається кількість різних символів (цифр), що можуть використовуватися у кожному з розрядів числа для його зображення в даній системі числення.

Системи числення зі змішаною базою можуть бути й при парній основі, але тоді можливе або застосування симетричних алфа­вітів без нуля (наприклад, при  можливий алфавіт +1, -1), або алфавітів, в яких кількість від’ємних значень цифр не дорівнює кількості додатних (наприклад, при  можливий алфавіт -1, 0, 1, 2).

Базис системи числення – це сукупність ваг окремих розрядів системи числення. Наприклад, базис десяткової системи є послідовністю: 1, 10, , …, . Вага роз­ряду числа в будь-якій системі числення – це відношення .Тому цифру  розряду з більшим значенням  називають більш значущою, ніж цифру розряду з меншим .

*Непозиційними системами числення* називають такі системи числення, алфавіт яких має необмежену кількість символів (цифр), при­чому кількісний еквівалент будь-якої цифри сталий і залежить тільки від її графічного написання, але не від позиції в числі. Такі системи будують за принципом адитивності, тобто кількісний еквівалент числа визначається як сума цифр, що стоять поруч:

,

де  – символи, що утворюють базис системи .

Найвідомішими представниками непозиційних систем числення є ієрогліфічні та алфавітні. *Ієрогліфічні системи числення* – це такі системи числення, в яких кожна цифра подана своїм символом, значком або ієрогліфом. *Алфавітні системи числення* – це системи числення, в яких буквам (всім або тільки деяким) приписуються числові значення, які, зазвичай, відповідають порядку букв в алфавіті. Переважно перші дев’ять букв отримують значення від 1 до 9, наступні дев’ять – від 10 до 90 й т.д. Для запису числа записуються букви, сума значень яких виражає це число.

До основних недоліків непозиційних систем числення слід віднести:

* відсутність нуля;
* необхідність використання нескінченної кількості символів;
* складність арифметичних дій із числами.

**3.2. Позиційні системи числення**

*Позиційними системами числення* називають такі, алфавіт яких має скінчену кількість символів, причому значення кож­ної цифри в числі визначається не тільки її написанням, але й знахо­диться в строгій залежності від позиції в записі числа.

Позиційні системи числення мають ряд переваг відносно непозиційних, основною з яких є зручність виконання арифметичних операцій.

У загальному вигляді число *А*в позиційній системі числення може бути записано таким чином:

**,** (3.1)

де  – цифра -горозряду числа, причому  є базою системи числення;  – основа системи числення;  – вага -го розряду числа.

Як бачимо, такі системи будуються не тільки за принципом ади­тивності, але й за принципом мультиплікативності, тобто кількісний еквівалент числа визначається як сума сусідніх цифр зі своїми вагами.

Позиційні системи числення, в свою чергу, поділяються на ряд підкласів (рисунок 3.1). У *неоднорідних позиційних системах числення*  не залежать один від іншого та можуть набувати будь-які значення, ці системи ще називають системами зі змішаною основою. У таких системах числення в кожному -му розряді кількість допустимих символів може бути різною, при цьому , де  – основа системи числе­ння в -му розряді. Прикладом неоднорідної позиційної системи числення є система числення часу.

*Однорідні позиційні системи числення* є частинним випадком позиційних систем числення при  для всіх  та , тобто в них ваги окремих розрядів є рядом членів геометричної прогресії зі знаменником .Тому число в однорідних системах може бути записано у вигляді поліному

, (3.2)

де , а знаменник геометричної прогресії має назву основи системи числе­ння. Очевидно, що основою однорідної позиційної системи може бути будь-яке ціле число, оскільки в означенні позиційних систем числення не накладено ніяких обмежень на величину основи. Тому можлива нескінченна множина позиційних систем числення.

Зазвичай число в однорідній позиційній системі записується у скороченому вигляді

,

а назву системи визначає її основа: десяткова, двійкова, вісімкова тощо.

Для будь-якої позиційної системи числення справедливо, що її основа зображується символами 10 у своїй системі, тобто будь-яке число в своїй системі можна записати символами цієї системи у вигляді

.

Правильний дріб у  системі числення буде мати вигляд



або

 (3.3)

*Кодовані позиційні системи числення* – це такі, в яких цифри однієї системи числення кодуються за допомогою цифр іншої системи, а число в загальному вигляді записується так:

(3.4)

де  – основа системи числення, символами якої коду­ються числа;  – основа початкової системи числення.

При побудові кодованих позиційних систем як ваги розрядів можуть бути вибрані члени геометричної прог­ресії (тобто ваги однорідної позиційної системи числення) та довільні числа. У першому випадку системи називають кодованими системами числення з природними вагами розрядів, у другому – зі штучними вагами розрядів.

Перевагою *систем числення спеціального призначення*, що створені для спрощення або прискорення обчислень у цифрових пристроях, є простота алгоритмів виконання деяких арифметичних операцій, недоліком – необхідність переведення з класичних позиційних систем у спеціальні. Їх використовують для реалізації деяких обчислювальних процесів, у яких не вимагається змінювати систему числення при введенні та виведенні даних, або ця зміна досягається простими засобами.

Наприклад, позиційні системи числення з від’ємною ос­новою дають змогу подати без знака будь-яке натуральне чис­ло, додатне або від’ємне. Однією з таких систем є зрівноважена трійкова система числення, тобто система за основою 3 з цифрами +1, 0, -1 (або ). Приклади записування чисел у цій системі: ; ; .

Перевагами зрівноваженої трійкової системи є:

* знак числа задається його найбільш значущою (старшою) ненульовою цифрою;
* перехід до числа з протилежним знаком відбувається заміною всіх 1 на  та навпаки;
* операція заокруглення до найближчого цілого зводиться до відкидання дробової частини.

Додавати в цій системі дуже просто, якщо врахувати, що , . Віднімання зводиться в переході до числа, що є протилежним за знаком, із наступною операцією додавання. Правила множення на +1 звичайні, а при множені на  знак добутку змінюється на протилежний.

*Приклад 3.1*. Помножити  на :



У *символічних системах*, на відміну від позиційних, цифри є символами, кожен з яких окремо ніяк не ха­рактеризує яке-небудь число. Визначеним комбінаціям цифр умовно поставлені у відповідність визначені числа. Прикладами символічних систем числення є знакологарифмічна система числення та система представлення чисел у залишках або сис­тема залишкових класів (СЗК).

Якщо цілим числам  та відповідає один і той самий залишок від ділення на третє число , то числа  та  називають такими, що є порівняльними за , це відображається записом .Число в СЗК зображується у вигляді залишків від ділення заданого числа на ряд взаємно простих чисел . При цьому утворюється число з вагами розрядів, що відповідно дорівнюють , тобто , де  та , де  – ціла частина . Отже, . Залишок називають також вирахуванням числа *А*за модулем .

В СЗК порозрядними є операція рахунку, а також операції додавання, віднімання та множення. СЗК використовуються в спеціалізованих цифрових пристроях, в яких діапа­зон початкових чисел і проміжних результатів строго фіксований та операція ділення практично відсутня.

**3.3. Переведення чисел з однієї системи числення в іншу**

Існує два основних методи переведення числа з однієї системи числення в іншу: табличний та розрахунковий.

Табличний метод прямого переведення заснований на використанні таблиць відповідності чисел різних систем числення. Цей метод надто громіздкий та вимагає значного об’єму пам’яті для зберігання таблиць, але може бути використаний для будь-яких систем числення (не тільки для позиційних). Суть іншого виду табличного методу полягає в тому, що існують таб­лиці еквівалентів у кожній системі тільки для цифр, тобто баз цих систем та степенів основ (додатних та від’ємних), тобто базису систем. Задача переведення зводиться до того, що у виразах поліномів (3.2) та (3.3) для початкової системи числення подаються еквіваленти з нової системи для всіх цифр та їх ваг розрядів та проводять дії (множення та додавання) за правилами арифмети­ки за новою основою *р.*Отриманий при цьому результат буде зображати число в новій системі числення. Цей метод використовується тільки до позиційних систем числення.

Розрахунковий метод використовується тільки до однорідних позиційних систем числення.

**3.3.1. Переведення цілих чисел з однієї позиційної системи числення в іншу**

Нехай задано число *А*в довільній позиційній системі числення з основою  та його необхідно перевести в нову систему з основою *р*,тобто перетворити до виду

, (3.5)

де  – база нової системи числення. Вираз (3.5) можна записати у вигляді

,

де  – ціла частина частки;  – залишок від ділення *А*на *р*, який є цифрою молодшого розряду шуканого числа, записаного в символах старої системи числення.

При діленні числа  на *р*тим самим способом отримаємо залишок й т.д. Таким чином, вираз (3.5) можна записати за схемою Горнера

,

після чого його права частина послідовно ділиться на основу нової системи .

Отже, в результаті серії ділень початкового числа на основу нової системи числення знаходимо коефіцієнти



При цьому ділення продовжується до тих пір, поки не будуть виконані відношення ; .

*Правило переведення цілих чисел з однієї позиційної системи* *в іншу позиційну систему* формулюється таким чином: щоб перевести ціле число з однієї позиційної системи числення в іншу, необхідно початкове число послідовно ділити на основу нової системи числення, що за­писане в початковій системі, до отримання частки, яка дорівнює нулеві. Число в новій системі числення записується із залишків від ділення, починаючи з останнього.

*Приклад 3.2.* Перевести десяткове число 125 у двійкову та вісімкову систему.

При переведенні з десяткової системи послідовно ділимо початкове число на основи 2 та 8 відповідно:

125

124

1

2

62

62

2

31

30

2

15

14

2

7

6

2

3

2

2

1

0

1

1

1

1

125

120

5

8

15

8

1

7

8

Нове число записуємо з права на ліво, тобто отримаємо .

При переведенні з двійкової системи числення в десяткову початкове число не­обхідно ділити на основу нової системи, тобто .

Ділення виконати в двійковій системі важко. Тому на прак­тиці за необхідності переведення чисел із системи з малою основою в систему з більшою основою зручно користуватися загальним записом чисел у вигляді поліному (3.5). У загальному випадку можна обчислити многочлен  безпосередньо у вигляді

,

представивши в системі за основою *р* ****** та та виконавши всі дії за правилами арифметики системи числення з основою *р*.Наприклад, при переведенні двійкових чисел у десяткову систему числення на практиці зазвичай підраховують суму степенів основи 2, при яких коефіцієнти ****** дорівнюють одиниці. Розрахунки проводять в десятковій системі числення.

*Приклад 3.3.*Перевести число  в десяткову систему:

.

**3.3.2. Переведення правильних дробів**

Нехай правильний дріб *А*,заданий у довільній позиційній системі числення з основою , необхідно перевести в но­ву систему з основою *р*,тобто перетворити його до вигляду

. (3.6)

Якщо, аналогічно переведенню цілих чисел, поділити обидві частини виразу (3.6) на ,тобто помножити на *р*,то отримаємо

,

де  – дробова частина добутку;  – ціла частина результату. Отримана при цьому цифра цілої частини результату й буде пер­шою цифрою шуканого числа.

Помноживши дробову частину результату знову на *р*,отримаємо

,

де  – дробова частина добутку (нового);  – наступна цифра шуканого числа.

Таким чином, при переведенні вираз (3.6) подається за схемою Горнера

.

Проводячи множення його послідовно разів на основу *р*,отримаємо шукане число вновій системі числення.

На відміну від цілих чисел, точне переведення можливе не для всіх правильних дробів. Похибка при цьому складає  молодшого розряду числа в новій системі.

