ББК 32.965
Л 73

Рекомендовано Міністерством освіти і науки України як навчальний посібник для студентів вищих навчальних закладів
(Лист № 14/18.2—1702 від 18.09.2002 р.)

У навчальному посібнику розглянуто різноманітні аспекти забезпечення надійності обчислювальних пристроїв, персональних комп'ютерів (ПК) і комп'ютерних систем, методи контролю цифрових пристроїв. Особливу увагу приділено методам і засобам діагностування ПК, програмного забезпечення, від умілого використання яких значною мірою залежить надійність їх функціювання. Також висвітлено питання модернізації та експлуатаційного обслуговування ПК.

Адресований студентам вищих технічних навчальних закладів, спеціалістам, які займаються експлуатацією комп'ютерних систем і мереж та використовують при вирішенні виробничих і управлінських завдань комп'ютерні технології.

Автори:

доктор технічних наук, професор Ю. Г. Савченко — 1 і 2 розділи;
доктор технічних наук, професор В. М. Локазюк — 3,4,5,6 розділи

Рецензенти:

член-кореспондент НАНУ, доктор технічних наук,
професор *В. В. Грицик*;
доктор технічних наук, професор *Г. О. Козлик*;
доктор технічних наук, професор *М. А. Філінюк*

ISBN 966-580-168-6

Зміст

Список скорочень

АТС — автоматична телефонна станція
БПС — багатопроцесорні системи
БСА — багатоходовий сигнатурний аналізатор (велика інтегральна схема)
ВІС — інтегральна схема високого ступеня інтеграції
ЗП — запам'ятовуючий пристрій
ІС — інтегральна схема
КЕОМ — керуюча ЕОМ
КПСІ — компоненти підвищеного ступеня інтеграції
КС — комп'ютерна система
ЛКМ — локальна комп'ютерна мережа
МП — мікропроцесор
МПП — мікропроцесорний пристрій
МПП і С — мікропроцесорні пристрої і системи
НВІС — інтегральна схема надвисокого ступеня інтеграції (надвелика інтегральна схема)
НМД — накопичувач на магнітних дисках
ОД — об'єкт діагностування
ОЗП — оперативний запам'ятовуючий пристрій
ОК — об'єкт контролю
ОП — обчислювальний пристрій
ПЗ — програмне забезпечення
ПКПСІ — пристрій з компонентами підвищеного ступеня інтеграції
ПЛМ — програмовані логічні матриці
ПС — пристрій спрягання
СКД — система комбінованого діагностування
СПД — система покомпонентного діагностування
ССД — система структурного діагностування
ТЕЗ — типовий елемент заміни
ТІ — таймер інтервальний
Ц і МПП — цифрові і мікропроцесорні пристрої
ШІ — штучний інтелект

1.

Надійність обчислювальних пристроїв, ПК і комп'ютерних систем

Обчислювальні пристрої, ПК і комп'ютерні системи є важливою складовою сучасного виробництва. Нині ця галузь розвивається особливо інтенсивно. Однак розвиток засобів комп'ютерної техніки неможливий без забезпечення їх надійності. Її, у свою чергу, характеризують багато показників і параметрів.

1.1. Суть і основні елементи теорії надійності

Однією з особливостей комп'ютерних систем і мереж порівняно з іншими, наприклад механічними, є те, що їх надійність визначає не тільки надійність елементів і апаратури, а й надійність програмного забезпечення. Для одержання правильних результатів обчислень необхідно, щоб усі згадані елементи мали необхідну надійність.

Надійність *— властивість об'єкта зберігати в часі у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах та умовах застосування, технічного обслуговування, зберігання і транспортування.*

У мережу увімкнутий будь-який простий прилад, приміром лампочка розжарення. Він є досить зручним об'єктом, хоча б тому що легко визначити, коли втрачається його працездатність. Нехай лампочка перегорить через 500 год, а інша буде світити 1000 год. Інтуїтивно можна стверджувати, що інша лампочка надійніша. Проте цей висновок можливий лише тоді, коли отримана інформація вже не має сенсу, бо далі використовувати лампочки не можна. З іншого боку, якщо ці прилади різних виробників, то з'являються деякі підстави вважати, що лампочки другого виробника надійніші принаймні за ознакою тривалості роботи. Однак і ці твердження необґрунтовані, бо результат експерименту може бути випадковим, через те, що лампочка першого виробника мала, наприклад, дефект, який не був помічений відділом технічного контролю (ВТК).

Якщо ускладнити експеримент і взяти не по одній лампочці, а $N = 100$ шт. одного виробника і стільки ж іншого, увімкнути їх у мережу, то через певні проміжки часу Δt можна зафіксувати кількість лампочок $N(t)$, що вже перегоріли на момент часу t. Це відображатиме приблизно така залежність, яка подана на рис. 1.1.

Криві залежності можуть бути побудовані інакше, однак спільними для них завжди є такі властивості:

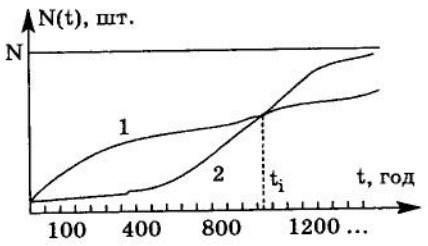


Рис. 1.1. Залежність кількості лампочок, що вийшли з ладу, від тривалості експлуатації

— криві починаються в точці з нульовими координатами, бо експеримент проводять зі справними об'єктами, тобто $N(0) = 0$;

— кожна крива не спадає, тому що з часом кількість несправних об'єктів не зменшується, тобто $dN(t)/dt \geq 0$;

— при $t \to \infty$ кожна крива асимтотично наближається до значення $N(t) = N$ (усі об'єкти вийдуть з ладу, а лампочки перегорять).

Криві, отримані в результаті експерименту або на основі оброблення статистичних даних про кількість відмов, називають *функціями ненадійності*. Якщо експеримент проведено з великою кількістю випадково вибраних виробів, то функція ненадійності містить вичерпну інформацію про очікувану поведінку об'єктів протягом

усього життєвого циклу. Отже, аналізуючи криві 1 і 2 (див. рис. 1.1), які відповідають об'єктам 1-ї і 2-ї групи, можна зробити висновок, що до моменту t_i лампочки першого виробника (крива 1) виявляються менш надійними, бо кількість об'єктів, що вийшли з ладу за весь період і в кожний момент часу, більша, ніж для об'єктів другого виробника (крива 2). Після моменту t_i, навпаки, крива 2 іде вище кривої 1, тобто для тривалішої роботи виявляються кращими лампочки другого виробника. Отже, надійність залежить від часу, протягом якого повинен працювати прилад. Саме тому не можна віддати безперечну перевагу першому чи другому виробникові — усе залежить від того, протягом якого часу буде експлуатуватися об'єкт і який час він уже пропрацював до моменту, з якого передбачається початок його експлуатації.

Лампочки яскраво ілюструють цей експеримент. Це — традиційні електротехнічні прилади. Є достовірна статистика про їх поведінку за тривалий час експлуатації. До того ж провести випробування лампочок на надійність неважко, оскільки це прості й дешеві вироби, перегорають вони часто і вихід їх з ладу можна зафіксувати. Проте є об'єкти більш складні й дорогі.

Якщо середня тривалість безперервної роботи звичайної лампочки становить 500—1000 год, то такі об'єкти, як транзистори, інтегральні схеми (ІС), навіть великого рівня інтеграції, працюють десятки і сотні тисяч годин. А це значить, що експеримент, про який йшлося, може тривати кілька років. З'являться нові компоненти, виготовлені за іншою технологією, і отримана в результаті експерименту інформація перестане бути актуальною. Така ситуація характерна не тільки для окремих компонентів, а й для багатьох електронних приладів (комп'ютерів, телевізорів, мобільних телефонів тощо). Можна стверджувати, що саме висока надійність зазначених об'єктів є перешкодою для її визначення, тому що сучасні електронні компоненти не дуже часто відмовляють, а самі відмови трапляються рідко.

Протилежною властивістю до надійності є ненадійність. Якщо знайти залежність ненадійності від часу для найпоширеніших електронних приладів і отримати її аналітичний вигляд, то з'явиться можливість прогнозувати надійність та проводити відповідні розрахунки для визначених приладів.

Перед тим як безпосередньо перейти до аналізу такої залежності, необхідно дати характеристику поняттю «відмова».

Відмова — подія, яка полягає в утраті функційним модулем (системою) здатності виконувати потрібну функцію.

Відносна кількість об'єктів $N(t)/N$, які втратили працездатність на момент часу t, є не що інше, як ймовірність відмови $Q(t)$ за час t. Це типове тлумачення при переході від статистичного розгляду масових подій до їх імовірнісного представлення. Величина:

$$P(t) = 1 - Q(t)$$

є імовірністю безвідмовної роботи об'єкта за час t. Залежність $P(t)$ є зворотною відносно $Q(t)$ (рис. 1.2). Конкретний характер залежності $Q(t)$ і відповідно $P(t)$ невідомий. Відомі лише кінцеві точки відповідної кривої: $P(0) = 1$ і $P(\infty) = 0$ та основна властивість — $Q(t)$ з часом не спадає $(dQ(T)/dt \geq 0)$ і $P(t)$ — не зростає $(dQ(T)/dt \leq 0)$.

***Ймовірність безвідмовної роботи P(t)** — функція розподілу ймовірностей безвідмовної роботи об'єкта, яка характеризує ймовірність того, що об'єкт за час t не втратить працездатності.*

Проте ця функція не дуже зручна для практичного використання. Адже споживача здебільшого цікавить не ймовірність безвідмовної роботи пристрою від початку експлуатації, а лише ця ймовірність на певному проміжку часу експлуатації, наприклад з моменту t_1 до моменту t_2.

Позначивши ймовірність $P(t_1,t_2)$, можна розглянути ситуацію, коли об'єкт у момент t_1 ще працездатний. Ймовірність такої події $P(t_1)$, а що об'єкт буде працездатним проміжок часу від t_1 до t_2 — $P(t_1,t_2)$. Тому ймовірність того, що об'єкт працездатний в момент t_2, можна записати як добуток:

$$P(t_2) = P(t_1)P(t_1,t_2),$$

звідси виводять формулу для обчислення ймовірності безвідмовної роботи об'єкта за довільний проміжок часу $t_1 \ldots t_2$:

$$P(t_1,t_2) = \frac{P(t_2)}{P(t_1)}. \quad (1.1)$$

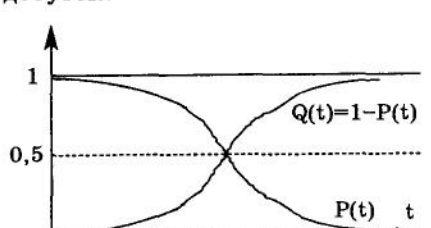


Рис. 1.2. Графік залежності ймовірності безвідмовної роботи об'єкта від часу

Ймовірність безвідмовної роботи є одним із основних показників надійності технічних виробів, але він не завжди зручний для розрахунків, тому що значення $P(t_1,t_2)$ за відносно короткі проміжки часу (година, доба) для сучасних електронних приладів близьке до 1. Наприклад, ймовірність безвідмовної роботи типової ІС за кілька годин роботи становить 0,999999...0,9999999. Проводити розрахунки з такими числами незручно. Тому доцільно перейти до відповідних ймовірностей відмови — $10^{-7}...10^{-6}$.

По суті, функція розподілу є інтегральною («накопичувальною») характеристикою надійності, тому що вона враховує всю попередню поведінку об'єкта з точки зору його працездатності.

Проте здебільшого зацікавленість викликає те, яке значення має надійність у певний момент. З огляду на це зручніше використовувати густину функції розподілу $dP(t)/dt$, що має фізичний зміст частоти відмов і розмірність 1/год:

$$f(t) = \frac{dQ(t)}{dt} = -\frac{dP(t)}{dt}.$$

Поведінка функції показує, як часто відбуваються втрати працездатності протягом життєвого циклу об'єкта. Величина $f(t)$ дає наочну картину зміни надійності об'єкта. Наприклад, для функції розподілу (рис. 1.3, а) густина розподілу зменшується так, як це показано на рис. 1.3, б.

В інтервалі $t_1 \ldots t_2$ частота відмов об'єкта максимальна, але далі зменшується. Така поведінка $f(t)$ (зменшення частоти відмов на кінцевому інтервалі життєвого циклу) не відповідає фізичній природі старіння об'єкта за тривалої експлуатації.

Глибша причина такого ефекту криється в самому експерименті. При випробовуванні фіксованої кількості об'єктів N з часом кількість об'єктів, які зали-

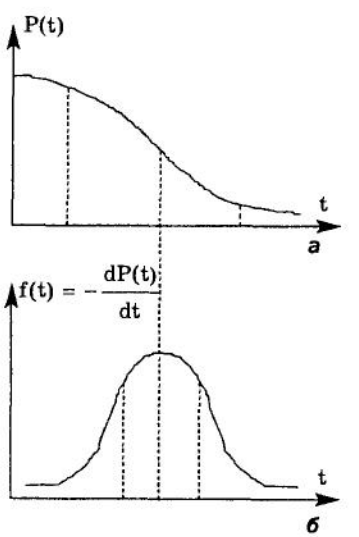


Рис. 1.3. Вигляд функцій розподілу та густини розподілу відмов об'єкта

шилися працездатними, поступово зменшувалася і відповідно зменшувалася частота відмов $f(t)$. Щоб усунути такий небажаний ефект, штучно збільшують частоту відмов. Математично це роблять так: ділять $f(t)$ на $P(t)$ і вводять новий показник надійності — *інтенсивність відмов*:

$$\lambda(t)=\frac{f(t)}{P(t)}=-\frac{dP(t)}{P(t)dt}. \qquad (1.2)$$

Ділення на $P(t)$ збільшує частоту відмов, бо величина ймовірності завжди менше 1, крім того, на кінцевому етапі вона різко зменшується.

Для сучасних електронних приладів та їх компонентів різниця між $f(t)$ і $\lambda(t)$ у відносно короткі проміжки невелика і має однаковий порядок.

Перед цим розглядалися функції розподілу довільного вигляду. Однак досвід і статистика, накопичена за багато років експлуатації технічних об'єктів, дають змогу стверджувати, що залежність інтенсивності відмов λ від часу t можна представити так, як показано на рис. 1.4.

На цій кривій чітко виділяють три етапи експлуатації виробу.

1. Етап припрацювання. Йому властивий високий рівень інтенсивності відмов, яка досить швидко зменшується і відповідає початковому періоду функціювання об'єкта. У цей час проявляються дефекти виробництва, непомічені вихідним контролем. Щоб подолати недосконалість контролю, застосовують прогони і тренування виробів у граничних режимах (за температурою, вологістю, напругами живлення тощо). Виробник повинен завершити цей етап, не передаючи вироби в експлуатацію.

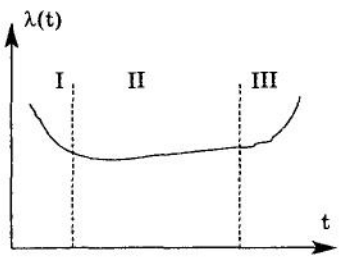


Рис. 1.4. Залежність інтенсивності відмов від часу експлуатації об'єкта

2. Період нормальної експлуатації виробу. Інтенсивність відмов стабілізується, має відносно невеликий рівень і загалом зростає через старіння компонентів та інші фактори.

3. Початок інтенсивної деградації виробу. Кількість відмов різко зростає, тому експлуатація може стати недоцільною через значні втрати на проведення відновлювальних (ремонтних) робіт. Початок третього етапу визначає технічний ресурс виробу.

> ***Технічний ресурс виробу*** *— прогнозована тривалість його експлуатації від початку до моменту, коли інтенсивність відмов збільшується до рівня, який робить подальшу експлуатацію неможливою або недоцільною з економічних міркувань.*

Отже, користувача цікавить насамперед етап нормальної експлуатації. Тому необхідно проаналізувати математичні способи опису розподілу ймовірності безвідмовної роботи саме в цей період.

1.2. Розподіли ймовірності безвідмовної роботи

Закономірності розподілу ймовірності безвідмовної роботи обчислювального пристрою чи системи дають можливість прогнозувати час їхньої експлуатації і вчасно замінювати їх на більш надійні у технологічному процесі, що розглядається.

Експоненціальний розподіл

Із практичного досвіду та аналізу фізичних або хімічних процесів, що відбуваються у внутрішній структурі компонентів електронного обладнання, можна зробити висновок, що за реальний час експлуатації електронного обладнання суттєвої деградації компонентів не відбувається, тобто $\lambda(t) = const$.

Будь-яка формула, зокрема і $\lambda(t) = const$, є лише математичною моделлю фізичного (хімічного) процесу, що проходить у внутрішній структурі того чи іншого елемента обладнання. Тільки з'ясувавши закономірності цього процесу, можна отримати точніші співвідношення для визначення часу, поки об'єкт буде залишатися працездатним. Вони повинні мати фізичний (хімічний) характер і описувати реальні процеси, а не поведінку моделі. Так, для лампочок розжарення такі процеси досить детально вивчені: вольфрамова спіраль з часом поступово випаровується, стає тоншою, з'являються неоднорідності за товщиною. Коли вона досягає критичної межі, воль-

фрамова спіраль перегорає. Цей процес має визначений характер, і можна досить достовірно прогнозувати середній час напрацювання лампочок певного виробника на відмову. (Лише середній час, тому що кожна лампочка має деякі відхилення своїх параметрів від середніх значень.)

Середній час напрацювання на відмову *— відношення сумарного часу напрацювання відновлювального об'єкта (пристрою) до математичного сподівання кількості його відмов протягом цього часу.*

За визначенням:

$$-\frac{dP(t)}{P(t)dt}=\lambda$$

або:

$$-\frac{dP(t)}{dt}=\lambda\cdot P(t).$$

Розв'язавши це диференціальне рівняння щодо $P(t)$, отримують:

$$P(t)=e^{-\lambda t}. \qquad (1.3)$$

Таким чином, ймовірність безвідмовної роботи об'єкта описує *експоненціальний закон розподілу*, тобто вона зменшується з часом як експонента. Закон має декілька властивостей:

1. Це один з найпростіших розподілів у теорії ймовірності, тому що визначається лише одним параметром λ і дає змогу отримати прості співвідношення для середнього часу, або математичного сподівання безвідмовної роботи об'єкта:

$$T=\frac{1}{\lambda}. \qquad (1.4)$$

Це так зване напрацювання на відмову, або очікуваний час між відмовами. Його не варто сприймати дослівно, бо ймовірність пропрацювати такий час без відмов досить мала:

$$P(T)=e^{-\lambda T}=e^{-T\frac{1}{T}}=\frac{1}{e}\approx 0{,}37.$$

Отже, напрацювання на відмову — лише формальний параметр, що дійсний тільки для об'єктів, які після відмови відразу відновлюються.

2. Визнають ймовірність безвідмовної роботи пристрою за час в інтервалі $t_1 \dots t_2$. Відповідно з (1.1) отримують:

$$P(t_1,t_2)=\frac{P(t_2)}{P(t_1)}=\frac{e^{-\lambda t_2}}{e^{-\lambda t_1}}=e^{-\lambda(t_2-t_1)}. \qquad (1.5)$$

Це значить, що об'єкт, для якого чинний експоненціальний розподіл, залишається весь час немов новий, оскільки ймовірність безвідмовної роботи залежить тільки від тривалості періоду експлуатації пристрою і не залежить від того, скільки пристрій пропрацював до цього.

3. Експоненціальний розподіл суттєво спрощує розрахунок надійності складних систем. Припустимо, що комп'ютерна мережа складається з N ПК. Вона працездатна лише, коли всі її компоненти працездатні. Таке *з'єднання — послідовне* з точки зору надійності (електрично воно не обов'язково повинно бути таким). Тоді ймовірність безвідмовної роботи мережі $P_c(t)$ є добутком ймовірностей безвідмовної роботи компонентів, тобто:

$$P_c(t)=\prod_{i=1}^{n}P_i(t)=\prod_{i=1}^{n}e^{-\lambda_i(t)}=e^{\left(-\sum_1^n\lambda_i\right)t}$$

або:

$$\lambda_c=\sum_{i=1}^{n}\lambda_i, \qquad (1.6)$$

де $P_i(t)$, $\lambda_i(t)$ — ймовірність безвідмовної роботи кожного ПК та інтенсивність відмов; $P_c(t)$, λ_c — параметри системи з N комп'ютерів.

Отже, коли всі компоненти системи необхідні для забезпечення її працездатності, то інтенсивність відмов системи є сумою інтенсивностей відмов компонентів. Це правило — дуже зручний інструмент для розрахунку надійності складних систем. Проте необхідно застерегти, що воно чинне лише для компонентів, для яких $\lambda(t)=const$. В інших випадках розрахунки суттєво ускладнюються через залежність інтенсивності відмов від часу. Тому залишається справедливою лише формула:

$$P_c(t)=\prod_{i=1}^{n}P_i(t),$$

а відповідний розподіл можна знайти, обчисливши значення $P_c(t)$ для ряду значень часу.

Нормальний розподіл (Гауса)

Не всі компоненти комп'ютерної техніки мають постійну інтенсивність відмов. У механічних, електромеханічних, гумотехнічних компонентах відбуваються помітні процеси старіння, тому тут застосовують інші розподіли. Серед них найуніверсальнішим є *нормальний розподіл (Гауса)*, для якого густина розподілу:

$$f(t)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(t-T)^2}{2\sigma^2}} \qquad (1.7)$$

і функція розподілу, тобто ймовірність, що об'єкт відмовить за час менший, ніж t, — площа під відповідною ділянкою кривої розподілу:

$$Q(t)=\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{t}e^{-\frac{(t-T)^2}{2\sigma^2}}dt. \qquad (1.8)$$

Нормальний розподіл визначається двома параметрами: T — середній час безвідмовної роботи об'єкта; σ — середнє квадратичне відхилення від середнього часу безвідмовної роботи.

Функція $f(t)$ досягає максимуму при $t = T$ — значення тривалості роботи об'єкта, при якому частота відмов максимальна. Графіки функцій $f(t)$ і $Q(t)$ зображені на рис. 1.5.

Чим більше значення σ, тим ширшою стає крива розподілу. Найбільша кількість відмов відбувається біля точки $t = T$. Таким чином, у межах від $T - \sigma$ до $T + \sigma$ виникає 68 % відмов.

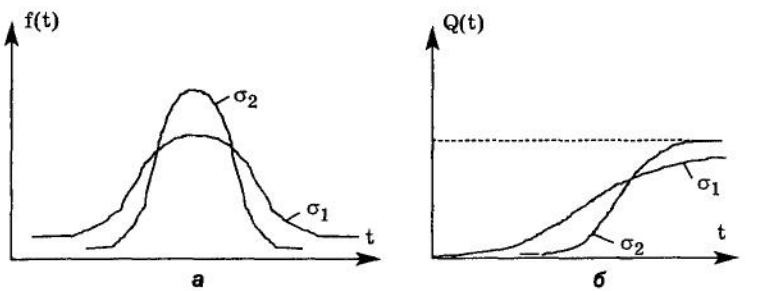


Рис. 1.5. Функції середнього квадратичного відхилення від середнього часу безвідмовної роботи об'єкта

Розподіл Вейбула

Досить часто у теорії надійності застосовують *розподіл Вейбула*. Це проста і штучна модель, що змінюється у широких межах залежно від її параметрів. Її запропонували для оцінювання міцності матеріалів, але далі часто використовували для оцінювання надійності. Ймовірність безвідмовної роботи у цьому випадку:

$$P(t)=e^{-\lambda t^{\beta}}, \qquad (1.9)$$

де λ та β — параметри моделі. $\beta = 0{,}2...0{,}4$ — для електронних пристроїв зі спадаючою функцією інтенсивності відмов і $\beta = 1{,}2...1{,}4$ — для механічних пристроїв з поступовим характером відмов.

Отже, всі згадані розподіли є лише математичними моделями фізичних процесів, що відбуваються у внутрішній структурі компонентів, з яких складаються об'єкти. Знання цих процесів дає змогу прогнозувати технічний стан об'єкта на тривалий період з більшою точністю. До того ж, враховуючи статистичний характер поняття надійності, не можна сподіватися на велику точність розрахунків надійності на основі лише математичних моделей.

У відповідальних випадках, коли, наприклад, комп'ютери застосовують для управління реальними об'єктами, які можуть становити небезпеку для людей або довкілля (ядерні установки, космічна апаратура), необхідно проводити натурні випробовування на надійність досить великої кількості об'єктів для того, щоб одержати розподіл, який відповідає реальним умовам експлуатації. У випадках, коли отримані показники надійності не відповідають вимогам безпеки чи іншого аналогічного критерію, необхідно застосовувати спеціальні методи забезпечення надійності.

1.3. Методи забезпечення надійності

Щодо забезпечення надійності будь-якого обладнання (не обов'язково електронного) принциповою є можливість (чи неможливість) відновлення працездатності об'єкта в разі його відмови. Тому в теорії надійності всі об'єкти поді-

ляють на два великих класи: *відновлювані* і *невідновлювані*. Підхід до забезпечення надійності обох класів зовсім різний.

Методи забезпечення надійності відновлюваних об'єктів

Забезпечення надійності цих об'єктів полягає в організації ефективного ремонту обладнання з мінімальними затратами часу.

Відновлювані об'єкти — об'єкти, які в період експлуатації в разі виникнення відмов можуть бути відремонтовані шляхом заміни несправних компонентів.

Для відновлюваних об'єктів основним показником надійності вважають *коефіцієнт готовності*, що визначає частину корисного часу $t_к$ протягом якого об'єкт нормально працює щодо загального часу експлуатації. Тобто:

$$K_г = \frac{t_к}{t_к + t_в}, \qquad (1.10)$$

де $t_в$ — час, що витрачено на відновлення працездатності. Він містить дві складові: t_1 — час, що витрачають на пошук несправності; t_2 — час власне відновлення.

Особливістю сучасного електронного обладнання є його модульна побудова, тому відновлення таких об'єктів найчастіше може полягати у простій заміні несправного модуля (блока) справним. Вона не потребує великих затрат часу. Пошук несправності, навпаки, — здебільшого трудомістка процедура і може тривати довго. На коефіцієнт готовності найбільше впливає саме цей час.

До основних методів забезпечення надійності відновлюваної електронної апаратури належать:

— створення максимально сприятливих умов для прискорення ремонту. Це означає, що конструкція приладу повинна бути ремонтопридатною, має бути забезпечено зручний доступ для заміни несправних компонентів, з'єднання між типовими елементами заміни (ТЕЗ) відбувається на основі роз'ємів і т. д.;

— застосування автоматизованого пошуку несправностей на основі апаратного і (або) програмного самоконтролю функціювання апаратури, тестового діагностування та використання спеціальних автоматизованих систем контролю і діагностики.

Згадані методи є обов'язковими при спробі забезпечити більш високу надійність обчислювальної техніки.

Методи забезпечення надійності невідновлюваних об'єктів

Для невідновлюваних об'єктів підхід до забезпечення надійності зовсім інший, ніж для відновлюваних, оскільки вони не підлягають ремонту.

Невідновлювані об'єкти — об'єкти, для яких ремонт із конструктивних причин або умов експлуатації є неможливим.

До пристроїв, що мають конструктивні особливості, належать інтегральні схеми, кінескопи. Пристрої, для яких ремонт неможливий через умови експлуатації — бортова і космічна апаратура, морські буї, глибоководне та підземне обладнання тощо.

Отже, запобігти виникненню несправностей принципово неможливо. Тому ситуацію, коли деякі компоненти об'єкта втрачають працездатність, потрібно сприймати як очікувану і нормальну. Крім того, необхідно враховувати сильну залежність надійності від складності об'єкта. Якщо ймовірність безвідмовної роботи одного компонента позначити через p, то відповідна ймовірність для системи, що складається з N таких компонентів, буде:

$$P_c = p^N$$

за умови, що для функціювання системи повинні бути працездатні всі N компонентів.

Припустимо, що $p = 0{,}99$. Тоді можна обчислити зменшення P_c при зростанні кількості компонентів (табл. 1.1). При $N = 70$ система стає непрацездатною (навряд чи кому потрібен прилад, який має ймовірність без-

Таблиця 1.1

Зменшення надійності пристроїв від їх складності

N	1	10	20	30	40	50	60	70
P_c	0,99	0,9	0,81	0,73	0,65	0,58	0,52	0,47
N	80	90	100	200	300	400	500...	
P_c	0,43	0,38	0,34	0,12	0,04	0,0136	0,0046...	

відмовної роботи меншу, ніж 0,5). Однак цьому можна запобігти, якщо застосувати методи введення надлишковості.

Щоб пояснити це поняття, найкраще провести аналогію з фабрикою або заводом, на якому працює, наприклад, 100 робітників. Виробничий процес організовано так, що в будь-який день кожен зі 100 працівників має бути на своєму робочому місці для нормального функціювання закладу. Важко уявити, що ніхто з персоналу не хворіє і не запізнюється. Такий заклад ніколи не зміг би нормально функціювати. Однак сотні фабрик і заводів, навіть зі значно більшою кількістю робітників, працюють, незважаючи на те, що частина персоналу щодня не виходить на роботу. Це зумовлено тим, що при чіткій організації виробничого процесу завжди передбачені надлишкові («зайві») працівники, щоб навіть неповний склад робітників зміг нормально виконати заплановану роботу. Коли хтось захворіє, інший повинен виконати його функцію. Саме на цьому і ґрунтується ідея побудови електронних систем з компонентами з обмеженою надійністю.

Цей приклад яскраво ілюструє саме поняття надлишковості. Її, у свою чергу, поділяють на часову, інформаційну і структурну.

Часова надлишковість. Передбачає, що за рахунок швидкого здійснення операцій їх можна виконати двічі або й тричі, і порівняти результат для того, щоб переконатися в його правильності.

Інформаційна надлишковість. Її особливість у тому, що у випадку, коли інформація, яка є вихідною для обчислень або утворення керуючих впливів у системах управління, містить помилки, то для нейтралізації їх дії застосовують спеціальні методи кодування (виявляють і (або) виправляють помилки).

Структурна надлишковість. У структуру системи вводять надлишкові компоненти, які відіграють роль запасу (резерву) на випадок відмови основного обладнання.

Ідея структурної надлишковості проста і була відома задовго до появи не тільки комп'ютерів, а й електроніки. Д. Ладнер у статті про обчислювальну машину Бебіджа в 1834 р. писав: «Найбільш конкретний і ефективний спосіб контролю помилок, що виникають у процесі обчислень, полягає в тому, щоб забезпечити виконання тих самих розрахунків на різних, не зв'язаних один з одним,

обчислювачах; цей контроль виявляється ще більш ефективним, якщо обчислення виконують різними способами».

Ґрунтовні наукові дослідження в цій сфері започаткував американський математик Дж. фон Нейман (1903—1957) у класичній праці «Синтез надійних організмів з ненадійних компонентів». Тут уперше було доведено можливість довільної побудови надійних пристроїв з компонентів з обмеженою надійністю шляхом цілеспрямованого введення структурної надлишковості. Сьогодні методи, що базуються на структурній надлишковості, стають ледве не стандартним способом забезпечення надійності комп'ютерних систем промислового і спеціального призначення.

Так, у системі автоматичного керування польотом (автопілот) літака «Boueng 737/300» стандартний блок складається з двох каналів обчислень, представлених трьома різними центральними процесорами. Один з них реалізує всю систему програмного забезпечення керування польотом, а інші два — тільки критичні функції. Цей стандартний блок резервований, нейтралізація помилок відбувається за допомогою моніторів, що порівнюють сигнали. Їх у кожному стандартному блоці — два.

Метою введення структурної надлишковості є зберігання можливості системи виконувати задані функції за наявності помилок через несправності самої системи. Помилки при цьому виявляють і усувають, а постійні несправності ліквідують у процесі технічного обслуговування апаратури.

Комп'ютерні системи як синтез відновлюваних і невідновлюваних об'єктів

Комп'ютерні системи, які використовують для управління реальними об'єктами, не можна беззастережно зарахувати ні до відновлюваних, ні до невідновлюваних об'єктів. З одного боку, такі системи здебільшого є відновлюваними, а з іншого — максимальний час їх відновлення обмежений технологічними вимогами з боку об'єкта управління. До методів забезпечення надійності комп'ютерних систем належать як методи, що застосовують для відновлюваних систем (прискорення пошуку несправностей), так і властиві невідновлюваним системам (структурна надлишковість) методи.

Принципова особливість таких систем полягає в багаторазових обчисленнях. Вони утворюються шляхом n-кратного повторення обчислювального процесу в таких розрізах:

— часовому (повторний рахунок);

— просторовому (на інших апаратних засобах);

— інформаційному (з використанням інших програм і даних).

За багатоканальної архітектури обчислення проводять рівноцінним виконанням екземплярів однієї і тієї самої програми в апаратних каналах. Такою є система керування космічним кораблем багаторазового використання «Шатл» НАСА, де n =4.

Широко застосовують інтелектуальні системи з архітектурою RAID (Redundant Array of Inexpensive Disks — матриця недорогих дисків з надлишковістю). Ці системи використовують для захисту інформації від помилок при збереженні в корпоративних і банківських мережах. Вони не тільки зберігають дані, а й захищають їх від збоїв та надають важливу інформацію про технічний стан твердих дисків.

RAID-система складається з керуючої програми або контролера і групи твердих дисків, що працюють спільно для забезпечення вищої, порівняно з окремим диском, продуктивності, а також стійкості до збоїв. Відмовостійкість досягається завдяки простому дублюванню інформації на двох окремих дисках (стандарт RAID рівня 1) та збереженню інформації за рахунок корекції помилок за допомогою завадостійких кодів (стандарт RAID рівня 3 і 5), що дають змогу навіть у випадку відмови одного з дисків відновити втрачені дані.

Необхідно згадати також про структурну надлишковість у біологічних системах, в яких природа широко використовує цей метод. Йдеться не тільки про дублювання органів (очі, вуха, легені, нирки), а, головним чином, про надлишковість на рівні окремих клітин і нейронів головного мозку. За даними нейрофізіології, ця надлишковість має дуже високий рівень: є підстави вважати, що в головному мозку людини кількість нейронів у тисячі разів більша, ніж це мінімально необхідно. Людина в середньому живе 60—70 років, а комп'ютер або телевізор, які набагато простіші за складом, ніж мозок людини, навряд чи можуть безперервно функціювати кілька років. За рахунок саме надлишковості відмирання частини нейронів суттєво

не впливає на роботу мозку (їх так багато, що вистачає на все життя). Крім того, на клітинному рівні відбуваються процеси самовідновлення. Це дає змогу забезпечити дуже високий рівень надійності людського організму порівняно з технічними об'єктами.

Відомий вражаючий історичний приклад, який доводить це твердження. У 1848 р. американському залізничнику Ф. Гейджу внаслідок аварії металевою палицею було наскрізь пробито голову. Досить швидко він одужав і працював на залізниці далі. Будь-яких помітних змін у поведінці і психіці Ф. Гейджа не сталося. Металеву палицю зараз експонують у Гарвардському університеті як символ винятково високої надійності людського мозку.

Щоб краще зрозуміти суть надійності, необхідно перейти до більш детального розгляду методів і засобів її забезпечення.

1.4. Резервування апаратури

Одним з методів підвищення надійності обчислювальних систем є резервування її складових. Його, у свою чергу, поділяють на окремі види.

Основні види резервування

Резервування ускладнює апаратуру. Цей метод застосовують не для всіх видів електронного обладнання та його рівнів.

> ***Резервування*** *(франц. reserve — запас) — метод підвищення надійності шляхом включення резервних елементів при розробленні системи або в процесі її експлуатації.*

Розрізняють поелементне і загальне резервування.

Поелементне резервування. Воно найдоцільніше з точки зору апаратних витрат, проте на практиці його задіюють, головним чином, на рівні функційно завершених блоків (виняток — цифрова апаратура, де поелементне постійне резервування застосовують на будь-якому рівні). Щодо окремих радіоелементів (резисторів, конденсаторів, транзисторів тощо) цей вид резервування не використовують через неможливість забезпечення незмінності параметрів резервованої схеми у процесі виникнення несправностей.

Загальне резервування. У цьому разі резервують увесь пристрій. Цей метод застосовують здебільшого у великих багатокомп'ютерних системах і мережах. Він вимагає складного програмного узгодження функцій кожного комп'ютера при втраті працездатності елементів системи.

За способом включення резерву виокремлюють постійне резервування і резервування заміщенням.

Постійне резервування. Полягає у включенні резервної апаратури протягом усього часу функціювання основної.

Резервування заміщенням. Цей метод є найефективнішим. Він полягає у задіянні (включенні) резервного елемента структури замість елемента, який вийшов з ладу. За такого резервування резерв може бути:

— ненавантаженим (апаратура не увімкнута у процесі функціювання);

— полегшеним (увімкнута частина резервної апаратури);

— навантаженим (увімкнута вся резервна апаратура).

До переваг цього способу резервування належать:

— один резервний елемент використовують для кількох основних;

— резервні елементи знаходяться у пасивному стані (наприклад, знеструмлені), що зберігає їх технічний ресурс.

Упевнитися, що цей метод підвищує надійність, можна, розглянувши рис. 1.6. Для цього спершу слід розрахувати ймовірність безвідмовної роботи найпростішої структури (один основний A_0 елемент і один резервний A_p).

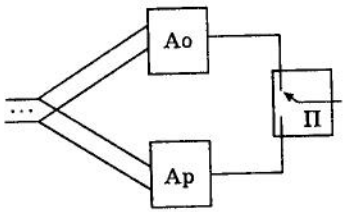


Рис. 1.6. Найпростіша структура резервування комп'ютерної системи

Заміщення здійснює перемикач *П* або оператор. Це залежить від природи елементів. Ймовірності безвідмовної роботи основного і резервного елементів — P.

Структура залишатиметься працездатною в таких ситуаціях:

— обидва елементи (основний і резервний) працездатні, ймовірність такої події — p^2;

— основний елемент справний, резервний вийшов з ладу, відповідна ймовірність — $p(1-p)$;

— резервний елемент справний, основний відмовив, відповідна ймовірність — $(1-p)p$.

Тоді ймовірність безвідмовної роботи структури можна записати як суму:

$$P_c = p^2 + 2p(1-p) = 2p - p^2,$$

враховуючи реальну надійність перемикача:

$$P_c = p_n(2p - p^2).$$

Припустивши, що перемикач абсолютно надійний і ймовірність його безвідмовної роботи $p_n = 1$, можна розрахувати ефект підвищення надійності наведеної схеми резервування. Нехай $p = 0{,}9$, тоді маємо $P_c = 0{,}99$, тобто ймовірність відмови зменшилася на порядок з $q = 0{,}1$ до $Q_c = 0{,}01$. У реальних умовах $P_n < 1$. При ненадійному перемикачеві можна навіть погіршити надійність. Необхідно знайти, за яких умов цього не трапиться. Очевидно, якщо:

$$P_c > p$$

або:

$$p_n(2-p^2) > p \;\; i \;\; p_n > \frac{p}{2p-p^2}$$

будемо мати підвищення надійності. Для цього прикладу при $p = 0{,}9$:

$$P_n > 0{,}91,$$

тобто перемикач повинен мати надійність того ж порядку, що основний і резервний елементи.

Необхідно розглянути структуру, коли один резервний елемент використовують для двох основних (рис. 1.7). У цьому разі структура залишатиметься працездатною, поки працездатними будуть перемикач і хоча б два з трьох елементів (два основних і один резервний). Для ймовірності безвідмовної роботи структури можна записати:

$$P_c = p_n[p^3 + 3p^2(1-p)] = p_n(3p^2 - 2p^3).$$

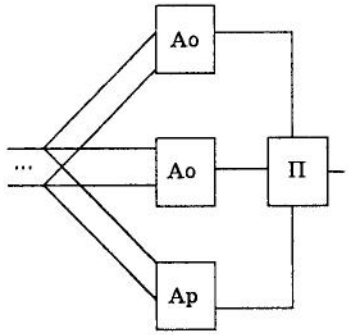


Рис. 1.7. Структура резервування системи з двома основними і одним резервним елементом

За тієї ж надійності основних і резервного елементів та $p_n = 1$ отримують $P_c = 0{,}982$. Порівняно з попереднім прикладом ефект підвищення надійності трохи гірший. Це можна пояснити меншим рівнем надлишковості.

Резервування заміщенням, коли з n елементів k є резервними, можна записати:

$$P_c = p_n \sum_{i=k}^{n} C_n^i p^i (1-p)^{n-i}. \qquad (1.11)$$

Реалізація функції переключення на резерв залежить від природи і функцій елементів, які потрібно переключати. Перемикачем може бути людина (оператор), коли операція переключення складна. Наприклад, для того щоб переключитися на резервне живлення від дизель-генератора при зникненні напруги в мережі, слід виконати ряд досить складних операцій, які важко автоматизувати. А коли таким компонентом є лампочка розжарення, неважко створити простий автомат, який при перегоранні однієї лампочки включить іншу. Загалом проблема у створенні відповідного автомата полягає в тому, що потрібно мати чіткі й однозначні критерії працездатності (непрацездатності) відповідного компонента системи. Їх не завжди легко знайти, особливо для багатофункційних компонентів.

Мажоритарний метод резервування

Мажоритарний (франц. majorité — більшість) метод резервування вважають класичним. Він був запропонований Дж. фон Нейманом. Реалізують його як постійне резервування з логічним перемиканням на резервні елементи. За найпростішої ситуації він потребує збільшення апаратних витрат щонайменше втричі. Схема включення елементів структури показана на рис. 1.8.

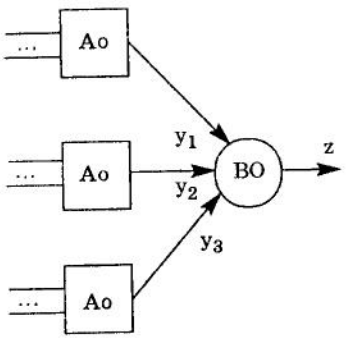


Рис. 1.8. Одноланцюгова структура відновлення роботи системи

На всі три елементи подають однакові вхідні сигнали $x_1, x_2, \ldots x_i$. Вихідні сигнали y_1, y_2, y_3 надходять на *відновлюючий орган (ВО)*, функція

якого полягає в утворенні результуючого сигналу за більшістю значень y_1, y_2, y_3 (вхідні сигнали для ВО наче голосують). Для двійкових сигналів функцію ВО можна записати так:

$$Z = y_1 y_2 \vee y_1 y_3 \vee y_2 y_3.$$

(Це мажоритарна функція алгебри логіки, звідси й назва методу.) Значення сигналу, що відрізняється від інших, ігнорують. Обґрунтувати цей алгоритм прийняття рішень можна так: якщо позначити ймовірність відмови одного елемента через q, то ймовірність одночасної відмови зразу двох елементів (незалежно один від одного) буде q^2. Враховуючи, що $q << 1$,

$$q^2 << q,$$

тому є всі підстави вважати, що відмовив саме один елемент, а не два одночасно. Так, для сучасних елементів обчислювальної техніки ймовірність відмови за кілька годин роботи має порядок $10^{-7} \ldots 10^{-6}$, тоді $q^2 \approx 10^{-14} \ldots 10^{-12}$, і можна безпомилково прийняти рішення про відмову саме одного елемента, а не двох одночасно. Слід ще раз наголосити, що ці твердження чинні лише за умови незалежності відмов, тобто тільки в разі відсутності загальної першопричини відмов, наприклад завад по колах живлення.

За більших рівнів надлишковості алгоритм прийняття рішень залишається таким самим — вихідний сигнал ВО визначається більшістю значень вхідних сигналів. Для ймовірності безвідмовної структури при n-кратному резервуванні можна записати:

$$P_c = P_{BO} \sum_{i=l}^{n} C_n^i p^i (1-p)^{n-i}, \qquad (1.12)$$

де P_{BO} — відповідна ймовірність ВО; $L = \frac{n}{2} + 1$ для парних n і $L = \frac{n+1}{2}$ для непарних. Зауважимо, що в (1.12) P_{BO} входить як множник і тому обмежує значення P_c, тобто P_c завжди менше P_{BO}. Це принципова вада *одноланцюгової структури* (див. рис. 1.8).

Для того щоб зняти це обмеження, застосовують *багатоланцюгові структури* (рис. 1.9), в яких ВО теж n-кратно резервують. Для таких структур:

$$P_c = \sum C_n^i (PP_{BO})^i (1 - PP_{BO}). \qquad (1.13)$$

Найважливішим для цих структур є зниження обмежень на надійність ВО та можливість досягнення будь-якої високої надійності структури за зростання рівня надлишковості n. З певними несуттєвими обмеженнями на початковий рівень надійності елементів $P_c \to 1$ при $n \to \infty$ (рис. 1.10). Тобто за рахунок надлишковості можна побудувати практично абсолютно надійні структури з компонентів, які мають обмежену надійність.

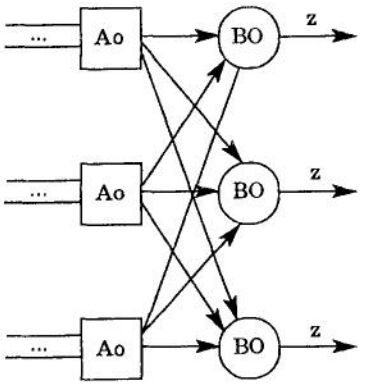


Рис. 1.9. Багатолінійна структура відновлення роботи системи з трьома ВО

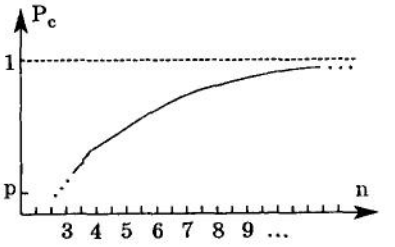


Рис. 1.10. Графік залежності ймовірності безвідмовної структури від кратності резервування

Крім надлишковості, для багатолінійних структур характерні багатолінійні зв'язки між послідовними каскадами структури (Дж. фон Нейман назвав їх «пучками зв'язків»). Усі сигнали в такій структурі поширюються багатьма каналами, що також сприяє підвищенню надійності. Проте виникає проблема останнього ВО, оскільки користувач повинен мати не n сигналів, а один, тому на кінцевому виході їх необхідно об'єднати. Таким чином, останній ВО буде обмежувати надійність цієї структури.

На користь ефективності багатолінійних мажоритарних структур свідчать і дослідження нейрофізіологів: є деякі підстави вважати, що саме за таким принципом побудована більшість біологічних систем, зокрема мозок людини. Окрім того, ті ж дослідження показують, що надлишковість біологічних систем на клітинному рівні дуже висока (принаймні йдеться про порядки).

Для технічних об'єктів, навіть для сучасної мікроелектроніки, такий рівень, звісно, нереальний. Сьогодні в ситуаціях, коли необхідно забезпечити високий рівень надійності, найчастіше застосовують потроєння апаратури ($n = 3$), в деяких відповідальних випадках — $n = 4{,}5$, але вищі рівні надлишковості є вже винятком.

Незважаючи на поширеність мажоритарного методу, при $n > 3$ його не можна вважати оптимальним. Це можна довести, розглянувши структуру з $n = 4$. При відмові одного елемента вона ще залишається працездатною, але вже при двох несправних елементах перестає правильно функціювати (значення двох сигналів на входах ВО дорівнює 1, двох — 0, більшості немає). Водночас у стуктурі ще залишаються два повністю працездатні елементи, проте вона вже непрацездатна. Із цієї точки зору принцип більшості не є найкращим.

Інший алгоритм, що не має цієї вади, такий: елемент, вихідна реакція якого відрізняється від реакцій інших елементів, відключають, надалі він не бере участі в утворенні результуючого сигналу. Така своєрідна «ампутація» дає змогу зберегти працездатність структури, коли хоча б два її елементи є працездатними. Може виникнути питання, чому саме два, а не один. За ситуації, коли лишаються два останні елементи і виходить з ладу передостанній, неможливо впевнено віддати перевагу одному з них і таким чином утворити правильний результуючий сигнал. Реалізація такого алгоритму складніша за мажоритарний.

Відновлюючий орган з пам'яттю як метод резервування

Ефект підвищення надійності можна отримати, застосовуючи ВО з пам'яттю. Ймовірність безвідмовної роботи, як і за мажоритарного ВО, зростає. Формули для обчислення цієї ймовірності збігаються з (1.12) для однолінійних і (1.13) для багатолінійних структур. Відмінність лише в кількості доданків, які утворюють суму, — вони зростають з $n - 1$ до $n - 2$, тобто для однолінійних структур, що мають ВО з пам'яттю:

$$P_c = P_{BO} \sum_{i=2}^{n} C_n^i p^i (1-p)^{n-i} \qquad (1.14)$$

і багатолінійних:

$$P_c = \sum_{i=2}^{n} C_n^i (P_{BO} p)^i (1 - P_{BO} p)^{n-i}. \qquad (1.15)$$

Порівнявши формули (1.12) з (1.14) та (1.13) з (1.15), можна зробити висновок: при $n > 3$ надійність структур, що мають ВО з пам'яттю, вища, ніж у мажоритарних.

Незалежно від алгоритму відновлення сигналів (мажоритарного чи іншого), поріг L (кількість однакових сигна-

лів, на основі яких приймається рішення) може змінюватися в широких межах, а саме від 2 до n.

Крім надійності, важливими є й інші показники, наприклад ймовірність того, що результат обчислень правильний — *достовірність вихідних сигналів* $P_Д$. Значення достовірності $P_Д$ залежить від кількості сигналів, що збігаються. На цій основі приймається рішення про їх правильність, а це в даній ситуації величина порога L. Якщо ймовірність відмови (помилки) одного елемента структури $(1-p)$, то ймовірність одночасної відмови L елементів:

$$(1-p)^L.$$

Достовірність можна визначити як ймовірність протилежної події, тобто:

$$P_Д = 1-(1-p)^L. \qquad (1.16)$$

Як видно з формули, достовірність результату зростає зі збільшенням L і буде мати максимальне значення при $L = n$, тобто, коли рішення приймається одноголосно, а не просто більшістю. При зростанні порога L ймовірність безвідмовної роботи зменшується, тому що при високих значеннях L для безвідмовної роботи потрібно більше працездатних елементів у структурі. На рис. 1.11 наведені залежності $P_c = f(L)$ та $P_Д = f(L)$ для різних значень порога при $p = 0{,}9$ однолінійної структури з $n = 5$ (спрощуючи розрахунки, вважають $P_{BO} = 1$).

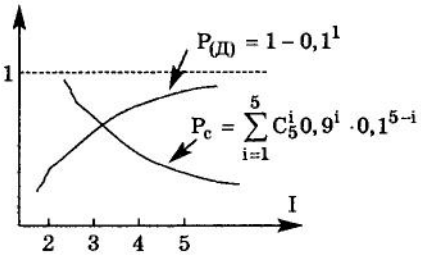


Рис. 1.11. Графік залежності $P_{(Д)}$ і P_c від порога L

Поріг L є зручним інструментом для вибору тих значень P_c і P, що найбільше відповідають умовам експлуатації системи і, головне, вимогам до надійності, яка в багатьох випадках визначається вимогами до безпеки об'єкта.

Структурна надлишковість, хоч і важливий, але не єдиний аспект, що визначає рівень надійності комп'ютерних систем та їх складових. Не менш суттєвий вплив на надійність і достовірність має інформаційна надлишковість.

1.5. Інформаційна надлишковість як універсальний засіб контролю

Традиційно поняття інформаційної надлишковості (ІН) найчастіше пов'язують з використанням завадостійких кодів для передавання і зберігання інформації. Щоб зрозуміти принципи їх побудови та використання, необхідно з'ясувати суть інформаційної надлишковості.

Суть інформаційної надлишковості

За твердженнями американського математика, засновника сучасної математичної теорії інформації К. Шеннона (1916—2001), рівень ІН визначають відносним перевищенням максимально можливої ентропії (грец. en — в; tropē — поворот, повторення) H_{max} над реальною ентропією $H_{реал}$ конкретного джерела інформації при використанні певного способу кодування:

$$J = 1 - \frac{H_{реал}}{H_{max}} \qquad (1.17)$$

або в абсолютному обчисленні:

$$J = H_{max} - H_{реал}, \qquad (1.18)$$

де $H_{max} = \log_2 N$ та $H_{реал} = \sum_{i=1}^{N} p_i \log p_i$, а N — кількість можливих повідомлень, p_i — ймовірність використання (появи) i-го повідомлення.

Таке перевищення виникає за будь-якого відхилення розподілу ймовірностей появи окремих повідомлень від рівномірного (при кодуванні повідомлень комбінаціями символів однакової довжини). Коли окремі комбінації джерело не використовує (ймовірність їх появи дорівнює 0), множина всіх можливих повідомлень X_0 природно розпадається на підмножину дозволених слів X_g (які джерело використовує) і підмножину заборонених слів $X_з$ (які джерело не використовує). Цей випадок з практичної точки зору найцікавіший. Про нього йтиметься далі.

Природна і штучна надлишковість

ІН характеризує специфічність джерела. Її досягають, використовуючи завадостійкі коди і додаючи перевіркові сигнали, утворені за певними правилами. Саме це і робить джерело індивідуальним. Перевірка виконання штучно введених правил дає змогу виявляти і (або) виправляти помилки при передаванні чи зберіганні інформації.

Однак наявність ІН не обов'язкова, щоб джерело було індивідуальним. (Адже навіть у випадку, коли всі повідомлення рівноймовірні ($H_{max} = H_{реал}$ та $J = 0$), джерело залишається специфічним.) Його індивідуальність полягає в тому, що всі повідомлення рівноймовірні, а відхилення від рівноймовірності може свідчити про наявність помилок при передаванні або збереженні інформації. Парадоксальність цієї ситуації полягає у відсутності надлишковості й наявності помилок.

Отже, будь-яке джерело з відомою і стаціонарною статистикою повідомлень є специфічним. За допомогою певних формальних правил можна визначити специфіку і відповідно виявити помилки. Звідси випливає, що контроль достовірності будь-якого джерела можна здійснити без введення *штучної надлишковості*. Таке твердження не є результатом формального доведення. Це припущення підтверджують приклади.

1. Повідомлення генеруються джерелом з різною ймовірністю, розподіл довільний, заданий спектрограмою, отриманою, наприклад, як результат тривалого спостереження за повідомленням джерела. Така спектрограма зображує відносні частоти появи того чи іншого слова. Приміром, в англійській мові символ (літера) *E* з'являється з частотою (ймовірністю) 0,12, *W* — 0,02 і т. д. В українській мові розподіл інший. Найбільшу частоту мають літери *О* (0,08) і *А* (0,07), а найменшу — *Ц* (0,0044) і *Є* (0,0037).

Саме розподіл частот характеризує індивідуальність відповідного джерела. Це — як відбитки пальців людини, за якими її можна ідентифікувати. Частотний розподіл вважають надійним критерієм достовірності інформації. Будь-яке його суттєве відхилення від зафіксованого протягом тривалого спостереження є ознакою виникнення помилок. Поодинокі відхилення (помилки) навряд чи призведуть до порушення розподілу, тому не можуть бути надійно виявлені. Це принципова вада всіх статистичних методів контролю. Крім того, навіть, коли помилки

мають регулярний характер їх можна виявити тільки із запізненням, після тривалого спостереження. Це є підставою для висновку, що статистичний метод контролю не може бути оперативним. Тому його не застосовують у тих випадках, коли отриману від джерела інформацію зразу використовують для управління реальними об'єктами, коли помилки, навіть поодинокі, можуть виявитися неприпустимими з точки зору можливих наслідків і втрат.

2. У частотному розподілі деякі повідомлення (слова або символи) мають нульову ймовірність за нормального функціювання об'єкта контролю (джерела). Вони утворюють підмножину заборонених слів $X_з$. Поява будь-якого слова з $X_з$ — ознака помилки, яку можна виявити практично миттєво апаратними чи програмними засобами. Для цього достатньо знати склад $X_з$. Враховуючи, що підмножини дозволених слів X_g і $X_з$ взаємно доповнюють одна одну до множини всіх можливих слів X_D, тобто:

$$X_D = X_з \cup X_з,$$

можна обмежитися фіксацією лише однієї підмножини. Для такої фіксації та реалізації відповідних процедур контролю необхідні вільні обчислювальні ресурси, насамперед пам'ять. Головною проблемою при цьому є часові обмеження. Адже процедури контролю необхідно виконувати щоразу, коли надходить чергове повідомлення. Тому вони мають бути простими і максимально короткими за часом.

Вихід із ситуації 1 можна знайти в класичній роботі К. Шеннона «Математична теорія зв'язку». Якщо розглядати частотний розподіл, утворений парами слів, які надходять від джерела в суміжні моменти часу, то з'ясовується, що певні пари мають нульову ймовірність надходження. Це відповідає вже ситуації 2, тобто можливості оперативного виявлення помилок. Щодо розподілу «трійок», «четвірок» і т. д. суміжних за часом повідомлень, то кількість штучно об'єднаних слів, що мають нульову ймовірність, зростатиме в геометричній прогресії.

Це досить детально ілюструють повідомлення українською мовою. На рівні окремих літер у частотному розподілі немає так званих провалів — у текстах використовують усі літери, але з різною ймовірністю. Проте в розподілі пар провали з'являються: не використовують м'який знак після голосних і на початку слова, сполучення літери «є» з приголосними та голосними («дє», «кє»).

Якщо уважно проаналізувати повідомлення від конкретного давача в комп'ютерну систему управління, на-

приклад технологічним об'єктом, то можна зауважити певні закономірності, наявні у статистичній структурі таких послідовностей. Тиск в об'ємі, що зменшується, не може теж зменшуватися, оскільки відповідно до закону Бойля—Маріотта добуток тиску на об'єм повинен залишатися постійним: $PV = const$.

Цю залежність можна використовувати у відповідних ситуаціях як надійну контрольну ознаку достовірності даних, що надходять від об'єкта в систему управління. У багатьох інших випадках саме взаємозалежність певних параметрів може виявитися найбільш корисною для організації ефективного контролю.

У типових комп'ютерних системах промислового призначення більшість інформації надходить до системи шляхом регулярного опитування давачів за певним (фіксованим) часовим регламентом. Повідомлення від окремих давачів, що характеризують параметри одного технологічного процесу, обов'язково будуть корельовані (це параметри одного і того самого процесу, який не може виходити за межі відповідних фізичних (хімічних) законів). Тому, як і в попередньому випадку, дотримання цих законів може бути ефективно використано для перевірки достовірності даних. Саме корельованість даних свідчить про наявність ІН, тому в таких ситуаціях вводити штучну ІН не обов'язково. І, головне, контроль на основі ІН, яку логічно назвати *природною*, охоплює не тільки помилки засобів передавання даних (характерно для штучної ІН), а й ввесь автоматизований комплекс, тобто порушення технологічного процесу і несправності засобів автоматизації. Це найважливіше для загальної ефективності контролю та запобігання аварійних ситуацій.

Слід з'ясувати, як природну надлишковість можна використати для організації контролю працездатності або технічного стану об'єкта.

Для цього необхідно, по-перше, оцінити ефективність контролю щодо його повноти, тобто визначити, яку частину всіх можливих помилок можна принципово виявити при його застосуванні; по-друге, побудувати основні процедури, що дасть змогу реалізувати контроль у складі існуючих або таких, що проектуються, комп'ютерних систем виробничого призначення.

Припустимо, що джерело (давач, первинний перетворювач, клавіатура, з якої оператор вводить виробничу інформацію в систему тощо) формує повідомлення у вигляді слів фіксованої розрядності n. Тоді загальну кількість усіх

можливих помилок (векторів помилок) можна записати як $(2^n - 1)$.

Вектор (лат. vectre — носій) помилки — двійкова n-розрядна комбінація, в якій одиниці відповідають тим розрядам, що спотворені помилками.

Якщо правильна комбінація:

$$X_i = (10110010),$$

а після спотворення помилки отримали:

$$X_i' = (10010110),$$

то відповідний вектор помилки:

$$E = X_i \oplus X_i' = (00100100),$$

де $\oplus$ — порозрядна сума за модулем 2. Для наочності цю операцію можна записати так:

$$\begin{array}{l} X_i = (10110010) \\ \oplus \\ X_i' = (10010110) \\ E = (00100100). \end{array}$$

З іншого боку, виявляють тільки такі помилки, які переводять (перетворюють) дозволене слово в заборонене. Тому кількість виявлених помилок дорівнює кількості заборонених слів (позначають її $m = (X_з)$). Тоді частина виявлених помилок:

$$\delta = \frac{m(X_з)}{2^n - 1}. \qquad (1.19)$$

Отже, чим менше використовувати слів (комбінацій з усіх можливих), тим більше помилок виявляється. Це надто загальне твердження, щоб можна було визначити конкретні помилки та побудувати відповідні процедури корекції.

Не всі можливі помилки реально виникають, крім того, не всі з них однаково серйозні. Тому варто диференціювати елементи підмножини $X_з$ відповідно до двох видів помилок:

1) які обов'язково повинні бути виявлені;

2) які коригувати не обов'язково.

Йдеться про скорочення кількості помилок, що підлягають виявленню, враховуючи тільки ті, які найімовірніші або найнебезпечніші для функціювання об'єкта управління. Це спрощує відповідні процедури контролю, що дуже важливо при їх апаратній реалізації.

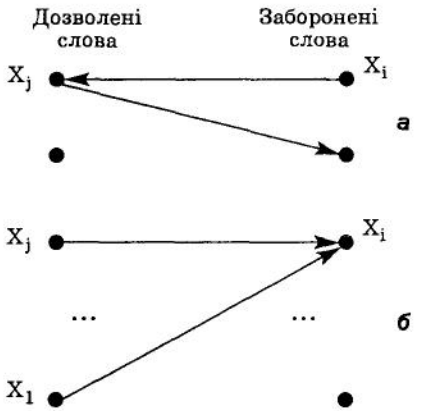


Рис. 1.12. Графічне представлення помилок у вигляді переходів між повідомленнями

Інколи рівень ІН дає змогу не тільки виявляти, а й виправляти помилки. Графічне представлення помилок у вигляді переходів між повідомленнями, або точніше перетворень дозволених слів у заборонені, наведено на рис. 1.12.

Необхідна умова для виправлення конкретної помилки — однозначність зворотного переходу від слова X_i, яке містить помилку, до слова X_j без помилок. Очевидно, така однозначність можлива лише за умови, коли в кожне заборонене слово переходить не більше одного дозволеного (ситуація а). Коли переходить кілька дозволених слів (ситуація б), то, не маючи підстав для того, щоб віддати перевагу одному з варіантів зворотного переходу, визначити «зворотний шлях» для виправлення помилки практично неможливо. Можна стверджувати, що для виправлення $m(E)$ помилок у кожному дозволеному слові повинно існувати $m(E)$ відповідних заборонених слів. Для всіх дозволених слів джерела можна записати так:

$$m(X_g) \cdot m(E) \le m(X_з),$$

звідси:

$$m(E) \le \frac{m(X_з)}{m(X_g)}. \qquad (1.20)$$

Це рівень ІН, необхідний для виправлення $m(E)$ векторів помилок при довільному способі кодування. Також це співвідношення є критерієм попередньої оцінки потенційної коригуючої здатності конкретного способу представлення інформації відповідним джерелом. Ця формула є універсальною. Її можна застосовувати у випадку штучної ІН, наприклад при використанні завадостійких кодів. Значна частина методів контролю цифрових пристроїв, що розглядатимуться у наступному розділі, ґрунтується саме на їхньому використанні.

Запитання. Завдання

1. Проаналізуйте статистичний зміст надійності.
2. Охарактеризуйте загальні властивості функції ненадійності.
3. Поясніть, чим відрізняється частота від інтенсивності відмов.
4. Які основні періоди життєвого циклу технічних об'єктів? У чому полягає їх зміст?
5. Обґрунтуйте поняття «технічний ресурс виробу».
6. Як з фізичної точки зору пояснити те, що інтенсивність відмов електронних компонентів не змінюється з часом?
7. Чому для експоненціального розподілу терміни «напрацювання на відмову» і «середній час безвідмовної роботи» за назвою не зовсім відповідають їхньому змісту?
8. Назвіть і охарактеризуйте інші розподіли, що застосовують у теорії надійності.
9. На які два класи можна поділити всі технічні об'єкти з точки зору надійності? Які шляхи її підвищення для об'єктів цих класів?
10. Обґрунтуйте поняття «структурна надлишковість».
11. Поміркуйте, як впливає складність об'єкта на його надійність.
12. Охарактеризуйте поняття «коефіцієнт готовності». Поясніть, як можна його підвищити.
13. У чому різниця між поелементним і загальним резервуванням?
14. Охарактеризуйте поняття «резервування заміщенням».
15. Який алгоритм відновлення сигналів при застосуванні мажоритарного методу?
16. Як впливає поріг прийняття рішень при відновленні сигналів на надійність і достовірність?
17. У чому суть інформаційної надлишковості?
18. Обґрунтуйте поняття «природна інформаційна надлишковість». Наведіть її приклади.
19. Охарактеризуйте поняття «штучна інформаційна надлишковість».
20. Чим визначається здатність до виявлення або виправлення помилок?
21. Завдяки чому можна штучно збільшити рівень інформаційної надлишковості?

2.

Методи контролю цифрових пристроїв

У наш час існують різні види і методи контролю цифрових пристроїв. Однак найпоширенішими є методи функційного і тестового контролю. Основну увагу приділяють апаратним методам контролю компонентів обчислювальних пристроїв, розрахованих на вентильну технологію ІС.

2.1. Апаратні методи функційного контролю

Функційний контроль цифрових пристроїв може здійснюватися як апаратними, так і програмними методами. Апаратні методи є одними з найпоширеніших.

Суть і завдання функційного контролю

Найвідомішим контролем цифрових пристроїв є функційний (робочий) контроль.

Функційний *(лат. functio — виконання)* ***(робочий) контроль*** *— контроль, під час якого на об'єкт контролю подають робочі впливи.*

Під час функційного контролю визначення технічного стану об'єкта (працездатний чи непрацездатний) проводять безпосередньо в робочому режимі, подаючи на входи пристрою лише робочі сигнали, передбачені алгоритмом функціювання пристрою. Функційний контроль ставить такі основні завдання:

— забезпечити його максимальну повноту, тобто можливість виявляти більшість помилок, імовірних при функціюванні A;

— знайти такі реалізації апаратури, що дають змогу мінімізувати апаратні витрати.

Кінцева мета функційного контролю полягає у визначенні, чи правильно виконує об'єкт свої функції в даний момент.

На рис. 2.1 показана узагальнена схема функційного контролю, де A — контрольований пристрій; X, Y — вхідні і вихідні сигнали пристрою, KO — певний контрольний орган, вихідний сигнал якого $R = 0$, коли A функціює правильно, і $R = 1$, коли в Y з'являються помилки. A разом з KO іноді називають схемою із самоконтролем.

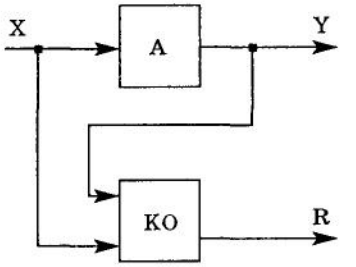


Рис. 2.1. Узагальнена схема функційного контролю

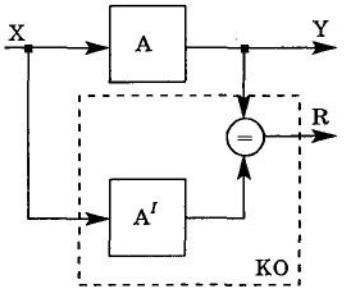


Рис. 2.2. Очевидний спосіб реалізації функційного контролю

Способом реалізації функційного контролю є схема на рис. 2.2, в якій функції A повторює дублікат A', після цього вихідні сигнали порівнюються для утворення сигналу помилки R. Така схема вимагає збільшення апаратних витрат удвічі, однак дає змогу виявляти всі помилки в контрольованому пристрої за винятком тих, що можуть одночасно виникнути в A'. Не виявляються тільки помилки, які виникають одночасно в A' і в тих самих розрядах, що і в A. Ймовірність цього за незалежної реалізації A і A' дуже мала. Незважаючи на вказані вади, цю схему застосовують на практиці завдяки високій ефективності з точки зору повноти контролю.

Пошук економічніших схем функційного контролю здійснюють за рахунок зменшення його повноти. Коли немає необхідності виявляти всі можливі помилки, замість повного дубліката A у складі KO можна використати його спрощену копію. Наприклад, для контролю арифметичних операцій часто застосовують контроль за модулем (рис. 2.3).

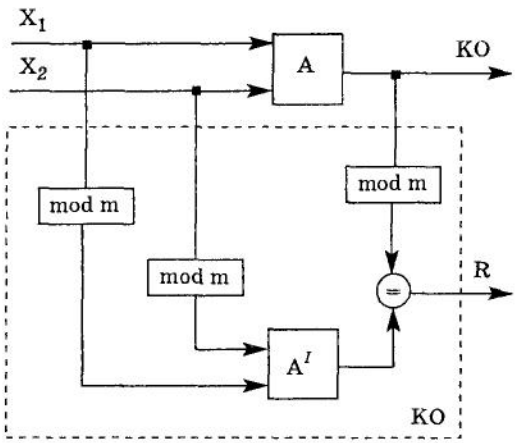


Рис. 2.3. Схема контролю за модулем

У цьому випадку в KO обчислюють остачі від ділення кожного з операндів X_1 і X_2 на певний модуль m. Потім над отриманими остачами виконують ту саму операцію, яку проводив пристрій A. Результат контрольованого пристрою Y теж згортається за модулем m, після цього результати обчислення в A' й отриману згортку порівнюють для утворення сигналу помилки.

Враховуючи, що A' працює зі значно меншою кількістю розрядів, складність KO може бути теж зменшена. Коли, наприклад, A виконує операції з 16- або 32-розрядними двійковими словами, то при $m = 7$ після згортки розрядність A' зменшується до 3, а при $m = 3$ — до 2. Контроль за модулем має обмежене застосування здебільшого арифметичними пристроями.

Виконання в KO спрощеної функції щодо функції пристрою A' заслуговує на увагу. Важливо уникати грубих помилок, які можуть вплинути на безпеку об'єкта управління, а помилки, які не впливають на безпеку (пов'язані з похибками давачів) можна ігнорувати. Із цією метою при реалізації A' використовують *усічений алгоритм*, за яким

обчислюють значення тієї самої функції, що й A', проте тільки над обмеженою кількістю старших розрядів, ігноруючи значення молодших. Складність KO при цьому зменшується, як і повнота контролю, тобто відбувається обмін повноти на складність. Тому необхідно ретельно аналізувати помилки щодо їх впливу на кінцевий результат обчислень.

Помилками, що принципово не можна виявити, не слід нехтувати. Йдеться про однотипні помилки, які можуть виникнути одночасно в A і A'. Щоб зменшити їх імовірність, бажано застосувати незалежну реалізацію A і A', але за змогою робити це порізному (вони не повинні бути однаковими).

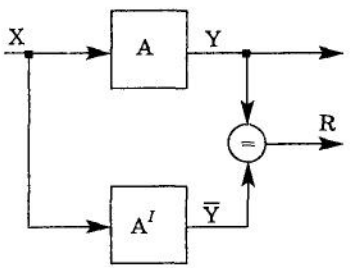


Рис. 2.4. Спосіб контролю зі схемою реалізації дублюючим пристроєм A′ інверсної функції

Припустимо, що дублюючий пристрій A' реалізує не ту саму функцію $Y = f(X)$, що A, а її інверсію, тобто для A' $Y = \overline{f(X)}$ (рис. 2.4) .

У цьому випадку A і A' будуть різними і ймовірність однотипних помилок зменшується. Такий *метод* отримав назву *«двопровідної логіки»*. На рис. 2.5 наведений приклад його застосування для простої комбінаційної схеми, що реалізує функцію:

$$y = x_1x_2 \vee \bar{x}_1x_3 \vee \bar{x}_2\bar{x}_3,$$

тоді:

$$\bar{y} = \bar{x}_1x_2\bar{x}_3 \vee x_1\bar{x}_2x_3.$$

Зміст сигналу помилки змінюється: $R = 0$ за наявності помилки і $R = 1$ за її відсутності.

У деяких випадках ефективний *метод «зворотної*

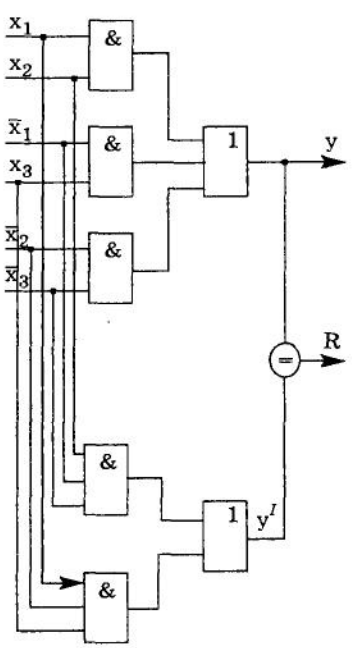


Рис. 2.5. Комбінаційна схема з дублюванням інформації методом «двопровідної логіки»

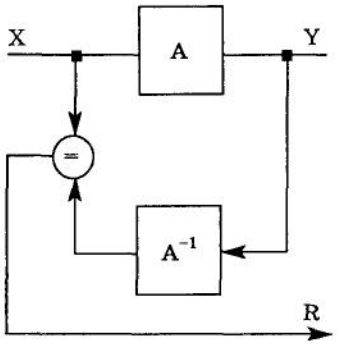


Рис. 2.6. Схема, що реалізує метод «зворотної машини»

машини» (рис. 2.6), який передбачає реалізацію зворотної функції:

$$X' = f'(Y)$$

і порівняння обчисленого значення вхідного сигналу з реальним значенням.

Необхідною умовою застосування цього методу є існування зворотного перетворення, тобто принципова можливість обчислити вхідні дані на основі вихідних. Вона існує для небагатьох перетворень. Навіть для арифметичних функцій немає зворотних перетворень (знаючи тільки суму, не можна знайти доданки, за добутком — множники і т. д.). Однак для деяких функцій це можливо, наприклад для піднесення у степінь зворотним є обчислення кореня. Для більшості тригонометричних функцій також легко знайти зворотні перетворення.

Класичним прикладом застосування методу «зворотної машини» є функційний контроль аналого-цифрового перетворення (АЦП), зворотним до якого є цифро-аналоговий перетворювач (ЦАП) (рис. 2.7). Оскільки ЦАП за складністю суттєво простіший за АЦП, такий контроль економний щодо апаратних витрат і виявляє практично всі можливі помилки АЦП.

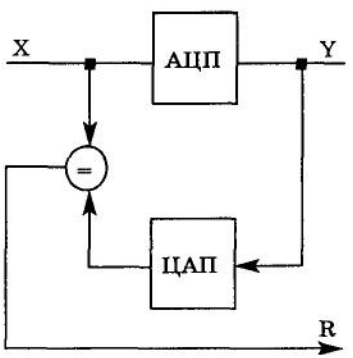


Рис. 2.7. Схема контролю АЦП методом «зворотної машини»

Метод «зворотної машини» досить універсальний. Його можна реалізувати і програмно, звичайно, лише в тих випадках, коли зворотне перетворення існує, тобто тоді, коли за результатом обчислень можна розрахувати вихідні дані й переконатись, що вони збігаються з тими, які були використані при обчисленнях.

Отже, арсенал методів функційного контролю великий. Це дає змогу змінювати як апаратні витрати, так і повноту контролю, що визначає його ефективність. Пошук оптимальних схемних рішень базується на ідеї селективності (лат. selectus — вибраний, вибірковість) контролю, тобто на обмеженні сукупності помилок тільки такими, які є найімовірнішими і найнебезпечнішими з точки зору впливу на об'єкт управління в комп'ютерних системах.

Тому треба аналізувати ті фізичні процеси, які відбуваються в компонентах пристрою і призводять до втрати працездатності. Первинними є несправності компонентів, а їх проявом — помилки на виходах пристрою. Враховуючи розгалужену топологію сучасних електронних пристроїв, однократні помилки здебільшого призводять на виходах пристрою до багатократних, і проявляються вони не на всіх вхідних сигналах. Щоб виявити цю специфічність і сформувати вимоги до засобів контролю, необхідно провести ретельний аналіз топології внутрішніх зв'язків контрольованого пристрою щодо поширення спотворених сигналів. Це дає змогу отримати множину найімовірніших помилок.

Наступним кроком оптимізації апаратних витрат на контроль є побудова *КО*, що виявляв би всі помилки зі сформованої множини. Для того щоб перейти до відповідних методів синтезу *КО*, необхідно розглянути основи теорії завадостійкого кодування.

Завадостійке кодування

Фундаментальним поняттям захисту від завад є інформаційна надлишковість. У подальшому важливий випадок, коли певну частину n-розрядних слів із загальної кількості $N = 2^n$ не використовують. Якщо кількість таких слів позначити через $N_з$ і назвати їх забороненими, то певні ймовірності будуть: $p_i = 0$ і $D > 0$. Підмножина слів, кількість яких N_g, — дозволені слова і використовуються джерелом інформації для кодування повідомлень. Тоді:

$$N = N_g + N_з.$$

Помилка в повідомленні може бути виявлена тільки в тому разі, коли вона перетворює дозволене слово на заборонене.

Для кожного дозволеного слова таких помилок — рівно $N_з$, а максимальна кількість усіх можливих при заданому способі двійкового кодування:

$$N_{E\max} = 2^n - 1,$$

де 1 віднесена до випадку, коли помилок немає. Відносну кількість помилок, яку можна виявити при заданому способі кодування, можна розрахувати як:

$$\delta = \frac{N_з}{2^n - 1}. \qquad (2.1)$$

Необхідно обчислити δ для таких випадків:

1. Код з однією перевіркою на парність. $N = 2^n$; $N_д = 2^{n-1}$ (дозволені слова мають парну кількість одиниць — їх половина).

$$\delta = \frac{2^n - 2^{n-1}}{2^n - 1} = \frac{2^{n-1}}{2^n - 1} \approx \frac{1}{2}\ (50\ \%).$$

2. Код, що має два додаткові розряди (код із контролем за модулем 3).

$$\delta = \frac{2^n - 2^{n-2}}{2^n - 1} \approx \frac{3}{4}\ (75\ \%);$$

$$\delta = \frac{2^5 - 10}{2^5 - 1} \approx 0{,}71\ (71\%).$$

3. Код із «повторенням». Кожне слово джерело передає двічі.

$$\delta = \frac{2^{2n} - 2^n}{2^{2n} - 1} = \frac{2^n(2^n - 1)}{2^{2n} - 1} \approx \frac{2^n \cdot 2^n}{2^n \cdot 2^n} = 1\ (100\ \%).$$

Отже, здатність коду до виявлення помилок збільшується зі зростанням надлишковості. Формула (2.1) універсальна, оскільки оцінювання здатності до виявлення помилок не залежить від способу кодування інформації.

У класичній теорії завадостійкого кодування за двійкового кодування застосовують підхід, який базується на визначенні кодової відстані та понятті «кратність помилок t».

***Кодова відстань між двома словами d** — кількість розрядів, за якими одне слово відрізняється від іншого.*

***Кратність помилок t** — кількість розрядів, спотворених помилкою.*

***Вектор помилки E = (e_1, e_2, ..., e_n)** — двійкова комбінація, яка містить одиниці в розрядах слова, що спотворені.*

Отже, кратність помилки — це кількість одиниць у векторі помилки, так звана вага вектора.

Легко довести, що мінімальна кодова відстань d для виявлення помилок кратності t і менше повинна бути:

$$d_{\min}^{\text{вияв}} = t_в + 1. \qquad (2.2)$$

При виконанні (2.2) жодна помилка кратності t і менше не зможе перетворити одне дозволене слово на інше дозволене.

Формули (2.1) і (2.2) у конкретних випадках еквівалентні. Це можна довести, взявши код з однією перевіркою на парність. Додавши один розряд таким чином, щоб кількість одиниць у слові стала парною, можна рознести слова на $d_{\min} = 2$ (досі було $d_{\min} = 1$). Будь-яка однократна помилка в цьому разі буде виявлятися, бо вона перетворює слово з парною кількістю одиниць на слово з непарною. Також виявлятимуться всі ті помилки, для яких t непарне, їх усього приблизно 50 %. В інших випадках потрібен більш детальний аналіз конкретних помилок, які виявляються і які не виявляються.

Якщо розглядати складніший випадок виправлення помилок, то слід з'ясувати, яка для цього необхідна надлишковість і найголовніше, що потрібно для того, щоб помилку можна було не тільки виявити, а й виправити.

Умова, необхідна для виявлення помилок, залишається справедливою і для їх виправлення: помилка повинна перетворювати дозволене слово на заборонене. Проте цього не досить. Якщо два чи більше дозволених слова можуть переходити в одне заборонене, то, отримавши це слово замість дозволеного (без помилок), отримувач не зможе визначити, яке саме дозволене слово було створено завадою. Щоб можна було це зробити, ймовірності помилок повинні бути суттєво різними. Тоді, отримавши заборонене слово, віддають перевагу тій помилці, у якої найбільша ймовірність. Однак у реальних умовах ймовірності не можуть вважатися відомими.

У теорії завадостійкого кодування існує твердження (це обґрунтовує теорія ймовірностей), що для незалежних помилок імовірність виникнення помилки дуже швидко зменшується зі зростанням її кратності, тобто найімовірніші однократні помилки, менш імовірні двократні й т. д. Множина можливих помилок завжди має бути обмежена або конкретною кратністю, або конкретним переліком.

Якщо обмеження виконано (а воно обов'язкове), то умову для виправлення помилок можна сформулювати так: кожна з помилок, що має бути виправлена, повинна переводити кожне з дозволених слів не більше, ніж в одне заборонене, а кількість таких помилок може бути визначена зі співвідношень (1.19, 1.20).

Для виправлення помилок певної кратності (частковий випадок) коригуючу здатність можна визначити через мінімальну кодову відстань:

$$d_{\min}^{k} = 2t_k + 1. \qquad (2.3)$$

З попереднього матеріалу може скластися враження, що, знаючи співвідношення (2.2) і (2.3), легко побудувати будь-який код. Для цього досить рознести дозволені слова на відповідну мінімальну відстань. Проте це не зовсім так. Якщо треба знайти, наприклад, кілька сотень дозволених слів ($N_g \approx 2^5 \ldots 2^{10}$), то за допомогою комп'ютера це можна зробити за кілька хвилин, отримавши таблицю дозволених слів, яку можна використати при декодуванні (виявленні або виправленні помилок). Однак, коли $N_g = 2^{20} \ldots 2^{100}$ або й більше, постає проблема збереження таблиці, якщо навіть її побудова можлива. (Це трудомісткий процес, оскільки завдання має комбінаторний характер.)

