А. М. Гуржій
Н. І. Поворознюк В. В. Самсонов

Інформатика та інформаційні технології

Підручник
для учнів професійно-технічних
навчальних закладів

Харків
«Компанія СМІТ» 2007

ББК 60.84
Г95
УДК 004

Затверджено Міністерством освіти і науки
(лист № 1/11–2043 від 25.06.2002 р.)

Рецензенти:

Тарасенко Володимир Петрович — д-р техн. наук, проф.;
Анненков Віктор Петрович — чл.-кор. АПН України,
канд. пед. наук, доц.

Гуржій А. М., Поворознюк Н. І., Самсонов В. В.
Г95 Інформатика та інформаційні технології: Підручник для учнів професійно-технічних навчальних закладів. — Харків: ООО «Компанія СМІТ», 2007. — 352 с.
ISBN 966-95983-2-X

У підручнику викладено загальні питання і основні поняття інформатики, апаратне і програмне забезпечення комп'ютерних систем, описано операційні системи і комп'ютерні мережі. Розглядаються широко вживані прикладні програми загального призначення. Детально описуються логічні елементи, цифрові пристрої обробки інформації, системи автоматизованого керування і автоматизованого проектування електронних пристроїв.

Для учнів і викладачів навчальних закладів системи професійно-технічної підготовки. Може бути корисна особам, які використовують комп'ютер у повсякденному житті.

ББК 60.84

ISBN 966-95983-2-X

Вступ

Сучасний стан розвитку суспільства характеризується різким зростанням інформаційних потоків не тільки в засобах масової інформації, але й у сфері виробництва, науки, культури. Якщо донедавна ступінь розвитку суспільства визначався ступенем його індустріалізації, то на сьогодні визначається ступенем інформатизації.

У такій важливій сфері людської діяльності як виробництво застосовуються станки з числовим програмним керуванням (ЧПК), гнучкі автоматизовані комплекси, робототехнічні системи, автоматизовані лінії тощо. Комп'ютерні технології інтенсивно запроваджуються як у традиційних галузях виробництва (металообробка, машинобудування, приладобудування тощо), так і в нових, виникнення яких неможливо уявити без застосування комп'ютерної техніки.

Широко застосовуються інформаційні технології у банківській справі, офісній діяльності. Проникають комп'ютери й у сферу управління, починаючи від державного рівня і закінчуючи рівнем міста, села.

Високий рівень інформатизації всіх сфер людської діяльності й, у першу чергу сфери виробництва ставить підвищені вимоги до підготовки фахівців. Комп'ютер стає невід'ємним атрибутом багатьох робочих місць.

Враховуючи важливе значення інформатизації, у школах, закладах професійно-технічної освіти, вузах до навчальних програм було введено курс інформатики. Курс інформатики має свої особливості порівняно з іншими дисциплінами. Це зумовлено, головним чином, надзвичайно швидкими темпами розвитку комп'ютерної техніки. Моделі комп'ютерів, які ще вчора вважалися останнім досягненням техніки, сьогодні стають морально застарілими.

Зважаючи на цю обставину, автори намагалися висвітлювати принципові моменти, основні тенденції розвитку інформаційної техніки, показати перспективи розвитку. З іншого боку, для наближення до практики, наповнення курсу конкретним змістом у підручнику описано пристрої і, особливо, програмні продукти, які вже зарекомендували себе і широко застосовуються у повсякденній практиці.

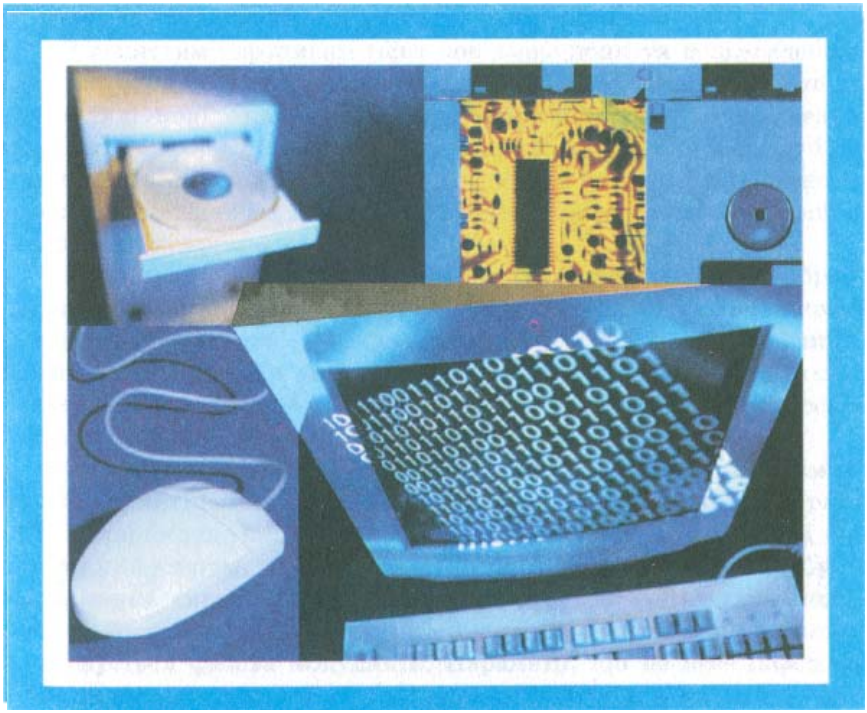
Підручник написано згідно з програмою інформатики, затвердженої Міністерством освіти і науки для навчальних закладів системи професійно-технічної освіти. Книга складається з двох частин: основи інформатики і обчислювальної техніки та комп'ютерно-інформаційні технології.

Першу частину призначено для загальної підготовки учнів професійно-технічної освіти всіх напрямків і профілів, вона складається з десяти розділів, у яких викладено загальні питання й основні поняття інформатики, апаратне і програмне забезпечення комп'ютерних систем, розглянуто операційні системи і комп'ютерні мережі. Крім того, у першій частині описано широко відомі прикладні програми загального призначення: текстовий і графічний редактори, електронні таблиці, бази даних, систему математичної обробки інформації.

Другу частину підручника призначено для професійно-орієнтованої підготовки учнів. Ця частина складається з п'яти розділів, у яких розглянуто логічні елементи, цифрові пристрої обробки інформації, мікропроцесори, викладено основи систем автоматизованого керування і автоматизованого проектування електронних пристроїв. Завершує книгу розділ, присвячений системам дистанційної освіти, — напрямок інформатики, який на сьогодні особливо стрімко розвивається.

1 частина

Основи інформатики та обчислювальної техніки



1

Інформатика. Основні поняття

1.1 Поняття інформації

Інформатика — це наука про інформацію та інформаційні процеси в природі та суспільстві, методи та засоби організації, одержання, зберігання, обробки, транспортування інформації та керування інформаційними процесами.

Основним поняттям інформатики є ***інформація***. Інформація — це одне з фундаментальних понять, тому строгого визначення інформації немає. Під інформацією розуміють факти, відомості, повідомлення, поняття, теоретичні положення, які зменшують початкову невизначеність про об'єкти матеріального світу.

З поняттям інформації тісно пов'язане поняття повідомлення. Людина отримує інформацію через органи зору, слуху, нюху тощо у вигляді повідомлень. Під ***повідомленням*** розуміють деяку кількість інформації, оформлену певним чином для передачі та сприйняття. ***Дані*** — це інформація, записана у формалізованому вигляді і призначена для обробки її технічними засобами, наприклад комп'ютерами.

Інформацію можна сприймати, передавати, зберігати, обробляти тощо. Ці процеси (їх називають інформаційними) пов'язані з сигналами. ***Сигнал*** — це фізичний процес, що несе інформацію. Залежно від природи фізичного процесу сигнали бувають звукові (акустичні), світлові (оптичні), електромагнітні, зокрема радіосигнали, електричні сигнали.

Сигнали мають ***інформативні та неінформативні*** параметри. Інформативний параметр — це параметр, що несе інформацію. Наприклад, якщо як сигнал використовується змінний синусоїдний струм, то інформативним параметром може бути ***амплітуда*** сигналу, якщо використовується амплітудна модуляція, ***частота*** при частотній модуляції, або ***фаза***, якщо використовується фазова модуляція. Параметр, що не несе інформацію, називається неінформативним параметром, наприклад, при амплітудній модуляції частота і фаза сигналу будуть неінформативними параметрами.

Крім сигналів з інформативними параметрами у пристроях обробки інформації діють сигнали, які не несуть інформацію й негативно впливають на інформаційні процеси. Такі паразитні сигнали називаються ***завадами***. Цей термін вперше почав застосовуватися в радіотехніці і позначав сторонні сигнали, які заважали прийому корисних сигналів. Згодом цей термін поширився на інші області техніки, в тому числі й на область інформаційної техніки.

Електричні коливання, миттєві значення яких змінюються хаотично, нерегулярно, непередбачуваним чином і мають широкий спектр, називаються ***шумами***. Цей термін також вперше з'явився в області радіотехніки й означав спочатку хаотичні електричні коливання у звуковому діапазоні частот, які, діючи на навушники телефонів чи гучномовці, створювали звук, схожий на шум моря чи вітру. У подальшому цей термін узагальнили і шумом почали називати хаотичні коливання з широким спектром не тільки у звуковому, а й у будь-якому діапазоні частот. Широко застосовується цей термін і в інформаційній техніці. Шум, спектр якого рівномірний в нескінченно широкій смузі частот від нуля до нескінченності, називається ***білим шумом***. Білий шум — це лише зручна математична модель для аналізу. Реальні сигнали можуть тільки наближатися до цієї моделі.

За місцем виникнення завади поділяються на ***внутрішні***, які виникають у каналах інформаційних пристроїв, і ***зовнішні***, що виникають за межами цих пристроїв.

Значення інформації, для людини зокрема і для суспільства у цілому дуже важливе. Не випадково існує прислів'я: «Хто володіє інформацією, той володіє всім».

У сучасному суспільстві існують заводи для виробництва машин, станків, устаткування, фабрики, що виробляють одяг, побутові товари, продукти харчування. Поряд з ними існують пошта, телеграф, телефон, радіо, телебачення, офіси різних фірм і підприємств, юридичні контори, державні установи. Яку продукцію виробляють ці установи? Відповідь напрошується сама собою — продукцією цих установ є інформація, тобто вони призначені для отримання, обробки, перетворення і поширення інформації.

Розвиток людини і суспільства нерозривно пов'язаний з процесом обміну інформацією.

Примітивні форми обміну інформацією спостерігаються вже у тваринному світі: окремими звуками, позами і жестами тварини повідомляють про свій стан, координують свої дії.

Виділення людини з тваринного світу сталося великою мірою завдяки різкому зростанню обміну інформацією між людьми. Пер-

вісні люди вирізнялися в тваринному світі значно більшим використанням жестів, міміки, звуків для координації своїх дій.

Важливим етапом, значення якого важко оцінити, є формування мови і мовлення, які різко збільшили інтенсивність обміну інформацією, сприяли формуванню суспільства.

Подальша інтенсифікація інформаційних процесів у суспільстві пов'язана із запровадженням писемності. Важливі історичні події та інформацію стало можливим фіксувати на матеріальних носіях інформації: папері, глиняних та дерев'яних дощечках тощо.

Винайдення радіо дало змогу передавати інформацію майже миттєво у різні куточки світу, утворився єдиний світовий інформаційний простір.

Із винайденням пристроїв обробки інформації — електронно-обчислювальних машин (ЕОМ) — різко зросла продуктивність обробки інформації.

1.2 Інформаційні процеси

Процеси зміни і розвитку інформації називаються ***інформаційними***. Розглянемо основні інформаційні процеси.

Сприйняття інформації

Щоб мати уявлення про реальні об'єкти і процеси у навколишньому світі, їх властивості і поведінку, потрібно спочатку сприйняти інформацію.

Людина отримує інформацію про навколишній світ за допомогою органів чуттів: зору, слуху, нюху тощо.

До складу технічних систем, зокрема систем обробки інформації, входять спеціальні пристрої, призначені для сприйняття інформації, вони називаються ***датчиками, сенсорами, рецепторами***. Наприклад, термопара або терморезистор є датчиками температури; фотодіод — датчик світлового потоку; мініатюрна турбінка, встановлена у водопровідній трубі,— датчик швидкості води. Фізичні величини, діючи на вхід датчика, призводять до відповідної зміни вихідного сигналу, наприклад, підвищення температури середовища, де знаходиться термопара, до збільшення вихідної електрорушійної сили термопари; збільшення світлового потоку, що падає на фотодіод, веде до збільшення струму фотодіода.

Велике значення датчики мають для технічних систем, які діють автоматично, без участі людини, наприклад технічні роботи.

Перетворення інформації

Існують різні форми і види інформації. Вони відіграють важливу роль в інформаційних процесах, наприклад, зберігати інформацію зручніше і простіше в одній формі, а передавати — в іншій, тому є потреба у перетворенні інформації з однієї форми в іншу. Технічні пристрої, призначені для перетворення інформації з однієї форми в іншу, називаються ***перетворювачами***. Наприклад, люди, спілкуючись між собою, обмінюються інформацією у звуковій формі. Мікрофон перетворює звуки на електричний сигнал, отже мікрофон є перетворювачем інформації. Електричний сигнал підсилюється і перетворюється у радіосигнал, який поширюється на велику відстань. Радіосигнал приймається антеною і перетворюється в електричний сигнал. Далі електричний сигнал за допомогою гучномовця перетворюється у звуковий сигнал.

Різноманітні перетворення інформації здійснюються й у комп'ютерних інформаційних системах. Процесор комп'ютера обробляє інформацію у формі електричних імпульсів, у пам'яті інформація зберігається у формі електричних зарядів або у магнітній формі, тому в процесі роботи інформація перетворюється з одного виду в інший. Крім того, потрібно перетворювати інформацію при виводі на екран дисплея, на друк. За допомогою клавіатури текстова, числова і керівна інформація перетворюється на електричні сигнали. Застосовуючи сканер, текстову і графічну інформацію перетворюють на електричні сигнали.

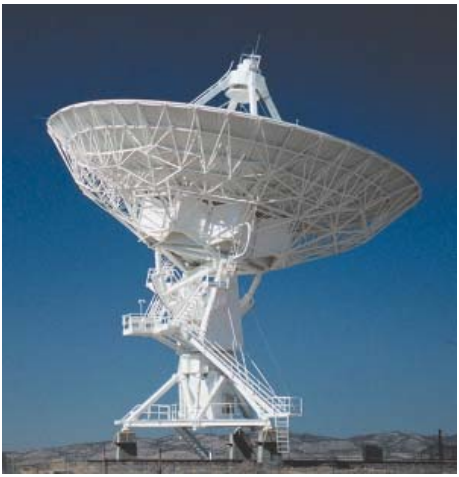


Рис. 1.1

Передача інформації

Цінність інформації багато в чому зумовлена тим, що її можна передавати на великі відстані за допомогою супутників. На рисунку 1.1 зображена антена для передачі і прийому інформації з супутника. Передають інформацію за допомогою сигналів ***каналами зв'язку***. Сигнал — фізичний процес, що несе інформацію. За фізичною природою розрізняють звукові, світлові, радіосигнали та ін.

Велике значення і широке застосування мають радіосигнали. Радіомовлення, телебачення, мобільний телефонний зв'язок

забезпечуються радіосигналами. Завдяки застосуванню радіосигналів функціонує глобальна комп'ютерна мережа Інтернет.

Останнім часом широкого поширення набули ***оптичні сигнали*** і ***волоконно-оптичні*** канали зв'язку. На відміну від радіоканалів волоконно-оптичні канали зв'язку мають більшу швидкість передачі інформації та захист від завад.

Зберігання інформації

Щоб скористатися інформацією у будь-який момент часу, її необхідно зберігати. Проблема зберігання інформації виникла задовго до винайдення і широкого застосування комп'ютера. Ще з давніх часів до нас дійшли наскальні зображення людей і тварин, виконані первісними людьми. У Стародавньому Вавилоні інформацію зберігали на глиняних плитках, у Стародавньому Єгипті — на папірусі, у Київській Русі — на дерев'яних дощечках. Глиняні плитки, папірус, дерев'яні дощечки — це приклади ***носіїв інформації***. Великого прогресу було досягнуто із винайденням, промисловим виготовленням і широким упровадженням паперу. Папір і в наш час широко застосовується як носій інформації. Люди в основному сприймають інформацію, що нанесена на папір.

У комп'ютері, де доступ до інформації вимірюється частками мікросекунди, знайшли застосування електронні, магнітні, оптичні носії інформації.

Технічний пристрій, призначений для зберігання інформації на носіях різноманітної фізичної природи, називається ***пам'яттю*** або ***системою пам'яті***. Однією з найважливіших характеристик пам'яті є ***доступ до пам'яті***, він визначається часом, необхідним для того, щоб записати або прочитати інформацію, що знаходиться в пам'яті.

Обробка інформації

Обробкою інформації називається процес отримання нової інформації на основі наявної. Наочним прикладом є числова інформація, де результат обчислення отримується в результаті операцій з вхідними даними. Основним засобом обробки інформації є комп'ютер.

1.3 Комп'ютер — основний технічний засіб обробки інформації

Розвиток виробництва і технологій ускладнюється, що призводить до багаторазового збільшення інформаційних потоків. Обробити ці інформаційні потоки без застосування комп'ютера неможливо, тому комп'ютер у наш час став основним технічним

засобом обробки інформації. На перших етапах свого розвитку комп'ютери застосовувалися здебільшого для виконання складних розрахунків і обчислень, тому цей технічний пристрій отримав назву ***електронно-обчислювальна машина*** (***ЕОМ***). І сьогодні складні розрахунки і обчислення здійснюються на комп'ютері, однак цим функції комп'ютера не обмежуються. Сфера застосувань комп'ютерів значно розширилася. За допомогою сучасних комп'ютерів здійснюється переклад книг різними мовами, керування пристроями, механізмами, машинами, складними виробничими процесами і технологіями, контролюється склад і якість різних виробів. Застосовуються комп'ютери і для підвищення ефективності навчального процесу.

Комп'ютер або електронно-обчислювальна машина будується за принципами, які вперше сформульовані американським вченим Дж. фон Нейманом.

1. Принцип програмного керування. Цей принцип полягає у тому, що розв'язання задач на комп'ютері здійснюється автоматично, під керуванням програми — послідовності команд, здійснюючи які комп'ютер виконує відповідну послідовність операцій обробки інформації. Ефективність програмного керування буде високою тоді, коли за раніше складеною програмою розв'язується велика кількість однотипних задач, що відрізняються одна від одної тільки різними вхідними даними.

2. Принцип програми, що зберігається в пам'яті комп'ютера у вигляді чисел. Вхідні і вихідні дані обробляються так само, як і числа, тими самими пристроями.

3. Принцип адресності, який полягає у тому, що вхідні дані і програми їх обробки зберігаються в пам'яті за певними адресами, що значно полегшує роботу пристроїв комп'ютера.

Комп'ютер є складним технічним пристроєм. Засоби, з яких він складається, можна розділити на дві частини:

- ❒ апаратну, до складу якої входять різні технічні пристрої, вузли, механізми, елементи;
- ❒ програмну, до складу якої входить комплекс різноманітних програм.

Апаратну частину називають ще ***апаратним забезпеченням*** (англ. термін hardware — твердий продукт), а програмну частину — ***програмним забезпеченням*** (англ. термін software — м'який продукт).

Апаратне забезпечення

Головним пристроєм комп'ютера є ***процесор*** — пристрій для виконання операцій з обробки інформації (даних). Процесор працює

під керуванням програми. Програма — це послідовність дій (операцій), які має виконувати процесор.

Вхідні дані для прикладних задач, а також проміжні і кінцеві результати зберігаються у пам'яті комп'ютера. У пам'яті зберігаються і коди команд програм, під керуванням яких здійснюється розв'язок прикладних задач.

Пристрій введення призначений для введення у комп'ютер вхідних даних і програм, ***пристрій виведення*** — для виведення результатів.

Програмне забезпечення

Як уже зазначалося раніше, одним із основних принципів побудови комп'ютерів є принцип програмного керування, який означає, що комп'ютер розв'язує задачу в автономному режимі під керуванням програми, тобто послідовності команд, яку склали заздалегідь і записали у пам'ять комп'ютера. Комплекс програм, призначених для забезпечення роботи комп'ютера, називається ***програмним забезпеченням*** комп'ютера. Його можна поділити на дві частини: ***базове*** (***системне***) програмне забезпечення і ***прикладне*** програмне забезпечення.

Системне програмне забезпечення складають програми, що призначені для взаємодії користувача з апаратними засобами і для взаємодії апаратних пристроїв один з одним і з центральним процесором. Основні програми системного забезпечення входять до складу ***операційної системи*** комп'ютера.

Крім операційної системи до системного програмного забезпечення відносяться ***сервісні програми***, які призначені для створення архівів, тестування пристроїв, боротьби з комп'ютерними вірусами тощо.

Для взаємодії з периферійними пристроями призначені спеціальні програми — ***драйвери***.

Для розробки прикладних програм призначені комплекси спеціальних програм — ***інтегровані програмні середовища***, які базуються на певній алгоритмічній мові програмування.

Для перетворення програм, написаних алгоритмічними мовами високого рівня у машинний код, призначені спеціальні програми — ***транслятори***, які в свою чергу поділяються на ***компілятори*** та ***інтерпретатори***.

Прикладне програмне забезпечення складають програми, призначені для розв'язання прикладних задач у різних галузях людської діяльності. Найпоширенішими прикладними програмами є:

- ❒ текстові редактори;
- ❒ електронні таблиці;
- ❒ системи керування базами даних;

- ❒ навчальні програми для шкіл, середніх і вищих навчальних закладів;
- ❒ програми перекладу з однієї мови на іншу;
- ❒ програми верстки друкованої продукції;
- ❒ системи автоматичного проектування тощо.

1.4 Історія і перспективи розвитку комп'ютерної техніки

З давніх часів люди створювали механізми і машини для полегшення своєї фізичної праці. Довгий час розумова праця людини здійснювалася без участі машин. Але з розвитком суспільства з'явилася потреба у складних обчисленнях і розрахунках. Особливо гострою стала проблема з відкриттям нових земель і пов'язаний з нею розвиток мореплавства. Для навігації потрібно було точно визначати координати судна в океані за зірками та іншими небесними тілами. Для цього необхідно було робити складні обчислення і розрахунки, складати таблиці руху небесних тіл. Ця робота вимагала багато сил і часу.

Перші машини для виконання обчислень були механічними. Так проект першого механічного комп'ютера був запропонований в Англії Бебіджем. Механічні обчислювальні пристрої були дуже повільними.

Електронні пристрої мали незрівнянно більшу швидкодію порівняно з механічними. Перший електронний комп'ютер створений у США в 1946 р. у Пенсільванському університеті. Ця машина мала назву ENIAC (Electronic Numeric Integrator and Calculator). Вона важила 30 т, містила 18 000 електронних ламп і могла виконувати 5000 операцій за секунду.

ENIAC та інші електронно-обчислювальні машини, в яких використовувались електронні лампи,— це машини першого покоління.

У 1955 році з'явилися машини другого покоління. Замість електронних ламп у них використовувалися напівпровідникові прилади — транзистори. Завдяки цьому електронно-обчислювальні машини стали меншими за розмірами, споживали менше електроенергії, мали більшу продуктивність — кілька десятків тисяч операцій за секунду. Тоді ж стали використовуватися мови програмування.

Наступний етап розвитку електроніки пов'язаний з розробкою і випуском інтегральних схем. ***Інтегральна схема*** — це електронний пристрій, що складається з напівпровідникових елементів

(транзисторів, діодів, резисторів, конденсаторів), виготовлених на напівпровідниковому кристалі в єдиному технологічному процесі (рис. 1.2). Технологія виготовлення інтегральних схем дала змогу в багато разів зменшити розміри, а відповідно і споживання енергії, напівпровідникових елементів (транзисторів, діодів тощо). Це призвело до значного збільшення швидкодії. Комп'ютери або як їх тоді називали електронно-обчислювальні машини, виготовлені із застосуванням інтегральних мікросхем, змонтованих на платах друкованого монтажу (рис. 1.3), — це електронно-обчислювальні машини третього покоління.

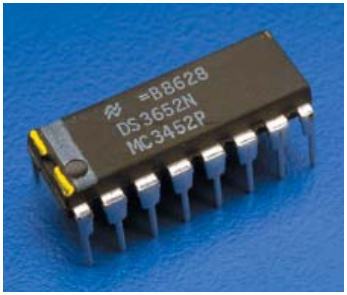


Рис. 1.2

Технологія виготовлення інтегральних мікросхем продовжує вдосконалюватися надзвичайно високими темпами, внаслідок чого з'являється мікропроцесор, тобто процесор, виготовлений на одному напівпровідниковому кристалі. За такими ж технологіями виготовляють мікросхеми пам'яті. Інтегральні мікросхеми, що містять на одному напівпровідниковому кристалі (чипі) кілька сотень тисяч транзисторів, стали називати ***великими інтегральними мікросхемами***. Електронно-обчислювальні машини, основними елементами яких є мікропроцесори та великі інтегральні схеми, вважаються машинами четвертого покоління.

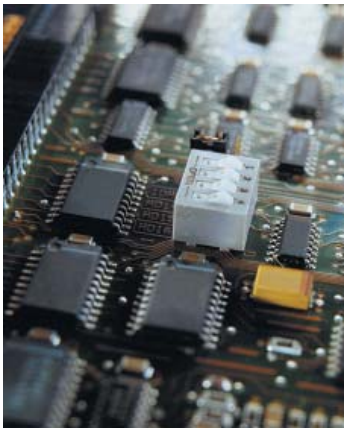


Рис. 1.3

Перехід до машин п'ятого покоління пов'язують із розробкою та впровадженням надвеликих інтегральних мікросхем, у яких кількість транзисторів на одному кристалі перевищує мільйон і тактова частота досягає кількох сотень мегагерц. Збільшення продуктивності апаратних засобів дало змогу розробити нові підходи і технології програмних засобів. Почали активно впроваджуватися об'єктно-орієнтовані й візуальні технології програмування. Нові апаратні та програмні засоби призвели до різкого зростання «інтелектуальних» можливостей комп'ютера. Стало можливим реалізувати тривимірну графіку (3D технології), засоби анімації і мультимедійні засоби.

1.5 Комп'ютерні мережі

Широке застосування комп'ютерів та інформаційних технологій, зростання інформаційних потоків гостро поставили питання про об'єднання комп'ютерів у єдину систему — комп'ютерну мережу. Створення комп'ютерних мереж забезпечує єдиний інформаційний простір для багатьох споживачів. Комп'ютерні мережі дають доступ користувачам до спільних ресурсів: бази даних, апаратних пристроїв (принтерів, накопичувачів тощо), програмних засобів.

Комп'ютерна мережа — сукупність територіально рознесених комп'ютерів, з'єднаних між собою каналами зв'язку. Кожний комп'ютер підключається до мережі, як правило, через спеціальний мережний засіб — вузол комутації або концентратор. Комп'ютери, підключені до мережі, мають свою адресу і часто називаються абонентськими пунктами або робочими станціями. Вони можуть обмінюватися між собою інформацією.

Апаратні засоби, що виконують у мережі функції керування та забезпечують взаємодію комп'ютерів, називаються серверами.

1.6 Числова інформація і системи числення

Властивості об'єктів матеріального світу в якісному і кількісному аспектах виражають через фізичні величини. Довжина, маса, об'єм, електричний заряд, магнітна проникність, світловий потік — всі ці фізичні величини характеризують властивості об'єктів і процесів матеріального світу. Кількісну сторону фізичної величини виражають числом.

На письмі числа позначаються за допомогою спеціальних символів — ***цифр***. Цифри, якими ми користуємося, називаються ***арабськими*** цифрами, тому що в Європу вони потрапили у середні віки через арабів, хоча виникли вони в Індії. Арабські цифри — це такі десять символів: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Отже, цифри — це своєрідний алфавіт числової писемності.

Числа виражаються у певній системі числення. ***Система числення*** — спосіб запису чисел на письмі за допомогою цифр. У повсякденному житті ми користуємося ***десятковою*** системою числення, в якій числа записуються за допомогою арабських цифр.

Системи числення поділяються на позиційні та непозиційні. Якщо значення кожної цифри залежить від місця (позиції), де стоїть цифра, то система називається ***позиційною***. Звична для нас десяткова система числення є позиційною. Наприклад, у десятковому числі 959, перша цифра 9 означає кількість сотень одиниць, друга цифра 5 — кількість десятків одиниць, а остання цифра 9 — кількість одиниць. В основному використовуються позиційні системи числення. Це обумовлено тим, що математичні дії над числами у позиційних системах числення виконувати значно зручніше і простіше, ніж у непозиційних. Крім того, запис чисел у позиційній системі значно компактніший, ніж у непозиційній.

Іноді використовують і ***непозиційні*** системи числення, в яких немає строгої залежності значення цифри від місця (позиції), яку вона займає. Прикладом такої системи є система римських цифр. У цій системі використовуються символи (цифри), які означають: I — одиницю, V — п'ять одиниць, X — десять одиниць, L — п'ятдесят одиниць, C — сотню одиниць і т. д. Будь-яке число записується комбінацією цих символів, наприклад, десяткове число 88 у системі римських цифр запишеться так: LXXXVIII. У цьому числі цифра X, яка стоїть на другому, третьому і четвертому місці (позиції), означає одне і те ж — десять одиниць.

Позиційні системи числення будуються за єдиним принципом. Обирається основа системи числення. ***Основа*** системи числення — кількість символів, тобто цифр, за допомогою яких виражається будь-яке число. Основою звичної для нас десяткової позиційної системи числення є десять символів (цифр) — 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Позиції, на яких стоять цифри, називаються ***розрядами числа***.

Будь-яке число в системі числення з основою p виражається через степені основи і цифри, що складають основу, за формулою:

$$a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + \ldots + a_1 \cdot p^1 + a_0 \cdot p^0 + \\ + a_{-1} \cdot p^{-1} + a_{-2} \cdot p^{-2} + \ldots + a_{-m} \cdot p^{-m}, \qquad (1.1)$$

де a_n, a_{n-1}, …, a_1, a_0, a_{-1}, a_{-2}, …, a_{-m} — числа, менші за основу і виражені цифрами. Кожний доданок у рівнянні (1.1) — це розряд числа. На письмі числа у системі числення з основою p записуються тільки своїми коефіцієнтами розрядів a_n, a_{n-1}, …, a_1, a_0, a_{-1}, a_{-2}, …, a_{-m}. Між коефіцієнтами, що стоять перед додатними і від'ємними показниками степеня, ставиться кома (у деяких країнах, наприклад у США, прийнято ставити не кому, а крапку). Розглянемо, наприклад, число $986{,}75_{10}$ у десятковій системі числення. Система числення позначається своєю основою, записаною як

нижній індекс, у даному випадку нижнім індексом $_{10}$ позначена десяткова система числення. Десяткове число $986{,}75_{10}$ має п'ять десяткових розрядів: сотень одиниць, десятків одиниць, одиниць, десятих часток одиниці і сотих часток одиниці, і його можна записати згідно з формулою (1.1) так:

$$9 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0 + 7 \cdot 10^{-1} + 5 \cdot 10^{-2},$$

тобто, десяткове число $986{,}75_{10}$ має дев'ять сотень одиниць, вісім десятків одиниць, шість одиниць, сім десятих часток одиниці і п'ять сотих часток одиниці.

Двійкова система числення

Як уже зазначалося, ми користуємося позиційною десятковою системою числення. Комп'ютер зберігає і обробляє числа, що записані у позиційній ***двійковій*** системі числення. Двійкова система числення має основу **2** і, відповідно, всього два символи (цифри) для кодування чисел: **0** і **1**. Застосування двійкової системи числення у комп'ютерних технологіях обумовлено зручністю і простотою кодування всього двох символів **0** і **1** технічними засобами, а саме: двом символам відповідають два альтернативні фізичні стани — наявність чи відсутність струму в транзисторах, намагніченість чи відсутність намагніченості ділянки магнітного матеріалу, віддзеркалення чи відсутність віддзеркалення світлового променя тощо.

Будь-яке число можна записати у двійковій системі за допомогою двох символів **0** і **1** і степенів числа **2** за формулою:

$$a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + \ldots + a_1 \cdot 2^1 + a_0 \cdot 2^0, a_{-1} \cdot 2^{-1} + \\ + a_{-2} \cdot 2^{-2} + \ldots + a_{-m} \cdot 2^{-m}, \qquad (1.2)$$

де a_n, a_{n-1}, …, a_1, a_0, a_{-1}, a_{-2}, …, a_{-m} — коефіцієнти розрядів числа, які набувають значення **0** і **1**. У формулу (1.2) входять степені числа 2. Наведемо таблицю значень степенів числа 2.

Таблиця 1.1

n	0	1	2	3	4	5	6	7	8	9	10
2^n	1	2	4	8	16	32	64	128	256	512	1024

Запишемо десяткові числа, які не перевищують основи системи числення, тобто 10, еквівалентними їм двійковими числами, користуючись таблицею 1.1.

$$0_{10} = 0 \cdot 2^0 = 0_2$$
$$1_{10} = 1 \cdot 2^0 = 1_2$$
$$2_{10} = 1 \cdot 2^1 + 0 \cdot 2^0 = 10_2$$
$$3_{10} = 1 \cdot 2^1 + 1 \cdot 2^0 = 11_2$$
$$4_{10} = 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 100_2$$
$$5_{10} = 1 \cdot 2^2 + 0 \cdot 2^0 + 1 \cdot 2^0 = 101_2$$
$$6_{10} = 1 \cdot 2^2 + 1 \cdot 2^0 + 0 \cdot 2^0 = 110_2$$
$$7_{10} = 1 \cdot 2^2 + 1 \cdot 2^0 + 1 \cdot 2^0 = 111_2$$
$$8_{10} = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 1000_2$$
$$9_{10} = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1001_2$$
$$10_{10} = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 1010_2$$

Розглянемо правила виконання арифметичних дій над одноразрядними двійковими числами.

Додавання	Множення
$0 + 0 = 0$	$0 \cdot 0 = 0$
$0 + 1 = 1$	$0 \cdot 1 = 0$
$1 + 0 = 1$	$1 \cdot 0 = 0$
$1 + 1 = 10$	$1 \cdot 1 = 1$

Додавання, віднімання, множення і ділення виконуються за тими ж правилами, що й для десяткових чисел, наприклад:

$$\begin{array}{r} 1010 \\ +\ 1100 \\ \hline 10110 \end{array} \qquad \begin{array}{r} 1100 \\ -\ 1010 \\ \hline 0010 \end{array} \qquad \begin{array}{r} 100 \\ \times\ 101 \\ \hline 100 \\ 100 \\ \hline 10100 \end{array} \qquad \begin{array}{r|l} 11001 & 101 \\ \cline{2-2} 101 & 101 \\ \cline{1-1} 101 & \\ 101 & \\ \cline{1-1} 0 & \end{array}$$

Перетворення чисел із однієї системи числення в іншу здійснюється перетворенням кожного розряду і основи згідно з формулою (1.2), наприклад:

$$286_{10} = 2 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0 = 11_2 \cdot (1010_2)^2 +$$
$$+\ 1000_2 \cdot (1010_2)^1 + 110_2 \cdot (1010_2)^0 = 100011110_2.$$

Інформація у комп'ютері зберігається і обробляється у двійковій системі числення, тому і вимірюється у двійкових одиницях.

Одиниця двійкової системи числення, тобто наймолодший розряд цілого двійкового числа, називається ***бітом*** **(bit)**. Інформацію прийнято вимірювати в одиницях двійкової системи числення, тобто у бітах.

Інформація, що містить восьмирозрядне двійкове слово, називається ***байтом***. Широко застосовуються також такі одиниці:

кілобайт $= 2^{10}$ байти $= 1024$ байти;
мегабайт $= 2^{10}$ кілобайти $= 1024$ кілобайти $= 2^{20}$ байти;
гігабайт $= 2^{10}$ мегабайти $= 2^{30}$ байти;
терабайт $= 2^{10}$ гігабайти $= 2^{40}$ байти.

Сукупність бітів, що зберігається і обробляється комп'ютером як одне ціле, називається ***словом*** (machine word, computer word). Слово може бути однобайтним, двобайтним і т. д.

Шістнадцяткова система числення. Крім двійкової системи в комп'ютері широко застосовується шістнадцяткова система числення, зокрема для запису адрес, за якими зберігається інформація в пам'яті машини. Основою шістнадцяткової системи числення є число 16, тобто числа у цій системі записуються за допомогою шістнадцяти символів — шістнадцяткових цифр. Перші десять символів шістнадцяткової системи — це цифри десяткової системи, тобто арабські цифри 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Наступні шість цифр — це перші шість букв латинського алфавіту (латиниці) A, B, C, D, E, F.

Запишемо відповідність між шістнадцятковими числами, що не перевищують основи, і числами двійкової та десяткової систем.

$$
\begin{aligned}
\mathbf{0}_{16} &= 0_{10} = 0_2 \\
\mathbf{1}_{16} &= 1_{10} = 1_2 \\
\mathbf{2}_{16} &= 2_{10} = 10_2 \\
\mathbf{3}_{16} &= 3_{10} = 11_2 \\
\mathbf{4}_{16} &= 4_{10} = 100_2 \\
\mathbf{5}_{16} &= 5_{10} = 101_2 \\
\mathbf{6}_{16} &= 6_{10} = 110_2 \\
\mathbf{7}_{16} &= 7_{10} = 111_2 \\
\mathbf{8}_{16} &= 8_{10} = 1000_2 \\
\mathbf{9}_{16} &= 9_{10} = 1001_2 \\
\mathbf{A}_{16} &= 10_{10} = 1010_2 \\
\mathbf{B}_{16} &= 11_{10} = 1011_2 \\
\mathbf{C}_{16} &= 12_{10} = 1100_2 \\
\mathbf{D}_{16} &= 13_{10} = 1101_2 \\
\mathbf{E}_{16} &= 14_{10} = 1110_2 \\
\mathbf{F}_{16} &= 15_{10} = 1111_2
\end{aligned}
$$

Перетворення чисел з шістнадцяткової системи у двійкову і навпаки здійснюється дуже просто: кожній двійковій тетраді (сукупності з чотирьох розрядів) відповідає одне однорозрядне шістнадцяткове число і навпаки, наприклад:

$$10110111_2 = B7_{16} \qquad 3C9_{16} = 001111001001_2.$$

1.7 Знакові системи

Важливою формою передачі інформації є письмо (книги, брошури, журнали, газети є письмовою формою інформації). На письмі інформація передається за допомогою символів або ***знаків***. Людство у процесі свого розвитку виробило дві форми писемності: піктографічне письмо за допомогою ієрогліфів і текстове письмо за допомогою букв або літер. Сукупність букв, які використовуються для даної мови, називається ***алфавітом***, ***азбукою***. Більшість народів світу користується алфавітною системою письма. Піктографічна система письма збереглася тільки в деяких східних країнах (Китай, Японія).

У країнах Заходу використовується латинський алфавіт — латиниця. Україна, Росія і деякі інші слов'янські країни використовують кирилицю — алфавіт, створений великими слов'янськими просвітителями Кирилом і Мефодієм.

Кодування символів алфавіту. Як уже зазначалося, інформація у комп'ютері зберігається й обробляється у вигляді двійкових чисел. Значна частина інформації, яку сприймає людина, уявляється у формі тексту. Отже, щоб обробляти текстову інформацію за допомогою комп'ютера, необхідно її перетворити у форму двійкових чисел. Для цього достатньо перетворити у форму двійкових чисел кожний текстовий символ, тобто кожну букву алфавіту, розділові знаки, цифри і спеціальні знаки.

1. Що означають терміни інформація, повідомлення, сигнал?
2. Якими параметрами характеризується сигнал?
3. Що таке завада і шум?
4. Які інформаційні процеси ви знаєте і чим вони характерні?
5. Що таке комп'ютер і для чого він призначений?
6. Назвіть основні етапи розвитку комп'ютерної техніки.
7. Що таке система числення?
8. Що є основою і розрядом числа у позиційній системі числення? Які особливості двійкової системи числення?
9. Що таке біт, байт, кілобайт, мегабайт?

2

Апаратні засоби комп'ютерної системи обробки інформації

2.1 Структура комп'ютерної системи обробки інформації

Обробка інформації — один із важливих і трудомістких інформаційних процесів. Власне обробка інформації і є змістом розумової діяльності людини. З прогресом суспільства і з різким зростанням інформаційних потоків неможливо собі уявити обробку такої кількості інформації без застосування спеціальних технічних пристроїв для обробки інформації. На даному етапі розвитку суспільства таким пристроєм є електронно-обчислювальна машина (комп'ютер). Значна частина інформації обробляється за допомогою комп'ютерних систем, основним елементом яких є персональні комп'ютери.

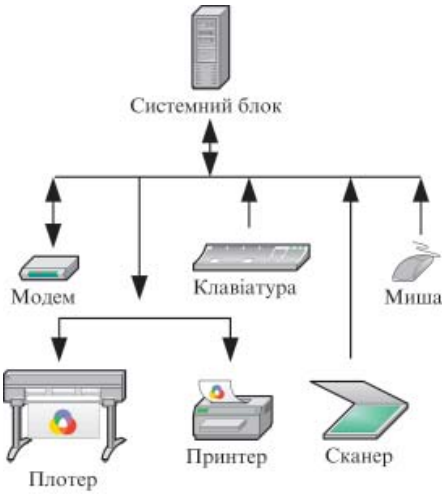


Рис. 2.1

Комп'ютерна система обробки інформації, як вже зазначалося, складається з апаратної (апаратне забезпечення) і програмної (програмне забезпечення) частин.

Розглянемо детальніше апаратну частину. Апаратна частина комп'ютерної системи — це сукупність різноманітних технічних пристроїв для обробки інформації. Структура апаратної частини наведена на рисунку 2.1. До складу апаратної частини комп'ютерної системи входять такі пристрої: системний блок, призначений для обробки інформації і керування іншими пристроями; периферійні пристрої, призначені для введення, виведення і обміну інформацією.

Системний блок

До складу системного блока входять пристрої, що здійснюють основні операції обробки інформації.

Конструкція системного блока може бути або у вигляді плоского прямокутного корпусу (desktop), на якому зверху розміщується монітор, або у вигляді вузького високого корпусу баштового (tower) типу. Всередині корпусу кріпиться так звана материнська плата (Materboard) з багатошаровим друкованим монтажем. На материнській платі монтуються мікросхеми мікропроцесора, пам'яті, спеціальні контактні пристрої (слоти) для монтажу вторинних (дочірніх) плат. Конструкція цих контактних пристроїв забезпечує швидку заміну мікросхем чи вторинних плат у разі їх несправності або модернізації. На вторинних платах монтуються мікросхеми оперативної пам'яті, мікросхеми для взаємодії з периферійними пристроями, додаткові мікросхеми і пристрої. Деякі мікросхеми монтуються не на вторинних платах, а безпосередньо в контактних пристроях на материнській платі.

Крім материнської плати з мікросхемами і вторинними платами у корпусі системного блока кріпляться пристрої для запису і читання інформації: твердий магнітний диск (вінчестер), гнучкий магнітний диск (флопі-диск), оптичний диск (CD–ROM).

Електрична енергія, необхідна для роботи пристроїв, поступає від ***блока живлення***, який також кріпиться всередині корпусу системного блока (рис. 2.2).

Рис. 2.2

Пристрої, що входять до системного блока, з'єднані між собою джгутами для передачі електричних сигналів. Електричні сигнали надходять до вхідних і вихідних контактних пристроїв, розміщених на задній стінці корпусу.

Системний блок побудований за магістрально-модульним принципом, тобто всі пристрої виконані у вигляді окремих модулів, що з'єднуються магістраллю (системною шиною), за допомогою якої модулі обмінюються інформацією.

Материнська плата

Структура материнської плати (рис. 2.3) також магістрально-модульна: всі пристрої виготовлені у вигляді окремих мікросхем або сукупності мікросхем і обмінюються інформацією через систему шин (магістралей): шину даних, шину адреси, шину керування. На материнській платі знаходяться такі пристрої:

- ***центральний процесор***, який виконує основні операції з обробки інформації, керує роботою і координує дії інших пристроїв апаратної частини;
- ***співпроцесор***, призначений для виконання складних математичних операцій;
- ***електронна пам'ять***, призначена для зберігання даних, команд і адресної інформації;
- ***пристрої керування (контролери)*** периферійним обладнанням.

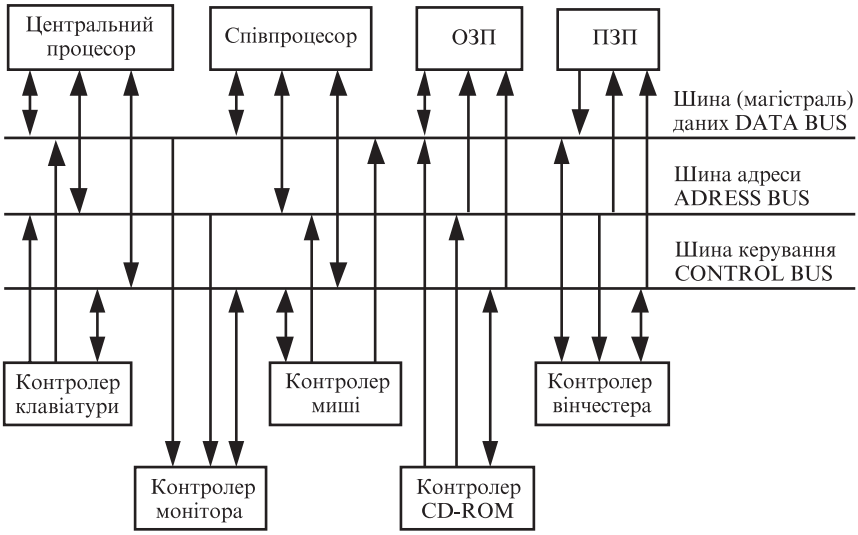


Рис. 2.3

Роль центрального процесора у сучасних комп'ютерах виконує мікропроцесор. Перша частина слова мікро (грецьке micro — маленький) означає, що процесор виготовлений за сучасними технологіями на одному кристалі (чипі) і містить десятки тисяч транзисторів. Досягнення технічного прогресу в галузі виробництва електронних мікросхем дає змогу зменшити розміри елементів до мікронів і часток мікрона, завдяки чому вдалося досягти таких показників, які ще донедавна здавалися фантастикою.

Мікропроцесор складається з таких частин:

- арифметико-логічного пристрою;
- надоперативної пам'яті;
- пристрою формування адреси;
- пристрою керування.

Арифметико-логічний пристрій призначений для виконання арифметичних і логічних операцій. Під час виконання цих операцій

необхідно зберігати проміжні результати. Крім того, результати попередніх операцій здебільшого використовуються як дані для виконання наступних операцій. Для зберігання результатів операцій і проміжних операцій призначена ***надоперативна*** пам'ять. Для ефективної роботи арифметико-логічного блока необхідно, щоб час доступу до надоперативної пам'яті не перевищував тривалості операції, яку виконує мікропроцесор, інакше мікропроцесор простоюватиме, очікуючи даних з пам'яті.

Виконуючи арифметичні і логічні операції, арифметико-логічний блок звертається до пам'яті як за кодом чергової команди, так і за даними, над якими здійснюються операції. Крім того, звернення до пам'яті необхідне для запису результатів операції, воно завжди виконується за певною адресою, яку необхідно сформувати. Для виконання цієї процедури є пристрій формування адреси. Слід зазначити, що пам'ять сучасних комп'ютерів має складну структуру і великий об'єм, тому операція формування адреси за складністю і трудомісткістю зрівнюється, а в деяких задачах перевищує операції обробки інформації, які виконує арифметико-логічний блок.

Пристрій керування центрального процесора призначений для координації дій інших пристроїв під час виконання операцій обробки інформації.

Центральний процесор взаємодіє з іншими пристроями за допомогою системи шин: даних, адреси, керування. Шина або магістраль — сукупність провідників, якими передаються електричні сигнали. Кількість провідників відповідає кількості розрядів двійкових чисел, якими оперує комп'ютер. Ця кількість провідників або розмір шини є однією з найважливіших характеристик процесора і комп'ютера. Сучасні комп'ютери мають 32-розрядну шину даних.

Ефективність системи обробки інформації у цілому значно зростає, якщо центральний процесор звільнити від виконання складних математичних операцій і перекласти виконання цих операцій на спеціалізований процесор, так званий ***співпроцесор***. Співпроцесор під час роботи вихоплює з потоку операцій «свої» і виконує їх паралельно із центральним процесором, чим значно прискорює виконання поставленої задачі у цілому. Таке розподілення функцій між процесором і співпроцесором застосовується у більшості сучасних комп'ютерів.

Пам'ять комп'ютера призначена для зберігання такої інформації:

- ❒ послідовності команд (програм), тобто інформацію про те, які дії має виконувати комп'ютер;
- ❒ вхідні дані, тобто операнди, над якими виконуються дії;

- ❒ проміжні та кінцеві результати операцій;
- ❒ адресна інформація про команди і дані.

Пам'ять складається з ***постійної пам'яті*** (англ. ROM — readily only memory — пам'ять, з якої можна тільки читати), інформації, яка записується під час виготовлення і не може бути зміненою, і ***оперативної пам'яті*** (англ. RAM — random access memory — пам'ять з довільним доступом), інформація в якій змінюється під час виконання програм. Якість пам'яті визначається такими показниками як ***місткість***, тобто кількість байтів інформації, яку можна зберігати, ***швидкодія***, яка характеризується часом доступу, тобто мінімальним часом, впродовж якого можна записати або прочитати інформацію. Крім того, важливою характеристикою пам'яті є ***енерговитрати***, які визначаються потужністю, що споживається, у розрахунку на один біт.

Периферійні пристрої взаємодіють з процесором через ***порти***, тобто спеціальні контактні пристрої, що розміщені на задній стінці системного блока. Швидкість роботи периферійних пристроїв набагато менша за швидкість роботи процесора. Наприклад, за допомогою клавіатури можна вводити у кращому разі до десятка знаків за секунду, в той час як тривалість елементарних операцій, які виконує центральний процесор, становить соті частки мікросекунди, тому безпосередньо керувати роботою периферійних пристроїв за допомогою центрального процесора недоцільно. У сучасних комп'ютерах центральний процесор звільняють від безпосереднього керування периферійними пристроями і покладають ці обов'язки на спеціалізовані процесори, так звані ***контролери*** (від англійського control — керування), які працюють під керуванням центрального процесора. Кожний периферійний пристрій: монітор, клавіатура, принтер тощо — обслуговує свій контролер. У деяких випадках спеціалізовані процесори для керування периферійними пристроями називають ***адаптерами***.

2.2 Пристрої введення інформації

Інформація, яку обробляє комп'ютер, вводиться в нього за допомогою спеціальних технічних пристроїв, які часто називають периферійними пристроями. Комп'ютер у цілому, і центральний процесор зокрема, обробляють інформацію у формі чисел двійкової системи. У пам'яті комп'ютера інформація також зберігається у вигляді двійкових чисел. Людина ж сприймає

інформацію у формі тексту, графічних зображень, звуків тощо. Тому одним з основних призначень пристроїв введення інформації є перетворення інформації з різноманітних форм у двійкову. Розглянемо найбільш поширені пристрої введення інформації в комп'ютерну систему.

Клавіатура

Клавіатура призначена для введення у комп'ютер текстової і цифрової інформації. За допомогою клавіатури вводиться також інформація для керування режимами роботи комп'ютера, переміщенням курсора тощо, тому клавіатура є досить універсальним засобом для введення інформації.

Рис. 2.4

Клавіатура складається з набору клавіш, закріплених у пластмасовому корпусі й нагадує клавіатуру друкарської машинки (рис. 2.4). Центральна група клавіш з нанесеними на них буквами кирилиці та латиниці призначена для введення текстової інформації. Перехід від введення букв одного алфавіту до введення букв іншого алфавіту здійснюється за допомогою керівних клавіш, здебільшого **Ctrl + Shift**. Для переходу від режиму введення малих букв до великих і навпаки служить клавіша **Caps Lock**. Для введення у комп'ютер цифр над буквеними клавішами зверху розміщений рядок клавіш з цифрами. Крім того, для введення великих масивів цифр у правій частині знаходиться ще одна компактна група клавіш з цифрами, розміщеними аналогічно розміщенню на кишенькових калькуляторах. Активізується ця група цифрових клавіш клавішею **Num Lock**. Крім букв і цифр, за допомогою клавіатури можна вводити розділові знаки і спеціальні символи.

Функціональними клавішами **F1...F12** вводяться спеціальні режими роботи комп'ютера. Наприклад, за допомогою клавіші **F1** викликається довідкова інформація для даної програми.

Для керування роботою комп'ютера, задання режимів роботи призначена група керівних клавіш: **Ctrl**, **Shift**, **Enter**, **Esc** тощо.

Миша

Миша — невелика пластмасова коробочка (здебільше сірого кольору) з двома–трьома клавішами, яка легко розміщується у долоні. Разом з електричним проводом для з'єднання з комп'ютером

цей пристрій дійсно нагадує мишу з хвостом (рис. 2.5). Миша призначена для введення в комп'ютер керівної інформації. Якщо мишу переміщувати по плоскій поверхні, то це переміщення перетворюється на електричний сигнал, під дією якого на екрані монітора (дисплея) переміщується світловий вказівник (курсор) у вигляді стрілки. Підводячи курсор до графічного об'єкта або тексту і клацаючи лівою клавішею, виділяють їх. Подвійне клацання лівою клавішею на виділеному графічному об'єкті призводить до запуску програм, зміни режиму роботи. Якщо на виділеному об'єкті клацнути правою клавішею, то це викличе довідкову інформацію про властивості об'єкта.

Рис. 2.5

Трекбол

У комп'ютерах, що випускаються у портативному варіанті: у наколінному (лептоп) або блокнотному (ноутбук) виконанні — роль миші виконує трекбол. Трекбол — невеличка кулька, яка може вільно обертатися, якщо по ній проводити долонею (рис. 2.6). Обертання кульки перетворюється на електричний сигнал, під дією якого на екрані монітора переміщується світловий вказівник. Введення керівної інформації за допомогою трекболу аналогічне введенню інформації за допомогою миші.

Рис. 2.6

Джойстик

Для введення керівної інформації у комп'ютер під час комп'ютерних ігор застосовують джойстик (від англ. joy — радість, задоволення, stick — палиця). Джойстик служить для імітації керування різними рухомими об'єктами, що відтворюються на екрані комп'ютера під час комп'ютерних ігор. Вводити інформацію у комп'ютер можна також і за допомогою гнучких магнітних дисків, які виконують одночасно роль пристроїв пам'яті.

Сканер

Для введення у комп'ютер текстової і графічної інформації, надрукованої на папері, призначені сканери (від англ. to scan — ретельно читати, вивчати). Для введення інформації за допомогою сканера є пристрій зчитування, який рівномірно рухається над зображенням. Графічне зображення або текст, які необхідно ввести

в комп'ютер, освітлюють джерелом світла. Відбиті від зображення промені потрапляють на фоточутливий елемент і перетворюються на послідовність двійкових чисел, які поступають у комп'ютер і обробляються.

а) *б*)

Рис. 2.7

Для перетворення послідовності двійкових чисел у текстову або графічну інформацію призначена спеціальна програма розпізнавання. Пристрій зчитування рівномірно переміщують або вручну в ручних сканерах (рис. 2.7, *а*), або за допомогою спеціального механізму в настільних сканерах (рис. 2.7, *б*). Однією з найважливіших характеристик сканерів є роздільна здатність.

2.3 Пристрої виведення інформації

Оброблену комп'ютером інформацію слід вивести і зобразити у зручній і звичній для людини формі: тексту, графічних зображень тощо. Для цього застосовуються такі пристрої виведення: монітор (дисплей), принтер, плотер.

Дисплей (монітор)

Для виведення оброблюваної комп'ютером інформації та зображення її в необхідній формі призначений дисплей (від англ. display — показувати). Крім зображення результатів, за допомогою дисплея можна слідкувати за ходом виконання програм комп'ютером, змінювати параметри програм і режими роботи комп'ютера. Тому цей пристрій виведення інформації називають ще монітором (від англ. monitor — пристрій для слідкування) (рис. 2.8).

Найпоширенішим є дисплей на основі електронно-променевої трубки, але останнім часом стрімко розвиваються плазмові дисплеї.

Електронно-променева трубка, яка є основою більшості сучасних дисплеїв, діє таким чином: у скляній колбі, де створюється високий вакуум, розміщується пристрій (катод), з якого під дією високої температури випускаються електрони. Далі за допомогою електричних полів електрони прискорюються і формується вузький пучок рухомих електронів, так званий електронний промінь. Електронний промінь можна відхиляти по вертикалі і горизонталі за допомогою відхилювальних систем. Як відомо, будь-який рух можна розкласти на вертикальну і горизонтальну складову. Електронний пучок, потрапивши на екран, вкритий шаром спеціальної речовини (люмінофором), викликає його світіння. Якщо пучок переміщувати, то світитиметься ряд точок — лінія. Таким чином, переміщуючи електронний пучок, можна отримати лінію довільної форми, розклавши цю лінію на вертикальне і горизонтальне переміщення електронного променя. У дисплеях, як і в побутових телевізорах, електронний промінь рухається вздовж горизонтальних ліній, які щільно лежать одна під одною, утворюючи так званий ***растр***. Збільшуючи чи зменшуючи інтенсивність електронного променя, можна змінювати яскравість точок світіння, відтворюючи яскравість точок зображуваного об'єкта. Таким чином, зображення на екрані дисплея — це сукупність світних точок певної інтенсивності. Кожна така світна точка називається пікселем (від англ. pixel — скорочення від picture element — елемент зображення). Кількість пікселів на екрані визначає одну з найважливіших характеристик дисплея — роздільну здатність. Чим більше пікселів, тим детальніше зображений графічний об'єкт на екрані і тим вища якість зображення. Зображення з роздільною здатністю 640×480 пікселів вважають зображеннями з низькою роздільною здатністю, відповідно з низькою якістю, а зображення з роздільною здатністю 1024×768 пікселів — з високою роздільною здатністю та якістю.

Рис. 2.8

Дисплеї можуть давати зображення з градацією кольорів від чорного до білого (монохромні дисплеї) і кольорове зображення (кольорові дисплеї). Щоб отримати кольорове зображення, екран покривають люмінофором, що світиться різними кольорами. Відомо, що будь-який колір можна розкласти на три основні кольори, три складові. Таким чином, збуджуючи випромінювання у кожній точці люмінофора з трьома кольорами світіння, отримаємо світну точку заданого кольору і заданої інтенсивності. Кольорові дисплеї мають набагато більшу інформативність, ніж монохромні, і тому поступово витісняють останні.

Важливою характеристикою дисплея є частота кадрів, тобто кількість зображень, що змінюється за одну секунду. Завдяки здатності сітківки людського ока утримувати зображення впродовж певного часу (близько кількох десятих часток секунди), зображення рухомого об'єкта, що записане послідовністю зображень через невеликі проміжки часу, сприймається людським оком як плавне переміщення об'єкта. Встановлено, що чим більше послідовних зображень відтворено на екрані за одну секунду, тобто чим більша частота кадрів, тим менше втомлюються очі оператора, що працює за екраном комп'ютера. У сучасних дисплеях частота кадрів доведена до 150 Гц порівняно з 50 Гц у перших моделях.

Ще однією важливою характеристикою дисплея є розмір екрана по діагоналі. Переважна більшість дисплеїв має розмір від 14 дюймів до 21 дюйма.

Принтер

Традиційна форма зображення і збереження текстової і графічної інформації на папері продовжує займати важливе місце, незважаючи на бурхливий розвиток електронних форм інформації. Інформацію, зафіксовану на папері, називають ще «твердою копією» документа, вважаючи «м'якою» копією електронний варіант. Паперовий варіант документа зберігає ряд переваг над електронним: звична для всіх форма, нечутливість до комп'ютерних вірусів, немає потреби у спеціальному пристрої для відтворення інформації, гарний естетичний вигляд.

Для створення документа на папері призначені принтери. Технічним прообразом сучасного комп'ютерного принтера є друкарська машинка, яка широко застосовувалася для створення паперових документів задовго до комп'ютерної епохи.

За принципом дії принтери поділяються на матричні, струминні та лазерні (рис. 2.9).

У матричних принтерах графічні зображення і знаки (букви, цифри, розділові та спеціальні) утворюються як сукупність точок,

що наносяться на папір ударом тонкого сталевого стержня (голки) через синтетичну стрічку, просякнуту спеціальною друкарською фарбою. Голки розміщені стовпчиком і разом з механізмом їх подачі змонтовані у друкувальній головці. Кількість голок у головці визначає якість друку і може бути 9 або 24. Найпоширеніші принтери з 9 голками. За допомогою спеціального механізму друкувальна головка рівномірно рухається вздовж рядка, а голки в певний момент ударяють по паперу крізь фарбувальну стрічку.

Рис. 2.9

У струминних принтерах зображення на папері створюється сукупністю точок, які наносяться мікрокрапельками спеціальних чорнил, що видуваються струменем повітря через сопла. Якість зображення у струминних принтерах вища, ніж у матричних. Крім того, струминні принтери можуть друкувати кольорові зображення, використовуючи кольорові чорнила.

Щоб створити зображення на папері за допомогою лазерного принтера, поверхню діелектричного барабана електризують дією лазера і створюють потенціальний рельєф, який повторює рельєф зображення. Часточки спеціального фарбувального порошку притягуються до наелектризованої поверхні барабана, а потім переносяться на папір. Якість зображення на лазерному принтері вища, ніж на струминному.

Якість зображення, нанесеного за допомогою принтера на папір, характеризується кількістю точок на дюйм. Сучасні принтери створюють зображення понад 600 dpi (англ. абревіатура dpi — dot per inch). Ще однією важливою характеристикою принтера є швидкість друкування.

Плотер

Важливими технічними документами були і продовжують залишатися технічні креслення та інша конструкторська документація. Конструктори та інженери, перш ніж створити новий пристрій чи

механізм з усіма його вузлами і деталями, втілюють свої задуми у технічних кресленнях. Виготовлення якісної технічної документації — копітка і напружена робота. Застосування комп'ютерів дає змогу звільнити конструктора від рутинної роботи і сконцентруватися на творчій роботі. Створена за допомогою комп'ютера технічна документація зберігається у пам'яті в електронній формі. Для відтворення її на папері призначені спеціальні технічні пристрої — плотери (від англ. plotter — пристрій для побудови графіків).

Звукові колонки

У деяких випадках, наприклад під час комп'ютерних ігор, необхідно виводити звукову інформацію. Крім того, за допомогою сучасних комп'ютерів можна відтворювати музичні твори. Для виведення звукової інформації існують звукові колонки. Перш ніж занести звукову інформацію у пам'ять комп'ютера, її перетворюють за допомогою спеціальних програм у компактну і зручну для зберігання форму. Для зворотного перетворення інформації призначений спеціальний електронний пристрій (звукова карта), який можна розмістити на материнській платі. Координує дії звукової карти центральний процесор. Відтворення музики на комп'ютері може здійснюватися одночасно з виконанням інших програм.

2.4 Пристрої обміну інформацією

Ефективність роботи комп'ютера зростає багаторазово, якщо його підключити до мережі, і користувач отримає доступ до гігантських інформаційних ресурсів.

Інформація каналами зв'язку передається в одній формі, а в комп'ютері обробляється і зберігається в іншій формі. Для перетворення інформації призначені спеціальні пристрої — модеми (скорочення від МОДуляція і ДЕМодуляція). Комп'ютер через модем підключається до телефонної лінії, й користувач отримує доступ до потрібної йому інформації.

2.5 Зовнішні пристрої зберігання інформації

Зовнішні пристрої зберігання інформації є складовою частиною складної ієрархічної системи пам'яті комп'ютера. Зовнішня пам'ять має набагато більшу місткість, ніж внутрішня пам'ять,

але час доступу до зовнішньої пам'яті набагато більший, ніж до внутрішньої, тому в зовнішній пам'яті розміщують інформацію великих обсягів, до якої звертаються порівняно рідко. Зрозуміло, що користувач не працює з усіма програмами одночасно, а здебільшого з однією, у крайньому випадку з кількома, тому звертається до зовнішньої пам'яті лічені рази протягом робочого дня. Розглянемо детальніше зовнішні пристрої зберігання інформації.

Твердий магнітний диск (вінчестер)

Цей пристрій зберігання інформації складається з кількох керамічних або металевих (алюмінієвих) дисків, закріплених на одній осі (рис. 2.10). На кожний диск наноситься тонкий шар магнітного матеріалу, який служить носієм інформації. Вся площина магнітного матеріалу поділена на концентричні смужки (доріжки). Кожна доріжка поділена на сектори так, щоб у кожному секторі могла розміститися однакова кількість інформації. Диски жорстко закріплені з рівним інтервалом на вертикальному стержні (осі). Стержень приводиться в обертальний рух за допомогою спеціального двигуна. Частота обертання дисків становить кілька тисяч обертів за хвилину.

Рис. 2.10

Для записування і зчитування інформації з секторів призначені головки запису і зчитування. Вони закріплені на спеціальних важелях, за допомогою яких переміщуються у певну позицію, щоб отримати доступ до інформації у потрібному секторі.

Час доступу до інформації на твердому диску становить одиниці мілісекунд. Місткість твердих дисків — десятки гігабайтів.

Гнучкі магнітні диски

Гнучкі магнітні диски призначені для зберігання порівняно невеликих обсягів інформації, а також для перенесення інформації з комп'ютера на комп'ютер. У такому випадку їх можна розглядати як зручний засіб введення і виведення інформації.

Гнучкий магнітний диск виготовляється з пластика, на який наноситься тонкий шар магнітного матеріалу. Щоб захистити диск від пилу і пошкоджень, його розміщують у твердому пластмасовому футлярі. Діаметр гнучкого магнітного диска — 3,5 дюйма.

Раніше застосовувалися диски діаметром 5 дюймів, але вони поступово вийшли з ужитку. Гнучкий магнітний диск з футляром називається ***дискетою*** (рис. 2.11). Щоб записати або прочитати інформацію, дискета вставляється у дисковід і приводиться в обертальний рух за допомогою спеціального двигуна. Швидкість обертання гнучкого диска набагато менша, ніж твердого, і становить сотні обертів за хвилину. Місткість дискети — 1,44 мегабайта. Інформація записується і зчитується за допомогою головок запису і зчитування.

Рис. 2.11

Оптичні диски (CD–ROM)

Інформацію на оптичні або компакт-диски можна перезаписувати, як і на гнучкі магнітні диски, але на відміну від дискет, інформація на компакт-диски записується за допомогою лазера не на концентричні доріжки, а на одну суцільну спіральну доріжку. Читається інформація з компакт-диска також за допомогою лазерного променя.

На лазерних дисках можна записати довідники, енциклопедії, книги, у тому числі підручники, музичні твори, кінофільми.

1. Які технічні пристрої входять до складу комп'ютерної системи обробки інформації?
2. З яких частин складається системний блок?
3. З яких елементів складається материнська плата?
4. Назвіть призначення центрального процесора.
5. Назвіть пристрої для введення інформації в комп'ютер?
6. Що таке дисплей, його основні технічні характеристики?
7. Які групи клавіш є на клавіатурі, їх призначення?
8. Поясніть, як за допомогою миші у комп'ютер вводиться інформація?
9. Для чого призначений плотер?
10. Назвіть основні технічні характеристики принтера?
11. Яка будова твердого і гнучкого магнітних дисків?
12. Як зберігається інформація на оптичних дисках?

3

Основи алгоритмізації та програмування

3.1 Прикладні задачі та їх розв'язок за допомогою комп'ютера

Бурхливий розвиток комп'ютерної техніки та інформаційних технологій викликаний потребою розв'язку задач у різних галузях людської діяльності. З розвитком суспільства ускладнювалося промислове виробництво, з'являлися і розвивалися нові технології, що призводило до ускладнення задач, які потрібно було розв'язувати. Стало зрозуміло, що без адекватних засобів цю проблему не вирішити.

Незважаючи на те, що прикладні задачі у різних предметних областях мають свою специфіку і можуть істотно відрізнятися одна від одної, в них є багато спільних рис.

Знайти розв'язок задачі — це отримати потрібний результат з вихідних даних.

Вихідні дані — це те, що дано в умові задачі.

Потрібні результати — це те, що потрібно отримати в результаті розв'язку.

У розв'язанні задач є загальні етапи.

1. Постановка задачі, де аналізуються вихідні умови, уточнюється, що саме необхідно отримати в результаті розв'язання.

2. Будується модель. Під ***моделлю*** розуміють аналог, образ об'єкта, процесу чи явища, який зберігає їх істотні властивості. Дослідження реальних об'єктів, процесів і явищ за допомогою моделей є основним методом пізнання реального світу.

Моделі можуть бути різних видів і форм:

- ❒ словесний опис природною мовою;
- ❒ за допомогою різних графічних зображень (схем, графіків, діаграм тощо);
- ❒ за допомогою математичних співвідношень, рівнянь, залежностей.

Під час побудови моделі необхідно знаходити компроміс між двома тенденціями.

Перша тенденція полягає в ідеалізації, тобто спрощенні досліджуваного реального об'єкта, процесу чи явища. Неістотні для

даної конкретної задачі властивості і параметри реального об'єкта не враховуються. Це дає змогу глибше дослідити і зрозуміти взаємозв'язок істотних для даної задачі параметрів об'єкта. Чим вищий ступінь спрощення чи схематизації, тим менша складність задачі і тим менше потрібно ресурсів (інформаційних, апаратних, часових тощо) для розв'язання даної задачі.

Друга тенденція полягає у намаганні дослідника врахувати якомога більше параметрів, щоб отримати найбільш ***адекватну модель***, тобто модель, яка найповніше відображує характеристики, особливості та поведінку реальних об'єктів. Зі збільшенням ступеня адекватності зростає складність задачі і потрібно більше ресурсів для її вирішення.

3. Вибирається метод розв'язання задачі.
4. Розробляється алгоритм розв'язання задачі.
5. Розробляється програма за розробленим алгоритмом.
6. Програму тестують і налагоджують.
7. Виконують програму з вихідними даними.

3.2 Алгоритми

Процес розв'язання прикладної задачі — не одномоментний акт, він складається з послідовності певних дій чи операцій. Під ***алгоритмом*** розуміють скінченну послідовність операцій чи дій, яка забезпечує розв'язання задачі певними ресурсами за певний час.

Алгоритмізацією задачі називається процес зведення задачі до послідовності операцій, що виконуються одна за одною.

Алгоритм має задовольняти певні вимоги.

1. Масовість. Застосовуватися не для однієї, а для цілого класу однотипних задач.

2. Скінченність. Складатися зі скінченної кількості операцій чи дій, кожна з яких потребує скінченного відрізка часу.

3. Результативність. Після закінчення виконання алгоритму має бути певний результат.

4. Однозначність. Виконання кожної дії алгоритму і всієї послідовності дій має здійснюватися однозначно. Це означає, що скільки б разів алгоритм не застосовувався до одних і тих самих вихідних даних, результат виконання кожної дії і всієї послідовності у цілому буде один і той самий.

5. Правильність. При застосуванні алгоритму до допустимих вхідних даних має бути отриманий потрібний результат. Доведення

правильності алгоритму — один з найскладніших етапів його створення. Найпоширеніша процедура перевірки правильності алгоритму — обґрунтування правомірності та перевірка правильності кожного з кроків при наборі тестів, підібраних так, щоб охопити всі допустимі вхідні і вихідні дані.

6. Ефективність. Забезпечувати розв'язання задачі за мінімальний час з мінімальними затратами апаратних і програмних ресурсів.

Для оцінки алгоритмів існує багато критеріїв. Найчастіше аналіз алгоритму (або, як кажуть, аналіз складності алгоритму) полягає в оцінці затрат часу на розв'язок задачі у розрахунку на одиницю вхідних даних. Фактично, ця оцінка зводиться до оцінки кількості базових елементарних операцій, на які можна розкласти даний алгоритм, оскільки кожна така операція виконується за конкретний, відомий відрізок часу.

Складність алгоритму оцінюється також кількістю апаратурних ресурсів, зокрема обсягом пам'яті, задіяної для виконання даного алгоритму.

Алгоритм може задаватися різними способами:

- ❒ словесно, словами і реченнями природної мови — української, англійської тощо;
- ❒ графічно, за допомогою спеціальних схем із застосуванням умовних графічних позначень;
- ❒ символами спеціальної алгоритмічної мови.

Запис алгоритму словами і реченнями природної мови найдоступніший для ширшого кола користувачів і не потребує спеціальної підготовки. Водночас така форма найменш точна та визначена, оскільки допускає різні тлумачення одного і того самого поняття внаслідок синонімічного багатства природних мов.

Застосування спеціальних графічних схем для запису алгоритму дає змогу наочно бачити взаємозв'язок вхідних і вихідних величин на різних етапах виконання алгоритму. Крім того, алгоритм у такій формі доволі просто змінювати і коригувати.

Алгоритм, записаний спеціальною алгоритмічною мовою, може бути використаний на комп'ютері для розв'язку прикладних задач.

Графічне зображення алгоритмів

Алгоритм зручно зображати графічно, у вигляді спеціальних схем. ***Схемою*** називається умовне наочне графічне зображення алгоритму за допомогою умовних графічних позначень. Умовні графічні позначення на схемах алгоритмів унормовані державними стандартами.

Розглянемо основні умовні графічні позначення на схемах.

1. Зв'язок між окремими операціями чи діями алгоритму позначають лінією (рис. 3.1, *а*).

2. Операцію, або послідовність операцій позначають прямокутником. Всередині прямокутника записується зміст операції чи послідовності операцій (рис. 3.1, *б*).

3. Перевірка умови позначається ромбом. Всередині ромба записується сама умова. Залежно від того, виконується умова чи ні, подальші операції вказуються стрілками, що виходять з вершин ромба, позначених «так» і «ні» (рис. 3.1, *в*).

4. Початок і кінець алгоритму зображуються заокругленими прямокутниками із написом усередині «початок» і «кінець» (рис. 3.1, *г*).

5. Програма чи алгоритм, що входять до даного алгоритму як складова частина, зображуються прямокутником з подвійними вертикальними сторонами (рис. 3.1, *д*).

6. Коментарі, що пояснюють операції обробки інформації, виділені квадратною дужкою зі штриховою лінією (рис. 3.1, *е*).

7. З'єднання розірваних зв'язків зображується колом з позначенням номера зв'язку (рис. 3.1, *є*).

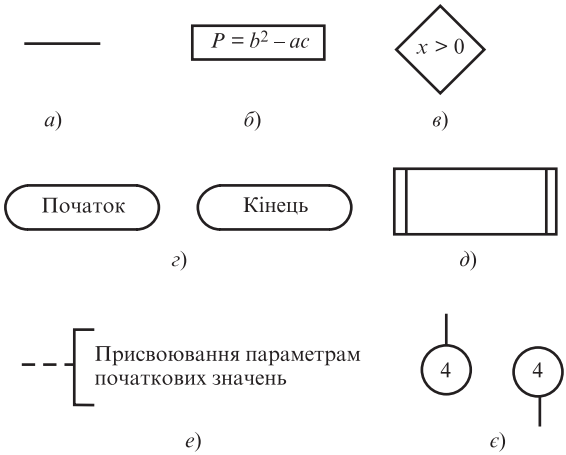
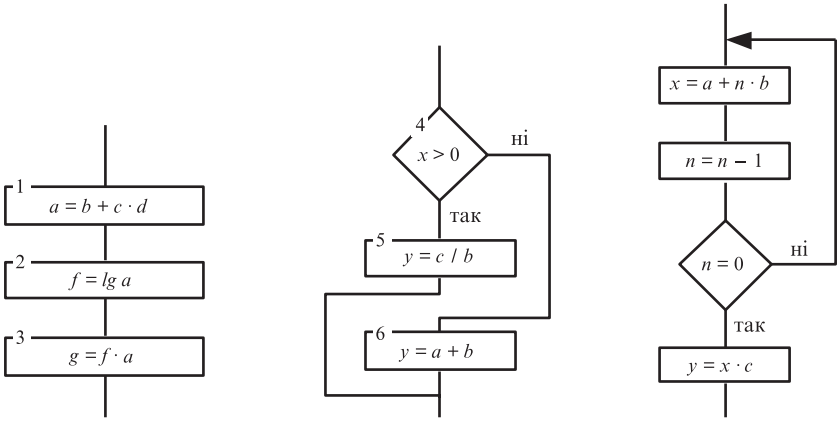


Рис. 3.1

Алгоритми розв'язку складних задач мають схему зі складною структурою. Водночас, у будь-якій складній структурі можна виділити базові структури і розглядати складні структури як сукупність базових структур. Це значно спрощує і прискорює створення, коригування, перевірку алгоритму на правильність.

Розглянемо базові структури алгоритму:

- послідовна структура. У цій структурі операції обробки інформації виконуються послідовно, одна за одною (рис. 3.2, *а*);
- розгалужена структура. Визначальною операцією цієї структури є операція перевірки умови, залежно від чого процес обробки інформації розгалужується на два альтернативні напрями (рис. 3.2, *б*);
- циклічна структура. Замкнена послідовність операцій, коли після виконання останньої операції переходять до виконання першої (рис. 3.2, *в*).



а) *б*) *в*)

Рис. 3.2

3.3 Програмування

Програмуванням називається процес створення програм відповідно до певного алгоритму. Розрізняють кілька видів і стилів програмування.

Структурне програмування. Поняття структурного програмування вперше запропоновано голландським ученим Е. Дейкстроєм. Основою структурного програмування є те, що основну увагу під час створення програм слід звертати на структуру програм, тобто виділення основних елементів, їх типізацію і взаємодію.

У структурному підході виділяються такі типові структури, як послідовність операторів, оператор умовного переходу, оператор

циклу, звернення до підпрограм. З точки зору структурного програмування є вкрай небажаним застосування операторів безумовного переходу, які переривають природний хід обчислень, роблять алгоритм і програму заплутаною і малозрозумілою. Значно ускладнюється у такому разі налагодження програм.

Структурне програмування припускає необхідним порушення природної послідовності виконання операцій тільки у таких випадках: для вибору одного оператора з кількох можливих залежно від виконання певних умов; для організації повторення операцій (циклу); для звернення до функції, процедури чи підпрограми. Вживання операторів переходу, з точки зору прихильників структурного програмування, невиправдане і призводить, з одного боку, до ускладнення роботи програм, а з іншого, до втрати наочності, до плутанини.

Текст програми, що займає кілька сторінок і має складну логіку, не дає змоги людині охопити її в цілому, якщо не передбачити заздалегідь строгу дисципліну її написання. Програма, яку складно і незручно читати, як правило, містить численні помилки, які важко виявити і виправити.

Структурне програмування передбачає також певні правила оформлення тексту програм: текст підпрограми або циклу мусить мати відступ від тексту основної програми; вкладена підпрограма нижчого рівня, тобто підпрограма, яку викликають, мусить мати відступ від підпрограми вищого рівня, тобто підпрограми, яка викликає; на початку і в кінці програми, підпрограми чи циклу має бути однаковий відступ.

Модульне програмування. Оскільки виготовлення програм — це трудомісткий процес, то доцільно виготовлену один раз програму використовувати для розв'язання багатьох задач. Цього можна досягнути, якщо, по-перше, зробити програму однаковою для багатьох типів вхідних даних; по-друге, збирати програму з готових частин — ***модулів*** чи блоків. Самі ж модулі мають бути достатньо універсальними і типовими, аби вони могли використовуватися у різних програмах.

Такий спосіб конструювання програм зі стандартних модулів отримав назву ***модульного програмування***. Дієвість і ефективність такого підходу виявляються не тільки у програмуванні, а й у багатьох галузях виробництва, зокрема у будівництві, коли будівлі з різною архітектурою, кількістю поверхів тощо виготовляються зі стандартних блоків.

При модульному програмуванні складна задача розкладається на ряд простих задач, які розкладаються на ще простіші.

Слід зазначити, що у програмуванні використання стандартних модулів дає особливо значний ефект тому, що використання будь-якого модуля не означає його фізичного споживання (на відміну, наприклад, від будівництва) і неможливість його використання для інших задач.

Програмування за принципом «зверху вниз» і «знизу вгору». Якщо поставлена задача складна і вимагає створення складної програми, людині не вдається зразу уявити собі детальний вигляд цієї програми та її ефективну реалізацію, оскільки у пам'яті прийшлося б втримувати непомірно великий обсяг інформації. Програмування за принципом «зверху вниз» ґрунтується на тому, що задача розбивається на кілька більш простих задач і подальшу деталізацію кожної частини здійснюють окремо. Частини задачі виділяються таким чином, щоб їх можна було програмувати незалежно одна від одної. Для цього необхідно встановити чітку систему зв'язків між окремими частинами програми.

Програмування за принципом «зверху вниз» буде ускладнено, якщо не зрозуміло, як і на які частини ділити задачу. Разом з тим, часто буває зразу видно, як розв'язувати окремі частини цієї задачі. У такому разі можна намагатися оформити ці фрагменти задачі і, ґрунтуючись на них, конструювати потрібну програму. Такий метод програмування називається програмуванням за принципом «знизу вгору».

Процедурне програмування. Програма процедурною мовою програмування складається з послідовності ***операторів***, які називають також ***командами***, ***інструкціями***, виконання яких призводить до виконання поставленої задачі. У процедурному програмуванні пам'ять трактується як сховище вхідних даних, що перетворюються операторами програми в результати. Таким чином з точки зору програміста існує тільки програма і пам'ять, вміст якої перетворює програма.

Функціональне програмування. Функціональне програмування розглядає програму як послідовний виклик функцій, що здійснюють обробку даних. Функції можуть складатися з інших функцій і входити до складу функцій. Мова програмування з точки зору функціонального програмування має такі елементи: класи постійних величин (констант), якими можуть оперувати функції; комплект базових функцій, які програміст може використовувати без попереднього об'явлення і опису; правила побудови нових функцій на основі базових; правила формування виразів на основі виклику функцій.

Логічне програмування. Логічне програмування — напрям у програмуванні, що ґрунтується на поняттях і методах математичної

логіки. Основним поняттям у логічному програмуванні є ***відношення***. Програма у логічному програмуванні є сукупністю тверджень про об'єкти, функції, відношення. Виконанню програми відповідає поняття ***запиту***. Методи логічного програмування реалізовано мовою Пролог (програмування на основі логіки — Programming in Logic).

Об'єктно-орієнтоване програмування. В основі об'єктно-орієнтованого програмування лежить поняття об'єкта, а суть цього виду програмування виражається формулою:

Об'єкт = дані + процедури.

Кожний об'єкт складається з структур даних і йому доступні процедури їх обробки, що називаються ***методами***. Об'єднання даних і процедур в одному об'єкті називається ***інкапсуляцією***. Для опису об'єктів служать ***класи***. Клас визначає сукупність об'єктів, об'єднаних спільними властивостями і методами. Будь-який об'єкт можна визначити як ***екземпляр*** класу. Програмування полягає у виборі наявних і створенні нових об'єктів, організації взаємодії між ними. Під час створення нових об'єктів можуть додаватися нові властивості або успадковуватися вже створені. В процесі роботи над об'єктами допускається ***поліморфізм*** — можливість використання однакових методів для обробки даних різних типів. До об'єктно-орієнтованих мов програмування належать **C++** і **Java.**

Останнім часом широко застосовується ***візуальне програмування***, в якому для складання програм, особливо об'єктно-орієнтованих, застосовуються графічні об'єкти. До числа об'єктно-орієнтованих систем візуального програмування належать: **Visual Basic, Delphi, C++ Builder, Visual C++.**

3.4 Мови програмування

Алгоритм може бути записаний спеціальною мовою — алгоритмічною. ***Алгоритмічна мова*** — це система позначень і правил для запису алгоритму розв'язку задачі. Слід зазначити, що за допомогою комп'ютера можна виконати тільки той алгоритм, який записаний алгоритмічною мовою.

Як і природна мова, алгоритмічна має ***алфавіт***, тобто систему символів, за допомогою яких записуються команди. Алфавіт алгоритмічної мови складається з букв латиниці, грецького алфавіту, а в деяких випадках кирилиці, цифр, знаків математичних і логічних операцій, спеціальних знаків.

Із символів алфавіту складаються ***ключові слова*** алгоритмічної мови, за допомогою яких записується послідовність дій (***алгоритм***), які має виконати комп'ютер. Алгоритм, записаний мовою програмування, називається ***програмою***.

Інформація (дані), команди обробляються і зберігаються в пам'яті комп'ютера у вигляді двійкових чисел. Щоб написати програму, програміст повинен пам'ятати комбінації нулів і одиниць двійкового коду кожної команди, а також двійкові коди адрес даних. Ефективність його роботи істотно збільшиться, якщо писати програми природною мовою, а переклад на мову машинних команд доручити комп'ютеру. Мови, спеціально призначені для написання програм, називаються ***мовами програмування***.

Розроблено і застосовується багато мов програмування, орієнтованих на певні класи прикладних задач, ці мови можна поділити на дві групи: низького рівня і високого рівня.

До мов низького рівня належать мови Асемблера (від англ. to assemble — збирати, компонувати). У мові Асемблера двійкові коди команд замінені іменами, які запам'ятати набагато легше. Замість послідовності двійкових кодів команд записуються їх символічні імена, а замість двійкових адрес даних — символьні імена цих даних. Інколи мову Асемблера називають мнемокодом або автокодом.

3.5 Мова програмування Паскаль

Мова програмування **Паскаль** є мовою високого рівня, вона створена Нікласом Віртом і призначалася спочатку як навчальна мова для програмування, а потім нею почали писати програми й для широкого вжитку. У мові Паскаль реалізовано ідею структурного програмування та концепцію структур даних.

Алфавіт. Алфавіт будь-якої мови, в тому числі й Паскаль,— це сукупність символів, за допомогою яких кодується інформація. Алфавіт мови Паскаль містить букви, десяткові (арабські) цифри, розділові знаки, знаки математичних операцій, спеціальні символи, а також службові слова, які сприймаються як один символ. У деяких варіантах цієї мови алфавіт може бути розширений за рахунок букв кирилиці.

Імена (ідентифікатори). Для позначення констант, змінних, типів, процедур, функцій, файлів, програм та інших інформаційних об'єктів використовують імена (ідентифікатори). Іменем може

бути будь-яка послідовність букв, цифр, знаків, підкреслення _ , яка починається з букви. Пропуски в іменах не припустимі. На кількість символів у імені не має обмежень. Для кращого розуміння програми доцільно вибирати значущі імена, а не довільний набір символів. Знак підкреслювання зручно використовувати у тому разі, коли ім'я складається з кількох слів, наприклад, **розрахунок_струмів, вхідні_дані** тощо.

Числа. Числа у Паскалі записані в десятковій системі числення у двох формах: у звичайній, з фіксованою десятковою точкою: **1.2, 53.436, 0.00986**, і в експоненціальній, з плаваючою точкою: **5.2E-2, 16.3E+3, 40E-2.** Числа в експоненціальній формі складаються з ***мантиси*** і ***порядку***, розділених символом E, який слід читати як «помножити на десять у степені...».

Константи–рядки. ***Рядок*** — це довільна послідовність символів, яка сприймається як окремий інформаційний об'єкт. Ознакою рядка є те, що він напочатку і вкінці обмежений апострофами, наприклад, **'Розв''язання рівняння'**, **'Струм ='**. Якщо апостроф потрібно вжити всередині рядка, то слід набрати подвійний апостроф.

Структура програми

Правила побудови програми мовою Паскаль цілком визначені. Вона складається із ***заголовка*** та ***блока*** і закінчується крапкою.

Заголовок програми починається зі службового слова **PROGRAM**, за яким іде ім'я програми з круглими дужками в кінці, де записуються розділені комами параметри програми. Закінчується заголовок крапкою з комою, наприклад:

```
PROGRAM current (input, output)
{Розрахунок струму у колі за законом Ома}
VAR U,R,I;
Begin
read (U,R);
I:= U/R;
Write ('Струм дорівнює',I)
end.
```

У заголовку програми, за службовим словом **PROGRAM** вказується ім'я програми. Наведена вище програма має ім'я current (струм). У круглих дужках записано список параметрів, розділених комами. За допомогою параметрів програма взаємодіє з апаратними і програмними засобами. В ній список параметрів складається з імен двох файлів, які є стандартними у Паскалі: input (введення) і output (виведення). Вхідні дані читаються з файла input, а результати роботи програми записуються у файл output. Якщо в програмі вхідні дані вводити не потрібно, то параметр

input можна не вказувати. Параметр output вказується завжди. У комп'ютер вхідні дані вводяться з пристроїв введення (клавіатури, пристрою читання магнітних дискет тощо), а результати виводяться на пристрої виведення (екран дисплея, принтер тощо).

Блок програми може мати до шести розділів, які записуються у певному порядку:

1. Розділ опису міток.
2. Розділ означень констант.
3. Розділ означень типів.
4. Розділ опису змінних.
5. Розділ опису процедур і функцій.
6. Розділ операторів (тіло блока, тіло програми).

Кожний опис чи означення закінчується розділовим знаком — крапкою з комою. Розділ операторів у кожній програмі є обов'язковим, решта — необов'язкова. Розділ операторів обмежений службовими словами **BEGIN** і **END**. Самі службові слова не є операторами, вони відіграють роль операторних дужок.

Довільна послідовність операторів, обмежена службовими словами **BEGIN** і **END**, називається ***складеним оператором,*** його можна розміщувати у довільному місці тіла програми (блока).

Між будь-якими операторами програми має стояти крапка з комою, яка відділяє операторів один від одного. Між останнім оператором у тілі програми і службовим словом **END** крапка з комою не ставиться, тому що слово **END** не є оператором.

Між двома довільними сусідніми символами у програмі допускається довільна кількість ***пропусків***, але пропуски неприпустимі всередині імен (ідентифікаторів), службових слів, чисел і складених символів. Між будь-якими двома рядками програми є, щонайменше, один пропуск, тому імена, службові слова, числа і складені символи не можна розривати, переходячи від одного рядка до іншого.

Програми Паскаль використовують ***іменовані константи***, яким присвоюється ім'я у розділі означення констант. У подальшому замість констант завжди використовуються імена.

Щоб краще зрозуміти програму, призначення її та елементів, у мові Паскаль використовують ***коментарі***, що допускають довільну кількість символів і обмежуються фігурними дужками. Коментарі, як пропуски і кінці рядків, не впливають на роботу програми і можуть стояти у довільному місці програми.

Типи даних

За допомогою мови Паскаль можна програмувати задачі з різних предметних областей, тому дані можуть мати різний тип.

У Паскалі використовуються чотири стандартні скалярні типи: ***цілий*** **(integer)**, ***дійсний*** **(real)**, ***булевий*** **(boolean)**, ***символьний*** **(char)**. Інші скалярні типи можуть вводитися програмістом.

Тип кожної змінної, яка використовується програмою, має бути явно заданий за допомогою описів у розділі опису змінних. Розділ опису змінних починається зі службового слова **VAR** і має вигляд:

VAR список ідентифікаторів змінних: тип;
список ідентифікаторів змінних: тип;
.
.
.
список ідентифікаторів змінних: тип;

Ідентифікатори змінних відокремлюються один від одного комою, наприклад:

VAR n, m, k: **integer** ;
R1, i1, DR: **real**;
g, f, z: **char**;

У наведеному прикладі змінні n, m, k мають тип **integer**, змінні R1, i1, DR — тип **real**, змінні g, f, z — тип **char**.

На відміну від змінних, вказувати тип константи не потрібно, комп'ютер сам відрізняє її за значенням.

Цілий тип (integer). Змінні цілого типу можуть набувати тільки цілих значень. Цілі числа можуть бути додатними і від'ємними. У мові Паскаль можна використовувати тільки скінченну множину цілих чисел від **−maxint** до **+maxint**. Константа **maxint** залежить від конкретного комп'ютера.

До операндів цілого типу можна застосовувати такі операції: додавання **(+)**, віднімання **(−)**, множення **(*)**, цілочислового ділення **(DIV)**, операція визначення остачі від цілочислового ділення **(MOD)**.

Дійсний тип (real). Змінні цього типу можуть набувати як цілих, так і дробових значень. Комп'ютер може здійснювати операції тільки зі скінченною множиною дійсних чисел. До дійсних чисел можна застосувати такі операції: додавання **(+)**, віднімання **(−)**, множення **(*)**, ділення **(/)**.

Булевий тип (boolean). Булеві (логічні змінні можуть набувати тільки одного з двох значень: істинно (true), хибно (false). Ці слова є булевими константами. Булеві змінні, як і змінні інших типів, мають бути описані у розділі опису змінних. До булевих операндів можна застосувати такі логічні операції: **AND** (логічне **І**, кон'юнкція), **OR** (логічне **АБО**, диз'юнкція), **NOT** (логічне **НЕ**, заперечення).

Крім булевих операцій, результат типу **boolean** видають операції відношення: **< ,<=, =, >, =>**.

Символьний тип (char). Змінна символьного (літерного) типу набуває значення лише одного символу. У мові Паскаль кожний допустимий символ має свій порядковий номер. Символьні константи — це символи, обмежені з обох сторін апострофами. Константи і змінні символьного типу можуть бути операндами в операціях порівняння.

Для визначення порядкового номера символу у впорядкованій множині символів застосовується функція **ord** (символ). Обернену операцію, тобто визначення символу за його порядковим номером виконує функція **chr** (порядковий номер).

Вирази

Вираз — сукупність констант, змінних, функцій, з'єднаних символами операцій (дій). Вирази задають дії й послідовність обчислень. Під час обчислення виразів, операції виконуються в строго визначеній послідовності з урахуванням загальноприйнятих правил ієрархії операцій і дужок. Значення функцій обчислюються в першу чергу. Якщо вираз не має дужок, то операції виконуються у такій послідовності:

1. **NOT;**
2. *** / DIV, MOD, AND;**
3. **+ − OR;**
4. **= < > <= =>.**

Операції одного рівня виконуються у послідовності зліва направо. Якщо потрібно змінити вказану послідовність операцій, застосовують круглі дужки.

Оператор присвоювання

Оператор присвоювання має вигляд:

Ім'я змінної := вираз.

Змінна і вираз мають бути одного типу.

Оператори введення

Для введення даних використовують оператори:

read (список змінних)

або

readln (список змінних).

Оператори **read** і **readln** не є операторами у звичайному розумінні, а зверненням до стандартних процедур введення.

Якщо пристроєм введення є клавіатура, то, виконуючи оператори **read** чи **readln**, комп'ютер переходить у режим очікування введення інформації з клавіатури. Числа відокремлюються одне від одного принаймні одним пропуском. Оператор **read** відрізняється від оператора **readln** тим, що вводить дані з одного рядка.

Оператори виведення

Виведення інформації здійснюється операторами:

write (список елементів виведення)

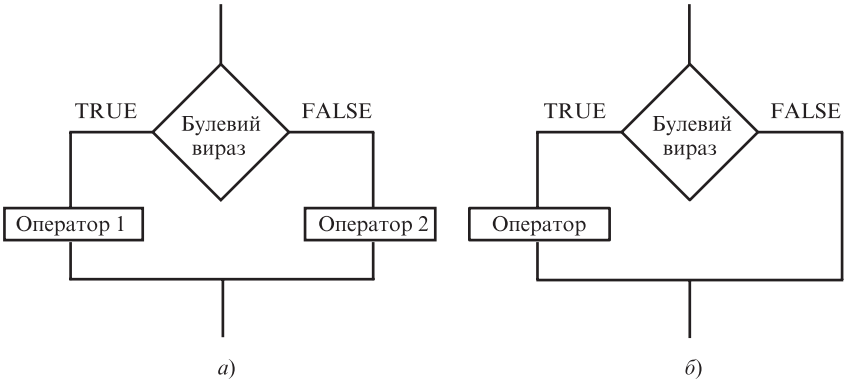
або

writeln (список елементів виведення).

Елементами списку виведення є вирази, змінні або константи всіх чотирьох стандартних типів. Значення арифметичних виразів виводяться у десятковій системі числення, причому дійсні значення виводяться у формі з плаваючою точкою. Оператор **write** відрізняється від оператора **writeln** тим, що виводить дані в одному рядку.

Умовний оператор. У мові Паскаль він має вигляд (рис. 3.3, *а*):

IF *булевий вираз* **THEN** *оператор 1* **ELSE** *оператор 2*.



а) *б*)

Рис. 3.3

Оператори 1 чи 2 можуть бути простими і складеними. Використовується також неповна форма умовного оператора (рис. 3.3, *б*):

IF *булевий вираз* **THEN** *оператор*.

Якщо після службових слів **THEN** і **ELSE** потрібно записати більше одного оператора, тобто складений оператор, то послідовність операторів потрібно взяти в операторні дужки **BEGIN**, **END**.

Умовний оператор діє таким чином: якщо значення булевого виразу **TRUE**, то виконується оператор 1, або складений оператор 1, а якщо значення булевого виразу **FALSE**, то виконується оператор 2. Таким чином, умовний оператор реалізує базову алгоритмічну структуру розгалуження.