Щоб перевести правильний дріб з однієї позиційної системи в іншу, необхідно початкове число послідовно множити на основу нової системи числення, записану в старій системі числення до отримання заданої точності. Дріб у новій системі числення запишеться у вигляді цілих частин добутків, починаючи з першої частини.

*Приклад 3.4.*Перевести правильний дріб 0,224 з десяткової системи числення в двійкову та вісімкову системи числення.

При переведенні з десяткової системи в двійкову систему числення множимо початковий дріб на 2 (див. таблицю 3.1), а при переведенні у вісімкову – на 8 (див. таблицю 3.2).

Отримаємо .

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблиця 3.1. Переведення 0,224 з десяткової системи у двійкову систему числення   |  |  | | --- | --- | | Ціла частина | Дробова частина | | 0 | 224 | | 0 | 448 | | 0 | 896 | | 1 | 792 | | 1 | 584 | | 1 | 168 | | 0 | 336 | | 0 | 672 | | 1 | 334 | | 0 | 668 | | Таблиця 3.2. Переведення 0,224 з десяткової системи у вісімкову систему числення   |  |  | | --- | --- | |  | Дробова частина | | 0 | 224 | | 1 | 792 | | 6 | 336 | | 2 | 688 | |

При переведенні правильного дробу з двійкової в десяткову систему числення перемножують початкове двійкове число на .

Правильний дріб  можна також перевести у нову систему числення, записавши його у вигляді

.

У цьому разі всі дії виконуються за правилами арифметики но­вої основи ( та  подаються за основою *р*).У цьому випадку необхідно уважно слідкувати за помилками, які можуть з’явитися в результаті відкидання або заокруглення при діленні на .

**3.3.3. Переведення неправильних дробів**

При переведенні неправильних дробів необхідно окремо перевести цілу та дробову частини числа за наведеними вище правилами переведення та записати в новій системі числення, залишивши незмінним поло­ження коми. У випадках, коли бажано забезпечити одноманітність дій, необхідних для переведення, задане число *А*спочатку або ділять на (****** – ціле додатне) так, щоб ,або множать на  ( – число необхідних розрядів дробової частини чис­ла *А*,представленого в новій системі числення за основою *р*)та заокруглюють до найближчого цілого числа. Потім отриманий дріб або ціле число переводять в *р*-ту систему числення. Для збереження кількісного еквівалента отриманий *р*-тий ре­зультат необхідно помножити або поділити відповідно на  або .Практично це означає перенесення коми на *п*розрядів вправо в першому випадку та на ****** розрядів вліво в другому випадку.

**3.4. Двійкова система числення**

Під *двійковою системою числення* розуміють таку систему, в якій для зображення чисел використовуються два символи, а ваги розрядів змінюються за законом  (де  – довільне ціле число). *Класичною двійковою системою* є система з символами 0, 1. Її двійкові цифри називають бітами.

Щоб опанувати будь-яку систему числення, потрібно вміти додавати та множити в ній будь-які цифри. Арифметичні операції в двійковій системі числення виконуються так само, як й в десятковій відповідно до таблиць порозрядних обчислень.

Таблиця 3.3 множень двійкових чисел повністю визначається двома правилами: 1) множення будь-якого числа на 0 дає 0; 2) множення будь-якого числа на 1 залишає його без зміни.

Для додавання є тільки одне правило, згідно з яким додавання 0 до будь-якого числа не змінює цього числа. Тоді таблиця додавання матиме вигляд згідно з таблицею 3.4.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблиця 3.3. Таблиця множення двійкових чисел   |  |  |  | | --- | --- | --- | |  | 0 | 1 | | 0 | 0 | 0 | | 1 | 0 | 1 | | Таблиця 3.4. Таблиця додавання двійкових чисел   |  |  |  | | --- | --- | --- | | + | 0 | 1 | | 0 | 0 | 1 | | 1 | 1 | 10 | |

*Приклад 3.7*. Знайти суму чисел  та :



Як вже зазначалося, в загальному вигляді всі двійкові числа подаються у вигляді поліному

. (3.8)

Переведення в двійкову систему числення з десяткової здійснюється або за загальним правилом переведення чисел з однієї позиційної системи числення в іншу, або десяткові числа переводяться у вісімкову систему за загальним правилом, а потім вісімкові числа переводяться в двійкові за правилом переведення чисел для однорідних позиційних систем з кратною основою. Зворотне переведення робиться аналогічно або за допомогою загального вигляду запису числа (3.8) у вигляді поліному.

**3.5. Двійкова система числення з цифрами 1, **

З означення двійкової системи числення випливає, що для зображення чисел можуть бути використані не лише символи 0, 1, але й символи 1, -1 або 0, -1.

Символ «-1» записують як , а систему, в якій використовуються символи 1, -1, називають системою (1, ). Число  запишеться, наприклад, у цій системі як , якщо у формулі (3.8) припустити, що  набуває значень 1 або .

Особливістю двійкової системи числення з цифрами 1,  є подання єдиним кодом як додатних, так і від’ємних чисел. Проте в ній відсутній нуль та немає можливості подати деякі числа у вигляді скінченної множини.

У системі (1, ) деякі цілі та дробові числа, наприклад парні (), подають у вигляді нескінченних дробів, тобто із певною похибкою.

Водночас, існують числа, які не мають єдиного зображення, наприклад, число 1 може бути подано у вигляді

, . (3.9)

Співвідношення (3.9) відображає зв’язок між звичайною двійковою системою та системою (1, ).

*Приклад 3.8*. Перевести число *А*=100101 у систему (1, ). Використовуючи співвідношення (3.10), зводимо переведення до заміни комбінацій 001 та 01 комбінаціями відповідно  та , тобто .

Для отримання скінченного подання як парних, так і непарних чисел у системі (1, ) використовується такий запис чисел:

. (3.10)

У цьому випадку через адитивний додаток доданих чисел буде менше на одне число, ніж від’ємних. Всього при  можна зображувати  чисел:  від’ємних,  додатних.

Для переведення чисел у систему (1, ) необхідно враховувати наступне: у разі непарного числа переклад здійснюють за правилом (3.9), а потім у розряд  записують одиницю; при переведенні парного числа, його спочатку перетворюють на непарне додаванням одиниці в молодший розряд і тільки після цього переводять у систему (1, ) за правилом (3.9) як непарне число. Потім до отриманого результату в розряд  записують .

*Приклад 3.9*. Перевести в систему (1, ) двійкове число *А*=11000.

Спочатку перетворюємо на непарне число 11001. Після цього замінюємо в зображенні числа комбінацію 001 на комбінацію , приписуємо в розряд після коми цифру  та отримуємо .

Правило переведення з двійкової системи числення з цифрами 0,1 у систему з цифрами 1,  можна сформулювати таким чином.

Для переведення додатних чисел спочатку до початкового числа приписується справа ще один розряд, значення якого є 1. Потім у початковому зображенні виділяють конструкції, що складаються з послідовності нулів та одиниць справа, тобто конструкції виду 00..01. Ці конструкції на основі (3.9) перетворяться до виду , тобто найстарший нульовий розряд замінюється на 1, а інші розряди, включаючи 1 у молодшому розряді конструкції, замінюються на .

*Приклад 3.10*. Число .

Приписуємо справа розряд зі значенням 1: . Виділяємо конструкції виду 000..01. . Перетворюємо вибрані конструкції та отримаємо .

У цьому випадку, додавши 1 справа до початкового запису, реалізується врахування адитивної поправки. Значення цієї поправки повинно бути одне й те саме для всіх чисел, тому якщо в початковому записі дробова частина має менше, ніж  значущих цифр, то її необхідно заповнити нулями до  розрядів та вже в -й розряд записати 1.

*Приклад 3.11*. Число  при , .

Спочатку підготуємо запис числа до перетворення: . Приписуємо справа розряд зі значенням 1 та виділяємо конструкції виду 000..01: . Перетворюємо виділені конструкції та отримаємо .

Для представлення від’ємних чисел у системі з цифрами 1,  спочатку перетворюється запис абсолютної величини числа *А* шляхом дописування зліва та справа необхідної кількості нулів. Потім у цьому зображенні всі 1 замінюються на , а потім до отриманого зображення дописується 1 справа (адитивний доданок). Далі виділяють конструкції виду 00…0 та 00..01, які перетворюються відповідно до вигляду 1…11 та 1….

Двійкову систему числення з цифрами 1,  раціонально використовувати як проміжну при виконанні операцій множення та ділення. Пряма операція арифметичних дій у цій системі суттєво складніша, ніж у системі з цифрами 0 та 1.

**3.6. Шістнадцяткова та вісімкова системи числення**

Двійкова система, зручна для комп'ютерів, для людини незручна через її громіздкість та незвичний запис. Переведення чисел з десяткової системи в двійкову та навпаки виконує машина. Проте, щоб професійно використати комп'ютер, слід навчитися розуміти слово машини. Для цього й розроблені вісімкова та шістнадцяткова системи. Числа в цих системах читаються майже так само легко, як десяткові, вимагають відповідно в три (вісімкова) та в чотири (шістнадцяткова) рази менше розрядів, ніж в двійковій системі (адже числа 8 та 16 – відповідно, третя і четверта степені числа 2).

*Шістнадцяткова система числення* – це позиційна система числення з основою 16. Для запису чисел в **шістнадцятковій системі** використовуються 10 цифр від нуля до дев’яти (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) та латинські букви A, B, C, D, E, F, що позначають числа від 10 до 15. Цифри шістнадцяткової системи числення та їх двійкові еквіваленти наведені в таблиці 3.5.

Таблиця 3.5. Двійкові, десяткові та шістнадцяткові числа

|  |  |  |
| --- | --- | --- |
| Десяткова система | Двійкова система | Шістнадцяткова система |
| 0 | 0000 | 0 |
| 1 | 0001 | 1 |
| 2 | 0010 | 2 |
| 3 | 0011 | 3 |
| 4 | 0100 | 4 |
| 5 | 0101 | 5 |
| 6 | 0110 | 6 |
| 7 | 0111 | 7 |
| 8 | 1000 | 8 |
| 9 | 1001 | 9 |
| 10 | 1010 | A |
| 11 | 1011 | B |
| 12 | 1100 | C |
| 13 | 1101 | D |
| 14 | 1110 | E |
| 15 | 1111 | F |

Числа з шістнадцяткової системи числення досить просто можуть переводитися в двійкову та навпаки, завдячуючи тому, що основа шістнадцяткової системи () рівна двом (основі двійкової системи) в четвертій степені, одному шістнадцятковому розряду відповідає чотири двійкових. Для того щоб перевести двійкове число в шістнадцяткове, необхідно просто розбити двійкове число на групи (тетради) по чотири цифри та перевести кожну таку групу в шістнадцятковий розряд. При цьому розбиття на групи при переведенні цілих чисел відбувається вліво від коми, а дробової частини – вправо. У випадку необхідності неповні тетради доповнюються нулями.

*Приклад 3.12*. Перевести з двійкової системи числення у шістнадцяткову числа: , , .

,

,

.

Для переведення шістнадцяткового числа у двійкове просто замінюють цифру шістнадцяткового числа на відповідну тетраду двійкових чисел.

Якщо необхідно перевести шістнадцяткове число у десяткове, то використовується вже наведена вище схема, наприклад:

.

Максимальне дворозрядне число, яке можна отримати за допомогою шістнадцяткового запису – це FF:

.