Головна мета класичної теорії кодування полягає у створенні конструктивних процедур кодування і декодування, що мають прості апаратні або програмні реалізації.

Класифікація кодів

З огляду на функційні особливості коди поділяють на групові, циклічні та коди БЧХ.

Групові коди. Теорія кодування вивчає головним чином так звані групові коди. Група — замкнена сукупність об'єктів щодо групової операції. Вона обов'язково містить також елемент, який називають одиничним або нульовим залежно від того, яку групову операцію використовують. Поняття «замкненість» полягає в тому, що застосування групової операції до двох чи більше елементів групи дає в результаті також елемент групи, а групова операція між будь-яким елементом групи і одиничним (нульовим) елементом дає сам елемент. Щоб проаналізувати ці визначення, необхідно навести прості приклади.

Усі дійсні числа — група відносно операції додавання (+) з нульовим елементом 0. Складаючи їх, завжди можна отримати теж дійсні числа (замкненість), а додавши 0 до будь-якого числа — те саме число. Щодо операції множення (·) з одиничним елементом 1, також є можливість отримати замкнену групу. Інший приклад: числа, що без остачі діляться на 3. Складаючи або перемножуючи їх, отримують числа, які теж без остачі діляться на 3.

У теорії кодування двійкові n-розрядні слова $X_i = (x_1, x_2, \ldots x_n)$ — це група відносно операції порозрядного додавання за модулем 2 ($\oplus$), а нульовий елемент — кодове слово $X_0 = (0,0,\ldots,0)$.

З математики відомо, що група може бути представлена своїм базисом. Базис — мінімальна сукупність незалежних елементів, застосовуючи до яких групову операцію (до різних поєднань елементів базису), можна відтворити всю групу.

$X_1 = 100$	$X_1 \oplus X_2 = 110$	100
$X_2 = 010$ $\Rightarrow$	$X_1 \oplus X_3 = 101$ $\Rightarrow$	010
$X_3 = 001$	$X_2 \oplus X_3 = 011$	001
Базис	$X_1 \oplus X_2 \oplus X_3 = 111$	110
		101
		011
		111
		Повна сукупність

Базис породжує повну сукупність елементів групи плюс нульовий елемент, що повинен бути в групі за визначенням. Базис є компактом групи. Наведений приклад ілюструє лише ідею представлення групи своїм базисом. Щоб задати всі 3-розрядні комбінації, не потрібно запроваджувати нові спеціальні поняття.

Візьмемо інший приклад:

100110		100110
010011	$\Rightarrow$	010011
001101		001101
Базис		110101
		101011
		011110
		111000
		000000
		Повна сукупність

(Задано вісім слів за допомогою лише трьох.)

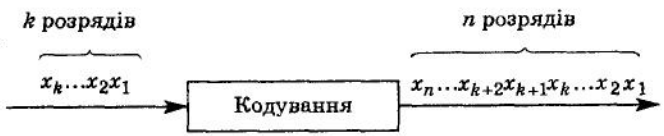
Базис зручно представляти (у цьому можна переконатися далі) відповідною матрицею. У даному випадку:

$$Q = \begin{Vmatrix} 100110 \\ 010011 \\ 001101 \end{Vmatrix} \qquad (2.4)$$

Таку матрицю називають породжуючою і майже завжди позначають символом Q.

***Породжуюча матриця Q кода** — матриця, рядки якої в сукупності утворюють базис групи, що відповідає множині всіх кодових слів.*

Процедура кодування полягає в доповненні вихідного слова певною кількістю додаткових (надлишкових або перевіркових) символів (розрядів). Позначивши кількість вихідних розрядів k, а кількість додаткових — $(n - k)$, отримують:



Значення перших k розрядів кодування повинні зберігатися. Це так звані *систематичні (роздільні) коди*. Тоді в породжуючій матриці Q перші k стовпців мають утворювати одиничну квадратну матрицю рангу k. Тільки так перші k розрядів кодової комбінації зможуть приймати довільні значення. Це необхідно для того, щоб не накладати будь-яких обмежень на інформацію, яка підлягає кодуванню. Таким чином:

$$Q = I_k A_{k,n-k}, \qquad (2.5)$$

де I_k — одинична квадратна матриця рангу k; $A_{k,n-k}$ — матриця розмірності $k(n - k)$.

Множину кодових слів задає Q, а кожен рядок Q — це теж кодове слово. Тому всі вимоги до множини кодових слів необхідно виконувати і для рядків Q. Якщо кодові слова повинні бути рознесені на відстань $d_{\min}$ одне від одного, то ця вимога виконується і для рядків Q. У даному прикладі як для Q, так і для повної множини кодових слів, породжених Q, $d_{\min} = 3$. Тобто, виконавши певні вимоги до Q, можна забезпечити задані властивості всієї множини кодових слів.

З точки зору математики процедура кодування еквівалентна матричному множенню довільного k-розрядного слова $X^{(k)} = (x_1, x_2, \ldots x_k)$ на породжуючу матрицю Q. Результат — добуток у вигляді n-розрядного слова $X^{(n)} =$

$= (x_1, x_2, \ldots x_k x_k, x_{k+1}, \ldots, x_n)$. Причому $X^{(k)}$ і $X^{(n)}$ треба розглядати як вектор-рядки, а операції множення і додавання виконують за правилами алгебри логіки:

x_i	x_j	$x_i \cdot x_j$	$x_i \oplus x_j$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Необхідно проаналізувати, що відбувається при такому множенні:

$$\|x_1, x_2, \ldots, x_k\| \cdot \begin{Vmatrix} 10 \ldots 0 a_{11} a_{12} \ldots a_{1,n-k} \\ 01 \ldots 0 a_{21} a_{22} \ldots a_{2,n-k} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ 00 \ldots 1 a_{k1} a_{k2} \ldots a_{k,n-k} \end{Vmatrix} = \|x_1', x_2', \ldots, x_k', x_{k+1}, x_{k+2}, \ldots, x_n\|,$$

де i-й розряд добутку — це скалярний добуток i-го розряду $X^{(k)}$ та i-го стовпця Q. Тобто:

$$\begin{aligned} x_1' &= x_1 \\ x_2' &= x_2 \\ &\ldots \\ x_k' &= x_k \end{aligned}$$

$$\begin{cases} x_{k+1} = a_{11} x_1 \oplus a_{21} x_2 \oplus \ldots \oplus a_{k1} x_k, \\ x_{k+2} = a_{12} x_1 \oplus a_{22} x_2 \oplus \ldots \oplus a_{k2} x_k, \\ \ldots \quad \ldots \quad \ldots \quad \ldots \\ x_n = a_{1,n-k} x_1 \oplus a_{2,n-k} x_2 \oplus \ldots \oplus a_{k,n-k} x_k. \end{cases}$$

Останні $(n - k)$ рівнянь називають *рівняннями кодування*. Вони визначають залежність значень додаткових розрядів від інформаційних. Кожне рівняння кодування можна записати як:

$$x_{k+i} = a_{1i} x_1 \oplus a_{2i} x_2 \oplus \ldots \oplus a_{ki} x_k, \quad (i = 1, 2, \ldots n - k)$$

або переписати так:

$$x_{k+i} + a_{1i} x_1 \oplus a_{2i} x_2 \oplus \ldots \oplus a_{ki} x_k = 0, \quad (i = 1, 2, \ldots n - k), \qquad (2.6)$$

якщо перенести доданок x_{k+i} з лівої частини в праву. Коли виникнуть помилки, рівняння (2.6) може не виконуватися, бо воно справедливе тільки за їх відсутності:

$$x_{k+i} \oplus a_{1i}x_1 \oplus a_{2i}x_2 \oplus \ldots \oplus a_{ki}x_k = r_i, \quad (i = 1,2,\ldots n-k), \qquad (2.7)$$

де $r_i = \begin{cases} 0, \text{ якщо помилок немає або вони не виявляються;} \\ 1, \text{ якщо виникла помилка.} \end{cases}$

Рівняння (2.7) є рівняннями перевірки, а r_i — результатом і-ї перевірки.

Сукупність $(n-k)$ результатів перевірки називається *перевірковою послідовністю* або *синдромом*:

$$R = (r_1, r_2, \ldots, r_{n-k}).$$

У математичній формі обчислення синдрому може бути записане як результат матричного множення n-розрядного слова на матрицю H, яку називають *перевірковою*. Спосіб її утворення:

$$Q = \left\| I_n A_{k,n-k} \right\|; \quad H = A^T_{k,n-k} I_{n-k},$$

де I_{n-k} — одинична квадратна матриця рангу $(n-k)$, $A^T_{k,n-k}$ — транспонована матриця $A_{k,n-k}$, тобто матриця, отримана з $A_{k,n-k}$ заміною рядків на стовпці:

$$A_{k,n-k} = \left\| \begin{matrix} a_{11}a_{12}\ldots a_{1,n-k} \\ a_{21}a_{22}\ldots a_{2,n-k} \\ a_{k1}a_{k2}\ldots a_{k,n-k} \end{matrix} \right\|; \quad A_{k,n-k} = \left\| \begin{matrix} a_{11}a_{21}\ldots a_{k1} \\ a_{12}a_{22}\ldots a_{k2} \\ a_{1,n-k}a_{2,n-k}\ldots a_{k,n-k} \end{matrix} \right\|$$

Перевіркова матриця H має $(n-k)$ рядків і n стовпців, а кожен рядок відповідає одній перевірці. Отже, результатом множення n-розрядного слова на H є саме синдром, тобто:

$$R = X^{(n)} \cdot H.$$

Тепер необхідно розглянути відповідність між помилками і синдромами. Спотворену помилкою кодову комбінацію можна записати так:

$$X' = X \oplus E,$$

де X — комбінація без помилок, її передало джерело інформації. Синдром, що відповідає X':

$$R = X' \cdot H = (X \oplus E) \cdot H = X \cdot H \oplus E \cdot H.$$

Однак за визначенням:

$$X \cdot H = 0 \text{ та } R = E \cdot H, \qquad (2.8)$$

тобто синдром для конкретної помилки E_i можна отримати, помноживши вектор помилки E_i на перевіркову матрицю.

Найімовірнішими є однократні помилки (помилки в одному розряді слова). Тому насамперед потрібно з'ясувати, яка відповідність між векторами однократних помилок і синдромами. Необхідно обчислити синдроми, що відповідають однократним помилкам у n-розрядному кодовому слові:

$$R = E \cdot H = \left\| \begin{matrix} e_1 \\ e_2 \\ \ldots \\ e_i \\ \ldots \\ e_n \end{matrix} \right\| \cdot \left\| \begin{matrix} \overbrace{\phantom{a_{11}a_{21}\ldots a_{k1}}}^{k} \overbrace{}^{n-k} \\ a_{11}a_{21}\ldots a_{k1}\ 10\ldots0 \\ a_{12}a_{22}\ldots a_{k2}01\ldots0 \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{1,n-k}a_{2,n-k}\ldots a_{k,n-k}00\ldots1 \end{matrix} \right\|$$

Оскільки E — вектор однократної помилки, то серед його компонентів є тільки одна одиниця. Це означає, що синдром помилки в i-му розряді кодового слова збігається з i-м стовпцем матриці H. Стовпці матриці H — це синдроми однократних помилок. Тому:

$$E = (e_1, e_2, \ldots, e_i, \ldots e_n) \Rightarrow R = (a_{i1}, a_{i2}, \ldots, a_{in}),$$

де лише $e_i = 1$.

Отже, вимоги до перевіркової матриці групового коду можна сформулювати так:

1) виявлення помилок:

— помилка E_i виявляється кодом, якщо $E_i \cdot H \neq 0$;

— перевіркова матриця коду, що виявляє всі однократні помилки, повинна мати n ненульових стовпців. Наприклад:

$$H = \|11\ldots1\| \text{ і відповідно } Q = \left\| \begin{matrix} 10\ldots01 \\ 01\ldots01 \\ \\ 00\ldots11 \end{matrix} \right\|$$

Це код з однією перевіркою на парність, що має один додатковий символ:

$$x_{k+1} = x_1 \oplus x_2 \oplus \ldots \oplus x_n.$$

Рівняння перевірки:

$$R = r = x_{k+1} \oplus x_1 \oplus x_2 \oplus \ldots \oplus x_n;$$

2) виправлення помилок:
якщо код виправляє сукупність помилок $E_1, E_2, \ldots, E_i, \ldots, E_e$, то всі відповідні цим помилкам синдроми повинні бути різними, тобто відповідність $E_i \leftrightarrow R_i$ має бути взаємно однозначною. Тільки в цьому випадку можна за значенням синдрому знайти відповідний йому вектор помилки і виправити помилку. З цього витікає, що перевіркова матриця коду, який виправляє всі однократні помилки, повинна мати n різних ненульових стовпців. Матрицю, в якій стовпці пробігають усі ненульові значення $(n - k)$-розрядних чисел:

$$H = \begin{Vmatrix} 000\ldots11 \\ 000\ldots11 \\ \ldots \\ 011\ldots11 \\ 101\ldots01 \end{Vmatrix} \qquad (2.9)$$

використовують як перевіркову коду, який виправляє всі однократні помилки. Така матриця відрізняється від канонічної форми:

$$H = \left\| A^T_{k,n-k} I_{n-k} \right\| \qquad (2.10)$$

лише розташуванням стовпців. Якщо стовпці, які містять по одній одиниці, перенести праворуч і сформувати з них одиничну матрицю, то можна отримати форму (2.9). Така процедура еквівалентна лише перенумерації позицій символів і не є принциповою з точки зору властивостей коду.

Форма матриці (2.10) є цікавою з іншого погляду. По-перше, при її використанні синдром у двійковому вигляді вказує номер розряду, який спотворено:

E	$\Rightarrow$	R
100...00	$\Rightarrow$	00...01
010...00	$\Rightarrow$	00...10
001...00	$\Rightarrow$	00...11
...		...
000...01	$\Rightarrow$	11...11,

по-друге, саме у такій формі американський вчений та інженер Р. Хеммінг (1915—1998) у 1948 р. запропонував перші коди для виправлення помилок.

Коди Хеммінга поділяють на три класи.

1. Код з виявленням помилок непарної кратності з однією перевіркою на парність. Для нього $d_{\min} = 2$ і $H = \|111\ldots1\|$.

2. Коди з виправленням однократних помилок. Для них $d_{\min} = 3$ і $n - k = \log_2(n+1)$. (2.11)

Наприклад, код з $n = 7$ і $(n - k) = 3$:

$$H = \begin{Vmatrix} 0001111 \\ 0110011 \\ 1010101 \end{Vmatrix}$$

1234567

З перевіркової матриці можна викреслити певну кількість стовпців. Це не вплине на коригуючу здатність коду. Головне — не зменшувати кількість перевіркових розрядів. Такі коди називають *укороченими*.

3. Коди з виявленням помилок кратності $t = 1,2,3$ і $t = 2$ або виправленням помилок кратності $t = 1$. Для них $d_{\min} = 4$.

Саме коди третього класу використовують найчастіше. Щоб отримати код з $d_{\min} = 4$, досить перевіркову матрицю H коду з $d_{\min} = 3$ доповнити ще одним рядком з одиниць:

$$H = \begin{Vmatrix} a_{11} & a_{12}\ldots a_{1n} & 10\ldots0 \\ a_{21} & a_{22}\ldots a_{2n} & 01\ldots0 \\ \ldots & \ldots & \ldots \\ a_{n-k,1} & a_{n-k,2}\ldots a_{n-k,n} & 00\ldots1 \\ 1 & 1\ldots1 & 11\ldots1 \end{Vmatrix}$$

Додатковому рядку відповідає один $(n-k+1)$-й додатковий розряд і одне рівняння перевірки:

$$r_0 = x_1 \oplus x_2 \oplus \ldots \oplus x_{n+1}.$$

При декодуванні можна отримати $(n-k+1)$ результатів перевірки:

$$R = (r_1, r_2, \ldots, r_{n-k}) \text{ та } r_0.$$

Наступний етап декодування коду базується на аналізі результатів перевірки. Вони можуть бути такі:

— $R = (0,0,\ldots0)$; $r_0 = 0$ — помилок немає;

— $R \neq (0,0,\ldots0)$; $r_0 \neq 0$ — однократна помилка, яку можна виправити на основі відповідності $R \rightarrow E$;

— $R \neq (0,0,\ldots0)$; $r_0 = 0$ — двократна помилка, що лише виявляється, але не виправляється.

Таким чином, можна отримати універсальний метод побудови кодів з $d_{min} = 2,3,4$, які виявляють і виправляють однократні помилки та виявляють двократні. Для інших випадків не існує універсальних методів, тобто придатних для довільних значень n або k, побудови кодів. У такому випадку код формують так: задають сукупність векторів помилок $E_1, E_2, \ldots, E_i, \ldots, E_N$. Далі необхідно знайти таку перевіркову матрицю H, для якої всі добутки $E_i \cdot H = R_i \neq 0$. У випадку виявлення заданої сукупності помилок та їх виправлення всі синдроми повинні бути різними, тобто:

$$R_i \neq R_j \text{ для всіх } i, j = 1, 2, \ldots, N.$$

Не існує іншого алгоритму, крім прямого перебору, побудови кодів, що коригують (виявляють або виправляють) довільну сукупність помилок. Тут корисною є така властивість: якщо вектору помилки E_i відповідає синдром R_i, а $E_j \to R_j$, то вектору помилки:

$$(E_i \oplus E_j) \to (R_i \oplus R_j). \qquad (2.12)$$

Ця властивість є наслідком лінійності всіх операцій, які застосовують у теорії кодування.

Завдання аналізу кодів, які виправляють помилки, досить складне, тому що потрібно аналізувати не окремі вектори помилки, а їх повну сукупність. Усі синдроми, що відповідають такій сукупності, повинні бути різними. Як цього досягти для однократних помилок — уже розглянуто. Для деяких інших сукупностей цю процедуру проводять за допомогою циклічних кодів.

Циклічні коди. Вони є підкласом групових кодів, але мають свою специфіку. Головним чином, це стосується математичного апарату і символіки, які використовують для їх побудови та аналізу.

Циклічний код — кільце, тобто замкнена сукупність елементів, для яких задані дві операції та зворотні до них.

У даному разі це порозрядне додавання $\oplus$ (зворотна операція збігається із самою операцією) і множення (зворотна — ділення). Самі елементи — двійкові n-розрядні комбінації — записують як поліноми. Правило, за яким встановлюють відповідність, можна пояснити прикладом:

$$\overbrace{1 \quad 1 \quad 0 \quad 1 \quad \ldots \quad 1 \quad \ldots 1 \; 0 \; 1}^{n \text{ розрядів}} \Rightarrow$$
$$n{-}1 \;\; n{-}2 \;\; n{-}3 \;\; \ldots \;\; \ldots \;\; i \;\; \ldots \;\; 2 \;\; 1 \;\; 0$$
$$\Rightarrow 1 \cdot x^{n-1} \oplus 1 \cdot x^{n-2} \oplus 0 \cdot 1 \cdot x^{n-3} \oplus \ldots \oplus x^i \oplus \ldots \oplus 1 \cdot x^2 \oplus 0x \oplus 1 \cdot x^0.$$

Більш компактно цей поліном можна записати так:

$$F(x) = x^{n-1} \oplus x^{n-2} \oplus \ldots \oplus x^i \oplus \ldots \oplus x^2 \oplus 1.$$

Такі поліноми додають, множать і ділять за звичайними правилами з урахуванням специфіки операції суми за модулем 2, а саме:

$$x^i \oplus x^i = 0; \quad x^i \oplus x^i \oplus x^i = x^i.$$

По-іншому можна дати таке (спрощене) визначення циклічного коду. Циклічний код утворює множина дозволених кодових комбінацій, відповідні яким поліноми без остачі діляться на поліном $P(x)$, який називають *породжуючим*.

Існує два способи утворення циклічних кодових комбінацій:

1) множенням довільної k-розрядної комбінації у вигляді відповідного їй полінома $F(x)$ ступеня $(k-1)$ на породжуючий поліном $P(x)$. Кодове слово може бути представлене як $F(x) \cdot P(x)$. Воно завжди без остачі ділиться на $P(x)$.

У цьому випадку отримують *несистематичний (нероздільний) код* — інформаційна частина $F(x)$ у кодовій комбінації не зберігається. Це суттєво ускладнює процедуру декодування за наявності помилок. Отже, з математичної точки зору цей спосіб найпростіший, але на практиці його застосовують рідко;

2) дописуванням до $F(x)$ з боку молодших розрядів остачі $R(x)$ від ділення $F(x) \cdot x^{n-k}$ на породжуючий поліном $P(x)$. (Множення на x^{n-k} — просте дописування до $F(x)$ $(n-k)$ нулів.) Це можна записати так: $F(x) \cdot x^{n-k} \oplus R(x)$. Отриманий таким чином поліном завжди без остачі ділитиметься на $P(x)$.

Будь-який циклічний код задають також породжуючою матрицею Q (бо це підклас групових кодів). Для цього досить знайти остачі для перших k рядків одиничної матриці I_n:

$$Q = \begin{Vmatrix} 10\ldots0R_1(x) \\ 01\ldots0R_2(x) \\ \\ 00\ldots1R_k(x) \end{Vmatrix}$$

Слід наголосити, що рядки матриці Q — це насамперед кодові слова, вони можуть бути утворені за звичайни-

ми правилами. Для прикладу можна взяти код з $P(x) = x^3 \oplus x \oplus 1$ і $k = 4$. Рядки одиничної матриці і відповідні операції:

$$1000 \to F_1(x) = x^3;\ F_1(x) \cdot x^3 = x^6 \to 101;$$

$$0100 \to F_2(x) = x^2;\ F_2(x) \cdot x^3 = x^5 \to 111;$$

$$0010 \to F_3(x) = x;\ F_3(x) \cdot x^3 = x^4 \to 110;$$

$$0001 \to F_4(x) = 1;\ F_4(x) \cdot x^3 = x^3 \to 011.$$

У результаті породжуючому поліному $P(x) = x^3 \oplus x \oplus 1$ для коду з $k = 4$ інформаційними розрядами відповідає породжуюча матриця:

$$Q = \begin{Vmatrix} 1000101 \\ 0100111 \\ 0010110 \\ 0001011 \end{Vmatrix}$$

Циклічний код з $P(x)$ і груповий код з Q — еквівалентні, а обчислення остачі $R(x)$ — подібне до обчислення синдрому R у групових кодах. Будь-якому циклічному коду можна знайти еквівалентний груповий код, але не навпаки, тому що клас групових кодів значно ширший, ніж клас циклічних.

На практиці саме циклічні коди використовують найчастіше. Це пояснюється їх простою апаратною (схемною) реалізацією на регістрах зсуву зі зворотними зв'язками та простим математичним апаратом, за допомогою якого можна описати циклічні коди та проаналізувати їх коригуючу здатність.

Для побудови циклічних кодів найчастіше використовують *незводимі поліноми* (аналог — прості числа). Не всі вони придатні для побудови ефективних кодів. Для отримання максимального числа різних остач при декодуванні породжуючий поліном $P(x)$ повинен бути дільником $x^n \oplus 1$. Варто розглянути деякі конкретні коди.

1. Код з $P(x) = x \oplus 1$. Він забезпечує $d_{\min} = 2$ і дає змогу виявляти всі однократні помилки, а також помилки непарної кратності. Це легко довести, аналізуючи подільність полінома, що відповідає вектору помилки $E(x) = x^i \oplus x^j \oplus \ldots \oplus x^z$ (кількість доданків непарна), на $P(x)$.

2. Код з $P(x) = x^3 \oplus x \oplus 1$. Цей поліном незводимий і є дільником $x^7 \oplus 1$, тому існує код з $n = 7$ та $(n - k) = 3$. Код з

такими параметрами дає $d_{\min} = 3$, тому він може виправляти однократні помилки або виявляти двократні. Такі самі характеристики має код з $P = x^3 \oplus x^2 \oplus 1$. У таблицях містяться інші поліноми для інших значень n і k. Наприклад:

n	k	$P(x)$
15	11	$x^4 \oplus x \oplus 1$
31	26	$x^5 \oplus x \oplus 1$
63	57	$x^6 \oplus x \oplus 1$
127	120	$x^7 \oplus x^3 \oplus 1$
255	247	$x^8 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1$
...	...	...

Усі коди оптимальні і є аналогами вже розглянутих кодів Хеммінга. В таблиці наведено лише по одному поліному для кожного значення n і k (насправді це може бути кілька еквівалентних варіантів).

3. Код з $d_{\min} = 4$ для виправлення однократних і виявлення двократних помилок. Такі коди отримують з попередніх, помноживши $P(x)$ на $(x \oplus 1)$. Тоді кількість перевіркових розрядів і довжина кодових слів збільшується на 1. Наприклад, з (31,26)-коду отримують (32,26)-код з породжуючим поліномом.

$$P'(x) = P(x) \cdot (x \oplus 1)(x^5 \oplus x \oplus 1)(x \oplus 1) =$$

$$= x^6 \oplus x^2 \oplus x \oplus x^5 \oplus x \oplus 1 = x^6 \oplus x^5 \oplus x^2 \oplus 1.$$

Цей код виправляє однократні та виявляє двократні помилки у 32-розрядних двійкових словах. Процедура декодування аналогічна відповідній процедурі для кодів Хеммінга з $d_{\min} = 4$. Нероздільність прийнятої комбінації на $P'(x)$ і роздільність на $(x \oplus 1)$ — ознака двократної помилки.

Коди БЧХ. Отримали назву від прізвищ їхніх авторів (Р. Боуз, Д. Рой-Чоудхурі та А. Хоквінгем). Коди БЧХ є узагальненням кодів Хеммінга і дають змогу виправляти кратні помилки ($t = 2,3,4,\ldots$). Породжуючий поліном для цих кодів утворюють як найменше загальне кратне певної сукупності незводимих поліномів (найчастіше — це добуток поліномів). Деякі з кодів БЧХ наведені в табл. 2.1.

Таблиця 2.1

Коди БЧХ

№ n/n	n	k	t	P(x)
1	15	7	2	$x^8 \oplus x^7 \oplus x^6 \oplus x^4 \oplus 1$
2	15	4	3	$x^{10} \oplus x^8 \oplus x^5 \oplus x^4 \oplus x^2 \oplus x \oplus 1$
3	31	21	2	$x^{10} \oplus x^9 \oplus x^8 \oplus x^6 \oplus x^5 \oplus x^3 \oplus 1$
4	31	16	3	$x^{15} \oplus x^{11} \oplus x^{10} \oplus x^9 \oplus x^8 \oplus x^7 \oplus x^5 \oplus x^3 \oplus \oplus x^2 \oplus x \oplus 1$
5	63	51	2	$x^{12} \oplus x^{10} \oplus x^8 \oplus x^5 \oplus x^4 \oplus x^3 \oplus 1$
6	63	45	3	$x^{18} \oplus x^{17} \oplus x^{16} \oplus x^{15} \oplus x^9 \oplus x^7 \oplus x^6 \oplus x^3 \oplus \oplus x^2 \oplus x \oplus 1$
7	63	36	5	$x^{27} \oplus x^{22} \oplus x^{21} \oplus x^{19} \oplus x^{18} \oplus x^{17} \oplus x^{15} \oplus x^8 \oplus \oplus x^4 \oplus x \oplus 1$

Загалом кодові методи функційного контролю обчислювальної техніки є найбільш поширеними.

Кодові методи функційного контролю

У сучасних обчислювальних (комп'ютерних) пристроях і системах широко використовують вбудовані засоби контролю, які конструктивно нероздільні із засобами, що виконують основні обчислювальні функції. Великого поширення у вбудованих засобах контролю набули кодові методи функційного контролю. Серед них найбільш ефективними і використовуваними є методи функційного контролю, що базуються на завадостійких кодах.

Насамперед необхідно розглянути традиційну схему використання завадостійких кодів у системах зв'язку (рис. 2.8).

Рис. 2.8. Схема використання завадостійких кодів у системах зв'язку

У повідомлення, які генеруються джерелом, за допомогою кодера повідомлень додається певна кількість надлишкових (перевіркових) розрядів, які забезпечують можливість виявлення або (і) виправлення помилок. Кодове слово $X = (X_1, X_2, \ldots, X_k, X_{k+1}, X_{k+2}, \ldots, X_n)$ надходить до кодера каналу (наприклад, модуляція радіочастотою). Сформовані таким чином сигнали передаються каналом, у якому діють завади, що може спричинити спотворення сигналів. Після отримання повідомлення декодер каналу проводить зворотне перетворення сигналів на електричному рівні (наприклад, демодуляцію), після чого в повідомленні декодер отримувача виявляє або (і) виправляє помилки. У цій схемі увага акцентується насамперед на функції кодера повідомлень і декодера отримувача, які майже не пов'язані з електричними трансформаціями сигналів.

У наведеній схемі канал передавання інформації (телефонна лінія, коаксіальний кабель, радіоканал, оптоволокно тощо) не змінює кодове слово X, якщо не враховувати впливу завад. Тому коригуючі властивості, закладені в повідомлення при кодуванні, не змінюються (кодова відстань між окремими словами, яка забезпечує можливість корекції, залишається незмінною).

Інша річ, коли за цією схемою необхідно використати завадостійкі коди в цифрових пристроях. У цьому випадку кодове слово після проходження через будь-яку цифрову схему зміниться так, що закладені при кодуванні коригуючі властивості зникнуть або зміняться (передусім порушаться кодові відстані між словами).

Для контролю і корекції помилок на виходах цифрових пристроїв застосовують схему (рис. 2.9, а), що узагальнює більшість розглянутих раніше методів. Так, схема апаратного самоконтролю з порівнянням виходів (рис. 2.9, б) та мажоритарне відновлення сигналів (рис. 2.9, в) можуть бути представлені цією універсальною структурою.

Відмінність полягає лише в класі завадостійких кодів, які використовують. Для функційного контролю — коди з виявленням помилок, для створення пристроїв, нечутливих до відмов компонентів, — коди з виправленням помилок. Саме тому кодовий підхід є найуніверсальнішим, він узагальнює більшість інших підходів, оскільки з'являється можливість варіації повноти контролю або рівня надійності шляхом вибору відповідних кодів (з більшою або меншою коригуючою здатністю).

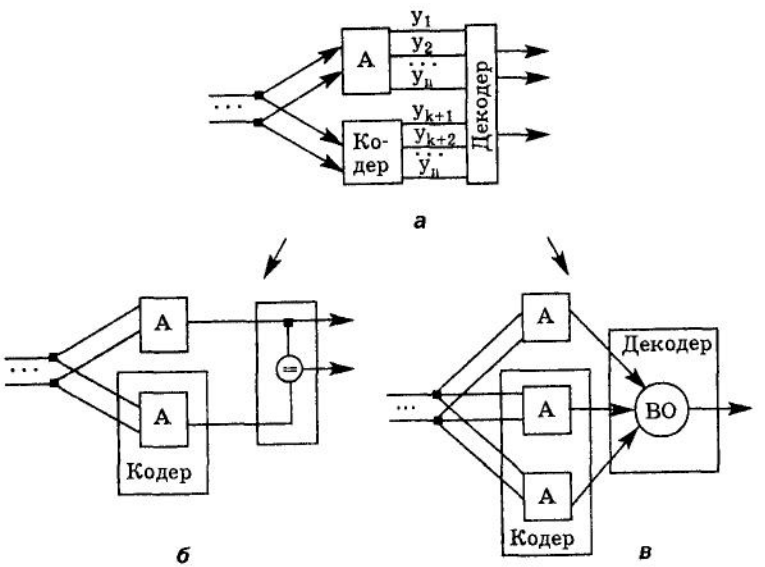


Рис. 2.9. Схеми контролю і корекції помилок на виходах цифрових пристроїв

Для виправлення помилок можна застосувати код Хеммінга з $k = 4$ інформаційними розрядами і $(n - k) = 3$ перевірковими, які задають рівняннями кодування:

$$y_5 = y_1 \oplus y_2 \oplus y_4;$$
$$y_6 = y_1 \oplus y_3 \oplus y_3;$$
$$y_7 = y_2 \oplus y_3 \oplus y_4.$$

Зробивши аналогічну підстановку, можна отримати:

$$y_5 = \bar{x}_1 \bar{x}_3;$$
$$y_6 = x_1 x_2;$$
$$y_7 = (x_1 \vee \bar{x}_3)\bar{x}_2.$$

Це функції, що реалізує кодер. Декодування проводять за стандартними методами, що застосовують у техніці зв'язку.

За потреби посилити коригуючу здатність контролю збільшують кількість перевіркових розрядів, додавши, наприклад, ще один розряд y_8 як суму за модулем 2 всіх вихідних сигналів схеми, тобто:

$$y_8 = y_1 + y_2 + y_3 + y_4 + y_5 + y_6 + y_7.$$

Зробивши відповідну підстановку, отримують:

$$y_8 = \bar{x}_1 x_2 \vee \bar{x}_2 x_3 \vee x_2 \bar{x}_3.$$

Це пристрій, в якому можна не тільки виправляти однократні помилки, але й виявляти двократні. Слід звернути увагу на такі моменти.

По-перше, в усіх прикладах складність кодера не перевищує складності вихідного пристрою. Порівняно з мажоритарним методом це можна вважати суттєвим виграшем, але при цьому зменшилася коригуюча здатність до виправлення лише однократних помилок.

По-друге, для корекції можна вибрати інший код з іншими рівняннями кодування або коригуючою здатністю. Ця властивість кодового підходу найважливіша щодо оптимізації за критеріями мінімуму апаратних витрат або максимуму коригуючої здатності. Це створює сприятливі умови, за яких задані вимоги до надійності забезпечують мінімальними апаратними витратами.

Можна стверджувати, що при виявленні всіх можливих помилок на виходах пристрою необхідні апаратні витрати збільшують удвічі, а для виправлення — утричі. Зменшення витрат можливе лише за рахунок зниження коригуючої здатності і, як наслідок, надійності.

Реальні апаратні витрати значною мірою залежать від специфіки конкретного пристрою, до якого застосовують методи забезпечення надійності, елементної бази та інших технічних факторів. Порівнявши класичні методи забезпечення надійності (мажоритарний, функційний контроль (див рис. 2.2)) з кодовим, зрозуміло, що їх відрізняють тільки коди, які застосовують. У першому випадку (мажоритарний метод) — коди з повторенням, що мають потужну коригуючу здатність і відповідно вимагають великої надлишковості, у другому — коди з меншою коригуючою здатністю, але й з меншою надлишковістю. Який метод вибрати для конкретного пристрою, залежить від вимог до надійності та наявних ресурсів (допустимих габаритів, ваги, вартості).

Розглянутий матеріал і накопичений досвід розроблення високонадійної цифрової апаратури дають змогу зробити висновок, що саме на основі кодового методу можна забезпечити селективність і керованість уведення надлишковості. Це важливо у випадках комп'ютерного управління реальними об'єктами (саме в таких ситуаціях виникає проблема забезпечення надійності), де окремі помилки можуть бути найбільш небезпечними і навіть неприпустимими. Наприклад, коли людина піднімається ліфтом, то (y_1, y_2, ...) — команди на пуск маршового двигуна, від-

чинення (зачинення) дверей і т. д. Щодо вимог безпеки, неприпустиме одночасне відчинення дверей кабіни і пуск двигуна, який її рухає. Саме така помилка повинна бути принаймні виявлена, щоб заблокувати виконання відповідних команд. Інші помилки не є такими небезпечними, їх не обов'язково виявляти і тим паче виправляти. Адже все, що безпосередньо не загрожує життю людини, можна вважати вторинним. Отже, всі наявні ресурси (надлишковість — це зайві витрати) необхідно спрямовувати саме на найнебезпечніші помилки. Тому кодовий підхід, який надає можливість цілеспрямованого введення надлишковості й узагальнює більшість інших підходів, є найуніверсальнішим.

Крім функційного контролю, значне місце у визначенні працездатності цифрових пристроїв займає тестовий контроль.

2.2. Тестовий контроль

Для поглиблення достовірності контролю працездатності обчислювальних пристроїв інколи недостатньо перевірити об'єкт на виконання ним робочих функцій. Тому виникає необхідність контролю працездатності кожного елемента і компонента структури пристрою. Це можна здійснити за допомогою тестового контролю.

Суть і особливості тестового контролю

Суть тестового контролю полягає в тому, що тестування проводять у спеціальному режимі, коли об'єкт не виконує своїх робочих функцій.

Тестовий (англ. test — випробування) контроль — контроль, під час якого на об'єкт контролю подають тестові впливи.

Для цифрових (мікропроцесорних) пристроїв і систем тестові впливи — це сукупність наборів логічних стимулюючих сигналів (логічна «1» і логічний «0»), що послідовно подають у вхідні контрольні точки (контакти) в заданих інтервалах часу (тактах) і призначені для контролю правильності функціювання елементів і компонентів цих пристроїв. Мета тестування полягає у виявленні наявності або відсутності несправностей і визначенні технічного стану об'єкта. Результатом тестування є висновок: «Об'єкт працездатний або непрацездатний» або «в об'єкті несправний блок (модуль) №...».

Основна ідея, на якій ґрунтується тестовий підхід, полягає в переході від перевірки правильності виконання

об'єктом своїх функцій (функційний контроль) до перевірки технічного стану обладнання, тобто наявності або відсутності несправностей.

Необхідно з'ясувати, чому ця відмінність є принциповою. Визначення технічного стану об'єкта на основі перевірки правильності виконання ним своїх функцій для більшості цифрових пристроїв практично не може бути реалізоване.

Для прикладу можна перевірити правильність виконання функції складання звичайним калькулятором. Необхідно складати по черзі всі можливі доданки, наприклад, довжиною 8 десяткових розрядів. Це значить, що слід виконати 10^{16} додавань тільки для того, щоб остаточно переконатися в правильності виконання калькулятором тільки функції додавання. Нескладний підрахунок показує, що якщо на кожне складання витрачати 1 с, то вся перевірка займе $3 \cdot 10^8$ років. Зрозуміло, що цю процедуру можна проводити і в автоматичному режимі. Тоді при швидкості 10^8 операцій/с вона буде здійснена лише за 3 с.

Проте ситуація ускладнюється, коли необхідно протестувати сучасний комп'ютер або комп'ютерну мережу.

Перед тим як перейти до методів побудови тестів, необхідно з'ясувати вплив наявності несправності на функції цифрового пристрою. На рис. 2.10 показана вентильна реалізація схеми «І» (а — справної схеми, б — схеми, в якій у колі змінної x_1 є обрив).

З табл. 2.2 видно, що несправність змінює функцію, яку реалізує схема. Щоб її виявити, не перебираючи на входах усі вісім наборів змінних, досить подати набір $(x_1x_2x_3) = (011)$. Якщо $y = 0$, то цієї конкретної несправності немає, а якщо $y = 1$, то є. Набір 011 — це тест, який виявляє задану несправність, його називають *перевірковим (контролюючим)*.

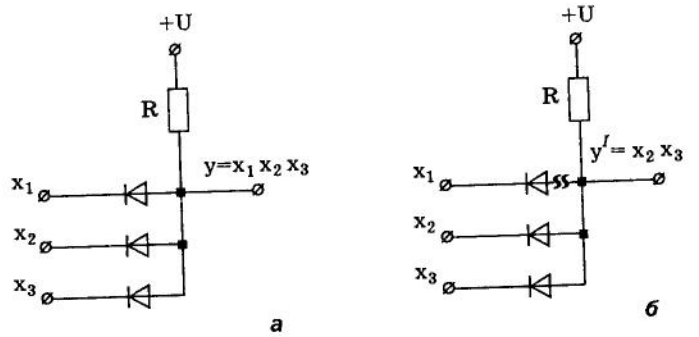


Рис. 2.10. Вентильна реалізація схеми «І»

Таблиця 2.2

Тести схеми «І»

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

Тепер слід розглянути аналогічний приклад для схеми «АБО» (рис. 2.11 і табл. 2.3).

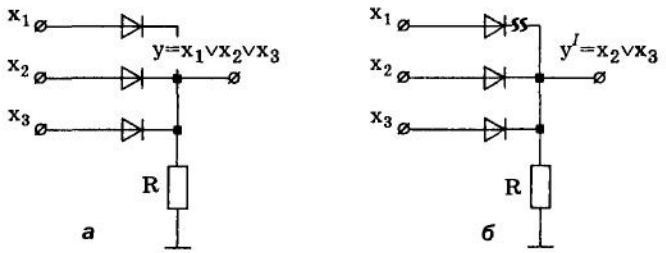


Рис. 2.11. Вентильна реалізація схеми «АБО»

Таблиця 2.3

Тести схеми «АБО»

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

Для цієї схеми і заданої несправності (обрив у колі x_1) перевірковим тестом буде набір (100). Для інших несправностей неважко знайти відповідні тестові набори, а потім об'єднати їх в одну тестову послідовність, що називається тестом. Здебільшого один тестовий набір виявляє більше, ніж одну несправність, тому довжина загальної тестової

послідовності (тесту) практично завжди значно менша суми тестових наборів.

Перевірковий тест для заданої несправності Li *— вхідний вплив, при якому вихідна реакція об'єкта на нього різна за наявності й відсутності* L_i.

Цьому визначенню цілком відповідає набір 011 для схеми «І» та 100 для схеми «АБО».Кожен тестовий набір будують для кожної заданої несправності.

Важливими є не тільки наявність чи відсутність несправностей, але й те, яка саме несправність має місце.

У цих випадках застосовують *діагностичні тести* (тести, для визначення помилок функціювання апаратури), ідея побудови яких така сама, але вихідні реакції об'єкта для різних несправностей різні. Отже, для побудови тестової послідовності необхідно задати (перелічити) конкретні несправності, які вона повинна виявляти.

Це можливо, коли несправності досить прості, наприклад короткі замикання на входах чи виходах схеми або навіть в електрорадіоелементах, що не мають безпосередніх електричних зв'язків ні з входами, ні з виходами. А якщо це великі інтегральні схеми (ВІС), де кількість вентилів може сягати десятків тисяч, враховуючи інші типи несправностей (зміна провідності, пробої діелектричних шарів, коливання затримок при проходженні сигналів тощо), то кількість несправностей, яку необхідно задати при побудові тесту, занадто велика. Крім того, несправності можуть бути не тільки однократні, але й двократні, трикратні й т. д.

У таких ситуаціях використовують *алгоритмічний підхід*. Принципова перевага його перед іншими методами полягає в тому, що всі задані несправності виявляють тестовою послідовністю на 100 %.

Проте здебільшого несправний елемент знаходиться всередині пристрою або компонента (ІС) і до його входів та виходів доступу не існує, тобто немає змоги безпосередньо подати на входи тестові сигнали і спостерігати реакцію елемента на них.

Основою побудови тестів є процедура *активізації шляхів* до недоступних входів і виходів об'єкта. Її пояснює рис. 2.12, де зображений фрагмент пристрою, в якому на вентиль B_i подають тестові сигнали.

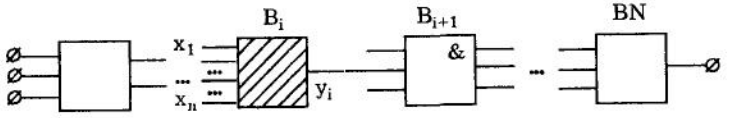


Рис. 2.12. Фрагмент пристрою, в якому немає безпосереднього доступу до вентилів

До виходу B_i немає безпосереднього доступу. Значення сигналу y_i на ньому можна спостерігати лише крізь елементи B_{i+1}, B_{i+2}, ..., B_N. Тому треба забезпечити їх так звану прозорість, тобто будь-яка зміна сигналу y_i повинна транспортуватись без перешкод до кінцевого виходу пристрою, доступного для спостереження. Це можна зробити, подавши на входи B_{i+1}, B_{i+2}, ..., B_N відповідні сигнали, значення яких визначають з рівняння (для вентиля B_{i+1}):

$$\frac{df(y_1,\dots y_i,\dots y_n)}{dy_i}=1,$$

тоді:

$$\frac{df(y_1,\dots y_i,\dots y_n)}{dy_i}=f(y_1,\dots y_i,\dots y_n)\oplus f(y_1,\dots \bar{y}_i,\dots y_n).$$

Це булева похідна по змінній y_i функції $f(y_1,\dots y_i,\dots,y_n)$, яку реалізує B_{i+1}.

Для елемента «І» (див. рис. 2.10):

$$f(y_1,\dots y_i,\dots y_n)=y_1y_2\dots y_i\dots y_n;$$

$$\frac{df(y_1,\dots y_i\dots y_n)}{dy_i}=y_1\dots y_i\dots y_n\oplus y_1\dots\bar{y}_i\dots y_n=y_1\dots y_{i-1}y_{i+1}\dots y_n=1.$$

Розв'язок рівняння:

$$y_1=1;\ y_2=1;\ \dots\ y_{i-1}=1;\ y_{i+1}=1;\ \dots\ y_n=1,$$

тобто для прозорості елемента «І» на всі його входи (крім y_i) потрібно подати одиничні сигнали.

Для елемента «АБО» (див. рис. 2.11):

$$y_1=0;\ y_2=0;\ \dots\ y_{i-1}=0;\ \ y_{i+1}=0;\ \dots,\ y_n=0.$$

Провівши аналогічні обчислення і подавши відповідні сигнали на B_{i+1}, B_{i+2}, ..., B_N, можна забезпечити прозорість усього шляху до кінцевого виходу для інформації від елемента, який перевіряють. Це і є активізація шляху $B_i, B_{i+1}, \dots, B_N$.

Наступний крок — забезпечення подачі на входи B_i необхідних тестових сигналів. Проте до цих входів немає безпосереднього доступу. Проблему розв'язують визначенням функцій, які формують відповідні сигнали для B_i, і довизначенням їх для тих входів, які активізують шляхи до B_i.

Найпопулярнішим серед алгоритмічних методів до цього часу залишається *d-алгоритм*. У його основі — активізація всіх можливих шляхів від місця несправності до всіх доступних для спостереження виходів пристрою. Цей алгоритм є складовою більшості програмних пакетів, призначених для побудови тестів.

Будь-який алгоритмічний метод принципово потребує переліку несправностей, які необхідно виявити. Ця вимога може бути виконана лише щодо так званих *константних* (лат. constans — незмінний, постійний) *несправностей*, тобто таких, які еквівалентні заміщенню булевих змінних константами 0 або 1, а на фізичному рівні — обривам або коротким замиканням.

Оскільки несправності інших типів для сучасних інтегральних компонентів мають далеко не нульову ймовірність, то застосування алгоритмічних методів обмежується здебільшого пристроями на дискретній (лат. discretus — поділений, переривчастий) елементній базі, релейними (франц. relayek — замінити) схемами тощо. Тести, побудовані за такими методами, можуть бути використані як базові, які в процесі застосування доповнюють додатковими тестовими наборами, що виявляють не виявлені несправності.

Тести є програмним продуктом, і фірми, що постачають їх на ринок, зі зрозумілих причин не розголошують методи побудови найефективніших з них. У конкретних випадках (для апаратури певного класу) існує багато варіантів розв'язання цієї проблеми на основі інтуїтивних підходів, комп'ютерного моделювання несправностей і експериментального пошуку вхідних сигналів, що їх виявляють. Можна вважати, що завдання побудови тестів залишається відкритим.

Імовірнісний метод тестового контролю

З появою ВІС із функціями рівня мікропроцесора стало очевидним, що алгоритмічний підхід не може забезпечити необхідну ефективність діагностування пристроїв такої і вищої складності. Це стало очевидним, коли значну частину несправностей при тестуванні у виробничих умовах не можна було виявити, що призводило до невиявлення дефектних виробів. Вади алгоритмічних методів стимулювали пошук нових підходів до тестового контролю цифрових пристроїв.

Основна ідея цих методів ґрунтується на пошуку інтегральних ознак працездатності, які не обов'язково пов'язані з виконанням об'єктом своїх функцій. Прикладом такого інтегрального підходу є *імовірнісний метод*. Суть його

полягає в тому, що на входи об'єкта подають випадкові двійкові послідовності, які мають фіксовану ймовірність появи 0 і 1 — P_0 і P_1 (це еквівалентно відносній частоті появи 0 і 1). Будь-який цифровий пристрій є одночасно і перетворювачем імовірностей p_0 і p_1. Враховуючи, що $p_0 = 1 - p_1$, можна записати:

$$P_{i0}^{вих} = F(P_0), i = 1,2,...,m,$$

де $P_{i0}^{вих}$ — ймовірність появи 0 на i-му виході пристрою, $F(P_0)$ — певна цілком визначена арифметична функція, що залежить від логічних функцій, які виконує пристрій, m — кількість виходів пристрою. Оскільки будь-яка несправність змінює логічні функції пристрою (якби це було не так, то її не можна було б виявити), то, фіксуючи $P_{i0}^{вих}$, визначають наявність (відсутність) несправності.

Наприклад, якщо на входи схеми «І» подавати сигнали з $p_0 = p_1 = \frac{1}{2}$, то на виході:

$$P_i^{вих} = \frac{1}{4} \text{ і } P_0^{вих} = \frac{3}{4}.$$

Це можна довести, аналізуючи істинність цієї функції:

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

На кожні чотири рівноймовірні набори вхідних змінних (00,01,10,11) схема видає одну 1.

Для схем реальної складності можна скористатися відповідними формулами алгебри логіки для трансформації логічних функцій в арифметичні та знайти залежності відносних частот 0 і 1 у вихідних змінних від частот 0 і 1 у вхідних змінних. Реалізація цього методу вимагає застосування нескладного обладнання: генератора послідовностей з фіксованими частотами появи 0 і 1 та лічильників, які підраховують кількість 1 за задані проміжки часу.

Як і будь-який статистичний метод, імовірнісне тестування вимагає досить довгих вхідних послідовностей, щоб забезпечити нечутливість до незначних відхилень

від імовірнісних розподілів, а також невисоку роздільну здатність до деяких типів несправностей. Крім того, існують несправності, які не змінюють вихідну статистику 0 та 1 і тому не можуть бути виявлені імовірнісним тестуванням.

Ось чому цей метод є лише проміжним етапом у пошуку ефективних «неалгоритмічних» методів тестування.

Сигнатурний метод тестового контролю

Наступним етапом у розробленні і розвитку нових методів тестування слід вважати метод, який отримав назву компактного тестування або *сигнатурного аналізу*.

Наприклад, для комп'ютера ефективним тестом може бути будь-яка велика (складна) програма обчислень з відомим результатом, отриманим залежним способом. Справді, ймовірність того, що на несправному комп'ютері можна отримати правильний результат, який вимагає багатьох звертань до пам'яті, використання підпрограм та опцій, дуже мала. Взявши для прикладу таке екзотичне завдання, як комп'ютерна оркестровка музичного твору для оркестру із заданим складом інструментів, можна стверджувати, що будь-яку помилку в результаті виконання програми помітить музикант-професіонал. Те саме стосується і складного математичного обчислення, комп'ютерного оброблення графічних матеріалів тощо.

Якщо подивитись на електричну схему аналогової апаратури (підсилювача, генератора сигналів, телевізора, радіоприймача) (рис. 2.13), то можна помітити, що вона має позначки, які вказують напруги, струми, осцилограми сигналів тощо (у даному випадку — напруги). Це значно полегшує пошук несправностей.

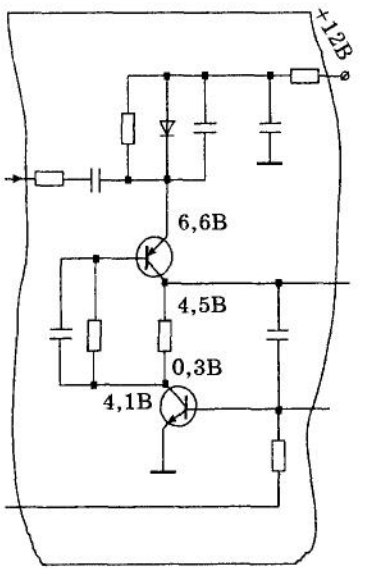


Рис. 2.13. Фрагмент електричної схеми аналогового пристрою

Майстер, який може навіть не знати принципу роботи пристрою, перевіривши значення напруг, струмів і т. д. у контрольних точках, має змогу локалізувати несправність. Це буде переважно той елемент схеми, сигнали на виходах якого не відповідають нормі при нормальних, правильних значеннях вхідних сигналів. Необхідно наголосити на найсуттєвішому: при такому підході до кваліфікації майстра не ставлять високих вимог. Це дуже важливо при обслуговуванні сучасної електронної техніки, враховуючи, що дефіцит кваліфікованого персоналу зростає.

Аналогічний підхід при пошуку несправностей у цифрових схемах не використовують, тому що в будь-якій точці схеми можна побачити і зафіксувати лише високий або низький потенціал у статичному (грец. statos — нерухомий) режимі, а в динамічному (грец. dunamikos — сильний) — послідовності безперервних змін високого і низького потенціалів, які важко відрізнити один від одного.

Розглянутий матеріал дає змогу визначити основну суть сигнатурного аналізу. Вона полягає в тому, що на входи цифрового пристрою подають послідовність нулів та одиниць, вибрану випадково, але щоразу одну й ту саму (тому її називають *псевдовипадковою*). За такої ситуації в будь-якій точці пристрою фіксують послідовність нулів та одиниць, що залишатиметься однаковою при повторенні експерименту. Завдяки цьому визначають, яка послідовність відповідає справному технічному стану об'єкта. Її вважають ознакою, що несправності немає, а за наявності несправності вона буде спотворена.

Отже, з'ясовано ситуацію, яку використовують для пошуку несправностей в аналогових схемах. Для застосування цього підходу необхідно виконати такі умови:

1. Вхідна псевдовипадкова послідовність повинна бути достатньо довгою для того, щоб будь-яка несправність встигла проявити себе. Для реальних об'єктів довжина послідовності має порядок $2^{20}...2^{30}$, тобто приблизно $10^6...10^{10}$ біт. Тому будують апаратний або програмний генератор псевдовипадкових послідовностей і звідси подають їх на об'єкт при діагностуванні пристрою.

2. Проводячи діагностичний експеримент за відсутності несправностей, необхідно зафіксувати для подальшого використання ті послідовності, які виникають у контрольних точках. Їх довжина має той самий порядок, що й довжина вхідної послідовності. Це еталонна ознака працездатності об'єкта.

Існує багато способів генерації псевдовипадкових послідовностей, розрахованих на апаратну або програмну реалізацію. Однак найпопулярнішим і найпоширенішим є спосіб генерації за допомогою регістрів зсуву зі зворотними зв'язками за модулем 2. Теорія схем цього класу існує давно, а самі схеми широко використовують при кодуванні і декодуванні циклічних завадостійких кодів.

На рис. 2.14 зображений 4-розрядний регістр зі зворотними зв'язками, а в поданій разом зі схемою табл. 2.4 зведені його стани, що змінюються з кожним тактом. Повний цикл до повторення комбінації становить 15 різних двійкових слів. Глянувши на таблицю по вертикалі, можна дійти висновку, що кожен її стовпчик — псевдовипадкова двійкова послідовність, яку використовують як тестову. Головна її особливість у тому, що цю послідовність можна генерувати безліч разів і вона залишатиметься без змін, а кожен вихід можна з'єднати з одним з виходів пристрою, на який подають тестові впливи.

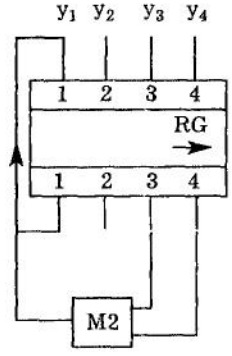


Рис. 2.14. 4-розрядний регістр зсуву зі зворотними зв'язками і сумато-ром за модулем 2

Таблиця 2.4

Послідовності, що генеруються регістром зі зворотними зв'язками

	y_1	y_2	y_3	y_4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	1	0	0	1
5	1	1	0	0
6	0	1	1	0
7	1	0	1	1
8	0	1	0	1
9	1	0	1	0
10	1	1	0	1
11	1	1	1	0
12	1	1	1	1
13	0	1	1	1
14	0	0	1	1
15	0	0	0	1
16	1	0	0	0

а

Послідовності, що генеруються лічильником

	y'_1	y'_2	y'_3	y'_4
1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	0	0
14	1	1	0	1
15	1	1	1	0
16	1	1	1	1

б

Залежно від початкового стану регістра (за прикладом — це 1000) будуть змінюватися і послідовності, що генеруються. Для 4-розрядного генератора таких різновидів 15 (див. табл. 2.4, а). Отже, цю процедуру можна провести за допомогою дуже простих апаратних засобів.

Зворотні зв'язки в регістрі зсуву вибирають відповідно до конкретного полінома, який забезпечує генерацію послідовностей максимальної довжини. Він повинен бути дуже примітивним, або незводимим (так само, як і для утворення циклічних кодів). Ступінь полінома m відповідає довжині регістра, а довжина послідовності — $2^m - 1$. Зі зростанням m кількість таких поліномів швидко збільшується і визначається функцією Ейлера:

$$\Phi(m) = \frac{2^m - 1}{m}.$$

Здебільшого обирають поліноми з мінімальним числом членів (доданків). У даному прикладі використаний поліном $Q(x) = x^4 + x + 1$. Перевага такого методу формування псевдовипадкових послідовностей полягає в дуже простій апаратній і програмній реалізації відповідних алгоритмів. Генератор містить лише m-розрядний регістр зсуву і набір суматорів за модулем 2 в колі зворотного зв'язку. Сам регістр виконує функції запам'ятовування m-розрядної послідовності та її зсуву на один розряд праворуч. Суматори за модулем 2 в колі зворотного зв'язку обчислюють значення чергових символів, які послідовно записують у лівий (молодший) розряд регістра.

Наприклад, для реалізації 32-розрядного генератора псевдовипадкових послідовностей знадобиться лише регістр відповідної розрядності та 4—5 двовходових суматори за модулем 2. При реалізації на сучасній елементній базі це лише одна ІС середнього рівня інтеграції. Водночас довжина послідовності, яка генерується таким пристроєм, складає понад 10^{10} 32-розрядних комбінацій.

На користь регістра зсуву зі зворотними зв'язками свідчить і порівняння зі звичайним двійковим лічильником. Подавши на його вхід тактові імпульси, отримують послідовність m-розрядних комбінацій довжиною 2^m. Послідовності, які генеруються лічильником, можна порівняти (див. табл. 2.4, б). Вони мають певну регулярність, не є випадковими. Можна, наприклад, побачити, що 0 і 1 в молодших розрядах розташовані через один, у другому молодшому розряді після 00 обов'язково йде комбінація 11 і т. д. Послідовності на виходах регістра зсуву не випадкові, але їх структура (періодичність появи, наприклад, 2-х, 3-х і т. д. одиниць) змінюється. Тому вважають, що послідовності, які генеруються регістром зсуву зі зворотними зв'язками ближче до випадкових. Важливо також, що їх імовірнісні характеристики, тобто частоти появи тих чи інших сполучень 0 і 1, легко змінюються відповідним вибором зворотних зв'язків регістра. До того ж різноманіття послідовностей, які генеруються регістром зі зворотними зв'язками, суттєво більше, ніж кількість різних послідовностей, які можна створити за допомогою лічильника.

Щодо проблеми фіксації реакцій об'єкта в контрольних точках при подаванні на його входи псевдовипадкових послідовностей, слід зауважити, що фіксувати всю послідовність, яка виникає в певній точці схеми, недоцільно. Передусім з точки зору складності реалізації (вони занадто довгі), а також через величину надлишковостей таких послідовностей. Тому їх слід скоротити (або згорнути).

Існує безліч способів такої згортки. Найпоширенішим є обчислення остачі від ділення полінома, що відповідає довгій послідовності, на поліном суттєво коротший (наприклад, на той, що використовують при генерації псевдовипадкової послідовності). Саме ця остача і є *сигнатурною*, а метод пошуку несправностей за допомогою сигнатур отримав назву сигнатурного аналізу.

Пристрій, який здійснює обчислення остачі, називають *сигнатурним аналізатором*. Уперше його застосувала фірма «Hewlett-Packard». Узагальнена схема пристрою для згортки послідовності $Y = y_1, y_2, \ldots, y_t, \ldots, y_N$ має структуру, зображену на рис. 2.15, і подібна до структури для генерації псевдовипадкових послідовностей (див. рис. 2.14). Управління аналізатором здійснюють команди «Старт», «Зсув», «Стоп».

Сигнали «Старт» і «Стоп» формують часовий інтервал, протягом якого здійснюється процедура стиснення. Крім того, команда «Старт» встановлює початковий стан регістра, здебільшого він нульовий. Робочий режим по-

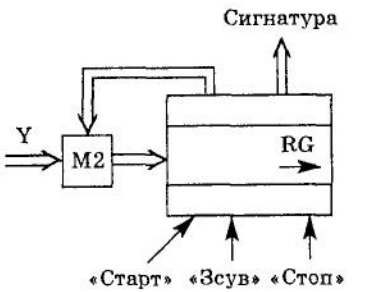


Рис. 2.15. Узагальнена структура сигнатурного аналізатора

чинається зі зсуву на один розряд праворуч під впливом імпульсів синхронізації «Зсув». Після проходу кожного розряду вхідної послідовності Y обчислюється значення:

$$r_1(t) = y(t) \oplus \sum_{i=1}^{t-1} a_j y(t-i),$$

де $y(t)$ — поточний символ послідовності Y, що стискається, $y(t-i)$ — двійкові символи, які відповідають стану регістра в момент часу $(t-1)$, a_j — коефіцієнти полінома, що визначають зворотні зв'язки регістра. Після проходження цієї послідовності Y в регістрі фіксується двійковий код, який і є сигнатурою.

Застосування сигнатурного аналізу складається з таких етапів:

— на входи еталонного пристрою (без несправностей) подають псевдовипадкову послідовність, згенеровану регістром зі зворотними зв'язками;

— у контрольних точках, доступних для спостереження, фіксують і запам'ятовують еталонні сигнатури, які в подальшому використовують для пошуку несправностей;

— перевіряють відповідності реальних сигнатур еталонним (процедура пошуку). Будь-яке відхилення реально отриманої сигнатури від еталону свідчить про несправність відповідного вузла (блока) пристрою.

У цьому полягає основна перевага сигнатурного аналізу. Він максимально спрощує процедури виявлення і локалізації несправностей цифрових пристроїв, не застосовуючи складної стендової апаратури. І, головне, цей метод не потребує висококваліфікованого персоналу для проведення ремонтно-відновлювальних робіт.

Загалом технічний контроль розглядають як одне із завдань діагностування. Поняття діагностування є ширшим, тому що воно має визначити не тільки справність чи несправність об'єкта, але й знайти місце і причину прояву несправностей того чи іншого типу. Тому логічно розглянути методи і засоби діагностування цифрових і мікропроцесорних пристроїв, зокрема персональних комп'ютерів.

Запитання. Завдання

1. Поясніть, що, на вашу думку, є визначальним при виборі методу функційного контролю.
2. Охарактеризуйте метод «зворотної машини». У яких випадках його не можна застосовувати?

3. Яким чином формується множина помилок, які необхідно виявляти при функційному контролі?
4. Обґрунтуйте поняття «коригуюча здатність завадостійкого коду». Які показники її характеризують?
5. Поясніть, що мають спільного і чим відрізняються між собою групові та циклічні коди.
6. Опишіть структуру породжуючої матриці коду.
7. Назвіть і охарактеризуйте способи утворення циклічних кодів.
8. Обґрунтуйте поняття «незводимі поліноми».
9. Розкрийте суть процедури декодування циклічних кодів.
10. У чому полягає основний спосіб утворення кодів БЧХ?
11. Поясніть, чому для цифрових пристроїв не можна застосувати класичну схему корекції помилок на основі завадостійких кодів.
12. Чому кодовий підхід вважають універсальним і узагальнюючим? Обґрунтуйте відповідь.
13. Назвіть апаратні витрати, необхідні для виявлення і корекції помилок на виходах цифрового пристрою. Охарактеризуйте їх.
14. Яким чином забезпечують селективність щодо помилок, які коригуються?
15. У чому полягає основна мета тестового контролю?
16. Яка суттєва відмінність між функційним і тестовим контролем? Обґрунтуйте відповідь.
17. Охарактеризуйте поняття «тест». Яка різниця між перевірковим і діагностичним тестом?
18. Поясніть, як здійснюють активізацію шляху від несправності до виходів пристрою.
19. Назвіть і охарактеризуйте вади, які має алгоритмічний підхід щодо побудови тестів.
20. Зіставте імовірнісний і сигнатурний методи контролю. Що перевіряють при обох підходах?
21. Поясніть, як утворюється сигнатура. Обґрунтуйте, чому вхідну послідовність називають псевдовипадковою.
22. Охарактеризуйте апаратну основу побудови сигнатурних аналізаторів.
23. У чому переваги і вади сигнатурного підходу порівняно з алгоритмічним?

3.

Методи діагностування цифрових, мікропроцесорних пристроїв і ПК

Застосування методів і засобів контролю обчислювальних пристроїв і систем дає змогу визначити, чи придатний об'єкт контролю для подальшого використання за основним призначенням. У разі непридатності використання за основним призначенням його вважають несправним. Для переведення об'єкта з несправного стану у справний необхідно знайти місце і причину прояву несправності, а також усунути її. Вивчення процесів виявлення несправностей, знаходження місця і причини їх прояву належать до галузі технічної діагностики.

3.1. Основні поняття і завдання технічної діагностики обчислювальних пристроїв і систем

Перш ніж перейти до розгляду основних завдань технічної діагностики обчислювальних пристроїв і систем слід розглянути її суть.

Суть технічної діагностики обчислювальних пристроїв і систем

У найширшому розумінні поняття «діагностика» вивчає методи і засоби визначення стану об'єктів будь-якої природи. Так, у медицині таким об'єктом є людина, а в техніці — вироби.

Технічна діагностика *— галузь знань, що досліджує стани ОД і розробляє методи їх розпізнавання, визначає принципи проектування та організації пристроїв і систем технічного діагностування.*

Предметом технічної діагностики є стан будь-яких технічних об'єктів. Їх називають *об'єктами діагностування (ОД)*.

Перевірка працездатності та пошук несправностей у процесі визначення стану ОД є найменш упорядкованими роботами. Якість їх виконання істотно залежить від індивідуальних здібностей людини-оператора і засобів, за допомогою яких вона виконує ці роботи. Процес визначення стану ОД і є технічною діагностикою. Вона розв'язує такі основні завдання:

1) перевірка працездатності ОД. Якщо показники задовільні, переходять до використання пристрою або системи за призначенням, якщо ні — до аналізу їх стану;

2) пошук несправних компонентів і елементів ОД, у результаті якого необхідно їх знайти, а також вказати їхні дефекти (несправності) або причини відмови;

3) прогнозування стану об'єкта на чітко визначений час, коли відомий закон поступової зміни параметрів ОД. Їх зміна призведе до того, що пристрій або система не зможе виконувати своїх функцій у майбутньому.

Ці завдання технічної діагностики пов'язані з розпізнаванням технічного стану пристроїв і систем.

Технічний стан об'єкта *— стан, який характеризують у певний момент часу за певних умов зовнішнього середовища значення параметрів, установлених технічною документацією на об'єкт.*

Обчислювальні пристрої і системи можуть перебувати у справному або несправному стані. Несправних станів може бути безліч, справний — тільки один.

При вмиканні обчислювальної системи перевіряють працездатність одного чи кількох блоків (модулів), що підключаються. Якщо результат позитивний, підключають наступні. Якщо це відбувається на всіх етапах, то пристрій або систему вважають справною і її використову-

ють за прямим призначенням. Коли, хоча б на одному з етапів підключення, одержано негативний результат, то це свідчить, що пристрій або система перебувають в одному з несправних станів. У цій ситуації постають такі завдання:

1) пошук несправних елементів і компонентів. При цьому пошук несправних елементів є нижчим рівнем, ніж пошук несправних компонентів;

2) визначення вихідних параметрів компонентів і елементів. Якщо вони відхилилися від номінальних або не виконують своїх логічних функцій, то їх вважають несправними. Надалі компоненти і елементи підлягають відновленню чи заміні. В такому разі виконують такі операції:

— виділяють компоненти і елементи, параметри яких відхилились від номінальних, але ще не перейшли граничні допуски;

— визначають швидкість виходу параметрів за граничні і на основі цього прогнозують стан об'єкта на майбутнє.

Залежно від типу елементів і компонентів об'єкта застосовують такі підходи: регульовані компоненти і елементи у разі оборотних змін параметрів підстроюють до номіналу, у протилежному випадку — замінюють справними, як і несправні нерегульовані.

Технічне діагностування — процес визначення технічного стану об'єкта з означеною (заданою) точністю.

Під ним розуміють процедуру локалізації несправностей об'єкта, тобто виявлення несправної частини ОД. Технічне діагностування є самостійним процесом при дослідженні об'єкта з невизначеними показниками технічного стану і визначенні типу і місця прояву дефектів. Завдання технічного діагностування об'єкта іноді включають перевірку справності, працездатності, правильності функціювання, пошук місця прояву несправностей. *Об'єктом технічного діагностування* є виріб або його складові частини, технічний стан яких потрібно визначити.

У сучасній обчислювальній техніці об'єктами технічного діагностування є цифрові пристрої і комп'ютери різноманітного призначення, комп'ютерні системи і мережі різної конфігурації та їх складові: конструктивно завершені модулі, блоки, типові елементи заміни, розміщені на окремих друкованих платах, інтегральні схеми різного ступеня інтеграції та інші радіоелектронні елементи, що входять до складу ОД.

Крім того, об'єктами технічного діагностування, особливо на етапах виробництва і експлуатації комп'ютерної техніки, є:

пристрої з компонентами підвищеного ступеня інтеграції (ПКПСІ) — змонтовані на друкованих платах цифрові пристрої, що є функційними вузлами модулів і блоків комп'ютерів та комп'ютерних систем, з наявними в них компонентами підвищеного ступеня інтеграції (КПСІ) — цифровими ВІС та інтегральними схемами надвисокого ступеня інтеграції (НВІС);

програмні засоби — засоби, що складаються з програм і документації, яка стосується їх функціювання;

системи програмування — програмні засоби, які функціюють у програмному середовищі та призначені для розроблення й використання програм.

Діагностування програмного забезпечення (ПЗ) здійснюють за допомогою:

програм контролю — діагностичних програм для перевірки вхідних програм або даних на наявність синтаксичних, семантичних та інших помилок;

програм трасування — діагностичних програм, що можуть простежити виконання кількох або всіх команд програми і записати результати кожного етапу.

Діагностування ПЗ спрямоване на пошук помилок — діяльність, внаслідок якої у програмі виявляють помилки. Її здійснюють в основному на етапі налагоджування — виявлення, локалізації та усунення помилок у програмі.

Коли поведінка програми відповідає специфікації на цю програму — це верифікація програми. (Детальніше діагностування програмного забезпечення буде розглянуто в розділі 5.)

Особливості контролю як елемента діагностування

Діагностування передбачає пошук несправностей об'єкта на нижчому ієрархічному рівні. Контроль є елементом, складовою діагностування. Він дає змогу досягати будь-якого рівня ієрархії об'єкта. Наприклад, коли об'єктом діагностування є пристрій, змонтований на друкованій платі, то складовою діагностування є контроль друкованої плати за принципом «придатний-непридатний», а діагностуванням вважатимуть процедуру пошуку несправних компонентів і елементів, вмонтованих у друковану плату.

Контроль технічного стану об'єкта. Його використовують для вирішення головного завдання технічної діагностики — перевірки працездатності ОД. Суть контролю полягає у перевірці відповідності значень параметрів об'єкта вимогам технічної документації та визначенні на цій основі одного із заданих видів технічного стану (справний, працездатний, несправний, непрацездатний та ін.) у даний момент.

> ***Технічний контроль*** *— процес, що забезпечує виявлення несправностей у роботі цифрових пристроїв, комп'ютерів, комп'ютерних систем і мереж, викликаних відмовою або збоями апаратурних засобів, помилками в програмах, помилкою оператора тощо.*

Об'єктами технічного контролю є будь-яка продукція, процеси її створення, використання, транспортування, зберігання, технічного обслуговування і ремонту, а також відповідна документація.

Контроль ПКПСІ. Процес, який забезпечує встановлення їх відповідності заданим технічним нормам. Результат фіксують у формі «придатний-непридатний».

Будь-який контроль здійснюють у два етапи: 1) отримання первинної інформації про фактичний стан об'єкта контролю (ОК) та про ознаки і показники його властивостей; 2) співставлення первинної інформації із заздалегідь відомими вимогами, нормами, критеріями і отримання вторинної інформації про розбіжність фактичних і необхідних даних або виявлення відповідності (невідповідності) фактичних даних очікуваним.

Залежно від завдань, що вирішують, виокремлюють контроль працездатності, прогнозуючий і діагностичний.

Контроль працездатності. Встановлює відповідність параметрів ОК певним, наперед заданим граничним значенням. При цьому констатують тільки правильність функціювання ОК без зазначення причини відмови (виходу параметрів ОК за допустимі значення).

Прогнозуючий контроль. На основі проведених контрольно-вимірювальних операцій з певною ймовірністю передбачають поведінку ОК протягом певного часу після цієї процедури.

Діагностичний контроль. Визначають не тільки відповідність ОК його технічним характеристикам, а й встановлюють причину відмови. Таке визначення співзвучне технічному діагностуванню, але треба мати на увазі, що при діагностуванні встановлення несправності ОК проводять на нижчому ієрархічному рівні.

Контроль здійснюють за допомогою апаратури або засобів контролю технічного стану, що включають апаратуру і програми. Щодо засобів, які використовуються для реалізації контролю, його поділяють на апаратний, програмний і змішаний.

Апаратний контроль. Здійснюють за допомогою апаратури контролю, його характеризує висока швидкість.

Програмний контроль. Проводять засобами контролю, але при цьому відпрацьовують діагностичні програми. Він пов'язаний з додатковими витратами процесорного часу і необхідністю збільшувати об'єм пам'яті для розміщення програм контролю.

Змішаний контроль. Поєднує апаратний і програмний контролі з метою їх спрощення та підвищення ефективності. Як засвідчує досвід, така практика є найдоцільнішою.

За характером взаємодії засобів контролю з ОК розрізняють активний і пасивний контроль.

Активний контроль. Об'єкт контролю вивчають шляхом подачі тестових впливів на його входи, а також генерації певних режимів з відповідними параметрами;

Пасивний контроль. Стан об'єкта визначають на основі вивчення вихідних параметрів, наявних в ОК за фізичною природою його функціювання (амплітуда, частота та інші параметри вихідних сигналів).

Залежно від часу проведення контроль поділяють на оперативний і тестовий.

Оперативний контроль. Здійснюють у процесі вирішення основних експлуатаційних завдань. Він дає змогу виявляти несправності безпосередньо під час відпрацювання завдань. Такий контроль є неповним, оскільки його виконують на невідповідних для нього завданнях. Прикладом оперативного апаратного контролю є контроль за модулем, який широко застосовують в обчислювальній техніці.

Тестовий контроль. Проводять у спеціально відведені проміжки часу з метою розв'язання тестових завдань. Це забезпечує контроль об'єкта в цілому або його компонентів і елементів, а також контроль виконання команд програми. Чим нижчий ієрархічний рівень тестового контролю, тим більше він наближається до тестового діагностування.

Тести — вхідні впливи, які називають тестовими діями або тестовими наборами.

Після подавання на ОК (ОД) тестових дій він певним чином реагує на них. Як правило, це призводить до зміни

сигналів, які виробляє ОК на вихідних лініях або в контрольованих точках.

Під *відповідною реакцією* (вихідною реакцією) розуміють сукупність логічних контрольованих сигналів на вихідних контактах у заданому такті контролю чи діагностування, які є реакцією об'єкта на подані на його входи тестові впливи.

Еталонні реакції (реакції, що вимагаються) — сукупність логічних контрольованих сигналів на вихідних контактах, які відповідають розрахованим для справного ОК (ОД) відповідним реакціям у заданих тактах контролю (діагностування).

Еталонні реакції можна розраховувати, використовуючи схему електричну принципову, з використанням іншої документації на пристрій або знімати з явно справного об'єкта.

Головна вада тестового контролю полягає в необхідності додаткових витрат часу на його проведення. Однак він дає змогу спростити аналіз результатів та забезпечити необхідну повноту контролю. Оскільки тестовий контроль проводять у спеціально відведений час, це дає змогу багатократно повторювати той чи інший тест. Таким чином, можна накопичувати помилки, пов'язані зі збоями пристроїв або систем, і визначати їх причини та місце виникнення. Для з'ясування оцінюючих характеристик стану ОК використовують допусковий, кількісний та інформаційний контроль.

Допусковий контроль. Встановлює дійсне значення параметра відносно його гранично допустимих значень без вимірювання значення параметра;

Кількісний контроль. Дає кількісну оцінку відхилення параметрів ОК від їх номінальних і граничних значень, а також вимірює і оцінює абсолютні значення параметрів, що контролюються;

Інформаційний контроль (доповнення до кількісного контролю). Забезпечує локалізацію місця виникнення передбачуваної несправності й надає інструкції для її усунення. Співзвучний з поняттям технічного діагностування.

Крім цих видів контролю, в обчислювальній техніці використовують *параметричний контроль* — контроль, що здійснюють шляхом вимірювання електричних параметрів пристроїв і сигналів, які вони обробляють. Його поділяють на контроль статичних параметрів і контроль динамічних параметрів.

Поширеним є *функційний контроль*, що перевіряє здатність ОК правильно виконувати функції, покладені на нього. У процесі цього контролю перевіряють працездатність об'єкта в цілому. У функційному контролі виокремлюють функційно-статистичний і функційно-динамічний.

Функційно-статичний (низькочастотний) контроль. Перевірку правильності відповідних реакцій проводять на тактових частотах, що значно нижчі за мінімально допустимі робочі.

Функційно-динамічний контроль. Перевірку правильності функціювання ОК здійснюють у діапазоні мінімальних і максимальних робочих частот.

За функційного контролю друкованих плат доступ до внутрішніх контрольних точок не обов'язковий. Друковану плату підключають до зовнішніх засобів контрольно-діагностичного обладнання за допомогою розташованих на платах крайових з'єднувачів.

Використовують і *внутрішньосхемний контроль*, за якого тестові впливи подають на внутрішні контрольні точки друкованих плат, а з інших внутрішніх контактів контрольних точок цієї плати знімають відповідні реакції.

Оскільки всі розглянуті види контролю мають певні недоліки і обмеження, на практиці найчастіше використовують їх комбінації.

Особливості і види діагностування

Під діагностуванням у радіоелектроніці та в обчислювальній техніці розуміють процедуру локалізації несправностей ОД. Виокремлюють декілька видів діагностування (контролю).

Робоче технічне діагностування (робоче діагностування). Його суть полягає в подачі на об'єкт робочих впливів.

Тестове технічне діагностування (тестове діагностування). На об'єкт подають тестові впливи.

Експрес-діагностування. Проводять за обмеженої кількості параметрів протягом заздалегідь установленого часу.

Оперативне технічне діагностування (оперативне діагностування). Надходження інформації про технічний стан об'єкта за заздалегідь спланованою стратегією в процесі функціювання об'єкта.

Безперервне технічне діагностування (безперервне діагностування). Надходження інформації про технічний стан об'єкта відбувається безперервно.

Періодичне технічне діагностування (періодичне діагностування). Надходження інформації про технічний стан об'єкта відбувається через встановлені інтервали часу.

Самодіагностування. Здійснюється за допомогою вмонтованих засобів діагностування або спеціальних програм.

Часто використовують також функційне, тестове і змішане діагностування.

1. Функційне діагностування. Вхідними впливами, що надходять на ОД, є робочі впливи, передбачені робочим алгоритмом функціювання об'єкта. Не слід плутати функційне діагностування з діагностуванням об'єкта в процесі його функціювання (це більш загальне поняття). Функційне діагностування, наприклад ТЕЗа чи окремого модуля, може проводитись не під час функціювання системи, складовою частиною якої вони є, а здійснюватися за допомогою зовнішніх засобів шляхом емуляції роботи цієї системи. Це, по суті, робоче діагностування. Функційне діагностування поділяють на апаратне, програмне і змішане.

2. Тестове діагностування. На ОД подають тестові дії, що відрізняються від робочих. Вихідні реакції ОД і вхідні впливи набувають специфічного характеру, невластивого робочим режимам об'єкта. У тестовому діагностуванні виокремлюють: загальне (структурне), покомпонентне і комбіноване.

— Тестове загальне (структурне) діагностування. Його здійснюють у цілому, тобто контролюють цілісність і правильність монтажу, вихідних параметрів ОД та правильність виконання відповідних функцій всією структурою об'єкта. Тестові дії подають на входи ОД через крайові з'єднувачі, а відповідні реакції знімають з вихідної частини з'єднувачів, інколи — із внутрішніх контрольних точок структури об'єкта.

Загальне діагностування поділяють на функційне тестування, параметричний контроль, функційно-параметричний контроль (йому відповідає термін «контроль технічного стану» або визначення продуктивності комп'ютерної системи).

— Тестове покомпонентне діагностування. Це послідовність окремих перевірок кожного компонента структури, за умови, що на нього не впливають зв'язані з ним компоненти. У таких випадках діагностуванням вважають ідентифікацію несправностей (процес виявлення несправності із заданою точністю) елементів контрольованих компонентів. Її поділяють на етапи: констатація факту наявності несправності; встановлення типу і класу несправності; встановлення місця прояву несправності.

У покомпонентному діагностуванні виокремлюють поелементне, пофрагментне і змішане.

• Поелементне діагностування. Передбачає проведення допускового контролю під час оцінювання параметрів кожного електрорадіоелемента пристрою. Тому такий вид діагностування інколи називають внутрішньосхемним параметричним контролем, внутрішньосхемним діагностуванням або внутрішньосхемним контролем.

• Пофрагментне і змішане діагностування. Ці види діагностування характеризують об'єкт, до якого належать.

— Тестове комбіноване діагностування. Проведення певних послідовних структурних і покомпонентних перевірок як об'єкта в цілому, так і його фрагментів, компонентів і елементів спільними апаратурними засобами: Це спрощує процес тестування за рахунок його максимального наближення до робочих режимів функціювання пристрою, який діагностують, пошуку несправностей різних класів, досягнення заданої глибини пошуку дефектів.

3. Змішане діагностування. Охоплює комбінації різноманітних видів діагностування.