Оператор вибору. В умовному операторі перевіряється тільки одна умова, виражена булевим виразом. Якщо ж умов кілька, то доводиться застосовувати кілька вкладених один в один умовних операторів. Це призводить до того, що програми стають менш наочними і складнішими для розуміння, тому в мові Паскаль

передбачено спеціальний оператор — оператор вибору, застосування якого допомагає уникнути складностей.

Оператор вибору записується у вигляді:

CASE *вираз* **OF**
список констант: оператор;
список констант: оператор;
.
.
.
список констант: оператор

Вираз, що стоїть між службовими словами **CASE** і **OF**, називається селектором. Константи у списку відокремлюються одна від одної комами. Тип констант вибору має збігатися з типом селектора.

Виконується команда вибору так. Обчислюється значення виразу, а тоді виконується той оператор, константа вибору якого збігається зі значенням виразу. Решта операторів не виконується.

Якщо в операторі вибору потрібно помістити не один оператор, а кілька, то їх потрібно об'єднати в один складений оператор, тобто взяти в операторні дужки **BEGIN, END.**

Оператори повторення (циклу). У мові Паскаль є три форми оператора повторення, за допомогою яких реалізуються базові циклічні алгоритмічні структури.

Оператор циклу **WHILE (ПОКИ)** записують у вигляді:

WHILE *булевий вираз* ***DO*** *оператор.*

Оператор, що стоїть після **DO** утворює ***тіло*** циклу. Оператор працює так. Обчислюється булевий вираз. Якщо значення булевого виразу **TRUE**, то виконується оператор, що стоїть після **DO**, а якщо значення булевого виразу **FALSE**, то програма переходить до виконання оператора, що стоїть після оператора циклу. Щоб виконати кілька операторів, що стоять після **DO**, тобто виконати складений оператор, їх слід об'єднати, розміщуючи між службовими словами **BEGIN** і **END**. Оператор циклу **ПОКИ** реалізує базову алгоритмічну циклічну структуру (рис. 3.4).

Оператор циклу **ДО** записують у вигляді:

REPEAT
оператори
UNTIL булевий вираз

Службові слова **REPEAT** (повторити) і **UNTIL** (до) відіграють роль операторних дужок і між ними можна помістити послідовність операторів.

Оператор циклу **ДО** виконується таким чином. Обчислюються оператори, що стоять після службового слова **REPEAT**. Потім обчислюється булевий вираз. Якщо значення булевого виразу

FALSE, то знову виконуються оператори після **REPEAT**, а якщо значення булевого виразу **TRUE**, то виконується наступний оператор після оператора циклу (рис. 3.5).

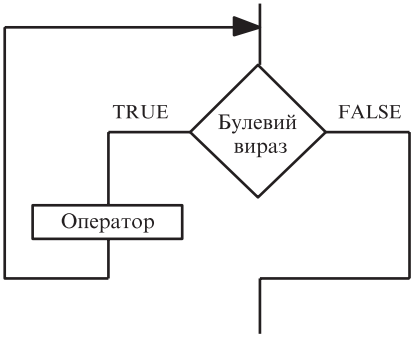


Рис. 3.4

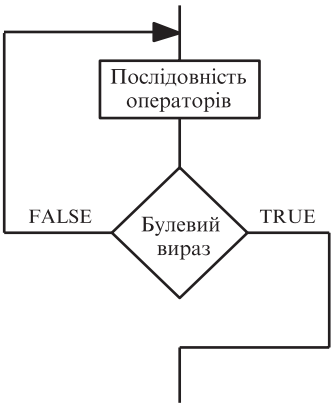


Рис. 3.5

Між операторами циклу існують такі основні відмінності:

- у циклі **REPEAT** перевірка умови виходу здійснюється після виконання операторів, а в циклі **WHILE** — до їх виконання, тому в циклі **REPEAT** у будь-якому випадку оператори виконуватимуться принаймні один раз;
- вихід з циклу в операторі **WHILE** здійснюється, якщо значення виразу — **FALSE**, а в операторі **REPEAT**, якщо значення **TRUE**;
- у циклі **WHILE** послідовність операторів записується в операторних дужках **BEGIN** і **END**, а у циклі **REPEAT** — без дужок.

Оператор циклу з параметром (цикл **FOR** (ДЛЯ)) може мати дві форми:

FOR *ім'я змінної* := *вираз 1* **TO** *вираз 2* **DO** оператор

і

FOR *ім'я змінної* := *вираз 1* **DOWNTO** *вираз 2* **DO** *оператор*

Змінна, ім'я якої стоїть після службового слова **FOR**, називається ***керівною змінною*** або ***параметром циклу***. Щоб виконати у циклі не один, а послідовність операторів, то їх необхідно об'єднати у складений оператор так, як і у циклі **WHILE**.

Виконання оператора циклу **FOR** (ДЛЯ) починається з обчислення виразів. Цикл забезпечує виконання оператора, який стоїть після слова **DO** (виконати), для кожного значення параметра, починаючи від початкового, що дорівнює виразу 1, до кінцевого

значення, яке дорівнює значенню виразу 2, включно. Значення параметра не можуть бути зміненими всередині циклу (в операторі). Крім того, вони мають бути одного і того самого скалярного типу. Якщо застосовується форма оператора з **TO**, то параметр циклу змінюється у прямій послідовності (збільшуватиметься у разі цілих чисел), а якщо з **DOWNTO**, то у зворотній послідовності (зменшуватиметься у разі цілих чисел).

Оператор переходу. Оператори у програмі виконуються, як правило, у послідовності їх запису. Змінити цю послідовність можна за допомогою оператора переходу, який має вигляд:

GO TO *мітка*.

Мітка — ціле число, без знака, що має не більше чотирьох розрядів і призначене для того, щоб помітити певний оператор. Мітки відокремлюються від операторів двокрапкою. Довільна мітка, що використовується у програмі, має бути описаною у розділі опису міток. Цей розділ починається службовим словом **LABEL** (мітка) і розміщується перед розділами визначення констант і опису змінних. Не можна увійти за допомогою міток у середину складеного оператора або в середину процедури.

Процедури і функції

Для виконання складних розрахунків і обчислень у Паскалі є потужний засіб — процедури і функції. На практиці часто трапляється ситуація, коли одну і ту саму групу дій (операцій) потрібно виконати у різних місцях програми. Доцільно оформити цю групу операцій у вигляді окремого модуля, до якого можна звертатися кілька разів.

Підпрограмою називається іменована логічно закінчена група операторів алгоритмічної мови, яку можна викликати для виконання з різних місць програми за іменем будь-яку кількість разів. У мові Паскаль для реалізації підпрограм використовуються процедури і функції.

Процедура — це відносно незалежна частина програми, що має ім'я, призначена для виконання певних дій. Вона складається із ***заголовка*** і ***тіла***. За структурою її можна розглядати як самостійну програму в мініатюрі. Після одноразового опису процедуру можна викликати за іменем з різних частин основної програми. В основній програмі описи процедур і функцій розміщуються після опису змінних. Використання імені процедури у програмі називається ***оператором процедури*** або ***викликом процедури***. Коли процедура виконає свою задачу, виконання основної програми продовжиться з оператора, який стоїть першим після оператора виклику даної процедури. Ім'я процедури не може входити у вираз як операнд.

Функція аналогічна процедурі, але має дві відмінності: функція після виклику повертає у результаті своєї роботи скалярне значення; ім'я функції може входити у вираз як операнд. Функція, що зустрічається у виразі, називається ***вказівником функції*** або ***викликом функції***.

Всі процедури і функції мови Паскаль поділяються на дві групи: вбудовані і визначені користувачем.

Вбудовані (стандартні) процедури і функції є частиною мови Паскаль. До них можна звертатися за іменем без попереднього їх опису в розділі опису. Мова Паскаль має велику бібліотеку вбудованих процедур і функцій, які можна використовувати для розв'язку різноманітних задач.

Процедури і функції користувача є окремими модулями і створюються самим користувачем відповідно до синтаксису мови. Попередній (перед використанням) опис процедур і функцій користувача обов'язковий.

Процедура користувача — це група операторів, що має ім'я і виконує певну частину загальної задачі. Звертатися до процедури або викликати процедуру за іменем можна з довільної позиції розділу операторів.

Опис процедури складається із заголовка процедури і тіла процедури. Заголовок, у свою чергу, складається із зарезервованого слова **Procedura**, ідентифікатора (імені) процедури і необов'язкового списку формальних параметрів із вказівкою їх типу, розміщеного у круглих дужках:

Procedure <ім'я > {(формальні параметри)};.

Наприклад:

Procedure Potential;

Procedure Current (U_1,U_2,R_1,R_2);

У процедури Potential відсутній список формальних параметрів, а у процедури Current чотири формальні параметри U_1, U_2, R_1, R_2.

Ім'я процедури або ідентифікатор процедури має бути унікальним у межах однієї програми.

Тіло процедури — локальний модуль, за структурою аналогічний програмі:

Procedure <ім'я > {(формальні параметри)};
<розділ описів>
begin
<розділ операторів>
end;

Для звернення до процедури з довільної позиції в розділі операторів програми використовується оператор процедури. Він скла-

дається з імені (ідентифікатора) процедури і списку фактичних параметрів, відокремлених один від одного комами, розміщеного у круглих дужках. Список параметрів може бути відсутнім. Оператор процедури має такий формат:

<ім'я процедури> {(формальні параметри)};

Фігурні дужки в запису оператора процедури означають, що цей елемент оператора необов'язковий, факультативний.

Запишемо кілька прикладів операторів процедури (викликів процедури):

Potential;

Current (220.00,380.00,50E-01,78E-01);

Параметри забезпечують механізм заміни, який дає змогу виконувати процедуру з різними початковими даними. Між фактичними параметрами в операторі виклику процедури і формальними параметрами у заголовку опису процедури існує взаємно-однозначна відповідність у результаті їх однакового розміщення зліва направо в однаковій послідовності. Кількість і тип формальних параметрів дорівнюють кількості і типу фактичних параметрів.

Якщо процедура повертає у програму певні значення змінних, то ці змінні мають бути описані як параметри-змінні з використанням слова **VAR**.

Функції, визначені користувачем, складаються із заголовка і тіла функції. Заголовок містить зарезервоване слово **Function**, ідентифікатор функції, поміщений у круглі дужки список формальних параметрів і тип значення, яке повертає функція. Список формальних параметрів необов'язковий і може бути відсутнім.

Формат функції, визначеної користувачем, такий:

Function <ім'я функції> {(формальні параметри)}; <тип результату> ;

Наведемо кілька прикладів визначення функції:

Function Power1(I,R: real); real;

Function Resist1(I,U: real); real;

Ім'я функції має бути унікальним у межах модуля. Результат, що повертає функція, може бути будь-якого скалярного типу.

Тіло функції — локальний модуль, за структурою аналогічний структурі програми:

Function <ім'я > {(формальні параметри)}; <тип результату>;
<розділ описів>
begin
<розділ операторів>
end;

У розділі операторів має знаходитися хоча б один оператор, що присвоює імені функції певне значення. Якщо таких присвоювань кілька, то результатом виконання функції буде значення останнього оператора присвоювання.

Звертання до функції здійснюється за її іменем з вказівкою списку аргументів. Список аргументів необов'язковий і його можна не вказувати. Кожний аргумент має відповідати формальним параметрам, вказаним у заголовку, і мусить мати такий самий тип.

Формат звертання до функції такий:

<ім'я функції> {(фактичні параметри)};

Наведемо кілька прикладів звертання до функцій:

Power1(50.00E-3,36.678E03);

Resist1(34.678E-3,220.00);

1. Назвіть основні етапи розв'язку задач за допомогою комп'ютера?
2. Що називається моделлю, її види та призначення?
3. Дайте визначення алгоритму.
4. Які вимоги має задовольняти алгоритм?
5. Якими графічними елементами зображуються алгоритми?
6. Назвіть основні різновиди програмування?
7. Опишіть принцип модульного програмування.
8. Що таке мови програмування, якими вони бувають?
9. Назвіть особливості мови Паскаль?
10. Опишіть структуру програми мовою Паскаль?
11. Назвіть типи даних у мові Паскаль.
12. Як записуються і виконуються оператори мови Паскаль?
13. Що таке процедура, як вона описується і викликається?
14. Як користуватися функціями мови Паскаль?

4

Операційні системи

4.1 Поняття операційної системи. Її функції, склад і види

Операційна система — комплекс взаємопов'язаних програм, призначених для взаємодії між користувачем, пристроями апаратної частини та прикладними програмами, вона є невід'ємною частиною програмного забезпечення комп'ютерних систем обробки інформації.

Операційна система виконує такі функції:

- ❒ забезпечує взаємодію користувача з апаратними і програмними засобами комп'ютера (інтерфейс користувача);
- ❒ здійснює керування та координацію дій пристроїв апаратної частини та взаємодію цих пристроїв з прикладними програмами;
- ❒ забезпечує запуск і виконання прикладних програм;
- ❒ розподіляє й перерозподіляє ресурси між задачами.

Операційна система складається з таких частин:

- ❒ командного процесора, тобто програми або комплексу програм, який сприймає керівну інформацію оператора і, перетворивши її у послідовність внутрішніх команд, керує роботою пристроїв апаратної частини й виконанням прикладних програм;
- ❒ файлової системи;
- ❒ драйверів пристроїв апаратної частини.

Операційні системи поділяються на такі види:

- ❒ за кількістю користувачів, що одночасно працюють за комп'ютером, на одноосібні (персональні) та багатоосібні (колективні);
- ❒ за кількістю задач, які одночасно виконуються під керівництвом системи, на однозадачні та багатозадачні;
- ❒ за кількістю процесорів, що задіяні для виконання задачі, на однопроцесорні та багатопроцесорні;
- ❒ за типом взаємодії з користувачем на об'єктно-орієнтовані (здебільшого з графічним інтерфейсом) і командні (з текстовим інтерфейсом);
- ❒ за типом доступу користувача до комп'ютера на системи пакетної обробки, розподілу часу і реального часу;

- за розрядністю системної шини на 8-, 16-, 32-, та 64-розрядні;
- за типом використання ресурсів на локальні і мережні.

4.2 Операційна система MS–DOS. Основні складові частини

Операційна система побудована за модульним принципом, тобто у вигляді окремих частин (модулів), що взаємодіють один з одним. Така структура системи значно спрощує її розробку, дає змогу поліпшувати характеристики і розширювати функціональні можливості системи в цілому, модифікуючи і замінюючи окремі модулі.

MS–DOS складається з таких основних модулів:

- базової системи введення/виведення (BIOS — Basic Input Output System);
- блока початкового завантаження (Boot Record);
- модуля розширення базової системи введення/виведення (IO.SYS);
- модуля обробки переривань (MS–DOS.SYS);
- командного процесора (COMMAND.COM);
- драйверів пристроїв апаратної частини (ANSI.SYS, RAMDRIVE.SYS тощо).

Базова система введення/виведення розміщується в постійній пам'яті на материнській платі та вмикається разом з увімкненням живлення. Перша функція цієї програми — автоматичне тестування пристроїв апаратної частини комп'ютера безпосередньо після його ввімкнення. Якщо під час тестування виявляться несправності або помилки, то видається на екран відповідне повідомлення. Другою функцією BIOS є обслуговування переривань, тобто сигналів, якими периферійні пристрої повідомляють процесор про свій стан. Третьою функцією є виклик блока початкового завантаження після закінчення тестування.

Блок початкового завантаження розміщений у першому секторі нульової доріжки твердого або гнучкого диска і призначений для завантаження в оперативну пам'ять двох модулів операційної системи — розширення і обробки переривань.

Модуль розширення базової системи введення/виведення є додатком до BIOS і призначений для взаємодії з додатковими зовнішніми пристроями.

Модуль обробки переривань є основним модулем операційної системи і забезпечує взаємодію з прикладними програмами за допомогою системи переривань.

Командний процесор призначений для виконання внутрішніх і зовнішніх, а також прикладних програм. Внутрішніми командами вважаються ті, які командний процесор виконує безпосередньо.

Комплект зовнішніх команд призначений для реалізації складних функцій операційної системи. Кожна зовнішня команда здійснюється програмою, якій командний процесор передає керування.

Файлова система MS–DOS. Інформація, яку обробляє комп'ютер за допомогою операційної системи, організована в складну ієрархічну структуру. Сама операційна система також має таку структуру. Основним елементом цієї структури є файл. ***Файлом*** називається інформація, об'єднана за певною ознакою в одне ціле. У пам'яті файл також зберігається як одне ціле. Кожному файлу присвоюється унікальне ім'я, яке складається з двох частин, розділених крапкою: назви і розширення файла, наприклад, COMMAND.COM. Назва файла не може мати більше восьми символів, а розширення — не більше трьох. Розширення служить для позначення типу файла. Наприклад, **.doc**, **.txt** позначають текстові, **.exe**, **.com** — файли команд, готових до виконання. Для операцій з файлами зручно застосовувати символи * ?. Позначення ***.txt** стосується сукупності всіх текстових файлів з довільним іменем, які мають розширення **.txt**. Знак питання ? в імені файла означає довільний символ, який може стояти на цьому місці. Ім'я **Unit??.txt** позначає групу текстових файлів, які мають однакові чотири перші символи у назві та відрізняються один від одного п'ятим і шостим символами. Імена, які містять символи * ?, називаються ***шаблонами***.

Крім імені кожний файл характеризується розміром, тобто кількістю байтів інформації, що міститься у ньому.

Щоб позначити особливості різних файлів, застосовуються такі ***атрибути*** (параметри) файлів: **A** — неархівований; **R** — призначений тільки для читання; **S** — системний; **H** — захований. Атрибут **A** вказує на те, що інформацію, яка міститься у файлі, можна обробляти безпосередньо, в той час як архівований, перш ніж обробляти, потрібно розархівувати. Інформацію, яка міститься у файлах з атрибутом R, не можна змінити, її можна тільки прочитати. Цим забезпечується захист інформації від несанкціонованої зміни і випадкового стирання. Файли з атрибутом S входять до складу операційної системи і забезпечують її функціонування. Імена захованих (hidden file) файлів (що мають атрибут **H**) та їх характеристики не виводяться на екран.

Щоб забезпечити ефективність файлової структури, файли об'єднують у групи за певною ознакою, наприклад, які входять

до складу однієї програми, призначені для взаємодії з певним пристроєм апаратної частини тощо. Інформація, що містить імена файлів, їхній розмір, атрибути, час створення тощо, розміщується у спеціальному файлі, який називається ***каталогом*** або ***директорією***. Каталог може містити інші каталоги, які називаються ***підкаталогами***, і входити до каталогів вищого рівня. Каталог найвищого рівня (***головний каталог***), що не входить ні до якого каталога, називається ***кореневим***. Така розгалужена ієрархічна структура нагадує дерево і тому отримала назву ***дерева каталогів***. Кореневий каталог є стовбуром цього дерева, підкаталоги є вітками дерева, від яких відгалужуються тонші вітки (другого рівня) і так далі. На вітках розміщені листочки (файли). Зображення файлової структури у вигляді дерева широко застосовується у багатьох операційних системах, оскільки зручне у користуванні і має велику наочність. Щоб ефективно використовувати файлову структуру, необхідно вказати місце даного файла чи каталога на дереві каталогів, або так званий шлях до файла. ***Шляхом до файла*** називається послідовність підкаталогів, яка починається з назви диска, кореневого каталога, каталога першого рівня, другого і так далі аж до потрібного файла. Всі імена підкаталогів відділяються один від одного символом \ (***зворотний слеш***). Якщо до імені файла додати шлях до нього, то отримаємо ***повне ім'я***.

4.3 Операційна система Windows

Операційна система Windows, як і будь-яка операційна система, призначена для взаємодії програмних і апаратних засобів між собою і з користувачем (оператором). Операційна система Windows набула широкого поширення і фактично є стандартом для операційних систем.

Операційна система Windows має певні особливості порівняно з іншими операційними системами, зокрема з MS–DOS.

1. Windows є об'єктно-орієнтованою операційною системою, тобто все, з чим оперує система, трактується як ***інформаційний об'єкт***, що має певні властивості, і тим самим адекватно відображає об'єкти реального світу. Виконання програм здійснюється звичним для людини способом як взаємодія об'єктів. Інформаційними об'єктами є ***прикладні програми*** Windows (**Application**), які працюють під керуванням Windows (або, як кажуть, «під Windows»). За допомогою прикладних програм створюються ***документи***, які

теж є інформаційними об'єктами Windows. ***Апаратні засоби*** також трактуються як інформаційні об'єкти, оскільки взаємодія операційної системи з програмними і апаратними засобами здійснюється на загальних засадах, універсальним чином. ***Програми*** самої операційної системи, взаємодіючи одна з одною і з зовнішніми програмами і документами, також є інформаційними об'єктами. Таким чином, об'єктно-орієнтований підхід дає змогу зробити взаємодію різнорідних елементів системи універсальною і розглядати операційну систему як ***середовище***, в якому взаємодіють інформаційні об'єкти.

2. Windows є графічною операційною системою, в якій всі інформаційні об'єкти зображені на екрані графічним способом (Icons) (рис. 4.1). Це значно спрощує та унаочнює взаємодію (інтерфейс) користувача (GUI, Graphical User Interface) з комп'ютером, робить його набагато зручнішим. Переміщення об'єктів виконується просто й зручно за принципом Drag and Drop — перетягни і покинь.

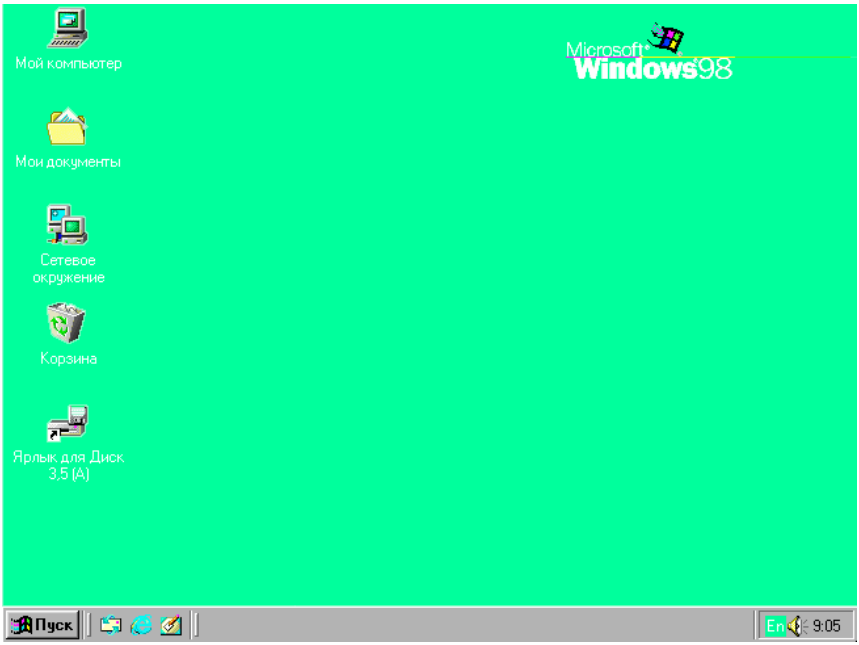


Рис. 4.1

3. Програми виконані в інших операційних системах, зокрема у MS–DOS, також можуть оброблятися Windows. Це надзвичайно цінна властивість, тому що у MS–DOS розроблено й експлуатується велика кількість програм.

4. Windows має засоби для підключення до локальних і глобальних комп'ютерних мереж, зокрема до всесвітньої мережі Internet.

5. Передбачено режим виконання кількох задач одночасно (multitasking) і режим одночасної роботи кількох користувачів (багатокористувацька система).

6. Реалізовано принцип точного відображення, так званий WYSIWYG (скорочення від англ. — What You See Is What You Get — що бачимо, те і отримуємо).

7. Використано технологію Plug and Play — увімкни і працюй. Ця технологія забезпечує самонастроювання апаратних засобів без відома користувача й автоматичну взаємодію апаратних і програмних засобів.

8. Одна програма може використовувати дані, створені іншою програмою за технологією DDE (скорочено від англ. Dynamic Data Exchange — динамічний обмін даними).

9. У Windows використовується механізм OLE (скорочено від англ. Object linking and Embedding — зв'язування і вбудовування об'єктів), який забезпечує використання об'єктів, створених однією прикладною програмою, іншими прикладними програмами, наприклад, графічних об'єктів, створених за допомогою графічних редакторів у текстових документах, створених текстовими редакторами. ***Вбудований*** у певну програму об'єкт сприймається цією програмою як свій власний, і передбачена можливість редагування об'єкта безпосередньо у даній програмі, для чого необхідно клацнути на ньому двічі мишею, що викличе програму, в якій був створений об'єкт, і дасть можливість редагувати. ***Зв'язаний*** об'єкт у даній прикладній програмі змінюватиметься автоматично зі зміною оригіналу в прикладній програмі, де він був створений.

10. Різні прикладні програми можуть обмінюватися інформацією через буфер обміну (Clipboard), частину оперативної пам'яті, в яку одна прикладна програма може розмістити інформацію, а інша цю інформацію прочитати і використати.

Файлова система Windows

Файлова система Windows успадкувала від файлової системи MS–DOS основні властивості, але має деякі додаткові можливості. Як і в MS–DOS, вся інформація в системі Windows розміщується й обробляється у вигляді файлів. Всі інформаційні об'єкти: прикладні програми; програми самої операційної системи; документи, створені за допомогою прикладних програм; апаратні засоби комп'ютерної системи — зберігаються і обробляються як файли.

Таким чином, ***файл*** — це контейнер (оболонка), в якому розміщується інформаційний об'єкт. Оскільки Windows — графічна операційна система, то кожний файл позначається умовним графічним зображенням — значком. Це дає змогу операції з файлами замінити маніпуляціями зі значками за допомогою миші чи клавіатури.

Часто зручніше маніпулювати не зі значками файлів, а тільки з вказівниками на файл — ярликами. ***Ярлик*** — це умовне графічне позначення файла з маленькою стрілочкою у нижньому лівому куті, який містить шлях до нього і забезпечує доступ з будь-якого місця операційної системи (рис. 4.2). Одному файлу можуть відповідати кілька ярликів, тобто до одного і того самого файла можна за допомогою ярликів забезпечити доступ з різних точок. Вилучення одного чи навіть всіх ярликів не призводить до вилучення самого файла.

Рис. 4.2

На відміну від MS–DOS, де назва файла не може перевищувати восьми символів, у Windows кількість символів у назві може бути до 255. Розширення може мати не більше трьох довільних символів. Незважаючи на те, що допускаються довільні символи у розширенні, доцільно використовувати на практиці загальноприйняті, уніфіковані розширення, наприклад, розширення **.doc** вказує, що об'єкт створений у текстовому редакторі **Word; .xls** — за допомогою **Excel** тощо. У Windows кожному розширенню, яке позначає тип файла, відповідає особливий значок, яким позначено на екрані файли даного типу. Деякі файли у Windows, здебільшого текстові, носять назву ***документи***.

Крім імені та розширення, кожний файл Windows характеризується розміром, часом створення або останньої зміни і ознакою, аналогічно файлу MS–DOS. Ці характеристики (атрибути) файла можна виводити на екран.

Рис. 4.3

Файли можна об'єднати за певною ознакою і розмістити дані про їх характеристики в особливий файл, який називається ***папкою*** (Folder) і позначається спеціальним значком на екрані комп'ютера (рис. 4.3). Наприклад, файли кожної прикладної програми доцільно об'єднати в одну папку. Папці присвоюється ім'я. Вимоги до імені папки такі ж, як і до імені файла, за винятком того, що розширення для імені папки не використовується. Інформаційні об'єкти — (папки) є метафорами реальних папок, у яких розкладають документи в офісному діловодстві. У MS–DOS аналогічні інформаційні об'єкти називаються каталогами. Папки, як і файли, також можна об'єднувати за певною ознакою

і розміщувати в окрему папку, наприклад, папки одного користувача. В окрему папку рекомендується розміщувати файли документів, створених за допомогою певної прикладної програми. Отже, кожна папка може містити у собі інші і входити до складу папок. Таким чином, папки і файли створюють розгалужену ієрархічну деревоподібну структуру, яка зберігається на магнітних (твердих та гнучких) і оптичних дисках. Самі диски є кореневими папками для всіх папок і файлів, які на ньому зберігаються. Такі папки позначаються спеціальним значком, відмінним від значка звичайної папки (рис. 4.4). Весь комп'ютер у цілому, з точки зору операційної системи є також папкою, позначеною спеціальним значком з написом **Мій комп'ютер (My computer)**.

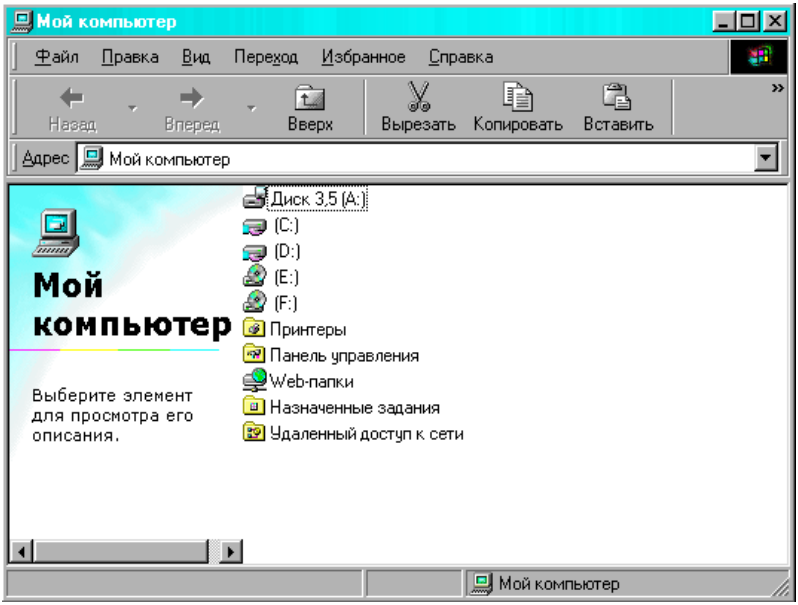


Рис. 4.4

Для орієнтації у складній, розгалуженій деревоподібній файловій структурі призначена спеціальна програма **Провідник** (Explorer), за допомогою якого можна знаходити потрібну папку чи файл, отримати їх характеристики і властивості, перемістити з однієї папки в іншу та виконувати деякі інші операції.

Засоби взаємодії користувача з комп'ютером

Користувач у об'єктно-орієнтованій системі Windows взаємодіє з інформаційними об'єктами, які представлені графічними

зображеннями за допомогою такої графічної форми як ***вікно*** (Window). Вікно є простим, зручним і універсальним графічним засобом взаємодії (інтерфейса). Саме за допомогою вікон користувач запускає прикладні програми, створює з їх допомогою документи, задає режими роботи апаратних пристроїв. Враховуючи виняткову важливість вікон, графічну операційну систему назвали Windows. Кожному інформаційному об'єкту відповідає своє вікно. ***Вікно прикладної програми*** виникає на екрані відразу після її запуску і дає змогу користувачеві працювати з цією програмою. За допомогою ***вікна документа*** створюється сам документ і обробляється відповідними прикладними програмами. На рисунку 4.5 як приклад наведено вікно прикладної програми Adobe PageMaker 6.5, призначеної для верстки книг. Книга, яку ви тримаєте у руках, зверстана саме за допомогою цієї програми.

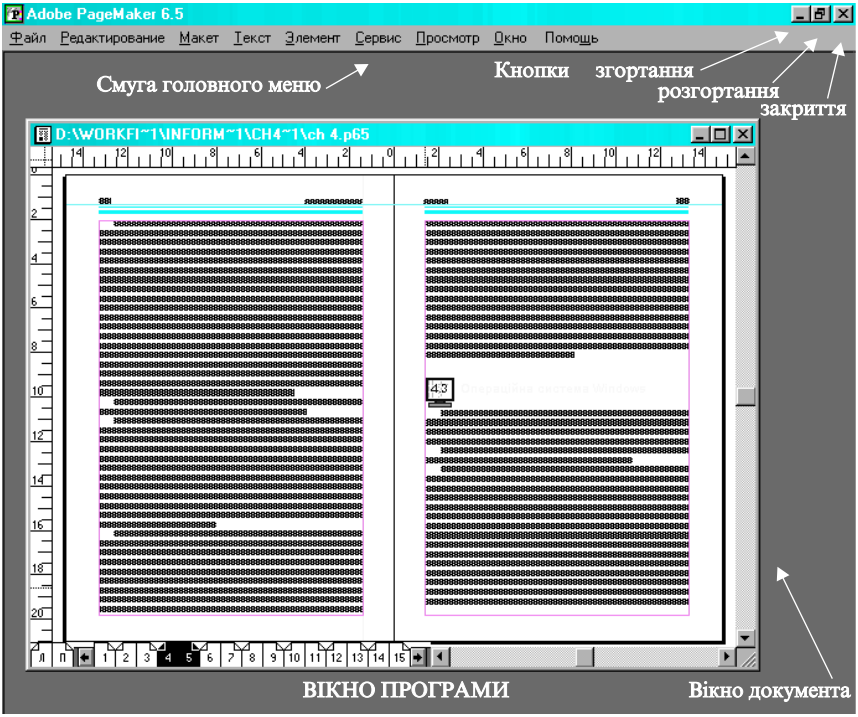


Рис. 4.5

Процес взаємодії користувача з інформаційним об'єктом за допомогою вікон здійснюється в режимі діалогу, тому вікна називаються ***діалоговими***. Для надання користувачеві інформації про властивості об'єктів призначені вікна ***властивостей***.

Незважаючи на те, що вікна призначені для взаємодії з різними інформаційними об'єктами, вони мають багато схожих і навіть однакових елементів. Розглянемо основні елементи і засоби керування вікон. Основою їх є рамка. Щоб змінити розмір вікна, слід підвести вказівник миші (стрілку) до межі рамки. Тоді форма вказівника змінюється, і він перетворюється на двоспрямовану стрілку, клацнувши лівою клавішею миші і утримуючи її в натиснутому стані, перетягуємо рамку на потрібну відстань. Відпустивши ліву клавішу, отримуємо вікно зміненого розміру в даному напрямі, наприклад горизонтальному. Аналогічним чином можна змінити розміри на екрані й у вертикальному напрямі. Щоб змінити розміри вікна одночасно у горизонтальному і вертикальному напрямках, застосовують вушко, розміщене у правому нижньому куті вікна. Якщо підвести вказівник миші до вушка, воно змінює свою форму і набуває вигляду двоспрямованої стрілки, розміщеної під кутом 45° до горизонту. Клацнувши лівою клавішею і утримуючи її у натиснутому стані, перетягнемо рамку по діагоналі вікна. Відпустивши ліву клавішу, отримуємо вікно нового розміру, зміненого одночасно у горизонтальному та вертикальному напрямку.

Для керування розмірами вікна призначені також три кнопки згортання (рис. 4.5), що розміщені у верхньому куті вікна. ***Кнопка згортання*** призначена для згортання вікна до значка з написом, який розміщується на панелі задач. У лівій частині цього значка розміщено графічне зображення (символ) даної прикладної програми чи документа, а праворуч — назва відповідного об'єкта. ***Кнопка розгортання*** служить для розгортання вікна на весь екран. Після того як вікно розгорнеться, кнопка змінює свій вигляд. Якщо клацнути мишею на такій кнопці, то вікно набере попереднього вигляду. ***Кнопка закриття*** служить для повного завершення роботи.

У верхній частині вікна розміщена ***смуга заголовка*** (рис. 4.5). На лівому її кінці розміщується умовне графічне позначення (значок) інформаційного об'єкта, а поряд — його назва. Якщо клацнути лівою клавішею миші на значку, то відкривається меню керування вікном, у якому повторюються останні три операції, ініційовані вже згаданими кнопками у правій частині смуги заголовка. Вікно можна переміщувати по екрану, якщо, клацнувши лівою клавішею миші й утримуючи її в натиснутому стані, пересувати у потрібному напрямі.

Під смугою заголовка розміщена смуга ***головного меню*** (рис. 4.5), яка складається з ряду кнопок з назвою функцій (команд). Якщо

клацнути мишею на кнопці, то відкривається додаткове підменю з назвами команд. Команди бувають доступними (enable) і недоступними (рис. 4.6). Назви доступних команд набрані насиченим чорним кольором. Їх можна виконати, клацнувши лівою клавішею миші. Недоступні (disables) команди набрані світло-сірим кольором, і їх у даному режимі роботи виконати не можна. Якщо напис закінчується чорним трикутником, то це вказує на групу команд, об'єднаних за певною ознакою. Ця група команд розміщується на додатковому підменю, що з'являється, якщо клацнути на написі з трикутником лівою клавішею миші.

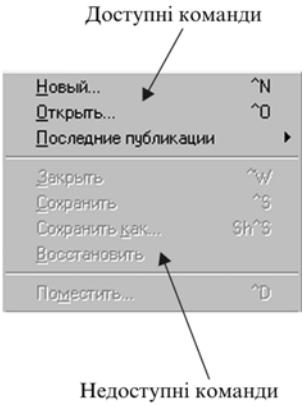


Рис. 4.6

Напис команди, що закінчується трьома крапками ..., вказує на те, що для їх виконання буде виведено додаткове діалогове вікно, в якому необхідно буде ввести параметри, режими роботи, дані. Щоб знайти потрібну команду в головному меню, досить клацнути мишею на будь-якому значку і, переміщуючи вказівник миші на інші написи, проглянути підменю, які розкриваються, як тільки на них буде наведено вказівник.

Нижче смуги головного меню може розміщуватися, у разі необхідності, панель інструментів. ***Панель інструментів*** — це ряд кнопок тих команд, які найчастіше використовуються в роботі. Кнопки та відповідні команди можна додавати на панель інструментів у разі необхідності або вилучати їх.

Більшу частину вікна займає ***робоча область***, у якій розміщено той інформаційний об'єкт (прикладна програма, документ, файл апаратного засобу, інші файли та папки), для якого призначено це вікно. Якщо інформаційний об'єкт не вміщується в робочу область, то внизу і справа з'являються лінійки прокручування. Клацнувши мишею на кнопках із зображенням трикутника (∇, або Δ), можна перемістити об'єкт у відповідному напрямі. Плавне переміщення вмісту робочої області можна здійснити пересуванням повзунка за допомогою миші. Щоб перемістити вміст на розмір екрана, слід клацнути мишею на лінійці між кнопкою з трикутником і повзунком.

Діалогові вікна

Діалогові вікна призначені для задання потрібних параметрів прикладній програмі, документу, файлу, папці чи іншому

інформаційному об'єкту. Від звичайних вікон вони відрізняються незмінністю розмірів. Розглянемо елементи керування діалогових вікон (рис. 4.7).

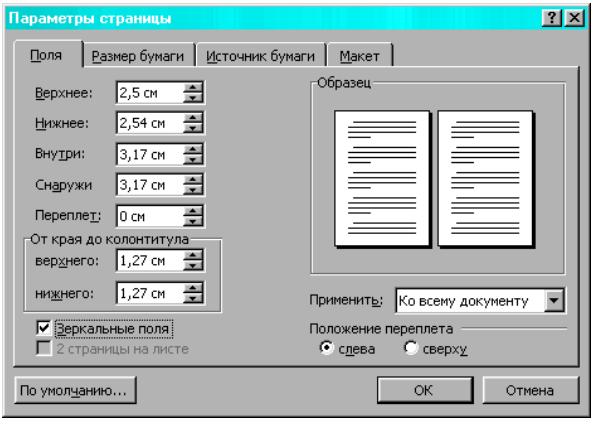


Рис. 4.7

Поле введення (***текстове поле***) використовується для введення текстової інформації (наприклад, імені файла).

Список призначений для вибору одного параметра з кількох можливих, тобто список — це меню значень параметра. Зовнішньо список має вигляд поля введення, в якому стоїть значення параметра, який діє в даний час. Щоб розкрити список, необхідно клацнути на кнопці з трикутником, що розміщена справа. Після того, як список розкриється, можна знайти і вибрати в ньому потрібне значення параметра. Часто всього списку не видно, і, щоб його побачити, необхідно скористатися смугою прокручування. На смузі прокручування розміщено дві кнопки зі стрілками і повзунок. Клацнувши лівою клавішею миші на одній із кнопок, можна прокрутити список на один елемент у відповідному напрямі. Якщо ж клацнути мишею на смузі прокручування між повзунком і відповідною кнопкою, то список переміститься на значення, що відповідає розміру екрана, тобто буде виведена наступна порція даних. Щоб плавно переміщуватися вздовж списку на великі відстані, слід перетягувати за допомогою миші повзунок по смузі прокручування, положення якого відповідає положенню тієї частини списку, яка відображується на даний момент. Якщо у фрагменті списку, що відображується, з'явиться потрібний параметр, можна вибрати його, клацнувши на ньому лівою клавішею миші. Залежно від роду параметра, значення якого відображається у списку, можливі два види списків: список, у якому значення можна тільки

вибрати із наявних, і список, у який крім наявних можна ввести нове значення параметра за допомогою поля введення. Другий вид списку називається ще комбінованим списком.

Регулятор призначений для плавної зміни значення параметра (як правило числового) у діапазоні значень, він аналогічний за функціональним призначенням до регуляторів фізичних величин електронних чи вимірювальних приладів, розміщених на їх передніх панелях. За зовнішнім виглядом регулятор нагадує поле введення, справа від якого розміщені дві кнопки зі стрілками. Значення параметра, виведене в полі, збільшуватиметься, якщо натиснути мишею на верхній кнопці й утримувати її натиснутою, і зменшуватиметься, якщо виконати ці операції з нижньою кнопкою. Ввести потрібне число можна і з клавіатури.

Перемикачі (***option buttons***) призначені для вибору одного з режимів роботи, для чого необхідно клацнути на вибраному режимі лівою клавішею миші, після чого проти вибраного режиму з'явиться символ.

Прапорці (***check boxes***), на відміну від перемикачів, не залежать один від одного. Їх можна встановлювати або знімати у будь-яких сполученнях. Може бути встановлено кілька прапорців або ні одного. Щоб встановити прапорець, необхідно клацнути на ньому, і він позначається галочкою чи хрестиком. Повторне клацання на вже встановленому прапорці знімає його і позначка зникає.

Якщо параметрів, що задаються, багато і вони не вміщуються у вікні, то їх розміщують на кількох аркушах, які називаються ***вкладками***. У вікні виводиться вміст тільки однієї вкладки. Решту вкладок не видно, а показуються тільки їх язички, на яких розміщена назва вкладки. Щоб відкрити потрібну вкладку, слід клацнути на її язичку.

Багато діалогових вікон мають кнопку виклику допомоги, яка позначена знаком питання **?** і розміщена поруч з кнопкою закриття вікна. Якщо клацнути клавішею миші на цій кнопці і відпустити її, то поруч із вказівником миші з'явиться знак питання. Його слід перемістити за допомогою миші до того об'єкта, робота з яким викликала утруднення, і необхідна додаткова інформація.

Після того як у діалоговому вікні встановлено потрібні параметри, можна або прийняти їх, або відмовитися від зроблених змін. Для цього призначено ***командні кнопки***. Зміни параметрів будуть прийняті, якщо натиснути командну кнопку **OK** або кнопку **Apply** (застосувати), і відкинуті, якщо натиснути кнопку **Cancel**. Закриття діалогового вікна за допомогою відповідної кнопки у правому верхньому куті рівноцінно відмові від змін і натискуванні кнопки **Cancel**.

Робочий стіл

Після вмикання живлення комп'ютера на екрані через деякий час з'являється головне вікно Windows, що має назву ***робочий стіл*** **(Desktop)** (рис. 4.1). Ця назва обумовлена тим, що екран комп'ютера нагадує робочий стіл офісного працівника, на якому розкладено документи у папках і засоби, за допомогою яких ці документи створюються. На робочому столі можна розмістити стільки папок, файлів і ярликів, скільки це потрібно для роботи, але мінімально необхідні такі об'єкти: **Мій комп'ютер (My Computer)**, **Мережне оточення (Network Neighborhood)**, **Кошик (Recycle Bin)**.

Папка **Мій комп'ютер** є кореневою папкою всієї ієрархічної файлової системи комп'ютера. Вона містить папки всіх програмних і апаратних засобів комп'ютера: твердих і гнучких магнітних дисків, оптичного диска, принтера тощо.

Папка **Мережне оточення** містить значки комп'ютерів, до яких є доступ у локальній комп'ютерній мережі. Відкривши доступ до певного комп'ютера, можна скористатися його програмними й апаратними ресурсами.

У папці **Кошик** зберігаються раніше вилучені файли і папки. Помилково вилучену інформацію можна поновити, знайшовши її в **Кошику**, оскільки файли, що знаходяться в **Кошику**, остаточно не вилучені з диска і займають колишній обсяг, тому час від часу треба звільняти **Кошик**.

На робочому столі доцільно розміщувати тільки ті документи, прикладні програми, папки, з якими користувач працює часто. Небажано надмірно захаращувати поверхню робочого столу документами і програмами, які застосовуються дуже рідко і викладаються на робочий стіл «про всяк випадок». Це надає робочому столу неохайного вигляду і заважає у роботі.

Розкрити документ або запустити прикладну програму, що знаходиться на робочому столі, дуже просто: для цього достатньо двічі клацнути на відповідному значку або ярлику лівою клавішею миші.

У нижній частині вікна **Робочий стіл** розміщено **Панель задач** — смуга сірого кольору, на якій, у середній її частині, розміщено значки документів і прикладних програм, які відкрито на даний момент. У правому кінці **Панелі задач** розміщено годинник і різні індикатори, наприклад, режим роботи клавіатури — для вибору мови, гучномовець — для регулювання гучності, екран — для вибору кольорової палітри і встановлення роздільної здатності. Поряд розміщують значки деяких програм, що можуть знадобитися у будь-який момент роботи і тому мають бути «під рукою»,

наприклад, словник для перекладу незнайомого слова, антивірусна програма тощо.

У лівому кінці **Панелі задач** міститься кнопка **Пуск (Start)** для доступу до **Головного меню** комп'ютера. **Головне меню** складається з таких пунктів: **Програми** (Programs), **Документи** (Documents), **Настройка** (Setting), **Пошук** (Find), **Довідка** (Help), **Виконати** (Execute) (рис. 4.8).

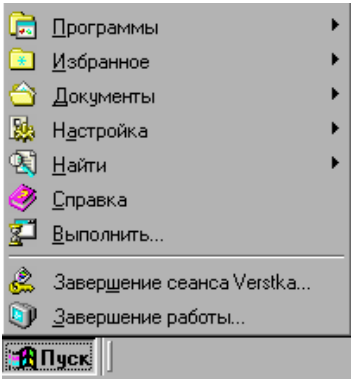


Рис. 4.8

Програми є каскадним меню, що забезпечує швидкий і зручний доступ до будь-якої прикладної програми, встановленої на комп'ютері, а також до стандартних програм, що поставляються разом з операційною системою. Для запуску програми, що входить до цього меню, необхідно її вибрати за допомогою миші з меню, навівши на неї вказівник миші (тоді вона виділиться інтенсивним кольором), і запустити, клацнувши лівою клавішею миші.

Документи — це меню, що показує останні папки і документи, що були відкриті за останній час.

Настройка є каскадним меню, що забезпечує доступ до всіх програмних і апаратних пристроїв комп'ютера, і дає змогу змінювати конфігурацію системи чи її частин.

Програма **Пошук**, як випливає з її назви, призначена для пошуку потрібного файла чи папки у файловій системі комп'ютера.

Щоб знайти довідкову інформацію про операційну систему Windows та її складові і взаємодію користувача з нею, призначена **Довідка**.

Команда **Виконати** забезпечує швидкий запуск прикладних програм.

Крім зазначених вище пунктів, у меню **Пуск** є пункт **Завершення роботи**, який забезпечує безпечне і надійне завершення роботи комп'ютера. Вимикати живлення комп'ютера не можна у довільний час, оскільки це може призвести до виходу з ладу пристроїв комп'ютера, тому для цього передбачена спеціальна процедура. Для правильного завершення роботи комп'ютера слід виконати такі дії:

- ❒ зберегти дані у всіх документах і закрити їх;
- ❒ натиснути кнопку **Пуск** і вибрати команду **Завершення роботи** (Shut Down). На екрані з'явиться діалогове вікно;
- ❒ вибрати режим **Вимкнути комп'ютер** (Shut Down the Computer);

- ❒ натиснути командну кнопку **ОК** у нижній частині вікна;
- ❒ дочекатися появи на екрані повідомлення **Вимкніть електроживлення** і виконати його.

Провідник

Файлова система комп'ютера має складну ієрархічну деревоподібну структуру, що утруднює пошук і переміщення файлів та документів, обмін даними між інформаційними об'єктами. Щоб зробити операції над інформаційними об'єктами легкими, зручними і наочними, призначена програма **Провідник (Explorer)**.

Запускати **Провідник** можна або з головного меню комп'ютера, з розділу **Програми**, або за допомогою значка цієї програми, заздалегідь розміщеного на **Робочому столі**. Після запуску програми на екрані з'являється вікно **Провідника** (рис. 4.9).

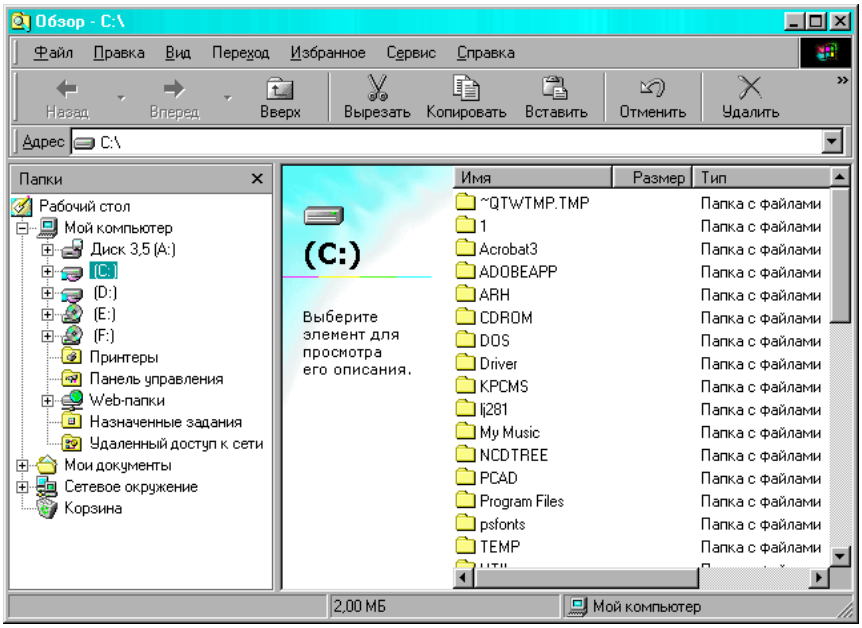


Рис. 4.9

Вікно **Провідника** складається з двох частин: на лівій половині зображаються програмні та апаратні ресурси комп'ютера у вигляді ієрархічної, деревоподібної структури, а на правій — вміст вибраної папки. Кожний рівень ієрархії зображується вертикальною штриховою лінією. На цій лінії розміщено вузли, від яких відходять горизонтальні вітки деревоподібної структури. Кожна горизонтальна вітка вказує на папку, і тоді вузол зображується

штриховим квадратиком, або на файл, і тоді вузол зображується як відгалуження лінії. Якщо у квадратику, що зображує вузол, стоїть знак + , то це означає, що папка, приєднана до цього вузла, містить інші папки. Щоб проглянути їх, необхідно клацнути лівою клавішею миші на вузлі. Після цього знак + зміниться на −, і виведеться на екран вміст даної папки. Щоб знову згорнути папки і файли всередину даної папки, потрібно клацнути мишею на вузлі зі знаком − , або двічі клацнути на самій папці. Якщо розгорнуто багато вузлів, то для перегляду довгої деревоподібної структури зручно скористатися вертикальною лінійкою прокручування, яка з'являється, коли файлова структура не вміщується у вікні.

Переміщення по дереву в лівій панелі — панелі ресурсів — не змінює вмісту правої панелі.

Кореневою папкою файлової структури комп'ютера є папка **Робочий стіл**. Від **Робочого столу** вниз відходить вертикальна лінія найвищого рівня ієрархії. Від неї відходять горизонтальні лінії, на яких розміщені папки, що входять до складу папки **Робочий стіл** і мають нижчий рівень. Це папки: **Мій комп'ютер**, **Мережне оточення**, **Кошик**. У свою чергу папка **Мій комп'ютер** містить папки всіх ресурсів даного комп'ютера, папка **Мережне оточення** — ресурси всіх комп'ютерів, до яких є доступ у локальній мережі. У **Кошику** містяться папки і файли, вилучені з дерева файлів, але остаточно не знищені.

За допомогою **Провідника** можна виконувати такі операції над файлами і папками: переміщення по деревоподібній структурі, проглядаючи об'єкти, виділення, створення, переміщення об'єктів та їх копіювання, перейменування, вилучення.

За допомогою **Провідника** зручно проглядати вміст всіх папок структури, переміщуючись по лівій панелі і виводячи вміст на правій панелі.

Файл чи папку можна виділити, клацнувши на об'єкті один раз мишею. Після цього об'єкт виділяється темним кольором. Для виділення групи файлів потрібно спочатку виділити один з них, а потім, натиснувши клавішу **Ctrl** і утримуючи її у натиснутому стані, продовжувати виділяти потрібні файли чи папки. Якщо клацнути мишею на вже виділеному об'єкті, то виділення скасовується. Коли потрібна група папок і файлів буде виділена, клавішу **Ctrl** відпускають.

Для виділення кількох файлів чи папок, що розміщені поряд, необхідно клацнути мишею на першому файлі в групі, а потім, натиснувши клавішу **Shift**, — на останньому.

Можна також виділити файли, обвівши їх прямокутною рамкою за допомогою миші.

Крім того, у меню вікна **Провідника** у пункті **Правка** є дві команди: **Виділити все** (Select All) і **Обернути виділення** (Invert Selection).

Створення нової папки. Спочатку слід виділити у лівій частині вікна **Провідника** папку, в якій міститиметься створена папка. Далі вибрати у меню **Файл** (File) команду **Нова-Папка** (New-Folder). На правій панелі з'явиться значок нової папки з текстовим полем, у якому можна набрати ім'я нової папки, після чого натиснути клавішу **Enter**.

Можна створити нову папку і за допомогою контекстного меню, яке з'явиться, якщо клацнути правою клавішею миші у правій половині вікна **Провідника**. У цьому меню також є команда **Нова-Папка**, результат дії якої аналогічний попередньому.

Переміщення і копіювання файлів і папок. Є два варіанти виконання цих дій. Один пов'язаний з використанням **Буфера обміну** (Clipboard), а другий — з перетягуванням значків мишею.

Через **Буфер обміну** файли копіюються таким чином. Після того, як потрібні файли і папки знайдені і виділені, вибирається команда **Копіювати** із меню **Правка**, і тим самим файли заносяться у **Буфер обміну**. Потім на лівій панелі виділяється папка, в яку потрібно помістити скопійований файл, і вибирається команда **Вставити** (Paste) з меню **Правка**. Якщо файли переміщуються, а не копіюються, то слід використовувати команду **Вирізати** (Cut). Таких самих результатів можна досягнути, якщо користуватися кнопками панелі інструментів або команд, що задаються комбінацією командних клавіш на клавіатурі.

Щоб скопіювати або перемістити файли чи папки перетягуванням за допомогою миші, необхідно виконати такі дії. Виділити потрібні файли і папки у правій панелі **Провідника**, а в лівій панелі домогтися того, щоб папка, в яку буде переміщений файл чи папка, стала видимою. Якщо потрібно перемістити виділену папку чи файл, то її перетягують за допомогою миші з правої панелі у потрібну папку на лівій панелі, а якщо скопіювати, то під час перетягування слід натиснути й утримувати у натиснутому стані клавішу **Ctrl**. Під час копіювання за допомогою перетягування, поряд з вказівником миші з'являється знак плюс, який підтверджує, що це копіювання, а не переміщення.

Перейменування файлів і папок. Щоб перейменувати файл чи папку, слід спочатку виділити його, а потім клацнути лівою клавішею миші на імені, після чого навкруги імені з'являється рамка текстового поля, всередині якої знаходиться курсор. Далі

потрібно перемістити курсор у потрібну позицію, набрати за допомогою клавіатури нове ім'я і натиснути на клавішу **Enter**. Дуже важливо, щоб між виділенням файла і виділенням імені була достатня пауза, інакше вони сприйматимуться як подвійне клацання, і замість перейменування запуститься програма, пов'язана з даним файлом.

Вилучення файлів і папок. Для вилучення слід виділити непотрібні файли і папки, а потім вибрати команду **Вилучити** (Delete) з меню **Файл** (File). У діалоговому вікні, що з'явиться після цього, слід підтвердити вилучення.

Після вилучення файли поміщаються у спеціальну папку **Кошик** і зберігаються там упродовж певного часу. З цієї папки їх можна відновити, якщо вони були вилучені помилково. Якщо потрібно остаточно знищити файл, і він не потрапив у Кошик, то під час вибору команди **Вилучити** слід натиснути і утримувати клавішу **Shift**, або скористатися комбінацією клавіш **Shift-Delete**.

Форматування дискет

На магнітні диски тверді і гнучкі наносяться концентричні доріжки (треки), що діляться на сектори. У сектори записується інформація. Перед першим застосуванням диск форматується.

Повторне форматування диска виконується у разі:

- ❒ виникнення фізичних вад або дефектних місць, коли інформація не читається;
- ❒ зараження вірусом, коли не можна вилікувати без втрати інформації;

Під час форматування диска на ньому позначаються дефектні місця, що з'явилися. Їх позиції заносяться у таблицю розподілу файлів FAT (File Allocation Table).

Форматування диска здійснюється за командою **Форматувати** (Format), яку можна вибрати такими способами:

1. Відкрити папку **Мій комп'ютер**. Потім клацнути правою клавішею миші на значку диска, який треба відформатувати. У контекстному меню, що з'явиться на екрані, слід вибрати команду **Форматувати**.

2. Відкрити програму **Провідник**, вибрати значок диска на панелі вмісту вибраної папки і клацнути правою клавішею миші. У контекстному меню, що з'явиться на екрані, потрібно вибрати команду **Форматувати**.

Захист інформації від комп'ютерних вірусів

Комп'ютерний вірус — невелика за обсягом програма, що може проникати в інші програми (заражати їх), дія якої призводить до негативних наслідків, які можуть проявлятися по-різному:

- ❒ пошкоджуються деякі файли;

- ❒ програми перестають виконуватися або виконуються неправильно; на екран монітора виводяться непередбачені повідомлення або символи;
- ❒ робота комп'ютера уповільнюється.

Заражені програми з одного комп'ютера можуть бути перенесені на інші комп'ютери за допомогою дискет або локальної мережі.

Для захисту інформації від комп'ютерних вірусів використовуються загальні та програмні засоби.

До загальних засобів належать:

- ❒ резервне копіювання інформації (створення копій файлів і системних областей дисків);
- ❒ розмежування доступу до інформації (запобігання несанкціонованого використання інформації).

До програмних засобів захисту належать різні антивірусні програми.

Програми-детектори призначені для знаходження заражених файлів.

Програми-лікарі призначені для лікування заражених дисків і програм. Відновлення програм полягає у вилучені із зараженої програми тіла вірусу.

Програми-ревізори запам'ятовують стан програм і системних областей дисків, а потім порівнюють його з поточним станом. У разі невідповідності даних виводиться повідомлення про це.

Лікарі-ревізори призначені для виявлення змін у файлах і системних областях і в разі виявлення цих змін повертають файли у початковий стан.

Програми-фільтри призначені для перехоплення звернень до операційної системи, які використовують віруси для розмноження, і повідомляють про це користувача.

1. Що таке операційна система і для чого вона призначена?
2. З яких частин складається операційна система MS–DOS?
3. Охарактеризуйте файлову систему MS–DOS?
4. Які особливості операційної системи Windows?
5. Які засоби взаємодії з користувачем має система Windows?
6. Які елементи мають діалогові вікна системи Windows і для чого вони призначені?
7. Що таке Робочий стіл і які функції він виконує?

5

Обробка текстової інформації за допомогою комп'ютера

5.1 Структура документа

Текстова інформація — одна з найбільш поширених видів інформації. До недавнього часу основним носієм текстової інформації був папір. З часу винайдення друку на папері написано багато книг, газет, журналів, наукових праць тощо. Але книги мають чималий обсяг, їх потрібно десь зберігати, з цією метою побудовані великі споруди (бібліотеки).

Комп'ютери дали змогу створювати, обробляти і зберігати текстову інформацію в ***електронному*** вигляді. Наприклад, на один компакт-диск можна записати і зберігати в такому вигляді близько 60 книг по 500 сторінок кожна. Порівняйте маленький компакт-диск і 60 томів.

Текстова інформація існує у формі ***документів***, які є основною одиницею текстової інформації. Документами є стаття у газеті чи журналі, доповідь, замітка, лекція, літературний твір (оповідання, вірш), лист тощо.

Найменшим неподільним елементом текстової інформації є ***символ***. Символи поділяються на такі види як букви, розділові знаки, цифри, спеціальні символи.

Основними символами є ***букви***. До алфавіту української мови входить 32 букви, його основою є ***кирилиця***. Більшість народів світу користується латинським алфавітом, він був створений у Стародавньому Римі.

У деяких текстах, особливо науково-технічних, широко застосовуються ***спеціальні символи***: букви грецького алфавіту, знаки математичних операцій, спеціальні математичні символи тощо.

Числа на письмі записуються цифрами.

Із букв (символів) складаються слова, які об'єднуються у ***рядки***, між словами і символами в рядку ставляться пробіли.

Сукупність рядків, об'єднаних єдиним змістом, утворюють ***абзац***. За традицією текст поділяється на ***сторінки*** для нанесення його на папір.

5.2 Введення, редагування, форматування тексту

Для введення, редагування, форматування та інших операцій з текстом застосовуються спеціальні програми, що називаються текстовими редакторами або текстовими процесорами. Одним із найпоширеніших текстових редакторів є Microsoft Word. На рисунку 5.1 наведено вікно цієї програми.

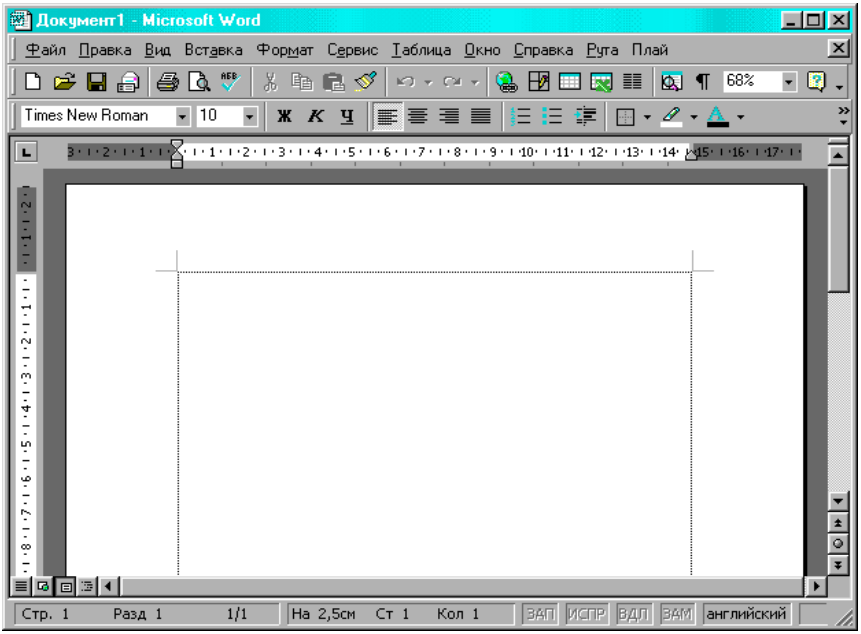


Рис. 5.1

Введення, редагування, форматування символів. Символи вводяться, як правило, з клавіатури на місце курсора — темної вертикальної риски, що періодично підсвічується в одному з двох режимів: ***вставки*** або ***заміни***, які перемикаються за допомогою клавіші **Insert**. У режимі заміни індикатор **ЗАМ** рядка стану має чорний колір, а в режимі вставки — сірий. У режимі вставки символи вводяться на місце курсора, а символи, що знаходяться за курсором зсуваються праворуч. У режимі заміни символи вводяться на місце існуючих, замінюючи їх.

Але не всі символи є на клавіатурі, щоб увести символ, якого немає на клавіатурі, можна скористатися командою **Символ** з меню **Вставка**. За цією командою відкривається вікно **Символ**, що має

дві вкладки: **Символи** і **Спеціальні символи** (рис. 5.2). У полі **Шрифт** встановлюється шрифт, що містить потрібні символи. Як правило, це шрифт **Symbol**, хоч у деяких випадках можна скористатися іншими типами шрифту.

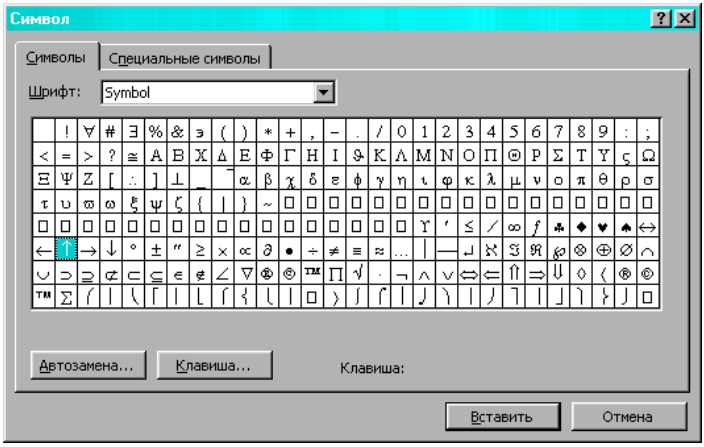


Рис. 5.2

Якщо клацнути мишею на потрібному символі, то з'явиться його збільшене зображення. Щоб вставити потрібний символ на місце курсора у тексті, слід двічі клацнути мишею на ньому або натиснути на кнопку **Вставити.** Оскільки кожний символ має свій номер, то його можна ввести, набравши його порядковий номер на цифровій клавіатурі при натиснутій клавіші **Alt**. Призначивши символу комбінацію клавіш за допомогою кнопки **Клавіша** і відповідного діалогового вікна, можна вводити цей символ з клавіатури.

Щоб вилучити символ, зліва від курсора слід натиснути клавішу **Back Space**, а справа — клавішу **Delete**.

Форматування символів, тобто встановлення зовнішнього виду символів, здійснюється командою **Формат–Шрифт** і відповідного діалогового вікна (рис. 5.3). За допомогою цієї команди можна встановити такі параметри символів:

- ❒ шрифт або гарнітуру символу, що вибирається зі списку шрифтів вкладки **Шрифт**. Найчастіше застосовуються такі шрифти (гарнітури): **Times New Roman, Arial, Courier New;**
- ❒ стиль шрифту, що вибирається зі списку: **Звичайний, Напівжирний, Курсив, Напівжирний курсив;**
- ❒ розмір шрифту, що вибирається зі списку. Розмір шрифту за традицією вимірюється у пунктах (пт);
- ❒ поле **Підкреслення** дає змогу задати параметри для тексту. В списку є варіанти вибору різних видів підкреслень;

- ❒ колір символів задається у полі введення **Колір**;
- ❒ спеціальні ефекти (закреслення, подвійне закреслення, верхні та нижні індекси, тінь, контур) для виділення символів, які можна задати групою параметрів **Ефекти.**

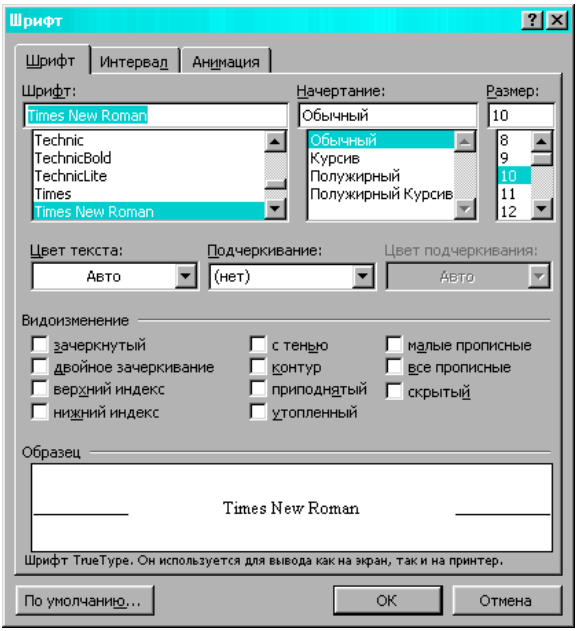


Рис. 5.3

На вкладці **Інтервал** можна задати інтервал між символами у слові. Міжсимвольний інтервал вибирається зі списку **Інтервал** однойменної вкладки і може бути звичайним, ущільненим, розрідженим. Ступінь ущільнення чи розрідження задається відповідним регулятором.

За допомогою поля **Зміщення** можна задати зміщення символів вгору і вниз відносно базової лінії. Значення зміщення можна встановити за допомогою відповідного регулятора, розміщеного на вкладці справа від поля.

Різні анімаційні ефекти, що привертають увагу користувача до певного слова чи фрагмента тексту, можна задати за допомогою вкладки **Анімація**. Це можуть бути маленькі рисочки, різних форм зірочки тощо.

Для керування регістром введених символів використовується команда **Регістр** з меню **Формати.** Перед її виконанням необхідно виділити фрагмент тексту, в якому необхідно змінити регістр. У діалоговому вікні **Регістр**, що з'являється під час виконання

відповідної команди, оператор може встановити такі режими:

- **Як у реченнях** — перший символ у першому слові речення переводиться у верхній регістр;
- **всі малі** — всі символи у вибраному фрагменті переводяться у нижній регістр;
- **ВСІ ВЕЛИКІ** — всі символи у вибраному фрагменті переводяться у верхній регістр;
- **Починати З Великих** — у верхній регістр переводиться перший символ кожного слова виділеного фрагмента;
- **зМІНИТИ РЕГІСТР** — символи верхнього регістру переводяться у нижній і навпаки.

Форматування абзаців

Абзац (Item) — це частина текстової інформації, об'єднана за спільним змістом і обмежена символом ¶ (Кінець абзацу). Абзац — структурний елемент документа. Щоб під час введення текстової інформації перейти до нового абзацу, потрібно натиснути клавішу **Enter**, а щоб перейти на новий рядок у межах абзацу, слід скористатися комбінацією клавіш **Shift-Enter**.

Основні параметри форматування абзаців встановлюються командою **Абзац** (Item) меню **Формат** (Format) і відповідного діалогового вікна (рис. 5.4), яке з'являється при виборі цієї команди.

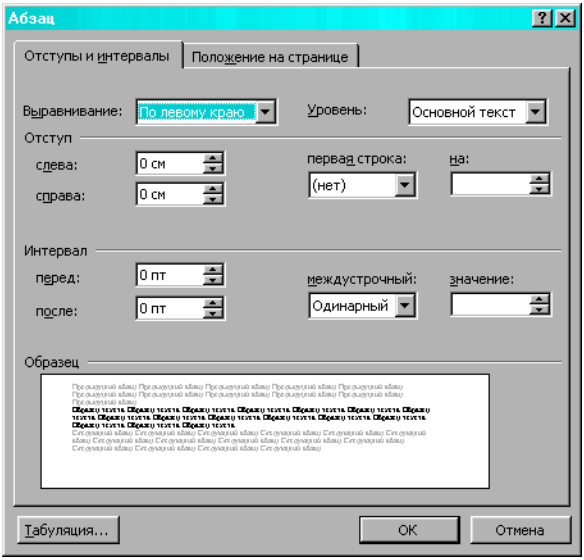


Рис. 5.4

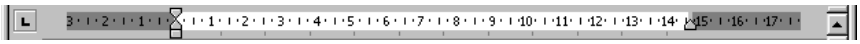
Цю команду можна вибрати із контекстного меню, що з'являється, якщо клацнути у довільному місці правою клавішею миші.

Діалогове вікно **Абзац** має дві вкладки: **Відступи й інтервали** та **Положення на сторінці**.

Вирівнювання рядків абзацу можна здійснити по лівому краю, по центру, по правому краю або по ширині (по обох краях). За замовчанням встановлюється режим вирівнювання по лівому краю, при цьому лівий край тексту буде рівним, а правий — зубчастим. Запровадити спосіб вирівнювання можна за допомогою списку **Вирівнювання** вкладки **Відступи й інтервали** діалогового вікна **Абзац**, кнопками  або клавішами **Ctrl+L, Ctrl+R, Ctrl+E, Ctrl+S**.

Відступи абзацу зліва і справа відносно робочого поля сторінки задаються за допомогою відповідних регуляторів вкладки **Відступи й інтервали** діалогового вікна **Абзац**. Аналогічно встановлюється відступ першого рядка кожного абзацу. На практиці межі робочого поля й абзаців тексту, як правило, збігаються, кожний абзац має відступ для першого, «червоного рядка».

Встановлення відступів можна швидко і наочно виконувати також за допомогою горизонтальної лінійки форматування пересуванням відповідних індикаторів.



Інтервал між рядками тексту, абзацу можна вибирати зі списку **Міжрядковий** (рис. 5.4). Можна вибрати одне із таких значень:

- ***одинарний*** — відстань, що дорівнює висоті шрифту найбільшого розміру, який застосовується в рядку, і додатковий простір, розмір якого залежить від шрифту;
- ***полуторний*** — відстань, яка у 1,5 рази перевищує відстань при одинарному інтервалі;
- ***подвійний*** — відстань між рядками, яка вдвічі перевищує відстань при одинарному інтервалі;
- ***мінімум*** — мінімальний інтервал, який вибирається для дуже великих шрифтів;
- ***точно*** — інтервал, який точно дорівнює значенню, вказаному в полі **Значення**;
- ***множник*** — інтервал формується перемножуванням одинарного інтервалу на значення, вказане у полі **Значення**.

Міжрядкові інтервали **одинарний, полуторний, подвійний** можна встановити також кнопками  .

Щоб поліпшити зовнішній вигляд тексту, між абзацами задаються інтервали як перед абзацом, так і після абзацу за допомогою відповідних регуляторів вкладки **Відступи й інтервали** діалогового вікна **Абзац**.

Для надання тексту більшої виразності його окремі абзаци та заголовки іноді вкладають у рамки з тінню і фоном за командою

Формат–Границі і **Заливка**, яка викликає на екран однойменне вікно з вкладками: **Границя**, **Сторінка** і **Заливка**. Засобами вкладки **Границя** (рис. 5.5) встановлюється вид рамки, тип, колір і ширина ліній. Результат виводиться у вікні **Зразок**. У разі використання вкладки **Сторінка** характер дій залишається той самий, але вони поширюються тепер на цілу сторінку. Засобами вкладки **Заливка** можна вибрати колір заливання і фон, тип візерунка.

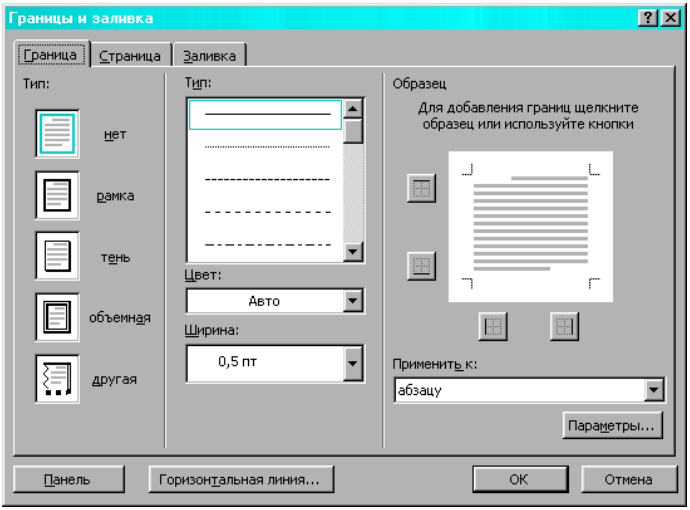


Рис. 5.5

Швидкий і спрощений варіант обрамлення виділених заголовка, абзацу, фрагмента тексту реалізується командою **Зовнішні границі** кнопкою . Параметри задаються за допомогою діалогового вікна, яке з'являється під час виконання цієї команди.

Щоб уникнути розривання абзаців під час форматування сторінок тексту, слід встановити перемикач **Не розривати абзац** вкладки **Положення на сторінці** вікна **Абзац**.

Форматування сторінок

Поділ документа на сторінки здійснюється автоматично, якщо ввімкнений режим **Фоновий поділ на сторінки**, розміщений на вкладці **Загальні** діалогового вікна **Параметри** з меню **Сервіс**. У багатьох випадках користувача це не задовольняє, і поділ здійснюється вручну. Для цього встановлюють курсор на місце поділу і набирають команду **Розрив** з меню **Вставка**, внаслідок чого на екрані з'являється однойменне діалогове вікно, за допомогою якого здійснюється бажаний поділ на сторінки. Цю операцію можна також виконати за допомогою клавіатури комбінацією клавіш **Ctrl+Enter**.

Робота зі сторінками документа здійснюється у режимі **Розмітка сторінки** за допомогою однойменної команди меню **Вид**.

Розглянемо найпоширеніші елементи форматування сторінок документа, що задаються за допомогою діалогового вікна **Параметри сторінки** з вкладками **Поля**, **Розмір паперу**, **Джерело паперу**, **Макет**, які з'являються під час виконання однойменних команд з меню **Файл**.

Розмір сторінки вибирається на вкладці **Розмір паперу** з однойменного списку або встановлюється за допомогою регуляторів **Ширина** і **Висота**, якщо в списку вибрати розмір **Інший** (рис. 5.6, *а*). Тут же вибирається орієнтація сторінки: книжкова або альбомна.

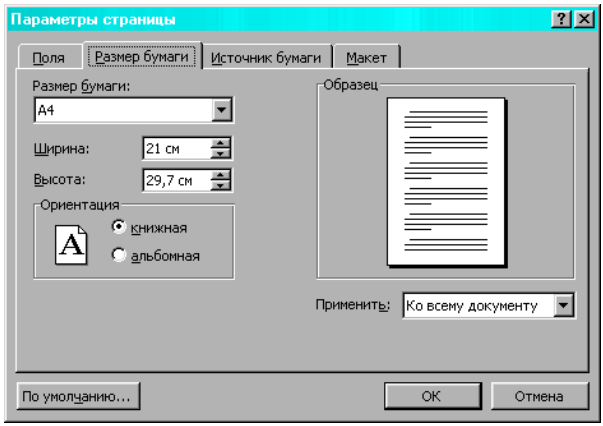


Рис. 5.6, *а*

Розміри полів встановлюються за допомогою відповідних регуляторів на вкладці **Поля** (рис. 5.6, *б*).

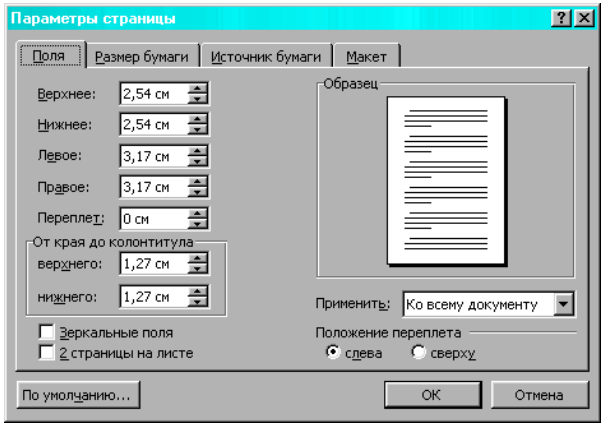


Рис. 5.6, *б*

Змінювати поля на сторінці можна легко і наочно, переміщуючи індикатори полів, що знаходяться на лінійці форматування. Границі тексту можна зробити видимими, якщо встановити позначку **Границі області тексту** на вкладці **Вид** діалогового вікна **Параметри** з меню **Сервіс**. Зовнішній вигляд сторінок тексту стає набагато привабливішим і інформативним, якщо на сторінці є ***верхні*** і ***нижні колонтитули*** — написи, відповідно на верхньому і нижньому полях сторінки. Верхній колонтитул, це, як правило, назва розділу і підрозділу тексту, які можуть бути оформленими з рамками, орнаментами та іншими ефектами. Нижній колонтитул, це, як правило, номер сторінки. Колонтитули оформляються за допомогою діалогового вікна **Колонтитули**, яке з'являється під час виконання однойменної команди з меню **Вид** (рис. 5.7).

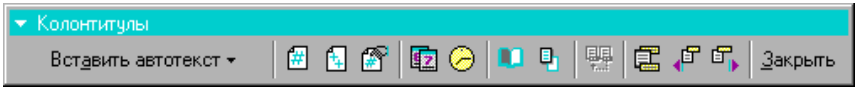


Рис. 5.7

Нумерацію сторінок можна включити в колонтитули, а можна виконати, за допомогою діалогового вікна **Номери сторінок**.

5.3 Таблиці

Таблиця — зручна і наочна форма представлення інформації, тому дуже широко застосовується у текстах, особливо економічного і соціального спрямування. Інформація у таблицях розміщується у ***клітинках***, які утворюють ***шпальти*** і ***рядки***. Шпальти прийнято позначати зліва направо буквами англійського алфавіту (A, B, C, ..., X, Y, Z), а рядки — зверху вниз цифрами (1, 2, 3, ...). У клітинках таблиці можуть розміщуватися дані будь-якого виду: числа, текст, графіка.

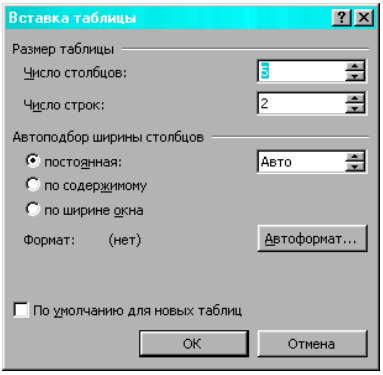


Рис. 5.8

Створити таблицю і вставити її в текст можна різними способами. Першим способом є застосування команди **Додати таблицю** з меню **Таблиця**. У діалоговому вікні, що з'являється на екрані (рис. 5.8),

можна вибрати кількість шпальт і рядків за допомогою відповідних полів. Так само можна вибрати ширину шпальти.

Можна також створити таблицю за допомогою кнопки **Додати таблицю.** Діалогове вікно, що з'являється під час виконання цієї команди, містить трафарет таблиці розміром 4x5 клітинок (рис. 5.9). Пересуваючи вказівник миші праворуч і вниз, встановлюємо потрібну кількість шпальт і рядків.

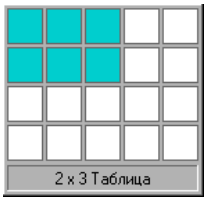


Рис. 5.9

У меню **Таблиця** є команда **Нарисувати таблицю,** що створює таблицю переміщенням вказівника миші, який набуває вигляду олівця, вниз і праворуч до отримання потрібного розміру.

Таблицю можна створити також на основі вже існуючого тексту, для чого слід виконати такі операції:

- виконати поділ тексту на шпальти за допомогою символу абзацу (Enter), табуляції, крапки з комою або будь-якого символу вибраного користувачем;
- виділити перетворений текст і виконати команду **Таблиця-Перетворити у таблицю.** У діалоговому вікні, що з'явиться, задати потрібну кількість шпальт і рядків.

Введення даних у таблицю. Дані вводяться у клітинки таблиці, починаючи з позиції курсора. По мірі заповнення клітинки її розміри по вертикалі автоматично збільшуються.

Переміщення між клітинками таблиці здійснюється за допомогою клавіатури такими комбінаціями клавіш:

- **Tab, Shift+Tab** — перехід до наступної та попередньої клітинки;
- **Alt+Home, Alt+End** — перехід до першої та останньої клітинки рядка;
- **Alt+PgUp, Alt+PgDn** — перехід до першої та останньої клітинки шпальти.

Операції зі шпальтами і рядками таблиці. Як і під час роботи зі звичайним текстом, перед тим, як застосувати операцію до певного об'єкта чи групи об'єктів таблиці, їх необхідно виділити.

Об'єднання клітинок. Виділити групу клітинок і виконати команду **Об'єднати клітинки** з меню **Таблиця.**

Поділ клітинок. Для поділу клітинки на кілька у рядку використовується команда **Розділити клітинки** з меню **Таблиця.** У діалоговому вікні, що відповідає цій команді, слід у полі **Кількість шпальт** вказати, на яку кількість клітинок розділити вибрану клітинку.

Збільшення кількості рядків і шпальт. Щоб додати новий рядок у таблицю, потрібно виділити один із рядків, а потім виконати

команду **Додати рядок** з меню **Таблиця**. Аналогічно виконується команда **Додати шпальту**.

Вилучення клітинок таблиці. Попередньо вибрані клітинки таблиці вилучаються командою **Вилучити клітинки** з меню **Таблиця**. У діалоговому вікні цієї команди слід вибрати одну з чотирьох операцій:

- **із зсувом ліворуч** — вибрані клітинки вилучаються, а на їх місце зсуваються клітинки, що розміщені справа;
- **вилучити весь рядок** — вилучається рядок, що містив вибрані клітинки;
- **із зсувом угору** — здійснюється зсув клітинок угору. Вибрані клітинки вилучаються, а на їх місце зсуваються клітинки, що розміщені внизу;
- **вилучити всю шпальту** — вилучається шпальта, що містила вибрані клітинки.

Нумерація клітинок. Щоб пронумерувати виділені клітинки таблиці, слід натиснути на кнопку **Нумерований список** на панелі інструментів. Якщо вносити зміни в нумеровану частину таблиці, то номери клітинок автоматично коригуються.

Сортування. Щоб виконати сортування в таблиці, слід помістити курсор у довільну клітинку і вибрати команду **Сортування** з меню **Таблиця**. Відкривається діалогове вікно **Сортування**, за допомогою якого можна виконати сортування у кількох шпальтах, але не більше ніж у трьох (рис. 5.10).

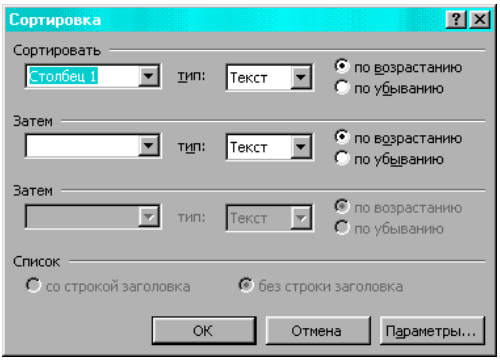


Рис. 5.10

У полі **Сортувати** слід вказати шпальту, в якій сортування має здійснюватися в першу чергу. У полях **Потім** вказати, якщо потрібно, решту шпальт, у яких виконуватиметься сортування. Для кожної вказаної шпальти потрібно вказати тип інформації, який вказується у полі **Тип**. Можна вибрати один з таких типів: **текст, число, дата**. Напрям сортування вибирається за допомогою параметра **За зростанням** і **За спаданням**.

Форматування таблиць. Форматування тексту всередині таблиці здійснюється тими самими засобами, що й звичайного тексту.

Під час створення таблиці або під час роботи з вже існуючими можна скористатися автоматичним форматуванням (команда

Автоформат таблиці з меню **Таблиця**). У діалоговому вікні цієї команди можна вибрати один із стилів рамок таблиці, а також скористатися готовими розробками з оформлення клітинок таблиці.

Керування шириною і висотою клітинок таблиці. Ширину шпальти або висоту рядка зручно змінювати безпосередньо в таблиці за допомогою миші. Якщо підвести вказівник миші до лінії, що розділяє шпальти чи рядки, то курсор змінює свій зовнішній вигляд на дві паралельні лінії. Клацнувши в цей момент лівою клавішею миші й утримуючи її в натиснутому стані, можна перемістити лінію розмежування у потрібну позицію, формуючи тим самим висоту рядка чи ширину шпальти.

За допомогою миші можна змінювати розміри клітинок, користуючись вертикальною і горизонтальною лінійками форматування, на яких розміщено спеціальні елементи. Переміщуючи мишею ці елементи на лінійці форматування, можна легко і наочно змінювати висоту і ширину клітинок таблиці.

Ще один спосіб змінити ширину і висоту клітинок: застосувати команди **Властивості таблиці** з меню **Таблиця**. Діалогове вікно цієї команди (рис. 5.11) містить дві вкладки: **Рядок** і **Шпальта**.

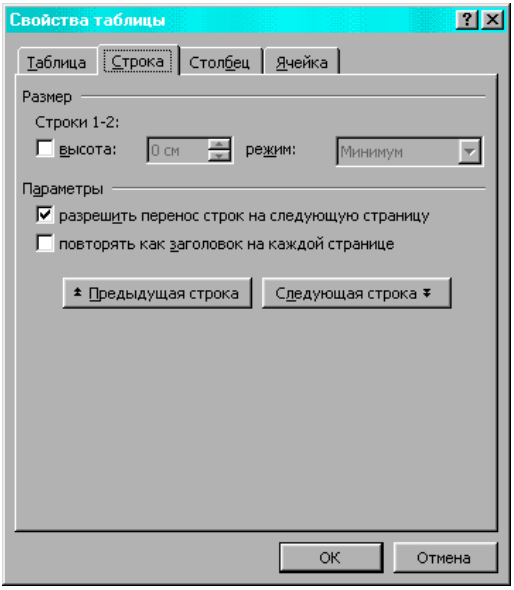


Рис. 5.11

Висота рядка встановлюється в полі **Висота рядка** N. Замість N стоїть число, що вказує номер рядка, висоту якого потрібно змінити. Перехід між рядками здійснюється за допомогою кнопок **Попередній рядок** і **Наступний рядок**. Щоб задати однакові параметри одночасно для кількох рядків, перед активізацією вікна слід виділити ці рядки.

У полі **Відступ зліва** вказується відстань від лівого краю тексту в рядку до лівого поля сторінки.

Параметр **Вирівнювання** визначає спосіб вирівнювання рядка відносно полів сторінки: по лівому краю, по центру, по правому

краю. Слід розрізняти вирівнювання тексту всередині клітинок і вирівнювання рядків таблиці відносно полів сторінки.

Параметри шпальт таблиці встановлюються за допомогою вкладки **Шпальта** (рис. 5.12). Як і при форматуванні рядків, якщо потрібно відформатувати зразу кілька шпальт, потрібно їх попередньо вибрати. Кнопки **Попередня шпальта** і **Наступна шпальта** служать для переходу між ними.

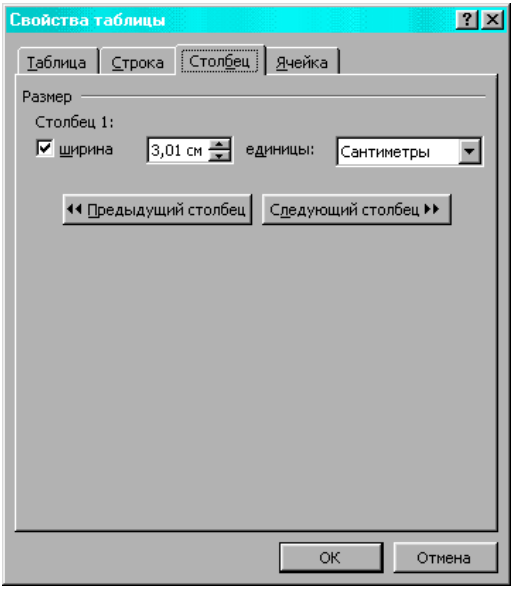


Рис. 5.12

Ширина шпальти (чи кількох) вказується у полі **Ширина шпальти**.

У полі **Інтервал між шпальтами** вказується відстань між границями шпальти та їх вмістом. Кнопка **Автодобір** дає змогу автоматично змінювати ширину згідно з вмістом.

Спрямування тексту. Текст всередині клітинок таблиці можна розмістити не тільки горизонтально, а й вертикально за допомогою команди **Спрямування тексту** з меню **Формат**. У діалоговому вікні цієї команди (рис. 5.13) можна вибрати бажаний напрям тексту. Для зміни напряму тексту можна скористатися також кнопкою на панелі інструментів **Таблиці та границі**.

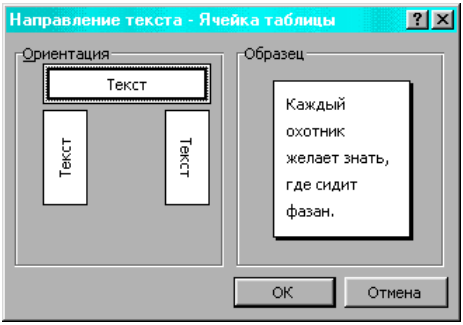


Рис. 5.13

Заголовки шпальт таблиці. Деякі таблиці не вміщуються на одній сторінці, тому їх доводиться розміщувати на кількох сторінках. Сприйняття документа поліпшується, якщо заголовки довгих таблиць повторюються з переходом на нову сторінку. Щоб задати автоматичне повторення заголовка, призначена команда **Заголовки** з меню **Таблиця**. Перед виконанням цієї команди слід

вибрати рядок чи кілька рядків, які використовуватимуться як заголовки.

Перетворення таблиці у текст. Інколи виникає необхідність перетворити таблицю у текст, для цього потрібно виділити таблицю, а потім вибрати команду **Перетворити у текст** з меню **Таблиця**. У діалоговому вікні цієї команди потрібно вказати, які символи використовувати як роздільники клітинок: маркери абзаців, символи табуляції, коми тощо.

5.4 Формули у тексті

Багато текстів, особливо науково-технічних, містять символи, яких немає не тільки на клавіатурі, а й на вкладках діалогового вікна **Символ**, яке викликається однойменною командою з меню **Вставка**. До таких складних символів належать, перш за все, математичні символи. Для введення таких символів у текст призначена спеціальна програма — **Редактор формул Microsoft Equation**.

Щоб вставити в текст математичну формулу, рівняння чи вираз, починаючи з позиції курсору, слід скористатися командою **Об'єкт** з меню **Вставка**. У діалоговому вікні **Вставка об'єкта** вибирається пункт **Microsoft Equation** зі списку на вкладці **Створити** і натискається кнопка **ОК** (рис. 5.14).

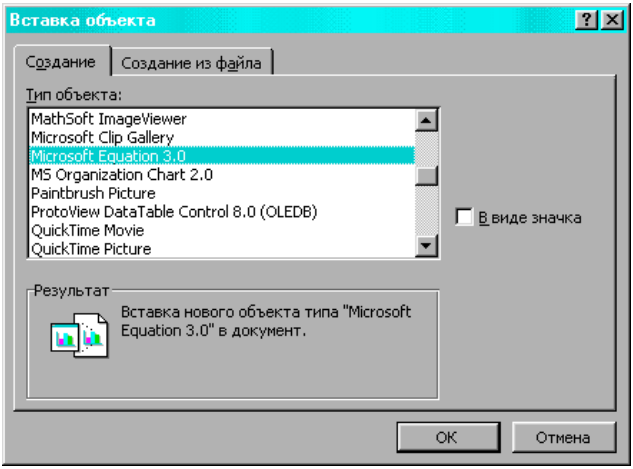


Рис. 5.14

Аналогічну команду можна вибрати за допомогою кнопки **Редактор формул** на панелі інструментів.

Виконання цієї команди супроводжується зміною виду меню (**Microsoft Word** замінюється на **Microsoft Equation**), виводом на екран нової панелі інструментів **Формула** і поля для введення математичних символів (рис. 5.15).

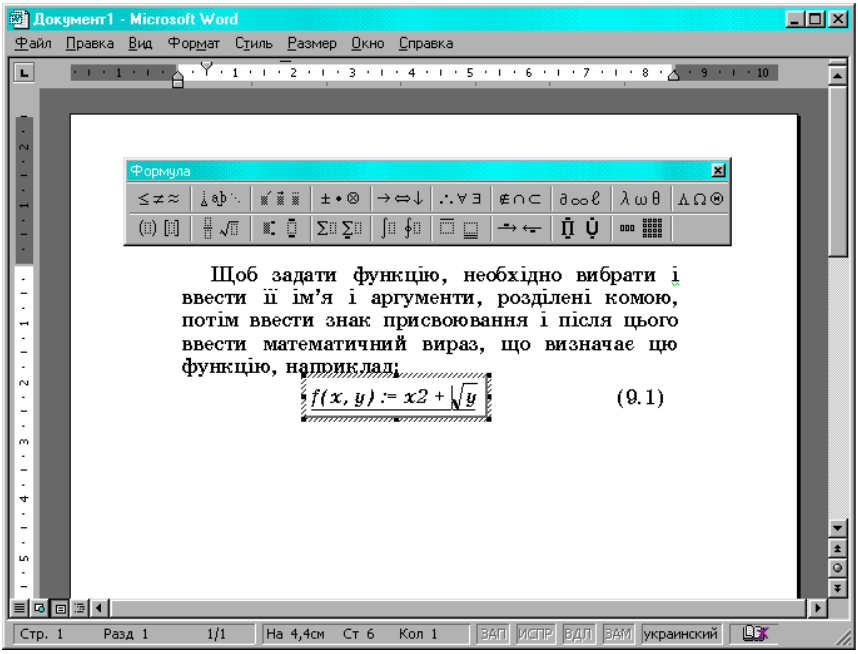


Рис. 5.15

У верхньому рядку панелі інструментів **Формула** розміщено понад 150 символів, згрупованих за певною ознакою в палітри. У нижньому рядку розміщено біля 120 шаблонів для складних математичних виразів, згрупованих у палітри. Щоб вставити символ чи шаблон, необхідно натиснути відповідну кнопку і в палітрі, яка відкриється, вибрати потрібний об'єкт. Щоб вийти з **Редактора формул**, слід натиснути клавішу **Esc** або клацнути мишею поза полем формули.