255 – це максимальне значення одного байта, що дорівнює 8 бітам: 1111 1111 = FF. Тому за допомогою шістнадцяткової системи числення дуже зручно коротко (за допомогою двох цифр-знаків) записувати значення байтів.

Шістнадцяткова система використовується в цифровій електроніці та комп’ютерній техніці, зокрема у низкорівневому програмуванні на мові асемблера.

*Вісімкова система числення* – це позиційна система числення з основою 8. Для запису чисел у **вісімковій системі** використовується вісім цифр від нуля до семи (0,1,2,3,4,5,6,7). Переведення чисел з вісімкової в двійкову відбувається досить просто за допомогою таблиці 3.6, в якій всі цифри вісімкової системи представлені у вигляді двійкових тріад.

Таблиця 3.6. Таблиця відповідності чисел

у вісімковій та двійковій системах числення

|  |  |
| --- | --- |
| Вісімкова система | Двійкова система |
| 0 | 000 |
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

Для перетворення двійкового числа у вісімкове достатньо розбити його на групи по три числа у кожній та замінити їх відповідними їм цифрами з вісімкової системи числення. Розбивати на трійки потрібно починати з права для цілої частини числа та зліва для дробової частини числа, а цифри, яких не вистачає, замінювати нулями.

*Приклад 3.13.* Перевести з двійкової системи числення у вісімкову числа: , , .

,

,

.

Для переведення вісімкового числа у двійкове просто замінюють цифру вісімкового числа на відповідну тріаду двійкових чисел.

Переведення вісімкового числа в десяткове можна здійснити за вже знайомою схемою, наприклад:



**Вісімкова система числення зараз** практично повністю витіснена шістнадцятковою, але функції переведення числа **з десяткової системи в вісімкову** та навпаки збереглися у всіх програмних та електронних калькуляторах та деяких мовах програмування. Насьогодні вісімкова система використовується для призначення прав доступу до файлів та каталогів в UNIX-подібних системах (Linux, FreeBSD).

**3.7. Вибір системи числення для подання даних у цифрових комп’ютерних системах**

Очевидно, що непозиційні системи числення непридатні для застосування в цифрових автоматах через свою громіздкість і важкість виконання арифметичних операцій. З позиційних систем найзручніші однорідні системи числення, оскільки однакова основа, тобто однакова кількість символів в усіх розрядах призводить до найраціональнішого користування цифровим автоматом та найпростіших алгоритмів виконання арифметичних операцій. При аналізі однорідних позиційних систем числення на предмет їх застосування в цифрових автоматах враховують такі чинники [13-15]:

1. Наявність фізичних елементів, здатних зображувати символи системи.
2. Економічність системи, тобто мінімальну кількість елементів, необхідних для подання багаторозрядних чисел.
3. Громіздкість виконання арифметичних операцій у цифрових автоматах.
4. Швидкодію обчислювальних систем.
5. Наявність формального математичного апарата для аналізу та синтезу обчислювальних пристроїв.
6. Зручність роботи людини з машиною.
7. Рівень завадостійкості кодування цифр на носіях інформації.

Отже, завдання вибору системи числення для застосування в цифрових автоматах зводиться, по-суті, до завдання вибору раціональної величини основи системи *р*.

Виходячи з перерахованих критеріїв, найприйнятнішою – для застосування в цифрових автоматах є однорідна позиційна система числення з основою, що дорівнює двом. Проте в деяких випадках при синтезі обчислювального пристрою якому-небудь критерію надається більше значення, ніж іншим. Тоді для застосування вибирається система числення, оптимальна за вибраним критерієм. Наприклад, у деяких випадках перевагу віддають десятковому численню, керуючись при цьому не міркуваннями економічності вибираного числення, а зручністю спілкування людини з машиною.

У сучасних універсальних комп’ютерах (цифрових автоматах) застосовується як двійкова, так і десяткова системи числення. Причому цифри останньої кодуються двійковими символами, тобто йдеться насправді не про десяткову, а про двійково-десяткову систему числення. Кожна із зазначених систем має свої переваги та недоліки, а також свої сфери застосування.

Перевагами двійкової системи числення відносно двійково-десяткової є: 1) економія близько 20 % устаткування; 2) приблизно в 1,5 раза вища швидкодія розрахунків (не плутати з десятковою системою, у якої швидкодія в 2,7 рази нижче, ніж у двійкової!); 3) спрощення логічної побудови та значна економія устаткування в схемах керування та в допоміжних ланцюгах.

Перевагами двійково-десяткової системи є: 1) відсутність необхідності переведення початкових даних та результатів розрахунків з однієї системи в іншу; 2) зручність контролю проміжних результатів шляхом виведення їх на індикацію для візуального спостереження; 3) ширші можливості для автоматичного контролю через наявність у двійково-десятковому коді надмірних комбінацій.

Двійкову систему числення переважно застосовують для вирішення науково-технічних завдань, для яких характерний великий об’єм обчислень та порівняно малий об’єм початкових даних і результатів обчислень. Її також доцільно застосовувати в комп’ютерах, призначених для управління технологічними процесами.

Двійково-десяткову систему числення переважно застосовують для вирішення економічних завдань, які характеризуються великим об’ємом початкових даних, порівняною простотою та малим об’ємом перетворень, що виконуються над ними, й великою кількістю результатів обчислень. Цю систему також застосовують у калькуляторах, комп’ютерах, призначених для інженерних розрахунків, а також у персональних комп’ютерах.

**3.8. Форми подання чисел у цифрових автоматах**

*Формою подання чисел* у цифрових автоматах називають сукупність правил, що дають змогу встановити взаємну відповідність між записом числа та його кількісним еквівалентом.

*Машинне (автоматне) зображення числа* є поданням числа в розрядній сітці цифрового автомата. Умовне позначення машинного зображення числа, наприклад *A,* записується як [*A*]. Тоді справедливе співвідношення , де  – коефіцієнт, величина якого залежить від форми подання чисел в автоматі.

Через обмежену довжину машинних слів, множина чисел, яку можна подати в машині, скінченна. Порівняння різних форм подання чисел у комп’ютерах зазвичай робиться на основі порівняння *діапазону й точності подання числа*.

У повсякденній практиці найпоширенішою є форма подання чисел у вигляді послідовності цифр, що розділені комою на цілу та дробову частини. Числа, подані в такій формі, називають числами *з природною комою або числами в природній формі*. У природній формі число записується в природному натуральному вигляді, наприклад, 12560 – ціле число, 0,003572 – правильний дріб, 4,89760 – неправильний дріб.

При поданні чисел у такій формі обов’язково потрібно для кожного числа давати вказівку про положення його коми в розрядній сітці, виділеній для подання числа в машині, що вимагає додаткових апаратних витрат досить великого об’єму. Тому в комп’ютерах (цифрових автоматах) набули поширення дві інші форми подання чисел: *з фіксованою та плаваючою комою*.

**3.8.1. Форма подання двійкових чисел із фіксованою комою**

Необхідність у вказанні положення коми відпадає, якщо місце коми в розрядної сітки машини заздалегідь фіксовано раз та назавжди. Така форма подання чисел називають поданням *з фіксованою комою (крапкою)*.

Щоб спростити функціонування автомата, необхідно обмежити вхідну інформацію будь-якою однією областю чисел (на вхід автомата бажано подавати або цілі числа, або правильні дроби, або будь-які числа), це дасть змогу визначити величину масштабного коефіцієнта . Наприклад, якщо на вхід цифрового автомата надходять тільки правильні дроби, то

, (3.11)

де  – машинне зображення числа для форми представлення з фіксованою комою.

Тоді число *А* буде подане у вигляді .

Задачу вибору масштабного коефіцієнта ускладнює необхідність збереження відповідності розрядів усіх чисел, якими оперує цифровий автомат. Нехай є цифровий автомат з розрядною сіткою 12 двійкових розрядів. Необхідно визначити масштабний коефіцієнт для чисел  та . Для того щоб виконувалася умова (3.11), необхідно число, яке більше за абсолютним значенням, записати у вигляді . Звідси , що відповідає величині масштабного коефіцієнта . Число  повинно ввійти в розрядну сітку автомата зі збереженням відповідності розрядів, тобто . Отже, , або .

З прикладу бачимо, що подання чисел у формі з фіксованою комою може призвести до похибки подання числа, що буде розглянуто в кінці цього розділу.

Величина масштабного коефіцієнта , що задовольняє умову (3.11), визначає той факт, що в машинному зображенні кома завжди стоїть після цілої частини дробу, тобто перед її старшим розрядом. Отже, можна зберігати тільки дробову частину числа (цифрову частину), а в розряді цілої частини писати додаткову інформацію.

Оскільки числа бувають додатні та від’ємні, то формат (розрядна сітка) машинного зображення розбивається на знакову частину та поле числа (рисунок 3.2). У полі числа розміщується саме зображення числа, що називається мантисою числа. Для кодування знака числа використовується найстарший розряд розрядної сітки, відведеної для зображення двійкового числа, а інші розряди відводяться під мантису числа.

0

1

2

3



Поле числа

Номер

розряду

Знакова

частина



…

Рисунок 3.2. Подання чисел у форматі з фіксованою комою

Положення коми в розрядній сітці строго фіксується, зазвичай або правіше самого молодшого розряду мантиси, або лівіше від найстаршого. У першому випадку число подається як ціле, в другому – як правильний дріб. Нині в переважній більшості в комп’ютерах у форматі з фіксованою комою подаються цілі числа.

У знакову частину записується інформація про знак числа. Прийнято, що знак позитивного числа "+" зображується символом 0, а знак від’ємного числа "–" – символом 1.

Наприклад, у двійковому коді, використовуючи 6-розрядну сітку, число 7 у формі з фіксованою комою можна подати у вигляді: , де цифра лівіше за точку це знак числа, а п’ять цифр правіше за точку – мантиса числа в прямому коді. Тут мається на увазі, що кома фіксована правіше молодшого розряду, а точка в зображенні числа в даному випадку просто розділяє знаковий біт від мантиси числа. Можна використовувати й іншу форму подання числа в машинній формі: , де знаковий розряд виділяється квадратними дужками.

Кількість розрядів у розрядній сітці, що відведені для зображення мантиси числа, визначає діапазон і точність подання числа з фіксованою комою. Максимальне за абсолютною величиною двійкове число зображується одиницями в усіх розрядах, виключаючи знаковий, тобто для цілого числа

,

де *n* - повна довжина розрядної сітки. У випадку 16-розрядної сітки

,

тобто діапазон подання цілих чисел у цьому випадку буде від  до .

Для випадку, коли кома фіксується правіше молодшого розряду мантиси, тобто для цілих чисел, числа, у яких модуль більший, ніж  та менше одиниці не подаються у формі з фіксованою комою. Числа, за абсолютною величиною менші одиниці молодшого розряду розрядної сітки, називають в цьому випадку *машинним нулем*. Від’ємний нуль заборонений.

У деяких випадках, коли можна оперувати тільки модулями чисел, уся розрядна сітка, включаючи найстарший розряд, відводиться для подання числа, що дає змогу розширити діапазон зображення чисел.

**3.8.2. Подання від’ємних чисел у формі з фіксованою комою**

У комп’ютерах з метою спрощення виконання арифметичних операцій застосовуються спеціальні двійкові коди для подання від’ємних чисел: зворотний та додатковий. За допомогою цих кодів спрощується визначення знака результату операції при алгебраїчному додаванні. Операція віднімання (чи алгебраїчне додавання) зводиться до арифметичного додавання операндів, полегшується вироблення ознак переповнювання розрядної сітки. У результаті спрощуються пристрої комп’ютера, що виконують арифметичні операції.