Засоби діагностування (контролю) поділяють на такі види:

— автоматичний засіб технічного діагностування. Функціює без участі оператора;

— автоматизований засіб технічного діагностування. Функціює з частковою участю оператора;

— вмонтований засіб технічного діагностування. Є складовою частиною об'єкта;

— зовнішній засіб технічного діагностування. Конструктивно відокремлений від об'єкта;

— бортовий засіб технічного діагностування. Як самостійний виріб входить до складу бортового літального чи іншого рухомого апарата;

— наземний засіб технічного діагностування. Входить до складу наземного устаткування;

— спеціалізований засіб технічного діагностування. Призначений для діагностування одного об'єкта або групи однотипних об'єктів;

— універсальний засіб технічного діагностування. Призначений для діагностування об'єктів різних типів;

— уніфікована апаратура технічного діагностування. Входить до складу спеціалізованих і (або) універсальних засобів технічного діагностування.

Засоби діагностування (контролю) визначають в основному загальний підхід до методу діагностування чи його місцезнаходження. Крім того, їх назва може вказувати на ступінь уніфікації чи універсальності засобу. Ступінь об'єднання засобів діагностування, ОД (ОК) та оператора-виконавця процесу діагностування розкриває поняття «система технічного діагностування».

Система технічного діагностування. Сукупність засобів, об'єктів і виконавців, необхідна для проведення діагностування (контролю) за правилами, встановленими технічною документацією. Як правило, вона охоплює зовнішні засоби технічного діагностування.

Автоматичні системи технічного діагностування. Їх поділяють на два типи, що забезпечують: 1) проведення діагностування без участі оператора; 2) проведення діагностування за допомогою засобів автоматизації та частковою участю оператора.

Діагностування обчислювальних пристроїв і систем передбачає розгляд *показників діагностування* — параметрів ОД, що використовують для визначення його технічного стану. Більшість із них мають подвійну природу і є як технічними, так і діагностичними. Висновок щодо працездатності об'єкта роблять на основі оцінювання сукупності діагностичних показників.

Для порівняння методів і засобів технічного діагностування і визначення їх ефективності використовують показники діагностування (контролю).

Тривалість діагностування. Це інтервал часу, необхідний для проведення діагностування об'єкта.

Повнота діагностування. Характеристика, яка визначає можливість виявлення відмов (несправностей) в об'єкті з використанням обраного методу його діагностування (контролю).

Глибина пошуку місця відмови (несправності). Характеристика, де вказують складову частину об'єкта, з точністю до якої визначають місце відмови.

Достовірність діагностування. Ступінь об'єктивної відповідності результату контролю дійсному технічному стану об'єкта.

Ймовірність невиявленої відмови (несправності) при діагностуванні. Умовна ймовірність того, що непрацездатний об'єкт у результаті діагностування визнають працездатним.

Ймовірність хибної відмови (несправності) під час діагностування. Ймовірність того, що працездатний об'єкт у результаті діагностування виявиться непрацездатним.

Ймовірність невиявленої відмови (несправності) в даному елементі (групі). Ймовірність того, що за наявності відмови в результаті діагностування приймається рішення про її відсутність у даному елементі (групі).

Ймовірність хибної відмови (несправності) в даному елементі (групі). Ймовірність того, що за відсутності відмови в результаті діагностування приймається рішення про її наявність у даному елементі (групі).

Технічне діагностування обчислювальних пристроїв і систем здійснюють як на етапі їх проектування та виробництва, так і під час технічного обслуговування і ремонту. Тому показники якості, що характеризують надійність пристрою або його складових частин, можуть бути одночасно і показниками пристрою як об'єкта діагностування.

Діагностування пристроїв і систем здійснюють за допомогою систем технічного діагностування, що ускладнює відокремлення показників об'єкта від системи діагностування.

Діагностичні показники поділяють на групи, що характеризують: показники об'єкта діагностування; показник діагностованості об'єкта; конструктивну пристосованість об'єкта до діагностування чи контролю.

Показниками об'єкта діагностування є:

T_g — періодичність проведення діагностування або напрацювання пристрою, після якої він потребує діагностування;

τ_g — середній час проведення діагностування як функція напрацювання $\tau_g = f_g(T)$.

Показник діагностованості об'єкта. Це сукупність показників для контролю працездатності. Кількісно його визначають множина показників $U_p = U\{U_1, U_2, \ldots, U_i, \ldots, U_n\}$ і коефіцієнт повноти перевірки працездатності — K_n. $K_n = \lambda_k/\lambda_b$, де λ_k — сумарний показник потоку відмов k-ї складової частин пристрою чи системи; λ_b — сумарний показник потоку відмов усіх складових частин пристрою або системи. Якщо показники потоку відмов пристрою і його складових частин невідомі, то наближено $K_n = n_k/n_b$, де n_k — кількість діагностичних показників; n_b — кількість показників технічного стану, використання яких забезпечує методичну достовірність перевірки.

Існують ще й такі діагностичні показники:

L — довжина тестової послідовності, яку визначають кількістю елементарних тестових дій;

$P_{i,j}$ — ймовірність помилки діагностування виду (i, j) чи ймовірність одночасного настання двох подій: ОД перебуває в технічному стані i, а в результаті діагностування вважають, що він перебуває в стані j;

D — ймовірність правильного діагностування — повна ймовірність того, що система діагностування визначає той технічний стан, у якому справді перебуває ОД.

Конструктивна пристосованість об'єкта до діагностування чи контролю. Її характеризують контролепридатність, а кількісно — показники діагностування і контролепридатності:

$t_{n.g}$ — середня оперативна тривалість діагностування (математичне сподівання оперативної тривалості однократного діагностування);

$t_{m.g}$ — середня оперативна трудомісткість діагностування;

C_g — середня оперативна вартість діагностування;

$K_{y.з}$ — коефіцієнт уніфікації пристроїв з'єднання із засобами діагностування $K_{y.з} = N_y/N_з$, де N_y — кількість уніфікованих пристроїв; $N_з$ — загальне кількість пристроїв з'єднання;

$K_{y.n}$ — коефіцієнт уніфікації показників сигналів пристрою $K_{y.n} = b_y/b_з$, де b_y — кількість уніфікованих діагностичних показників; $b_з$ —загальне кількість показників;

$K_{m.g}$ — коефіцієнт трудомісткості підготовки пристроїв або систем до діагностування $K_{m.g} = (W_g - W_b)/W_b$, де W_g — сумарна трудомісткість підготовки пристрою до діагностування; W_b — середня трудомісткість підготовки пристрою до діагностування; $W_g = W_o + W_b$, а W_o — основна трудомісткість діагностування;

$K_{b.c.}$ — коефіцієнт використання спеціальних засобів діагностування $K_{b.c.} = (G_{c.д.} - G_{c.з.д.})/G_{c.д.}$, де $G_{c.д.}$ — кількість серійних засобів діагностування; $G_{c.з.д.}$ — кількість спеціальних засобів діагностування.

При організації процесу діагностування сукупність діагностичних показників повинна бути мінімальною, але визначати повноту діагностування (контролю), достовірність, можливість пошуку несправностей, прогнозування їх виникнення, чутливість до зміни стану окремих фрагментів і компонентів пристрою, тобто зберігати якість діагностування.

Зростання безвідмовності кожного нового покоління обчислювальних пристроїв, у яких використана новітня елементна база, наводить на роздуми про доцільність процесів контролю і діагностування, особливо на етапі експлуатації пристрою. На етапах проектування і виробництва доцільність їх не викликає сумнівів. У кожному випадку необхідне кількісне оцінювання і врахування багатьох факторів.

Кількісне оцінювання доцільності контролю і діагностування може бути описане *коефіцієнтом доцільності* $K_д = D_1/D_2$, де D_1 — достовірність інформації про технічний стан пристрою чи системи за певного варіанта контролю, D_2 — достовірність інформації про технічний стан цього об'єкта за іншого варіанта контролю або діагностування. Одержані результати дають змогу порівняти варіанти діагностування (контролю) і обрати найдоцільніший.

> *Достовірність роботи обчислювальних пристроїв — властивість, що характеризує істинність вихідного результату роботи пристрою, яка визначається здатністю засобів контролю фіксувати правильність чи помилковість його роботи.*

Залежить вона тільки від помилок, викликаних відмовами і збоями ОД, засобів контролю або діагностування, характеристик методу контролю. Помилки оператора або розробника тестів не беруть до уваги, тому що на входи ОД подають безпомилкові тести.

Достовірність роботи обчислювального пристрою розглядають з точки зору таких основних понять: достовірності функціювання, достовірності правильного функціювання і достовірності помилкового функціювання.

Достовірність функціювання. Властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати результат роботи пристрою правильним або помилковим за наявності пропуску помилок чи видачі хибних сигналів помилок засобами контролю.

Достовірність правильного функціювання. Властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати правильним результат роботи пристрою за наявності пропуску помилок засобами контролю.

Достовірність помилкового функціювання. Властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати помилковим результат роботи пристрою за наявності хибних сигналів помилок, що видаються засобами контролю.

Правильним результатом роботи обчислювального пристрою є такий результат, коли пристрій справді працює правильно, відсутні сигнали помилок, або результат, коли пристрій справді працює неправильно, про що свідчить сигнал помилки.

Пропуск помилки засобами контролю — результат неправильної роботи пристрою і відсутності сигналу помилки. Якщо пристрій працює правильно, а засоби контролю сигналізують про помилку, то це свідчить, що вони контролюють ОК не повністю або зовсім не контролюють.

Наведені визначення створюють сукупність подій, яку описують таким виразом:

$$P_{пр.п}(t) + P_{н.п}(t) + P_{н.в}(t) + P_{п.п}(t) = 1, \qquad (3.1)$$

де $P_{пр.п}(t)$ — ймовірність правильної (безпомилкової) роботи пристрою; $P_{н.п}(t)$ — ймовірність неправильної роботи пристрою, про що свідчить сигнал помилки; $P_{н.в}(t)$ — ймовірність того, що пристрій працює неправильно, але сигналу помилки немає (ймовірність пропуску помилки засобами контролю); $P_{п.п}(t)$ — ймовірність того, що пристрій працює правильно, але є сигнал помилки (ймовірність появи хибного сигналу помилки); t — період роботи пристрою, за який оцінюють достовірність роботи.

Для цифрових і мікропроцесорних пристроїв без врахування відновлення використовують такі показники: $D_ф(t)$ — достовірність функціювання; $D_{п.ф}(t)$ — достовірність правильного функціювання і $D_{н.ф}(t)$ — достовірність помилкового функціювання.

Достовірність функціювання $D_ф(t)$. Умовна ймовірність того, що засоби контролю відображають правильний результат роботи пристрою, коли є пропуск помилки засобами контролю і хибний сигнал помилки на виході засобів контролю:

$$D_ф(t) = \frac{P_{пр.п}(t) + P_{н.п}(t)}{P_{пр.п}(t) + P_{н.п}(t) + P_{н.в}(t) + P_{п.п}(t)}, \qquad (3.2)$$

якщо врахувати (3.1), то одержують:

$$D_ф(t) = P_{пр.п}(t) + P_{н.п}(t), \qquad (3.3)$$

або:

$$D_ф(t) = 1 - P_{н.в}(t) - P_{п.п}(t). \qquad (3.4)$$

Із формули (3.4) випливає, що достовірність функціювання визначають достовірністю невидачі цифровим чи мікропроцесорним пристроєм неправильних результатів.

Достовірність правильного функціювання $D_{п.ф}(t)$. Умовна ймовірність того, що пристрій працює правильно, коли є пропуск помилки засобами контролю:

$$D_{п.ф}(t) = \frac{P_{пр.п}(t)}{P_{пр.п}(t) + P_{н.в}(t)}. \qquad (3.5)$$

Достовірність помилкового функціювання $D_{н.ф}(t)$. Умовна ймовірність того, що пристрій працює неправильно, коли видається хибний сигнал помилки засобами контролю:

$$D_{н.ф}(t) = \frac{P_{н.п}(t)}{P_{н.п}(t) + P_{п.п}(t)}. \qquad (3.6)$$

Це свідчить, що показники достовірності визначають ступінь довіри до результатів роботи пристрою. Вони оцінюють працездатність як об'єкта контролю, так і засобів контролю і тому суттєво залежать від показників засобів контролю. Завдяки цьому їх використовують як критерії, за якими обирають засоби контролю на етапі проектування апаратури і оцінювання її ефективності.

Оцінювання ефективності діагностування обчислювальних пристроїв

Оцінювання ефективності дає змогу зробити висновок, наскільки вдалим виявилось використання методів і засобів діагностування.

Ефективність *— комплексна властивість процесу використання даної системи за призначенням на певний момент часу.*

У результаті роботи системи одержують позитивний або негативний ефект. Системи діагностування можуть бути реалізовані на досить високому рівні і значно поліпшувати результати використання попередніх засобів-аналогів. Проте, з огляду на певні об'єктивні причини, наприклад високу вартість апаратури нових засобів, ефективність використання системи діагностування може виявитись низькою або навіть недоцільною.

Ефективність діагностування обчислювальних пристроїв є комплексним поняттям, яке охоплює:

— якість системи (сукупність властивостей системи, що дає змогу задовольняти певні потреби відповідно до її призначення);

— якість експлуатації (сукупність властивостей процесу експлуатації системи, від яких залежить відповідність цього процесу і його результатів встановленим вимогам);

— експлуатаційну ситуацію (обставини, що зумовлюють вплив зовнішнього середовища, мету і режими функційного використання системи, запит на систему і результати її функціювання).

Ефективність і якість системи оцінюють за сукупністю відповідних показників:

— показник ефективності використання діагностичних засобів (кількісна характеристика ступеня досягнення корисних результатів при використанні системи в конкретній експлуатаційній ситуації з урахуванням експлуатаційних витрат);

— показник якості (кількісна характеристика однієї або кількох властивостей системи, які складають її якість, що розглядається стосовно певних умов створення і споживання).

Показники якості поділяють на інтегральні, одиничні та комплексні.

Інтегральний показник якості. Є близьким за змістом до показника ефективності використання діагностичних засобів. Його визначають як відношення сумарного корисного ефекту від експлуатації діагностичних засобів до сумарних витрат на їх створення і експлуатацію.

Одиничний показник якості. Це показники функційного використання, технічні і експлуатаційні показники діагностичних засобів, до яких належать достовірність інформації, ймовірність помилок діагностування технічного стану, безвідмовність, довговічність та ін.

Комплексний показник якості. Він характеризує кілька простих властивостей чи одну складну властивість діагностичних засобів. Ним є коефіцієнт $K_{т.в}$ — технічного використання. Йдеться про діагностичні засоби у зв'язку з можливостями відмов вузлів і блоків, що входять до їх складу, і необхідність певних робіт для підтримання якості.

Комплексні показники ефективності можуть бути представлені як добуток показників якості K_i, тобто:

$$K = K_1, K_2, \ldots, K_i, \ldots, K_m \qquad (3.7)$$

або частка, одержана в результаті ділення одних показників на інші:

$$K = \frac{K_1 K_2, \ldots, K_m}{K_{m+1}, K_{m+2}, \ldots, K_n}. \qquad (3.8)$$

Найзагальніші вирази для показників ефективності повинні, крім технічних, включати й економічні показники систем і процесів.

Вираз для інтегрального показника якості може бути заданий формулою:

$$I = \frac{E}{B_c + B_e}, \qquad (3.9)$$

де E — сумарний корисний ефект від функційного використання діагностичних засобів; B_c — сумарні витрати на їх створення; B_e — сумарні витрати на їх експлуатацію, в тому числі технічне обслуговування, ремонт та інші складові. Аналогічно може бути визначений і показник ефективності використання. Часто для характеристики показника якості і показника ефективності використовують один термін *показник якості і ефективності* K_e. Основним є визначення ефективності використання діагностичних засобів, тому для представлення K_e до нього повинні входити елементи корисного ефекту діагностичних засобів. Це можуть бути: підвищення безвідмовності обчислювальних пристроїв (зменшення потоку відмов $\lambda(t)$), скорочення часу відновлення пристроїв, збільшення коефіцієнта технічного використання, підвищення надійності пристроїв загалом, підвищення обсягу інформації в системі інформаційного забезпечення та сукупність цих елементів.

Натепер немає чітких і однозначних рекомендацій щодо складових ефективності для визначення K_e. Проблема особливо важлива при розробленні варіантів систем діагностування і їх порівнянні.

Ефективність операцій контролю і діагностування можна представити як різницю:

$$K_e(t) = \Delta E = E(t/t_g) - E(t);\ t > t_g, \qquad (3.10)$$

де $E(t/t_g)$ — ефективність діагностування за умови, що в момент t_g проведено діагностування і технічне обслуговування об'єкта; $E(t)$ — ефективність діагностування за умови, що технічне обслуговування не проводилось. Нормований показник ефективності використання діагностичних засобів:

$$K_e = [E(t/t_g) - E(t)]/E(t);\ 0 < K_e < 1. \qquad (3.11)$$

Визначення ефективності діагностування обчислювальних пристроїв має дуже важливе практичне значення, оскільки дає відповідь на питання, чи є сенс взагалі його реалізовувати.

3.2. Цифрові і мікропроцесорні пристрої як об'єкти діагностування

Невиконання функцій, для яких призначені цифрові і мікропроцесорні пристрої, як правило, зумовлено наявністю дефектів, які є фізичним явищем (обрив провідника, псування кристала та ін.). Дефекти, у свою чергу, формалізовано описують несправностями. Для успішної ідентифікації несправностей (визначення типу та знаходження місця прояву) у цифрових і мікропроцесорних пристроях необхідно спочатку розглянути і визначити їх властивості як об'єктів діагностування. Насамперед це стосується математичних моделей і типів та моделей несправностей, що виникають у них.

Дефекти і несправності цифрових і мікропроцесорних пристроїв

Основними компонентами сучасних цифрових і мікропроцесорних пристроїв (Ц і МПП) є компоненти підвищеного ступеня інтеграції — ВІС і НВІС.

Несправність ВІС або НВІС. Це стан, зумовлений дефектами одного чи кількох елементів внутрішньої структури кристала або провідників, що з'єднують його з виводами цієї ВІС (НВІС). Враховуючи це, несправністю цифрового чи мікропроцесорного пристрою вважають стан, зумовлений дефектами одного чи кількох компонентів (елементів) їх структури або провідників, що з'єднують компоненти в цю структуру.

Несправність Ц або МПП. Це формалізоване представлення факту прояву дефекту цих пристроїв у вигляді неправильних значень сигналів на входах і виходах.

Дефекти і несправності можуть бути сталими (постійними) чи несталими (короткочасними, що перемежовуються).

У життєвому циклі обчислювальних пристроїв виокремлюють три основних стадії: проектування, виробництво та експлуатацію. На кожній з них існує ймовірність виникнення конструктивних і фізичних дефектів. Ці процеси залежать від різних факторів, насамперед від технології проектування і виготовлення, а також від елементної бази. Дефекти одного чи кількох компонентів пристрою, порушення друкованого монтажу, призводять до збоїв і відмов.

Збій — короткочасне порушення правильної роботи цифрового чи мікропроцесорного пристрою, після якого його працездатність відновлюється або її відновлює оператор без проведення ремонту.

Це порушення зумовлено дефектом, суть якого в тому, що в результаті тимчасової зміни показників окремих елементів пристрою чи схеми його з'єднань протягом певного часу він буде функціювати неправильно.

У такому разі під відмовою будемо розуміти подію, яка полягає в порушенні працездатності об'єкта, що не самовідновлюється. Це безповоротне порушення характеристик окремих елементів обчислювального пристрою або схеми з'єднань.

Залежно від місця виникнення в інтегральній схемі визначають такі типи дефектів:

— пов'язані з фізичними явищами у внутрішній структурі кристала;

— зумовлені явищами на поверхні кристала.

За причинами виникнення виокремлюють групи дефектів:

— пов'язані з недосконалістю конструкції і технології, що з розробленням і впровадженням досконалішої технології та процесів виробництва зменшуються;

— спричинені короткочасними перевантаженнями напруги або потужності;

— пов'язані з вадами проектування і конструювання.

Найпоширеніші дефекти ІС наведені в табл. 3.1.

Надзвичайно важливим є пошук дефектів компонентів, що проектуються, зокрема ВІС та НВІС. На *стадії проектування* особливу увагу звертають на методи контролепридатного проектування, тестопридатність і верифікацію. Новий клас систем тестування — верифікатори (тестери) дослідних зразків — орієнтований на інтегральні схеми з коротким циклом проектування. Верифікатори належать до засобів імітаційного моделювання. Під час проектування їх застосовують ширше, ніж методи і засоби безпосереднього контролю, що використовують у лабораторних умовах (візуалізація дефектів, мікрозондування, засоби модифікації та ремонту кристалів).

Важливу роль у життєвому циклі Ц і МПП відіграє *стадія виробництва*, що охоплює вхідний контроль компонентів, відбір комплектуючих виробів, монтаж і налаго-

Таблиця 3.1

Дефекти компонентів інтегральних схем Ц і МПП

Дефекти маркування	Дефекти виробничі	Дефекти механічні	Дефекти, пов'язані з відхиленням електричних параметрів
— компоненти з помилковим маркуванням; — неправильне позначення компонентів	— наявність домішок у кристалі; — забруднення поверхонь кристала; — дефекти кристала як напівпровідника; — дефекти покриття; — дефекти металізації; — неякісне під'єднання виводів; — корозія	— тріщини в кристалах; — зігнуті або зламані виводи; — неправильне формування виводів; — порушення герметичності корпусу	— дрейф; — відхилення параметрів ІС; — зміна ємностей

дження пристрою. Виробництво пристроїв і систем, побудованих на їх базі, пов'язане з можливістю внесення дефектів на кожній стадії технологічного процесу складання.

Найважливішими причинами виникнення дефектів на етапі виробництва пристроїв є:

— дефекти компонентів;

— відхилення технологічного процесу складання від встановлених норм;

— суб'єктивні фактори (психофізичні особливості людини-оператора).

До найтиповіших дефектів виробництва належать:

— короткі замикання між друкованими провідниками на платі;

— обрив провідників на друкованій платі;

— неправильна орієнтація мікросхем щодо шин живлення;

— монтаж елемента з іншими реалізуючими функціями.

Кількість і типи дефектів залежать насамперед від рівня технології виготовлення пристроїв і на різних підприємствах і фірмах можуть значно відрізнятись. Виробнича статистика фізичних дефектів обчислювальних пристроїв відображена в табл. 3.2.

З розглянутого матеріалу видно, що значна частина несправностей зумовлена дефектами комплектуючих ви-

Таблиця 3.2

Статистичні дані про дефекти обчислювальних пристроїв[1]

№ п/п	Тип дефектів	Відсотковий вміст, %
1	Коротке замикання друкованих провідників	35—70
2	Обрив друкованих провідників	0—3
3	Монтаж компонентів з іншими реалізуючими функціями	3—15
4	Невмонтовані компоненти	2—5
5	Неправильна орієнтація ІС щодо шин живлення	2—10
6	Незадовільні робочі характеристики компонентів	0—22
7	Дефекти елементів компонентів	2—40
8	Інші дефекти	0—2

[1] Статистичні дані пораховані так, що в конкретному пристрої може бути та чи інша несправність. Тому їх кількість перевищує 100 %.

робів, короткими замиканнями, обривами провідників та ін. Тому на етапі виробництва особливу увагу слід звертати на організацію вхідного контролю компонентів обчислювальних пристроїв, а також на технологію і якість їх монтажу.

Важливо мати на увазі, що багато несправностей Ц і МПП та їх компонентів проявляються в динамічних режимах, а ігнорування ними призводить до значних економічних втрат. Несправностей Ц і МПП є чимало. У табл. 3.3 наведено основні з них, крім того, способи їх виявлення і усунення.

Таблиця 3.3

Основні несправності обчислювальних пристроїв, способи їх виявлення і усунення на етапі виробництва[1]

№ п/п	Етапи життєвого циклу і несправності, що виникають	Способи	
		виявлення	усунення
1	2	3	4
I	Відбір і контроль комплектуючих виробів		
1	Несправності компонентів	вхідний контроль (статичний)	заміна компонентів

Закінчення таблиці 3.3

1	2	3	4
2	Незадовільні робочі характеристики	вхідний контроль (динамічний)	заміна компонентів
II	**Монтаж, складання, налагодження**		
1	Коротке замикання провідників	діагностування (статичне)	усунення хибних перемичок
2	Обрив провідників	діагностування (статичне)	ремонт провідників
3	Неустановлені компоненти	контроль (візуальний)	установка компонентів
4	Неправильна орієнтація компонентів	контроль (візуальний статичний)	правильна установка компонентів
5	Встановлено компоненти з іншою функцією	контроль (статичний)	установка компонентів з необхідною функцією
6	Несправності динамічного типу	діагностування (динамічне)	заміна несправних компонентів
7	Збої	діагностування (динамічне)	заміна несправних компонентів

[1] На етапі експлуатації виникають несправності II (1; 2; 6; 7). Способи їх виявлення і усунення такі самі.

На *стадії експлуатації* дефекти і відповідні їм несправності, які з'явилися під час виготовлення пристрою та в період його експлуатації, призводять до збоїв і відмов. Вони пов'язані з фізичним руйнуванням компонентів пристрою чи поступовим погіршенням його характеристик. Дефекти, несправності, збої та відмови взаємно залежать один від одного. Цю особливість показано на рис. 3.1.

Рис. 3.1. Взаємна залежність дефектів, несправностей, збоїв і відмов

Багато дефектів ІС, зокрема ВІС і НВІС, не завжди проявляються як константні несправності (наприклад, інверсія (лат. inversio — перестановка) біт в елементах компонентів на МОН-структурах та ін.).

Крім того, ВІС і НВІС до постачання споживачеві піддають переважно обмеженому контролю. Тому відмови можливі на етапі експлуатації, коли подають заборонені комбінації сигналів, які використовують у процесі вихідного контролю або коли змінюють певні параметри компонентів, що викликають непередбачувану поведінку пристрою.

Інтенсивність збоїв ІС та Ц і МПП на один-два порядки вища, ніж інтенсивність відмов. Збої можуть проявлятися при зміні: напруги живлення; допустимих завад; статичних і динамічних показників компонентів; граничних внутрішніх співвідношень вхідних сигналів і сигналів стробів; внутрішніх імпульсних завад у колах живлення і сигнальних провідниках; внутрішніх завад і значень сигналів на фронтах.

Певними особливостями характеризують відмови НВІС запам'ятовуючих пристроїв (ЗП) бортових цифрових обчислювальних комплексів. Найбільше впливають на їх роботу збої, що виникають у ЗП і спотворюють не тільки дані, але й програми роботи. Головна причина появи таких збоїв — наявність у матеріалі корпусу НВІС молекул урану і торію, при ядерному розпаді яких з'являються альфа-частинки. При гальмуванні вони створюють заряди, які призводять до утворення хибного сигналу на виході ЗП. Дія альфа-частинок на запам'ятовуючі елементи динамічних ЗП викликає перехід зі стану логічного «0» і «1», що, в свою чергу, призводить до несправностей на момент передавання даних. Статичні ЗП також схильні до впливу альфа-частинок, але значно менше.

Другим найбільш помітним джерелом збоїв є космічні промені. Накопичення енергії під їх дією в чутливому об'ємі НВІС призводить до її хибного функціювання.

Статичні і динамічні несправності. Дефекти (несправності) за характером прояву поділяють на детерміновані (дефекти, що проявляються постійно) і випадкові.

У детермінованих несправностях виокремлюють статичні і динамічні.

До ***несправностей статичного типу*** належать несправності Ц і МПП, що проявляються на частотах значно нижчих від робочих. Вони є декількох типів.

Логічні несправності — несправності логічних елементів і компонентів, побудованих на основі Ц і МПП, обриви зв'язків, які призводять до зміни логічних функцій, що реалізують ці елементи і компоненти. Питома вага логічних несправностей може становити до 90 % і більше від кількості всіх несправностей, що ідентифікують.

Логічні несправності описують математичні моделі константних несправностей — «константний нуль» і «константна одиниця». Їх подають як модель «константного нуля» («≡ 0»): $y_i \equiv 0$ при $\{x_i\}$, $(i = \overline{0,1})$, $x_i = \{0,1\}$; модель «константної одиниці» («≡ 1»): $y_i \equiv 1$ при $\{x_i\}$, $(i = \overline{0,1})$, $x_i = \{0,1\}$; де $x_1, x_2, \ldots, x_n$ — вхідні змінні; y_i — стан i-го виходу структури цифрового пристрою.

Суть моделі полягає в тому, що, подаючи на входи цифрової стуктури будь-які дозволені набори змінних двійкової алгебри Буля, на одному або кількох функційних виходах структури встановлюється «константний нуль» або «константна одиниця». Таку модель часто називають класичною і використовують для опису інших типів несправностей.

Раніше при розробленні методів ідентифікації несправностей за допомогою систем тестового діагностування цифрових пристроїв, що були побудовані на основі ІС малого та середнього ступенів інтеграції, найчастіше використовувалась саме така модель константних несправностей.

Зміна логічної функції елемента — замість однієї функції виконується інша.

Коротке замикання — виникає в результаті короткого замикання сигнальних ліній входів, виходів, сигнальних ліній і шин електроживлення, короткого замикання типу зворотного зв'язку.

Інверсні несправності — поява фіктивного інвертора (інверторів) на вході (входах) чи виході (виходах) структури або її елементів.

Переплутування зв'язків — полягає у переплутуванні зв'язків цифрової структури.

Контактні несправності — результат обриву провідників і відсутності контакту в крайових з'єднувачах пристроїв.

Несправності статичного типу цифрових пристроїв, побудованих на ІС малого і середнього ступенів інтеграції, аналогічні несправностям статичного типу пристроїв, побудованих на ВІС і НВІС.

До ***несправностей динамічного типу*** належать несправності Ц і МПП пристроїв, що проявляються в динамічних режимах роботи, тобто в діапазоні від мінімальних до максимальних робочих частот.

Питома вага несправностей динамічного типу в обчислювальних пристроях, побудованих на ІС малого і середнього ступенів інтеграції, незначна. Проте матеріальні витрати, яких можуть зазнати виробники, нехтуючи ними, будуть досить відчутні. Несправності динамічного типу поділяються на логічні і нелогічні.

Логічна динамічна несправність — це стан, який призводить до спотворення логічних функцій, що реалізують елементи структури в динамічних режимах роботи.

Серед логічних динамічних несправностей виокремлюють:

— несправності, що призводять до зміни логічних функцій, які реалізуються цифровою структурою пристрою;

— несправності, що пов'язані з видом вхідних тестових послідовностей Ц або МПП і призводять до спотворення тест-векторів контрольованих сигналів відповідних реакцій порівняно з еталонними.

Нелогічна динамічна несправність — це стан, який призводить до появи неправильних значень сигналів на виходах структури в динамічних режимах роботи і не пов'язаний зі спотворенням елементами структури логічних функцій, що ними реалізуються.

Нелогічні динамічні несправності поділяють на:

— несправності, виявлені в результаті дії завад. Виникають у лініях при перемиканні груп елементів структури пристрою. Причина багатьох несправностей цього підкласу — некоректне проектування (конструювання). Для вивчення іншої частини несправностей підкласу на ОД на максимальних робочих частотах подають набори стимульованих сигналів тестових впливів, які перевіряють усі можливі варіанти перемикання груп шин;

— несправності, причини яких «змагання» сигналів. Одна з основних причин таких несправностей — помилки розробників, а також різна інерційність компонентів пристрою, збільшення паразитних ємностей в міру старіння компонентів, зміна умов експлуатації та ін. Тому лише коректне проектування не знімає актуальності виявлення таких несправностей;

— несправності, викликані спотворенням фронтів імпульсних сигналів. Вони можуть бути зумовлені збільшенням тривалості фронтів імпульсних сигналів. Причинами таких змін є фізичне старіння компонентів структури, їх роз-

герметизація, дії, пов'язані зі зміною параметрів навколишнього середовища, та ін. Отже, треба якнайдовше затримувати поширення імпульсних сигналів у структурі пристрою;

— несправності, зумовлені шинною структурою МПП. Одна з їх головних причин — зменшення навантажувальної здатності буферних елементів структури в динамічних режимах.

Взаємозв'язок розглянутих класів і підкласів несправностей Ц і МПП показано на рис. 3.2.

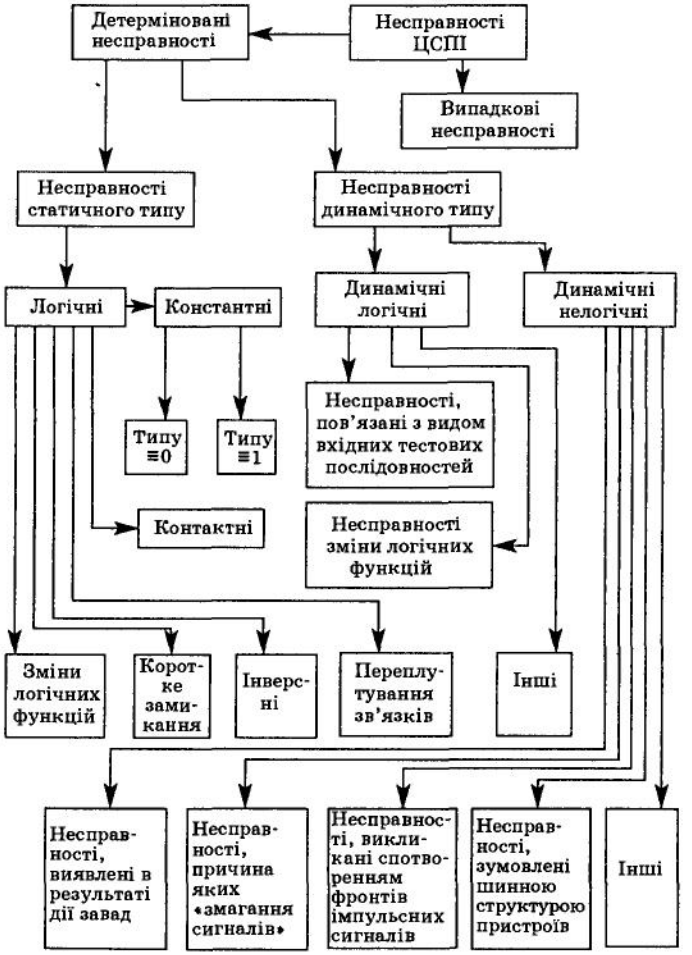


Рис. 3.2. Класи і підкласи несправностей обчислювальних пристроїв

Наведена класифікація дає змогу складати алгоритми ідентифікації несправностей Ц і МПП, враховуючи, що в структурі пристрою одночасно можуть бути несправності різних класів і підкласів і їх ліквідацію необхідно проводити в певній послідовності.

Ідентифікація несправностей передбачає розроблення і аналіз моделей цифрових пристроїв як об'єктів діагностування.

Моделі цифрових пристроїв як об'єктів діагностування

У технічній діагностиці етапи моделювання і діагностування природно пов'язані між собою, тому що моделювання Ц і МПП як об'єктів діагностування само по собі не має сенсу без подальшого його використання в процесі діагностування, а процес діагностування неможливий без розробки моделей Ц і МПП як об'єктів діагностування. Однак щодо методів діагностування, ці питання розглядають окремо: на моделях ОД ґрунтується сам метод, тобто підхід до діагностування, а моделі процесу діагностування, як правило, покладені в основу методик діагностування.

З огляду на те, що Ц і МПП — складні системи, складовими яких є елементи, компоненти і фрагменти МПП, змонтовані на друкованих платах (картриджах) в основному з ІС підвищеного ступеня інтеграції, загальну модель представляють як багаторівневу (стратову), де кожному компоненту, фрагменту чи пристрою відповідає певний рівень (страт) деталізації. Узагальнена модель Ц і МПП чи системи (С) представлена множиною моделей $M = \{m_i\}$, $(i = 1,n)$, де модель m_i відображає поведінку МПП чи системи на i-му рівні деталізації. Така схема МПП і С зображена на рис. 3.3.

Якщо прийняти за k-й рівень модель, що описує МПП, яка відповідає платі об'єднувальній, то цей рівень можна представити алгоритмічною моделлю:

$$m_k = \langle X, Y, Q, F_\Delta, F_\lambda, P, S_A \rangle,\ m_k \in m_{k+1},$$

де X, Y, Q — множини станів входів, виходів і внутрішніх станів вузлів плати об'єднувальної ; F_Δ, F_λ — множини функцій переходів і виходів вузлів об'єднувальної плати; P — множина предикатів, за допомогою яких в алгоритмі реалізуються умовні переходи; S_A — граф-схема алгоритму, що має два типи вершин: операторні (обчислюють значення функцій на основі F_Δ і F_λ) і предикатні (перевіряють

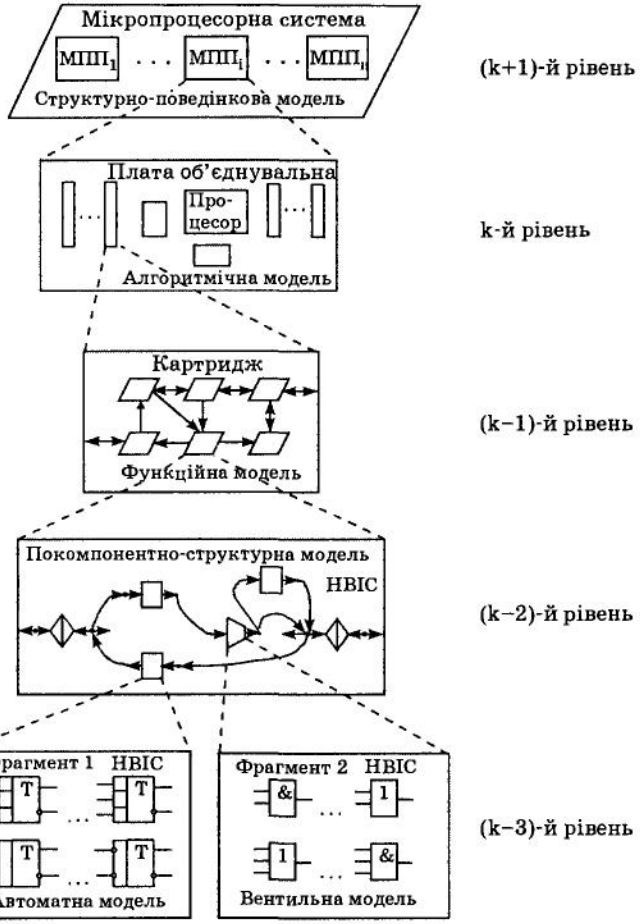


Рис. 3.3. Структурна схема рівнів узагальненої моделі МПП і С

правильність заданого предиката і за його результатом визначають наступну ділянку розрахункового шляху в алгоритмі).

При описі МПП алгоритмічною моделлю домінуючу роль відіграють програмні засоби діагностування. Таким чином, процес тестування спрощують, але за допомогою алгоритмічної моделі не можна визначити місце несправності з точністю до компонента. Це можливо при переході до функційної моделі на (k-1)-му рівні узагальненої моделі.

(k-1)-рівень представляють функційною моделлю:

$$m_{k-1} = \langle X, Y, Q, F_\Delta, F_\lambda, S_\Phi \rangle,\ m_{k-1} \in m_k,$$

де X, Y, Q — множини вхідних, вихідних і внутрішніх станів функційних вузлів, які відповідають компонентам (ІС різного ступеня інтеграції) картриджа у даному прикладі; F_Δ, F_λ — множини функцій переходів і виходів функційних вузлів картриджа; S_Φ — розмічена функційна схема, відображена графом, з вершинами якого зіставлені функційні вузли, реалізовані як окремі компоненти, а з ребрами — зв'язки між ними.

Моделювання на (k-1)-му рівні дає змогу визначити місце прояву несправності з глибиною до компонента інтегральної схеми. Якщо необхідно збільшити глибину пошуку несправності до структурного вузла НВІС, то переходять до моделювання на (k-2)-му рівні.

Модель на (k-2)-му рівні подано у вигляді:

$$m_{k-2} = \langle X, Y, Q, F_\Delta, F_\lambda, S_C \rangle,\ m_{k-2} \in m_{k-1},$$

де X, Y, Q — множини вхідних, вихідних і внутрішніх станів структурних вузлів; F_Δ, F_λ — множини функцій переходів і виходів структурних вузлів НВІС; S_C — структурна схема, відображена графом, з вершинами якого зіставлені структурні вузли НВІС, а з ребрами — зв'язки між ними.

(k-2)-й рівень моделювання представлений покомпонентно-структурною моделлю абстрактної НВІС (див. рис. 3.3):

$$m_{k-1} = \langle V, I, S_{V,I} \rangle,\ m_{k-2} \in m_{k-1},$$

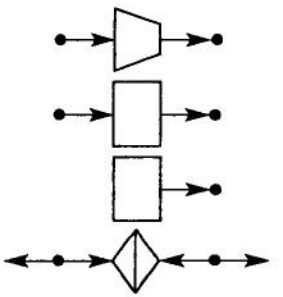
де V — множина вершин $S_{V,I}$-графа; I — множина ребер $S_{V,I}$-графа; $S_{V,I}$ — граф, вершини якого — сукупність операційних вузлів, регістрів, буферів, елементів керування, а ребра — процедури (акти передачі певної кількості тест-векторів заданої розмірності) обміну інформацією між ними. Вузли абстрактної НВІС мають такі позначення:

1) операційний вузол (ОВ) —

2) операційний регістр (ОР) —

3) регістр стану (РС) —

4) буфер двонапрямлений (БД) —



За необхідності збільшити глибину пошуку несправностей переходять до (k-3)-го рівня моделювання, на якому фрагменти НВІС або цифрового пристрою можуть бути представлені вентильною чи автоматною моделлю. Вентильну модель подають у вигляді:

$$m_B = \langle X, Y, F, S_B \rangle,\ m_B \in m_{k-3},\ m_{k-3} \in m_{k-2},$$

де X, Y — множини входів і виходів вентилів фрагмента НВІС; F — множина описів функцій, що реалізується логічними елементами (вентилями) фрагмента НВІС; S_B — принципова схема, що відображена графом, вершини якого зіставлені з логічними елементами фрагмента НВІС, а ребра — зі зв'язками між ними. Така модель дає змогу виявляти несправності вентилів НВІС.

Якщо розглядати фрагмент НВІС на рівні цифрових автоматів, то (k-3)-му рівню моделювання відповідатимуть автоматні моделі, які можна представити так:

$$m_{aj} = \langle X, Y, Q, F_\Delta, F_\lambda, S_a \rangle,\ m_{aj} \in m_{k-3},\ m_{k-3} \in m_{k-2},$$

де X, Y, Q — множини вхідних, вихідних і внутрішніх станів j-го автомата; F_Δ, F_λ — множини однокрокових функцій переходів і виходів j-го автомата; S_a — граф, вершини якого зіставлені з автоматами фрагмента НВІС. Знання внутрішньої структури фрагмента НВІС на рівні вентилів не обов'язкове.

Узагальнена модель відкрита щодо рівнів моделювання, тобто її можуть доповнювати вищі, загальніші рівні моделювання (k+1, k+2 і т. д.) і нижчі, детальніші (k–4, k–5 і т. д.). У ній (див. рис. 3.3) вищим рівнем моделювання є (k+1)-й рівень, представлений структурно-поведінковою моделлю мікропроцесорної системи, а нижчим (k–4)-й рівень моделювання, якому відповідають цифрові автомати або вентилі фрагмента НВІС, представлені транзисторними схемами.

Рівні моделювання обирають залежно від того, на якому етапі життєвого циклу Ц чи МПП відбувається процес діагностування. На етапі проектування використовують вказані вище рівні моделювання МПП як об'єктів діагностування. На етапі виготовлення МПП задають рівні з k–2 по k+1, а на етапі експлуатації тільки рівні з k–1 по k+1, тому що в цьому випадку достатньо глибини пошуку несправності до компонента. У багаторівневій моделі можна описувати одні й ті самі фрагменти і компоненти МПП моделями різних типів. Наприклад, НВІС як об'єкт діагностування можна описувати покомпонентно-структурною і

функційною, а МПП — алгоритмічною і структурно-поведінковою моделями.

При такому описі в узагальнену модель можна не включати рівень моделювання, що не впливає на значення вихідних реакцій ОД, і не деталізувати кожний елемент певного рівня моделі.

Засоби діагностування Ц і МПП. Вбудовані засоби і методи контролю та діагностування мають багато переваг: здійснення контролю в процесі використання пристрою за призначенням (без витрат додаткового часу), простота спрягання вмонтованих засобів діагностування з основною апаратурою та ін. Проте їм властиві й недоліки, що знижують достовірність і ефективність процесу діагностування. Зокрема, надлишковість компонентів і елементів у пристроях і відповідно додаткові витрати замовників, особливо тоді, коли йдеться про виробництво порівняно простої обчислювальної апаратури. Крім того, не можна здійснити такі операції:

— у певних випадках виділити частину апаратурних ресурсів на вбудовані засоби діагностування;

— змінити діагностичні програми у вбудованих засобах;

— досягнути необхідної глибини пошуку несправностей;

— провести діагностування, коли вийшло з ладу діагностичне ядро.

Тому для контролю і діагностування обчислювальних пристроїв і систем, особливо на етапах проектування та виробництва, широко використовують зовнішні засоби і відповідні їм методи. Узагальнена класифікація методів діагностування цифрових пристроїв і систем, у тому числі за допомогою зовнішніх засобів, представлена на рис. 3.4.

Виходячи із загальної класифікації методів діагностування обчислювальних пристроїв, для кожного окремого пристрою обирають найбільш прийнятний та ефективний метод.

Методологія тестового діагностування

На сьогодні найбільш використовуваними є функційне, тестове і змішане діагностування МПП, визначення яким було дано у викладеному раніше матеріалі.

Функційне діагностування. При реалізації цього методу вхідними впливами, що надходять на об'єкт діагностування, є робочі впливи, які передбачені робочим алгоритмом функціювання об'єкта.

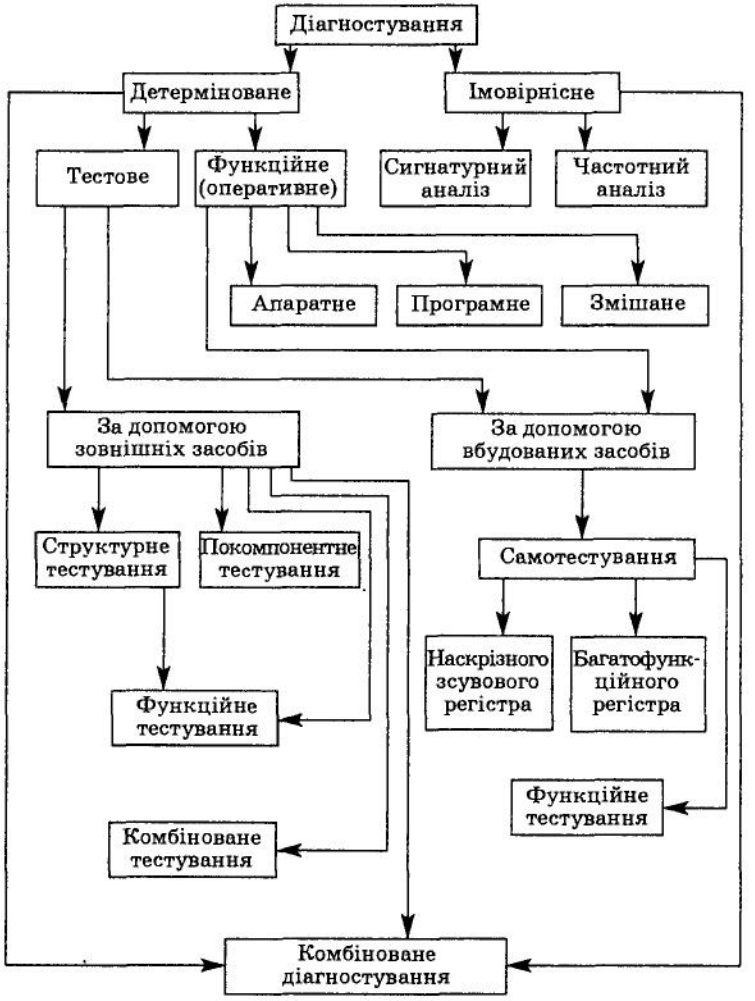


Рис. 3.4. Класифікація методів діагностування обчислювальних пристроїв

Тестове діагностування. Цей метод дещо відрізняється від функційного. Його реалізація передбачає подавання на ОД тестових впливів, що відрізняються від робочих. При цьому відповідні реакції ОД набувають специфічного характеру, якого не буває в робочих режимах. У розрізі цього методу найбільшого поширення набули структурне (загальне), покомпонентне і комбіноване діагностування.

Тестове структурне (загальне) діагностування. Його здійснюють у цілому. На входи МПП подають тестові впливи, а на виходах реєструють відповідні реакції, за якими аналізують правильність виконання функцій, що реалізуються цифровою структурою пристрою.

Необхідними умовами здійснення структурного методу тестування є вимірювання і аналіз різноманітних діагностичних параметрів ОД, а також виконання складних процедур і програм перевірок, що, в свою чергу, ускладнює процедуру діагностування.

Тестове покомпонентне діагностування. Його суть полягає у проведенні послідовних перевірок кожного компонента структури окремо при виконанні умови виключення взаємного впливу зв'язаних з ним компонентів. Компонентами в даному випадку є вмонтовані в друковану плату інтегральні схеми та ін.

Діалектично методи структурного і покомпонентного тестування взаємодоповнюють один одного з точки зору повноти діагностування і глибини пошуку несправностей. Тому комбінація цих методів є виправданою. Під час її здійснення структурні і покомпонентні перевірки поєднують за певними стратегіями.

Найбільш повно комбінацію цих методів реалізують у методі тестового комбінованого діагностування обчислювальних пристроїв. При цьому комбіноване тестування не ототожнюють з простим поєднанням структурних і покомпонентних перевірок. Тестове комбіноване діагностування описує власна теорія, стратегії і алгоритми.

Тестове комбіноване діагностування. Під ним розуміють тестування, що полягає у проведенні послідовних структурних і покомпонентних перевірок як ОД у цілому, так і його фрагментів і компонентів спільними апаратурно-програмними засобами. Це здійснюють з метою спрощення процесу тестування за рахунок його максимального наближення до робочих режимів роботи обчислювального пристрою чи системи, які діагностують, пошуку несправностей різних класів, досягнення необхідної глибини пошуку несправностей.

Методи тестового діагностування реалізують в основному за допомогою зовнішніх щодо МПП засобів. Внутрішні щодо ОД засоби діагностування в основному використовують для контролю і діагностування функцій, які повинен виконувати МПП, правильності передавання інформації у вигляді кодів (наприклад, контроль на парність чи непар-

ність) через інформаційні магістралі й запису її у функційні вузли МПП або зчитування з них.

Основними принципами методології тестового структурного діагностування є:

— подавання тестових впливів і зняття відповідних реакцій з крайових з'єднувачів ОД, щоб перевірити правильність виконуваних функцій, які реалізує структура МПП;

— знання законів функціювання кожного структурного фрагмента МПП;

— вимірювання й аналізування різноманітних діагностичних показників МПП або їх складових;

— імітація функціювання МПП за допомогою додаткового обладнання імітаторів і навантажень.

Методологія тестового покомпонентного діагностування ґрунтується на таких основних принципах:

— подавання тестових впливів на внутрішні точки структури МПП;

— інваріантність тестування компонентів досягають вимушеним наведенням сигналів тестових впливів на входи активних компонентів;

— непошкодження компонентів у процесі діагностування, що реалізують шляхом встановлення на відповідних компонентах початкових умов тестування.

Методологія тестового комбінованого діагностування МПП ґрунтується на структурному і покомпонентному тестуванні, однак має свої принципи, основними з яких є:

— синтез математичної моделі структури МПП за їх фрагментами, що можуть бути подані на різних рівнях опису;

— подавання сигналів тестових впливів і приймання сигналів відповідних реакцій як з крайових з'єднувачів, так і з внутрішніх контрольних точок МПП;

— максимальне наближення форми сигналів тестових впливів до робочих (тих, якими оперує МПП під час виконання своїх основних функцій);

— аналіз сигналів відповідних реакцій за різноманітними діагностичними параметрами;

— покомпонентно-структурне тестування інтегральних схем підвищеного ступеня інтеграції, що входять до складу МПП. Цей принцип передбачає структурне (загальне) тестування ВІС і НВІС у змонтованому типовому елементі заміни (ТЕЗі) — МПП з функційно закінченими вузлами, де заздалегідь виключають взаємний вплив зв'язаних компонентів. Шини МПП (адреси, даних та ін.), де можна, переводять у високоімпедансний стан (стан, при якому виходи мікросхеми мають як завгодно великий опір), а на тих лініях, де це неможливо, наводять логічну «1». Потім за допомогою багатоконтактного пристрою на входи ВІС (НВІС) подають послідовності сигналів тестових впливів, а з виходів знімають послідовності сигналів відповідних реакцій.

Порівнюючи структурне і покомпонентне тестування, можна простежити низку переваг і недоліків, що властиві кожному з методів. Так, головними перевагами структурного тестування є простота підключення зовнішніх засобів діагностування до ОД, менша кількість каналів зв'язку обладнання діагностування з ОД та вища швидкість перевірки за принципом «придатний-непридатний». Недоліками структурного тестування є складність і порівняно велика трудомісткість розроблення тестових програм за вимогою моделювання об'єкта в цілому, необхідність попереднього генерування, вимірювання і аналізу різноманітних вхідних і вихідних параметрів сигналів для діагностування обчислювального пристрою та ін.

Головною перевагою покомпонентного тестування порівняно зі структурним є відносно мала трудомісткість складання тестів, оскільки їх складають для кожного компонента окремо. Недоліками такого тестування є проведення перевірок у статичних режимах, а вони відрізняються від робочих, неможливість виявлення помилок синхронізації, обов'язкова наявність окремого спеціального багатоконтактного пристрою, що часто знижує надійність контактування його контактів з контактами інтегральних схем, необхідність зняття захисного шару на друкованій платі, коли цей метод використовують на етапі експлуатації, та ін.

Комбінації і стратегії систем тестування

Для підвищення ефективності процесу діагностування на виробництві використовують різні комбінації систем тестування в одній технологічній лінії та різні стратегії. Серед них — послідовна (тандемна), паралельна і паралельно-послідовна (паралельно-тандемна).

Рис. 3.5. Послідовна конфігурація і відповідна їй стратегія тестування

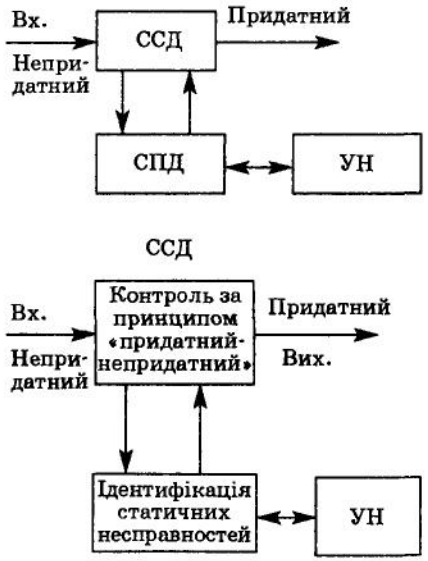


Рис. 3.6. Паралельна конфігурація і відповідна їй стратегія тестування

За *послідовної стратегії* (рис. 3.5) за допомогою систем покомпонентного діагностування (СПД) виявляють насамперед статичні несправності. Якщо такі є, то їх усувають у процесі усунення несправності (УН). Після цього ОД подають на систему структурного діагностування (ССД) для контролю за принципом «придатний-непридатний».

При *паралельній стратегії* (рис. 3.6) за допомогою ССД проводять передусім відбракування за принципом «придатний-непридатний». Потім несправні ПКПСІ надходять для

ідентифікації несправностей статичного типу на СПД, після їх усунення вони повертаються для повторного контролю за принципом «придатний-непридатний».

При *паралельно-послідовній стратегії* (рис. 3.7) за допомогою першої ССД здійснюють спочатку контроль за принципом «придатний-непридатний». Потім вироби, що не пройшли контроль, надходять на СПД для ідентифікації несправностей статичного типу. Після їх усунення вироби передають на іншу ССД для контролю за принципом «придатний-непридатний» та ідентифікації окремих класів динамічних несправностей.

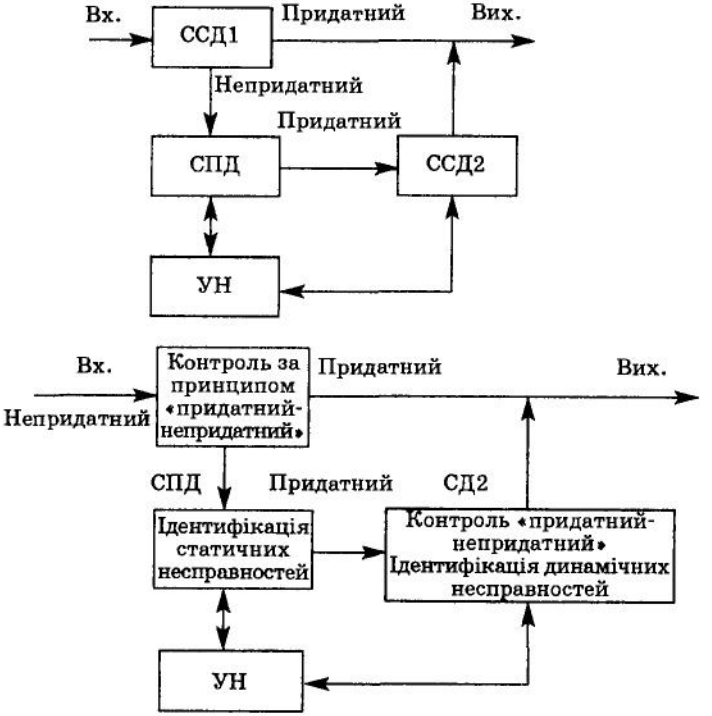


Рис. 3.7. Паралельно-послідовна конфігурація і відповідна їй стратегія тестування

При реалізації тестування ПКПСІ за допомогою системи комбінованого діагностування (СКД) воно істотно спрощується. Конфігурація технологічної лінії для всіх стратегій буде одна й та сама. До її складу входять СКД і дільниця (технологічне місце) усунення несправностей (рис. 3.8).

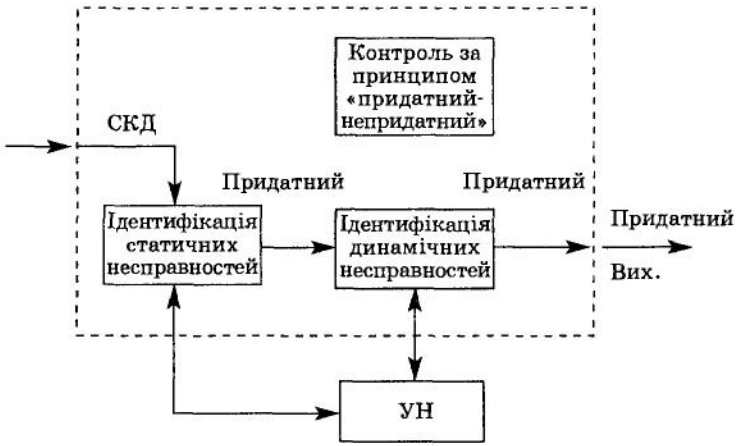


Рис. 3.8. Послідовна стратегія тестування, що реалізується на СКД

При послідовній стратегії тестування, що реалізується на СКД, контроль за принципом «придатний-непридатний» не проводять. Спочатку ідентифікують і усувають несправності статичного типу. Якщо їх немає, то ті самі операції здійснюють з несправностями динамічного типу і на цій основі роблять висновок про придатність виробу. Така стратегія найефективніша, коли кількість несправностей статичного і динамічного типів досить велика і вироби випускають порівняно невеликими серіями.

У паралельній стратегії тестування ПКПСІ, що реалізується на СКД, виокремлюють два типи (рис. 3.9).

1. Здійснюють контроль за принципом «придатний-непридатний» (рис. 3.9, а). У відбракованих виробах ідентифікують і усувають несправності статичного типу. Після цього вони повторно проходять контроль за принципом «придатний-непридатний». Ця стратегія найефективніша, якщо несправності динамічного типу мають незначну питому вагу і вироби випускають великими серіями.

2. Здійснюють контроль за принципом «придатний-непридатний» (рис. 3.9, б). У відбракованих виробах ідентифікують і усувають несправності статичного типу. Після цього ті самі операції здійснюють з несправностями динамічного типу. На цій основі роблять висновок про придатність виробу.

Така стратегія найефективніша, коли ігнорують ідентифікацію несправностей динамічного типу, і це призво-

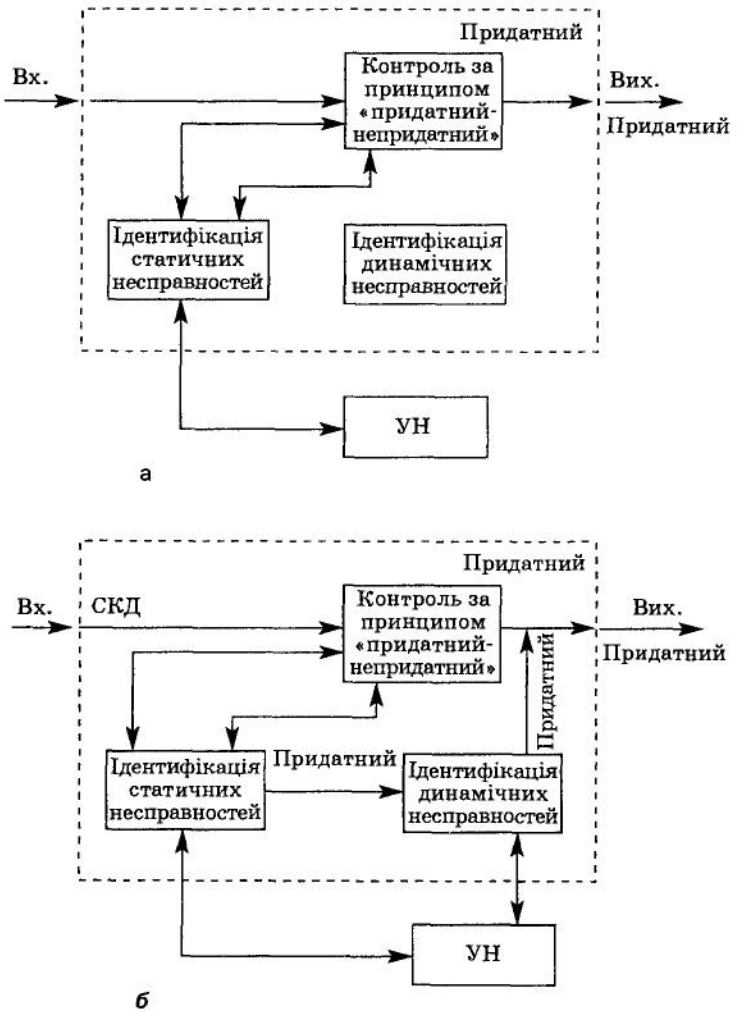


Рис. 3.9. Паралельна стратегія тестування ПКПСІ, що реалізується на СКД

дить до необґрунтованих втрат при різній серійності виробництва ПКПСІ.

Стратегії тестового комбінованого діагностування порівняно зі стратегіями поетапного застосування систем структурного і покомпонентного діагностування простіші й ефективніші. Цього досягають, використовуючи насам-

перед тільки одну систему діагностування — СКД у процесі діагностування ПКПСІ. Це дає змогу уникнути необхідності передавати вироби для тестування від однієї системи до іншої, внаслідок чого зменшується кількість операторів і обслуговуючого персоналу.

Крім того, комбінації ССД і СПД орієнтовані в основному на ідентифікацію несправностей статичного типу. А використовуючи стратегії СКД та ідентифікуючи водночас несправності статичного і динамічного типів, можна підвищити достовірність контролю та діагностування ПКПСІ.

Отже, за послідовної стратегії тестування, що реалізується на СКД, контроль за принципом «придатний-непридатний» взагалі не проводять, а за паралельної стратегії тестування не потрібно проводити два рази контроль за принципом «придатний-непридатний» для відбракованих виробів. Це зменшує час і витрати на тестування. Все засвідчує очевидні переваги стратегії СКД.

Для розроблення алгоритмів пошуку несправностей за тестового комбінованого діагностування ПКПСІ використовують такий ієрархічний підхід:

1) визначення несправних ПКПСІ шляхом відбракування виробів за принципом «придатний-непридатний»;

2) визначення несправних фрагментів ПКПСІ певних функційних вузлів;

3) визначення несправних компонентів і елементів у фрагменті структури ПКПСІ.

Етап 1. Проводять структурне тестування ОД шляхом подавання тестових впливів, що найбільш наближені до робочих. Тестові послідовності складають на основі інформації про алгоритм роботи пристрою. Це спрощує їх складання.

Етап 2. Здійснюють тестування фрагментів структури пристроїв (фрагментів ТЕЗів) шляхом подавання сигналів тестових впливів, що генерують відповідно до часової діаграми роботи цих фрагментів, і зчитування відповідних реакцій в запрограмований момент. Модель фрагмента складають на алгоритмічному або функційному рівні, що робить простішим складання тестових послідовностей.

Етап 3. Проводять покомпонентно-структурне тестування, передусім компонентів підвищеного ступеня інтеграції — ВІС і НВІС фрагмента структури. Тестову програму для діагностування розробляють на основі їх моделей.

Сигнали тестових впливів формують відповідно до часової діаграми роботи ПКПСІ.

На всіх етапах можна зняти еталонні реакції зі справного ПКПСІ чи його фрагментів, щоб надалі порівняти їх з відповідними реакціями пристроїв, які діагностують.

За можливості подавати сигнали тестових впливів і приймати відповідні реакції тільки з крайових чи інших з'єднувачів здійснюють структурне тестування об'єкта. Для розроблення тесту використовують математичну модель усієї структури ОД. Завдання моделювання об'єкта і розроблення тесту суттєво спрощується, коли використовують багатоконтактні пристрої, зокрема типу кліпс. Тоді достатньо розробити математичні моделі й тести для окремих компонентів структури, які є стандартними компонентами мікропроцесорних комплектів і серій ІС. Часто їх розробляють заздалегідь і зберігають у бібліотеці тестів компонентів.

Комбінації подавань сигналів тестових впливів і знять відповідних реакцій з крайових з'єднувачів і внутрішньосхемних контрольних точок дають змогу протестувати окремі компоненти і фрагменти схеми ОД. Здійснюють покомпонентно-структурне тестування ВІС і НВІС вузла за допомогою контактного пристрою типу кліпса або перехідних колодок. Заздалегідь нейтралізують взаємний вплив зв'язаних компонентів шляхом подання необхідних сигналів з крайових з'єднувачів і внутрішніх контрольних точок.

Переважно при складанні тесту ПКПСІ в розпорядженні спеціалістів не завжди є опис їх детальної внутрішньої структури. Формальний опис ВІС і НВІС на функційному або алгоритмічному рівні дає змогу, не знаючи їх детальної внутрішньої структури і подробиць функціювання, визначити послідовність передавання інформації і застосувати її при діагностуванні.

Тестове комбіноване діагностування ПКПСІ, порівняно зі структурним і покомпонентним тестовим діагностуванням, дає змогу:

— спростити моделювання ПКПСІ як об'єктів діагностування, тому що можна отримати доступ до частини внутрішніх контрольних точок не через крайові з'єднувачі, а за допомогою простіших багатоконтактних пристроїв;

— емулювати (замінювати іншими засобами) роботу пристрою (блока), складовою частиною якого є діагностований ОД;

— тестувати ВІС і НВІС у складі цифрової структури (змонтованої друкованої плати);

— ідентифікувати, крім несправностей статичного типу, несправності динамічного типу, в тому числі нелогічні динамічні несправності;

— автоматизувати виявлення дефектних компонентів у цифрових шинних структурах без порушення друкованого монтажу ПКПСІ;

— виключити деякі операції параметричного контролю, зокрема оцінку часу затримки поширення сигналів, величини вимірювання струму сигналів відповідних реакцій та ін.;

— скоротити сумарні витрати часу і засобів на діагностування ПКПСІ.

Із зазначених можливостей зрозуміло, що тестове комбіноване діагностування є найефективнішим.

Особливості МПП як об'єктів діагностування

Подальший розвиток тестового комбінованого діагностування Ц і МПП зумовлений побудовою їх на базі компонентів підвищеного ступеня інтеграції — ВІС і НВІС. При цьому окреслилися особливості МПП як об'єктів діагностування:

— великі витрати часу на побудову тестів, що зумовлено підвищеним ступенем інтеграції компонентів МПП і, як правило, виключає синтез тестів на вентильному або автоматному рівні;

— шинна структура МПП та складність ідентифікації несправностей у таких структурах;

— відсутність доступу до внутрішніх контрольних точок ВІС і НВІС мікропроцесорного пристрою через крайові з'єднувачі. Це зумовлює формування тестових послідовностей значної довжини. Для цього необхідне знання детальної внутрішньої структури МПП і його компонентів, що не завжди відоме розробникам тестів;

— необхідність діагностування МПП на максимальних робочих частотах, які сягають кількох сотень і тисяч МГц, оскільки відсутність збоїв і відмов на порівняно низьких частотах не забезпечує безвідмовну роботу пристрою або системи на максимальних робочих частотах;

— синхронізація роботи певних пристроїв чи компонентів з головним мікропроцесором (мікропроцесорами) з

метою забезпечення необхідних часових співвідношень між подаванням на ОД сигналів тестових впливів і зчитуванням відповідних реакцій;

— різні підходи до реалізації діагностування протягом життєвого циклу МПП, зокрема на етапах проектування, виробництва, експлуатації.

Однією з важливих особливостей діагностування Ц і МПП, побудованих на базі КПСІ, є їх тестування за допомогою комплексу апаратних і програмних засобів, що вимагає знання принципів функціювання та структури програмного складу ОД.

При цьому під час експлуатації і ремонтів широко використовують програмне тестування, що передбачає використання діагностичних програм, розміщених на зовнішніх носіях.

Такі програми належать до третього рівня і мають вузьку спеціалізованість. Вони потребують значного часу для різноманітних перевірок. Найчастіше їх використовують, коли несправність проявляється при певній кодовій операції або дестабілізуючих факторах протягом тривалого часу роботи. Ці програми можуть конкретизувати місце прояву несправностей в комп'ютерах. Окрім них, існує ще чимало програмних продуктів. Зокрема, для PC-комп'ютерів — це програмні засоби діагностування CHECKIT, утиліта DISK DOKTOR з пакета NU 7.0 NDIAGS. Вони дають змогу перевіряти процесор, системну плату, підсистему пам'яті, клавіатуру, порти вводу-виводу, відео- і звукові плати та інші складові комп'ютера. Всі текстові утиліти мають досить зручні інтерфейси.

Серед програмних засобів дуже важливе значення має перевірка інформаційних каналів на віруси. Так, утиліта NORTON ANTIVIRUS із NU 7.0 NDIAGS розпізнає біля 700 вірусів та 1400 їх модифікацій.

Часто для розв'язання завдань, пов'язаних з пошуком несправностей в комп'ютерах, використовують програми налагоджувача DEBUG чи AFD.

Дії, ініційовані налагоджувачами, виконуються автоматично діагностичними тестами. Однак можна перейти на ручний режим або записувати програми в налагоджувачі, використовуючи мнемокоди. Налагоджувач дає змогу переглядати і змінювати інформацію в потрібних ділянках пам'яті, регістрах мікропроцесора і портах введення-виведення, запускати програми на виконання, знайомитися із вмістом секторів накопичувачів на маг-

нітних дисках (НМД) і записувати на диск необхідну інформацію. (Докладніше програмне тестування описано у розділі 4.)

Моделі несправностей Ц і МПП

Розглянуті раніше моделі несправностей статичного типу досить прості. Моделі ж прояву несправностей динамічного типу складніші.

Підходи до дослідження цієї проблеми, що ґрунтуються на описі зміни часової послідовності початкових і кінцевих значень тестових сигналів («0» чи «1») і послідовності миттєвих стрибків («1»:«0» → «1») і («0»:«1» → «0»), які здійснюються у випадкові моменти часу і виключають інтервали перемикання стану «0» → «1» («1» → «0»), прийнятні для опису цифрових автоматів, побудованих на компонентах малого ступеня інтеграції. Однак вони непридатні для опису прояву несправностей динамічного типу в цифрових компонентах підвищеного ступеня інтеграції та обчислювальних пристроях, побудованих на їх основі. Такі компоненти — це складніші функційно закінчені пристрої, достовірність роботи яких залежить від часових інтервалів, що задаються генератором тактових імпульсів. Зі збільшенням частоти генерації тактових імпульсів не можна ігнорувати час переходу цифрової структури з одного стану в наступний.

Не беруть до уваги час переходу структури з одного стану в наступний при розгляді ОД як n зв'язаних між собою синхронних блоків.

Якщо розглянути математичну модель прояву несправностей динамічного типу в ПКПСІ, в якій враховують час переходу структури з одного стану в наступний, то це буде цифрова структура — набір асинхронно-синхронних функційних вузлів. Відомо, що асинхронність зводиться до k-синхронності (асинхронність кратна синхронності).

Модель структури справного пристрою подають на функційно-динамічному рівні:

$$M = < X, Y, Q, F_\lambda, S_\phi, T >, \qquad (3.12)$$

де X, Y, Q — множини вхідних, вихідних і внутрішніх станів структури; F_λ — множина функцій виходів функційних вузлів структури; S_ϕ — функційна схема структури, що відображена графом; T — тактові інтервали роботи пристрою. Можна припустити, що в системі тестового діагностування реєстрацію відповідних реакцій з ОД проводять у кожному такті T роботи пристрою. Тестові дії подають на ОД на початку такту, другу частину тактового інтервалу витрачають на перехідні процеси в структурі, а третю, що залишилася від усієї тривалості такту, витрачають на реєстрацію відповідних реакцій. Тому головна умова достовірної реєстрації сигналів відповідних реакцій — це тривалий період T для здійснення цих трьох процесів. Якщо знехтувати часом встановлення тестових дій на входах структури, то:

$$T_i = t_i + \tau_i;$$
$$X(T_i) = \{X_1(T_i), X_2(T_i),\ldots,X_n(T_i)\};$$
$$Q(T_i) = \{Q_1(T_i), Q_2(Ti),\ldots,Q_r(T_i)\};$$
$$Y(t_i + \tau_i) = \{Y_1(t_i + \tau_i), Y_2(t_i + \tau_i),\ldots,Y_k(t_i + \tau_i)\};$$
$$Y(t_i + \tau_i) = F(X(T_i), Q(T_i)),$$

де T_i — i-й тактовий інтервал роботи пристрою; t_i — тривалість перехідних процесів в i-му такті роботи пристрою; τ_i — тривалість стану пристрою, що встановився в i-му такті.

Якщо в структурі пристрою є несправності, то її модель подають так:

$$M = < X, Y', Q', F_\lambda', S_\phi, T >, \qquad (3.13)$$

де Y', Q' — множини спотворених вихідних і внутрішніх станів структури несправного пристрою; F_λ' — множина спотворених функцій виходів функційних вузлів структури несправного пристрою. Тоді:

$$Y'(t_i + \tau_i) = F_\lambda'(X(T_i), Q'(T_i)). \qquad (3.14)$$

При цьому:

$$Q'(T_i) = \{Q_1'(T_i), Q_2'(T_i),\ldots,Q_r'(T_i)\};$$
$$Y'(t_i + \tau_i) = \{Y_1'(t_i + \tau_i), Y_2'(t_i + \tau_i),\ldots,Y_k'(t_i + \tau_i)\}.$$

Функція F_λ відома і визначена цифровою структурою об'єкта, для якого вона еталонна (франц. étalek — виставляти, показувати). Якщо $F_\lambda' \neq F_\lambda$, то в структурі пристрою є несправність, яку представляють як $Y'(t_i + \tau_i) \neq Y(t_i + \tau_i)$. Це призводить до того, що множина зареєстрованих кодів відповідних реакцій не відповідає множині наборів еталонних відповідних реакцій. Множину несправностей динамічного типу H_g цифрової структури пристрою подають як підмножину логічних динамічних несправностей $H_л$ і підмножину нелогічних динамічних несправностей $H_{нл}$.

Тоді:

$$H_g = H_{л.} \cup H_{н.л.} \qquad (3.15)$$

Ідентифікуючи несправності цифрових пристроїв з компонентами підвищеного ступеня інтеграції, необхідно насамперед визначити частотні області, в яких будуть проявлятися несправності динамічного типу. Якщо відповідні реакції не відповідатимуть еталонним, це можливо з огляду на такі причини: 1) моменти реєстрації сигналів відповідних реакцій потрапляють в інтервал t_i; 2) якщо $t_i \to T_i$, відповідно $t_i \to 0$ через збільшення тривалості перехідних процесів в i-му такті роботи пристрою.

У першому випадку можна виправити ситуацію, якщо реєструвати сигнали відповідних реакцій у різних точках тактового інтервалу. В другому — слід констатувати, що в структурі пристрою є логічні динамічні несправності або недостатня тривалість τ_i стану пристрою, який встановився в i-му такті за причини перевищення частоти подачі тестових послідовностей.

Визначають мінімальну $\nu_{p.\min}$ і максимальну $\nu_{p.\max}$ — робочі частоти подачі тестових дій і прийому відповідних реакцій на ОД у динамічних режимах діагностування структури пристрою.

Якщо відомо тільки тривалість t перехідних процесів усієї структури і тривалість τ_ϕ стану, що встановився, то:

$$\nu_{p.\min} = \frac{l}{t + \tau_{\max}} \text{ і } \nu_{p.\max} = \frac{l}{t + \tau_{\min}}.$$

У випадку, коли відомі тривалості перехідних процесів t_ϕ кожного з функційних вузлів структури і тривалості τ_ϕ станів, що встановилися, кожного з вузлів, то:

$$\nu_{p.\min} = \frac{l}{\sum_{j=1}^{l} t_{\phi.j} + \sum_{i+1}^{l} \tau_{\phi.j}} = \frac{l}{\sum_{j=1}^{l} (t_{\phi.j} + \tau_{\phi.j})} = \frac{l}{l \cdot T},$$

де l — кількість функційних вузлів структури; $t_{\phi.j}$ — тривалість перехідних процесів j-го функційного вузла; $\tau_{\phi.j}$ — тривалість стану, що встановився, j-го функційного вузла.

Максимальна робоча частота тестування:

$$\nu_{p.\max} = \frac{l}{\sum_{j=1}^{l} t_j + \tau_{\min}}.$$

При зменшенні частоти відносно $\nu_{p.\min}$ вважають, що тестування здійснюють у статичному режимі.

Отже, прийнятною виявилась ідея опису логічних і нелогічних динамічних несправностей ідентичними моделями. Це можливо, якщо несправності обох класів призводять до одних і тих самих наслідків — появи спотворених логічних значень сигналів на виходах пристрою.

Слід розглянути, зокрема, несправності, що зумовлені «змаганнями» сигналів, не як завади, які спотворюють генеровані структурою логічні сигнали, а як логічні сигнали, що з'явилися додатково і які змінюють множину $\{Y\}$ вихідних змінних у множину $\{Y'\}$, а також спотворюють множину функцій виходів фукційних вузлів $\{F_\lambda\}$ у $\{F_\lambda'\}$.

У цьому випадку множину несправностей, що зумовлена «змаганнями» сигналів, визначають як:

$$(H_{н.л})_з = F_\lambda'(X(T_i), Q(t_i)). \qquad (3.16)$$

Щоб проаналізувати методику виявлення несправностей цього типу, необхідно задати сигнал завади, який виник внаслідок «змагань», тривалістю τ і амплітудою рівня логічної «1» чи «0». Мінімальна тривалість $\tau_{\min}$ цього сигналу повинна бути такою, щоб елементи цифрової структури не сприйняли його як логічний сигнал (такі дані є в довідниках), в іншому разі він не буде причиною несправності. У відомих системах тестового діагностування допускають, що сигнали відповідних реакцій не змінюють свого логічного рівня протягом усього періоду реєстрації T. Нехай сигнал завади, який виник, за тривалістю менший або дорівнює періоду реєстрації T ($\tau_{\min} \le T$). Складність його виявлення в тому, що виникає він протягом періоду реєстрації T у коротшому проміжку часу, ніж T. Це значить, що система тестового діагностування повинна реєструвати сигнали такого типу завад у різні моменти відносно початку такту прийому відповідних реакцій, що показано на рис. 3.10.

Ідентифікацію несправностей, причини яких «змагання» сигналів, проводять шляхом проб реєстрації сигналу завади в різні моменти часу щодо початку такту прийому відповідних реакцій. Якщо сигнал зареєстрований у певному інтервалі періоду T, то в цифровій структурі є несправності даного підкласу. Завдання зводиться до визначення кроку дискретизації $T_{диск}$ періоду T або коефіцієнта поділу k періоду T реєстрації відповідних реакцій.