Редагування формул. Перед редагуванням формулу спочатку виділяють, клацнувши на ній лівою клавішею миші. Потім за командою **Правка–Об'єкт Equation-Змінити** або подвійним клацанням лівою клавішею миші на формулі активізується діалогове вікно **Формула** і здійснюється редагування. Для цього слід скористатися командою **Правка-Об'єкт Equation-Відкрити**. Розглянемо окремі операції редагування, які можна виконати у цьому вікні.

Зміна масштабу зображення виконується за допомогою команди **Масштаб** з меню **Вид**, яка відкриває однойменне діалогове вікно, де і встановлюється масштаб формули.

Інтервали між символами у формулах встановлюються автоматично, але можуть бути змінені користувачем у діалоговому вікні **Інтервал**, яке з'являється за однойменною командою з меню **Формат** (рис. 5.16). Результати внесених змін можна побачити на спеціальному транспаранті, якщо клацнути на кнопці **Застосувати**. Щоб відновити стандартні відстані, слід скористатися кнопкою **За замовчанням**. Позиціями окремих символів зручно керувати вручну. Наприклад, комбінація клавіш **Ctrl+↑** призводить до переміщення символу або групи вибраних символів вгору, а комбінація **Ctrl+→**— вправо.

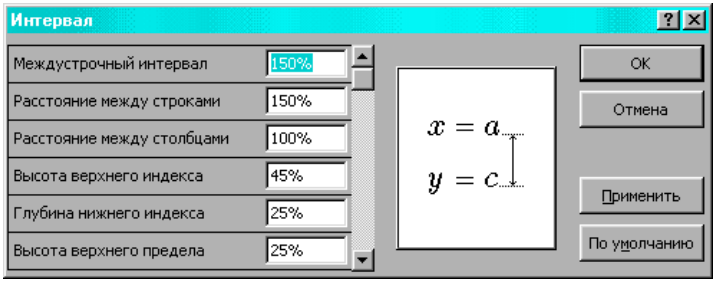


Рис. 5.16

Стандартні розміри символів, що вставляються у звичному режимі, можна змінити за допомогою команди **Визначити** з меню **Розмір**. У діалоговому вікні, яке з'являється під час виконання цієї команди (рис. 5.17), є поля введення, за допомогою яких встановлюються розміри символів верхніх і нижніх індексів. Функції кнопок **Застосувати** і **За замовчанням** такі самі, як і в попередньому вікні.

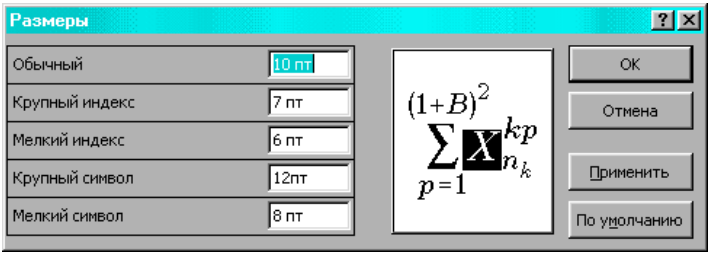


Рис. 5.17

Стилі форматування застосовуються для надання формулам наочності і гарного зовнішнього вигляду і вибирається з меню

Стиль. Якщо користувача не влаштовують запропоновані стандартні параметри стилів, то їх можна змінити за допомогою діалогового вікна **Стилі**, яке з'являється на екрані під час виконання вибраної команди **Визначити** з меню **Стиль** (рис. 5.18).

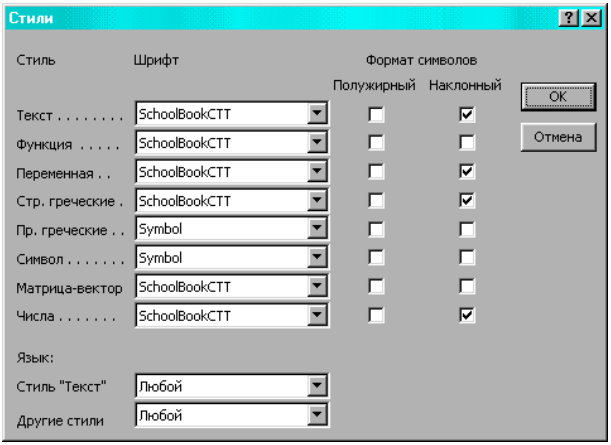


Рис. 5.18

За допомогою цього меню можна змінити шрифт (найбільш уживаними є Arial, Symbol, Times New Roman, SchoolBook, але користувач може вибрати будь-який інший), задати текстові курсивне або (і) напівжирне написання.

1. Схарактеризуйте структуру текстового документа?
2. Як вводиться текстова інформація в комп'ютер?
3. Як редагується текстова інформація в комп'ютері?
4. Що таке форматування тексту та як воно здійснюється?
5. Як форматуються абзаци?
6. Як здійснити форматування сторінок?
7. Як створити таблицю засобами текстового редактора?
8. За допомогою яких засобів вводяться і редагуються формули і яким чином це здійснюється?
9. Як за допомогою текстового редактора створити і відредагувати список?
10. Що таке колонтитули, і як їх створити засобами текстового редактора?

6

Обробка графічної інформації за допомогою комп'ютера

6.1 Основні поняття комп'ютерної графіки

Графічні зображення є важливою формою інформації. Комп'ютерна графіка — це створення і обробка зображень за допомогою комп'ютера. За способом створення розрізняють два види графіки: растрову і векторну.

Растрова графіка. Растрові зображення складаються із сукупності різнокольорових крапок (пікселів від англ. picel picture element — елемент зображення). Розмір визначається кількістю пікселів по горизонталі та вертикалі, наприклад, растрові зображення операційної системи Windows на екрані дисплея мають такі стандартні розміри: 640 × 480, 1024 × 768, 1240 × 1024.

Растрова графіка дає змогу точно передавати найменші деталі, тому растрові графічні зображення мають високу якість. Разом з тим, для збереження растрових зображень необхідний великий обсяг пам'яті. Наприклад, для збереження зображення на екрані дисплея системи з розмірами 1024 × 768, за умови, що колір кодується трьома байтами, необхідно 1024 × 768x3 = 2,3 Мбайт пам'яті. Крім того, досить складно змінювати масштаб растрового зображення і редагувати його.

Векторна графіка. Векторне графічне зображення складається з простих графічних елементів: прямих і кривих ліній, геометричних фігур (трикутників, прямокутників, кіл, еліпсів, дуг). Для побудови кожного такого елемента складається програма, в якій задаються параметри даного графічного об'єкта: довжина, товщина, орієнтація, колір лінії, радіус і координати центра кола тощо.

Перевагою векторних зображень порівняно з растровими є те, що обсяг пам'яті для збереження векторних зображень у десятки, сотні і навіть тисячі разів менший, ніж для таких самих растрових зображень.

Векторна графіка має особливі переваги при передачі штучних графічних об'єктів: технічних креслень, схем, графіків, діаграм тощо. Природні об'єкти мають, здебільшого, плавний перехід лі-

ній і кольорів, тому є певні труднощі зображення таких об'єктів засобами векторної графіки.

6.2 Графічний редактор і його призначення

Графічний редактор **CorelDRAW** є прикладною програмою Windows, він призначений для створення графічних зображень у векторному форматі. Документ, який створюється за допомогою **CorelDRAW**, називається ***ілюстрацією***. Крім того, **CorelDRAW** має засоби для створення і обробки растрових зображень і текстової інформації. **CorelDRAW**, як і інші прикладні програми, що працюють у середовищі Windows, має графічні засоби взаємодії (інтерфейс) з користувачем — ***вікна***.

Під час запуску програми на екрані дисплея з'являється вікно запрошення (рис. 6.1), на якому відображаються такі значки: **Новий документ** (New Graphic), **Відкрити останню редакцію** (Open Last Edited), **Відкрити Документ** (Open Graphic), **Шаблон** (Template), **Підручник** (CorelTUTOR), **Що нового?** (What's New?). Якщо клацнути на значку лівою клавішею миші, запускається відповідна програма.

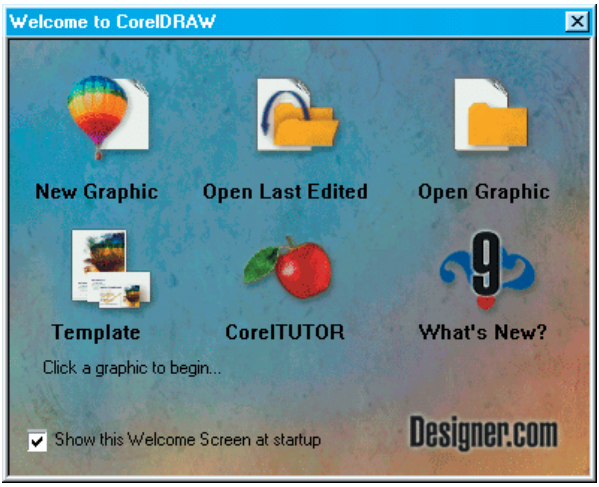


Рис. 6.1

Значок **Новий документ** запускає програму створення нового документа (ілюстрації). **Відкрити останню редакцію** відкриває останній документ, з яким працював користувач. **Відкрити Документ**

дає можливість відкрити будь-який документ, створений за допомогою **CorelDRAW**. За допомогою **Шаблону** можна створити документ на основі готового. **Підручник** (CorelTUTOR) запускає інтерактивний підручник **CorelDRAW**. **Що нового?** знайомить корис- тувача з новими можливостями даної версії **CorelDRAW**.

Основне вікно CorelDRAW (рис. 6.2). Верхню частину вікна займає рядок заголовка, в якому відображується ім'я прикладної програми **(CorelDRAW)**, ім'я відкритого активного документа, а також три стандартні кнопки керування вікном: згортання, розгортання і закриття. На початку рядка розміщена системна кнопка зі значком програми **CorelDRAW**. При натисканні на цю кнопку розкривається меню з командами керування вікном.

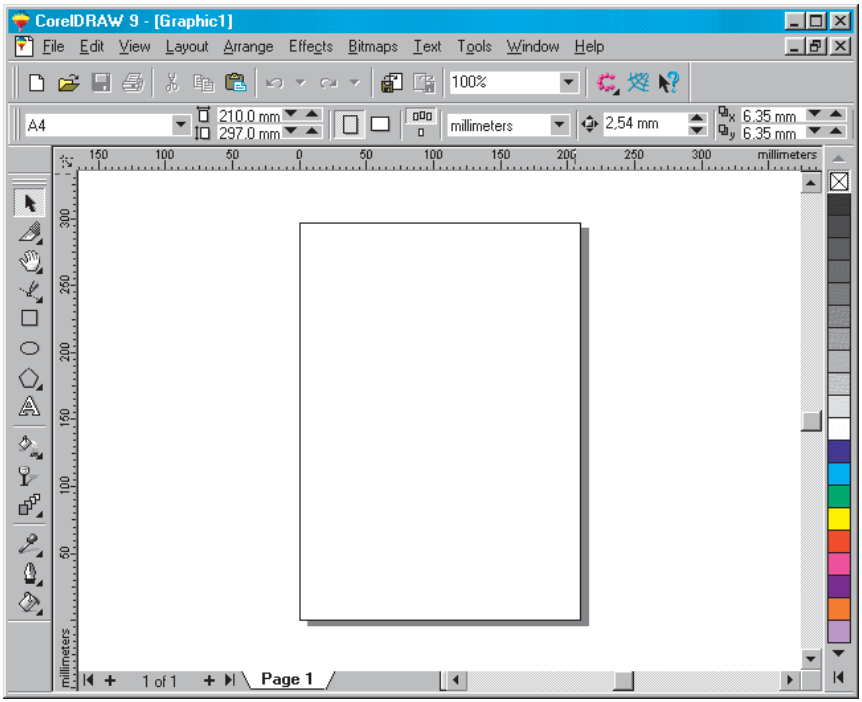


Рис. 6.2

Нижче рядка заголовка розташований рядок головного меню із заголовками додаткових меню: **Файл** (File), **Правка** (Edit), **Вид** (View), **Макет** (Layout), **Монтаж** (Arrange), **Ефекти** (Effects), **Растрові зображення** (Bitmaps), **Текст** (Text), **Сервіс** (Tools), **Вікно** (Window), **?** , що розкриваються, якщо клацнути на них лівою клавішею миші.

Під головним меню розміщена панель **Стандартна**, на якій знаходяться значки найбільш вживаних команд.

У лівій частині головного вікна знаходиться панель **Набір інструментів** (Toolbox), яка відкриває доступ до інструментів для створення, редагування, перетворення графічних об'єктів.

У правій частині головного вікна знаходиться палітра кольорів, за допомогою якої вибирається колір контуру графічного об'єкта і колір заливання. За замовчанням палітра подана одним рядком. За бажанням палітру кольорів можна розмістити в будь-якому місці вікна, наприклад, у правій його частині. Для перегляду елементів палітри ліворуч і праворуч є кнопки прокручування.

Щоб вибрати колір контуру певного графічного об'єкта, слід підвести вказівник миші до клітинки з бажаним кольором і клацнути правою клавішею миші. Колір заливання вибирається лівою клавішею миші. Для скасування вибраного кольору, слід клацнути відповідною клавішею на клітинці зі знаком ×.

У нижній частині головного вікна знаходиться ***рядок стану***, в якому відображається інформація про виділений графічний об'єкт або обрану команду меню. Зовнішній вигляд і параметри рядка стану можна змінювати за допомогою контекстного меню, яке відкривається, якщо клацнути по рядку стану правою клавішею миші.

Вікно документа. Велику частину головного вікна займає вікно документа. Документ у технічній документації **CorelDRAW** називається ***ілюстрацією***. Здебільшого вікно документа розгорнуте на все головне вікно, і тоді кнопки керування вікном документа: згортання, розгортання і закриття — розміщені у правій частині рядка головного меню.

Головним елементом вікна документа є сторінка документа, яка зображується на екрані монітора аркушем з тінню. У **CorelDRAW** документ може мати кілька сторінок, тоді внизу відображені язички цих сторінок документа. Якщо клацнути мишею по язичку сторінки, то вона виведеться на екран монітора. У лівій нижній частині вікна розташований лічильник сторінок документа і кнопки керування багатосторінковим документом. Кнопки + дають змогу додати до документа нові сторінки перед першою сторінкою або після останньої. Сторінки в документі можна гортати за допомогою кнопок зі стрілками на смузі прокручування.

Уздовж верхньої та лівої межі вікна документа знаходяться ***вимірювальні лінійки***, за допомогою яких легко визначати координати об'єкта на сторінці.

Вікно документа має також ***смуги прокручування*** для переміщення сторінки всередині вікна.

6.3 Графічні об'єкти, їх створення і редагування

CorelDRAW призначений для створення векторних графічних зображень, хоч має засоби для створення растрових зображень і тексту.

Векторні графічні об'єкти складаються з найпростіших елементів: прямих і кривих ліній, прямокутників, еліпсів, багатокутників.

Будь-який графічний об'єкт у **CorelDRAW** має такі дві основні характеристики (атрибути): контур **(абрис)** і геометрична фігура, обмежена контуром, колір якої можна задати, тобто здійснити **заливання** фігури. Заливання відсутнє у розімкнутих лініях.

Створення графічних об'єктів

Лінії. Прямі і криві лінії можна створити за допомогою інструментів, розміщених на панелі **Крива** (Curve Flyout) 

. Панель можна згорнути до розміру однієї кнопки, розміщеної на панелі інструментів. Щоб розгорнути панель, слід клацнути мишею на чорному трикутнику, розміщеному в правому нижньому куті кнопки. На розгорнутій панелі розміщено кнопки, якими можна активізувати такі інструменти:

- **Довільна крива** (Freehand Tool);
- **Крива Без'є** (Bezier Tool);
- **Товста крива** (Artistic Media Tool);
- **Розмірна лінія** (Dimension Tool);
- **З'єднувальна лінія** (Connector Tool);
- **Інтерактивна з'єднувальна лінія** (Interactive Connector Tool).

Крива складається з окремих частин — ***сегментів***. ***Вузол*** — точка на кривій, у якій з'єднуються два сегменти і в якій крива змінює свій напрям. Інструменти дають змогу рисувати прямі та криві лінії переміщенням миші (аналогічно тому, як це робиться олівцем на папері).

Щоб створити пряму лінію за допомогою інструмента **Довільна крива** або **Крива Без'є**, слід клацнути лівою клавішею миші в тому місці, де починається пряма, а потім у тому місці, де вона закінчується. Щоб створити ламану лінію, потрібно послідовно, одна за одною, створювати прямі лінії.

Криву лінію можна створити за допомогою тих самих інструментів, якщо клацнути лівою клавішею миші і, продовжуючи утримувати її в натиснутому стані, переміщувати вказівник по екрані. Якщо під час переміщення вказівника відпускати ліву клавішу миші, то на лінії створюватимуться вузли.

Якщо закінчити лінію у початковій точці, то отримаємо замкнену ламану або криву лінію.

Для завершення операції створення лінії слід натиснути на клавішу **Пробіл**.

Геометричні фігури CorelDRAW. Дозволяє користувачеві створювати прості геометричні фігури: прямокутники, квадрати, еліпси, кола, багатокутники, спіралі, сітки. Щоб створити потрібну фігуру, слід вибрати за допомогою кнопки відповідний інструмент на панелі інструментів. Розглянемо, наприклад, створення прямокутника. Для цього спершу слід клацнути лівою клавішею миші на кнопці **Прямокутник** панелі інструментів. Далі встановити вказівник миші у тому місці документа, де має бути дана фігура. Натиснути ліву кнопку миші й, утримуючи її у натиснутому стані, переміщувати вказівник, доки не утвориться фігура потрібного розміру, після чого відпустити клавішу. Якщо під час переміщення миші утримувати у натиснутому стані клавішу **Ctrl**, то утворюється квадрат.

Виділення графічних об'єктів

Виділення графічних об'єктів потрібне для подальшого їх редагування і перетворення. Щоб виділити потрібний графічний об'єкт, необхідно встановити режим виділення за допомогою кнопки **Вказівник** (Pick Tool). У режимі виділення вказівник миші перетворюється на чорну стрілку. Якщо підвести вказівник миші до графічного об'єкта і клацнути лівою клавішею, то об'єкт виділиться. Це проявляється в тому, що навколо виділеного об'єкта з'являються маленькі чорні квадрати (маркери) в кутках прямокутника і на його сторонах. У центрі виділеного об'єкта з'являється косий чорний хрест (маркер переміщення).

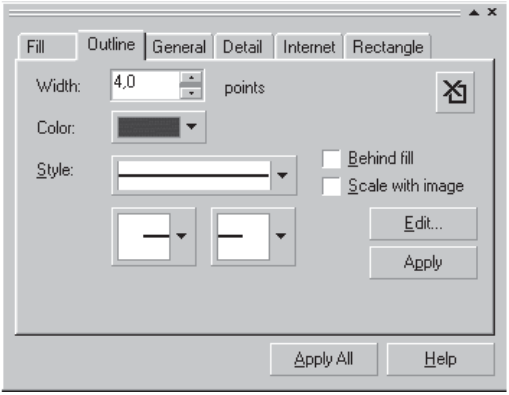


Рис. 6.3

Параметри виділеного об'єкта з'являються на панелі **Властивості** (Property) (рис. 6.3).

Щоб виділити групу об'єктів, слід у режимі виділення клацнути мишею й, утримуючи ліву клавішу миші у натиснутому стані, переміщувати вказівник миші. Під час переміщення з'являється штриховий контур прямокутника синього кольору, розміри якого змінюються відповідно до переміщення. Змінюючи розміри

прямокутника, слід пересвідчитися, що об'єкти, які потрібно виділити, потрапили всередину контуру і тоді відпустити клавішу. Навколо виділеної групи об'єктів з'являться такі самі маркери, як і навколо одного виділеного об'єкта.

Редагування графічних об'єктів

Редагування графічних об'єктів здійснюється за допомогою інструментів, розміщених на панелі **Редагування** (Shape Edit Flyout)  . Цю панель можна згорнути і записати на панелі інструментів однією кнопкою з маленьким чорним трикутником у нижньому правому куті кнопки. Щоб розгорнути кнопку в панель, слід натиснути на цей трикутник. На панелі розміщуються кнопки, якими можна викликати такі інструменти:

- **Форма** (Shape);
- **Ніж** (Knife);
- **Ластик** (Eraser);
- **Вільне перетворення** (Free Transform).

Форму ліній і фігур можна змінити за допомогою інструмента **Форма** (Shape Tool), який активізується, якщо натиснути на відповідну кнопку панелі інструментів. Вказівник миші в цьому режимі стає чорною стрілкою.

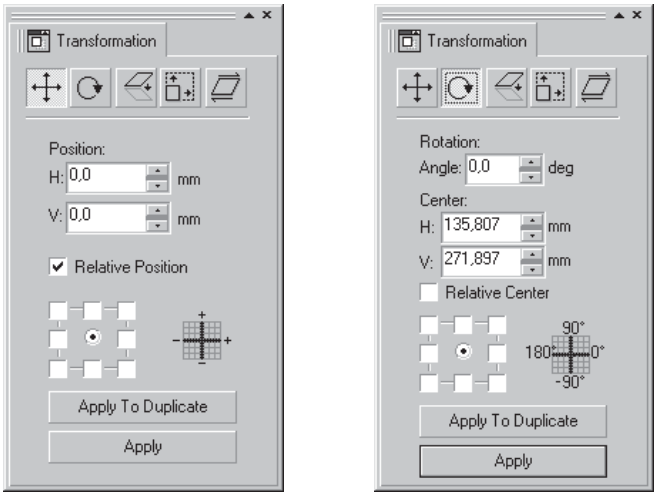
Якщо виділити потрібний графічний об'єкт за допомогою миші у режимі **Форма**, вузли об'єкта відображаються у вигляді маленьких кружечків. Підвівши вказівник миші до потрібного вузла, побачимо, що поряд із вказівником з'являється маленький чорний хрестик зі стрілочками на кінцях, а сам вузол змінює форму на маленький світлий квадратик. Якщо в цей час клацнути лівою клавішею миші, то вузол виділиться. Світлий квадратик зміниться на чорний. Крім того, з'явиться штрихова лінія, дотична до сегмента у даній точці. Утримуючи ліву клавішу миші у натиснутому стані, можна переміщувати вузол на потрібну відстань у потрібному напрямі. Кривизну виділеного сегмента кривої і його розмір можна змінювати за допомогою дотичної штрихової лінії, змінюючи її довжину і орієнтацію.

6.4 Перетворення графічних об'єктів

Однією з найважливіших переваг векторної графіки є можливість перетворення створюваних графічних об'єктів. **CorelDRAW** має широкий набір інструментів для перетворення графічних об'єктів, які зібрані на панелі **Перетворення** (Transformation)

(рис. 6.4, *а*). На цій панелі містяться такі інструменти для перетворення графічних об'єктів (верхні п'ять кнопок):

- **Положення** (Position);
- **Обертання** (Rotation) (рис. 6.4, *б*);
- **Масштаб** (Scale) і **Віддзеркалення** (Mirror);
- **Розмір** (Size);
- **Скіс** (Skew).



а) *б*)

Рис. 6.4

Перетворення графічних об'єктів у **CorelDRAW** здійснюються ***інтерактивно***, тобто користувач бачить на екрані результати своїх дій і може їх коригувати. Об'єкти на екрані монітора в процесі перетворень набувають яскравого синього кольору.

Зміна положення графічних об'єктів

Перш ніж змінювати положення графічних об'єктів, їх необхідно виділити за допомогою інструмента **Вказівник** (Pick Tool) на панелі інструментів (рис. 6.2).

Змінити положення графічного об'єкта можна за допомогою таких засобів:

- миші;
- панелі **Властивості** (Property);
- панелі **Перетворення** (Transformation).

Об'єкт, виділений за допомогою інструмента **Вказівник** (Pick Tool), обмежений прямокутником з чотирма маленькими чорними квадратами у вершинах і чотирма на сторонах цього прямокутника. Крім того, в центрі прямокутника є косий хрест. Якщо підвести

до нього вказівник миші, то вказівник змінить форму на прямий хрест зі стрілками на кінцях. Якщо в цей час клацнути лівою клавішею миші й, утримуючи її у натиснутому стані, переміщувати, то можна перемістити виділений об'єкт на потрібну відстань у потрібному напрямі. Відпустивши клавішу миші, отримаємо графічний об'єкт на новому місці. Під час переміщення об'єкта на екрані видно прямокутник синього кольору, що обмежує об'єкт.

Більш точно можна переміщувати графічний об'єкт за допомогою полів введення **X** і **У** панелі **Властивості**. Ввівши у поля **X** і **У** потрібні координати, слід натиснути клавішу **Enter** або клацнути мишею у будь-якому місці документа поза виділеним об'єктом. Об'єкт переміститься у положення, задане введеними координатами.

Аналогічним чином можна перемістити об'єкт за допомогою інструмента **Переміщення** (Move) панелі **Перетворення** (рис. 6.4, *а*). Координати виділеного за допомогою інструмента **Вказівник** об'єкта відображуються у полях **H** і **V** панелі **Перетворення**. Якщо у поля **H** і **V** ввести нові координати, то після завершення операції введення за допомогою кнопки **Застосувати** (Apply), об'єкт переміститься у позицію, задану новими координатами. Координати об'єкта можна задавати відносно будь-якої вершини або середини сторони робочого аркуша, а також відносно центра аркуша, для чого потрібно поставити прапорець у відповідному місці на панелі **Перетворення**.

Обертання графічних об'єктів

Щоб виділити певний об'єкт для обертання, слід двічі клацнути на ньому інструментом **Вказівник**, після чого у вершинах прямокутника навколо об'єкта з'являються ***мітки повороту***, а на сторонах прямокутника ***мітки скосу*** у вигляді двосторонніх стрілок. Крім того, з'являється ***центр обертання*** у вигляді маленького кола з крапкою посередині. Центр обертання можна за допомогою миші перемістити в будь-яку точку. Захопивши мишею потрібну мітку повороту й утримуючи кнопку миші в натиснутому стані, повертають об'єкт навколо центра обертання, доки він не займе бажаного положення.

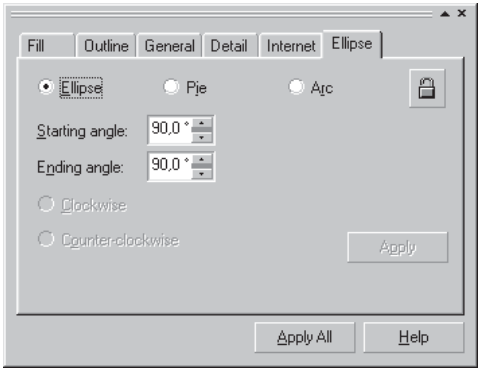


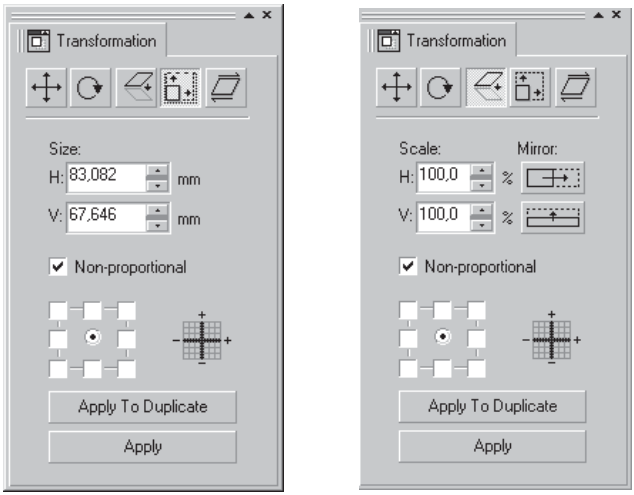
Рис. 6.5

Також це можна зробити за допомогою панелі **Властивості**, ввівши в поле **Кут повороту** значення кута повороту (рис. 6.5).

Ще одним способом обертання графічних об'єктів є обертання за допомогою однойменного інструмента панелі **Перетворення**. Ввівши значення кута повороту (в градусах) у поле **Кут**, отримаємо бажаний поворот, вибраний інструментом **Вказівник** графічного об'єкта. Позицію центра обертання можна змінити, якщо ввести в поля введення **H**(orizontal) **Горизонтально** і **V**(ertical) **Вертикально** відповідні координати.

Зміна масштабу графічних об'єктів

Щоб змінити розміри об'єкта в певну кількість разів відносно існуючих розмірів, застосовують інструмент **Масштаб** (рис. 6.6, *а*). Масштабування може здійснюватися як з дотриманням пропорцій об'єкта (при знятому прапорці **Без пропорцій**), так і без дотримання. Перетворення за замовчанням виконується відносно центра об'єкта. Щоб здійснити перетворення об'єкта відносно інших маркерів, слід встановити відповідний прапорець на панелі **Перетворення**. Масштаб перетворення здійснюється введенням відповідних значень у поля **Горизонтально** і **Вертикально**. Значення коефіцієнта масштабування 100 % не змінює розмірів об'єкта, значення 200 % збільшить розміри вдвічі, а значення 50 % зменшить розміри вдвічі.



а) *б*)

Рис. 6.6

Віддзеркалення об'єкта

Віддзеркалення об'єкта — отримання дзеркального відбитка об'єкта відносно горизонтальної або вертикальної лінії.

Віддзеркалення об'єктів можна здійснити за допомогою таких засобів: миші, панелі **Властивості**, інструмента **Дзеркало** (рис. 6.6, *б*) на панелі **Перетворення**.

Щоб отримати дзеркальний відбиток об'єкта за допомогою миші, необхідно виділити об'єкт інструментом **Вказівник**, захопити мишею мітку на боковій стороні і перемістити її при натиснутій клавіші на протилежну сторону обмежувального прямокутника. Пересвідчившись, що об'єкт, зображений синьою лінією, віддзеркалився, відпустити кнопку миші та клавішу **Ctrl**.

Для здійснення віддзеркалення за допомогою панелей **Властивості** або **Перетворення** слід скористатися відповідними кнопками, які є на цих панелях.

Зміна розміру графічних об'єктів

Розмір графічного об'єкта можна змінювати за допомогою таких засобів: миші, панелі **Властивості**, інструмента **Розмір** (рис. 6.6, *а*) панелі **Перетворення**.

Щоб змінити розміри об'єкта за допомогою миші, необхідно виділити об'єкт інструментом **Вказівник**, захопити мишею мітку у вершині обмежувального прямокутника і переміщувати її на потрібну відстань. Пересвідчившись, що об'єкт набув потрібного розміру, відпустити клавішу миші. Якщо захопити мишею бокову мітку і переміщувати її, то об'єкт розтягуватиметься по горизонталі.

Розміри виділеного об'єкта можна змінити, якщо ввести нові значення у поля на панелі **Властивості**.

Щоб змінити розміри виділеного об'єкта за допомогою інструмента панелі, потрібно ввести необхідні розміри у поля **Горизонтально** і **Вертикально** безпосередньо, або скористатися лічильниками у правій частині полів.

Вибраний об'єкт можна розтягнути по горизонталі або вертикалі, якщо встановити прапорець **Не пропорційно** на панелі **Перетворення**. Якщо зняти цей прапорець, то зміна одного розміру призводить до пропорційної зміни іншого розміру.

Скіс графічних об'єктів

CorelDRAW має засоби для здійснення такого перетворення, як скіс графічних об'єктів по горизонталі і вертикалі.

Скіс об'єктів може здійснюватися за допомогою миші, панелі **Властивості**, інструмента **Скіс** панелі **Перетворення**.

Щоб здійснити скіс графічного об'єкта за допомогою миші, слід двічі клацнути лівою клавішею миші на потрібному об'єкті в режимі **Вказівник**, в результаті чого у вершинах обмежувального прямокутника з'являються мітки повороту, а на сторонах цього

прямокутника — мітки скосу у вигляді двобічних стрілок. Далі слід захопити мишею мітку скосу й, утримуючи клавішу в натиснутому стані, переміщувати її на потрібну відстань. Графічний об'єкт на екрані дисплея під час переміщення стане синього кольору і змінюватиметься. Пересвідчившись, що об'єкт набрав потрібного вигляду, слід відпустити клавішу миші.

Скіс вибраного об'єкта здійснюватиметься в покроковому режимі з кроком у 15°, якщо під час переміщення миші утримувати у натиснутому стані клавішу **Ctrl**.

Щоб здійснити скіс об'єкта з більшою точністю, застосовують інструмент **Скіс** панелі **Перетворення**. Спочатку необхідно виділити об'єкт інструментом **Вказівник**. Далі слід задати центр перетворення, поставивши мітку у відповідному квадратику на панелі. Ввести у поля **Горизонтально** і **Вертикально** значення скосу в градусах з клавіатури або за допомогою лічильників у правій частині полів. Для завершення перетворення слід натиснути кнопку **Застосувати** на панелі.

1. Назвіть види графіки за способом створення?
2. З яких основних частин складається вікно **CorelDRAW**?
3. Які види ліній можна створити у **CorelDRAW** і за допомогою яких інструментів?
4. Як створюються геометричні фігури в **CorelDRAW**?
5. Опишіть процедуру виділення графічних об'єктів, її призначення?
6. Як здійснюється редагування графічних об'єктів?
7. Для чого призначена панель **Набір інструментів**?
8. Які перетворення графічних об'єктів можна здійснити у **CorelDRAW**?
9. За допомогою яких засобів і яким чином можна змінити положення графічних об'єктів?
10. Як здійснити обертання графічних об'єктів?
11. Як можна змінити розмір і масштаб графічних об'єктів?

7

Обробка інформації за допомогою електронних таблиць. Програма Microsoft Excel

7.1 Електронні таблиці

Інформацію у певних предметних областях, перш за все в економічній, фінансовій, соціальній, доцільно подавати у вигляді таблиць. Для обробки інформації таблиці застосовувалися задовго до виникнення комп'ютера, тому така форма обробки інформації звична для великої кількості людей.

Комп'ютери дали змогу створювати таблиці в електронному вигляді. Для обробки інформації у формі електронних таблиць призначені спеціальні прикладні програми. Однією з найпоширеніших програм є Microsoft Excel, вікно якої наведено на рисунку 7.1.

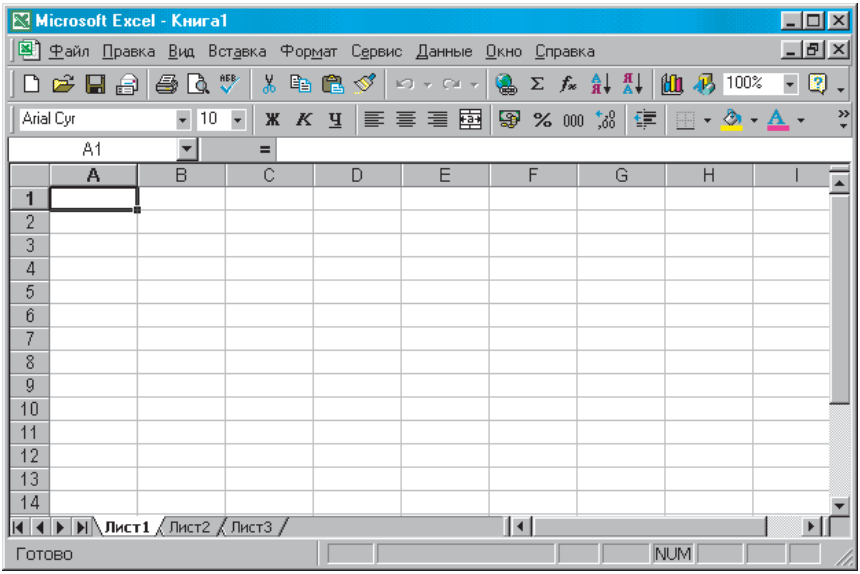


Рис. 7.1

Електронні таблиці можна зберігати у пам'яті комп'ютера і застосовувати для обробки інформації, що зберігається у цих таблицях, весь арсенал потужних засобів, які має комп'ютер.

Електронні таблиці дають змогу:

- ❒ подати інформацію у зручному і звичному вигляді;
- ❒ здійснювати просту і зручну адресацію інформації за координатами клітинок електронних таблиць;
- ❒ використовувати різні форми представлення числової і текстової інформації;
- ❒ здійснювати заміну даних в автоматичному режимі;
- ❒ форматувати інформацію, використовуючи різні види і стилі форматування;
- ❒ упорядковувати інформацію в таблицях за різними ознаками;
- ❒ фільтрувати інформацію, тобто вибирати інформацію з таблиць, яка задовольняє комплекс заданих критеріїв;
- ❒ сортувати інформацію за заданими критеріями;
- ❒ створювати нові таблиці на основі наявних;
- ❒ виконувати складні математичні обчислення, використовуючи потужний арсенал формул і функцій;
- ❒ відслідковувати й аналізувати зміну одних даних залежно від зміни інших;
- ❒ здійснювати статистичний аналіз даних;
- ❒ наочно відображати дані за допомогою діаграм і графіків різних видів;
- ❒ здійснювати автоматичне заповнення таблиць;
- ❒ використовувати захист інформації від несанкціонованого втручання;
- ❒ використовувати інформацію з інших прикладних програм.

7.2 Основні елементи електронних таблиць

Інформація в електронних таблицях аналогічно зі звичайними таблицями, зберігається в ***клітинках*** прямокутної таблиці. Клітинки таблиці утворюють ***рядки*** і ***стовпчики***. Рядки таблиці в Excel позначаються цифрами натурального ряду **1**, **2**, **...**, **65536**, а стовпчики — літерами латинського алфавіту **A**, **B**, **C**, **...**, **Z**, **AA**, **AB**, **...**, **AZ**, **BA**, **...**, **BZ**. Цифри і літери, якими позначаються рядки і стовпчики розміщені в окремих клітинках і називаються ***заголовками*** рядків і стовпчиків. Таким чином, кожна клітинка електронної таблиці має свою адресу, яка складається з позначення стовпчика і рядка, на перетині яких знаходиться дана клітинка, наприклад, адреса **BA45** вказує, що клітинка знаходиться на

перетині стовпчика **BA** і рядка **45.** Кожна електронна таблиця в Excel називається ***робочим аркушем***. Робочі аркуші мають ім'я, що розміщене на ярличку внизу аркуша. Сукупність робочих аркушів складає ***робочу книгу***. Кожна робоча книга зберігається як окремий файл під своїм унікальним іменем. Стандартний робочий аркуш має 65536 рядків і 256 стовпчиків, тому у вікні на екрані монітора видно лише його невелику частину. Щоб вивести на екран потрібну частину робочого аркуша, призначені горизонтальна і вертикальна смуги прокручування. Переміщуючи за допомогою миші повзунки на лінійках прокручування, виводять на екран і роблять видимими потрібну частину робочого аркуша.

Виділення об'єктів. Потрібну клітинку можна виділити, підвівши до неї курсор, що має форму білого хреста, і клацнувши лівою клавішею миші. Виділена клітинка називається ***активною*** і позначена товстою темною рамкою з квадратиком у нижньому правому куті, який називається **маркером заповнення.** Адреса активної клітинки виводиться у лівому полі рядка формул. Рядок і стовпчик, у яких знаходиться активна клітинка, також стають **активними.**

Щоб виділити кілька суміжних клітинок, тобто ***діапазон*** клітинок, достатньо провести курсором по цих клітинках при натиснутій лівій клавіші миші. Це ж саме можна зробити за допомогою комбінації клавіш **Shift** + ↑ або **Shift** + →

Виділити стовпчик чи рядок можна, клацнувши лівою клавішею миші на його імені.

Для виділення всього робочого аркуша досить клацнути мишею на кнопці **Виділити все**, розміщеній у верхньому правому куті робочого аркуша.

Зміна розмірів рядків і стовпчиків. Щоб розмістити всі символи, що вводяться у клітинку, і забезпечити гарний зовнішній вигляд, необхідно у багатьох випадках змінювати розміри клітинок, рядків, стовпчиків. Щоб змінити ширину стовпчика, потрібно курсор підвести до правої межі потрібного стовпчика. Тоді курсор змінює свій вигляд. Клацнувши лівою клавішею миші й утримуючи її у натиснутому стані, переміщують границю стовпчика на потрібну відстань. Аналогічну операцію можна виконати і з рядком таблиці.

Типи даних. За допомогою електронних таблиць може оброблятися інформація різного роду. Дані, з якими оперує Excel, можуть бути такого типу:

- ❒ числа, включаючи дату і час;
- ❒ текст;
- ❒ формула.

У свою чергу, числа можуть бути записані у різних формах: ціле, дійсне, дробове, в експоненціальній формі. Ціле число складається з цифр 0, 1, ..., 9 і знака + і −. Дійсне число складається з цілої і дробової частин, розділених десятковою комою. Число в експоненціальній формі складається з мантиси і порядку, розділених латинською літерою e або E, яка трактується як основа десяткової системи числення, тобто число 10. Мантиса — ціле, або дійсне число, а порядок — показник степеня.

Числа в **Excel** записують у таких форматах:

- ❒ загальний;
- ❒ числовий;
- ❒ фінансовий;
- ❒ грошовий;
- ❒ процентний;
- ❒ дробовий.

Формат числа задається в діалоговому вікні (рис. 7.2) командою **Клітинка** в меню **Формат.**

У загальному форматі можуть записуватися як числа, так і текстова інформація. Числа у форматі **Загальний** записуються так, як вводяться. Для чисел, записаних у числовому форматі, вказується кількість розрядів після десяткової коми.

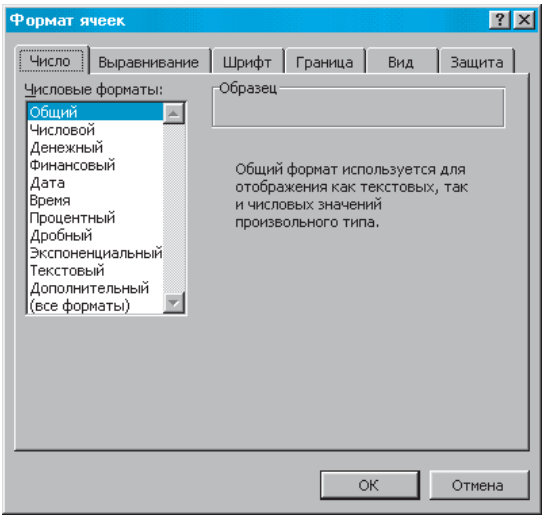


Рис. 7.2

У форматах **Фінансовий** і **Грошовий** поряд з числом вказана національна грошова одиниця, в **Процентний** числа подані у відсотках. **Дробовий** передбачає запис числа у вигляді простих дробів, десяткових дробів тощо. Вводиться текстова інформація у форматі **Загальний**.

Формула — це математичний вираз, що починається зі знака = і містить знаки арифметичних операцій +, −, *, / і назви функцій з аргументами. Формула вводиться у полі рядка формул.

Введення інформації. Щоб ввести інформацію в клітинку таблиці, потрібно цю клітинку виділити, тобто зробити активною.

Під час введення першого символу курсор змінює свою форму на тонку вертикальну риску, яка вказує на позицію, куди буде вводитися наступний символ. Символи можуть вводитися в режимі **Заміни** або **Вставки**. У першому випадку введений символ замінює той символ, у позиції якого знаходився курсор. У режимі **Вставка** символи праворуч від позиції курсору зсуваються і символ вводиться на звільнене місце. Для вилучення символу в позиції курсору слід натиснути клавішу **Del**, а символу зліва від курсору — клавішу **Back Space**. Відразу, після початку введення, у рядку формул з'являються дві кнопки: × — кнопка скасування і **v** — кнопка введення. Щоб завершити введення, слід натиснути клавішу **Enter** або кнопку **v**, а щоб скасувати введення — клавішу **Esc** або кнопку ×. Після завершення введення інформації у вибраній клітинці автоматично вибирається (активізується) клітинка, розміщена під даною. Щоб змінити напрям зсуву активізації клітинок, слід вибрати пункт **Параметри** з меню **Сервіс**, і на вкладці **Правка** вибрати потрібний напрям зсуву.

Щоб заповнити клітинки однаковою інформацією, потрібно ввести цю інформацію у першу клітинку діапазону. Потім підвести курсор до маркера заповнення (маленького темного квадратика у правому нижньому куті виділеної клітинки). Курсор змінить свій вигляд на маленький темний хрестик. Клацнувши лівою клавішею миші й утримуючи її у натиснутому стані, пересунути маркер заповнення у потрібному напрямку і на потрібну кількість клітин.

Щоб заповнити діапазон клітинок членами арифметичної прогресії, слід ввести у дві сусідні клітинки два перші члени цієї прогресії. Потім, виділивши мишею ці дві клітинки, захопити мишею маркер заповнення і перемістити рамку виділення на потрібний діапазон у потрібному напрямку.

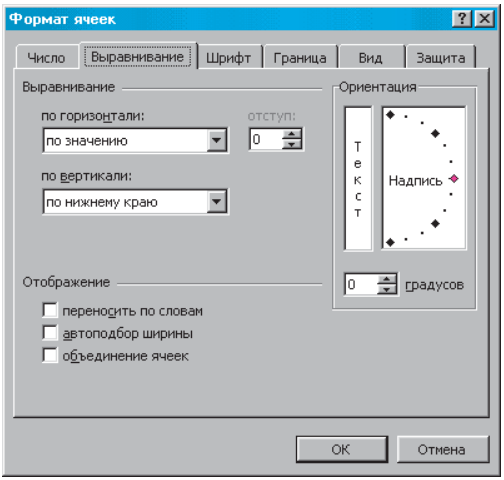


Рис. 7.3

Форматування інформації. Форматування інформації, введеної у клітинку електронної таблиці, здійснюється за допомогою команди **Клітинки** з меню **Формат** (рис. 7.3). Цю команду можна вибрати з контекстного меню,

яке з'являється, якщо клацнути правою клавішею миші на активній (виділеній) клітинці, або викликати комбінацією клавіш **Ctrl+1**. Діалогове вікно **Формат клітинок** цієї команди, містить вкладки **Число, Вирівнювання, Шрифт, Границя, Вигляд** і **Захист**.

На вкладці **Число** можна зі списку **Числові формати** вибрати потрібний формат. Слід зауважити, що формат **Загальний**, придатний як для форматування числової, так і текстової інформації.

На вкладці **Вирівнювання** у полях **По горизонталі** і **По вертикалі** можна задати вид вирівнювання і значення відступу. За допомогою транспаранта **Орієнтація** і регулятора **Градуси** можна задати орієнтацію написів під довільним кутом до горизонталі.

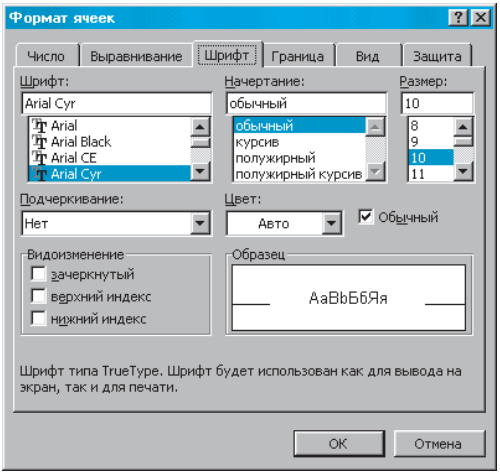


Рис. 7.4

Параметри шрифту: гарнітуру, стиль, розмір, колір тощо — можна задати на вкладці **Шрифт** (рис. 7.4).

Щоб текст чи числові дані мали привабливий вигляд і легко читалися, їх можна гарно оформити за допомогою рамок, вибравши їх зовнішній вигляд на вкладці **Границя**, і розмістити їх на кольоровому фоні з візерунком, який можна вибрати на вкладці **Вигляд**.

Щоб уберегти дані від несанкціонованих змін, призначена вкладка **Захист**.

7.3 Обробка інформації за допомогою формул

Для обробки інформації в Excel є потужний механізм формул. ***Формула*** — сукупність операндів, сполучених між собою знаками математичних операцій і дужками. Формальною ознакою формули є те, що першим символом, з якого починається формула, має бути символ =. Операндами формули можуть бути: числа, включаючи дату і час; текст; адреси клітинок таблиці, тобто ***посилання*** на клітинки, функції.

Математичні операції в Excel поділяються на:

- ❒ арифметичні: додавання (+), віднімання (−), множення (*), ділення (/), піднесення до степеня (^);
- ❒ логічні: більше (>), менше (<), дорівнює (=), не менше (=>), не більше (<=).

Посилання на клітинки. Адреса клітинки таблиці включає заголовок колонки (стовпчика) і номер рядка, і використовується у формулі як аргумент функції чи операнд. Таке використання адреси називається ***посиланням*** на клітинку і є дуже зручним механізмом обробки інформації. Розрізняють відносні, абсолютні і змішані посилання.

Відносні посилання характерні тим, що зі зміною адреси клітинки (наприклад, під час додавання або вилучення рядка чи стовпчика) автоматично змінюється і посилання у формулі, яка його використовує.

Абсолютні посилання не змінюються під час зміни адреси клітинки. Прикметою абсолютних посилань є знак $, який ставиться перед заголовком рядка чи стовпчика в адресі, наприклад, посилання **AA128** на клітинку з адресою **AA128** є абсолютним. Відсутність знака $ в адресі клітинки є прикметою відносних посилань.

У змішаних посиланнях абсолютною є адреса стовпчика, а відносною — адреса рядка, або навпаки (наприклад, **$AA128**, **AA$128**). Зі зміною адреси клітинки, на яку йде посилання, змінюється тільки відносна частина посилання, а абсолютна залишається без змін.

У формулі можуть використовуватися посилання на діапазон клітинок, наприклад, посилання AB345:AB355 вказує на десять сусідніх клітинок стовпчика AB з 345 до 355 рядка. Щоб адресувати прямокутний діапазон клітинок, у посиланнях зазначають адресу лівої верхньої і правої нижньої клітинки діапазону, наприклад, посилання AB45:AE51 адресує прямокутний діапазон клітинок шириною чотири клітинки і висотою шість клітинок.

Імена клітинок і діапазонів. У певних задачах зручно користуватися не адресою клітинки чи діапазону клітинок, а іменем. Excel дає змогу призначати імена виділеним клітинкам чи діапазонам. Для цього слід виконати команду **Вставка-Ім'я-Присвоїти**. На екрані з'явиться діалогове вікно **Присвоїти ім'я** (рис. 7.5), де можна набрати потрібне ім'я. Воно має починатися з букви, може мати будь-які букви і цифри, а також знак підкреслення, знак питання, крапку. Адреса виділеної клітинки чи діапазону з'явиться у полі **Формула** цього вікна. Для вилучення імені слід розкрити

список **Ім'я** діалогового вікна **Присвоїти ім'я**, виділити потрібне ім'я і натиснути на кнопку **Скасувати.**

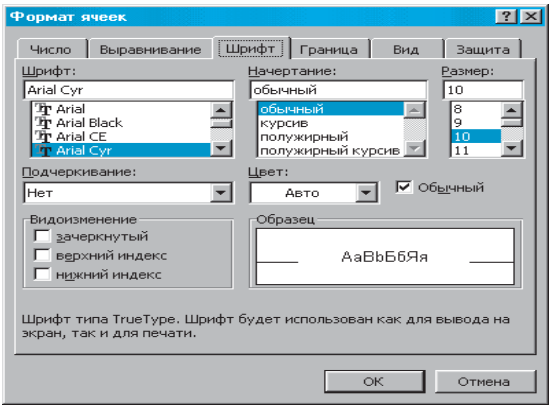


Рис. 7.5

Функції Excel. Вони призначені для виконання складних операцій обробки інформації в Excel. Кожній функції в Excel присвоюється унікальне ім'я. Після імені у круглих дужках вказуються аргументи (або аргумент), над якими здійснюється операція, яку позначає функція. Пробіл між іменем функції і круглими дужками не допускається. Значення, які повертаються функціями як відповіді, називаються ***результатами***.

Excel має понад 400 вбудованих функцій, поділених на категорії, відповідно до галузей знань (рис. 7.6).

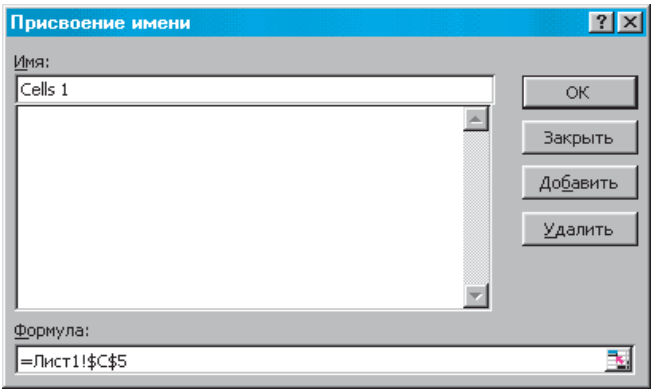


Рис. 7.6

Аргументами функцій можуть бути числові та текстові константи, посилання на клітинки і діапазони клітинок.

Ввести функцію у формулу можна вручну, за допомогою клавіатури або за допомогою спеціального програмного засобу — **Майстра функцій**. Майстер функцій викликається або за допомогою однойменної кнопки на панелі інструментів, або за допомогою команди **Функції** з меню **Вставка**. Виконання цієї команди здійснюється у два етапи (два кроки). Діалогове вікно, що виникає на екрані під час виконання цієї команди, має різний вигляд на кожному етапі. На першому етапі у вікні відкрито два списки: список категорій (різновидів) функцій і список самих функцій у вибраній категорії (рис. 7.6). Вибравши потрібну функцію, слід натиснути кнопку **Далі**, у результаті чого здійснюватиметься другий етап виконання команди і діалогове вікно набуде іншого вигляду (рис. 7.7).

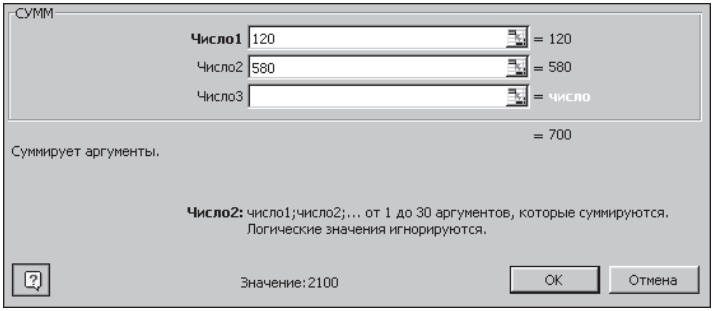


Рис. 7.7

У поле тепер потрібно ввести аргументи функції, тоді в полі **Значення** з'явиться обчислене значення функції. Після натискання на кнопку **Готова**, формула стає в активну клітинку. В полі клітинки може відображатися формула або значення. Вид відображення залежить від мітки **Формули** вкладки **Вигляд** діалогового вікна **Параметри** з меню **Сервіс**.

7.4 Зображення інформації за допомогою діаграм і графіків

Інформація, записана за допомогою діаграм і графіків, має більшу наочність і зручна для сприйняття. Діаграмами і графіками зручно ілюструвати функціональну залежність однієї величини від іншої, виявляти тенденції зміни певного параметра в часі, порівнювати характеристики різних об'єктів, виявляти взаємозв'язки різних величин і вплив певних чинників на протікання досліджуваних процесів.

Для відображення таких складних інформаційних взаємозв'язків Excel має понад 14 різноманітних типів діаграм, кожна з яких має кілька різновидів (рис. 7.8). Будь-який тип діаграми відображує залежність однієї чи кількох величин від іншої. Незалежні змінні (аргументи) називаються ***категоріями***, а залежні — ***значеннями***. Відповідно, горизонтальна вісь (вісь X) називається ***віссю категорій***, а вертикальна (вісь Y) — ***віссю значень***.

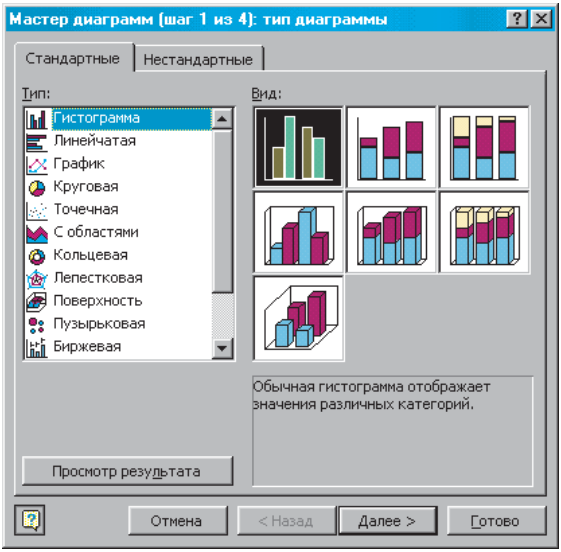


Рис. 7.8

Excel передбачає два варіанти відображення: «за рядками» і «за стовпчиками». У першому випадку заголовки рядків відображуються на осі категорій (осі X), а вміст клітинок певного стовпчика — на осі значень (осі Y), у другому ж — на осі категорій відображуються заголовки стовпчиків, а на осі значень — вміст клітинок певного рядка.

Тип діаграм. Для відображення залежностей в Excel можна будувати діаграми таких стандартних типів:

- ❒ гістограма;
- ❒ лінійна;
- ❒ графік;
- ❒ кругова;
- ❒ кільцева;
- ❒ точкова;
- ❒ пелюсткова;
- ❒ поверхнева та ін.

Складний процес побудови діаграм в Excel автоматизується за допомогою спеціальної програми — **Майстра діаграм.** Створення діаграм будь-якого типу починається з виділення діапазону даних електронної таблиці, які підлягають відображенню. Потім потрібно активізувати програму **Майстер діаграм** за командою **Вставка-Діаграма** або за допомогою кнопки **Майстер діаграм** панелі інструментів. Робота програми **Майстра діаграм** здійснюється за чотири етапи (кроки), причому на кожному етапі діалогове вікно цієї програми має різний вигляд.

Перший етап після запуску Майстра діаграм — етап вибору типу діаграми. Діалогове вікно, що виводиться на екран на цьому етапі, має дві вкладки: **Стандартні** й **Нестандартні** типи діаграм (рис. 7.8). На вкладці **Стандартні** в списку **Тип** користувачеві запропоновано вибрати один з таких типів. На транспаранті **Вид справа** наводиться зовнішній вигляд різновидів діаграм даного типу, який можна вибрати, клацнувши на ньому мишею. Натискуючи на кнопку **Далі** у нижній частині вікна, здійснюють перехід до наступного етапу побудови — вибору рядка і діапазону даних електронної таблиці, що будуть відображені на діаграмі. Вибравши діапазон і ряд даних, за допомогою кнопки **Далі** переходять до етапу встановлення параметрів діаграми в однойменному діалоговому вікні, що має шість вкладок: **Заголовки, Осі, Лінії сітки, Легенда** (назва), **Підписи даних, Таблиці даних.** Вибравши параметри діаграми, зрозумілі з назв вкладок, переходять до етапу розміщення діаграми. Діаграму можна розміщувати на одному з аркушів робочої книги або відкрити власний аркуш у робочій книзі.

1. Що таке електронна таблиця?
2. Для чого призначена програма Microsoft Excel і які її основні функції?
3. Як вводиться і редагується інформація в електронних таблицях?
4. Які основні елементи має електронна таблиця?
5. Які типи даних можуть міститися у клітинках електронних таблиць?
6. Як можна форматувати інформацію в клітинках електронної таблиці?

8

Система керування базами даних. Програма Microsoft Accsess

8.1 Основні поняття баз даних

У сучасному суспільстві в будь-якій сфері діяльності (промислове виробництво, наука, культура) необхідний потужний інформаційний супровід. Наприклад, сучасне промислове виробництво випускає складні вироби за складною технологією. Таке виробництво неможливе без розробки креслень виробу з усіма його найменшими деталями, картки технологічних процесів та іншої технологічної документації. Для зберігання таких великих інформаційних потоків необхідні відповідні засоби.

Сукупність об'єктів реального світу, що мають певні характеристики і об'єднані за певною ознакою, складають ***предметну область***, наприклад, сукупність учнів у даній навчальній групі, сукупність деталей, що виробляються у даному цеху, сукупність виробів, що зберігаються на даному складі тощо. Кожний реальний об'єкт має нескінченно багато характеристик, які визначають його суть, проте в кожній галузі важливими є тільки певна сукупність характеристик. Такою сукупністю характеристик наділяється абстрактний об'єкт, який називається ***моделлю***, або ***інформаційною моделлю*** об'єкта. Вибираючи модель об'єкта, слід подбати про її адекватність, тобто здатність моделі максимально відтворювати поведінку реального об'єкта за певних умов.

Крім характеристик, для реальних об'єктів важливими є також їх взаємозв'язки, внутрішній розвиток.

Для діяльності в певній галузі необхідно мати інформацію про реальні об'єкти даної предметної області, взаємозв'язок між цими об'єктами, розвиток і зміни цих об'єктів. Така інформація зберігається в базах даних. Таким чином, ***база даних*** — це сукупність взаємопов'язаних даних про властивості об'єктів даної предметної області, їх взаємозв'язки і взаємний вплив. Існують такі види баз даних: реляційна, ієрархічна, мережна. Найбільшого поширення у наш час набули реляційні (від англ. relation — відношення) бази даних.

Для взаємодії користувача з базою даних призначений комплекс програм, який називається ***система управління базою даних***

(***СУБД***). Крім забезпечення взаємодії користувача з базою даних, системи керування базами даних виконують такі функції:

- створення баз даних і їх основних елементів;
- введення, коректування і вилучення даних;
- упорядкування даних за певними критеріями;
- вибір сукупності даних, що відповідають певним вимогам;
- оформлення і зображення вихідних даних тощо.

Сукупність бази даних і системи керування нею часто називають ***банком*** даних.

8.2 Основні характеристики Microsoft Access

Система керування базами даних Microsoft Access призначена для реляційних баз даних. У реляційних базах даних інформація про характеристики певного класу реальних об'єктів заноситься в спеціальні таблиці. Кількість характеристик (атрибутів), що визначають сутність об'єктів даного класу, відповідає кількості стовпчиків таблиці. Кожному об'єкту даного класу відповідає один рядок таблиці. Стовпчики таблиці реляційних баз даних називаються ***полями***, а рядки — ***записами***.

Таблиці є основою реляційної бази даних, вони мають такі властивості:

- кожний елемент таблиці (клітинка) є елементом даних;
- кожний стовпчик таблиці має унікальне ім'я;
- кожний стовпчик у таблиці однорідний, тобто складається з однотипних елементів. Це можуть бути числа, текст, дати, логічні значення, графічні об'єкти тощо;
- у таблиці не може бути однакових рядків;
- рядки мають бути з однаковою кількістю елементів (полів), які є різнорідними і взаємопов'язаними;
- послідовність розміщення рядків і стовпчиків довільна.

У СУБД перших поколінь для кожної таблиці створювався свій файл. Для спільного використання таблиць застосовувалися спеціальні програми (утиліти). З розвитком СУБД почали застосовуватися спеціальні команди, які мають виконуватися перед початком роботи, для так званого ***зв'язування*** таблиць. Пізніше почали зберігати кілька таблиць з описом зв'язків між ними. У Microsoft Access є прості й наочні засоби зв'язування таблиць, які зберігаються в одному файлі. Зберігання кількох таблиць в одному файлі має такі переваги:

- можливість розподілу даних за таблицями згідно зі змістом даних, тобто інформація про властивості об'єктів одного класу знаходиться в одній таблиці;
- зберігання інформації в окремих таблицях (модулях) дає змогу змінювати і поповнювати інформацію в кожному модулі окремо, згідно зі змінами об'єктів предметної області;
- редагування даних в одному місці призводить до автоматичного редагування в інших місцях.

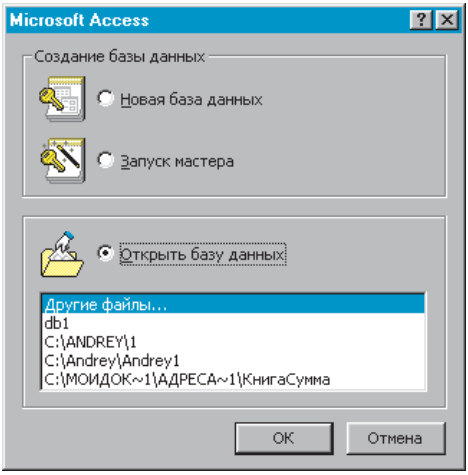


Рис. 8.1

Під час запуску **Microsoft Access** на екрані з'являється діалогове вікно (рис. 8.1) за допомогою якого можна створити нову базу даних або базу даних на основі набору типових баз даних.

Основні об'єкти Microsoft Access. Основними об'єктами Microsoft Access є таблиці, форми, запити, звіти, макроси, модулі (рис. 8.2).

Таблиці баз даних призначені для збереження даних про характеристики об'єктів предметної області.

Форми створюються для введення, перегляду, коригування взаємопов'язаних даних у базі. Зовнішній вигляд форми відповідає звичайному документові.

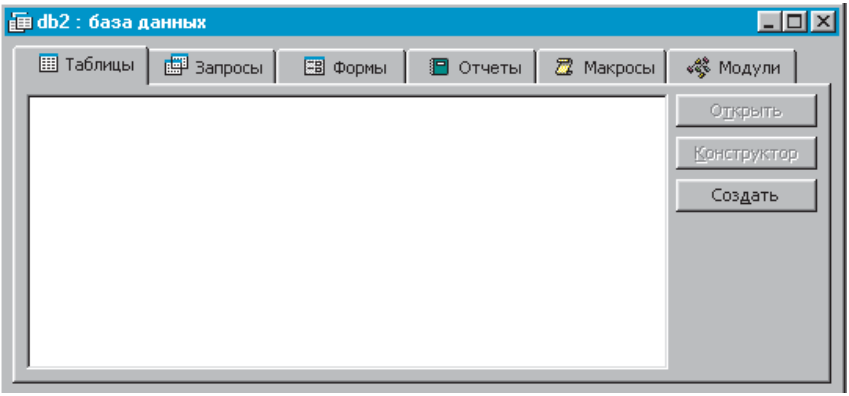


Рис. 8.2

Запити потрібні для вибору з бази даних інформації за певним критерієм або системою критеріїв. За допомогою запиту мож-

на також додати, вилучити чи відновити дані в таблиці або на основі вже існуючих таблиць створити нові.

Звіти призначені для формування вихідного документа, що виводиться, як правило, на друк.

Макроси необхідні для об'єднання певної послідовності дій під час виконання багатоетапної процедури обробки інформації в одне ціле.

Модулі містять програми, написані однією з алгоритмічних мов, для реалізації нестандартних процедур обробки інформації.

8.3 Таблиці і робота з ними

Основні операції обробки інформації у Microsoft Access здійснюються за допомогою двовимірних прямокутних таблиць. У таблицях знаходиться інформація про найважливіші характеристики певного класу об'єктів. Кожному об'єкту відповідає один ***рядок*** таблиці. Дані, що містяться в одному рядку, називаються ***записом***. Таким чином, кожний об'єкт з певного класу об'єктів характеризується одним записом. Кожний рядок складається з однакової кількості клітинок, які називаються ***полями***. У поля заноситься інформація (дані) про певну характеристику (***атрибут***) об'єкта. Однотипні поля, розміщені одне під одним, складають ***стовпчик*** таблиці, якому присвоюється ***ім'я***. Отже, кожний стовпчик (шпальта) таблиці описує характеристику (атрибут) об'єктів даного класу.

Створюючи структуру таблиці, обов'язково вказують імена і типи полів. У Microsoft Access використовуються такі типи полів:

- ❒ текстове;
- ❒ числове;
- ❒ логічне;
- ❒ поле типу лічильник;
- ❒ поле типу MEMO;
- ❒ поле об'єкта OLE.

Оскільки у стовпчику таблиці розміщені поля одного типу та імені, то ім'я і тип поля є іменем і типом стовпчика. Тип поля визначає тип даних, що зберігаються у цьому полі.

Створення таблиць

Таблиці в Microsoft Access можна створити різними способами, зокрема, можна створити таблиці під час створення нової бази даних за командою **Створити** з меню **Файл**. Діалогове вікно, що виникає під час виконання цієї команди, містить вкладку **Таблиці**. Після

вибору цієї вкладки і натискання на кнопку **Створити**, на екран виводиться наступне вікно **Нова таблиця** (рис. 8.3). На правому транспаранті виведені такі способи створення таблиць: **Режим таблиці, Конструктор, Майстер таблиць, Імпорт таблиць, Зв'язок з таблицями.** Розглянемо детальніше кожний зі способів.

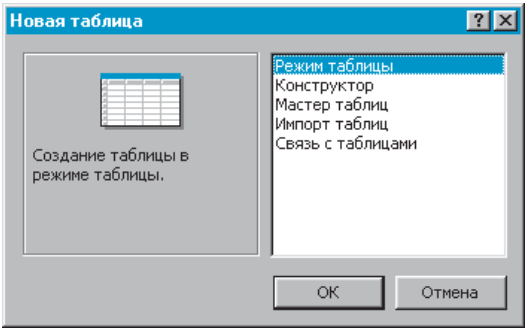


Рис. 8.3

Режим таблиці. У діалоговому вікні **Нова таблиця** вибираємо **Режим таблиці,** внаслідок чого на екран виводиться трафарет стандартної таблиці розміром 20 стовпчиків на 30 рядків із системним іменем **Таблиця N: таблиця** (рис. 8.4).

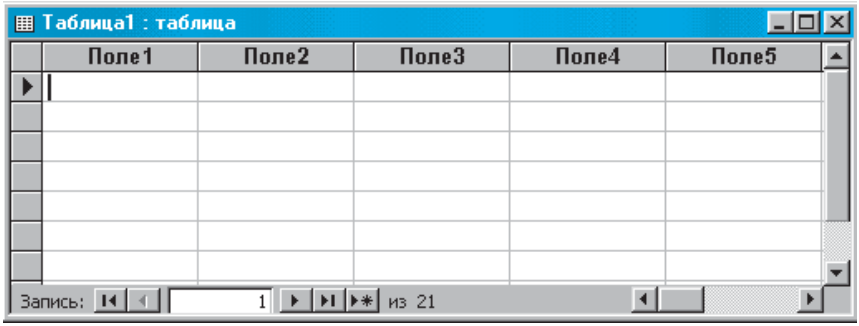


Рис. 8.4

На цьому трафареті системні імена полів замінюються на потрібні, для чого необхідно двічі клацнути мишею на вибраному полі і ввести нове ім'я. Далі послідовно, рядок за рядком, вводяться дані про характеристики об'єкта у відповідні поля. Створюється первинне ключове поле. Щоб пересвідчитися у правильності вибору типу полів та виконати у разі необхідності потрібні зміни, викликається командою **Вид-Конструктор таблиць** вікно **Конструктора** (рис. 8.5). Після завершення цих дій таблиця з вибраним іменем зберігається за допомогою команди **Файл-Зберегти** або відповідної кнопки панелі інструментів.

У режимі **Конструктор** користувач має широкі можливості вибору параметрів таблиці. Вікно **Конструктор: Таблиця N:таблиця,** яке відкривається при виборі цього режиму, містить дві панелі. За допомогою верхньої панелі присвоюється ім'я кожному полю

таблиці і визначається тип даних кожного поля. Нижня панель забезпечує встановлення властивостей кожного поля.

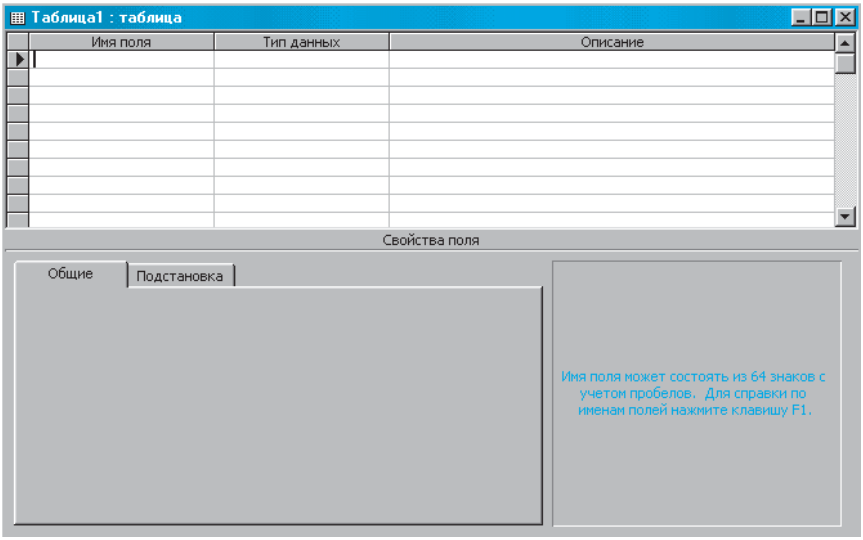


Рис. 8.5

Ім'я поля, яке вводиться в поля першого стовпчика, має бути коротким, інформативним і не перевищувати 64 символів. Тип поля (текстовий, числовий тощо) вибираємо зі списку **Тип даних**, що розкривається.

Опис — це пояснення до кожного поля.

Властивості полів відображуються на нижній панелі вікна **Конструктор**. Засобами цієї панелі користувач може задавати розмір і формат поля, маску введення даних у поле, умови на значення тощо. Вміст вкладки **Загальні** змінюється залежно від типу поля. Створення структури таблиці, як правило, завершується визначенням первинного ключа, який однозначно визначає кожний запис таблиці. Первинний ключ вводиться за командою **Ключове поле** контекстного меню вибраного поля або за допомогою однойменної кнопки панелі інструментів. Як первинний ключ можна вибрати також поле таблиці, що має унікальні значення. Таких полів може бути два і більше. Якщо користувачем не визначене ключове поле, то система автоматично як ключове поле використовує порядковий номер запису, вводячи додаткове поле типу **Лічильник**.

У режимі **Конструктор** можна створювати і змінювати тільки структуру таблиці. Заповнення ж таблиці даними виконується в режимах **Режим таблиці** чи **Майстер таблиць**.

Після формування структури таблиці її необхідно зберегти за командою **Файл-Зберегти** й у діалоговому вікні, що відкривається, задати розміщення її в пам'яті, а у вікні **Нове Ім'я** вказати її ім'я.

Створення таблиці в режимі Майстер таблиць. Термін створення таблиць у цьому режимі істотно зменшується завдяки використанню вже готових зразків таблиць і полів, тобто процес створення таблиць автоматизується. Після вибору режиму **Майстер таблиць** у списку діалогового вікна **Нова таблиця** відкривається діалогове вікно **Створення таблиці** з трьома списками (рис. 8.6).

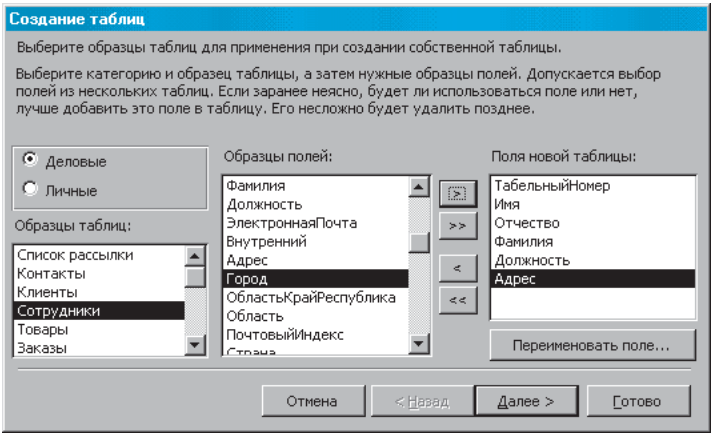


Рис. 8.6

У списку **Зразки таблиць** вибирається потрібний зразок таблиці, а у списку **Зразки полів** — імена полів поточної таблиці. Вибраний зразок поля передається в таблицю, що створюється, за допомогою кнопки >, що знаходиться справа від списку. Дія кнопки >> поширюється на всі зразки полів. Кнопки < і << забезпечують виконання зворотних операцій: вилучають з таблиці, що створюється, вказані поля. Змінити ім'я поля можна за допомогою кнопки **Перейменувати поле**. Типи полів призначаються Майстром за замовчанням. У разі необхідності їх можна переглянути і змінити у режимі **Конструктор**.

Після завершення формування списку полів **Майстер** переходить до наступного кроку (кнопка **Далі**). На цьому кроці вводиться ім'я нової таблиці і вибирається спосіб визначення первинного ключа — автоматично або самостійно — користувачем. Під час самостійного визначення ключа після натискання кнопки **Далі** на екрані з'являється відповідне діалогове вікно, за допомогою якого вибирається ключове поле.

На четвертому кроці роботи **Майстра** встановлюються можливі зв'язки новостворюваної таблиці з іншими таблицями за допомогою діалогового вікна **Зв'язки.**

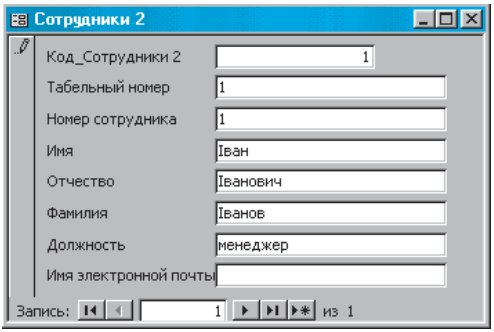


Рис. 8.7

На завершальному кроці роботи **Майстра** встановлюється один з режимів введення даних у створену таблицю за допомогою альтернативних перемикачів **Змінити структуру таблиці, Ввести дані безпосередньо в таблицю, Ввести дані в таблицю за допомогою форми, створеної автором.** Залежно від вибраного режиму на екран виводиться відповідне вікно (рис. 8.7). Новостворена таблиця наведена на рисунку 8.8.

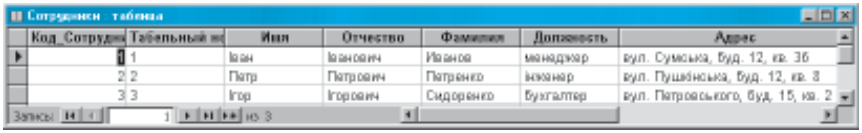


Рис. 8.8

Редагування таблиць

Створені таблиці можна редагувати в разі необхідності у режимі **Конструктор** або **Режим таблиці.**

Редагування структури таблиць у режимі Конструктор. Створюючи таблицю, можна припуститися помилок. Їх можна виправити у вікні **Конструктор**, що містить усі поля та параметри редагованої таблиці. Засобами цього вікна можна:

- ❒ змінювати імена полів, їх тип і параметри;
- ❒ вилучати поля з таблиці та додавати нові;
- ❒ змінювати послідовність полів;
- ❒ змінювати або задавати нові ключові поля тощо.

Ім'я та тип поля можна змінювати за допомогою клавіші **Backspase** або **Delete.** Ім'я редагується чи вводиться заново, а новий тип поля вибирається зі списку стовпчика **Тип даних.** Для кожного типу даних за допомогою елементів нижньої панелі можуть бути встановлені властивості.

Щоб змінити послідовність розміщення полів, необхідно виділити потрібний рядок за допомогою миші. Виділений рядок позначається стрілкою. Якщо клацнути мишею на стрілці, то під

стрілкою з'явиться маленький штриховий прямокутник. Захопивши його за допомогою миші й утримуючи натиснуту ліву клавішу, переміщуємо виділений рядок у потрібне місце.

Щоб вилучити поле з таблиці, потрібно спочатку його виділити, а потім натиснути на клавішу **Delete**. Можна вилучати також групи суміжних і несуміжних полів. Група суміжних полів виділяється так: спочатку виділяється перше поле з діапазону, а далі при натиснутій кнопці **Shift** виділяється останнє поле діапазону. Поля другої несуміжної групи виділяються при натиснутій кнопці **Ctrl**.

Додавання нового рядка виконується за допомогою команди **Вставка-Рядок**. Новий рядок додається над виділеним рядком з позначкою у вигляді маленької чорної стрілки. Можна також додати рядок за допомогою команди **Додати рядок** з контекстно залежного меню виділеного рядка.

8.4 Зв'язки між таблицями. Створення схеми даних у базі

Таблиці містять інформацію про об'єкти предметної області, взаємозв'язані між собою в процесі функціонування, тому для відображення реальних процесів таблиці теж мають бути взаємозв'язаними. Зв'язки між таблицями встановлюються відповідно до інформаційно-логічної моделі предметної області (рис. 8.9).

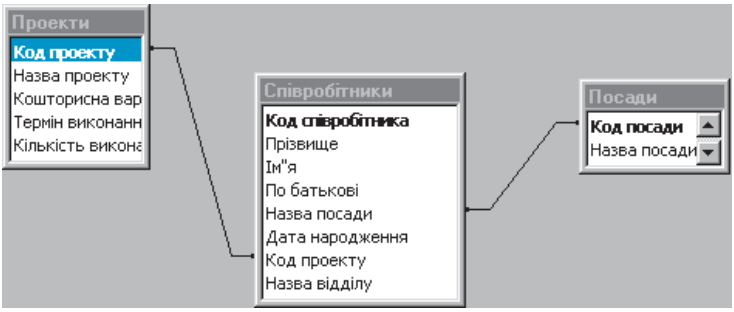


Рис. 8.9

Програма автоматично визначає за вибраним полем тип зв'язку між таблицями. Якщо зв'язані поля головної і підпорядкованої таблиці є унікальними ключами, то встановлюється зв'язок 1:1 «один до одного». Якщо поле зв'язку в головній таблиці є унікальним ключем, а в підпорядкованій ні, то встановлюється зв'язок 1:М «один до багатьох». Для зв'язків 1:1 та 1:М може бути заданий параметр забезпечення цілісності даних, що означає виконання таких умов:

не можна вилучати записи з головної таблиці, якщо не вилучено пов'язані з ними записи у підпорядкованій таблиці; у підпорядковану таблицю не можна вводити запис з неіснуючим у головній таблиці значенням ключа; зміну значень ключа зв'язку головної таблиці слід здійснювати відповідно до змін підпорядкованої таблиці.

Створення схеми даних здійснюється виконанням команди **Сервіс-Схема даних** або за допомогою відповідної кнопки на панелі інструментів. З діалогового вікна цієї команди потрібно викликати вікно **Додавання таблиці** (рис. 8.10) командою **Зв'язки-додати таблицю** або клацнути лівою клавішею миші на відповідній кнопці панелі інструментів **Зв'язок**. Після чого вікно **Додавання таблиць** закривається відповідною кнопкою.

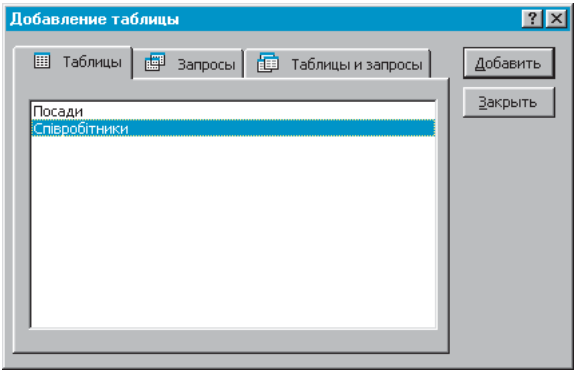


Рис. 8.10

У вікні **Схема даних** зв'язок між відповідними полями двох таблиць здійснюється перенесенням за допомогою миші головної таблиці у відповідне поле підпорядкованої таблиці. Під час здійснення цієї операції відкривається діалогове вікно **Зв'язки**, в якому задаються параметри зв'язків.

8.5 Застосування запитів для обробки і зображення інформації

Інформація, що зберігається у базах даних, відображає стан і функціонування реальних об'єктів. Для обробки інформації, що зберігається у таблицях, застосовується потужний і гнучкий інструмент для обробки такого роду інформації — ***запити***. За допомогою запитів можуть виконуватися обчислення, відновлюватися дані у таблицях, додаватися і вилучатися записи. Результати виконання запиту подаються у зручній формі — у формі таблиці.

За допомогою запитів можна виконувати такі операції: вибрати записи, що задовольняють певні вимоги, з кількох таблиць; включити у підсумкову таблицю додаткові поля й у разі необхідності виконати обчислення для них; згрупувати запити за певними ознаками у визначеному полі; користуючись наявними таблицями, створити нову таблицю; вилучити зі зв'язаних таблиць записи, що відповідають певним умовам тощо.

У Microsoft Access є такі види запитів: запит на вибірку, в результаті виконання якого вибираються дані зі зв'язаних таблиць, а також таблиць, побудованих під час здійснення інших запитів; запит на створення таблиці, призначений для створення нової таблиці; запит на оновлення, який дає змогу вносити зміни у групу записів; запит на додавання, за допомогою якого додаються записи у таблиці бази даних; запит на вилучення, який забезпечує вилучення даних.

Створення запиту здійснюється командою **Створити** на вкладці **Запити** діалогового вікна бази даних. У діалоговому вікні команди слід вибрати один із засобів створення запиту: **Конструктор**, **Простий запит**, **Перехресний запит**, **Повторні записи**, **Записи баз підлеглих**.

8.6 Застосування форм для роботи з базою даних

Дані, що зберігаються у базах даних, відображують стан і властивості об'єктів прикладної області, а тому їх потрібно змінювати по мірі того, як змінюються самі об'єкти. Для занесення оновлених даних у зв'язані таблиці бази даних, коригування, обробки і вилучення призначений такий інструмент як ***форми***. Конструкція форми визначається користувачем, причому, слід вказати дані яких таблиць і яких полів оброблятиме дана форма, які графічні елементи мають бути включені у дану форму. Для обробки інформації до складу форми включаються елементи керування: поле, напис, група, перемикач, позначка, вимикач, поле зі списком тощо. Елементи керування форми можуть бути зв'язаними, вільними або такими, що визначаються.

Зв'язаний елемент керування, приєднаний до поля таблиці або запиту, застосовується для відображення, введення або оновлення значень у полі таблиці.

Вільний елемент керування, не прив'язаний до певного поля таблиці, служить для виведення на екран цифрової, текстової чи графічної інформації.

Елемент керування, що визначається, призначений для здійснення розрахунків.

Для створення форми слід у вікні бази даних вибрати вкладку **Форми**, а на цій вкладці активізувати команду **Створити**. У діалоговому вікні **Нова форма** цієї команди слід вибрати один з таких режимів створення форми: **Конструктор, Майстер форм, Автоформа у стовпчик, Автоформа: стрічкова, Автоформа : таблична, Діаграма, Зведена таблиця.**

8.7 Створення звітів

Звіт, як правило, складається за певний період діяльності в предметній області, для інформаційного супроводу якої і призначена база даних. Особливістю звіту, в порівнянні з іншими інструментами бази даних, є його орієнтація на викладення інформації за певний період діяльності на папері. Звіт, створений за допомогою системи Microsoft Access, забезпечує видачу даних у різних форматах, з різним ступенем деталізації. Для більшої наочності до звіту можуть бути включені графічні об'єкти (рамки, рисунки, графіки, діаграми тощо).

Створення звітів здійснюється командою **Створити** на вкладці **Звіти** діалогового вікна бази даних. У діалоговому вікні **Новий звіт** цієї команди слід вибрати один з режимів створення звітів: **Конструктор, Майстер звітів, Автозвіт: у стовпчик, Автозвіт: рядок, Майстер діаграм, Поштові наклейки.**

Створення звітів за допомогою **Майстра звітів** здійснюється у діалоговому режимі, коли програма на кожному кроці задає питання і на основі отриманих відповідей створює звіт.

1. Що таке база даних?
2. Які функції виконує система керування базою даних?
3. Схарактеризуйте прикладну програму Microsoft Access?
4. З якими об'єктами оперує Microsoft Access?
5. Що таке таблиці бази даних і для чого вони призначені?
6. Як обробляється інформація за допомогою таблиць?
7. Як зв'язуються таблиці у базі даних?

9

Система обробки математичної інформації MathCAD

9.1 Особливості обробки математичної інформації

Математичні обчислення і розрахунки завжди супроводжували технічний прогрес. З розвитком новітніх технологій, розробкою і впровадженням нових машин і пристроїв роль математичної обробки інформації неухильно зростає. Слід зазначити, що перші комп'ютери якраз і призначалися для виконання складних математичних обчислень і у зв'язку з цим отримали назву «електронно-обчислювальні машини». І в наш час обробка математичної інформації, математичні обчислення становлять вагому частину інформаційних технологій.

Для обробки математичної інформації, виконання складних розрахунків і обчислень розроблено ряд пакетів спеціальних програм.

Однією з найпотужніших математичних програм є система **MatLab**. За допомогою цієї системи можна виконувати складні розрахунки в таких галузях як: обробка сигналів, статистика, дискретна математика, економіка тощо. Для кожної прикладної галузі в **MatLab** розроблено набір спеціальних функцій та інструментів. До її складу входить програма **Simulink**, за допомогою якої можна моделювати функціонування реальних пристроїв і систем.

Для здійснення математичних операцій у символьному вигляді призначена система **Maple**, за допомогою якої можна виконувати складні перетворення математичних виразів, спрощувати їх, розв'язувати рівняння і системи рівнянь. Символьний процесор **Maple** використовується в інших математичних системах обробки, зокрема в **MatLab** і **MathCAD**.

Складні математичні обчислення і перетворення можна виконувати за допомогою системи **Mathematica**, що виконує також символьні перетворення, працює з текстовою і графічною інформацією.

До систем обробки математичної інформації належить також система **MathCAD**, можливості якої розглянемо докладніше.

9.2 Особливості системи MathCAD

Система **MathCAD** (від англ. Mathematical Computer Aided Design — математична система автоматичного проектування) призначена для виконання складних математичних обчислень, розрахунків і перетворень.

MathCAD забезпечує такі функції:

- зображення величин цілими, раціональними, ірраціональними, комплексними числами;
- зображення величин логічними змінними;
- зображення чисел у десятковій, двійковій, вісімковій, шістнадцятковій системах числення;
- зображення чисел у формі з фіксованою і плаваючою точкою;
- виконання арифметичних операцій з цілими, раціональними, ірраціональними, комплексними числами;
- можливість задавати точність обчислень і точність зображення результатів обчислень;
- обробку інформації за допомогою великої кількості вбудованих функцій;
- можливість створювати функції користувачем;
- розв'язання лінійних і нелінійних рівнянь, систем рівнянь;
- виконання диференціювання, інтегрування та інших операцій вищої математики;
- розв'язання диференціальних рівнянь;
- виконання операцій у символьному вигляді.

Робота в **MathCAD** не вимагає від користувача знань і навичок програмування, операції записуються у звичному вигляді. Це дає змогу фахівцю у певній галузі — математику, інженеру, економісту, хіміку розв'язувати задачі без спеціальної підготовки звичним способом. Така особливість **MathCAD** привертає до себе увагу студентів і учнів.

Особливістю системи **MathCAD** є те, що операції в математичних виразах виконуються відразу після їх введення. Будь-яка заміна початкових даних призводить до того, що одразу починають перераховуватися всі математичні вирази, в які ці змінені дані входять прямо або опосередковано.

Крім зазначених математичних функцій, **MathCAD** має засоби для зображення інформації в графічному вигляді за допомогою графіків, діаграм тощо.

Математичні обчислення та їх результати, зображені у графічній формі, супроводжуються, як правило, текстовою інформацією: викладом засадних положень, умов, пояснень, доведень, коментарів, висновків. Щоб виконувати ці функції, до складу **MathCAD** входить вбудований текстовий редактор. Це дає змогу за допомогою **MathCAD** готувати наукові статті і звіти, монографії, курсові та дипломні проекти в готовому вигляді, тобто так, як вони надруковані в наукових журналах і книгах. За допомогою вбудованого текстового редактора можна готувати електронні книги з математичних, технічних, економічних та інших дисциплін, у яких математична обробка становить істотну частку. Електронні книги та підручники набирають останнім часом все більшої популярності.

MathCAD має потужну довідкову систему, вікно якої з'являється на екрані під час запуску програми (рис. 9.1).

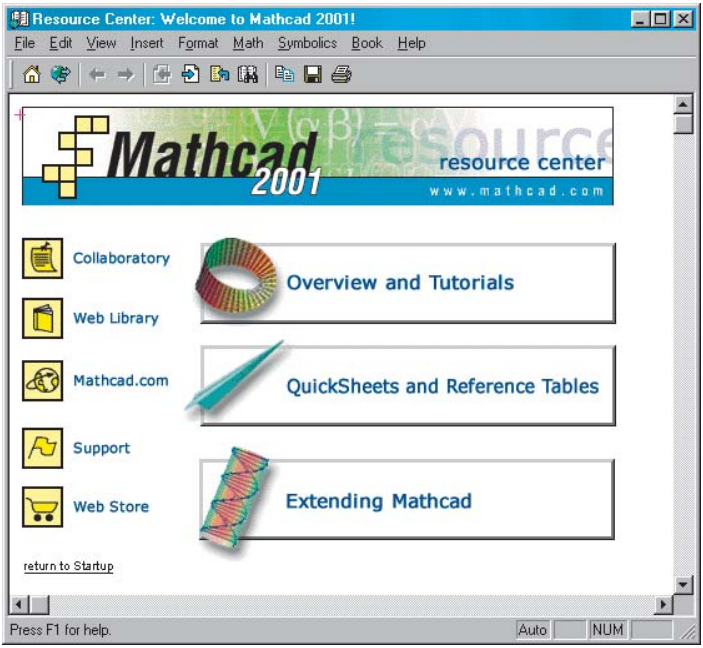


Рис. 9.1

MathCAD, як і багато інших програм, що працюють у середовищі Windows, підтримують технологію **OLE**, тобто в документ можна вставляти об'єкти, створені за допомогою інших прикладних програм. Таким чином, вставляючи у документ кольорові та

чорно-білі графіки, діаграми, схеми, фотографії, можна значно поліпшити зовнішній вигляд документа.

9.3 Вікно MathCAD і його елементи

Система **MathCAD** працює у середовищі Windows, тому має графічний інтерфейс, схожий на інші прикладні програми Windows. Основним засобом взаємодії (інтерфейса) **MathCAD** з користувачем є вікно.

Вікно системи складається з таких елементів (рис. 9.2):

- рядок заголовка;
- рядок меню;
- панель інструментів;
- панель форматування;
- рядок стану;
- вікно документа.

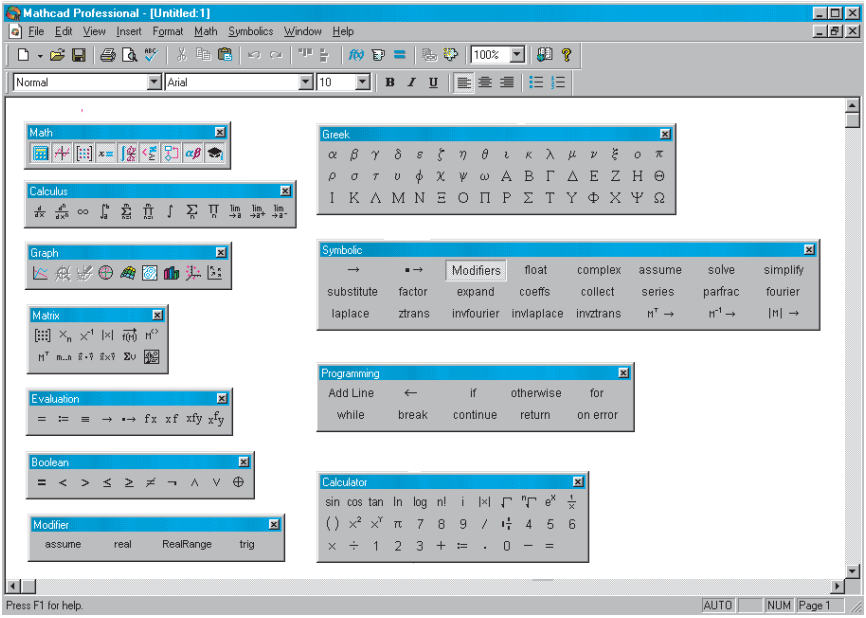


Рис. 9.2

Рядок заголовка містить системну кнопку з піктограмою (картинкою) системи **MathCAD**, при натисканні на яку випадає меню керування вікном з командами **Розгорнути**, **Згорнути**, **Закрити**. Ці команди

можна виконати також за допомогою відповідних кнопок справа у рядку заголовка. Поруч із системною кнопкою знаходиться назва прикладної програми **MathCAD Provessional** і назва документа.

Рядок меню містить такі кнопки з випадними меню: **Файл** (File), **Правка** (Edit), **Вид** (View), **Вставка** (Insert), **Формат** (Format), **Математика** (Math), **Символьний** (Symbolic), **Вікно** (Window), **Допомога** (Help).

На ***панель інструментів*** у вигляді кнопок винесені найбільш вживані команди з меню.

Панель форматування містить інструменти для надання текстовій інформації потрібного вигляду.