Відомо, що одним із способів виконання операції віднімання є заміна знака від’ємника на протилежний та додавання його до зменшуваного:

.

Цим операцію арифметичного віднімання замінюють операцією алгебраїчного додавання, яку можна виконати за допомогою двійкових суматорів.

Для машинного подання від’ємних чисел використовують *прямі, додаткові та зворотні коди*. Спрощене означення цих кодів може бути дане таким чином. Якщо число *А* в звичайному двійковому коді – *прямому* *двійковому коді*, зображувати як , тоді число „–*А”* в цьому самому коді подається як , а в *зворотному* (*інверсному*) *коді* це число буде мати вигляд , де , якщо  та , якщо  ( – цифра *i*-того розряду двійкового числа). Отже, при переході від прямого коду до зворотного всі цифри розрядів мантиси числа інвертуються.

*Додатковий код числа*  – це таке машинне зображення цього числа , для якого , якщо  та , якщо , за винятком останнього значущого розряду, для якого  при . Наприклад, число  запишеться в додатковому коді як .

Таким чином, для отримання додаткового коду від’ємних чисел необхідно спочатку інвертувати цифрову частину початкового числа, внаслідок чого виходить його зворотний код, а потім додати одиницю в молодший розряд цифрової частини числа.

Наголосимо, що *додатковий та зворотний коди використовуються тільки для подання від’ємних двійкових чисел у формі з фіксованою комою*. Додатні числа в цих кодах не змінюють свого зображення та подаються як у прямому коді.

На сьогодні в комп’ютерах для подання від’ємних чисел у форматі з фіксованою комою зазвичай використовується додатковий код.

**3.8.3. Форма подання чисел з плаваючою комою**

У нормальній формі

, (3.12)

де  – мантиса числа *А*;  – порядок числа *А* (характеристика числа).

Таке подання чисел не є однозначним, тому для визначеності зазвичай вводять деякі обмеження. Найпоширеніше та зручне для подання в цифрових автоматах обмеження виду

, (3.13)

де  – основа системи числення.

*Нормальна форма подання чисел* – форма подання чисел, для якої справедлива умова (3.13).

Оскільки в цьому випадку абсолютне значення мантиси лежить у межах від  до , де  – кількість розрядів для зображення мантиси без знака, положення розрядів числа в його автоматному зображенні не постійне. Тому таку форму подання чисел називають також *формою подання з плаваючою комою*. Формат машинного зображення числа з плаваючою комою повинен мати знакові частини та поля для мантиси й порядку (рисунок 3.3). Виділяються спеціальні розряди для зображення знака числа (мантиси) та знака порядку або характеристики (рисунок 3.3) [14]. Кодування знаків залишається таким самим, як було з фіксованою комою.

0

1

2

3

4

5

6

7

8

9

10

11

12

13

15

14

Поле порядку

Знак порядку

Номер

розряду

Поле мантиси

Знак мантиси

Мантиса

Порядок

Мантиса

Знак числа

Характеристика

Рисунок 3.3. Подання чисел у формі з плаваючою комою

Розглянемо приклад записування чисел у формі з плаваючою комою.

*Приклад 3.14*. У розрядну сітку цифрового автомата (рисунок 3.3) записати двійкові числа  та  [14].

Спочатку ці числа необхідно записати в нормальній формі (рисунок 3.4). Порядок чисел вибираємо таким чином, щоб для них виконувалась умова (3.13), тобто  та , він має бути записаний у двійковій системі числення. Так як система числення для заданого автомата залишається сталою, то немає необхідності вказувати її основу, достатньо лише представити показник порядку (характеристику числа).

1

1

0

1

1

0

1

1

1

1

0

0

0

1

1

0





0

1

1

0

0

1

0

1

1

1

1

0

0

0

1

1





Рисунок 3.4. Запис чисел  та  у нормальній формі

Оскільки для зображення порядку виділено п’ять цифрових розрядів та один розряд для знака, їх машинне зображення та машинне зображення їх мантис відповідно дорівнюють: , , , .

**3.9. Похибки подання чисел**

Подання числової інформації в цифровому автоматі, як правило, тягне за собою появу похибок, величина яких залежить як від форми подання чисел, так і від довжини розрядної сітки цифрового автомата, тобто комп’ютера. Отже, при переході з однієї системи числення в іншу неминуче виникають похибки.

*Абсолютна похибка подання числа* – це різниця між істинним значенням вхідної величини *А* та її значенням, отриманим із машинного зображення , тобто

. (3.14)

*Відносна похибка подання числа*  дорівнює

. (3.15)

Вхідні величини незалежно від кількості значущих цифр можуть мати грубі похибки, що виникають через помилки при друкуванні, хибних відліків показів деяких пристроїв, некоректної постановки задачі або відсутності повнішої та точнішої інформації. Часто одна величина в одній системі числення має скінченне значення, а в іншій системі числення стає нескінченною величиною. Отже, при переведенні з однієї системи числення в іншу можуть виникнути похибки, оцінити які неважко, якщо відомі істині значення вхідних чисел.

Відповідно до (3.11) числа зображуються в машині у вигляді , де масштабний коефіцієнт  вибирають так, щоб абсолютне значення машинного зображення *А* в системі числення з основою  було завжди менше 1: .

Оскільки довжина розрядної сітки дорівнює  двійкових розрядів після коми, то абсолютна похибка переведення десяткової інформації в систему з основою  буде дорівнювати

. (3.16)

Якщо , то при  максимальне значення цієї похибки дорівнює

. (3.17)

З (3.17) випливає, що максимальна похибка переведення десяткової інформації в двійкову не буде перевищувати одиниці молодшого розряду розрядної сітки автомата. Мінімальна похибка переведення дорівнює нулеві.

Усереднена абсолютна похибка переведення числа в двійкову систему числення .

Для чисел, поданих у формі з фіксованою комою, абсолютне значення машинного зображення числа знаходиться в межах

. (3.18)

Отже, відносна похибка подання для мінімального значення числа . (3.19)

Для комп’ютерів, як правило, , тому , звідки .

Аналогічно для максимального значення

. (3.20)

З (3.20) бачимо, що похибки подання малих чисел у формі з фіксованою комою можуть бути досить значними.

Для подання чисел у формі з плаваючою комою абсолютне значення мантиси

. (3.21)

Похибка (3.17) – похибка мантиси. Для знаходження похибки подання числа в формі з плаваючою комою величину цієї похибки необхідно помножити на величину порядку числа 

 (3.22)

, (3.23)

де  – кількість розрядів для подання мантиси числа.

З (3.22) та (3.23) випливає, що відносна точність подання чисел у формі з плаваючою комою майже не залежить від величини числа.

**Висновки**

1. Системою числення називається метод позначення (запису) чисел, який, в загальному випадку, є спеціальною мовою, алфавіт якої є множина символів, які називаються цифрами, а синтаксисом – правила, що дають змогу однозначно здійснити запис чисел.
2. Позиційними системами числення називаються такі системи числення, алфавіт яких має обмежену кількість символів, причому значення кож­ної цифри в числі визначається не тільки її написанням, але й знахо­диться в строгій залежності від позиції в записі числа.
3. Щоб перевести ціле число з однієї позиційної системи числення в іншу, необхідно початкове число послідовно ділити на основу нової системи числення, що за­писане в початковій системі, до отримання частки, що дорівнює нулеві. Число в новій системі числення записується із залишків від ділення, починаючи з останнього.
4. Щоб перевести правильний дріб із однієї позиційної системи в іншу, необхідно початкове число послідовно множити на основу нової системи числення, записану в старій системі числення до отримання заданої точності. Дріб у новій системі числення записується у вигляді цілих частин добутків, починаючи з першої частини.
5. При переведенні неправильних дробів необхідно окремо перевести цілу та дробову частини числа за вищенаведеними правилами переведення та записати в новій системі числення, залишивши незмінним поло­ження коми.
6. Найприйнятнішою для застосування в цифрових автоматах є однорідна позиційна система числення з основою, що дорівнює двом.
7. Під двійковою системою числення розуміють таку систему, в якій для зображення чисел використовуються два символи, а ваги розрядів змінюються за законом  (де  – довільне ціле число). Класичною двійковою системою є система з символами 0, 1.
8. У цифрових автоматах (комп’ютерах) набули поширення дві форми подання чисел: з фіксованою та плаваючою комами.
9. З метою спрощення виконання арифметичних операцій застосовуються спеціальні двійкові коди для подання від’ємних чисел: зворотний та додатковий. Додатковий та зворотний коди використовуються тільки для подання від’ємних двійкових чисел у формі з фіксованою комою. Додатні числа в цих кодах не змінюють свого зображення та подаються як в прямому коді.
10. Числа у формі з плаваючою комою записуються у нормальній формі. Формат машинного зображення такого числа повинен мати знакові частини та поля для мантиси й порядку (характеристики).

**Контрольні запитання та завдання**

1. Що таке система числення? Назвіть типи систем числення.
2. Як утворюються числа в різних системах числення?
3. Назвіть методи переведення чисел з однієї системи числення в іншу.
4. Наведіть правила переведення цілих чисел та дробів з однієї позиційної системи числення в іншу.
5. Охарактеризуйте двійкову систему числення (класичну двійкову систему та систему (1, ), арифметичні операції в цих системах).
6. Охарактеризуйте шістнадцяткову та вісімкову системи числення.
7. Охарактеризуйте представлення символьної інформації у комп’ютері.
8. Наведіть форми подання чисел у цифрових автоматах та особливості подання від’ємних чисел.
9. Перевести числа 3261, 17671 та  з десяткової системи в двійкову систему числення.
10. Перевести вісімковий дріб  у двійкову систему числення.
11. Перевести числа DDD0, 5BC1 та 213214 з шістнадцяткової системи числення в двійкову систему.
12. Перевести двійкове число 0,0110001002 в систему (1, ).
13. Перевести число  з системи (1, ) у двійкову систему числення.
14. Знайдіть зворотний та додатковий коди чисел -0,1110002 та 0,001012.
15. Записати машинне зображення у формі з плаваючою комою десяткового числа -5,482, якщо для мантиси існує шість двійкових розрядів зі знаком та для порядку – три двійкових розряди зі знаком.
16. Записати машинне зображення у формі з плаваючою комою десяткового числа -6,1382, якщо для мантиси існує вісім двійкових розрядів зі знаком та для порядку – п’ять двійкових розрядів зі знаком.
17. Визначити масштабні коефіцієнти для чисел  та  за умови, що машинне зображення числа має десять двійкових розрядів зі знаком.

**4. АРИФМЕТИЧНІ ДІЇ З ДВІЙКОВИМИ ТА ДЕСЯТКОВИМИ ЧИСЛАМИ В ЦИФРОВИХ КОМП’ЮТЕРНИХ СИСТЕМАХ**

* *Додавання чисел на двійкових суматорах*
* *Множення чисел на двійкових суматорах*
* *Ділення чисел на двійкових суматорах*
* *Виконання арифметичних операцій над Д-кодами в цифрових автоматах*

**4.1. Виконання операцій над двійковими числами**

**4.1.1. Формальні правила двійкової арифметики**

Важливою функцією більшості обчислювальних пристроїв є виконання арифметичних операцій. У зв’язку з цим у комп’ютерах (цифрових автоматах) виділяють спеціальний функціональний блок – арифметико-логічний пристрій, призначений для виконання операцій над числовими кодами, та який у комп’ютерах переважно є складовою процесора.