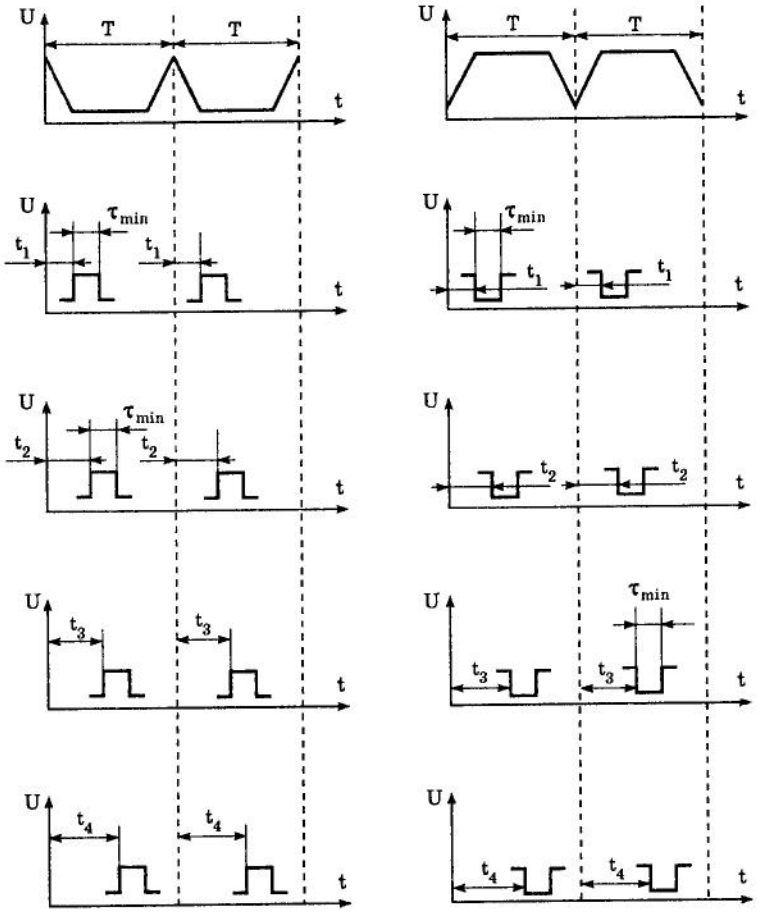


Рис. 3.10. Часові діаграми, що моделюють сигнал завад, зумовлений «змаганнями» сигналів

В аналізі часових діаграм (рис. 3.11), що дають уявлення про взаємне розташування сигналу завади і кроку дискретизації $T_{диск}$, можливі два випадки:

$$\tau_{\min} > T_{диск};\ \tau_{\min} \le T_{диск}.$$

Для $\tau_{\min} > T_{диск}$, якщо сигнал завади є в трьох кроках дискретизації, він беззаперечно був би зареєстрований у другому кроці ($\tau_{\min} < T_{диск}$). Однак, якщо сигнал завади

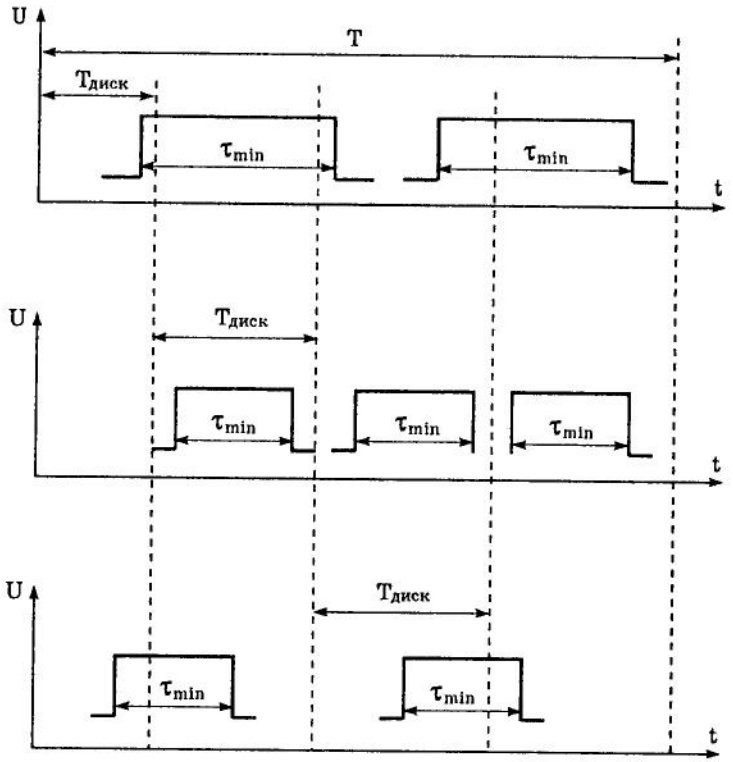


Рис. 3.11. Часові діаграми, що моделюють взаємне розташування сигналу завади і кроку дискретизації

фіксують за час, що дорівнює двом крокам, то він може бути і не зареєстрований, оскільки $T_{диск} > \tau_{\min}$.

Для $\tau_{\min} \le T_{диск}$ можливі два варіанти: 1) сигнал завади повністю знаходиться в одному з періодів дискретизації $T_{диск}$; 2) сигнал завади міститься у двох періодах $T_{диск}$, причому в одному з кроків — більше половини (чи половина) його тривалості. Тоді завдання можна вирішити, розв'язавши систему рівнянь:

$$\begin{cases} T = k \cdot T_{диск}, \quad k = 1,2,\ldots,n; \\ T \ge T_{диск}; \\ T_{диск} = \dfrac{\tau_{\min}}{2} + \Delta\tau_{\min}, \end{cases} \tag{3.17}$$

де $\Delta\tau_{min}$ — діапазон зміни тривалості сигналу завади (береться з довідникових видань).

Із системи рівнянь отримують:

$$k = \frac{T}{\frac{\tau_{min}}{2} + \Delta\tau_{min}} = \frac{2T}{\tau_{min} + 2\Delta\tau_{min}} = \frac{2}{\nu(\tau_{min} + 2\Delta\tau_{min})}, \quad (3.18)$$

де ν — частота реєстрації відповідних реакцій.

З формули 3.18 випливає, що для описаного методу ідентифікації нелогічних динамічних несправностей коефіцієнт поділу періоду T тим менший, чим вища частота реєстрації відповідних реакцій. Крім того, достовірність реєстрації сигналу завади, причина появи якого «змагання», залежить від того, як змінюється діапазон тривалості цього сигналу, що, як правило, дорівнює сумі тривалостей переднього і заднього фронтів імпульсних сигналів, які генеруються логічними елементами конкретного типу логіки.

Інерційність напівпровідникових приладів, їх паразитні ємності є причиною того, що кожне перемикання супроводжується перехідними процесами. В міру зношування компонентів ПКПСІ, а також при зміні температури навколишнього середовища, в них відбуваються спотворення (подовження) фронтів імпульсних сигналів. Тобто спотворюються часові параметри сигналів. Це, у свою чергу, призводить до виникнення помилок при обробленні інформації.

За допомогою подавань на об'єкт логічних сигналів з постійними параметрами й аналізу відповідних реакцій, тільки як кодових наборів, несправності, пов'язані зі спотвореннями фронтів імпульсних сигналів, виявити неможливо. У таких випадках застосовують параметричний контроль, але його реалізація досить складна і не завжди економічно вигідна. Крім того, низка операцій параметричного контролю не піддається автоматизації.

Існує часо-імпульсний спосіб ідентифікації несправностей, пов'язаних з подовженням фронтів імпульсних сигналів і відповідно зі збільшенням часу затримки поширення імпульсних сигналів під час оброблення їх цифровою структурою.

Суть способу полягає в подаванні на входи компонентів структури часо-імпульсних сигналів певного логічного рівня й аналізі сигналів відповідних реакцій як за логічним рівнем, так і за тривалістю.

З огляду на бурхливий розвиток проектування і виробництва мікропроцесорів (МП) та МПП окреслилися їх нові особливості як ОД. Серед них:

— обмеженість доступу до контрольних точок МПП і виводів компонентів, у зв'язку зі значним зменшенням кроку розведення друкованих провідників ТЕЗів, збільшенням кількості виводів (наприклад, від 40 до 370 (для МП) при порівняно однакових розмірах корпусів НВІС), що призвело до неможливості використання багатоконтактних пристроїв типу поле із цвяхів та кліпса;

— неможливість розроблення моделей МПП при описі його як ОД на певних етапах життєвого циклу пристрою, зокрема на етапі виробництва, коли немає інформації про внутрішню детальну структуру (фірми-виробники МП і МПП таких відомостей, як правило, не приводять, вважаючи їх комерційною таємницею);

— ускладнення процесу діагностування МПП, з огляду на появу нових, невідомих досі класів і типів несправностей;

— необхідність створення об'ємних баз даних для запам'ятовування інформації про структуру та елементи моделей МПП і типи несправностей;

— створення нового компонента процесу діагностування МПП — бази знань, без якої неможливо ідентифікувати більшу частину несправностей та реалізувати самонавчання систем контролю і діагностування;

— ускладнення процесу діагностування сучасних МПП через використання в них D-, S-, SD-RAM високої швидкодії (2—10 нс) та великої ємності (десятки і сотні Мбайт);

— наявність у складі МПП компонентів програмованих логічних матриць (ПЛМ), що ускладнює процес побудови тестів, з огляду на нестандартність МПП, побудованих з використанням ПЛМ, і зміну ступеня інтеграції МПП при введенні в їх структуру ПЛМ;

— ускладнення функцій, що виконують компоненти і МПП в цілому, зокрема паралельне функціювання складових МПП.

Прикладом пристроїв і систем, яким властива ця особливість, можуть бути комп'ютери, побудовані на базі мікропроцесорів Pentium II — Pentium IV, в яких є кілька конвеєрів (U-, V-конвеєри та ін.), що виконують інструкції паралельно і незалежно один від одного. Кількість інструкцій, які вони виконують за одиницю часу, значно збільшилась порівняно з попередніми поколіннями мікропроцесорів.

Інша особливість, що залежить від фактора часу, — це пошук несправностей у реальному часі. Використовуючи МПП за їх основним призначенням, цей процес переривають з метою пошуку несправностей, а потім знову використовують МПП за основним призначенням.

У наш час суттєво скорочують строки проектування, підготовки до виробництва і самого виробництва МПП. Тому необхідно переглянути стратегію їх контролю і діагностування.

Частина особливостей процесу діагностування, що з'явилася на початку впровадження МПП (80-ті роки) у технічні системи, перестала відігравати значну роль. Стали другорядними проблеми, пов'язані з обмеженістю оперативної і зовнішньої пам'яті ЕОМ систем діагностування. Достатній об'єм пам'яті сучасних комп'ютерів знімає проблему запам'ятовування блоків тест-векторів великої ємності (Мбайти). Потужність і швидкодія су-часних комп'ютерів, що використовують як керуючі ЕОМ, забезпечують швидке оброблення великої кількості інформації, але вони не можуть вирішити багатьох проблем контролю і діагностування МПП.

Принципи діагностування МПП

Особливості сучасних мікропроцесорних пристроїв і систем як об'єктів діагностування зумовили нові проблеми процесу тестування порівняно з моментом появи перших мікропроцесорів, наприклад 18086. Перераховані труднощі не можна подолати без істотної модифікації відомих методів тестування або розроблення нової методології тестового діагностування МПП.

Основними принципами нової методології тестування МПП є:

1) перевага програмного контролю і діагностування МПП над апаратним, призначеним тільки для виконання тестування мікропроцесорних пристроїв і систем;

2) відмова від методів структурного і покомпонентного тестового діагностування для сучасних МПП в класичному їх розумінні, тому що неможливо використати багатоконтактні пристрої при реалізації покомпонентного діагностування і перевірити, чи правильно виконує функції кожний компонент МПП, не знаючи їх детальної структури, при реалізації загального тестування;

3) використання для подавань сигналів тестових впливів і зняття сигналів відповідних реакцій тільки з тих контрольних точок, що відповідають контактам слотів (крайових з'єднувачів) і сокетів МП і МПП;

4) покомпонентно-структурне тестування НВІС мікропроцесорів, мікропроцесорних модулів і картриджів різного призначення;

5) багаторівневе моделювання МПП як об'єктів діагностування;

6) інтелектуалізація процесу тестового діагностування МПП на основі застосування компонентів штучного інтелекту (ШІ).

Отже, всі наведені принципи пов'язані з новими особливостями МПП і стосуються в основному етапів виробництва і експлуатації. Кожен з них відображає одну або й кілька особливостей сучасних МПП. Так, принцип 1 — перевага програмного діагностування над апаратним — пов'язаний зі складністю (особливо при переході на частоти синхронізації МПП у сотні і тисячі МГц) і високою вартістю (десятки і сотні тисяч доларів) зовнішніх засобів тестування. Тому природним є намагання розробників і споживачів сучасних МПП, зокрема персональних комп'ютерів, створити широкий спектр діагностичних програм.

Застосування принципів 2—4 нової методології діагностування зі зрозумілих причин не передбачене на етапі проектування компонентів, зокрема ВІС і НВІС, сучасних МПП, у той час, як принципи 5—6 у певних випадках застосовують тут.

Принцип 3 випливає з конструктивних особливостей МПП і певною мірою пов'язаний з принципом 2 тестового комбінованого діагностування. Саме його покладено в основу нової методології, тому її принципи мають багато спільного з методом тестового комбінованого діагностування.

Принцип 6 — новий і раніше в методології тестового діагностування МПП не розглядався, хоча спроби застосування окремих компонентів ШІ, зокрема експертних систем і самонавчання, були.

Нові принципи методології тестування МПП пов'язані насамперед з інтенсивним розвитком архітектури і технології розроблення та виготовлення МПП, з огляду на підвищення ступеня інтеграції. Завдання, що розв'язують при цьому, є важкоформалізованими і потребують посилення значущості інтелектуальних підходів.

Обчислювальні пристрої, зокрема МПП, призначені для розв'язання, як правило, одного або двох-трьох функційних завдань. Проте під час автоматизації виробництва або технологічних процесів потрібно вирішувати значно більшу їх кількість. Із цією метою пристрої об'єднують у системи. На цьому відповідно і ґрунтується діагностування багатопроцесорних систем.

3.3. Контроль і діагностування багатопроцесорних систем

Контроль і діагностування багатопроцесорних систем за можливостями реалізації є складнішими порівняно з тестуванням обчислювальних пристроїв. Це зумовлено наявністю і одночасним функціюванням кількох мікропроцесорів для вирішення конкретного завдання. Тому для підвищення ефективності процесу діагностування багатопроцесорних систем необхідно глибоко розуміти його суть.

Суть процесу діагностування багатопроцесорних систем

Тестове діагностування багатопроцесорних систем (БПС) здійснюють різними способами. Найпоширенішим з них є спосіб організації діагностування БПС за допомогою ядра.

Ядро — сукупність апаратних і програмних засобів, що забезпечують реалізацію тестового діагностування системи.

Ядро завжди вважають справним. Є кілька типів ядер. Серед них виокремлюють плаваюче, жорстке, централізоване і розподілене ядро.

Плаваючим ядром називають ядро, склад якого непостійний.

Жорстке ядро — ядро, склад якого постійний.

Централізованим вважають ядро, що організують на одному заздалегідь визначеному справному комп'ютері.

Розподілене ядро — ядро, що може бути організоване на будь-якому комп'ютері комп'ютерної системи (КС), навіть на неперевіреній. Істинний стан ситеми при цьому визначають у процесі взаємних перевірок.

Існують різні комбінації ядер: централізоване жорстке ядро, централізоване плаваюче ядро, розподілене жорстке ядро, розподілене плаваюче ядро. Можливі поєднання цих способів.

Одні з найпоширеніших — БПС із централізованим ядром, для побудови яких використовують спеціальний *обслуговуючий сервісний процесор*, призначений для контролю чи діагностування системи, а також підтримання працездатності основної системи, зокрема центрального процесора. Як обслуговуючий процесор, використовують

автономний комп'ютер, що має свою конструкцію. Щоб зменшити ймовірність спільних відмов обслуговуючого і основного процесора, їх часто функційно і конструктивно відділяють один від одного. Обслуговуючий процесор має своє окреме програмне забезпечення, яке сумісне з програмним забезпеченням основного процесора, але не перебуває під його контролем.

Апаратура обслуговуючого процесора складається з контролера (пристрою керування) із засобами порівняння кодових комбінацій, набору адаптерів, призначених для спрягання обслуговуючого процесора з пристроями основної системи, одного або кількох пультів керування з дисплеями. До системи можуть входити і дистанційні пульти керування, зв'язані з обслуговуючим персоналом через модеми, що служать для передавання сигналів через канали зв'язку. Вони знаходяться в центрі обслуговування різних обчислювальних систем.

Один обслуговуючий процесор може забезпечувати роботу кількох основних процесорів. Він виконує такі функції: запуск системи; керування системою; ідентифікація несправностей; тестування системи при її розробленні, інсталяції та під час роботи.

При відмовах системи і регламентних роботах обслуговуючий процесор починає роботу з того, що вводить з локальної пам'яті програмне забезпечення. Потім він виконує операції, які приводять у дію основний процесор (процесори).

У нормальному режимі обслуговуючий процесор використовує свої адаптери, під'єднані до центрального процесора, для керування всією системою. Він перевіряє його логіку і оперативну пам'ять, виконує регламентні діагностичні програми, інколи — керує енергоживленням і тепловим режимом. Ці функції здійснюються у перервах між роботою основного процесора.

Коли виникає відмова, обслуговуючий процесор відновлює обчислювальний процес основного процесора. Якщо це не вдалося, то обслуговуючий процесор відключає підсистему, що відмовила, і повідомляє про це обслуговуючий персонал. Інколи, усунувши причини відмови, він сам автоматично запускає основний процесор.

Важливу роль відіграють БПС, у яких здійснюють взаємне тестування процесорів. Вони належать до систем з розподіленим плаваючим ядром.

Взаємне діагностування процесорів у БПС проводять у ситемах, що мають спеціальні діагностуючі процесори, та в системах, де основні процесори діагностують один одного.

Одним з основних процесів при діагностуванні БПС є вимірювання. Це джерело найдостовірніших даних про функціювання і стан системи. Вимірювання спрямовують на діагностування як усієї системи, так і її підсистем. Для діагностування до системи підключають монітори, що вимірюють параметри станів системи: моменти зміни станів, тривалість перебування в них та ін. Виміряні дані від моніторів надходять до архіву, накопичуються там і потім обробляються.

Функціювання системи розглядають як зміну станів процесорів і ресурсів. Їх відображають у керуючих таблицях, які підтримують керуючі програми операційної системи. Стани процесів і ресурсів змінюються в моменти виконання спеціальних команд — звертання до супервізора привілейованих команд, за допомогою яких супервізор керує процесами і ресурсами, та ін. Крім того, стани пристроїв відображають відповідні електричні сигнали.

Класифікація моніторів

Монітори будують на основі різноманітних методів вимірювань і засобів їх реалізації. Серед них виокремлюють трасуючий і вибірковий методи.

Трасуючий метод. Полягає у реєстрації подій, що відповідають моментам зміни стану обчислювальної системи. До них належать початок і кінець введення завдання, крок завдання, етап процесорної обробки, звертання до зовнішньої пам'яті та ін. Монітори, що вимірюють процес функціювання системи цим методом, називають *трасуючими*.

Монітор реєструє події у вигляді подійного набору даних T, що складається з послідовності записів $S_1,S_2,S_3,\ldots$, які відповідають послідовності подій. Тут реєструють момент виникнення події, імена процесу $(I_1,I_2,I_3,\ldots)$ і ресурсу, з якими вона пов'язана, і параметри події, наприклад число байтів даних, що передаються. Використання пристрою системи можуть відображати діаграми.

Вибірковий метод. Його функцією є реєстрація стану обчислювальної системи в задані моменти часу через проміжки d. У моменти $t = n_\delta$, $n = 1,2,3\ldots$, *вибірковий монітор* реєструє стан системи, фіксуючи у відповідних записах дані з керуючих таблиць. Можна фіксувати електричні сигнали, що характеризують стан пристроїв системи в задані моменти. Отримані дані дають змогу оцінити тривалість процесів і розміри ресурсів, а також імовірність станів.

Тривалість станів визначають як $P_i = n_i/n$, де n_i — кількість вибірок, при яких зареєстровано даний стан i, n — тривалість процесу вимірювань, що визначають кількістю вибірок. Головна перевага вибіркового монітору — вимірювання безлічі швидких процесів за обмеженої швидкодії.

Залежно від виконуваних функцій монітори поділяють на універсальні і спеціалізовані.

Універсальний монітор. Реєструє всі події чи більшість із них. Тому подійного набору достатньо, щоб побудувати трасу процесів і використання ресурсів. Обсяг даних, що вимірюються, досить великий і становить 10^7 і більше байт на один процес. Універсальні монітори використовують в окремих випадках, наприклад для оцінювання конкретних системних або прикладних процесів.

Спеціалізований монітор. Реєструє певну частину подій чи станів. Це спрощує процес вимірювань. Такі монітори широко використовують для обліку виконаних робіт та оцінювання завантаженості ресурсів, а також як постійно діючі вимірювальні засоби для діагностування систем протягом усього часу їх функціювання.

Для діагностування комп'ютерних систем використовують *програмні монітори.* Вони реалізовані у вигляді програми, яку виконує ця сама комп'ютерна система. Залежно від їх місцезнаходження в комп'ютерній системі програмні монітори поділяють на вбудовані й автономні.

Вбудований програмний монітор. Це сукупність програм, що входять до складу операційної системи. Він має статус керуючої програми операційної системи і призначений для видачі мінімальних відомостей про систему.

Автономний програмний монітор. Це сукупність прикладних програм операційної системи. Вони дають змогу фіксувати широкий спектр подій при трасуючому і вибірковому методах вимірювань. Призначені для видавання розширених відомостей про систему.

Крім того, існують *апаратурні монітори.* Комплекс технічних і програмних засобів, призначених для діагностування процесів функціювання комп'ютерних систем. Монітор, аналізуючи електричні сигнали, що характеризують стан окремих блоків і пристроїв системи, отримує інформацію про її стан. Вимірювання проводять у визначених точках блоків і пристроїв за допомогою зондів. Дані вимірювань записують на певні носії та аналізують. Їх оперативні оцінювання використовують для контролю за правильністю функціювання системи та ідентифікації несправностей.

Локальні комп'ютерні мережі

В останні роки бурхливо розвивається мережна інтеграція. У світі встановлюють десятки тисяч локальних (лат. locus — місце) комп'ютерних мереж (ЛКМ). Відмова в їхній роботі спричиняє підприємствам, що їх використовують, значні збитки. За часткового або повного зупинення ЛКМ більш ніж на півдня два рази в місяць фінансові збитки становлять більше трьох мільйонів доларів на рік. Тому для збереження працездатності ЛКМ і зниження збитків від їх простоїв підприємства вимушені йти на значні витрати для підвищення надійності й відновлення їх роботи після відмови.

Міжнародна організація стандартів ISO (International Standards Organization) стосовно мереж, насамперед глобальних, розробила *семирівневу ієрархічну модель*. З точки зору діагностування мереж найбільшу зацікавленість викликають три рівні: фізичний, інформаційний (канальний) і мережний.

1. Фізичний рівень. Охоплює засоби створення, підтримання і порушення фізичних з'єднань для передавання двійкової інформації. Він є найнижчим рівнем.

2. Інформаційний (канальний) рівень. Забезпечує встановлення, виконання і завершення обміну кодами між комп'ютерами мережі в інформаційному каналі. Одна з головних функцій цього рівня — виявлення і усунення несправностей (діагностування), що виникли на фізичному рівні.

3. Мережний рівень. При взаємодії з інформаційним рівнем формується повідомлення про виявлені несправності.

Це свідчить, що контроль технічного стану та інформаційних потоків мережі на базі семирівневої моделі ISO забезпечують передусім другий і третій рівні.

Асоціація західноєвропейських виробників комп'ютерної техніки за стандартну обрала архітектуру ЛКМ, що охоплює чотири нижніх рівні, які збігаються з американським стандартом IEEE 802 (Institute of Electrical and Electronic Engineers — Інститут інженерів з електротехніки і електроніки, що встановлює стандарти).

Майже всі відомі методи контролю в ЛКМ ґрунтуються на апаратній, програмній або часовій надлишковості. Використання апаратної надлишковості в основному скероване на усунення відмов. Програмну і часову надлишковість використовують для запобігання випадкових збоїв. Програмна надлишковість передбачає використання тестів, контрольних кодів та ін. Часова надлишковість реалізується багатократним виконанням програм для найважливіших функцій мережі.

На надійність функціювання мережі істотний вплив мають комунікаційні лінії. Для підвищення її застосовують надлишковість ліній та засобів автоматичного налагодження обладнання мережі, що відмовило, використовуючи різноманітні методи контролю або діагностування.

Один з найпоширеніших методів діагностування з метою пошуку несправних компонентів ЛКМ — *самотестування за допомогою останніх функційних компонентів*. Він ґрунтується на процедурі, що здійснює обчислення у спільному каналі. Контролюючий абонент надсилає мережею циркулярний контрольний кадр. Кожний з під'єднаних до мережі абонентів, у тому числі й контрольований, додає за способом спільного каналу вміст контрольного кадру зі своїм номером і направляє суму далі. Пройшовши мережею і повернувшись до контролюючого абонента, кадр записує суму номерів усіх абонентів, що відізвались. Якщо вона збігається з контрольною сумою всіх абонентів, у ЛКМ робиться висновок, що всі вони, в тому числі й контрольований, а також комунікаційні лінії, справні. Коли кадр не повертається — це свідчить про несправність у комунікаційних лініях. Якщо він повертається без зміни — значить на нього не прореагували інші абоненти. Отже, в мережі є один або кілька несправних абонентів. Коли контрольна сума не збігається із заданою контрольною сумою всіх абонентів — це також свідчить про те, що в ЛКМ є несправні абоненти. Якщо помилка одна, то номер несправного абонента визначають за різницею вищезгаданих контрольних сум.

За відмови в ЛКМ одного з абонентів міжмашинного каналу або його підсистеми архітектура системи передбачає побудову нової конфігурації з обходом модуля, що відмовив. ЛКМ стає неповнозв'язаною. У такій ситуації в ній є не менше двох незалежних шляхів передавання інформації між двома будь-якими станціями. Для реконфігурації мережі в кожному комутаційному модулі необхідно мати інформацію про стан каналів і сусідніх модулів.

У такій ЛКМ використовують протоколи, що створюють дві групи.

До *першої групи* належать протоколи двох нижніх рівнів семирівневої ієрархічної моделі ISO, які в системі обміну реалізуються апаратно-мікропрограмним способом, а в абонентних станціях — за допомогою відповідних підпрограм. Вони забезпечують спрягання абонентних станцій із систе-

мою обміну, створення і роз'єднання каналів міжабонентських зв'язків, а також контроль даних, що передаються.

Протоколи *другої групи* забезпечують зв'язок між прикладними процесами. Вони реалізуються монітором, що пов'язує робочі станції, і транспортним інтерфейсом.

Ідентифікація несправностей ЛКМ

Керування комп'ютерними мережами, їх ефективна експлуатація є наукою і своєрідним мистецтвом. Вони потребують глибоких знань функціювання мереж для вирішення завдань діагностування. Щоб знайти несправність при діагностуванні ЛКМ, спеціалісти витрачають 3—4 роки. Вони виявляють симптоми, складають перелік можливих причин, виконують тести для аналізу причини і аналізують результати.

Виникнення несправностей у мережі супроводжується зменшенням її продуктивності, тому що необхідно повторювати повідомлення при невдалій спробі передавання. Це підвищує надійність роботи ЛКМ, але й ускладнює процес ідентифікації несправностей через ідентичність симптомів.

Процес ідентифікації несправностей поділяють на три етапи.

Етап 1. Для того щоб ідентифікувати несправність, що виникла в ЛКМ, ще до її появи з'ясовують типові характеристики комп'ютерної мережі:

— середнє завантаження ЛКМ, його зміну протягом робочого дня;

— найпопулярнішу прикладну програму;

— протоколи, що використовують при роботі, та їх показники продуктивності;

— виробників апаратних засобів ЛКМ і їхні характеристики продуктивності;

— виробників підсилювачів, мостів, маршрутизаторів, міжмережних шлюзів, характеристики цього обладнання.

Етап 2. Аналізуючи причини виникнення несправностей, звертають увагу на такі чинники:

— дія несправності, що виникла, на користувача (у виділеному сегменті чи на одиничних робочих станціях);

— тривалість дії несправності в процесі роботи;

— характер прояву несправності (детермінований (стабільний) або випадковий);

— останні зміни, зроблені в архітектурі мережі, доповнення ЛКМ новими пристроями;

— при виході з ладу яких пристроїв можливі такі симптоми, хто виробник цих приладів, які версії комп'ютерних систем, мережних адаптерів, концентраторів, маршрутизаторів, мостів, прикладних програм і мережної операційної системи.

Зібравши цю інформацію, складають перелік можливих причин симптомів, які спостерігають. На основі відомих характеристик нормального режиму роботи ЛКМ проводять їх зіставлення із симптомами несправностей і визначають їх причини. Щоб скласти перелік можливих несправностей, необхідно знати, як впливає вихід з ладу окремого компонента на роботу комп'ютерної мережі.

Етап 3. Проводять тестування ЛКМ з метою ідентифікації конкретної несправності із загального списку. Після виявлення місця прояву несправності замінюють несправний компонент чи реконфігурують мережу і проводять повторне тестування.

Засоби діагностування і усунення несправностей ЛКМ. Більшість несправностей в ЛКМ виникають на фізичному рівні моделі OSI (Open System Interconnection — зв'язок відкритих систем), що є складовою організації ISO. Це — обриви, короткі замикання в кабелях (саме в цій мережі найчастіше виникають несправності), неправильне функціювання адаптерів та ін. Численні засоби діагностування ЛКМ і усунення несправностей поділяють на стратегічні і тактичні.

Стратегічні засоби. З їх допомогою проводять моніторинг і контроль параметрів роботи всієї мережі, вирішують проблеми, пов'язані із загальною конфігурацією ЛКМ, та ін. До них належать системи керування мережею, вбудовані системи діагностування, розподілені системи моніторингу, засоби діагностування операційних систем хостмашин і серверів.

Тактичні засоби. Дають змогу відшукати і усунути несправності в конкретних мережних сегментах. Це аналізатори протоколів і сканери кабельної системи.

Застосування в комплексі стратегічних і тактичних засобів діагностування ЛКМ дає змогу досягти необхідної глибини пошуку несправностей і таким чином підвищити ефективність процесу діагностування.

Діагностування МПП і С не слід розглядати як абстрактний процес. Його необхідно пов'язувати з етапами життєвого циклу цих пристроїв. Це зумовлено тим, що технологічний процес діагностування МПП і С на кожному етапі має свої особливості.

3.4. Методологія поетапного діагностування цифрових і мікропроцесорних пристроїв

Кожен пристрій проходить процеси проектування, виробництва і експлуатації. Виробничий процес можна поділити на сукупність часткових процесів — основних, допоміжних і обслуговуючих. Слід розглянути основні процеси, тобто технологічні процеси виробництва, а також процеси проектування і експлуатації. Зокрема, звернути увагу на поетапний процес діагностування Ц і МПП.

Технологічний процес діагностування

Технологічний процес діагностування Ц і МПП складається з послідовних технологічних діагностичних операцій. Може бути представлений лінійно, циклічно або комбіновано.

Лінійний технологічний процес діагностування — це послідовний ланцюжок операцій.

Циклічний технологічний процес діагностування — замкнута послідовність операцій ланцюжка.

Комбінований технологічний процес діагностування — послідовний ланцюжок технологічних операцій з паралельними розгалуженнями і (або) замкнутими ланцюжками операцій.

Основні принципи технологічного процесу діагностування Ц і МПП: можливість реалізації; розумна достатність і раціональність (економічна ефективність); поетапна наступність та ієрархічність реалізації.

Структурною одиницею технологічного процесу діагностування Ц і МПП є *технологічна діагностична операція* — частина технологічного процесу діагностування, яку здійснюють над одним чи кількома ОД, на одному робочому місці, одним чи кількома спеціалістами-експертами або спеціалістами-розробниками.

Технологічні процеси діагностування Ц і МПП базуються на відповідних методах діагностування. Найпоширеніші з них: функційний, тестовий і змішаний.

Методи тестового діагностування реалізують в основному за допомогою зовнішніх щодо МПП засобів. Внутрішні засоби діагностування використовують для контролю діагностування функцій, які має виконувати МПП.

Вибір методу діагностування залежить від обраного рівня моделі мікропроцесорного пристрою як ОД. Наприклад, для цифрових пристроїв системи модернізації АТС електрозв'язку найоптимальніші моделі на вентильному, автоматному, покомпонентно-структурному і функційному рівнях. Компоненти малого та середнього ступенів інтеграції моделюють на вентильному й автоматному рівнях, компоненти підвищеного ступеня інтеграції (ВІС, НВІС) — на покомпонентно-структурному, а цифрові пристрої (змонтовані друковані плати) — на функційному. Однак на етапі проектування ВІС і НВІС можна моделювати на вентильному або автоматному рівнях. МПП (змонтовані друковані плати з НВІС мікропроцесора (мікропроцесорів) чи програмованими НВІС) краще моделювати на алгоритмічному або інших рівнях, що містять елементи програмування.

Основними принципами моделювання Ц і МПП як об'єктів діагностування є: багаторівневе моделювання МПП; синтез математичної моделі МПП або цифрових пристроїв за їх фрагментами, які можна подавати на різних ієрархічних рівнях; опис компонентів підвищеного ступеня інтеграції (ВІС і НВІС) покомпонентно-структурними моделями на етапах виробництва і експлуатації.

На початку XXI ст. набирають актуальності принципи, пов'язані з інтелектуалізацією процесу діагностування МПП, зокрема:

— моделювання і розв'язання важкоформалізованих завдань діагностування шляхом використання компонентів ШІ;

— застосування знань експертів при розробленні моделей МПП;

— моделювання ОД на основі теорії штучних нейронних мереж;

— використання теорії нечітких множин для розроблення математичних моделей ОД і розмитих процесів та ін.

Головні цілі моделювання:

— розроблення детального формалізованого опису технологічного процесу діагностування Ц і МПП;

— задання послідовності й взаємозв'язку технологічних діагностичних операцій процесу діагностування;

— одержання необхідних значень діагностичних параметрів ОД;

— вибір критеріїв оптимізації моделей діагностування з метою подальшої оцінки ефективності технологічного процесу.

Щоб змоделювати процес діагностування Ц і МПП, повинен бути відомий перелік діагностичних операцій. Це основа для виявлення причинно-наслідкових зв'язків, властивих цьому процесу. Задавши послідовності діагностичних операцій та знаючи їх взаємозв'язок, врахувавши відомі параметри діагностування, розробляють математичну модель процесу діагностування, до якої включають показники, що необхідно визначити. Їх знаходять як невідомі з рівнянь чи нерівностей. Визначивши, як вони залежать від показників, що вже відомі й задані, проводять оцінку достовірності технологічного процесу діагностування Ц і МПП на різних етапах життєвого циклу.

Діагностичні операції, що виконують на певному етапі життєвого циклу, описують множиною:

$$E = \{E_1,\ldots,E_i,\ldots,E_n\}, \qquad (3.19)$$

де E_1, E_i, E_n — перша, i-та, n-на діагностичні операції.

Кожна діагностична операція має свої параметри, що складають множину $\overline{P}$ з підмножин $\overline{P}_1,\ldots,\overline{P}_i,\ldots,\overline{P}_n$:

$$\overline{P} = \{\overline{P}_1,\ldots,\overline{P}_i,\ldots,\overline{P}_n\}, \qquad (3.20)$$

де $\overline{P}_1 = \{p_1,\ldots p_k\}$, $\overline{P}_i = \{p_{i1},\ldots p_{im}\}$, $\overline{P}_n = \{p_{n1},\ldots p_{nz}\}$.

Якщо прийняти E_1 за початкову, а E_n — за кінцеву діагностичні операції, тоді всі операції $[E_2 \ldots E_{n-1}]$ будуть проміжними. Кожну з них характеризує відповідна множина значень вихідної функції $\overline{E}_2,\ldots,\overline{E}_{i-1}, \overline{E}_i \ldots \overline{E}_{n-1}, \overline{Y}$.

На початкову діагностичну операцію E_1 діє множина вхідних параметрів $\overline{X} = \{x_1,\ldots,x_s\}$. Вхідними параметрами для кожної наступної діагностичної операції є множина значень вихідної функції попередньої діагностичної операції. Тоді:

$$\begin{aligned}
&\overline{F}_1 = \overline{X} \cup \overline{P}_1; \quad && \overline{X} \subseteq \overline{F}_1, \overline{P}_1 \subseteq \overline{F}_1;\\
&\qquad\vdots\\
&\overline{F}_i = \overline{F}_{i-1} \cup \overline{P}_i; && \overline{F}_{i-1} \subseteq \overline{F}_i, \overline{P}_i \subseteq \overline{F}_i;\\
&\qquad\vdots\\
&\overline{Y} = \overline{F}_{n-1} \cup \overline{P}_n; && \overline{F}_{n-1} \subseteq \overline{Y}, \overline{P}_n \subseteq \overline{Y}.
\end{aligned} \qquad (3.21)$$

Виключивши вектори множин проміжних вихідних функцій з рівнянь (3.21), одержують:

$$\begin{aligned}
&\overline{F}_1 = \overline{X} \cup \overline{P}_1;\\
&\overline{F}_2 = \overline{X} \cup \overline{P}_1 \cup \overline{P}_2;\\
&\qquad\vdots\\
&\overline{F}_i = \overline{X} \cup \overline{P}_1 \cup \overline{P}_2 \cup \ldots \cup \overline{P}_i;\\
&\qquad\vdots\\
&\overline{Y} = \overline{X} \cup \overline{P}_1 \cup \overline{P}_2 \cup \ldots \cup \overline{P}_i \cup \ldots \cup \overline{P}_n.
\end{aligned} \qquad (3.22)$$

Формалізована схема технологічного процесу діагностування пристроїв модернізації АТС з прямонапрямленими зв'язками зображена на рис. 3.12.

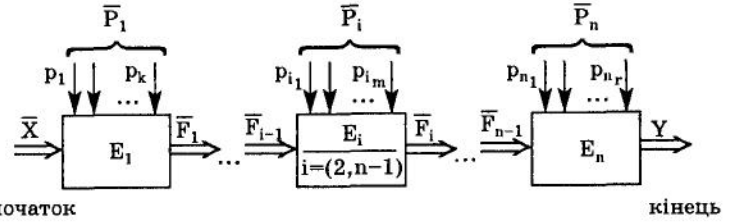


Рис. 3.12. Формалізована схема технологічного процесу діагностування Ц і МПП з прямими послідовними зв'язками

Запропонована модель процесу діагностування дає змогу враховувати вплив параметрів початкової, проміжних і кінцевої діагностичних операцій.

Особливості моделювання на різних етапах життєвого циклу пристрою

Моделювання процесу діагностування Ц МПП здійснюють на етапах проектування, виробництва та експлуатації пристрою.

Моделювання на етапі проектування. Особливості моделювання процесу діагностування Ц і МПП на етапі проектування пов'язані з особливостями діагностичних операцій. Щодо пристроїв, змонтованих на друкованих платах (картриджах), такими діагностичними операціями є:

— вибір тестопридатних мікропроцесорних комплектів (наборів мікросхем системної логіки) інтегральних схем (ІС);

— розроблення фрагментів тестових структур пристрою на базі спеціальних ІС;

— розроблення методик компонування елементів і компонентів пристроїв, враховуючи вимоги контролю і діагностування (необхідна кількість контрольних точок, транспортування несправностей на вихід структури пристрою або на контрольні точки для спостереження їх прояву);

— вибір додаткових контрольних точок для діагностування пристрою;

— декомпозиція структури (цифрової, аналого-цифрової) пристрою з метою реалізації обраного методу діагностування;

— аналіз і вибір міжкомпонентних та міжфрагментних зв'язків (ліній, шин, магістралей) для діагностування пристрою.

Діагностичні операції класифікують на три групи.

1. Діагностичні операції при розробленні узагальненої моделі процесу діагностування. Цю групу описує множина загальних діагностичних операцій E, які спільні для всіх етапів життєвого циклу пристрою або системи.

2. Діагностичні операції, пов'язані з особливостями моделювання процесу діагностування на етапі проектування. Вони становлять множину $E_{осп}$.

3. Усі діагностичні операції, задіяні в процесі діагностування пристроїв на етапі проектування, становлять множину $E_{пр}$. Моделювання процесу діагностування полягає у вибірковому поєднанні загальних діагностичних операцій множини $E_{виб} \subset E$ з діагностичними операціями множини $E_{осп}$. У результаті утворюється ланцюжок діагностичних операцій, який описує множина $E_{пр}$, що є об'єднанням діагностичних операцій процесу діагностування пристроїв на етапі проектування.

Цю класифікацію можна описати такими математичними виразами:

$$\begin{aligned} E &= \{E_1,\ldots,\ E_g,\ldots E_h,\ldots E_i,\ldots E_n\};\\ E_{виб} &= \{\ldots,\ E_g,\ldots E_h,\ldots\};\\ E_{виб} &\subset E; \qquad (3.23)\\ E_{осп} &= \{E_{oc_1}, E_{oc_2},\ldots, E_{oc_k},\ldots, E_{oc_q}\};\\ E_{пр} &= E_{виб} \cup E_{осп}. \end{aligned}$$

Схема технологічного процесу діагностування Ц і МПП на етапі проектування представлена на рис. 3.13. Множину значень параметрів вихідної функції $\overline{Y}_{пр}$ кінцевої діагностичної операції $E_{пр_r}$ визначають з рівняння:

$$\overline{Y}_{пр} = \overline{P}_{осп_1} \cup \overline{P}_g \cup \overline{P}_{осп\,2} \cup \overline{P}_h \cup \ldots \cup \overline{P}_{осп_k} \cup \ldots \cup \overline{P}_{осп_q}. \quad (3.24)$$

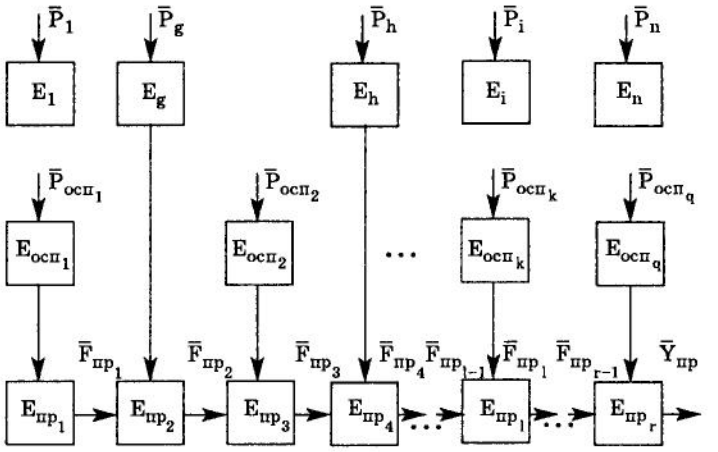


Рис. 3.13. Формалізована схема технологічного процесу діагностування Ц і МПП на етапі проектування

Етап проектування є вихідним процесу моделювання Ц і МПП як об'єктів діагностування.

Моделювання на етапі виробництва. Діагностичними операціями процесу діагностування Ц і МПП на етапі виробництва, що пов'язані з особливостями моделювання, є:

— вхідний контроль компонентів пристрою з метою виявлення несправностей компонентів (невиконання заданих згідно з технічними умовами функцій, незадовільні робочі характеристики та ін.);

— візуальний контроль пристроїв з метою виявлення механічних пошкоджень плати або компонентів, неправильної орієнтації мікросхем на платі відносно шин живлення, незмонтованих компонентів, компонентів з іншими реалізуючими функціями та ін.;

— програмний контроль за принципом «придатний-непридатний» для встановлення правильності функціювання ОД і відбракування несправних пристроїв;

— виявлення детермінованих несправностей;

— ідентифікація статичних несправностей (коротке замикання провідників, обриви, константні («≡0» чи «≡1») і логічні несправності та ін.);

— ідентифікація динамічних несправностей (логічні і нелогічні несправності), зокрема, тих, причини яких «змагання» сигналів, несправностей, що проявляються в результаті дії завад, які виникають у лініях під час перемикання груп елементів компонента чи компонентів пристрою та ін.);

— ідентифікація несправностей, пов'язаних із шинною структурою пристрою;

— виявлення випадкових несправностей;

— повторний програмний контроль за принципом «придатний-непридатний» після усунення виявлених несправностей.

Ці операції теж поділяють на три групи.

1. Діагностичні операції, перераховані при розробленні узагальненої моделі процесу діагностування. Вони складають елементи множини E. Їхня основна властивість та, що частина їх або всі вони можуть входити до складу діагностичних операцій будь-якого життєвого циклу пристрою.

2. Діагностичні операції, властиві процесу діагностування тільки на етапі виробництва. Вони — елементи множини $E_{осв}$.

3. Усі діагностичні операції задіяні в процесі діагностування пристроїв на етапі виробництва. Це елементи множини $E_{вир}$.

Моделювання процесу діагностування пристроїв на етапі виробництва — вибіркове поєднання загальних діагностичних операцій множини E з діагностичними операціями множини $E_{осв}$. Утворюється ланцюжок діагностичних операцій, що є елементами множини $E_{вир}$. Він і складає послідовність діагностичних операцій процесу діагностування Ц і МПП на етапі виробництва.

У цьому випадку:

$$\begin{aligned} E &= \{E_1,\ldots,E_g,\ldots,E_h,\ldots,E_i,\ldots,E_n\};\\ E_{вир} &= \{\ldots,E_g,\ldots,E_h,\ldots\};\\ E_{вир} &\subset E;\\ E_{осв} &= \{E_{осв_1},E_{осв_2},\ldots,E_{осв_k},\ldots,E_{осв_q}\};\\ E_{вир} &= E_{вир}\cup E_{осв}. \end{aligned} \qquad (3.25)$$

Схема технологічного процесу діагностування пристроїв на етапі виробництва представлена на рис. 3.14. Множину значень параметрів вихідної функції $Y_{пр}$ кінцевої діагностичної операції $E_{пр_r}$ визначають з рівняння:

$$\overline{Y}_{вир} = \overline{P}_1\cup\overline{P}_{осв_1}\cup\overline{P}_{осв_2}\cup\overline{P}_i\cup\ldots\cup\overline{P}_{осв_k}\cup\ldots\cup\overline{P}_{осв_q}. \quad (3.26)$$

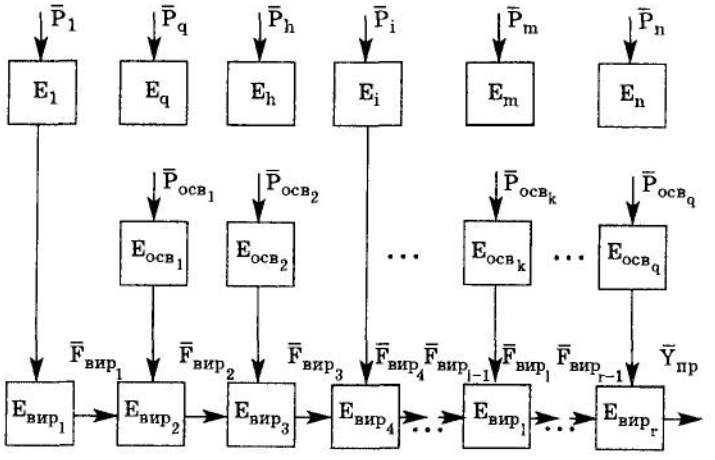


Рис. 3.14. Формалізована схема технологічного процесу діагностування Ц і МПП на етапі виробництва

Послідовність проведення діагностичних операцій — це стратегія діагностування Ц і МПП. Її вибір ґрунтується на застосованому методі діагностування і відповідних апаратних і програмних засобах.

Моделювання на етапі експлуатації. Головне завдання процесу діагностування Ц і МПП на етапі експлуатації полягає в контролі працездатності пристрою та визначенні дефектних компонентів і елементів у разі його несправності. Особливість цього процесу в тому, що контроль працездатності здійснюють на місці використання об'єкта за основним призначенням. За наявності резервний справний пристрій замінює несправний. Діагностування пристрою з глибиною пошуку несправності до компонента чи елемента проводять здебільшого в діагностичних лабораторіях. При здійсненні контролю працездатності переважають програмні засоби, а при ідентифікації несправних компонентів і елементів пристрою використовують як програмні, так і апаратні. Це слід враховувати, моделюючи процес діагностування пристроїв на етапі експлуатації.

Особливості моделювання процесу діагностування пристроїв на етапі експлуатації пов'язані з такими діагностичними операціями:

— періодичний візуальний контроль пристроїв щодо механічних та інших пошкоджень;

— програмний контроль пристрою під час вмикання в мережу живлення або проведення профілактичних робіт;

— контроль і діагностування за допомогою діагностичних програм операційних систем на місці використання пристрою за основним призначенням;

— контроль і діагностування за допомогою програм фірм-виробників пристроїв, який проводять на місці їх використання за основним призначенням;

— контроль і діагностування за допомогою програм фірм-виробників пристроїв, що здійснюють у спеціалізованих діагностичних лабораторіях;

— контроль і діагностування пристроїв за допомогою програм фірм, які спеціалізуються на їх розробленні, здійснюють у спеціалізованих діагностичних лабораторіях.

Ці діагностичні операції можуть поділятися на підоперації нижчого ієрархічного рівня залежно від типу контрольованого або діагностованого пристрою і типів несправностей, що ідентифікують у ньому.

Схема технологічного процесу діагностування пристроїв на етапі експлуатації зображена на рис. 3.15. Множину значень параметрів вихідної функції $\overline{Y}_{екс}$ кінцевої діагностичної операції $E_{екс_s}$ визначають з рівняння:

$$\overline{Y}_{екс} = \overline{P}_{осе_{11}} \cup,\ldots,\cup \overline{P}_{осе_{1n}} \cup \overline{P}_1 \cup \overline{P}_{осе_{i1}} \cup \overline{P}_{осе_{i2}} \cup \ldots$$
$$\ldots \cup \overline{P}_{осе_{ik}} \cup \ldots \cup \overline{P}_k \cup \overline{P}_{осе_{k1}} \cup \ldots \cup \overline{P}_{осе_{km}} \cup \overline{P}_{осе_{k(m+1)}} \cup \ldots \quad (3.27)$$
$$\ldots \cup \overline{P}_{осе_{q1}} \cup \ldots \cup \overline{P}_{осе_{qn}}.$$

На етапі експлуатації виявляють недоліки і помилки моделювання попередніх етапів. З метою їх усунення інформацію про них передають у підрозділи, що проектують і виробляють Ц і МПП.

Комбінована поетапна модель діагностування

Суть комбінованої поетапної моделі діагностування полягає у відображенні процесу діагностування пристроїв на різних етапах життєвого циклу. Процес діагностування неперервний від початку проектування до закінчення експлуатації пристрою. Тобто організацію і проведення діагностичних операцій на різних етапах життєвого циклу пристроїв здійснюють кілька підприємств, що працюють на одну мету. Саме у цьому — основна відмінна риса комбінованого поетапного методу діагностування Ц і МПП.

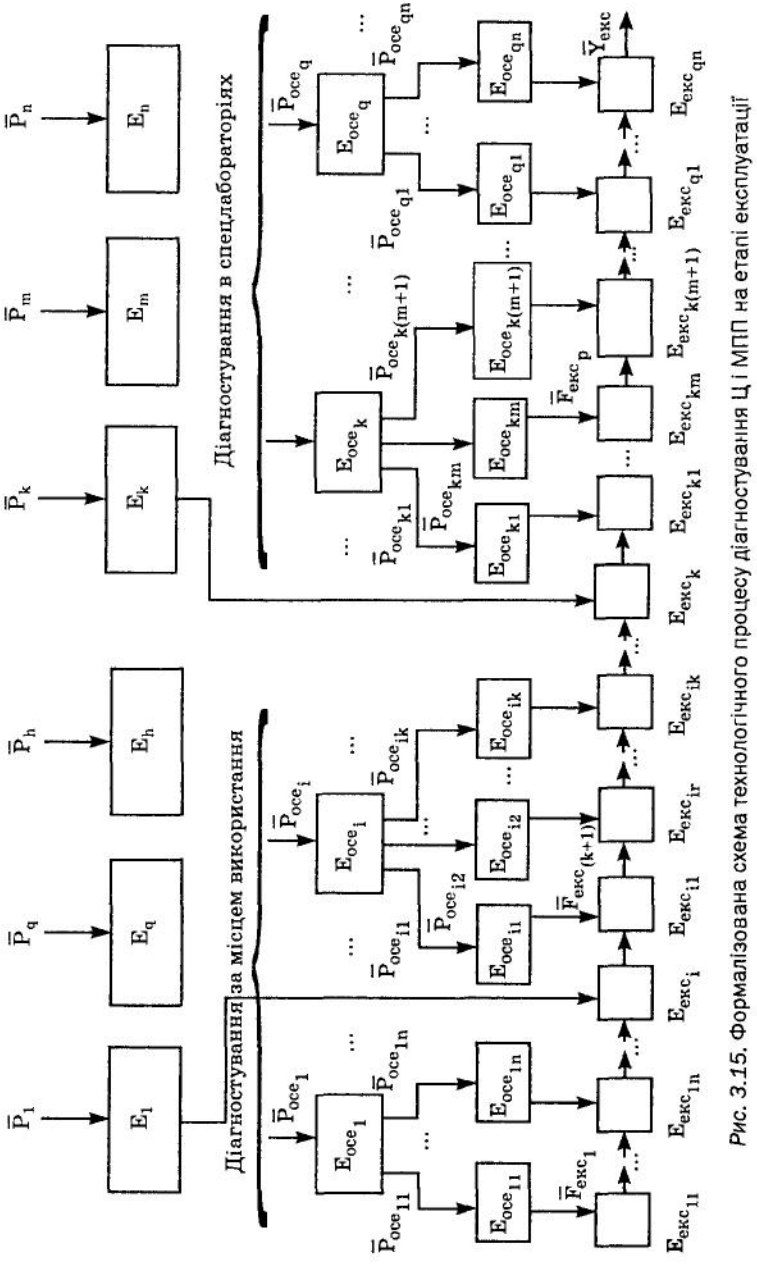


Рис. 3.15. Формалізована схема технологічного процесу діагностування Ц і МПП на етапі експлуатації

Схема комбінованого поетапного технологічного процесу діагностування пристроїв представлена на рис. 3.16. Множину значень параметрів вихідної функції $\overline{Y}$ кінцевої діагностичної операції $E_{екс_{qn}}$ визначають з рівняння:

$$\begin{aligned}
&\overline{Y} = \overline{Y}_{пр} \cup \overline{Y}_{вир} \cup \overline{Y}_{екс};\\
&\overline{Y}_{пр} = \overline{P}_{пр} \cup \overline{P}_{екс\,/пр} \cup \overline{P}_{вир\,/\,пр};\\
&\overline{Y}_{вир} = (\overline{P}_{вир} \cup \overline{P}_{екс\,/\,вир} \cup \overline{Y}_{пр}) \setminus \overline{P}_{вир\,/\,пр};\\
&\overline{Y}_{екс} = (\overline{P}_{екс} \cup \overline{Y}_{вир}) \setminus (\overline{P}_{екс\,/\,вир} \cup \overline{P}_{екс\,/\,пр}).
\end{aligned} \tag{3.28}$$

А множини параметрів вихідних функцій $\overline{Y}_{пр}, \overline{Y}_{вир}, \overline{Y}_{екс}$ через підмножини параметрів елементарних діагностичних операцій поетапного діагностування (3.28) визначають з рівнянь:

$$\begin{aligned}
&\overline{Y}_{пр} = \overline{P}_{пр_1} \cup \ldots \cup \overline{P}_{пр_\ell} \cup \ldots \cup \overline{P}_{пр_r};\\
&\overline{Y}_{вир} = \overline{P}_{вир_1} \cup \ldots \cup \overline{P}_{вир_\ell} \cup \ldots \cup \overline{P}_{вир_r};\\
&\overline{Y}_{екс} = \overline{P}_{екс_{11}} \cup \ldots \cup \overline{P}_{екс_k} \cup \ldots \cup \overline{P}_{екс_{qn}}.
\end{aligned} \tag{3.29}$$

Тоді множину значень вихідної функції $\overline{Y}$ отримують:

$$\begin{aligned}
&\overline{Y} = \overline{P}_{пр} \cup \overline{P}_{екс\,/пр} \cup \overline{P}_{вир\,/\,пр} \cup\\
&\cup ((\overline{P}_{вир} \cup \overline{P}_{екс\,/вир} \cup \overline{Y}_{пр}) \setminus \overline{P}_{вир\,/\,пр}) \cup\\
&\cup ((\overline{P}_{екс} \cup \overline{Y}_{вир}) \setminus (\overline{P}_{екс\,/вир} \cup \overline{P}_{екс\,/пр})) =\\
&= \overline{F}_{пр_1} \cup \ldots \cup \overline{F}_{пр_\ell} \cup \ldots \cup \overline{F}_{пр_r} \cup \overline{F}_{вир_1} \cup \ldots \cup \overline{F}_{вир_\ell} \cup \ldots\\
&\ldots \cup \overline{F}_{вир_r} \cup \overline{F}_{екс_{11}} \cup \ldots \cup \overline{F}_{екс_k} \cup \ldots \cup \overline{F}_{екс_{qn}}.
\end{aligned} \tag{3.30}$$

Співставивши послідовності діагностичних операцій моделі й відповідних зв'язків між ними з реальним процесом діагностування пристроїв на кожному етапі життєвого циклу, можна оцінити достовірність моделі процесу діагностування. За цими результатами вводять ланцюжок діагностичних операцій, що не були враховані при моделюванні. Після цього виявляють і встановлюють необхідні зв'язки між ними.

Достовірність моделі D_M описують адитивною (лат. additivus — додавальний) величиною. $E_{заг}$ — загальна кількість діагностичних операцій реального процесу діагностування пристрою. Ця величина відповідає цілому і дорівнює сумі величин, які відповідають його частинам.

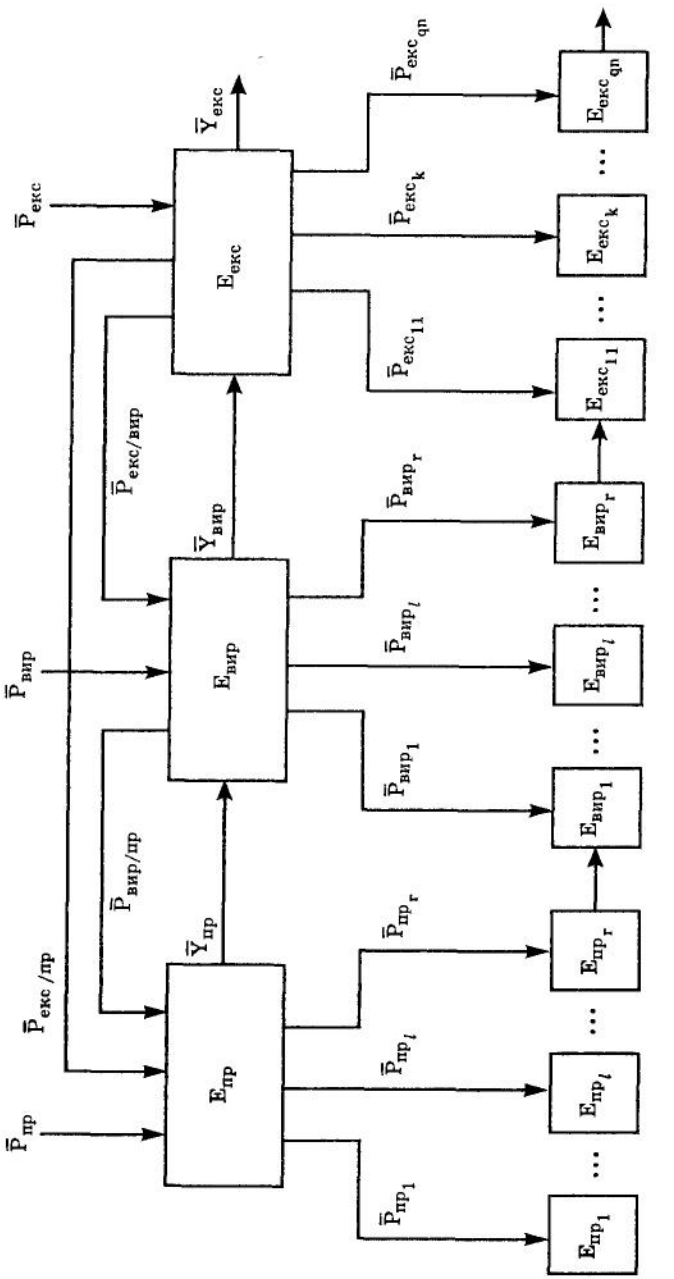


Рис. 3.16. Формалізована схема комбінованого поетапного технологічного процесу діагностування Ц і МПП

$E_{M(врах)}$ — кількість діагностичних операцій, що враховані в моделі процесу діагностування; $E_{M(неврах)}$ — кількість діагностичних операцій, що не враховані в моделі процесу діагностування. Тоді:

$$D_M = \frac{E_{M(врах)}}{E_{M(врах)} + E_{M(неврах)}} \qquad (3.31)$$

Якщо до оцінювання достовірності включити наявні й ненаявні в моделі зв'язки між діагностичними операціями, то їх описують адитивною залежністю, а залежність між діагностичними операціями і зв'язками між ними — мультиплікативною (лат. multiplicatio — множення) залежністю. У цьому випадку:

$$D_M = \frac{E_{M(врах)} \cdot F_{M(наяв)}}{E_{M(врах)} \cdot F_{M(наяв)} + E_{M(неврах)} \cdot F_{M(ненаяв)}}, \qquad (3.32)$$

де $E_{M(наяв)}$ — кількість зв'язків між $E_{M(врах)}$ у моделі; $E_{M(ненаяв)}$ — кількість зв'язків між врахованими і неврахованими діагностичними операціями ($E_{M(врах)} + E_{M(неврах)}$) у моделі.

Визначаючи ступінь адекватності моделі та об'єкта, слід брати враховані й невраховані фактори. Це будуть параметри діагностичних операцій з множин $P_1,\ldots,P_i,\ldots,P_n$ (див. рис. 3.12). Їх ділять на дві множини: множину $P_{M(врах)}$ врахованих у моделі параметрів і множину $P_{M(неврах)}$ неврахованих у моделі параметрів. Нехай залежність між параметрами всередині множин буде мультиплікативною, а залежність між параметрами різних множин — адитивною. Тоді ступінь адекватності $Q_{p/y}$ моделі, що описує процес діагностування, щодо реального процесу діагностування визначають за формулою:

$$Q_{p/y} = \frac{p_{1в} \cdot p_{2в} \cdots p_{iв}}{p_{1в} \cdot p_{2в} \cdots p_{iв} + p_{1нв} \cdot p_{2нв} \cdots p_{knв}}, \qquad (3.33)$$

де $P_{1в},\ldots,P_{2в},\ldots,P_{iв}$ — коефіцієнти врахованих параметрів діагностичних операцій; $p_{1нв},\ldots,p_{2нв},\ldots,p_{knн}$ — коефіцієнти неврахованих параметрів діагностичних операцій. Вони приймають значення у проміжку від 0 до 1 і характеризувати, наскільки важливим є значення параметрів щодо успішного проведення діагностичних операцій. Схематично поетапне діагностування Ц і МПП можна представити як певну множину технологічних діагностичних операцій і взаємозв'язків між ними.

Комбіноване поетапне діагностування на базі штучної нейронної мережі Хопфілда

Перед тим як перейти до аналізу комбінованого поетапного діагностування на базі штучної нейронної мережі Хопфілда, слід розглянути його загальну структуру (рис. 3.17), де X,Y — множини входів у технологічні операції і виходів з них; $E_{пр},E_{вир},E_{екс}$ — множини технологічних операцій діагностування на етапах проектування, виробництва й експлуатації.

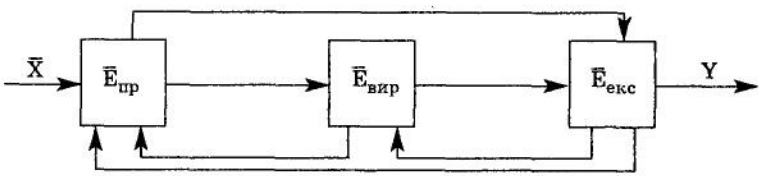


Рис. 3.17. Загальна структура комбінованого поетапного процесу діагностування Ц і МПП

Процесові діагностування Ц і МПП на етапі проектування властиві такі технологічні діагностичні операції:

— вибір тестопридатної (легкотестованої) елементної бази ($E_{пр1}$);

— визначення компонентів і фрагментів структури пристрою, що повинні пройти обов'язкове систематичне діагностування ($E_{пр2}$);

— визначення компонентів і фрагментів структури, які мають пройти епізодичне діагностування ($E_{пр3}$);

— визначення необхідної і достатньої глибини пошуку несправностей (до елемента, компонента або фрагмента) ($E_{пр4}$);

— вибір методів діагностування пристроїв, його фрагментів і компонентів ($E_{пр5}$);

— вибір засобів реалізації обраних методів діагностування пристроїв, його фрагментів і компонентів ($E_{пр6}$);

— розроблення вбудованих тестових структур пристрою для реалізації обраних методів діагностування ($E_{пр7}$);

— розроблення тестових програм для діагностування пристроїв на кожному з етапів життєвого циклу ($E_{пр8}$).

У процесі діагностування Ц і МПП на етапі виробництва виокремлюють такі технологічні діагностичні операції:

— вхідний контроль елементів і компонентів, з яких монтують пристрій ($E_{вир1}$);

— візуальний контроль змонтованих пристроїв з метою виявлення механічних пошкоджень ($E_{вир2}$);

— контроль змонтованих пристроїв за принципом «придатний-непридатний» ($E_{вир3}$);

— ідентифікація статичних несправностей пристрою та його компонентів і елементів ($E_{вир4}$);

— ідентифікація динамічних несправностей пристрою та його компонентів і елементів ($E_{вир5}$);

— ідентифікація несправностей, пов'язаних із шинною структурою пристрою ($E_{вир6}$);

— ідентифікація випадкових несправностей пристрою ($E_{вир7}$);

— повторний контроль пристроїв за принципом «придатний-непридатний» ($E_{вир8}$).

Процес діагностування Ц і МПП на етапі експлуатації поділяють на такі технологічні діагностичні операції:

— періодичний візуальний контроль пристроїв та їх компонентів і елементів з метою виявлення механічних пошкоджень ($E_{екс1}$);

— контроль і діагностування пристроїв при вмиканні живлення або подаванні сигналу скидання ($E_{екс2}$);

— епізодичний контроль і діагностування пристроїв при появі збоїв та помилок ($E_{екс3}$);

— періодичний контроль і діагностування пристроїв під час проведення профілактичних робіт ($E_{екс4}$);

— апаратний і програмний контроль та діагностування пристроїв при відмовах та ремонтах ($E_{екс5}$).

На основі структур процесу діагностування на кожному з етапів будують узагальнену структуру процесу діагностування протягом повного життєвого циклу пристрою. Вона представлена на рис. 3.18 і є базовою для розроблення методу поетапного діагностування.

Опис цієї структури ґрунтується на теорії штучних нейронних мереж, зокрема мережі видатного американського вченого Дж. Хопфілда. Процес діагностування починають з певної технологічної операції будь-якого етапу життєвого циклу пристрою. Її вважають початковою. Якщо процес діагностування починають з кількох операцій, то всі вони — початкові. Операції, якими закінчують процес, — кінцеві. Мережа Хопфілда одношарова, неповнозв'язна, що описує процес діагностування Ц і МПП, представлена на рис. 3.19.

Для її розрахунків використовують такі рівняння:

$$\overline{Y} = f(\overline{WX}), \qquad (3.34)$$

де $\overline{X}$ — вектор вхідних величин; $\overline{Y}$ — вектор вихідних величин; $\overline{W}$ — вектор вагових коефіцієнтів; f — активізацій-

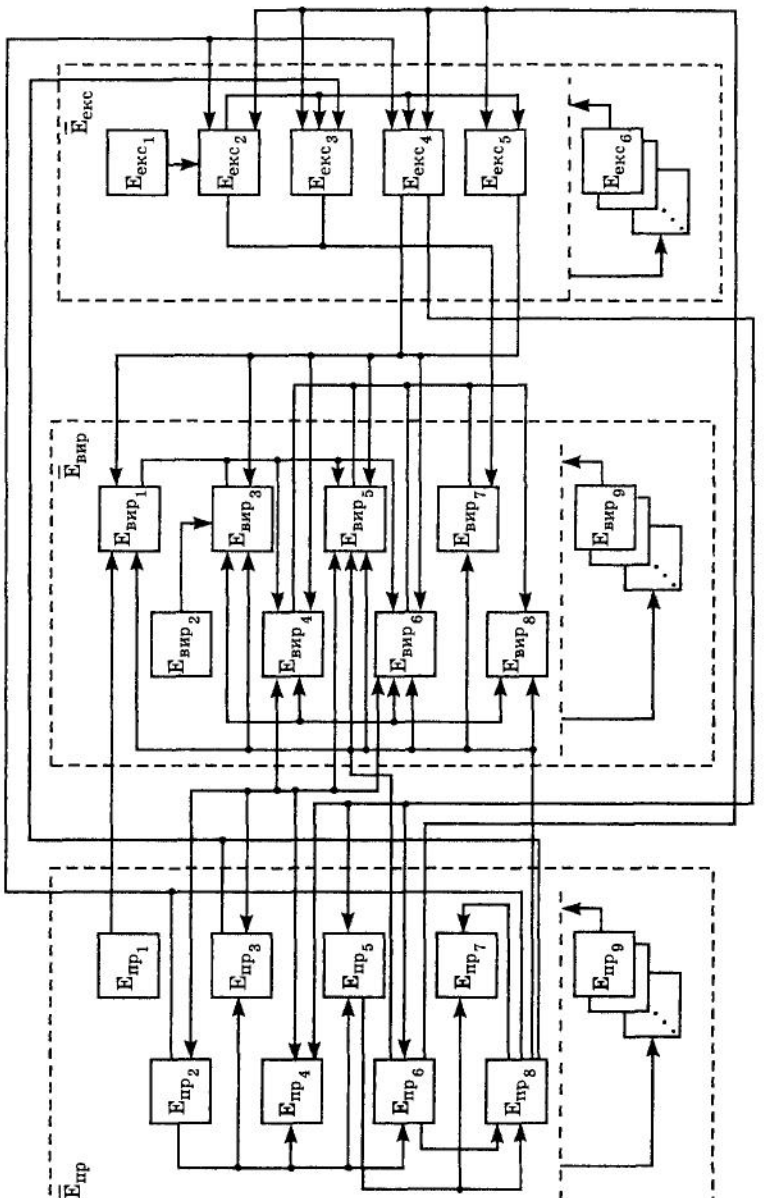


Рис. 3.18. Структура процесу комбінованого поетапного діагностування Ц і МПП протягом повного життєвого циклу пристрою

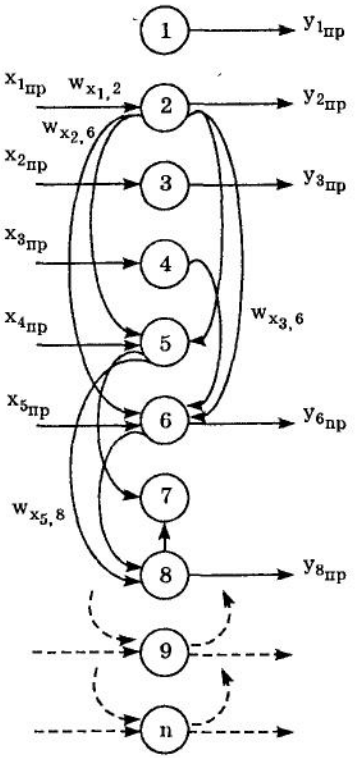


Рис. 3.19. Одношарова мережа Хопфілда, що відображає процес діагностування пристроїв на етапі проектування

на функція. Вага порогового значення (зміщення) дорівнює нулеві.

Значення j-го виходу визначають за формулою:

$$y_j = \sum w_{ij} x_i, \quad (3.35)$$

де x_i — значення i-го входу мережі; w_{ij} — вага (значення синапсу) лінії, що з'єднує x_i вхід з k-ю технологічною операцією, яка відповідає тілу k-го нейрона, або вага лінії, що з'єднує i-й нейрон з j-тим.

Процес діагностування буде найефективнішим, коли виконуватимуться всі технологічні операції діагностування і враховуватимуться всі зв'язки між ними з відповідними вагами. Вихідний вектор $\overline{Y}$ визначають шляхом ітерацій (лат. iteratio — повторення). Вектор вважають знайденим за $(n-1)$ ітерацій. Якщо на n-му кроці не змінюється вектор Y. Вагові коефіцієнти бажано обирати в інтервалі (0, 1].

На рис. 3.20 зображена мережа, що відображає процес діагностування пристроїв на етапах виробництва, а на рис. 3.21 — на етапах експлуатації.

Якщо розглядати процес діагностування на кожному етапі життєвого циклу окремо, то всі діагностичні операції повинні мати входи і виходи. Аналізуючи процес діагностування як комбінований поетапний, найголовніше з'ясувати входи на етапі проектування і виходи на етапі експлуатації.

Виходи з етапу проектування і виробництва, як і входи в етапи виробництва та експлуатації, вважають внутрішніми зв'язками. Тоді нейронна мережа, що відображає комбіноване поетапне діагностування Ц і МПП, матиме інший вигляд (рис. 3.22), ніж класична одношарова мережа Хопфілда.

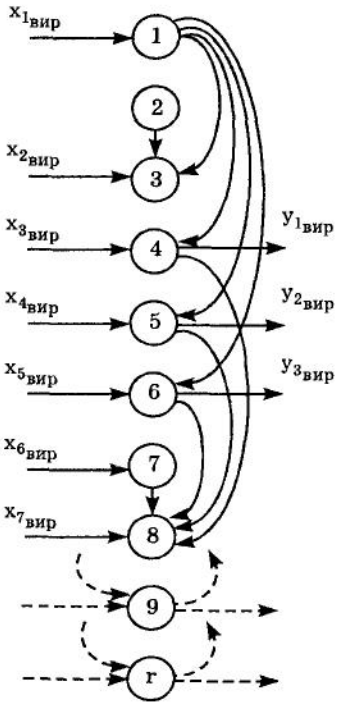


Рис. 3.20. Одношарова мережа, що відображає процес діагностування пристроїв на етапі виробництва

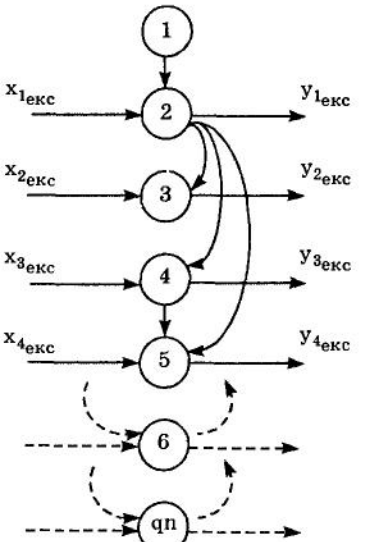


Рис. 3.21. Одношарова мережа, що відображає процес діагностування пристроїв на етапі експлуатації

Вона може будуватись як для конкретного, так і для всіх етапів життєвого циклу пристроїв. При побудові мережі для комбінованого поетапного діагностування виходи з неї, що відображають етап проектування, будуть внутрішніми для зв'язку з мережами, які відображають етапи виробництва й експлуатації, і навпаки.

Значення виходів для мережі Хопфілда, що відображає процес комбінованого поетапного діагностування пристроїв протягом усього життєвого циклу, визначають з рівнянь:

$$y_{j_{екс.}} = f \sum_{i,j}^{n} w_{ij} x_{i_{пр.}} \quad (3.36)$$

$$Y = \sum_{j=1}^{n} y_{j_{екс.}} \quad (3.37)$$

На основі розробленої штучної нейронної мережі можна побудувати множину алгоритмів діагностування. Кожен з них починатиметься з операції, що відповідає певно-

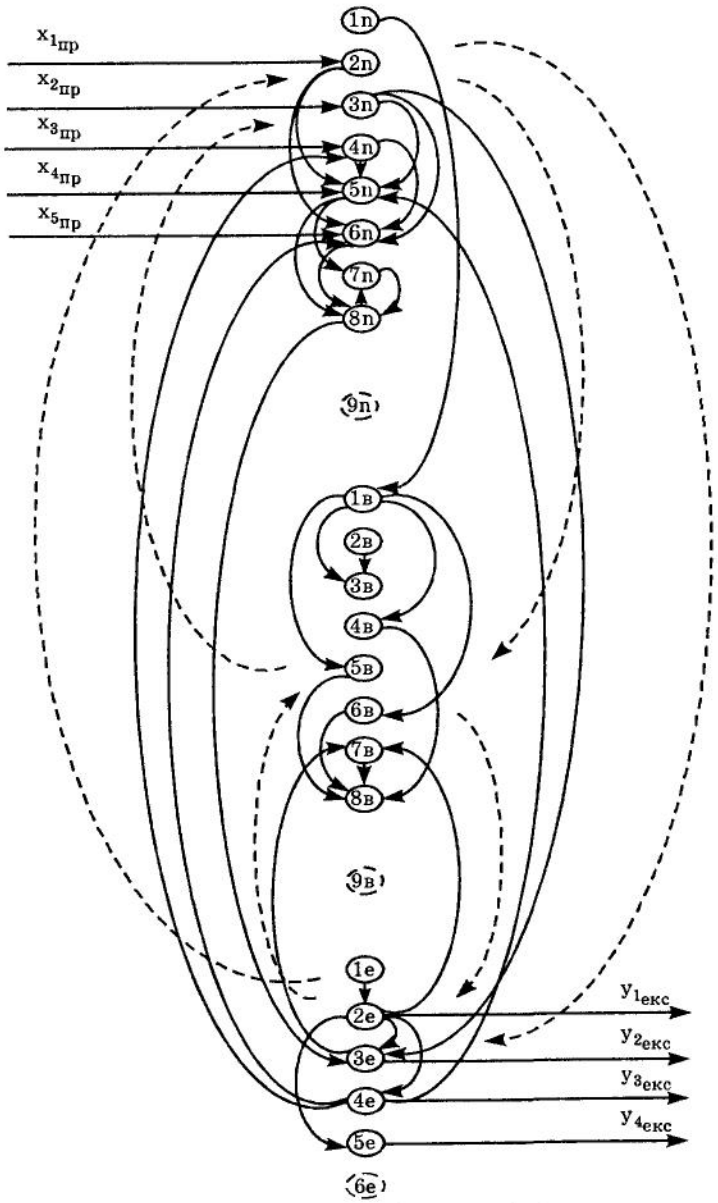


Рис. 3.22. Одношарова мережа Хопфілда, що відображає процес комбінованого поетапного діагностування пристроїв протягом усього життєвого циклу

му входу мережі x_i і закінчуватиметься операцією, що відповідає певному виходу y_i мережі. Алгоритм описує шлях у мережі. Наприклад, $x_i \rightarrow$.

Процес діагностування пристрою на етапі проектування, як складову поетапного процесу діагностування, можна реалізувати такою множиною алгоритмів:

A_i із шляхом $x_{1пр} \rightarrow E_2 \rightarrow y_{2пр}$;

$\vdots$

A_n із шляхом $x_{5пр} \rightarrow E_6 \rightarrow E_8 \rightarrow y_{8пр}$.

Шляхи алгоритмів прості, тобто безумовні. Алгоритми з умовами будують так:

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad y_{2пр.} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad\uparrow \qquad E_7 \\
A_1' : x_{1пр.} \rightarrow E_2 \rightarrow E_5 \uparrow\!\rightarrow E_8 \rightarrow y_{8пр.} \\
\vdots \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad\downarrow \\
\qquad\qquad\qquad\qquad\qquad\qquad E_6 \rightarrow y_{6пр.} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad\downarrow \qquad E_8 \rightarrow y_{8пр.}
\end{array}
$$

Аналогічно визначають множини алгоритмів процесу діагностування на етапах виробництва, експлуатації та при комбінованому поетапному діагностуванні Ц і МПП (див. рис. 3.20, 3.21, 3.22).

Процесові діагностування Ц і МПП на різних етапах життєвого циклу властиві різні технологічні діагностичні операції, а їм, у свою чергу, параметри. Певну особливість вони мають для технологічних діагностичних операцій процесу діагностування Ц і МПП на етапі проектування (див. рис. 3.22).

1. *Діагностична операція* $E_{пр1}$ $(1n)$ вибору тестопридатної елементної бази. Її характеризують параметри:

— діагностованість — визначають через коефіцієнт повноти перевірки працездатності $K_n = \dfrac{n_з}{n_в}$, де $n_з$ — кількість діагностичних параметрів, що використовують для діагностичної операції; $E_{пр1}$ — $n_в$ кількість діагностичних параметрів, що забезпечують методичну достовірність результату діагностичної операції $E_{пр1}$;

— тестованість — визначають через коефіцієнт складності K_T процедур діагностування ОД на етапі проектування. Цей параметр вказує, як легко встановлюється ОД у вихідний стан або в будь-який внутрішній стан шляхом подавання тестових дій на його зовнішні входи;

— керованість — характеризує можливість подавання тестових дій на окремі фрагменти ОД через його зовнішні входи. Її описує коефіцієнт керованості $K_к = \frac{n_{к.ф}}{n_{к.ф} + n_{нк.ф}}$, де $n_{к.ф}$ — кількість керованих фрагментів ОД, а $n_{нк.ф}$ — кількість некерованих фрагментів ОД;

— спостережуваність — вказує, на скільки однозначно ідентифікують стани ОД через його зовнішні виводи. Її характеризує коефіцієнт спостережуваності $K_c = \frac{n_{с.о}}{n_{с.о} + n_{с.но}}$, де $n_{с.о}$ — кількість станів ОД, що ідентифікують однозначно; $n_{с.но}$ — кількість станів ОД, які ідентифікують неоднозначно;

— самотестованість — ступінь охоплення вузлів ОД засобами самотестування. Цей параметр визначає коефіцієнт самотестованості $K_{ст} = \frac{n_{ст}}{n_{ст} + n_{нст}}$, де $n_{ст}$ — кількість вузлів ОД, що самотестуються, а $n_{нст}$ — кількість вузлів ОД, які не самотестуються.

2. *Діагностична операція* $E_{пр2}$ $(2n)$ визначення компонентів або фрагментів ОД, що повинні пройти оперативне діагностування. Її характеризує коефіцієнт оперативного діагностування $K_{оп.д} = \frac{n_{н.о.д}}{n_{п.о.д} + n_{н.о.д}}$, де $n_{н.о.д}$ — кількість компонентів (фрагментів) ОД, які не підлягають оперативному діагностуванню; $n_{п.о.д}$ — кількість компонентів (фрагментів) ОД, що підлягають оперативному діагностуванню.

3. *Діагностична операція* $E_{пр3}$ $(3n)$ визначення компонентів (фрагментів) ОД, що повинні пройти епізодичне діагностування. Її описує коефіцієнт епізодичного діагностування $K_{е.д} = \frac{n_{н.е.д}}{n_{п.е.д} + n_{н.е.д}}$, де $n_{н.е.д}$ — кількість компонентів (фрагментів) ОД, які не підлягають епізодичному діагностуванню; $n_{п.е.д}$ — кількість компонентів

(фрагментів) ОД, що підлягають епізодичному діагностуванню.

4. *Діагностична операція* $E_{пр4}$ $(4n)$ визначення глибини пошуку несправностей на k-му рівні ОД (тобто пошук елемента, компонента, фрагмента, з точністю до яких визначають несправність). Її визначає коефіцієнт глибини пошуку несправності $K_{г.п.д} = \frac{n_{о.р.ч.од}}{n_{з.с.ч.од}}$, де $n_{о.р.ч.од}$ — кількість однозначно розрізнювальних складових ОД на прийнятному рівні, з точністю до яких визначають місце несправності; $n_{з.с.ч.од}$ — загальна кількість складових ОД, з точністю до яких необхідно визначити місце несправності.

5. *Діагностична операція* $E_{пр5}$ $(5n)$ вибору методів діагностування ОД в цілому або його фрагментів чи компонентів. Її характеризує коефіцієнт вибору методів діагностування $K_{в.м.д} = \frac{DL}{W}$, де D — ймовірність правильного діагностування об'єкта, яка відповідає повній імовірності того, що визначають той технічний стан, у якому справді знаходиться ОД; L — глибина пошуку несправностей (чим на вищому рівні деталізації виявляють несправності, тим вище значення вона має); W — трудомісткість операції діагностування, яка прямо пропорційна часу і вартості реалізації операції діагностування.