Засоби (інструменти) обробки математичної інформації згруповані за певними ознаками і розміщені на таких ***панелях*** **(Toolbar): Калькулятор** (Calculator), **Графіка** (Graph), **Матриці** (Matrix), **Оцінювання** (Evaluation), **Обчислення** (Calculus), **Бульові** (Boolean), **Програмування** (Programming), **Грецькі** (Greek), **Символьний** (Symbolic), **Модифікація** (Modifier) (див. рис. 9.2). Щоб вивести будь-яку панель на екран, необхідно вибрати відповідну команду з випадного меню **Вид** (View). Крім того, потрібну панель можна вивести на екран, якщо натиснути відповідну кнопку на ***Панелі математичних інструментів***, де зведено всі панелі.

У ***рядку стану***, що знаходиться у нижній частині вікна, виводиться інформація про команди.

У вікні **MathCAD** можна відкрити ***вікно документа***. Обробка інформації в ньому здійснюється у формі окремих електронних об'єктів, які називаються ***документом MathCAD***. Кожний документ зберігається і обробляється у вигляді окремого файла, що має розширення .mcd. Файли **MathCAD**, як і файли інших прикладних програм, входять до складу файлової системи Windows.

На екрані дисплея документ **MathCAD** зображений у вигляді аркуша і називається ***робочий аркуш***, що у перекладі з англійської означає буквально «робоче простирадло». Цим жаргонним виразом називають великі аркуші паперу, на які наноситься інформація, зведена з багатьох джерел.

Робочий документ поділяється на ***сторінки***, які в разі потреби, можна надрукувати. Розміри сторінок можна вибрати зі списку стандартних або задати свої.

Робочий документ розділений вертикальною лінією на дві частини. Ліва частина документа виводиться на друк, а праву можна не виводити і розміщувати на ній допоміжну інформацію, проміжні громіздкі обчислення, коментарі. Тому сторінки в правій частині іноді називають прихованими сторінками.

Інформація на робочому аркуші зображена у вигляді блоків інформації. ***Блок*** (Regions) — це місце на робочому аркуші, обмежене

невидимим прямокутником, на якому розміщена математична, графічна, текстова інформація.

Обробка інформації у блоках виконується в такій послідовності: зліва направо і зверху вниз.

Блоки можна виділити більш світлим забарвленням, якщо вибрати команду **Блок** з меню **Вид**.

Щоб створити блок, необхідно клацнути лівою клавішею миші у вибраному місці. У цьому місці з'явиться маленький хрестик (+) червоного кольору. Тепер потрібно почати вводити інформацію з клавіатури. Автоматично, з першим введеним з клавіатури символом, **MathCAD** створить блок.

За змістом інформації блоки поділяються на ***математичні***, ***графічні***, ***текстові*** (рис. 9.3). З кожним блоком можна працювати самостійно: переміщувати, копіювати, редагувати тощо. За замовчанням тип блока буде математичним. Тип блока автоматично зміниться на текстовий, якщо натиснути на клавішу **Пробіл**.

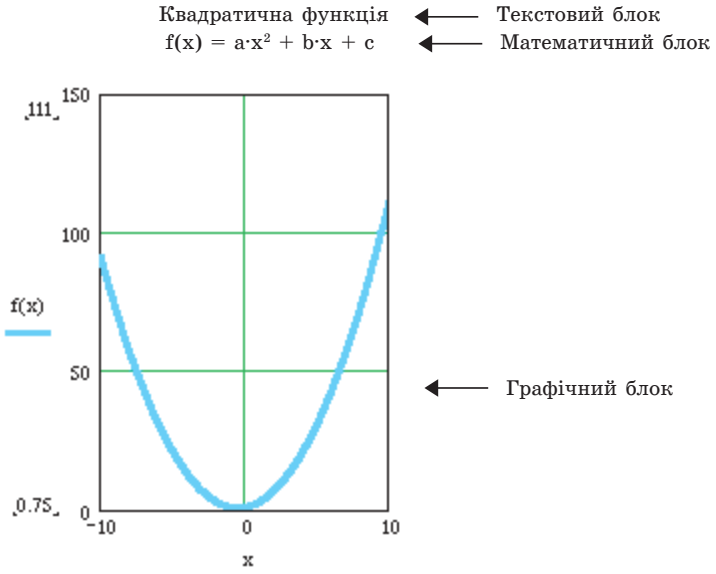


Рис. 9.3

Щоб виділити блок будь-якого типу, досить клацнути на ньому лівою клавішею миші. Тоді навколо блока з'являється чорна рамка. Якщо блок текстовий або графічний, то рамка має маркери — маленькі чорні квадрати у правому нижньому куті, а також посередині на нижній і правій стороні рамки. За допомогою маркерів можна змінювати розміри рамки, а, відповідно, розміри блока. Для цього слід підвести вказівник миші до маркера. Вказівник

змінить свою форму на двоспрямовану стрілку. Якщо тепер клацнути лівою клавішею миші й, утримуючи її у натиснутому стані, переміщувати маркер, то рамка змінюватиме свій розмір. Відпустивши клавішу, отримаємо блок потрібного розміру.

Якщо потрібно виділити не один, а кілька блоків, то слід клацнути лівою клавішею миші й, утримуючи її у натиснутому стані, переміщувати вказівник так, щоб штриховий прямокутник, що з'являється, охопив потрібні блоки. Якщо тепер відпустити клавішу миші, то навколо блоків, які потрапили в зону дії виділення, з'являться штрихові рамки. Виділену таким чином групу блоків можна переміщувати, копіювати, вилучати, вирізати і розміщувати в буфері обміну тощо.

Виділену групу блоків можна вирівняти вздовж горизонтальної або вертикальної ліній. Для цього слід клацнути мишею на відповідній кнопці панелі інструментів або скористатися відповідною командою з меню **Формат**.

9.4 Основні об'єкти MathCAD і операції над ними

У математичних блоках розрізняють такі математичні об'єкти: числа, змінні, функції, вектори, матриці. Вони входять до складу арифметичних і алгебраїчних виразів, а також рівнянь.

Числа (Numbers). MathCAD може оперувати з такими числами: цілими, раціональними, ірраціональними, комплексними.

23	Ціле число
23.0897	Раціональне число у десятковій формі запису
2.591×10^2	Раціональне число у експоненціальній формі запису
$5.9 + j \cdot 6.7$	Комплексне число

Цілі числа записуються без десяткової точки. Раціональні числа зображуються **MathCAD** у вигляді двох цілих чисел. Для ірраціональних чисел є дві форми запису: десяткова (decimal) і експоненціальна (exponential або scientific notation). Комплексні числа складаються з дійсної та уявної частин. Ознакою уявної частини є уявна одиниця, яка позначається літерою ***i*** або ***j***.

Арифметичні вирази. Числа, з'єднані символами математичних дій (операцій), утворюють арифметичні вирази.

Арифметичні вирази вводяться у будь-якому місці робочого аркуша в математичний блок. Числа і знаки математичних операцій

вводяться з клавіатури. Крім клавіатури в **MathCAD** використовуються спеціальні засоби — панелі. На панелях розміщені цифри і знаки математичних операцій. Щоб вставити цифру чи символ операції у математичний блок, досить клацнути на ньому мишею.

Арифметичний вираз обчислюється відразу, як тільки після нього ввести знак =. Цей знак є сигналом для математичного процесора виконувати обчислення, записані зліва від нього. Використання знака = у **MathCAD** повністю аналогічне використанню цього знака в калькуляторах.

$$\left(23 \cdot 0.583^{\,2}\right) + \frac{254}{78.05}$$ Арифметичний вираз

$$\left(23 \cdot 0.583^{\,2}\right) + \frac{254}{78.05} = 1.107 \times 10^{1}$$ Значення арифметичного виразу

Якщо змінити будь-яке число в арифметичному виразі, процесор автоматично перерахує і виведе результат, відповідно до нового значення. У цьому режимі працює як звичайний калькулятор.

Змінна (Variables). ***Змінною*** у **MathCAD** називають величину, яка змінює своє значення у процесі виконання програми. Змінною може бути число, арифметичний чи алгебраїчний вираз, якому присвоєно унікальне ім'я, тобто змінна у **MathCAD** має те саме значення, що і в алгебрі. Ім'я змінної може складатися з довільної кількості допустимих символів. Допустимими символами в імені є букви (як правило латинського алфавіту), цифри, знаки _ і %, першим символом може бути тільки буква. Ім'я змінної має бути коротким і легко запам'ятовуватися. Не слід давати ім'я змінній, яке є іменем вбудованої функції чи константи. Ім'я змінної вводять, як правило, з клавіатури.

Рангові змінні. Крім звичайних змінних у **MathCAD** використовуються так звані рангові змінні, тобто ряд чисел з однаковим інтервалом між ними.

Функції. ***Функція*** — залежність однієї змінної від однієї чи кількох інших змінних, які називаються ***аргументами***. У **MathCAD** функція, як і змінна, має унікальне ***ім'я***. Вимоги до імені функції такі самі, як і до імені змінної. Після імені в дужках вказуються аргументи чи аргумент функції. При зверненні до функції вона повертає результат.

Функції поділяються на ***вбудовані*** і ***задані користувачем***. Вбудовані функції входять до складу **MathCAD** і можуть використовуватися для розрахунків і обчислень. **MathCAD** надає у розпорядження

користувача широкий набір елементарних функцій, функцій вищої математики, спеціальних функцій, зокрема користувач може скористатися такими елементарними функціями, як логарифмічна, показникова, тригонометричні, обернені тригонометричні, прямі й обернені гіперболічні та ін. Вбудовані функції викликаються за допомогою команди **Function** з меню **Insert**, або за допомогою комбінації клавіш **Ctrl+E**. В діалоговому вікні цієї команди вбудовані функції поділені на категорії. В лівому полі вікна слід вибрати категорію функції, а правому — саму функцію (рис. 9.4).

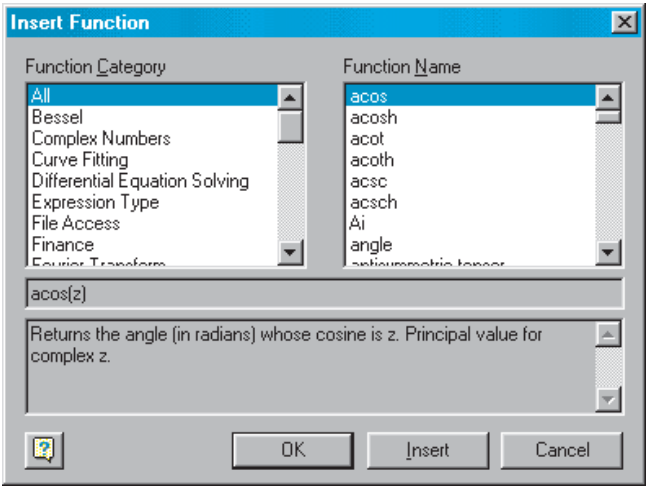


Рис. 9.4

Щоб задати функцію, необхідно вибрати і ввести її ім'я й аргументи, розділені комою, потім ввести знак присвоювання і після цього ввести математичний вираз, що визначає цю функцію, наприклад:

$$f(x, y) := x^2 + \sqrt{y}.$$

Якщо тепер задати значення аргументам x = 37.345, y = 6.839, то функцію можна обчислити при цих значеннях аргументів таким чином $f(x, y) = 1397.264$.

Функцію можна визначити через інші функції, що були визначені раніше, а також через вбудовані функції.

Слід зауважити, що **MathCAD** сприймає імена змінних і функцій однаково, тому не слід давати їм однакові імена.

Матриці. Масив (сукупність) чисел, записаних у формі прямокутної таблиці, називається матрицею. Матриця в **MathCAD** має те саме значення, що й у математиці.

Щоб створити матрицю, користувачеві необхідно вибрати команду **Matrix** з меню **Insert** або скористатися комбінацією клавіш **Ctrl+M**. Після вибору цієї команди на екрані з'являється діалогове вікно, в якому в полях **Рядки** і **Стовпчики** слід набрати потрібне число рядків і стовпчиків матриці. Після натискання на кнопку діалогового вікна на екрані з'являється трафарет матриці з маленькими чорними прямокутниками на місці елементів матриці $\begin{pmatrix} \blacksquare & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \\ \blacksquare & \blacksquare & \blacksquare \end{pmatrix}$.

Тепер можна послідовно, один за одним, вводити елементи матриці. Для введення елемента матриці в потрібному місці, слід спочатку виділити відповідний чорний прямокутник, клацнувши на ньому лівою клавішею миші. Елементом матриці може бути число, змінна або рядкова змінна.

Матриці, як і будь-якій змінній, можна присвоїти ім'я за допомогою оператора :=, наприклад

$$M_1 := \begin{pmatrix} a & 4 & \frac{4}{5} \\ 7{,}45 & x^2 & 78 \\ 67 & 89 & 6{,}09 \cdot 10^{-5} \end{pmatrix}$$

Будь-який елемент створеної матриці можна відкоригувати, для чого достатньо клацнути мишею на потрібному елементі і ввести новий елемент замість старого.

Вектори. Під ***вектором*** розуміють масив (сукупність) чисел, розміщених у певній послідовності. Вектор можна розглядати як матрицю, в якої є тільки один стовпчик (вектор-стовпчик) або тільки один рядок (вектор-рядок). Вектор має багато спільного з ранговою змінною.

Алгебраїчні вирази. Числа, змінні, вбудовані і задані споживачем функції можуть входити до складу алгебраїчних виразів. ***Алгебраїчний вираз*** — імена змінних, функцій і цифри, з'єднані символами математичних операцій, наприклад:

$$\frac{3 \cdot a^2 + 6 \cdot \sin(x)}{\sqrt{b}}$$

Щоб отримати значення алгебраїчного виразу, слід після нього ввести знак =.

9.5 Операції над математичними об'єктами

Над математичними об'єктами здійснюються різні операції.

Операція присвоювання. Після імені змінної обов'язково має стояти знак присвоювання :_. Праворуч від знака присвоювання може стояти або цифра, або алгебраїчний вираз. Якщо праворуч від знака присвоювання стоїть число, то це означає, що змінній з даним іменем присвоюється значення, виражене цим числом. Якщо ж справа стоїть алгебраїчний вираз, то це означає, що дана змінна таким чином визначається через інші змінні і числа, що входять до складу алгебраїчного виразу. Змінні можуть входити до складу математичного виразу, бути аргументом функції або операндом.

Операція виконання. Щоб отримати значення змінної, необхідно після імені змінної ввести знак =. Після введення цього знака, **MathCAD** одразу виводить результат. До цього всі змінні, що входять до складу алгебраїчного виразу, мають бути визначені через інші змінні або через числа. Якщо ця умова не виконана, то після введення знака =, ім'я змінної або алгебраїчний вираз змінює колір на яскраво-червоний і виводиться повідомлення «This variable or function is not define above».

Арифметичні та алгебраїчні операції. У **MathCAD** застосовуються такі математичні дії:

- ❒ + додавання;
- ❒ − віднімання;
- ❒ x множення;
- ❒ / ділення;
- ❒ піднесення до степеня;
- ❒ добування кореня, степеня;
- ❒ добування квадратного кореня;
- ❒ визначення модуля числа;
- ❒ визначення комплексно-сполученого числа.

9.6 Розв'язок рівнянь і систем рівнянь

Одним з найважливіших частин математики є розв'язок рівнянь і систем рівнянь.

Щоб розв'язати одне рівняння з одним невідомим, застосовується функція root. Якщо задане рівняння

$$f(x) = g(x), \tag{9.1}$$

де $f(x)$, $g(x)$ — математичні вирази (9.1), перетворити до вигляду

$$f(x) - g(x) = 0, \tag{9.2}$$

то корені рівняння можна знайти за допомогою функції

$$root(f(x) - g(x),\ x) = \ , \tag{9.3}$$

яка повертає значення x, що задовольняє рівняння (9.3), тобто корінь рівняння.

Систему N лінійних рівнянь можна розв'язати за допомогою функції

$$lsolve(M,\ v) = \ , \tag{9.4}$$

де M — квадратна матриця коефіцієнтів лівої частини системи лінійних рівнянь;

v — вектор вільних членів правої частини системи лінійних рівнянь.

Розв'язок системи нелінійних рівнянь можна отримати за допомогою ключового слова **Given (Дано)**, після чого записується система рівнянь і нерівностей і функції **Find** (x_1, x_2, …, x_n). Функція **Find** повертає значення x_1, x_2, ..., x_n, які задовольняють умови, записані після слова **Given**, наприклад:

$x := 1 \quad y = 1$

Given

$x^2 + y^2 = 6$

$x + y = 2$

$$\text{Find(x, y)} = \begin{pmatrix} 2.414 \\ -0.414 \end{pmatrix}$$

9.7 Математичні операції у символьному вигляді

Для розв'язання багатьох прикладних задач необхідно виконати перетворення математичних виразів у символьному вигляді, наприклад, перш ніж виконати складні обчислення і розрахунки, доцільно спростити математичний вираз чи рівняння. Для виконання таких операцій до складу **MathCAD** включено ядро символьного процесора системи **Maple**.

Символьні операції можна задати такими способами:

- ❒ вибрати операцію з випадного меню **Symbolic**;
- ❒ вибрати потрібну символьну операцію з панелі **Symbolic**;
- ❒ за допомогою символьного знака рівності.

Операції у символьному вигляді можна поділити на такі групи:

- ❒ символьна алгебра: спрощення, розкладання виразів, факторизація, зведення, знаходження поліноміальних коефіцієнтів, розкладання у ряди, розкладання у раціональні дроби, знаходження сум і добутків у символьному вигляді;
- ❒ символьні операції вищої математики: знаходження інтегралів, похідних і границь;
- ❒ розв'язання рівнянь і систем рівнянь у символьному вигляді;
- ❒ операції з матрицями;
- ❒ прямі та обернені перетворення: Лапласа, Фур'є, Z-перетворення;
- ❒ операції з вбудованими і створеними функціями.

Як приклад нижче наведено розв'язок квадратного рівняння у символьному вигляді за допомогою команди **solve**:

$$\left(-a\cdot x^2 + b\cdot x + c\right) solve, x \rightarrow \begin{bmatrix} \frac{1}{2\cdot a}\cdot\left[b + \left(b^2 + 4\cdot a\cdot c\right)^{\frac{1}{2}}\right] \\ \frac{1}{2\cdot a}\cdot\left[b - \left(b^2 + 4\cdot a\cdot c\right)^{\frac{1}{2}}\right] \end{bmatrix}.$$

9.8 Створення графіків засобами MathCAD

Інформація, зображена у вигляді графіків, краще сприймається, має більшу наочність, дає змогу виявити взаємозв'язок між досліджуваними величинами.

MathCAD надає у розпорядження користувача широкий набір засобів для створення двовимірних і тривимірних графіків різноманітної форми.

Двовимірні графіки можуть бути двох видів:

- ❒ у прямокутній декартовій системі координат;
- ❒ у полярній системі координат.

Для створення двовимірних графіків вибираємо команду **X–Y Plot** з меню **Insert**. Також можна скористатися нанеллю **Graph** або комбінацією клавіш **Ctrl+@**. Як приклад, зображено графік функції $f(x) := \frac{(\sin(x)\cdot\cos(x))}{x}$ у декартовій системі координат (рис. 9.5):

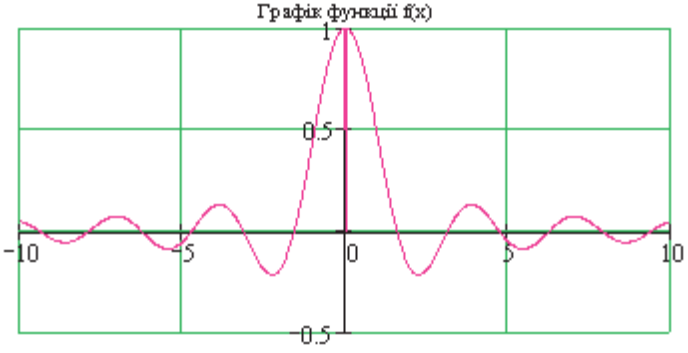


Рис. 9.5

На рисунку 9.6 наведено графік функції $y = ax$ у полярній системі координат:

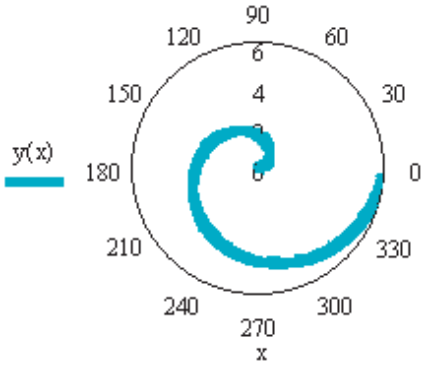


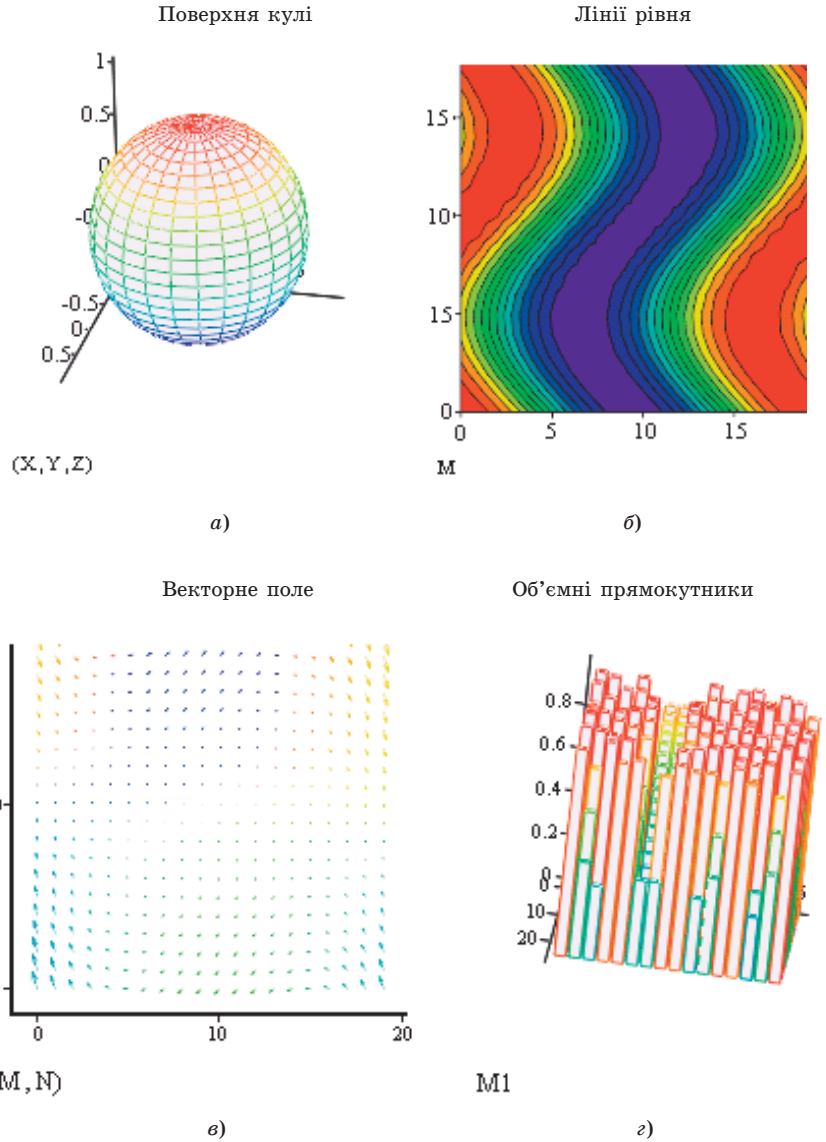
Рис. 9.6

Тривимірні графіки можуть мати вигляд:

- ❒ поверхні (рис. 9.7, *а*);
- ❒ сукупності ліній рівня (рис. 9.7, *б*);

- ❒ векторного поля (рис. 9.7, *в*);
- ❒ об'ємних прямокутників (рис. 9.7, *г*);
- ❒ сукупності розсіяних точок.

На рисунку наведені приклади таких графіків:



а) *б*) *в*) *г*)

Рис. 9.7

9.9 Використання текстової інформації у MathCAD

У наукових статтях, технічних звітах, дисертаціях, курсових і дипломних проектах поряд з математичними виразами, рівняннями, формулами необхідні пояснення, коментарі, виклад засадних положень, аналіз отриманих результатів та інша текстова інформація.

Текстова інформація оформлена у вигляді текстових блоків. Для створення текстового блока, слід клацнути мишею у тому місці робочого аркуша, де потрібно розмістити текстовий блок. Далі слід вибрати команду **Текстовий блок** з меню **Вставити**. Цю саму команду можна викликати за допомогою клавіатури комбінацією клавіш **Ctrl** + **"**. Далі слід вводити символи з клавіатури. Після вводу першого символу навколо нього з'явиться прямокутний контур з трьома маркерами. Із вводом наступних символів контур автоматично розширюватиметься.

Щоб закінчити введення інформації у текстовий блок, необхідно клацнути мишею в будь-якому місці робочого аркуша поза блоком. Натискання на клавішу **ENTER**, як це часто робиться в інших прикладних програмах, призводить не до закінчення операції введення, а до введення нового рядка тексту.

1. Як обробляється математична інформація за допомогою комп'ютера?
2. Які функції виконує система MathCAD?
3. З яких елементів складається вікно MathCAD і для чого вони призначені?
4. Схарактеризуйте основні об'єкти системи MathCAD?
5. Вкажіть відмінності між операторами присвоювання і виконання?
6. Як виконуються у MathCAD арифметичні та алгебраїчні операції над числами?
7. Як розв'язуються рівняння і системи рівнянь за допомогою MathCAD?
8. Яким чином виконуються математичні операції у символьному вигляді?

10

Глобальні і локальні комп'ютерні мережі. Інтернет

10.1 Комп'ютерні мережі. Основні поняття та класифікація

Застосування комп'ютерів у всіх сферах людської діяльності дало змогу обробляти великі масиви інформації. На сучасному етапі розвитку суспільства застосування поодиноких комп'ютерів вже не задовольняє зростаючі вимоги до обробки інформації.

Ускладнення виробництва, застосування новітніх технологій призводить до різкого зростання інформаційних потоків. Крім того, в сучасному суспільстві спостерігаються інтенсивні процеси глобалізації. Все це гостро ставить питання про об'єднання комп'ютерів у систему, створення єдиного інформаційного простору.

Комп'ютерна мережа — це сукупність взаємопов'язаних через канали комп'ютерів, а також апаратних, програмних, інформаційних та інших засобів для забезпечення цієї взаємодії.

Фізичне середовище передачі сигналів. Комп'ютери мережі віддалені один від одного в просторі й для обміну сигналами їх необхідно з'єднати. Комп'ютери можуть з'єднуватися провідними і безпровідними середовищами передачі сигналів.

У ***провідних*** середовищах використовуються такі види кабелів:

- неекранована **(unshielded)** і екранована **(shielded)** скручена пара **(twisted pair)**;
- коаксіальний кабель **(coaxial cable)**;
- оптоволоконний кабель **(fiber cable)**.

Вибирають тип кабелю, виходячи з таких його характеристик: простота встановлення, завадостійкість, швидкість передачі, кількість вузлів, які можна підключити, максимальна відстань прокладання, ціна.

Скручена пара — це два перевиті один навколо одного ізольовані електричні проводи. Скручування проводів зменшує рівень завад, що наводяться сусідніми кабелями і кабелем живлення. Використовується неекранована пара **UTP (unshielded twisted pair)** (рис. 10.1, *а*) і екранована пара (рис. 10.1, *б*) **STP (shielded twisted pair)**. Неекранована скручена пара порівняно дешева і широко застосовується у локальних комп'ютерних та в телефонних

мережах. Максимальна довжина відрізка кабелю — 100 м. Основний недолік неекранованої скрученої пари — низька завадостійкість.

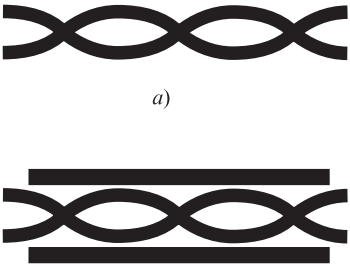


Рис. 10.1

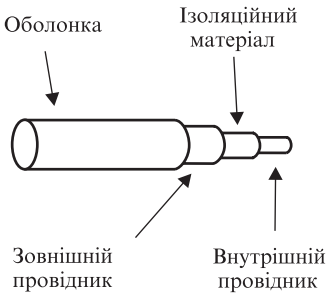


Рис. 10.2

Екранована скручена пара має екран, який зменшує вплив зовнішніх електромагнітних полів і тим самим підвищує завадостійкість.

Коаксіальний кабель складається із внутрішнього провідника (як правило, мідна жила), оточеного шаром ізоляційного матеріалу (поліхлорвініл, тефлон), зовнішнього провідника (спеціальний екран з переплетених мідних проводів) та оболонки (рис. 10.2).

У локальних мережах застосовують два типи коаксіальних кабелів: тонкий і товстий.

Тонкий коаксіальний кабель — це гнучкий кабель діаметром 5 мм. Простий у застосуванні, придатний для будь-якого типу мереж. Підключається безпосередньо до плати мережного адаптера комп'ютера. Максимальна відстань передачі сигналів становить 185 м. Хвильовий опір кабелю 50 Ом.

Товстий коаксіальний кабель — відносно твердий кабель діаметром 10 мм. За ціною він дорожчий за тонкий кабель. Максимальна відстань передачі сигналів до 500 м.

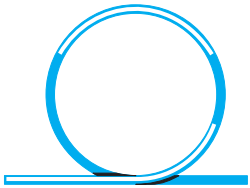


Рис. 10.3

Оптоволоконний кабель. Оптоволоконний кабель складається з оптичного волокна (жили), вкритого скляною оболонкою і має зовнішню захисну оболонку (рис. 10.3). Інформація передається по оптоволоконному кабелю у вигляді модульованих світлових імпульсів. Пропускна спроможність оптоволоконного кабелю сягає 200 000 Мбіт/с.

Оптоволоконний кабель має високий ступінь захисту від завад. Водночас слід зазначити, що цей кабель досить дорогий порівняно з іншими видами кабелів.

Крім провідних каналів зв'язку в комп'ютерних мережах, особливо глобальних, застосовується безпровідний зв'язок.

У безпровідних мережах використовується передача даних за допомогою інфрачервоного випромінювання лазерів і радіохвиль.

Інфрачервоне випромінювання має широкий діапазон частот, що дає змогу передавати сигнали зі швидкістю 10 Мбіт/с на відстань до 30 м.

Канали передач ***на основі лазерів*** мають велику пропускну спроможність, але передача може вестися тільки в межах прямої видимості.

Радіоканал схожий на канал передачі звичайної радіостанції. Пряма видимість необов'язкова, покривається велика площа.

Мережний адаптер — це пристрій для забезпечення взаємодії (інтерфейса) між комп'ютером і мережним каналом передачі даних. Плати мережного адаптера вставляються у слоти розширення мережних комп'ютерів і серверів.

Мережний адаптер призначений для виконання таких функцій:

- ❒ підготовки даних, що надходять від комп'ютера, до передачі через канал зв'язку;
- ❒ здійснення керування потоком даних між комп'ютером і каналом зв'язку;
- ❒ для перетворення інформації, що надходить з каналу зв'язку, у форму, прийнятну для роботи центрального процесора.

Плата мережного адаптера складається з апаратної частини і вбудованих програм, записаних у постійний пристрій пам'яті.

Інформація на адаптер поступає по 16- і 32-розрядних шинах комп'ютера і перетворюється адаптером у послідовність бітів. Далі трансивер перетворює біти інформації в електричні чи оптичні сигнали для їх передачі каналами зв'язку.

Класифікація комп'ютерних мереж. Існуючі комп'ютерні мережі можна класифікувати за такими ознаками: за ступенем територіального розподілу, топологією, методами з'єднання (комутації) комп'ютерів.

За ступенем територіального розподілу комп'ютерні мережі поділяються на:

- ❒ ***локальні*** (місцеві) **(LAN — Local Area Network)**, що охоплюють відносно невеликі території (кілька квадратних кілометрів), наприклад, територію підприємства, організації

чи фірми, і характеризуються наявністю відносно простої, але доволі швидкісної системи передачі даних;

- ***регіональні*** **(RAN — Regional Area Network)**, розташовані в межах визначеного територіального регіону (групи споріднених підприємств, міста, області тощо) і характеризуються більшою складністю порівняно з локальними мережами. Можуть використовуватися для об'єднання кількох локальних мереж у інтегровану систему;
- ***глобальні*** системи **(GAN — Global Area Network)** охоплюють територію держави, кількох держав чи навіть континентів.

Топологія мереж. ***Топологія*** — це геометрична схема з'єднання вузлів мережі. Найбільшого поширення набули мережі з такою топологією: кільце, шина (магістраль), зірка, дерево, повнозв'язкова.

У мережах з шинною топологією канал зв'язку, що з'єднує комп'ютери в мережу, утворює розімкнену лінію, на кінцях якої встановлюються спеціальні заглушки (термінатори). Дані від вузла-передавача поширюються по каналу в обох напрямках до кінців каналу. Дані одночасно отримують всі вузли. Вузол-приймач розпізнає дані, призначені для нього, і читає їх. Шинна топологія характеризується стійкістю роботи в разі несправності окремих вузлів, можливістю нарощування. Разом з тим мережі цього типу мають невелику довжину.

У кільцевій топології вузли з'єднані послідовно один з одним і утворюють кільце. Сигнали передаються в одному напрямку по кільцю і проходять через мережний адаптер кожного комп'ютера. Кожний комп'ютер виступає у ролі повторювача: приймає сигнали і передає наступному. Кільцева топологія має ряд переваг: простота виявлення несправних ділянок, автоматичне підтвердження прийому, можливість використання різних фізичних каналів на різних ділянках, забезпечення гарантованого доступу до каналу навіть у дуже завантаженій мережі.

Основними недоліками кільцевої топології є: залежність надійності мережі від усіх ділянок каналу, складність розширення мережі без переривання її функціонування, затримка сигналу в мережних адаптерах.

У мережах із зіркоподібною топологією всі комп'ютери за допомогою сегментів кабелю підключаються до центрального компонента — ***концентратора*** **(hub)**. Сигнали від комп'ютера-передавача надходять через концентратор до всіх інших комп'ютерів. Мережа з такою топологією надає змогу використання у різних

напрямках різних типів каналів. Вихід з ладу будь-якого комп'ютера, клієнта або кабельного з'єднання не впливає на роботу інших комп'ютерів мережі. Майже всі недоліки цієї топології пов'язані з центральним компонентом (концентратором): при виході його з ладу порушується робота всієї мережі, складність технології та висока вартість центрального вузла, обмежена кількість портів.

У мережах із деревоподібною чи ієрархічною топологією кожний вузол зв'язаний з одним керівним вузлом вищого рівня і одним чи кількома керованими вузлами нижчого рівня. Топологія деревоподібної мережі здебільшого відображає ієрархічну організаційну структуру установи, для якої ця мережа створювалася. Така мережа має переваги з точки зору простоти керування, можливості розширення. Однак, якщо виникне несправність у вузлі, то всі вузли нижніх рівнів виявляться вимкненими з мережі.

У повнозв'язній мережі кожний вузол зв'язаний безпосередньо з усіма іншими вузлами. Перевагою такої мережі є висока надійність функціонування. Недолік — значне зростання кількості каналів зв'язку зі збільшенням кількості вузлів.

Слід зазначити, що розглянуті топології характерні, здебільшого, для локальних мереж. Регіональні, а тим паче глобальні мережі, мають у своєму складі ділянки з різними топологіями.

Комутація у комп'ютерних мережах. Комплекс заходів, призначених для передачі інформації (даних) між абонентами мережі, задіяні при цьому апаратні і програмні засоби мережі, називаються ***комутацією***. Іншими словами, комутація — це технологія передачі даних у комп'ютерних мережах.

Передача інформації (даних) по комп'ютерній мережі здійснюється послідовно через кілька вузлів. Кожний вузол комутує, тобто підключає, з'єднує вхідний канал, по якому приходить пакет даних, з одним з вихідних каналів. Комутація — основне функціональне призначення вузла. Залежно від методів комутації, мережі поділяються на мережі з комутацією каналів, повідомлень і пакетів.

У мережах з ***комутацією каналів*** між передавачем і приймачем повідомлень заздалегідь, до початку передачі, встановлюється фізичне нерозривне з'єднання, тобто утворюється наскрізний тракт із послідовно з'єднаних окремих фізичних каналів.

Перевагами способу комутації каналів є:

- ❒ забезпечення незначних затримок сигналу, що особливо важливо для передачі відео- і аудіоінформації;
- ❒ немає небезпеки виникнення перевантаження чи блокування лінії.

До недоліків цього способу належить:

- ❒ неефективне використання ресурсів мережі, оскільки під час перерв у передачі (коли абонент обробляє інформацію, що надійшла, готує відповідь), ні одна з ділянок тракту не може бути використана для передачі даних між іншими абонентами мережі;
- ❒ зниження пропускної спроможності мережі у цілому через монопольне виділення каналів зв'язку;
- ❒ великий час встановлення зв'язку між абонентами;
- ❒ пропускна спроможність всього тракту передачі визначається каналом з мінімальною пропускною спроможністю.

У мережах з ***комутацією повідомлень*** інформація передається у вигляді ***повідомлень***, які містять заголовок і дані за маршрутом, що визначається кожним вузлом мережі. ***Маршрутом*** називається шлях руху повідомлення між вузлами. Для проходження повідомлень по мережі дані мають бути оформлені у певному ***форматі***. Формат повідомлення і процедура його застосування визначаються мережними протоколами. Між передавачем і приймачем повідомлень фізичний наскрізний тракт передачі не встановлюється, а передавач передає повідомлення в сусідній вузол комутації. Повідомлення приймається цим вузлом і розміщується у його пам'яті. Вузол комутації за адресою приймача, що міститься у заголовку, визначає маршрут, тобто наступний вузол, куди має бути передане повідомлення. Процес прийому, обробки і передачі повідомлення повторюється послідовно всіма вузлами на маршруті від абонента-передавача до абонента-приймача (адресата) повідомлення.

Переваги способу:

- ❒ можливість використання на різних ділянках маршруту каналів зв'язку різної фізичної природи і з різною пропускною спроможністю;
- ❒ незалежні дії окремих вузлів мережі, наявність черг, можливість очікування під час ретрансляції забезпечує високий рівень використання каналів, а наявність буферної пам'яті вузла робить мережу тривкою до перевантажень;
- ❒ послідовна передача повідомлень від вузла до вузла з ретрансляцією і контролем правильності на кожній ділянці зменшує ймовірність доставки повідомлень не за адресою;
- ❒ можливість введення різноманітних категорій терміновості повідомлень і пріоритетів у використанні каналів.

Основним недоліком способу комутації повідомлень є неможливість ведення діалогу між абонентами в реальному часі.

Спосіб ***комутації пакетів*** передбачає поділ повідомлень на окремі частини — ***пакети*** (порції) однакового стандартного обсягу. Кожний пакет має власний заголовок, який містить номер пакета в повідомленні, і адресу приймача повідомлень. Пакети передаються мережею незалежно один від одного, можливо навіть за різними маршрутами. Адресат (приймач) збирає повідомлення з пакетів, що надійшли мережею. Перевагами даного способу є:

- максимальне використання пропускної спроможності каналів за допомогою мультиплексування — розподілення часу роботи каналу для одночасної передачі потоків даних;
- зменшення кількості помилок під час передачі внаслідок зменшення обсягу інформації за сеанс передачі;
- зменшення обсягу буферної пам'яті порівняно з комутацією повідомлень.

Взаємодія компонентів комп'ютерної мережі. Можливість функціонування різнотипних комп'ютерів у складі комп'ютерної мережі можна забезпечити у тому випадку, якщо ці комп'ютери відповідають певним ***єдиним системним стандартам***, які б забезпечували єдність інтерфейсів і правил взаємодії. Різні міжнародні організації ведуть розробку таких стандартів і впровадження їх на практиці. Одним з таких стандартів є еталонна модель взаємодії відкритих систем **(Open System Interconnection reference model — OSI).** Модель базується на трьох базових поняттях:

- системах, що відображають реальні комп'ютери, які надають чи споживають мережні ресурси;
- прикладних процесах, що характеризують ці інформаційні ресурси;
- з'єднаннях, що забезпечують обмін інформацією між прикладними процесами.

Відкрита система — це система, що взаємодіє з іншими системами згідно з прийнятими стандартами.

Згідно з моделлю відкритих систем стандартизація апаратних і програмних засобів проводиться на підставі протоколів. ***Протокол*** — це ієрархічна взаємозв'язана система правил взаємодії.

Основу моделі відкритих систем OSI складає багаторівнева організація взаємодії. Згідно з цією моделлю у комп'ютерній мережі може використовуватися до семи рівнів взаємодії: прикладний (application), представницький (presentation), сеансовий (session), транспортний (transport), мережний (network), канальний (data link), фізичний (physical).

Прикладний, сьомий рівень, є найвищим рівнем в ієрархії, оскільки безпосередньо зв'язаний з прикладними процесами і обслуговує взаємодію між ними. Цей рівень забезпечує взаємодію (інтерфейс) між прикладними програмами і мережею. На цьому рівні обробляють дані в тому вигляді, як вони надходять від прикладного процесу і доставляються в інших прикладний процес.

Представницький рівень забезпечує перетворення формату, синтаксису даних для їх передачі мережею. Іншими словами, здійснюється перетворення (трансляція) з різних мов, форматів, кодів, з якими працюють різнотипні комп'ютери, у форму, придатну для передачі мережею.

Сеансовий рівень забезпечує керування передачею інформації між прикладними процесами. Він дозволяє звертатися до будь-якого процесу за його іменем, незалежно від того, на якому комп'ютері знаходиться цей процес.

Транспортний рівень визначає адресацію фізичних пристроїв у мережі. Однак, головним завданням цього рівня є забезпечення ефективних, зручних і надійних форм передачі інформації. Часто інформація ділиться на окремі блоки — кадри або пакети, які пересилаються мережею від одного комп'ютера до іншого.

На ***мережному*** рівні визначаються маршрути руху пакетів каналами мережі, керування інформаційними потоками, виявлення помилок передачі і повідомлення про них.

На ***канальному*** рівні здійснюється відповідне оформлення пакетів даних і надійна передача через фізичний канал.

Найнижчий, **фізичний** рівень, забезпечує взаємодію комп'ютера з фізичним каналом. На цьому рівні формуються сигнали, тобто фізичні процеси, що несуть інформацію.

Відповідно до семирівневої моделі розрізняють і сім видів протоколів.

10.2 Глобальна комп'ютерна мережа Інтернет

Світова глобальна комп'ютерна мережа **Інтернет** — це об'єднання глобальних, регіональних та локальних мереж, окремих комп'ютерних систем і комп'ютерів, які використовують для обміну інформації комплекс стандартних правил взаємодії (протоколів). Обсяг інформаційних ресурсів та кількість користувачів **Інтернету** гігантські і постійно збільшуються, розширюється і різноманітність послуг, що надаються цією мережею (рис. 10.4). На

сьогодні, за даними статистики, мережа **Інтернету** об'єднує понад 100 мільйонів користувачів більш як у 100 країнах світу, кожні 30 хвилин до **Інтернету** приєднується чергова комп'ютерна мережа.

Рис. 10.4

Взаємодія об'єктів в Інтернеті

Як уже зазначалося, **Інтернет** є системою, що об'єднує локальні, регіональні мережі, окремі комп'ютери. Всі ці об'єкти мають кожний свої особливості і характеристики апаратних і програмних засобів.

Основним завданням, яке необхідно вирішити, створюючи такі неоднорідні системи, є забезпечення сумісності устаткування за електричними і механічними характеристиками і сумісності програмних засобів і даних за форматом, системою кодування тощо.

Вирішення цієї проблеми у мережі **Інтернет** ґрунтується на ***моделі відкритих систем*** **(OSI)**. Згідно з цією моделлю (див. попередній підрозділ) стандартизація апаратури і програмного забезпечення здійснюється на підставі ієрархічної системи протоколів, тобто комплексу норм і правил взаємодії. У мережі **Інтернету** згідно з моделлю **OSI** можуть використовуватися до семи рівнів взаємодії.

Основу системи **OSI** в **Інтернеті** складає комплекс протоколів **TCP/IP**. Основна властивість, яка забезпечується **TCP/IP**, — можливість передачі даних від вузла до вузла у структурі, що постійно змінюється і складається з різнорідних елементів.

Свою назву **TCP/IP** отримав від двох протоколів. Це **Transmission Control Protocol (TCP)** — протокол транспортного рівня, який визначає, *як* відбувається передача інформації, та **Internet Protocol (IP)** — адресний протокол мережного рівня, що визначає *куди* передається інформація.

Протокол TCP. Згідно з протоколом **TCP** дані, які відправляються, поділяються на невеликі ***пакети***, після чого до кожного пакета додаються адреси відправника та одержувача, а також відомості, необхідні для того, щоб правильно зібрати документ з пакетів на комп'ютері одержувача.

Протокол IP. За протоколом **IP** здійснюється адресація і маршрутизація пакетів у мережі. Для цього кожний користувач мусить мати свою унікальну адресу (**IP**-адреса). **IP**-адреса служить ідентифікатором комп'ютера споживача і складається з чотирьох чисел, розділених крапками. Чотири числа **IP**-адреси передаються чотирма байтами. На кожному вузлі мережі за чотирма числами **IP**-адреси визначається маршрут відправлення пакетів у даний момент, з урахуванням умов зв'язку, пропускної спроможності каналу, наявної черги. Таким чином, пакети навіть одного документа можуть передаватися незалежно один від одного різними маршрутами і каналами.

Незважаючи на те, що у мережі Інтернет використовуються протоколи інших рівнів, її називають **TCP/IP**-мережею, оскільки ці два протоколи є найважливішими.

Система адресації в Інтернеті

Для обміну інформацією кожному комп'ютерові необхідно мати свою адресу. В Інтернеті використовуються два типи адрес: цифрові, або IP-адреси, і доменні (від англ. domain — область, сфера). Розглянемо структуру кожного з типів.

IP-адреса аналогічна поштовому індексу, що містить інформацію про місто (перші три цифри) і поштове відділення у ньому (другі три цифри). IP-адреса складається з чотирьох чисел, розділених крапками, наприклад 198.162.201.204. Кожне з чисел займає один байт, тобто вісім двійкових розрядів, тому може набувати значення від 1 до 255. Ліва частина IP-адреси визначає конкретну мережу в Інтернеті й називається мережним ідентифікатором (англ. network ID). Права частина IP-адреси визначає конкретний комп'ютер у цій мережі і називається ідентифікатором комп'ютера (англ. host ID). Залежно від того, скільки байтів у IP-адресі відведено під мережний ідентифікатор — один, два, або три, розрізняють відповідно класи A, B, C IP-адрес.

IP-адреси використовують програмні засоби мережі для пересилання інформації між комп'ютерами. Однак для сприйняття користувачем IP-адреса незручна. Людина краще сприймає і запам'ятовує слова, а не числа. Тому користувачі працюють з доменними адресами — унікальними іменами комп'ютерів у мережі. Доменна адреса, так само як і IP-адреса, складається з частин, розділених крапками. На відміну від IP-адреси, яка уточнює місце призначення зліва направо, доменна адреса робить це у зворотній послідовності — справа наліво: спочатку йде ім'я комп'ютера, а потім ім'я мережі, в якій він знаходиться.

Щоб користувачам Інтернету було зручніше зв'язуватися один з одним, весь простір адрес розділено на області — домени. Можливий також поділ за певними ознаками всередині доменів. Доменна адреса комп'ютера складається як мінімум з двох рівнів доменів:

Ім'я комп'ютера ... Домен другого рівня · Домен першого рівня.

Крайній праворуч — домен першого рівня, наступний зліва — домен другого рівня і т. д.

Домен першого рівня визначає країну або тип організації, якій належить комп'ютер. Існують скорочені до двох букв домени країн, наприклад, Україна — **ua**, Росія — **ru**, США — **us**, Франція — **fr** та ін. Домени організацій скорочуються до трьох букв, наприклад, університети та інші учбові заклади — **edu**, урядові установи — **gov**, комерційні організації — **com** тощо.

Домен другого рівня визначає організацію, яка володіє або керує мережею, в якій знаходиться даний комп'ютер. Здебільшого ім'я цього домену збігається з назвою відповідної фірми або її торгової марки.

Ім'я комп'ютера вказує на конкретний комп'ютер у мережі, визначений доменами першого і другого (а у деяких випадках наступних) рівнів. Воно реєструється тільки у цій мережі і тільки ця мережа «відповідальна» за передачу інформації конкретному комп'ютеру-адресату.

Щоб мережний комп'ютер «зрозумів», куди слід передати повідомлення, доменна адреса має бути перетворена в **IP**-адресу. Таке перетворення здійснюється на спеціальних серверах мережі, де зберігаються таблиці відповідності між доменними адресами і **IP**-адресами. Такі сервери називаються ***DNS-серверами*** (від англ. Domain Name System — система доменних імен). DNS-сервери розміщені по всій мережі Інтернету. Кожний з них зберігає інформацію про велику кількість комп'ютерів. Якщо **IP**-адреса потрібного комп'ютера невідома даному DNS-серверу, він звертається

до найближчого сусіднього DNS-сервера, і далі, доки не буде знайдена потрібна адреса. Адресу одного з DNS-серверів користувач має вказати під час настройки комп'ютера для роботи в Інтернеті.

Служби Інтернету

Користувач (клієнт) може скористатися різними послугами (сервісами), які надає йому через свої служби Інтернет. Розглянемо найбільш відомі і поширені послуги, які забезпечує Інтернет.

Пересилання файлів згідно з протоколом FTP (File Transfer Protocol). Ця служба дає змогу надсилати і отримувати текстові файли і файли з інформацією у вигляді двійкових чисел будь-якими клієнтами Інтернету, згідно з протоколом **FTP**. Протокол **FTP** — це протокол передачі файлів, один із перших послуг Інтернету. Встановивши зв'язок з віддаленим комп'ютером, користувач може скопіювати довільний файл, що міститься на цьому комп'ютері, на свій комп'ютер або зі свого на віддалений комп'ютер.

Gopher — це удосконалена система пересилання файлів. На відміну від **FTR**, вона працює з файлами, що розміщені на численних комп'ютерах, а також дає змогу за допомогою меню не тільки переглянути списки ресурсів, але і переслати потрібний матеріал, причому знати, де він знаходиться, не обов'язково. **Gopher —** це одна з найбільш повних систем перегляду, інтегрована з іншими системами **(FTR, Telnet)**. Мережні засоби за допомогою розподілених індексів зв'язані в єдину систему, що називається **Gopherspace (Gopher-простір)**. Доступ до ресурсів **Gopher-простору** здійснюється за допомогою запропонованих меню, а пошук — за допомогою спеціальних пошукових програм.

Електронна пошта (E-mail). Електронна пошта — одна з найпопулярніших і найпоширеніших послуг в Інтернеті. Потоки інформації Інтернету складаються, в основному, з електронної кореспонденції, і багато людей, підключившись до Інтернету, користуються, в основному, саме цією послугою. В Інтернеті кожне поштове повідомлення складається з двох частин: ***заголовка*** і ***тіла***. Заголовок повідомлення містить службову інформацію: адресу відправника, адресу отримувача, дату і час відправки тощо. Тіло повідомлення містить основний текст.

Поштова адреса в Інтернеті складається з двох частин, розділених знаком @: **ім'я користувача @ ім'я комп'ютера**. Існує багато прикладних програм, що реалізують поштову службу в Інтернеті: Mail, Eudora, Microsoft Exchange, Outlook Express та ін.

Mail Lists (списки розсилання). Дану послугу створено на основі протоколу електронної пошти. Підписавшись (безкоштовно) на списки розсилання, можна регулярно отримувати електронною

поштою повідомлення на певні теми: науково-технічні та економічні огляди, презентація нових апаратних і програмних засобів тощо.

Мережний протокол передачі новин (Network News Transfer Protocol — NNTP). Цей протокол розроблений з метою розповсюдження та отримання статті з новинами. **UseNet** — одна з областей застосування **NNTP**. Цей сервіс забезпечує обмін інформацією у формі статей, повідомлень, оголошень тощо, і є образом (метафорою) електронної дошки оголошень і об'яв, на яку будь-який учасник може помістити своє повідомлення, і воно стає доступним для всіх інших.

Новини поділяються за темами на ***групи новин*** або ***телеконференції***. Вони працюють 24 години на добу, 365 днів на рік. Щоб отримати доступ до телеконференцій, потрібно мати на комп'ютері відповідну програму і передплатити конференції, які зацікавили користувача.

Послуга WWW (World Wide Web — всесвітня павутина). **WWW** — єдиний інформаційний простір, що складається з сотень мільйонів гіпертекстових електронних документів, що зберігаються на **WWW** — серверах у різних куточках земної кулі. Окремий гіпертекстовий документ називається ***Web-сторінкою***. Група тематично об'єднаних сторінок утворюють ***Web-вузол*** або ***Web-сайт***. Для передачі в Інтернеті використовується протокол **http (*hyper text transfer protocol*)**.

Гіпертекстовим документом називають документ, у якому є **гіперпосилання**. Що ж таке гіперпосилання? Ще задовго до появи комп'ютерів автори у книгах і статтях користувалися ***примітками***. Якщо коло певного слова автор поставить знак примітки, то внизу сторінки під таким самим знаком розміщено пояснення вжитого терміна. Крім пояснення, і це дуже важливо, містилося посилання на інші джерела (книги, статті, дисертації тощо), де можна більш докладно ознайомитися з даною темою. Гіперпосилання є подальшим розвитком цієї ідеї на новій технічній базі із застосуванням нових засобів.

Гіперпосилання реалізуються таким чином: слово або термін у документі виділяється кольором або формою шрифту. З цим виділеним словом зв'язаний інший документ, який детальніше пояснює цей термін і може бути розміщений на сервері в іншій частині світу. Цей гіпертекстовий документ, у свою чергу, пов'язаний з іншим документом через гіперпосилання. Таким чином, у гіпертекстовому документі немає ні початку ні кінця. Такі документи утворюють єдину взаємозв'язану систему, яка нагадує павутину, що покриває весь світ.

Для створення гіпертекстових документів використовується спеціальна мова **HTML (Hyper Text Markup Language)**.

Крім тексту, на **Web-сторінках** можуть розміщуватися графічні зображення (картини, фотографії), звукова та відеоінформація, анімаційні об'єкти тощо. Посилання також можуть бути на інформацію будь-якого роду.

У всіх ресурсів **Інтернету**, в тому числі у **Web-сторінок**, є своя адреса, яка задається за допомогою **URL (Uniform Resource Locator** — уніфікований локатор (визначник положення) ресурсів). **URL** — стандарт, прийнятий в **Інтернеті** для визначення місця знаходження будь-якого ресурсу. **URL** складається з трьох частин:

Схема Хост Шлях

Схема описує протокол прикладної програми. Найчастіше використовується протокол передачі гіпертекстових документів **http**. За назвою протоколу ставляться символи ://.

Хост — це доменне ім'я комп'ютера, на якому знаходиться ресурс.

Шлях — це повний маршрут до документа і, можливо, його ім'я.

Перегляд **Web-сторінок** і перехід від одного до іншого документа через гіперпосилання здійснюються за допомогою спеціальних програм з графічним інтерфейсом — ***браузерів*** (від англ. to browse — переглядати). Найпоширенішими **Web-браузерами** в наш час є **Internet Explorer** (від англ. explorer — дослідник) фірми **Microsoft** і **Netscape Navigator** (від англ. navigator — навігатор) фірми **Netscape**.

Щоб орієнтуватися у гігантських потоках інформації, які надходять до користувача через браузери, і вибрати потрібну, створені спеціальні ***пошукові програми***, які знаходяться на пошукових серверах. Отримати доступ до них просто, достатньо знати їх **URL**.

Послуга Telnet (віддалений доступ). Служба **Telnet** дає змогу одному комп'ютеру, підключеному до **Інтернету**, отримати віддалений доступ до іншого комп'ютера, також підключеного до **Інтернету**. Служба **Telnet** використовується, здебільшого, для доступу до баз даних, відкритих для широкої публіки: каталогів бібліотек і електронних дошок об'яв. Часто за допомогою цієї служби здійснюють дистанційне керування складними технічними об'єктами: телескопами, відеокамерами, промисловими роботами. У минулому ця послуга широко використовувалася для проведення складних математичних розрахунків і обчислень на віддаленому супер-комп'ютері.

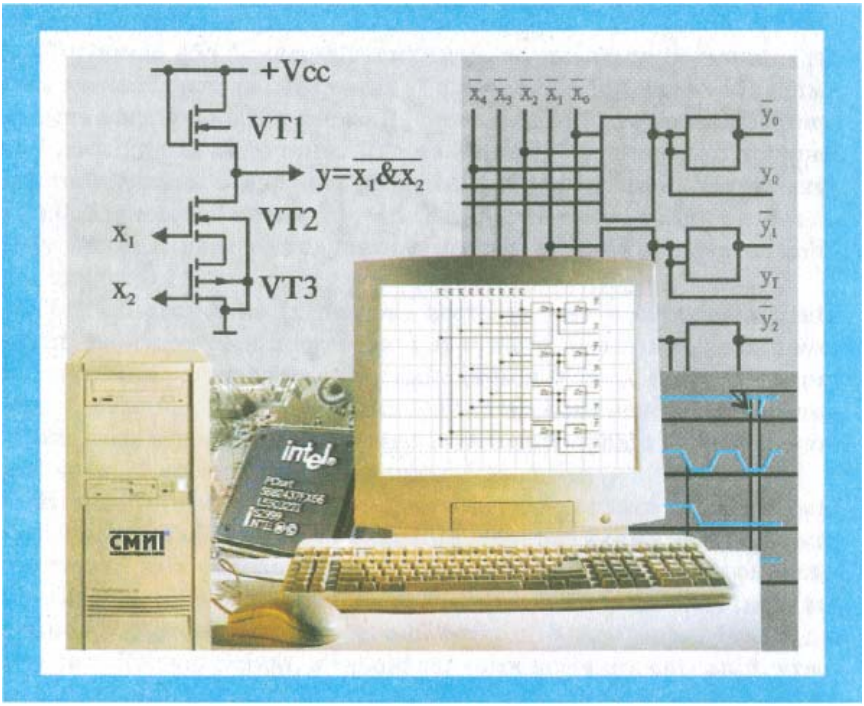
Послуга IRC. Служба **IRC (Internet Relay Chat** — трансляція бесід в **Інтернеті)** дає змогу спілкуватися з реальними людьми

в реальному часі. Розмови (англ. chat — дружня розмова, бесіда) відбуваються у так званих «кімнатах» (англ. room) або «каналах» (англ. channel), які можуть бути як відкриті (англ. public), так і закриті (англ. private). Кожна «кімната» характеризується своєю темою. Для участі в розмові, яка ведеться у «кімнаті» за допомогою **IRC**, необхідно підключитися до комп'ютера, на якому виконується серверна програма **IRC**. Самі сервери з'єднані між собою й утворюють кілька окремих мереж. Розмову можна вести тільки між людьми, комп'ютери яких підключені до однієї й тієї ж мережі. Після підключення до сервера слід вибрати «кімнату» з потрібною тематикою. Далі процес нагадує розмови, які ведуться на аматорських каналах, тільки замість мікрофонів використовується клавіатура. Для участі в таких розмовах слід встановити на комп'ютері одну з програм-клієнтів **IRC**: **Netscape Chat**, **mIRC**, **Microsoft Chat** тощо.

1. Що таке комп'ютерна мережа?
2. Яке фізичне середовище застосовується для передачі інформації у комп'ютерних мережах?
3. Що таке мережний адаптер і для чого він призначений?
4. Схарактеризуйте види комп'ютерних мереж?
5. Що таке комутація і як вона застосовується у комп'ютерних мережах?
6. Скільки та які рівні взаємодії використовуються у відкритих системах?
7. Що таке протокол TCP/IP?
8. Які системи адресації застосовуються в Інтернеті?
9. Схарактеризуйте основні служби Інтернету?
10. Що таке гіпертекст і для чого він застосовується?

2 частина

Комп’ютерно-інформаційні технології



11

Логічні елементи та цифрові пристрої обробки інформації

11.1 Логічні (булеві) змінні та операції над ними

Інформація існує в різних формах: текстова (книги, газети, журнали, доповіді, брошури тощо), графічна (картини, технічні креслення деталей тощо), відеоінформація (фільми, кліпи, репортажі), аудіоінформація, тобто звукова (доповіді, лекції тощо). Засоби обчислювальної техніки обробляють інформацію, подану у вигляді двійкових чисел, тобто чисел, що мають тільки два значення, тому всі види інформації перетворюють у форму двійкових чисел. Обчислювальні пристрої будь-якого ступеня складності складаються із сукупності елементарних пристроїв, що виконують елементарні логічні операції.

Логічною або ***булевою*** величиною називається величина, що може мати тільки два значення. Такі величини названі булевими на честь англійського вченого Д. Буля, котрий уперше запропонував і дослідив ці величини. Два альтернативні значення логічної величини можна позначати різними способами: двійковими цифрами **0, 1**; словами «істинно», «хибно»; «так», «ні» тощо. У подальшому викладі використовуватимемо для логічних змінних двійкові цифри **0** і **1**.

Логічні змінні та їх широке застосування в обчислювальній техніці пояснюються простотою і зручністю реалізації за допомогою технічних пристроїв, що знаходяться у двох станах, наприклад, перемикач у станах «замкнено» і «розімкнено», транзистор у станах «відкритий» і «закритий», магнітопровід у станах «намагнічено» і «розмагнічено» тощо.

Над логічними величинами можна здійснювати різні математичні ***операції***. Сукупність операцій над логічними величинами, властивості цих операцій і правила виконання називаються ***булевою алгеброю***. Результат логічної операції є також логічною величиною, тобто результат логічної операції може мати тільки два значення. Логічні змінні, з'єднані знаками логічних операцій, утворюють ***логічні вирази***. Значення логічного виразу присвоюється логічній змінній, яка залежить від змінних, що входять до складу

логічного виразу, і називається ***логічною функцією***. Логічні функції є одним із видів ***математичних функцій***.

Логічна функція, як і будь-яка математична функція, може виражатися аналітично, тобто за допомогою формули. Крім аналітичного способу, логічні функції зручно подавати в табличній формі. Таблиці, що визначають логічні функції, називаються ***таблицями істиності***. В них відображуються значення логічної функції при всіх можливих значеннях аргументів.

Елементарні логічні функції (операції)

Логічні функції можна представити через ***елементарні логічні функції***. Розглянемо найпоширеніші елементарні операції над двома логічними величинами.

Операція логічного додавання (диз'юнкція). Операція логічного додавання позначається знаком + або $\vee$. Логічна функція, що відповідає операції логічного додавання, називається функцією **АБО**. Запишемо логічну операцію додавання для всіх можливих значень аргументів:

$$0 \vee 0 = 0, \quad 0 \vee 1 = 1, \quad 1 \vee 0 = 1, \quad 1 \vee 1 = 1. \qquad (11.1)$$

Результат операції логічного додавання має значення 0 тільки в тому випадку, якщо обидва аргументи мають значення 0, а значення 1, якщо хоч один з аргументів має значення 1. Співвідношення (11.1) зручно подавати у формі таблиці:

Таблиця 11.1

x_1	x_2	$f(x_1, x_2) = x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1

Операція логічного множення (кон'юнкція). Операція логічного множення позначається знаком математичного множення, тобто крапкою, яку можна не писати, або символом &. Логічна функція, що відповідає операції логічного множення, називається функцією **І**. Запишемо логічну операцію множення для всіх можливих значень аргументів:

$$0 \cdot 0 = 0, \quad 0 \cdot 1 = 0, \quad 1 \cdot 0 = 0, \quad 1 \cdot 1 = 1. \qquad (11.2)$$

Результат операції логічного множення має значення 1 тільки в тому випадку, якщо обидва аргументи мають значення 1, а значення 0, якщо хоч один з аргументів має значення 0.

Співвідношення (11.2) зручно подавати у формі таблиці:

Таблиця 11.2

x_1	x_2	$f(x_1, x_2) = x_1 \cdot x_2 = x_1 \,\&\, x_2$
0	0	0
0	1	0
1	0	0
1	1	1

Операція логічного заперечення (інверсія). Операція логічного заперечення позначається рискою над іменем змінної $\overline{x}$. Запишемо операцію логічного заперечення для всіх можливих значень аргументу

$$\overline{0} = 1, \qquad \overline{1} = 0. \tag{11.3}$$

Запишемо співвідношення 11.3 у вигляді таблиці:

Таблиця 11.3

x	$y = \overline{x}$
0	1
1	0

Властивості логічних операцій

Операції логічного додавання (диз'юнкції) і множення (кон'юнкції) мають певні властивості, які визначаються тотожностями. Довести ці тотожності можна, наприклад, за допомогою таблиць істиності. Розглянемо найважливіші властивості логічних операцій.

$$x \vee 0 = x, \qquad x \cdot 0 = 0, \tag{11.4}$$

$$x \vee 1 = 1, \qquad x \cdot 1 = x. \tag{11.5}$$

Комутативність:

$$x_1 \vee x_2 = x_2 \vee x_1,\ x_1 \cdot x_2 = x_2 \cdot x_1. \tag{11.6}$$

Асоціативність:

$$(x_1 \vee x_2) \vee x_3 = x_1 \vee (x_2 \vee x_3);$$
$$(x_1 \cdot x_2) \cdot x_3 = x_1 \cdot (x_2 \cdot x_3). \tag{11.7}$$

Дистрибутивність:

$$x_1 \cdot (x_2 \vee x_3) = x_1 \cdot x_2 \vee x_1 \cdot x_3;$$
$$x_1 \vee x_2 \cdot x_3 = (x_1 \vee x_2) \cdot (x_1 \vee x_3). \tag{11.8}$$

Ідемпотентність:

$$x \vee x = x,\ x \cdot x = x; \tag{11.9}$$

$$\overline{\overline{x}} = x. \tag{11.10}$$

Поглинання:

$$x_1 \vee x_1 \cdot x_2 = x_1,\ (x_1 \vee x_2) \cdot x_1 = x_1. \tag{11.11}$$

Склеювання:

$$x_1 x_2 \vee x_1 \overline{x_2} = x_1,\ (x_1 \vee x_2) \cdot (x_1 \vee \overline{x_2}) = x_1. \tag{11.12}$$

Правило де Моргана:

$$\overline{x_1 \vee x_2} = \overline{x_1} \cdot \overline{x_2},\ \overline{x_1 \cdot x_2} = \overline{x_1} \vee \overline{x_2}. \tag{11.13}$$

Правило де Моргана:

$$x_1 \vee x_2 = \overline{\overline{x_1} \cdot \overline{x_2}},\ x_1 \cdot x_2 = \overline{\overline{x_1} \vee \overline{x_2}}. \tag{11.14}$$

$$x_1 \vee \overline{x_1} = 1,\ x_1 \cdot \overline{x_1} = 0. \tag{11.15}$$

$$\overline{0} = 1,\ \overline{1} = 0. \tag{11.16}$$

$$x_1 \cdot (\overline{x_1} \vee x_2) = x_1 \cdot x_2,\ x_1 \vee \overline{x_1} \cdot x_2 = x_1 \vee x_2. \tag{11.17}$$

За допомогою співвідношень (11.4)–(11.17) можна виконувати тотожні перетворення логічних виразів і функцій для їх спрощення.

Виконуючи тотожні перетворення, слід дотримуватися ***послідовності*** у виконанні операцій: у першу чергу виконується операція заперечення, в другу — кон'юнкції, в третю — диз'юнкції. Якщо потрібно змінити послідовність виконання операцій, то застосовують дужки.

Функціонально повні системи елементарних функцій

Сукупність елементарних логічних функцій, якими можна записати або через які можна виразити логічну функцію довільного ступеня складності, утворюють ***функціонально повну систему елементарних функцій***. Операції логічного додавання, множення й заперечення складають функціонально повну систему логічних функцій. Це означає, що будь-яку логічну функцію довільної складності можна виразити через ці три логічні функції.

Крім розглянутих логічних функцій диз'юнкції, кон'юнкції й заперечення існують інші функціонально повні системи елементарних функцій.

Функція Шефера (штрих Шефера). Функцію Шефера для двох аргументів, яку позначають $y = x_1 \mid x_2$, можна задати таблицею істиності.

Таблиця 11.4

x_1	x_2	$y = x_1 \mid x_2$
0	0	1
0	1	1
1	0	1
1	1	0

Як випливає з таблиці істиності, функція Шефера має значення 0 тільки тоді, якщо обидва аргументи мають значення 1, у решті ж випадків вона має значення 1, тому цю функцію називають ***запереченням кон'юнкції***, тому що

$$y = x_1 \mid x_2 = \overline{x_1 \cdot x_2}. \qquad (11.18)$$

Функція Шефера складає функціонально повну систему елементарних логічних функцій, тобто будь-яку логічну функцію можна виразити через функцію Шефера. Для того, щоб довести це, достатньо виразити через функцію Шефера елементарні функції заперечення, кон'юнкції й диз'юнкції, що складають функціонально повну систему логічних функцій. Функція заперечення виражається через функцію Шефера так:

$$x_1 \mid x_1 = \overline{x_1 \cdot x_1} = \overline{x_1} \vee \overline{x_1} = \overline{x_1}. \qquad (11.19)$$

У перетвореннях (11.19) використовувалося правило де Моргана й властивість ідемпотентності.

Функція кон'юнкції виражається через функцію Шефера, використовуючи властивості (11.18), (11.10), (11.9),

$$(x_1 \mid x_2) \mid (x_1 \mid x_2) = \overline{(\overline{x_1 \cdot x_2}) \cdot (\overline{x_1 \cdot x_2})} =$$

$$= (\overline{\overline{x_1 \cdot x_2}}) \vee (\overline{\overline{x_1 \cdot x_2}}) = (x_1 \cdot x_2) \vee (x_1 \cdot x_2) = x_1 \cdot x_2. \qquad (11.20)$$

Функція диз'юнкції виражається через функцію Шефера, використовуючи властивості (11.19), (11.13), (11.10),

$$(x_1 \mid x_1) \mid (x_2 \mid x_2) = \overline{(\overline{x_1 \cdot x_1}) \cdot (\overline{x_2 \cdot x_2})} =$$

$$= (\overline{\overline{x_1 \cdot x_1}}) \vee (\overline{\overline{x_2 \cdot x_2}}) = \overline{\overline{x_1}} \vee \overline{\overline{x_2}} = x_1 \vee x_2. \qquad (11.21)$$

Функція Пірса (стрілка Пірса). Функція Пірса двох змінних, яка позначається $y = x_1 \downarrow x_2$, задається таблицею істиності:

Таблиця 11.5

x_1	x_2	$f(x_1; x_2) = x_1 \downarrow x_2$
0	0	1
0	1	0
1	0	0
1	1	0

Зіставляючи таблицю 11.1 і 11.5, бачимо, що стрілка Пірса є запереченням диз'юнкції, тобто

$$x_1 \downarrow x_2 = \overline{x_1 \vee x_2}. \qquad (11.22)$$

Стрілка Пірса, як і функція Шефера, складає функціонально повну систему елементарних логічних функцій, тобто можна будь-яку логічну функцію виразити через стрілку Пірса.

Спрощення логічних виразів

Логічний вираз або функцію можна представити в різних формах, використовуючи функціонально повну систему елементарних логічних функцій. У цифрових пристроях обробки інформації, зокрема в комп'ютері, логічні функції реалізуються за допомогою логічних елементів, тому для зменшення апаратурних затрат дуже важливо максимально спростити логічну функцію і знайти її форму з мінімальною кількістю операцій.

Довільний логічний вираз можна звести до канонічної форми — ***суми добутків*** змінних, причому в кожний добуток входить або сама змінна, або її заперечення. Така канонічна форма логічного виразу носить назву ***диз'юнктивна нормальна форма*** (***ДНФ***). Наведемо кілька прикладів диз'юнктивних нормальних форм логічних виразів:

$$y_1(x_1, x_2) = \overline{x_1} \cdot x_2 \vee x_1 \cdot \overline{x_2}. \qquad (11.23)$$

Таблиця 11.6

x_1	x_2	$y_1(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	0

$$y_2(x_1, x_2, x_3) =$$

$$= \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \vee \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot x_2 \cdot x_3. \qquad (11.7)$$

Таблиця 11.7

x_1	x_2	x_3	$y_2(x_1, x_1, x_1)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Другою канонічною формою, до якої можна звести будь-який логічний вираз, є ***добуток сум*** логічних змінних, причому в кожну суму входить або сама змінна, або її заперечення. Така канонічна форма логічного виразу носить назву ***кон'юнктивна нормальна форма*** (***КНФ***). Наведені вище логічні функції $y_1(x_1, x_2)$ $y_2(x_1, x_2, x_3)$ можна записати в кон'юнктивній нормальній формі

$$y_1(x_1, x_2) = (x_1 \vee x_2) \cdot (\overline{x_1} \vee \overline{x_2}), \qquad (11.8)$$

$$y_2(x_1, x_2, x_3) =$$

$$= (x_1 \vee \overline{x_2} \vee x_3) \cdot (x_1 \vee \overline{x_2} \vee \overline{x_3}) \cdot (\overline{x_1} \vee x_2 \vee x_3) \cdot (\overline{x_1} \vee \overline{x_2} \vee x_3). \qquad (11.9)$$

Можна показати, що для будь-якого логічного виразу існує тільки одна диз'юнктивна нормальна форма і тільки одна кон'юнктивна нормальна форма.

Спрощувати логічні вирази можна за допомогою послідовних тотожних логічних перетворень (11.4) ... (11.17). Наприклад, застосувавши до першого і другого, а також до третього й четвертого доданків функції $y_2(x_1, x_2, x_3)$ (11.7) операцію склеювання (11.12), отримуємо спрощену форму виразу

$$y_2(x_1, x_2, x_3) = \overline{x_1} \cdot \overline{x_2} \vee x_1 \cdot x_3. \qquad (11.10)$$

Крім тотожних перетворень, для спрощення логічних виразів, у яких кількість аргументів не перевищує шести, доцільно

застосовувати так звані карти Карно. ***Карти Карно*** — спеціальна таблиця істиності, рядки й стовпчики якої відповідають усім можливим комбінаціям значень не більше двох змінних, причому ці комбінації змінних розміщені в такій послідовності, що дві сусідні комбінації, і відповідно дві сусідні клітинки таблиці відрізняються значенням тільки однієї змінної. Клітинки, що розміщуються на краю таблиці, також вважаються сусідніми. В ті клітинки карти Карно, що відповідають доданкам диз'юнктивної нормальної форми, записуються одиниці, в решті ж клітинок — записуються нулі. Сусідні клітинки з одиницями об'єднують у блоки по 2, 4, 8, ... одиниць. Кожному блоку одиниць на карті Карно відповідає один доданок у диз'юнктивній формі функції, причому в блоках з двох одиниць кількість множників у доданку на одиницю менша кількості змінних, у блоках із чотирьох одиниць — на дві одиниці менша і т. д. Запишемо для прикладу карту Карно для функції $y_2(x_1, x_2, x_3)$ (11.7).

Таблиця 11.8

	$x_2\ \overline{x}_3$	x_2x_3	$\overline{x}_2x_3$	$\overline{x}_2\ \overline{x}_3$
x_1		1	1	
$\overline{x}_1$			1	1

Об'єднавши одиниці на карті Карно в два блоки, отримуємо спрощений вираз для функції $y_2(x_1, x_2, x_3)$,

$$y_2(x_1, x_2, x_3) = x_1 \cdot x_3 \vee \overline{x_1} \cdot \overline{x_2}. \qquad (11.11)$$

Порівнюючи вирази (11.10) і (11.11), отримані у результаті спрощення різними методами, бачимо, що вони однакові.

11.2 Логічні елементи

Логічні елементи — елементарні пристрої цифрової техніки, призначені для реалізації елементарних операцій над логічними змінними. Для реалізації операцій над логічними змінними, що можуть мати тільки два значення **0** і **1**, застосовують технічні пристрої, які можуть знаходитися в одному з двох станів. Такими технічними пристроями є різного роду ***ключові елементи***, що можуть перебувати в одному з двох станів: ***увімкнено*** й ***вимкнено***.

Ключові елементи можуть мати різну фізичну природу: пневматичний ключовий елемент — клапан на пневмопроводі, що може знаходитися у двох станах — закритому й відкритому, гідравлічний ключовий елемент — клапан на трубопроводі з рідиною, який також може бути або відкритий, або закритий. Відомі пристрої, що виконують доволі складні логічні операції, побудовані на основі пневматичних ключових елементів.

Однак найбільшого поширення набули логічні пристрої на основі електричних ключових елементів. Електричний ключ може бути механічної дії, коли електричне коло замикається чи розмикається вручну за допомогою електричного вимикача чи кнопки. Електричний ключ механічної дії застосовується для реалізації найпростіших логічних функцій на передніх панелях електричних пристроїв і вимірювальних приладів, задаючи різноманітні режими їх роботи.

Досконалішим електричним ключем є електромагнітне реле. До винайдення і широкого застосування електронних ключових елементів електромагнітні реле були основним елементом для реалізації пристроїв обробки цифрової інформації.

Види електронних логічних елементів

У наш час для реалізації логічних функції і операцій застосовуються пристрої, що побудовані на основі електронних ключових елементів: біполярних і польових транзисторах, які працюють у ключовому режимі.

Логічні змінні і функції можуть мати, як відомо, лише два значення — **0** і **1**. У технічних пристроях значення логічного **0** реалізується рівнем напруги від 0 В до 0,8 В, а значення логічної **1** — рівнем напруги 2,4 ... 5 В.

Розглянемо логічні елементи, що реалізують елементарні логічні операції.

Логічний елемент НІ (логічний інвертор). Логічний елемент **НІ** реалізує елементарну логічну операцію (функцію) заперечення, таблицею істиності якої є таблиця 11.3. На рисунку 11.1, *г* наведено умовне графічне позначення логічного елемента **НІ** згідно з державним стандартом. Логічний елемент **НІ** будується на основі біполярного (рис. 11.1, *а*) або уніполярного (польового) транзистора (рис. 11.1, *б*, *в*), що працює в ключовому режимі. Ключовий режим транзистора характеризується тим, що транзистор знаходиться у одному з двох станів: стані насичення або стані відсікання. Перехід з одного стану в інший здійснюється стрибком.

На рисунку 11.1 наведено логічний елемент **НІ**, що реалізує елементарну логічну функцію заперечення, на основі біполярного транзистора.

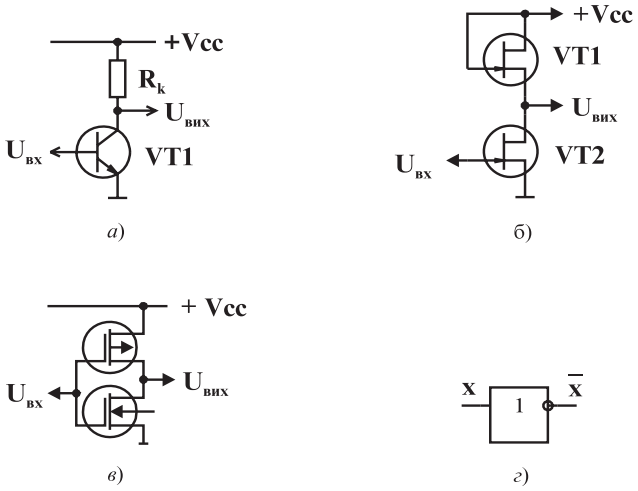


Рис. 11.1

Якщо на базу біполярного транзистора подати напругу 2,4 ... 5 В, що відповідає логічній 1, то транзистор стрибком перейде у стан ***насичення*** і вихідна напруга, що знімається з колектора транзистора, дорівнюватиме напрузі на двох відкритих *p-n* переходах, тобто 0,4 ... 0,7 В, що відповідає рівню логічного 0. Якщо ж на базу транзистора подати напругу 0 ... 0,7 В, то транзистор стрибком перейде у стан відсікання, і вихідна напруга дорівнюватиме напрузі живлення 5 В, тобто рівню логічної 1. Таким чином, транзистор інвертує вхідний сигнал, тобто змінює значення сигналу на протилежне, отже, такий пристрій виконує логічну операцію заперечення.

Логічний елемент І. Логічний елемент **І**, умовне графічне позначення якого згідно з державними стандартами наведено на рисунку 11.2, *б*, реалізує елементарну логічну операцію кон'юнкції або логічного множення. Логічний елемент **І** реалізується на послідовно ввімкнених ключах (рис. 11.2, *а*).

Вихідний сигнал послідовно ввімкнених ключів, що відповідає логічній 1, буде тільки в тому разі, якщо на обидва ключі подати сигнали, що відповідають рівню логічної 1 і вони обидва перейдуть у замкнений стан. Якщо хоч на один з ключів не подати сигнал, що відповідає логічній 1, то він залишиться у розімкненому стані, і вихідний сигнал дорівнюватиме рівню логічного 0.

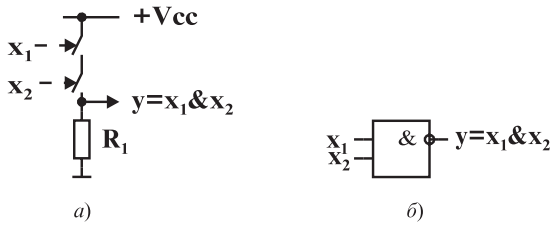


Рис. 11.2

Логічний елемент АБО. Логічний елемент **АБО**, умовне графічне позначення якого згідно з державними стандартами наведено на рисунку 11.3, *б*, реалізує елементарну логічну операцію диз'юнкції або логічного додавання. Логічний елемент **АБО** реалізується на паралельно ввімкнених ключах (рис. 11.3, *а*).

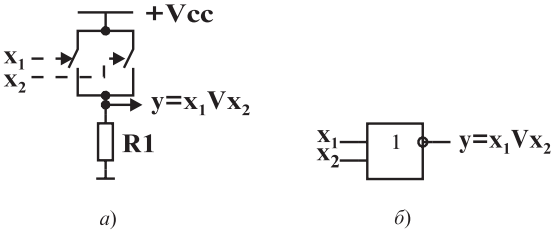


Рис. 11.3

Вихідний сигнал паралельно ввімкнених ключів, що відповідає логічному 0, буде тільки в тому випадку, якщо на обидва ключі подати сигнали, що відповідають рівню логічного 0, і вони обидва будуть розімкненими, і вихідний сигнал елемента дорівнюватиме рівню логічного 0. Якщо на один з ключів або на обидва разом подати сигнал, що відповідає логічній 1, то він замикається і на виході елемента встановиться рівень сигналу, що дорівнює логічній 1.

Мікросхеми логічних елементів

Логічні елементи реалізуються у вигляді мікросхем, що виготовляються на кристалах кремнію за допомогою новітніх високоефективних технологій.

Мікросхеми логічних елементів характеризуються рядом параметрів.

Швидкодія. Швидкодія логічного елемента визначається середньою затримкою сигналу, яка обчислюється як середнє арифметичне затримок увімкнення й вимкнення елемента.

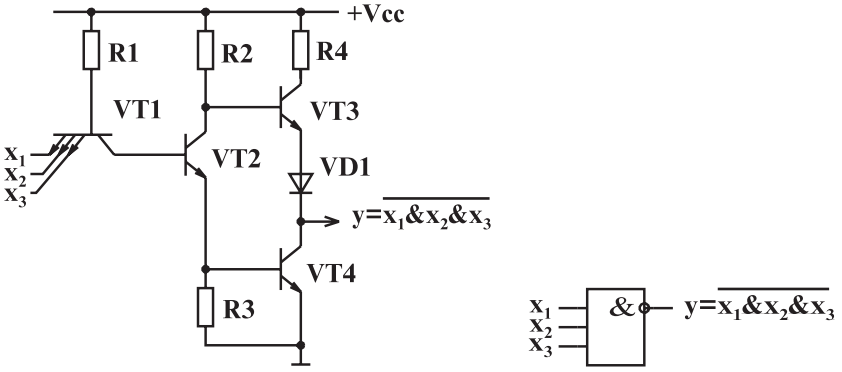
Споживана потужність. Споживана потужність визначається як середня арифметична потужність, що споживається логічним елементом в увімкненому й вимкненому станах. Швидкодія і споживана потужність пов'язані, як правило, прямою пропорційною залежністю: чим більша швидкодія, тим більша споживана потужність, тому для мікросхем логічних елементів часто застосовують таку характеристику, як добуток середньої споживаної потужності на середню затримку сигналів.

Коефіцієнт об'єднання за входами. Коефіцієнт об'єднання за входом — це максимальна кількість входів, яку може мати логічний елемент. Для більшості мікросхем логічних елементів коефіцієнт об'єднання за входами не перевищує восьми.

Коефіцієнт розгалуження за виходами. Коефіцієнт розгалуження за виходами або навантажувальна здатність визначається кількістю входів логічних елементів такого ж типу, які можуть бути приєднаними до виходу даної мікросхеми без порушення її працездатності.

Крім цих параметрів, мікросхеми логічних елементів характеризуються також завадостійкістю, надійністю, ступенем інтеграції тощо.

Транзисторно-транзисторні логічні елементи й пристрої. Транзисторно-транзисторні логічні елементи й пристрої (ТТЛ логіка) побудовані на основі багатоемітерного транзистора, що реалізує елемент І, та транзисторного інвертора (рис. 11.4, *а*).



а) *б*)

Рис. 11.4

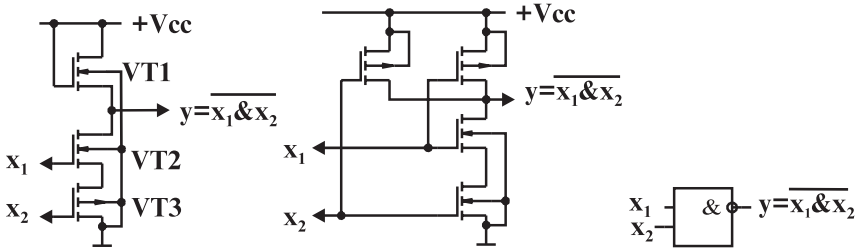
Якщо хоча б на один з емітерів транзистора *VT*1 подати потенціал <0,5 В, то транзистор відкриється і з'єднає базу транзистора *VT*2 з нульовою шиною живлення. Транзистор *VT*2 закриється,

база транзистора *VT*3 виявиться під високим потенціалом, що відкриє транзистор *VT*3, і на виході мікросхеми востановиться напруга високого рівня, що відповідає логічній 1. Таким чином, мікросхема в цілому виконує роль логічного елемента **І-НІ** і реалізує логічну функцію Шефера.

Щоб збільшити швидкодію ТТЛ — логічних елементів, під час виготовлення мікросхеми додатково створюють *p-n* переходи, або так звані діоди Шоткі, які запобігають режиму насичення транзисторів і тим самим зменшують час переходу транзистора з активного режиму в режим відсікання. Транзисторно-транзисторні логічні елементи з додатковими діодами Шоткі називають ТТЛШ логічними елементами або ТТЛШ логікою.

Мікросхеми логічних елементів на МДН-транзисторах. У порівнянні з біполярними транзисторами МДН-транзистори, тобто уніполярні транзистори, що виготовлені за структурою метал-діелектрик-напівпровідник, набагато менше споживають енергії, мають великий вхідний опір, тобто забезпечують велику навантажувальну здатність. Логічні елементи будуються на основі базової структури — інвертора, схема якого наведена на рисунку 11.1, *б*. Транзистор *VT*1, затвор якого з'єднаний зі стоком, відіграє роль резистора навантаження, а транзистор *VT*2 — роль ключа. Якщо на затвор транзистора *VT*2 подати низьку напругу, рівень якої відповідає логічному 0, то транзистор *VT*2 закривається, і на виході елемента буде висока напруга, рівень якої відповідає логічній 1. Якщо ж на затвор подати високу напругу, то транзистор *VT*2 відкриється, і вихід елемента з'єднається з нульовою шиною через відкритий транзистор.

На рисунку 11.5, *а* показано логічний елемент **І-НІ**, а на рисунку 11.6, *а* — **АБО-НІ**, що побудовані на основі базової структури інвертора на МДН-транзисторах.



а) *б*) *в*)

Рис. 11.5

Мікросхеми логічних елементів на комплементарних МДН-транзисторах. Комплементарними називають пару МДН-транзисторів, один з яких має канал з *n*-провідністю, а другий — з *p*-провідністю. Такі два послідовно з'єднані транзистори (рис. 11.1, *в*) утворюють інвертор. Якщо на затвори обох транзисторів подати високий потенціал, що відповідає логічній одиниці, то відкриється нижній транзистор *VT*2, а верхній *VT*1 — закриється, і вихід елемента через відкритий транзистор *VT*2 з'єднається з нульовою шиною живлення, тобто матиме рівень логічного 0. Якщо ж подати на затвори низький потенціал, то навпаки, відкриється верхній транзистор, а закриється нижній, і потенціал на виході матиме рівень логічної 1. У будь-якому разі один із комплементарних транзисторів завжди закритий, опір його становить сотні мегом, і струм через транзистори практично завжди дорівнює 0. Мікросхема споживає енергію тільки в момент перемикання, тому мікросхеми на комплементарних МДН-транзисторах (КМДН логічні мікросхеми) характеризуються надзвичайно малою споживаною потужністю. На рисунку 11.5, *б*, як приклад, наведено логічний елемент **І-НІ**, а на рисунку 11.6, *б* — **АБО-НІ**, що побудовані на основі комплементарних МДН-транзисторів.

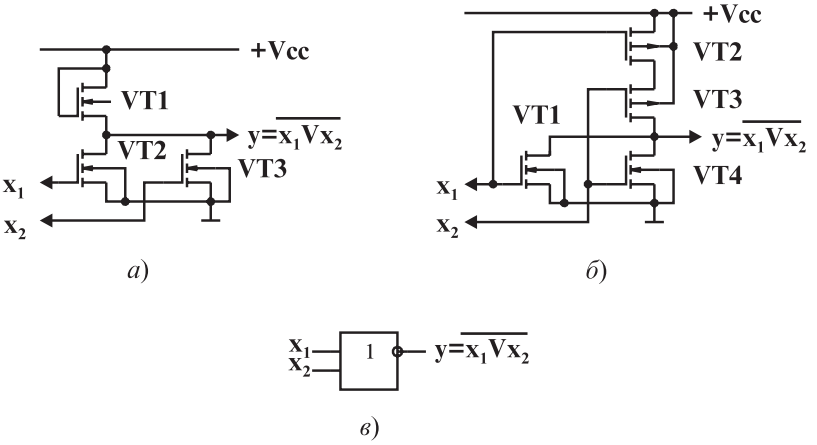


Рис. 11.6

Мікросхеми логічних елементів на емітерно-зв'язаних транзисторах (ЕЗЛ–елементи). Базовою структурою таких логічних елементів є диференціальний каскад (рис. 11.7), побудований на транзисторах *VT*1 і *VT*2. На базу *VT*2 подається опорна напруга, яка формується на транзисторі *VT*3. Емітерні повторювачі напруги на транзисторах *VT*4 і *VT*5 (рис. 11.8, *а*) призначені для узгодження рівнів сигналів.

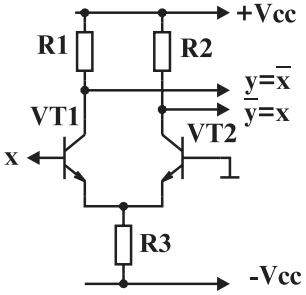
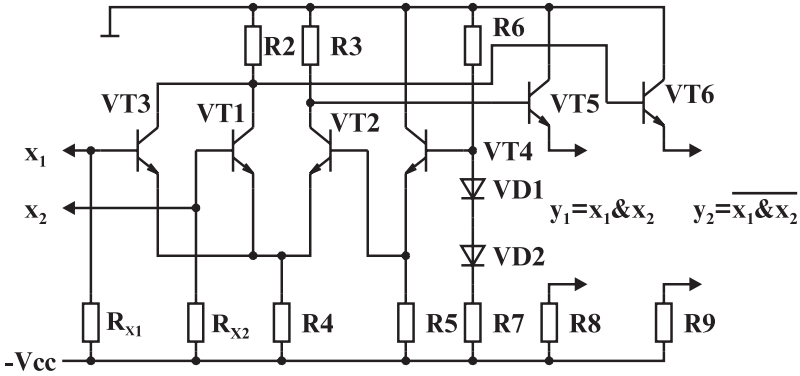
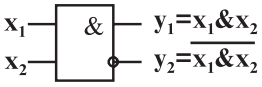


Рис. 11.7



а)



б)

Рис. 11.8

Особливістю ЕЗЛ–елементів є висока швидкодія, обумовлена тим, що транзистори диференціального каскаду ніколи не переходять у стан насичення. Якщо на вхід пристрою подати низький потенціал (близько −1,7 В), то через транзистор $VT2$ протікає струм I_0, на виході 1 буде потенціал низького рівня, а на виході 2 — високого (близько −0,7 В). Якщо ж на вхід пристрою подати високий потенціал, то транзистор $VT2$ практично закриється і струм I_0 протікатиме через транзистор $VT1$. Потенціали на виходах зміняться на протилежні.

На рисунку 11.8, *а,* як приклад, наведено логічний елемент на емітерно-зв'язаних емітерах, який має два входи і два виходи і може використовуватися як логічний двовходовий елемент **І** (вихід 1) або **І-НІ** (вихід 2), а на рисунку 11.8, *б* — умовне графічне зображення.

Мікросхеми логічних елементів на транзисторах з інжекційним живленням (І²Л–елементи). У логічних елементах, виготовлених у вигляді мікросхем за допомогою сучасних інтегральних технологій, широко застосовуються біполярні транзистори з інжекційним живленням. На відміну від звичайного біполярного транзистора транзистор з інжекційним живленням має додатковий електрод — інжектор, який вмикається до джерела струму. Транзистор з інжекційним живленням використовується як ключ при побудові логічних елементів. Якщо базу транзистора з'єднати з емітером, то такий ключ знаходитиметься в розімкнутому стані, якщо ж базу роз'єднати з емітером, то ключ перейде в замкнений стан. Ключ на основі такого транзистора має велику швидкодію і малі розміри. Швидкодія ключа збільшується зі збільшенням струму живлення, але при цьому зростає споживана потужність.

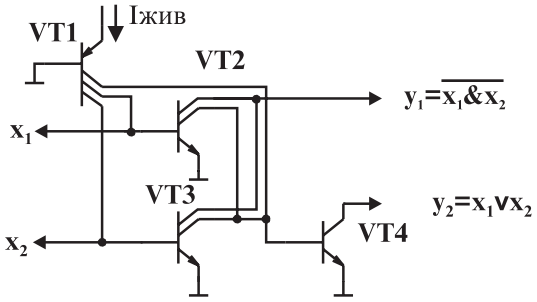


Рис. 11.9

На рисунку 11.9 наведено, як приклад, двовходовий елемент **І-НІ**, який виконано на основі біполярних транзисторів з інжекційним живленням.

11.3 Цифрові комбінаційні пристрої

Цифровими комбінаційними пристроями називаються електронні пристрої, які реалізують логічні функції, і вихідний сигнал яких залежить тільки від сигналів на входах цього пристрою

в даний момент часу. Цифрові комбінаційні пристрої використовуються як складові частини комп'ютерів, контролерів та інших цифрових пристроїв. Розглянемо комбінаційні цифрові пристрої, що широко застосовуються для обробки інформації.

Шифратори. Шифратори використовуються у системах переривання комп'ютерів, цифрових пристроях автоматичного керування, цифрових вимірювальних пристроях. Шифратори призначені для перетворення числа в одиничній позиційній системі числення у двійкове число. В одиничній позиційній системі числення натуральне число N зображується одиницею у N-ому розряді, в той час як у решти розрядів стоять нулі, наприклад, число 5 зображується так: 00001000, число 7 — 00000010.

На рисунку 11.10, як приклад, наведено шифратор 15-розрядного числа в одиничній позиційній системі числення у 4-розрядне двійкове число на 8-входових логічних елементах **АБО-НІ.**

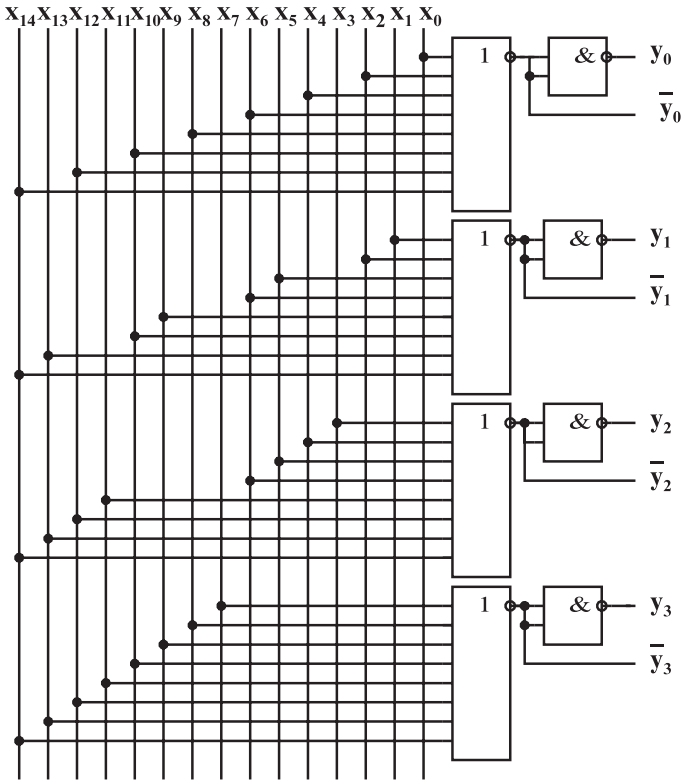


Рис. 11.10

На рисунку 11.11 наведено умовне графічне зображення шифратора згідно з державними стандартами.