При виконанні арифметичних дій завжди беруть участь два та більше числа. У результаті арифметичної операції з’являється нове число

, (4.1)

де  – знак арифметичної дії (додавання, віднімання, множення, ділення).

Числа, які беруть участь у арифметичних операціях, що виконуються цифровими автоматами, називають *операндами*.

Оскільки цифровий автомат оперує тільки автоматними зображеннями чисел, то для машинних операцій вираз (4.1) правильніше записати у вигляді

, (4.2)

де в квадратних дужках  – позначення автоматних зображень операндів.

Розглянемо формальні правила виконання арифметичних операцій додавання та віднімання на рівні розрядів операторів.

На основі правил двійкової арифметики (таблиця 3.4) можна записати правила додавання двійкових цифр так, як це показано в таблиці 4.1, де ,  – розряди операндів  та  відповідно;  – розряд суми;  – перенесення з даного розряду в сусідній розряд.

*Двійковий напівсуматор* – пристрій, що виконує арифметичні дії за правилами, вказаними в таблиці 4.1.

Поява одиниці перенесення при додаванні двох розрядів дещо змінює правила додавання двійкових чисел (таблиця 4.2).

Узагальнюючи сказане вище, можна сформулювати правила порозрядних дій при додаванні операндів  та :

, (4.3)

де  – перенесення з (*і*-1)-го розряду;  – перенесення в (*і*+1)-ший розряд (переноси набувають значення 0 або 1).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблиця 4.1. Правила додавання двійкових цифр   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | | 0  0  1  1 | 0  1  0  1 | 0  1  1  0 | 0  0  0  1 | | Таблиця 4.2. Правила додавання двійкових цифр з урахуванням переносу   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  | | 0  0  1  1  0  0  1  1 | 0  1  0  1  0  1  0  1 | 0  0  0  0  1  1  1  1 | 0  1  1  0  1  0  0  1 | 0  0  0  1  0  1  1  1 | |

*Двійковий суматор* – пристрій, що виконує арифметичні дії за правилами, вказаними в таблиці 4.2. Умовні позначення двійкових напівсуматорів та суматорів показані на рисунках 4.1 та 4.2 відповідно.

|  |  |
| --- | --- |
| Рисунок 4.1. Умовне позначення двійкового напівсуматора | Рисунок 4.2. Умовне позначення двійкового суматора |

На основі правил двійкової арифметики можна записати правила віднімання двійкових чисел так, як це показано в таблиці 4.3, де  – запозичення зі старшого розряду. Запозичення рівносильне відніманню одиниці зі старшого розряду. З урахуванням одиниці запозичення зі старшого сусіднього розряду, правила віднімання двійкових чисел можна записати так, як це показано в таблиці 4.4 (щоб відрізняти запозичення від перенесення, перед одиницею поставлений знак мінус).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблиця 4.3. Правила віднімання двійкових цифр   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | | 0  1  1  0 | 0  0  1  1 | 0  1  0  1 | 0  0  0  -1 | | Таблиця 4.4. Правила додавання двійкових цифр з урахуванням запозичення   |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  |  |  |  | | 0  1  1  0  0  1  1  0 | 0  0  1  1  0  0  1  1 | 0  0  0  0  -1  -1  -1  -1 | 0  1  0  1  1  0  1  0 | 0  0  0  -1  -1  0  -1  -1 | |

Якщо  – зменшувальне (перший операнд),  – віднімальне (другий операнд), то для порозрядних дій справедлива рівність

. (4.4)

**4.1.2. Додавання двійкових чисел**

Арифметичні операції можна виконувати з двійковими числами, поданими в прямому, зворотному та додатковому кодах. Якщо операнди подані в прямому коді та мають однакові знаки, то над ними при алгебраїчному додаванні природно виконується процедура додавання. Якщо ж операнди мають різні знаки – процедура віднімання. Як вже зазначалося, для спрощення апаратних засобів комп’ютера процедура віднімання замінюється додаванням завдяки тому, що від’ємний операнд подається у зворотному або додатковому коді.

**4.1.2.1. Алгебраїчне додавання чисел, поданих у формі з фіксованою комою.** Розглянемо кілька видів двійкових суматорів.

*Двійковий суматор прямого коду* (ДСПК) – суматор, у якому відсутній ланцюг порозрядного переносу між старшими цифровими та знаковими розрядами. На ДСПК можна додавати тільки числа, що мають однакові знаки, тобто такий суматор не може виконувати операцію алгебраїчного додавання.

Нехай задані операнди

, ,

де ,  – відповідно вміст знакових розрядів автоматних зображень для  та  (символ походить від англійського слова *sign* – знак); ,  – цифрові розряди зображень.

Якщо , то сума чисел буде мати знак будь-якого з доданків, а цифрова частина результату буде отримана після додавання цифрових частин операндів.

*Приклад 4.1*. Додати числа  та  на ДСПК.

*Приклад 4.2*. Додати числа  та  на ДСПК.

При додаванні чисел на ДСПК можливий випадок, коли абсолютне значення суми операндів перевищує одиницю. Тоді кажуть, що має місце переповнення розрядної сітки автомата. Ознака переповнення – наявність одиниці перенесення зі старшого розряду цифрової частини суматора. У цьому випадку генерується сигнал переповнення , за яким відбувається автоматична зупинка машини та коректування масштабних коефіцієнтів з таким розрахунком, щоб запобігти появі переповнення.

*Двійковий суматор додаткового коду* (ДСДК) – суматор, що оперує зображеннями чисел у додатковому коді. Характерна риса ДСДК – наявність ланцюга порозрядного перенесення зі старшого розряду цифрової частини в знаковий розряд. Правило додавання чисел на ДСДК формулюють так: *сума додаткових кодів чисел є додатковий код результату*. Правило справедливе для всіх випадків, у яких не виникає переповнення розрядної сітки, що дає змогу складати автоматні зображення чисел за правилами двійкової арифметики (таблиця 4.2), не розділюючи знакову та цифрову частини автоматного зображення числа.

*Приклад 4.3*. Додати числа  та  на ДСДК.



*Приклад 4.4*. Додати числа  та  на ДСДК.



*Приклад 4.5*. Додати числа  та  на ДСДК.



*Двійковий суматор зворотного коду* (ДСЗК) – суматор, що оперує зображеннями чисел у зворотному коді. Характерна особливість ДСДК – наявність ланцюга циклічного перенесення зі знакового розряду в молодший розряд цифрової частини. Правило додавання чисел на ДСЗК формулюють так: *сума зворотних кодів чисел є зворотний код результату*. На ДСЗК машинні зображення чисел додаються за правилами, поданими в таблиці 4.2.

*Приклад 4.6*. Додати числа  та  на ДСЗК.



*Приклад 4.7*. Додати числа  та  на ДСЗК.



*Приклад 4.8*. Додати числа  та  на ДСЗК.



*Приклад 4.9*. Додати числа  та  на ДСЗК.



**4.1.2.2. Переповнення розрядної сітки.** При додаванні чисел однакового знака, поданих у формі з фіксованою комою, у цифрових автоматах може виникнути переповнення розрядної сітки.

1. Ознакою переповнення розрядної сітки при додаванні чисел у прямому коді є поява одиниці перенесення зі старшого розряду цифрової частини числа.

2. Ознака переповнення розрядної сітки при додаванні чисел у додатковому та зворотному кодах – отримання знака результату, протилежного до знаків операндів.

Для виявлення переповнення розрядної сітки у складі цифрового автомата мають бути передбачені апаратні засоби, які автоматично генерують ознаку переповнення – сигнал переповнення .

Один із методів виявлення переповнення розрядної сітки передбачає введення допоміжного розряду в знакову частину зображення числа (рисунок 4.3), який називають *розрядом переповнення*. На рисунках 4.4 та 4.5 відповідно подано зображення додатного та від’ємного чисел [14]. Таке зображення числа називають *модифікованим*. Тоді у випадку появи переповнення сигнал ,

, , (4.5)

в інших випадках .

**4.1.2.3. Алгебраїчне додавання чисел, які подано у формі з плаваючою комою.** При виконанні будь-яких арифметичних дій над операндами, поданими у формі з плаваючою комою, операції, що виконуються над мантисами та порядками (чи характеристиками) цих чисел, різні. Тому перед початком будь-якої арифметичної процедури кожен з операндів "розрізається": порядок (характеристика) відділяється від мантиси операнда, щоб можна було над ними виконувати необхідні окремі процедури. Після виконання конкретної арифметичної дії та обов’язкової процедури нормалізації результату, його порядок або характеристика та мантиса "склеюються" у звичайний формат з плаваючою комою.

0

0

1

1

0

1

0

0

1

1

1

Рисунок 4.4. Зображення додатного числа в модифікованому коді

1

1

0

0

1

0

1

1

1

0

0





1

2

3

…

*n-2*

*n-1*

*n*

Знакова

частина

Поле числа

Рисунок 4.5. Зображення від’ємного числа в модифікованому коді

Рисунок 4.3. Зображення числа в модифікованому коді

Розряд переповнення

Оскільки для чисел з плаваючою комою справедлива умова (3.13), то будь-який результат, що не задовольняє цю умову, має бути приведений відповідно до формули (3.13). Таку операцію називають *нормалізацією числа*. Операція нормалізації числа полягає в перевірці умови (3.13) та зсуву автоматного зображення мантиси в той чи інший бік. Зсуви можуть здійснюватися на один розряд та більше, в лівий або правий бік у межах розрядної сітки машини.

*Простий зсув* – операція, що виконується за правилами таблиці 4.5.

Таблиця 4.5. Операція простого зсуву

|  |  |  |
| --- | --- | --- |
| Вхідна комбінація | Зсунута вліво  на один розряд | Зсунута вправо  на один розряд |
|  |  |  |

*Модифікований зсув* – операція над модифікованими зображеннями, яка виконується згідно з таблицею 4.6 (величина залежить від коду: для додаткового коду , для зворотного ).

Таблиця 4.6. Операція модифікованого зсуву

|  |  |  |
| --- | --- | --- |
| Вхідна комбінація | Зсунута вліво  на один розряд | Зсунута вправо  на один розряд |
|  |  |  |

*Порушення нормалізації числа* – невиконання умови (3.13). Оскільки умова (3.13) містить дві нерівності, то може бути порушення справа та зліва. Ознакою порушення нормалізації числа справа  (коли величина результату дорівнює або перевищує одиницю) є наявність різнойменних комбінацій у знакових розрядах суматора, тобто

якщо , (4.6)

(в інших випадках ), де  – ознака порушення нормалізації числа справа, що вказує на необхідність зсуву числа вправо на один розряд.

Ознакою порушення нормалізації числа зліва  (коли результат за власною величиною виявляється менше ) є наявність однакових комбінацій у розряді переповнення та старшому розряді цифрової частини суматора :

якщо  (4.7)

(в інших випадках ), де  – ознака порушення нормалізації, що вказує на необхідність зсуву числа вліво на один розряд.

Таким чином, операція нормалізації числа складається з сукупності зсувів та перевірки наявності ознак порушення  та .

Розглянемо додавання чисел  та , що мають однаковий порядок . Обидві мантиси задовольняють умову нормалізації.

Додавання мантис здійснюється на відповідному суматорі за правилами, викладеним раніше для чисел, які подано у формі з фіксованою комою. Якщо після додавання мантиса результату задовольняє умову нормалізації (тобто ), то до цього результату приписується порядок будь-якого з операндів. В іншому випадку – проходить нормалізація числа.

*Приклад 4.10*. Знайти суму  чисел  та , якщо мантиси та порядок обробляються на суматорах додаткового коду (шість розрядів для мантиси та чотири розряди для порядку) [14].