6. *Діагностична операція* $E_{пр6}$ $(6n)$ вибору засобів реалізації обраних методів діагностування ОД. Її описує коефіцієнт вибору засобів діагностування $K_{в.з.д} = \frac{K_у \cdot K_{р.м}}{C}$, де $K_у$ — коефіцієнт уніфікації пристроїв і параметрів сигналів впливів засобів діагностування і відповідних реакцій ОД. Визначають з відношень кількості уніфікованих пристроїв до загальної кількості пристроїв та кількості уніфікованих параметрів сигналів впливів засобів діагностування і відповідних реакцій ОД до загальної кількості параметрів сигналів; $K_{р.м}$ — коефіцієнт ступеня реалізації методу діагностування, який визначають мірою адаптації засобів діагностування до обраного методу діагностування пристроїв (його значення знаходять у діапазоні від «0» до «1»); C — вартість засобів діагностування.

7. *Діагностична операція* $E_{пр7}$ $(7n)$ розроблення вбудованих тестових структур пристрою. Її характеризує коефіці-

єнт вбудованих тестових структур $K_{в.т.с} = \frac{N_{т.с} \cdot C_{т.с}}{N_{о.с} \cdot C_{о.с}}$, де $N_{т.с}$ — кількість компонентів тестової структури ОД; $N_{о.с}$ — кількість компонентів основної структури ОД; $C_{т.с}$ — вартість одного компонента тестової структури; $C_{о.с}$ — вартість одного компонента основної структури.

8. *Діагностична операція* $E_{пр8}$ $(8п)$ розроблення тестових програм діагностування пристрою для кожного з етапів його життєвого циклу. Її характеризують такі параметри:

— коефіцієнт трудомісткості розроблення тестових програм для реалізації етапу проектування $K_{W_{пр}} = \frac{W_{т.п_{пр}}}{W_{пр}}$, де $W_{т.п_{пр}}$ — трудомісткість розроблення тестових програм для реалізації етапу проектування; $W_{пр}$ — загальна трудомісткість етапу проектування;

— коефіцієнт трудомісткості розроблення тестових програм для реалізації етапу виробництва $K_{W_{вир}} = \frac{W_{Т.П_{вир}}}{W_{вир}}$, де $W_{т.п_{вир}}$ — трудомісткість розроблення тестових програм для реалізації етапу виробництва; $W_{вир}$ — загальна трудомісткість одиниці продукції на етапі виробництва;

— коефіцієнт трудомісткості розроблення тестових програм для реалізації етапу експлуатації $K_{W_{екс}} = \frac{W_{Т.П_{екс}}}{W_{екс}}$, де $W_{т.п_{екс}}$ — трудомісткість розроблення тестових програм для реалізації етапу експлуатації; $W_{екс}$ — загальна трудомісткість одиниці продукції на етапі експлуатації.

Діагностична операція $E_{пр8}$ $(8п)$ розроблення тестових програм діагностування пристрою для кожного з етапів його життєвого циклу завершує етап проектування. Крім того, існують діагностичні операції $E_{пр9...}$ $(9п)$. Їх описують відповідні параметри.

Наступним етапом у життєвому циклі Ц і МПП є етап виробництва. Йому властиві інші технологічні операції процесу діагностування цифрових і мікропроцесорних пристроїв (див. рис. 3.22).

1. *Діагностична операція* $E_{вир1}$ $(1в)$ вхідного контролю елементів і компонентів, з яких монтують пристрій. Її характеризують параметри:

— повнота контролю $K_{п.к} = \frac{n_{о.к}}{n_{ком.п}}$, де $n_{о.к}$ — кількість компонентів пристрою, охоплених контролем; $n_{ком.п}$ — загальна кількість компонентів пристрою;

— повнота контролю щодо відмов $K_{п.к.в} = \frac{n_{в.в}}{n_в}$, де $n_{в.в}$ — кількість виявлених відмов даним методом; $n_в$ — загальна кількість відмов ОД;

— повнота контролю щодо контрольованих функцій $K_{п.к.ф} = \frac{n_{к.ф}}{n_ф}$, де $n_{к.ф}$ — кількість контрольованих функцій, що виконує пристрій; $n_ф$ — загальна кількість функцій, які виконує пристрій.

2. *Діагностична операція* $E_{вир2}$ $(2в)$ візуального контролю з метою виявлення пошкоджень. Її визначає коефіцієнт повноти візуального контролю $K_{п.в.к_{вир}} = \frac{n_{в.п_{вир}}}{n_{п_{вир}}}$, де $n_{в.п_{вир}}$ — кількість візуально виявлених пошкоджень пристрою; $n_{п_{вир}}$ — загальна кількість пошкоджень.

3. *Діагностична операція* $E_{вир3}$ $(3в)$ контролю змонтованих пристроїв за принципом «придатний-непридатний». Її описує коефіцієнт повноти контролю за принципом «придатний-непридатний» $K_{п.к.п} = \frac{n_{к.п\text{-}нп}}{n_{п.з}}$, де $n_{к.п\text{-}нп}$ — кількість змонтованих і контрольованих за принципом «придатний-непридатний» пристроїв; $n_{п.з}$ — загальна кількість змонтованих пристроїв.

4. *Діагностична операція* $E_{вир4}$ $(4в)$ ідентифікації статичних несправностей пристрою та його компонентів. Її характеризує коефіцієнт повноти ідентифікації статичних несправностей $K_{п.і.с.н} = \frac{n_{і.с.н}}{n_{с.н}}$, де $n_{і.с.н}$ — кількість ідентифікованих статичних несправностей ОД; $n_{с.н}$ — загальна кількість статичних несправностей ОД.

5. *Діагностична операція* $E_{вир5}$ $(5в)$ ідентифікації динамічних несправностей пристрою та його компонентів. Її визначає коефіцієнт повноти ідентифікації динамічних

несправностей $K_{п.і.д.н} = \frac{n_{і.д.н}}{n_{д.н}}$, де $n_{і.д.н}$ — кількість ідентифікованих динамічних несправностей ОД; $n_{д.н}$ — загальна кількість динамічних несправностей ОД.

6. *Діагностична операція* $E_{вир6}$ (6*в*) ідентифікації несправностей, пов'язаних із шинною структурою пристрою. Її характеризує коефіцієнт повноти ідентифікації несправностей, пов'язаних із шинною структурою, $K_{п.і.н.ш.с} = \frac{n_{і.н.ш.с}}{n_{н.ш.с}}$, де $n_{і.н.ш.с}$ — кількість ідентифікованих несправностей шинної структури; $n_{н.ш.с}$ — загальна кількість несправностей, пов'язаних із шинною структурою пристрою ОД.

7. *Діагностична операція* $E_{вир7}$ (7*в*) ідентифікації випадкових несправностей пристрою. Її характеризує ймовірність $P_{в.в.н}$ виявлення випадкових несправностей до загальної кількості випадкових несправностей, що проявилися за час діагностування пристрою на етапі виробництва.

8. *Діагностична операція* $E_{вир8}$ (8*в*) повторного контролю пристроїв за принципом «придатний-непридатний». Її описує коефіцієнт повноти контролю за принципом «придатний-непридатний», як і діагностичну операцію $E_{вир3}$.

Діагностична операція $E_{вир8}$ (8*в*) повторного контролю пристроїв за принципом «придатний-непридатний» завершує етап виробництва Ц і МПП. Варто зауважити, що існують діагностичні операції $E_{вир9...}$ (9*в*). Їх характеризують відповідні параметри.

Після цього пристрій проходить етап експлуатації, якому властиві свої технологічні операції процесу діагностування Ц і МПП (див. рис. 3.22).

1. *Діагностична операція* $E_{екс1}$ (1*е*) періодичного візуального контролю пристроїв та їх компонентів з метою виявлення механічних пошкоджень. Її характеризують такі параметри:

— коефіцієнт періодичності візуального контролю $K_{пер.в.к_{екс}} = \frac{\overline{\Delta t}_{п.в.к}}{t_{екс}}$, *де* $\overline{\Delta t}_{п.в.к}$ — середній час між попереднім і наступним візуальним контролем з метою виявлення механічних пошкоджень, $t_{екс}$ — час між початком і закінченням експлуатації пристрою (протягом доби);

— коефіцієнт повноти візуального контролю $K_{п.в.к_{екс}} = \frac{n_{в.п_{екс}}}{n_{п_{екс}}}$, де $n_{в.п_{екс}}$ — кількість візуально виявлених пошкоджень пристрою під час i-го візуального контролю на етапі експлуатації; $n_{п_{екс}}$ — загальна кількість пошкоджень пристрою на момент i-го візуального контролю.

2. *Діагностична операція* $E_{екс2}$ (2*е*) контролю і діагностування пристроїв при вмиканні живлення або подаванні сигналу скиду. Її визначають параметри:

— коефіцієнт періодичності вмикання $K_{пер.д.в.ж} = \frac{\overline{\Delta t}_{п.в}}{t_{екс}}$, де $\overline{\Delta t}_{п.в}$ — середній час між попереднім і наступним вмиканням пристрою; $t_{екс}$ — загальний час експлуатації пристрою;

— коефіцієнт періодичності скидання пристрою $K_{пер.д.с} = \frac{\Delta t_{п.с}}{t_{в.в}}$, де $\overline{\Delta t}_{п.с}$ — середній час між попереднім і наступним скиданням; $t_{в.в}$ — час між вмиканням і вимиканням живлення при експлуатації пристрою.

3. *Діагностична операція* $E_{екс3}$ (3*е*) оперативного контролю і діагностування в процесі експлуатації пристрою. Її характеризують:

— коефіцієнт прояву збоїв і помилок $K_{о.к_{вир}} = \frac{K_{п.зі_{екс}}}{K_{зі_{екс}}}$, де $K_{п.зі_{екс}}$ — кількість збоїв і помилок, що проявилася при експлуатації пристрою і була ідентифікована під час оперативного контролю і діагностування; $K_{зі_{екс}}$ — загальна кількість збоїв і помилок, що проявилася під час експлуатації;

— коефіцієнт повноти охоплення оперативним контролем і діагностуванням компонентів пристрою $K_{о.о.кід} = \frac{n_{о.о.кід}}{n_п}$, де $n_{о.о.кід}$ — кількість компонентів пристрою, охоплених оперативним контролем і діагностуванням; $n_п$ — загальна кількість компонентів пристрою.

4. *Діагностична операція* $E_{екс4}$ (4*е*) періодичного контролю і діагностування пристроїв під час проведення профілактичних робіт. Її характеризують тривалість проведення профілактичних робіт $t_{п.р}$ та коефіцієнт періодич-

ності проведення профілактичних робіт $K_{п.п.п.р} = \frac{\overline{\Delta t_{п.п.р}}}{t_{екс}}$, де $\overline{\Delta t_{п.п.р}}$ — середній час між проведенням профілактичних робіт; $t_{екс}$ — загальний час експлуатації пристрою.

5. *Діагностична операція* $E_{екс5}$ (5*е*) апаратного і програмного контролю і діагностування пристроїв при їх відмовах на етапі експлуатації. Її описують такі параметри:

— тривалість контролю і діагностування $t_{кід}$, що відповідає часу, який витрачається на ідентифікацію і усунення відмови;

— коефіцієнт вартості використання апаратних засобів контролю і діагностування при відмовах $K_{в.а} = \frac{C_{а.д.в}}{C_{а.з}}$, де $C_{а.д.в}$ — вартість використання апаратних засобів контролю і діагностування при відмовах; $C_{а.з}$ — загальна вартість апаратних засобів діагностування;

— коефіцієнт вартості використання програмних засобів контролю і діагностування при відмовах $K_{в.п} = \frac{C_{п.д.в}}{C_{п.з}}$, де $C_{п.д.в}$ — вартість використання програмних засобів контролю і діагностування при відмовах; $C_{п.з}$ — загальна вартість програмних засобів діагностування;

— коефіцієнт вартості використання апаратних засобів контролю і діагностування при усуненні відмов $K_{в.в.а} = \frac{C_{а.д.в}}{C_{кід}}$, де $C_{кід}$ — загальна вартість засобів контролю і діагностування;

— коефіцієнт вартості використання програмних засобів контролю і діагностування при усуненні відмов $K_{в.в.п} = \frac{C_{п.д.в}}{C_{кід}}$.

Окрім того, існують також інші діагностичні операції, наприклад $E_{екс6...}$ (6*е*). Їх описують відповідні параметри.

Отже, розглянуто основні технологічні діагностичні операції процесів діагностування цифрових і мікропроцесорних пристроїв на всіх етапах життєвого циклу приладів, зокрема на етапах проектування, виробництва й експлуатації.

На основі часткових структур процесу діагностування на кожному з етапів життєвого циклу Ц і МПП розробляють формалізований опис технологічного процесу комбінованого поетапного діагностування, що ґрунтується на базі штучної нейронної мережі Хопфілда.

Мережа Хопфілда, що моделює процес діагностування (див. рис. 3.22), є одношаровою і неповнозв'язною. Значення j-го виходу визначають за формулою:

$$y_j = f_j \sum w_{ij} x_i, \qquad (3.38)$$

де x_i — значення i-го входу мережі; w_{ij} — вага (значення синапсу) лінії, що з'єднує x_i вхід з k-ю технологічною операцією, яка відповідає тілу k-го нейрона, або вага лінії, що з'єднує i-й нейрон з j-тим; f_j — j-та активаційна функція.

Активаційною функцією може бути двійкова логічна функція зі значенням «1» у випадку, якщо вихід з n_k нейрона нейронної мережі існує, і «0» — якщо його немає. Передаючи сигнали всередині мережі від одного нейрона до іншого, приймають значення активаційної функції, присвоєне відповідній технологічній операції діагностування. Ефективність процесу діагностування:

$$E = \frac{y_j}{\overline{Y}}. \qquad (3.39)$$

Слід наголосити, що завдання визначення ваг зв'язків і присвоєння значень активаційної функції є важкоформалізованим. Значення кожній активаційній функції присвоюють таке, щоб воно відповідало порівняній важливості технологічної операції процесу діагностування, а це, у свою чергу, — конкретному нейрону штучної нейронної мережі. Однак доцільніше значення активаційної функції обирати в діапазоні додатніх чисел від «1» до «2» (інколи до «10»), щоб числа в розрахунках були не досить великими. Це стосується і присвоєння ваг міжнейронних зв'язків. Найкраще їх обирати в діапазоні (0, 1].

Значення активаційних функцій для різних діагностичних операцій і вагових коефіцієнтів ліній зв'язку найчастіше присвоюють досвідчені спеціалісти-експерти. Значення вагових коефіцієнтів ліній, які з'єднують входи мережі з нейронами, що відповідають діагностичним операціям (синапсам (грец. synapsis — з'єднання)), приймають рівними 1.

Параметри окремих діагностичних операцій взаємопов'язані. Для обчислення їх загального впливу і визначення компонентів вектора X припускають, що вони залежать між собою мультиплікативно. Тоді загальне значення параметра вхідного сигналу:

$$P_{i_{np}} = \prod_{l=1}^{r} p_{l_{np}}, \qquad (3.40)$$

де r — кількість параметрів j-ї діагностичної операції з входом $x_{i_{np}}$; $p_{l_{np}}$ — значення l-го параметра i-го вхідного сигналу.

Для взаємонепов'язаних параметрів діагностичних операцій загальне значення параметра вхідного сигналу обчислюють за формулою:

$$P_{i_{np}} = \sum_{l=1}^{r} p_{l_{np}}. \qquad (3.41)$$

Значення вхідних сигналів вектора:

$$\overline{X} = \overline{X}_{np} \cup \overline{X}_{вир} \cup \overline{X}_{екс}, \qquad (3.42)$$

де $\overline{X}_{np}$ — множина вхідних ліній етапу проектування; $\overline{X}_{вир}$ — множина вхідних ліній етапу виробництва; $\overline{X}_{екс}$ — множина вхідних ліній етапу експлуатації.

Загальна кількість вхідних ліній мережі, що відображає процес діагностування:

$$X = X_{np} + X_{вир} + X_{екс}. \qquad (3.43)$$

Математичний вигляд одношарової мережі Хопфілда, що відображає процес діагностування пристроїв на етапі проектування, можна записати через множину значень параметрів вхідних ліній:

$$\overline{P}_{np} = \{P_{i_{np}}\}, \quad i = \overline{1, m}, \qquad (3.44)$$

де $P_{i_{np}}$ — параметр i-го входу мережі; m — кількість вхідних ліній.

Множина вагових коефіцієнтів ліній, що з'єднують входи мережі з діагностичними операціями (нейронами) діагностування:

$$\overline{W}_{x_{np},E} = \{w_{x_{i_{np}}, E_{j_{np}}}\}, \quad x_{i_{np}}, E_{j_{np}} = \overline{1, m}, \qquad (3.45)$$

де $w_{x_{i_{np}}, E_{j_{np}}}$ — ваговий коефіцієнт лінії, що з'єднує $x_{i_{np}}$-й вхід з $E_{j_{np}}$-м нейроном.

Потім визначають множину вагових коефіцієнтів ліній, що з'єднують E_r-й нейрон з E_s-м нейроном, її представляють таким чином:

$$\overline{W}_{E_r, E_s} = \{w_{E_{r_{i_{np}}}, E_{s_{j_{np}}}}\}, \quad r_i, s_j = \overline{1, n}, \qquad (3.46)$$

де $w_{E_{r_{i_{np}}}, E_{s_{j_{np}}}}$ — ваговий коефіцієнт лінії, що з'єднує E_r-й нейрон з E_s -м нейроном і виходить з E_r-го нейрона, а входить в E_s-й нейрон; n — кількість нейронів штучної нейронної мережі, що відображає процес діагностування на етапі проектування.

Мережа містить множину нейронів $\overline{E}_{np} = \{E_{k_{np}}\}, k = \overline{1, n}$, де $E_{k_{np}}$ — k-й нейрон.

Множина виходів мережі $\overline{Y}_{np} = \{y_{l_{np}}\}$, $l = \overline{1, q}$, де $y_{l_{np}}$ — l-й вихід мережі; q — кількість виходів мережі.

Наступна операція полягає у визначенні l-го виходу мереж. Його представляють як:

$$y_{l_{np}} = E_{s_{np}}\left(\sum_{r,s=1}^{n} f(E_{r_{np}})\, w_{E_{r_{np}}, E_{s_{np}}} + P_{x_{i_{np}}} w_{x_{i_{np}}, E_{s_{np}}}\right), \quad i = \overline{1, n}, \qquad (3.47)$$

де $E_{s_{np}}$ — активаційна функція $E_{s_{np}}$-го нейрона; $f(E_{r_{np}})$ — сумарна активаційна функція $E_{r_{np}}$-го нейрона; $w_{E_{r_{np}}, E_{s_{np}}}$ — ваговий коефіцієнт лінії, що з'єднує $E_{r_{np}}$-й нейрон з $E_{s_{np}}$-м нейроном; $P_{x_{i_{np}}}$ — загальне значення параметра лінії $x_{i_{np}}$; $w_{x_{i_{np}}, E_{s_{np}}}$ — ваговий коефіцієнт лінії $x_{i_{np}}$, що з'єднана з $E_{s_{np}}$-ю діагностичною операцією.

Далі переходять до визначення сумарної активаційної функції. Його здійснюють у такий спосіб:

$$f(E_{r_{np}}) = E_{r_{np}}\left(\sum_{r,k=1}^{n} f(E_{k_{np}})\, w_{E_{k_{np}}, E_{r_{np}}} + P_{x_{j_{np}}} w_{x_{j_{np}}, E_{r_{np}}}\right), j = \overline{1, n}, \qquad (3.48)$$

де $E_{r_{np}}$ — активаційна функція $E_{r_{np}}$-го нейрона; $f(E_{k_{np}})$ — сумарна активаційна функція $E_{k_{np}}$-го нейрона; $w_{E_{k_{np}}, E_{r_{np}}}$ —

ваговий коефіцієнт лінії, що з'єднує $E_{r_{np}}$-й нейрон з $E_{s_{np}}$-м нейроном; $P_{x_{j_{np}}}$ — загальне значення параметра лінії $x_{j_{np}}$; $w_{x_{j_{np}}, E_{r_{np}}}$ — ваговий коефіцієнт лінії $x_{j_{np}}$, що з'єднується з $E_{r_{np}}$-ю діагностичною операцією. Тоді:

$$y_{l_{np}} = E_{s_{np}} \left[\sum_{r,s=1}^{n} E_{r_{np}} \left(\sum_{r,k=1}^{n} f(E_{k_{np}}) w_{E_{k_{np}}, E_{r_{np}}} + P_{x_{j_{np}}} w_{x_{j_{np}}, E_{r_{np}}} \right) \cdot \right.$$

$$\left. \cdot w_{E_{r_{np}}, E_{s_{np}}} + P_{x_{i_{np}}} w_{x_{i_{np}}, E_{s_{np}}} \right]. \qquad (3.49)$$

Якщо з $E_{r_{np}}$-го нейрона немає виходів на $E_{s_{np}}$-й нейрон, то з цього нейрона йде вихідна лінія $y_{1_{np}}$ і значення $y_{1_{np}}$-го виходу дорівнює $y_{1_{np}} = f(E_{r_{np}})$, якщо ж $E_{r_{np}}$-й нейрон не має входів, то $y_{1_{np}} = E_{r_{np}}$:

$$Y_{np} = \sum_{l=1}^{q} y_{l_{np}}. \qquad (3.50)$$

Ідентично визначають значення вихідних ліній штучних нейронних мереж, що відображають процес діагностування на етапах виробництва, експлуатації та протягом усього життєвого циклу. Так:

$$y_{l_{вир}} = E_{s_{вир}} \left[\sum_{r,s=1}^{n} E_{r_{вир}} \left(\sum_{r,k=1}^{n} f(E_{k_{вир}}) w_{E_{k_{вир}}, E_{r_{вир}}} + P_{x_{j_{вир}}} w_{x_{j_{вир}}, E_{r_{вир}}} \right) \cdot \right.$$

$$\left. \cdot w_{E_{r_{вир}}, E_{s_{вир}}} + P_{x_{i_{вир}}} w_{x_{i_{вир}}, E_{s_{вир}}} \right]; \qquad (3.51)$$

$$Y_{вир} = \sum_{l=1}^{q} y_{l_{вир}}. \qquad (3.52)$$

$$y_{l_{екс}} = E_{s_{екс}} \left[\sum_{r,s=1}^{n} E_{r_{екс}} \left(\sum_{r,k=1}^{n} f(E_{k_{екс}}) w_{E_{k_{екс}}, E_{r_{екс}}} + P_{x_{j_{екс}}} w_{x_{j_{екс}}, E_{r_{екс}}} \right) \cdot \right.$$

$$\left. \cdot w_{E_{r_{екс}}, E_{s_{екс}}} + P_{x_{i_{екс}}} w_{x_{i_{екс}}, E_{s_{екс}}} \right]; \qquad (3.53)$$

$$Y_{екс} = \sum_{l=1}^{q} y_{l_{екс}}. \qquad (3.54)$$

Значення вихідної лінії $y_{1_{екс(К)}}$ штучної мережі, що відображає комбіноване поетапне діагностування пристрою протягом усього життєвого циклу (див. рис. 3.22), визначають за формулою (3.54):

$$y_{l_{екс(К)}} = E_{s_{np}} \left(\sum_{r,s=1}^{n} f(E_{r_{np}}) w_{E_{r_{np}}, E_{s_{np}}} + P_{x_{i_{np}}} w_{x_{i_{np}}, E_{s_{np}}} \right) +$$

$$+ E_{s_{вир}} \sum_{r,s=1}^{n} f(E_{r_{вир}}) w_{E_{r_{вир}}, E_{s_{вир}}} + E_{s_{екс}} \sum_{r,s=1}^{n} f(E_{r_{екс}}) w_{E_{r_{екс}}, E_{s_{екс}}} +$$

$$+ f(E_{r_{np}}) w_{E_{r_{np}}, E_{s_{вир}}} + f(E_{r_{np}}) w_{E_{r_{np}}, E_{s_{екс}}} +$$

$$+ f(E_{r_{вир}}) w_{E_{r_{вир}}, E_{s_{екс}}} + f(E_{r_{вир}}) w_{E_{r_{вир}}, E_{s_{np}}} +$$

$$+ f(E_{r_{екс}}) w_{E_{r_{екс}}, E_{s_{вир}}} + f(E_{r_{екс}}) w_{E_{r_{екс}}, E_{s_{np}}}. \qquad (3.55)$$

Отже, сумарне значення вихідних ліній мережі при реалізації комбінованого поетапного діагностування пристрою:

$$Y_{екс(К)} = \sum_{l=1}^{q} y_{l_{екс(К)}}. \qquad (3.56)$$

Процес діагностування буде найефективнішим, коли виконуватимуться всі технологічні операції діагностування і враховуватимуться всі зв'язки між ними з відповідними вагами.

Щодо кількісної оцінки, то ефективність процесу діагностування буде тим більша, чим більше числове значення отримають на задіяній вихідній лінії або кількох лініях згідно з алгоритмом діагностування пристрою:

$$Y_K = \max \sum_{l=1}^{q} y_{l_{np,вир,екс}}. \qquad (3.57)$$

Критеріями оптимізації технологічного процесу поетапного комбінованого діагностування будуть витрати на його реалізацію. Зменшити їх можна за рахунок мінімізації кількості діагностичних операцій, що реалізують у процесі діагностування, а також вартості цих операцій. Однак при оптимізації процесу діагностування треба враховувати відносне вагове значення результату реалізованих діагностичних операцій до вагового значення загальної кількості можливих для реалізації діагностичних операцій.

Тоді коефіцієнт оптимізації $K_{o.д.}$ визначають як:

$$K_{o.д.} = \frac{Y_p \cdot C}{Y \cdot C_д}, \qquad (3.58)$$

де Y_p — вагове значення результату реалізованих діагностичних операцій; Y — вагове значення загальної кількості можливих для реалізації діагностичних операцій. Y_p і Y розраховують за вищезгаданою методикою:

$$Y_P = \max \sum_{u=1}^{V} y_u, \qquad (3.59)$$

де y_u — вагове значення результату u-ї реалізованої діагностичної операції, що відповідає y_u-му виходу мережі й відображає процес комбінованого поетапного діагностування; V — кількість виходів мережі, що відповідають реалізованим діагностичним операціям.

$$Y = \sum_{l=1}^{q} y_{l_{пр,\,вир,\,екс}}, \qquad (3.60)$$

де $y_{l_{пр,\,вир,\,екс}}$ — вагове значення результату l-ї діагностичної операції, що можлива для реалізації процесу комбінованого поетапного діагностування і відповідає y_l-му виходу мережі; q — кількість виходів мережі, що відповідають можливим для реалізації діагностичним операціям.

$$K_{o.д} = \frac{\max\left(\sum_{u=1}^{V} y_u\right) \cdot C}{\sum_{l=1}^{q} y_{l_{пр,\,вир,\,екс}} \cdot C_д}. \qquad (3.61)$$

Загальну вартість C одиниці продукції (пристрою) при оптимізації процесу діагностування задають на основі запропонованої методики. Це стала величина для певного обрахунку. Вартість реалізації кожної із задіяних діагностичних операцій і вартість реалізації задіяних зв'язків між ними також відомі і задаються. За таких умов вартість $C_д$ реалізації комбінованого поетапного діагностування визначають за формулою:

$$C_Д = \min\left(C_E + C_{w_{E_r,E_s}}\right) = \min\left(\sum_{i=1}^{T} C_{E_i} + \sum_{r,s=1}^{T} C_{w_{E_r,E_s}}\right), \qquad (3.62)$$

де C_{E_i} — вартість E_i-ї діагностичної операції; $C_{w_{E_r,E_s}}$ — вартість w_{E_r,E_s}-го зв'язку між задіяними діагностичними операціями; T — кількість задіяних діагностичних операцій. Тоді:

$$K_{O.Д} = \frac{\max\left(\sum_{u=1}^{V} y_u\right) \cdot C}{\min\left(\sum_{i=1}^{T} C_{E_i} + \sum_{r,s=1}^{T} C_{w_{E_r,E_s}}\right) \cdot \sum_{l=1}^{q} y_{l_{пр,\,вир,\,екс}}}. \qquad (3.63)$$

Ефективним діагностування вважають, якщо $\frac{C_Д}{C} \le 0,25$. У такому випадку, врахувавши формулу для визначення $K_{o.д}$, отримують $Y_P \le \frac{Y}{4}$. В іншому випадку процес діагностування вважають неоптимальним.

Величину Y визначають за формулою 3.59 мережі Хопфілда, що відображає процес комбінованого поетапного діагностування пристроїв протягом усього життєвого циклу, і приймають сталою. Y_p визначають згідно з обраним алгоритмом A_i діагностування.

Отримані математичні вирази, подані як функції виходів штучної нейронної мережі, визначають ефективність процесу діагностування залежно від задіяних діагностичних операцій і зв'язків між ними. При цьому враховують сумарну активізаційну функцію нейрона, яка відображає діагностичну операцію, ваговий коефіцієнт зв'язків між діагностичними операціями та загальне значення параметрів діагностичних операцій.

Процес діагностування буде тим оптимальнішим, чим більше вагове значення результату реалізованих діагностичних операцій з відповідними зв'язками між ними щодо вагового значення загальної кількості можливих для реалізації діагностичних операцій і зв'язків між ними, а також чим менше вартість діагностування одиниці продукції (пристрою) щодо вартості продукції (пристрою).

Такий підхід стосовно діагностування Ц і МПП дає змогу охопити діагностуванням усі етапи життєвого циклу пристроїв, описати технологічний процес діагностування, задати взаємозв'язок технологічних операцій, на основі якого можна розробити алгоритми діагностування прист-

роїв, оптимізувати процес діагностування як на кожному з етапів життєвого циклу пристроїв, так і при комбінованому поетапному діагностуванні.

Моделювання і розроблення процесу діагностування МПП і С є важкоформалізованим завданням. За допомогою звичайних алгоритмічних підходів не завжди можливо його розв'язати. Тому зараз для діагностування МПП і С застосовують системи штучного інтелекту.

3.5. Інтелектуальне діагностування мікропроцесорних пристроїв і систем

Ефективність процесу тестування, крім врахування особливостей сучасних МПП як ОД, значною мірою залежить від обраного методу тестового діагностування. Крім того, в технічній діагностиці, зокрема діагностуванні МПП і С, простежуються спроби використати елементи і компоненти ШІ, зокрема самонавчання та експертні системи. Однак при цьому не вироблено чіткої методології та відповідних стратегій. Не розглядають питання взаємодоповнення компонентів ШІ в реалізації процесу тестового діагностування та оцінювання ефективності їх використання.

Складові інтелектуального діагностування

З переліку компонентів ШІ, а саме: подання знань, розв'язування задач, експертні системи, засоби спілкування з ЕОМ на природній мові, навчання, когнитивне моделювання, стратегічні ігри, оброблення візуальної інформації, робототехніка та інше в діагностуванні МПП і С найбільше зацікавлення викликають самонавчання, експертні системи, штучні нейронні мережі та нечітка логіка.

Однак, розробляючи системи ШІ, необхідно враховувати й елементи, що складають процес прийняття рішення людиною, — цілі, факти, правила, механізми висновку і спрощення.

Цілі. На першому етапі проектування засобів і процесу діагностування обчислювальної техніки слід конкретизувати цілі, виходячи з інформації про ОД. Тут уточнюють, до якого класу належить завдання, яке потрібно вирішити, обирають форму і терміни опису.

Факти. Їх визначення є одним з важливих етапів. По-перше, слід вибрати тільки потрібні факти, по-друге, вказати їх відносну важливість (вагу). Це дає змогу досягти поставлених цілей. Чим більша вага факту, тим більше значення він має при розв'язанні поставленого завдання. Далі переходять до визначення конкретних даних про ОД і відомим змінним присвоюють значення. Факти подають у певній формі. Ті, що містять конкретну інформацію, стають даними, їх заносять у базу даних, а ті, що будуть використовуватись у подальшому процесі діагностування МПП і С, зберігають в комп'ютері автоматизованої системи діагностування. Вони представлені базою знань.

Правила. Як складові ШІ, вони допомагають системі діагностування правильно оцінити дані й досягти мети. Правило узагальнює декілька фактів і спрощує їх використання.

Механізм висновку. Реалізують за допомогою способів прямого та оберненого ланцюжків суджень. У першому випадку судження йде від даних до логічного висновку, в другому — навпаки, а саме, коли висновок використовують для пошуку даних, які його підтверджують.

Механізм спрощення. Керує пошуком додаткових правил для верифікації мети доти, доки не будуть перевірені всі можливі способи досягнення мети діагностування ОД. Крім того, він ігнорує непотрібні для діагностування судження.

На рис. 3.23 зображені основні компоненти інтелектуалізації процесу діагностування МПП та С і взаємозв'язки між ними.

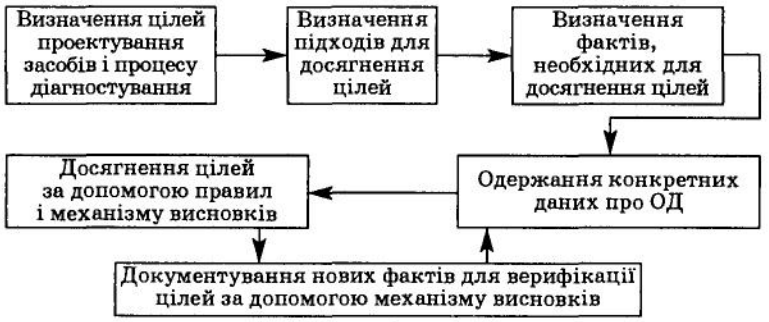


Рис. 3.23. Компоненти інтелектуалізації процесу діагностування МПП і С та взаємозв'язки між ними

Отже, слід розглянути конкретні компоненти штучного інтелекту щодо діагностування МПП і С, зокрема самонавчання, експертні системи і нечітку логіку.

Самонавчання

Елементи самонавчання ШІ автоматизованих систем діагностування почали використовувати раніше, ніж інші. Самонавчання тісно пов'язане з навчанням і в системах технічного діагностування доповнює його. Тобто те, що не вдається досягти безпосередньо навчанням, намагаються досягти самонавчанням. Це роблять тоді, коли, наприклад, для опису поведінки ОД не вистачає знань. Реалізація цих принципів може бути представлена автоматизованою системою поелементного діагностування АПД. Адаптація АПД передбачає автоматичне поповнення системи потрібною інформацією про ОД і за її надходженням — зміну змісту тест-векторів. Те саме стосується і несправностей. Отже, принцип самонавчання дає змогу оптимізувати пошук дефектів шляхом накопичення знань і забезпечити необхідну зміну алгоритмів функціювання системи.

Самонавчання систем діагностування здійснюють за певними стратегіями. Зокрема, за стратегією індуктивного навчання на основі дерева прийняття рішень, аналітичного навчання, навчання з поясненням несправностей, навчання за аналогією, навчання, що базується на теорії штучних нейронних мереж та ін. Найефективнішим є застосування комбінованої стратегії, що поєднує в собі елементи кількох стратегій, та поєднання знань експерта про несправності ОД зі знаннями, взятими з математичної моделі об'єкта.

Експертні системи технічної діагностики

Навчання і самонавчання автоматизованих систем діагностування МПП і С — оптимальний спосіб підвищити достовірність та знизити вартість процесу тестування. Однак більший ефект можна отримати при комбінації їх з іншими компонентами і елементами ШІ.

Експертні системи технічної діагностики — пакет програм, що класифікує ОД і несправності, які виникають у них, проводить їх аналіз, видає консультації і ставить діагноз.

Він орієнтований на завдання, вирішити які може тільки експертиза, зроблена спеціалістом з технічного діагностування МПП і С. Створення експертної системи технічного діагностування МПП і С потребує вирішення таких завдань:

— з'ясування цілей функціювання експертної системи діагностування;

— з'ясування та вибір способів представлення фактів і знань з технічного діагностування конкретного класу мікропроцесорних пристроїв і систем;

— вибір способу опису функції, що виконує ОД при тестуванні, та заходи щодо спрощення цього опису;

— вибір способу взаємодії оператора із системою діагностування;

— вибір системи та мови програмування для реалізації експертної системи діагностування;

— вибір методів і способів розвитку експертної системи діагностування на ґрунті виявлення нових і зміни існуючих правил.

Зібрані дані про діагностування ОД подають у формі таблиць і графіків.

Наступним компонентом, який широко застосовують для моделювання і розроблення процесу діагностування МПП і С, є штучні нейронні мережі (див. підрозділ 3.4).

У технічній діагностиці обчислювальних пристроїв і систем можна використати штучні нейронні мережі як з оберненими зв'язками, так і без них. Узагальнена структурна схема системи тестового діагностування як зовнішнього засобу тестування представлена на рис. 3.24.

До складу системи тестового діагностування (СД) входять керуюча ЕОМ (КЕОМ), генератор тестових дій (Г) і блок проміжної буферної пам'яті (БП) для запам'ятовування відповідних реакцій. КЕОМ формує блок тест-векторів і записує їх у генератор. За командою з КЕОМ на основі записаного блоку тест-векторів генератор формує сигнали тестових впливів, які через паралельні незалежні канали подають на ОД. У результаті на його виходах з'являються сигнали відповідних реакцій, що фіксують у БП. Він формує блок векторів відповідних реак-

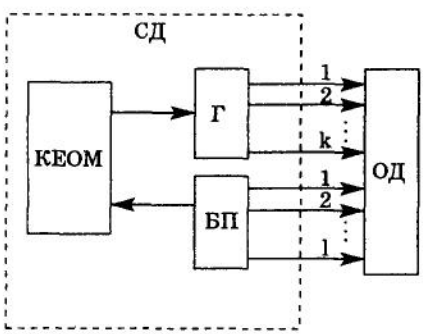


Рис. 3.24. Узагальнена структура системи тестового діагностування

цій, які зчитуються КЕОМ для подальшого аналізу. В системах тестового комбінованого діагностування блоки Г і БП суміщають. Це дає змогу змінювати напрям передавання інформації на ОД у кожному такті синхронізатора. Тобто в одному такті по кожному з каналів на ОД може подаватись сигнал тестового впливу, а далі по цьому самому каналу прийматись сигнал відповідної реакції.

Вбудовані засоби тестування цифрових структур мають приблизно таку саму структуру, тільки система діагностування міститься в складі основної апаратури.

На основі штучних нейронних мереж без обернених зв'язків можна вибудовувати системи тестування, в яких генератор і буферна пам'ять однонапрямлені (напівруплексні) щодо ОД.

Однак найбільше цікавить побудова систем діагностування цифрових структур на базі штучних нейронних мереж з оберненими зв'язками. Це дає змогу системі самонавчатись, використовуючи еталон. Маючи еталонний цифровий пристрій, наприклад змонтовану друковану плату, система при під'єднанні його як ОД сама генерує тест. У несправному ОД автоматично вказують місце прояву несправностей, їх опис заносять до бази даних, зокрема і тих несправностей, що проявилися вперше. Отже, такий підхід значно підвищує ефективність процесу діагностування МПП і С. Немає необхідності складати тест для еталону. Операції знаходження місця прояву несправностей та поповнення бази даних автоматизують. Фрагмент структури генератора системи тестового комбінованого діагностування представлений на рис. 3.25.

Принцип роботи генератора такий: з виходів ОД y_1-y_n сигнали відповідних реакцій через елементи M_1-M_n, які виконують функцію дозволу або заборони проходження сигналів, надходять для сумування на входи опрацьовуючих одиниць (нейронів) 1-n. Вхідний сигнал кожної опрацьовуючої одиниці складається із зовнішнього щодо замкнутої системи «генератор — ОД» сигналу x_1, зваженого «синаптичного» сигналу, який є результатом дії сигналу відповідної реакції з ОД, і суми сигналів $\sum_{j=1}^{n-1} w_j$, що надходять з дозволених виходів інших одиниць. Вихідний сигнал i-ї опрацьовуючої одиниці визначають з рівнянь:

$$S_i = x_i + y_i + \sum_{j=1}^{n-1} w_j; \qquad z_i = f(S).$$

Тестування ОД закінчують при періодичному повторенні стану мережі. Якщо стан мережі збігається зі станом

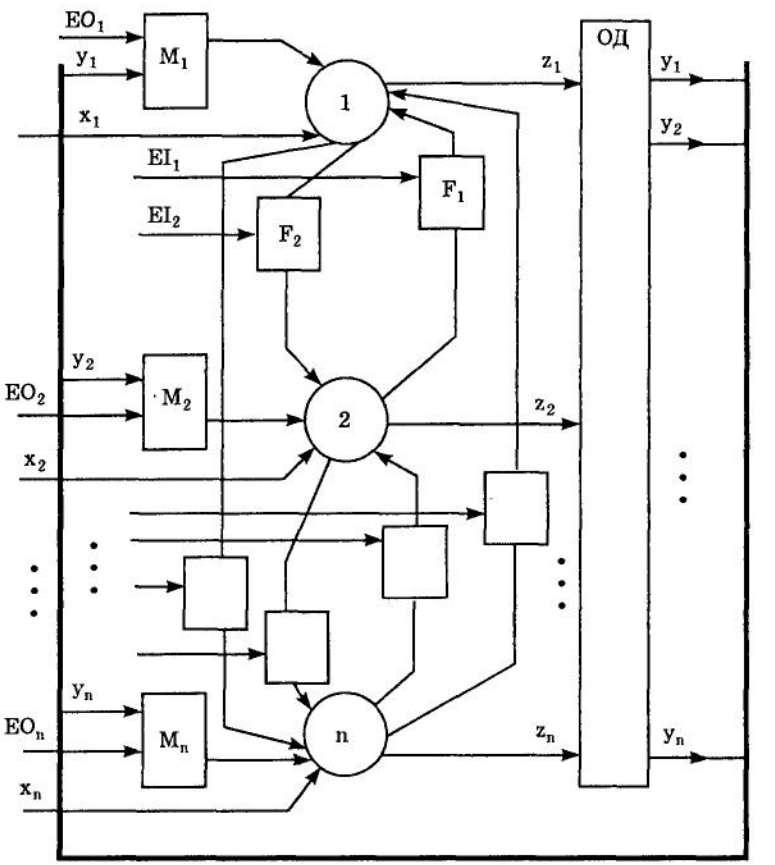


Рис. 3.25. Фрагмент структури генератора системи тестового комбінованого діагностування

для еталонного ОД, тестований ОД вважають справним, в іншому випадку, навпаки, — несправним. Аналіз розбіжності станів еталонного і діагностованого ОД дає інформацію про тип і місце прояву несправності.

Нечітка логіка

У багатьох ситуаціях є необхідність описувати різні процеси, зокрема і технічні, множинами, для яких не можна прийняти беззаперечне рішення про належність певної частини елементів.

Щодо технічної діагностики МПП і С це можуть бути такі вирази: «справний», «частково несправний», «несправний». Ступінчаста характеристична функція «0/1» чіткої множини може переходити в неперервну функцію належності нечіткої (фуцці) множини.

Комбінація компонентів ШІ значно підвищує ефективність тестового комбінованого діагностування МПП і С. Цього досягають, зменшуючи витрати на розроблення тестів і час реалізації процесу діагностування, водночас збільшуючи глибину пошуку дефектів.

Отже, дослідницькі роботи в напрямі інтелектуалізації процесу тестування МПП і С хоч і мають певні напрацювання, однак вони ще не достатньо розвинуті. Детальнішого вивчення потребують питання внеску кожного з компонентів ШІ для підвищення ефективності діагностування. Тому ці напрями є перспективними.

У попередньому матеріалі основну увагу приділяли методології діагностування Ц і МПП та обчислювальних систем. Реалізацію методів діагностування можна здійснювати за наявності відповідних апаратних і програмних засобів. Вони різні для кожного зі згаданих методів. Тому необхідно розглянути їх окремо.

Запитання. Завдання

1. У чому суть технічної діагностики? Які основні завдання вона вирішує?
2. Поясніть, у чому суть технічного діагностування.
3. Обгрунтуйте поняття «технічний контроль в обчислювальній техніці».
4. Назвіть і охарактеризуйте види контролю.
5. Які основні відмінності кількісного контролю від інформаційного?
6. Що розуміють під діагностуванням в обчислювальній техніці? Назвіть його види.
7. Охарактеризуйте процедуру ідентифікації несправності.
8. У чому суть функційного діагностування і чим воно відрізняється від тестового?
9. Дайте характеристику поняттю «ефективність системи діагностування». Яким чином її визначають?
10. Обгрунтуйте поняття «несправність Ц і МПП».
11. У чому суть збою Ц або МПП?
12. Що, на вашу думку, є причинами виникнення дефектів на етапі виробництва обчислювальних пристроїв?
13. Назвіть типи несправностей обчислювальних пристроїв. Дайте їм загальну характеристику.

14. Поясніть взаємозв'язок класів і підкласів несправностей обчислювальних пристроїв.
15. Які існують відмінності між представленням цифрового пристрою на вентильному рівні від його представлення на автоматному рівні?
16. У чому суть представлення МПП багаторівневою (стратовою) моделлю?
17. Назвіть і охарактеризуйте основні принципи тестового структурного діагностування.
18. У чому полягають основні принципи тестового комбінованого діагностування?
19. Які, на вашу думку, переваги тестового комбінованого діагностування перед покомпонентним і структурним?
20. Які комбінації систем діагностування використовують на виробництві для реалізації різних стратегій діагностування?
21. Охарактеризуйте етапи пошуку несправностей пристроїв з компонентами підвищеного ступеня інтеграції, згідно з ієрархічним підходом при тестовому комбінованому діагностуванні.
22. Назвіть особливості мікропроцесорних пристроїв як об'єктів діагностування. Чим вони зумовлені?
23. Які основні принципи сучасної методології тестування МПП?
24. У чому суть способу організації діагностування багатопроцесорних систем за допомогою централізованого жорсткого і плаваючого ядра, розподіленого жорсткого і плаваючого ядра?
25. Охарактеризуйте функції, які виконує обслуговуючий сервісний процесор.
26. Які події реєструють універсальний і спеціалізований монітори?
27. У чому полягає принцип діагностування локальних комп'ютерних мереж?
28. Розкрийте суть технологічного процесу діагностування Ц або МПП.
29. Які діагностичні операції характерні процесові діагностування МПП на етапі проектування? Дайте їм характеристику.
30. Назвіть і охарактеризуйте діагностичні операції, які застосовують при реалізації процесу діагностування МПП на етапі виробництва.
31. У чому особливості реалізації процесу діагностування МПП на етапі експлуатації?
32. Дайте характеристику комбінованій поетапній моделі процесу діагностування МПП.
33. За яким принципом визначають ефективність процесу діагностування?
34. Назвіть компоненти штучного інтелекту, які є найприйнятнішими для застосування в технічній діагностиці обчислювальних пристроїв та систем? Дайте їм характеристику.
35. Наведіть приклади використання самонавчання в автоматизованих системах діагностування.
36. Які завдання вирішують при створенні експертних систем діагностування МПП і С?
37. З якою метою в системах діагностування застосовують елементи нечіткої логіки?

4.

Засоби діагностування цифрових і мікропроцесорних пристроїв

Для визначення місця прояву несправностей з метою їх подальшого усунення застосовують апаратні, програмні або змішані засоби діагностування Ц і МПП. Усі вони, залежно від випадку, тією чи іншою мірою придатні для діагностування ОД. Отже, необхідно проаналізувати кожен з них. На перших етапах розвитку обчислювальної техніки найбільшого поширення набули апаратні засоби діагностування Ц і МПП. Тому їх слід розглянути насамперед.

4.1. Апаратні засоби діагностування цифрових і мікропроцесорних пристроїв

Сьогодні важко надати перевагу апаратним або програмним засобам діагностування Ц і МПП. Найкращий результат дає їх комбінація,оскільки програмні засоби діагностування, зокрема діагностичні програми, мають відпрацьовуватись за допомогою апаратних засобів. Розгляд апаратних засобів слід почати з їх класифікації.

Класифікація апаратних засобів контролю і діагностування

Апаратні засоби діагностування Ц і МПП поділяють на зовнішні і вбудовані. Зовнішні засоби конструктивно від'єднані від ОД, а вбудовані входять до його складу.

До *зовнішніх апаратних засобів контролю і діагностування Ц і МПП* належать: контрольно-вимірювальні прилади, пульти, стенди; аналізатори і тестери; контрольно-діагностичні комплекси і системи.

Контрольно-вимірювальні прилади, пульти і стенди. Їх використовують для ремонтів, сервісного обслуговування, вхідного і вихідного контролю при ручному (неавтоматизованому) способі та слабко вираженій автоматизації цих процесів.

Аналізатори і тестери. Застосовують для параметричного, функційного, тестового або комбінованого контролю і діагностування ІС, у тому числі ВІС і НВІС, а також змонтованих на їх базі вузлів, пристроїв, що розміщують на друкованих платах. За допомогою більшості цих засобів процес контролю автоматизують, а самі вони можуть бути як універсальні, так і спеціалізовані. Керування їхньою роботою та оброблення діагностичної інформації здійснює вмонтована в них мікроЕОМ.

Контрольно-діагностичні комплекси і системи. Їх використовують для здійснення всіх чи окремих видів контролю і діагностування ІС різного ступеня інтеграції, змонтованих друкованих плат, цифрових пристроїв і систем. Процес контролю і діагностування за допомогою цих засобів частково або повністю автоматизований. У структурі комплексу чи системи є одна або кілька ЕОМ.

Існує багато критеріїв класифікації апаратури контролю і діагностування обчислювальних пристроїв. Серед них можна виокремити: функційне призначення; види контролю і діагностування; ступінь автоматизації процесу контролю і діагностування; етапи контролю і діагностування; конструктивне виконання засобів контролю і діагностування.

Функційне призначення. Характеризує засоби контролю і діагностування за спеціалізацією. Однак це не означає, що апаратуру для певного виду контролю чи діагностування не можна використати для інших операцій. Так, системи діагностування змонтованих друкованих плат з ІС різного ступеня інтеграції успішно використовують для контролю самих ІС.

Засоби контролю та діагностування обчислювальних пристроїв і систем за функційним призначенням можна поділити на апаратуру для контролю і діагностування інтегральних схем малого і середнього ступенів інтеграції, друкованих плат із цими ІС, ВІС (НВІС) та мікропроцесорів, мікропроцесорних пристроїв, комп'ютерних систем.

Види контролю і діагностування. За цим критерієм засоби поділяють на апаратуру параметричного, функційного, тестового контролю і діагностування.

Ступінь автоматизації процесу контролю і діагностування. Визначає продуктивність і технічні можливості засобів. Тому цей критерій дуже важливий. Ступінь автоматизації має градацію.

Апаратура з ручним керуванням. Їх використовують як контрольно-вимірювальні прилади та прості стенди на різних етапах життєвого циклу обчислювальних пристроїв. Це зумовлено простотою користування і порівняно низькими вартісними витратами. Однак вона малопродуктивна.

Автоматизовані засоби (системи) контролю і діагностування. Їх використовують найширше. Вони мають значно більшу продуктивність. Використовуючи такі системи вручну, можна проводити контактування ОД із системою, початкове завантаження системи і деякі інші операції. Контроль або діагностування, як правило, проводять у діалоговому режимі.

Засоби автоматичного контролю і діагностування. Вони найефективніші. Увесь процес контролю чи діагностування проходить автоматично після задання початкових умов.

Етапи контролю і діагностування. Тісно пов'язані з етапами життєвого циклу обчислювальних пристроїв і систем, зокрема етапами проектування, виробництва і експлуатації.

Етап проектування. За допомогою лабораторних досліджень перевіряють правильність синтезованих цифрових структур, а також налагодження і коригування технологічного процесу виготовлення пристроїв.

Етап виробництва. Його починають з відбраковування непридатних комірок кристала. До складу такої апаратури входить зондова установка для перевірки фрагментів структури кристала. До апаратури, за допомогою якої здійснюють вихідний контроль, ставлять досить високі вимоги: велика продуктивність, стабільність заданих умов, точність. Це здебільшого автоматизовані системи контролю або діагностування, до складу яких входить одна чи кілька керуючих ЕОМ. За допомогою апаратури вхідного контролю проводять відбір ІС різних ступенів інтеграції за критеріями надійності, електричної і функційної сталості параметрів.

Етап експлуатації. Під час діагностування, сервісного обслуговування і ремонтів обчислювальних пристроїв і систем за допомогою апаратних засобів проводять контроль ІС, змонтованих друкованих плат і вузлів, а також пристроїв у цілому з метою пошуку дефектних компонентів і заміни їх справними. Також при цьому широко використовують програмні засоби.

Конструктивне виконання засобів контролю і діагностування. Залежить воно від експлуатаційного призначення засобів. Стаціонарну апаратуру використовують на етапах проектування і налагодження обчислювальних пристроїв і під час їх масового серійного виробництва. При виробництві невиликих серій частіше застосовують настільні варіанти діагностичної апаратури. Експлуатація сервісного обслуговування та ремонтів потребує здебільшого портативної апаратури.

Прилади з ручним керуванням. Інколи для пошуку дефектних компонентів обчислювальних пристроїв достатньо простих приладів з ручним керуванням, зокрема вольтметрів, низьковольтних тестерів цілісності провідників, багатоканальних осцилографів, частотомірів та ін. Найчастіше під час ручного пошуку несправностей використовують логічний пробник, логічний пульсатор, струмовий зонд.

Логічний пробник. Призначений для реєстрації логічних станів та імпульсів у логічних схемах. Він не потребує інтерпретації напруг. При підключенні до схеми світлодіоди цього приладу показують наявність *H*-рівня, *L*-рівня або імпульсів. Деякі логічні пробники можуть працювати як з ТТЛ-, так і з МОН-схемами і виявляти імпульси різних типів. Щоб почати роботу з приладом, його підключають до відповідного джерела живлення. У процесі реєстрації логічних рівнів чи імпульсів видаються відповідні звукові сигнали.

Один з типових логічних пробників має три світлодіоди, які показують сім комбінацій, зафіксовані в табл. 4.1.

Таблиця 4.1

Комбінація сигналів логічного пробника

Код комбінації	Світлодіоди			Сигнал	Пояснення
	High	Low	Pulse		
0	○	○	○	Немає	Пробник не підключений. Контрольна точка обірвана. Високоімпедансний стан
4	▨	○	○	5 В «1» 0 В	H-рівень (високий рівень)
2	○	▨	○	5 В «0» 0 В	L-рівень (низький рівень)
5	▨	○	▨		H-рівень з імпульсами (високий рівень)
3	○	▨	▨		L-рівень з імпульсами (низький рівень)
1	○	○	▨		Прямокутні імпульси з частотою більш ніж 100 кГц
7	▨	▨	▨		Прямокутні імпульси з частотою менш ніж 100 кГц

Комбінації 5 і 3 показують, що є потік несиметричних імпульсів, які в кожному циклі відповідають *H*- або *L*-рівню. Комбінації 1 і 7 дають змогу приблизно оцінити частоту імпульсів.

Логічний пульсатор. Генерує прямокутні сигнали. Натиском кнопки на корпусі приладу в контрольну точку схеми подають один цикл прямокутного сигналу (*H*- чи *L*-рівень). Він може змінити один рівень контрольної точки на інший. Головна складність цього способу в тому, що треба досконало знати схему, яку діагностують.

Проте є випадки (двонапрямлені шини, високоімпедансний стан), коли для ідентифікації несправностей недостатньо простого вимірювання напруги. Необхідно знати ще і величини струмів, що протікають по шинах. У таких ситуаціях використовують індуктивні перетворювачі магнітного потоку. Зокрема, струмовий зонд (індикатор струму).

Струмовий зонд. Прилад належить до безконтактних методів вимірювання імпульсних струмів. Чутливий елемент зонда — трансформаторний перетворювач. Прилад дає змогу простежити шлях струму від генератора імпульсів до дефектного елемента (компонента), що є, наприклад, джерелом нульового сигналу на шині, яке спричинене коротким замиканням між двома виходами логічних елементів. Сприймаюча катушка струмового зонду завдяки електромагнітній індукції виявляє змінний струм імпульсного сигналу, коли він переключається з одного рівня в інший, однак зовсім не реагує на постійний струм.

Такі прилади мають велику похибку і потребують прецизійного (франц. précision — точність, визначеність) виконання через вузькі провідники і малий крок між виводами ІС на друкованих платах сучасних обчислювальних пристроїв.

Автоматизовані засоби контролю і діагностування. Вони забезпечують якісно новий рівень контролю і діагностування обчислювальних пристроїв. Серед них найчастіше використовують логічний і сигнатурний аналізатори.

Логічний аналізатор. За його допомогою проводять логічний аналіз стану ОД (ОК) у часових проміжках. Він автоматизує етапи реєстрації та відображення інформації про реальні процеси в ОД. Цей прилад скорочує час налагодження МПП у 10—20 разів. Його застосовують на різних етапах життєвого циклу пристроїв, у тому числі на етапах проектування, виробництва та експлуатації.

Порівняно з ручними засобами контролю, зокрема осцилографом, вольтметром, логічним пробником та логічним пульсатором, він може реєструвати послідовності логічних сигналів одночасно і синхронно в багатьох контрольних точках протягом тривалих інтервалів часу в однократному і періодичних режимах. Також логічний аналізатор реєструє послідовності станів у зв'язку з однократними подіями (керуючі сигнали, команди), стани в контрольних точках в інтервалі часу, що перемежовується з обраним або до, або після обраного, асинхронні події. Крім того, логічний аналізатор оперативно відображає зареєстровану інформацію в різних режимах і форматах (це зручно для спостереження і дає змогу ідентифікувати несправності, що перемежовуються), серед них — режим порівняння таблиць, виявлення коротких імпульсів («гліч»-імпульсів) збирання даних з розділенням пам'яті.

Режими відображення діагностичної інформації в часі не залежать від режимів реєстрації. Індикація (лат. indica-

tio — вказування, визначення) зареєстрованої інформації може тривати як завгодно довго.

Робочі операції, що проводить користувач логічного аналізатора під час контролю чи діагностування, переважно автоматизовані. Їх зміст полягає в аналізі інформаційного потоку системою запуску аналізатора, синхронізації та реєстрації діагностичної інформації, генерації вхідних дій. Суть операцій оброблення в тому, щоб відображати відповідні реакції, перетворювати, вимірювати, порівнювати, шукати і приймати рішення. У логічних аналізаторах вони не автоматизовані і слабко формалізовані.

Під час діагностування МПП прилади підключають насамперед до шин адреси, даних, керування. Для виявлення місця прояву несправності підбирають контрольні точки синхронізації та запуску. Логічний аналізатор складається з декількох елементів.

Запам'ятовуючий пристрій (ЗП). Призначений для приймання і зберігання діагностичної інформації з ОД. Його ємність залежить від кількості вхідних каналів аналізатора.

Пристрій перетворення діагностичної інформації (ППІ). Побудований на базі мікропроцесорного контролера (мікроЕОМ) або в більш ранніх розробках (70—80-ті) у вигляді автомата з жорсткою логікою.

Пристрій введення (ПВ). Здійснює перетворення інформації, що надходить на входи аналізатора у двійковий код.

Пристрій виведення (ПВив.). Це — дисплей для відображення діагностичної інформації після її оброблення пристроєм керування.

Пульт керування (ПК). Призначений для задання оператором режимів роботи аналізатора.

Пристрій аналізатора інформації (ПАІ). Генерує сигнали керування роботою логічного аналізатора при збіганні кодів на виході пристрою введення з кодами, що встановлює оператор.

Наведений перелік узагальнює спрощена структурна схема логічного аналізатора (рис. 4.1). Аналізатор розгортає стан ОД у дискретному часі. Генератор задає моменти часу, впродовж яких реєструється стан ОД у вигляді вектора, кількість елементів якого відповідає кількості вхідних каналів аналізатора. Кожному елементу в пам'яті відповідає логічна двійкова змінна. Інформація у вигляді наборів векторів відображається на дисплеї.

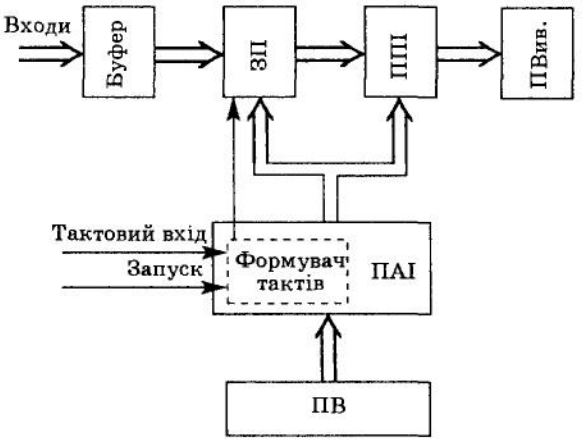


Рис. 4.1. Структурна схема логічного аналізатора

Основними характеристиками логічних аналізаторів є кількість вхідних каналів, глибина ЗП (біт/канал) та максимальна частота реєстрації сигналів діагностичної інформації.

За функційним призначенням логічні аналізатори поділяють на: 1) аналізатори логічних часових послідовностей (АЛЧП), що використовують синхронну реєстрацію сигналів; 2) аналізатори логічних станів (АЛС) із синхронною реєстрацією.

АЛЧП — складніші прилади, ніж АЛС. Частота їх реєстрації, як правило, на порядок вища від частоти магістралі мікропроцесорних пристроїв. Прикладом такого аналізатора є аналізатор VP-3662A фірми «Matsustita». Він має 32 вхідних канали при частоті реєстрації 100 МГц. Прилад аналізує процеси в системних магістралях і призначений для налагодження і ремонту МПП і мікроЕОМ.

Аналізатор фірми «Omni Logic» може реєструвати до тисячі відліків цифрового сигналу по 16-ти каналах і 330 відліків по 8-ми каналах залежно від модифікації. Передбачена можливість фіксації коротких імпульсів завад з мінімальною тривалістю 10 нс по 4-х каналах.

Подальше удосконалення аналізаторів спрямоване на комбінацію властивостей різних контрольно-вимірювальних приладів, удосконалення архітектури і програмного забезпечення. Так, аналізатор 2610 фірми «Genrad» поєднує властивості логічного аналізатора, АЛЧП, цифрового мультиметра, осцилографа та має засоби проектування діагностичних програм.

Сигнатурний аналізатор. Принцип його роботи багато в чому збігається з методами контролю за допомогою логічного аналізатора. Це, зокрема, вибір контрольних точок ОД, складання тестових впливів, аналіз відповідних реакцій та ін. Проте він має і певні відмінності. Сигнатурний аналізатор використовують тоді, коли істотно зростає довжина тестових послідовностей, особливо для пристроїв з компонентами підвищеного ступеня інтеграції. Це вимагає враховувати надійність обладнання для генерації тестів. Тому щоб знизити надлишковість цієї апаратури, використовують компактну форму представлення тестових послідовностей для її аналізу після проходження через пристрій, що контролюють чи діагностують.

Стискання відповідних реакцій реалізують на регістрах зсуву зі зворотними зв'язками. Принцип роботи сигнатурного аналізатора полягає в тому, що відповідні реакції з ОД перекодовуються в ньому на короткі коди (сигнатури). Їх порівнюють з еталонними, які отримують або шляхом розрахунків, або з тими, що знімають із заздалегідь справного ОД. Подавання тестів здійснюють зі спеціального генератора або програмними методами. Необхідно дотримуватись, щоб інтервал часу подавання на ОД сигналів тестових впливів дорівнював часу аналізу.

Прикладом технічної реалізації сигнатурного аналізатора є функційна схема аналізатора фірми «Hewlett-Packhard», яку описують поліномом $a(x) = 1 \oplus x^7 \oplus x^9 \oplus x^{12} \oplus x^{16}$. Цей аналізатор довгий час був неофіційним стандартом пристроїв подібного типу. Його схема складається з 5-вхідного суматора за модулем 2, 16-розрядного регістра зсуву і кількох логічних елементів для організації сигналів «Старт» і «Стоп». Розряди регістра зсуву розбивають на чотири тетради, вміст кожної з них визначає значення 16-річної цифри сигнатури. Її представлення може здійснюватись, наприклад, семисегментним цифровим індикатором.

Засоби автоматичного контролю і діагностування. Їх використання дає істотні переваги, а саме: задавши для приладу початкові умови, подальший процес контролю і діагностування обчислювальних пристроїв і систем відбувається автоматично. Це значно спрощує керування ним. Отже, детально розглядати цей вид автоматизації процесу контролю і діагностування немає потреби.

Розмаїття апаратних засобів діагностування дає змогу обрати їх щодо конкретних ОД. При цьому головну увагу звертають на доцільність використання того чи іншого засобу та економічну ефективність процесу діагностування, реалізованого на його базі.

Вимоги до систем діагностування цифрових і обчислювальних пристроїв

В основу архітектурних вимог до систем діагностування цифрових і мікропроцесорних пристроїв покладені загальні принципи, яких дотримуються, розробляючи обчислювальні пристрої і системи.

Невід'ємними властивостями систем діагностування є гнучкість, нарощуваність, простота спрягання окремих вузлів і модулів системи з керуючою ЕОМ (КЕОМ). Керуюча ЕОМ повинна використовуватись як для оброблення результатів тестування, так і для розроблення і налагодження тестових програм. Цього можна досягти, дотримуючись основних принципів схемотехнічного проектування, зокрема модульності, магістральності та мікропрограмованості.

Принцип модульності. Передбачає проектування системи діагностування на основі відкритості архітектури. При цьому, з одного боку, повинна бути можливість комплектування і нарощування системи діагностування за рахунок нових пристроїв і блоків, а з іншого — можливість нарощування пристроїв і блоків необхідною кількістю конструктивно закінчених модулів і вузлів (друковані плати з вмонтованими в них ВІС і НВІС та іншими компонентами). Всі модулі повинні забезпечувати конструктивно-технологічну та електричну сумісність між собою. Це значить, що перероблення нижчого модуля не повинно призводити до перероблення верхнього.

Принцип магістральності. Полягає в тому, що проектування систем діагностування проводять на основі такої структури, де блоки з'єднані між собою однією чи кількома магістралями передавання інформації, які утворюють багаторівневу систему каналів зв'язку. Магістраль вищого рівня повинна мати вищу пропускну здатність для передавання тестової інформації і забезпечувати необхідну швидкодію модулів нижчого рівня.

Магістрально-модульна структура. Цей тип систем діагностування узгоджує продуктивність КЕОМ (швидкість передавання тест-векторів) з пропускною здатністю модулів і блоків системи. Спрягання КЕОМ з іншими блоками системи забезпечує високошвидкісна магістраль. Магістрально-модульна структура побудови систем діагностування дає змогу організувати автономну роботу необхідних модулів.

Принцип мікропрограмованості. Це реалізація частини функцій протоколів, що не піддаються мікромініатюризації за допомогою схемної логіки, у вигляді мікропрограм. Вони містяться у напівпровідниковому постійному запам'ятовуючому пристрої і забезпечують вищу швидкодію системи під час обміну тестовою інформацією.

Головною функцією систем діагностування є реалізація відповідних методів діагностування обчислювальних пристроїв і систем.

Структура систем діагностування. Сучасні системи діагностування обчислювальних пристроїв є складними мультипроцесорними системами з різноманітними уніфікованими і оригінальними вузлами, модулями і блоками. Натепер ще немає змоги цілком формалізувати процес розроблення систем діагностування, тому передусім поєднують елементи проектування і досвіду розробника.

Під час вибору структури системи діагностування розглядають і комплексно вирішують такі завдання:

— аналіз ОД;

— вивчення етапів життєвого циклу ОД;

— визначення функцій, які повинна виконувати система діагностування;

— з'ясування можливості й рівня автоматизації процесу діагностування ОД;

— аналіз вартості апаратних і програмних засобів та експлуатаційних витрат на систему діагностування.

Системи діагностування складаються з таких основних структурних частин: керуюча ЕОМ (КЕОМ); пристрій накопичення і видавання сигналів тест-векторів на ОД (ПВ); пристрій прийому сигналів відповідних реакцій з ОД (ПП).

КЕОМ призначена для формування блоку (файла) тест-векторів впливів; запису блоку в ПВ; зчитування з ПП тест-векторів реакцій; керування роботою пристроїв ПВ і ПП; аналізу тест-векторів реакцій і видавання результатів.

ПВ виконує запам'ятовування необхідної кількості тест-векторів впливів у буферній пам'яті і видавання їх із заданою частотою на ОД (КПСІ чи ПКПСІ).

ПП здійснює прийом сигналів відповідних реакцій і формування на їх основі тест-векторів реакцій для зчитування КЕОМ.

Розробляючи системи діагностування, крім визначення їхніх функцій, розв'язують два основних завдання: розрахунок параметрів системи і вибір структури.

До основних параметрів системи належать: кількість каналів спрягання з ОД; об'єм буферної пам'яті тест-векторів; швидкість (частота) подавання сигналів тестових впливів на ОД.

Кількість каналів спрягання з ОД. Визначають зі схеми електричної принципової ОД. Вона повинна, як мінімум, дорівнювати кількості входів і виходів об'єкта, який діагностують.

Обсяг буферної пам'яті тест-векторів. Він має бути таким, щоб забезпечувати подавання всього блока тест-векторів на ОД без розділення його на кілька окремих модулів.

Швидкість подавання сигналів тестових впливів на ОД. Вона повинна забезпечувати подавання сигналів тестових впливів за прийнятний час у процесі тестування обчислювального пристрою, а також діагностування ОД у динамічних режимах на максимальних робочих частотах. Ця швидкість залежатиме від структури системи діагностування. Тому одне з головних завдань розробників — вибрати структуру, яка б забезпечувала задану швидкість тестування.

Узагальнена структура системи діагностування показана на рис. 4.2.

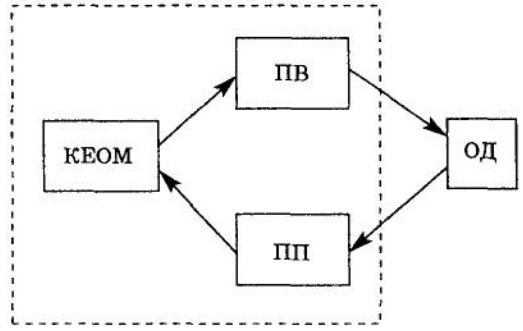


Рис. 4.2. Структурна схема системи діагностування, до складу якої входять КЕОМ, ПВ і ПП

На схемі зображено пристрої та інформаційні потоки для систем, у яких буферна пам'ять тест-векторів впливів і тест-векторів реакцій розділена. Однак у певних системах вона може бути і суміщена.

Структура систем тестового комбінованого діагностування. Питання аналізу та структури систем тестового загального і покомпонентного діагностування досить широко висвітлені в літературі. Тому основну увагу слід приді-

лити системі тестового комбінованого діагностування (СКД).

Найчастіше до складу СКД входять чотири основні структурні частини. Це КЕОМ, ПВ, ПП і додатково спецпроцесор керування введенням-виведенням тестової інформації (СП КВВ). Функціями СП КВВ є керування подаванням тестових впливів та інших сигналів на ОД, керування прийомом сигналів відповідних реакцій, роботою інших пристроїв системи. Відповідно з КЕОМ знімають перераховані функції, але додають функцію керування СП КВВ.

За такого складу СКД можливі різні варіанти її конфігурації. Слід розглянути найзагальніші з них.

До складу системи входять: одна КЕОМ, один СП КВВ і по одному ПВ і ПП (рис. 4.3).

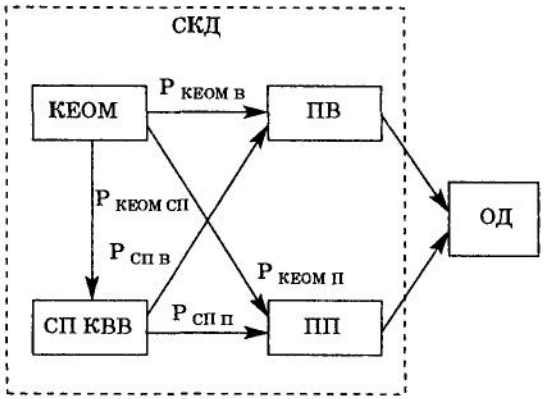


Рис. 4.3. Структурна схема СКД, до складу якої входить спецпроцесор керування введенням-виведенням тестової інформації

Багато функцій, які реалізують СКД, ідентичні функціям систем діагностування цифрових пристроїв, однак мають свої особливості реалізації окремих функцій і відповідні додаткові засоби.

Важливим є формування сигналів тестових впливів відповідно до часових діаграм роботи КПСІ і ПКПСІ. Тому для формування сигналів, що подають на керуючі входи, у складі системи повинні бути спеціальні пристрої, здатні генерувати сигнали запрограмованої тривалості і затримки стосовно сигналів, що подають на шини адреси даних. Щоб досягти максимальної швидкості подавання сигналів тестових впливів на ОД, цю генерацію доцільно проводити в одному такті синхронізатора.

З метою ідентифікації динамічних несправностей подавання сигналів тестових впливів на ОД слід здійснювати на максимальних робочих частотах. Для цього у складі СКД має бути спеціальний програмований генератор, швидкість подання сигналів тестових впливів і прийому відповідних реакцій якого не залежить від швидкодії КЕОМ.

Під час відпрацювання тестів необхідно змінювати напрямок передавання тест-векторів у кожному з наступних тактів синхронізатора. Відповідно і канали блока буферної пам'яті СКД повинні поєднувати функції подавання тестових впливів і прийому відповідних реакцій.

Для аналізу сигналів реакцій не тільки за потенційними рівнями, а й за іншими діагностичними параметрами, зокрема за величиною струму, у складі СКД мають бути засоби вимірювання цих параметрів і засоби для вимірювання часу затримки поширення сигналів.

СКД повинна реєструвати сигнали відповідних реакцій в різних точках тактових інтервалів, що генеруються синхронізатором. Це забезпечує ідентифікування нелогічних динамічних несправностей.

Засоби СКД мають своїм завданням діагностувати шинні структури, ЦАП і АЦП у складі обчислювальних пристроїв. Наявність генератора аналогових впливів у складі СКД пов'язана з діагностуванням АЦП в складі ПКПСІ. За допомогою такого генератора на вхід АЦП подають сигнал із запрограмованою величиною струму.

Для реалізації імовірнісного та псевдовипадкового тестування до складу СКД входить сигнатурний аналізатор.

З урахуванням завдань тестового комбінованого діагностування ПКПСІ і розглянутих вище особливостей до складу СКД можуть входити: КЕОМ, СПКВВ, оперативний запам'ятовуючий пристрій (ОЗП) спецпроцесора; блок пам'яті тестових впливів і відповідних реакцій (БП ТВ і ВР); генератор програмований (ГП); таймер інтервальний (ТІ); блок формування затримок сигналів тестових впливів (БФЗ); блок формування тривалості сигналів тестових впливів (БФТ); генератор аналогових сигналів (ГАС); блок вимірювання тривалостей часо-імпульсних сигналів (БЧІВ); блок вимірювання параметрів аналогових сигналів (БАВ); багатовходовий сигнатурний аналізатор (БСА); блок формування рівнів напруг тестових сигналів (БФРН); пристрій спрягання з ОД (ПС); багатоконтактні пристрої для контактування з внутрішніми точками ОД (БПК).

— *КЕОМ* системи призначена для формування дій тест-векторів та запису їх у БП ТВ і ВР, формування і запису в СП КВВ керуючої інформації блоками системи, прийому і аналізу тест-векторів відповідних реакцій і видавання результатів тестування.

— *СП КВВ* призначений для керування роботою блоків БП ТВ і ВР, ГП, ТІ, БФЗ, БФТ, ГАС, БЧІВ, БСА, БФРН та часткового оброблення результатів тестування.

— *БП ТВ і ВР* формує послідовності тестових дій і через БФРН і ПС подає їх на ОД, а також реєструє сигнали відповідних реакцій.

— *ГП* задає частоту подавання сигналів тестових впливів на ОД та частоту прийому відповідних реакцій.

— *ТІ* задає кількість тест-векторів, що подають на ОД.

— *БФЗ* затримує сигнали тестових впливів відносно початку такту з метою формування їх відповідно до часових діаграм.

— *БФТ* формує сигнали тестових дій відповідно до заданої тривалості.

— *ГАС* генерує сигнали тестових впливів із запрограмованою величиною струму.

— *БЧІВ* призначений для вимірювання затримки поширення сигналів у КПСІ і ПКПСІ, а також тривалості сигналів відповідних реакцій.

— *БАВ* вимірює величину струму сигналів відповідних реакцій та сигналів тестових впливів.

— *БСА* дає змогу знімати сигнатури з відповідних виходів ОД.

— *БФРН* спільно з програмованим джерелом живлення задає необхідні рівні тестових сигналів (ТТЛ, МОН та ін.).

— *ПС* призначений для спрягання системи з ОД.

Узагальнена структура такої СКД подана на рис. 4.4. Блоки БЧІВ, БАВ, БСА, ГАС включають до складу системи залежно від типу ОД та обраних методів діагностування.

Різновидів систем тестового комбінованого діагностування існує багато.

СКД, створені в колишньому СРСР. Однією з перших була *система* тестового контролю і діагностування мікропроцесорних засобів обчислювальної техніки *на базі мікроЕОМ «Электроника МС 1212»*. При паралельному діагностуванні за допомогою засобів структурного діагностування вона вирішувала завдання «придатний-непридатний». Це пришвидшувало вихід придатних ПКПСІ. Відбраковані передавались для покомпонентного діагностування. Після виявлення несправних компонентів і ремонту плат їх знову повертали для повторного структурного діагностування.

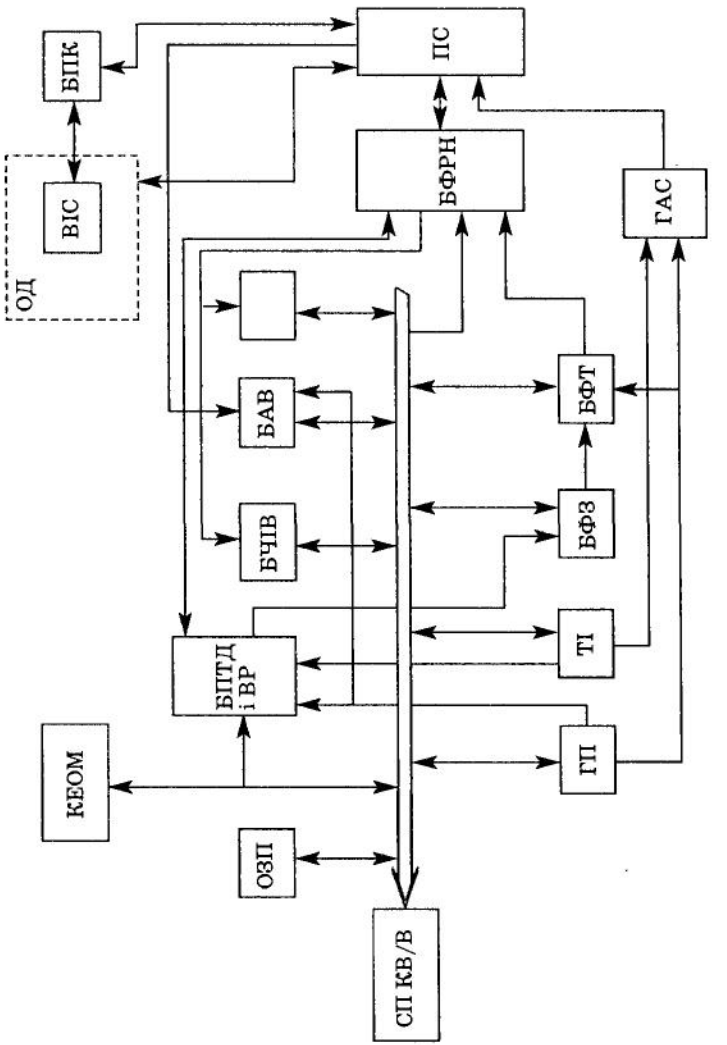


Рис. 4.4. Узагальнена структура СКД

СКД, що серійно випускали в СНД. Серед них, зокрема, виокремлюють такі:

автоматизована система контролю (АСК) типу «ИПЭ». Її використовували для контролю цифрових ПКПСІ, що містили інтегральні схеми ТТЛ- і МОН-структур. До системи підключали ОД за допомогою контактного пристрою типу поле з цвяхів або через крайові з'єднувачі. До її складу входили контактні пристрої типу кліпса і одноконтактний щуп;

система комбінованого діагностування «Европа-6». Вона мала можливість підключення ОД через адаптер типу поле з цвяхів. До неї входили сигнатурний аналізатор і пробник, а також 176 каналів для спрягання з ОД через крайові з'єднувачі та більш ніж 2000 — через адаптер типу поле з цвяхів. Частота подавання сигналів тестових дій на ОД — приблизно 2 МГц.

Зарубіжні СКД. До них належать такі системи:

тестер Z3200 компанії «Zehntel Inc.». Це одна з раніше розроблених СКД для тестування КПСІ. У ній реалізовано метод емуляції шинних циклів, що дає змогу користуватися власними тестовими програмами. Тестер підключають до КПСІ за допомогою пристрою типу кліпса. В більшості випадків він підтримує на лінії скидання компонента фіксований потенціал;

тестер виробничих випробувань S1650 компанії «Schlumberger». Це продуктивніша СКД. Система функціює з похибкою 800 нс і призначена для багатофункційного контролю цифрових і аналого-цифрових КПСІ, а також НВІС пам'яті. Швидкість обміну тестовими даними становить 20—40 МГц. Є можливість вимірювати струми, що споживаються, в діапазоні від 800 пА до 1 А за наявності перехідних процесів з викидами.

Одними з перших зарубіжних СКД ПКПСІ були тестери L200 фірми «Teradyne Inc.» і HP3065AT фірми «Hewlett Packard»:

тестер L200. Подає тест-вектори на ОД з частотою до 10 МГц. Тестування може здійснюватись як через крайові з'єднувачі, так і через контактний пристрій типу поле з цвяхів з 2304-ма контактами. Система працює під керуванням ЕОМ PDP 11/44. Випускалась модифікація цієї системи, орієнтована на діагностування ПКПСІ з НВІС;

комбінований тестер HP3065AT. Він складається з контролера 3065C, пристрою тестування цифрових ПКПСІ 3065D і аналого-цифрових схем 3065H. Системне програмне забезпечення розроблено на базі мови Бейсік та містить екранний символьний редактор і генератор програм діагностування.