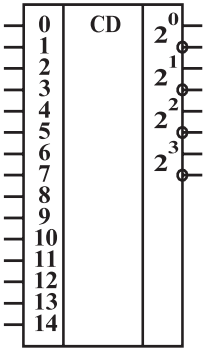
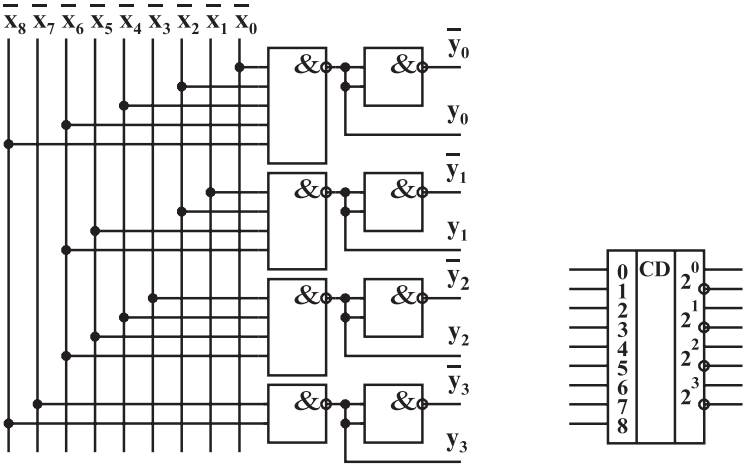


Рис. 11.11

Приклад шифратора на елементах **І-НІ** наведено на рисунку 11.12, *а*, а його умовне позначення згідно з державними стандартами на рисунку 11.12, *б*.

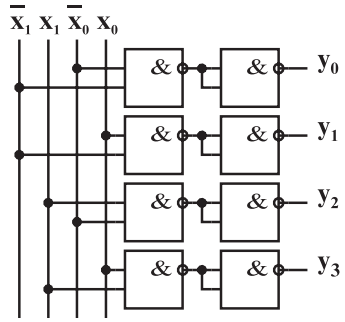


а) *б*)

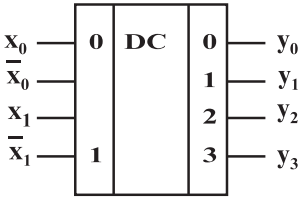
Рис. 11.12

Дешифратори. Дешифратори застосовуються у пристроях керування центрального процесора, у системах електронної пам'яті для операцій з адресами, у цифрових контролерах тощо. Дешифратори призначені для перетворення чисел із двійкової системи числення в одиничну позиційну систему числення. За структурою

дешифратори поділяються на матричні, каскадні і пірамідальні. На рисунку 11.13, *а* наведено дешифратор 2-розрядного двійкового числа на 2-входових логічних елементах **І-НІ**, а його умовне графічне позначення згідно з державними стандартами на рисунку 11.13, *б*.



а)



б)

Рис. 11.13

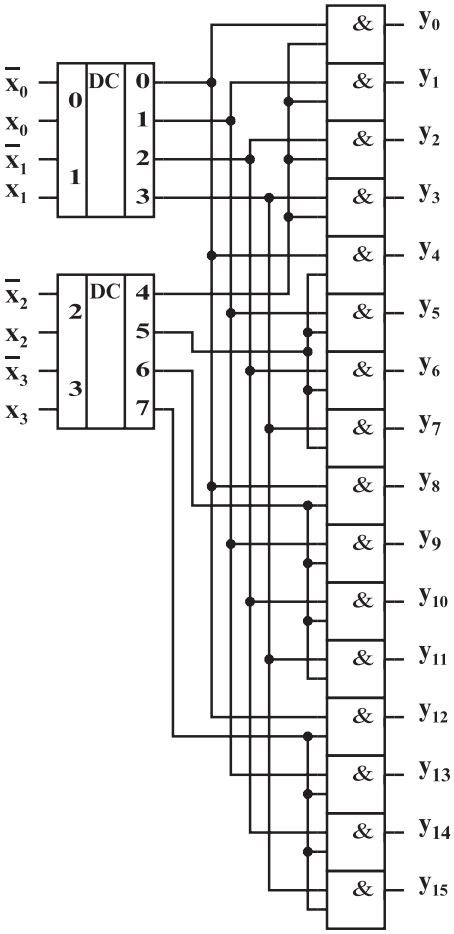


Рис. 11.14

Якщо кількість розрядів вхідного двійкового числа велика, то доцільніше дешифрувати це число каскадно, тобто поділити його на частини, дешифрувати кожну частину окремо, а потім результати дешифрації на першому етапі використати як вхідні дані для другого етапу дешифрації (рис. 11.14).

Пірамідальний дешифратор — це частковий варіант каскадного дешифратора, в якому на кожному етапі додається один двійковий

розряд. На рисунку 11.15 наведено пірамідальний дешифратор 3-розрядного двійкового числа.

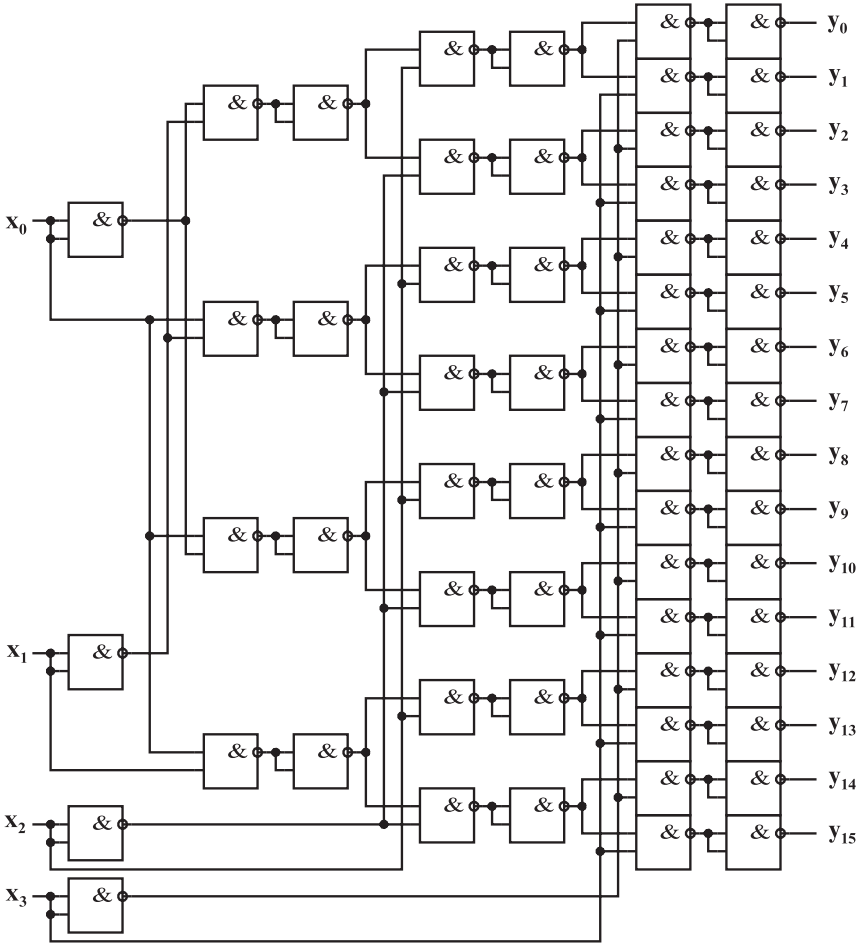


Рис. 11.15

Перетворювачі кодів. Шифратори й дешифратори є частковими випадками більш загального класу комбінаційних цифрових пристроїв — перетворювачів кодів. До перетворювачів кодів належить перетворювач двійкового позиційного коду в код семисегментного індикатора (рис. 11.16), який застосовується для відображення десяткових цифр у калькуляторах та інших цифрових

пристроях обробки інформації. Схему цього перетворювача реалізовано на елементах **І-НІ** і наведено на рисунку 11.17.

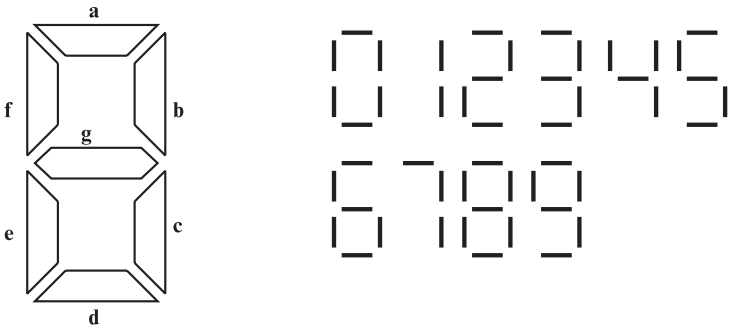


Рис. 11.16

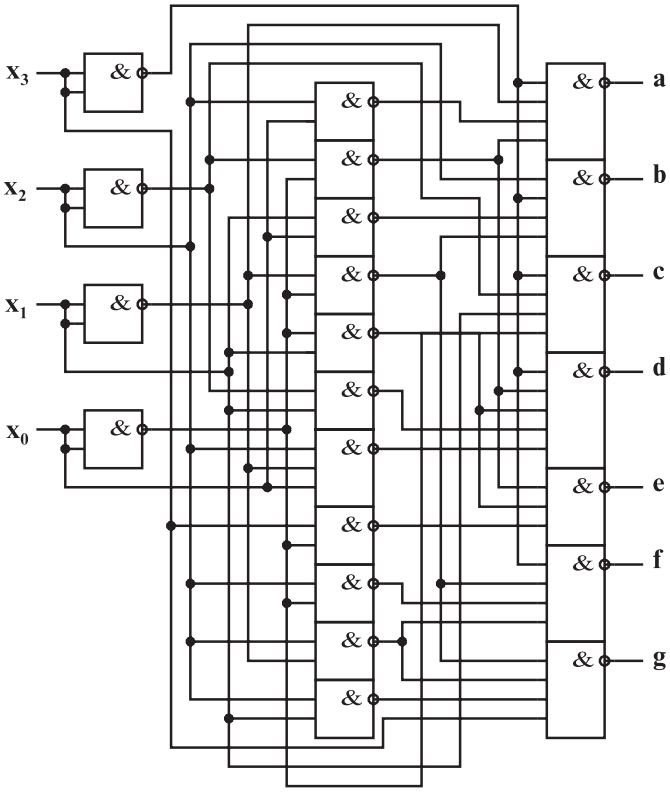


Рис. 11.17

У цифрових пристроях обробки інформації широко застосовується обернений код двійкового числа, який отримаємо, якщо в прямому коді двійкового числа нулі замінимо на одиниці, а одиниці на нулі. На рисунку 11.18 наведена схема перетворювача прямого коду двійкового числа в обернений.

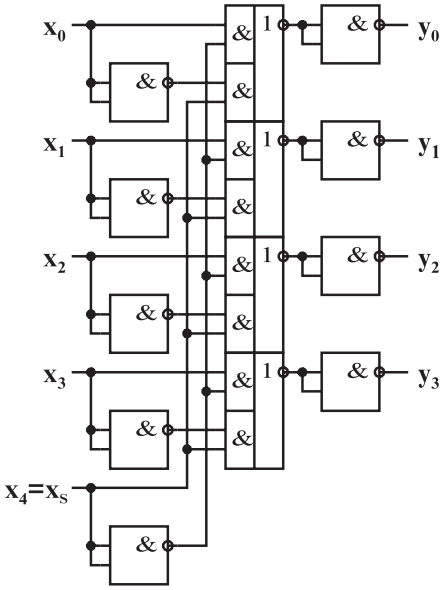


Рис. 11.18

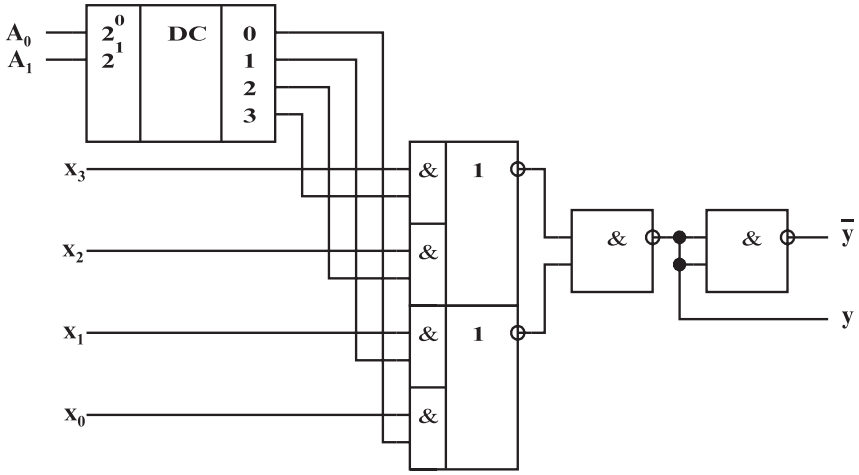


Рис. 11.19

Мультиплексори. Мультиплексор призначено для підключення на вхід каналу передачі інформації одного з вхідних сигналів. На рисунку 11.19 наведено як приклад мультиплексор одного з чотирьох сигналів x_0, x_1, x_2, x_3, на вхід y каналу передачі інформації, а на рисунку 11.20 — умовне графічне зображення мультиплексора згідно з державними стандартами.

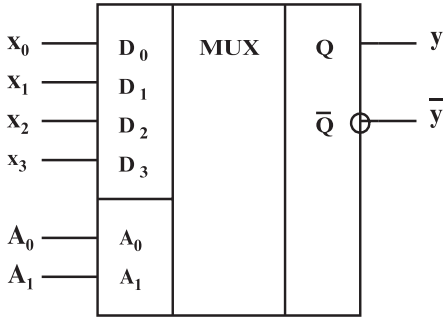
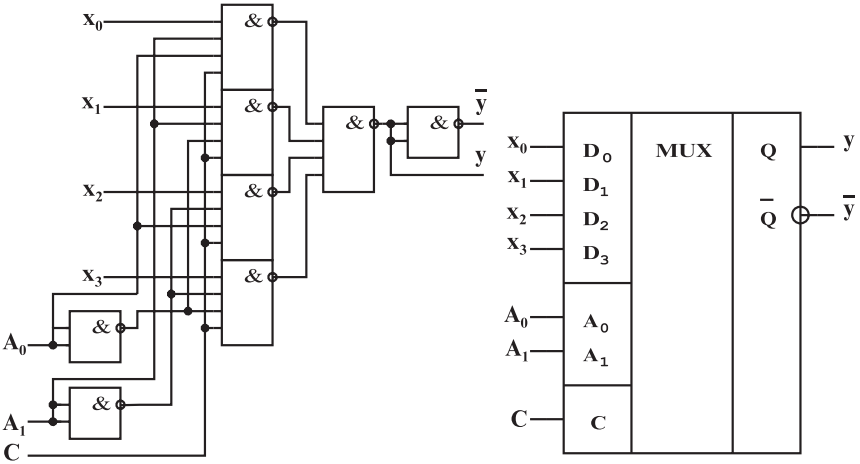


Рис. 11.20

Мультиплексор на чотири входи, що реалізований на елементах **І-НІ**, а також його умовне графічне позначення наведено на рисунку 11.21, даний мультиплексор синхронізований імпульсною послідовністю, що подається на вхід **С**.

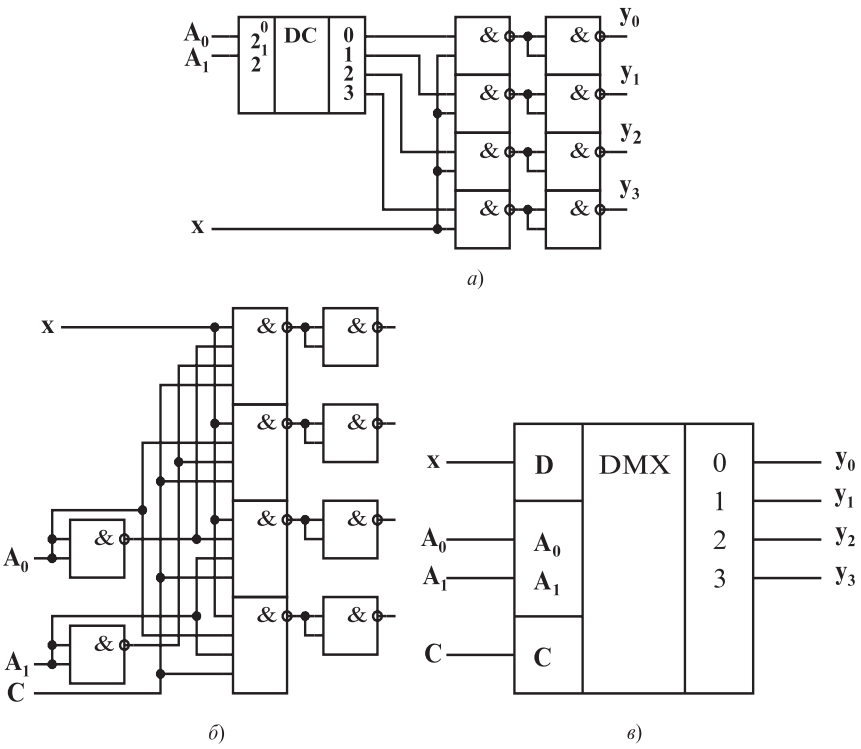


а) *б)*

Рис. 11.21

Демультиплексори. У деяких цифрових пристроях необхідно здійснювати операцію, обернену мультиплексуванню, тобто

демультиплексування сигналів. Ця операція полягає у підключенні одного сигналу на вхід одного з каналів передачі інформації. На рисунку 11.22, *а* наведено демультиплексор, який підключає сигнал *x* на вхід одного з чотирьох каналів передачі інформації. Номер каналу, що є його адресою, формується за допомогою дешифратора адреси. Для застосування демультиплексорів у синхронних пристроях вони мають синхронізуватися послідовністю імпульсів. Приклад такого синхронізованого демультиплексора, що реалізований на елементах **І-НІ**, наведено на рисунку 11.22, *б*, а його умовне графічне позначення — на рисунку 11.22, *в*.



а) *б*) *в*)

Рис. 11.22

Комбінаційні пристрої для зсуву даних. Зсув даних широко застосовується у різноманітних цифрових пристроях. Операцію зсуву на один розряд можна виконувати на регістрах зсуву. Якщо ж потрібно зсунути число на довільну кількість розрядів у довільному напрямі, то таку операцію доцільно виконувати за допомогою комбінаційних пристроїв зсуву на мультиплексорах (рис. 11.23, 11.24).

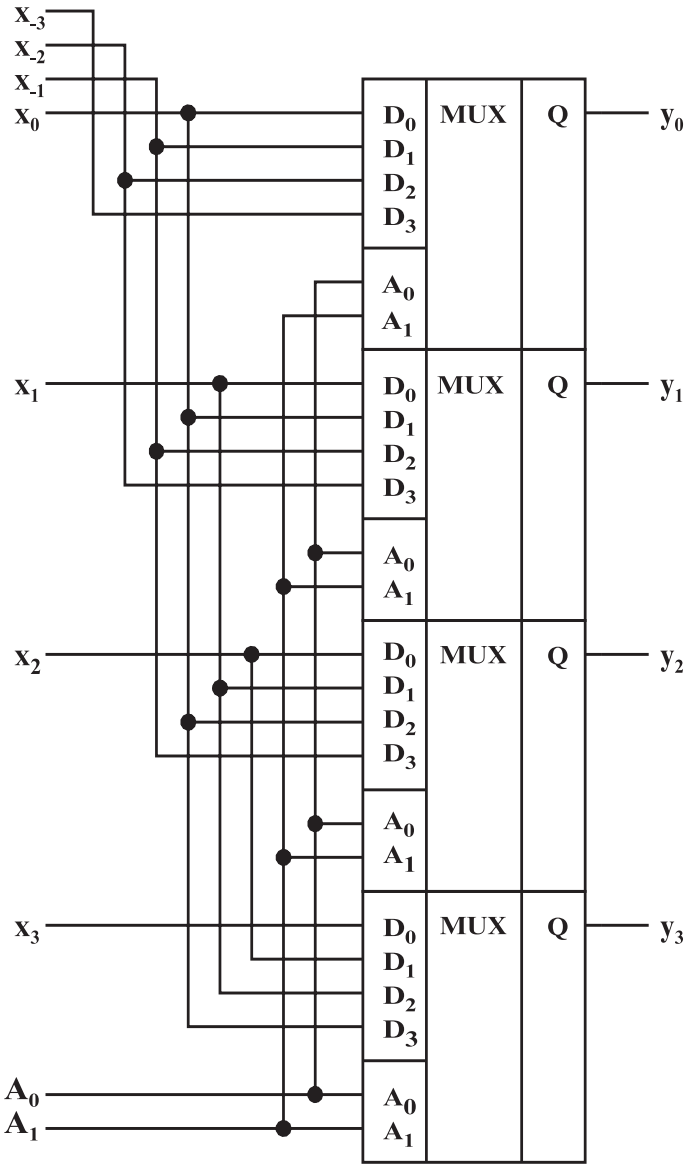


Рис. 11.23

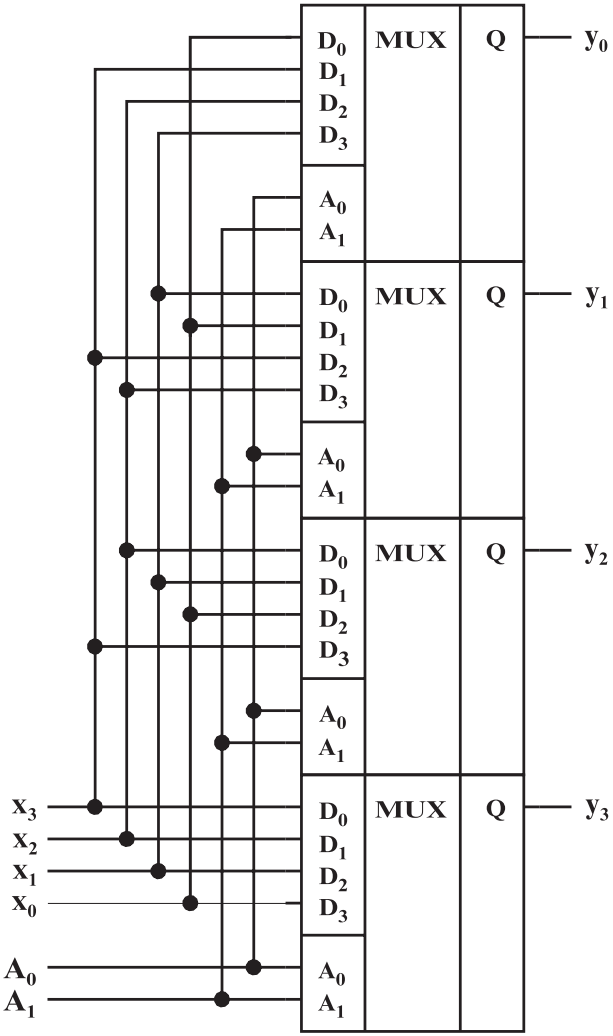


Рис. 11.24

Суматори. Суматори призначені для додавання двійкових чисел. Сума багаторозрядних чисел формується на основі однорозрядного додавання. Розглянемо додавання двох однорозрядних чисел, записаних у вигляді таблиці.

Таблиця 11.9

x_1	x_2	s	p
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

У таблиці 11.9 наведено два однорозрядні доданки x_1, x_2, сума s і перенос у старший розряд p. Для суми s можна записати такі логічні вирази

$$s = \overline{x_1} \cdot x_2 \vee x_1 \cdot \overline{x_2} = (x_1 \vee x_2) \cdot (\overline{x_1} \vee \overline{x_2}) =$$

$$= (x_1 \vee x_2) \cdot \overline{(x_1 \cdot x_2)}. \qquad (11.12)$$

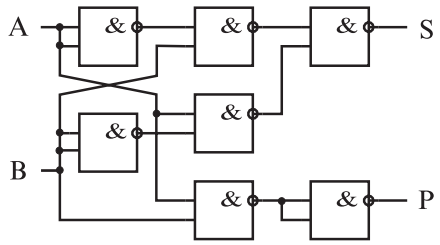
Для переносу p у старший розряд логічний вираз буде таким:

$$p = x_1 \cdot x_2, \qquad (11.13)$$

тобто перенос можна сформувати за допомогою операції диз'юнкції над однорозрядними доданками. Враховуючи вираз (11.13) для p, останній вираз у (11.12) можна записати так:

$$s = (x_1 \vee x_2) \cdot \overline{p}. \qquad (11.14)$$

Сума і перенос p у старший розряд відповідно до виразів (11.12) і (11.13) наведені на рисунку 11.25.



а)

A	B	S	P
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

S=$\overline{A}$*B+A*$\overline{B}$

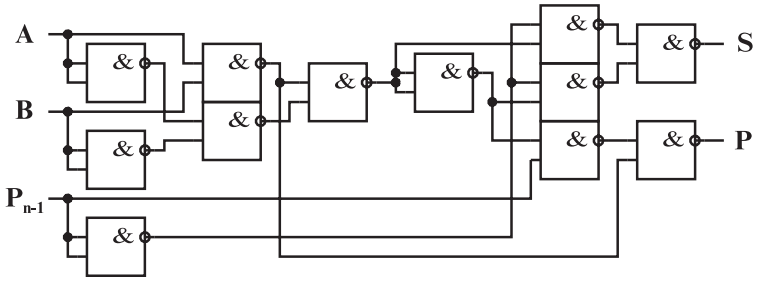
P=A*B

б)

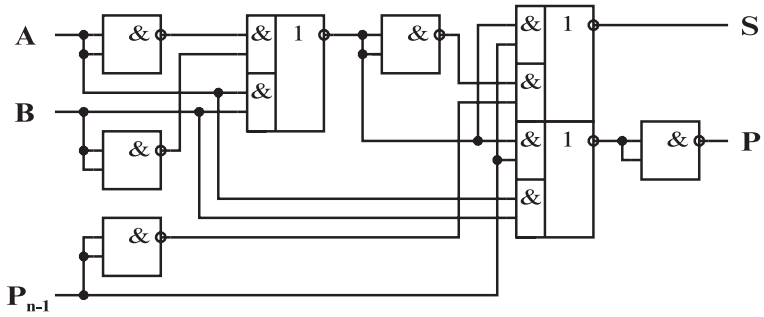
Рис. 11.25

Логічний пристрій, що реалізує обидві логічні функції s і c називається ***напівсуматором***. Напівсуматор, реалізований відповідно до виразів (11.13), (11.14), наведено на рисунку 11.25, *а*.

Під час додавання двох багаторозрядних двійкових чисел у кожному розряді крім двох однорозрядних доданків, необхідно додавати також перенос із попереднього розряду. Суматор, що здійснює цю операцію називається ***повним однорозрядним суматором***. Таблицю істиності для повного однорозрядного суматора, що здійснює додавання двох однорозрядних двійкових чисел A і B та переносу з попереднього молодшого розряду P_{n-1}, наведено на рисунку 11.26, *г*. Виходом однорозрядного двійкового повного суматора є однорозрядна сума і перенос у старший розряд P. Карти Карно для суми і переносу в старший розряд P наведено на рис. 11.26, *д*, *е*. На рисунку 11.26, *а*, *б*, *в* наведена реалізація повного суматора відповідно до наведених карт Карно на основі різних логічних елементів. Зображення повного однорозрядного суматора на схемах наведено на рисунку 11.26, *є*.

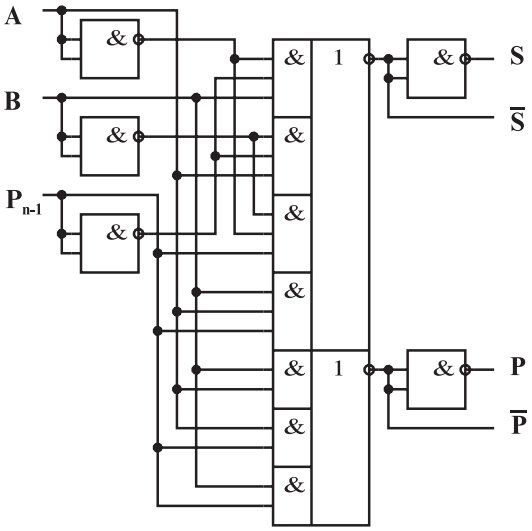


а)



б)

Рис. 11.26



в)

A	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
B	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**
P_{n-1}	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
S	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**1**
P	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**

г)

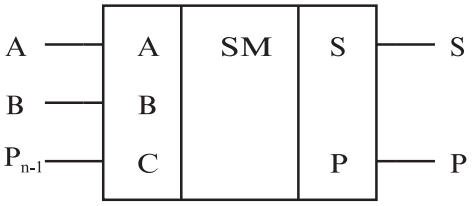
	$\overline{A}\,\overline{B}$	$\overline{A}\,B$	$A\,B$	$A\,\overline{B}$
$\overline{P}_{n-1}$	0	1	0	1
P_{n-1}	1	0	1	0

$S=\overline{A}*B*\overline{P}_{n-1}+A*\overline{B}*P_{n-1}+\overline{A}*\overline{B}*P_{n-1}+A*B*P_{n-1}=$
$=(\overline{A}*B+A*\overline{B})*\overline{P}_{n-1}+(\overline{A}*\overline{B}+A*B)*P_{n-1}$

	$\overline{A}\,\overline{B}$	$\overline{A}\,B$	$A\,B$	$A\,\overline{B}$
$\overline{P}_{n-1}$	0	0	1	0
P_{n-1}	0	1	1	1

$P=A*B*\overline{P}_{n-1}+\overline{A}*B*P_{n-1}+A*B*P_{n-1}+A*\overline{B}*P_{n-1}=$
$=A*B+A*P_{n-1}+B*P_{n-1}$

Рис. 11.26



є)

Рис. 11.26

Багаторозрядні суматори двійкових чисел будуються на основі повних однорозрядних суматорів, що зв'язані лініями переносу і виготовляються у вигляді однієї мікросхеми.

Додавати багаторозрядні двійкові числа можна за допомогою повного однорозрядного суматора послідовно, розряд за розрядом, починаючи з наймолодшого і закінчуючи найстаршим, запам'ятовуючи розряди суми в регістрі зсуву (див. наступний підрозділ), а перенос у старший розряд на спеціальному тригері переносу (рис. 11.27). Позитивною якістю такого послідовного суматора є незначні апаратурні затрати, а негативною — мала швидкодія.

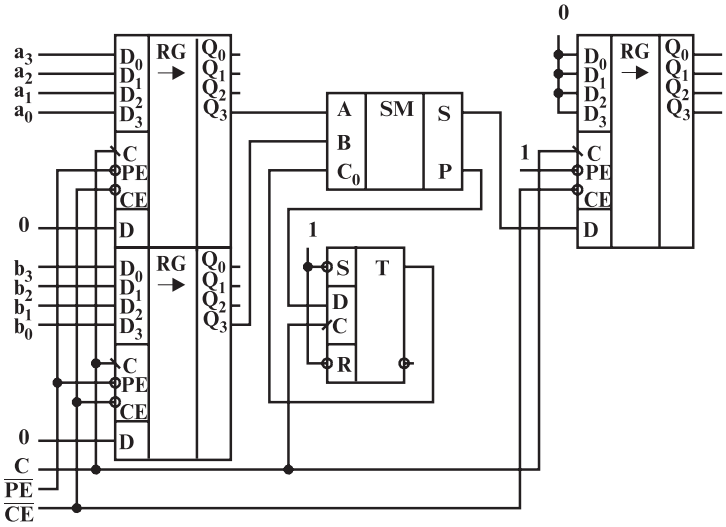


Рис. 11.27

Для зменшення тривалості операції додавання застосовують паралельний багаторозрядний суматор, у якому кількість однорозрядних суматорів дорівнює кількості розрядів (рис. 11.28).

Тривалість операції додавання визначається тривалістю поширення переносу з наймолодшого розряду до найстаршого і залежить від кількості розрядів. Такі паралельні суматори називаються суматорами з послідовним переносом.

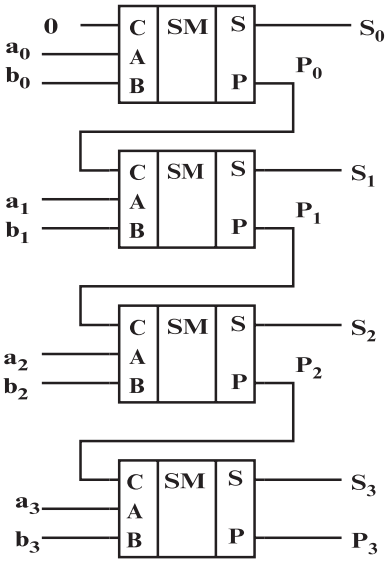
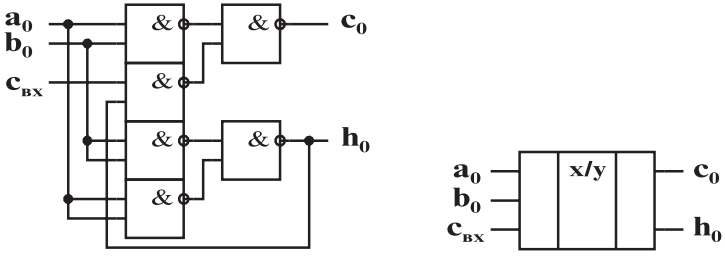


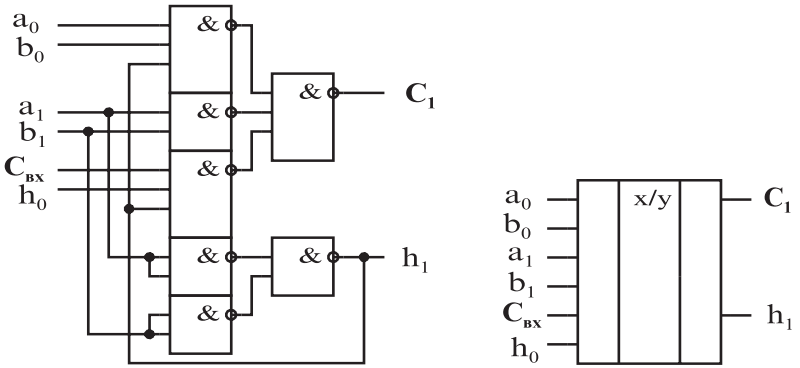
Рис. 11.28

Тривалість операції додавання можна істотно зменшити, якщо перенос поступатиме не послідовно, від одного розряду до іншого, а одночасно на всі розряди. Комбінаційні пристрої, що формують перенос на всі розряди паралельно, називаються ***схемами прискореного переносу***. Складність таких схем різко збільшується зі збільшенням номера розряду. На рисунку 11.29 наведено схеми прискореного переносу перших двох розрядів.



а) *б)*

Рис. 11.29



в) *г)*

Рис. 11.29

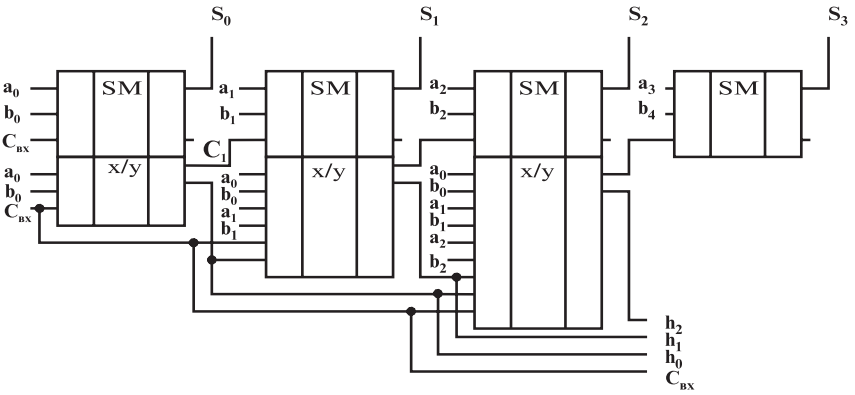


Рис. 11.30

На рисунку 11.30 наведено паралельний суматор із прискореним переносом.

Щоб досягти компромісу між зменшенням тривалості операції додавання й апаратурними затратами, що пов'язані зі схемами прискореного переносу, застосовують ***суматори з груповим переносом*** (рис. 11.31), в яких розряди суматора ділять на кілька груп. Для кожної групи застосовують схеми прискореного переносу, а між групами розрядів існує послідовний перенос.

Комбінаційні перемножувачі. Комбінаційні суматори завдяки великій швидкодії застосовуються в різноманітних цифрових пристроях обробки інформації, зокрема на їх основі будуються пристрої

множення двійкових чисел. Для перемножування двох чисел a і b можна число a додати з самим собою b разів за допомогою комбінаційного суматора. Тривалість операції множення у такому разі різко збільшується зі збільшенням кількості розрядів.

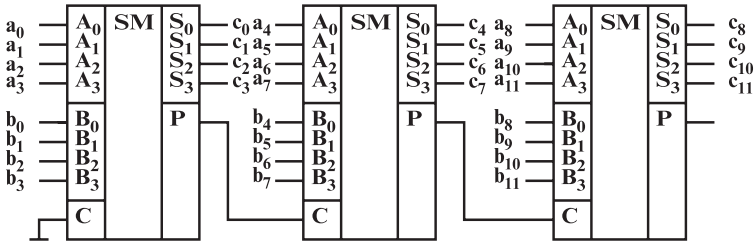


Рис. 11.31

Щоб зменшити тривалість операції множення можна використати додавання за допомогою комбінаційних суматорів часткових добутків двох чисел. На рисунку 11.32 наведено схему виконання операції множення через ***часткові добутки***, тобто добутки одного співмножника на розряди іншого.

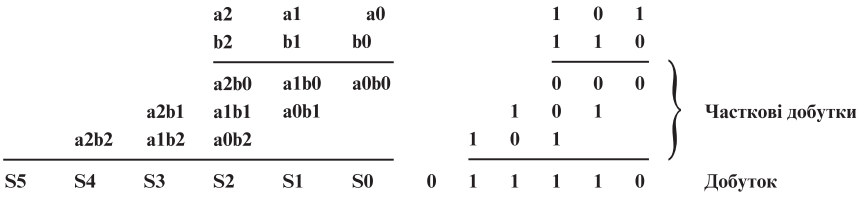


Рис. 11.32

Як видно зі схеми, частковий i-й добуток дорівнює множеному, якщо i-й розряд множника дорівнює 1, і дорівнює нулю, якщо i-й розряд множника дорівнює нулю. Кожний наступний частковий добуток зсувається на один розряд відносно попереднього. Добуток чисел визначається у результаті послідовного додавання часткових добутків. Схему перемножувача, що реалізує алгоритм додавання часткових добутків на комбінаційних суматорах, наведено на рисунку 11.33. Добутки окремих розрядів чисел отримані за допомогою схем **І**.

Основна позитивна якість такого комбінаційного перемножувача — велика швидкодія, що досягається ціною значних апаратурних затрат: для реалізації перемножувача m–розрядного множеного і m–розрядного множника потрібно $m \times n$ логічних елементів **І** і $m \times (n - 1)$ повних однорозрядних суматорів.

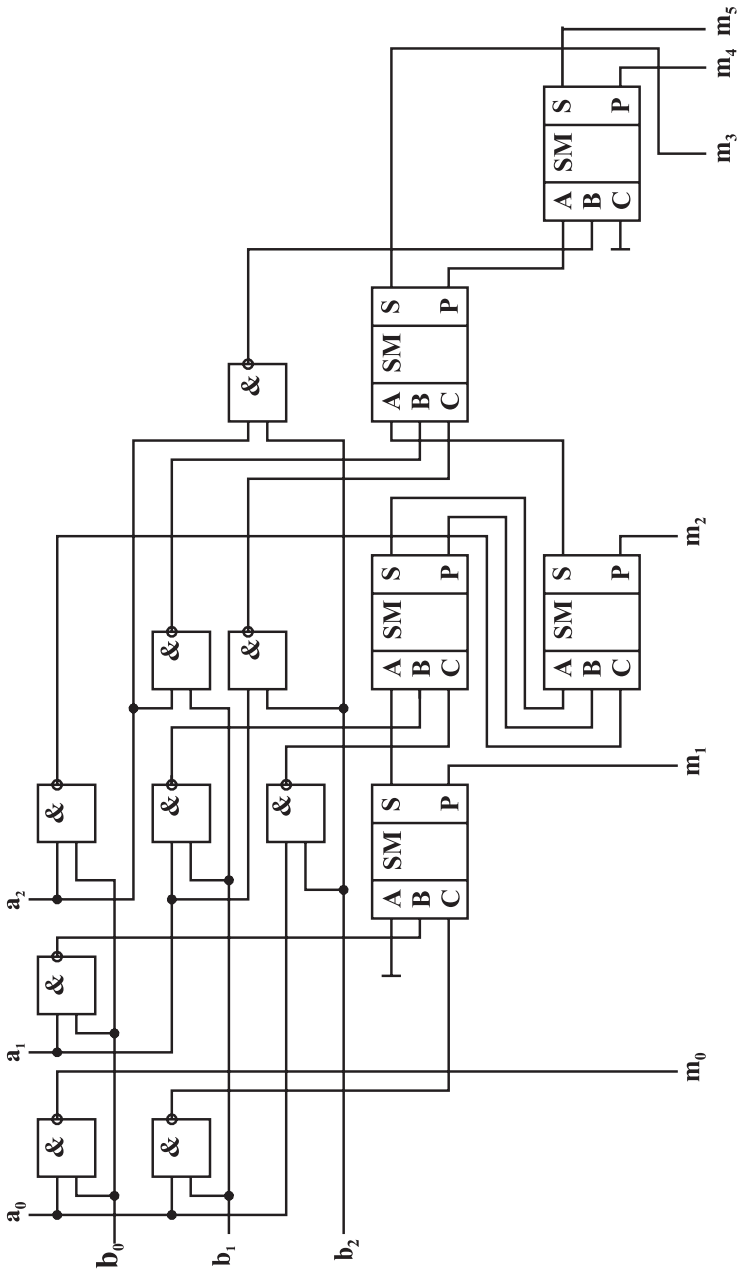


Рис. 11.33

11.4 Цифрові пристрої зі зворотними зв'язками

Цифрові пристрої зі зворотними зв'язками призначені для реалізації логічних функцій, значення яких залежить не тільки від значень аргументів у поточний момент часу, а й від значень аргументів у попередні моменти часу. Такі цифрові пристрої називають ще пристроями з пам'яттю. Розглянемо найбільш поширені пристрої.

Тригери

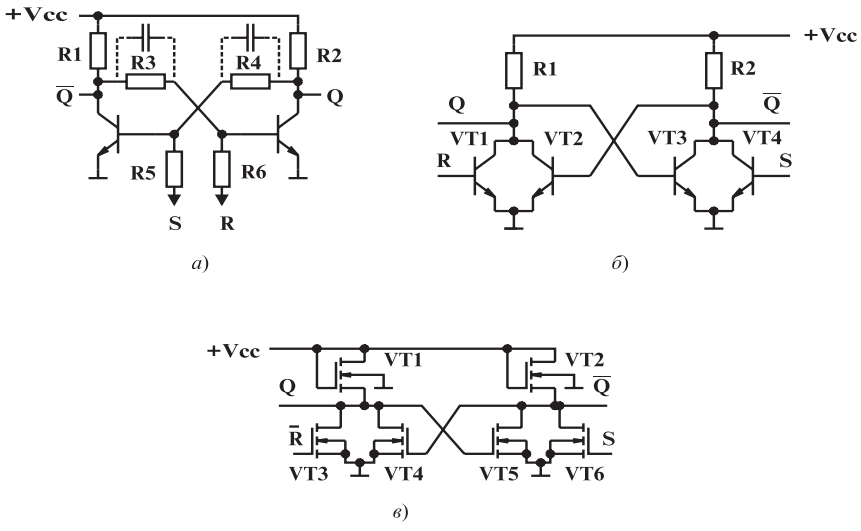
Тригер — це елементарний пристрій зі зворотними зв'язками. Тригер має два стійкі стани і може знаходитися в одному з них як завгодно довго. Переведення тригера з одного стану в інший здійснюється вхідними логічними сигналами. Одному стійкому стану відповідає логічна 1, а іншому — логічний 0. Таким чином, тригер може використовуватися для збереження однієї одиниці інформації — біта.

Застосовується кілька різновидів тригерів, які розрізняються за видом вхідних і вихідних сигналів, а також за способом запису інформації у тригер. За видом вхідних сигналів розрізняють тригери з імпульсним і потенціальним керуванням. За способом запису інформації тригери поділяються на ***асинхронні***, в яких запис інформації здійснюється в момент зміни інформаційного сигналу, і ***синхронні***, в яких запис інформації здійснюється у певні, визначені моменти дії спеціальних імпульсів синхронізації.

Загальна схема тригера (рис. 11.34) складається з елемента пам'яті і пристрою керування, який забезпечує керування тригером залежно від комбінації вхідних сигналів.

Рис. 11.34

Для побудови тригера застосовуються різні активні елементи, в основному біполярні й польові транзистори. Тригер складається з двох інверторів на транзисторах (рис. 11.35) з'єднаних так, що з виходу першого інвертора сигнал поступає на вхід другого, а з виходу другого — на вхід першого. Для тригерів використовуються ТТЛ-інвертори (рис. 11.35, *а*), інвертори на емітерно-зв'язаних транзисторах (рис. 11.35, *б*), інвертори на МДН-транзисторах (рис. 11.35, *в*).



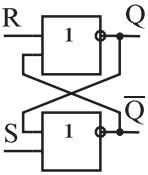
а) *б*) *в*)

Рис. 11.35

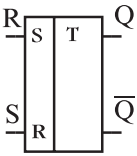
Розглянемо різновиди тригерів, що використовуються у цифрових пристроях обробки інформації.

RS-тригер. RS-тригер має два входи (S і R) і два виходи (Q і $\overline{Q}$) і може бути реалізований на двох логічних елементах **АБО-НІ** (рис. 11.36, *а*), або на двох логічних елементах **І-НІ** (рис. 11.37, *а*). Умовне графічне зображення RS-тригера на принципових схемах згідно з державними стандартами, наведено на рисунку 11.36, *б* і рисунку 11.37, *б*. Робота RS-тригера ілюструється картами Карно (рис.11.36, *в*, 11.37, *в*).

Робота тригера, що реалізований на логічних елементах **АБО-НІ** і зображений на рисунку 11.36, *а*, ілюструється часовими діаграмами зміни вхідних і вихідних сигналів (рис. 11.36, *г*). Передній фронт сигналу на вході S призводить до переходу вихідного сигналу $\overline{Q}$ з високого на низький рівень, а сигналу Q з низького до високого рівня. Якщо на вхід S приходить повторний імпульс,



а)

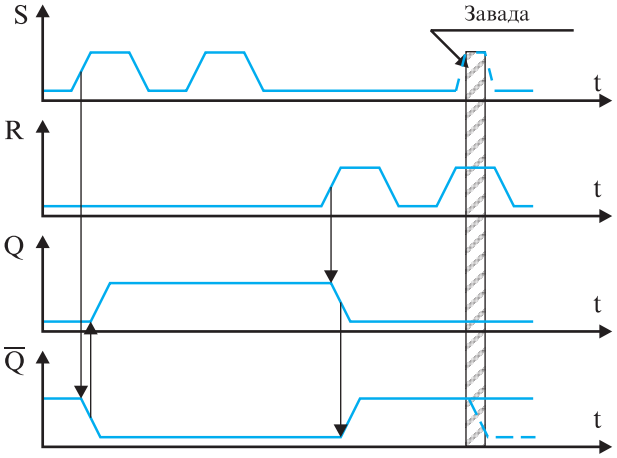


б)

	$\overline{R}\,\overline{S}$	$\overline{R}\,S$	$R\,\overline{S}$	$R\,S$
$\overline{Q}_{n-1}$	0	1	0	x
Q_{n-1}	1	1	0	x

x - невизначений стан

в)



г)

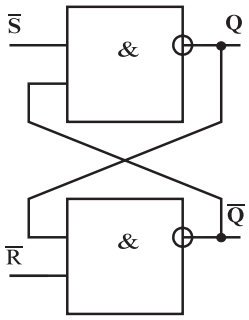
S	R	Q_{n-1}	Q_n
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

x - невизначений стан

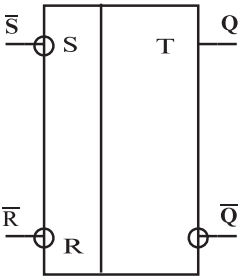
д)

Рис. 11.36

то це не змінює стану тригера. Якщо тригер знаходиться в стані збереження 1, то дія на вході R імпульсу, призводить до переходу сигналу Q з високого рівня у низький.



а)

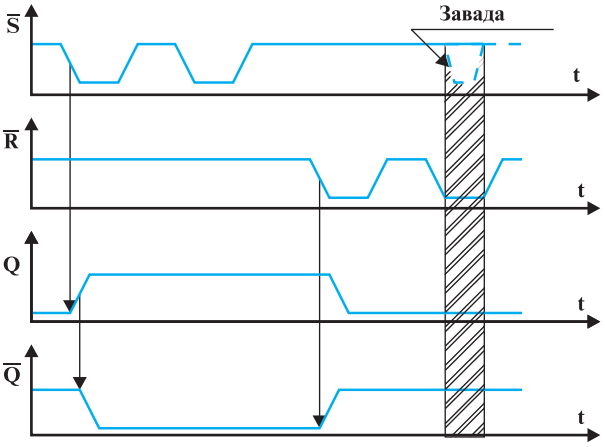


б)

	R S	R $\overline{S}$	$\overline{R}$ S	$\overline{R}$ $\overline{S}$
$\overline{Q}_{n-1}$	0	1	0	x
Q_{n-1}	1	1	0	x

x - невизначений стан

в)



г)

Рис. 11.37

$\overline{S}$	$\overline{R}$	Q_{n-1}	Q_n
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	1
0	0	1	x
0	0	0	x

x - невизначений стан

д)

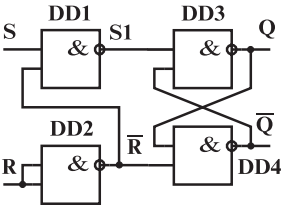
Рис. 11.37

Керівними вхідними сигналами для тригера на основі елементів **І-НІ** (рис. 11.37, *а*) є сигнали низького рівня, тобто рівня логічного 0. Якщо на вхід *S* (від англ. Set — установка) подати сигнал низького рівня (рівня логічного 0), то на виході *Q* RS-тригера встановиться 1, а на виході $\overline{Q}$ встановиться 0. Якщо ж сигнал низького рівня (рівня логічного 0) подати на вхід *R* (від англ. Reset — скидання, повернення до початкового стану), то на виході *Q* RS-тригера встановиться 0, а на виході $\overline{Q}$ встановиться 1. Одночасна подача сигналів низького рівня на обидва входи *S* і *R* заборонена, тому що стан тригера буде невизначений. RS-тригер (рис. 11.37, *а*) зберігає попередній стан, якщо на обидва входи *S* і *R* подати сигнали високого рівня (рівня логічної 1). У цьому режимі RS-тригер відіграє роль елементарного пристрою пам'яті, призначеного для зберігання одного біта інформації.

На часових діаграмах заштриховано інтервали дії на входи тригера заборонених комбінацій. Стан, у який переходить тригер пам'ять після закінчення дії забороненої комбінації, залежить від того, який із сигналів триває довше.

Для усунення неоднозначності реакції RS-тригера на заборонену комбінацію сигналів, необхідно встановити стан, у який тригер має перейти. За реакцією на заборонену комбінацію вхідних сигналів розрізняють R-тригери, що переходять у нульовий стан, S-тригери, які переходять у одиничний стан, і E-тригери, що зберігають попередній стан.

R-тригер на елементах **І-НІ** наведено на рисунку 11.38, *а*, а його карту Карно на рисунку 11.38, *б*. Якщо на вхід такого тригера поступає заборонена комбінація S = R = 1, то через те, що на вхід логічного елемента DD1 поступає інверсний сигнал $\overline{R}$, вихід елемента DD1 незалежно від сигналу S встановлюється на 1. Таким чином, комбінація сигналів S = R = 1 встановлює тригер на нуль. Часові діаграми (рис. 11.38, *в*) ілюструють співвідношення між сигналами.

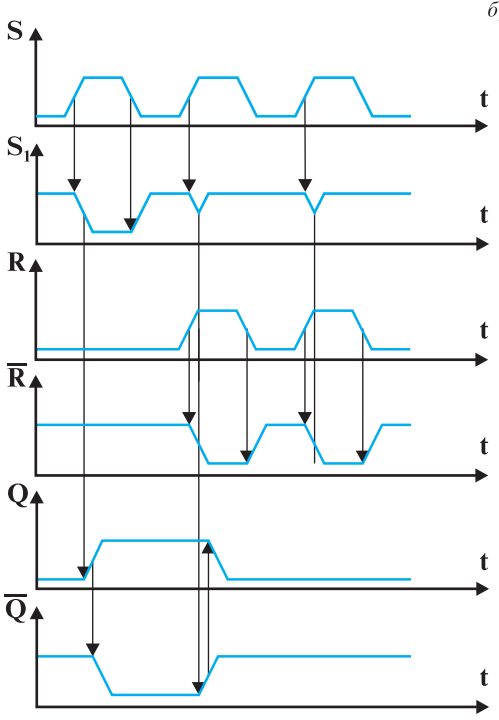


а)

	$\overline{R}\,\overline{S}$	$\overline{R}\,S$	$R\,\overline{S}$	$R\,S$
$\overline{Q}_{n-1}$	0	1	0	0
Q_{n-1}	1	1	0	0

$$Q_n=\overline{R}*S+\overline{R}*Q_{n-1}=$$
$$=\overline{\overline{\overline{R}*S}*\overline{\overline{R}*Q_{n-1}}}$$

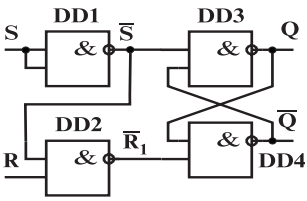
б)



в)

Рис. 11.38

На рисунку 11.39, *а* наведено схему S-тригера на елементах **I-HI**, а на рисунку 11.39, *б* — його карту Карно. Якщо на вхід такого тригера поступає заборонена комбінація S = R = 1, то через те, що на вхід логічного елемента DD2 поступає інверсний сигнал $\overline{S}$, вихід елемента DD2 незалежно від сигналу R встановлюється на 1. Таким чином, комбінація сигналів S = R = 1 встановлює тригер на одиницю. Часові діаграми (рис. 11.39, *в*) ілюструють співвідношення між сигналами і принцип дії тригера.



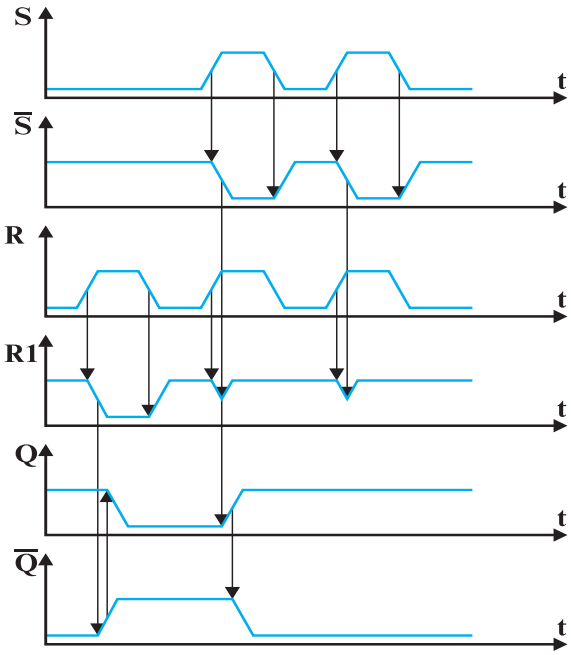
а)

	$\overline{R}\,\overline{S}$	$\overline{R}\,S$	$R\,S$	$R\,\overline{S}$
$\overline{Q}_{n-1}$	0	1	0	0
Q_{n-1}	1	1	1	0

$$Q_n=\overline{R}*S+\overline{R}*Q_{n-1}+S*Q_{n-1}=$$

$$=\underbrace{\overline{\overline{R}*S}}_{S_1}*Q_{n-1}*\underbrace{\overline{R*\overline{S}}}_{R_1}=\overline{S_1*Q_{n-1}*R_1}$$

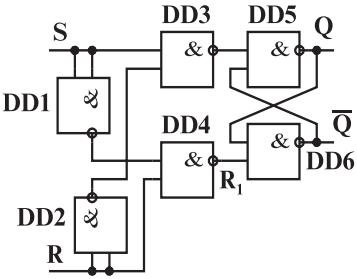
б)



в)

Рис. 11.39

Е-тригер на елементах **І-НІ** наведено на рисунку 11.40, *а*. Перехід тригера з одного стану в інший під дією вхідних сигналів ілюструє карта Карно (рис. 11.40, *б*) і часові діаграми (рис. 11.40, *в*). Якщо на вхід тригера поступає комбінація сигналів S = R = 1, то, інвертуючись на елементах DD1, DD2, сигнали поступають на входи елементів DD1, DD2. У такому разі їх вихідні сигнали S_1, R_1 дорівнюють одиниці і вихідний тригер на елементах DD5, DD6 зберігає попередній стан.

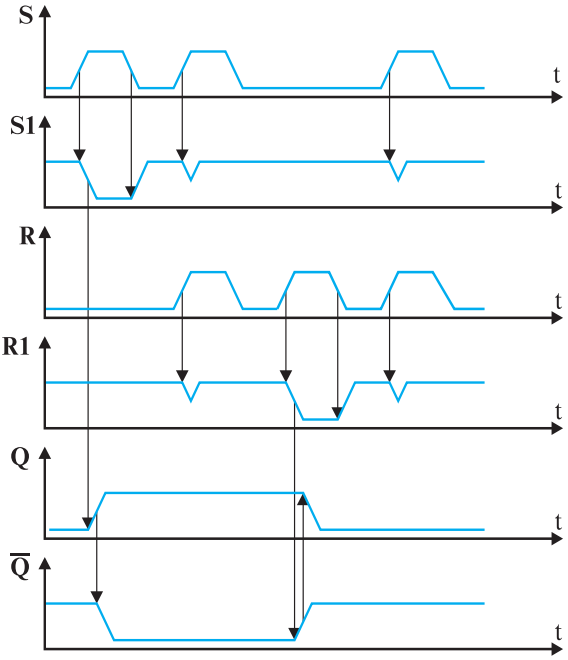


а)

	$\overline{R}\,\overline{S}$	$\overline{R}\,S$	$R\,S$	$R\,\overline{S}$
$\overline{Q}_{n-1}$	0	1	1	0
Q_{n-1}	1	1	1	0

$$Q_n = S + \overline{R} * Q_{n-1} = \overline{\overline{S} * \overline{\overline{R} * Q_{n-1}}}$$

б)



в)

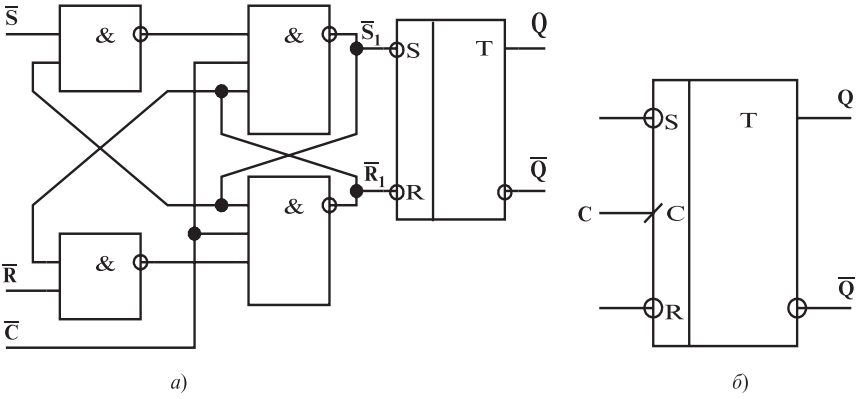
Рис. 11.40

Синхронізований RS-тригер. Завади на входах тригерів за певних умов можуть бути причиною хибних спрацювань тригерів. Для зменшення ймовірності хибних спрацювань доцільно розглянуті тригери синхронізувати послідовністю імпульсів, які слід подавати на додаткові входи логічних елементів.

Крім зменшення залежності від завад, синхронізація тригера застосовується для його роботи у складі синхронізованих цифрових пристроїв обробки інформації.

У більшості цифрових пристроїв необхідно, щоб RS-тригер змінював свій стан не в довільні, а в цілком визначені моменти часу. Щоб задати такі моменти часу, в цифрових пристроях обробки інформації використовують спеціальну послідовність так званих ***імпульсів синхронізації*** (англ. *clock*) або ***тактових імпульсів***. Імпульси синхронізації подаються через такий інтервал часу (період синхронізації), щоб закінчилися перехідні процеси переходу тригерів та інших цифрових пристроїв з одного стану в інший. Такі процеси можуть синхронізуватися як фронтом імпульсу синхронізації, так і спадом цього імпульсу. Застосування імпульсів синхронізації усуває неоднозначність стану тригерів і різко збільшує надійність їх роботи.

На рисунку 11.41, *а* наведено схему синхронного RS-тригера на елементах **І-НІ**, а на рисунку 11.41, *б* — його умовне графічне позначення.



а) *б*)

Рис. 11.41

Часові діаграми, що ілюструють роботу тригера, наведено на рисунку 11.42.

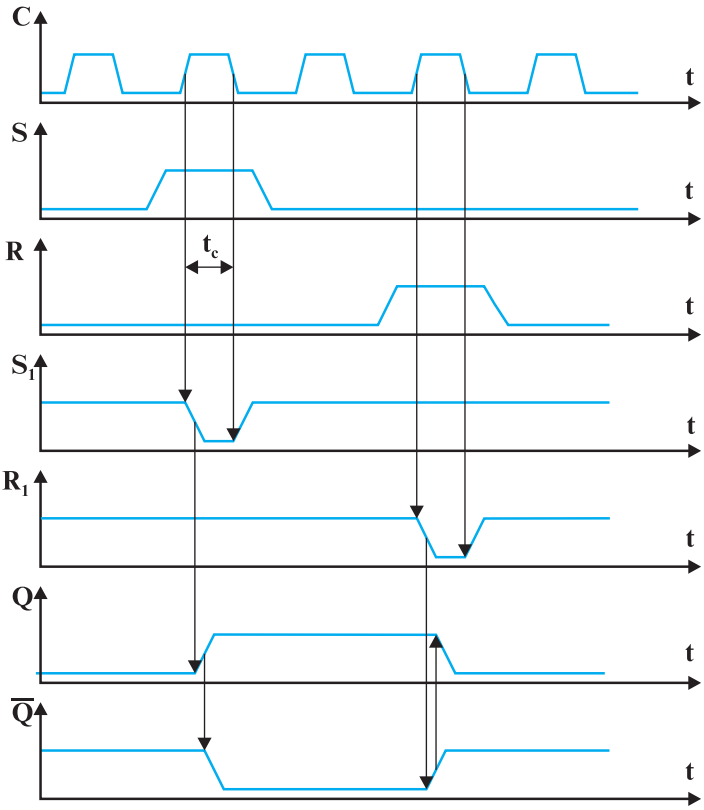
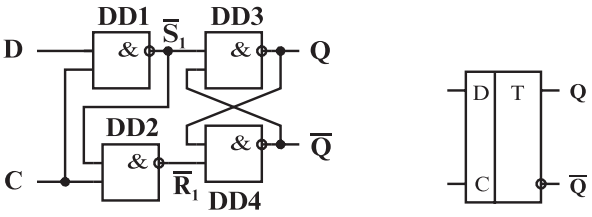


Рис. 11.42

D-тригер. D-тригер має один інформаційний вхід (D-вхід) і один вхід синхронізації (C-вхід) і два виходи: прямий (Q) та інверсний ($\overline{Q}$) (рис. 11.43, *а*). У моменти дії на вході синхронізації C рівня або фронту імпульсу здійснюється запис у тригер інформації, яка в цей момент поступає на інформаційний вхід (D-вхід). D-тригер називають також тригером із затримкою (від англ. Delay — затримка), тому що інформація на прямому виході тригера з'являється із затримкою на один період імпульсів синхронізації (рис. 11.43, *г*). Умовне графічне зображення D-тригера згідно з державними стандартами наведено на рисунку 11.43, *б*.

Робота D-тригера пояснюється картою Карно (рис. 11.43, *в*) і часовими діаграмами вхідних і вихідних сигналів (рис. 11.43, *г*). Якщо на вході синхронізації діє сигнал низького рівня (логічний 0), то на виході елементів DD1, DD2 (сигнали $\overline{S}_1$, $\overline{R}_1$) буде сигнал високого рівня незалежно від сигналу на інформаційному



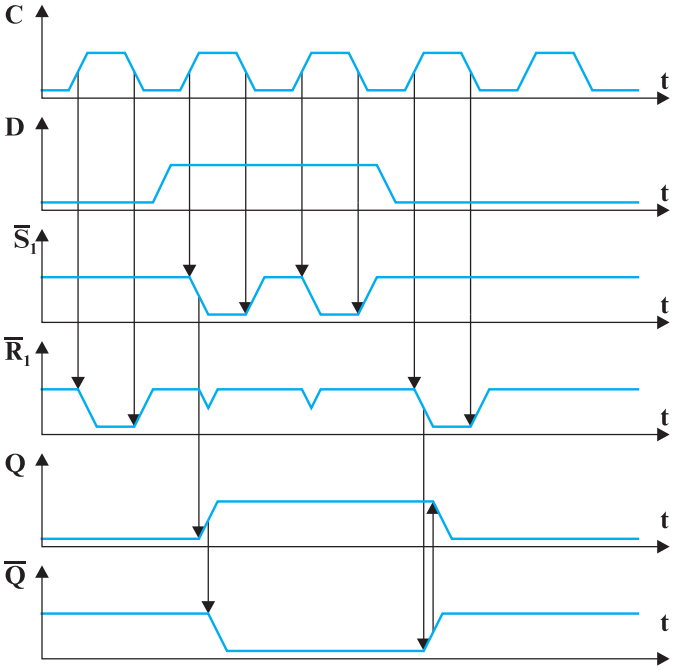
а) *б*)

	$\overline{C}\,\overline{D}$	$\overline{C}\,D$	$C\,D$	$C\,\overline{D}$
$\overline{Q}_{n-1}$	0	0	1	
Q_{n-1}	1	1	1	

$$Q_n = C\,D + \overline{C}\,Q_{n-1} =$$

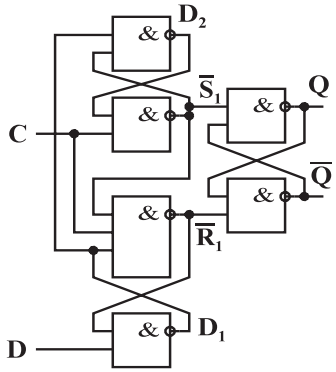
$$= \overline{\overline{C\,D} * \overline{\overline{C}\,Q_{n-1}}}$$

в)

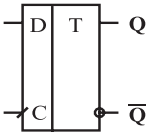


г)

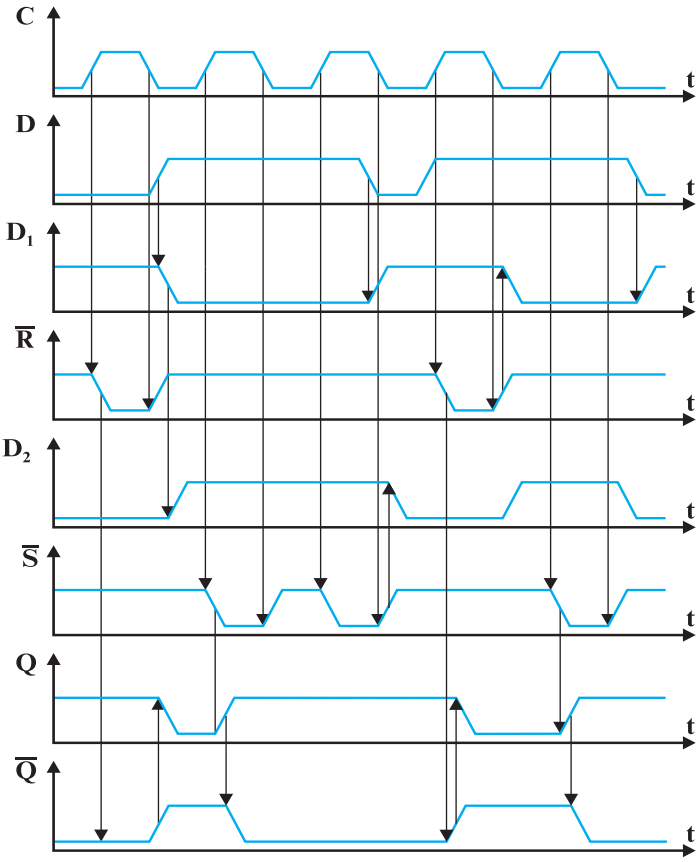
Рис. 11.43



а)



б)



в)

Рис. 11.44

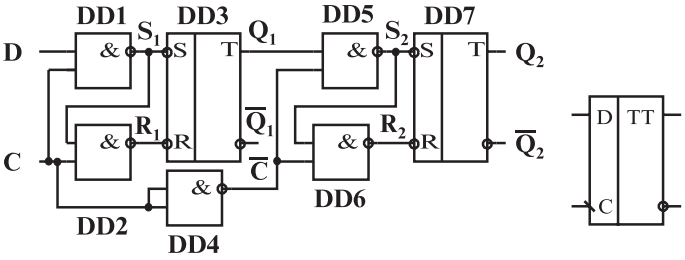
вході і тригер на логічних елементах DD3, DD4 перебуває у стані збереження інформації. Якщо ж сигнал синхронізації має високий рівень (рівень логічної 1), то в тригер на логічних елементах DD3, DD4 записується інформація, що присутня на його інформаційному вході D.

Щоб забезпечити надійне функціонування D-тригера необхідно зменшити ймовірність зміни стану тригера за D-входом (так зване наскрізне керування за D-входом) під час дії імпульсу синхронізації. Це досягається застосуванням синхронізації D-тригера фронтом імпульсу. D-тригер, синхронізований фронтом імпульсу, наведено на рисунку 11.44, *а*, а його умовне графічне позначення — на рисунку 11.44, *б*.

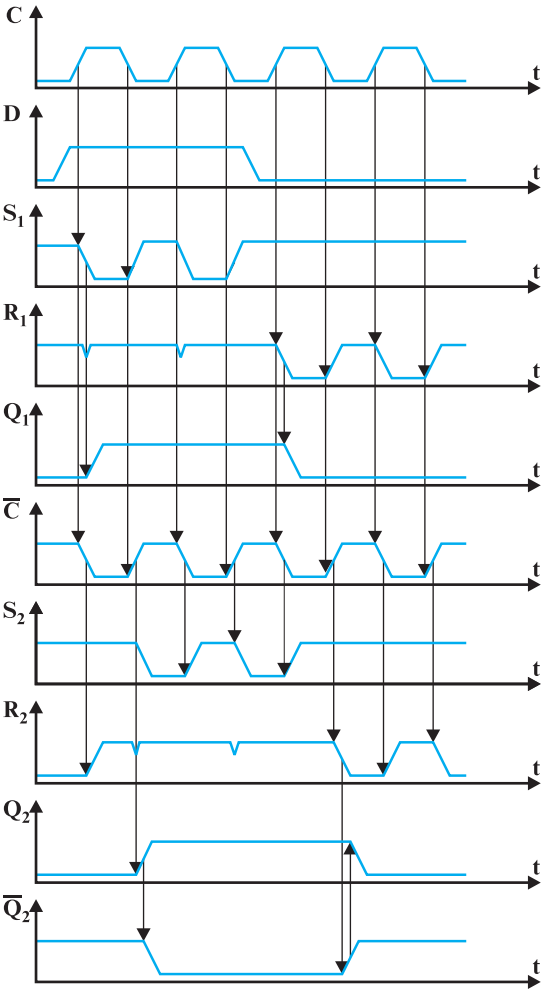
Робота D-тригера, що синхронізований фронтом імпульсу, ілюструється часовими діаграмами вхідних і вихідних сигналів (рис. 11.44, *в*).

Ще одним способом усунення наскрізного керування D-тригером полягає в застосуванні двоступеневих структур. Тригери з двоступеневими структурами називаються тригерами MS-типу. Двоступеневий тригер (рис. 11.45, *а*) складається з двох D-тригерів, які синхронізуються протилежними рівнями сигналу, завдяки чому запис інформації в тригери розділений у часі: якщо один з тригерів знаходиться в стані запису інформації, то інший — у стані її збереження. Робота двоступеневого D-тригера ілюструється часовими діаграмами сигналів (рис. 11.45, *в*). Умовне графічне зображення наведено на рисунку 11.45, *б*.

Т-тригери. Тригери з одним входом, які з кожним імпульсом вхідного сигналу переходять у протилежний стан, називаються ***Т-тригерами*** або ***лічильними*** тригерами, оскільки на їх основі зручно реалізувати лічильники імпульсів. Щоб забезпечити такий алгоритм роботи, необхідно ввести зворотні зв'язки з виходу тригера на його вхід. Найзручніше це зробити у тригерах, побудованих за двоступінчатою схемою (рис. 11.46, *а*). Т-тригер можна також реалізувати на основі двоступінчатого D-тригера, з'єднавши інверсний вихід тригера з його інформаційним входом D (рис. 11.46, *б*). Умовне графічне зображення Т-тригерів наведено на рисунку 11.46, *в*, а часові діаграми сигналів, що ілюструють роботу тригера, на рис. 11.46, *г*.

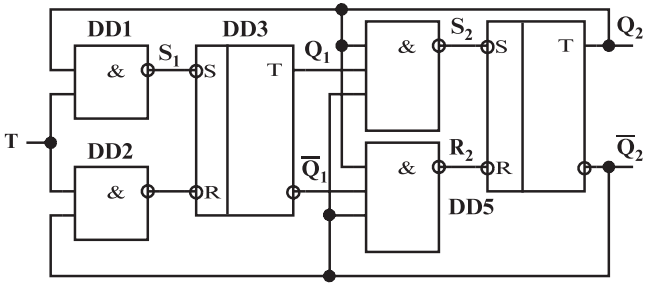

DD1
DD3
DD5
DD7
D
C
&
S
T
R
S_1
R_1
Q_1
$\overline{Q}_1$
$\overline{C}$
S_2
R_2
Q_2
$\overline{Q}_2$
DD2
DD4
DD6
D
TT
C

а) *б)*

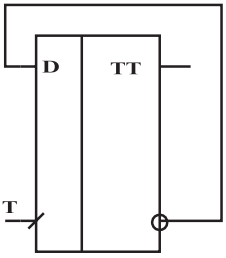

C
D
S_1
R_1
Q_1
$\overline{C}$
S_2
R_2
Q_2
$\overline{Q}_2$
t

в)

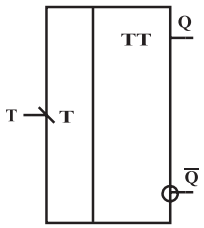
Рис. 11.45



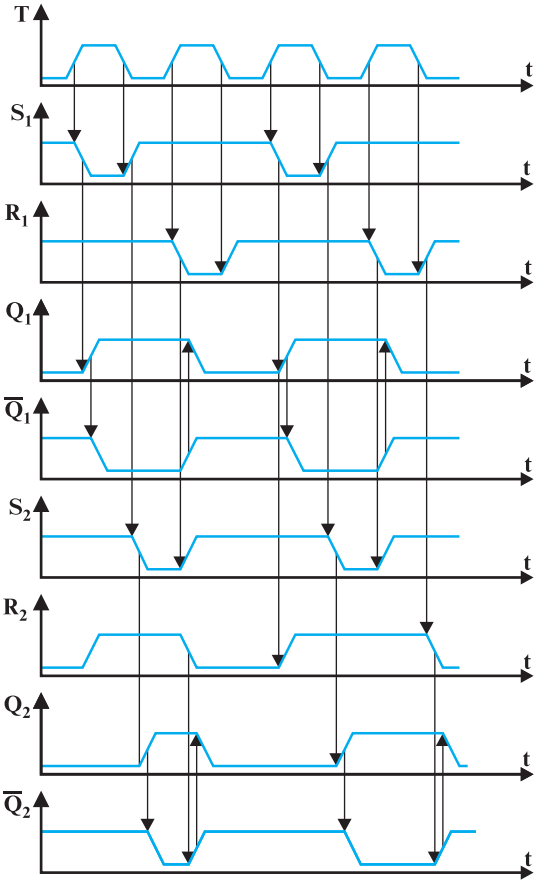
а)



б)



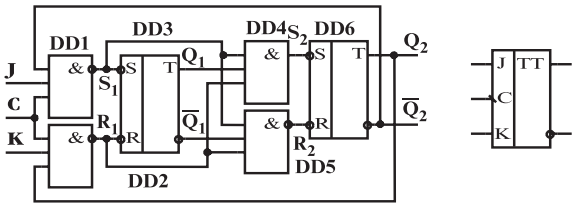
в)



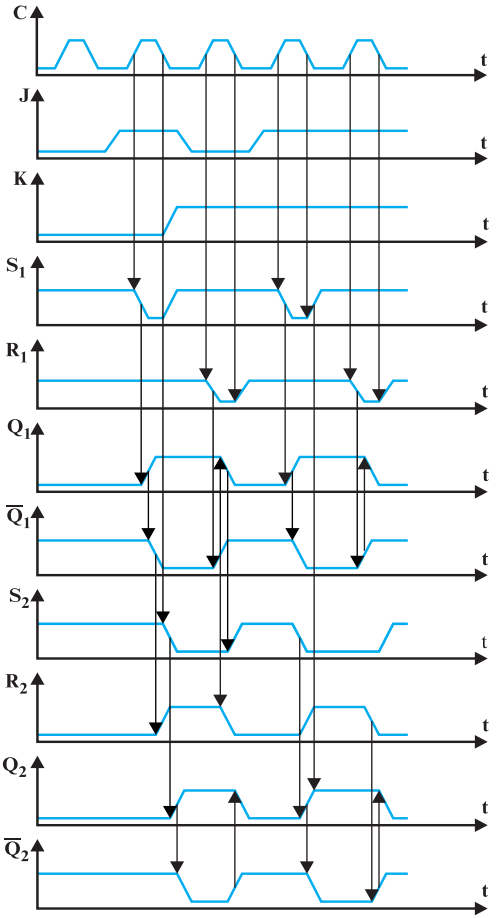
г)

Рис. 11.46

JK-тригер. JK-тригер має як і RS-тригер два керівні входи J і K (рис. 11.47, *а*). Якщо на вхід J подати сигнал з рівнем логічної 1, а на вхід K сигнал із рівнем логічного 0, то тригер установиться на 1, тобто на виході встановиться сигнал логічної 1.



а) *б*)



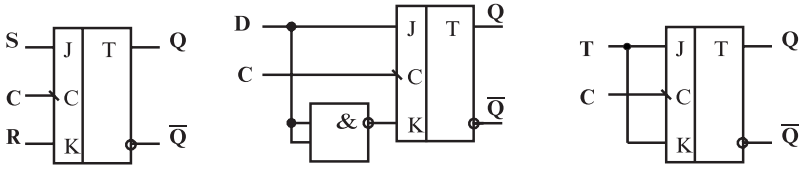
в)

Рис. 11.47

Сигнал з рівнем логічної 1 на вході K встановлює тригер на 0, тобто при такій комбінації сигналів JK-тригер діє, як RS-тригер. Якщо ж на обидва входи JK-тригера подати сигнал логічної одиниці, то під дією імпульсу синхронізації тригер перейде у протилежний стан, тобто коли на J і K входах діє сигнал логічної одиниці, то JK-тригер працює, як T-тригер. JK-тригер зберігає попередній стан, тобто знаходиться в режимі зберігання інформації, якщо на обидва входи J і K подати сигнали логічного 0.

Схему двоступеневого синхронного JK-тригера наведено на рисунку 11.47, *а*, його умовне графічне позначення — на рисунку 11.47, *б*, карту Карно —на рисунку 11.48, *г*.

JK-тригер називається універсальним, тому що на його основі можна реалізувати RS-тригер (рис. 11.48, *а*), D-тригер (рис. 11.48, *б*), T-тригер (рис. 11.48, *в*).



а) *б*) *в*)

	$\overline{J}\,\overline{K}$	$\overline{J}\,K$	$J\,K$	$J\,\overline{K}$
$\overline{C}\,\overline{Q}_{n-1}$	0	0	0	0
$\overline{C}\,Q_{n-1}$	1	1	1	1
$C\,Q_{n-1}$	1	0	0	1
$C\,\overline{Q}_{n-1}$	0	0	1	1

$$Q_n = \overline{C}\,Q_{n-1} + \overline{K}Q_{n-1} + CJ\overline{Q}_{n-1}$$

г)

Рис. 11.48

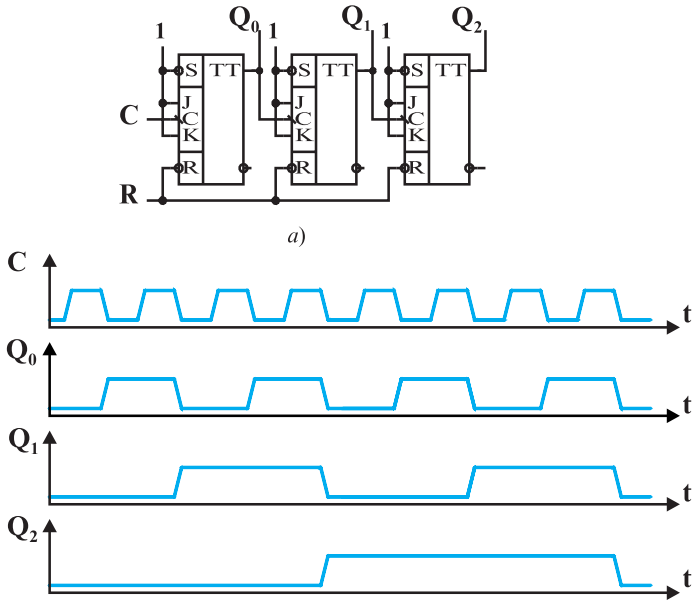
Лічильники імпульсів

Лічильниками імпульсів називаються логічні електронні пристрої з пам'яттю, що призначені для підрахунку кількості вхідних імпульсів і збереження цієї інформації. За способом підрахунку

лічильники поділяються на лічильники додавання, віднімання, реверсивні. У лічильників може бути організовано послідовний, наскрізний, паралельний і комбінований переноси. Лічильники з послідовним і наскрізним переносом називаються ***асинхронними***, а з паралельним — ***синхронними***.

Основними параметрами лічильників імпульсів є місткість лічильника й швидкодія. ***Місткість*** лічильника визначається максимальною кількістю імпульсів, яку може порахувати лічильник, і залежить від кількості тригерів, що входять до складу лічильника. ***Швидкодія*** лічильника імпульсів залежить від швидкодії тригерів, що входять до його складу, і визначається ***середньою тривалістю*** переходу лічильника з одного стану в інший або оберненою величиною — ***максимальною частотою*** імпульсів, які лічильник підраховує без перебоїв.

Асинхронні лічильники. В асинхронних лічильниках тригери переходять з одного стану в інший послідовно, розряд за розрядом, починаючи від наймолодшого і закінчуючи найстаршим. На рисунку 11.49, як приклад, наведено 3-розрядний асинхронний лічильник на основі JK-тригерів, які з'єднані послідовно, тобто вихід першого тригера подається на вхід синхронізації другого,



а)

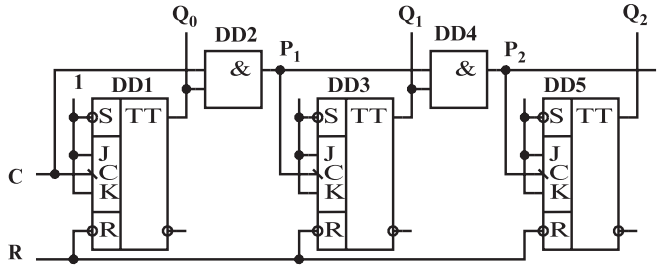
б)

Рис. 11.49

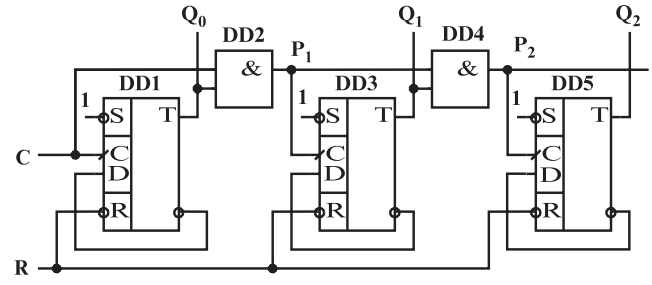
вихід другого — на вхід третього і т. д. Входи J і K об'єднані між собою, і на них подається сигнал із рівнем логічної одиниці. У такому режимі, як відомо, JK-тригер із кожним новим імпульсом, що поступає на його вхід, переходить у протилежний стан.

Середня тривалість переходу лічильника з одного стану в інший визначається сумою тривалостей переходу всіх його тригерів. Таким чином, зі збільшенням кількості розрядів лічильника його швидкодія зменшується, і це є основним недоліком лічильників такого типу.

Один із способів збільшення швидкодії асинхронних лічильників полягає в організації переносів між розрядами через додаткові логічні елементи (рис. 11.50, *а*, *б*). Якщо перший тригер лічильника знаходиться у стані логічної 1, то наступний вхідний імпульс C переводить його в нульовий стан заднім фронтом. Як видно з часових діаграм (рис. 11.50, *в*), ще до переключення виходу Q_1 тригера DD1 лічильний імпульс C через елемент DD2 поступає як імпульс переносу P_1 на тригер DD3 другого розряду і на елемент DD4. Таким чином лічильний імпульс поступає одночасно на всі тригери, що знаходяться в одиничному стані.

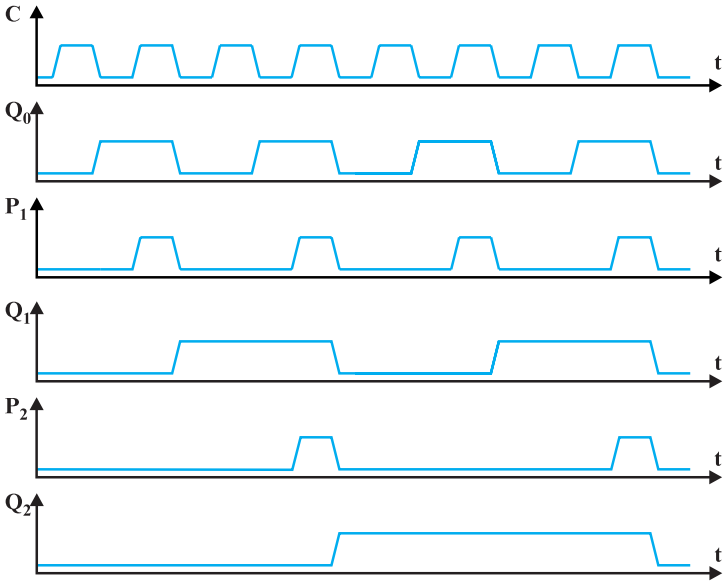


а)



б)

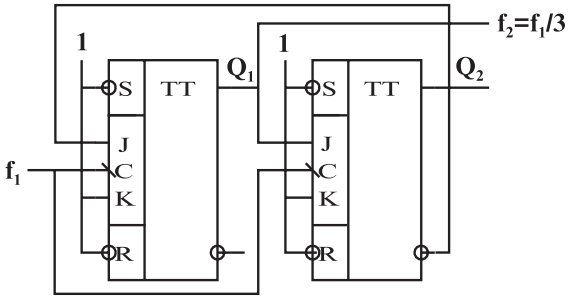
Рис. 11.50



в)

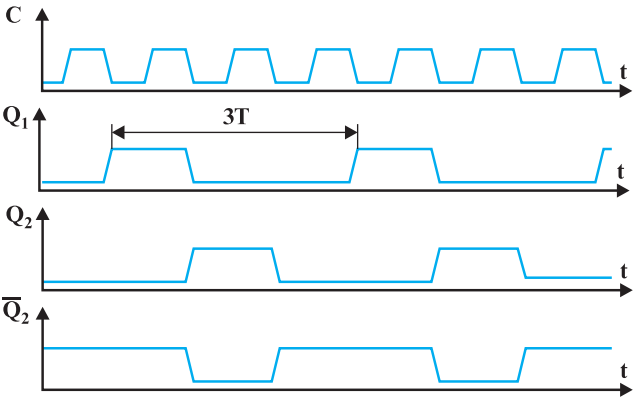
Рис. 11.50

У цифрових пристроях обробки інформації застосовуються дільники частоти імпульсів у певну кількість разів. Дільники частоти реалізуються на лічильниках імпульсів. Коефіцієнт поділу частоти, який є цілим числом, можна розкласти на множники, які є простими числами. Маючи у своєму розпорядженні дільники частоти з коефіцієнтом поділу, що виражаються простими числами, можна реалізувати дільники з довільним коефіцієнтом поділу. На рисунках 11.51 і 11.52 наведено дільники частоти на 3 і 5. Для здійснення таких поділів у лічильники імпульсів необхідно ввести зворотні зв'язки.



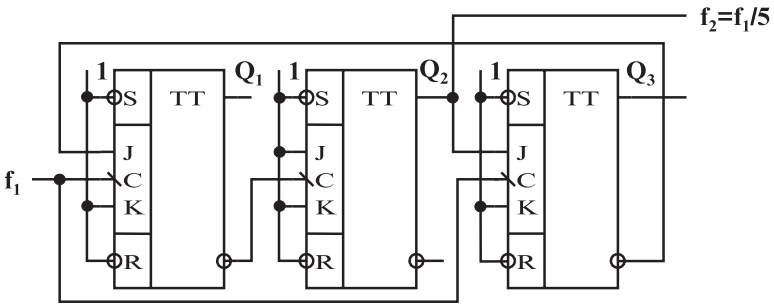
а)

Рис. 11.51

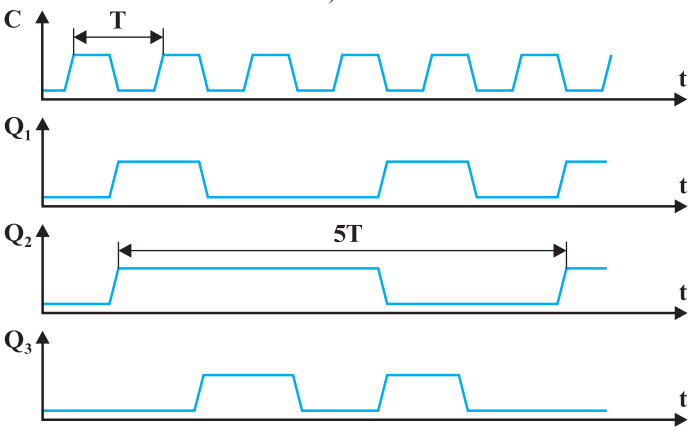


б)

Рис. 11.51



а)



б)

Рис. 11.52

У багатьох цифрових пристроях обробки інформації необхідні такі дільники імпульсів, коефіцієнт поділу яких можна було б автоматично змінювати у процесі роботи, тобто дільники з керованим коефіцієнтом поділу. На рисунку 11.53 наведено схему дільника імпульсів, коефіцієнт поділу якого задається кодом на входах A_0, A_1, A_2, A_3.

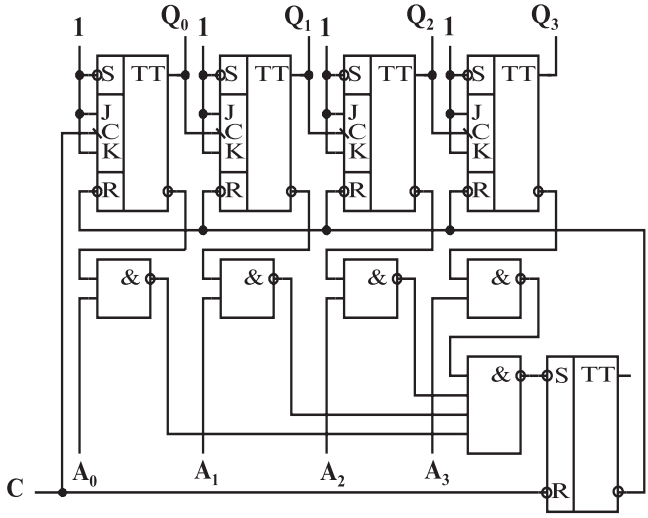
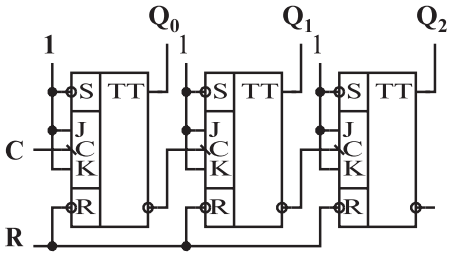


Рис. 11.53

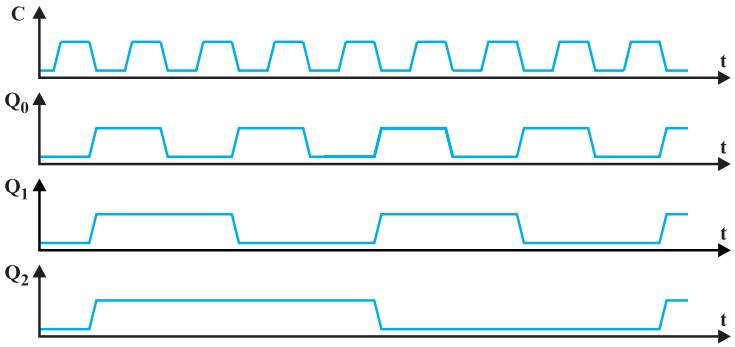
Розглянуті вище асинхронні лічильники імпульсів належать до лічильників ***додавання***. Для реалізації режиму ***віднімання***, достатньо на лічильний вхід тригера подавати сигнал не з прямого виходу попереднього тригера, а з інверсного (рис. 11.54).

Лічильники імпульсів, у яких можна вибрати один з двох режимів: режим додавання чи режим віднімання, називаються ***реверсивними*** лічильниками імпульсів. На рисунку 11.55 наведено схему реверсивного лічильника імпульсів на JK-тригерах, у якому режим віднімання чи додавання вибирається сигналом E.

Синхронні лічильники. До синхронних належать лічильники, в яких переключення розрядів відбувається одночасно, синхронно, для чого на всі розряди лічильника подається послідовність імпульсів синхронізації. Завдяки такій синхронізації досягається мінімальна тривалість встановлення лічильника, що не перевищує тривалості встановлення одного тригера, тобто забезпечується максимальна частота лічби.



а)



б)

Рис. 11.54

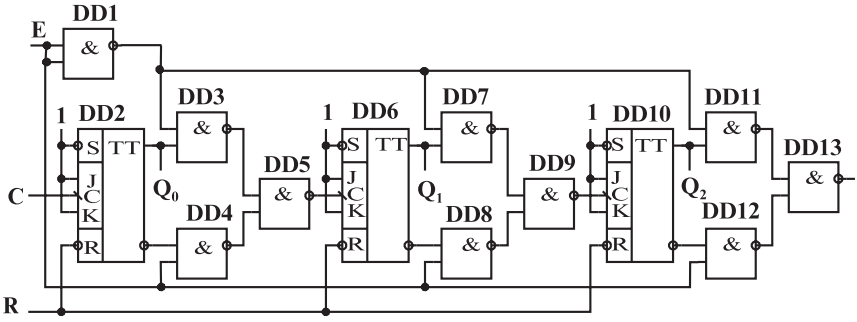


Рис. 11.55

Схему простого синхронного лічильника на JK-тригерах наведено на рисунку 11.56, у цьому лічильнику перенос у старший

розряд поширюється послідовно через елементи DD2, DD4, DD6, ... по мірі поширення логічної 1. Аналогічний лічильник на D-тригерах наведено на рисунку 11.57.