Спочатку записуються машинні зображення операндів :



потім мантиси додаються



Тут , тобто , значить, потрібний зсув мантиси вліво на розряд:

, .

Одночасно зі зсувом вліво потрібна корекція порядку, тобто зменшення його величини на одиницю (що рівнозначне збільшенню коду на 1,111):



Оскільки після зсуву знову , то здійснюється ще раз зсув та корекція порядку:

Так триватиме до тих пір, поки величина  не дорівнюватиме нулеві. Отже,  задовольняє умову нормалізації та результат дорівнює

, .

Отже, .

*Приклад 4.11*. Знайти суму  чисел  та , якщо числа додаються на суматорі зворотного коду (шість розрядів для мантиси та чотири розряди для порядку) [14].

Машинні зображення операндів записуються в такому вигляді:



Потім складаються мантиси



У цьому випадку сталося порушення нормалізації справа, що вимагає модифікованого зсуву мантиси результату вправо на один розряд:

.

Одночасно зі зсувом робиться корекція порядку результату на величину +0,001, або , в результаті отримуємо .

Розглянемо загальний випадок додавання чисел, які представлені у формі з плаваючою комою, коли їх порядки не дорівнюють один одному, тобто . Для операції додавання чисел необхідною умовою є відповідність розрядів операндів один одному. Отже, спочатку треба зрівняти порядки, що спричинить тимчасове порушення нормалізації одного з доданків. Вирівнювання порядків означає, що порядок меншого числа потрібно збільшити на величину , що означає зсув мантиси меншого числа вправо на кількість розрядів, що дорівнює .

Отже, цифровий автомат повинен самостійно визначати, який із двох операндів менший. На це вкаже знак різниці : додатний знак буде при , а від’ємний – .

Операції додавання та віднімання чисел у формі з плаваючою комою здійснюються в усіх сучасних машинах за наведеними вище правилами.

*Приклад 4.12*. Додати числа  та  на суматорах зворотного коду (шість двійкових розрядів для мантиси та чотири двійкових розряди для порядку) [14].

Спочатку записуються машинні зображення чисел та визначається, який з двох порядків більше:



Величину  позначимо , що означає зміну знака числа  на зворотний, тобто . Тоді .

Оскільки величина  додатна, то . Отже, потрібно зсунути мантису числа вправо на кількість розрядів, що дорівнює , тобто на один розряд:  (зсув модифікований, стрілка над символом  показує зсув у відповідну сторону). Тепер порядки операндів рівні та подальші дії виконуються в послідовності, що аналогічна послідовності попереднього прикладу.

Додається зображення мантис



Здійснюється нормалізація мантиси  та відповідна корекція порядку



Оскільки порушень нормалізації немає, то отримуємо остаточний результат .

*Приклад 4.13*. Додати числа  та , задані у формі з фіксованою точкою: ; ; ;  [14]. Для виконання операції додавання використаємо суматор додаткового коду, що має сім бітів для мантиси зі знаком, чотири біти для характеристики зі знаком. Запишемо машинні зображення мантис ; .

Початкові числа в пам’яті машини можна зберігати або в прямому, або в зворотному (додатковому) кодах. Якщо числа зберігаються в пам’яті машини в прямому коді, то при виконанні операції додавання (віднімання) на суматорах зворотного (додаткового) коду необхідно зробити перетворення з прямого коду у зворотний (додатковий) код. Після закінчення операції необхідно перетворити результат зі зворотного (додаткового) коду в прямий.

При виконанні цього прикладу передбачається, що числа в пам’яті машини зберігаються в додатковому коді.

Спочатку необхідно порівняти характеристики

.

Різниця характеристик – додатна: другий порядок менший першого на 2. Отже, мантиса другого числа зсувається на два розряди (зсув модифікований) та після цього мантиси додаються



Оскільки , то здійснюється зсув вліво на один розряд з корекцією характеристики

.

Таким чином, остаточний результат отриманий у нормалізованому вигляді , .

Останній приклад наведено для випадку, коли мантиса – ціле число та подається у формі з фіксованою комою перед старшим розрядом.

**4.1.3. Множення двійкових чисел**

У двійковій системі числення операція множення може виконуватися у два способи.

1. Множення, починаючи з молодших розрядів множника,



2. Множення, починаючи зі старших розрядів множника,



В обох випадках операція множення складається з ряду послідовних операцій додавання частинних добутків. Операціями додавання управляють розряди множника: якщо в якомусь розряді множника знаходиться одиниця, то до суми частинних добутків додається множене з відповідним зсувом (вліво або вправо), якщо в розряді множника – нуль, то множене не додається, але враховується, що у подальшій операції аналізу розряду множника потрібно зробити додатковий зсув.

Якщо, наприклад, у наступному після нульового розряду множника зустрічається 1, то множене зсувається на 2 розряди та додається до суми частинних добутків. Скільки підряд зустрічатиметься 0, – стільки додаткових зсувів множеного потрібно буде зробити, коли в черговому розряді зустрінеться 1, а потім додавати множене до суми частинних добутків.

Таким чином, окрім операції додавання чисел для отримання добутку потрібна операція зсуву чисел. При цьому з’являється можливість зсувати множене на суму частинних добутків, що дає підставу для різних методів реалізації операції множення.

*Метод 1*. Нехай  – множене (),  – множник (),  – добуток. Тоді у випадку подання чисел у формі з плаваючою комою отримаємо: ; .

Звідси

(4.8)

Множення на  означає зсув на  розрядів вправо числа, яке знаходиться в дужках, тобто в даному випадку зсувається вправо множене та множення починається зі старших розрядів.

*Метод 2*. Нехай  – множене,  – множник.

Множник можна перетворити, використовуючи метод Горнера,

.

Тоді

 (4.9)

Тут множення починається з молодших розрядів та зсувається вправо сума частинних добутків.

*Метод 3*. Нехай  – множене,  – множник.

Множник, використовуючи метод Горнера, можна записати так:



У цьому випадку

 (4.10)

Це означає, що множення починається з молодших розрядів, множене зсувається вліво на один розряд в кожному такті.

*Метод 4*. Нехай  – множене,  – множник.

Якщо множник  записати за методом Горнера

 (4.11)

то множення починається зі старшого розряду та в кожному такті зсувається вліво сума частинних добутків.

Аналіз формул (4.8)-(4.11) показує, що з формальної точки зору процес множення двох чисел може бути поданий:

- при послідовному виконанні – у вигляді багаторазового повторення за кількістю розрядів циклу

, (4.12)

де ,  – суми частинних добутків на -му та -му кроках відповідно;

- при паралельному виконанні – сумою членів діагональної матриці, для якої задані в рядках , в стовпцях – , де  – поточний номер розряду.

**4.1.3.1. Множення чисел, які подано у формі з фіксованою комою, на двійковому суматорі прямого коду.** Нехай задані машинні зображення двох чисел  та . Тоді їх добуток , де  ( – знак операції додавання за модулем 2, що дорівнює , , , ).

*Приклад 4.14*. Перемножити числа  та .

Знак добутку визначаємо окремо від цифрової частини відповідно до рівняння . При множенні будемо використовувати метод 1:



Отже, .

**4.1.3.2. Множення чисел, які подано у формі з фіксованою комою, на двійковому суматорі додаткового коду.** У випадках, коли числа в машині зберігаються в додаткових кодах, доцільно всі операції над числами проводити на суматорах додаткового коду. Але при цьому виникає низка особливостей, які необхідно враховувати.

*Добуток додаткових кодів множників дорівнює додатковому коду результату тільки у випадку додатного множника.*

Нехай множене  – будь-яке число, тобто , а множник . Тоді

. (4.13)

На основі правила про додавання додаткових кодів можна стверджувати, що в правій частині рівняння (4.13) стоїть додатковий код результату.

Таким чином, множення на суматорі додаткового коду полягає в аналізі розрядів множника при  та в додаванні додаткового коду множеного до вмісту суматора. При цьому мають здійснюватися модифіковані зсуви.

Тепер розглянемо випадок, коли  – будь-яке число, а множник . Тоді .

На основі того, що додатковий код є математичним доповненням основі системи числення (, де  – абсолютне значення числа ), то можна записати, що , або . Отже, добуток чисел

. (4.14)

Формула (4.14) показує, що при від’ємному множнику добуток додаткових кодів операндів не дорівнює додатковому коду результату. Якщо ввести заміну  на , то можна вивести наступне правило. *Якщо множник від’ємний, то добуток чисел на суматорі додаткового коду отримується додаванням поправки  до добутку додаткових кодів множників*.

**4.1.3.3. Множення чисел на двійковому суматорі зворотного коду.** Розглянемо правила множення операндів, заданих у зворотному коді.

*Добуток зворотних кодів множників дорівнює зворотному коду результату тільки у випадку додатного множника.*

Нехай множене , а множник . Тоді

. (4.15)

За правилом про додавання зворотних кодів у правій частині даного рівняння отримується зворотний код результату.

Отже, множення на суматорі зворотного коду також полягає в аналізі розрядів множника, та якщо виявиться, що черговий розряд множника дорівнює одиниці, то до вмісту суматора додається зворотний код множеного.

Нехай  та . Тоді . Відповідно до правил перетворення чисел у зворотний код . Отже, .

. (4.16)

На основі (4.16) можна сформулювати наступне правило. *Якщо множник від’ємний, то добуток чисел на суматорі зворотного коду отримується додаванням поправок  та*  *до добутку зворотних кодів множників*.

**4.1.3.4. Множення чисел, які подано у формі з плаваючою комою.** Длячисел, які подано у формі з плаваючою комою, обов’язковим є їх подання у вигляді мантиси та порядку (характеристики). При операції множення дії, що виконуються над мантисами та порядками, різні: мантиси перемножуються, порядки додаються. Очевидно, що результат множення може виявитися ненормалізованим, тоді буде необхідна нормалізація з відповідною корекцією порядку результату.

*Приклад 4.15*. Перемножити числа  та , що задані в прямому коді [14].

Мантиси перемножуємо за правилами, що розглянуті для чисел, поданих у формі з фіксованою комою. Для перемноження мантис використаємо суматор прямого коду, а для додавання порядків – суматор зворотного коду.

Спочатку записуємо машинне зображення чисел



Мантиса добутку дорівнює . Додамо порядки

.

Оскільки мантиса результату не задовольняє умову нормалізації (порушена ліва границя: , ), то проводиться зсув мантиси вліво на один розряд , та корекція порядку

.

Якщо суматор мантис містить тільки  розрядів, то після заокруглення отримаємо результат .

**4.1.4. Ділення двійкових чисел**

**4.1.4.1. Ділення двійкових чисел, які подано у формі з фіксованою комою.** Ділення двійкових чисел багато в чому аналогічне діленню десяткових чисел. У комп’ютерах, як правило, реалізується "шкільний" алгоритм ділення чисел. "Шкільний" алгоритм ділення полягає в тому, що дільник на кожному кроці віднімається від діленого стільки разів (починаючи зі старших розрядів), скільки це можливо для отримання найменшого додатного залишку. Тоді в черговий розряд частки записується цифра, що дорівнює числу дільників, які містяться в діленому на цьому кроці. Інакше кажучи, при діленні операцію віднімання повторюють до тих пір, поки зменшуване не стане менше за те, що віднімається. Число цих повторень показує, скільки разів від’ємник укладається в зменшуваному.

*Приклад 4.16*. Поділимо на .

ділене 11001100 | 1100

дільник 1100 | 10001

00001

- 0

11

- 0

110

- 0

1100

- 1100

0000.