СКД серії 900 фірми «Wayne Kerr». Вона має модульну структуру. Побудована на базі двох процесорів і забезпечує

діагностування складних ПКПСІ з шинною організацією структури. Система містить спеціальний модуль, що забезпечує прискорене тестування в режимі покомпонентних перевірок КПСІ;

експертна СКД PRO-1990 фірми «Protech». Вона ефективна в умовах серійного виробництва ПКПСІ. Її використовують на ремонтних станціях у період їх експлуатації. Система має спеціальний генератор опису структури ОД з його топології;

тестер схемних плат S790 компанії «Schlumberger». Це СКД, що реалізує методи штучного інтелекту для діагностування ПКПСІ. В системі використовують удосконалені програмні засоби і АРМ для генерації тест-програм, моделювання і локалізації помилок, налагодження та збирання даних для тестування;

комбінований тестер друкованих плат GR2288 фірми «Genrad». Нараховує 1152 канали підключення ОД. Кожен з них комплектують окремим запам'ятовуючим пристроєм. Канал виконує такі діагностичні процедури: перевірку та спрягання вузлів; покомпонентне і структурне тестування; контроль цифрових, аналогових і гібридних КПСІ, чипів пам'яті, інтерфейсів.

Кожна зі згаданих систем і тестерів орієнтована на певні класи ОД і не є абсолютно універсальною. Ця тенденція зберігається і сьогодні.

Засоби тестування мікропроцесорів

Одним з перших методів діагностування Ц і МПП є, зокрема, самотестування процесорів ПК.

Самотестування процесорів ПК. Вбудований механізм самотестування процесора BIST (Built in Self-Test — програма самоперевірки процесора при вмиканні живлення) забезпечує постійний контроль збоїв у мікрокомандах і великих логічних матрицях, а також тестування кеша команд, даних, буферів трансляції TLB і сегментів пам'яті ROM (Read Only Memory — постійний запам'ятовуючий пристрій (ПЗП)).

Після закінчення сигналу RESET процесор починає виконувати внутрішній тест BIST. Тестується більша частина процесора. Тест відпрацьовується за кілька десятків секунд. Закінчивши самотестування, процесор починає роботу як після звичайного скидання. В цей час у регістрах EAX записується сигнатура результату тестування. Її нульове значення свідчить про успішне виконання тесту.

Скидання переводить процесор у реальний режим і встановлює регістри у відповідні стани. Зокрема: FLAGS=0002h, його біти VM і RF (розряди розширення) встановлюються в нуль; у регістрі CR0 онуляються біти PG, TS, EM, MP і PE; регістр CS=F000h; регістр EIP=0000FFF0h. Уміст регістрів DS, ES, SS, FS, GS дорівнює 0000h; регістр DX містить інформацію про тип процесора.

Апаратне скидання онульовує рядки кеш-пам'яті, буферів трансляції і таблиць переходів (BTB), встановлює регістри FPU і MMX у певний стан. У вищеописаному стані CS:IP. Перебуваючи в реальному режимі, процесор починає виконувати інструкцію, що зчитується за фізичною адресою FFFFFFF0h. Це триває до першого перезавантаження регістра CS при виконанні команди міжсегментного переходу чи виклику або під час обслуговування переривання чи виключення. Ця програма повинна забезпечити ініціалізацію регістрів процесора і структур даних у пам'яті.

Після скидання до першої команди міжсегментного переходу або виклику на шині адреси в реальному режимі біти A20—A31 у циклах вибірки команд мають одиничне значення. Так, після сигналу RESET комп'ютер має образ BIOS (Basic Input-Output Sustem — базова система введення-виведення). Це програма, що забезпечує передавання інформації з комп'ютера на периферійні пристрої, і навпаки. Вона наявна в адресах FFFFFFF0-FFFFFFFFh. У I8086 ROM BIOS розташовувалась під границею 1-го Мбайта, в I80286 — 16-го Мбайта. Видалити BIOS з 1-го Мбайта пам'яті в режимі нормальної роботи неможливо, оскільки вектори переривань, що посилаються на сервіси BIOS, у реальному режимі можуть адресуватися тільки до пам'яті в діапазоні адрес 0-0FFFFFh.

Процесор Pentium і старші його моделі мають додатковий вхід INIT. Дії цього сигналу порівняно з апаратним скиданням мають такі особливості:

— тест BIST не запускається;

— внутрішня кеш-пам'ять не очищається, але TLB і BTB скидаються;

— значення регістрів MSR, включаючи регістр MTRR, не змінюється;

— стан FPU не змінюється.

Цей сигнал можна використати для переводу процесора в реальний режим зі збереженням даних у кеші.

Тестування апаратних засобів процесора можливе не тільки за допомогою тесту BIST. Починаючи з I80386, у процесорах стали використовувати тестові регістри TRn, за допомогою яких виконують тестування буферів, а в i486 і внутрішній кеш. У процесорах Pentium і старших моделях процесорів фірми «Intel» тестові регістри значно змінені щодо номенклатури і включені до складу регістрів MSR.

У процесорах Pentium уперше застосовано вбудовані пристрої контролю функціювання апаратних засобів під час виконання основних його функцій. За виявлення апаратних несправностей процесори подають команду #MC (Machine Check Exception — виключення машинного контролю). Дані про помилку записують і зберігають у спеціальних регістрах MSR. За їх вмістом дають повідомлення операційній системі і визначають, чи можливий рестарт команди, під час якої була виявлена помилка.

Апаратний контроль процесорів Pentium обмежується перевіркою паритету (лат. paritas — рівність) шини даних при операціях зчитування і правильності завершення шинних циклів обміну. Для обслуговування #MC призначені регістри P5 MC TYPE (тип помилки) та P5 MC ADDR (адреса).

У процесорах P6 застосовують архітектуру MCA. Порівняно з P5 її контроль розширено. Контролюється паритет на шині адреси, помилки ECC-контролю, кеш-пам'яті та буферів TLB.

Склад регістрів розширений глобальними регістрами опису можливостей контролю MCG CAP, стану MCG STATUS і керування MCG CTL, а також банком регістрів повідомлень про помилки. Кожному апаратному вузлові, що тестується, відповідає набір регістрів стану MC і STATUS, адреси MC і ADDR, керування MC і CTL та змішаного призначення MC і MISC. Наявність засобів апаратного контролю визначають за прапорцем MCE, а для розширеної архітектури — за прапорцем MCA в переліку властивостей, що повідомляються за інструкцією CPUID. Загалом особливості тестування процесорів ПК залежать від їх типу і моделі.

Тестування співпроцесорів. Процесори Pentium перевіряють щодо помилок наступним шляхом. Виконують тест за допомогою операції ділення і перевіряють правильність одержаного результату. Для того щоб пересвідчитися, що тестується співпроцесор, використовують будь-яку спеціальну команду чи встановлюють програму, необхідну для цього додатку. Такий підхід забезпечить коректну пе-

ревірку операції ділення незалежно від справності основного процесора.

У процесорах Pentium найсерйозніші помилки трапляються в третьому і п'ятому значущих розрядах результату ділення чисел з плаваючою крапкою. Перевірити, як виконує операції ділення з плаваючою крапкою процесор Pentium можна без розрахунків, скориставшись програмою CPUID фірми «Intel». Вона ідентифікує і тип процесора. Програма повідомляє про наявність помилки при операції ділення з плаваючою крапкою, якщо процесор має дефект.

Зовнішні засоби тестування компонентів ПК. Порівняно простими ручними апаратними засобами контролю компонентів комп'ютера є генератори поодиноких імпульсів, тест-роз'єми і тестери мережної розетки.

Генератор поодиноких імпульсів. Прилад використовують у комплексі з логічним пробником. Він призначений для подавання до схеми імпульсів високого рівня певної тривалості. Ці реакції порівнюють з еталонними.

Тест-роз'єми. Їх використовують для контролю працездатності послідовних і паралельних портів. Це прилади для 9-ти і 25-ти контактних послідовних портів і для 25-ти контактного паралельного порту. Такі роз'єми випускає фірма «Intel» (зокрема, універсальний роз'єм, що поєднує всі три різновиди в одному корпусі. Він найзручніший і входить до більшості діагностичних і ремонтних наборів) та багато інших фірм. Якщо тест-роз'єм установити замість з'єднувальних кабелів, то при перевірці працездатності сигнали з вихідних контактів порту будуть подаватися на вхідні контакти, тобто самі на себе.

Тестер мережної розетки. Призначений для перевірки електричних розеток. З його допомогою це роблять дуже швидко. Тестер вставляють у розетку і за світінням трьох світлодіодів визначають правильність підключення електричних ліній. Найчастіше трапляються неправильно підключені заземлювальні провідники. Мережні перешкоди в незаземленій системі потрапляють на спільний дріт комп'ютера, щодо якого відраховують логічні сигнали. Це породжує помилки при передаванні даних, періодичні збої. Незалежне заземлення комп'ютера має свої вади, передусім імовірність електричних розрядів, коли торкатися його корпусу.

Тестери модулів пам'яті. Для контролю стану *модулів пам'яті SIMM* (Single In-Line Memory Modyle — модуль пам'яті з однорядним розташуванням штирків (ОЗП)),

зокрема індивідуальних модулів кеш-пам'яті, використовують тестери. Тестують модулі за допомогою програм діагностування, хоча вони записують тільки коди в пам'ять, зчитують і перевіряють їх. Крім вищезазначених, тестер SIMM виконує такі операції:

— визначає тип і швидкодію модуля пам'яті;

— змінює час регенерації і швидкодію доступу до оперативної пам'яті;

— визначає, чи використовує модуль пам'яті емуляцію контролю парності, чи має свій контроль парності;

— виявляє збої в окремому біті;

— знаходить помилки зберігання даних;

— визначає час збою;

— визначає потужність шуму, пов'язаного зі збоєм;

— знаходить неспаяні контакти і короткі замикання.

Тестування модуля пам'яті комп'ютера за допомогою тестера модулів пам'яті SIMM є найефективнішим способом.

Контроль стану *модулів пам'яті SD RAM* (Synchronous DRAM — надшвидкодіюча синхронна динамічна пам'ять) проводять за допомогою тестерів, до яких належить і тестова система TURBO CATSTMII. Її з'єднують із підсистемою на базі процесора P6. Це дає змогу використовувати TURBO CATSMII для контролю роботи тестової системи, системи калібрування, графічного інтерфейсу користувача, роботи з базами даних, створення звітів та ін. Цю систему створювали як комплексне рішення для тестування SD RAM через системну шину з частотою від 66 до 133 МГц. Реальна частота тестування при цьому відповідала реальній швидкості системної шини. TURBO CATSMII підтримує до 2 Гбайт адресного простору. Щоб знаходити помилки оперативної пам'яті, існує більше 38 тестів. Вони виявляють помилки з майже стовідсотковою ймовірністю.

Тестер працює з шиною пам'яті RAMBUS: 300 MHz & & 400 MHz configurable clock rates; 600 MHz & 800 MHz RIMM module testing. Тестову систему комплектують операційними системами Windows 95/98 чи Windows NT.

Апаратні засоби тестування мікропроцесорів

Починаючи з окремих моделей i486, до складу процесорів включено підтримку тестового інтерфейсу JTAG.

Інтерфейс JTAG. Щоб його використовувати, необхідні зовнішні засоби діагностування, зокрема комп'ютер.

Для підключення через JTAG (Boundary Scan Architecture — інтерфейс для тестування складних логічних схем (процесорів) використовують перехідні колодки між сокетом і процесором. Сигнали тестового інтерфейсу не пов'язані з основним (системним) інтерфейсом процесора. За допомогою JTAG можна запускати тест BIST, а також подавати сигнали тестових дій і знімати відповідні реакції, згідно з програмою тестування процесора.

Тестовий інтерфейс JTAG є комбінацією зовнішніх і внутрішніх засобів діагностування процесорів. Інтерфейс JTAG (IEEE 1149.1) має всього чотири сигнали:

1) TMS (Test Mode Select — сигнал вибору тестового режиму);

2) TDI (Test Data Input — вхідні дані в послідовному двійковому коді);

3) TDO (Test Data Output — вихідні дані в послідовному двійковому коді);

4) TCK (Test Clock — сигнал синхронізації послідовних даних).

Частота синхронізації може досягати 16 МГц. Сигнали інтерфейсу JTAG подають на тестовий порт TAP, через який обладнання, що мають тестувати, підключають до обладнання, за допомогою якого його будуть тестувати. Основні завдання обладнання, яке тестує, — формування тестових сигналів згідно з програмою тестування пристрою, який діагностують, і порівняння одержаних результатів з еталонними. Один і той самий порт можна використовувати для тестування кількох пристроїв, що підтримують JTAG. Стандартизований логічний формат дає змогу контролеру JTAG окремо звертатися до кожного зі з'єднаних у ланцюжок тестованих пристроїв. При цьому головна вимога — справність комірок JTAG у тестованих пристроях.

На рис. 4.5 показана умовна цифрова схема, що має вхідні, вихідні й двонапрямлені сигнальні лінії, в тому числі з високоімпедансним (лат. impedio — перешкоджаю) станом. Комірка тестування B/S розташована між зовнішніми виводами процесора і самим логічним пристроєм, тобто на логічній границі пристрою. Контролер тестового порту (TAP-контролер) сканує комірки — керує ними і зчитує з них інформацію. При включеному тестовому режимі TAP-контролер логічно від'єднує сигнальні лінії від зовнішніх виводів, задає вхідні впливи і зчитує результати. Це всі операції, необхідні для тестування послідовних схем. Такий підхід до тестування пристроїв комп'ютера є, по суті, тестовим комбінованим діагностуванням цифрових структур.

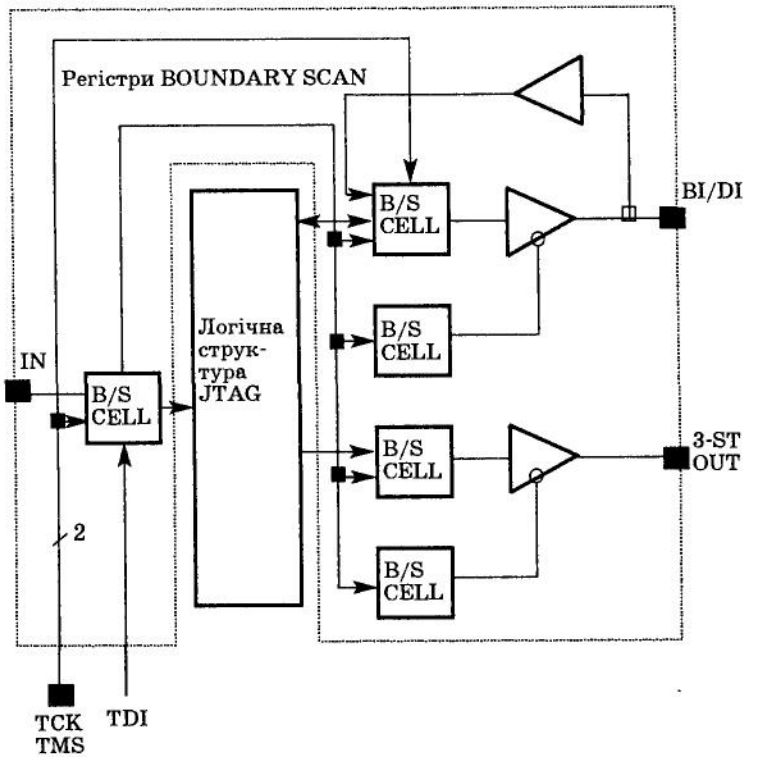


Рис. 4.5. Схема включення комірок сканування

Підтримка інтерфейсу JTAG. У пристрій, який підтримує JTAG, вбудована тестова логіка. Вона складається з таких елементів (рис. 4.6):

— тестового порту TAP, що має чотири сигнальні лінії;

— TAP-контролера, що керує тестовими регістрами;

— регістра інструкцій IR, який приймає послідовний код із входу TDI. Код інструкції використовують для вибору тестової операції, що виконується, чи регістра тестових даних, до яких робиться звертання;

— регістрів тестових даних BPR, BSR і DID у процесорах i486 та додаткових регістрів для зондового режиму в

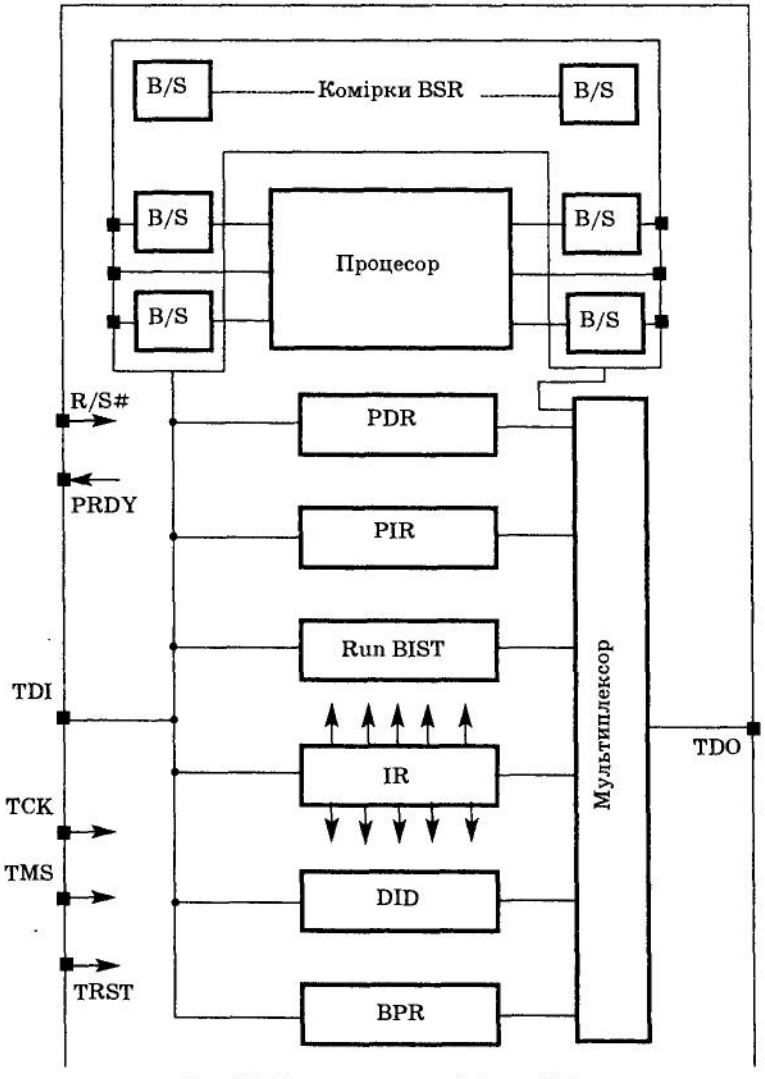


Рис. 4.6. Процесор з інтерфейсом JTAG

процесорах Pentium і старших моделях процесорів фірми «Intel».

Регістри інструкцій і даних є незалежними зсуваючими регістрами, з'єднаними паралельно. На їхні входи (старші розряди) поступає сигнал TDI, а з виходів (молодші розряди) знімають сигнали TDO. При кожному позитивному перепаді дані зсуваються на один біт.

Регістр BPR. Має довжину в один розряд. Його використовують як найкоротший обхідний шлях для послідовних даних, коли останні регістри не беруть участі в обміні.

Регістр BSR. Це «довгий» зсуваючий регістр. Кожний його біт — прикордонні комірки, встановлені на всіх вхідних і вихідних сигнальних лініях процесора. Для двонапрямлених ліній, крім інформаційних комірок регістра, що відповідають зовнішнім сигналам, є і керуючі комірки, що задають напрям передавання інформації. Довжина BSR у процесорів i486 — 104 розряди, в P5 — 175 розрядів, у P6 — 159.

Регістр DID (32-розрядний). Містить ідентифікатор виробника, код пристрою і номер версії. За ним TAP-контролер може розпізнати, з яким пристроєм він працює.

Регістр IR. Призначений для зберігання виконуваної тестової інструкції. Його довжина становить у i486 — 4 розряди, в Pentium — 13 розрядів, у P6 — 6.

Для всіх процесорів обов'язковими є інструкції EXTEST, SAMPLE/PRELOAD, IDCODE і BYPASS. Крім перерахованих інструкцій, процесори виконують інструкцію RUNBIST, а також низку інших інструкцій, характерних для кожної окремої моделі. Інструкції, непризначені для даної моделі процесора, на нього не діють, а на виході TDO встановлюватимуться нулі.

Інструкція EXTEST (молодші біти коду — 00). Перевіряє зовнішні ланцюжки. Вихідні сигнали подають на вихідні виводи. Попередньо їх записують у регістр BSR. Стан вхідних сигналів фіксують у цих самих регістрах. Сигнали, що передають у двох напрямках однією лінією, заздалегідь конфігурують керуючими бітами комірок BSR. Виконавши дану інструкцію, процесор може перейти в нормальний режим роботи після сигналу RESET.

Інструкція SAMPLE/PRELOAD (молодші біти коду — 01). Вона має два призначення. Дає змогу миттєво запам'ятати стан усіх зовнішніх сигналів без впливу на роботу пристрою, коли TAP-контролер знаходиться в стані Capture-DR. Значення сигналів фіксують за позитивним перепадом TCK. За цією інструкцією в стані Update-DR дані завантажують (скидом сигналу TCK) у вихідні комірки тестового порту (але не на виходи пристрою), звідки згодом виводитимуться на виводи процесора за інструкцією EXTEST.

Інструкція ідентифікації IDCODE (молодші біти коду — 10). Підключає до інтерфейсу регістр DID. Вона дає змогу зчитати його вміст.

Інструкція BYPASS (усі біти коду одиничні). Призначена для підключення однобітного обхідного регістра. Вхід TDI установлений резистором до високого рівня. Розрив ланцюжка JTAG зумовлює підключення обхідних регістрів у всіх пристроях. Це виключає непередбачувані дії пристроїв у разі обриву.

Інструкція RUNBIST (код залежить від моделі процесора). Підключає однойменний однобітний регістр, записує в нього одиниці й запускає тест BIST. Після цього TAP-контролер переводиться у стан Run-Test/Idle і очікує завершення тесту. З регістра RUNBIST може бути зчитаний код результату виконання тесту. Його нульове значення свідчить про те, що тест завершився успішно.

TAP-контролер є синхронним кінцевим автоматом, що змінює свій сигнал по фронту сигналу TCK і при вмиканні живлення. Зміною стану керує сигнал TMS, що сприймається за позитивним перепадом сигналу TCK. На рис. 4.7 зображено граф станів і переходів керуючого автомата. Біля стрілок переходів вказано значення сигналу TMS під час фронту TCK.

Стан Test-Logic-Reset. У цей вхідний стан контролер переходить автоматично при вмиканні живлення і може бути переведений з будь-якого стану високим рівнем сигналу TMS за п'ять тактів TCK. У цьому стані тестова логіка заборонена, і процесор працює в нормальному режимі.

Стан Run-Test/Idle. Він є проміжним між виконанням тестових операцій. У ньому регістри не змінюють своїх значень.

Стан Capture-DR. У ньому скануючий регістр фіксує тільки дані на вхідних лініях під час виконання інструкцій EXTEST і SAMPLE/PRELOAD.

Стан Shift-DR. Тут дані з входу TDI рухаються на вихід TDO через підключений зсуваючий регістр.

Стан Pause-DR. Контролер тимчасово забороняє просування даних через зсуваючий регістр.

Стан Update-DR. За спадом сигналу TCK сигнали із зсуваючого регістра фіксують на виходах тестових комірок.

Стан Capture-IR. У ньому контролер завантажує у зсуваючий регістр інструкцій код інструкції SAMPLE.

Стан Shift-IR. У ланцюжок між TDI і TDO вмикають зсуваючий регістр інструкцій, але попередню інструкцію ще виконують.

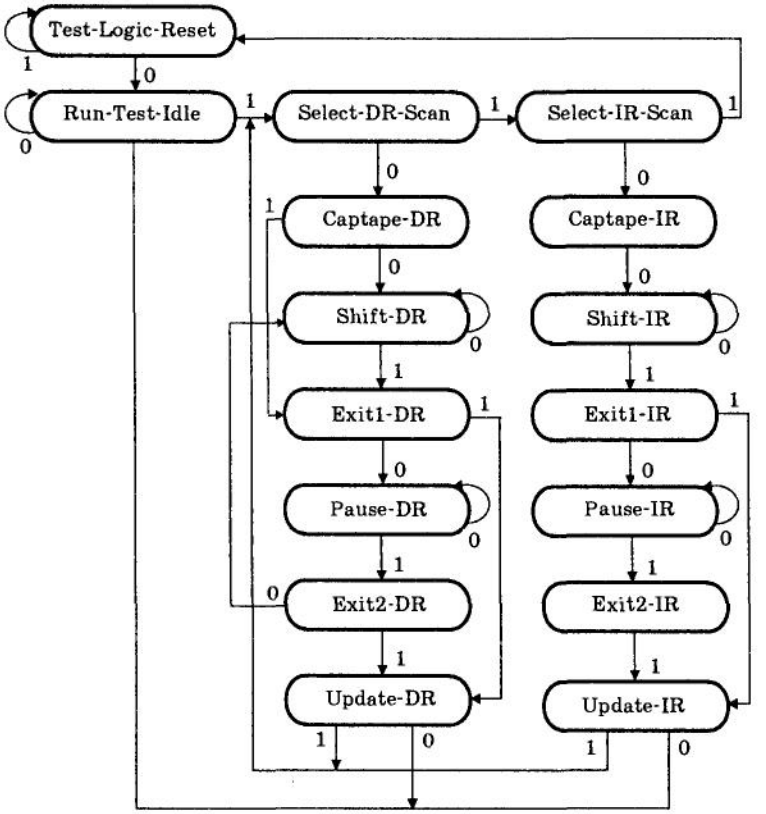


Рис. 4.7. Граф станів і переходів контролера TAP

Стан Pause-IR. У цьому стані контролер тимчасово забороняє просування даних через зсуваючий регістр інструкцій.

Стан Update-IR. За спадом сигналу TCK фіксують нову виконувану інструкцію і в ланцюжок TDI-TDO вмикають відповідний регістр.

Крім основних станів контролера, що визначають дії тестового обладнання, є тимчасові проміжні стани. Вони необхідні для реалізації переходів автомата. До них належать *Select-DR-Scan, Exit1-DR, Exit2-DR, Select-IR-Scan, Exit1-IR, Exit2-IR.*

Для інтерфейсу JTAG існує спеціальна мова опису пристроїв BSDL (Boundary Scan Description Language —

мова опису пристроїв для діагностування за допомогою інтерфейсу JTAG). Склад і послідовність проходження інформаційних і керуючих комірок у зсувному регістрі даних специфічний для кожного пристрою.

У порту TAP процесорів P5 і P6 є додаткові регістри інструкцій і сигнальні лінії, за допомогою яких реалізують зондовий режим налагодження. У ньому зчитують регістри процесора. Це дає змогу використати налагоджувальні засоби, що не залежать від робочих програм, які виконує процесор.

Індикатор D-LED™. Будь-який несправний пристрій ПК чи дефект їх монтажу може спричинити відмову всієї системи. Оперативне одержання діагностичної інформації про стан системи дало б змогу швидко і ефективно приймати рішення щодо ідентифікації несправностей та способів їх усунення. Тому для користувачів і налагоджувачів комп'ютерів потрібен інструмент налагодження, здатний виявляти несправності системи і повідомляти про її стан. Ним є індикатор D-LED™ (Dual-Light Emitting Diode — світлодіодний індикатор для визначення несправностей комп'ютера, розташований на системній платі).

Він може діагностувати і простежувати процедуру завантаження, перевіряти стан системи, включаючи її основні компоненти. Роботу системи відображає комбінація сигналів на 4-х світлодіодних індикаторах. Це забезпечує 16 комбінацій сигналів, що відображають стан системи. Завдяки цьому пристрій вказує на несправності системи під час її зупинки.

Кожний світлодіодний індикатор має два різних режими світіння — червоний і зелений. Порівнюючи комбінацію сигналів, що відображає стан системи на певний момент, з комбінацією кодів у табл. 4.2 («0» — червоний, «1» — зелений), можна легко ідентифікувати стан системи, а якщо він відповідає певній несправності, то визначити її тип і місце прояву.

Отже, за допомогою індикатора D-LED™ користувачі мають змогу швидко визначити несправності комп'ютера і усунути їх. Крім того, що пристрій дає можливість зекономити час для усунення відмов, він вирішує ще багато проблем, пов'язаних з відмовами апаратури, її неправильним встановленням та налагодженням, визначенням конфігурації обладнання безпосередньо на місці його встановлення.

Таблиця 4.2

Кодові комбінації сигналів індикатора D-LED™

№ п/п	Кодова комбінація сигналів, що відповідає стану плати				Суть несправностей
	R5	R4	R3	R1	
1	0	0	0	0	System Power ON — системи живлення підключено
2	0	0	0	1	Early Chipset Initialization — передчасна ініціалізація чіпсета
3	0	0	1	0	Memory Detection Test — тест виявлення пам'яті
4	0	0	1	1	Decompressing BIOS image to RAM — скидання відображення BIOS в оперативній пам'яті
5	0	1	0	0	Initializing Keyboard Controller — ініціалізація контролера клавіатури
6	0	1	0	1	Testing VGA (Video Graphics Array — графічний адаптер) BIOS — тестування VGA BIOS
7	0	1	1	0	Processor Initialization — ініціалізація процесора
8	0	1	1	1	Testing RTC (Real Time Clock — годинник реального часу) — тестування RTC
9	1	0	0	0	Initializing Video Interface — ініціалізація відеоінтерфейсу
10	1	0	0	1	BIOS Sign On — ознака роботи BIOS наявна
11	1	0	1	0	Testing Base and Extended Memory — тестування базової і розширеної пам'яті
12	1	0	1	1	Assign Resource to all ISA — призначення ресурсів для всіх ISA
13	1	1	0	0	Initializing Hard Drive Controller — ініціалізація контролера жорсткого диска
14	1	1	0	1	Initializing Floppy Drive Controller — ініціалізація контролера гнучкого диска
15	1	1	1	0	Boot Attempt — спроба початкового завантаження
16	1	1	1	1	Operating System Booting — операційна система завантаження

Ефективність систем діагностування

З огляду на складність засобів тестового діагностування сучасних обчислювальних приладів існують різноманітні підходи до оцінювання їх ефективності. Усі вони певною мірою дають можливість з тою чи іншою достовірністю оцінити ефективність досліджуваних систем.

Ефективність *(лат. effectivus — дійовий)* ***системи діагностування*** *— ступінь її пристосованості до процесу визначення технічного стану ОД і пошуку в ньому дефектів.*

Оцінювання ефективності автоматизованих систем діагностування проводять за показниками $\{D, \tau_D, C_D, L\}$, де D — ймовірність правильного діагностування; τ_D — середня оперативна тривалість діагностування; C_D — середня вартість діагностування; L — глибина пошуку дефектів.

Також оцінювання ефективності систем визначають з відносного приросту ефективності через *показник надійності*. Проте його визначення складне, а часом і неможливе, оскільки при цьому необхідно враховувати додатково багато факторів, що характеризують ОД, систему діагностування та їх взаємодію.

Для порівняльного аналізу систем діагностування запроваджено поняття *«питомі витрати»* — відношення об'єму буферної пам'яті до швидкості передавання тест-векторів. За методикою порівняльного оцінювання ефективності при розрахунках беруть до уваги тільки два параметри системи, хоча її характеризує і багато інших.

У згаданих методиках ефективність систем діагностування постає як функція параметрів системи діагностування і ОД. Проте, з одного боку, не завжди можливо точно визначити параметри ОД (наприклад, за відсутності відомостей про них), а з іншого — порівняльне оцінювання систем значно спростилося б, коли при їх розрахунках не треба було знати точні значення параметрів кожної з порівнюваних систем.

За такого підходу до уваги беруть тільки порівнювальне значення параметрів, тобто більше чи менше значення кожного з них щодо параметрів інших систем, які порівнюють.

Для порівняльного оцінювання систем діагностування існує поняття *«пріоритетна функція»* $P = \{X_1, X_2, \ldots, X_k\}$, яку визначають як функцію скінченного числа k параметрів, за якими порівнюють системи діагностування. Їх поділяють на три групи, за умови, що ОД і діагностовані системи, які порівнюють, мають однакову складність або ідентичні.

1. Параметри експлуатаційної ефективності систем (з вагою p_1): L — глибина пошуку дефектів; τ — час діагностування; ν — частота подавання тестових дій і прийому відповідних реакцій; N — кількість каналів зв'язку з ОД; l — глибина генератора тестових дій; n — кількість рівнів тестових дій, що генеруються.

2. Конструктивні параметри систем (з вагою $p_2 < p_1$): W — споживана потужність; G — габарити (обсяг) апаратури; m — маса.

3. Вартістні параметри систем (з вагою $p_2 < p_3 < p_1$): C — ціна системи діагностування.

Для введених змінних записують пріоритетну функцію:

$$P = P[(L,\tau,\nu,N,l,n)_{P_1}, (W,G,m)_{P_2}, C_{P_3}] \qquad (4.1)$$

Вид функції (4.1) невідомий, але при обчисленні її значення для різних систем діагностування використовують теоретико-множинний підхід. Порівнявши скінченне число q систем зі скінченним числом ознак (параметрів), за результатом порівняння складають пріоритетну таблицю. В її першому рядку записують множину $V = \{V_1, V_2, \ldots, V_k\}$ ознак, у всіх наступних — множини Q_i систем діагностування, яким віддають перевагу (пріоритет) за цими ознаками. Чим вище рядок розташований в таблиці, тим більшу вагу він має. Якщо за певною ознакою V_i системи, що порівнюють, однакові, то на місці перетину стовпця і рядка таблиці вказують індекси обох систем. Наприклад, порівнюють три системи діагностування, які позначено через a, b, c. Щоб обчислити пріоритет кожної з них, необхідно побудувати таблицю пріоритетів (табл. 4.3). З неї можна обчислити значення пріоритетів для кожної із систем a, b, c.

Отже, для системи a:

$$P(a) = 4p_1p_4 + p_2p_4 + p_3p_4 + p_1p_5 + p_1p_6 + 2p_2p_6 = \\ = p_4(4p_1 + p_2 + p_3) + p_1p_5 + p_6(p_1 + 2p_2). \qquad (4.2)$$

Для системи b:

$$P_b = p_4(p_1 + p_2) + p_5(3p_1 + 2p_2) + p_6(2p_1 + p_3). \qquad (4.3)$$

Для системи c:

$$P_c = p_4(p_1 + p_2) + p_5(2p_1 + p_2 + p_3) + p_6(3p_1 + p_2). \qquad (4.4)$$

Таблиця 4.3

Пріоритетні параметри систем діагностування

	p_1						p_2			p_3
	L	τ	v	N	I	n	W	G	m	C
p_4	a	a	a	c	b	a	c	a	b	a
p_5	c	b	b	a	c	b	b	b	c	c
p_6	b	c	c	b	a	c	a	c	a	b

Якщо прийняти, що $p_1 + p_2 + p_3 = 1$ і $p_4 + p_5 + p_6 = 1$ і присвоїти $p_1 = 0,5$; $p_2 = 0,2$; $p_3 = 0,3$; $p_4 = 0,5$; $p_5 = 0,3$; $p_6 = 0,2$, то отримують $P_a = 1,58$; $P_b = 1,18$; $P_c = 1,04$.

Щоб точніше обчислювати пріоритетні функції систем, що порівнюють, задають конкретну вагу кожному з параметрів груп p_1 і p_2.

Методика порівняльної оцінки дає змогу спрощувати оцінювання як систем, які розробляють, так і систем, які вже використовують для контролю і діагностування обчислювальних пристроїв.

4.2. Програмні засоби діагностування ПК

У наш час широкого поширення набули програмні засоби діагностування ПК, особливо на етапі експлуатації. Тут вони переважають за причини їх спрощеного застосування. Діагностичні програми, якими є програмні засоби, відпрацьовуються засобами самого ПК і в більшості випадків не потребують додаткового діагностичного обладнання. Це одна з головних переваг програмних засобів діагностування ПК перед апаратними.

Класифікація діагностичних програм

Для IBM-сумісних комп'ютерів існують діагностичні програми, що дають змогу користувачеві виявляти несправності в комп'ютері. Вони локалізують несправність до функційного вузла. Умовно їх можна поділити на п'ять видів, які подані нижче за рівнем складності.

1. POST (Power-On Self Test — самотестування комп'ютера при вмиканні живлення). Програма виконується під час кожного вмикання комп'ютера.

2. Діагностичні програми фірм-виробників. «IBM», «Compag», «Hewlett-Pachard», «Dell», «Genrad» та інші випускають для своїх систем спеціалізоване діагностичне програмне забезпечення, яке складається з наборів тестів для контролю працездатності компонентів комп'ютера.

3. Діагностичні програми обладнання комп'ютерів і комп'ютерних систем фірм-виробників. Такі фірми розробляють діагностичні програми для контролю і діагностування певного пристрою.

4. Діагностичні програми операційних систем (ОС). Операційні системи Windows 9x і Windows NT розробляють з кількома діагностичними програмами для перевірки різних компонентів комп'ютера.

5. Діагностичні програми загального призначення. Забезпечують належне тестування будь-яких IBM-сумісних комп'ютерів.

Класифікація діагностичних програм наведена за ознаками їх функційного призначення. Вона є найпоширенішою. Однак можливі класифікації й за іншими ознаками.

Суть і особливості діагностичних програм

Щоб забезпечити ефективне використання діагностичних програм, недостатньо знати їх класифікацію, необхідно визначити суть і особливості діагностичного програмного забезпечення.

1. Самоперевірка при вмиканні комп'ютера (POST). Ця процедура є послідовністю підпрограм, яка зберігається на материнській платі в ROM BIOS. Вони призначені для діагностування основних компонентів системи відразу після її включення. Під час цієї процедури завантаження операційної системи затримується.

При включенні комп'ютера автоматично тестується центральний процесор, ПЗП (програмований запам'ятовуючий пристрій), допоміжні елементи материнської плати, оперативної пам'яті і периферійних пристроїв. Ці тести виконуються дуже швидко і не так високоякісно, як діагностичні програми виробників. Виявивши несправність компонента, видається попередження або повідомлення про помилку чи несправність.

Процедура POST передбачає такі способи індентифікації несправностей, як звукові сигнали; повідомлення, що

виводяться на екран монітора; шістнадцяткові коди помилок, які подаються в порт введення-виведення.

За серйозної несправності завантаження системи зупиняється і з'являється повідомлення про несправність, за яким визначають її причину. Такі несправності називають *фатальними помилками*.

— Звукові коди помилок, що видає процедура POST. Якщо процедура POST виявляє несправності, то комп'ютер подає характерні звукові сигнали, за якими визначають несправний пристрій або компонент (під час вмикання справного комп'ютера чути один короткий звуковий сигнал). Видається серія довгих чи коротких звукових сигналів або їх комбінація. Це залежить від фірми-розробника BIOS та її версії. Звукові коди несправностей IBM-сумісних комп'ютерів наведені в табл. 4.4 — 4.7.

Таблиця 4.4

IBM POST

№ п/п	Звуковий сигнал	Місце виникнеиия несправності
1	1 короткий	Система справна. Процедура POST завершена
2	2 коротких	Є несправність, код помилки виведений на екран
3	Немає сигналу	Несправний блок живлення або системна плата
4	Неперервний сигнал	
5	Короткі сигнали, що повторюються	
6	1 довгий, 1 короткий	Системна плата
7	1 довгий, 2 коротких	Адаптер дисплея (MDA, CGA)
8	1 довгий, 3 коротких	Розширений графічний адаптер (EGA)
9	3 довгих	Плата клавіатури 3270

Таблиця 4.5

Звукові сигнали процедури POST AMI BIOS

№ п/п	Звуковий сигнал	Фатальна помилка
1	2	3
1	1 короткий	Помилка регенерації динамічного ОЗП
2	2 коротких	Помилка схеми контролю парності

Продовження таблиці 4.5

1	2	3
3	3 коротких	Несправність у перших 64 Кбайт ОЗП
4	4 коротких	Несправність системного таймера
5	5 коротких	Помилка процесора
6	6 коротких	Помилка у схемі керування лінією A20 у контролері клавіатури
7	7 коротких	Помилка переключення у віртуальний режим
8	8 коротких	Помилка зчитування-запису відеопам'яті
9	9 коротких	Помилка контрольної суми ROM BIOS
10	10 коротких	Помилка зчитування-запису CMOS (Complimentary Metal Oxide Semiconductor — комплементарна структура метал-оксид-напівпровідник (КМОН))-пам'яті
11	11 коротких	Помилка кеш-пам'яті
12	Звуковий сигнал	Нефатальна помилка
13	1 довгий, 3 коротких	Помилка в основній або розширеній пам'яті
14	1 довгий, 8 коротких	Не виконується тест на відповідний сигнал дисплея

Таблиця 4.6

Критичні помилки, які визначають під час виконання процедури POST Phoenix BIOS

№ п/п	Звуковий код	Код порту 80h	Поясиення
1	2	3	4
1	Немає	01h	Виконується тестування регістрів CPU (Central Processor Unit — центральний процесор)
2	1—1—3	02h	Помилка зчитування або запису в CMOS-пам'ять
3	1—1—4	03h	Неправильна контрольна сума системної BIOS
4	1—2—1	04h	Несправність програмованого таймера інтервалів

Продовження таблиці 4.6

1	2	3	4
5	1—2—2	05h	Невдала спроба ініціалізації прямого доступу до пам'яті
6	1—2—3	06h	Помилка зчитування або запису в регістри сторінок прямого доступу до пам'яті
7	1—3—1	08h	Помилка під час перевірки схеми регенерації пам'яті
8	Немає	09h	Тестування перших 64 Кбайт пам'яті
9	1—3—3	0Ah	Несправність мікросхеми або лінії даних у перших 64 Кбайт пам'яті (декілька бітів)
10	1—3—4	0Bh	Логічна помилка парності-непарності в перших 64 Кбайт пам'яті
11	1—4—1	0Ch	Несправність лінії адреси в перших 64 Кбайт пам'яті
12	1—4—2	0Dh	Помилка контролю парності в перших 64 Кбайт пам'яті
13	2—1—1	10h	Помилка в біті 0 перших 64 Кбайт пам'яті
14	2—1—2	11h	Помилка в біті 1 перших 64 Кбайт пам'яті
15	2—1—3	12h	Помилка в біті 2 перших 64 Кбайт пам'яті
16	2—1—4	13h	Помилка в біті 3 перших 64 Кбайт пам'яті
17	2—2—1	14h	Помилка в біті 4 перших 64 Кбайт пам'яті
18	2—2—2	15h	Помилка в біті 5 перших 64 Кбайт пам'яті
19	2—2—3	16h	Помилка в біті 6 перших 64 Кбайт пам'яті
20	2—2—4	17h	Помилка в біті 7 перших 64 Кбайт пам'яті
21	2—3—1	18h	Помилка в біті 8 перших 64 Кбайт пам'яті
22	2—3—2	19h	Помилка в біті 9 перших 64 Кбайт пам'яті

Закінчення таблиці 4.6

1	2	3	4
23	2—3—3	1Ah	Помилка в біті 10 перших 64 Кбайт пам'яті
24	2—3—4	1Bh	Помилка в біті 11 перших 64 Кбайт пам'яті
25	2—4—1	1Ch	Помилка в біті 12 перших 64 Кбайт пам'яті
26	2—4—2	1Dh	Помилка в біті 13 перших 64 Кбайт пам'яті
27	2—4—3	1Eh	Помилка в біті 14 перших 64 Кбайт пам'яті
28	2—4—4	1Fh	Помилка в біті 15 перших 64 Кбайт пам'яті
29	3—1—1	20h	Помилка в регістрі прямого доступу до пам'яті
30	3—1—2	21h	Помилка у «ведучому» регістрі прямого доступу до пам'яті
31	3—1—3	22h	Помилка у «ведучому» регістрі маски переривань
32	3—1—4	23h	Помилка в регістрі маски переривань
33	Немає	25h	Завантаження векторів переривань
34	3—2—4	27h	Помилка при виконанні тесту контролера клавіатури
35	Немає	28h	Несправність живлення CMOS-пам'яті або виконується підрахунок контрольної суми CMOS-пам'яті
36	3—3—4	2Bh	Помилка при ініціалізації екрана
37	3—4—1	2Ch	Помилка під час перевірки повернутого сигналу дисплея
38	3—4—2	2Dh	Виконується пошук ПЗП відеоадаптера
39	Немає	2Eh	Забезпечує виведення на екран ПЗП відеоадаптера
40	Немає	30h	Відеосистема працездатна
41	Немає	31h	Монохромний монітор працездатний
42	Немає	32h	Кольоровий (на 40 стовпців) монітор працездатний
43	Немає	33h	Кольоровий (на 80 стовпців) монітор працездатний

Таблиця 4.7

Некритичні помилки, які визначають під час виконання процедури POST Phoenix BIOS

№ п/п	Звуковий код	Код порту 80h	Пояснення
1	4—2—1	34h	Перевіряється переривання синхроімпульсів таймера або виявлена несправність
2	4—2—2	35h	Перевіряється відключення або виявлена несправність
3	4—2—3	36h	Несправність схеми керування лінією A20
4	4—2—4	37h	Непередбачене переривання в захищеному режимі
5	4—3—1	38h	Виконується перевірка ОЗП або виявлена несправність за адресою, що перевищує FFFFh
6	4—3—3	3Ah	Перевіряється канал 2 таймера або виявлена несправність
7	4—3—4	3Bh	Перевіряються години поточного часу або виявлена несправність
8	4—4—1	3Ch	Перевіряються послідовні порти або виявлена несправність
9	4—4—2	3Dh	Перевіряються паралельні порти або виявлена несправність
10	4—4—3	3Eh	Перевіряється співпроцесор або виявлена несправність
11	Low* 1—1—2	41h	Помилка вибору системної плати
12	Low 1—1—2	42h	Несправність розширеної CMOS-пам'яті

* Low — звук нижчого тону; передує решті звуків.

За появи критичних помилок система зупиняє роботу і подальше виконання операцій стає неможливим. Наслідки некритичних помилок менш серйозні.

— Повідомлення про помилки, що видає на екран процедура POST. Процедура POST у комп'ютерах XT, PS/2 і IBM-сумісних комп'ютерах відображає на екрані хід тестування оперативної пам'яті. Останнє виведене число вказує на кількість пам'яті, що успішно пройшла перевірку. Наприклад, у сучасних комп'ютерах з'являється повідомлення: 32768 KB OK. У справному комп'ютері це число повинно збігатися з кількістю встановленої в ньому пам'яті.

Якщо під час тестування процедура POST виявила несправність, то на екран дисплея виводиться відповідне повідомлення, наприклад 1790-Disk 0 Error. Далі, скориставшись керівництвом з експлуатації, з'ясовують, яка несправність відповідає цьому кодові.

— Коди помилок, що видає процедура POST у порти введення-виведення. На початку виконання кожного тесту за адресою спеціального порту введення-виведення POST видає коди, що прочитуються тільки за допомогою спеціальної плати адаптера, яку встановлюють у роз'єм розширення. Раніше (90-ті роки) такі плати розробляли для тестування материнських плат. Тепер окремі фірми випускають їх для сервісного обслуговування.

Під час виконання процедури POST на вбудованому індикаторі POST-плати, встановленої в роз'єм розширення, швидко змінюються двозначні шістнадцяткові числа. Якщо числа не змінюються, то на індикаторі відображається код тесту, під час якого відбувся збій. Це дещо локалізує місце прояву несправності.

У більшості комп'ютерів із системними шинами ISA (Industry Standart Architecture — тип системної шини) і EISA (Extended ISA — розширена шина ISA) BIOS видає POST-коди в порт введення-виведення за адресою 80h.

Використовують здебільшого плати, які вставляють у 8-розрядні частини роз'ємів шин ISA і EISA, і плати, які підключають до шини MCA. Є універсальні пристрої, які за допомогою додаткового адаптера підключають ISA/EISA-плату до MCA-шини. У наш час більшість виробників випускають тестові плати тільки для шин PCI (Periphrial Component Interconnect bus — шина взаємодії периферійних пристроїв) та ISA.

2. Діагностичні програми фірм-виробників. Базові діагностичні програми для визначення правильності функціювання комп'ютерів випускають фірми виробники цих самих комп'ютерів. Це так зване спеціалізоване діагностичне програмне забезпечення, що складається з тестів, які дають змогу визначити працездатність комп'ютера.

3. Діагностичні програми обладнання комп'ютерів і комп'ютерних систем фірм-виробників. Багато видів діагностичного програмного забезпечення призначено для певних типів апаратних засобів. Такі програми надходять разом з комп'ютерними пристроями.

Більшість адаптерів інтерфейсу SCSI (Small Computer System Interface — системний інтерфейс малих комп'ютерів) мають вбудовану BIOS, за допомогою якої можна настроювати і діагностувати їх. Наприклад, SCSI-адаптери фірми «Adaptec» отримують разом із програмою SCSI SELECT, що дає змогу правильно сконфігурувати та протестувати прилад.

Деякі виробники мережних плат пропонують діагностичне програмне забезпечення, яке тестує інтерфейс шини, пам'ять, вектори переривань, а також виконує циклічний тест. Ці програми записують на гнучких або компакт-дисках і додають до пристрою. З їх допомогою можна звернутися на Web-вузол виробника.

4. Діагностичні програми операційних систем. Систему можна протестувати не тільки за допомогою спеціальних діагностичних програм, а й за допомогою засобів операційної системи. У Windows 9x і NT є кілька діагностичних програм.

— Microsoft Diagnostics (MSD). Ця програма діагностування є неповною, більше схожа на утиліту для конфігурації системи. MSD дає змогу швидко вирішити завдання щодо використання переривань і розподілу пам'яті.

Ця програма видає основну інформацію про версію BIOS, тип процесора, відеоадаптер, мережу (якщо вона є), мишу, дисковод, CD-ROM (Compact Disc-Read Only Memory — постійна пам'ять на компакт-дисках тільки для зчитування), паралельні і послідовні порти та версію DOS. Крім того, за допомогою MSD можна отримати інформацію про завантажені в пам'ять драйвери пристроїв та резидентні програми. Програма у графічній формі показує їх розташування в пам'яті більш наглядно, ніж текст.

MSD міститься на компакт-диску разом з Windows 9x. Щоб її запустити, треба перезавантажити комп'ютер з Windows у режим MS DOS.

— Диспетчер пристроїв Windows 9x. Він досконаліший, ніж програма MSD. У вкладці «Обладнання» («Device Manager») відображено встановлені у комп'ютері пристрої. Тут їх можна конфігурувати і переглядати їхні ресурси, обновлювати драйвери. Якщо встановити перемикач «Обладнання за підключенням» («Viev devices by connection»), то є можливість переглянути різні порти та інтерфейси комп'ютера.

На елементі «Комп'ютер» («Computer») є діалогове вікно, в якому переглядають розподіл адрес введення-виведення, переривань, прямого доступу до пам'яті.

Якщо встановити пристрої з технологією автоматичного налагодження Plug and Play у Windows 9x, то між ними виникають конфлікти. Їх усувають за допомогою диспетчера пристроїв.

— Індикатор ресурсів Windows 9x. Програма відображає системні ресурси, зокрема модулі USER. EXE і GDI. EXE. На початку її завантаження піктограма програми «Індикатор ресурсів» («Resource Meter») міститься у правій частині панелі завдань. Якщо ввійти до неї, то з'явиться діалогове вікно «Індикатор ресурсів». При запускові будь-якої програми або відкритті кількох вікон папок системні ресурси зменшуються, що відображається у вищезгаданому діалоговому вікні.

— Системний монітор. Програма «Системний монітор» («System Monitor») є у Windows 9x і у Windows NT. З її допомогою графічно відображають параметри системи, зокрема використання пам'яті, файлової системи, мережі, ядра, кеш-пам'яті та ін.

Встановлюючи деякі програми, можна додати параметри для відображення у вікні «Системний монітор». Це дає змогу, переглядаючи діаграми, побачити недоліки діагностованої системи.

— Відомості про систему. У Windows 98/2000 і у Windows NT є програма «Відомості про систему» («System Information»). З її допомогою одержують детальну інформацію про параметри комп'ютера. Програма «Перегляд подій» («Event Viewer») у Windows NT зберігає записи про всі системні події в спеціальному файлі. Ця інформація допомагає при пошуку несправностей або вирішенні проблеми, що виникла з обладнанням.

— Програми для тестування комп'ютерного обладнання, що підтримує операційні системи. Основні з них поділяють на два класи: програми для підтримання операційної системи MS DOS; програми, що підтримують операційні системи Windows 95/98 і Windows NT.

• До класу програм, що підтримують операційну систему MS DOS належать: AMI DIAGNOSTIC V5.42; CHECKIT V3.0; DR. HARDWARE SYSINFO V5.0E; MICROSOFT DIAGNOSTIC V2.13; NORTON DIAGNOSTIC V9.0. Усі вони призначені для комплексного діагностування вищезгаданого обладнання.

• Програми, що підтримують операційні системи Windows 95/98 і Windows NT: BCM DIAGNOSTICS V1.02 і CHECKIT 98 V6.00.035 (призначені для комплексного діагностування обладнання); HMONITOR V3.1.0.4 PROFES-

SIONAL і MOTHERBOARD MONITOR V3.831 (призначені для моніторингу стану системної плати — напруга живлення, перегрів процесора та ін).

У переліку наведено програми, що підтримують найпоширеніші й раніше розроблені операційні системи.

5. Діагностичні програми загального призначення. Для діагностування IBM-сумісних комп'ютерів існує багато різних програм. Більшість з них орієнтована на користувачів з підготовкою середнього рівня.

Тестові програми запускають у пакетному режимі. Це дає змогу без втручання оператора виконувати серії тестів. Можна скласти програму автоматизованого діагностування, яка ідентифікуватиме можливі дефекти і несправності. Її використовують для діагностування групи комп'ютерів.

Найефективніші з них — програми для діагностування системної пам'яті різних типів: основної, розширеної і додаткової. Локалізація несправностей досягає глибини окремої мікросхеми чи біта модуля.

— AMIDiad. У сучасних IBM-сумісних комп'ютерах здебільшого використовують AMI BIOS. Це одна з популярних програм фірми «AMI» («American Megatrends Inc.»). Окрім того, вона випускає на дискетах розширену версію цієї програми, записаної в ПЗУ. AMIDiad — програма загального діагностування, що підходить для будь-яких IBM-сумісних систем і тестує більшість нових процесорів, систем з кількома процесорами (до 16-ти), оперативну пам'ять об'ємом до 4 Гбайт, контролер USB (Universal Serial Bus — універсальна послідовна шина), SCSI-адаптери та ін. Крім того, вона дає змогу тестувати мульти-медіа функції комп'ютера та їх апаратне забезпечення (CD-ROM, звукову карту, відеоадаптер), а також мережні функції системи.

— CHECKIT PRO. Пакет фірми «Touchstone Software Corp.» складається з набору тестів, що діагностують процесор, основну, розширену і додаткову пам'ять, жорсткий диск, дисководи, відеоадаптер, монітор, мишу і клавіатуру (в тому числі, пристрої стандарту VESA). Набір призначений для операційних чисток систем Windows і DOS.

Одна з версій цього пакета — CHECKIT PROFESSIONAL EDITION — виконує аналіз швидкодії. З її допомогою одержують повну інформацію про апаратні засоби комп'ютера, зокрема про об'єм і поточний розподіл встановленої пам'яті, тип і ємність жорсткого диска, доступні використовувані переривання, швидкість передавання даних модема чи факсмодема та ін. Текстовий редактор, що входить до пакета, дає змогу оперативно змінювати вміст файлів CONFIG. SYS AUTOXEC. BAT, SYSTEM. INI і WIN. INI, а також системного реєстру Windows 9x.

— MICRO-SCOPE. Повнофункційна діагностична програма для IBM-сумісних комп'ютерів. Її розробила фірма «Micro 2000». Вона перевіряє порти введення-виведення, лінії запиту переривань, дає змогу точно визначити використовуване конкретним адаптером або пристроєм переривання. Це необхідно при вирішенні конфліктів між адаптерами. MICRO-SCOPE має свою операційну систему, і тестування проходить без участі системної BIOS. Програму використовують для діагностування комп'ютерів під керуванням інших операційних систем, зокрема UNIX чи NOVELL. З її допомогою інсталюють жорсткий диск і запускають DOS. MICRO-SCOPE може тестувати мікропроцесори фірм «Intel», «AMD», «Cyrix», а також накопичувачі CD-ROM і DVD-ROM.

— Пакет діагностичних програм NORTON UTILITIES. До складу цього пакета версії 8.0 (для DOS і Windows) і версії для Windows 9x, який є невід'ємною частиною системи зберігання і відновлення даних, тестування і пошуку несправностей апаратної частини, входить програма NORTON DIAGNOSTICS (NDIAGS). Це одна з найкращих діагностичних програм загального призначення. NDIAGS дає інформацію про тип процесора і співпроцесора, версії системної BIOS, відеоадаптера, миші і клавіатури, тип жорсткого диска, дисководів гнучких дисків, обсяг встановленої оперативної пам'яті (розширеної і додаткової), тип системної шини, кількість послідовних і паралельних портів та ін.

NDIAGS тестує основні компоненти системи і навіть перевіряє роботу індикаторів, розташованих на клавіатурі. Також програма шукає несправності монітора.

Крім того, пакет NORTON UTILITIES дає доступ до програм SPEEDISK, DISK DOCTOR і CALIBRATE. Вони є еталоном програм, що використовують для діагностування жорсткого диска і відновлення програмного забезпечення. Утиліта SYSINFO із цього пакета тестує швидкодію системи.

— PC TECHNICIAN. Програму розробила фірма «Windsor Technologies». Цей продукт уже давно існує на ринку діагностичних програм, однак постійно модифікується і сьогодні відповідає останнім досягненням комп'ютерної техніки. Дає змогу перевірити функціювання всіх основних компонентів комп'ютерної системи. PC TECHNICIAN

має свою операційну систему для уникнення можливих програмних конфліктів. Оскільки програма написана на Асемблері, то під час тестування безпосередньо звертається до апаратури. До PC TECHNICIAN додають роз'єми-заглушки для тестування послідовних і паралельних портів.

— QUAPlus/FE. Розробила фірма «Diagsoft». Це одна з найскладніших багатофункційних програм. Дає змогу тестувати будь-які комп'ютери. QUAPlus/FE точно оцінює продуктивність системи. Вона міститься на завантажувальних дискетах. Це забезпечує незалежність її завантаження від встановленої операційної системи (DOS, UNIX, OS/2).

За допомогою QUAPlus/FE тестують системну плату, основну, розширену і додаткову пам'ять, відеоадаптер, жорсткий диск, дисководи, CD-ROM, мишу, клавіатуру, принтер, послідовні і паралельні порти. Крім того, можна отримати вичерпну інформацію про конфігурацію системи і встановлене обладнання, тип процесора, об'єм оперативної пам'яті, а також повну таблицю переривань, список драйверів пристроїв і резидентних програм та ін.

До пакета QUAPlus/FE входять утиліти, призначені для спеціалістів з ремонту комп'ютерів. Зокрема, програма-редактор вмісту CMOS-пам'яті, що дає змогу змінити системний час і дату, тип жорсткого диска, об'єм встановленої пам'яті тощо. Також пакет містить налагоджувач для послідовного COM-порта, утиліту низькорівневого форматування і тестування жорсткого диска, програму тестування гнучких дисків, редактор файлів конфігурації, програму дистанційного керування і контролю, яка дає змогу спеціалістам зі служби технічного обслуговування керувати вашим комп'ютером через модем.

Щоб виявити приховані дефекти, тривалий час роботу комп'ютера тестують способом завантаження, залучаючи всі його системні ресурси. Це підвищує ймовірність прояву дефектів під час тестування, а не після нього. Як правило, тест триває дві-три доби.

— ACTIWE SMART. Програма дає змогу простежувати стан жорстких дисків і передбачати можливі збої в їх роботі. З її допомогою можна отримати інформацію про жорсткі диски, зокрема продуктивність диска, режим UDMA, ємність, файлову систему та ін.

— CD-R DIAGNOSTIC. Призначена для одержання інформації про CD-R (Compact Disc Recordable — компакт-диск з можливістю запису), CD-RW (Compact Disc ReWritable — компакт-диск з можливістю багаторазового перезапису) та звичайні диски. Програма перевіряє, чи є на диску помилки, намагається відтворити недоступні файли CD-RW.

— CPU STABILITY TEST. Призначена для перевірки стабільності роботи процесора. Після виконання 12-годинного тесту програма видає сертифікат стабільності, короткий звіт про кількість помилок та ін.

— COLD MEMORY. Діагностична програма, що призначена для тестування оперативної пам'яті (до 1 Гбайт).

— HDD (Hard Disc Drive — накопичувач на жорстких дисках (вінчестер)) UTILITY. Пакет утиліт, що надає інформацію про жорсткі диски, зокрема про моніторинг тестування і керування різними їх функціями. Використовують для знаходження і усунення різних дефектів. Це одна з найкращих розробок у своєму класі.

— MODEM DOCTOR FOR WIN 2000/95/98/NT4 VER 2.0. Програма для діагностування модема. Має велику кількість тестів, портів тощо. Виводить інформацію про модеми, S-регістри та ін.

— NOKIA MONITOR TEST. Одна з високоякісних програм для тестування монітора. Є русифікований варіант опису програми.

— WIN TUPE 98. Програма видає інформацію і тестує різні компоненти системи. Виконує тести процесора, пам'яті, відео, зокрема DIRECT 3D і OPEN GL, тести жорсткого диска. Підтримує MMX, 3d Now і SSE.

Широкий асортимент діагностичних програм дає змогу користувачеві або спеціалістам з технічного обслуговування ПК обирати найефективніші з них для швидкого пошуку несправностей, що виникають під час експлуатації комп'ютера.

Щодо діагностування периферійних пристроїв ПК, то воно має свої особливості, тому засоби, що його реалізують, слід розглядати окремо.

4.3. Засоби діагностування периферійних пристроїв ПК

Під периферійними пристроями часто розуміють сукупність периферійної апаратури (пристрої введення і виведення інформації — клавіатура, монітор, принтер) та їх блоків керування, оскільки в деяких випадках вони є одним цілим з логічної і конструктивної точки зору. Блок керування периферійним апаратом виконує функцію перетворення інформації і керування конкретними його меха-

нізмами. Сукупність уніфікованих технічних і програмних засобів та правил спрягання процесорного (системного) блока з периферійною апаратурою називають інтерфейсом введення-виведення.

Загальна характеристика інтерфейсів введення-виведення

У сучасних комп'ютерах як засоби комунікації використовують послідовні і паралельні порти. Це — інтерфейси введення-виведення.

Послідовні порти. До них підключають двонапрямлені пристрої, що передають інформацію в комп'ютер і приймають її з нього. Серед них — модеми, миша, сканери та ін.

Паралельні порти. Їх використовують для підключення принтерів, які працюють в однонапрямленому режимі, мережних адаптерів, дисководів для жорстких і гнучких дисків та CD-ROM.

Послідовні і паралельні порти тестують програмним і апаратно-програмним способом. *Програмне тестування* проводять за допомогою спеціальних програм, *апаратно-програмне* — за допомогою роз'ємів-заглушок, що підключають до портів.

Тестування послідовних портів. Цю процедуру здійснюють за допомогою MICROSOFT DIAGNOSTICS (MSD) і тесту із замиканням петлі. Крім того, проконтролювати роботу послідовних портів можна у Windows 95/98.

MICROSOFT DIAGNOSTICS. Ця діагностична програма належить до програмних засобів тестування і входить до складу MS DOS 6.x, Microsoft Windows і Windows 9x. На компакт-диску з Windows 95 вона міститься в каталозі \other\msd, а на компакт-диску з Windows 98 — у каталозі \tools\oldmsdos. При інсталяції операційної системи програма автоматично не встановлюється. Її треба запустити безпосередньо з компакт-диска або скопіювати з нього заздалегідь на жорсткий диск.

Щоб запустити MSD, переходять у каталог (він може вказуватись у змінній PATH), в якому розташований файл MSD.EXE. В командному рядку DOS вводять MSD і натискують ENTER. На екрані з'являється меню. Для послідовних портів обирають опцію COM PORTS. У цьому випадку з'являється інформація про мікросхему UART, що встановлена в послідовному порту комп'ютера, та про доступні порти. Якщо один з портів не реагує на тест, то він не потрапляє

у звіт програми. Також MSD видає інформацію щодо портів, які в цей момент використовують. Багато програм типу MSD працюють краще в середовищі DOS, тому їх рекомендують запускати повторно в цьому самому режимі.

Windows 95/98. У цій операційній системі також відображається інформація про те, чи працюють порти. Щоб перевірити і проконтролювати їх роботу, треба клацнути правою кнопкою миші на піктограмі «Мій комп'ютер» («My Computer») і в меню, що з'явилося, обрати пункт «Властивості» («Properties»). Далі у діалоговому вікні обирають вкладку «Пристрої» («Device Manager»). На екрані відобразиться список підключених до комп'ютера пристроїв. Якщо пристрій функціює неправильно, то поряд з назвою з'явиться знак оклику в жовтому колі. Далі розкривають список портів і двічі клацають на потрібному. Windows 95/98 вкаже, чи працює цей порт або назве пристрій, з яким він конфліктує.

Тест із замиканням петлі. Він є найнадійнішим для діагностування послідовних портів. Дає змогу перевірити справність самого порту та проконтролювати підключені кабелі. При цьому замикають внутрішню (цифрову) і зовнішню (аналогову) петлі. Тест з внутрішньою петлею виконують за допомогою діагностичної програми без додаткових пристроїв, тест із зовнішньою — за допомогою спеціального роз'єму-заглушки, який підключають до гнізда порту, що діагностується. Це — простий інтерфейсний кабель, що замикає порт на самого себе. Дані, які передає послідовний порт, проходячи через заглушку, повертаються на приймальні контакти роз'єму. Під час цієї операції порт працює одночасно в режимах передавання і прийому. Більшість діагностичних програм виконує тестування із замиканням петлі. Потрібні роз'єми додають до тестових дискет.

Щоб перевірити послідовні порти (DB25S) програмами фірми «IBM», у 25-контактному роз'ємі-заглушці виводи з'єднані так:

1—7;
2—3;
4—5—8;
6—11—20—22;
15—17—23;
18—15.

До пакета NORTON UTILITIES версій 7.x і вище входять програми NORTON DIAGNOSTICS (NDIAGS.EXE). Для них потрібні нестандартні заглушки. Їх випускає фірма «Symantec». У 25-контактному роз'ємі-заглушці для

контролю послідовних портів (DB25S) програмами фірми «Symantec» виводи з'єднують за таким принципом:

2—3;
4—5;
6—8—20—22.

У 9-контактному роз'ємі-заглушці для перевірки послідовних портів (DB9S) програмами фірми «IBM» виводи з'єднують так:

1—7—8;
2—3;
4—6—9.

Щоб перевірити послідовні порти (DB9S) програмами фірми «Symantec», у 9-контактному роз'ємі-заглушці виводи з'єднують таким чином:

2—3;
7—8;
1—4—6—9.

Для того щоб проконтролювати кабель, необхідно підключити заглушку на інший його кінець.

Послідовні порти, як правило, установлюють на системній платі ПК.

Тестування паралельних портів. Для тестування паралельних портів застосовують ті самі способи, що і для послідовних. Зокрема, програми тестування та допоміжні пристрої аналогічні засобам діагностування послідовних портів. Типи роз'ємів-заглушок залежать від типів програм, що використовують для тестування.

У 25-контактному роз'ємі-заглушці для контролю паралельних портів (DB25P) програмами фірми «IBM» виводи з'єднані так:

1—13;
2—15;
10—16;
11—17.

У 25-контактному роз'ємі-заглушці для контролю паралельних портів (DB25P) програмами «Norton Utilities» виводи з'єднують за таким принципом:

2—15;
3—13;
4—12;
5—10;
6—11.

Отже, розглянувши способи тестування послідовних і паралельних портів, можна зробити висновок, що їх проводять за аналогічними методиками.

Несправності периферійних пристроїв

До несправностей периферійних пристроїв належать: несправності адаптерів, моніторів, звукових плат, конфлікти переривань і каналів прямого доступу до пам'яті DMA (Direct Memory Access — передавання даних у режимі прямого доступу до пам'яті) тощо. Розв'язання цих питань за допомогою засобів діагностування дає змогу покращити роботу ПК.

Несправності адаптерів і моніторів. Більшість проблем, пов'язаних з несправностями цих пристроїв, розв'язують шляхом їх заміни на справні. Проте коштує це дорого. Часто проблему вирішують, налагоджуючи монітор. Інформація про несправності й відмови цих пристроїв може бути на Web-сервері фірми-виробника.

Найпростіший спосіб самостійно відремонтувати, наприклад кабель монітора, — вирівняти зігнуті штирі, якщо вони є в роз'ємі. Однак краще робити це в сервісних фірмах, оскільки самостійно відремонтувати такий пристрій практично неможливо. Не завжди можна знайти докладний опис електричних схем монітора чи адаптера.

Здебільшого несправності в системах відображення пов'язані з неправильним налагодженням програмних драйверів. Тому одразу після виявлення збоїв треба перевірити, чи правильно встановлена версія драйвера і як він налагоджений.

Ідентифікація несправностей звукових плат. Під час функціювання звукова плата використовує такі ресурси: номер переривання IRQ (Interrupt-Request — лінія запиту на переривання), базову адресу введення-виведення та канали прямого доступу до пам'яті. Вибираючи ресурси, необхідно уникати конфліктів з іншими пристроями. У процесі пошуку несправностей інколи змінюють положення перемичок або перемикачів на платі чи навіть конфігурацію інших плат.

Апаратні несправності та конфлікти звукової плати з іншими пристроями виникають з різних причин. Плата може не працювати або повторювати одні й ті самі звуки, або приводити до зависання комп'ютера. Це зумовлено су-

місним використанням звукової плати з іншими пристроями лінії запиту переривань IRQ, каналів прямого доступу до пам'яті, адрес введення-виведення. Звукова плата — це кілька пристроїв, кожен з яких, починаючи з базового, потребує свого діапазону адрес.

До більшості звукових плат додають установчі програми, що аналізують системи конфігурації і намагаються знайти ресурси, ще не використані іншими пристроями. За необхідності користуються послугами диспетчера пристроїв, до якого можна ввійти через панель керування Windows 9x. Розподіл ресурсів за умовчанням між компонентами стандартних звукових плат SOUND BLASTER 16 представлено в табл. 4.8.

Таблиця 4.8

Розподіл ресурсів SOUND BLASTER 16

Пристрій	Переривання	Порт введення-виведення	16-розрядний DMA	8-розрядний DMA
Аудіо	IRQ5	220h—223h	5	1
MIDI-порт	—	330h—331h	—	—
Частотний синтезатор	—	388h—388h	—	—
Ігровий порт	—	200h—207h	—	—

Ресурси звукової плати змінюються у разі конфліктів з іншими пристроями. Такі компоненти, як ігровий порт MIDI (Musical Instrument Digital Interface — цифровий інтерфейс звукової плати (музичних інструментів)) та частотний синтезатор не використовують переривань і прямого доступу до пам'яті. За конфлікту звукової плати з будь-яким пристроєм слід віддати перевагу зміні настроювання саме цього пристрою, а не звукової плати.

Якщо у Windows 95/98 плата підтримує технологію автоматичного налагодження Plug and Play, то для вирішення конфлікту можна використати диспетчер пристроїв.

Розв'язання конфліктів переривань. Звукова плата може використовувати будь-яке з доступних переривань. При розробленні її налагоджують на конкретне переривання, як правило, на IRQ5, і в разі необхідності краще змінити налагодження інших плат.

Стандартні звукові плати, зокрема SOUND BLASTER 16, налагоджують на переривання 2, 5, 7, 10. За умовчанням

використовують переривання 5 переважно у паралельному порту LPT (Line Printer — порт комп'ютера для підключення принтера) 2. Проте його немає в більшості комп'ютерів і комп'ютерних систем. При використанні IRQ7 може виникнути конфлікт з першим паралельним портом LPT1. За переривання IRQ10 пристрої в комп'ютері здебільшого не працюють, тому його може застосовувати звукова плата SOUND BLASTER. Її встановлюють також на використання IRQ2. Тоді система буде сприймати звукову плату як за переривання IRQ9.

Розв'язання конфліктів каналів прямого доступу до пам'яті. Стандартні і додаткові системні пристрої в IBM-сумісних комп'ютерах використовують 7 каналів прямого доступу до пам'яті. Для звукових плат типу SOUND BLASTER необхідні два синтезованих DMA канали.

Канали DMA 0, 1, 2 і 3 забезпечують тільки 8-розрядний режим передавання інформації. Канали DMA 5, 6 і 7 — тільки 16-розрядний. Основною причиною конфлікту каналів DMA є повна відсутність звуку.

Звукова плата SOUND BLASTER 16 використовує канал DMA1 для 8-розрядного звуку (нижній DMA) і DMA5 — для 16-розрядного звуку (верхній DMA). Призначення каналів DMA в стандартному IBM-сумісному комп'ютері з установленою в ньому платою SOUND BLASTER наведено в табл. 4.9.

Таблиця 4.9

Канали DMA та їх призначення

Канал DMA	Стандартна функція	Наявність слота	Тип і розрядність плати	Розрядність інформації, що передається
1	2	3	4	5
0	Доступно для використання	Так	16	8
1	Звукова плата (нижній DMA)	Так	8/16	8
2	Контролер гнучких дисків	Так	8/16	8
3	Паралельний порт з розширеними можливостями (ECP)/є доступ для користування	Так	8/16	8
4	Каскад першого контролера DMA	Так	—	16

Закінчення таблиці 4.9

1	2	3	4	5
5	Звукова плата (верхній DMA)	Так	16	16
6	SCSI-адаптер/є доступ для користування	Так	16	16
7	Є доступ для користування	Так	16	16

Апаратні конфлікти. При роботі з Windows 9x з наявністю технології автоматичного налагодження Plug and Play для розв'язання конфлікту слід скористатися вбудованим засобом «Device Manager» («Диспетчером пристроїв»).

Найчастіше конфлікти виникають при використанні SCSI-адаптерів; мережних плат; плат адаптерів миші; адаптерів послідовних портів COM3 і COM4; адаптера паралельного порту LPT2; внутрішнього модему; інтерфейсної плати сканера.

Для визначення пристрою, що конфліктує зі звуковою платою, необхідно вийняти всі плати розширення, крім звукової і відеоадаптера. Потім вставляти їх по одній. Пристрій, після якого звукова плата перестане працювати, і буде причиною конфлікту. Далі необхідно змінити настроювання плати, що викликала конфлікт, або звукової плати. У таких випадках змінюють канал DMA, канал IRQ чи адресу введення-виведення. Цього досягають шляхом зміни положень перемичок і перемикачів на платі або використовують програму конфігурації звукової плати.

Пошук несправностей клавіатури. Найпоширеніщими несправностями клавіатури є замикання клавіш і дефекти кабелю. До цього часто призводить введення неправильного символу. Пересвідчитися в несправності кабелю можна, перевіривши за допомогою тестера або цифрового мультиметра його з'єднання. З огляду на низьку вартість клавіатури інколи краще замінити її повністю, ніж замовляти кабель.

Перші повідомлення про несправність пристрою з'являються під час виконання процедури POST. Код помилки в такому разі починається із цифри 3. Ремонт клавіші здебільшого полягає в тому, що з неї знімають ковпачок і чистять контактуючу поверхню.

Щоб визначити, чи є несправність роз'єму клавіатури на системній платі, вимірюють напругу на певних контактах. Спочатку вимикають комп'ютер, потім від'єднують

клавіатуру. Далі вмикають живлення і перевіряють напругу між загальним дротом і останніми контактами. Якщо всі напруги знаходяться в межах, зазначених у табл. 4.10, то вузли системної плати, які стосуються клавіатури, справні.

Таблиця 4.10

Сигнали на роз'ємі клавіатури

Контакт DIN[1]	Контакт міні DIN	Сигнал	Напруга, В
1	5	Синхронізація клавіатури	+2,0—5,5
2	1	Дані з клавіатури	+2,0—5,5
3	—	Зарезервований	—
4	3	Загальний	—
5	4	+5 В	+4,8—5,5

[1] DIN (Deutsch Industrie Norm — малогабаритний круглий багатоконтактний роз'єм)

Якщо виміряні напруги відрізняються від поданих у таблиці, то вийшла з ладу системна плата. Якщо — ні, то несправність слід шукати в кабелі чи клавіатурі. У разі знаходження на системній платі плавкого запобіжника необхідно пересвідчитися в його справності. Крім того, в деяких моделях комп'ютерів контролер клавіатур знімають. У такому випадку його можна замінити справним. У табл. 4.11 наведено список стандартних кодів несправностей клавіатури для процедури POST і діагностичних програм.

Таблиця 4.11

Коди несправностей клавіатури для процедури POST

Код помилки	Опис
1	2
3xx	Несправність клавіатури
301	Несправність скидання клавіатури чи замикання клавіші (xx301, xx — шістнадцятковий сканкод)
302	Заблокований вимикач клавіатури на системному блоці
302	Помилка тесту клавіатури, яку визначає користувач
303	Несправність клавіатури або системної плати; несправність контролера

Закінчення таблиці 4.11

1	2
304	Несправність клавіатури або системної плати; висока частота синхронізації клавіатури
305	Несправність джерела живлення +5 В клавіатури; у PS/2 вийшов з ладу запобіжник клавіатури
341	Несправність клавіатури
342	Несправність кабелю клавіатури
343	Несправність кабелю чи плати світлодіодів клавіатури
365	Несправність кабелю чи плати світлодіодів клавіатури
366	Несправність інтерфейсного кабеля клавіатури
367	Несправність кабелю чи плати світлодіодів клавіатури

Клавіатура ПК є периферійним засобом, з яким найбільше працюють користувачі. Вона може частіше виходити з ладу за інші пристрої ПК. Тому знання методик пошуку несправностей клавіатури дуже важливе для забезпечення постійної працездатності комп'ютера.

Несправності миші. Для контролю працездатності миші необхідно перевіряти її апаратні і програмні засоби. Апаратні засоби досить прості, тому їх діагностування не займає багато часу. Складніше відшукати несправності програмного забезпечення. Дефекти і несправності миші насамперед пов'язані із забрудненням або перериванням.

Забруднення. У цьому випадку достатньо почистити пристрій, оскільки пил і бруд, що накопичуються на кульках і валиках, не дають змоги вказівнику вільно рухатися екраном.

Переривання. Виникає під час використання миші, коли передається інформація від пристрою до програми-драйвера. Це зумовлено тим, що відведене для миші переривання використовує ще один пристрій. Тоді миша або працює неправильно, або зовсім не працює. Подібний конфлікт виникає, якщо для підключення миші використовують послідовний інтерфейс при умові доукомплектування системної плати комп'ютера третім або четвертим послідовним портом. Це відбувається тому, що в комп'ютерах із шиною ISA непарні (1, 3) і парні (2, 4) послідовні порти часто налагоджують для використання одного переривання. Мишу та інші пристрої (наприклад, модем) слід підключати до непарного і парного портів, оскільки для них відведені різні переривання.

Усунути конфлікти, пов'язані з перериваннями, можна, налагодивши систему так, щоб два будь-які пристрої не використовували одне і те саме переривання.

Якщо миша комп'ютера підключена до шинного інтерфейсу, то для виявлення конфлікту через переривання необхідно скористатися діагностичною програмою MSD. Приміром, програма MSD намагається ідентифікувати мишу, а її несправність призводить до зависання комп'ютера після завантаження драйвера. У такому випадку MSD запускають ключем /I, тоді програма пропускає етап попередньої перевірки всіх пристроїв у комп'ютері. Після цього слід виконувати всі тести окремо, в тому числі й миші, доти, доки система не заблокується. Якщо під час тестування миші комп'ютер зависне, то — несправна миша або її порт.

Якщо пристрій не працює з конкретною прикладною програмою, то перевіряють настроювання програми або самої миші. Необхідно також перевірити, чи повідомлено програму про наявність миші. Пересвідчившись, що пристрій не працює саме із цією програмою, необхідно звернутися до документації програми.

Виявлення несправностей блока живлення. Про несправності блока живлення сигналізує багато ознак. Наприклад, поява помилок парності. Це зумовлено тим, що з блока живлення мікросхеми отримують напругу. Якщо вона не відповідає певним вимогам, то виникають збої. Інший критерій — хаотичність адрес комірок пам'яті, в яких виникають збої. Відсутність цього свідчить, що несправною передусім може бути сама пам'ять.

Несправність блока живлення спричиняє такі наслідки:

— будь-які помилки і зависання комп'ютера при вмиканні;

— спонтанне перезавантаження і зависання під час роботи;

— хаотичні помилки парності та інші помилки пам'яті;

— одночасна зупинка жорсткого диска і вентилятора (немає напруги +12 В);

— перегрівання комп'ютера, оскільки з ладу виходить вентилятор;

— перезапуск комп'ютера без певних дій користувача;

— ураження електричним струмом при доторканні до корпусу комп'ютера або роз'ємів;

— невеликі статичні розряди, що порушують роботу системи.

За несправності в комп'ютері блока живлення можливі такі збої:

— зупинка в роботі комп'ютера;

— поява диму;

— згорання на розподільчому щитку мережного запобіжника.

Для усунення несправності необхідно провести вимірювання і виконати відповідні тести.

Вимірювання вихідних напруг блока живлення. Це один з найпростіших тестів. Він дає змогу з'ясувати, чи виробляються відповідні напруги і чи їх значення перебуває в допустимих межах. Цю процедуру проводять здебільшого при підключених номінальних навантаженнях. Для цього слід використовувати тільки цифрові прилади, випробувальна напруга яких +1,5 В. У стрілкових цей показник становить +9 В. Отже, випробувальна напруга стрілкових приладів може зруйнувати електронні компоненти з напругою живлення +5 В. Якщо значення вимірювальних напруг блока живлення виходять за межі допустимих (табл. 4.12), то його замінюють.

Таблиця 4.12

Значення допусків вихідних напруг блоків живлення

Номінальна напруга, В	Широкий допуск		Жорсткий допуск	
	Мін. (−10%)	Макс. (+8%)	Мін. (−5%)	Макс. (+5%)
3,3	2,97	3,63	3,135	3,465
±5,0	4,5	5,4	4,75	5,25
±12,0	10,8	12,9	11,4	12,6

Перевіряючи блок живлення за межами комп'ютера для виходів +5 В і +12 В, використовують навантажувальні резистори (еквівалент навантаження). Для 12-вольтового джерела живлення вмикають паралельно кілька резисторів по 6 Ом (50 Вт). Кожен доповнюють вимикачем для підключення або відключення від джерела живлення. Через кожний резистор протікатиме струм 2 А (потужність — 24 Вт). Для 5-вольтового джерела живлення використовують 50-ватні дротові резистори номіналом 1 Ом, з'єднані так само, як на схемі для контролю 12-вольтового джерела живлення. Кожний такий резистор у мережі +5 В споживатиме струм 5 А і розсіюватиме потужність 25 Вт.