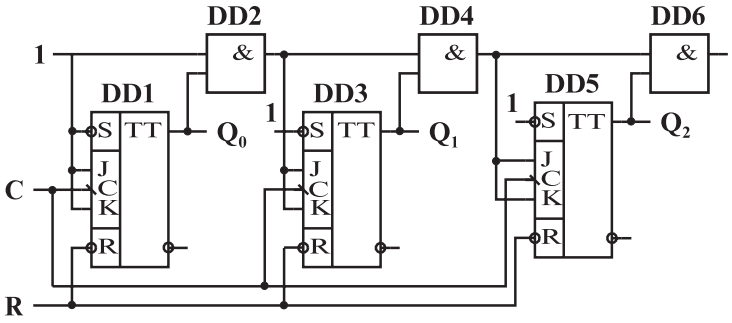


Рис. 11.56

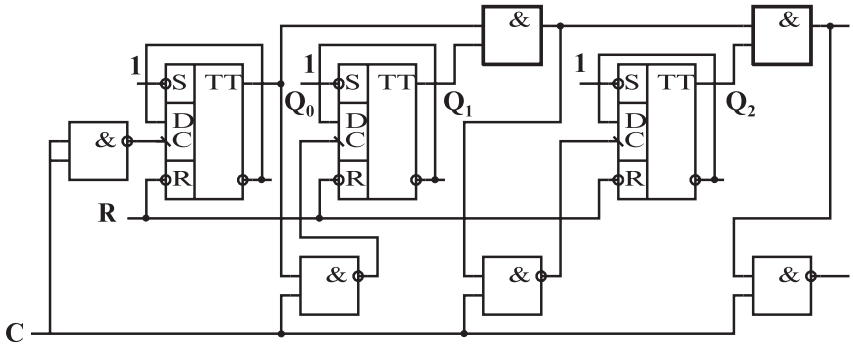
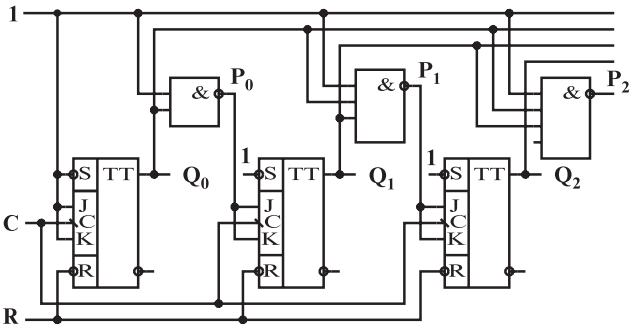


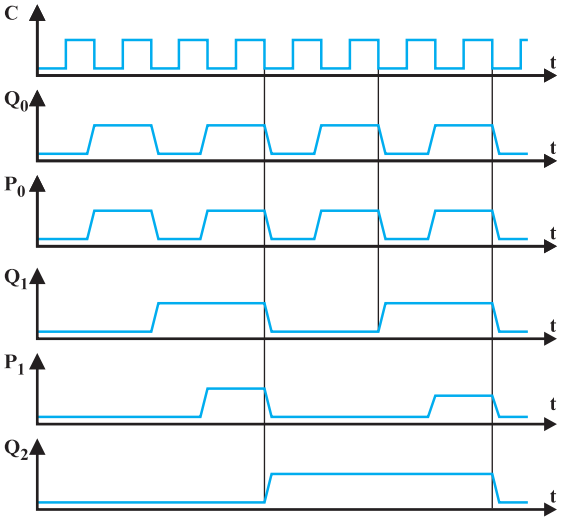
Рис. 11.57

За рахунок логічних елементів схеми переносу DD2, DD4, DD6, накопичується затримка імпульсів, що накладає обмеження на максимальну частоту лічби. Затримка лічильних імпульсів збільшується зі збільшенням кількості розрядів лічильника.

Для зменшення затримки лічильних імпульсів, а отже для збільшення швидкодії лічильника застосовують схеми наскрізного переносу (рис. 11.58, *а*, рис. 11.59), дія яких ґрунтується на тому факті, що довільний розряд лічильника переключається тільки в тому разі, коли всі попередні розряди знаходяться в стані логічної 1. Зі збільшенням порядку розряду збільшується кількість входів логічних елементів схеми переносу, тобто збільшуються апаратурні затрати.



а)



б)

Рис. 11.58

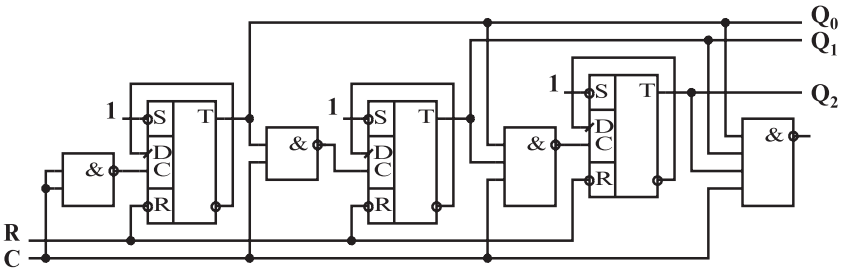


Рис. 11.59

Щоб досягнути компромісу між збільшенням швидкодії лічильника і необхідними для цього апаратурними затратами застосовують поділ розрядів лічильника на групи. Всередині кожної групи розрядів реалізується паралельний перенос, а між групами — послідовний перенос (рис. 11.60). Такі лічильники називаються лічильниками з ***груповим*** переносом.

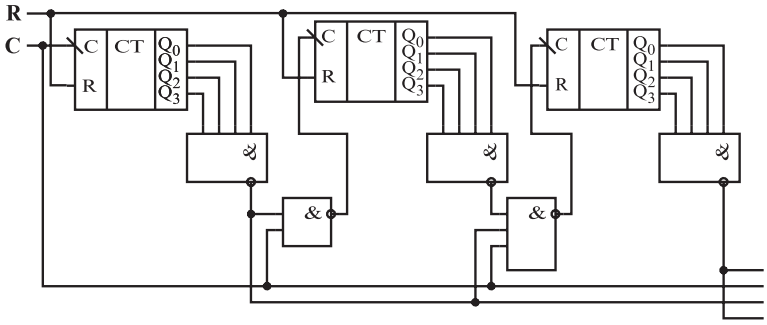


Рис. 11.60

Синхронний лічильник, що працює у режимі ***віднімання***, як і асинхронний, реалізується за рахунок зв'язків входів тригерів з інверсними виходами попереднього молодшого розряду (рис. 11.61). Як правило, такі лічильники мають входи паралельної попередньої установки числа (входи x_0, x_1, x_2, ...), від якого починається зворотна лічба. Занесення паралельного коду в лічильник здійснюється сигналом A.

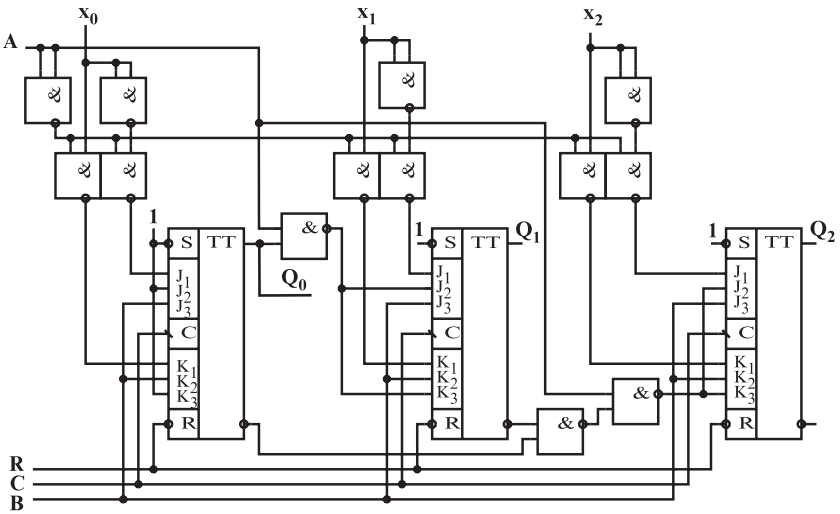
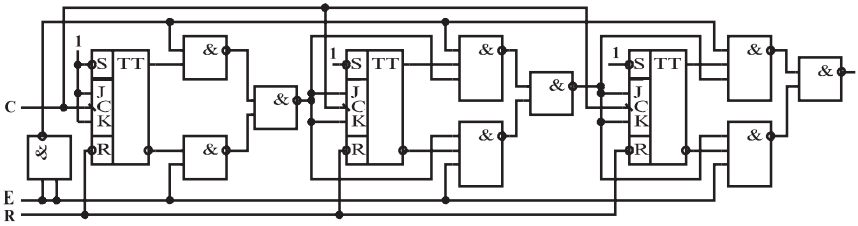


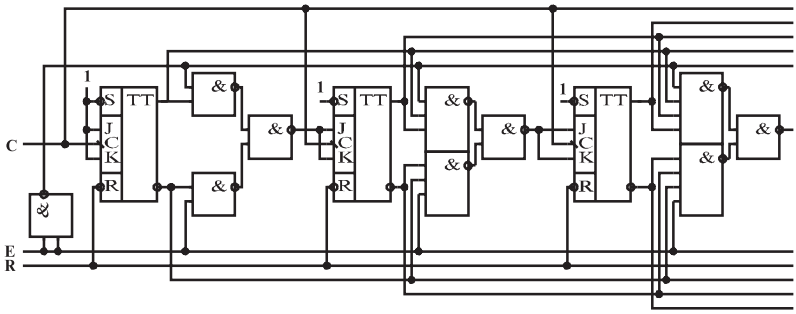
Рис. 11.61

Реверсивні синхронні лічильники будуються за тим самим принципом, що й асинхронні: режим додавання імпульсів реалізується формуванням переносів з прямих виходів тригерів, а режим віднімання — з інверсних.

На рисунку 11.62, *а* наведена схема реверсивного синхронного лічильника з послідовним переносом, а на рисунку 11.62, *б* — з паралельним. Переведення лічильника у режим додавання чи віднімання здійснюється сигналом E.



а)



б)

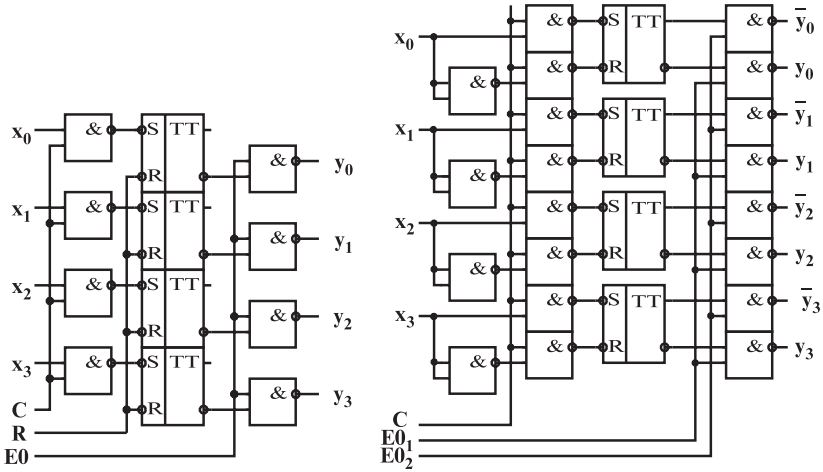
Рис. 11.62

Регістри

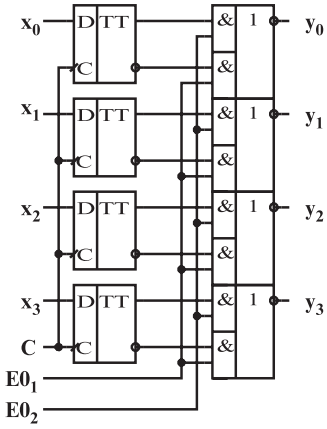
Регістри — це електронні пристрої, побудовані на основі тригерів і призначені для прийому, зберігання, перетворення і передачі інформації у формі двійкових чисел. ***Розрядність*** регістра визначається кількістю тригерів, що входять до його складу, а отже, і кількістю розрядів двійкового числа, яке може зберігати регістр. Другою важливою характеристикою регістрів є їх ***швидкодія***, яка визначається швидкодією тригерів, що входять до складу регістрів.

Залежно від способу прийому й передачі інформації в двійковій формі регістри можуть поділятися на послідовні, паралельні й універсальні.

Паралельний регістр. У паралельний регістр всі розряди двійкового числа вводяться одночасно, паралельно в усі розряди. Паралельні регістри використовуються як пам'ять з великою швидкодією у цифрових пристроях обробки інформації. На рисунку 11.63 наведено схеми паралельних регістрів, що реалізовані на різних типах тригерів і з різною реалізацією вхідних і вихідних схем. У регістрі на основі асинхронних RS-тригерів (рис. 11.63, *а*) перед занесенням інформації через входи x_0, x_1, x_2, ... необхідне попереднє встановлення всіх тригерів у нульовий стан сигналом *R*. Введення (запис) інформації здійснюється сигналом *C*.



а) *б*)



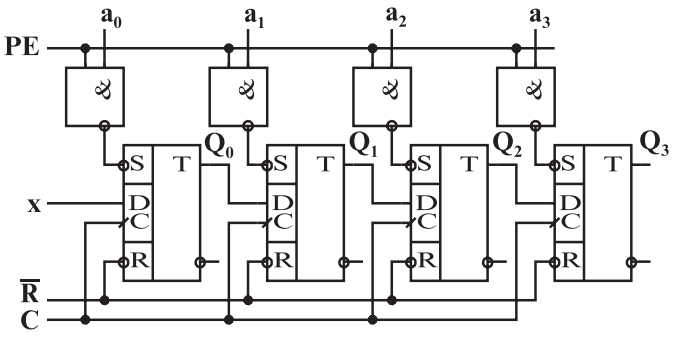
б)

Рис. 11.63

У регістрі (рис. 11.63, *б*), що також побудований на основі асинхронних RS-тригерів, попереднє очищення регістра не потрібне, і запис інформації здійснюється за один такт. Видача інформації в цьому регістрі може здійснюватися в прямому коді (сигнал $ЕО_2 = 1$) або в оберненому коді (сигнал $ЕО_1 = 1$). У регістрі, що побудований на основі D-тригерів (рис. 11.63, *в*), занесення інформації здійснюється імпульсами синхронізації С. Видача інформації може здійснюватися як у прямому коді, так і в оберненому.

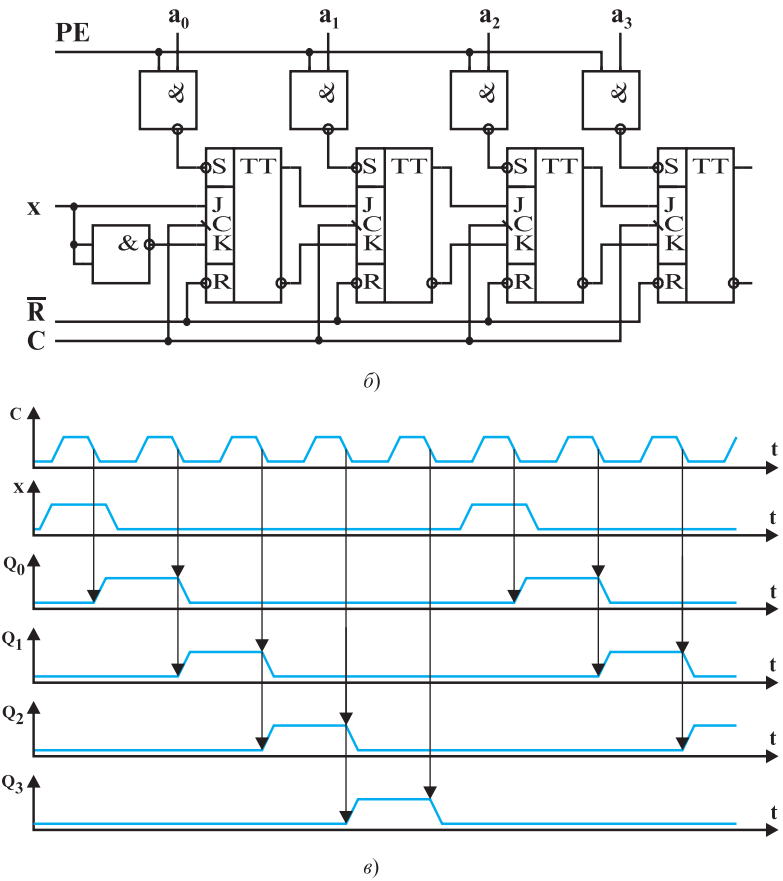
Послідовний регістр. Послідовні регістри виконують операцію зсуву під час послідовного введення чи виведення інформації. Зсув двійкового числа реалізується перезаписом стану між сусідніми тригерами регістра у напрямі зсуву. Таким чином, кожний тригер регістра одночасно приймає інформацію з попереднього розряду й передає в наступний. Для уникнення перебоїв і підвищення надійності застосовуються двотактні тригери. Якщо зсув інформації здійснюється в одному напрямі, то достатньо з'єднати виходи попереднього тригера MS-типу із входами наступного (рис. 11.64). Введення інформації можна здійснювати ***послідовно***, розряд за розрядом по входу X з наступним зсувом інформації вправо на один розряд кожним імпульсом синхронізації. Можна також здійснити введення інформації у регістр ***паралельно*** по входах x_0, x_1, x_2 сигналом РЕ. Схеми таких регістрів, що реалізовані на основі D-тригерів, наведено на рисунку 11.64, *а*, а на JK-тригерах — на рисунку 11.64, *б*. Часові діаграми смгналів послідовного регістра наведено на рисунку 11.64, *в*.

Крім зберігання інформації, регістр зсуву використовується також для перетворення послідовного коду двійкового числа у паралельний.



а)

Рис. 11.64



б)

в)

Рис. 11.64

У ***реверсивних*** регістрах зсув інформації може здійснюватися в обох напрямках, тому вихід кожного розряду має бути зв'язаний через логічні елементи із входами попереднього й наступного розрядів (рис. 11.65). Напрям зсуву задається сигналом E.

Універсальний регістр. Універсальним регістром називається реверсивний регістр зсуву, в якому передбачена можливість паралельного занесення інформації, тобто в універсальному регістрі поєднані властивості паралельних і послідовних регістрів. На рисунку 11.66 наведена схема універсального регістра на D-тригерах, який може працювати в трьох режимах: послідовного введення інформації через вхід x із зсувом праворуч, послідовного введення інформації через вхід y із зсувом ліворуч, паралельного введення інформації через входи a_0, a_1, a_2, a_3.

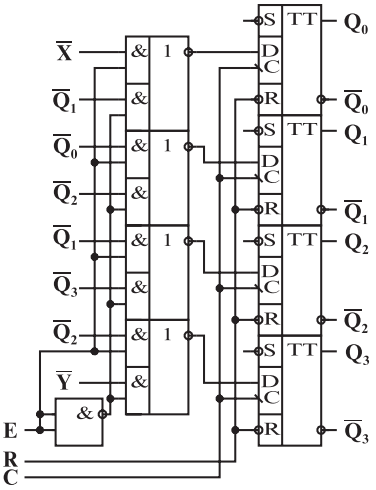


Рис. 11.65

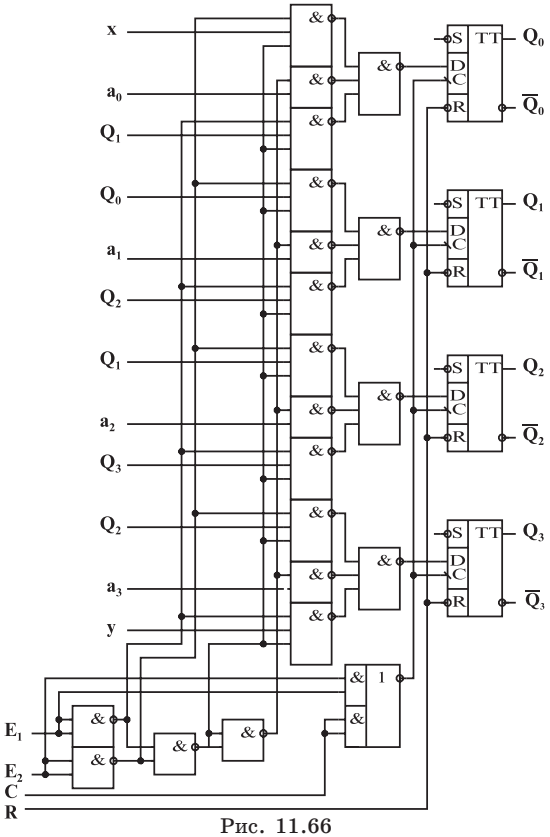


Рис. 11.66

Режим роботи універсального регістра задається за допомогою сигналів E_1 і E_2 згідно з таблицею 11.10.

Таблиця 11.10

Керівні сигнали		Режим роботи
E_1	E_2	
0	0	Паралельне введення інформації
0	1	Зсув зліва
1	0	Зсув справа
1	1	Блокування входів

Сучасна електронна промисловість випускає універсальні регістри у вигляді інтегральних мікросхем, які залежно від спеціальних керівних сигналів можуть працювати в режимах як послідовного, так і паралельного регістрів. На рисунку 11.67, як приклад, наведено універсальний 4-розрядний регістр К531ИР11.

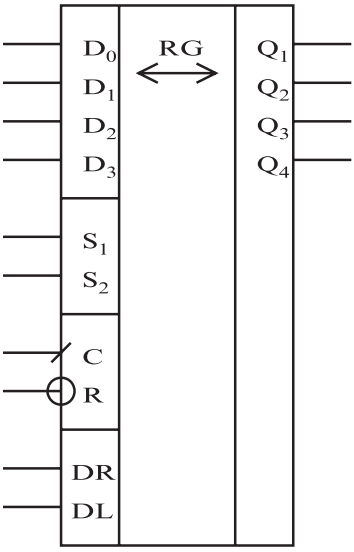


Рис. 11.67

Входи D_0, D_1, D_2, D_3 призначені для паралельного введення інформації, вхід DR призначений для послідовного введення інформації із зсувом вправо, вхід DL — для послідовного введення інформації із зсувом вліво, вхід C — вхід синхронізації, вхід R призначений для очищення регістра і встановлення всіх його розрядів у нульовий стан. Входи керування S_0, S_1 призначені для

встановлення режиму роботи регістра. Залежність режиму роботи регістра від комбінації сигналів на входах S_0, S_1 наведено у таблиці 11.11.

Таблиця 11.11

S_1	S_0	Режим роботи
0	0	Блокування
0	1	Зсув справа
1	0	Зсув зліва
1	1	Паралельне введення

1. Що таке логічні величини?
2. Які елементарні операції можна виконувати над логічними змінними?
3. Дайте визначення функціонально повним системам елементарних функцій?
4. Як можна спростити логічні вирази за допомогою карт Карно?
5. Що таке логічні елементи?
6. Які види мікросхем логічних елементів ви знаєте?
7. Схарактеризуйте цифрові комбінаційні пристрої?
8. Для чого застосовуються шифратори і дешифратори?
9. Які види тригерів ви знаєте, який їх принцип дії?
10. Який принцип дії електронного лічильника імпульсів?
11. Схарактеризуйте структуру і принцип дії комбінаційних суматорів?

12

Мікропроцесори

12.1 Будова мікропроцесора

Мікропроцесором називається програмований електронний пристрій для обробки інформації, виконаний у вигляді однієї чи кількох мікросхем високого ступеня інтеграції. Перші мікропроцесори були у вигляді інтегральних мікросхем на кристалах кремнію. На початку у вигляді мікросхем виготовлялися окремі логічні елементи. Такі мікросхеми вважалися мікросхемами з малим ступенем інтеграції. Використовуючи мікросхеми логічних елементів, змонтованих на платах із друкованим монтажем, можна було створювати окремі цифрові пристрої обробки інформації: регістри, шифратори, дешифратори, суматори, лічильники імпульсів тощо.

Удосконалення інтегральних технологій дало змогу виготовляти у вигляді мікросхем не тільки окремі логічні елементи, але цифрові пристрої в цілому. Такі мікросхеми належать до мікросхем із середнім ступенем інтеграції.

Подальший бурхливий розвиток інтегральних технологій дав змогу виготовляти у вигляді однієї мікросхеми окремі блоки електронно-обчислювальних машин. Такі мікросхеми називаються мікросхемами з високим ступенем інтеграції.

Наступний етап розвитку інтегральних технологій дав змогу виготовити у вигляді однієї чи кількох мікросхем основну частину електронно-обчислювальної машини, яка виконує основний обсяг обробки інформації — процесор. Такий процесор, виготовлений у вигляді однієї чи кількох мікросхем, отримав назву ***мікропроцесор***.

Мікропроцесор здійснює обробку інформації за програмою, що записана в його пам'ять. Змінивши програму, можна змінювати функції й області застосування мікропроцесора. У цьому полягає його універсальність.

Мікропроцесор, як уже зазначалося, призначений для виконання програм з обробки інформацій. Програма складається з певної кількості команд (інструкцій), які мікропроцесор виконує у певній

послідовності. Команди, як і дані, над якими мікропроцесор виконує операції, знаходяться у пам'яті.

Виконання команд програми мікропроцесором — це певна циклічна послідовність дій:

- ❒ формування адреси чергової команди;
- ❒ зчитування цієї команди за сформованою адресою й пересилка її з пам'яті у мікропроцесор;
- ❒ дешифрування отриманої з пам'яті команди, тобто розкладання команди на елементарні дії, які мають виконувати пристрої мікропроцесора;
- ❒ власне виконання команди, тобто виконання у певній послідовності елементарних дій, з яких складається команда;
- ❒ формування адреси операндів, над якими виконується певна послідовність елементарних операцій даної команди;
- ❒ зчитування операндів з пам'яті за сформованою адресою і пересилання їх із пам'яті у мікропроцесор;
- ❒ формування адреси, за якою буде записано результат виконання даної команди;
- ❒ пересилання результату за сформованою адресою з мікропроцесора у пам'ять;
- ❒ формування адреси наступної команди.

Щоб виконати таку послідовність дій, мікропроцесор має такі складові частини: пристрій формування адрес команд і операндів; операційний пристрій, тобто пристрій виконання команд; пристрій керування; система шин, призначена для взаємодії пристроїв мікропроцесора між собою.

Розглянемо будову, принцип дії й основні характеристики цих складових частин мікропроцесора. Пристрій формування адрес команд і операндів більш детально розглянемо у наступному підрозділі.

Пристрій виконання команд

Пристрій виконання команд призначений для прийому команд із пам'яті через шину даних, дешифрування команд, тобто розкладання команд на елементарні дії, виконання цих дій у певній послідовності.

У свою чергу, пристрій виконання команд складається з таких основних частин: блок попередньої вибірки команд; блок дешифрування команд; виконавчий блок.

Блок попередньої вибірки команд призначений для отримання команд із шини даних і розміщення їх у черзі наперед вибраних команд. Режим попередньої вибірки команд є майже в усіх

сучасних мікропроцесорах і призначений для суміщення виконання певних операцій, що дає істотне прискорення виконання програми у цілому.

Блок дешифрування команд вилучає чергову команду з черги попередньої вибірки, перетворює її на послідовність елементарних операцій і розміщує дешифровану команду в чергу, де вона очікує виконання. Команду можна дешифрувати апаратним і програмним способами. У мікропроцесорах, де реалізовано ***апаратний спосіб*** дешифрування команд, команда розкладається на елементарні дії, і кожну таку дію виконує спеціальний пристрій, наприклад, операцію додавання виконує суматор, операцію множення — перемножувач, добування квадратного кореня — спеціальний пристрій для добування кореня тощо.

У мікропроцесорах із ***програмною реалізацією*** кожна команда розкладається на послідовність елементарних операцій (мікрокоманд), які виконуються у певній послідовності, тобто для кожної команди складається спеціальна програма — ***мікропрограма*** виконання цієї команди. Наприклад, команда додавання двох чисел розкладається на такі елементарні операції, як запис першого доданку в один регістр суматора, запис другого доданку в другий регістр суматора, пересилання результату додавання з регістра результату в пам'ять тощо. Команда множення двох чисел розкладається на ряд послідовних додавань. Щоб виконати команду добування квадратного кореня з числа, функція кореня розкладається у ряд, тобто у послідовність операцій множення, ділення, піднесення до степеня. Операція піднесення до степеня замінюється послідовністю операцій множення, а операція множення — послідовністю операцій додавання. Мікропрограми зберігаються у спеціальному пристрої пам'яті, який називається ***пам'яттю мікропрограм***. Дешифрування команд при мікропрограмному способі реалізації команд полягає у зчитуванні відповідної мікропрограми з пам'яті й виконанні послідовності мікрокоманд.

У багатьох випадках реалізується змішаний спосіб виконання команд — частина команд виконується апаратним способом, а решта — програмним способом.

Виконавчий блок безпосередньо здійснює виконання дешифрованих команд. До складу блока входять такі складові частини: регістри загального призначення, спеціальні регістри, суматор із можливістю зсуву чисел на певну кількість одиниць, перемножувач, пристрій обробки послідовностей, пристрій місцевого керування, пристрій перевірки захисту тощо. У деяких мікропроцесорах суматор і кілька регістрів загального призначення утворюють

так званий арифметико-логічний пристрій (в англомовній літературі він позначається абревіатурою ALU — arithmetic and logic unit).

У сучасних мікропроцесорах спостерігається стійка тенденція до розподілу функцій. Наприклад, виконання складних математичних операцій покладається на спеціалізований пристрій, так званий ***співпроцесор***. Співпроцесор може розміщуватися на тому самому кристалі, що й сам мікропроцесор, а може бути виготовлений як окрема мікросхема.

Система шин мікропроцесора

Мікропроцесор побудований за модульним принципом, тобто складається з окремих, відносно самостійних частин, кожна з яких виконує свої спеціальні функції. Щоб забезпечити взаємодію та узгоджену роботу всіх частин мікропроцесора, складові частини мікропроцесора з'єднані системою шин (магістралей).

Шина або ***магістраль*** — це сукупність електричних з'єднувальних провідників, по кожному з яких передається один розряд двійкового числа.

Кількість провідників, що входить до складу шини, називається ***шириною*** шини. Сучасні мікропроцесори використовують шини шириною 32, 64, 80 розрядів. Ще однією важливою характеристикою шини є її ***швидкодія***, яка характеризується кількістю двійкових слів, які здатна передати шина за одиницю часу.

Якщо шиною можна передавати інформацію тільки в одному напрямі, наприклад, від постійної пам'яті до мікропроцесора, то така шина називається ***однонапрямленою***, а якщо у двох напрямах, наприклад, від оперативного пристрою пам'яті до мікропроцесора і навпаки, то така шина називається ***двонапрямленою***.

За типом інформації, що передається, розрізняють ***шину даних*** (data bus), ***шину адреси*** (address bus) і ***шину керування*** (control bus). Мікропроцесор має внутрішню систему шин, яка зв'язує всі його складові частини. Крім того, мікропроцесор у цілому пов'язаний системою зовнішніх шин з іншими пристроями мікропроцесорної системи.

Шини мікропроцесора можуть працювати у режимі розподілу часу, тобто певний відрізок часу виконання команд шина працює як шина адреси, а в інший відрізок часу як шина даних. Такий режим роботи шини називається ***мультиплексуванням*** шини. Такий режим реалізований, наприклад, у мікропроцесорі К1810ВМ86.

Для підвищення продуктивності обробки інформації сучасні мікропроцесорні системи побудовано як мультипроцесорні, тобто, коли в системі паралельно працюють два і більше процесорів,

наприклад, центральний процесор і співпроцесор, який виконує тільки складні математичні операції. Розподілом задач між паралельно працюючими процесорами досягається істотне підвищення продуктивності системи в цілому. Така будова мікропроцесорних систем відповідає модульному принципу, що спрощує модернізацію та експлуатацію такої системи.

Водночас у мультипроцесорних системах виникають специфічні проблеми, пов'язані головним чином з організацією взаємодії й розподілом спільних ресурсів. Одним із таких ресурсів є система шин, яку використовують кілька паралельно працюючих пристроїв, проте у довільний момент часу шину може використовувати (керувати) тільки один пристрій.

Для розв'язку цієї проблеми у сучасних мікропроцесорних системах передбачені апаратні і програмні засоби, наприклад, спеціальний процесор — контролер шини, який аналізує запити центрального процесора, співпроцесора, контролерів периферійних пристроїв на користування шиною, зважує їх пріоритети і передає керування шиною процесору, що має в даний момент часу максимальний пріоритет.

Пристрій керування

Модульний принцип побудови мікропроцесора, складний характер взаємодії складових частин мікропроцесора, необхідність продуктивної роботи зумовлюють необхідність мати спеціальний пристрій керування у складі мікропроцесора, який керував би роботою складових частин мікропроцесора. Крім того, на пристрій керування покладається завдання розподілу спільних ресурсів, наприклад, внутрішньої системи шин.

Пристрій керування мікропроцесором має складну ієрархічну будову. Кожний модуль мікропроцесора має свій локальний пристрій керування, який генерує сигнали, призначені для керування даним модулем. Крім того, для здійснення координації взаємодії модулів у цілому призначений пристрій керування вищого рівня ієрархії.

12.2 Структура пам'яті й система адресації

Як уже зазначалося, команди програм, які виконує мікропроцесор, а також дані, які обробляються мікропроцесором, зберігаються в пам'яті. Структура пам'яті та її характеристики мають дуже важливе значення і багато в чому визначають продуктивність і ефективність усієї мікропроцесорної системи в цілому.

Однією з найважливіших характеристик пам'яті є її ***місткість***, яка характеризується кількістю одиниць інформації, яку можна зберігати в даній пам'яті. Другою важливою характеристикою пам'яті є ***швидкодія***, яка характеризується ***часом доступу***, тобто тривалістю отримання даних із пам'яті. Між цими двома характеристиками пам'яті є обернена залежність: чим більша ємність пам'яті, тим менша її швидкодія і навпаки, тому для кожного випадку потрібно вибирати компромісне співвідношення.

Конструюючи систему пам'яті, слід враховувати частоту звертання до пам'яті. Як показують статистичні дослідження, дані дуже різко відрізняються за частотою звертання, наприклад, імовірність того, що до пам'яті звернуться за результатами попередніх обчислень дуже висока, а ймовірність того, що до пам'яті звернуться за результатами проміжних обчислень, які виконувалися давно, — дуже низька.

Враховуючи все це, пам'ять мікропроцесора має ієрархічну будову з різними рівнями ієрархії. На верхньому рівні ієрархії знаходиться пам'ять з максимальною швидкодією і незначної місткості. Це, як правило, пам'ять на регістрах загального призначення і спеціальних регістрах, що входять до складу арифметико-логічного пристрою. На регістрах запам'ятовуються результати обчислень, які будуть потрібні на наступних кроках алгоритму.

Крім регістрової пам'яті, до складу мікропроцесора входить так звана ***кеш-пам'ять***. Внутрішня кеш-пам'ять мікропроцесора призначена для розміщення інформації, до якої найчастіше звертається мікропроцесор. Регістрова пам'ять і внутрішня кеш-пам'ять розміщені на одному кристалі з мікропроцесором і становлять ***внутрішню пам'ять*** комп'ютера.

Програма, яка виконується мікропроцесором, завантажується в ***оперативну пам'ять***. Елементи оперативної пам'яті виконані на електронних компонентах у вигляді мікросхем і розміщені на материнській платі комп'ютера. Місткість оперативної пам'яті сучасних мікропроцесорів становить сотні мегабайт, час доступу — частки мікросекунди.

Крім внутрішньої кеш-пам'яті, сучасні мікропроцесорні системи мають ***зовнішню кеш-пам'ять***, куди операційна система переміщує інформацію, за якою мікропроцесор найчастіше звертається до оперативної пам'яті.

Операційна система, а також прикладні програми зберігаються в пам'яті зовнішніх пристроїв: гнучких і твердих магнітних дисків, а також на оптичних дисках. Місткість пам'яті на магнітних і оптичних дисках становить десятки гігабайтів і спостерігається стійка тенденція до її збільшення.

Для продуктивної та ефективної роботи мікропроцесора застосовуються системи адресації різного ступеня складності, тобто системи формування адрес команд під час виконання програми.

Будь-яка команда програми складається з двох частин: в одній частині міститься код команди, власне сама команда, а в другій — одна чи кілька адрес операндів. За кількістю адрес, які містяться в адресній частині команди, команди поділяються на чотириадресні, триадресні, двоадресні, одноадресні та безадресні.

Чотириадресна команда містить адреси двох операндів операції, адресу результату та адресу наступної команди. Якщо в програмі чи її частині відсутні команди розгалуження й циклу, то команди у пам'яті розміщені послідовно одна за одною, тобто наступна команда має адресу на 1 більшу, ніж попередня. У такому разі для формування адреси застосовується ***лічильник команд*** (PC-program counter). Лічильник команд — це спеціальний регістр, вміст якого автоматично збільшується на 1 після виконання чергової команди. Таким чином автоматично формується адреса наступної команди й потреба у чотириадресних командах відпадає.

Триадресна команда містить адреси двох операндів і адресу результату операції. Часто адресу результату вказувати недоцільно, оскільки результат знаходиться у визначеному регістрі. Такий регістр часто називають ***акумулятором***. Команди, в яких вказані адреси двох операндів, називаються ***двоадресними***.

Адресу одного з операндів також можна встановити фіксованою, тому її можна не вказувати, команда міститиме адресу тільки одного операнда і називатиметься ***одноадресною***.

У багатьох сучасних мікропроцесорних системах організована так звана ***стекова пам'ять***. Стекова пам'ять організується для зберігання результатів послідовності команд. Результат чергової операції записується за однією й тією ж наперед визначеною адресою, яка називається ***вершиною стека***, при цьому результати всіх попередніх операцій зсуваються на одну позицію вперед або назад. Зчитуються результати операцій також за однією й тією ж адресою, тобто у зворотному порядку: результат, який був записаний останнім, зчитуватиметься першим. Така організація пам'яті дає змогу використовувати ***безадресні команди***, оскільки вершина стека має наперед визначену адресу.

Безпосередньою адресацією називається така адресація, коли операнд міститься у самій команді. Наприклад, у мікропроцесорах серії 80x86 є команда

MOV AL, 0111B,

яка означає, що необхідно двійкове число 0111 занести в регістр AL. У цій команді операнд 0111 міститься в самій команді.

Якщо адреса операнда міститься в адресній частині команди, то така адресація називається ***прямою***. Перевагою прямої адресації є те, що для визначення адреси операнда не потрібно робити ніяких обчислень. Недоліком прямої адресації є неможливість змінити адресу в процесі трансляції програми або в процесі виконання програми. Крім того, сучасні мікропроцесори оперують з великими обсягами пам'яті, тому адреси мають велику кількість розрядів і, як наслідок, для розміщення команд із прямою адресацією необхідний великий обсяг пам'яті.

Сучасні мікропроцесори здійснюють обробку інформації за складними алгоритмами, в них реалізований багатозадачний режим роботи мікропроцесора. Це призводить до того, що в мікропроцесорах застосовуються складні системи адресації. Однією з різновидів таких систем адресації є ***опосередкована адресація***. На відміну від прямої адресації, в якій адреса операнда містилася в самій команді, в опосередкованій системі адресації в адресній частині команди міститься не адреса операнда, а адреса спеціального регістра, в якому міститься адреса операнда, тобто в адресній частині команди міститься «адреса адреси».

Системи адресації сучасних мікропроцесорних систем мають забезпечувати виконання ряду задач.

Системи адресації мають забезпечувати можливість переміщувати програми і дані в пам'яті, не змінюючи самих програм. Програми для мікропроцесорних систем складаються, як правило, з окремих, відносно самостійних модулів, які розробляються окремо, причому різними програмістами. Для того, щоб їх об'єднати в одне ціле, необхідно забезпечити можливість переміщувати модулі програми в пам'яті, не вносячи змін у самі програми. Важливо зазначити, що переміщення програм або частин програм у пам'яті бажано здійснювати незалежно від переміщення даних. Це потрібно для того, щоб програма, під яку відведений за попередніми підрахунками певний обсяг пам'яті, могла розділити його власне між програмою і даними. У процесі виконання програми може виявитися, що попередні розрахунки не точні, і тоді пам'яті або не вистачатиме, або вона виявиться зайвою. Крім того, окремі частини масиву даних можуть призначатися для обміну даними між різними частинами програми для забезпечення їх взаємодії. Оскільки різні частини програми розроблялися незалежно одна від одної і можливо різними виконавцями, то може виникнути необхідність перемістити у пам'яті обмінний масив незалежно від інших частин даних і незалежно від частин програми.

До складу програмного забезпечення сучасних мікропроцесорних систем входять бібліотеки стандартних програм, до яких звертаються різні користувачі. Якщо мікропроцесор працює у багатозадачному режимі, тобто обробляє інформацію для кількох прикладних програм, то ці програми можуть звертатися до бібліотеки за однією й тією ж стандартною програмою, яку потрібно кожний раз транслювати. Доцільно здійснити трансляцію один раз і користуватися цією програмою, але для цього потрібно передбачити можливість переміщувати її в пам'яті.

Таким чином, наявність засобів для переміщення програм, даних, або їх частин у пам'яті без зміни самої програми машинною мовою еквівалентно збільшенню швидкодії машини та обсягу пам'яті.

Можливість переміщувати програми, дані та їх частини незалежно одна від одної і без зміни самих програм забезпечує ***ущільнення*** інформації в пам'яті.

Якщо мікропроцесор використовується для виконання кількох програм паралельно, то потрібно передбачити ***захист*** інформації однієї програми від несанкціонованого доступу іншої програми.

Складні системи адресації обумовлені також і організацією в мікропроцесорних системах так званої віртуальної пам'яті. ***Віртуальна пам'ять*** — такий спеціальний режим роботи пам'яті, коли мікропроцесор звертається до програми так, ніби вся ця програма чи кілька одночасно працюючих програм розміщені в оперативній пам'яті, хоч насправді в цій пам'яті розміщується тільки частина (сегмент) програми, а інші частини (сегменти) програми розміщені, наприклад, на магнітних дисках. Операційна система автоматично переміщує ті частини (сегменти) програми, до яких найчастіше звертається мікропроцесор із зовнішньої пам'яті в оперативну і навпаки: частини програми, до яких мікропроцесор звертається зрідка, автоматично переміщуються з оперативної пам'яті в зовнішню пам'ять. Така організація пам'яті дає змогу значно збільшити швидкодію мікропроцесорної системи в цілому і підвищити її продуктивність і ефективність.

Необхідність застосування складних систем опосередкованої адресації обумовлена також забезпеченням ефективності роботи програм у режимі переривання.

У складних системах адресації застосовується ***сегментація*** пам'яті, тобто поділ пам'яті на окремі частини — ***сегменти*** пам'яті. Сегменти пам'яті можуть мати різний розмір і перекривати один одного. Ніяких обмежень на розміщення сегментів у пам'яті

немає. Поділ пам'яті на сегменти дає змогу переміщувати програми й частини програм незалежно одна від одної.

Крім поділу пам'яті на сегменти у мікропроцесорних системах застосовується ***сторінкова*** організація пам'яті, коли пам'ять ділиться на частини (сторінки) однакового розміру (здебільшого 4 кілобайти).

Системи адресації поділяються на системи, в яких адреси не обчислюються, і системи, в яких адреси обчислюються. До систем, в яких адреси не обчислюються, належить система ***безпосередньої*** адресації, в якій операнд міститься в самій команді. До систем, в яких адреси не обчислюються, належить ***регістрова*** система адресації, в якій операнд знаходиться в регістрі мікропроцесора. Код регістра міститься в самій команді. В системах безпосередньої й регістрової адресацій команди виконуються за мінімальний час.

Крім регістрової адресації, у мікропроцесорах застосовується ***опосередкована регістрова*** адресація, коли адреса операнда знаходиться в одному з регістрів мікропроцесора.

До систем опосередкованої адресації, в яких адреса знаходиться за допомогою обчислень, належить ***базова*** система адресації. В цій системі адреса операнда обчислюється як сума ***базової адреси***, що зберігається в одному з регістрів, і ***зміщення***, що міститься у самій команді. Така адреса, знайдена в результаті обчислень, називається ***ефективною*** адресою. Таким чином, схема визначення ефективної адреси базової системи адресації має вигляд:

$$\textit{Ефективна адреса} = \textit{базова адреса} + \textit{зміщення}.$$

Базову систему адресації зручно застосовувати для обробки масивів даних, коли зміщення, тобто номер елемента в масиві відомий до виконання програми, а початкова, тобто базова адреса, обчислюється в процесі виконання програми.

У системах ***індексної*** адресації ефективна адреса знаходиться як сума зміщення, що міститься в самій команді, та вмісту спеціального ***індексного*** регістра. У мікропроцесорних системах індексація адреси часто суміщується з ***масштабуванням***, тобто перемножуванням вмісту індексного регістра на масштабний коефіцієнт, значення якого дорівнює 2, 4, 8, 16. Крім звичайної індексної адресації застосовується також ***відносна*** або ***базова індексна*** адресація, в якій ефективна адреса обчислюється як сума вмісту базового та індексного регістрів.

У сучасних мікропроцесорних системах застосовуються складні комбінації різних видів адресації, наприклад, у мікропроцесорі Intel 80386 адреса може обчислюватися в три етапи.

На першому етапі обчислюється ефективна адреса, як сума трьох адресних компонент: базової адреси, зміщення і масштабованого індексу за схемою:

Ефективна адреса = база + зміщення + індекс-коефіцієнт.

На другому етапі обчислюється ***лінійна*** адреса, як сума ефективної адреси, отриманої на першому етапі, і базової адреси сегмента пам'яті за схемою:

Лінійна адреса = ефективна адреса + базова адреса сегмента.

Третій етап виконується тільки в разі сторінкової організації пам'яті. На третьому етапі лінійна адреса, отримана на другому етапі, перетворюється (транслюється) на ***фізичну*** адресу, тобто адресу, яка встановлюється на шині адреси. Фізична адреса складається з фізичної адреси сторінки й зміщення відносно початку сторінки.

12.3 Система переривань

У сучасних мікропроцесорних системах мікропроцесор, як правило, здійснює обробку інформації кількох процесів паралельно, з розподілом часу. Наприклад, здійснює введення інформації для однієї програми й виконує іншу програму. У такому режимі роботи часто виникають ситуації, коли потрібно тимчасово призупинити й відкласти виконання однієї програми і перейти до виконання іншої програми. Тимчасове призупинення називається ***перериванням*** (interrupt) програми.

Переривання можуть викликатися зовнішніми пристроями мікропроцесорної системи, наприклад, пристроями введення й виведення інформації, пристроями зовнішньої пам'яті. Крім того, переривання можуть викликатися пристроями системи автоматичного керування, в якій мікропроцесор застосовується як керівний пристрій.

Переривання програми слід здійснити таким чином, щоб програму можна було продовжувати виконувати в будь-який момент часу, як тільки мікропроцесор звільниться від роботи над іншими програмами. Щоб здійснити це, в сучасних мікропроцесорах передбачено цілий комплекс апаратних і програмних засобів, який називається ***системою переривань***. Зокрема до складу цієї системи, входить спеціалізований мікропроцесор (контролер), призначений для обробки переривань. У центральному мікропроцесорі

для запитів на обслуговування переривань передбачено спеціальні виводи.

Класифікацію переривань за причинами й характером наведено на рисунку 12.1.

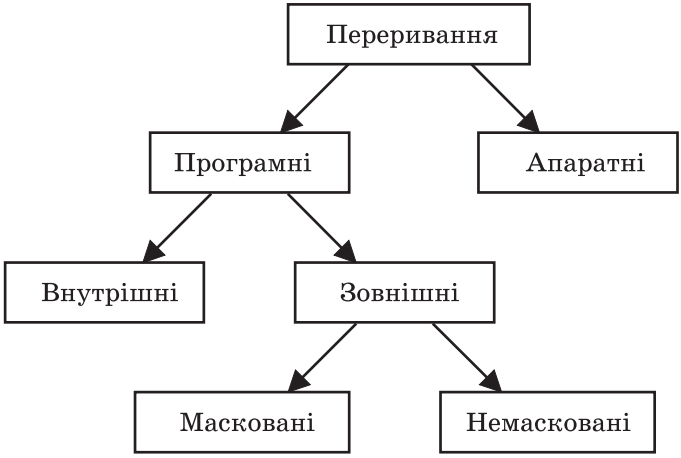


Рис. 12.1

Програмні переривання викликаються самою програмою. Програмні переривання зручно використовувати для організації доступу до певних модулів, спільних для всіх програм. Наприклад, програмні модулі операційної системи доступні для прикладних програм саме через систему переривань.

Апаратні переривання викликаються фізичними пристроями і можуть траплятися у довільний момент часу. Ці переривання інформують мікропроцесор про стан цих фізичних пристроїв. Мікропроцесор має відповідним чином реагувати на сигнали від зовнішніх пристроїв. На деякі події мікропроцесор має реагувати невідкладно, наприклад, на аварійне вимкнення живлення, на вихід із ладу зовнішнього пристрою тощо.

Використання механізму переривань дозволяє узгодити в часі роботу зовнішніх пристроїв з низькою швидкодією та роботу швидкодіючого мікропроцесора. Наприклад, введення символів текстової інформації з клавіатури здійснюється зі швидкістю кілька знаків за секунду. В перервах між введенням символів мікропроцесор може здійснювати виконання інших програм, наприклад, виводити на друк документи, відтворювати музичні твори, записані на оптичний диск тощо.

Внутрішні (логічні) переривання формуються самим процесором, коли трапляються особливі події, наприклад, ділення числа на нуль, помилка парності тощо.

Зовнішні апаратні переривання викликаються сигналами, зовнішніми по відношенню до мікропроцесора. Ці сигнали подаються на спеціальні виводи мікросхеми мікропроцесора.

Зовнішні апаратні переривання, які можна заборонити або дозволити програмно за допомогою спеціального біта в регістрі ознак, називаються ***маскованими***. Переривання, які програмно заборонити не можна, вважаються ***немаскованими***. У мікросхемі процесора для маскованих і немаскованих переривань передбачено окремі виводи: INT — для маскованих і NMI — для немаскованих. Немасковані переривання застосовуються тоді, коли мікропроцесор має негайно реагувати на сигнали переривання, наприклад, аварійне вимкнення живлення, виявлення помилок у програмі тощо.

Оскільки мікропроцесор здійснює обробку інформації кількох процесів паралельно, то кожний з цих процесів ініціює переривання незалежно один від одного, і може трапитися ситуація, коли на мікропроцесор поступить кілька сигналів переривань одночасно. Для коректної роботи мікропроцесорної системи в цілому слід установити ***рівні пріоритету*** для кожного переривання. Мікропроцесор віддає перевагу та обробляє переривання з найвищим пріоритетом. Наведемо для прикладу перелік переривань у послідовності зменшення їх пріоритетів:

- ❒ переривання, спричинені помилкою ділення двох чисел;
- ❒ програмні переривання, ініційовані спеціальною командою переривань;
- ❒ переривання через переповнення кількості розрядів;
- ❒ немасковані переривання;
- ❒ масковані переривання;
- ❒ переривання покрокового режиму роботи.

Сучасні системи переривання дають змогу призначати пріоритети і змінювати призначені пріоритети. Більше того, пріоритети можна змінювати в процесі роботи програм.

Переривання, які поступили в мікропроцесор, обробляються спеціальними програмами — ***обробниками переривань***, що розміщені в пам'яті за певними адресами. Для кожного типу переривань призначений свій обробник. Базові адреси обробників називаються ***векторами переривань***. Вектори переривань зведені у спеціальні таблиці — ***таблиці векторів переривань*** і розміщені в оперативній пам'яті мікропроцесора.

Мікропроцесор обробляє переривання в такій послідовності:

- ❒ завершується поточна команда;
- ❒ перевіряється тип переривань.

Якщо переривання внутрішнє або немасковане, то мікропроцесор переходить до занесення інформації про стан перерваної програми в пам'ять. Якщо ж переривання масковане, то перевіряється біт заборони маскування в регістрі ознак. Якщо переривання заборонено, то запит на переривання відхиляється й виконується наступна команда перерваної програми. Якщо ж масковане переривання дозволено, то формується сигнал підтвердження переривання:

- ❒ занесення інформації, необхідної для відновлення перерваної програми в пам'ять. Як правило для цього застосовується стекова пам'ять;
- ❒ викликається процедура обробки переривань;
- ❒ виконується процедура обробки переривань;
- ❒ викликається з пам'яті інформація, необхідна для продовження виконання перерваної програми;
- ❒ виконується наступна команда перерваної програми.

Для апаратного й програмного забезпечення переривань призначений спеціальний контролер переривань (PIC — Programmable Interrupt Controller) виготовлений у вигляді мікросхеми 8259A фірми Intel (вітчизняний аналог КР580ВН59).

12.4 Система команд мікропроцесорів

Системою команд мікропроцесора називається сукупність команд, які може виконувати мікропроцесор. Залежно від сукупності команд, які може виконувати мікропроцесор, вони поділяються на такі види:

- ❒ CISC (Complex Instruction Set Computer) мікропроцесор із повним набором команд (інструкцій) збільшеної довжини. Для підвищення продуктивності передбачається збільшувати тактову частоту мікропроцесора;
- ❒ RISC (Reduced Instruction Set Computer), де застосовано спрощену систему команд однакового формату. Основними командами є команди типу регістр-регістр. Команди поділено на поля, тому дешифрування таких команд спрощується;
- ❒ MISC (Multipurpose Instruction Set Computer), де застосовано поєднання команд типу RISC із мікропрограмним пристроєм пам'яті.

Як приклад, розглянемо систему команд мікропроцесора Intel 8086 (вітчизняний аналог К1810ВМ86). Всю сукупність команд мікропроцесора можна поділити на такі види:

- команди передачі даних;
- команди арифметичних операцій;
- команди логічних операцій і зсувів;
- команди передачі керування;
- ланцюжкові команди;
- команди керування мікропроцесором.

Для виконання команд мікропроцесор використовує регістри операційного пристрою, які є доступними програмісту для програмування команд. Для зручності регістри мікропроцесора зручно поділити на групи.

Група регістрів загального призначення або регістрів даних складається з таких двобайтних регістрів: **AX, BX,CX, DX**. Особливістю цих регістрів є те, що старший **(H)** і молодший **(L)** байти цих регістрів можуть адресуватися окремо.

До складу групи вказівних та індексних регістрів входять такі двобайтні адресні регістри: **SP, BP, SI, DI**. Вони призначені для зберігання двобайтних адрес.

Група сегментних регістрів складається з регістра коду команд **CS**, регістра даних **DS**, регістра стека **SS**, регістра додаткових даних **ES**.

Останню групу регістрів складають регістр-вказівник команд **IP** або програмний лічильник **PC**, і регістр ознак. Біти регістра ознак фіксують властивості результатів арифметичних і логічних операцій, а також призначені для керування певними діями мікропроцесора. Бітами регістра ознак фіксуються такі ознаки:

- **AF —** додатковий перенос із молодшої тетради (молодшого напівбайта) у старшу тетраду (старший напівбайт);
- **CF** — перенос, який виникає під час виконання арифметичних і логічних операцій;
- **OF** — переповнення, яке виникає під час виконання арифметичних операцій;
- **SF**— знак результату;
- **PF**— парність кількості одиниць, які містяться у молодшому байті результату;
- **ZF** — наявність нульового результату операції;
- **DF** — визначає напрям перегляду ланцюжкових даних;
- **IF** — ознака переривання;
- **TF** — перехід мікропроцесора в покроковий режим.

Команди передачі даних

Команди передачі даних призначені для переміщення інформації між пам'яттю й регістрами, а також між одним регістром та іншим. Команди передачі даних можна поділити на такі групи:

- ❒ загальні команди передачі даних;
- ❒ команди передачі даних із застосуванням стека (стекові команди);
- ❒ команди введення/виведення.

Загальні команди передачі даних. До даної групи входять команди, що здійснюють передачу інформації регістр-регістр, регістр-пам'ять, пам'ять-регістр. Найпоширенішею є команда **MOV**.

Загальний вигляд команди такий:

MOV	Приймач	Передавач

Перший байт команди містить код операції (КОП) і два однобітних поля — напряму ***d*** і довжини слова ***w***.

Поле ***d*** визначає напрям переміщення інформації: якщо ***d***=1, то інформація передається у мікропроцесор, а якщо ***d*** = 0, то в пам'ять. Якщо поле ***w*** = 1, то операнд має два байти (слово), а якщо ***w*** = 0, то один байт.

Другий байт команди складається з трьох полів: поле режиму ***mod***, поле регістра ***reg*** і поле ***r/m*** — регістр/пам'ять.

Якщо ***mod*** = 00, то зміщення адреси відсутнє, якщо ***mod*** = 01, то команда містить однобайтне зміщення адреси, якщо ***mod*** = 10, то команда містить двобайтне зміщення адреси, а якщо ***mod*** = 11, то здійснюється регістрова адресація.

Поле ***r/m*** визначає: регістр чи пам'ять є другим операндом.

Крім команди **MOV** до складу загальних команд передачі даних входить команда **XCHG** обміну інформацією між регістром і пам'яттю чи між двома регістрами і має формат:

XCHG	*reg*	*mem/reg*

Однобайтна команда **XLAT** замінює вміст акумулятора на байт з 256-байтної таблиці, початкова адреса якої знаходиться в регістрі **BX**.

Команди **LEA, LDS, LES** відрізняються від інших команд передачі даних тим, що при їх виконанні в адресований регістр передаються не дані, а адреси.

Стекові команди. Стекові команди призначені для занесення інформації в стек (команда **PUSH**) і вилучення інформації зі стека (команда **POP**). Для адресації вершини стека, що знаходиться

у поточному сегменті, і містить останній занесений у стек елемент даних, призначений вказівник SP. Всі стекові команди маніпулюють тільки двома байтами (словами) і супроводжуються модифікацією вказівника стека: під час занесення інформації у стек автоматично здійснюється зменшення вмісту вказівника стека на 1 (декремент вказівника стека), а під час вилучення даних зі стека автоматично здійснюється збільшення вказівника стека на 1 (інкремент вказівника стека).

Команди введення/виведення. Команда введення призначена для введення інформації в акумулятор мікропроцесора від зовнішнього пристрою введення інформації через порт, а команда виведення — для виведення інформації на зовнішній пристрій також через порт. Команда введення має формат:

IN	Акумулятор	Порт введення

а команда виведення — формат:

OUT	Порт виведення	Акумулятор

Команди арифметичних операцій

Команди арифметичних операцій призначені для виконання складних обчислень. Наявність у системі таких команд дає змогу застосовувати мікропроцесор у складних системах обробки інформації. Арифметичні операції здійснюються над цілими числами чотирьох типів: беззнаковими двійковими, знаковими двійковими, упакованими десятковими й неупакованими десятковими. Під час виконання арифметичних операцій ознаки результату арифметичної операції фіксуються у регістрі ознак. Арифметичні операції складаються з команд додавання, віднімання, множення, ділення й порівняння.

Команди додавання **ADD** мають формат:

ADD	Пам'ять або регістр	Регістр або пам'ять

і дає змогу додавати два операнди, причому кожний з них може розміщуватися або в пам’яті, або в регістрі мікропроцесора. Різновидність цієї команди має формат:

ADD	Пам'ять або регістр	Дані

і дає змогу додати до вмісту регістра або елемента пам’яті 8- або 16-бітну константу, яка міститься в самій команді. Скорочений варіант цієї команди:

ADD	Акумулятор	Дані

дає змогу додати до вмісту акумулятора, тобто спеціального регістра константу, що міститься в команді.

Команда **ADC** має такий самий формат, що й **ADD**, але на відміну від **ADD** до суми двох доданків додає ще й біт переносу CF, який міститься у регістрі ознак.

Команди віднімання **SUB** та **SBB** мають такі ж формати й різновиди, що і команди додавання **ADD** та **ADC**, і відрізняються тільки видом операції.

Команда множення двох беззнакових цілих чисел

MUL	Регістр

перемножує вміст адресованого в команді регістра з вмістом акумулятора. Команда

MUL	Пам'ять

перемножує вміст елемента пам'яті, адреса якого міститься в команді, з вмістом акумулятора.

Команда множення **IMUL** має такі ж формати й різновиди, що і команда **MUL**, але відрізняється від останньої тим, що операндами її є знакові двійкові числа, записані у додатковому коді.

Команди ділення мають формати, аналогічні форматам команд множення

DIV	Регістр

DIV	Пам'ять

IDIV	Регістр

IDIV	Пам'ять

Команди логічних операцій і зсувів

Команди логічних операцій призначені для обробки булевих величин. У мікропроцесорі є команди для виконання таких логічних операцій: інверсії **NOT**, кон'юнкції **AND**, диз'юнкції **OR**, додавання за модулем 2 **XOR** і команда **TEST**, яка виконує кон'юнкцію операндів, але не змінює їх значень.

Команди логічних операцій мають такі формати:

Код логічної операції (NOT, AND, OR, XOR, TEST)	Пам'ять або регістр	Регістр або пам'ять

Код логічної операції (NOT, AND, OR, XOR, TEST)	Пам'ять або регістр	Дані

Код логічної операції (NOT, AND, OR, XOR, TEST)	Акумулятор	Дані

Команди логічних операцій, що мають перший тип формату, виконують операцію, визначену кодом операції (КОП), що міститься у першому байті команди, над операндами, розміщеними в регістрі або елементі пам'яті. У другому випадку здійснюється безпосередня адресація і логічна операція виконується над операндом, що міститься в пам'яті або регістрі, й операндом, що міститься в самій команді. У третьому випадку також здійснюється безпосередня адресація і логічна операція виконується над операндом, що міститься в акумуляторі та операндом, що міститься в самій команді.

Команди логічних операцій діють на кожний біт даних окремо, тому ознаки переносу в регістрі ознак завжди переводяться в нульове положення.

Зсув двійкового числа на певну кількість розрядів уперед або назад використовується у багатьох процедурах обробки інформації. У системах команд мікропроцесорів для здійснення таких операцій передбачені спеціальні команди. В мікропроцесорі К1810ВМ86 реалізовано такі команди зсуву:

- RCL — циклічний зсув вліво, включаючи біт переносу;
- RCR — циклічний зсув вправо, включаючи біт переносу;
- ROL — циклічний зсув вліво, не включаючи біт переносу;
- ROR — циклічний зсув вправо, не включаючи біт переносу;
- SHL, SHR — логічний зсув, відповідно вліво і вправо. Для логічного зсуву характерно, що у біт, який звільняється, записується нуль, а біт, що виходить за межі слова, втрачається;
- SAL, SAR — арифметичний зсув, відповідно, вліво і вправо. Арифметичний зсув вправо відрізняється від логічного тим,

що знаковий біт не зсувається, а повторюється у сусідньому правому біті, зберігаючи знак числа. Арифметичний зсув вліво SAL у додатковому коді не відрізняється від логічного зсуву SHL. Два позначення SAL та SHL однієї й тієї ж операції введено для зручності програмування.

Команди зсуву мають такий формат:

Код операції зсуву	Пам'ять або регістр	Кількість зсувів

Команди передачі керування

Команди передачі керування призначені для виконання програм із розгалуженнями, циклічних фрагментів програм, підпрограм. Якщо виконується лінійний фрагмент програми, то операційний пристрій одну за одною вибирає команди з черги команд і послідовно їх виконує. Під час вибору чергової команди значення програмного лічильника (PC —Program Counter) збільшується на одиницю. Фізична адреса програмної пам'яті визначається вмістом сегментного регістра **CS** і програмного лічильника **PC**.

У циклічних програмах, програмах із розгалуженням, а також під час організації підпрограм, необхідно виконувати не наступну команду, а команду, що знаходиться в пам'яті за адресою переходу. Спеціальні команди, що змінюють значення у регістрах **CS** та **PC**, називаються ***командами передачі керування***. Сегментна організація пам'яті програм визначає дві основні різновидності команд передачі керування. Передача керування в межах поточного сегмента називається близькою (тип **NEAR**) і змінює вміст тільки програмного лічильника **PC**. Адреса переходу записується одним словом. Передача керування за межі поточного сегмента називається міжсегментною або далекою (тип **FAR**) і для неї необхідно змінювати вміст обох регістрів **CS** і **PC**.

Команди передачі керування поділяються на команди безумовних переходів, команди умовних переходів, викликів, повернень, керування циклами й команди переривань.

Команди безумовних переходів мають такі формати:

JMP	Однобайтне зміщення

JMP	Двобайтне зміщення

JMP	Пам'ять або регістр

JMP	Адреса

JMP	Пам'ять

Перші три команди мають тип **NEAR**, а останні дві — **FAR**.

Двобайтна команда першого типу містить у другому байті зміщення, яке додається до вмісту програмного лічильника **PC**. Трибайтна команда другого типу здійснює ті ж дії, що й попередня команда. Команда третього типу реалізує безумовний перехід з опосередкованою адресацією. Адресою переходу, що заноситься у **PC**, служить вміст регістра або слова в пам'яті. В разі адресації пам'яті, команда містить адресу елемента пам'яті. Четверта команда реалізує міжсегментний перехід з прямою адресацією, а п'ята — з опосередкованою. Останні два байти команди здійснюють запис інформації у регістр **CS**, а передостанні два — у регістр **PC**.

Команди умовних переходів мають такий єдиний формат:

Код команди умовного переходу	Однобайтне зміщення

Під час виконання цих команд аналізується певна умова, закодована поточними станами полів регістра ознак, і залежно від виконання умови перехід здійснюється або не здійснюється.

Команда виклику підпрограми **CALL** передає керування з автоматичним збереженням адреси повернення. У полі операнда цієї команди знаходиться мітка першої команди підпрограми, що викликається. Переходячи до підпрограми, слід тимчасово запам'ятати адресу команди, що стоїть після команди **CALL**. Ця адреса називається адресою повернення. Після того як підпрограма закінчить свої дії, її кінцева команда повернення **RET** передає керування за збереженою адресою повернення. Зручним місцем збереження адрес повернення є стек, який забезпечує дуже просту організацію вкладених підпрограм.

Сегментна організація пам'яті впливає на реалізацію команд викликів. Як і команди безумовного переходу, виклики можуть бути в межах сегмента і між сегментами.

Команди виклику підпрограм мають такі формати:

CALL	Двобайтне зміщення

CALL	Пам'ять або регістр

CALL	Адреса

CALL	Пам'ять

Команди виклику підпрограм **CALL** мають такі ж формати, що й команди безумовних переходів **JMP**. За дією на регістри команди **CALL** також відповідають командам **JMP**, але додатково вони зберігають у поточному сегменті стека, що адресується спеціальним регістром **SS**, адресу повернення з відповідною зміною вказівника стека **SP**.

Кожна підпрограма мусить мати як мінімум одну команду повернення **RET**, що повертає керування програмі, яка викликала дану підпрограму. Така передача керування здійснюється вилученням із стека адреси повернення, яка була занесена в стек командою виклику підпрограми. Таким чином, команда повернення не містить ніякої адресної інформації, неявно адресуючи вершину стека.

Три команди керування циклами застосовуються для організації програмних циклів. У них передбачено використання регістра **CX** як лічильника циклів. Команди керування циклами мають такі формати:

LOOP	Однобайтне зміщення

LOOPE або LOOPZ	Однобайтне зміщення

LOOPNE або LOOPNZ	Однобайтне зміщення

Виконання команд циклу супроводжується зменшенням вмісту регістра **CX** на одиницю (декремент регістра **CX**) і, якщо значення двійкового числа в цьому регістрі не дорівнює нулю, то зміщення, що міститься в команді, додається до числа в програмному лічильнику і здійснюється перехід на початок циклу. Якщо ж число в регістрі **CX** дорівнює нулю, то виконання циклу закінчено і виконується наступна команда. Команди **LOOPE** і **LOOPZ** означають одну і ту ж команду, яка на відміну від команди **LOOP** аналізує значення поля **ZF** регістра ознак, і якщо це значення дорівнює 1, то цикл виконується. Команди **LOOPNE** і **LOOPNZ** аналогічні командам **LOOPE** і **LOOPZ** , тільки в командах **LOOPNE** і **LOOPNZ** цикл виконується, якщо значення поля **ZF** регістра ознак дорівнює 0.

Команди переривань призначені для здійснення програмних переривань і мають такі формати:

INT	Тип переривання

INTO

IRET

Перші дві команди викликають підпрограму обробки переривань приблизно так, як це роблять апаратні переривання.

Команда **INT** викликає програму обробки переривань, яка визначається типом переривань.

Однобайтна команда переривань у разі переповнення **INTO** генерує програмне переривання, якщо в результаті операції встановлено ознаку переповнення **(OF=1)** у регістрі ознак.

Однобайтна команда повернення з переривань **IRET** призначена для виходу з підпрограми обробки переривань.

Ланцюжкові команди

Ланцюжкові команди призначені для операцій над ланцюжками даних, тобто послідовності зв'язаних даних, що знаходяться в суміжних елементах пам'яті. У системі команд мікропроцесора К1810ВМ86 є п'ять однобайтних команд, призначених для обробки одного елемента ланцюжка даних. Перед ланцюжковою командою може стояти префікс повторення **REP**, який викликає повторення дії команди над наступним елементом. Завдяки такому префіксу, обробка елементів ланцюжка здійснюється швидше, ніж при організації циклу.

Команди можуть мати операнд-передавач, операнд-приймач, те й інше одночасно.

Команда передачі ланцюжка **MOVS** має такий формат:

MOVS	Ланцюжок-приймач	Ланцюжок-передавач

Ця команда передає байт або слово з ланцюжка, що адресується регістром **SI**, у ланцюжок, що адресується регістром **DI**.

Команда порівняння ланцюжків даних **CMPS** має такий формат:

CMPS	Ланцюжок-приймач	Ланцюжок-передавач

Цією командою здійснюється віднімання байта чи слова з ланцюжка-приймача від відповідного байта чи слова ланцюжка-передавача. Залежно від результату операції встановлюються ознаки в регістрі ознак, а самі операнди не змінюються.

Команда сканування (переглядання) елементів ланцюжка **SCAS** має такий формат:

CMPS	Ланцюжок-приймач

Ця команда віднімає від значення акумулятора елементи ланцюжка-приймача. Залежно від результату операції встановлюються ознаки в регістрі ознак, а самі операнди не змінюються.

Команда завантаження елементів ланцюжкових даних в акумулятор **LODS** має такий формат:

LODS	Ланцюжок-передавач

За цією командою елементи ланцюжкових даних, що адресуються регістром **SI**, заносяться в акумулятор, при цьому значення в регістрі **SI** змінюється.

Команда збереження вмісту акумулятора в ланцюжку даних **STOS** має такий формат:

STOS	Ланцюжок-приймач

За допомогою цієї команди вміст акумулятора заноситься в пам'ять як елемент ланцюжкових даних.

Команди керування мікропроцесором

Команди керування мікропроцесором можна умовно поділити на дві групи. Команди першої групи призначені для керування бітами ознак у регістрі ознак, а другої групи — для синхронізації мікропроцесора.

До першої групи належать такі команди:

- **CLC** — встановити біт переносу **CF** на нуль;
- **CMC** — інвертувати біт переносу **CF**;
- **STC** — встановити біт переносу **CF** на одиницю;
- **CLD** — встановити біт додаткового переносу **DF** на нуль;
- **STD** — встановити біт додаткового переносу **DF** на одиницю;
- **CLI** — встановити біт переривань **IF** на нуль;
- **STI** — встановити біт переривань **IF** на одиницю.

У другу групу входять такі команди:

- **HLT** — зупинити мікропроцесор;
- **WAIT** — очікувати зовнішню подію;

- **LOCK** — видати сигнал блокування шини $\overline{\text{LOCK}}$;
- **ESC** — переключитися на співпроцесор;
- **NOP** — команда пустої операції.

1. Що таке мікропроцесор?
2. Яка послідовність дій мікропроцесора під час виконання команд?
3. З яких частин складається мікропроцесор?
4. Для чого призначена систем шин мікропроцесора?
5. Які функції виконує пристрій керування?
6. Чим характеризується пам'ять мікропроцесора?
7. Яка структура пам'яті мікропроцесора?
8. Які системи адресації застосовуються в мікропроцесорі?
9. Що таке сегментація пам'яті та її призначення?
10. Які функції виконує система переривань?
11. Що таке система команд мікропроцесора?
12. Схарактеризуйте команди передачі даних.
13. Як обробляється інформація за допомогою арифметичних команд?
14. Які логічні операції може виконувати мікропроцесор?
15. Де застосовуються операції зсуву, і яким чином вони виконуються?
16. У яких випадках виконується передача керування в мікропроцесорі?
17. Як обробляються ланцюжки даних у мікропроцесорах?
18. Як можна керувати мікропроцесором, і які команди для цього використовуються?

13

Використання комп'ютерно-інформаційних технологій для автоматизації промисловості

13.1 Основні поняття

Сучасний етап розвитку суспільства характеризується різким збільшенням обсягів виробництва і номенклатури виробів, зростанням їх складності, широким застосуванням складних машин і механізмів, впровадженням сучасних технологій. Різко зросли інформаційні потоки у виробництві. Здійснення всього цього неможливо без застосування інформаційних і комп'ютерних технологій.

Виробництво товарів здійснюється в ***технологічному процесі***. Технологічний процес є окремим випадком фізичних процесів, що відбуваються у навколишньому світі. Фізичним процесом називається послідовна зміна стану об'єкта. Фізичні процеси протікають у навколишній природі (сходить і заходить сонце, віє вітер, йде дощ, відбувається виверження вулканів, утворюються і зникають циклони, проходять сонячні і місячні затемнення тощо). Розрізняють такі види фізичних процесів:

- механічні (рух фізичних тіл, тобто зміна положення у просторі і часі);
- теплові процеси (нагрівання та охолодження тіл, що пов'язані зі зміною теплової енергії);
- електромагнітні процеси (процеси перетворення електричної та магнітної енергії);
- оптичні процеси (випромінювання чи поглинання світлових променів, інтерференція, дифракція, поляризація);
- хімічні процеси (хімічні реакції, тобто перетворення одних речовин в інші, утворення розчинів);
- процеси ядерного перетворення тощо.

На деякі фізичні процеси людина не може впливати, тобто не може ними керувати, вона може їх тільки спостерігати, наприклад, зміни пір року, утворення припливів і відпливів, землетруси і виверження вулканів.

Фізичні процеси, параметри яких можна вимірювати і на хід яких можна впливати, вважаються технічними процесами.

Технічні процеси, спеціальним чином організовані для виробництва продукції, називаються ***технологічними процесами***. Технологічними процесами є також виробництво енергії, зокрема електроенергії, теплової енергії.

Вплив на технічний процес з метою зміни його протікання в потрібному напрямку називається ***керуванням*** процесом. Керування може здійснюватися безпосередньо людиною, і тоді воно називається ***ручним керуванням***, а може і без участі людини, і в такому разі воно називається ***автоматичним керуванням***. Якщо керування здійснюється за частковою участю людини, то таке керування називається ***автоматизованим***.

Термін автоматичний і автоматизований походить від грецького слова αυτοματοζ — самодіючий, тобто такий, який діє сам по собі, без участі людини. Наука, яка займається вивченням таких пристроїв, їх побудовою і застосуванням на практиці, називається ***автоматикою***. ***Автоматизація*** — це практичне втілення досягнень автоматики в практику для вирішення конкретних завдань керування технологічними процесами.

З автоматикою тісно пов'язана ***кібернетика*** — наука про керування у живій і неживій природі. Термін кібернетика походить від грецького слова κυβερνητηζ — мистецтво водити кораблі.

Структура керування наведена на рисунку 13.1.

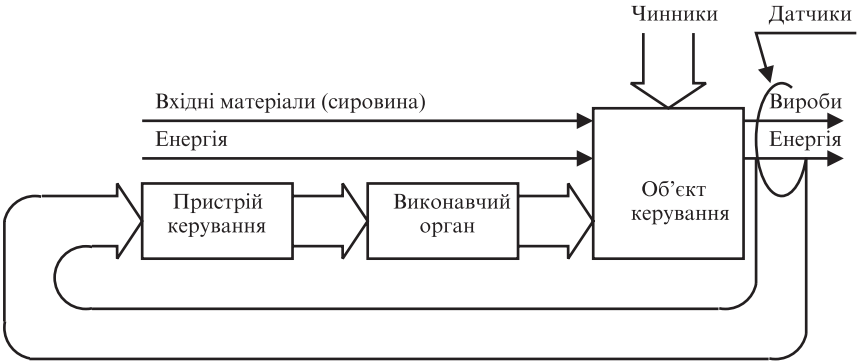


Рис. 13.1

Будь-який технічний, і зокрема технологічний процес як об'єкт керування, можна представити як процес перетворення вхідних матеріалів (сировини) і енергії на вихідні матеріали (готову продукцію) і енергію. Інформація про вхідні і вихідні параметри речовини і енергії сприймається спеціальними технічними пристроями — датчиками. Датчики, виконавчі органи, пристрої керування

та інші допоміжні пристрої — це складова частина ***системи автоматичного керування*** **(САК).** У системах автоматичного керування реалізується ***принцип зворотного зв'язку***: вхідний сигнал об'єкта керування, що формується системою автоматичного керування, залежить від вихідного сигналу. Якщо вихідний сигнал системи автоматичного керування лінійно залежить від вхідного сигналу, то такі системи називаються ***лінійними***, а якщо між вхідним і вихідним сигналами існує складна нелінійна залежність — то ***нелінійними***.

Виробництво є однією з найважливіших сфер людської діяльності, тому значення автоматизації виробництва для розвитку суспільства важко переоцінити. Автоматизація виробництва є надзвичайно складною проблемою, яку необхідно вирішувати на різних рівнях, починаючи від автоматизації елементарних виробничих операцій і закінчуючи автоматизацією на рівні заводу, регіону і країни в цілому.

Особливо великий економічний ефект дає автоматизація елементарних виробничих операцій внаслідок масовості цих операцій. Однією з найпоширеніших є операція переміщення робочих органів (наприклад, переміщення різця у токарному верстаті, фрези у фрезерному верстаті тощо), зокрема у такій важливій галузі виробництва, як машинобудування. Переміщення робочих органів верстатів і їх позиціонування здійснюється за допомогою електроприводів, пневмоприводів та гідроприводів. Приводи з механічними і електромеханічними пристроями керування мають невисоку точність, малу швидкодію, забезпечують нескладні алгоритми керування. Застосування електронних пристроїв керування дало змогу значно поліпшити ці показники, але й вони вже не задовольняють зростаючі вимоги сучасного виробництва.

Значний стрибок у поліпшенні якості приводів дало застосування у пристроях керування приводами спеціальних мікропроцесорів — контролерів. Застосування контролерів дало змогу здійснити цифрове програмне керування приводами, що значно підвищило точність позиціонування, швидкодію приводів. Крім того, це дало змогу реалізувати складні алгоритми керування, переходити від одних алгоритмів до інших у процесі роботи, можливість здійснювати переміщення робочих органів складними траєкторіями руху.

Важливою проблемою для виробництва є перехід від випуску одного виду продукції до іншого або перехід від випуску однієї моделі виробу до іншої. Щоб випускати нову продукцію потрібно замінити одне устаткування іншим, перебудувати технологічний

процес, здійснити перепідготовку персоналу. Сучасне виробництво змушене досить часто переходити на випуск нової продукції, що пояснюється швидким моральним старінням виробів. Яскравим прикладом такої тенденції є виробництво персональних комп'ютерів. Кожна нова модель персонального комп'ютера за своїми параметрами у багато разів перевищує попередню модель, що була випущена менше року тому, тобто моральне старіння йде набагато швидшими темпами ніж фізичне. Крім того, потрібно враховувати також гостру конкуренцію на ринках. Таким чином, для підтримання високого рівня конкурентноспроможності виробів необхідно час від часу переходити на випуск нової продукції, а з іншого боку кожний такий перехід пов'язаний зі значними затратами. Щоб досягти компромісу між цими тенденціями, застосовуються ***гнучкі автоматизовані виробничі комплекси*** **(*ГВК*)**. Вони складаються з окремих, відносно самостійних частин, так званих ***гнучких автоматизованих виробничих модулів*** **(*ГВМ*)**. Кожний такий модуль здатний виконувати певний комплекс виробничих операцій. Перехід на інший комплекс операцій здійснюється програмним шляхом, тобто заміною однієї програми іншою. Перехід на випуск нової продукції здійснюється зміною послідовності і номенклатури модулів, а також перепрограмуванням у разі потреби цих модулів.

Застосування гнучких виробничих комплексів дає змогу різко збільшити ефективність виробництва, підвищити якість і конкурентноспроможність продукції.

Важливим засобом інтенсифікації виробництва є ***роботизація***, тобто застосування у виробництві промислових роботів. ***Промисловий робот*** — це технічний пристрій, призначений для виконання комплексу виробничих операцій в автоматичному режимі. У виробництві застосовується велика кількість різновидів і типів роботів і робототехнічних комплексів від найпростіших до складних інтелектуальних роботів, здатних самостійно приймати рішення на основі отриманої інформації у складних виробничих умовах, адаптуватися до змін у навколишньому середовищі. У роботах і робототехнічних комплексах знайшли застосування останні досягнення інформаційної техніки: пристрої і системи сприйняття інформації, цифрові пристрої і мікропроцесори для перетворення і обробки інформації, приводи робочих органів з цифровим програмним керуванням, сучасні програмні засоби.

Для робототехнічного виробництва характерним є те, що це виробництво здійснюється без участі або майже без участі людини. Застосування роботів дає змогу звільнити людину від важкої

одноманітної праці, від роботи у шкідливих для організму людини умовах, а також у недоступних для людини середовищах.

Сучасні програмні та апаратні засоби дають змогу автоматизувати не тільки окремі виробничі операції чи комплекс операцій, а й весь технологічний процес у цілому. Такі системи називаються ***автоматизованими системами управління технологічним процесом*** **(АСУ ТП).** Особливо значний ефект дає застосування таких систем у галузях неперервного виробництва, зокрема у хімічній і нафтогазовій галузях. Виробництво сірчаної, азотної та інших кислот, мінеральних добрив, переробка нафти та інші процеси здійснюються із застосуванням АСУ ТП.

Вироби, що випускаються сучасними підприємствами, складаються з великої кількості деталей, вузлів, систем. Для їх виробництва необхідна велика кількість матеріалів і заготовок. Щоб вчасно і безперебійно постачати виробництво деталями, заготовками, напівфабрикатами і матеріалами, підприємству необхідно мати складну і потужну транспортно-складську систему. Для здійснення керування такими значними матеріальними потоками застосовуються ***автоматизовані транспортно-складські системи*** **(АТСС).**

Сучасне підприємство є складною ієрархічною системою, яка характеризується значними матеріальними, енергетичними, інформаційними потоками, тому керувати такою системою дуже складно. Застосування сучасних апаратних і програмних засобів для автоматизації керування підприємством, його підрозділами і службами дає змогу значно підвищити ефективність виробництва, зменшити собівартість продукції і підвищити її якість. Такий комплекс апаратних, програмних та інших засобів називається ***автоматизованою системою управління підприємством*** **(*АСУП*).** Автоматизована система управління підприємством, як і саме підприємство, є складною багаторівневою ієрархічною системою, що складається з підсистем різного рівня. Наприклад, системи керування технологічними процесами (АСУ ТП), автоматизовані транспортно-складські системи (АТСС), робототехнічні комплекси і гнучкі автоматизовані виробничі комплекси можуть входити як підсистеми до АСУП. Крім того, АСУП має, як правило, підсистему бухгалтерського обліку, економічного аналізу, обліку кадрів тощо. Ступінь автоматизації на кожному рівні ієрархії може бути різним. Різними за складом, функціями та параметрами є також апаратні й програмні засоби на різних рівнях ієрархії.

13.2 Пристрої для сприйняття інформації про об'єкт керування

Керування об'єктом, зокрема технологічним процесом, можливе лише за умови, що на систему керування поступає інформація про стан об'єкта. Для сприйняття інформації про стан керованого об'єкта призначені спеціальні вимірювальні пристрої. Такі вимірювальні пристрої називають ***первинними вимірювальними перетворювачами***, ***рецепторами***, ***сенсорами***. Елемент такого вимірювального пристрою, який безпосередньо взаємодіє з об'єктом керування і сприймає інформацію, називається ***чутливим елементом***. Вимірювальний перетворювач, який разом з допоміжними елементами виготовлений у вигляді окремого виробу і встановлений безпосередньо на об'єкті керування, називається ***датчиком***.

Характеристики датчиків

Для забезпечення потрібного рівня керування об'єктом необхідно, щоб параметри датчиків відповідали заданим вимогам. Характеристики датчиків поділяються на статичні, які характеризують датчик у ***статичному режимі***, і динамічні, що характеризують його в ***динамічному режимі***. Статичним режимом датчика називається режим роботи, коли на вході датчика діє ***стала*** фізична величина. Однак стала фізична величина — лише зручна математична модель, абстракція, яка застосовується для теоретичного аналізу. В природі всі величини змінюються з тією чи іншою швидкістю. З іншого боку, реальні технічні пристрої, зокрема датчики, мають інерцію, тобто реагують на вхідну дію не миттєво, а з деяким запізненням. Враховуючи все це, доцільно вважати ***статичним режимом*** реального технічного пристрою (датчика) такий режим, коли інерційними властивостями пристрою можна знехтувати порівняно зі швидкістю зміни сигналу на його вході.

Динамічним режимом роботи технічного пристрою вважається режим роботи, в якому динамічні характеристики пристроїв, обумовлені інерційністю, істотно впливають на результат роботи.

Характеристики датчиків у статичному режимі. Найбільш повно властивості датчиків у статичному режимі представляє функція перетворення (статична характеристика).

Функція перетворення — це залежність вихідної фізичної величини датчика y від вхідної x, тобто $y = f(x)$. Для більшості

датчиків залежність $y = f(x)$ — нелінійна (рис. 13.2). Функція перетворення характеризує вимірювальний перетворювач у всьому діапазоні зміни вхідної і вихідної величини і є повною статичною характеристикою. В окремих випадках достатньо знати тільки окремі параметри вимірювального перетворювача.

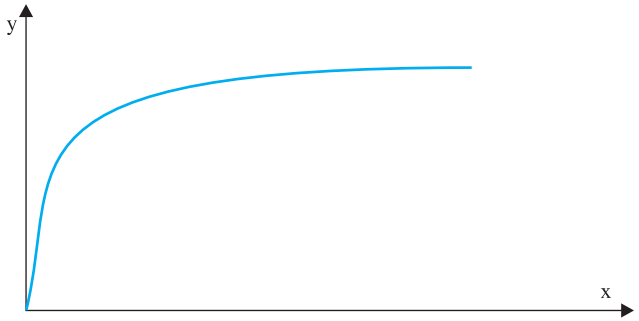


Рис. 13.2

Одним із таких важливих параметрів є ***чутливість*** (*sensivity*), яка визначається як відношення вихідної величини вимірювального перетворювача y до його вхідної величини x:

$$S = \frac{y}{x}. \qquad (13.1)$$

Крім статичної чутливості, що визначається формулою (13.1), користуються також динамічною чутливістю

$$S = \frac{\Delta y}{\Delta x}. \qquad (13.2)$$

Роздільна здатність датчика (*resolution*) — це властивість розрізняти два близькі значення вхідної величини.

Лінійність датчика — це ступінь наближення функції перетворення датчика до прямої лінії. Для аналого-цифрового перетворення сигналу необхідна пряма пропорційна залежність між вихідним значенням перетворювача і фізичною величиною на вході датчика, тому лінійність датчика є дуже важливим параметром. Якщо функція перетворення датчика є істотно нелінійною, то необхідне додаткове перетворення вихідного сигналу датчика, щоб лінеаризувати канал вимірювання в цілому.

Робочий або ***динамічний*** діапазон датчика (*operating range*) — це діапазон зміни вхідної величини, в якому параметри датчика не виходять за задані межі.

Повторюваність (*repeatability*) — це властивість датчика зберігати свої параметри від екземпляра до екземпляра.

Відтворюваність (*reproducivity*) — властивість відтворити задані характеристики датчика під час його виготовлення.

Вхідний і ***вихідний*** імпеданс датчика — це вхідний і вихідний повні опори. Ці характеристики датчиків дуже важливі для узгодження параметрів датчика з іншими пристроями в каналі вимірювання, а також для оцінки впливу датчика на досліджуваний процес.

Характеристики датчиків у динамічному режимі. Якість керування багато в чому залежить від того, як швидко система керування в цілому та її складові частини, зокрема датчики, реагують на зміни в об'єкті керування. В деяких випадках несвоєчасна реакція систем автоматичного керування на зміни в об'єкті керування призводить до аварійних ситуацій і виходу з ладу. Динамічні характеристики показують ступінь інертності датчиків.

Динамічні характеристики поділяються на повні, які мають всю повноту інформації про інерційні властивості об'єктів, і часткові, які характеризують тільки певні властивості об'єктів.

До повних динамічних характеристик належать: імпульсна і перехідна характеристики в часовій області й амплітудно-частотна, фазочастотна та комплексна характеристики в частотній області.

Перехідною характеристикою датчика $h(x)$ називається реакція датчика, тобто його вихідний сигнал, на одноступінчатий сигнал на вході (рис. 13.3).

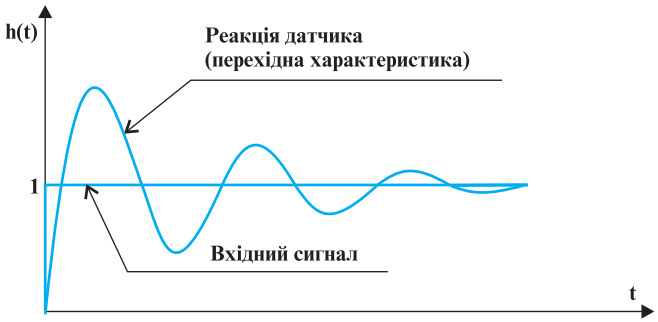


Рис. 13.3

Імпульсною характеристикою датчика $g(x)$ називається реакція датчика на вхідний імпульсний сигнал надзвичайно малої тривалості (рис. 13.4). З математичної точки зору імпульсна характеристика є похідною від перехідної характеристики.

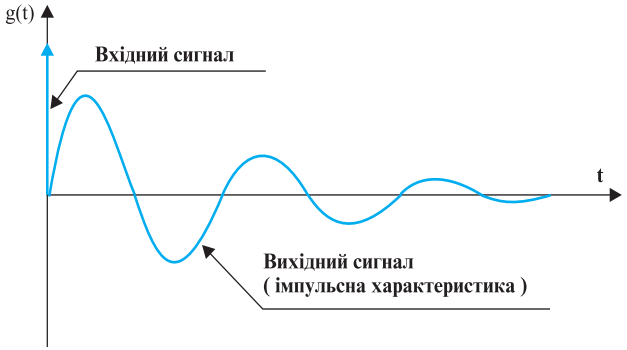


Рис. 13.4

Амплітудно-частотною характеристикою датчика називається залежність відношення амплітуд вихідного і вхідного синусних сигналів від частоти:

$$K(\omega) = \frac{A_{вих}(\omega)}{A_{вх}(\omega)}. \qquad (13.3)$$

Фазочастотна характеристика датчика — це залежність різниці фаз вихідного і вхідного сигналів від частоти.

$$\varphi(\omega) = \psi_{вих}(\omega) - \psi_{вх}(\omega). \qquad (13.4)$$

Комплексна частотна характеристика — це комплексна функція, модулем якої є амплітудно-частотна, а аргументом — фазочастотна характеристики.

$$K(j\omega) = K(\omega)e^{j\varphi(\omega)}; \quad K(\omega) = \left| K(j\omega) \right| ; \varphi(\omega) = \arg(K(j\omega)) \qquad (13.5)$$

Часткові динамічні характеристики містять інформацію про найбільш важливі властивості датчиків. До часткових динамічних характеристик, які можна визначити за перехідною характеристикою, належать такі характеристики:

- ❒ час проходження зони нечутливості (dead time);
- ❒ запізнення (delay time), яке визначається відрізком часу від моменту подачі вхідного ступінчастого сигналу до моменту

досягнення вихідним сигналом 50 % рівня усталеного значення;

- ❒ тривалість наростання сигналу (rise time), яка визначається інтервалом часу від моменту досягнення вихідним сигналом рівня 10 % до моменту досягнення 90 % від усталеного значення;
- ❒ момент досягнення першого максимуму (peak time);
- ❒ тривалість перехідного процесу (settling time);
- ❒ відносне перерегулювання (percentag overshot);
- ❒ статична похибка (steady-state error).

Аналогові датчики

Переважна більшість об'єктів керування характеризується неперервними фізичними величинами, які поступають на вхід датчиків.

Вихідним сигналом аналогового датчика є неперервна фізична величина. За видом вхідної величини аналогові датчики поділяються на такі види: датчики руху (кутового і лінійного переміщення, швидкості прискорення), датчики сили, моменту, тиску, датчики наближення (індуктивні, ємнісні, магнітні), датчики температури, датчики витрати, хімічні і біохімічні датчики.

Датчики руху (motion sensors). Датчики руху широко застосовують для автоматизації технологічних процесів у машинобудуванні, наприклад, для автоматичного керування робочими органами різноманітних верстатів (токарних, фрезерних, шліфувальних тощо) і роботів. Механічний рух характеризується такими кінематичними величинами: переміщенням (зміною положення, відстані, ступеня наближеності, розміру), лінійною і кутовою швидкістю, прискоренням тощо. Датчики руху ґрунтуються на різноманітних фізичних принципах. Розглянемо найбільш поширені датчики руху.

Реостатний перетворювач. Це резистор з рухомим контактом, що змінює своє положення залежно від зміни вхідної величини (лінійного або кутового переміщення). Реостатний перетворювач можна ввімкнути у вимірювальну схему послідовно (рис. 13.5, *а*) або паралельно (рис. 13.5, *б*).

а) *б*)

Рис. 13.5

Залежно від конструкції реостатного перетворювача його статична характеристика може бути плавною або східчастою (рис. 13.6).

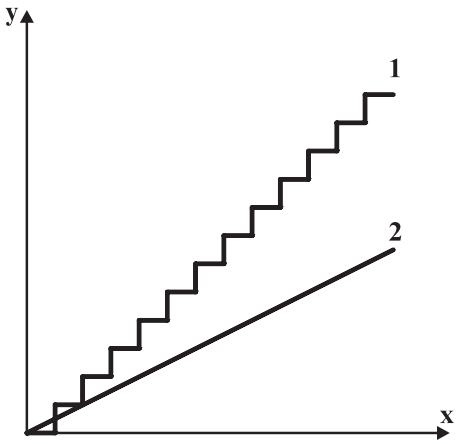


Рис. 13.6

Плавну статичну характеристику мають струнні реостатні перетворювачі у вигляді проводу, по якому ковзає рухомий контакт (рис. 13.7, *а*).

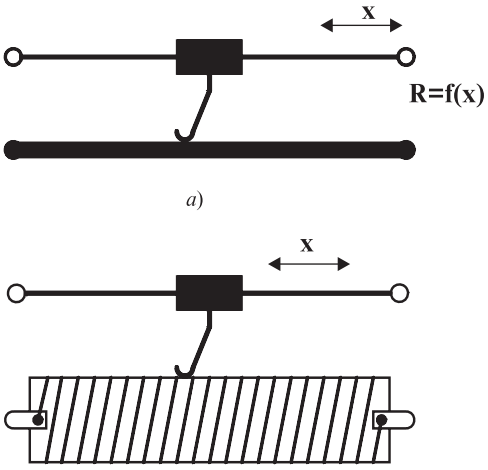


Рис. 13.7

Східчаста характеристика властива для реостатного перетворювача каркасного типу (рис. 13.7, *б*). Реостатний перетворювач

з кутовим вхідним переміщенням наведений на рисунку 13.8. У таких перетворювачів тонкий провід в ізоляції намотується в один ряд на каркас з ізоляційного матеріалу. Рухомий контакт переміщується по поверхні, оголеній від ізоляції. Під час переміщення рухомого контакту з одного витка на другий відбувається стрибкоподібна зміна вихідного опору резистора. Для виготовлення каркаса застосовують такі ізоляційні матеріали як текстоліт, гетинакс, кераміка, ебоніт тощо. Для намотки застосовують тонкий провід (0,03...0,3 мм) з манганіну, константану, ніхрому та інших високоомних спеціальних сплавів. Інколи використовуються реостатні перетворювачі із заданою функціональною залежністю вихідного опору від вхідного переміщення. Для виготовлення таких перетворювачів використовується каркас спеціальної форми, профіль якого відтворює задану функціональну залежність.

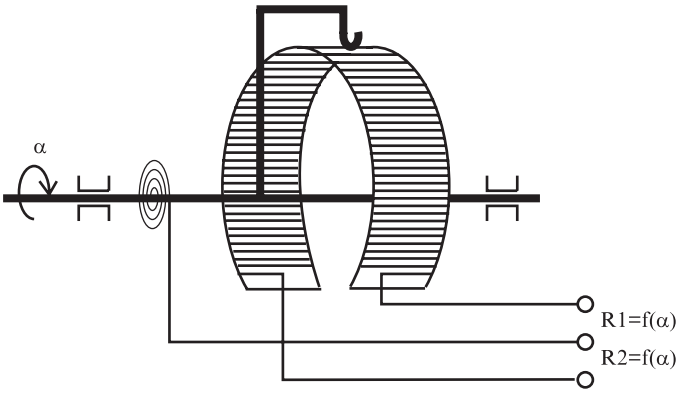


Рис. 13.8

Індуктивні і взаємоіндуктивні перетворювачі. Індуктивний і взаємоіндуктивний перетворювачі складаються з котушки і магнітопроводу із зазором. Дія індуктивного (рис. 13.9) і взаємоіндуктивного перетворювача (рис. 13.10) ґрунтується на залежності індукції (взаємної індукції) перетворювача від переміщення якоря (магнітопроводу або його частини). Від переміщення якоря змінюється зазор у магнітопроводі котушки індуктивності чи взаємної індуктивності, отже змінюється магнітний опір магнітопроводу. Зміна магнітного опору призводить до зміни індуктивності, від якої залежить повний опір котушки.