Тут цифри частки отримуються послідовно, починаючи зі старшого розряду, шляхом віднімання дільника з отриманого залишку. Якщо отриманий додатний залишок, то цифра частки дорівнює одиниці; якщо залишок від’ємний, то цифра частки дорівнює нулеві, при цьому відновлюється попередній додатний залишок.

У разі додатного залишку для отримання наступної цифри останній залишок зсувається вліво на один розряд (або дільник управо на один розряд) та з нього віднімається дільник й т. д.

У разі від’ємного залишку відновлюється попередній додатний залишок додаванням до від’ємного залишку дільника та відновлений залишок зсувається на один розряд вліво (або зсувається дільник вправо на один розряд) та з нього віднімається дільник. Такий алгоритм отримав дістав назву алгоритму ділення з відновленням залишку. Формально всі дії можна описати таким чином.

Нехай  – ділене,  – дільник та  – частка, при цьому , , .

Для реалізації алгоритму ділення двійкових чисел, поданих у формі з фіксованою комою, необхідно, щоб виконувалась умова . Якщо ця умова не виконується, то на першому кроці виникає переповнення розрядної сітки та операція не виконується. Якщо , то на першому кроці операції проводиться зсув дільника та визначається залишок .

Нехай , тоді . Процес ділення триває далі: .

Нехай , тоді  та робиться відновлення залишку : .

Цей залишок приймається за  та ділення триває далі таким чином: .

Отже, алгоритм ділення можна описати в загальному вигляді на -му кроці

 , (4.17)

якщо , то  та перехід до наступного кроку;

якщо , то  та відновлення залишку

 , (4.18)

який приймається за залишок  та процес продовжується згідно з формулою (4.17). Отже, операція ділення зводиться до послідовного виконання віднімань (додавань) у суматорі та зсувами дільника. Зсув дільника може бути замінений зсувом вмісту суматора в протилежний бік.

Алгоритм ділення, заснований на реалізації формул (4.17) та (4.18), називається *діленням з відновленням залишку*.

Розглянемо процес відновлення залишку. Коли  та , то на наступному кроці виконуються дії за формулою . Після перетворення отримаємо

. (4.19)

Таким чином, з’являється можливість побудувати алгоритм ділення за схемою , якщо , то  та перехід до наступного кроку; якщо , то  та продовження за формулою . При цьому, якщо , то  та переходимо до формули (4.17), якщо , то  та переходимо до формули (4.19). Такий алгоритм називається *діленням без відновлення залишків*.

**4.1.4.2. Ділення двійкових чисел, які подано у формі з фіксованою комою на суматорах додаткового коду.** При діленні знакова та цифрова частини частки отримуються окремо. Ділення чисел відбувається в три етапи:

1. Знаходиться знак частки за формулою

.

2. Проводиться нульовий крок ділення для перевірки частки на переповнення розрядної сітки шляхом алгебраїчного додавання діленого з дільником, якому приписується знак, протилежний знаку дільника.

3. Для знаходження цифр частки використовують співвідношення згідно з таблицею 4.7 (символ «-» над  вказує на зміну знака на протилежний на наступному кроці ділення). В кінці ділення в -й розряд частки обов’язково додається одиниця.

Таблиця 4.7. Співвідношення для знаходження цифр частки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Знак діленого | + | + | - | - |
| Знак дільника | + | - | + | - |
| Операція в суматорі |  |  |  |  |

За своїм характером операція ділення відноситься до операцій, що дає не завжди точний результат, тому ознакою завершення операції ділення може бути досягнення заданої точності. Якщо в процесі ділення отримали залишок , то операція зупиняється та в розряди частки, що залишилися, записуються нулі. Зазвичай формальною ознакою завершення операції ділення є кількість зсувів: при досягненні кількості зсувів, що дорівнює кількості розрядів частки, в цифрових автоматах генерується сигнал завершення операції ділення.

**4.1.4.3. Особливості ділення чисел, які подано у формі з плаваючою комою.** Для отримання частки від ділення двох чисел, які подано у формі з плаваючою комою, необхідно визначити  та .

Оскільки мантиси діленого та дільника – нормалізовані числа, то при діленні можливі випадки, коли  та .

Якщо мантиса діленого більше або дорівнює мантисі дільника, то у кінці операції ділення знадобиться нормалізація частки (порушення правої границі). Таким чином, алгоритм ділення розпочинається з операції віднімання дільника від діленого та запису одиниці в цілу частину частки. Всі інші дії над мантисами аналогічні діям над числами, які подані у формі з фіксованою комою.

Якщо мантиса діленого менше мантиси дільника, то після операції віднімання на першому кроці отримується від’ємний залишок, що означає нуль в цілій частині мантиси частки та продовження алгоритму ділення за розглянутими вище правилами для чисел, які подані у формі з фіксованою комою. Таким чином, частка завжди виходить у прямому коді, а дії над мантисами здійснюються на ДСЗК або ДСДК.

**4.2. Виконання арифметичних операцій над десятковими числами**

**4.2.1. Подання десяткових чисел в Д-кодах**

*Д-код (двійково-кодоване подання) десяткового числа* – таке його зображення, в якому кожна десяткова цифра зображується тетрадою з двійкових символів

, (4.20)

де  – двійкові розряди тетради ;

*п* – кількість десяткових розрядів.

Кількість різних Д-кодів визначається кількістю можливих поєднань по 10 з 16 комбінацій, які допускає тетрада.

При утворенні Д-коду слід виходити із загальних вимог, що висуваються до систем числення:

* + різним десятковим цифрам повинні відповідати різні тетради;
  + більша десяткова цифра повинна зображуватися більшою тетрадою (якщо розряди тетради мають вагу по двійковій системі числення);
  + для десяткових цифр  та , що пов’язані співвідношенням , повинна задовольнятися умова

  (4.21)

Для однозначності переведення чисел в Д-код та навпаки бажано, щоб розряди тетрад мали певну вагу. Тоді значення десяткової цифри , відповідає виразу

, (4.22)

де  – вага розряду тетради.

У таблиці 4.8 представлено кодування десяткових цифр у різних Д-кодах.

У таблиці для кожного Д-коду вказані дозволені комбінації. Всі інші комбінації – заборонені.

Наявність дозволених та заборонених комбінацій – дуже важлива властивість Д-кодів. Вона відрізняє їх від звичайних позиційних систем числення, в яких усі комбінації дозволені.

Розглянемо найпоширеніші Д-коди.

Код Д1 прямого заміщення (система 8421 – ваги розрядів тетради відповідно дорівнюють 8, 4, 2, 1). У коді Д1 дозволені комбінації відповідають двійковим еквівалентам десяткових цифр з вагами розрядів, що дорівнюють степеням основи 2. Для цього коду не виконується умова (4.21), оскільки цифри, що є доповненням до 9, не виходять простим інвертуванням наборів тетрад.

Таблиця 4.8. Кодування десяткових цифр у різних Д-кодах

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Десяткові числа | Еквіваленти в кодах | | | | | | |
| Д1  (система 8421) | Д2  (система 2421) | Д3  (система 5121) | Д4  (система 8421+3) | Д5  (система 53-21) | Д6  (система 75-31) | Д7  (система 5421) |
| 0  1  2  3  4  5  6  7  8  9 | 0000  0001  0010  0011  0100  0101  0110  0111  1000  1001 | 0000  0001  0010  0011  0100  1011  1100  1101  1110  1111 | 0000  0001  0010  0011  0111  1000  1001  1010  1011  1111 | 0011  0100  0101  0110  0111  1000  1001  1010  1011  1100 | 0000  0001  0111  1010  0101  1000  1001  1111  1100  1101 | 0000  0001  0110  0111  1010  0100  0101  1000  1001  1110 | 0000  0001  0010  0011  0100  1000  1001  1010  1011  1100 |

Для коду Д2 (система 2421) ваги розрядів тетради відповідно дорівнюють 2, 4, 2, 1; таблиця кодування поділяється на дві частини: від 0 до 4 – тетради повторюють двійкові еквіваленти; від 5 до 9 – у порівнянні з двійковою системою кожна тетрада містить надлишок +0110. Це дає можливість будь-яку цифру однієї частини таблиці перетворити на її доповнення до 9 простим інвертуванням.

Для коду Д4 (система 8421+3) усі тетради мають значення на три одиниці більше, ніж тетради коду Д1 (звідси назва коду), та для нього не існує цілочисельних значень ваги, які задовольняли б вираз (3.22).

Коди Д5 (система 53-21) та Д6 (система 75-31) відрізняються від вищеназваних кодів тим, що для них деякі ваги мають від’ємні значення. В обчислювальних машинах різного призначення переважно використовуються коди Д1 та Д4.

**4.2.2. Формальні правила порозрядного додавання десяткових чисел у Д-кодах**

Для задавання формальних правил порозрядного додавання чисел, поданих в Д-коді, розглянемо ті особливості, які властиві цим кодам:

1. Наявність дозволених та заборонених комбінацій. Поява забороненої комбінації при виконанні будь-яких дій над числами свідчить про виникнення помилки або ж про необхідність ввести коригування результату.

2. При додаванні тетрад виникає потетрадне перенесення  замість порозрядного перенесення . Це призводить до необхідності корекції результату.

Насправді, якщо додаються числа  та , то сума  та

, (4.23)

де  – *і-*йрозряд суми;  – перенесення з молодшої тетради;  – перенесення в старшу тетраду; , , .

Далі виведемо правила додавання стосовно Д-кодів.

При додаванні чисел в коді Д1 можуть виникнути такі випадки:

1. Нехай , де  – тетради коду Д1. При додаванні в даному розряді числа утворюється сума менше 10. Якщо дії над розрядами тетради проводяться за правилами двійкової арифметики, то правильний результат отримують без корекції.

*Приклад 4.17*.Додати тетради  та  при значенні :

.

2. Нехай , тобто виникає десяткове перенесення та сума дорівнює , де .

Показником того, що результат невірний, є або поява забороненої комбінації, якщо , або поява потетрадного перенесення , що перевищує значення десяткового перенесення на 6. Отже, необхідна корекція результату в даній тетраді введенням поправки, що дорівнює +0110.

*Приклад 4.18*.Додати тетради  та  при значенні :

.

Величина  – заборонена комбінація. Отже, необхідно ввести поправку



тобторезультат дорівнює 0101 у даній тетраді та утворилося перенесення  в старшу тетраду.

*Приклад 4.19*.Додати тетради  та  при значенні :

.

Поява потетрадного перенесення вимагає корекції результату: . Отже, , .

Приклади, розглянуті вище, дають можливість сформулювати наступні правила потетрадного додавання чисел в Д1-коді:

* якщо при потетрадному додаванні перенесення в сусідню старшу тетраду не виникає (), то результат підсумовування не вимагає корекції;
* корекція результату потетрадного сумування, шляхом додавання поправки +0110 вимагається у разі, якщо:

а) виникає потетрадне перенесення в старшу тетраду ();

б) виникає заборонена комбінація.

Пристрій, який працює за сформульованими вище правилами, називається *однорозрядним десятковим суматором* для Д-коду.

При додаванні чисел у коді Д2 можуть виникнути такі випадки:

1. Нехай  та , де , – тетради для коду Д2. Тоді якщо – результат додавання не вимагає корекції; якщо  – результат попадає в другу частину таб­лиці кодування, де .

Тут необхідна корекція результату, шляхом введення поправки 0110. Ознака цього – поява заборонених комбінацій.

2. Нехай  та ; .