Плата PC POWER CHECK фірми «Data Depot». Її використовують для перевірки блока живлення у складі комп'ютера чи без нього. Плату підключають до шини ISA. На ній встановлено кілька світлодіодів для індикації (фіксації показників) перевищення або зниження необхідних рівнів напруг, перешкод і викидів на шинах живлення. Вона працює як у режимі неперервної індикації, так і з пам'яттю. Її підключають у слот, як і будь-яку плату адаптера. В автономному режимі плата PC POWER CHECK імітує системну плату.

Тестер PC POWER SYSTEM ANALYZER фірми «TCE». Це один зі складних приладів для діагностування блоків живлення. Він дає змогу виконати три тести. При цьому блок живлення не виймають з комп'ютера. До комплекту аналізатора входять адаптери шин ISA/EISA і MCA. З їх допомогою контролюють напруги в мережах +5 В і +12 В. Аналізатор визначає навантажувальну здатність джерела живлення. В процесі вимірювання напруг реєструють усі параметри, пов'язані з виявленою несправністю, відображають режими роботи аналізатора і поточний стан блока живлення. Результати тестування можна надрукувати. Аналізатор PC POWER SYSTEM ANALYZER — один з найбільш багатофункційних і високоефективних приладів для діагностування блоків живлення. З огляду на свою складність призначений для професіоналів.

Знання структури і принципів функціювання апаратних засобів діагностування, змісту й основ функціювання діагностичних програм є передумовою успішної і ефективної експлуатації Ц і МПП та ПК під час використання їх за основним призначенням.

Обчислювальні пристрої та системи є програмно керованими, тобто комплексом апаратних і програмних засобів. Завдання, що відпрацьовуються ними, представлені у вигляді програмних кодів. Тому є необхідність розглянути програмне забезпечення ПК як окремий об'єкт діагностування.

Запитання. Завдання

1. Назвіть типи засобів контролю і діагностування обчислювальних пристроїв. Дайте їм характеристику.
2. Розкрийте принцип дії та призначення логічного пробника.
3. Які функції виконує логічний аналізатор стану? Охарактеризуйте його структурну схему.
4. У чому полягає принцип дії сигнатурного аналізатора?
5. Назвіть основні принципи схемотехнічного проектування автоматизованих систем діагностування Ц і МПП.

6. Охарактеризуйте структуру системи тестового комбінованого діагностування Ц і МПП.

7. Дайте характеристику вбудованим засобам самотестування процесорів персональних комп'ютерів, зокрема механізму BIST.

8. У чому суть внутрішнього апаратного контролю процесорів Pentium?

9. Назвіть функції, які виконують тестери модулів пам'яті.

10. Дайте загальну характеристику інтерфейсу JTAG.

11. Для чого призначений індикатор D-LED™ і як він функціює?

12. Яким чином здійснюють порівняльну оцінку ефективності систем діагностування?

13. Назвіть і охарактеризуйте види діагностичних програм, які існують для IBM-сумісних програм.

14. У чому полягає суть самоперевірки POST при вмиканні ПК у мережу живлення?

15. Дайте характеристику звуковим сигналам, які видає процедура POST.

16. Які повідомлення про помилки видає на екран процедура POST?

17. Обгрунтуйте механізм використання кодів помилок, що видає процедура POST у порти введення-виведення.

18. З якою метою розробляють діагностичні програми фірми-виробники?

19. Наведіть приклади діагностичних програм фірм-виробників і розкрийте їх функційні можливості.

20. Проілюструйте на конкретних прикладах діагностичних програм операційних систем їх функційне призначення.

21. Назвіть діагностичні програми загального призначення і розкрийте їх функційні можливості.

22. Назвіть програмні засоби тестування послідовних портів. Які програми існують для здійснення цієї процедури?

23. Обгрунтуйте принцип діагностування послідовних портів у Windows 98.

24. У чому суть тестування послідовних портів із замиканням петлі?

25. Охарактеризуйте принцип тестування паралельних портів.

26. Розкрийте методи пошуку і усунення несправностей адаптерів і моніторів.

27. У чому суть методів ідентифікації несправностей звукових плат?

28. Назвіть типи конфліктів переривань. Яким чином їх розв'язують?

29. Як вирішують питання конфліктів каналів прямого доступу до пам'яті (DMA)?

30. Наведіть приклади розв'язування апаратних конфліктів.

31. За яким принципом проводять пошук і усувають несправності клавіатури?

32. Яким чином виявляють несправності миші?

33. Як здійснюють пошук і ліквідують несправності блоків живлення?

34. Розкрийте функції та призначення тестера PC POWER SYSTEM ANALYSER.

5.

Діагностування програмного забезпечення

Надійність програмного забезпечення (ПЗ) як складова надійності ПК і обчислювальних системі відіграє важливу роль у забезпеченні їх працездатності. Про надійність апаратних засобів вже йшлося, але програмне забезпечення є досить важливою складовою ПК, тому виникає необхідність детально розглянути його надійність.

5.1. Надійність програмного забезпечення

Забезпечення надійності програмних засобів ПК і обчислювальних систем істотно відрізняється від забезпечення надійності апаратури. Щоб зрозуміти це потрібно ознайомитися із загальною методологією діагностування ПЗ.

Порівняльний аналіз надійності апаратних засобів ПК і програмного забезпечення

Умова успішного дослідження або розроблення обчислювальної техніки — системний підхід до завдання, що вирішується. Це стосується і програмного забезпечення.

Воно має складну ієрархічну будову, що вимагає застосування до нього системного підходу.

Системний підхід — сумісний аналіз і розроблення всіх елементів системи.

ПЗ складається з багатьох модулів. Кожен з них виготовляють автономно. Модуль призначений для вирішення певного завдання, але всі вони в межах конкретного ПЗ спрямовані на досягнення єдиної мети. Тому всі модулі повинні бути взаємопов'язаними.

Модуль (лат. modulus — міра) — елемент (компонент) програми, стандартизований за формою запису і зовнішніми зв'язками.

Отже, з'ясувавши, що ПЗ складається з модулів і вимагає застосування системного підходу, можна стверджувати, що останній передбачає сумісний аналіз і розроблення керуючого і керованого об'єктів, давачів інформації, ліній зв'язку та ін. До керуючих і керованих об'єктів належать апаратні засоби і керуючі програми ПЗ (розробляючи їх, аналізують сукупність вихідних даних, допустимих кінцевих і проміжних результатів та ін.). Тому надійність роботи апаратних засобів розглядають у комплексі з надійністю програмного забезпечення.

Надійність програмного забезпечення — властивість програми виконувати задані функції в заданих умовах роботи і на заданій ЕОМ.

Це аналогічно тому, як визначають поняття надійності апаратури (властивість об'єкта зберігати в часі у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах та умовах застосування, технічного обслуговування, зберігання і транспортування).

З імовірнісного підходу під надійністю ПЗ розуміють імовірність того, що при функціюванні системи протягом певного часу не буде виявлено помилок. Проте тут не враховано відмінність між помилками різних типів.

Надійність ПЗ краще розглядати з точки зору впливу помилок на користувача системи. Усе залежить від того, на якому етапі (розроблення, експлуатація) і в якому компоненті допущено помилку. Інколи її допускають на етапі проектування, і це призводить до незначних і непомітних наслідків, а на етапі експлуатації — до катастрофічних, і навпаки. Наприклад, запуск космічного корабля на Венеру був невдалим тому, що в операторі DO програми, написаної на Фортрані, була пропущена кома.

Надійність ПЗ не варто розглядати як внутрішню властивість програми. Вона тісно пов'язана з тим, як використовують програму. А слово «ймовірність» наголошує на тому, що користувач з певною ймовірністю не введе в ПЗ певний набір даних, який виведе систему з ладу. З цього погляду надійністю програмного забезпечення є ймовірність його роботи без помилок і відмов протягом певного часу.

Надійність є комплексним поняттям, яке залежно від призначення об'єкта і умов його застосування відображає такі властивості, як безвідмовність, довговічність, ремонтопридатність і збережуваність, або їх поєднання. Базовим поняттям серед зазначених властивостей є безвідмовність ПЗ.

Тут слід знову згадати таке поняття, як «відмова», під якою в цьому аспекті розуміють втрату функційним модулем (системою) здатності виконувати потрібну функцію.

Безвідмовність програмного забезпечення. Оцінюють імовірністю роботи ПЗ без відмов протягом заданого часу в певних умовах зовнішнього середовища.

Безвідмовність ПЗ з точки зору надійності принципово відрізняється від безвідмовності апаратних засобів обчислювальної техніки. Програми з часом не «зношуються». Характеристики функціювання ПЗ залежать тільки від його якості, що зумовлена процесом розроблення. Не існує поняття виходу з ладу програми. Безвідмовність ПЗ визначає його правильність (коректність) і цілком залежить від наявності помилок у ньому, допущених на етапі проектування, а безвідмовність апаратури визначають випадковими відмовами, що залежать від зміни параметрів на етапі експлуатації.

На функціювання ПЗ значною мірою впливають вхідні дані, які обробляє обчислювальний пристрій. З часом можуть трапитися такі, які програма не зможе правильно обробити. Це і спричинює появу помилок програмного забезпечення.

Відмови при функціюванні ПЗ є наслідком порушення кодів запису програми в пам'яті, нормального проходження обчислювального процесу. Це також спричиняють витирання або перекручування даних в оперативній чи довготривалій пам'яті комп'ютера. Крім того, відмови проявляються як програмне зупинення, систематичне пропускання певної групи команд, спотворення накопичуваних даних та ін. Це призводить до того, що кількість інформації та керуючих дій, які видають абонентам, скорочується. Наслід-

ком цього є порушення працездатності ПЗ. Найчастіше це трапляється в результаті конфлікту між реальними вхідними даними, що мають оброблятися, і програмою, що здійснює це оброблення. До вхідних даних належать ті, що вперше надійшли, і ті, що вже накопичені за попередній період функціювання ПЗ.

За цих умов необхідно ще раз акцентувати на тому, що з поняттям «відмови» взаємопов'язане поняття «збою», під яким у теорії надійності розуміють відмову, що самоусувається. Її усунення не потребує зовнішнього втручання для заміни компонентів, що відмовили.

Щодо апаратури, то поняття «збою» і «відмови» відрізняються між собою ступенем фізичного руйнування елементів і компонентів та відповідно необхідністю їх заміни. Головна особливість полягає в тому, що неможливо визначити чіткий часовий поріг розподілу збоїв і відмов.

У процесі відпрацювання програми або комплексу програм фізичного руйнування апаратури обчислювального пристрою чи комп'ютера немає, таким чином, зникає необхідність заміни компонентів або ремонту.

Відновлюваність ПЗ. Її характеризують витрати часу і праці на усунення відмов через помилки в програмі та їхні наслідки. В такій ситуації надійна програма повинна забезпечувати низьку ймовірність відмов при її відпрацюванні. Високу надійність програми забезпечує швидке реагування на її спотворення, а також спотворення даних або обчислювального процесу, відновлення працездатності програм за час, менший, ніж поріг між збоєм і відмовою. При цьому некоректна програма може функціювати надійно, тобто неправильні результати не спричинятимуть відмови, а коректна — ненадійно, насамперед через системну помилку при визначенні області зміни вихідних даних.

Після відмови в програмі відновлення полягає в: коригуванні й відновленні тексту програми; виправленні даних; внесенні змін в організацію обчислювального процесу та ін.

Відновлюваність ПЗ оцінюють за середньою тривалістю усунення помилки (помилок) в програмі й відновлення її працездатності. Відновлюваність ПЗ залежить від: складності структури комплексу програм; структурованості програм; алгоритмічної мови, якою розроблено програму; якості документації на програму та ін.

Оскільки усунення відмов у програмах не потребує ремонту, то можна автоматизувати відновлення ПЗ. Головне, щоб його час не перевищував граничного значення часу між збоєм і відмовою. За такого підходу відмови перетворюють на збої і завдяки цьому поліпшують показники надійності ПЗ. Тому в комплексі програм необхідно мати засоби, які дають змогу:

— здійснювати систематичний контроль, оперативно виявляти аномалії стану програм і даних та процесу функціювання ПЗ;

— діагностувати виявлені спотворення;

— обирати методи і засоби оперативного відновлення ПЗ;

— реалізовувати оперативне відновлення працездатності програмного комплексу;

— реєструвати кожний збій і відмову для встановлення їх систематичності з метою удосконалення ПЗ.

Поняття «відновлюваність» використовують і щодо апаратних засобів, але суть його щодо ПЗ різна.

Усталеність функціювання. Це одна з характеристик ПЗ, яка полягає в здатності обмежувати наслідки власних помилок і несприятливих впливів зовнішнього середовища (некоректність вхідних даних, помилки оператора та ін.) або протистояти їм.

Механізм виникнення відмов. Механізм виникнення відмов ПЗ істотно відрізняється від механізму виникнення відмов апаратури. Відмова апаратури зумовлена руйнуванням її певних елементів і компонентів, а відмова ПЗ — його невідповідністю поставленим завданням. Таку невідповідність спричинюють порушення специфікації чи неточна або неповна специфікація.

Порушення специфікації (технічних вимог до програми). Трапляється це насамперед у складних програмних системах, де окремі помилки програміста важко виявити.

Неточна або неповна специфікація. Виникає, коли при її складанні невідомі фактори, що впливають на роботу програми. Їх з'ясовують поступово, протягом експлуатації виробу. Особливо це стосується керуючих програм.

Оскільки програма є коротким записом дуже складних функцій, вказати в специфікаціях усі їх властивості інколи не легше, ніж розробити відповідну програму.

За ступенем точності специфікації є програми:

— функції яких повністю визначає специфікація;

— функції яких коректують зіставленням обчислювальних і вимірювальних результатів (моделюючі програми, що реалізують математичну модель фізичного об'єкта);

— які діють у постійно змінюваному середовищі (складаються з інших програм, даних і реальних систем; операційні системи; програми керування ресурсами та ін.).

Механізм виникнення відмов характеризують поняття «коректність програми» і «прихованість помилок».

Коректність програми. Це відповідність її специфікації. Оскільки специфікація може не відповідати фактичним вимогам до програми, то трапляються випадки, коли некоректна програма працює надійно або, навпаки, коректна — ненадійно.

Прихованість помилок. Полягає у виявленні помилки в результаті великої кількості комбінацій вихідних даних. Тому це можливо тільки протягом тривалої експлуатації. На відміну від них помилки, що виявляють у результаті будь-яких вихідних даних, не такі небезпечні, оскільки їх виявляють відразу при перших пробних прогонах програми.

Надійність програмного забезпечення $H_{ПЗ}$ певною мірою залежить від кількості помилок, внесених і не усунутих у процесі розроблення. Їх виявляють і усувають під час експлуатації. Якщо при виправленні помилок не вносять нові або їх вносять менше, ніж усувають, то надійність ПЗ безперервно зростає. Чим інтенсивніша експлуатація, тим частіше виявляють помилки і швидше зростає надійність ПЗ. Ця закономірність підтверджується на практиці, що засвідчує принципову відмінність надійності ПЗ від надійності апаратних засобів $H_{АЗ}$. Характер цих залежностей показаний на рис. 5.1, а зміна частоти відмов типового обчислювального пристрою зображена на рис. 5.2.

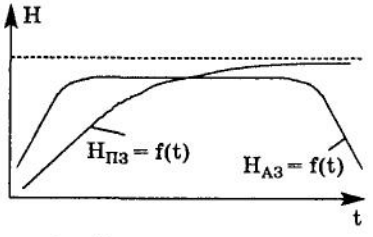


Рис. 5.1. Графік залежності надійності апаратних засобів $H_{АЗ}$ і програмних засобів $H_{ПЗ}$ від часу t

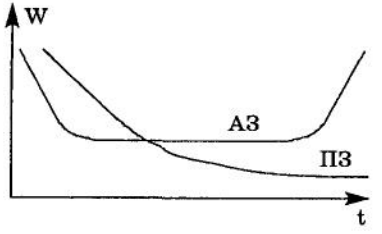


Рис. 5.2. Різниця між надійністю апаратних засобів (АЗ) і програмного забезпечення (ПЗ)

Ненадійність ПЗ є наслідком здебільшого помилок проектування. Якщо помилки ПЗ після виявлення виправляють і вони більше не повторюються, то зміна надійності відповідає кривій ПЗ (див. рис. 5.2). Ця крива відображає ймовірність, що при виправленні виявлених помилок не будуть внесені нові. Проте це не завжди так.

Зміна надійності за часом апаратних засобів і ПЗ різна. Надійність апаратних засобів визначають в основному прихованими збоями, а надійність ПЗ — прихованими в ньому помилками. Прояв помилок у програмах на відміну від апаратних засобів залежить від вхідних даних. Їх ідентифікацію найчастіше здійснюють на основі оброблення послідовності таких вхідних даних, які раніше не траплялися. Помилки в програмах проявляються не як випадкові, а як систематичні події. Зважаючи на те, що надійність є функцією помилок, які залишилися в ПЗ після вводу його в експлуатацію, то ПЗ, яке не має помилок, — абсолютно надійне. Однак для великих програм абсолютна надійність практично недосяжна.

Отже, надійність є показником якості, який характеризує властивість ПЗ виявляти в процесі експлуатації помилки, що залишилися в ньому, за певної сукупності вихідних даних. Для оцінювання надійності найбільше відповідають імовірнісні (статистичні) показники. Зокрема, ймовірність безвідмовної роботи, ймовірність відмови та інтенсивність відмов. У цьому аспекті під безвідмовністю розуміють властивість функційного модуля (системи) виконувати належні функції в певних умовах протягом заданого інтервалу часу чи наробітку. Оскільки, з огляду на відмови, розглядають тільки помилки в програмі, що не можуть самоусуватися, то ПЗ відносять до класу невідновлювальних систем. За основний критерій беруть тривалість наробітку T до першої відмови.

Ймовірністю безвідмовної роботи в теорії надійності вважають імовірність того, що при заданих умовах експлуатації протягом інтервалу часу t не виникне відмови:

$$P(t) = P(T \geq t).$$

Ймовірність відмови за час t визначають за формулою:

$$Q(t) = P(T < t) = 1 - P(t).$$

Інтенсивність відмов — інтегральний критерій $\lambda(t)$, що характеризує густину розподілу напрацювання до першої відмови, розраховану за умови, що до моменту часу, який розглядають, система пропрацювала безвідмовно:

$$\lambda(t) = a(t) / P(t) = -\frac{d \ln P(t)}{dt}.$$

При кожному прояві нову помилку, як правило, локалізують і виправляють. З огляду на це показники надійності ПЗ поліпшуються. Тому їх оцінювання матиме тимчасовий і наближений характер. Із цієї причини потрібні інші характеристики, адекватні реальному процесу зміни надійності. Зокрема, залежність інтенсивності виявлення помилок від кількості прогонів програми (реалізацій), залежність відмов від часу функціювання ПЗ та ін.

Оцінювання і прогнозування надійності програм

Ці процедури здійснюють на основі математичних моделей Джелінські-Моранди, Шумана, Шика-Волвертона.

Модель Джелінські-Моранди. Вона є однією з найбільш використовуваних. Ґрунтується на параметрах: 1) час до наступної відмови розподіляють експоненціально; 2) інтенсивність відмов програми пропорційна кількості помилок, що залишилися в програмі. Згідно з ними ймовірність безвідмовної роботи програм, як функція часу t_i, дорівнює:

$$P(t_i) = e^{-\lambda_i t_i}, \quad \text{де } \lambda_i = C_D(N - (i-1)), \tag{5.1}$$

де λ_i — інтенсивність відмов; C_D — коефіцієнт пропорційності; N — початкова кількість помилок програми. У формулі (5.1) відлік часу починають з моменту останньої $(i-1)$-ї відмови програми. Надійність програми буде тим вища, чим менше λ_i.

Модель Шумана. Відрізняється від моделі Джелінські-Моранди тільки тим, що періоди часу відлагодження і експлуатації розглядають окремо.

Модель Шика-Волвертона. Ґрунтується на тому, що інтенсивність прояву помилок пропорційна не тільки кількості помилок, що залишилися в програмі, але й часу, витраченому на їх відлагодження. Вади попередніх моделей полягають у тому, що за неточного визначення кількості помилок програми на початку відлагодження і експлуатації інтенсивність відмов стає від'ємною, що призводить до результату, який не має сенсу (інтенсивність відмов не може бути від'ємною). В них не враховано і те, що після виявлення помилки вона може бути повністю не усунута, до того ж можуть бути внесені нові. Модель Шика-Волвертона усуває ці вади, оскільки в ній враховано, що потік виникнення ситуацій, за яких можлива відмова програми, є пуассонівським з параметром λ. Отже, під час виявлення і виправлення помилок відмови в цих ситуаціях виникають з імовірністю, меншою за одиницю. Тому потік відмов подають як розріджений потік зі змінним коефіцієнтом розрідження p_i, де i — номер відмови, тоді:

$$p_i = 1 - (1 - p_n)q^{i-1},$$

де q — певний коефіцієнт $(0 < q1)$; $p_n = p_1$ — початковий коефіцієнт розрідження потоку. За допомогою цієї моделі прогнозують не тільки інтенсивність виникнення наступної відмови програми, а й параметри потоку відмов.

Описані моделі краще спрацьовують для прогнозування відмов у процесі налагодження та експлуатації програми. Значення параметрів визначають за даними про моменти виникнення відмов, використовуючи метод максимальної правдоподібності.

Експериментальне оцінювання кількості помилок у програмі. Кількість природних помилок у програмі визначають стосовно кількості виявлених природних і штучних помилок, якщо в програму була введена певна кількість однотипних з природними штучних помилок.

Помилки у програму вносять хаотично. Їх кількість відома заздалегідь. Припускають, що ймовірність виявлення помилок, внесених штучно, і ймовірність виявлення помилок, що є в програмі до їх внесення, однакова при реалізації тестування і залежить тільки від кількості помилок. Тестуючи програму і відсортовуючи внесені й наявні помилки, оцінюють початкову кількість помилок у програмі.

Наприклад, у програму внесли s помилок і після цього розпочали тестування. При тестуванні виявили $(n + \nu)$ помилок, де n — кількість виявлених помилок, що були в програмі, а ν — кількість виявлених помилок, що внесені в програму. Тоді їх початкову кількість N у програмі за методом максимальної правдоподібності визначають як:

$$N = \frac{sn}{\nu}. \tag{5.2}$$

Це чітко ілюструють змодельовані ситуації.

1. Якщо в програму внесено 30 помилок і на певний момент тестування з них виявлено 20 власних і 10 внесених помилок, тоді N матиме значення 60.

Припустивши, що в програмі наявно не більше k власних помилок, у неї вносять ще s помилок. Тестують її, поки не виявлять усі внесені помилки, підраховуючи кіль-

кість виявлених власних помилок n. Тоді міру довіри (рівень значущості) C до моделі визначають за формулою:

$$C = \begin{cases} 1 & при \quad n > k, \\ \dfrac{s}{s+k+1} & при \quad n \le k. \end{cases} \qquad (5.3)$$

Міра довіри C — ймовірність того, що модель правильно відхилятиме хибне припущення.

2. Якщо в програмі немає помилок ($k = 0$) і, вніcши в неї 4 помилки, всі їх виявляють, не натрапивши на жодну вихідну помилку, то $C = 0{,}8$. Щоб досягти рівня значущості 0,95, потрібно було б внести в програму 19 помилок. Стверджуючи, що в програмі не більше 3-х вихідних помилок, і, вніcши ще 6, виявляють їх усі, рівень значущості дорівнює 0,6.

Формули (5.2) і (5.3) мають достатню статистичну основу. Це довів американський вчений Х. Мілс. Перша — передбачає кількість помилок, другу — використовують для встановлення довірчого інтервалу прогнозу. Вада цієї формули в тому, що її не використовують для визначення C доти, доки не будуть виявлені всі внесені помилки. А це може бути і в завершальній фазі тестування. Щоб усунути цю ваду, формулу (5.3) модернізовано:

$$C = \begin{cases} 1 & при \quad n > k, \\ \dbinom{s}{j-1} \Big/ \dbinom{s+k+1}{k+j} & при \quad n \le k, \end{cases} \qquad (5.4)$$

де j — кількість виявлених внесених помилок при оцінюванні C, коли $j \le s$. У наведених прикладах C змінюється з 0,6 до 0,33, якщо виявлені 5 із 6-ти внесених помилок при $k = 3$, а $s = 6$. Виходячи з формул (5.3) і (5.4), можна будувати графік у процесі тестування, на якому відображати поточне значення верхньої границі k для певного фіксованого довірчого рівня, наприклад 0,9; 0,95.

Усі внесені помилки слід реєструвати, щоб їх можна було поділити на власні і внесені. Однак природу внесених помилок не розголошують. Головне, щоб вони були типовими. Х. Мілс пропонує інтуїтивну модель, яка не потребує введення штучних помилок. Згідно з нею кількість помилок у програмі оцінюють як $Y = \dfrac{Y_1 Y_2}{Y_{12}}$, де Y_1, Y_2 — кількість помилок, виявлених першим і другим програмістами, які налагоджують незалежно один від одного первісний текст програми. Y_{12} — кількість помилок, виявлених першим і другим програмістами. При цьому первісний текст програми повинен розробляти третій програміст, щоб поставити перших двох в однакові (рівні) умови.

Один з простих інтуїтивних методів прогнозування надійності ПЗ — тестування одних і тих самих програм незалежними групами програмістів, які використовують різні набори тестів. Певний час вони тестують ПЗ паралельно, а потім збирають і порівнюють результати.

Нехай множина $\overline{N}$ відображає невідому повну кількість помилок у програмі, підмножини $\overline{N}_1$ і $\overline{N}_2$ — відповідно кількість помилок, виявлених першою і другою групою, а підмножина $\overline{N}_{12}$ — кількість помилок, виявлених двічі, тобто обома групами. Це відношення показано на рис. 5.3.

Тоді: $\overline{N}_{12} = \overline{N}_1 \cap \overline{N}_2$.

Ефективність тестування програми кожною з груп визначають за формулами:

$$E_1 = \frac{N_1}{N}; \; E_2 = \frac{N_2}{N}.$$

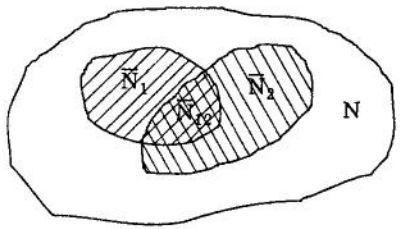


Рис. 5.3. Підмножини помилок, виявлених незалежними групами

Вважаючи, що можливість виявлення помилок для обох груп однакова, кожну підмножину простору $\overline{N}$ розглядають як апроксимацію (лат. approximatio — зближення) всього простору. Це дає змогу зробити висновок, що:

$$E_1 = \frac{N_1}{N} = \frac{N_{12}}{N_2},$$

виконуючи підстановку для N_2, можна отримати:

$$E_1 = \frac{N_{12}}{E_2 \cdot N}$$

або:

$$N = \frac{N_{12}}{E_1 \cdot E_2}.$$

N_{12} відомо, а E_1 і E_2 визначають як $\dfrac{N_{12}}{N_2}$ і $\dfrac{N_{12}}{N_1}$. Маючи ці дані, отримують наближення для N.

Це лише середні оцінювання, що ґрунтуються на інформації, яку використовували для нових методів програмування, наприклад структурного програмування.

Загалом найкраще послуговуватись кількома моделями і об'єднувати їх результати. Дані раніше розроблених підходів використовують для приблизного оцінювання кількості помилок ПЗ у сучасних підходах.

Надійність програмного комплексу. Визначають її як функцію надійності складових частин програмного комплексу. Ця процедура значно полегшується, якщо він побудований за модульним принципом. Мірою надійності програмного модуля вважають імовірність того, що модуль коректно реалізовує задану функцію та коректно передає керування іншому модулю, передбаченому алгоритмом роботи програмного комплексу. Надійності модулів є незалежними величинами. Щодо коректності, то загальний результат роботи програми буде некоректним, якщо хоча б в одному з виконаних прогонів і модулів виявлять помилку. Надійність кожного з модулів вважають визначеною при визначенні надійності програмного комплексу. Тобто, по суті, визначають надійність керування модулями.

Процес керування модулями програми представляють як *марківський процес*. Це значить, що вибір наступного модуля для виконання залежить тільки від того модуля, який виконується в даний момент і не залежить від попереднього. Ймовірності передавання керування між модулями — сталі величини і повністю характеризують спосіб використання програми користувачем.

Структуру керування програмами представляють у вигляді напрямленого графа, де кожна вершина N_i $(i = \overline{1,n})$ відповідає певному програмному модулю, а дуга (N_i, N_j) — можливому передаванню керування від модуля N_i до модуля N_j. Кожній дузі (N_i, N_j) слід співвіднести ймовірність переходу з вершини N_i по дузі (N_i, N_j), а кожній вершині N_i — ймовірність безвідмовної роботи R_i відповідного модуля. Граф програми має вхідну вершину N_1, вихідну вершину N_n та дві додаткові вершини, що відповідають коректному C і помилковому F вихідним результатам.

Вважають, що при появі помилки по дузі (N_i, F) з імовірністю $1 - R_i$ незалежно від правильності наступного оброблення здійснюють перехід у стан F, а при видаванні правильного результату модулем N_i — перехід до виконання модуля N_j з імовірністю $R_i P_{ij}$. Коректному завершенню програми відповідатиме перехід з вихідного стану N_n у стан C з імовірністю R_n. Тоді матрицю перехідних імовірностей P, де її елементи відповідатимуть переходові марківського процесу із стану i в стан j, представляють як:

$$P = \begin{array}{c|cccccccc} & N_1 & N_2 & \dots & N_j & \dots & N_n & C & F \\ N_1 & 0 & R_1P_{12} & \dots & R_1P_{1j} & \dots & R_1P_{1n} & 0 & 1-R_1 \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_i & 0 & R_iP_{i2} & \dots & R_iP_{ij} & \dots & R_iP_{in} & 0 & 1-R_i \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_{n-1} & 0 & R_{n-1}P_{(n-1)2} & \dots & R_{n-1}P_{(n-1)j} & \dots & R_{n-1}P_{(n-1)n} & 0 & 1-R_{n-1} \\ N_n & 0 & 0 & \dots & 0 & \dots & 0 & R_n & 1-R_n \\ C & 0 & 0 & \dots & 0 & \dots & 0 & 1 & 0 \\ F & 0 & 0 & \dots & 0 & \dots & 0 & 0 & 1 \end{array}$$

У такому разі надійність усього програмного комплексу вираховують як імовірність досягнення кінцевого стану C з початкового стану графа. Так, матрицю Q, яку отримують шляхом викреслювання стовпців і рядків, що відповідають кінцевим станам C і F матриці P, зображують так:

$$Q = \begin{array}{c|cccccc} & N_1 & N_2 & \dots & N_j & \dots & N_n \\ N_1 & 0 & R_1P_{12} & \dots & R_1P_{1j} & \dots & R_1P_{1n} \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_i & 0 & R_iP_{i2} & \dots & R_iP_{ij} & \dots & R_iP_{in} \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_{n-1} & 0 & R_{n-1}P_{(n-1)2} & \dots & R_{n-1}P_{(n-1)j} & \dots & R_{n-1}P_{(n-1)n} \\ N_n & 0 & 0 & \dots & 0 & \dots & 0 \end{array}$$

Необхідно визначити P^k для кожного цілого $k > 0$ як k-у степінь P. P^k (i,j) — ймовірність того, що марківський процес за k кроків перейде зі стану i в стан j. Тоді матриця $T = I + P + P^2 + \dots = \sum_{k=0}^{\infty} P^k$ визначає ймовірність переходу з одного стану в інший за довільну кількість кроків.

Якщо S — квадратична матриця розмірності n, то

$$S = I + Q + Q^2 + \dots = \sum_{k=0}^{\infty} Q^k.$$

Якщо матриця $W = I - Q$, згідно з теорією ймовірностей, тоді отримують:

$$S = W^{-1} = (I - Q)^{-1},$$

звідки надійність програмного комплексу визначають як:

$$R = S(\overline{1,n})R_n.$$

Отже, ймовірність безвідмовної роботи програмного комплексу залежить від імовірності безвідмовної роботи модулів і зв'язків між ними.

Помилки програмного забезпечення

У загальному під поняттям «помилка» розуміють неправильність, похибку або ненавмисне спотворення об'єкта чи процесу. При цьому приймають, що правильний стан об'єкта або процесу відомий. Існує поняття «помилка в програмі». Спільне в цих трактуваннях те, що помилку не можна вважати тільки відхиленням від формалізованого еталону. Адже серед помилок ПЗ бувають такі, коли програма відповідає еталону, але порушуються окремі правила побудови програм, не передбачені ним. Якщо програма не виконує того, що користувач від неї очікує, тоді вважають, що в ній є помилка. Це помилка як ПЗ, так і неформалізованих сподівань його користувачів. Відсутність повністю визначеної правильної програми (еталону) не дає змоги локалізувати помилки, безпосередньо порівнявши програму, що діагностують, з програмою без помилок.

Крім поняття «помилки в програмі», існує ще поняття «програмна помилка». Її характеризує багато чинників. Один з них полягає в тому, що програмна помилка виникає тоді, коли програма працює не відповідно зі своїми специфікаціями. Його недолік у тому, що припускають абсолютну коректність специфікацій. Проте це не завжди так. Програма може працювати відповідно зі специфікаціями, але не можна стверджувати, що в ній немає помилок.

З іншого боку, твердять, що помилка в програмі виникає тоді, коли вона працює не відповідно зі специфікаціями при умові, що програму експлуатують у заздалегідь визначених межах. Це твердження є не досить точним, тому що, коли систему використовують випадково, навіть поза заздалегідь установленими межами, вона повинна видавати певні правильні результати. В іншому разі в системі наявна помилка.

Можливе також і таке визначення: помилка наявна тоді, коли робота програми не відповідає документації. Проте й воно має недоліки. Може статися так, що програма працює відповідно до документації, але помилка все одно є, оскільки і в документації, і в програмі існують помилки.

З точки зору замовника вірогідне інше визначення: помилка — це невідповідність роботи програми початковим вимогам замовника-користувача. Недоліком цього визначення є те, що замовник не завжди може достатньо детально викласти вимоги для опису бажаної поведінки програми при всіх можливих умовах.

Помилкою слід вважати таку роботу програми, яка не відповідає очікуванням користувача. Такий підхід цілком його задовольняє, але розробника ПЗ задовольнить лише тоді, коли він точно знатиме, що очікує користувач від системи. Адже він є суб'єктом, і йому властиві помилки. Завдання ускладнюється, коли користувачами системи є декілька осіб, тому що кожен з них може по-своєму вводити дані в систему.

Основні причини відмов ПЗ, що призводять до порушення нормального функціювання: приховані в програмі помилки; спотворення вхідної інформації, що підлягає обробленню; неправильні дії користувача; несправності апаратних засобів, на яких проводять обчислювальний процес. Типи помилок ПЗ та їх відсоток від загальної кількості наведено в табл. 5.1.

Таблиця 5.1

Типи помилок ПЗ

№ п/п	Причини помилки	Частота виникнення, %
1	Неповне або помилкове завдання	28
2	Відхилення від завдання	12
3	Нехтування правилами програмування	10
4	Помилкова вибірка даних	10
5	Помилкова логіка чи послідовність операцій	12
6	Помилкові арифметичні операції	9
7	Нестача часу для розв'язання	4
8	Неправильне оброблення переривань	4
9	Неправильні сталі або вихідні дані	3
10	Неточний запис	8

Характеристики помилок ПЗ використовують для вибору найефективніших методів технології програмування і супроводжень програм із заданими показниками якості. Статистика прояву помилок може слугувати орієнтиром

для розробників ПЗ, щоб правильно розподілити операції під час процесу налагодження. На етапі проектування ПЗ характеристики помилок допомагають оцінювати реальний стан розробленого пристрою, планувати трудомісткість і тривалість робіт до завершення його розроблення, здійснювати відбір показників складності компонентів ПЗ, розраховувати ефективність засобів оперативного захисту від невиявлених помилок, необхідні ресурси комп'ютерів, у тому числі і продуктивні, враховуючи витрати на усунення помилок та ін.

Спотворення в тексті програм називають *первинними помилками*. Однак помилку, як правило, виявляють за її вторинними проявами — спотворенням вихідних результатів при виконанні програми. Таку помилку називають *вторинною*. Її локалізація і усунення вимагають виконання відповідних операцій.

Залежно від складності первинні помилки ПЗ поділяють на такі типи:

— технологічні помилки підготовки машинних носіїв і документації, а також виведення і введення програм у комп'ютер та його засоби відображення;

— алгоритмічні помилки, пов'язані з неповним формуванням необхідних умов розв'язання і некоректною постановкою завдання;

— системні помилки, зумовлені відхиленням функціювання ПЗ у реальній системі від очікуваних під час проектування;

— програмні помилки, пов'язані з некоректною постановкою завдання.

На рис. 5.4 зображена схема зв'язків первинних помилок у ПЗ із методами їх виявлення.

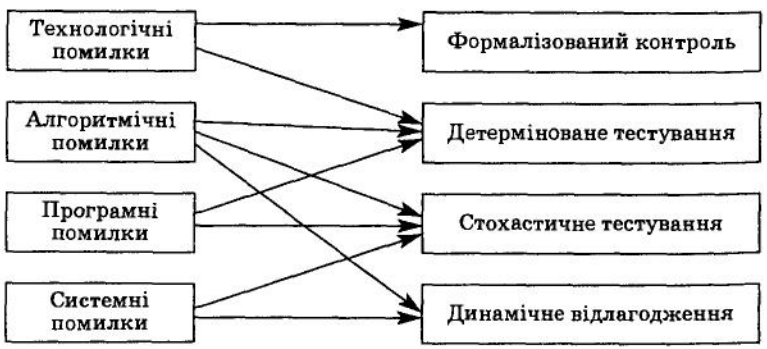


Рис. 5.4. Первинні помилки ПЗ і методи їх виявлення

Наведені помилки також розрізняють за частотою появи і методами їх виявлення на різних етапах проектування ПЗ.

Технологічні помилки документації і запису програм у пам'ять комп'ютера. Виявляють їх автоматичними методами при налагодженні. Вони становлять приблизно 5—10 % загальної кількості помилок, що виявляють під час цього процесу. Навіть за дуже високого рівня функційної налагодженості ПЗ трапляються окремі технологічні помилки.

Алгоритмічні помилки. Їх важко виявити за допомогою методів автоматичного контролю. Це зумовлено відсутністю повністю формалізованої постановки завдання і точної специфікації. Зокрема, коли в специфікаціях не висвітлюють усі умови, необхідні для отримання правильного результату. Такі помилки трапляються найчастіше і становлять до 70 % усіх алгоритмічних помилок та до 30 % загальної кількості помилок на початкових етапах проектування. До алгоритмічних помилок належать також помилки спрягання програмних модулів і функційних груп програм. Їх зараховують до помилок некоректної постановки завдання.

Частина помилок є наслідком прорахунків у визначенні ресурсів обчислювальних систем для вирішення певного завдання. Це стосується як нестачі, так і перевищення виділених ресурсів. Найвідчутнішими є алгоритмічні помилки, зумовлені неправильним розрахунком часу реалізації програмних модулів і програм у цілому. Вони знижують достовірність визначення часу оброблення інформації за різними технологічними маршрутами.

Системні помилки. Зумовлені неповною інформацією про реальні процеси приймання, оброблення і видавання даних у комп'ютерних системах. Адже на практиці не завжди є змога досить точно визначити й описати згадані процеси, не досліджуючи заздалегідь функціювання системи або її модулів у взаємодії із зовнішніми користувачами. Крім того, на початковій стадії проектування інколи не вдається точно визначити цільове завдання всієї системи та основних модулів програми. Здебільшого його уточнюють у процесі подальшого проектування і налагодження.

Відсоток системних помилок зростає на завершальних етапах комплексного налагодження системи. Водночас за автономного налагодження він порівняно невеликий — до 10 %. Найбільшу кількість (до 80 % усіх помилок) становлять системні помилки, що виявляють на етапі експлуатації. Виправлення однієї помилки у програмі зумовлює ко-

ректування до 25 команд, що свідчить про велику трудомісткість цього процесу.

Також важко прогнозувати і час, необхідний для виявлення та усунення системних помилок. Під час цього процесу при налагодженні складається враження, що всі помилки виявлені й усунуті, тому після цього різко знижується інтенсивність їх виявлення, а інколи воно припиняється взагалі. Проте під час серійного випуску комп'ютерних систем розширюються умови експлуатації й інтенсивність помилок зростає. Протягом певного часу зростає і сумарна інтенсивність їх виявлення. Це дає змогу усунути багато помилок, зумовлених умовами експлуатації систем, і збільшує тривалість між їх проявами у процесі експлуатації.

Програмні помилки. Серед цього типу виокремлюють помилки: зумовлені обмеженими можливостями програми (неможливість отримання результатів за прийнятний час або при заданих обмеженнях за обсягом обчислень) і логічні помилки (отримання неправильних результатів, незважаючи на час і обсяг обчислень).

Всебічний аналіз помилок у програмах можливий за кількості даних про відмови програми та причини їх виникнення. Вони є основою для побудови математичних моделей надійності ПЗ, оцінювання надійності програм з метою її підвищення і забезпечення показників надійності, згідно з вимогами і технічними завданнями (ТЗ).

На різних етапах життєвого циклу проявляються помилки таких типів: при постановці завдання; у проектуванні; помилки програмування; при записі на носії даних; помилки обчислень; логічні помилки; введення-виведення; помилки спрягань; маніпулювання даними; ініціалізації бази даних; помилки в документації; інші помилки (помилки у визначенні глобальних змінних; порушення технічних вимог; помилки оператора; нерозпізнані помилки).

Помилки при постановці завдання. Постановку завдання на розроблення ПЗ оформлюють як ТЗ і технічні умови (ТУ). Правильно сформульоване завдання повинно врахувати всі вимоги користувачів. Для цього потрібно точно встановити очікувані діапазони вихідних даних і одержуваних результатів. Нерозуміння завдання матиме наслідком помилки при постановці завдання, що позбавить сенсу подальше проектування ПЗ. Вимоги до якості програм є невід'ємною складовою загальних технічних вимог. Помилки ТЗ, що найчастіше трапляються в програмах, наведені в табл. 5.2.

Таблиця 5.2

Типи помилок ТЗ

№ п/п	Причини помилки	Частота виникнення, %
1	Помилки в числових значеннях	12
2	Недостатні вимоги до точності	4
3	Помилкові символи або знаки	2
4	Помилки оформлення	15
5	Неправильний опис апаратури	2
6	Неповні чи неточні основи розроблення	52
7	Двозначність вимог	13

Помилки проектування. Це помилки при виборі параметрів, технічні помилки в проектній документації, недосконалість обраних математичних методів розв'язання задач, неузгодженість даних у часі, неврахування кореляції між окремими змінними та ін. Більшість із них виникає через неадекватність математичної моделі реальному процесу. Інколи вони належать до алгоритмічних помилок.

Помилки програмування. Сюди належать помилки вибору чисельних методів розв'язання задач, кодування чисел і команд (синтаксичні помилки), розподілу пам'яті, трактування алгоритмічних конструкцій (семантичні помилки), спрягання програмних модулів і програм, реалізації умовних пропонувань, в описі даних, у документації. Навіть сучасні засоби відлагодження і перевіряння програм не завжди забезпечують безпомилкове програмування.

Помилки при записі на носії даних. Суть цих помилок полягає в спотворенні двійкових кодів.

З огляду на види робіт зі створення і експлуатації ПЗ серед іншої частини наведених у загальному переліку помилок виокремлюють декілька типів.

Помилки обчислень. Вони містяться в обраних алгоритмах розв'язання задач. Це — неправильна рівність чи співвідношення, неадекватність розроблених математичних моделей реальним процесам, неправильно розрахований логічний або фізичний номер елемента.

Логічні помилки. Здебільшого мають алгоритмічний характер. Серед них ті, що призводять до зациклення програ-

ми, неправильне передавання керування, неправильна послідовність дій та ін.

Помилки введення-виведення. Вони призводять до введення непередбачених або дублюючих елементів даних, зокрема неправильних форматів даних, та ін. Як правило, містяться в модулях введення-виведення (драйверах) і в операторах інтерфейсу.

Помилки спрягань. Можуть бути в міжпрограмних інтерфейсах програмних додатків, спрягань прикладних програм з операційною системою та іншими базовими програмами ПЗ, інтерфейсах користувачів. Це помилки в даних, які оператор вводить вручну, в спряганнях з базами даних через невідповідність структури бази даних її опису.

Помилки маніпулювання даними. Їх виявляють в операціях зчитування, запису, пересилання, зберігання і зміни даних. Це — присвоєння вхідним даним неправильних початкових значень, втрата даних, помилки сортування, генерування зайвих елементів інформації, зокрема таблиць, масивів тощо, пошук даних у неіснуючих таблицях і записах.

Помилки ініціалізації бази даних. Вони трапляються при описі даних або в даних, що запитуються в номінальних, максимальних і мінімальних значеннях, у текстах повідомлень про помилки та ін.

Помилки в документації. Це помилки в описах режимів роботи, можливостей програм, вихідних форматів, формату карт та ін.

Інші помилки. Вони трапляються у визначенні глобальних змінних, порушенні технічних вимог, також це можуть бути помилки оператора, нерозпізнані помилки.

Частота прояву описаних помилок різна (табл. 5.3).

Таблиця 5.3

Частота прояву помилок

№ п/п	Причини помилки	Частота прояву, %
1	Помилки обчислень	7
2	Логічні помилки	22
3	Помилки введення-виведення даних	10
4	Помилки спрягань	10
5	Помилки маніпулювання даними	15
6	Помилки ініціалізації бази даних	6
7	Помилки в документації	8
8	Інші помилки	22

Симптомами найпоширеніших помилок є: переповнення розрядної сітки — 30,4 %, неправильне передавання керування — 16,4, несумісність з базами даних — 14,5, несумісність програм за даними, що пересилаються, — 9, невиконання програмою додаткових функцій — 4,9, несумісність програм — 7, інші симптоми — 18,2 %.

Причинами найпоширеніших помилок у ПЗ є:

— передчасне (аварійне) закінчення виконання програми;

— збільшення часу виконання програми;

— зациклення на виконанні певної послідовності команд однієї з програм;

— часткова або повна втрата даних, необхідних для успішного виконання завдань, що вирішуються;

— порушення послідовності виклику окремих програм, що призводить або до пропуску необхідних програм, або до непередбачених їх викликів;

— спотворення окремих елементів даних, зокрема вхідних, проміжних і вихідних, у результаті спотвореної первинної інформації.

Суттєву роль відіграють несправності апаратури, яку використовують для обчислювального процесу. Вони відчутно впливають на надійність ПЗ. Відмови чи збої у роботі апаратури призводять до порушення обчислювального процесу, зокрема до спотворення даних і кодів програм у пам'яті.

Щодо виникнення помилок у життєвому циклі ПЗ, то їх класифікують у часі за двома найважливішими фазами: перша — специфікація, що ґрунтується на ТЗ, і друга — реалізація.

1. Фаза специфікації. Вона охоплює помилки: «багатозначність специфікації» і прості «помилки специфікації». Їх спричинюють:

— помилкове чи коректне, але не повне формулювання вимог користувача;

— неможливість сформулювати завдання однозначно і точно формалізовано;

— помилкове або поверхове розуміння поставленого завдання.

Помилки специфікації є критичними, їх важко виявляти і коректувати.

2. Фаза реалізації. Під час цієї фази проводять послідовні кроки доводки. Помилки програмних кодів дають змогу попередити появу помилок реалізації. Останні поділяють на такі класи:

— проектні помилки;
— помилки структури даних;
— алгоритмічні помилки;
— логічні помилки;
— помилки, зумовлені особливостями мов програмування.

(Програмні помилки сучасних персональних комп'ютерів подано в Додатку).

Відповідний рівень надійності при розробленні ПЗ щодо помилок забезпечує дотримання чотирьох основних принципів.

1. Уникнення помилок. Описує засоби і методи, виконання яких мінімізує помилки, що виникають у процесі створення ПЗ;

2. Виявлення помилок. Базується на засобах і методах, які забезпечують виявлення помилок у програмах, які розробляють.

3. Виправлення помилок. Здійснюють на основі конструювання пристрою і методології використання функцій, що коректують виправлені помилки чи усувають їх дії.

4. Допущення помилок. Забезпечують засобами і методами, що дають змогу виконувати задані функції при наявності помилок.

1. Засоби і методи уникнення помилок. Сюди належать ті, які попереджують появу помилок у програмах. Серед них виокремлюють такі засоби і методи:

— мінімізації складності ПЗ як основної причини появи помилок трансляції;
— досягнення точності процесу трансляції;
— удосконалення інформаційних зв'язків розробників;
— негайного виявлення і усунення помилок трансляції після кожного її кроку, а не після її повного закінчення.

Уникнення помилок є оптимальним підходом у досягненні надійності ПЗ, яке розробляють. Однак практика свідчить, що навіть старанне розроблення програм не убезпечує від помилок.

2. Засоби і методи виявлення помилок. Вони є різними, однак їх можна звести до таких узагальнених процедур:

— перевіряння ознак кожного вхідного елемента даних. Якщо вхідні дані є цифровими, то в цьому треба переконатися, а також перевірити, чи має число відповідний розмір і знак;
— перевіряння значень вхідних даних на відомі обмеження;

— повнота перевіряння. Якщо для числа задані межі його значення (верхня і нижня), то варто перевірити, як вони дотримуються;
— перевіряння надлишкових вхідних даних на сумісність;
— передбачення надлишковості вхідних даних, якщо вона відсутня;
— порівняння вхідних даних з повними внутрішніми даними.

Розроблення засобів для виявлення помилок базується на таких принципах:

— взаємної підозри (кожен модуль передбачає, що всі інші зв'язані з ним модулі містять помилки. Це вимагає перевіряти інформацію, яка надходить до цього модуля від інших);
— негайного виявлення помилок (до моменту реалізації функцій модуля);
— надлишковості (всі заходи щодо виявлення помилок ґрунтуються на певній явній або неявній надлишковості).

Існують методи пасивного і активного виявлення помилок у програмах.

Пасивний метод. Він полягає в тому, що помилки виявляють під час виконання запланованих програмних функцій (робочих алгоритмів). При такому підході їх можна виявити тільки тоді, коли здійснюють контрольну функцію.

Активний метод. Може бути реалізований за рахунок спроектованих програмних функцій, які активно використовують програмну систему з метою пошуку помилок. Цю процедуру здійснюють через монітор виявлення, що проводить паралельний процес періодичного сканування даних. В окремих (екстремальних) ситуаціях використовують монітор, який виконує діагностичні тести системи.

3. Засоби і методи виправлення помилок. Це одні з основних і специфічних засобів підвищення надійності ПЗ. Тут проводять коректування даних чи усувають джерела їх спотворення. Засоби сучасних ПК виявляють несправні апаратні модулі й компоненти, що дає змогу замінити їх ідентичними резервними. Такий підхід не завжди прийнятний щодо ПЗ, оскільки наявність помилки в конкретному програмному модулі свідчить, що вона є і в резервному. Тому виправлення наслідків помилок ПЗ потребує від розробника передбачення можливих типів

руйнувань, а також розроблення спеціальних програмних функцій, що виправляють ці наслідки дій помилок. Якщо функції коригування помилок не достатньо потужні, то під час проектування ПЗ переважають засоби уникнення помилок.

4. Засоби і методи допущення помилок. Вони повинні забезпечувати функціювання ПЗ, якщо в ньому є помилки. До їх складу входять динамічна надлишковість, допоміжні методи та ізоляція помилок.

Динамічна надлишковість. Вона полягає в тому, що одні й ті самі дані обробляють паралельно кількома ідентичними пристроями (процесорами, комп'ютерами), а одержані результати зіставляють. Коли вони збігаються, їх вважають правильними. Якщо результат основного модуля вважають некоректним, для забезпечення динамічної надлишковості використовують додаткові варіанти програмних модулів. За такої ситуації система автоматично викликає запасний модуль. Якщо ситуація повторюється, викликають наступний модуль-дублер і т. д.

Допоміжні методи. Їх застосовують, маючи на меті коректно закінчити роботу системи. Це можуть бути функції завантаження і відпрацювання допоміжного модуля після контролю і виявлення помилки, щоб гарантовано завершити перевіряння всіх контрольованих системою процесів, або функції, необхідні в операційних системах для попередження користувачів про збій і можливе спотворення даних.

Ізоляція помилок. Полягає в спробі ізолювати помилки мінімальною частиною програмної системи, щоб уся система при виникненні помилки не втратила своєї працездатності. Наприклад, помилка в прикладному додатку, що опрацьовує операційна система, позначиться тільки на роботі цієї програми і не вплине на операційну систему та інші прикладні програми.

Тільки сучасні засоби і методи допуску помилок прийнятні для більшості програмних систем.

Шляхи забезпечення надійності ПЗ

Основні фактори, що визначають надійність функціювання ПЗ, об'єднують у три групи:

1. Фактори, які безпосередньо визначають надійність програм. До них належать:

— особливості, пов'язані з користувачами результатів виконання програм: вимоги до показників надійності; інерція користувачів, час, необхідний для відклику на вхідні дані;

— спотворення початкових даних, що надходять від людини-оператора або телекомунікаційними каналами зв'язку, даних у процесі накопичення і зберігання в комп'ютері;

— помилки в програмах та їх прояв: статистичні характеристики помилок і спотворень програм, масивів даних та обчислювального процесу.

2. Методи проектування коректних програм. Йдеться про:

— структурне проектування програм і даних: проектування програмних модулів та їх взаємодії, структурування даних;

— тестування програм: детерміноване, статистичне і динамічне тестування, контроль пропускної здатності в реальному часі.

3. Методи контролю і забезпечення надійності програм:

— методи використання часової, інформаційної та програмної надлишковості;

— методи контролю програм, даних і обчислювального процесу: передпусковий і оперативний контроль;

— методи програмного відновлення: відновлення тестів програм, виправлення даних, коректування обчислювального процесу;

— методи випробування на надійність: випробування в нормальних умовах експлуатації, форсовані випробування, розрахунково-експериментальні методи визначення надійності;

— методи забезпечення надійності при супроводженні: забезпечення зберігання програм еталонних версій та коректності внесення змін і розвитку версій.

Визначальне значення для забезпечення надійності програм має третя група факторів. Для її підвищення і захисту інформації програмно-алгоритмічними методами використовують часову, інформаційну та програмну надлишковість.

Часова надлишковість. Це використання певної частини апаратних засобів і продуктивності ЕОМ, зокрема ПК, для контролю виконання програм і відновлення обчислювального процесу. Передбачається певний резерв продуктивності, необхідний для контролю та підвищення надійності функціювання ПЗ. Резерв часу використовують для контролю і виявлення спотворень, їх діагностування і

прийняття рішень щодо відновлення обчислювального процесу або інформації, реалізації операцій відновлення.

Інформаційна надлишковість. Вона полягає у дублюванні накопичених початкових і проміжних даних. Її використовують для забезпечення достовірності даних, які потребують значного часу для відновлення чи найбільше впливають на нормальне функціювання ПЗ. Вони характеризують певні інтегральні відомості про зовнішній керований процес, і в разі їх руйнування можуть надовго припинитися керування цим процесом та відповідне оброблення інформації. Інформаційна надлишковість дає змогу не тільки виявляти спотворення даних, а й усувати помилки.

Програмна надлишковість. Вона контролює і забезпечує достовірність найважливіших рішень з оброблення інформації та керування. Її сутність полягає в застосуванні кількох варіантів програм, що відрізняються методами розв'язання певного завдання або програмною реалізацією одного й того самого методу. Це дає змогу виключати спотворення результатів, зумовлені програмними помилками чи збоями ЕОМ.

Часто введенням надлишковості для забезпечення надійності ПЗ нехтують, що знижує показники надійності програм, які розробляють.

Небажані наслідки спотворень обчислювального процесу і даних зумовлюють проведення контролю функціювання програм. Кожний його метод орієнтований на виявлення певного типу наслідків спотворень чи їх певної сукупності. З огляду на це застосовують ієрархічні підходи, на яких ґрунтується використання послідовності методів, що дає змогу досягти заданої глибини пошуку спотворень у програмах.

Щодо процесу експлуатації, то контроль надійності функціювання ПЗ проводять протягом трьох етапів, відповідно до яких він і має назви.

1. Профілактичний контроль. Здійснюють у неробочому режимі системи керування чи оброблення інформації під час проведення регламентних профілактичних робіт.

2. Передпусковий контроль. Проводять при підготовці до ввімкнення нормального робочого режиму функціювання ПЗ.

3. Оперативний контроль. Здійснюють у процесі розв'язання системою основних функційних завдань у нормальному робочому режимі.

1. *Профілактичний контроль.* На цьому етапі розв'язування завдань у нормальному робочому режимі призупиняють, детально аналізують характеристики і виконують значний обсяг профілактичних і відновлювальних робіт. При цьому проводять аналіз даних про зареєстровані спотворення інформації та обчислювального процесу, що накопичилися під час робочого функціювання. За цими даними і діагностичними засобами встановлюють причини спотворень, які надалі намагаються усунути. За допомогою контрольної і діагностичної апаратури та діагностичних програм джерела ненадійності автоматизовано локалізують. Після цього проводять ремонт апаратури чи виправлення програм. Заміна версії програм після усунення в них помилок, забезпечення достовірності й збережуваності кожної версії дають змогу підвищити надійність функціювання ПЗ.

2. *Передпусковий контроль.* За обмежений час здійснюють необхідний мінімум перевірок надійності функціювання ПЗ для включення нормального режиму роботи. Визначають перелік контрольних операцій та послідовність їх проведення. Головне завдання полягає в перевірці збережуваності програм і масивів даних, необхідних для початкового запуску системи. Передпусковий контроль також проводять без оброблення реальної інформації і видавання її зовнішнім користувачам.

3. *Оперативний контроль.* Його здійснюють у процесі розв'язання системою основних функційних завдань. Крім того, проводять максимально швидке автоматизоване відновлення при будь-яких спотвореннях даних чи обчислювального процесу. Методи оперативного контролю, у свою чергу, поділяють на такі методи:

— контролю стану і збережуваності програм у різних видах пам'яті;

— контролю динаміки проходження програм, збережуваності зв'язків і взаємодії модулів при робочому функціюванні ПЗ;

— контролю стану і даних у пам'яті ПК.

Хоча етапи контролю ПЗ доповнюють один одного, але найбільший вплив на надійність мають оперативний і передпусковий контроль.

Метод оперативного відновлення ПЗ передбачає певні заходи для усунення наслідків спотворень. Зокрема, здійснюють таке:

— ігнорують виявлене спотворення, коли воно слабо впливає на вихідні результати та сам процес оброблення інформації;

— повторюють розв'язання функційного завдання чи виконання програм при тих самих вхідних даних;

— виключають повідомлення з оброблення за причиною спотворення або важкості продовження обчислювального процесу;

— короткочасно призупиняють розв'язання завдання до оновлення вихідних даних чи усунення джерела спотворювання;

— переналагоджують режим роботи ПЗ з огляду на появу чинників спотворень;

— переходять на резервний комп'ютер (модуль комп'ютера) з накопиченою інформацією про стан зовнішніх процесів або відновлення інформації за рахунок її дублювання;

— відновлюють процес оброблення інформації чи процес керування з режиму початкового пуску.

Основна мета випробувань ПЗ на надійність полягає в тому, щоб розроблені програми відповідали вимогам технічного завдання та іншим документам, зокрема ДСТУ. Базовий документ — план проведення серії детермінованих і статистичних експериментів. Для цього визначають обсяг і послідовність кожного експерименту за критеріями мінімізації та погодженої із замовником достовірності одержуваних результатів.

Програма випробувань має містити такі розділи:

— об'єкт випробувань, його призначення і перелік основних документів, що визначили розроблення цього об'єкта;

— мету випробувань з визначеними основними вимогами ТЗ, які підлягають перевірянню, та обмеженнями на проведення випробувань;

— програму випробувань, що містить характеристику загальної перевірки складу спроектованого ПЗ відповідно до технічних завдань і повноти технічної документації, програму спеціальної перевірки показників якості з усіх розділів технічного завдання;

— методики випробувань, що однозначно визначають усі показники якості, які перевіряють, умови їх перевірки, засоби, що застосовують для випробувань, методики оброблення і оцінювання результатів з кожного розділу програми випробувань.

Великий обсяг перевірок та їх складність зумовлюють використання засобів автоматизації на різних етапах випробувань. Їх поділяють на такі групи:

— засоби автоматизації випробувань дослідного зразка ПЗ відповідно до всіх вимог ТЗ;

— засоби автоматизації контролю фірми-виробника серійних зразків ПЗ відповідно до затверджених технічних умов;

— засоби автоматизації контролю функціювання кожного комплексу ПЗ у процесі його експлуатації, тобто оброблення інформації і розв'язування завдань керування реальними об'єктами.

Перші дві системи майже завжди є унікальними, остання є в кожному зразку.

Методи теорії надійності, що дають змогу визначати характеристики надійності складних систем, застосовують для статистичної перевірки надійності функціювання ПЗ. Серед них виокремлюють такі основні групи: прямі експериментальні методи визначення показників надійності систем в умовах нормального функціювання; форсовані методи випробувань на надійність реальних систем; розрахунково-експериментальні методи, коли вихідні дані для розрахунків одержують експериментально, а показники надійності систем розраховують з використанням згаданих даних.

Прямі експериментальні методи визначення надійності ПЗ. Їх інколи важко застосувати через необхідність проводити тривалі напрацювання (тисячі або десятки тисяч годин). Зменшення цього часу при випробуваннях призводить до отримання малих вибірок зареєстрованих відмов. Тому, щоб підвищити достовірність випробувань, враховують дані про виявлені помилки на завершальних етапах відлагодження програм, а також обробляють дані про виявлені помилки за допомогою математичних моделей. Це дає змогу коректніше прогнозувати показники надійності.

Форсовані методи випробувань на надійність реальних систем. Здійснюють їх шляхом підвищення інтенсивності спотворень вихідних даних, а також спеціальним понаднормовим збільшенням завантаження ПЗ. Планування цих випробувань повинно передбачати перерахунок отриманих показників надійності на нормальні умови експлуатації ПЗ. Один з особливих видів форсованих випробувань — перевірка ефективності засобів контролю і відновлення програм, даних і обчислювального процесу, зокрема якості функціювання засобів підвищення надійності. У такому разі оцінювання інтегральних показників відходить на задній план.

Розрахунково-експериментальні методи аналізу надійності ПЗ. Вони більш обмежені, ніж для оцінювання

надійності апаратури. Це зумовлено насамперед неоднорідністю характеристик надійності програмних модулів, груп програм, масивів даних та ін. Однак у певних випадках їх оцінюють розрахунковим шляхом. Наприклад, поєднання експериментальних і аналітичних методів оцінювання надійності використовують для визначення пропускної здатності комплексу програм на конкретному комп'ютері та впливу перевантаження на надійність його функціювання.

Після завершення випробування і передавання ПЗ замовнику його експлуатують і тиражують протягом одного-двох десятків років. На цьому етапі можливі спотворення програм у процесі їх зберігання та регулярної експлуатації. Крім того, з огляду на зміну вхідних даних може виникнути необхідність внесення змін у програму. Тому навіть одне незначне низькоякісне коректування може спричинити істотне зниження напрацювання на відмову. Щоб зберегти і поліпшити показники надійності ПЗ під час його тривалої експлуатації, необхідно чітко регламентувати передавання програм користувачам. Зміни в ПЗ вводять групами. Це формує чергову версію ПЗ зі зміненими характеристиками.

Серед версій ПЗ виокремлюють еталонні й користувальницькі.

Еталонні версії. Їх доопрацьовують і модернізують основні розробники ПЗ або спеціалісти з експлуатації. Вони мають скоректовану технічну документацію, що відповідає програмам, і точний перелік усіх змін, введених у дану версію порівняно з попередньою. Після того як розроблено наступну еталонну версію, її випробовують і перевіряють на працездатність.

Користувальницькі версії або *версії конкретної системи.* Їх формують за інструкціями і правилами, що є в експлуатаційній документації з еталонних версій. Зміни в них обмежені й стосуються окремих компонентів ПЗ. Щоб перевірити, чи коректно їх виконують, додають методики перевірки та правила розроблення контролюючих тестів.

Відносна легкість зміни окремих програм створює в користувачів, особливо недосвідчених, враження, що програму можна самостійно поліпшити, виходячи за дозволені інструкціями межі. Це, як правило, призводить до негативних наслідків. Такі зміни повинні бути організаційно заблоковані правилами впровадження ПЗ і технічно обмежені передаванням технічної документації. Крім того, доцільно зменшувати доступ широкого кола користувачів,

оскільки технічна документація містить детальні відомості про зміст і логіку функціювання програм.

Загальними способами забезпечення і підвищення надійності ПЗ є:

— розроблення і вибір алгоритмів функціювання ПЗ, нечутливих до порушень обчислювального процесу (використання алгоритмічної надлишковості);

— удосконалення технології програмування;

— контроль і тестування програм з наступною їх корекцією;

— резервування програм та інші методи введення структурної надлишковості.

Застосування конкретного методу надійності залежить від стану налагодження програми, але на всіх етапах слід дотримуватись перерахованих вимог. Головним способом підвищення надійності ПЗ, як зазначалось раніше, є його діагностування.

5.2. Методологія діагностування програмного забезпечення

Методологія діагностування ПЗ пояснює суть методів і способів тестування програм, методики побудови тестів, процедур, які доводять програми до робочого стану. Вона визначає також послідовність видів тестування.

Загальні відомості діагностування ПЗ

Діагностування як процес виявлення помилок у ПЗ є одним з основних методів підвищення його надійності. Здійснюють діагностування на всіх етапах життєвого циклу ПЗ, а саме на етапі планування, проектування і кодування.

Етап планування. Тут розглядають не самі програми, а аналізують і оцінюють вимоги до ПЗ як продукту, його функційних можливостей і характеристик. При цьому спеціалісти намагаються з'ясувати такі аспекти:

— адекватність вищезгаданих вимог. Чи саме такий продукт має бути створено;

— повноту вимог. Чи не пропущено корисних або життєво необхідних функцій, і чи не можна послабити певні з перерахованих вимог;

— сумісність вимог між собою;
— можливість їх виконання;
— сенс вимог до якості продукту чи його вартості;
— можливість тестування ПЗ, щоб з'ясувати, чи відповідає проектна організація вимогам до програмного продукту.

Етап проектування. На цій стадії також ще немає коду програми. Тому діагностують лише ідеї, але вже формалізовані й описані детальніше, ніж на попередньому етапі. Складається чіткіше уявлення про роботу майбутньої програмної системи. Спеціалісти з тестування беруть участь у роботі з метою складання майбутніх тестів. Основну увагу проектувальники приділяють таким чинникам:

— чи дійсно на основі цього проекту буде створено компактний, ефективний, легкотестований, сприятливий для модернізації і супроводження програмний продукт;
— чи відповідає проект вимогам, вказаним у документації на етапі планування;
— чи проект повний, тобто чи описує всі взаємозв'язки і передавання даних між модулями, умови роботи кожного модуля та їх реалізацію;
— чи належно описана в проекті підсистема опрацювання помилок.

Етап кодування. Під час цього етапу програміст пише програму і сам її тестує, використовуючи технологію тестування «скляної («білої») скриньки» або протилежну їй технологію «чорної скриньки».

— При тестуванні «скляної скриньки» тестувальник розробляє тести на основі вихідного коду, до якого він має повний доступ. У такому разі тестування «скляної скриньки» є частиною процесу програмування.

— При тестуванні «чорної скриньки» програму розглядають як об'єкт, внутрішня структура якого невідома. Спеціаліст-тестувальник вводить дані в програму, яку тестують, й аналізує результат. Як саме працює програма, його не цікавить. Дані, які вводять у програму, добирають так, що при їх подаванні помилки тестованої програми повинні проявитися з найбільшою ймовірністю.

Виправлення однієї помилки може призвести до появи інших. Проте інколи програмісти, виявивши первісну помилку, не роблять повторного тестування, що знижує надійність ПЗ. Тому доцільно тестувати розроблювану програму кілька, а іноді й багато разів. Досвідчені спеціалісти не приймають нову версію програми доти, доки попередня не буде досконало і ґрунтовно протестована. Такий підхід до тестування чергової версії зі складанням підсумкового

звіту про всі відомі проблеми і знайдені помилки називають ***повним циклом***.

В окремих випадках обмежуються двома циклами тестування. На першому — виявляють усі помилки, а на другому — переконуються, що всі вони виправлені. Нерідко проводять 8—10 циклів тестування ПЗ.

Це дає підставу розглядати діагностування ПЗ як послідовність перевірянь з відповідними цілями, яка охоплює:

— тестовий контроль кожного окремого модуля — з метою встановлення відмінності між логічною схемою модуля і його інтерфейсу із зовнішніми специфікаціями системи;
— функційне тестування — з метою виявлення суперечностей між розроблюваною програмою і функціями, визначеними зовнішніми специфікаціями для реалізації в системі;
— системне тестування — з метою виявлення суперечностей між системою і цілями, визначеними користувачем для її створення;
— синтаксичний і семантичний контроль кодування модулів — цю процедуру проводять при їх трансляції в машинні програми з метою виявлення синтаксичних і семантичних помилок у кодах програми.
— випробування ПЗ — з метою перевірки працездатності програм.

При створенні ПЗ до 50 % коштів, призначених для розроблення програм, витрачають на діагностування. Однак навіть за таких витрат програми можуть бути не досить надійними і мати значну кількість помилок.

Особливості процесу тестування

Суть тестування полягає у перевірці роботи програм за результатами їх виконання зі спеціально підібраними наборами вихідних даних (тестами). Тобто це метод виявлення помилок у програмах шляхом опрацювання тестових даних і порівняння отриманих результатів з розрахунковими. Тестування є одним з методів діагностування.

Діагностування ПЗ здійснюють також і методом верифікації. Верифікація програми є аналітичним методом перевірки процесів і певних загальнозначущих умов, які можна вивести з характеру поведінки програм. Полягає вона в доведенні за допомогою математичних методів правильності програми. Для цього програму розглядають як

послідовність простих суджень, доведення яких порівняно просте. Однак автоматизація процесу верифікації щодо доведення навіть простих суджень є досить складною і не завжди можлива. Для цього потрібні висококваліфіковані спеціалісти, інакше можливі помилки в доведеннях попри їх вдавану строгість. Тому найпоширенішим методом діагностування ПЗ є тестування програм, завдання якого підтвердити правильність програми і відсутність у ній помилок. Проте відсутність помилок у програмі продемонструвати неможливо. Тому тестування, пов'язане тільки з намірами знайти помилки, не дає певності, що їх у програмі немає. Тому краще розглядати його як процес виконання програми з наміром знайти помилки.

Залежно від бачення суті тестування програміст-тестувальник або обирає простіші тести, які з високою ймовірністю виконуватимуться правильно, або готує такі, що дають змогу відшукати якомога більше помилок у програмі. Якщо тестувальник ставить за мету продемонструвати відсутність помилок, то він знайде їх менше, ніж намагаючись показати наявність помилок.

У проектуванні тестів найпоширенішими є такі підходи: побудова тесту на основі дослідження зовнішніх специфікацій чи специфікацій спрягання програм або модулів, які тестують; проектування тесту на основі вивчення логіки програми.

Побудова тесту на основі дослідження зовнішніх специфікацій чи специфікацій спрягання програм або модулів, які тестують. Програму, що тестують, розглядають як «чорну скриньку». Тестувальника не цікавить, як вона детально виглядає і чи виконує він при тестуванні всі команди і гілки програми. Його задовольняє те, що програма поводить себе, як зазначено в специфікаціях. При цьому головна мета — подати всі можливі комбінації на входи і перевірити правильність реакцій на виходах.

Проектування тесту на основі вивчення логіки програми. Тестувальник намагається підготувати такі тести, щоб кожна команда і команда умовного переходу в кожному з напрямів виконувались хоча б один раз. Його головна мета — перевірити всі гілки алгоритму. При цьому тестувальника не цікавлять специфікації.

Обидва підходи не є оптимальними, хоч перший з них прийнятніший. Проте для кількох входів його не можна застосувати через досить велику кількість (мільярди і більше) тестів для кожної послідовності й комбінації всіх вхідних даних.

Високоякісне тестування полягає в доборі певної кількості тестів з максимальною віддачею. Тобто тест повинен показати невиявлену раніше помилку за мінімальний час і з мінімальною вартістю підготовлення, виконання і перевірки результатів тестування.

Ще менш досконалим є другий підхід, згідно з яким тестують усі шляхи. Цього не можна досягти навіть для порівняно простих програм. Крім того, тестування всіх шляхів ще не гарантує відповідності програми її специфікаціям. Це ілюструє модуль, який складається з циклу, що виконується від 0 до 10-ти разів, за ним — розгалуження, а потім — другий цикл (рис. 5.5).

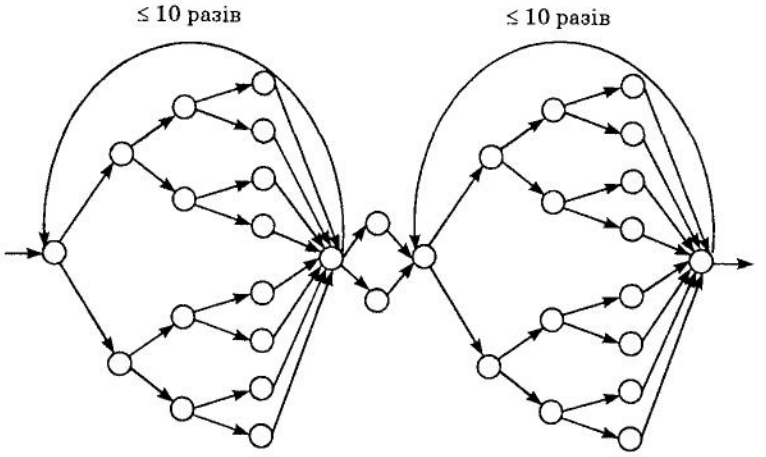


Рис. 5.5. Можливі шляхи порівняно невеликої програми

У кожному циклі є багато розгалужень. Вважаючи, що кожний розв'язок цілком незалежний від інших, кількість різних шляхів у модулі приблизно дорівнюватиме 10^{18}. Важко уявити, як і за який час можна протестувати таку кількість шляхів.

Раціональна стратегія розроблення тестів повинна поєднувати елементи обох підходів, наближаючись всередині спектра стратегій більше до першого. Вдалий вибір точки в спектрі стратегій підтверджує високий фаховий рівень тестувальника. Хоча загалом рівень проектування тестів на сьогодні ще не досить високий.

У процесі діагностування ПЗ використовують, крім тестування, і процедури доведення, контролю, випробування, атестації та відлагодження.

Доведення (Proof). Це спроба виявити помилки в програмі незалежно від її зовнішнього середовища. Такі методи передбачають формулювання тверджень і висновків про поведінку програми, доведення математичних теорем про правильність виконання програми. Їх можна розглядати як форму тестування, але вони не передбачають прямого виконання програми.

Контроль (Verification). Полягав у намаганні знайти помилки, виконуючи програму в тестовому або модельованому середовищі.

Випробування (Validation). Це спроба знайти помилки, виконуючи програму в заданому реальному середовищі.

Атестація (Certification). Авторитетне підтвердження правильності програми. Під час атестації програму порівнюють з певним, наперед визначеним, стандартом.

Відлагодження (Debugging). Охоплює діяльність, спрямовану на встановлення природи відомої помилки та її виправлення. Це не те саме, що тестування, оскільки тестування передбачає лише виявлення помилок. Результати тестування є вихідними для відлагодження.

Усі ці процедури характеризують діагностування ПЗ щодо середовища, на яке воно спирається, типів помилок, які мають бути виявлені, а також стандартів, з якими порівнюють програми, що тестують.

Не менш важливим є тестування ПЗ за певними ознаками.

Тестування модуля або автономне тестування (Module testing, unit testing). Воно полягає в контролі окремого програмного модуля в середовищі, ізольованому від інших модулів.

Тестування спрягань (Integration testing). Передбачає контроль спрягань між частинами системи (модулями, компонентами, підсистемами).

Комплексне тестування (System testing). Охоплює контроль або випробування системи щодо вихідних цілей. Якщо комплексне тестування виконують у модельованому середовищі, то воно є процесом контролю, а коли у реальному, то це — процес випробування.

Тестування зовнішніх функцій (External function testing). Воно спрямоване на контроль зовнішньої поведінки системи, визначеної зовнішніми специфікаціями.

Тестування прийнятності (Acceptance testing). Має своїм завданням перевірку кожного конкретного варіанта установлення системи з метою виявлення помилок, що виникли в процесі налагодження системи.

З огляду на обсяг тестування може бути повним або вибірковим. За *повного тестування* програму тестують повністю, а за *вибіркового* — в окремих точках блоку вхідних даних.

Під час вибіркового тестування надійність програми повністю не забезпечують. Частина помилок може бути прихованою і не виявитися. А повне тестування не можна реалізовувати, наприклад, через причини, які випливають з наведеного аналізу (див. рис. 5.5). Тому найчастіше використовують *структурне вибіркове тестування.* Воно полягає у розділенні блока вхідних даних на класи, кожен з яких дає змогу підтвердити певні властивості або працездатність окремих елементів структури. Вибір шляхів у структурі програми повинен здійснюватись так, щоб усі оператори програми були задіяні хоча б один раз.

Для перевірки результатів роботи програми використовують числове і символьне тестування.

Числове тестування. Полягає у перевірці правильності числових результатів роботи програм при певних тестових наборах.

Символьне тестування. Передбачає виконання процедур, що ґрунтуються на символьних входах. Вхідні змінні позначають так, що вони дають змогу виразити вихідні змінні також у символьному вигляді. При цьому різним шляхам у програмі відповідають різні символьні входи і виходи.

Контроль складових ПЗ необхідно проводити у зазначеній послідовності й повному обсязі. Його визначають залежно від вимог до надійності.

5.3. Тестування модулів

Процес тестування ПЗ починають з тестування модулів за наявності спроектованих тестів. Тому слід приділити цим процедурам особливу увагу.

Суть процесу тестування модулів

При тестуванні ПЗ увагу приділяють насамперед тестуванню автономних модулів. Суть цього процесу полягає в контролі окремого програмного модуля. Його метою є зна-

ходження невідповідностей між логікою модулів та їх спряганням, з одного боку, і зовнішніми специфікаціями, з іншого. Зовнішні специфікації охоплюють опис функцій, вхідних і вихідних даних та зовнішніх ефектів. Компіляцію модуля теж розглядають як тестування, оскільки компілятор виявляє більшість синтаксичних помилок, а також певні логічні і семантичні.

Процес тестування модуля здійснюють у кілька етапів:

— підготовка на основі зовнішніх специфікацій модуля тесту для кожної ситуації, для кожної з меж областей допустимих значень всіх недопустимих умов — усіх вхідних даних та областей зміни даних;

— перевірка тесту програми. Вона дає змогу з'ясувати, чи всі умовні переходи виконуватимуть у кожному напрямі. За необхідності потрібно додатково розробити відповідні тести;

— з'ясування того, чи охоплюють тести відповідно до тексту програми велику кількість можливих шляхів. Щодо циклів, то тест повинен відповідати шляху без кола циклу з однократним його виконанням і максимальною кількістю повторень;

— перевірка чутливості програми, згідно з її текстом, стосовно окремих особливих значень вхідних даних. За потреби необхідно доповнити тестову програму відповідними тестами.

Тестування автономних модулів здійснюють на основі кількох аксіом, що є фундаментальними принципами тестування:

— високоякісним є той тест, який має високу ймовірність виявлення помилок, а не той, що демонструє правильну роботу програми;

— однією з найважливіших проблем при тестуванні, виходячи з першої аксіоми, є з'ясування термінів закінчення тестування;

— неможливо тестувати особисту програму;

— необхідною частиною будь-якого тесту повинен бути опис очікуваних вихідних даних або результатів;

— слід уникати невідтворюваних тестів, тому що це призводить до необхідності тестувати програму повторно і відповідно повторно розробляти нові тести;

— необхідно розробляти тести як для правильних, так і для неправильних вхідних даних;

— слід детально аналізувати результати проходження кожного тесту;

— необхідно враховувати, що зі зростанням кількості виявлених у модулі ПЗ помилок підвищується відносна ймовірність існування в ньому невиявлених помилок. Тому такі модулі слід перевіряти особливо старанно;

— тестування проводять найкваліфікованіші програмісти;

— при розробленні ПЗ тестованість є ключовим завданням;

— проект системи повинен передбачати, що кожний модуль, який підключають до системи, має бути в єдиному варіанті;

— не можна змінювати програму з метою полегшення її тестування;

— тестування, як і будь-яку іншу діяльність, слід починати з визначення цілей;

— усі розгалуження потрібно виконувати в кожному з можливих напрямів хоча б один раз.

Тестування модулів ПЗ здійснюють на основі відповідних методів, зокрема статичного і динамічного тестування.

Статичне тестування. Полягає в тому, що одним з критеріїв обрання шляхів у структурі програми при тестуванні є задіяння всіма операторами хоча б один раз обраних шляхів. При цьому тестують команди за послідовністю їх розташування в тексті програми. Під час статичного тестування тестують не всі можливі шляхи в графі програми, тому частина помилок може залишитися невиявленою; програмний код взагалі не виконується — його тестують тільки шляхом логічного аналізу.

Для статичного тестування використовують такий інструментальний засіб, як компілятор. Виявивши, наприклад, синтаксичну помилку або неправильну операцію, він видає відповідне повідомлення. Компонувальник теж може видавати корисні повідомлення, зокрема імена змінних та інших об'єктів, що повторюються, посилання на неоголошені змінні та функції.

Статичний аналіз програми спеціалісти-тестувальники виконують неавтоматизовано. Вони читають вихідний код програми, обговорюють його і знаходять, як правило, досить багато помилок. Робота вичитування програмного коду досить рутинна, але необхідна.

Динамічне тестування (тестування гілок). Вимагає обрання таких шляхів, які перекривають усі гілки програми або всі розгалуження в усіх напрямах. Цей метод гарантує однократне тестування всіх гілок і операторів. Усі

можливі шляхи в модулі виявляють за допомогою керуючого графа.