Індуктивні перетворювачі бувають одинарними і диференціальними. Диференціальний перетворювач складається з двох однакових одинарних перетворювачів, увімкнених таким чином, що

переміщення якоря призводить до рівних за модулем і протилежних за знаком змін індуктивності перетворювачів. Сумарна зміна індуктивності диференціального перетворювача в цілому дорівнює подвоєній зміні індуктивності кожного одинарного перетворювача.

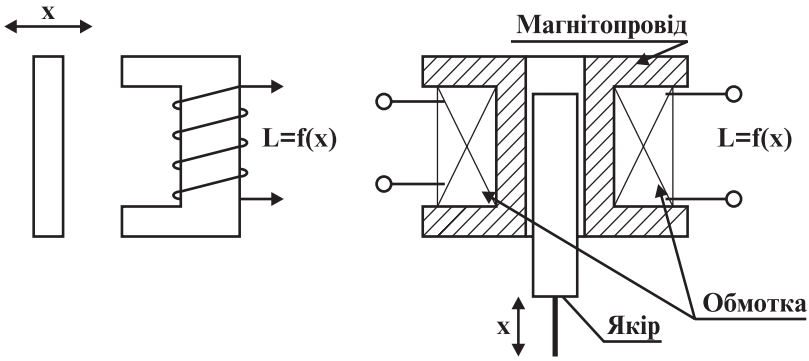


Рис. 13.9

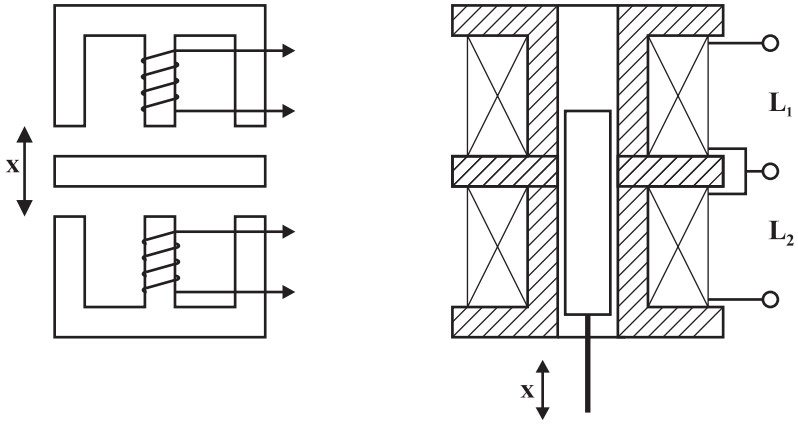


Рис. 13.10

Трансформаторні перетворювачі відрізняються від індуктивних наявністю ще однієї (вимірювальної) обмотки. Як і індуктивні перетворювачі, трансформаторні перетворювачі бувають одинарні (рис. 13.11) і диференціальні (рис. 13.12).

Ємнісний датчик. Ємнісний перетворювач — це плоский, рідше циліндричний конденсатор, ємність якого залежить від площі електродів ***S***, відстані між електродами δ і діелектричної проникності

матеріалу ε, що знаходиться між електродами (рис. 13.13). Для плоского конденсатора ця залежність виражається формулою:

$$C = \varepsilon\varepsilon_0 \frac{S}{\delta}. \qquad (13.6)$$

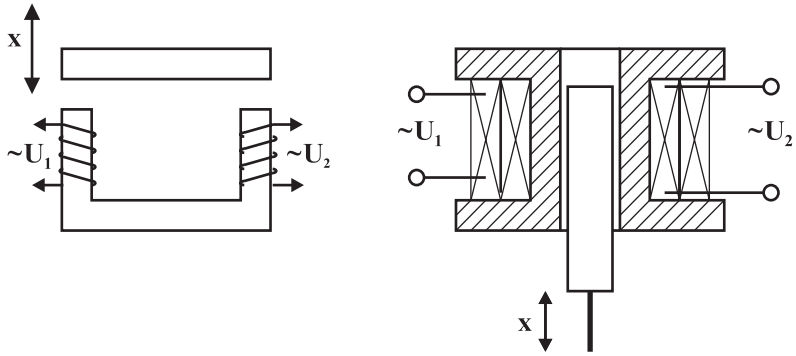


Рис. 13.11

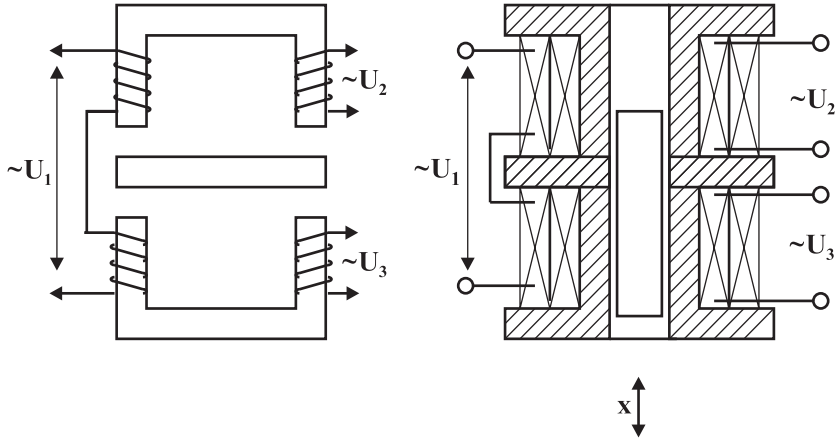


Рис. 13.12

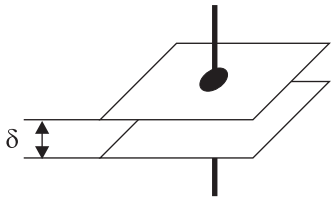


Рис. 13.13

Як датчики переміщення використовуються плоскі конденсатори, ємність яких залежить від відстані δ між електродами. Такі конденсатори вмикаються, як правило, в одне плече моста змінного струму, напруга в діагоналі якого залежатиме від ємності цього конденсатора, або у коливальний контур, частота власних коливань якого залежатиме від ємності.

Лазерні датчики. Для вимірювання з високою точністю відстаней застосовуються останнім часом лазерні датчики, принцип дії яких ґрунтується на залежності часу проходження світловим імпульсом від відстані між предметами.

Датчики кутового переміщення. У станках, маніпуляторах, робототехнічних комплексах широко застосовується обертальний рух, тому вимірювання кутового переміщення в широкому діапазоні і з високою точністю дуже важливе. Найбільше поширення знайшли перетворювачі кутового переміщення в різницю фаз електричних коливань. Одним з таких перетворювачів є синус-косинусний обертовий трансформатор (СКОТ). СКОТ — це електрична мікромашина змінного струму з двофазними обмотками на статорі і роторі. Обмотки СКОТ сконструйовані таким чином, що взаємна індуктивність відповідних фаз ротора і статора змінюється від кута повороту ротора з високим ступенем точності за законами синуса і косинуса.

Крім синус-косинусного обертового трансформатора широко використовуються також індуктосини. ***Індуктосин*** — це також електрична мікромашина змінного струму з кількома десятками полюсів, взаємна індуктивність між відповідними обмотками статора і ротора залежить від кута повороту ротора.

Щоб підвищити точність датчиків кутового переміщення, їх включають у компенсаційну схему перетворення різниці фаз у цифровий код.

Датчики швидкості обертання. За формою вихідного сигналу датчики швидкості обертання поділяються на аналогові, імпульсні і цифрові.

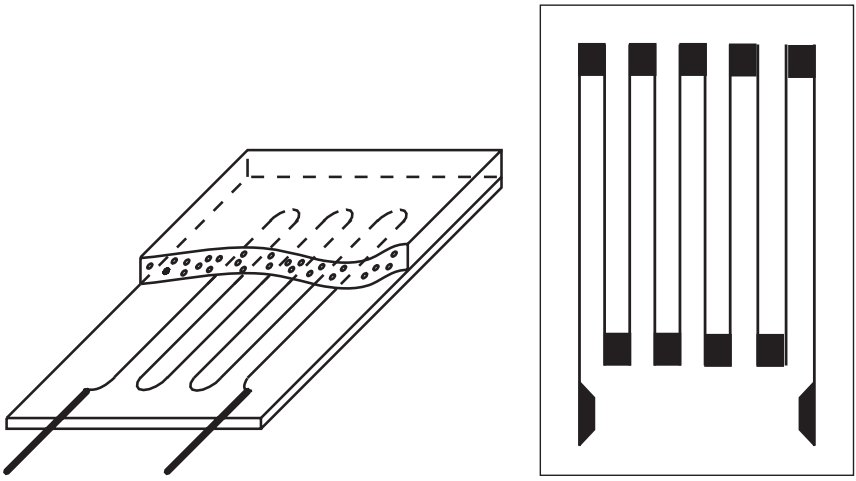
Як аналогові датчики швидкості обертання широкого застосування набули тахогенератори постійного і змінного струму.

Тахогенератор постійного струму — це електрична мікромашина постійного струму. Ротор такої мікромашини обертається в магнітному полі постійного магніту. Напруга, яка знімається з колектора, прямо пропорційна швидкості обертання.

У тахогенераторі змінного струму амплітуда і частота змінної напруги, наведеної магнітним полем ротора, прямо пропорційні швидкості обертання ротора.

Датчики прискорення (акселерометри). Датчики прискорення широко застосовуються в автоматичних системах керування рухомими об'єктами, зокрема літаками, ракетами тощо. Принцип дії акселерометрів ґрунтується на перетворенні прискорення у силу інерції відповідно до другого закону Ньютона $F = ma$. Далі сила перетворюється у переміщення, яке, в свою чергу, перетворюється в електричну величину (напругу, струм тощо).

Датчики сили, моменту, тиску. В цих датчиках сила, момент, тиск перетворюються на деформацію пружного елемента, яка сприймається датчиками, що називаються ***тензорезисторами***. Електричний опір тензорезистора залежить від деформації. За конструкцією і технологією виготовлення розрізняють провідникові, фольгові і плівкові тензорезистори. Провідниковий тензорезистор (рис. 13.14, *а*) — це зигзагоподібної форми провідник діаметром 0,02...0,005 мм, виготовлений з константану, наклеєний спеціальним клеєм на смужку паперу і покритий зверху лаком. Для вимірювання деформацій смужку паперу з тензорезистором наклеюють на досліджувану деталь або на пружний елемент.



а) *б*)

Рис. 13.14

Крім провідникових, застосовують також фольгові тензорезистори (рис. 13.14, *б*), які виготовляються хімічним травленням тонкого (близько 10 мкм) шару тензорезистивного матеріалу, нанесеного на діелектричну підкладку. Після травлення на підкладці залишається захищений шар тензорезистивного матеріалу у вигляді послідовного зигзагоподібного з'єднання тонких і вузьких провідників.

Ще одна технологія виготовлення тензорезисторів полягає у вакуумному осадженні розпиленого тензорезистивного матеріалу на діелектричну підкладку. Такі тензорезистори називаються ***плівковими***.

За використаним тензорезистивним матеріалом тензорезистори поділяються на провідникові і напівпровідникові.

Датчики наближення (*proximity sensors*). Датчики наближення широко застосовуються для автоматизації процесів у машинобудуванні, наприклад, при підрахунку кількості деталей на конвеєрі, у робототехнічних комплексах, в охоронних системах. Принцип дії датчиків наближення ґрунтується на зміні властивостей чутливого елемента при наближенні до нього певного об'єкта.

В ***індуктивного*** датчика наближення котушка, ввімкнена у коливальний контур, створює високочастотне електромагнітне поле, яке наводить у провідному матеріалі об'єкта, що наближається, вихрові струми. Вихрові струми призводять до втрат енергії, тому амплітуда коливань у контурі зменшується.

Ємнісні датчики наближення як чутливий елемент мають конденсатор, увімкнений у коливальний контур. Об'єкт, що наближається, змінює ємність конденсатора, отже і частоту власних коливань контура.

Магнітні датчики реагують на зміну характеристик магнітного поля і можуть базуватися на магніторезистивному ефекті, ефекті Холла, зміні магнітного опору тощо.

Датчики температури. Температура є важливим технологічним параметром, тому датчики температури широко застосовуються при автоматизації технологічних процесів у хімічній, текстильній, нафтовій та газовій промисловості. Найбільшого поширення набули такі датчики температури, як термоелементи і терморезистори.

Термоелемент — це з'єднані в одній точці провідники з різних металів. На межі двох різних металів виникає контактна різниця потенціалів, значення якої залежить від температури. Інколи термоелемент називають термопарою.

Для виготовлення позитивних полюсів термоелемента використовують мідь та сплави: хромель (Cr+Ni) або платинородій (Pt+Rh); для виготовлення негативних полюсів — сплави константан (Cu+Ni+Mn), копель (Cu+Ni), алюмель (Al+Si+Mg+Ni).

Терморезистор — це резистор, опір якого залежить від температури. Для виготовлення резисторів використовують мідь і платину. У провідникових металевих резисторах опір практично лінійно залежить від температури $R(t) = R_0 + \alpha t + \beta t^2 + \ldots$.

Коефіцієнти α, β для платинового терморезистора мають значення $\alpha = 0{,}004\ K^{-1}$, $\beta = 0{,}59 \cdot 10^{-6}\ K^{-2}$.

Терморезистор, виготовлений з напівпровідникового матеріалу, називається ***термістором***. Опір термістора нелінійно залежить від температури

$$R(t) = R_0 e^{\beta(1/T - 1/T_0)}, \qquad (13.7)$$

де T — температура (К),

R_0 — опір при опорній температурі (298 К),

β — стала (3000...5000 К).

Нелінійну залежність опору від температури лінеаризують аналоговим або цифровим способами.

Датчики витрати. Вимірювання витрати рідин і газів необхідно проводити в хімічній, нафтогазовій, харчовій та інших галузях промисловості.

Датчики витрати ґрунтуються на різноманітних фізичних принципах. ***Дросельний*** датчик витрати ґрунтується на законі Бернуллі для руху рідини: витрата рідини, що рухається у трубопроводі, стала для будь-якого перерізу, тому тиск рідини залежить від площі поперечного перерізу потоку. Різниця тисків рідини у звичайному і звуженому місцях трубопроводу буде прямо пропорційною витраті рідини. Щоб створити звуження трубопроводу, застосовують діафрагми або трубу Вентурі.

Якщо відома площа поперечного перерізу трубопроводу, то витрати рідини можна знайти, знаючи швидкість рідини. Швидкість рідини в трубопроводі можна визначити за частотою обертання маленької турбінки, поміщеної в потоці рідини.

Ультразвукові датчики витрати використовують поширення ультразвуку в двох напрямах: за напрямом руху рідини і проти напряму руху. Якщо напрям руху рідини й ультразвуку збігається, то швидкість ультразвуку в рухомій рідині дорівнює сумі швидкостей ультразвуку в нерухомій рідині і швидкості самої рідини, а якщо напрямки руху протилежні — то різниці цих швидкостей. Вимірюючи швидкість ультразвуку в цих двох протилежних напрямках, можна визначити швидкість руху рідини, а відтак, знаючи площу поперечного перерізу трубопроводу, і витрату рідини.

У ***магнітних датчиках витрати*** вихідною величиною є електрорушійна сила, що створюється в електропровідній рідині під час руху її в магнітному полі. Значення наведеної електрорушійної сили визначається швидкістю руху іонів електропровідної рідини, отже це значення прямо пропорційне витраті рідини.

Бінарні, імпульсні і цифрові датчики

Бінарні датчики — це датчики, вихідний сигнал яких може бути тільки у двох альтернативних станах, наприклад увімкнено-вимкнено. До бінарних датчиків належать датчики положення (positions sensor). Як датчики положення використовуються різного роду вимикачі. Одним з видів датчиків положення є так звані кінцеві вимикачі (limit switch), які призначені для фіксації меж робочого ходу виконавчих пристроїв.

Імпульсні датчики мають вихідний сигнал у вигляді імпульсів. Найбільшого поширення набули імпульсні датчики кутового переміщення вала і положення (позиції) вала. Перетворення кутового переміщення в кількість імпульсів може ґрунтуватися на таких фізичних принципах: відбивання і переривання світлового потоку (фотоелектричні імпульсні датчики), стрибкоподібна зміна взаємної індуктивності (індуктивні імпульсні датчики) або взаємної ємності (ємнісні імпульсні датчики) тощо. Недоліком імпульсних датчиків є великі похибки під час перебою в роботі і пропуск одного або кількох імпульсів.

Цифрові датчики перетворюють вхідну фізичну величину (здебільшого це кутове чи лінійне переміщення) у код, тобто в одному пристрої суміщено чутливий елемент і аналого-цифровий перетворювач. Для аналого-цифрового перетворення кутового чи лінійного переміщення використовують ряд паралельних чи концентричних доріжок, кожна з яких поділена на однакові ділянки. Властивості двох сусідніх ділянок кожної доріжки різко відрізняються одна від одної, наприклад: прозора-непрозора, намагнічена-ненамагнічена, провідна-непровідна тощо. На кожну доріжку встановлено чутливий елемент, що фіксує, яка саме ділянка знаходиться у даному положенні. Сукупність вихідних сигналів чутливих елементів є кодом кутового чи лінійного переміщення відносно початкового положення.

13.3 Виконавчі пристрої

Для здійснення впливу системи автоматичного керування на об'єкт керування призначені ***виконавчі пристрої*** або ***механізми*** (*aktuators*). Якщо датчики перетворюють фізичні величини, що характеризують об'єкт керування, в електричний сигнал, то виконавчі пристрої здійснюють обернену дію, перетворюють сигнал системи керування у фізичну величину, що змінює перебіг технологічного

процесу в потрібному напрямі. У сучасних автоматичних системах керування основні операції обробки інформації виконує комп'ютер або мікропроцесор, тому виконавчі пристрої мають здійснювати перетворення цифрового вихідного сигналу комп'ютера у фізичну величину. Наприклад, у станках з числовим програмним управлінням (ЧПУ) вихідний цифровий сигнал з керівного мікропроцесора перетворюється у переміщення робочого органу станка (різця, фрези тощо) і переміщення деталі, що обробляється на цьому станку. У хімічних процесах цифровий сигнал перетворюється у переміщення робочих органів, що регулюють надходження вхідних реагентів, температуру в реакторі тощо.

У складі виконавчого пристрою можна виділити дві частини: малопотужну частину, яка складається з ***перетворювача*** (*transducer*) і ***підсилювача*** (*amplifier*), і потужну частину, що складається з ***потужного перетворювача*** (*converter*) і ***вихідного виконавчого механізму***. В деяких виконавчих механізмах окремі частини можуть бути відсутніми.

Виконавчі механізми характеризуються такими параметрами, як: точність, робочий діапазон, швидкодія, потужність, габарити тощо.

Виконавчі механізми поділяються на двопозиційні (бінарні) й аналогові.

Двопозиційні (бінарні) виконавчі механізми

У системах автоматичного керування досить поширені бінарні виконавчі механізми. За родом фізичної величини двопозиційні виконавчі пристрої поділяються на електричні, механічні, гідравлічні, пневматичні тощо.

Електричні двопозиційні виконавчі пристрої це: вимикачі, перемикачі, комутатори, контактори, реле тощо.

Потужність вихідних сигналів комп'ютера дуже мала (<100 мВт), тому безпосередньо подавати такий сигнал на виконавчі пристрої не можна. Їх необхідно спочатку підсилити. Для цього використовуються керовані вимикачі.

Електромагнітні реле

До винайдення і застосування на практиці напівпровідникових приладів як керований вимикач застосовувалися ***електромагнітні реле***, які продовжують широко використовуватися і в наш час. Електромагнітне реле (рис. 13.15) складається з електромагніта, по обмотці якого протікає струм керування, і контактів, що механічно переміщуються під дією магнітного поля, створеного електромагнітом, замикаючи чи розмикаючи електричне коло виконавчого пристрою.

Таким чином, за допомогою електромагнітного реле можна керувати значними струмами, використовуючи незначні.

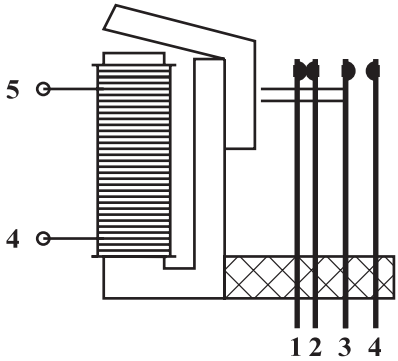


Рис. 13.15

Широкого застосування набули малогабаритні без'якірні електромагнітні реле постійного струму з магнітокерованими герметизованими контактами, так звані ***геркони***. Геркон (рис. 13.16) складається зі скляного балона, всередині якого створено технічний вакуум і розміщено пружні феромагнітні елементи з електричними контактами.

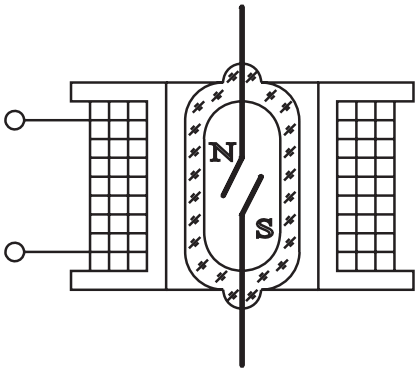


Рис. 13.16

Скляний балон з контактами розміщують у магнітному полі, створеному струмом в обмотці. Якщо вільні кінці контактних пластин намагнічені так, що мають різнойменні полюси, то при ввімкненні струму в обмотці, вони притягуються один до одного (замикаючий контакт), а якщо однойменні полюси — то відштовхуються (розмикаючий контакт). Можна створити також перемикаючі контакти.

Розміщення контактів у вакуумі збільшує опір між контактами в розімкненому стані і значно зменшує окислення контактів, що призводить, у свою чергу, до зменшення перехідного опору в місці контакту і до значного збільшення надійності роботи. Для зменшення окислення контактів, їх покривають тонким шаром неокислюваного металу (як правило, золота).

Важливою властивістю електромагнітного реле є електрична ізоляція керівного і виконавчого електричних кіл, завдяки чому значні струми виконавчих пристроїв не впливають на кола керування, зокрема на коло керування комп'ютера. Електромагнітні реле мають незначний опір контактів у замкненому стані (десяті і соті частки ома) і великий опір, який визначається опором повітряного проміжку в розімкнутому стані.

До недоліків електромагнітних реле слід віднести низьку швидкодію (кілька мілісекунд) порівняно з напівпровідниковими перемикачами, швидкодія яких становить мікросекунди і навіть частки мікросекунд. Крім того, для реле, як і для механічних перемикачів, характерне «брязкання» контактів, тобто багаторазове вмикання і розмикання контактів у перехідному процесі внаслідок механічної інерції після ввімкнення струму в обмотку реле. Обмотка електромагнітного реле вмикається, здебільшого, під номінальну напругу 12 В і споживає струм кілька десятків міліампер, тому не можна безпосередньо керувати реле з виходу комп'ютера, отже потрібний додатковий проміжний підсилювач, наприклад, транзисторний, який встановлюється між виходом комп'ютера і обмоткою реле.

Промисловістю випускаються різноманітні реле в широкому діапазоні потужностей від мініатюрних поляризованих реле потужністю кілька міліват (такі реле часто встановлюють на платах розширення комп'ютерів) до кіловатних контакторів, які встановлюються в окремих стійках, і призначені для керування двигунами значної потужності.

Напівпровідникові вимикачі. Для керування деякими пристроями, наприклад, для керування виконавчими двигунами на основі широтно-імпульсної модуляції, необхідно забезпечити перемикання струмів в електричних колах з швидкодією у кілька мікросекунд. Електромагнітні реле неспроможні забезпечити таку швидкодію, тому застосовують напівпровідникові прилади: біполярні і польові транзистори та тиристори.

Транзистор, що працює в ключовому режимі, виконує функцію керованого вимикача. ***Ключовим режимом*** транзистора називається такий режим, коли транзистор знаходиться в одному

з двох станів: стані насичення і стані відсікання. У стані насичення транзистор відкритий, тобто опір між електродами транзистора незначний, і через нього протікає струм. Стан насичення відповідає замкненому вимикачу. У стані відсікання транзистор закритий, між електродами транзистора значний опір. Стан відсікання відповідає розімкненому вимикачу.

Для керування пристроями з невеликою потужністю можна застосовувати транзистори загального призначення. Щоб керувати пристроями, в яких використовуються напруги більші за 1000 В і струми у кілька кілоампер, призначені напівпровідникові прилади, виготовлені за спеціальними технологіями.

Крім біполярних і польових транзисторів, для комутації потужних електричних кіл застосовуються тиристори. Тиристори можуть знаходитися в одному з двох станів: провідному, коли опір між катодом і анодом становить частки ома, і непровідному, коли опір між електродами становить десятки мегом. Щоб перевести тиристор у провідний стан, на керівний електрод слід подати імпульс. Тиристори порівняно з транзисторами мають значно менший опір у замкненому стані, тому кількість теплової енергії, що розсіюється тиристором, значно менша ніж у транзистора. У той же час, за швидкодією транзистори переважають тиристори.

На відміну від електромагнітних реле, у напівпровідникових приладах немає ***гальванічного розмежування*** між вхідними і вихідними електричними колами, і це великий недолік напівпровідникових перемикачів. У польових транзисторів вхідний опір набагато більший ніж у біполярних, і це забезпечує значно менший ступінь впливу вихідних кіл на вхідні. Для забезпечення гальванічного розмежування вхідних і вихідних кіл застосовують передачу сигналу через оптичний канал: керівний сигнал з комп'ютера подають на світлодіод, де він перетворюється у світловий імпульс, що діє на фототранзистор або фототиристор, і переводить його у провідний стан. Сукупність керованого джерела випромінювання (світлодіода) і приймача випромінювання (фототранзистора чи фототиристора), з'єднаного оптичним каналом і розміщеного в одному корпусі, називається ***оптроном***.

У системах керування з електричними перемикачами слід особливу увагу звернути на розмикання електричних кіл, що містять елементи зі значною індуктивністю: котушки електромагнітів, реле, обмотки електричних машин тощо. У таких елементах накопичена в магнітному полі енергія під час розмикання кола призводить до утворення різкого стрибка значення напруги у колі, що може призвести до виходу з ладу ввімкнених у коло пристроїв.

Щоб уникнути цього, слід паралельно до елемента зі значною індуктивністю ввімкнути діод у зворотному напрямку.

Тяговий електромагніт. Тяговий електромагніт складається з обмотки, магнітопроводу і рухомого штока, який є частиною магнітопроводу. Якщо по обмотці пропустити електричний струм, то на шток діятиме сила, спрямована так, щоб зазор зменшувався. Тяговий електромагніт — це виконавчий пристрій для перетворення електричного сигналу в механічне переміщення штока.

Клапани. У багатьох технологічних процесах, наприклад у хімічній промисловості, потрібно керувати подачею рідин і газів. Для цього на трубопроводах подачі встановлені клапани, які діють таким чином: якщо встановити клапан у закрите положення, то подача рідини або газу припиняється, а якщо у відкрите — то відновлюється. Для переміщення клапана застосовуються електричний, гідравлічний або пневматичний приводи.

Електропривід

Електропривід — це сукупність виконавчого електричного двигуна і електронної системи керування цим двигуном. Електропривід призначений для приведення в рух робочих органів верстатів, маніпуляторів роботів і робототехнічних комплексів. В електроприводі електрична енергія перетворюється у механічну енергію.

Таким чином, з одного боку виконавчий двигун електроприводу є об'єктом керування з боку електронної системи, а з іншого боку — електропривід у цілому є виконавчим пристроєм у системі автоматичного керування більш високого ступеня ієрархії, наприклад, верстата з числовим програмним керуванням.

Як виконавчі двигуни в електроприводах використовуються двигуни постійного і змінного струму.

Двигуни постійного струму. Для електроприводу використовуються найчастіше двигуни постійного струму з незалежним збудженням. Частота обертання ротора двигуна постійного струму визначається формулою:

$$n = \frac{U}{C_E \Phi} - \frac{R_я}{C_E C_M \Phi^2} M, \qquad (13.8)$$

де U — напруга живлення двигуна;
C_E, C_M — сталі, що залежать від конструкції двигуна;
Ф — магнітний потік;
$R_я$ — опір обмотки якоря двигуна;
M — момент на валу двигуна.

Змінювати швидкість обертання двигуна можна, як видно з формули (13.8), такими способами:

- ❒ змінюючи напругу живлення (якірне керування);
- ❒ змінюючи магнітний потік (полюсне керування);
- ❒ змінюючи опір у колі якоря двигуна.

Якірне керування має такі позитивні якості: висока лінійність і однозначність механічних і регулювальних характеристик; значна крутизна механічних характеристик; незначні втрати в обмотці якоря; незначна індуктивність кола керування (якоря), відповідно велика швидкодія; значний діапазон регулювання швидкості.

Недоліком керування швидкості обертання за допомогою зміни напруги є необхідність мати кероване джерело постійної напруги. Найпоширенішим способом створення такого джерела живлення є застосування широтно-імпульсної модуляції (ШІМ). Суть її полягає в тому, що для живлення двигуна використовується не постійна напруга, а імпульсна. Амплітуда і частота імпульсів стала, а тривалість імпульсів можна змінювати. Середнє значення такої імпульсної напруги прямо пропорційне тривалості імпульсів. Внаслідок значної інерційності ротора, двигун реагує тільки на середнє значення напруги. Таким чином, змінюючи тривалість імпульсів, можна змінювати середнє значення напруги живлення, а, отже, керувати швидкістю обертання двигуна постійного струму. Для формування імпульсної напруги живлення зі змінною тривалістю імпульсів використовуються прилади силової електроніки: потужні транзистори і тиристори.

Основним недоліком двигунів постійного струму є наявність у них механічного комутатора секцій обмотки якоря, так званого колекторно-щіткового механізму. По-перше, для виготовлення колектора використовуються дорогі матеріали і виготовляється колектор за складною технологією. По-друге, під час роботи між колектором і щітками виникає іскріння, що створює завади у роботі електронних схем і унеможливлює застосування двигунів постійного струму у вибухонебезпечних виробництвах, наприклад, у хімічній і нафтогазовій промисловостях.

Асинхронні двигуни змінного струму. Перевагою асинхронних двигунів, порівняно з двигунами постійного струму, є проста технологія виготовлення, простота обслуговування, висока надійність, відсутність електричного зв'язку між статором і ротором, відсутність колекторно-щіткового механізму, отже і відсутність іскріння.

Для електроприводу можуть використовуватися двофазні і трифазні асинхронні двигуни.

Частота обертання ротора асинхронного двигуна визначається формулою:

$$n = \frac{60f}{p}(1 - s), \qquad (13.9)$$

де f — частота коливань змінної напруги в мережі живлення;

p — кількість пар полюсів;

$s = \frac{n_1 - n_2}{n_1}$ — ковзання ротора, яке залежить від частоти обертання магнітного поля n_1 і частоти обертання ротора n_2.

Частоту обертання ротора асинхронного двигуна можна змінювати, змінюючи згідно з формулою (13.9), частоту коливань напруги f джерела живлення, кількість пар полюсів p, ковзання s, яке, у свою чергу, залежить від моменту навантаження двигуна, амплітуди напруги живлення, активних і реактивних опорів обмоток статора і ротора.

Для приводів середньої і великої потужностей застосовується здебільшого частотне керування швидкістю обертання. Енергозабезпечення здійснюється трифазними системами з незмінною частотою, тому для здійснення частотного керування асинхронних двигунів необхідно живити двигун від спеціального трифазного джерела змінної напруги, амплітудою і частотою якого можна керувати. Такі спеціальні керовані джерела живлення стало можливим створювати із розробкою і масовим випуском силових напівпровідникових приладів: транзисторів і тиристорів великої потужності.

Керування такими потужними напівпровідниковими приладами здійснюється за складними алгоритмами. Електронною промисловістю випускаються спеціально розроблені мікросхеми для керування силовою електронікою електроприводів.

Як виконавчі двигуни електроприводів малої потужності широко використовуються асинхронні двофазні двигуни з порожнистим і короткозамкнутим ротором. Для таких двигунів застосовують амплітудний, фазний і амплітудно-фазний спосіб керування частотою обертання ротора.

Для здійснення ***амплітудного керування*** на обмотку збудження і обмотку керування подаються змінні напруги однієї частоти і зсунуті одна відносно одної на фазовий кут $\pi/2$. Амплітуда напруги на обмотці керування змінюється від нуля до номінального значення.

Фазове керування кутовою швидкістю обертання здійснюється подачею на обмотки збудження і керування змінних напруг

однакової частоти й амплітуди і зміною фазового зсуву напруги керування відносно напруги збудження у діапазоні від $-\pi/2$ до $+\pi/2$.

Амплітудно-фазове керування здійснюється, якщо одночасно застосувати амплітудне і фазове керування.

Синхронні двигуни змінного струму. На відміну від асинхронних, у синхронних двигунах частота обертання ротора не залежить від навантаження, а визначається тільки частотою коливань напруги джерела живлення, тому синхронні двигуни доцільно застосовувати у приводах, які мають забезпечувати високу стабільність частоти обертання у широкому діапазоні зміни навантаження.

Крановий електродвигун. У кроковому електродвигуні ротор може знаходитися в одному з можливих стійких положень. Стійкі положення розміщено рівномірно по колу. Кількість стійких положень ротора визначається конструкцією крокового електродвигуна. Щоб перемістити ротор з одного стійкого положення у сусіднє, в обмотку слід подати імпульс струму. Таким чином, обертання ротора можна розглядати як послідовний перехід рядом стійких положень. До позитивних властивостей крокових електродвигунів належить висока точність позиціонування, узгодженість з імпульсними і цифровими пристроями керування. Водночас у крокових електродвигунів менший порівняно з двигунами неперервного типу рушійний момент, обмежена швидкість обертання, високий рівень вібрацій через стрибкоподібний рух.

Гідро- і пневмопривід

У гідро- і пневмоприводах енергія стисненого газу (здебільшого повітря) або рідини перетворюється в механічну енергію. Позитивними властивостями гідро- і пневмоприводів є великі зусилля, які вони розвивають, і здатність забезпечити прямолінійний рух. За будовою гідро- і пневмоприводи поділяються на мембранні і поршневі. Якщо у камеру під мембраною (рис. 13.17) подати робочу речовину під тиском, то робочий шток, долаючи опір пружини і самої мембрани, переміщуватиметься у крайнє положення. Аналогічним чином діє і поршневий привід. Якщо робочою речовиною заповнюють камери з обох боків від поршня, то поршень рухатиметься в той чи протилежний бік, залежно від співвідношення тисків у камерах.

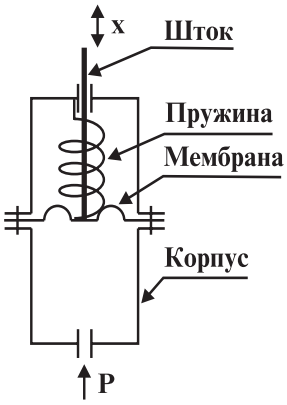


Рис. 13.17

На практиці знайшли застосування двокаскадні поршневі приводи: один поршень (керівний) керує перерозподілом робочої речовини у камерах другого (робочого) поршня. Перерозподіл робочої речовини в першому поршні (керівному) здійснюється за допомогою електромагнітного перетворювача.

13.4 Обробка сигналів у системах автоматичного керування

У системах автоматичного керування існує два потоки інформації: перший потік — це інформація про об'єкт керування, яка сприймається датчиками через канали, і засоби обробки і поступає у систему автоматичного керування; другий потік — це керівна інформація, яку видає комп'ютер системи автоматичного керування на виконавчі пристрої через засоби обробки інформації (рис. 13.18).

Рис. 13.18

Обидва потоки інформації проходять цілим каскадом різноманітних перетворень. Розглянемо найважливіші перетворення цих потоків інформації.

Узгодження рівнів сигналів

Вихідні сигнали датчиків, як правило, не можна передавати безпосередньо в комп'ютер, їх спочатку потрібно узгодити за рівнем. Багато датчиків (наприклад, термоелементи, тензорезистори тощо) мають рівень вихідного сигналу недостатній для передачі каналом зв'язку в комп'ютер, і тому їх потрібно підсилювати. Для підсилення сигналів у сучасних системах автоматичного керування застосовуються операційні підсилювачі, охоплені різними видами зворотних зв'язків. Сучасна електронна промисловість випускає широку гаму типів операційних підсилювачів, як загального призначення, так і спеціалізованих, наприклад, підвищеної

швидкодії, з високим вхідним опором, високої точності (інструментальні підсилювачі), з високою вхідною напругою тощо.

У системах автоматичного керування для підсилення сигналу широко застосовуються диференціальні підсилювачі в режимі інвертора (рис. 13.19, *а*), повторювача (рис. 13.19, *б*), диференціального підсилювача (рис. 13.19, *в*).

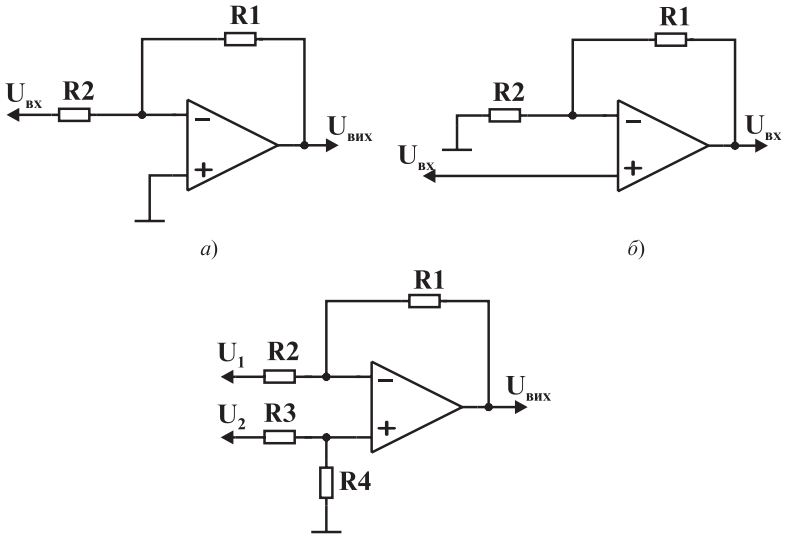


Рис. 13.19

Узгодження імпедансів

Крім узгодження сигналів за рівнем, необхідно ***узгоджувати імпеданси*** (тобто повні опори) послідовно з'єднаних пристроїв обробки інформації. Пристрій з низьким вхідним опором істотно впливає на процеси, що протікають в об'єкті, до якого він приєднаний. Наприклад, вольтметр з низьким вхідним опором, увімкнений паралельно до пристрою, напругу на якому слід виміряти, спотворює режим досліджуваного кола, оскільки зменшує еквівалентний опір цього кола, в результаті чого напруга на пристрої, до якого приєднаний вольтметр, відрізнятиметься від напруги на пристрої без вольтметра. Достатньо масивний датчик температури змінює тепловий режим досліджуваного процесу, в результаті чого виникає похибка вимірювання температури.

Пристрій з високим вихідним імпедансом дуже чутливий до опору навантаження, оскільки навантаження з малим опором спричинює значні вихідні струми, що може призвести до перевантаження пристрою.

Таким чином, для зменшення спотворення сигналів необхідно, щоб вхідний опір наступного пристрою у тракті обробки сигналу був би набагато більшим за вихідний опір попереднього пристрою, тобто чим більше відношення вхідного до вихідного опору двох послідовно з'єднаних пристроїв, тим більший ступінь узгодженості імпедансів. Щоб узгодити вихідний імпеданс попереднього пристрою і вхідний імпеданс наступного пристрою, між цими пристроями включають операційний підсилювач у режимі повторювача напруги, який має великий вхідний опір (десятки і сотні мегом) і малий (десяті і соті частки ома).

Для сигналів високих частот та імпульсних сигналів важливим є також узгодження вихідного імпедансу пристрою з імпедансом (хвильовим опором) лінії передач, а також узгодження імпедансу лінії передач з вхідним опором приймача інформації. Якщо такого узгодження не дотримуватися, то в лінії передач спостерігатиметься відбивання електромагнітних хвиль, що призведе до спотворення сигналів, що передаються.

Зменшення впливу електромагнітних завад

На роботу систем автоматичного керування негативно впливають завади (див. розділ 1, п. 1.1)

Канали обробки і передачі інформації через взаємні резистивні, індуктивні і ємнісні зв'язки впливають один на одного, чим створюють ***взаємні завади***. Частину сигналу одного каналу, яка через ці зв'язки проникає у розміщений поблизу сусідній канал, часто називають ***наводкою***. Особливо велику за інтенсивністю наводку створюють електричні кола живлення на високочутливі вхідні пристрої систем автоматичного керування.

Зовнішні завади поділяються на промислові, атмосферні і космічного походження.

Промислові завади створюються в результаті дії електромагнітних полів різних електротехнічних пристроїв: ліній електропередач, трансформаторних підстанцій, електроустаткування промислових підприємств, контактних мереж електротранспорту.

До ***атмосферних*** відносяться ***завади***, спричинені різними атмосферними явищами: грозовими розрядами, магнітними бурями, північним сяйвом.

Космічні завади спричинені електромагнітним випромінюванням Сонця, видимих і невидимих зірок та інших космічних об'єктів.

За характером дії на вхід вимірювального пристрою завади поділяються на ***синфазні*** або ***поздовжні завади*** і ***диференціальні*** або ***поперечні завади***. Синфазні завади називаються також

завадами загального виду, а диференціальні — завадами нормального виду. Поперечні завади діють як сигнал між вхідними полюсами пристрою, а поздовжні діють між точкою заземлення і вхідними полюсами пристрою.

Щоб мінімізувати дію завад в умовах одночасної роботи багатьох електромагнітних пристроїв системи, необхідно провести комплекс заходів, спрямованих для досягнення ***електромагнітної сумісності***. Для цього потрібно, щоб кожний пристрій системи мав низький рівень чутливості до зовнішніх завад, а з іншого боку, сам не генерував завади, які могли б негативно впливати на роботу інших пристроїв.

Якщо кілька електричних пристроїв мають спільне джерело живлення і спільне заземлення, то вони можуть створювати один одному завади через резистивний зв'язок. Один з можливих способів уникнути негативних наслідків — забезпечити для високочутливих пристроїв окреме джерело живлення. Друга можливість — гальванічна розв'язка між апаратурою і джерелом живлення.

Якщо електричні кола двох електронних пристроїв знаходяться під різними електричними потенціалами, то через ***взаємний ємнісний зв'язок*** один пристрій створюватиме завади іншому пристрою. Для зменшення впливу одного електричного пристрою на інший через взаємний ємнісний зв'язок необхідно здійснити електростатичне екранування найчутливіших пристроїв.

Контури зі струмом одного електричного пристрою можуть через ***взаємний індуктивний зв'язок*** згідно із законом електромагнітної індукції наводити електрорушійну силу в контурах інших пристроїв, створюючи тим самим завади.

Щоб зменшити вплив одного пристрою на інший через взаємний індуктивний зв'язок застосовують такі прийоми. Для зв'язку застосовують не паралельні, а скручені провідники. Сигнальні провідники і провідники електроживлення слід прокладати якомога далі один від одного. Чутливі пристрої слід розміщувати якнайдалі від потужних джерел магнітного поля (силових трансформаторів, індукторів тощо). Для потужних джерел магнітного поля слід застосовувати магнітні екрани, виготовлені з феромагнітного матеріалу з високою магнітною проникністю.

Аналогова фільтрація

У багатьох випадках смуги частот сигналу і смуги частот завад відомі заздалегідь. Наприклад, частоти вихідних сигналів датчиків температури, тензодатчиків, датчиків вологості лежать у діапазоні від нуля до кількох герц (рідше до десятків герц). Наводка від мережі живлення має, як відомо, частоту коливання 50 Гц. Смуга частот шумів простягається від часток герца до мегагерців.

За допомогою аналогової фільтрації корисні сигнали підсилюються у вибраній смузі частот, а завади, що лежать поза цією смугою, пригнічуються. Таким чином, за допомогою аналогової фільтрації можна збільшити відношення сигнал/шум.

Фільтрація електричних сигналів здійснюється за допомогою аналогових фільтрів. У сучасних системах автоматичного керування, як і в інших галузях, застосовуються активні аналогові фільтри на основі операційних підсилювачів.

Повну інформацію про властивості фільтра несе ***комплексна частотна характеристика***, а також ***амплітудно-частотна*** і ***фазочастотна характеристики***. Крім повних характеристик широко застосовуються часткові характеристики. Однією з таких характеристик є ***смуга пропускання***, тобто смуга частот, на якій коефіцієнт підсилення фільтра зменшується не більше ніж у 0,707 порівняно з коефіцієнтом підсилення на опорній частоті. Частота, на якій коефіцієнт підсилення фільтра зменшується у $\sqrt{2}$ рази, називається ***частотою зрізу*** фільтра.

За розміщенням смуги пропускання на осі частот фільтри поділяються на фільтри нижніх частот, верхніх частот, смугові (рис. 13.20).

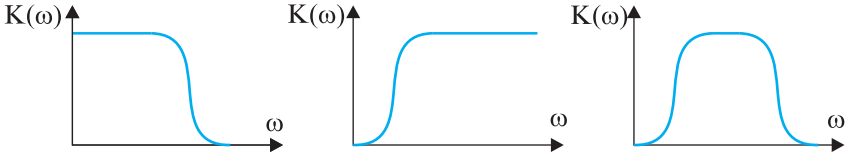
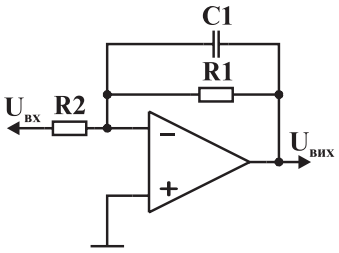


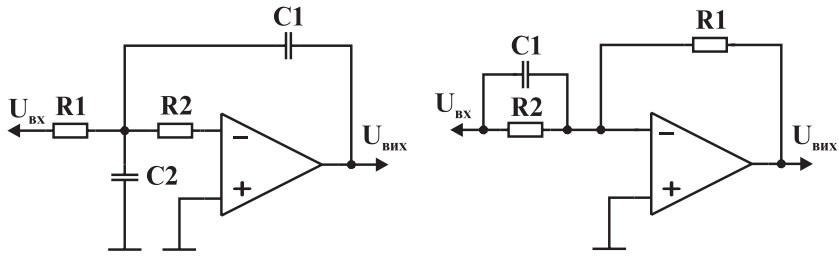
Рис. 13.20

Якість фільтра, відповідно і складність, визначає порядок фільтра. Застосовуються фільтри першого, другого і так далі порядків. На рисунку 13.21, як приклад, наведено фільтри нижніх частот першого (*а*) і другого (*б*) порядків і фільтр верхніх частот першого порядку (*в*).



а)

Рис. 13.21



б) *в*)

Рис. 13.21

Мультиплексування сигналів

Вихідні сигнали багатьох типів датчиків систем автоматичного керування змінюються дуже повільно внаслідок інерційності процесів в об'єктах керування. Швидкодія ж сучасних цифрових пристроїв обробки сигналів дуже велика. Для прикладу можна зазначити, що смуга частот термоперетворювачів становить одиниці герца, у той час як тактова частота цифрових пристроїв обробки інформації (процесорів, контролерів) становить сотні мегагерців. Пропускна здатність сучасних каналів зв'язку також є значною. Таким чином, цифрові пристрої обробки інформації мають таку продуктивність, а канали зв'язку таку пропускну здатність, що можуть одночасно обробляти сотні і тисячі вихідних сигналів датчиків паралельно. Щоб це здійснити, застосовуються ***мультиплексори***.

Мультиплексор — пристрій, який послідовно, за чергою підключає вихідні сигнали датчиків на вхід каналу обробки інформації. Таким чином, мультиплексор — перемикач або комутатор, який має кілька входів і один вихід (рис. 13.22). Мультиплексор може реалізовуватися на електромеханічних перемикачах (електромеханічних реле) або на електронних ключах. Електромеханічні реле мають такі позитивні якості, як незначний опір у замкненому стані та значний у розімкнутому. Сигнал керування електромеханічним ключем гальванічно розв'язаний від сигналу, що перемикається. Разом з тим, швидкодія електромеханічних реле низька (сотні перемикань за секунду). Строк служби обмежений природним зношенням рухомих частин. Швидкодія електронних ключів набагато більша (час комутації становить одиниці і частки мікросекунди) ніж в електромеханічних, але сигнал керування

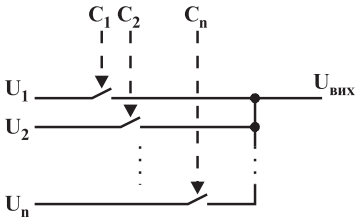


Рис. 13.22

гальванічно не розв'язаний з сигналом, що перемикається. Електронні ключі мають значно більший опір у замкненому стані порівняно з електромеханічними. Слід зазначити, що технологія виготовлення напівпровідникових приладів і електронних ключів зокрема, постійно вдосконалюється і параметри електронних ключів неухильно поліпшуються.

Дискретизація аналогових сигналів

Вихідні сигнали датчиків є неперервними, тобто існують у будь-який момент часу. Цифрові ж пристрої обробки сигналів обробляють окремі значення сигналів, що наведені у вигляді чисел. Таким чином, на пристрій обробки має надходити не аналоговий сигнал у кожний момент часу, а послідовність окремих значень сигналу через певні інтервали часу. Процес періодичної вибірки окремих значень із аналогового сигналу і представлення неперервного аналогового сигналу послідовністю дискретних значень називається ***дискретизацією*** (*sampling*) аналогового сигналу (рис. 13.23). Кожне вибране окреме значення сигналу називається ***дискретою***. Інтервал часу, через який вибираються окремі значення сигналу, називається ***інтервалом дискретизації*** T_S, а обернена величина — ***частотою дискретизації*** $f_S = 1/T_S$.

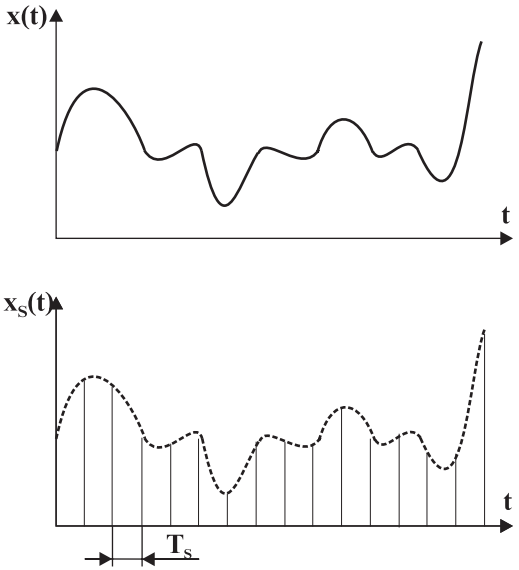


Рис. 13.23

Визначення частоти дискретизації є дуже важливою технічною задачею. З одного боку, зі збільшенням частоти дискретизації

різниця між значенням сигналу на початку і в кінці інтервалу дискретизації зменшуватиметься, оскільки зі зменшенням інтервалу дискретизації сигнал істотно не встигає змінитися і послідовність дискретних значень точніше відтворюватиме аналоговий сигнал. З іншого боку, якщо частота дискретизації занадто велика, то збільшуватиметься кількість однакових сусідніх дискрет сигналу, які не несуть нової інформації і тільки завантажують пристрої обробки.

З дискретизацією сигналів пов'язане специфічне спотворення сигналів (у англомовній літературі воно називається *aliasing*), яке проявляється в тому, що у дискретизованому сигналі при недостатній частоті дискретизації з'являються складові сигналу (псевдосигнали) з частотами, яких немає в аналоговому сигналі.

Щоб уникнути появи псевдосигналів, необхідно частоту дискретизації вибирати згідно з теоремою дискретизації (у вітчизняній літературі вона називається теоремою Котельникова, а в англомовній — теоремою Найквіста). За цією теоремою, щоб уникнути спотворення сигналів, частота дискретизації f_S має бути вдвічі більшою ніж максимальна частота $f_{\max}$ у спектрі сигналу, тобто $f_S = 2f_{\max}$. Така частота дискретизації називається ***частотою Найквіста***. Теорема дискретизації стосується ідеалізованого сигналу з обмеженим спектром. Реальні сигнали мають спектр у необмеженій смузі частот, тому щоб звести спотворення сигналу до заданого рівня, необхідно частоту дискретизації вибирати в кілька разів більшу за частоту Найквіста.

Ще одним способом зменшення спотворення сигналів під час дискретизації є застосування аналогової фільтрації. За допомогою аналогових фільтрів слід відфільтрувати сторонні сигнали, частоти яких більші за половину частоти дискретизації.

Дискретизацію здійснюють за допомогою спеціального пристрою — пристрою вибірки і зберігання (*sample-and-hold circuit*). Найпростіша схема цього пристрою, що ілюструє принцип його дії, наведена на рисунку 13.24.

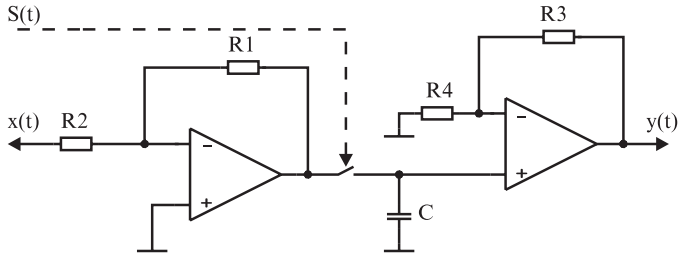


Рис. 13.24

Пристрій має два режими роботи, які задаються станом ключа *SW*. Якщо ключ ***замкнутий***, то пристрій працює у ***режимі вибірки*** і сигнал на виході пристрою практично повторює сигнал на його вході. У ***режимі зберігання*** ключ ***розімкнутий*** і сигнал на виході пристрою зберігає значення вхідного сигналу безпосередньо перед моментом розмикання.

Аналого-цифрове перетворення

Вимірювана фізична величина змінюється в межах деякого діапазону значень. Цей діапазон ділиться на багато ступенів або квантів. Наприклад, якщо напруга змінюється в діапазоні від нуля до 10 В, то діапазон можна поділити на 10 000 квантів або ступенів по 0,001 В. Далі кожне миттєве значення фізичної величини порівнюється (врівноважується) з сукупністю квантів (квантовою фізичною величиною, відтвореною мірою) до виконання системи нерівностей

$$N \cdot \Delta x \le x \le (N+1) \cdot \Delta x. \qquad (13.10)$$

Компаратор або пристрій для порівняння фіксує, тобто «сигналізує» про виконання системи нерівностей спеціальним сигналом. Наприклад, якщо система нерівностей виконується, то сигнал на виході компаратора 1, а якщо не виконується, то 0.

Після закінчення процесу зрівноважування вимірювальній величині X приписується або нижній рівень $N \cdot \Delta x$ (квантування «з недостачею»), або верхній рівень $(N + 1) \cdot \Delta x$ (квантування «з надлишком»).

Число N (або (N+1)) потрібно подати у певній ***системі числення***. Цей процес називається ***кодуванням*** квантового сигналу. Комп'ютери та цифрові прилади здійснюють обробку чисел у ***двійковій*** системі числення, в якій довільне число подається комбінацією тільки двох цифр 0 та 1.

Для візуального відображення інформації користуються звичною для людини ***десятковою*** системою числення, в якій кожне число подається комбінацією десяти цифр 0, 1, …, 9, які часто називають арабськими цифрами.

Вимірювальний перетворювач, який здійснює дискретизацію аналогового сигналу, квантування та кодування сигналу, називається ***аналого-цифровим перетворювачем*** (АЦП). Аналого-цифровий перетворювач є невід'ємною складовою частиною будь-якого цифрового приладу. Мікроелектронна промисловість випускає аналого-цифрові перетворювачі у вигляді однієї або кількох мікросхем.

Методи аналого-цифрового перетворення. Методи аналого-цифрового перетворення поділяються на:

- ❒ методи зіставлення;
- ❒ методи зрівноважування.

У методах зіставлення аналого-цифрове перетворення здійснюється за один прийом або такт, тобто з максимальною швидкістю. Щоб домогтися цього, потрібно за допомогою міри відтворити всі N значення, на які квантується діапазон, тобто міра має бути багатоканальною.

У методах зрівноважування аналого-цифрове перетворення здійснюється за кілька тактів. Методи зрівноважування поділяються на методи слідкуючого і розгортального зрівноважування.

АЦП зіставлення (паралельної дії). На рисунку 13.25, як приклад, зображено аналого-цифровий перетворювач напруги, який реалізує метод зіставлення.

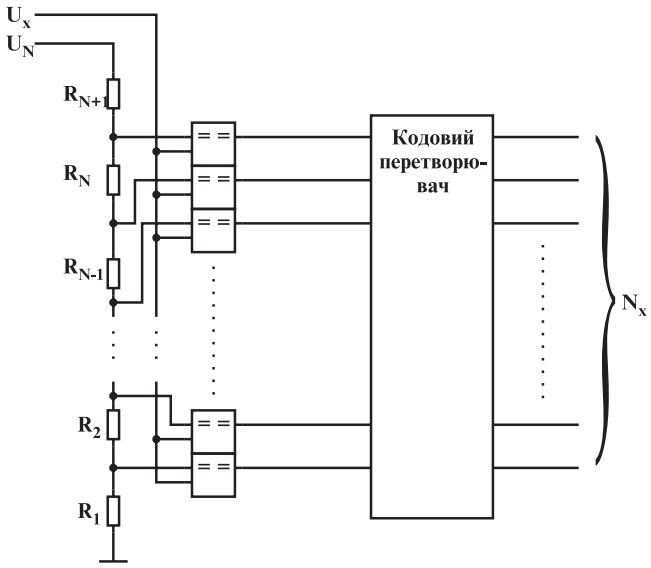


Рис. 13.25

Напруга з виходу стабілізованого джерела, подається на багатоступеневий дільник напруги. Кількість ступенів дільника напруги дорівнює кількості квантів діапазону вимірювання. На виході дільника напруги відтворюється N квантованих рівнів напруги

$$U_N \cdot \frac{1}{N},\ U_N \cdot \frac{2}{N},\ U_N \cdot \frac{3}{N},\ \ldots,\ U_N \cdot \frac{N-1}{N},\ U_N.$$

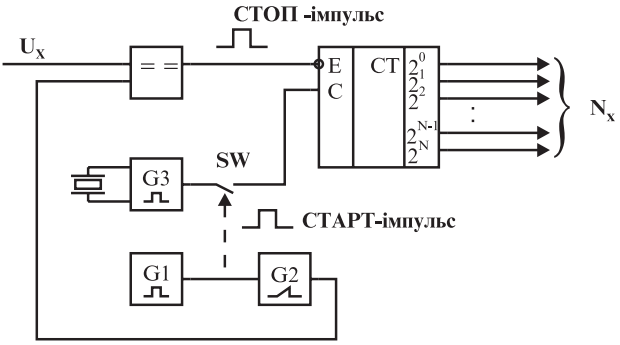
Ці рівні напруги подаються на перші входи N компараторів, а на їхні другі входи подається вимірювальна напруга U_x. Вихідний сигнал кожного n-го компаратора дорівнює 1, якщо $U_X \geq U_N \cdot \frac{n}{N}$; і дорівнює 0, якщо $U_X < U_N \cdot \frac{n}{N}$. Виходи всіх N компараторів подаються на пристрій кодування, який формує число N_X у певній, як правило, двійковій системі числення. Число N_X — це кількість компараторів, вихідний сигнал яких дорівнює 1, тобто це кількість квантів $N_X \cdot \Delta U$, які не перевищують вимірюваної напруги U_X,

$$U_X \geq N_X \cdot \Delta U.$$

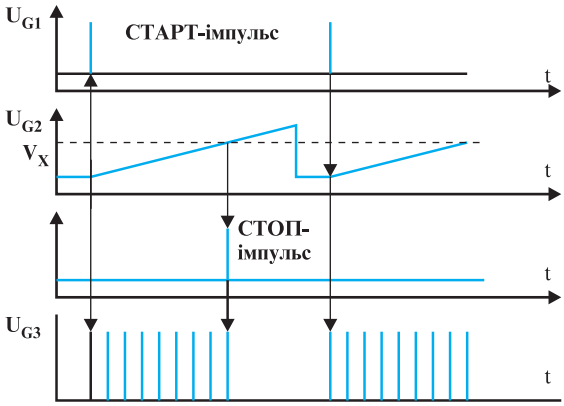
Аналого-цифрові перетворювачі, які реалізують метод зіставлення, випускаються серійно на кількість квантів 256, 512 і здійснюють до половини мільярда вимірювань за секунду.

АЦП розгортального зрівноважування. АЦП розгортального зрівноважування працюють циклами, які періодично повторюються. Нове значення вимірюваної величини періодично замінює попереднє значення. У кожному циклі вимірювана величина компенсується однорідною фізичною величиною, яка відтворюється мірою лінійно або згідно з певним законом. На рисунку 13.26 зображено структуру АЦП з лінійним наростанням зрівноважувальної напруги у кожному циклі. Таким чином, вимірювана напруга перетворюється в інтервал часу, який потім вимірюється, тому такі АЦП називаються часово-імпульсними.

Часово-імпульсний АЦП працює таким чином. Генератор стартових імпульсів G_1 генерує стартовий імпульс на початку кожного циклу. Стартовий імпульс запускає генератор лінійно-наростаючої напруги G_2 і одночасно під'єднує за допомогою комутатора генератор квантувальних імпульсів G_3 від входу лічильника імпульсів. Напруга на виході генератора лінійно-наростаючої напруги зростає і порівнюється з вимірюваною напругою. Коли досягається момент рівності напруг, компаратор генерує стоп-імпульс, який від'єднує генератор квантувальних імпульсів від входу лічильника імпульсів, і надходження імпульсів на лічильник припиняється. Число на виході лічильника імпульсів є результатом вимірювання у двійковій системі числення. За допомогою перетворювача кодів воно перетворюється у десяткову систему числення і відображується на цифровому індикаторі.



а)

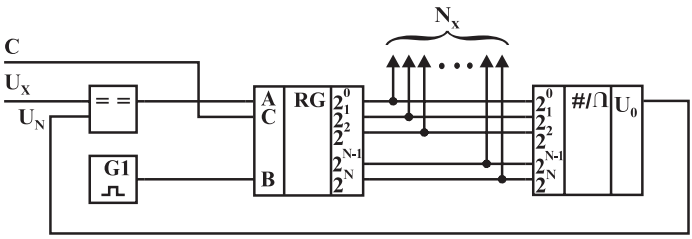


б)

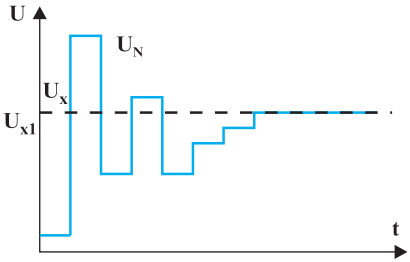
Рис. 13.26

Недоліком цифрових приладів розгортального зрівноважування з рівномірно ступінчастою зміною зрівноважувальної величини є велика тривалість циклу, яка збільшується з підвищенням точності, тобто збільшенні числа квантів, на які квантується інтервал вимірюваної величини.

Для зменшення тривалості аналого-цифрового перетворення, тобто для підвищення швидкодії, застосовується ***порозрядне зрівноважування*** вимірюваної величини. Здебільшого порозрядне зрівноважування здійснюється у двійковій системі і весь цикл зрівноважування ділиться на такти відповідно до кількості розрядів. На рисунку 13.27 наведена структура АЦП з порозрядним зрівноважуванням, який працює таким чином: стартовий імпульс, який на початку кожного циклу видає генератор стартових імпульсів, запускає процес зрівноважування, який здійснюється порозрядно, починаючи від найстаршого розряду і закінчуючи наймолодшим.



а)



б)

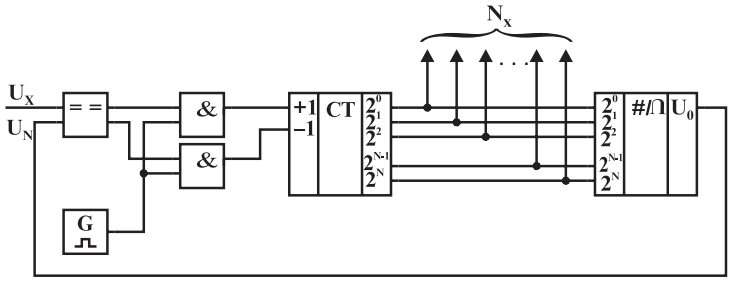
Рис. 13.27

Спочатку пристрій керування вмикає найстарший розряд, цифро-аналоговий перетворювач відтворює зрівноважувальну напругу відповідно до найстаршого розряду, і ця напруга порівнюється за допомогою компаратора з вимірюваною напругою. Якщо вимірювана напруга більша за зрівноважувальну, то найстарший розряд залишається, а якщо менша, то вимикається. У наступному такті вмикається сусідній молодший розряд і зрівноважування здійснюється в такій самій послідовності. Таким чином, починаючи з найстаршого послідовно, такт за тактом, переходячи від старшого до молодшого і закінчуючи наймолодшим, здійснюється аналого-цифрове перетворення вимірюваної напруги. Кількість тактів порозрядного зрівноважування значно зменшується у порівнянні з рівномірно-ступінчастим. Наприклад, якщо діапазон вимірювання квантується на 128 квантів, то максимальна кількість тактів рівномірно-ступінчастого зрівноважування дорівнює кількості квантів, тобто 128, в той час як кількість тактів порозрядного зрівноважування дорівнює

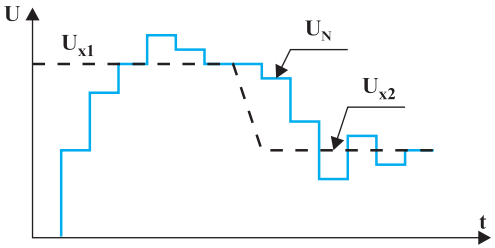
$$\log_2(128 - 1) = 7$$

АЦП слідкувального зрівноважування. Вихідний кодовий сигнал приладів слідувального зрівноважування змінюється вслід

за зміною вимірюваного сигналу. На рисунку 13.28 наведена структура АЦП слідкувального зрівноважування. Прилад працює таким чином: вимірювана напруга U_X порівнюється на компараторі з напругою $N_X \cdot \Delta U$, відтвореною цифро-аналоговим перетворювачем (ЦАП), який є мірою напруги, керованою кодом. Якщо $U_X \geq N_X \cdot \Delta U$, то вихідний сигнал компаратора 1, і імпульси від генератора імпульсів через комутатор подаються на вхід додавання (+1) реверсивного лічильника імпульсів. Число N_X на виході лічильника імпульсів зростає і зростатиме доти, поки U_X не зрівноважиться $N_X \cdot \Delta U$. Якщо ж $U_X \leq N_X \cdot \Delta U$, то вихідний сигнал компаратора дорівнює 0, імпульси від генератора імпульсів передаються тепер на вхід віднімання (−1) реверсивного лічильника імпульсів, і число N_X на виході лічильника зменшується і зменшуватиметься доти, поки U_X не зрівноважиться $N_X \cdot \Delta U$.



а)



а)

Рис. 13.28

Вихідний код реверсивного лічильника імпульсів формується, як правило, у двійковій системі числення і керує цифро-аналоговим перетворювачем. Для візуального спостереження вихідний код лічильника імпульсів за допомогою перетворювача кодів

перетворюється у звичний для нас десятковий код, який відображується на цифровому індикаторі.

Цифро-аналогове перетворення

У цифрових системах автоматичного керування керівний комп'ютер оперує з сигналами у вигляді двійкових чисел. Щоб передати ці сигнали для керування виконавчими органами, їх потрібно перетворити в аналогову форму. В аналогову форму потрібно також перетворити опорні сигнали для аналогових регуляторів. Пристрій, який перетворює цифрові сигнали в аналогові, називається ***цифро-аналоговим перетворювачем*** (ЦАП).

Цифро-аналогові перетворювачі складаються з опорного джерела напруги з високою стабільністю, матриці резисторів і набору електронних ключів. Використовуються два види матриці резисторів. У матриці першого виду опір наступного резистора вдвічі більший за опір попереднього (рис. 13.29). Таким чином, опори резисторів матриці відповідають розрядам двійкового числа. Якщо резистори матриці з'єднати паралельно й увімкнути до опорного джерела стабільної напруги, то через резистори матриці протікатимуть струми, значення яких відповідатимуть розрядам двійкового числа.

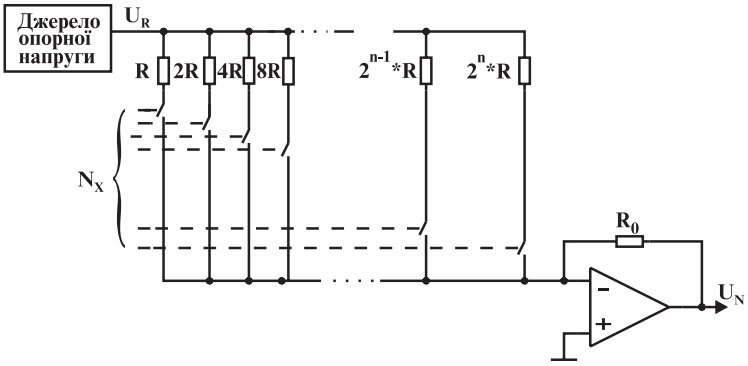


Рис. 13.29

Крім матриці резисторів, опори яких відповідають розрядам двійкового числа, застосовуються також матриці резисторів R – 2R, що мають регулярну структуру і містять резистори тільки двох номіналів, один з яких вдвічі більший другого (рис. 13.30). Якщо матрицю резисторів $R - 2R$ підключити до джерела опорної напруги, то у резисторах $2R$ протікатимуть струми, що відповідають розрядам двійкового числа.

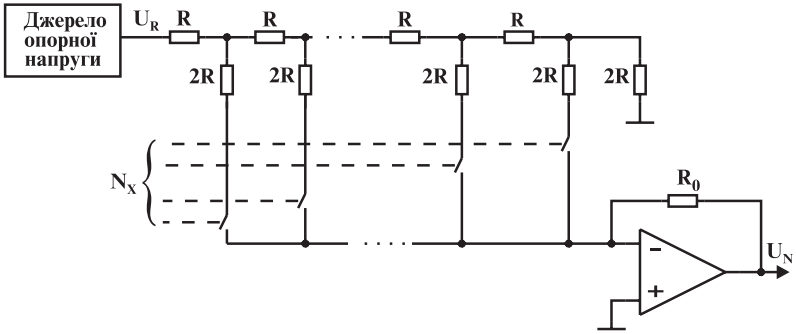


Рис. 13.30

Двійкові струми можна вмикати і вимикати за допомогою ключів. Якщо на цифро-аналоговий перетворювач подати двійкове число, кожний розряд якого керує ключем, то сумарний струм усіх увімкнених резисторів буде прямо пропорційний цьому двійковому числу.

Електронна промисловість випускає цифро-аналогові перетворювачі, виготовлені у вигляді мікросхем.

Цифрова обробка сигналів

Розглянуті вище способи обробки інформації здійснювалися до введення її у комп'ютер. Після введення інформації в комп'ютер здійснюється її обробка цифровими методами із застосуванням програмного забезпечення. Розглянемо основні способи обробки інформації, яка поступає від датчиків у комп'ютер.

Компенсація дрейфу. Дрейфом називається хаотичний, непередбачуваний вихідний сигнал аналогових пристроїв (датчиків, операційних підсилювачів тощо) за умови, що вхідний сигнал цього пристрою дорівнює нулю. Дрейф накладається на корисний сигнал і спотворює його. Щоб компенсувати дрейф, його періодично запам'ятовують, виключаючи на цей час вхідний сигнал, і віднімають від кожного значення сигналу.

Перевірка достовірності сигналу. Щоб переконатися у достовірності сигналу, слід перевірити, чи значення сигналу не виходять за межі робочого діапазону датчика. Вихід значень сигналу за межі робочого діапазону свідчить про аварійну ситуацію. Крім того, потрібно перевірити, чи швидкість зміни сигналу не виходить за межі діапазону швидкостей.

Статистична обробка сигналу. Сигнали, що поступили в комп'ютер, обробляються: обчислюється середнє значення, середнє

квадратичне відхилення. Значення сигналів, що різко відхиляються від середніх значень, слід відкинути.

Цифрова фільтрація. Крім аналогової фільтрації, над сигналами у цифровій формі можна здійснювати цифрову фільтрацію. Цифрові фільтри, на відміну від аналогових, реалізуються програмно. Це дає змогу просто змінювати параметри фільтра, реалізувати фільтри високих порядків. Цифрові фільтри мають значно більшу стабільність характеристик, ніж аналогові.

Лінеаризація. Датчики фізичних величин, що використовуються в системах автоматичного керування мають здебільшого нелінійну залежність між вхідним і вихідним сигналами. Цифрові сигнали, що обробляються системою автоматичного керування, мають бути лінійно залежними від відповідних вхідних сигналів. Щоб зробити цю залежність лінійною, необхідно здійснити нелінійне перетворення, обернене до нелінійного перетворення датчика. Така процедура називається лінеаризацією датчика.

1. Що таке автоматизація?
2. Які ви знаєте характеристики датчиків?
3. Схарактеризуйте датчики руху?
4. Які датчики сили, моменту, тиску застосовуються у системах автоматизованого керування?
5. Який принцип дії датчиків температури?
6. Які виконавчі пристрої застосовуються у системах автоматизованого керування?
7. Яким чином здійснюється обробка сигналів у системах автоматизованого керування?
8. Як працюють аналого-цифрові перетворювачі?
9. Як здійснюється цифро-аналогове перетворення сигналів?
10. Як обробляється цифровий сигнал у системах автоматизованого керування?

14

Системи автоматизованого проектування (CAD)

14.1 Основні поняття

Сучасне виробництво характеризується різким ускладненням виробів, що спричинює значне збільшення обсягу проектних і конструкторських робіт. На сучасних підприємствах авіакосмічної, електронної, біотехнологічної та інших високотехнологічних галузей штати конструкторських бюро складають значну частку від загальних штатів робітників. Крім того, проектно-конструкторськими роботами займаються спеціальні заклади: проектні інститути, спеціальні конструкторські бюро тощо.

Технічний прогрес і конкурентна боротьба змушують скорочувати терміни розробки нових виробів. Виграє в цій боротьбі той, хто перший почне випускати новий товар чи нову модель: комп'ютер, літак, автомобіль тощо.

Застосування комп'ютерно-інформаційних технологій у проектно-конструкторській роботі дає змогу значно збільшити продуктивність роботи конструктора, істотно скоротити терміни розробки.

У деяких галузях, наприклад в електронній промисловості, під час розробки інтегральних схем високого ступеня інтеграції, взагалі неможливо проводити проектні й конструкторські розробки без застосування комп'ютерів.

Для автоматизації проектних робіт у різних галузях виробництва розроблено й успішно експлуатуються системи автоматизованого проектування (САПР) (англомовна абревіатура CAD — Computer Aided Design).

У архітектурі для проектування різних споруд промислового і цивільного призначення застосовується система ArchiCAD.

У машинобудуванні та приладобудуванні для проектування різноманітних машин, пристроїв і виготовлення креслень та іншої технічної документації застосовується система автоматизованого проектування AutoCAD.

Найбільшого поширення системи автоматизованого проектування знайшли в електронній промисловості для проектування цифрових, аналогових та цифро-аналогових електронних пристроїв.

Бурхливий розвиток електронної, зокрема комп'ютерної техніки, неухильне зростання продуктивності мікропроцесорів обумовлені великою мірою саме широким застосуванням комп'ютерно-інформаційних технологій для автоматизації проектування.

Для автоматизації проектування електронних пристроїв розроблено й успішно експлуатується цілий ряд систем автоматизованого проектування різного рівня: MicroCAP, Electronic Workbench, OrCAD, MicroSIM, P-CAD.

Система автоматизованого проектування електронних пристроїв P-CAD є однією з найдосконаліших, широко застосовується на виробництві, має всі необхідні засоби для автоматизованого проектування, тому далі розглядатиметься саме ця система.

Сучасні електронні пристрої виробляються, в основному, у вигляді багатошарових друкованих плат, на яких змонтовані електронні компоненти. Застосовуються такі електронні компоненти:

- мікросхеми різного ступеня інтеграції;
- напівпровідникові прилади (біполярні і польові транзистори, діоди, тиристори);
- дискретні електричні елементи: резистори, конденсатори, індуктивні елементи (трансформатори, дроселі, соленоїди тощо).

Проектування і виготовлення багатошарових друкованих плат становить значну частку від загального обсягу робіт і є складним і трудомістким процесом, тому автоматизація проектування є дуже актуальною.

Система автоматизованого проектування P-CAD призначена для проектування багатошарових друкованих плат електронних пристроїв у середовищі Windows. Вона складається з таких основних модулів:

- менеджер бібліотек P-CAD Library Manager;
- графічний редактор принципових схем електронних пристроїв P-CAD Schematic;
- графічний редактор багатошарових друкованих плат P-CAD PCB;
- програми трасування друкованих провідників Quick Route, Shape-Based Router, SPECCTRA;
- програма моделювання аналогових і аналого-цифрових електронних пристроїв Protel Advanted Sim;
- програма аналізу електромагнітної сумісності P-CAD Signal Integrity;
- допоміжні програми: програма автоматизації створення графічної і текстової документації P-CAD Document Toolbox, програма доробки фотошаблонів CAMtastic.

Проектування електронних пристроїв за допомогою системи P-CAD розпочинається зі створення принципової електричної схеми, застосовуючи для цього графічний редактор P-CAD Schematic. Створення принципової електричної схеми полягає у перенесенні умовних графічних зображень електронних компонентів з бібліотеки на робочий аркуш і з'єднання виводів компонентів електричними провідниками.

Умовні графічні зображення (Symbol) електронних компонент зберігаються у базі даних системи P-CAD. У базі даних зберігається також зображення корпусів (конструкцій) електронних компонент (Pattern) і їх параметри. Бази даних у P-CAD названо бібліотеками (Library). Для роботи користувача з бібліотеками призначена спеціальна програма P-CAD Library Manager.

Створена принципова схема електронного пристрою переноситься у середовище редактора друкованих плат P-CAD PCB. Після перенесення схеми виконується розміщення корпусів компонент на друкованій платі, редагування схеми, перевірка правильності з'єднань.

Наступним важливим етапом проектування є трасування провідників, тобто розміщення на платі електричних провідників, що з'єднують виводи компонент. Трасування провідників виконуєть за допомогою спеціальних програм Quick Route, Shape-Based Router, SPECCTRA.

Роботу електричних пристроїв можна моделювати на етапі проектування за допомогою програми Protel. За допомогою цієї програми можна розрахувати режим постійного струму, перехідні процеси, провести спектральний аналіз, проаналізувати сигнали при варіації одного чи двох параметрів, дослідити дію температури і шумів на характеристики електронного пристрою.

Щоб проаналізувати передачу сигналів провідниками друкованої плати, взаємні наводки та інші паразитні ефекти ліній передач сигналів, перевірити електромагнітну сумісність електронних компонент друкованої плати, призначено програму P-CAD Signal Integrity.

Технологія виготовлення багатошарових плат друкованого монтажу передбачає виготовлення фотошаблонів рисунків друкованого монтажу. Фотошаблони виготовляються на фотоплотерах високої точності. Для керування виконавчими органами фотоплотера необхідно створити спеціальні програми керування. Такі програми можна створити на основі розробленої друкованої плати за допомогою прикладної програми CAMtastic.

14.2 Створення принципових електричних схем електронних пристроїв

Для створення принципових електричних схем електронних цифрових, аналогових і аналого-цифрових пристроїв призначено графічний редактор схем **P-CAD Schematic**. **P-CAD Schematic**, як і інші прикладні програми Windows, викликається традиційним для Windows способом.

Вікно P-CAD Schematic. Графічний редактор працює у середовищі операційної системи Windows, тому має головне вікно, типове для прикладних програм Windows (рис. 14.1).

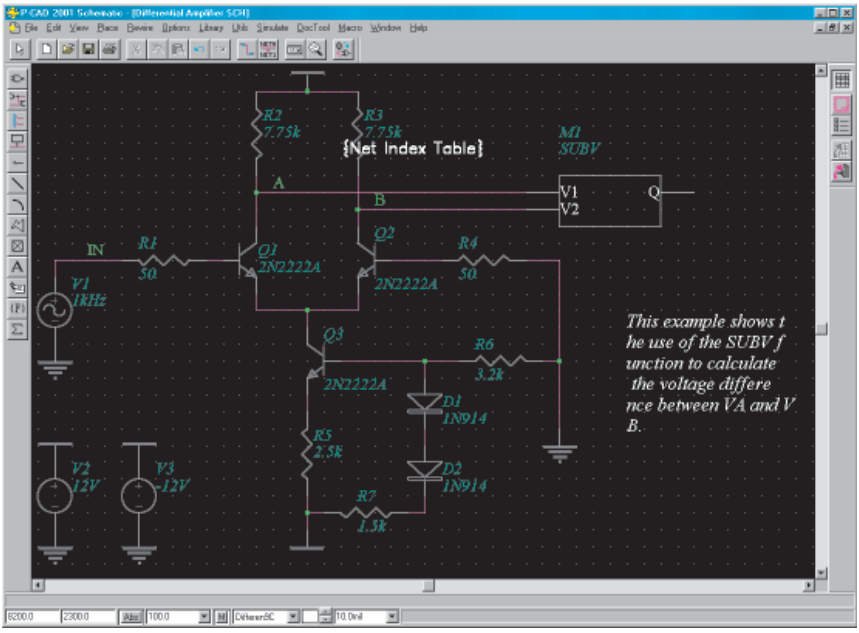


Рис. 14.1

У верхній частині вікна розміщується рядок заголовка, в якому відображується ім'я прикладної програми (**P-CAD Schematic**), ім'я відкритого активного документа, а також три стандартні кнопки керування вікном: згортання, розгортання і закриття. На початку рядка розміщена системна кнопка зі значком програми **P-CAD Schematic**. При натисканні на цю кнопку розкривається меню з командами керування вікном.

Нижче рядка заголовка розташований рядок головного меню команд: **Файл** (File), **Правка** (Edit), **Вид** (View), **Розміщення** (Place),

Зміна з'єднання (Rewire), **Настройка параметрів** (Option), **Бібліотека** (Library), **Утиліти** (Utils), **Моделювання** (Simulate), **Документування** (DocTool), **Макроси** (Macro), **Вікно** (Window), **Допомога** (Help), що розкриваються, якщо клацнути на них лівою клавішею миші.

Під головним меню розміщена панель **Стандартна**, на якій знаходяться значки найбільш вживаних команд.

Під панеллю знаходиться робоче вікно, що займає більшу частину екрана.

У лівій частині головного вікна розташована панель з інструментами розміщення умовних графічних зображень (символів) об'єктів.

Праворуч робочого вікна розміщується панель інструментів документування **DocTool**.

У нижній частині головного вікна під палітрою кольорів є ***рядок повідомлень***, у якому відображується інформація про виділений графічний об'єкт або обрану команду меню. Якщо повідомлень немає, то рядок залишається пустим.

Під рядком повідомлень розміщено ***рядок стану***, що складається з послідовності полів і кнопок. У перших двох полях праворуч вказуються поточні координати курсора **X**, **Y** під час його переміщення по робочому столі.

Поряд з полями координат знаходиться кнопка вибору типу сітки, що зображена на документі для зручного розміщення об'єктів. Абсолютна сітка, що вибирається кнопкою **Abs**, має початок координат у нижньому лівому куті робочого аркуша. Початок координат відносної сітки, що вибирається кнопкою **Rel**, вказують курсором після переходу в режим **Rel**.

Далі розміщено поле вибору кроку сітки. Щоб встановити новий крок сітки, значення кроку набирають у полі й натискають клавішу Enter. Поряд з полем — кнопка випадного меню кроків сітки, за допомогою якої можна вибрати і встановити необхідний крок.

Наступною є кнопка із зображенням літери **M**, призначена для створення макрокоманд. Натисканням цієї кнопки розпочинається запис у тимчасовий файл послідовності команд, які в цей час виводяться на екран. Повторне натискання цієї кнопки припиняє запис команд. Записана в тимчасовий файл послідовність команд виконується, якщо натиснути на клавішу **E**.

Принципова електрична схема, що створюється, може розміщуватися на кількох аркушах (Sheet), які нумеруються. Поточний аркуш схеми розміщено в полі, що знаходиться поруч з кнопкою макрокоманд. Потрібний аркуш схеми можна активізувати за допомогою випадного меню, кнопка якого знаходиться поруч з полем аркуша.

Далі розміщується поле товщини лінії разом зі списком значень товщини, який розкривається за допомогою відповідної кнопки. Потрібне значення зі списку вибирається за допомогою миші.

У кінці рядка стану, зліва розміщено інформаційне поле, в якому відображається така інформація: тип, позиційне позначення і кількість вибраних об'єктів; значення приросту координат під час проведення сегментів ліній чи переміщення вибраних об'єктів; ім'я вибраного електричного з'єднання; відстань між вибраними точками, їх проекції на координатні осі під час виконання команди **Edit-Measure.**

Настройка конфігурації

Перш ніж створювати принципові електричні схеми, слід вибрати параметри проекту, інструментів і засобів графічного редактора. Ці параметри задаються за допомогою діалогових панелей, що викликаються командами меню **Options.**

За допомогою панелі **Options-Configure** вибирається стандартний розмір (A0, A1, ...) робочого аркуша, розмір рамки, основні й додаткові написи. Вибирається система одиниць: mil — міли (1 міл = 0,001 дюйма = 0,0254 мм), мм — міліметри, inch — дюйми. Задається режим введення електричних з'єднань і ліній: ортогональних, діагональних або ліній під довільним кутом (рис. 14.2).

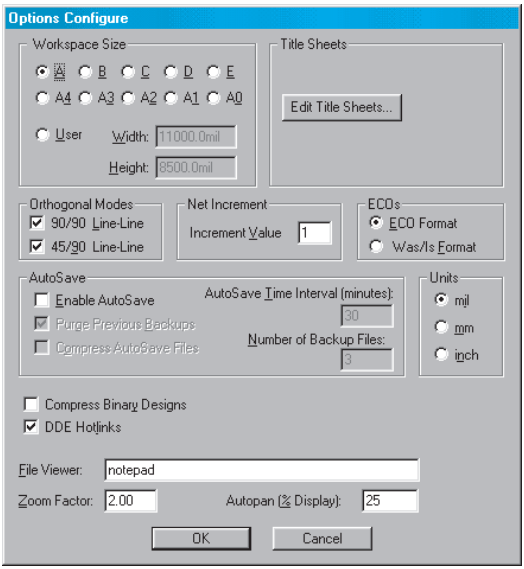


Рис. 14.2

Крок сітки задається в меню **Options-Grids** (рис. 14.3). Щоб додати до списку кроків нове значення, його слід набрати і натиснути кнопку **Add**.

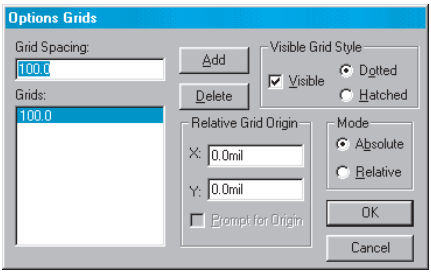


Рис. 14.3

Діалогова панель **Options Display** (рис. 14.4) має дві закладки: **Colors** і **Miscellaneous**. На закладці **Colors** задаються кольори різних графічних об'єктів: умовних графічних зображень (символів) компонент (Part), провідників (Wire), шин (Bus), точок з'єднання провідників (Junction), виводів компонент (Pin), ліній (Line), полігонів (Polygon), тексту (Text), а також колір тла (Background), сітки (Grid), виділеного (Highlight) і вибраного (Select) об'єкта, заголовка схеми (Title), атрибутів компонент (Part Attr) і з'єднань (Wire Attr).

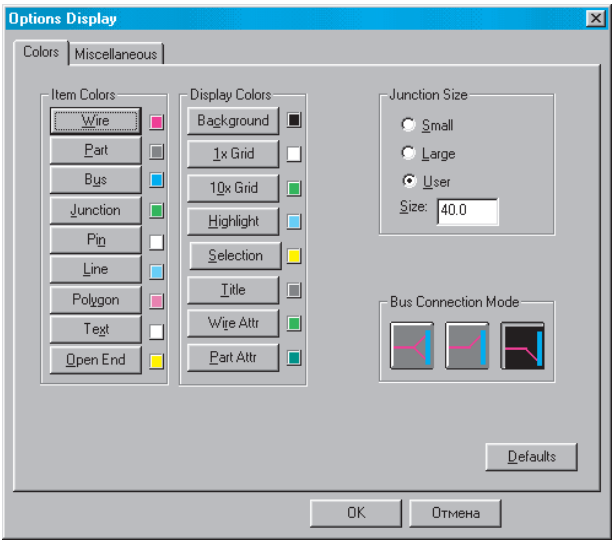


Рис. 14.4

На закладці **Miscellaneous** задається дозвіл чи заборона виводу на схему маркерів помилок, зовнішній вигляд курсора,

відображення на екрані не з'єднаних виводів і провідників, маркерів помилок, якщо вони перекриваються, номерів секцій компонент. Крім того, задається вивід на екран смуг прокрутки і підказок.

Натискання на кнопку **Default** на панелі **Options Display** призначає всім параметрам значення за замовчанням, **OK** — встановлені користувачем, **Cancel** — скасовує їх.

За допомогою команд **Options-Current Wire** і **Options-Current Line** вибирається ширина друкованих провідників і ширина ліній.

Текстові параметри встановлюються за допомогою команд меню **Options-Text Style.**

Параметри настройки конфігурації програми **P-CAD Schematic** заносяться у файл SCH.INI і зберігаються для наступної роботи.

Створення електричних принципових схем

Після настройки конфігурації програми **P-CAD Schematic** приступають до створення принципових схем.

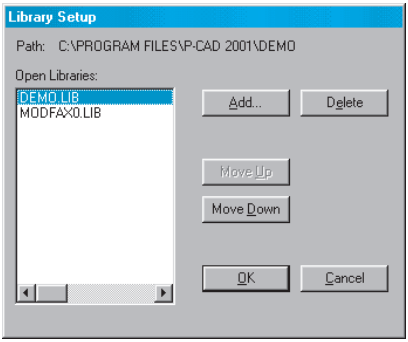


Рис. 14.5

Завантаження бібліотек. В інтегрованих бібліотеках **P-CAD** містяться такі об'єкти як: ***компоненти*** (*components*), ***корпуси*** (*patterns*), ***символи***, тобто умовні графічні зображення (symbols). Кожний компонент може складатися з однієї або кількох частин, наприклад, мікросхема 155ЛА3 складається з чотирьох логічних елементів **I-HI**. Перед нанесенням на схему символів компонент, за командою **Library-Setup** і діалоговою панеллю цієї команди (рис. 14.5) забезпечується доступ до необхідних бібліотек компонент. У разі потреби до списку відкритих бібліотек додають нові за допомогою кнопки **Add.**

Розміщення компонент на схемі. У режим розміщення символів компонент на схемі переходять за командою **Place>Part** (рис. 14.6). Якщо після вибору команди клацнути мишею в будь-якій точці схеми, то виникає діалогова панель **Place Part.**

На панелі **Library Setup** вибирається ім'я однієї з відкритих бібліотек, і список її компонент виводиться у полі **Component Name** панелі **Place Part.** Ім'я потрібної компоненти вибирають з цього списку або вводять у верхньому рядку поля. Якщо натиснути на кнопку **Browse**, то можна переглянути графічне зображення компоненти.

У графі **Alternate** можна вибрати одне із зображень, що виконані на основі різних стандартів: **Normal** — у звичайному стандарті,

De Morgan — за стандартом зображення логічних функцій, **IEEE** — за стандартом Інституту інженерів з електротехніки й електроніки (США).

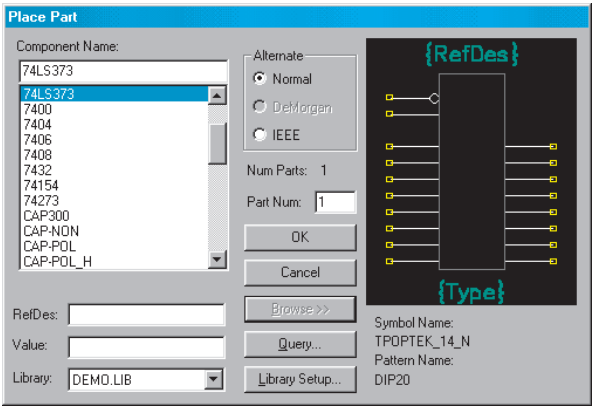


Рис. 14.6

У графі **Num Parts** вказується кількість частин, з яких складається компонента, наприклад, мікросхема 155ЛА3 складається з 4 логічних елементів.

У полі **Part Num** за замовчанням вказується номер 1 частини компоненти, хоч за потребою його можна змінити.

Процедура пошуку потрібної компоненти у відкритих бібліотеках активізується кнопкою **Query**. У відповідній діалоговій панелі, у стовпчику **Criteria** задають критерії пошуку, а в стовпчику **Show** відмічають, які характеристики знайдених компонент потрібно відображати в таблицях результату пошуку.

Позиційні позначення компонент на схемі проставляються автоматично. За потребою позиційне позначення можна змінити, вказавши його в полі **RefDes** в явному вигляді.

У полі **Value** проставляється номінал компоненти, наприклад, опір резистора, ємність конденсатора тощо.

Після вибору на панелі **Place Part** потрібної компоненти і введення всіх потрібних параметрів натискають **ОК**. Курсор набирає форму хрестика для точного позиціонування у вузлах сітки. Безпосереднє розміщення символу компоненти на схемі здійснюється, якщо клацнути лівою клавішею миші. Поки не відпущена ліва клавіша миші, символ можна переміщувати по екрану, повернути на 90° проти ходу годинникової стрілки, якщо натиснути клавішу **R**, віддзеркалити відносно вертикальної осі, якщо натиснути на клавішу **F**.

Розміщення електричних провідників. Компоненти, що розміщені на схемі, з'єднуються електричними провідниками. Для цього вибирається команда **Place>Wire** і курсор набирає форму хрестика. Початкову точку провідника відмічають, клацнувши лівою клавішею миші. Кожне натискання лівої клавіші миші фіксує точку зламу, а натискання на клавішу **F** змінює його орієнтацію. Щоб завершити операцію прокладання провідника, слід натиснути праву клавішу миші або клавішу **Esc**. Ширина провідника встановлюється командою **Options>Current Wire**.

У програмі передбачено автоматичне присвоювання імен провідникам, що прокладаються. Ім'я провідника відображається у полі інформації рядка стану. У разі потреби можна змінити ім'я, що присвоєно провіднику автоматично. Для цього потрібно вибрати потрібний провідник і відкрити правою клавішею миші панель редагування. На панелі редагування відкрити пункт **Властивості** (Properties) (рис. 14.7).

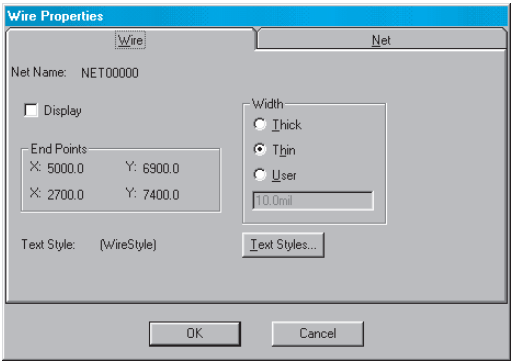


Рис. 14.7

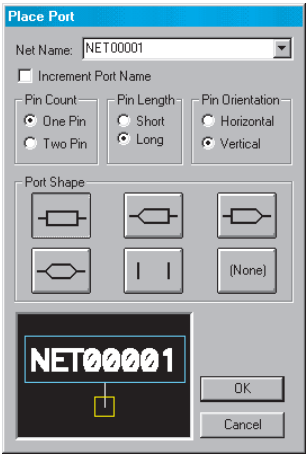


Рис. 14.8

У полі **Net Name** замінити призначене системою ім'я на інше. На Т-подібних з'єднаннях провідників автоматично проставляється точка «пайки» **(Junction)**. Щоб створити Т-подібне електричне з'єднання, необхідно клацнути мишею у точці перетину, а потім продовжити прокладання провідника (рис. 14.8).