Оскільки , то при додаванні поправка не вимагається.

3. Нехай  та , . Тоді якщо  – результат вимагає корекції шляхом введення поправки -0110, оскільки з’являється заборонена комбінація (виняток з цього випадку виникає при , коли немає необхідності вводити поправку, оскільки з’явиться дозволена комбінація); якщо  – результат не вимагає корекції.

При потетрадному додаванні в Д2-коді результат додавання без корекції виходить в усіх випадках, окрім наступних: якщо за відсутності перенесення в старшу тетраду () виникає заборонена комбінація, то необхідно ввести поправку +0110; якщо за наявності перенесення в старшу тетраду () виникає заборонена комбінація, то вимагається ввести поправку -0110 (1001 в зворотному коді або 1010 – в додатковому коді).

Оскільки поправки бувають додатні або від’ємні, то їх введення супроводжується блокуванням ланцюгів міжтетрадного перенесення в період корекції результату.

*Приклад 4.20.*Додати числа *А*=0001 0011 1101 та В=0100 1011 1101, записані в коді Д2 [14].

Спочатку проводиться потетрадне додавання, а потім кор­екція:



У молодшій тетраді виникає перенесення  та заборонена комбінація – вводиться поправка 1010 в додатковому коді; в старшій тетраді при  виникла заборонена комбінація – поправка вводиться 0110.

Отже, .

При додаванні чисел в коді Д4 можливі наступні випадки. Нехай , , де та – тетради для коду Д4. Тоді: якщо , то ; результат потребує корекції шляхом введення поправки ; якщо , то .

Тут виникає десяткове перенесення, яке, за умовою, "відносить" з собою шість надмірних комбінацій. Отже, в даному випадку потрібно провести корекцію результату шляхом введення поправки .

*Приклад 4.21*. Додати числа *А*=35=0110 1000 та *В*=28=0101 1011, записані в коді Д4 [14].

Спочатку робиться потетрадне додавання, а потім введення поправок:



Отже, *С* = 63=1001 0110Д4.

Тут поправки вводяться при блокуванні ланцюгів потетрадного перенесення.

Правила введення поправок можна сформулювати так: якщо при додаванні тетрад не виникає перенесення , то поправка дорівнює  (чи доповнення ); якщо ж виникає потетрадне перенесення , то поправка дорівнює .

Аналогічним чином можна розглянути правила додавання для інших Д-кодів.

**4.2.3. Подання від’ємних чисел у Д-кодах**

Подавання Д-коду в розрядній сітці машини може здійснюватися у формі з фіксованою або плаваючою комою. При цьому від’ємні числа повинні бути подані в прямому, зворотному або додатковому кодах. Тому, якщо , де  – тетради, то

 (4.24)

де  – доповнення до  в усіх тетрадах;  – доповнення до  в усіх тетрадах, за винятком молодшої, для якої це доповнення до .

З правил перетворення (4.24) виходить, що

. (4.25)

Це означає, що для Д-кодів, для яких виконується умова (4.25), зворотний код виходить простим інвертуванням набору тетрад.

*Приклад 4.22*. Знайти зворотний та додатковий коди в коді Д2 для числа .

На основі (4.24), .

Використовуючи співвідношення , знаходимо додатковий код .

Додавання одиниці в молодшу тетраду при утворенні додаткового коду в коді Д2 здійснюється за правилами додавання для цього коду.

Код Д1 відрізняється тим, що для нього не виконується умова (4.21). Ця особливість коду впливає на утворення зворотного або додаткового коду, оскільки інвертування набору тетрад означає отримання доповнення до . Отже, необхідно забрати різницю. Один з прийомів, який використовується при цьому, полягає в тому, що до всіх цифрових тетрад числа в коді Д1 додається  та після цього робиться інвертування набору. Отримане зображення є зворотним кодом числа.

*Приклад 4.23.*Отримати зворотний код у коді Д1 для числа .

Спочатку до всіх тетрад додається 0110:



Після інвертування цього набору отримаємо .

**4.2.4. Множення чисел у Д-кодах**

Виконання операцій множення в Д-кодах проводиться за класичною схемою: множення чисел зводиться до послідовного підсумовування частинних добутків, отриманих у результаті добутку множеного на чергову цифру множника. Оскільки кожна цифра множника подається у вигляді , де  – номер розряду, то множення супроводжується розшифруванням значення чергової тетради множника та зсувом на чотири розряди відразу. Розшифрування можна здійснити різними способами. Простим прийомом є послідовне віднімання одиниці зі значення тетради до отримання нуля та відповідно додавання множеного в суматор на кожному такті. Оскільки при множенні множеного на тетраду можливе переповнення розрядної сітки суматора (внаслідок того, що множене додається до суми частинних добутків стільки разів, скільки одиниць знаходиться в даному десятковому розряді множника), то в ньому необхідно передбачити додаткову тетраду для перенесень. З чотирьох можливих методів множення в Д-кодах доцільно використовувати тільки один: множення молодших розрядів множника із зсувом суматора частинних добутків вправо. Множення переважно проводиться в прямому коді. Тобто знак результату визначається сумою цифр множників за модулем 2.

**4.2.5. Ділення чисел у Д-кодах**

Ділення десяткових чисел у Д-кодах виконується методом послідовного віднімання дільника з діленого на першому кроці та із залишків – на подальших кроках. Віднімання на кожному кроці робиться до тих пір, поки не вийде від’ємний залишок. Кожного разу при отриманні додатного залишку додається одиниця в спеціальний лічильник, де накопичується чергова цифра частки. Потім здійснюється зсув на чотири двійкові розряди та додавання дільника до тих пір, поки не отримаємо додатний залишок. Кількість додавань (без останнього) є доповненням відповідної цифри частки до 9, що заноситься в лічильник чергової цифри частки.

Таким чином, процес ділення складається з ряду послідовних циклів додавання та віднімання з зсувом. Знак частки виходить як логічна сума за модулем 2 знаків чисел.

Усі дії при виконанні операції ділення повинні здійснюватися на суматорі додаткового (зворотного) коду, який працює за правилами множення – віднімання відповідного Д-коду.

**4.2.6. Переведення двійкових чисел у Д-коди та навпаки**

Розглянемо переведення цілих десяткових чисел, поданих у Д-коді, у двійкову систему числення.

Нехай задане десяткове число , де  – десяткова цифра, що має бути подана в Д-коді у вигляді .

Використовуючи рівність , будь-яке десяткове число можна записати



Множення на  означає зсув двійкового коду на  розрядів вліво. Отже, переведення чисел зводиться до зсуву відповідних тетрад та їх послідовному додаванню.

Для переведення правильних дробів використовують наступний метод. Заданий -розрядний десятковий дріб спочатку розглядають як ціле число та переводять за описаним вище алгоритмом, а потім ділять на , що записане двійковими числами.

*Приклад 4.24*. Перевести  у двійкову систему числення .

,



.

Переведення чисел з двійкової системи числення у Д-код може здійснюватися різними способами. Наприклад, діленням цілих двійкових чисел на число 1010. При цьому десяткові цифри отримуються послідовно одна за одною. При дробових числах ця операція видозмінюється таким чином, щоб при множенні на число 1010 можна було отримати відповідні цифри десяткових дробів.

*Приклад 4.25*. Задано . Знайти двійково-десятковий код цього числа [15].

При знаходженні коду кожної десяткової цифри числа множення  на 10102 замінюємо додаванням  та .



Отже, .

**Висновки**

1. Алгебраїчне додавання двійкових чисел у формі з фіксованою комою проводиться на двійкових суматорах прямого, додаткового та зворотного коду. При цьому від’ємні числа подаються у зворотному або додатковому коді.
2. Алгебраїчне додавання двійкових чисел у формі з плаваючою комою проводиться в такій послідовності: нормалізація числа, відокремлення мантис від порядків, додавання мантис за правилами додавання для чисел з фіксованою комою, перевірка результату умові нормалізації, дописування до мантиси порядку одного з доданків.
3. Множення двійкових чисел у формі з фіксованою комою полягає в послідовності циклів додавання частинних добутків на двійкових суматорах та операції зсуву.
4. При множенні двійкових чисел у формі з плаваючою комою обов’язковою умовою є подання числа у нормальній формі. Операція множення полягає в перемножуванні мантис та додаванні порядків.
5. Ділення двійкових чисел у формі з фіксованою комою полягає в послідовності циклів операцій віднімання на двійкових суматорах та операції зсуву до досягнення заданої точності.
6. При діленні двійкових чисел у формі з плаваючою комою проходить ділення мантис та знаходиться різниця порядків.
7. Д-код десяткового числа – таке подання числа, в якому кожна десяткова цифра зображується тетрадою з двійкових символів. Характерною особливістю Д-кодів є наявність дозволених та заборонених комбінацій.
8. Формальні правила додавання в Д-кодах полягають у потетрадному додаванні чисел з обов’язковою корекцією результату.
9. Виконання операцій множення в Д-кодах проводиться за класичною схемою: множення чисел зводиться до послідовного підсумовування частинних добутків, отриманих у результаті добутку множеного на чергову цифру множника.
10. Процес ділення в Д-кодах складається з ряду послідовних циклів додавання та віднімання зі зсувом.

**Контрольні запитання та завдання**

1. Наведіть формальні правила двійкової арифметики.
2. Дайте означення двійкового суматора. Наведіть їх види та особливості додавання на них.
3. Дайте означення поняття «переповнення розрядної сітки». Коли виникає таке переповнення?
4. Охарактеризуйте правила додавання чисел, що подані у формі з плаваючою комою.
5. Наведіть методи множення двійкових чисел.
6. Охарактеризуйте множення двійкових чисел, поданих у формі з фіксованою комою.
7. Охарактеризуйте особливість множення двійкових чисел, поданих у формі з плаваючою комою.
8. Наведіть принцип ділення двійкових чисел.
9. Дайте означення Д-коду. Назвіть їх особливість відносно інших позиційних систем числення.
10. Наведіть формальні правила порозрядного додавання в Д-кодах.
11. Охарактеризуйте особливості множення та ділення чисел у Д-кодах.
12. Наведіть правила переведення двійкових чисел у Д-коди та навпаки.
13. Додати числа  та  на двійковому суматорі прямого коду.
14. Додати числа  та  на двійковому суматорі зворотного коду.
15. Додати числа  та  на двійковому суматорі додаткового коду.
16. Знайти суму  чисел  та , якщо мантиси та порядок обробляються на суматорах додаткового коду (шість розрядів для мантиси та чотири розряди для порядку).
17. Додати числа  та  на суматорах зворотного коду (шість двійкових розрядів для мантиси та чотири двійкових розряди для порядку).
18. Перемножити числа  та  на суматорах прямого коду.
19. Перемножити числа  та  на суматорах додаткового коду.
20. Перемножити числа  та  на суматорах зворотного коду.
21. Перемножити числа  та , що задані в прямому коді.
22. Поділити задані в прямому коді числа  та .
23. Провести на суматорах зворотного коду ділення чисел  та , що подані у формі з плаваючою комою.
24. Перетворити число  в додатковий код у кодах Д1 та Д2.
25. Перетворити число  у зворотний код у кодах Д1 та Д2.
26. Додати числа  та  на суматорі додаткового коду в коді Д1.
27. Додати числа  та  на суматорі зворотного коду в коді Д2.
28. Додати числа  та  на суматорі додаткового коду в коді Д4.
29. Перемножити числа  та  на суматорі прямого коду в коді Д1.
30. Поділити число  на  на суматорі додаткового коду в коді Д2.