Розміщення шин. Щоб розмістити на схемі шину, необхідно вибрати команду **Place>Bus**. Початок шини і точки її зламу відмічають, клацнувши лівою клавішею миші. Побудова шини завершується, якщо клацнути правою клавішею миші або натиснути на клавішу **Esc**.

Щоб задати ім'я провідників, що утворюють шину, до них необхідно підключити спеціальні порти за командою **Place>Port**. У діалоговому вікні, що виникає після вибору цієї команди, в полі **Net Name** вказати ім'я провідника (рис. 14.8).

Редагування схеми. Щоб перемістити компоненту, провідник або шину, їх потрібно спочатку вибрати за допомогою миші, а потім переміщувати рухом миші. Якщо необхідно перемістити кілька об'єктів, наприклад компоненту зі зв'язаними з нею з'єднаннями, то одночасно з натисканням на ліву клавішу миші, слід натиснути і втримувати у натиснутому стані клавішу **Ctrl**.

Скопіювати вибрану групу об'єктів можна послідовним виконанням команд **Edit>Copy**, **Edit>Paste** або натиснути клавішу **Alt** і перемістити мишу.

Збереження проекту. Створену і відредаговану електричну принципову схему (проект) електронного пристрою зберігають за командою **File>Save** у початковому файлі, а за командою **File>Save As** у файлі з новим іменем.

Застосування модулів

Електронні пристрої часто містять однотипні вузли, що багаторазово повторюються у різних місцях електричної схеми, наприклад, підсилювачі, формувачі, модулятори, демодулятори в аналогових схемах, тригери, лічильники, мультиплексори, шифратори, дешифратори в цифрових схемах. Для спрощення побудови електричних схем такі однотипні вузли зручно оформити у вигляді окремого ***модуля*** і далі використовувати цей модуль як елемент електричної принципової схеми. Ефективність від застосування модулів значно збільшиться, якщо реалізувати ієрархічні багаторівневі структури модулів, тобто якщо модуль з одного боку міститиме модулі нижчого ієрархічного рівня, а з іншого — входитиме до складу модулів вищого ієрархічного рівня. У системі **P-CAD** є засоби для створення таких модулів.

Модулі ієрархічної структури створюються у діалоговому режимі за допомогою **Майстра створення модулів (Module Wizard)** (рис. 14.9), який викликається командою **Utils>Module Wizard**. У діалоговому вікні, що виникає під час вибору цієї команди, спочатку необхідно вказати яким чином створюватиметься модуль: на основі вже існуючих модулів чи створюватиметься новий модуль. Якщо вибрати другий варіант, то на

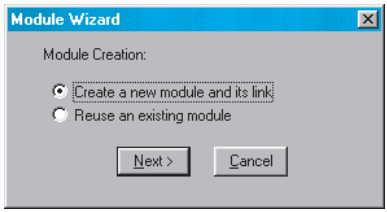


Рис. 14.9

екрані виникає нове діалогове вікно (рис. 14.10), в якому необхідно вказати такі параметри: Module Name — ім'я модуля; Number of input pins — кількість вхідних виводів модуля (розміщені зліва); Number of output pins — кількість вихідних виводів модуля (розміщені справа); Number of bidirect pins — кількість двоспрямованих виводів модуля (розміщені справа); Symbol width — ширина символу модуля; Pin length — довжина виводу; Pin spacing — відстань між виводами; Create Corresponding Link — створення зв'язку модуля з його схемою; Link Name — ім'я зв'язку модуля з його схемою; Save in Library — запис модуля в бібліотеку; Library — ім'я бібліотеки, в яку буде записано модуль; Pin Designator — номер виводу, що присвоюється за замовчанням; Pin Name — ім'я виводу, що призначається за замовчанням з можливістю його зміни. Після введення всіх параметрів натискають на кнопку **Далі** (Next).

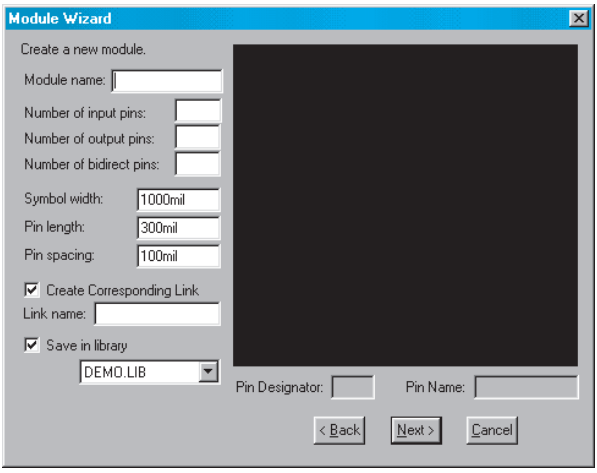


Рис. 14.10

У вікні, що з'явилося, подається інформація про зв'язки модуля: ім'я нового зв'язку; імена аркушів, на яких розміщена схема проекту; ім'я аркуша, на якому має бути розміщена схема модуля; позиційне позначення модуля, що збігається з іменем аркуша. Після натискання на кнопку **Далі** (Next), з'являється останнє вікно **Майстра створення модулів**. У цьому вікні повідомляється про готовність розміщення символу модуля на схемі і вказується присвоєне за замовчанням позиційне позначення модуля. Щоб завершити створення символу модуля, слід натиснути на кнопку **ОК**.

Після розміщення на схемі символів модулів слід виконати команду **Utils>Hierarchy** для того, щоб проставити всі позиційні

позначення компонент та імена електричних з'єднань. Цю команду слід виконати до коригування схеми і складання списків електричних зв'язків.

Перевірка (верифікація) схеми на наявність помилок

Після створення принципової електричної схеми доцільно виявити помилки, яких припустилися під час проектування, виправити їх і тільки після цього перейти до проектування друкованої плати. Перевірка здійснюється за допомогою спеціальної програми **ERC (*Electrical Rules Check*)**, що викликається командою **Utils>ERC**. У діалоговому вікні (рис. 14.11), що виникає після вибору команди, задається перелік перевірок, результати яких наводяться в текстовому звіті: з'єднання, що мають один вузол; з'єднання, що не мають вузлів; правильність електричних з'єднань. Наприклад, з'єднання двох вихідних виводів чи з'єднання вихідного виводу і джерело живлення є неправильним; непідключені виводи компонент; непідключені провідники; наявність провідників, що входять у шину, але не виходять з неї; правильність розміщення компонент (одна компонента не може бути розміщена поверх іншої); правильність підключення шин живлення і землі; правильність побудови ієрархічних структур.

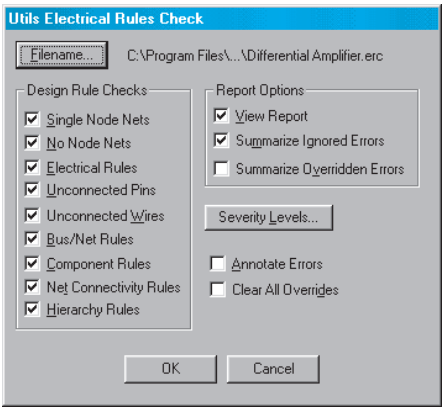


Рис. 14.11

Крім переліку перевірок, у діалоговому вікні, що відкривається після натискання на кнопку **Severity Level**, встановлюється ступінь важливості виявленої невідповідності: **Error** (помилка), **Warning** (попередження), **Ignored** (ігнорувати).

Щоб мати змогу переглянути звіт перевірки і повідомлення про помилки, слід увімкнути опцію **View Report**, а увімкнувши опцію **Annotate Errors**, отримують індикацію помилок на схемі.

Пошук помилок програма **ERC** починає після ввімкнення кнопки **OK**. Інформація про помилки відзначається на схемі індикаторами й виводиться у текстовий звіт.

Вивід даних

Результати проектування виводяться у **P-CAD Schematic** у вигляді: схеми, надрукованої на принтері або плотері; списку з'єднань; текстових звітів.

Для підготовки до друку принципової електричної схеми на принтері або плотері виконують команду **File>Print Setup** та у відповідному діалоговому вікні вибирають тип пристрою виводу, розмір і орієнтацію аркуша.

Параметри друку встановлюються у діалоговому вікні, що виникає з вибором команди **File>Print**. Клавішею **Page Setup** відкривається додаткове вікно, в якому встановлюється розмір креслення, відстань до країв аркуша тощо. У додатковому діалоговому вікні **Print Options**, що відкривається однойменною кнопкою, задається колір графічних об'єктів.

Список з'єднань складається зі списку компонент і провідників із зазначенням номерів виводів компонент, до яких вони підключені. Він використовується для розміщення на друкованій платі корпусів компонент із зазначенням їх електричних з'єднань згідно з електричною схемою. За командою **Utils>Generate Netlist** відкривається діалогове вікно, в якому зі списку вибирається формат списку з'єднань (рис. 14.12).

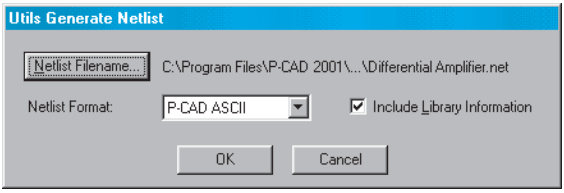


Рис. 14.12

За командою **File>Report** створюють текстові звіти про виконаний проект. У відповідному діалоговому вікні, в розділі **Report to Generate**, вибирається вид звіту: список атрибутів і провідників, список компонент, список глобальних електричних з'єднань, список позиційних позначень компонент тощо.

14.3 Створення проектів друкованих плат електронних пристроїв

Сучасні електронні пристрої побудовані на основі цифрових, аналогових, аналого-цифрових мікросхем різного ступеня інтеграції, змонтованих на багатошарових друкованих платах.

Конструкція багатошарової друкованої плати розробляється за допомогою графічного редактора **P-CAD PCB**. Вікно редактора має типовий для прикладних програм Windows вигляд і має багато спільного з вікном редактора **P-CAD Schematic** (рис. 14.13).

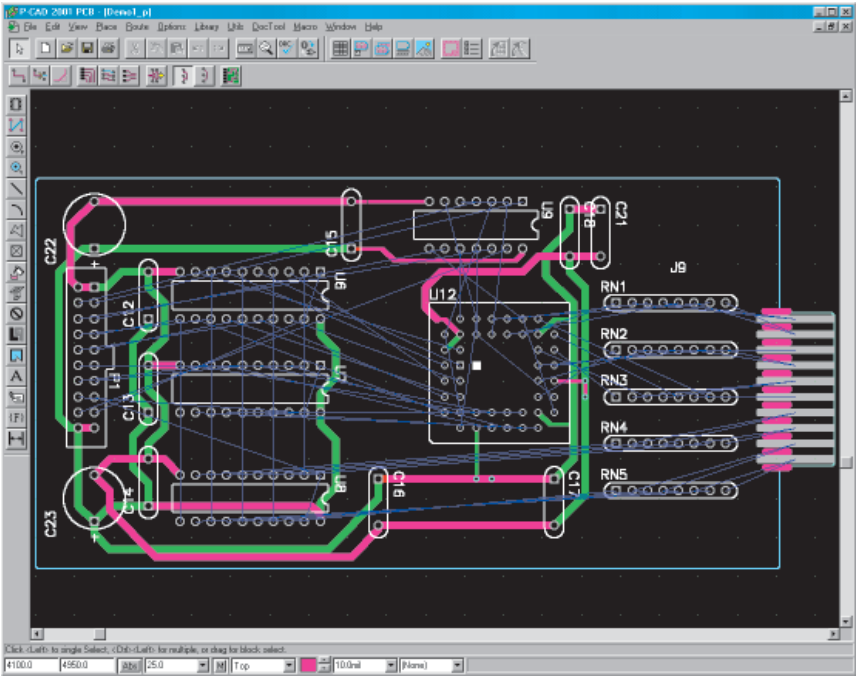


Рис. 14.13

Настройка конфігурації P-CAD PCB

На початку роботи з редактором **P-CAD PCB** необхідно настроїти його конфігурацію, вибравши для цього з меню **Options** команди **Configure, Display, Layers, Grids** та ін.

За командою **Options>Configure** вибирають такі глобальні параметри проекту як: систему одиниць, розміри робочої області тощо.

Колір об'єктів на різних шарах друкованої плати і ряд інших параметрів екрана встановлюють за командою **Options> >Display**. Для кожного об'єкта можна встановити однаковий колір на всіх шарах або всім об'єктам одного шару надати однакового кольору. Основними об'єктами, що розміщені на друкованій платі, є: провідники і лінії, виводи компонентів, тобто контактні площадки, перехідні отвори, полігони, текст.

У діалоговому вікні команди **Options>Selection Mask** встановлюють правила вибору групи об'єктів і окремого об'єкта.

Як уже зазначалося, сучасні електронні пристрої реалізуються на багатошарових друкованих платах. Кількість шарів **(Layers)**

на платі та їх властивості задаються у діалоговому вікні команди **Options>Layers** (рис. 14.14). За замовчанням редактор **P-CAD PCB** встановлює 11 шарів, хоч всього може бути до 99 шарів. Шари поділяються на такі типи: сигнальні — в яких розміщено провідники сигналів (позначаються символом **S**); металізації — в яких розведено шини живлення і землі (позначаються символом **P**); допоміжні (несигнальні позначаються символом **N**). Кожний шар може бути ввімкнений або вимкнений.

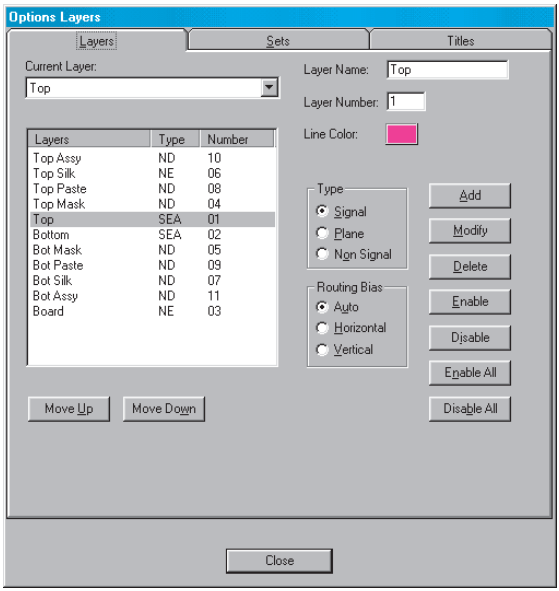


Рис. 14.14

Список значень ширини провідників і ліній складається за допомогою команди **Options>Current Line.** Ширину виділеного провідника чи лінії вибирають з цього списку за допомогою рядка стану.

Контактні площадки **(Pad)** і перехідні отвори **(Vie)**, що розміщені на різних шарах одні над одними, стовпчиком, називаються ***стеком***. У діалоговому вікні команди **Options>Pad Style** відкривається список стилів контактних площадок, а у вікні **Options>Vie Style** — список стилів перехідних отворів. У цих відкритих вікнах задають форму, геометричні розміри та інші параметри контактних площадок і перехідних отворів.

Стиль текстових написів на друкованій платі, тобто форму і розмір шрифту, та інші параметри вибирають у діалоговому вікні команди **Options>Text Style.**

Перед розміщенням на друкованій платі компонент вручну або за допомогою процедури упаковки принципової електричної

схеми необхідно забезпечити доступ до бібліотек, тобто до баз даних, у яких знаходяться ці компоненти. Бібліотеки підключаються за командою **Library>Setup**. За допомогою кнопки **Add** у діалоговому вікні цієї команди можна додавати потрібну бібліотеку до списку відкритих бібліотек, а за допомогою кнопки **Delete** вилучати бібліотеку з цього списку, щоб звільнити місце для інших.

Розробка конструкції друкованої плати

Розробку конструкції нової друкованої плати починають з виконання команди **File>New** і настройки конфігурації графічного редактора **P-CAD PCB**. Після цього на шарі **Board** за командою **Place Line** наносять контур друкованої плати у вигляді замкненої ламаної лінії.

Розміщувати компоненти і з'єднувати їх виводи лініями електричного зв'язку можна вручну за допомогою команд **Place> Component**, **Place>Connection**. Такий спосіб надзвичайно трудомісткий і застосовується дуже рідко для проектування нескладних пристроїв. Здебільшого на друковану плату переносять електричну схему, створену за допомогою редактора **P-CAD Schematic**. Процедура перенесення схеми на друковану плату називається ***упаковкою*** схеми. Починають упаковку завантаженням файла списку з'єднань за командою **Utils>Load Netlist**, цей файл створено за допомогою редактора **P-CAD Schematic**. У діалоговому вікні цієї команди задають необхідний формат списку з'єднань та інші параметри.

Після завантаження списку з'єднань (упаковки схеми) приступають до розміщення компонент всередині контуру друкованої плати. Оптимальне розміщення компонент багато в чому визначає вдале трасування електричних з'єднань і працездатність електронного пристрою. Засоби автоматичного розміщення компонент має тільки система **SPECCTRA**, що викликається командою **Place>Autoplace**, проте можливості автоматичного розміщення обмежені, особливо для аналогових пристроїв, тому розміщення компонент на друкованій платі виконується вручну.

Щоб перемістити об'єкт, його спочатку необхідно вибрати, клацнувши на ньому і утримуючи у натиснутому стані ліву клавішу миші. Рухом миші об'єкт переміщують, причому на екрані з'являється зображення точки прив'язки у вигляді квадратика з хрестиком. Вибраний об'єкт повернеться на кут 90° проти ходу годинникової стрілки, якщо натиснути клавішу **R**. Якщо одночасно натиснути клавіші **Shift+R**, то вибраний об'єкт повернеться на кут, заданий на діалоговому вікні команди **Options>Configure**. Вибраний об'єкт можна віддзеркалити відносно вертикальної осі

і перенести компоненти на протилежний бік друкованої плати, якщо натиснути клавішу **F**.

Лінії електричних зв'язків переміщуються разом з компонентами і це допомагає правильно їх розмістити. Розміщуючи однотипні компоненти, їх зручно автоматично вирівняти відносно вибраної точки відліку. Для цього, компоненти, що підлягають вирівнюванню, вибирають за чергою, натискуючи ліву клавішу миші. Вибираючи другий і наступні компоненти, слід натиснути й утримувати у натиснутому стані клавішу **Ctrl**. Після вибору групи однотипних компонент відкривають діалогове вікно, клацнувши правою клавішею миші. У діалоговому вікні вибирають опцію **Selections Point** і вказують курсором миші точку відліку, клацнувши лівою клавішею. Потім вибирають опцію **Align** (вирівнювання) й у вікні, що відкрилося, вибирають тип вирівнювання: по горизонталі відносно встановленої точки відліку; по вертикалі відносно точки відліку; у найближчі вузли сітки; на рівні відстані.

За потребою замінюють тип корпусу компоненти, для чого курсором миші вибирають компоненту і викликають діалогове вікно, клацнувши правою клавішею миші або за допомогою команди **Edit Properties**. У графі **Type** вибирають тип корпусу зі списку корпусів.

Після завершення розміщення компонент доцільно виконати мінімізацію електричних з'єднань шляхом перестановки логічно еквівалентних частин компонент і їх виводів за допомогою команди **Utils>Optimize Nets**.

Перед тим, як почати трасування електричних з'єднань, задають правила і параметри трасування і обмеження. Перед початком трасування в діалоговому вікні команди **Options>Grids** задають потрібний крок сітки і, за потребою, встановлюють режим нерегулярної сітки й задають її параметри. Закінчуючи підготовку до трасування, у діалоговому вікні команди **Options Rules** задають правила проектування й допустимі зазори для кожного шару між такими об'єктами, як: контактна площадка, провідник, перехідний отвір.

Для керування розміщенням компонент і трасуванням електричних з'єднань в автоматичному та інтерактивному режимах, а також для перевірки правильності електричних з'єднань, компонентам, провідникам та іншим об'єктам присвоюються атрибути. Атрибути вибраному електричному з'єднанню присвоюються в діалоговому вікні команди **Edit>Nets**. Атрибути компонент вводять аналогічно, клацнувши після вибору потрібної компоненти правою клавішею миші. У діалоговому вікні, що з'являється, вказують

пункт **Properties**, а у діалоговому вікні, що з'являється після вибору цього пункту, вибирають потрібні атрибути на панелі **Attributes**.

Трасування електричних з'єднань ***вручну*** здійснюється за командою **Route>Manual**. Початок траси фіксують клацанням лівою клавішею миші всередині контуру контактної площадки, перехідного отвору або вже існуючого провідника. Утримуючи ліву клавішу миші у натиснутому стані і переміщуючи курсор, прокладають сегмент провідника. Якщо клавішу відпустити, то фіксується точка зламу траси. Курсор також можна переміщувати за допомогою функціональних клавіш зі стрілками на один крок сітки, натискуючи ліву клавішу миші. У цьому випадку значення приросту координат курсора відносно останньої точки зламу з'являється у рядку стану. Натисканням клавіші **O** переключають характер зламу з ортогонального на діагональний і навпаки. Натисканням на клавішу **F** змінюють орієнтацію траси провідника.

Щоб перейти на інший шар і там продовжити трасування, натискають на клавішу **L**, а щоб повернутися на попередній шар — клавішу **Shift L**. При цьому автоматично встановлюється перехідний отвір між шарами з поточною конфігурацією, заданою заздалегідь командою **Options>Via Style**. Не перериваючи трасування, можна змінити товщину провідника командою **Options>Current Line** або за допомогою рядка стану.

Порушення зазору між трасою провідника і виводом компоненти, перехідного отвору чи трасою іншого провідника відмічається індикатором помилок у вигляді кола з хрестиком, якщо увімкнений режим перевірки **Online DRC**. Щоб витерти прокладені сегменти траси з помилками, необхідно натиснути на клавішу **Backspace** стільки разів, скільки сегментів траси потрібно витерти.

Щоб завершити трасування провідника, необхідно натиснути на праву клавішу миші або вибрати іншу команду. Прокладання траси можна припинити, але не завершити, якщо натиснути на клавішу з косою рискою \ або /.

Під час трасування корисно звертати увагу на рядок інформації у нижній частині екрана, де виводиться інформація про прирости координат кожного сегмента, загальну довжину провідника до точки зламу, кількість помилок під час прокладання траси.

Інтерактивне трасування з'єднань здійснюється за командою **Route>Interactive**. Трасу починають, натиснувши ліву клавішу миші на контактній площадці компоненти або на будь-якій точці раніше прокладеної траси, а прокладають рухом курсора при натиснутій лівій клавіші миші. Під час прокладання траси автоматично огинаються перепони: провідники, контактні площадки,

перехідні отвори, області металізації. Щоб завершити черговий сегмент траси, потрібно відпустити натиснуту клавішу миші. Натиснувши праву клавішу миші під час прокладання траси, відкривають діалогове вікно, за допомогою якого можна:

- ❒ завершити прокладання траси з дотриманням встановленого заздалегідь режиму введення провідників і допустимих зазорів;
- ❒ увімкнути режим відштовхування вже прокладених провідників;
- ❒ припинити прокладання траси зі скасуванням введення останнього сегмента;
- ❒ змінити параметри конфігурації за командою **Option>>Configure;**
- ❒ відкрити вікно команди **Option>Layers** для зміни структури шарів плати;
- ❒ відкрити вікно команди **Option>Via Style** для вибору типу перехідного отвору чи його редагування.

Клавіші **O, F,** \, / мають такі самі призначення, що й у режимі ручного прокладання траси.

Перевірка друкованої плати

Розроблену конструкцію друкованої плати слід перевірити за допомогою спеціальної програми **DRC(Design Rule Checking),** перш ніж виготовляти фотошаблони. Програма перевірки викликається командою **Utils>DRC.**

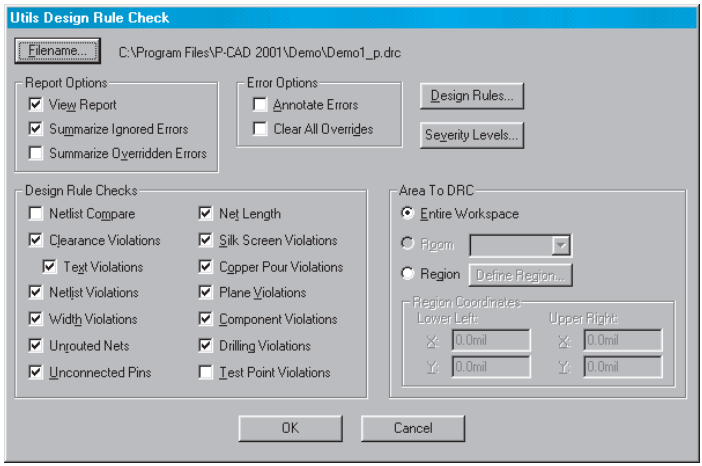


Рис. 14.15

У діалоговому вікні цієї команди (рис. 14.15) можна задати такі види перевірок:

- **Netlist Compare** — порівняння списку з'єднань розробленої друкованої плати з принциповою схемою;
- **Clearance Violations** — порушення зазорів;
- **Text Violations** — порушення зазорів між текстом, розміщеним на сигнальних шарах, і металізованими об'єктами;
- **Netlist Violations** — перевірка відповідності електричних з'єднань на друкованій платі зі списком електричних зв'язків;
- **Width Violations** — перевірка виконання обмежень на ширину провідника, заданого за допомогою атрибута **Width**;
- **Unrouted Nets —** не розведені з'єднання;
- **Unconnected Pin —** не з'єднані виводи компонент;
- **Net Legth —** виконання обмежень на довжину провідників тощо.

У результаті здійснення перевірок перелік похибок заноситься у файл з розширенням імені *.DRC. Щоб змінити ім'я цього файла, слід натиснути на кнопку **Filename**. Якщо ввімкнути опцію **View Report**, то на екран виведеться текст звіту про перевірку, а **Annotate Errors** — відмічає на друкованій платі місця помилок спеціальним значком — кружечком з хрестиком (⊗). Якщо виділити цей значок лівою клавішею миші, а потім клацнути правою клавішею, то відкривається вікно, в якому виводиться інформація про помилку. За допомогою кнопок **Next Previous** можна переглянути повідомлення про попередні і наступні помилки.

Оперативну перевірку друкованої плати у процесі її створення можна ввімкнути за допомогою опції **Enable Online DRC** на закладці **Online DRC** команди **Option>Configure**. У цьому режимі під час ручного трасування провідників виводяться маркери помилок трасування, перевіряється дотримання зазорів і вимог до ширини провідника.

Виведення результатів проектування

Результати електронного проектування конструкцій друкованих плат можна вивести на принтери і плотери різних типів і таким чином створити тверду копію технічної документації. За допомогою команди **File>Print Setup** встановлюється тип пристрою виведення, розмір паперу, спосіб подачі, орієнтація, кількість копій та інші параметри друку. В діалоговому вікні команди **File>Print** у відповідному полі встановлюється мінімальна ширина лінії, і складається завдання на друкування. Кожному завданню присвоюється ім'я і виводиться діалогове вікно цього завдання, в якому вказуються всі необхідні параметри. Кнопкою **Colors** вибирається потрібний колір документа. Щоб переглянути зображення перед

друкуванням, можна скористатися режимом **Print Preview**, натиснувши відповідну кнопку. Безпосереднє друкування документа здійснюється, якщо натиснути кнопку **Generate Printout**.

Для виготовлення багатошарової друкованої плати спочатку необхідно виготовити фотошаблони всіх її шарів. Вимоги до точності фотошаблонів дуже високі, і тому їх виготовляють на спеціальному прецизійному обладнанні — фотоплотерах Gerber. За допомогою **P-CAD PCB** можна створити керівні файли для керування пристроями фотоплотера в процесі виготовлення фотошаблону. Слід зазначити, що програма може створити керівні файли з помилками, тому безпосередньо використовувати такі файли недоцільно. Для усунення помилок і доробки фотошаблонів призначено спеціальні програми **CAM 350** і **CAMtastic**. Керівні файли для фотоплотера, створені за допомогою програми **P-CAD PCB**, обробляються програмами **CAM 350** і **CAMtastic**, а потім повторно завантажуються у програму **P-CAD PCB**.

Керівні файли для фотоплотера Gerber створюються за допомогою команди **File>Export>Gerber**. У діалоговому вікні цієї команди задають розширення імені керівного файла, призначають завдання, створюють і редагують апертури фотоплотера, вибирають формат керівного файла.

Після визначення завдань, призначення списку апертур і вибору формату можна відправити на виконання одне чи кілька завдань кнопкою **Generate Output Files**. Точність виконання фотошаблону перевіряється командою **File>Import>Gerber**.

Одним з важливих етапів під час виготовлення друкованих багатошарових плат є свердління отворів у платах. Ця операція вимагає високої точності і виконується на прецизійних координатних свердлильних верстатах (типу Excellon) з числовим програмним керуванням. З цією метою створюються керівні файли за допомогою команди **File>Export>N/C Drill**. У діалоговому вікні цієї команди складаються завдання для свердління, вказуються розширення імен керівних файлів, складається таблиця свердл, встановлюється формат керівних файлів, вибирається система одиниць вимірювання. Керівний файл, із заданою у діалоговому вікні конфігурацією, створюється після натискання на кнопку **Generate Output Files**.

Одним з важливих видів технічної документації на виготовлення багатошарової друкованої плати є список з'єднань, у якому вказуються всі компоненти й електричні з'єднання, а також номери виводів компонент, до яких вони ввімкнені. Список з'єднань складається на основі створеної конструкції друкованої плати за командою **Utils>Generate Netlist**.

Текстові звіти можна створити за допомогою команди **File>Report**. У діалоговому вікні цієї команди вибирається вид звіту, його формат, пристрій виведення.

14.4 Робота з бібліотеками компонент

Як уже зазначалося сучасні електронні пристрої побудовані на основі мікросхем різного ступеня інтеграції. Крім мікросхем, у електронних пристроях застосовуються також напівпровідникові прилади (біполярні і польові транзистори, діоди, тиристори), дискретні електричні елементи: резистори, конденсатори, індуктивні елементи (трансформатори, дроселі, соленоїди тощо). Всі ці елементи є компонентами друкованих плат. У системі P-CAD дані про компоненти зберігаються у спеціальних базах даних, які називаються бібліотеками. У бібліотеці зберігається інформація таких типів: текстова інформація про компоненти, графічні зображення корпусів компонент, графічні зображення символів компонент.

Для керування базами даних призначені спеціальні програми Library Manager і Library Executive. Library Executive має всі функціональні можливості Library Manager і, крім того, має додаткові засоби. Це можливість вибору компонент за запитом по заданому набору атрибутів. Крім того до складу Library Executive входять спеціальні програми: Symbol Editor призначена для створення і редагування умовних графічних зображень компонент; Pattern Editor призначена для створення і редагування зображення корпусів компонент.

Вікно програми Library Manager або Library Executive є типовим вікном прикладної програми Windows (рис. 14.16). Інформація про компоненти виводиться за допомогою чотирьох діалогових вікон.

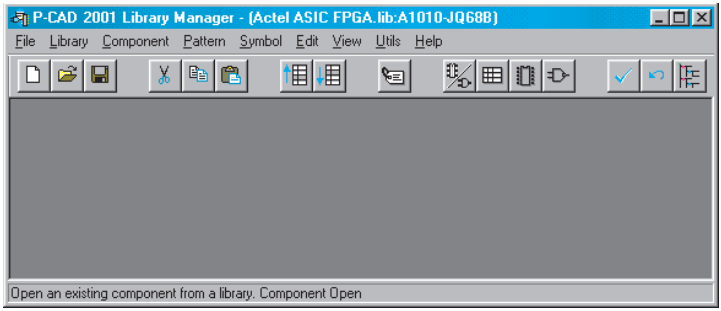


Рис. 14.16

Діалогове вікно **Component Information** з'являється під час вибору команд **Component>Open** і **Component>New**. У цьому діалоговому вікні можна задати тип корпусу компоненти, кількість секцій, загальну кількість виводів, префікс позиційного позначення, тип компоненти (звичайна компонента, джерело живлення, з'єднувач аркушів схеми, модуль ієрархічної структури), альтернативні зображення символів згідно з різними стандартами.

У діалоговому вікні **Symbol View** (рис. 14.17) наведено умовне графічне зображення (символ) компоненти і таблиця з інформацією про її виводи.

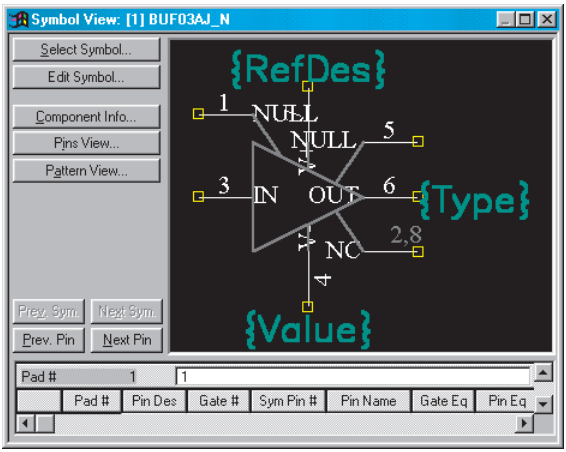


Рис. 14.17

Кожному виводу компоненти відповідає один рядок таблиці. У стовпчиках таблиці відображається така інформація: фізичний номер виводу відповідно до цоколівки, номер секції компоненти, порядковий номер та ім'я виводу символу секції, код логічної еквівалентності секції, код логічної еквівалентності виводу секції, електричний тип виводу, що необхідний для перевірки принципової схеми (вхід, вихід, виводи двоспрямований, живлення, пасивної компоненти тощо).

У діалоговому вікні **Pattern View** (рис. 14.18) наведено умовне графічне зображення корпусу компоненти і таблицю з інформацією про його виводи. Корпус, що призначений для даної компоненти, можна змінити на інший за допомогою кнопки **Select Pattern**.

У діалоговому вікні **Pins View** (рис. 14.19) наведено таблицю з інформацією про всі виводи компоненти, яку зручно редагувати. Якщо вибрати курсором будь-яку клітинку таблиці, то з'являється рядок з інформацією про вміст клітинки і панель для редагування клітинки.

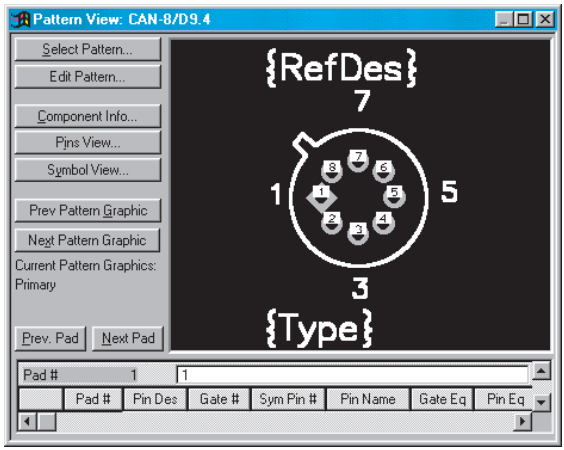
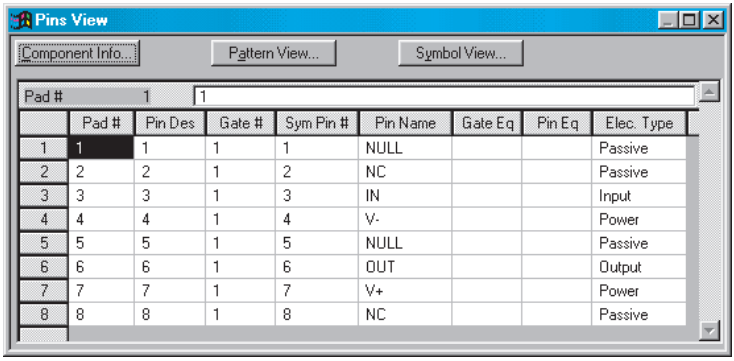


Рис. 14.18



	Pad #	Pin Des	Gate #	Sym Pin #	Pin Name	Gate Eq	Pin Eq	Elec. Type
1	1	1	1	1	NULL			Passive
2	2	2	1	2	NC			Passive
3	3	3	1	3	IN			Input
4	4	4	1	4	V-			Power
5	5	5	1	5	NULL			Passive
6	6	6	1	6	OUT			Output
7	7	7	1	7	V+			Power
8	8	8	1	8	NC			Passive

Рис. 14.19

Створення зображень корпусів і символів компонент

Зображення корпусів компонент можна створити за допомогою редактора P-CAD PCB і P-CAD Pattern Editor (рис. 14.20). Щоб створити зображення корпусу компоненти за допомогою редактора P-CAD PCB, необхідно вибрати систему одиниць за допомогою команди **Option> Configure** і встановити потрібний крок сітки за допомогою команди **Option> Grids.** Після цього необхідно розмістити виводи компоненти із заданим інтервалом і вибрати стиль стека контактних площадок. Далі наносять атрибути компоненти RefDes і Type. Автоматична нумерація виводів здійснюється командою **Utils>Renumber**, якщо перед цим ці виводи вибрати. На кінцевому етапі всі графічні об'єкти, що стосуються

зображення корпусу компоненти, вибирають і заносять в одну з відкритих бібліотек командою **Library>Pattern Save As.**

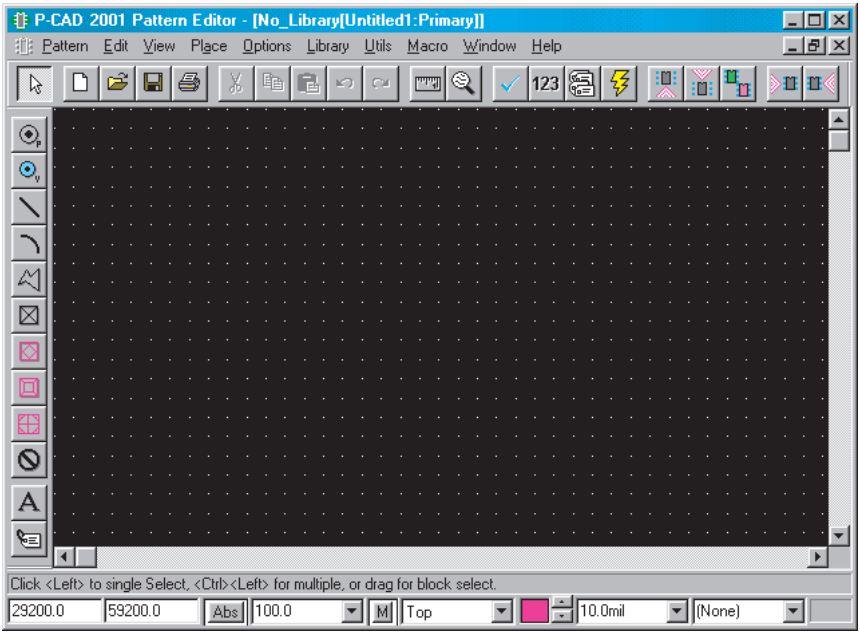


Рис. 14.20

Зображення корпусів створюються набагато простіше за допомогою спеціальної програми — редактора P-CAD Pattern Editor. До складу P-CAD Pattern Editor входить **Майстер створення зображення корпусів** Pattern Wizard, який дає змогу створювати зображення корпусів у автоматичному режимі.

Зображення умовних графічних зображень (символів) компонент можна створити за допомогою редактора P-CAD Schematic і P-CAD Symbol Editor.

Щоб створити зображення корпусу компоненти за допомогою редактора P-CAD PCB, необхідно вибрати систему одиниць за допомогою команди **Option>Configure** і встановити потрібний крок сітки за допомогою команди **Option>Grids.** Далі, за допомогою команд векторної графіки **Place>Line, Place>Arc** створюється умовне графічне зображення компоненти. Текстові написи наносяться за командою **Place>Text**, а виводи — **Place>Pin.** Атрибути символу RefDes і Type наносяться за командою **Place>Attribute.** Виводи нумерують за допомогою команди **Utils>Renumber.** На кінцевому етапі всі графічні об'єкти, що стосуються умовного

графічного зображення компоненти, вибирають і заносять в одну з відкритих бібліотек командою **Library>Pattern Save As.**

Зображення корпусів набагато простіше створюються за допомогою спеціальної програми — редактора P-CAD Symbol Editor, до його складу входить **Майстер створення зображення корпусів** Symbol Wizard, який дає змогу створювати умовні графічні зображення в автоматичному режимі.

Створити зображення корпусу й умовне графічне зображення компоненти можна безпосередньо засобами програми Library Manager або Library Executive. Для цього вибирається команда **Component>New.** У діалоговому вікні **Component Information** кнопкою **Select Pattern** відкривають вікно **Library Browse,** де розміщений список зображень корпусів, з якого можна вибрати необхідний корпус. Потім вводять таку інформацію про вибрану компоненту: кількість секцій, префікс позиційного позначення, ім'я умовного графічного зображення, тип, стиль, спосіб нумерації секцій, код логічної еквівалентності, альтернативні зображення символу. Створення компоненти завершується заповненням таблиці виводів, що виводиться на екран кнопкою **Pin Views.** Створену компоненту потрібно перевірити на наявність помилок командою **Component>Validate.** Якщо помилка виявлена, то виводиться повідомлення про неї. Після виправлення всіх помилок виконується команда збереження компоненти у поточній бібліотеці **Component>Save** або **Component>Save As.** Слід зазначити, що компоненту неможливо зберегти, якщо не виправити всі помилки.

14.5 Автоматичне прокладання трас провідників на друкованих платах

Після створення принципової електричної схеми електронного пристрою за допомогою програми P-CAD Schematic і конструкції друкованої багатошарової плати засобами програми P-CAD PCB приступають до прокладання трас електричних провідників на шарах друкованої плати. Щоб прокладати траси електричних провідників, призначено програми Quick Router і P-CAD Shape Router, що входять до складу системи автоматичного проектування і P-CAD. Крім цих двох програм для прокладання трас провідників застосовується також програма SPECCTRA, за допомогою якої здійснюється також автоматичне розміщення компонент на друкованих платах. Програма SPECCTRA не входить до складу P-CAD і постачається окремо, але в системі P-CAD є засоби для взаємодії та обміну інформацією між P-CAD і SPECCTRA.

Програма Quick Router

Програму Quick Router призначено для розведення нескладних друкованих плат, вона викликається з оболонки P-CAD PCB командою **Route>Autoroute** (рис. 14.21).

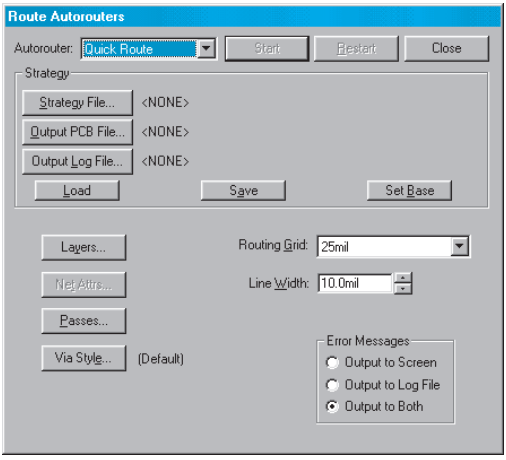


Рис. 14.21

У діалоговому вікні цієї команди програму вибирають зі списку **Autorouter** програм автоматичного трасування. У розділі **Strategy** діалогового вікна вибираються імена таких файлів:

- Strategy File — файл параметрів конфігурації з розширенням імені *.STR;
- Output PCB File — файл з результатами роботи програми Quick Router, який має розширення імені *.PCB;
- Output Log File — файл з протоколом трасування, що має розширення імені *.LOG.

За замовчанням всі ці файли мають ті самі імена, що й файл проекту, але на початку імені додається буква R. Файл стратегії містить параметри конфігурації програми прокладання трас. Для збереження параметрів конфігурації потрібно натиснути кнопку **Save**, розміщену в нижній частині розділу **Strategy**. Крім того, файл стратегії автоматично зберігається після початку трасування. Кнопка **Load** призначена для завантаження файла з параметрами конфігурації, встановленими користувачем, а кнопка **Set Base** — для завантаження параметрів конфігурації з файла, встановленого за замовчанням. У діалоговому вікні встановлюється також конфігурація шарів друкованої плати кнопкою **Layers**, атрибути з'єднувальних провідників — кнопкою **Net Attrs**, типи перехідних отворів — кнопкою **Via Style**.

Кнопкою **Passes** відкривається меню вибору **Pass Selection** типу проходу під час прокладання трас провідників:

- **Wide Line Routing** — прокладання трас широких провідників перед прокладанням інших трас;
- **Horizontal** — виконання найпростіших з'єднань по горизонталі без застосування перехідних отворів і з мінімальним відхиленням від прямих ліній;
- **Vertical** — виконання найпростіших з'єднань по вертикалі на довільному шарі без застосування перехідних отворів і з мінімальним відхиленням від прямих ліній;
- **"L" Routes (1via)** — формування перетину двох провідників і одного перехідного отвору, що мають форму букви **L**. Провідники розміщені на двох активних шарах перпендикулярно один одному;
- **"Z" Routes (2 vias)** — формування перетину трьох провідників і двох перехідних отворів, що мають форму букви **Z**;
- **"C" Routes (2 vias)** — формування перетину трьох провідників і одного перехідного отвору, що мають форму букви **C**. Провідники розміщені на двох активних шарах перпендикулярно один одному;
- **Any nodes (2 vias)** — спроба прокласти провідник між двома довільними вузлами електричного кола з використанням не більше двох перехідних отворів для забезпечення найповнішого розведення;
- **Maze Routes** — спосіб трасування типу «лабіринт», який ґрунтується на прив'язці сегментів траси до вузлів координатної сітки і здатний знайти шлях для оптимального прокладання траси провідника, якщо це фізично можливо;
- **Any nodes (maze)** — спроба прокласти провідник між двома довільними вузлами електричного кола з використанням алгоритму «лабіринт»;
- **Route Cleanup** — поліпшення зовнішнього вигляду розводки друкованої плати. Під час цього проходу частина провідників розводиться заново для їх випрямлення, якщо це можливо;
- **Via Minimization** — мінімізація кількості перехідних отворів.

Після вибору параметрів конфігурації програми переходять до безпосереднього автоматичного прокладання трас, для чого натискають на кнопку **Start** у діалоговому вікні програми. У процесі прокладання трас доступні команди з меню команд програми, зокрема за командою **Route>Info**, виводяться в спеціальному вікні **інформація про результати розводки**, а командою **Route> Cancel** трасування припиняється, і користувачеві пропонується вибрати

між двома режимами: припинити прокладання трас і зберегти результати роботи, чи припинити трасування і результати роботи не зберігати. У робочому вікні виводиться зображення друкованої плати, і після прокладання кожного провідника його зображення з'являється на екрані. У рядку стану виводяться повідомлення про етапи трасування. В кінці роботи програми виводиться повідомлення про завершення роботи. Одночасно з'являється інформація про відносну кількість (у відсотках) розведених з'єднань.

Інформація про стратегію трасування, результати виконання окремих етапів роботи і підсумкові дані заносяться до файла протоколу, що має розширення імені **.LOG**.

Як вже зазначалося, програма Quick Router призначена для прокладання трас провідників на друкованих платах невисокої складності, зокрема ця програма має такі обмеження: дозволено тільки прості контактні площадки і перехідні отвори; широкі провідники, що розводяться під час проходу **Wide Line Routing**, мусять мати відповідні атрибути; діаметр перехідного отвору не може більш ніж удвічі перевищувати поточний крок сітки; дозволені тільки певні розміри сітки, метрична сітка не допускається; ширина провідника не може перевищувати половину кроку сітки; для перехідних отворів неможливо створити окрему сітку; виводи компонентів можна повернути тільки на 90°; допускається не більше ніж чотири шари металізації.

Програма P-CAD Shape Router

Програма використовує принципи оптимізації нейронних мереж і призначена для прокладання трас провідників на багатошарових друкованих платах з високою насиченістю розміщення електронних компонент в автоматичному, інтерактивному і ручному режимах.

Програма ґрунтується на безсіткових технологіях, у яких кожний об'єкт моделюється як сукупність геометричних фігур (прямокутник, коло, дуга, відрізок прямої лінії тощо), а не прив'язується до координатної сітки.

Програма Shape-Based Router може викликатися безпосередньо з Windows, або з меню **Route** P-CAD PCB. У діалоговому вікні цієї команди у розділі **Strategy** можна вказати імена файлів: **PRF File** — початковий файл друкованої плати; **Output PCB File** — вихідний файл друкованої плати; **Output Log File** — вихідний файл протоколу трасування друкованої плати. За замовчанням ці файли отримують ті самі імена, що й файл проекту, тільки на початку імені додається буква **R**.

Настройка параметрів програми здійснюється командою **Option>Auto-router**. У діалоговому вікні цієї команди є три закладки.

На закладці **Routing Passes** вибирається спосіб прокладання трас провідників:

- **Memory** — трасування типу «пам'ять»;
- **Fan Out User SMD Pins —** генерація віялоподібно розміщених стрингерів для виводів планарних компонент;
- **Pattern** — трасування фрагментів друкованої плати з використанням типових зразків, що є в програмі;
- **Shape Route — Push and Shove** — розсування і відштовхування раніше прокладених трас, які заважають прокладанню нових;
- **Shape Route — Pip Up** — розривання і повторне прокладання трас, прокладених під час попередніх проходів з порушеннями;
- **Clean During Routing** — зменшення кількості згинів траси й усунення зайвих перехідних отворів під час прокладання трас;
- **Clean After Routing** — зменшення кількості згинів траси й усунення зайвих перехідних отворів після прокладання трас;
- **Evenly Space Traces** —рівномірний розподіл трас;
- **Add Testpoints** — додавання контрольних точок.

На закладці **Parameters** у полі **Router Directional** для кожного шару друкованої плати вибирають переважну: горизонтальну, вертикальну, з нахилом тощо. Крім того, на цій закладці призначаються такі параметри: система одиниць (дюйми, міліметри, сантиметри тощо); заборона або дозвіл використовувати перехідні отвори; заборона або дозвіл розміщувати перехідні отвори під контактними площадками; розмір каналу трасування, що дорівнює сумі ширини траси і допустимого зазору; діаметр контактних площадок і перехідних отворів; ширина трас; мінімально допустимий зазор між об'єктами.

На закладці **Testpoints** призначається пріоритет розміщення контрольних точок на друкованій платі, якщо на закладці **Routing Passes** відмічена опція **Add Testpoints**.

У діалоговому вікні команди **Edit>Net Attributes** призначаються атрибути провідникам друкованої плати: пріоритет трасування, мінімізація довжини провідників, спосіб трасування, шар трасування, ширина траси.

Автотрасування всієї друкованої плати починається за командою **Tools>Start Autorouter**. Інтерактивне прокладання трас

провідників починається за такими командами: **AutoRoute Connection, AutoRoute Net, AutoRoute Component, AutoRoute Area.** Ручне прокладання трас провідників здійснюється за командою **Manual Route.** Якщо вибрати команду **Tools>Sketch Route,** то можна вручну курсором приблизно прокласти трасу провідника, яка потім прокладатиметься точно в автоматичному режимі.

Після завершення трасування доцільно переглянути звіти про виконану роботу за командами **Reports>Routing Statistics, Reports>Reports.** Повернення в програму P-CAD PCB здійснюється командою **File>Save and Return.**

Загальні відомості про програму SPECCTRA

SPECCTRA — це програма для розміщення компонент і прокладання трас провідників в автоматичному та інтерактивному режимах. **SPECCTRA** використовує безсіткові алгоритми. Її основна перевага над аналогічною програмою Shape-Based Router полягає у підтримуванні складних правил проектування. Для кожного об'єкта можна задати ряд індивідуальних правил. За допомогою програми **SPECCTRA** можна редагувати розміщення компонент і трас провідників безпосередньо, без допомоги P-CAD PCB, що значно спрощує роботу. В інтерактивному режимі роздільна здатність **SPECCTRA** становить 0,01 мкм у метричній і 0,0001 мил (1 мил = 0,001 дюйма = 0,0254 мм) в англійській системі мір.

Програма **SPECCTRA** має два основні режими: режим прокладання трас провідників **(Routing)** і режим розміщення компонент на друкованій платі **(Placement).**

У режимі прокладання провідників **(Routing)** основним видом є робота програми в автоматичному режимі **AR (Auto Route),** який забезпечує: розривання (Rip-up) вже прокладених провідників і їх повторну розводку із застосуванням методу розсування і проштовхування провідників (Push and Shove); трасування провідників із застосуванням чи без застосування сіток для розміщення перехідних отворів і провідників; поліпшення технологічності виготовлення друкованої плати; згладжування прямокутних згинів провідників по діагоналі; трасування на основі набору ієрархічних правил. До складу **AR** входять дві додаткові програми, які забезпечують ручне редагування в середовищі **SPECCTRA: ER (Edit Route)** — редагування розміщення трас провідників і перехідних отворів у інтерактивному режимі та **EP (Edit Place)** — розміщення компонент. Програма **ER** забезпечує:

- ❒ переміщення одного чи кількох провідників у разі зсуву перехідного отвору (Plowing);
- ❒ розсування одним провідником кількох інших (Shoving);

- ❒ оцінка можливих результатів переміщення провідника чи перехідного отвору (Ghosting). Під час переміщення сегмента провідника чи перехідного отвору, сусідні провідники динамічно розсуваються і можна переглянути кілька варіантів та вибрати кращий;
- ❒ зображення дозволених точок розміщення перехідних отворів (Via search);
- ❒ скасування зайвих точок зламу провідників (Critic).

Під час прокладання провідників здійснюється поточний контроль дотримання допустимих зазорів між провідниками та іншими об'єктами.

Програма **EP (Edit Place)** забезпечує розміщення, зсув, поворот, вирівнювання та перестановку компонент, перенесення їх на протилежний бік друкованої плати. Переміщуючись, компонента може зсувати одну чи кілька компонент, які їй заважають. Можна вибрати компоненту з найбільшою кількістю зв'язків і вибрати найкраще місце для її розміщення. Оптимально розмістити компоненти допомагає гістограма густини зв'язків, яка наочно відображає розподіл зв'язків по площині плати. Під час розміщення компонент здійснюється контроль дотримання допустимих зазорів між об'єктами.

До складу програми **AR (Auto Route)** входять кілька додаткових програм (утиліт).

Програма **AD (Advanced)** розширює можливості настройки стратегії трасування складних друкованих плат: дозволяє прокладати на певних сигнальних шарах індивідуальні електричні зв'язки, класи та групи зв'язків; призначати різні значення ширини траси та зазорів на різних шарах плати; призначати правила трасування різним електричним з'єднанням і класам з'єднань; призначати перехідні отвори окремим класам з'єднань; підтримувати технології міжшарових перехідних отворів; розміщувати перехідні отвори під виводами планарних компонент; робити монтаж гнучкими провідниками і перемичками.

Програма **DF (Design for Manufacturability)** — програма для підвищення технологічності проекту за рахунок автоматичної генерації контрольних точок для всіх або виділених електричних з'єднань, збільшення зазорів за наявності вільного простору.

Програма **MV (Route Micro Via)** — для розміщення мініатюрних глухих перехідних отворів один над одним.

Програму **HP (High Performance)** призначено для врахування особливостей проектування високочастотних пристроїв: контроль максимальної довжини паралельних провідників, розміщених на одних і тих самих або на суміжних шарах для зменшення

перехресних наводок; контроль затримок поширення сигналу; прокладання диференціальних провідників; згладжування прямих кутів дугами; спеціальні правила розводки виділених областей; введення екранування провідників. До складу програми **HP (High Performance)** входить додаткова програма **EH (Edit High Performance)** призначена для інтерактивного редагування трас провідників високочастотних пристроїв з урахуванням обмежень на тривалість поширення сигналів: виведення на екран детальної інформації про довжину провідника з контролем обмежень на максимальну і мінімальну довжину; висвічування зеленим кольором області навколо курсора, в межах якої задовольняються обмеження на тривалість поширення сигналів.

Наведені відомості дають тільки загальне уявлення про програму **SPECCTRA**. Для роботи з програмою **SPECCTRA** слід скористатися спеціальною літературою.

14.6 Моделювання електронних пристроїв

Перш ніж виготовляти розроблений за допомогою системи автоматичного проектування P-CAD електронний пристрій, його роботу слід змоделювати на комп'ютері. Моделювання роботи розроблених електронних пристроїв на комп'ютері дає змогу усунути під час проектування помилки, дослідити поведінку пристрою в умовах дії різноманітних вхідних сигналів, визначити найважливіші параметри пристрою, знайти відхилення параметрів пристрою під дією різноманітних зовнішніх чинників тощо.

Для схемотехнічного моделювання електронних пристроїв призначена програма **Protel Design Explorer.** Принципова схема електронного пристрою, робота якого моделюватиметься, створюється за допомогою програми P-CAD Schematic. Створюючи принципову схему електронного пристрою, слід дотримуватися певних вимог:

- для моделювання використовуються спеціальні бібліотеки електронних компонент, у яких зберігається інформація про параметри цих компонент, що необхідні для моделювання: номінальні значення параметрів, температурні коефіцієнти, імена моделей тощо. Якщо модель необхідної компоненти відсутня в бібліотеці, її потрібно створити засобами P-CAD;
- крім компонентів електронного пристрою, робота якого моделюється, треба мати певні додаткові елементи: джерела

живлення, генератори вхідних сигналів, елементи заземлення, елементи, що враховують паразитні ефекти (опори котушок індуктивностей, паразитні індуктивності та ємності) тощо;

- ❒ присвоїти значення атрибутам компонентів: номінальні значення резисторів, конденсаторів, параметри джерел живлення і генераторів сигналів, імена математичних моделей;
- ❒ присвоїти імена електричним з'єднанням, на які є посилання.

Якщо в принциповій схемі є помилки, то вони заносяться в список, що зберігається в спеціальному файлі з розширенням **.ERR**.

Математична модель електронної компоненти

Для моделювання роботи електронного пристрою для кожної електронної компоненти пристрою створюється математична модель. Математична модель електронної компоненти — це сукупність фізичних параметрів, за допомогою яких адекватно відображується робота компоненти в усіх робочих режимах. Крім фізичних параметрів, кожна компонента електронного пристрою характеризується такими ***атрибутами*** як ім'я, тип тощо.

Атрибути компоненти розміщені на закладці **Attributes** (рис. 14.22) діалогового вікна **Properties**, що виникає, коли виділити потрібну компоненту і клацнути правою клавішею миші. Кожний атрибут має ім'я **(Name)** і значення **(Value)**.

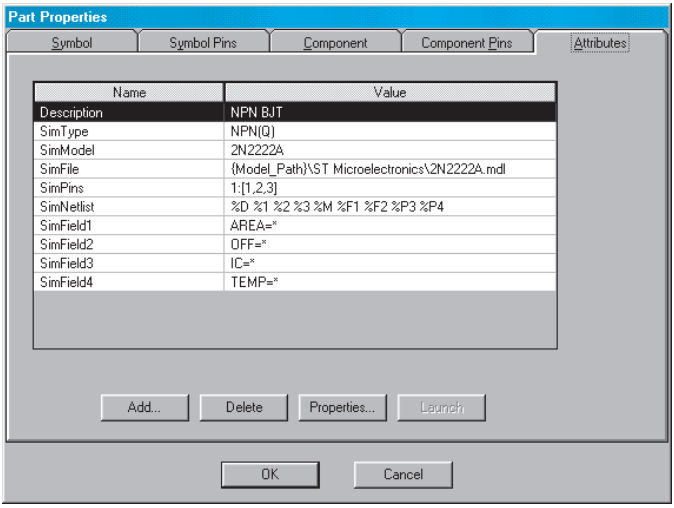


Рис. 14.22

Розглянемо найважливіші атрибути компонент.

SymType. Цей атрибут описує тип компоненти і префікс його позиційного позначення на принципових схемах, наприклад, біполярний *n-p-n* транзистор має атрибут **NPN(Q),** конденсатор — **CAP(C),** резистор — **RES(R)** тощо.

SymModel. Ім'я моделі компоненти, наприклад КТ315В — ім'я моделі біполярного транзистора.

SymFile. Ім'я файла, де зберігається математична модель компоненти.

SymPins. Список виводів усіх секцій компоненти.

SymNetlist. За допомогою цього атрибута призначається відповідність між виводами секцій компоненти і виводами електричних з'єднань.

SymDefaults. Містить ряд значень параметрів компоненти, які призначаються за замовчанням.

SymField1...SymField16. Поля **Value** цих атрибутів містять параметри компоненти, значення яких можна задавати в діалоговому вікні **Properties.**

Математична модель компоненти зберігається в окремому файлі. Файли математичних моделей компонент об'єднані в спеціальні бібліотеки.

Моделювання роботи електронного пристрою

Щоб моделювати роботу електронного пристрою, необхідно перш за все скласти завдання для моделювання. Це здійснюється командою **Simulate>Setup.** У діалоговому вікні цієї команди на закладці **General** вибирається один чи кілька видів аналізу. Засобами P-CAD можна здійснити такі види аналізу:

- **Operating Point Analysis** — аналіз в режимі постійного струму;
- **Transient/Fourier Analysis** — аналіз перехідних процесів і розрахунок частотних характеристик;
- **AC Small Signal Analysis** — аналіз роботи пристрою в режимі «малого» сигналу, тобто коли амплітуда змінної складової сигналу набагато менша постійної складової;
- **DC Sweep** — аналіз роботи в режимі постійного струму в разі зміни (варіації) одного чи двох параметрів;
- **Noise Analysis** — аналіз впливу шуму на роботу пристрою;
- **Transient Function** — визначення передаточних функцій у режимі «малого» сигналу;
- **Temperature Sweep** — аналіз впливу на роботу пристрою зміни температури;

- **Parameter Sweep** — аналіз впливу на роботу пристрою зміни одного чи двох параметрів;
- **Monte Carlo Analysis** — статистичний аналіз впливу на роботу пристрою випадкових флуктуацій параметра за методом Монте-Карло.

Кнопкою **Advanced** викликається додаткове діалогове вікно, за допомогою якого призначають параметри алгоритмів моделювання.

Завдання на аналіз перехідних процесів і спектральний аналіз встановлюється за допомогою закладки **Transient/Fourier**. Задаються такі параметри аналізу перехідних процесів: початковий і кінцевий момент моделювання, інтервал часу, максимальний інтервал часу, початкові умови. Для спектрального аналізу задаються такі параметри: частота першої гармоніки, кількість гармонік.

Під час аналізу частотних характеристик у режимі «малого» сигналу задаються на закладці **AC Small Signal** такі параметри аналізу: початкова і кінцева частоти, кількість розрахункових точок, характер зміни частоти (лінійний, логарифмічний в октавах, логарифмічний у декадах). Для здійснення такого аналізу необхідне джерело змінного струму, частоту якого змінюють у заданому діапазоні.

Вплив зміни параметра на характеристики електронного пристрою досліджують, задаючи такі параметри аналізу на закладці **Parameter Sweep**: початкові і кінцеві значення діапазону зміни вибраних параметрів, прирости параметрів на кожному кроці аналізу.

Аналогічно задають параметри і для решти видів аналізу.

Безпосередньо процес моделювання починається за командою **Simulate>Run** або після натискання на кнопку **Run Analyses** на закладці **General**.

Результати моделювання електронних пристроїв відображуються на екрані у вигляді графіків за допомогою програми **SimView**. Потенціали вузлів, струми пристроїв, які потрібно відобразити на графіках, встановлюються в полі **Collect Data For** закладки **General**.

У полі **Available Signals** наводиться перелік усіх сигналів у даному режимі, а в полі **Active Signals** задаються назви сигналів, графіки яких потрібно побудувати. На екрані можна розмістити один чи кілька графіків.

Для контролю точності моделювання на графіки наносяться контрольні точки.

Щоб зчитувати дані з графіків, на них розміщують два курсори, переміщуючи які, можна отримати їх координати у вікні **Measurement Cursors**.

Програма аналізу паразитних ефектів P-CAD Signal Integrity

На високих частотах реактивні опори, обумовлені ємністю та індуктивністю провідників, а також їх взаємними ємностями та індуктивностями, стають сумірними з активними опорами цих провідників і їх впливом вже не можна нехтувати. Це призводить до спотворення форми сигналів, що передаються за допомогою друкованих провідників. Крім того, збільшуються взаємні наводки сигнальних провідників і наводки, спричинені провідниками системи живлення.

Для аналізу таких процесів призначена програма P-CAD **Signal Integrity** (рис. 14.23), яка входить до складу системи P-CAD.

Для роботи програми аналізу **Signal Integrity** необхідно завантажити файл конструкції друкованої плати в P-CAD PCB. Перед початком сеансу роботи з P-CAD **Signal Integrity** необхідно задати параметри конфігурації програми за допомогою команди **Options>Configure**, у діалоговому вікні якої встановлюються такі параметри: система одиниць вимірювання; максимальна відстань між паралельними провідниками, для яких розраховується рівень взаємних наводок; максимальна довжина провідника; дозвіл на вибір з'єднань і перенесення їх у список з'єднань, що підлягають аналізу.

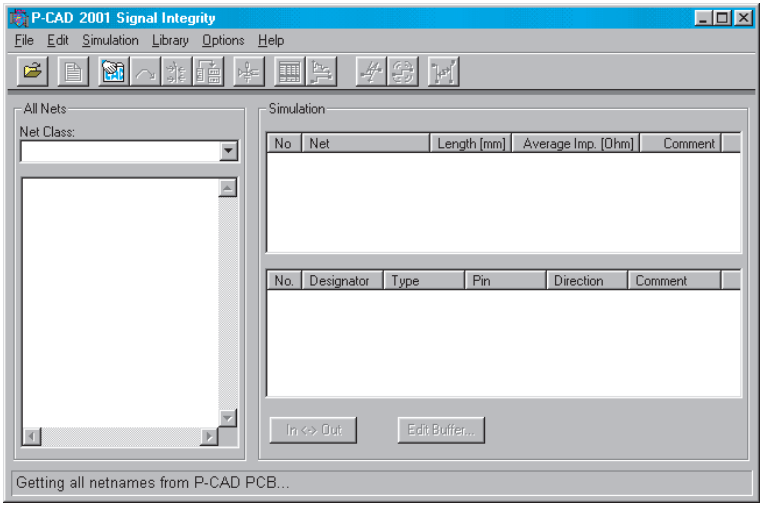


Рис. 14.23

Далі за командою **Options>Simulator** відкривається діалогове вікно, в якому задаються параметри алгоритмів розрахунку перехідних процесів на трьох закладках: **Integration** — вибір методу числового інтегрування; **Accuracy** — встановлення параметрів

точності розрахунків; **DC Analysis** — встановлення параметрів алгоритмів розрахунку за постійним струмом.

Після цього за командою **File>Get Nets** у **Signal Integrity** передається інформація про склад і конструкцію друкованої плати і вибирається одне чи кілька електричних з'єднань, для яких будуть розраховуватися перехідні процеси.

Для вибраних електричних з'єднань за командою **Edit>Take Over** у програму завантажується інформація про параметри цих з'єднань, а також про параметри шарів, у яких вони розміщені, і пристрої, до яких вони підключені. Далі, за командою **Edit> Component**, компонентам пристрою, які беруть участь в аналізі, призначається один із семи типів.

Параметри генератора імпульсного сигналу, який використовується для аналізу, встановлюються на закладці **Stimulus**, що відкривається, коли натиснути на клавішу **Edit Buffer**. Під час аналізу паразитних ефектів генератори імпульсних сигналів автоматично підключаються до всіх ліній передачі сигналів.

Після завантаження конструкції друкованої плати у програму **Signal Integrity**, на екрані відображається довжина вибраних провідників і середнє значення їх хвильових опорів. Детальніша таблиця результатів аналізу (середні, максимальні, мінімальні значення хвильових опорів провідників, викиди імпульсів на передньому і задньому фронтах, швидкості наростання і спаду імпульсів) виводиться за командою **Simulation>Screening**.

Узгодження з навантаженням вибраної лінії здійснюється за командою **Simulation>Terminator Advisor**. Користувач вибирає один із семи варіантів схеми узгодження, розраховує номінальні значення компонентів узгодження і вводить їх.

Розрахунок сигналів на виході кожного провідника із врахуванням відбивання сигналу здійснюється за командою **Simulation> Reflection**. Результати розрахунків відображаються у вигляді графіків на екрані **Waveform Analyzer**, де можна вибрати тип графіка.

За допомогою програми **Signal Integrity** здійснюється також аналіз перехресних спотворень, під час якого розраховуються сигнали із урахуванням взаємного електромагнітного зв'язку між паралельними ділянками провідників. Перед початком аналізу необхідно за командою **Simulation>Set Agressor Net** призначити ознаку «Agressor» провіднику, до якого приєднаний генератор сигналу, а за командою **Simulation>Set Victim Net** призначити ознаку «Victim» провіднику, який знаходиться під постійною напругою. Безпосередньо аналіз перехресних спотворень здійснюється

за командою **Simulation>Crosstalk**, результати якого відображуються у вигляді графіків на екрані **Waveform Analyzer**.

Текстові звіти про результати аналізу виводяться за командою **File>Report**, у діалоговому вікні якого задається їх перелік і формат.

14.7 Допоміжні програми

Крім основних операцій, за допомогою системи автоматизованого проектування можна здійснювати допоміжні операції, які поліпшують якість проекту і збільшують продуктивність праці конструкторів.

Програма внесення змін в проект ECO

Програма **ECO (Engineering Change Order)** дає змогу записувати в текстовий файл усі зміни, виконані у графічному редакторі P-CAD Schematic і передавати їх у P-CAD PCB для внесення відповідних змін у конструкцію друкованої плати (так зване пряме коригування), і навпаки, переносити зміни в конструкції друкованої плати в електричну принципову схему (зворотне коригування).

Файли програми **ECO** мають два формати: **Was/Is Format** — запис змін позиційних позначень компонент і **ECO Format** — запис усіх змін.

Внесення змін у проект у прямому і зворотному напрямку здійснюється в такій послідовності: за командою **Utils>Records ECOs** включається запис у файл усіх змін принципової схеми чи конструкції друкованої плати; потім виконується команда **Utils>Export ECOs**, за якою зміни записуються у зовнішній файл ECO. Попередній перегляд змін і їх редагування здійснюється за командою **View Pending**. Після попереднього перегляду і редагування зміни записують, натиснувши на кнопку **Save ECOs Now**. Після цього потрібно вибрати характер запису інформації у файл **ECO**. Якщо натиснути на кнопку **Append ECOs File**, то у файл запишуться поточні зміни, а попередні зміни позначаться як коментарі. Натискання на кнопку **Discard ECOs** стирає попередньо записану інформацію і на її місце записує нову; далі з редактора принципових схем у редактор конструкції друкованої плати (або навпаки) завантажують проект, у який необхідно внести зміни, і виконують команду читання файла змін **Utils>Import ECOs**. У меню цієї команди вибирають потрібний файл змін і вносять його у проект натисканням на кнопку **OK**.

Програма розміщення компонент і задавання правил розробки конструкцій друкованих плат InterPlace&Parametric Constraint Solver (InterPlace&PCS)

Програма InterPlace&PCS призначена для задавання правил проектування на етапах створення принципової електричної схеми і конструкції друкованої плати пристрою.

Для роботи програми InterPlace&PCS необхідно відкрити проект у редакторі P-CAD Schematic або P-CAD PCB, після чого викликається програма InterPlace&PCS командою **Utils>>InterPlace/PCS** (рис. 14.24) і виконується команда **File>Update Schematic** або **File>Update PCB.** Після виконання цих команд на екрані розміщуватиметься три вікна: вікно **Layout** у середній частині екрана, вікно **Design Manager** справа і вікно **Constraint Editor** внизу екрана.

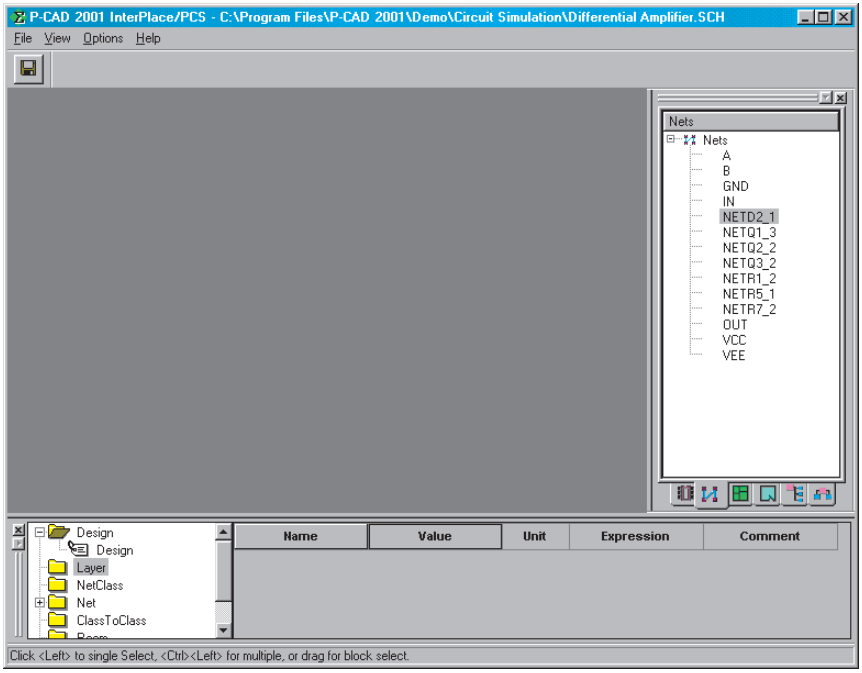


Рис. 14.24

Програма Parametric Constraint Solver(PCS) дає змогу задати ряд правил розміщення компонент, прокладання трас провідників та інших правил розробки проекту на етапах створення принципової електричної схеми та ранніх етапах розробки конструкції друкованої плати. Крім задання правил розробки у вигляді

сукупності обмежень (констант), допускається також використовувати математичні функції, в яких є посилання на константи. Всі обмеження відображуються у вікні **Constraint Editor** послідовно, за рівнями ієрархії. У правій частині вікна **Constraint Editor** наводиться список обмежень для об'єктів, вибраних у лівій частині вікна. Щоб задати нове обмеження, необхідно клацнути правою клавішею миші в діалоговому вікні, що з'явилося, вибрати з меню категорію обмеження, тип обмеження і одиниці вимірювання. Якщо є файл технологічних обмежень **Design Technology Parameters**, то він може бути завантажений командою **File>Design Technology Parameters**, і обмеження, які він містить, будуть об'єднані з обмеженнями, введеними за допомогою програми **PCS**.

Інтерактивне розміщення компонент виконується у вікні **Layout** за допомогою команд меню **Edit**, більшість яких дублюється піктограмами або вибирається з меню, що виникає, якщо клацнути правою клавішею миші: вирівнювання компонент; фіксація вибраних компонент, скасування фіксації, висвічування вибраних компонент, скасування висвічування, висвічування приєднаних до компоненти провідників, вибір місця точки «прив'язки», поворот компоненти на фіксовані кути.

За командою **Edit>Discrete Placement** здійснюється розміщення вибраної дискретної компоненти. Тип корпусу дискретної компоненти вказується в діалоговому вікні цієї команди.

1. Що таке система автоматизованого проектування?
2. Схарактеризуйте систему автоматизованого проектування електронних пристроїв P-CAD?
3. Як створюються принципові електричні схеми пристроїв за допомогою P-CAD Schematic?
4. Яким чином перевіряються принципові електричні схеми, що створені за допомогою P-CAD Schematic, на наявність помилок?
5. Схарактеризуйте графічний редактор проектів друкованих P-CAD PCB?
6. Що таке бібліотека компонент і як нею користуватися?
7. Як здійснюється автоматизоване прокладання трас провідників засобами системи P-CAD?
8. Які функції виконує система SPECCTRA?

15

Дистанційне навчання

15.1 Історія виникнення

Першим винаходом, що дозволив зрушити навчання в часі і просторі, була книга, яка використовується як засіб навчання і по сьогоднішній день. Книга дозволила зробити навчання індивідуальним і поклала початок дистанціюванню учня від учителя. Книга — головне джерело навчальної інформації, проте необхідний ще один обов'язковий компонент навчання — комунікація між вчителем і учнем. За допомогою друкованих матеріалів між студентом і викладачем організована комунікативна взаємодія.

Більше ста років люди в різних частинах світу беруть участь у процесі навчання й одержують освіту через систему заочного навчання. У цій системі студент одержує друковані матеріали, письмово виконує завдання, відсилає їх викладачу й одержує від нього зворотний зв'язок у вигляді рецензії, зауважень або оцінки. Таке навчання, хоч і дуже поширене в усьому світі, має суттєві обмеження для формування фахових навичок і вмінь, оскільки взаємодія з викладачем, що веде навчання, обмежена тільки письмовими повідомленнями. Враховуючи затримку в одержанні відповіді від викладача, ефективність зворотного зв'язку (керування навчанням і якістю засвоєння) виявляється недостатньою.

Студент-заочник практично не має умов для активної навчальної діяльності, тому що майже всю навчальну роботу він виконує, користуючись підручником, ручкою і зошитом. Головний недолік заочного навчання — відсутність усного живого спілкування, що лише частково компенсується на двох у рік короткотермінових передсесійних настановних сесіях. Доступ студентів до навчальних ресурсів вузу (навчального відео, комп'ютерних курсів і навіть бібліотечного фонду) також обмежений. Система заочної освіти сьогодні повинна шукати нові можливості ефективної професійної підготовки.

Ілюстрацією організації заочного навчання може послужити цитата з книги Дж. Вінцента «Рух Чатауква», виданої в Бостоні в 1886 р. і присвяченої заочному курсу латини, в якій викладаються основи заочного навчання, що не втратили свого значення і понині:

«Щотижня студент отримує поштою інструкцію. В цій інструкції:

а) формулюються завдання, які він повинен виконати, наприклад, які частини тексту перекласти, які розділи граматики вивчити;

б) вказується порядок виконання робіт, якого учень повинен дотримуватися;

в) даються пояснення до тих місць уроку, які учень може не зрозуміти;

г) пропонується допомога в тих випадках, коли це вважається доцільним;

д) виділяється певний матеріал для повторення;

е) дається контрольна робота, яку учень повинен виконати після вивчення матеріалу уроку.

Інакше кажучи, інструкція призначена для того, щоб спрямувати учня і надавати йому допомогу так само, як це робить вчитель у класі.

Так задається і вивчається кожний урок, і результати навчання надаються вчителеві для корекції, критики і рекомендацій.

З вищевикладеного випливає, що вчитель заочного навчання повинен бути старанним, терплячим, доброзичливим і «живим», а студент-заочник повинен бути чесним, честолюбним, сприйнятливим і таким же «живим». Чого б «мертвий» учитель не досяг у класі, він нічого не зможе зробити на відстані, але якщо студент на відстані і не має перерахованих вище якостей, то в заочному навчанні трапиться одне з двох: або він виробить у собі ці якості і досягне успіху, або він залишиться там, де він був на самому початку, і зазнає невдачі».

Як бачимо, структура заочного навчання була дуже чітко визначена ще більше 100 років тому, при цьому роль, функції й обов'язки як учителя, так і студента, описані в цій цитаті, зберігають своє значення і в наш час. У США, наприклад, існує заочне навчання, так зване «External Degree», при якому в навчальному закладі виконується не більше 25 процентів навчальної роботи. У нашій країні заочне навчання існує з давніх пір і потребує присутності студента на двох настановних сесіях загальною тривалістю до двох місяців із десяти місяців навчання.

Сучасна історія дистанційного навчання або другий етап розвитку заочного навчання почався із відкриттям у Великобританії Відкритого університету в 1969 році, хоча перші спроби запровадити нові технологічні ідеї були початі кількома роками раніше, коли в 1961 році Британський уряд прийняв рішення відкрити

«Університет в Ефірі» (University of the Air). Незабаром, у ФРГ відкрився «Телеуніверситет» (Fernsehuniversitat). У цих навчальних закладах навчальні програми транслювалися по радіо і телебаченню. В СРСР трансляція навчальних ТВ програм з різних предметів почалася в 1966 році, при цьому такі програми призначалися переважно як допомога саме студентам-заочникам.

Технічний відтінок у назві цих університетів ознаменував перехід до нової форми дистанційного навчання на основі сучасних технологій. Звернення до технології було викликано потребою, з одного боку, у масовому навчанні широких прошарків суспільства, з іншого боку, прагненням до розвитку й удосконалення освіти в цілому.

15.2 Основні поняття

Одним із досягнень сучасних комп'ютерних технологій є можливість навчати без фізичної присутності вчителя: урок може бути зсунутий у часі (не за розкладом, а в будь-який інший час) і за місцезнаходженням (не в аудиторії, а в будь-якому іншому місці).

У найзагальнішому вигляді ***дистанційне навчання*** — це навчання, в якому учень і вчитель фізично відділені один від одного. При цьому важливо розмежувати поняття **дистанційна освіта**, **дистанційне навчання** і **дистанційне вчення**.

Дистанційна освіта — повноцінна освіта, наприклад, вища, надана учням на відстані засобами телекомунікаційних технологій.

Дистанційне навчання передбачає поширення навчальних програм за допомогою деяких технічних засобів, у той час як **дистанційне вчення** має на меті надати будь-якій людині, що потребує одержання якоїсь освіти або професійної підготовки в тій чи іншій області, всіх необхідних умов, матеріалів, засобів і можливостей для досягнення поставленої мети.

В дистанційній освіті виділяються такі елементи:

- ❒ фізична віддаленість учня від учителя;
- ❒ вплив навчального закладу, що відрізняє дистанційне навчання від самонавчання;
- ❒ використання технічних засобів, насамперед телекомунікаційних, для навчально-інформаційного забезпечення навчального процесу і підтримки специфічної комунікації учня з учителем і навколишнім світом з метою формування фахової компетенції студента вузу;

- ❒ забезпечення двосторонньої комунікації з тим, щоб студент отримав користь від спілкування (із викладачем) і навіть міг ініціювати діалог;
- ❒ можливість періодичних зустрічей як з педагогічною, так і соціальною метою.

Дистанційне навчання — це автоматизований процес, головним засобом підтримки якого є комп'ютер. Керування процесом навчання здійснюється на основі науково розроблених моделей, навчального процесу учня. Навчання при цьому більш централізоване, ніж у традиційних системах навчання. Дистанційне навчання — масове навчання для широкого кола користувачів.

Дистанційне навчання — специфічна дидактична система, що надає будь-якому користувачу різноманітні освітні послуги (в тому числі систематичну освіту, включаючи вищу) на відстані без відвідування навчального закладу в зручний для нього час на основі сучасної інформаційно-комп'ютерної і телекомунікаційної технології. Ця система дає можливість одержання загальноосвітньої і спеціальної підготовки будь-якому користувачеві за допомогою спеціально організованого навчання й окремих курсів, які учень вивчає самостійно або під керуванням учителя. Система відкриває доступ учневі до розподілених навчальних ресурсів, забезпечує необхідний ступінь керування навчанням, зворотний зв'язок, контроль і методичну підтримку, а також дозволяє учневі вести активне спілкування з викладачем та іншими учнями. Дистанційне навчання не є обов'язковою формою навчання з регулярним відвідуванням занять у навчальному закладі, у ньому добровільно беруть участь тільки ті учні, які розраховують отримати в шуканий результат і в змозі це зробити.

Причини, які спонукають людину вчитися в системі дистанційного навчання:

- ❒ особи, що бажають одержати освіту в якійсь області знань або професійну підготовку з певної спеціальності, але з деяких причин не можуть регулярно відвідувати заняття (або навчальний заклад знаходиться далеко від місця їхнього проживання, або вони не можуть залишити роботу або сім'ю, чи за станом здоров'я, чи з інвалідності не можуть відвідувати навчальний заклад);
- ❒ людина хоче вчитися по-своєму: самостійно обираючи предмет, курс, навчальні матеріали, методику навчання, темп і час занять;
- ❒ учень, студент і фахівець мають потребу в доступі до численних і різноманітних джерел навчальної інформації: бібліотек, банків даних, а також викладачів у тій області

знань, у якій він не може одержати підготовку або консультацію за місцем проживання.

Дистанційне навчання забезпечує **масове навчання**, при цьому ціль досягається через **індивідуальне навчання** в рамках масового. Збільшеня кількості учнів не позначається на індивідуальній роботі викладача з кожним окремим учнем.

Дистанційна освіта відрізняється від заочної наявністю спеціально розробленого комплексу дидактичних засобів (книги, аудіо-посібники, візуальні посібники, аудіовізуальні і комп'ютерні посібники). Студенти мають доступ безпосередньо до викладача для одержання консультацій і до електронної навчально-інформаційної бази навчального закладу. Завдяки цьому студент контактує з викладачем не два рази на рік, як у заочному навчанні, а коли це необхідно, у зв'язку з чим ефект відірваності від навчального закладу усувається й ефективність його самостійної навчальної діяльності різко зростає.

Зворотний зв'язок у дистанційному навчанні існує у двох формах: автоматизована, здійснювана програмою автоматизованого навчального контролю; «жива», що встановлюється з викладачем або консультантом опосередковано, через телекомунікаційні засоби. У силу останнього існує два види дистанційного навчання:

- **пасивне**, при якому навчальний матеріал і дидактичні впливи здійснюються через навчальні посібники, фонограми, слайди, кіно і відеокурси, радіопередачі і електронну пошту;
- **активне**, в якому той, хто навчається, взаємодіє з навчальним центром або особисто викладачем безпосередньо — через телекомунікаційні мережі і комп'ютер, або опосередковано, вивчаючи різноманітні навчальні матеріали: відео, інтерактивні, автоматизовані й інші курси, які реалізуються за допомогою засобів мультимедіа і надають студентові можливість активної взаємодії з навчальною інформацією.

Другий вид дистанційного навчання має великі переваги в порівнянні з першим завдяки інтерактивній формі взаємодії між тим, хто навчається, і тим, хто вчить, і можливості контролю діяльності того, хто навчається, з боку викладача.

В останні роки все більшого поширення набули чотири види дистанційного навчання, основані на таких телекомунікаційних засобах:

- інтерактивному телебаченні;
- комп'ютерних телекомунікаційних мережах, у тому числі в інтерактивному режимі;

- ❒ комп'ютерних відеоконференціях;
- ❒ поєднанні першого і другого.

В останні роки розвиток дистанційної освіти призвів до появи нової форми — **віртуальної.** Він виник на основі розвитку технологій мультимедіа та її переростання в систему **віртуальної реальності.** Знаходячись у штучному світі, створеному комп'ютерними і мультимедійними засобами, у людини складається враження повної заглибленості в цей світ, де можна змінювати свій образ, навколишнє середовище, змінювати масштаби, переміщати свою точку огляду або самому переміщатися в просторі і часі. За допомогою цієї системи моделюється й середовище, яке надає людині можливість взаємодії з будь-яким об'єктом цього середовища. Система віртуальної реальності відкриває практично безмежні можливості для навчання.

Віртуальна освіта являє собою високоякісну, ефективну і доступну освіту, здійснювану в середовищі дистанційного навчання за допомогою телекомунікаційних і електронних засобів доставки. Серед віртуальних навчальних закладів найбільш поширені **віртуальні університети** — навчальні заклади нового, відкритого типу, що надають широкий спектр найсучасніших освітніх послуг усім бажаючим у будь-якій області знань, на відстані, без обов'язкового відвідування занять.

15.3 Інформаційно-освітнє середовище дистанційного навчання

Методологічним фундаментом, який об'єднує «дистанційного учня» з «дистанційним учителем», є мережа, що перекидає міст над простором між учителем і учнем. **Дистанційне навчання** розглядається як система і процес підключення учнів до віддалених розподілених навчальних ресурсів.

З технологічної точки зору дистанційне навчання є системою, яка дозволяє здійснювати підключення голосу, відео, графіки та даних між віддаленими учнями і вчителями, віртуальними бібліотеками, організаціями, предметними експертами, Інтернетом і множиною провайдерів освітніх матеріалів (рис. 15.1). У центрі системи знаходиться учень, підключений до вищезазначених ресурсів у режимі реального часу або в асинхронному режимі. Мережа дистанційного середовища може зробити навчання доступним, зручним для учня і освітніх провайдерів.

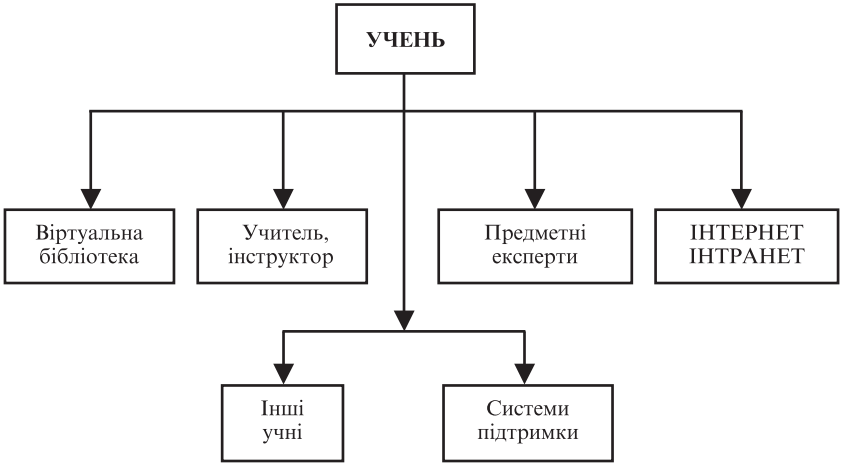


Рис. 15.1

Асинхронні комунікаційні технології (електронна пошта, телеконференції, аудіо- і відеокасети, електронна дошка, бібліотеки мультимедійних баз даних, Інтернет) підтримують взаємодію в розподіленому режимі і забезпечують доступ до інформаційного ресурсу з місця учня.

В розділі 10 ми розглядали, що таке ***електронна пошта***. Вона забезпечує передачу й обмін інформацією між великою кількістю учнів, зберігає інформацію в пам'яті центрального комп'ютера або електронної поштової скриньки до запитання, перенесення інформації з мережі до комп'ютера і навпаки, підключення до електронних банків і баз даних, демонстрацію текстової і графічної інформації, її інтерпретацію, отримання повідомлень про отримання інформації або її повернення, багато іншого.

Телеконференції мають велике поширення і включають аудіо-, відео- і комп'ютерні конференції. Аудіо і відеоконференції є складовою частиною синхронних комутаційних технологій, а комп'ютерні конференції — асинхронні. Вони дозволяють здійснювати підготовку, прийом та передачу інформації (текстової, графічної, звукової) через мережу будь-якому користувачу, що бере участь у конференції. Всі повідомлення нумеруються, систематизуються з метою полегшення доступу до них. При цьому створюються додаткові підтеми, які цікавлять окремих учасників конференції.

Аудіо- і відеокасети на CD. Ємність CD-ROM в 500 разів більша ємності дискет, вони зручніші для доставки та зберігання

аудіо- і відеоінформації. Сучасні DVD забезпечують перегляд відеозображення протягом 3 годин, мають високу якість, пошукові можливості, мультимедійні доріжки.

Електронна дошка оголошень має таке значення: розміщення і зберігання повідомлення; пошуку користувачем інформації; пошуку партнерів і багато іншого.

Синхронні комунікаційні технології (настільні відеоконференції, інтерактивні групові аудіо- і відеоконференції) забезпечують взаємодію в реальному часі між учнями і вчителем (інструктором). Інструктори, предметні експерти і студенти можуть бачити і слухати один одного з усіх місць та втягуються в діалог, як у традиційній аудиторії. В основі цієї технології лежить телекомунікаційний зв'язок «комп'ютер–комп'ютер», який дозволяє здійснити підготовку, редагування, обробку, зберігання, передачу і прийом інформації, перенесення її з мережі на магнітний носій або навпаки.

За допомогою телефону, аудіо-, комп'ютерних і двосторонніх відеотехнологій навчальними програмами можуть користуватися стільки учнів і в стількох містах, скільки потрібно. Користувачі виграють від доступу до різноманітних допоміжних матеріалів за рахунок можливості контактів з предметними експертами, проведення диспутів з іншими студентами (учнями), можливості організації електронних дискусійних груп або дошки оголошень, Інтернет–ресурсів, а також інших доступних освітніх послуг.

Дистанційне навчання сьогодні використовує **широкополосні мультимедійні мережі**, розташовані по всьому світі. Мережі дистанційного навчання можуть використовуватися для всіх форм навчання: поліпшувати засвоювання програми, підсилити ефект від навчання, а також забезпечити учнів найсучаснішими навчальними матеріалами в будь-якому місці і в будь-який час. Вони можуть мати багато форм і розмірів — від Інтернету до супутникових систем і магістральних оптичних інфраструктур. При цьому важливим є спроможність мережі працювати в різноманітному оточені. Це пов'язано з великими географічними відстанями та охопленням мереж зі змішаними технологічними структурами, які відрізняються можливостями.

Одна із характеристик дистанційного навчання — комунікативний характер будь-якої діяльності учнів у системі. Учень вступає в мовну взаємодію з різноманітними категоріями інших користувачів, вчителем і джерелами інформації і здійснює в процесі навчання обмін інформацією. Ця характеристика має особливе значення для вивчення, де комунікація — мета і засіб навчання.

У системі дистанційного навчання можливі такі варіанти комунікативного спілкування:

1. Взаємодія з комунікативними партнерами, тобто природний людський діалог у писемній формі. Такий діалог відбувається між учнем і: вчителем, інструктором, які здійснюють керівництво навчанням; предметним експертом; співучнями з курсу.

2. Взаємодія із системою дистанційного навчання, що є людино-машинним діалогом, тобто обмін інформацією з комп'ютером на основі зворотного зв'язку. Діалог ведеться між учнем і: автоматизованим навчальним курсом; довідково-інформаційною системою; різноманітними банками даних (наприклад, бібліотеками).

Комунікативна діяльність учня в системі дистанційного навчання може бути наведена у вигляді схеми (рис. 15.2).

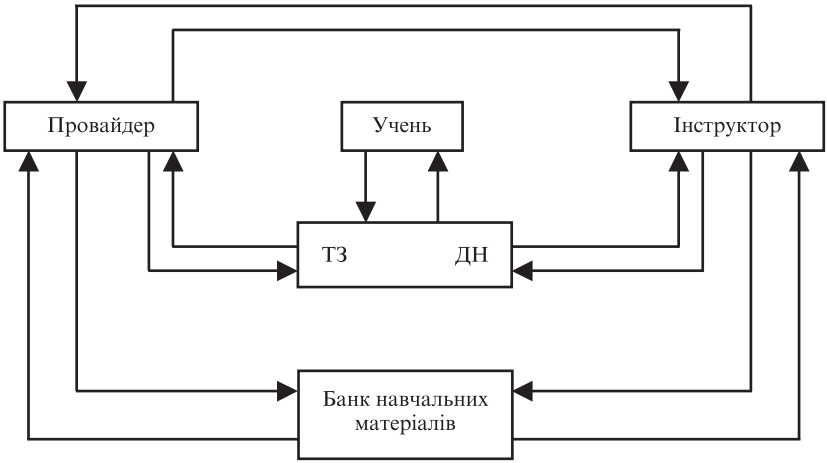


Рис. 15.2

Як випливає з рисунку 15.2, у навчальній комунікації в середовищі дистанційного навчання виділяються такі 3 види спілкування: навчальний (студент — вчитель/автоматизований навчальний курс); інформаційно-довідковий (студент — експерт/довідково-інформаційна система і банк даних); розмовний (студент – однокурсник).

У дистанційному навчанні:

- ❒ два типи комунікації — природна і штучна (людина-машина);
- ❒ дві форми комунікації — індивідуальна (діалог) і колективна (конференція);

- ❒ дві категорії комунікативних партнерів — природні (вчитель, експерт, однокурсник, носій мови) і штучні (автоматизований навчальний курс, банк даних);
- ❒ два режими комунікації — в реальному часі і відстрочений;
- ❒ два види доступу до спілкування — миттєвий та із затримкою;
- ❒ два способи опрацювання інформації — індивідуальний і пакетний.

1. Які особливості має заочна форма навчання?
2. Що таке дистанційна освіта і дистанційне навчання?
3. Чим характерна віртуальна освіта?
4. Яка схема взаємодії студента із компонентами дистанційної освіти?
5. Що таке асинхронні і синхронні дистанційні технології?

Зміст

Навчальне видання

ГУРЖІЙ Андрій Миколайович
ПОВОРОЗНЮК Назар Іванович
САМСОНОВ Валерій Васильович

Інформатика та інформаційні технології

Підручник для учнів
професійно-технічних закладів

Редактор *Попко О. Г.*
Комп'ютерна верстка *Горб В. С.*
Художнє оформлення *Харченко М. О.*

Підписано до друку 26.08.03. Формат 60 × 90/16. Друк ризографічний.
Папір офсетний. Гарнитура SchoolBook СТТ. Умов. друк. арк. 22,0.
Тираж 100 прим. Зам. № 72.

ТОВ «Компанія СМІТ».
61166, м. Харків, просп. Леніна, 14.
Тел.: 8-(057)-702-08-16
Факс: 8-(057)-702-13-07
E-mail: book@smit.com.ua
http://www.smit-book.com

Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру
видавців, виготівників і розповсюджувачів видавничої продукції
ДК № 435 від 26.04.2001

Друк ФОП «Васильєва Н. В.»
61166, м. Харків, просп. Леніна, 14.
Тел./факс: 8-(057)-702-13-07.