Протестувати всі можливі шляхи виконання програми нереально, тому спеціалісти виділяють ті, які необхідно протестувати обов'язково. Для цього користуються спеціальними критеріями охоплення. Їх інколи називають логічними критеріями охоплення або критеріями повноти. Основні з них — критерії охоплення рядків, розгалужень і умов.

Критерій охоплення рядків. Вимагає, щоб кожний рядок коду програми був виконаний хоча б один раз. Однак для змістовного тестування програми навіть цього не достатньо. За цим критерієм, якщо рядок містить оператор прийняття рішення, повинні бути перевірені всі значення, що керують рішенням.

Критерій охоплення розгалужень. Застосовують для ґрунтовнішого діагностування модулів ПЗ. Суть його полягає в тому, що програміст-тестувальник перевіряє дію програми за виконаної і невиконаної умови оператора IF. У такому разі програма проходить не тільки всі рядки коду, але й всі можливі гілки.

Критерій охоплення умов. Він пред'являє найвищі вимоги. За ним необхідно перевірити всі складові кожної логічної умови, що часто означає перевірку всіх варіантів рішень.

Тестування шляхів у програмі вважають завершеним, коли обраний критерій охоплення повністю виконаний. Для автоматизації цього процесу розроблено програми, які аналізують програмний код і обчислюють кількість шляхів, що мають тестуватися. Вони також вираховують, скільки шляхів уже перевірено. Такі програми називають *засобами моніторингу охоплення.*

Критерії охоплення дуже важливі при діагностуванні, але одного тільки тестування шляхів для ефективного виявлення помилок програм недостатньо. Щоб усунути неточності при формуванні умов виходу із циклів, кожний з них випробовують двома тестами: перший приводить до виконання циклу з поверненням, другий — до виконання циклу без повернення.

Прикладом динамічного тестування є базові стратегії — тестування «чорної скриньки» і «скляної скриньки». Різниця між ними полягає тільки в тому, на якій інформації ґрунтується підбір тестів. Якщо програму розглядають як «скляну скриньку», то вона може бути протестована структурним методом. Його головною ідеєю є правильний вибір тестованого програмного шляху. На противагу йому

методом функційного тестування тестують модулі, які розглядають як «чорну скриньку».

Головна вимога до будь-якої програми або програмної системи — виконувати задані функції. Вони найповніше реалізуються при застосуванні функційного, структурного, символьного і регресивного тестування.

Функційне тестування. Найкраще перевіряє програми з точки зору відповідності їх завданню, специфікації або еталону, тобто, чи виконує програма функції та поставлені вимоги. Тести обирають за змістовим принципом або стохастично (за принципом ймовірності).

Якщо *вибір* здійснюють за *змістовим принципом*, то виходять з конкретного завдання. За *стохастичного вибору* тестів вони повинні в стохастичному розумінні відповідати завданням, які вирішують. Можна кількісно оцінити ймовірність того, що в програмі, яку тестують, немає помилок, а також знайти оптимальну кількість необхідних тестів. Основою стохастичного тестування повинна бути адекватність тестових і вхідних наборів за статистичними критеріями.

Якщо входи програми, яку тестують, мають дискретний характер, то необхідно, щоб як безумовні, так і умовні ймовірності виникнення окремих дискретних значень входів одночасно збігалися для тестових і експлуатаційних наборів послідовностей. У більшості програм є дискретні й неперервні входи.

За функційного тестування кожну функцію програми тестують шляхом введення її вхідних даних і аналізу вихідних. Внутрішню структуру програми враховують лише інколи або зовсім не враховують.

Структурне тестування. Має потужнішу теоретичну базу порівняно з функційним. Однак більшість тестувальників використовує функційне тестування. Хоча в структурному краще здійснювати математичне моделювання, проте воно не завжди ефективніше. Найоптимальніше використовувати комбінацію цих методів, оскільки кожен з них дає змогу виявляти помилки, пропущені при виконанні іншого методу. В такому разі вони однаково ефективні.

Символьне тестування. Інколи його застосовують для діагностування модулів ПЗ. На відміну від числового тестування, яке перевіряє роботу програми на окремих числових значеннях тестових наборів, символьне дає змогу оперувати множинами вихідних даних, визначених обмеженнями. Вирази шляхів програми за символьного тестування одержують шляхом прямої або оберненої підстановки.

Пряма підстановка. Це дії, що виконують при реалізації певного шляху в структурі програм. При цьому символьне виконання здійснюють для кожного оператора, що виконується, із запам'ятовуванням проміжних виразів символьних змінних.

Обернена підстановка. Вона полягає в тому, що обмеження на вхідні змінні починають будувати знизу при проходженні шляху на графі програми в оберненому напрямі. Одержують такі ж обмеження, як і за прямої підстановки, однак за оберненої немає необхідності запам'ятовувати записи символьних змінних. Перевага прямої підстановки перед оберненою полягає в тому, що вона дає змогу виявити шляхи, що не реалізуються, із суперечними обмеженнями на вихідні дані на ранніх етапах тестування.

Здійснення символьного тестування має певні труднощі. Серед них тестування циклічних ділянок програми, наявність у ній окремих модулів, встановлення значень змінної тільки під час виконання програми. Першу проблему вирішують шляхом підстановки заздалегідь обчисленої кількості ітерацій (лат. iteratio — повторення), другу — символьним виконанням модулів, третю — введенням додаткових гіпотетичних (імовірних) обмежень, що відповідають різним можливим випадкам.

Регресивне тестування. Здійснення цього методу — одне з основних завдань тестувальника. Суттю його є повторне використання розроблених тестів. Цей вид тестування реалізують такими шляхами:

— відпрацьовують тест, виявляють помилку, програміст її виправляє. Так проводять кілька ітерацій, поки досконало не буде виправлено відповідний фрагмент програми. Крім того, щоразу, коли в програму вносять зміни, всі тести відпрацьовують знову. Ці операції повинні засвідчити, що тестувальник-програміст цілком виправив виявлену помилку і вона більше не з'явиться;

— відпрацьовують стандартну серію тестів, щоб упевнитися, що, виправивши одну частину програми, не було внесено помилок в іншу. Для цього випадку тестують незмінність і цілісність попередньої версії всієї програми, а не виправлення однієї помилки.

Отже, діагностування модулів ПЗ здійснюють шляхом реалізації одного або кількох видів тестування, суть яких розкрито вище. Вибір виду тестування залежить від вимог до ступеня надійності програм.

Проектування тестів

Важливим етапом тестування ПЗ є проектування тестів, яке, вимагаючи творчого підходу, у кожному випадку має специфічний характер. Ця робота поки що не цілком формалізована. Однак певні рекомендації для її здійснення можна сформулювати. Передусім необхідно здійснити такі операції:

— вивчити діапазони змін вхідних і вихідних даних, щоб розробити тест для кожної характерної області значень вхідних даних;

— скласти таблицю значень логічних функцій і відповідні їм тести, проконтролювати повноту перевірок і за необхідності доповнити тести тими, яких не вистачає;

— за наявності циклів перевірити їх спрацювання за однократної і максимальної кількості повторень;

— розробити тести для перевірки чутливості модулів до критичних ситуацій.

Якщо програмний модуль розглядати як «чорну скриньку», то процес тестування поділяють на такі етапи: перевірка модуля в нормальних, екстремальних і критичних умовах.

Перевірка роботи модуля в нормальних умовах. Під час цього етапу на входи модуля подають дані, що відповідають звичайним умовам роботи.

Перевірка роботи модуля в екстремальних умовах. При здійсненні цієї процедури вхідні дані повинні включати граничні значення в області зміни.

Перевірка роботи модуля в критичних умовах. Під час цього етапу на входи модуля подають дані, що перебувають за допустимою областю їх зміни.

Вхідні дані при тестуванні модулів можуть бути спрощеними або реальними (модифікованими або реальними у повному обсязі).

Спрощені дані. Програміст-тестувальник обирає їх для перевірки загальної працездатності програмного модуля. Цей вибір зумовлений його розумінням специфіки програми.

Реальні модифіковані дані. Отримують методом моделювання. Процес функціювання програми повинен бути максимально наближеним до реальних умов. Вхідні дані модифікують з метою забезпечення достовірності й повноти перевірок.

Реальні дані у повному обсязі. Їх використовують, як правило, на заключному етапі розроблення тесту, здебільшого при налагодженні всієї програмної системи.

Етапи проектування тестів досить складні й трудомісткі. Це підтверджує, наприклад, методика структурного тестування програмного модуля. Проектування тестів полягає у виборі множини шляхів, що цілком покривають граф програми, і визначенні вхідних даних, на основі яких ці шляхи реалізують.

Граф програми є структурною моделлю програми, що вказує зв'язки між її елементами. Вершини графа відображають оператори розгалуження і об'єднання, а дуги — оператори оброблення і передавання даних. У математичному вигляді граф подають як упаковану матрицю суміжності $A = \{a_{ij}\}$ з v вершинами. Це vxl матриця, де l — максимальний ступінь виходу i-ї вершини. Ступінь входу $d_{вх}(v_j)$ і виходу $d_{вих}(v_i)$ певної вершини графа означає відповідно кількість тих дуг, що входять і виходять з вершини. Кожний рядок i матриці суміжності заповнюють довільно номерами вершин, що є суміжними з вершиною i.

За такого підходу критерієм тестування береться критерій гілок, маючи на увазі, що гілка програми є певною послідовністю операторів, які виконують чітко один за одним, тобто лінійною ділянкою програми. Для побудови мінімального покриття граф ділять на DD-шляхи з використанням суміжної матриці вихідного графа. D-вершинами позначають множину вершин, у яких $d_{вих}(v_i) > 1$, та вхідну і вихідну вершини. У цьому разі DD-шлях є простим шляхом між двома D-вершинами, в його межах немає D-вершин. Враховуючи це, визначають цикли і петлі та виключають дуги, що їх замикають.

Запропонований алгоритм побудови мінімального покриття графа складається з декількох етапів.

Етап 1. Розглядають i-ту вершину, визначають суміжну вершину j, номер якої є максимальним серед номерів суміжних вершин ($i \in \overline{1,N-1}$, N — кількість вершин графа).

Етап 2. Розглядають дугу (v_i,v_j). Якщо $d_{вих}(v_i) > 1$ і $d_{вх}(v_j) > 1$, то дугу $q(v_i,v_j)$ виключають. Якщо $d_{вих}(v_i) > 1$ і $d_{вх}(v_j) = 1$, то дугу $h(v_i,v_j)$ залишають і відмічають.

Етап 3. Якщо $d_{вих}(v_i) = 1$, $d_{вх}(v_j) > 1$ та якщо на шляху немає дуги типу q, то виключають дугу, що розглядають, а також дугу типу h, що була відмічена на етапі 2.

Етап 4. Підставляють $i = j$ та повторюють етапи 1—3 до того часу, поки j не дорівнюватиме номеру кінцевої (вихідної) вершини. Фіксують шлях у вигляді послідовності значень j.

Етап 5. Повторюють етапи 1—4 доти, доки в побудованому шляху не залишиться дуг типу q і h.

Тестування всіх шляхів у структурі програми не завжди можливе, особливо для складних програм з багатьма розгалуженнями, та й не завжди доцільне. Компромісним є *тестування вибірки шляхів*, що забезпечує випробування найважливіших взаємодій між частинами програми. На графі, який описує програму, взаємодії можуть бути змодельовані введенням пар вершин, що повинні взаємодіяти хоча б в одному тесті.

Якщо покриття знайдене одним зі згаданих способів, то далі необхідно визначити тести, для реалізації шляхів на графі програми, які утворюють покриття.

Загальний вираз для обчислення вхідних даних, що дають змогу реалізувати i-й шлях, такий:

$$H_i = \bigcap_{j=1}^{m} H_{ij},$$

де H_{ij} — множина наборів вхідних даних, яка забезпечує переключення в потрібному напрямі в i-й точці розгалуження i-го шляху ($j = \overline{1,m}$).

Множину H_{ij} визначають через H'_{ij}, яка є множиною значень проміжних результатів, що входять безпосередньо до умови переключення. Їх описують через певні умови, що найчастіше є нерівностями. Відображення множини H'_{ij} у множину H_{ij} здійснюють шляхом послідовних підстановок проміжних результатів в умову H'_{ij} до того часу, поки не будуть підставлені вхідні дані програми.

Наприклад, якщо H'_{ij} виражають через умову переключення в j-му вузлі i-го шляху $H'_{ij}: h_{ij}(y_{ij}(x)) \geq 0$, а $y_{ij} = y_{ij}(x)$, де y_{ij} — проміжні результати в точці ij графа програми, а x — вхідні дані, то $H_{ij}: h_{ij}(y_{ij}(x)) \geq 0$.

Удосконаленню методик і методів тестування сприяє автоматизація процесу тестування. Програмні засоби автоматизації тестування використовують для статичного оброблення інформації про помилки, виявлені в програмах користувачами, передбачення впливу ймовірних змін, локалізації потенційних джерел помилок і мінімізації нових помилок, що вносять при зміні програм.

Після того як тести для модуля спроектовані, переходять до наступних етапів, зокрема їх написання, тестування і виконання. Тести повинні бути перевірені і протестовані перед їх використанням для тестування реальної програми. Після виконання тестів необхідно обов'язково провести аналіз результатів тестування. Форма представлення тестів залежить від методів інтеграції модулів.

5.4. Інтеграція модулів, тестування зовнішніх функцій і комплексів програм

Тестування модулів є дуже важливим етапом діагностування комплексу ПЗ. Наступними завданнями, які розв'язують під час розроблення і налагодження ПЗ, є інтеграція модулів у систему, її комплексне тестування і налагодження.

Суть і методи інтеграції програмних модулів

Після проектування здійснюють *інтеграцію* (лат. integratio — поповнення) *модулів* — злиття модулів у програму або програмну систему. Вибір послідовності злиття модулів у систему визначає форму, в якій записують тести, послідовність програмування модулів, вибір засобів тестування і відповідно ефективність тестування. Тому рішення щодо інтеграції модулів необхідно приймати на ранніх етапах проектування ПЗ.

Існують різні методи злиття програмних модулів у систему. Найпоширенішими з них є висхідне і низхідне тестування, метод «великого стрибка» та метод Сандвича.

Висхідне тестування. Суть його полягає в тому, що програму збирають і тестують знизу вгору. Тестують першими і автономно модулі найнижчого рівня. Потім — модулі, що безпосередньо викликають уже перевірені. Для тестування модулів вищого рівня повинні бути протестовані всі модулі нижчого, які він викликає. Взаємозв'язки між модулями встановлюють з графа взаємозв'язків модулів або з матриці взаємозв'язків. Після попереднього автономного тестування модулів нижчого рівня, що зв'язані з модулем вищого, здійснюють їх спільне тестування. При цьому можливе паралельне тестування груп модулів з попередньо протестованими модулями нижчого рівня. Цей процес повторюють, поки не буде досягнуто модуля, що відповідає вершині графа. Саме тоді завершують тестування модулів і спрягань між ним.

За висхідного тестування для кожного модуля потрібен *драйвер* — програма, що імітує надходження інформації з попередніх модулів. Під час висхідного тестування програмної системи тести на кожен модуль слід подавати відповідно зі спряганнями модуля, що тестують. Тестові дані формують як безпосередньо вбудовані в цю програму змінні й структури даних. Програма багато разів викликає модуль, який тестують, і з кожним викликом передає йому нові тестові дані.

Кращим є підхід, за яким використовують програму тестування модулів. Вона дає змогу описувати тести спеціальною мовою. При цьому немає потреби розробляти драйвери.

Низхідне тестування. Полягає в тому, що програму збирають і тестують зверху вниз. Автономно тестують тільки головний модуль, що відповідає вершині графа. Потім до нього приєднують один за одним модулі, безпосередньо зв'язані з ним, і тестують одержану комбінацію. Процес повторюють, поки не будуть зібрані й перевірені всі модулі. Низхідне тестування дає змогу чітко дотримуватися певної послідовності тестування.

За цього методу інколи складається ситуація, коли після перевірки одного модуля наступний ще не готовий. Тоді потрібно дочекатися, поки його буде розроблено або розробити програму-імітатор («заглушку») його вихідних даних. Приймаючи рішення про розроблення «заглушки», спочатку оцінюють складність модуля. Можливо, краще зосередити зусилля на завершенні розроблення його самого. Щодо форми підготовлення і передавання даних програмі, коли головний модуль містить усі необхідні операції введення-виведення, тести пишуть як звичайні зовнішні дані для користувачів і передають програмі через пристрої введення. Проте так роблять нечасто. Для підвищення ефективності тестування модулі додають не за чітко визначеною низхідною послідовністю, а так, щоб оперативно забезпечити функціювання операцій фізичного введення-виведення. Вони мають найнижчий рівень. Перевага в тому, що всі наступні тести готують у формі, яка розрахована на користувачів.

Однією з головних переваг низхідного тестування є змога одночасно здійснювати тестування модуля, спрягань між модулями і частково зовнішніх виконуваних функцій. Цей метод використовують за наявності у проектах програми та її специфікаціях серйозних вад, які потребують виправлення. Послідовна перевірка модулів спрощує цю процедуру.

Одним з недоліків низхідного тестування є поверховий, неглибокий контроль, зумовлений незавершеністю

частини модулів і значною кількістю спрощених «заглушок». Крім того, використовуючи цей метод, часто неможливо тестувати певні логічні умови, зокрема помилкові ситуації або захисні перевірки. Низхідне тестування ускладнює або зовсім унеможливлює перевірки виняткових ситуацій у певному модулі, якщо програма працює з ним лише в обмеженому контексті. У такому випадку модуль не отримує достатньо повного набору вхідних даних.

Цю проблему вирішує *модифікований низхідний метод*. Його суть полягає в тому, що кожний модуль перед підключенням до програми піддають автономному тестуванню. Проте цей метод потребує драйверів і «заглушок» для кожного модуля.

Метод «великого стрибка». Він є одним з найпоширеніших методів тестування. Використовують його лише для невеликих програм. Особливість цього методу полягає в автономному тестуванні спочатку кожного модуля. Потім усіх їх одночасно інтегрують у систему. При цьому для кожного автономного методу необхідно розробляти драйвери і «заглушки», що суттєво ускладнює метод «великого стрибка». Від низхідного тестування його відрізняє те, що всі модулі інтегрують у систему одночасно в останній момент. Це зумовлює ситуацію, коли у спряганнях між модулями тривалий час залишаються помилки, які можуть бути непоміченими. Для невеликих добре спроектованих програм, наприклад драйверів, метод «великого стрибка» ефективний, для великих програм — неприйнятний.

Метод Сандвича. Це компромісний варіант між висхідним і низхідним тестуванням. Головна мета його полягає у використанні переваг цих методів й усуненні їх недоліків. Суть методу полягає в тому, що висхідне і низхідне тестування починають одночасно. Програміст збирає програму як знизу, так і зверху. Ці два процеси з'єднуються приблизно посередині. Точка зустрічі залежить від конкретної тестованої програми і заздалегідь визначається при вивченні її структури.

Використання цього методу має сенс при інтеграції і тестуванні великих програмних систем (операційних систем) та пакетів прикладних програм. Особливість його застосування виявляється в інтеграції системи на ранньому етапі, що дає змогу отримати каркас програми, яка працюватиме. Елементи висхідного тестування методу Сандвича забезпечують зменшення кількості «заглушок», що використовують. Проте він має ту саму вади, що і низхідне тестування — недосконало тестуються окремі модулі. Висхідний етап методу Сандвича вирішує цю проблему для модулів нижчих рівнів, але вона залишається невирішеною для нижньої половини верхньої частини програми.

Вирішити цю проблему дає змогу *модифікований метод Сандвича*, який полягає в тому, що нижні рівні програми тестують так само чітко знизу вгору, а модулі верхніх рівнів спочатку тестують ізольовано, а потім збирають низхідним способом. Це є компромісом між висхідним і низхідним підходами.

Важливу роль при діагностуванні ПЗ відіграють тестування спрягань і зовнішніх функцій.

Тестування спрягань. Під час цієї процедури перевіряють взаємозв'язки між модулями щодо використовуваної інформації і часу її надходження. Тести для контролю спрягань розробляють на основі аналізу архітектури програмної системи і структури програми. Цей аналіз ґрунтується на схемі інформаційних зв'язків у графі програмної системи і матриці взаємозв'язків модулів. Характер вхідних даних для кожного модуля визначають з таблиць вхідних змінних.

Тестування зовнішніх функцій. Його здійснюють з метою виявлення розходжень між програмою і її зовнішніми специфікаціями. Тому необхідною умовою такого тестування є наявність чітких і точних специфікацій. Основну увагу зосереджують на контролі зовнішньої поведінки програми, тобто зв'язку із зовнішнім середовищем (джерелами інформації, каналами зв'язку, керованими об'єктами, пристроями комп'ютерів та ін.).

Щоб розробити тести зовнішніх функцій, зовнішні специфікації поділяють на окремі зовнішні функції, наприклад за типом вхідних повідомлень або команд. Після старанного вивчення кожної функції будують тести. Це можна здійснювати за описаними правилами проектування тестів для модулів. Тести слід будувати для всіх вхідних варіантів і умов, а також на межах всіх областей припустимих значень на вході та областей зміни значень на виході. Вони повинні перевіряти поведінку програми на функційних межах, а також тоді, коли вводять непередбачені або недопустимі дані.

Тести зовнішніх функцій розробляють так, щоб їх можна було використати багаторазово. Бажано, щоб кожен з них був самоперевіряючим. Найважливішу частину тестів виокремлюють як *тести регресії*, які необхідно ви-

конувати після кожної модифікації або виправлення програми, щоб з'ясувати, чи під час виправлення не внесено помилки. Тестування функцій компонентів системи виконують перед збиранням її в єдине ціле. Насамперед складають тести за зовнішніми специфікаціями, не звертаючись до внутрішньої структури і алгоритму програми. Однак у програмі перевіряють і те, наскільки охоплені всі умовні переходи.

Тестуючи зовнішні функції, слід орієнтуватися на обов'язковий мінімум, тобто слід виконати всі розгалуження хоча б один раз у кожному напрямі, за винятком тих гілок, що забезпечують захисне програмування.

Збирання і комплексне тестування програмної системи

Після завершення автономного тестування модулів, тестування спрягань і тестування зовнішніх функцій приступають безпосередньо до інтеграції програмної системи. Цей процес починають з того, що складають план поступового збирання системи.

Кожний рівень або версію системи називають *спіном*. Збирання починають з нульового базового спіну і закінчують *N*-м спіном. За нульовий спін приймають, наприклад, операційну систему. У плані збирання також вказують кількості спінів. При неперервній інтеграції вони створюються часто і відрізняються один від одного лише одним добавленим модулем.

План збирання розробляють на основі розбиття системи на збірні вузли — набори модулів, які розробник збирає в окремий спін. Вони відображають певну функцію системи, є більшими, ніж модуль, але меншими за підсистему. Збирання призначене виділити збірні вузли і скласти календарний план їх розроблення. Ця робота повинна відповідати таким критеріям:

— у плані збирання слід враховувати обмеження, пов'язані з наявними ресурсами і графіком розроблення діагностичного забезпечення;

— послідовність збирання необхідно базувати на каркасній (0-й спін) версії системи, що вже готова і працює;

— у плані має враховуватися залежність одних збірних вузлів від інших;

— послідовність збирання має забезпечувати включення найважливіших функцій в перші спіни;

— у плані збирання слід мінімізувати застосування програм, що імітують відсутні у всіх спінах, крім останнього, певні фрагменти системи.

Дотримання перерахованих критеріїв збирання програмної системи є обов'язковим, хоча цей перелік можна змінювати, уточнюючи певні твердження.

Після збирання системи здійснюють її комплексне тестування.

Комплексне тестування — пошук невідповідності системи своїй вихідній (початковій) меті.

Чітке формулювання такої мети має дуже важливе значення і є обов'язковою умовою. Без цього виконати комплексне тестування неможливо.

Компонентами під час розроблення комплексного тесту є вихідні цілі, загальна документація, документація для користувача, а також сама система. Всі комплексні тести готують на основі документації для користувача. До зовнішніх специфікацій звертаються тільки за необхідності проаналізувати суперечності між системою та відомостями про неї.

Комплексні тести не зводять до перевірки окремих функцій системи. Їх розробляють у формі *сценаріїв* — послідовних дій користувача. Комплексне тестування є важкоформалізованим, складним творчим процесом. Розроблення високоякісних тестів, як правило, потребує спеціаліста вищої кваліфікації, ніж саме проектування. Немає і чітких рекомендацій щодо розроблення комплексних тестів. Однак певні загальні тенденції все-таки можна означити. Отже, слід проводити тестування:

— стресових ситуацій реального середовища;

— великих обсягів даних протягом тривалого часу;

— конфігурації апаратури і ПЗ;

— сумісності з розробленими раніше версіями і програмними засобами;

— захисту від несанкціонованого доступу;

— вимог до пам'яті комп'ютера;

— продуктивності та ефективності системи;

— налагодження системи;

— надійності та готовності системи;

— засобів відновлення після втрати даних або відмов;

— зручності обслуговування системи;

— документації для користувача;

— психологічних факторів, зокрема невідповідності зовнішнім специфікаціям, неправильного формулювання відповідей і повідомлень та ін.;

— зручності експлуатації системи.

Комплексне тестування проводять перед приймальними випробуваннями на завершальній стадії розроблення.

Відлагодження програми

Особливе значення в процесі діагностування ПЗ має відлагодження (усунення помилок) програми. Його починають з виявлення певних ознак або симптомів помилки в ПЗ, а далі визначають місце її знаходження. Проте ним часто нехтують, вважаючи, що цією роботою можуть займатися менш кваліфіковані спеціалісти. Об'єктивно все повинно бути навпаки.

Відлагодження програм від тестування відрізняють час і мета проведення. Його проводять тоді, коли програма зазделегідь непрацездатна. Якщо вона зовні функціює правильно, тоді її тестують з метою виявлення прихованих помилок.

Кожну помилку слід уважно вивчити, щоб зрозуміти причину її виникнення і з'ясувати заходи, які потрібно здійснити для її запобігання або завчасного виявлення. Аналізуючи це, необхідно звертати увагу на такі чинники:

— чому виникла саме ця помилка. Вказати безпосередньо її джерело;

— як і коли виявлена помилка;

— чому її не виявлено раніше (під час проектування, контролю або на попередній фазі тестування);

— що потрібно робити при проектуванні або тестуванні для попередження появи цієї помилки чи завчасного її виявлення.

Такий аналіз необхідний для гарантії, що виявлені помилки в системах, які працюють або тестовані, не пропущено і що виправлення виконано відповідно до прийнятих норм.

У процесі відлагодження програми на комп'ютері можна провести такі операції:

— здійснити покрокове виконання програми;

— перевірити значення і місцезнаходження певної змінної під час виконання програми;

— проаналізувати значення змінних та виразів, щоб за потреби змінити їх при виконанні програми;

— розглянути послідовність виклику функцій в програмі;

— видалити і додати змінні та вирази, які розглядають, під час виконання програми.

Діагностування ПЗ є обов'язковим у процесі проектування і розроблення МПП і С. Нехтування ним призводить до негативних наслідків.

З часом апаратні засоби і ПЗ можуть потребувати модернізації. Тому слід розглянути методи модернізації операційних систем, зокрема ПК.

Запитання. Завдання

1. У чому полягають особливості системного підходу до забезпечення надійності обчислювальної техніки?
2. Охарактеризуйте суть механізму відмов ПЗ і розкрийте його відмінності від механізму відмов апаратури.
3. Порівняйте залежності надійності апаратних і програмних засобів від часу експлуатації.
4. Назвіть критерії, за якими визначають імовірність та інтенсивність відмови. Охарактеризуйте їх.
5. Чим відрізняється модель Шумана оцінки надійності ПЗ від моделі Джелінські-Моранди?
6. У чому суть моделі Шика-Волвертона і на яких припущеннях вона грунтується?
7. Визначте міру довіри С, коли в програму внесено 20 помилок і виявлено 15 власних помилок програми і 15 внесених помилок при наявних 17 власних помилках.
8. Охарактеризуйте інтуїтивну модель Х. Мілса.
9. Як визначають надійність програмних комплексів?
10. Розкрийте суть первинних і вторинних помилок ПЗ.
11. У чому суть технологічної помилки?
12. Дайте загальну характеристику алгоритмічних помилок.
13. Чим зумовлені системні помилки ПЗ?
14. Охарактеризуйте помилки проектування і програмування.
15. Дайте загальну класифікацію помилок ПЗ за видами робіт зі створення і експлуатації ПЗ.
16. У чому полягає суть часової надлишковості?
17. Які особливості інформаційної надлишковості?
18. У чому суть програмної надлишковості?
19. Назвіть операції, які проводять під час профілактичного контролю ПЗ. Охарактеризуйте їх.
20. Якими методами здійснюють оперативний контроль ПЗ?
21. Дайте характеристику прямим експериментальним методам визначення надійності ПЗ.
22. Опишіть розрахунково-експериментальні методи аналізу надійності ПЗ. Які завдання ставлять перед ними?
23. Назвіть операції, які виконують під час контролю, випробування, атестації програм. Охарактеризуйте їх.
24. Які особливості повного і вибіркового тестування?

25. Що розуміють під числовим тестуванням?
26. У чому суть символьного тестування?
27. Назвіть і охарактеризуйте етапи, з яких складається процес тестування модулів.
28. Дайте характеристику критерію охоплення шляхів виконання програми.
29. Розкрийте суть критерію охоплення розгалужень.
30. Як здійснюють перевірку автономного модуля в нормальних, екстремальних і критичним умовах?
31. Назвіть і охарактеризуйте етапи, з яких складається алгоритм побудови мінімального покриття графа програми.
32. Чим відрізняється низхідне тестування від висхідного?
33. У якому випадку використовують метод «великого стрибка» і в чому його суть?
34. Дайте загальну характеристику методу Сандвича.
35. Яким чином здійснюють тестування спрягань при інтеграції модулів?
36. У чому полягає суть комплексного тестування системи і яку форму мають комплексні тести?

6.

Модернізація і експлуатаційне обслуговування ПК

Сучасні комп'ютерні технології розвиваються особливо інтенсивно. Тому навіть нові ПК порівняно швидко фізично й морально старіють. Необхідність підвищення продуктивності ПК та установлення сучасних програмних продуктів з розширеними функційними можливостями створює проблему для багатьох користувачів. Одним з підходів до вирішення цієї проблеми є модернізація ПК.

6.1. Загальні питання модернізації ПК

Експлуатація є досить важливим і багатогранним процесом. Виробництво і діагностування апаратних і програмних засобів ПК здійснюють з метою їх довготривалої і ефективної експлуатації, хоч нерідко комп'ютери не використовують на повну потужність. Вони навіть не відпрацьовують ту кількість часу, яку могли б. Це зумовлено насамперед відсутністю кваліфікованих спеціалістів і незадовільною комп'ютерною грамотністю користувачів.

За масової комп'ютеризації виробництва і управління суттєвою проблемою, яка призводить до економічних втрат, є старіння комп'ютерних засобів за відносно незначний час. Істотно продовжити використання застарілої комп'ютерної техніки неможливо. Однак певний час можна використовувати їх модернізацію.

Модернізація (франц. modernisation — оновлення) — заміна в комп'ютері застарілих компонентів на нові з вищими параметрами, якщо це дає змогу зробити конструкція комп'ютера.

Завдяки цьому користуються новими технологіями за меншу вартість, ніж витрати на придбання нового комп'ютера. Такі компоненти, як процесор, чіпи додаткової оперативної пам'яті, жорсткі диски більшої ємності, відеоадаптер і навіть материнська плата, користувач має змогу встановити самостійно і домогтися значного підвищення продуктивності ПК. При цьому дії користувача щодо модернізації комп'ютера досить прості.

Під час модернізації оновлюється не все обладнання, але не слід надто переживати з приводу того, як нове обладнання спряжеться зі старим. Це, однак, не знімає необхідності добре продумувати таке «вторгнення» нових компонентів у стару систему. Попри наявність методик зваженої модернізації не завжди вдається одразу знайти оптимальний варіант модернізації.

Перед прийняттям рішення про модернізацію його слід обґрунтувати, тобто визначити доцільність з точки зору вартості й продуктивності майбутньої модернізованої системи. Насамперед метою модернізації ПК має бути вирішення нових завдань, які не може здійснити наявний комп'ютер. Модернізований ПК повинен мати нові функційні можливості. Щодо підвищення ефективності оброблення інформації, під час модернізації орієнтуються на поліпшення графіки, більший об'єм пам'яті для зберігання ПЗ, високоякісне відтворення звуку, прискорення виконання команд та ін. Модернізацію апаратних засобів слід здійснювати в комплексі з модернізацією ПЗ.

З економічної точки зору навіть прості нові комп'ютери (типу Celeron) часто виявляються продуктивнішими і дешевшими, ніж модернізовані старі. Вартість повної модернізації застарілих ПК може наближатися до ціни нового комп'ютера. За такої ситуації навряд чи доцільно їх повністю модернізувати. Це матиме сенс тільки за часткової модернізації комп'ютерів тих користувачів, які

виконують специфічні завдання. У цьому разі проводять одну або кілька модернізацій. Загалом модернізація буде успішною тільки тоді, коли користувач точно знає, оновлення яких ресурсів принесе найбільшу користь і на якому етапі слід його припинити. Отже, займатися модернізацією слід кваліфікованим спеціалістам, користуючись фаховою літературою, керівництвом з експлуатації для комп'ютера, який модернізують. Багато специфічної інформації про експлуатацію ПК та їх компонентів, надійність і діагностування міститься в мережі Інтернет. Там є чимало інформації, яку використовують при виникненні ускладнень у процесі модернізації комп'ютерів.

6.2. Модернізація компонентів системного блока ПК

Здебільшого модернізацію комп'ютера починають із системного блока. В ньому на системній платі міститься процесор (мікропроцесор) — «мозок» комп'ютера. Він визначає продуктивність і типи завдань, що вирішують за його допомогою. Отже, коли метою модернізації ПК є підвищення продуктивності комп'ютерної системи та виконання додаткових функцій (інструкцій), замінюють системну плату.

Заміна системної плати

Це одна з найважливіших операцій, завдяки якій отримують майже новий комп'ютер. Заміна системної плати дає змогу істотно підвищити продуктивність ПК за рахунок збільшення його швидкодії, використати можливості нового процесора, розширити систему загалом, збільшити обсяг оперативної та кеш-пам'яті, швидкість обміну інформацією з периферійними пристроями тощо. Встановлення системної плати зумовлює, як правило, встановлення нового процесора. Це збільшує вартість модернізації. Тому з економічної точки зору слід проаналізувати, чи не дешевше купити новий комп'ютер.

Установлення нової системної плати є складною операцією, пов'язаною з демонтажем усього комп'ютера.

Щодо надійності, майже всі системні плати рівноцінні і мають однакові якості, але можуть істотно відрізнятися між собою за іншими критеріями. Низька ціна системної плати певного типу часто свідчить про її низьку надійність. Тому, обираючи системну плату, варто проаналізувати її ціну.

Заміну системної плати здійснюють у два етапи.

1. Підготовка до заміни системної плати. Складається вона з таких підетапів:

— запис параметрів HDD і всіх істотних параметрів CMOS SETUP — їх повідомляють у STANDARD CMOS SETUP;

— відкриття корпусу;

— маркування старих плат розширення.

2. Вилучення старої системної плати із системного блока. Воно включає такі процедури:

— від'єднання всіх кабелів від системної плати;

— від'єднання кабелів, що заважають витягнути плату із системного блока;

— витягнення карт розширення;

— відкручування гвинтів кріплення системної плати;

— витягнення модулів пам'яті;

— видалення пластикових стояків.

Установлюючи нову системну плату, слід дотримуватися таких вимог:

— плату потрібно розпакувати і впевнитися, що вона повністю укомплектована;

— у комплекті, крім плати, повинні бути кілька кабелів для підключення пристроїв введення-виведення та документація;

— якщо плату було замовлено з процесором і пам'яттю, то вони можуть бути встановлені безпосередньо на ній або упаковані окремо;

— до комплекту може входити браслет для заземлення.

Якщо на системній платі не встановлено процесор і модулі оперативної пам'яті, то це роблять насамперед, а потім на основі документації встановлюють у відповідне положення перемички.

Щоб установити процесор у системну плату, необхідно виконати дії, послідовність яких залежить від типу роз'єму процесора — сокет або слот. Щодо процесорів, що установлюють у сокет, необхідно здійснити такі операції:

— знайти контакт номер 1. Він міститься в одному з позначених (крапка чи скіс) кутів і на гнізді з поміткою

ZIF (Zero Insertion Forse — гніздо для встановлення мікропроцесорної мікросхеми);

— підняти важіль, мікросхему процесора помістити в роз'єм, сумістивши контактні виводи з відповідними отворами роз'єму сокет;

— опустити важіль, щоб зафіксувати мікросхему в роз'ємі;

— установити тепловідвідний пристрій на процесорі, заздалегідь змастивши його тепловідвідною сполукою.

З процесорами, що установлюють у роз'ємі типу слот, необхідно здійснити операції:

— установити спеціальний механізм кріплення (складається зі стояків кріплення процесора і механізму підтримки тепловідвідного елемента);

— закріпити на системній платі тепловідвідний елемент;

— встановити процесор у роз'єм по напрямних стояках кріплення.

Правильне виконання зазначених операцій та дотримання їх послідовності гарантуватиме працездатність установленої системної плати, якщо вона до цього була справна.

Установлення модулів пам'яті

До внутрішньої пам'яті ПК належать оперативний запам'ятовуючий пристрій (ОЗП), кеш-пам'ять і спеціальна пам'ять, що входить до складу відео- і аудіообладнання. Можна модернізувати кожен із зазначених видів внутрішньої пам'яті. Однак найчастіше цю процедуру проводять з ОЗП для збільшення його об'єму, що значно поліпшує роботу комп'ютера. Модернізація ОЗП полягає в установленні модулів пам'яті у слоті.

У більшості комп'ютерів на системних платах установлюють модулі типу DIMM (Dual In-Line Memory Module — модуль пам'яті з двосторонніми друкованими виводами (ОЗП)) або RIMM. Ця операція має свої особливості щодо кожного типу. Першими використовують слоти з найменшими номерами. Здебільшого модулі встановлюють парами, а інколи навіть по чотири.

Перед виконанням цієї процедури варто ознайомитися з документацією на плату. Тут повинна бути інформація про тип модулів оперативної плати, що установлюють у неї, і послідовність цієї операції.

Модернізація системного блока ПК

Модернізацію системного блока здебільшого починають із заміни старої системної плати новою. При цьому звертають увагу, наскільки вона укомплектована, зокрема, чи наявні мікросхема або модуль процесора та модулі оперативної пам'яті.

Перед установленням системної плати в системний блок варто її протестувати окремо функційним тестом. Для цього потрібно мати джерело живлення, монітор, відеокарту і гучномовець. Плату розміщують поруч із системним блоком і підключають до неї роз'єми живлення. Тип блока живлення вказують на системній платі установленням відповідного джампера.

Після успішного завершення функційного тесту системної плати її встановлюють у системний блок, закріплюють у корпусі кількома гвинтами і пластиковими стояками. Якщо корпус новий, то спочатку слід установити пластикові (металеві) стояки в спеціально призначені для них отвори. Якщо навколо отворів напаяний металевий кант, то вони призначені для металевих стояків, якщо ні — для пластикових. Металеві стояки вгвинчують в отвори шасі корпусу так, щоб вони розташувались навпроти відповідних їм отворів у платі. Пластикові стояки установлюють знизу в плату. Для цього їх потрібно натиснути, і вони стануть на своє місце. Коли розміри нової плати відрізняються від старої, то перед установленням потрібно визначити, якому отвору відповідає стояк і куди слід загвинчувати гвинти. Дуже важливо підкласти діелектричні шайби, щоб попередити коротке замикання.

Установивши системну плату в корпус системного блока, знову перевіряють її працездатність. Для цього запускають програму POST. Якщо цей тест не проходить, насамперед перевіряють, чи немає короткого замикання: від'єднують системну плату від корпусу і установлюють пластикові шайби між корпусом і платою в тих місцях, де їх не було.

Наступним кроком модернізації системного блока є установлення плат розширення. На них найчастіше містяться аудіо-, відео-, SCSI- та мережний адаптери. Для їх установлення на системній платі є спеціальні роз'єми розширення. Тримаючи адаптер за краї, його установлюють з боку металічних контактів у відповідний роз'єм. Потім

сильно натискають на верхній край плати, щоб вона повністю стала на своє місце. Далі підключають усі необхідні кабелі до установленої плати адаптера.

Після цього тестують системний блок. Це роблять за допомогою різних наборів тестових програм. Найпростіше протестувати системну плату щодо повного функціювання, запустивши операційну систему Windows. Якщо тестування є успішним, то у першому наближенні можна вважати, що компоненти системної плати справні.

Перед тестуванням системного блока слід закрити корпус і підключити зовнішні кабелі. Для цього здійснюють такі операції:

— закривають корпус кришкою;

— під'єднують усі зовнішні кабелі;

— у 15-контактний роз'єм відеоадаптера вставляють кабель від монітора;

— підключають до модему телефонний шнур;

— підключають до відповідних роз'ємів, згідно з документацією, кабелі клавіатури і миші;

— якщо залишились не підключені пристрої (джойстик, акустичні системи), то їх підключають до відповідних роз'ємів;

— вмикають комп'ютер.

Операцію установлення системної плати здійснюють у два етапи: установлення нової системної плати в корпус системного блока; підключення до системної плати кабелів від з'єднувачів і пристроїв введення-виведення.

1. Установлюють модулі оперативної пам'яті. Проводять зовнішній тест системної плати, заздалегідь підключивши блок живлення і відеокарту. Коли тестування плати не проходить, то з'ясовують, чи є акустичне повідомлення, перевіряють живлення, спрягання монітора з відеокартою, коректність установлення модулів пам'яті. Якщо плата працює, то блок живлення і відеокарту від'єднують. Установлюють стояки, а потім установлюють і закріплюють у корпусі системну плату, підключають живлення, відеокарту і монітор. Відпрацьовують внутрішній тест. Якщо він пройшов успішно, то установлюють карти розширення, якщо ні — знову перевіряють виконання попередніх операцій або повторюють зовнішній тест.

Установивши карти розширення, до них підключають кабелі. Далі підключають кабелі (RESET, TURBO та ін.) до корпусу системного блока і запускають тест функціювання

обслуговуючих елементів. Для цього установлюють нові найважливіші опції STANDARD CMOS SETUP. Для їх виклику слід підключити клавіатуру і монітор. Після запам'ятовування установлених опцій перевіряють працездатність вимикачів RESET і TURBO-перемикача. Якщо тест пройшов успішно, то закривають корпус, якщо ні — здійснюють пошук помилок, зокрема тестують окремо карти розширення. Останньою операцією при установленні системної плати є її налагодження за допомогою CMOS SETUP, командних файлів та ін.

2. До системної плати підключають кабелі від з'єднувачів, що залишилися незадіяними, та пристрої введення-виведення. Кілька з'єднувальних дротів системної плати підключають до різних елементів корпусу комп'ютера, зокрема індикаторів живлення, жорсткого диска і кнопки RESET. У сучасних системних платах є кілька вбудованих портів введення-виведення: IDE (Integrated Drive Electronics — інтерфейс для дисків (жорстких) з інтегрованим у нього контролером) і SCSI-адаптери, контролер дисководів, послідовні і паралельні порти, аудіо- та відеоадаптери. У платі типу ATX роз'єми портів вбудовані в плату з протилежного боку. У застарілих платах, наприклад BABY-AT, роз'єми портів і адаптерів закріплюють на задній стінці корпусу системного блока і за допомогою додаткових кабелів під'єднують до системної плати.

Підключення з'єднувальних кабелів до системної плати з інтегрованими портами введення-виведення здійснюють у такій послідовності. На платі знаходять 34-контактний роз'єм контролера дисководів гнучких дисків і за допомогою плоского кабелю підключають до нього дисководи. Потім за допомогою плоского кабеля IDE підключають пристрої з інтерфейсом IDE-накопичувачі на жорстких дисках, CD-ROM і накопичувач на магнітній стрічці, якщо такий є, до розташованих на платі 40-контактних роз'ємів головного і підлеглого IDE-контролерів. Жорсткий диск підключають до головного контролера, а CD-ROM — до підлеглого. На платах, інших, ніж тип ATX, для паралельного порту використовують з'єднувач з 25-контактним роз'ємом (тип «мама»). Для двох послідовних портів один роз'єм (тип «тато») завжди 9-контактний, а інший може бути 9-ти або 25-контактним. Кабелі підключають до всіх трьох портів. За відсутності з'єднувачів портів з відповідними роз'ємами порт установлюють на задній панелі корпусу. Там необхідно знайти тільки відповідні отвори. Щоб роз'єми трималися на задній панелі, необхідно загвинтити гвинти. За відсутності роз'ємів для підключення виконують лише описану операцію. Останніми до системної плати підключають кнопки та індикатори передньої панелі системного блока, а також внутрішній гучномовець. Якщо при цьому виникли труднощі, треба звернутися до документації на системну плату.

Особливої уваги потребує запуск програми SETUP BIOS. Цю процедуру здійснюють, підключивши всі пристрої, що є у складі комп'ютерної системи. Програма дає змогу сконфігурувати системну плату, повідомивши їй інформацію про установлені в системі пристрої, установити системну дату і час. Після цього комп'ютер самотестується і виявляє наявні проблеми.

Отже, щоб запустити програму SETUP BIOS, спочатку вмикають монітор, а потім системний блок. Система проводить самотестування при вмиканні живлення (POST). Коли програма виявляє помилки, то комп'ютер видає попереджуючі звукові сигнали. За їх кількістю визначають, яка саме помилка виникла. Крім того, процедура POST відображає результати тестування на екрані. Якщо в процесі роботи натиснути відповідну клавішу (яку саме — залежить від типу BIOS, установленої на системній платі), то завантаження буде перервано і програма SETUP BIOS може сприймати введену в неї системну інформацію. Якщо під час виконання POST на екрані не з'явиться підказка про клавішу (клавіші), за допомогою якої викликають програму установлення параметрів BIOS, то цю інформацію знаходять у документації на системну плату.

Далі користувач, послуговуючись меню програми SETUP BIOS, вводить поточну дату і час, параметри жорсткого диска, тип дисковода і відеоадаптера, установки для клавіатури тощо. Сучасніші BIOS самі визначають параметри жорсткого диска, і вводити їх вручну немає потреби.

Послідовність збереження нових установок і виходу з програми установлення параметрів BIOS подається на екрані дисплея або в документації на системну плату.

Процес установлення портів і плат розширення

У сучасних системних платах порти введення-виведення вбудовані. Вони можуть містити інтегровані відео-, звукові й мережні адаптери. Це, по суті, готова система, в якій є кілька вільних роз'ємів, що могли б зайняти інтегровані пристрої. Продуктивність інтегрованих компонентів і компонентів у вигляді окремих плат різна. У системних платах з інтегрованими компонентами вона вища. Однак у такому разі можливості модернізації окремих плат обмежені, оскільки необхідно придбати нову системну плату. А це коштуватиме значно дорожче, ніж модернізація системи на базі системної плати з інтегрованими адаптерами.

Якщо системна плата містить неінтегровані порти введення-виведення, то її підключають до плати розширення. У такому разі компонент займає вільний роз'єм розширення. Системні плати (системні блоки) мають:

— два послідовних порти (з буфером типу 16550A);
— паралельний порт (типу EPP/ECP);
— порт підключення клавіатури;
— порт підключення миші;
— два порти USB;
— роз'єм відео (необов'язковий);
— роз'єм аудіо/ігровий (необов'язковий);
— два порти ENHANCED IDE на локальній шині (первинний і вторинний);
— роз'єм контролера на гнучких магнітних дисках.

При установленні плати розширення послідовно виконують такі дії:

— плату беруть за краї, не торкаючись мікросхем і електричних з'єднань;
— опускають її нижній край з нанесеними металевими контактними місцями у відповідний роз'єм;
— гвинтом прикручують плату до корпусу системного блока;
— до установленої плати підключають усі необхідні кабелі.

Часто виникає необхідність замінити стару карту розширення на нову або додатково установити певну карту. Це стосується насамперед відео- і аудіоплат. Усе ширше застосовують для мультимедіасистем графічні дисплеї. Відеоплата істотно впливає на обчислювальні характеристики комп'ютера, зокрема мультимедійні. При цьому зростає популярність відеоплат із 3D-прискорювачем. Сучасні комп'ютери містять шину типу AGP (Accelerated Graphic Port — покращений графічний порт), до якої підключають спеціалізовані тримірні відеоплати. За відсутності слота AGP і наявності слота PCI раціональною є така послідовність установлення тримірної відеоплати:

— вимкнути монітор і системний блок з електромережі;
— від'єднати роз'єм монітора від графічної плати;
— відкрити корпус системного блока і знайти графічну плату;
— викрутивши гвинти фіксації, витягти стару графічну плату;
— установити нову графічну плату в 32-розрядний слот PCI;
— під'єднати кабель монітора й увімкнути його;
— увімкнути комп'ютер і викликати CMOS SETUP;
— у STANDARD CMOS SETUP установити опцію PRIMARY DISPLAY у стан VGA і перезапустити комп'ютер;
— з'ясувати, чи функціює відеоплата.

Якщо плата не функціює, необхідно провести програмне тестування або закріпити гвинтами карту і роз'єм монітора, або налагодити карту, або закрити корпус, або інсталювати драйвер. Перед програмним тестуванням необхідно звернути увагу на звуковий сигнал (він повинен бути 8-кратним), а також перевірити правильність конфігурування плати, коректність вибору слота і регулювання монітора.

Інколи виникає необхідність наростити відеопам'ять графічного адаптера, оскільки 1 Мбайт відеопам'яті не дає змоги працювати зі складною графікою. У графічному середовищі Windows необхідно щонайменше 2 Мбайт відеопам'яті. Якщо потрібно систематично працювати з графічними проектами, то відеопам'ять нарощують до максимального обсягу. Найхарактернішими ситуаціями є:

— на двомегабайтній платі установлено 1 Мбайт пам'яті. Її нарощують ще 1 Мбайтом пам'яті;
— на чотиримегабайтній платі установлено 2 Мбайта пам'яті. Її нарощують ще 2 Мбайтами;
— є плата з 4 Мбайтами пам'яті, яку доповнюють ще певною кількістю пам'яті — від 4-х до 8-ми Мбайт або більше.

При нарощуванні відеопам'яті виконують такі дії: 1) вимикають комп'ютер з електромережі та знімають кришку системного блока; 2) виймають відеоплату зі слота; 3) вий-

мають чіп відеопам'яті із захисної упаковки, зіставляють мітку на корпусі чіпа з міткою на роз'ємі, що призначена для чіпа на платі, і втискають його в роз'єм; 4) упевнюються в тому, що чіп не випадає з роз'єму на платі; 5) вставляють відеоплату в комп'ютер, закривають корпус і вмикають ПК; 6) упевнюються в зміні об'єму відеопам'яті, аналізуючи заставку відеоплати.

Якщо повідомлення вказує, що змін не відбулося, то це може бути спричинене неправильним установленням (неустановленням) чіпа пам'яті або його несправністю. Для остаточного висновку слід повторити процедуру встановлення відеопам'яті.

Звукову плату не можна модернізувати. Зокрема, при наявності вільного слота PCI можна установити звукову плату з хвильовими таблицями.

Послідовність установлення модемів

На відміну від установки плат розширення установка модемів потребує певних підготовчих робіт. Їх виконують у такій послідовності:

— підготовлюють телефонну розетку шляхом подовження телефонної лінії, при цьому для зручності установлюють паралельний телефон. При перенесенні телефонної розетки для полегшення підключення модему установлюють розетку типу RJ11. Використовуючи вітчизняну розетку, виготовляють перехідник для підключення модему;

— обирають конфігурацію модему. Всі модеми поділяють на внутрішні і зовнішні. *Внутрішніми* називають ті, що знаходяться всередині конструкції системного блока, а *зовнішніми* — ті, що поза ним. Для плати внутрішнього модему перед її установленням обирають номер порту (COM1—COM4) і номер лінії переривання (IRQ2—IRQ7) (стандарти значення ресурсів COM-портів указані в документації на ПК). Для цього установлюють у відповідні положення DIP-перемикачі або джампери на платі модему. Якщо всі порти введення-виведення вже зайняті, то слід один з пристроїв (на вибір користувача) відключити. Більшість сучасних модемів підтримує технологію автоматичного налагодження Plug and Play, тому джамперів у них може не бути;

— плату внутрішнього модему установлюють у системну плату як звичайну карту розширення;

— під'єднуючи зовнішній модем, при виключеному живленні його з'єднують інтерфейсним кабелем з відповідним роз'ємом системної плати і фіксують гвинтами. Після цього підключають живлення модему і комп'ютера.

Для з'єднання зовнішнього модему з телефонним каналом є канальний роз'єм (LINE, WALL, SW, LL). Використовують також будь-який інший кабель, який додають до модему, з роз'ємами типу RJ11, RJ12, RJ41 або RJ45. Щоб підключити телефонний апарат, більшість модемів мають гніздо PHONE. Телефоном можна користуватися за відсутності модемного зв'язку.

Підключаючи внутрішній модем, використовувати кабелі для комп'ютера і блока живлення немає потреби. Заземлення зовнішнього модему здійснюють за допомогою спеціального кабелю живлення. При підключенні живлення, як правило, виконується автоматична самоперевірка модему.

Щоб перевірити працездатність модему, користуються будь-якою зв'язною термінальною програмою, яка дає змогу керувати модемом за допомогою команд AT (Advanced Technology — передова технологія; ПК з процесорами I80286 і вище) або протоколу V.25 bis. Зовнішні модеми і команди, що вводять з комп'ютера, формують відповіді в текстовому або цифровому коді. Наприклад, на команди AT модем відповідає повідомленням OK чи 0. За відсутності відповіді вводять команду ATQOV1, щоб перевести модем у текстовий режим і повторити введення команди AT. При відповіді ERROR перевіряють установлені швидкості порту модему і комп'ютера.

Якщо на комп'ютері встановлено операційну систему Windows 95/98, то користувач може протестувати модем, користуючись вікном діалогу «Властивості модему». Вікно з'являється після введення команди «Панель управління», «Модем», «Діагностика», «Властивості». Результат тестування видається у зручному для перегляду вигляді.

За першого включення модему за умовчанням завантажуються заводські налагодження, і пристрій може почати працювати із цією конфігурацією. Однак умови використання модему зумовлюють зміни окремих параметрів налагодження, що здійснювались при його виготовленні. Насамперед слід звернути увагу на швидкість передавання інформації, коефіцієнт паузи імпульсного набору, режим виправлення помилок. У табл. 6.1 наведено типовий набір параметрів модему, встановлених за умовчанням.

Таблиця 6.1

Параметри модему

AT-команда	Параметр	Установлення
ATE1	Відлуння команд	Дозволено
ATQ0	Відповіді модему	Дозволено
ATV1	Форма відповідей модему	Текстова
ATX4	Набір контрольованих параметрів	Виявлення відповіді станції і сигналу «Зайнято»
AT%E0	Захист від помилок	Заборонено
AT&G2	Захисний тон	Увімкнутий з частотою 1800 Гц
ATS0=1	Автовідповідь	Дозволена після одного дзвінка
ATY0	Роз'єднання після прийняття сигналу BREAK	Заборонено
AT&K3	Керування потоком	Дозволений метод RTS/CTS
AT&C0	Стан ланцюжка DCD (109)	Вимушене вмикання
AT&D0	Стан ланцюжка DTR (108)	Ігнорування сигналу DTR
AT&P1	Відношення замикання-розмикання	61/39
AT&X0	Синхронізація передавання	Внутрішня

Якщо необхідно, за допомогою AT-команд значення окремих параметрів змінюють. Щоб зберегти обрані значення в енергозалежній пам'яті, використовують команду AT&Wn, де *n* — номер варіанта конфігурації.

Вмикаючи живлення в оперативну пам'ять модему, за умовчанням завантажується варіант конфігурації з мінімальним номером. Для завантаження іншого профілю використовують команду AT&Fn або AT&Zn, де *n* — номер варіанта конфігурації. Якщо необхідно, щоб при вмиканні живлення за умовчанням викликалась конфігурація за номером відмінним від 0, то його потрібно записати в пам'яті модему за допомогою команди AT&Yn.

Операції при установленні жорсткого диска

Заміна жорсткого диска є досить складною процедурою. Це зумовлено необхідністю установити не тільки обладнання, але й змінити конфігурацію системи. Існує також певний ризик загубити інформацію, записану на вже установлених жорстких дисках і навіть вивести з ладу комп'ютер. Однак заміна жорсткого диска на новіший дає змогу істотно підвищити продуктивність ПК і розширити його функційні можливості. Тому такий ризик у певних випадках виправданий.

Установлення жорсткого диска передбачає розгляд конфігурації накопичувача, фізичне установлення і форматування жорсткого диска. Процедурно ця робота охоплює:

— налагодження накопичувача;
— налагодження контролера та інтерфейсного пристрою;
— установлення накопичувача в корпус ПК;
— налагодження системи в цілому;
— виконання логічного розподілу диска;
— виконання форматування високого рівня.

Конфігурація накопичувача з IDE-інтерфейсом потребує встановлення перемикача «ведучий-ведений», а для накопичувачів з SCSI-інтерфейсом — вибір його ID.

Щодо контролерів накопичувачів, то в ранніх моделях ПК контролер установлювали в роз'єм системної плати, а сучасні IDE-накопичувачі мають вбудований контролер. Їх конфігурують за допомогою програми установлення параметрів BIOS. Однак певні системні плати не підтримують накопичувачів ATA/33 і ATA/66. Тому перед їх установленням у ПК необхідно упевнитись, що системна плата підтримує цей клас пристроїв.

Для SCSI-пристроїв плату адаптера розміщують у роз'ємі системної плати. Для конфігурації SCSI-адаптера потрібно установити такі параметри:

— адреси BIOS;
— канали прямого доступу до пам'яті (DMA);
— сигнали запиту переривань (IRQ);
— адреси портів введення-виведення.

Необхідні ресурси для адаптера або їх частину виділяють також автоматично BIOS і операційною системою. При встановленні адаптера, який підтримує технологія автоматичного налагодження Plug and Play, у комп'ютер з BIOS конфігурацію здійснює автоматично Plug and Play і операційна система Windows 9x. Система сама виділяє

ресурси, необхідні для роботи пристрою, і вирішує конфлікти, які виникають.

Налагодження адаптерів, що не відповідають Plug and Play, виконують вручну шляхом установлення відповідних перемикачів, причому необхідно точно знати, які саме ресурси потрібні для конкретної плати.

Накопичувач IDE використовує BIOS системної плати. Вона забезпечує завантаження пристроїв цього типу. В адаптерах SCSI-накопичувачів установлюють свою ROM BIOS, що дає змогу виконувати завантаження системи із цього типу пристроями. Якщо SCSI-накопичувач не використовують для завантаження, то необхідно відімкнути її ROM BIOS за допомогою перемичок або перемикачів, а для доступу до SCSI-накопичувача потрібно завантажити стандартний драйвер пристрою з операційної системи. У ROM BIOS адаптера SCSI записані також програми, що реалізують наступні функції:

— форматування низького рівня;

— керування накопичувачем конкретного типу (залежно від його параметрів);

— конфігурацію адаптера;

— підтримання нестандартних адрес портів введення-виведення;

— діагностику.

Якщо підтримання контролера жорсткого диска передбачено системою BIOS, то наявність вбудованої BIOS небажана, оскільки для неї відведений адресний простір в області верхньої пам'яті. Там вона займає останні 384 Кбайт у межах першого мегабайта системної пам'яті.

Області пам'яті, що займають BIOS різних адаптерів, не повинні перетинатися. Тому на платах є перемикачі й перемички, за допомогою яких змінюють адреси BIOS. У деяких випадках це можна зробити і програмно. Завдяки цьому запобігають можливим конфліктам.

Накопичувач на жсрсткому диску, як і інші, установлюють у системному блоці комп'ютера. Для цього потрібні пластмасові напрямні, які дають змогу установити його на відповідне місце. Їх кріплять до пристрою з обох боків. Якщо монтаж здійснюють не за допомогою напрямних, то потрібні спеціальні накладки. Для установлення накопичувача на жорстких дисках виконують такі дії:

— перевіряють, чи є у комп'ютері невикористаний роз'єм IDE. Здебільшого в Pentium-комп'ютер можна установити чотири IDE-пристрої — по два на кожний канал;

— перевіряють, як підключений кабель до накопичувача. Як правило, червоний провід кабелю підключають до першого контакту роз'єму накопичувача. Всі роз'єми мають свій ключ, що забезпечує правильне підключення;

— поміщають накопичувач у корпус системного блока і загвинчують необхідні гвинти;

— під'єднують кабель до задньої частини накопичувача. За можливістю краще під'єднати кабель перед закріпленням накопичувача в корпусі системного блока;

— під'єднують кабель живлення до накопичувача. Він, як правило, чотирижильний зі стандартним роз'ємом;

— вмикають комп'ютер. Якщо двигун монтованого накопичувача не запрацював, перевіряють правильність підключення кабелів;

— перезавантажують комп'ютер і запускають програму установлення параметрів BIOS. Визначивши тип накопичувача за допомогою розділу параметрів накопичувачів, у розділі конфігурування установлюють автоматичне визначення накопичувача при запусканні комп'ютера. Зберігають установлені параметри і виходять з програми;

— перезавантажують комп'ютер, розділяють диск на розділи і відформатовують його.

Якщо операційною системою є MS DOS, то для перенесення даних необхідно здійснити такі операції:

— створити завантажувальний диск і перевірити наявність у ньому програм FDISK, FORMAT і XCOPY;

— створити розділи на диску, який установлюють, відформатувати його як системний, незважаючи на те, що система ідентифікує його як диск D:;

— для перенесення всіх несхованих файлів з диску C: на диск D: ввести команду: XCOPY C: \ D: \ /S /E.

За нею всі файли, крім схованих, будуть перенесені на новий диск. Тоді відмикають живлення і виймають старий жорсткий диск, а на установленому диску ставлять перемикач у положення «основний». Після цього вмикають живлення і за допомогою програми FDISK на новому диску установлюють активний розділ.

В операційній системі Windows 9x перенесення даних є складнішою процедурою, оскільки вона використовує сховані файли і папки. Цю операцію здійснює також програма XCOPY, але вона теж складніша, ніж у DOS.

При введенні команди XCOPY автоматично запускається команда XCOPY32.

Для перенесення даних на новий диск, що установлюється, вводять команду: XCOPY32 C: \ D: \ /E /C /H /R /K, де:

/E — копіювати папки і підпапки, включаючи пусті;

/C — продовжувати копіювання у випадку виникнення помилок;

/H — копіювати сховані і системні файли;

/R — замінювати файли тільки для зчитування;

/K — копіювати атрибути (програма XCOPY знімає атрибут «тільки зчитування»).

Подальші дії, щоб перенести дані на новий жорсткий диск в операційній системі Windows 9x, аналогічні діям операційної системи MS DOS. Для проведення цієї операції використовують також спеціальні програми, зокрема DRIVE COPY фірми «Power Quest».

Підключення другого IDE-вінчестера. Часто установлення другого жорсткого диска є не тільки прийнятним, а й єдиним правильним рішенням. Це дає змогу мати на одному комп'ютері дві операційні системи, а також є одним з найкращих способів захисту інформації. За можливістю при установленні нового диска залишають старий з інформацією, яка є на ньому.

На перший погляд установлення другого жорсткого диска, крім кількох невеликих відмінностей, ідентичне установленню першого. Однак для проведення цієї операції необхідно добре знати комп'ютер, розуміти принципи і способи об'єднання пристроїв ПК. Найкраще розмістити другий IDE-диск поруч з першим у корпусі системного блока. При цьому може виникнути необхідність перемістити компоненти ПК. Для цього спочатку знімають плати розширення і вимикають кабелі, під'єднані до першого жорсткого диска. Перед тим запам'ятовують розташування плат розширення і кабелів, щоб потім правильно відновити конфігурацію комп'ютера. Знімати системну плату і блок живлення, установлюючи другий жорсткий диск, немає потреби. Установлення другого IDE-вінчестера здійснюють за такою послідовністю:

— виймають жорсткий диск, що був у комплекті ПК;

— установлюють пріоритет першого і другого жорстких дисків. Для цього переключають джампери на платі. Найкраще, щоб перший вінчестер був «ведучим», а другий «веденим». Якщо до цього комп'ютер мав один вінчестер, то він був «ведучим». Отже, його виймають і конфігурують за бажанням користувача. Джампери в більшості жорстких дисків містяться на зворотній стороні або з боку монтажної плати. Інформацію про положення джамперів указують на корпусі жорсткого диска і в документації на нього. Головне, щоб при встановленні двох IDE-дисків їх конфігурували порізному;

— перевіряють, чи є за «заглушкою» на корпусі системного блока вільний відсік для установлення накопичувача жорсткого магнітного диска. За його відсутності встановлюють кріплення;

— тимчасово, але надійно установлюють привод у відсік і закріплюють його. Після цього підключають до вінчестера живлення;

— під'єднують кабель даних. Маркована жила кабеля даних повинна суміщатися з контактом 1 роз'єму;

— вносять відповідні зміни в CMOS SETUP, користуючись опціями конфігурації. Вони записані у заставці CMOS SETUP;

— сегментують другий вінчестер. Для цього викликають програму FDISK, що дає змогу розмітити вінчестер, утворити на ньому логічні диски і присвоїти їм ідентифікатори;

— якщо розмітка другого диска завершилась успішно, його відформатовують командою FORMAT;

— перезапускають комп'ютер. Якщо він працює, то змінюють конфігурацію системи і новий вінчестер стає «ведучим», а старий «веденим»;

— остаточно закріплюють обидва вінчестери і корпус системного блока;

— протестовують новий диск. Для цього за допомогою команди XCOPY копіюють дані зі старого диска на новий і порівнюють їх. Для детальнішого діагностування використовують тести (CORE-TEST, CHECKIT тощо).

Дотримання послідовності операцій при установленні другого IDE-вінчестера гарантує його високу працездатність у складі ПК.

Послідовність дій при установленні дисковода

Останніми роками переважає тенденція замінювати поширені 3,5″-дисководи (накопичувачі на гнучких дисках) високоякіснішими дисководами з більшою ємністю. Це, зокрема, перезаписувані компакт-диски (CD-R), дисководи типу LS-120 та ін. Сучасні комп'ютери оснащені ZIP чи DVD-ROM-пристроями. Нові дисководи зберігають набагато більше інформації, ніж кілька 3,5″-дисководів.

Однак вони є у більшості ПК, і зберігання інформації на дискетах ще не втратило своєї актуальності. Тому найближчим часом 3,5″-дисководи будуть застосовуватись у комп'ютерних системах. Вони дуже практичні, і фізичне передавання та копіювання файлів за їх допомогою є найдешевшим способом. З економічної точки зору неефективно зберігати і передавати разом з фізичним носієм порівняно невеликі файли обсягом до 1 Мбайт на нових носіях об'ємом до 100—500 Мбайт.

Хоча дисководи — досить надійні пристрої зберігання інформації, але й вони з часом виходять з ладу, що зумовлює необхідність їх заміни. Нерідко це спричинено і бажанням мати другий дисковод у складі комп'ютерної системи. Процес установлення нового дисковода здійснюють у два етапи: зняття старого дисковода; установлення нового.

1. Зняття старого дисковода. На цьому етапі виконують такі дії:

— вимикають ПК, знімають кришку системного блока і знаходять місце кріплення 3,5″-дисковода;

— від'єднують кабелі від дисковода. До нього під'єднані два кабелі. Перший — плоский стрічковий (сірий з червоною смугою вздовж краю) для передавання даних з дисковода на системну плату, і навпаки. Другий — з'єднує 3,5″-дисковод з блоком живлення. Він складається з кількох дротів, його і під'єднують до великого білого роз'єму;

— послаблюють гвинти, що кріплять декоративну кришку всередині корпусу, витягають дисковод із шасі і, відкрутивши гвинти, висувають його з відсіку.

2. Установлення нового дисковода. При цьому вдаються до таких дій:

— вставляють новий дисковод у відповідний відсік і прикріплюють до шасі гвинтами. Потім до нього під'єднують кабелі. Зокрема, стрічковий кабель під'єднують до роз'єму на зворотному боці нового дисковода. Край з червоною смугою з'єднують з лівим нижнім штирком (Pin) на роз'ємі дисковода. Після цього під'єднують кабель живлення до відповідного роз'єму на зворотному боці дисковода;

— пересвідчуються, чи правильно установлений дисковод у шасі, після цього пригвинчують шасі на відповідне місце;

— перевіряють, чи правильно під'єднані кабелі;

— установлюють кришку корпусу системного блока і вмикають комп'ютер;

— входять у процедуру SETUP і перевіряють установлення BIOS, що стосується 3,5″-дисковода. Якщо старий дисковод було замінено на однотипний новий, то комп'ютер повинен працювати зі старими уставками, змінювати їх немає потреби;

— викликають процедуру SETUP, знаходять розділ, що дає змогу змінювати дискові опції. Пересвідчуються, що диск A: установлено як 3,5″ 1,44 Мбайт дисковод;

— пересвідчуються, що при запуску ПК спочатку звертається до дисковода A:, а не до жорсткого диска C:. Це перевіряють в опціях початкового завантаження (BOOT OPTIONS). За негативного результату налагоджують комп'ютер так, щоб він спочатку звертався до пристрою A:;

— після перевірки установок і внесення необхідних змін їх зберігають і виходять з процедури SETUP.

Правильність виконання операцій при установленні нового дисковода забезпечує його тривалу працездатність.

Установлення накопичувача CD-ROM

Ефективність комп'ютера можна підвищити за рахунок установлення накопичувача типу CD-ROM. Це роблять з огляду на велику ємність компакт-дисків (близько 650 Мбайт). Також спрощується інсталяція об'ємних пакетів програм порівняно з їх розміщенням на дискетах.

Якщо у складі комп'ютерної системи є дисковод CD-ROM, то причинами модернізації може бути незадовільний час завантаження програмного забезпечення з наявною CD-ROM, час доступу до даних (проміжок часу між звертанням до даних і їх отриманням) та ін.

Накопичувачі CD-ROM є зовнішні (переносні) і внутрішні. *Зовнішні накопичувачі* CD-ROM конструктивно від'єднані від системного блока, а *внутрішні* знаходяться в його складі.

Зовнішні дисководи простіші в установленні тому, що для цієї процедури не потрібно розбирати корпус системного блока комп'ютера. Їх легко переміщувати з однієї комп'ютерної системи в іншу. Для підключення таких дисководів частіше використовують SCSI-інтерфейс, який набагато швидший, ніж IDE (EIDE)-інтерфейс (модернізована версія IDE-стандарту). Однак коштує зовнішній дисковод CD-ROM значно дорожче. Це обмежує його використання.

Внутрішні дисководи CD-ROM з IDE-інтерфейсом значно поширеніші. Багато типів системних плат містять вбудований контролер IDE-інтерфейсу, а частина має окремий IDE-контролер, який підключають до слота розширення. У сучасних ПК EIDE-слоти інтегровані в системній платі. EIDE-інтерфейс у 3—4 рази швидший за IDE-інтерфейс.

Перед установленням CD-ROM звертають увагу на конфігурацію нового накопичувача і правильно її виставляють. Для цього знаходять усі перемички і роз'єми на новому дисководі. Для дисковода CD-ROM з IDE-інтерфейсом необхідно установити перемички в такі положення: «основний» дисковод на вторинному IDE-роз'ємі; «додатковий» щодо установленого жорсткого диска.

Якщо дисковод установлюють у вторинний EIDE-інтерфейс, то його перемички будуть установлені правильно. Це можна перевірити за документацією, яку додають до дисковода CD-ROM. Використовуючи CD-ROM як вторинний пристрій, перевіряють, чи правильно установлені перемички і під'єднаний кабель до системної плати. Накопичувачу, який установлюють, присвоюють наступну вільну букву пристрою.

Не слід підключати накопичувач CD-ROM і жорсткий диск в один канал IDE. Це спричинить уповільнення роботи обох пристроїв. CD-ROM підключають за наявності вільного каналу IDE у вторинний IDE, а жорсткий диск залишають на первинному. Установлення накопичувача CD-ROM відбувається за такою схемою:

— установлення дисковода;
— під'єднання кабелів;
— установлення ПЗ;
— тестування працездатності накопичувача CD-ROM.

Щоб установити зовнішній SCSI-накопичувач, немає потреби мати окрему плату адаптера для кожного пристрою, оскільки кожен адаптер підтримує до 7-ми пристроїв, а ULTRA 2 SCSI-адаптер підтримує до 15-ти пристроїв. У деяких системних платах є інтегрований SCSI-адаптер, але для інших потрібна окрема плата SCSI-адаптера.

Знайшовши місце для установлення накопичувача, до нього під'єднують кабель живлення. Роз'єм для кабелю міститься на задній панелі системного блока. Далі необхідно під'єднати один кінець кабелю до роз'єму накопичувача, а інший — до роз'єму на платі адаптера. На задній панелі зовнішнього накопичувача CD-ROM є два роз'єми. Для підключення накопичувача до комп'ютера використовують

будь-який з них. Потім закріплюють роз'єми кабелю за допомогою петель, якщо вони є.

На задній панелі зовнішнього накопичувача є перемикач ідентифікаційного номера пристрою SCSI. За умовчанням завантажувальному диску присвоюють номер 0, адаптерам — 7. Необхідно перевірити, щоб накопичувачу, який установлюють, був присвоєний інший номер (наприклад, 4; 5; 6). Не можна установлювати значення, яке вже використовують для іншої плати або периферійного SCSI-пристрою. До складу вбудованого накопичувача CD-ROM входять:

— накопичувач;
— плоский кабель для підключення накопичувача до адаптера SCSI або IDE та кабель для внутрішнього підключення накопичувача до звукової плати;
— компакт-диск (чи дискети) з програмами-драйверами;
— напрямні для монтажу накопичувача і гвинти для кріплення.

Перед установленням плати SCSI в роз'єм розширення підключають до неї плоский стрічковий кабель. Обидва його кінці повинні бути однаковими. Уздовж краю плати SCSI-адаптера міститься 50 штиркових контактів жовтого кольору. Це роз'єм плати. На ній вказані номери контактів (можуть бути тільки першого і останнього). Необхідно сумістити роз'єм кабелю зі штирковими контактами на платі й уставити його. Потім вставляють плату в роз'єм, не зважаючи на інший кінець плоского кабелю. Обирають відсік на передній панелі системного блока для установлення накопичувача і знімають кришку відсіку. Якщо накопичувач щільно входить у відсік і в бокових стиках системного блока є отвори для монтажних гвинтів, то додаткові напрямні не потрібні. Якщо розміри накопичувача менші за відсік, то спочатку пригвинчують напрямні, а потім поміщають пристрій CD-ROM у відсік і фіксують його гвинтами. Далі знаходять маркований бік плоского кабелю і суміщають його з першим контактом роз'єму накопичувача.

Щоб підключити живлення до накопичувача CD-ROM, він має на зворотному боці 4-контактний роз'єм. Усередині системного блока живлення для різних пристроїв призначені окремі кабелі з кількома роз'ємами. Якщо один або кілька з них вільні, то їх під'єднують до накопичувача CD-ROM. Якщо це неможливо, то використовують двійник-подовжувач. Переконавшись, що пристрій CD-ROM працює,

закривають кришку системного блока. Комп'ютер можна вмикати, але для роботи накопичувача CD-ROM необхідно ще установити програми-драйвери.

У процесі завантаження комп'ютера BIOS визначає наявність у ньому накопичувача CD-ROM і видає відповідне повідомлення. Якщо працює операційна система Windows 95/98, то вона сама установлює драйвери високого і низького рівнів, які є у базі даних Windows для конкретної моделі CD-ROM. Якщо після завантаження Windows у вікні діалогу «Мій комп'ютер» не з'явилась підпрограма CD-ROM, то необхідно установити і прописати у стартових файлах драйвер низького і високого рівня (MSCDEX). Для цього необхідно запустити SETUP.EXE з інсталяційної дискети, яку додають до пристроїв CD-ROM, і драйвери автоматично скопіюються на вінчестер, а відповідні команди будуть записані в стартові файли. Якщо на дискеті міститься тільки драйвер для CD-ROM, то його копіюють на вінчестер і зміни в стартових файлах CONFIG.SYS та AUTOEXEC.BAT проводять вручну.

Для перевірки накопичувача CD-ROM у нього вставляють компакт-диск і копіюють на вінчестер кілька файлів. Як тестовий компакт-диск, використовують диск, записаний за допомогою CD-R. Після успішного проходження тесту закріплюють накопичувач у відсіку.

Особливості установлення DVD-ROM

Цифровий універсальний диск DVD — це новий стандарт, що значно збільшує об'єм пам'яті і кількість додатків, що використовують для компакт-дисків. Технологія CD-ROM обмежена об'ємом пам'яті диска. Він може містити максимум 650 Мбайт даних, але й цього недостатньо для багатьох нових додатків, зокрема відео. DVD-стандарт може замінити відеокасети, а також диски CD-ROM.

DVD-диск є одностороннім, одношаровим і містить 4,7 Гбайт інформації. Його діаметр збігається з діаметром компакт-диска CD-ROM, але він у два рази тонший (0,6 мм). Сучасні моделі накопичувачів DVD підтримують DVD-диски ємністю 8,3 Гбайт. Двосторонні диски мають ємність 9,4 Гбайт на одній стороні. Двошарові DVD-диски — 17 Гбайт.

Установлення накопичувача DVD аналогічне встановленню звичайного накопичувача CD-ROM. Перед цією процедурою його необхідно правильно сконфігурувати, тобто установити перемички «первинний» і «вторинний». Майже всі накопичувачі DVD підтримують інтерфейс BUS MASTER IDE. Особливість при встановленні DVD-накопичувача в тому, що встановлюють MPEG (Monitor Picture Experts Group — плата декодера) у незадіяний роз'єм PCI, а також виділяють йому необхідні ресурси. Для встановлення DVD-накопичувача виконують такі операції:

— для конфігурування DVD перевіряють правильність установлення перемичок — це «додатковий» пристрій. Накопичувач на жорстких дисках повинен бути установлений як «основний»;

— установлюють накопичувач DVD у відсік 5,25". Цей процес в основному аналогічний установленню накопичувача CD-ROM;

— підключають накопичувач DVD до контролера IDE за допомогою стрічкового кабелю;

— підключають накопичувач DVD до плати декодера. Для цього з'єднують накопичувач з платою декодера (MPEG або MPEG-2) перехідним аудіокабелем;

— установлюють плату декодера в роз'єм PCI;

— підключають з'єднувальний аудіокабель до гнізда «лінійний вхід» аудіоплати, а кабель-перемичку — до роз'єму графічного адаптера;

— установлюють програмне забезпечення для DVD-накопичувача. Установлення драйверів та іншого програмного забезпечення DVD-накопичувача аналогічне їх установленню для CD-ROM. Накопичувач DVD підтримує операційна система Windows 95 версії 2.1. або Windows 98 та ін.

Процес установлення блока живлення

У багатьох випадках старий блок живлення краще замінити на новий, ніж ремонтувати. Це надійніше і дешевше. При цьому звертають увагу на форм-фактор блока живлення і при установленні нового з'ясовують, чи вони взаємозамінні.

Установлюючи блок живлення, здійснюють такі операції:

— визначають розміщення і спосіб кріплення блока живлення. Він кріпиться кількома гвинтами на задній

частині корпусу системного блока і може закриватися кришкою або кронштейном;

— переміщають стрічкові кабелі, що з'єднують дисководи і роз'єми портів подавання, від блока живлення;

— запам'ятовують розміщення кабелів перед відімкненням живлення, від'єднують роз'єми живлення від системної плати;

— запам'ятовують, як підключені диски до блока живлення, від'єднують роз'єми живлення від системної плати і від дисководів;

— знявши роз'єм живлення з дисковода, від'єднують дріт живлення від вентилятора процесора;

— від'єднавши всі кабелі, знімають блок живлення;

— порівнюють знятий блок живлення з новим. Упевнюються, що діаметр і розміщення отворів для установлення збігаються, а кабелів достатньо, щоб підключити всі пристрої системи;

— поміщають новий блок живлення у відповідний відсік корпусу системного блока;

— загвинчують вставлені гвинти, установлюють і прикріплюють кронштейн з вимикачем у вихідному положенні;

— під'єднують провід заземлення;

— прикріплюють роз'єми кабелів блока живлення до вентилятора процесора і дисководів;

— укладають кабелі даних. Кабелі живлення і даних тримають далі від панелей корпусу, щоб не зіпсувати їх;

— вмикають комп'ютер і перевіряють, чи обертаються вентилятори блока живлення і центрального процесора, чи набирає оберти жорсткий диск;

— якщо комп'ютер завантажився, то установлюють кришку системного блока ПК.

Після проведення переліку операцій з установлення блока живлення ПК готовий для подальшого використання.

Під час експлуатації ПК може виникнути потреба в установленні нових периферійних пристроїв або заміні тих, що входять до його складу.

6.3. Установлення і підключення периферійних пристроїв під час модернізації ПК

Периферійні пристрої відіграють важливу роль у складі комп'ютерів і комп'ютерних систем. Вони, як правило, призначені для введення і виведення інформації, що обробляється комп'ютерною системою. Тому зрозуміло, що без них її робота неможлива.

З часом периферійні пристрої і плати-адаптери, які підключають до системної плати, виходять з ладу та старіють за незначний час. Тому в процесі експлуатації комп'ютерних систем, у тому числі й ПК, виникає необхідність модернізації системи шляхом установлення і підключення нових периферійних засобів.

Установлення нової графічної плати і монітора

При тривалій експлуатації комп'ютера може виникнути необхідність в установленні нової графічної плати і (чи) монітора.

Для установлення нової графічної плати необхідно виконати такі операції:

— вимкнути монітор — від'єднати його кабелі від електромережі і системного блока;

— вимкнути системний блок і зняти з нього кришку;

— відгвинтити і зняти стару плату;

— визначити тип слота нової плати і вставити в нього нову плату. Закріпити її гвинтом;

— установити кришку системного блока і пригвинтити її;

— під'єднати монітор до графічної плати;

— під'єднати монітор і системний блок до електромережі;

— встановити програмні драйвери.

Після виконання згаданих операцій установлення нової графічної плати вважають закінченим.

Установлення монітора. Ця процедура є наступною після установлення графічної плати. Для цього необхідно виконати такі операції:

1) вимкнути старий монітор. Від'єднати його і ПК від електромережі;

2) установити монітор на робочий стіл;

3) під'єднати відеокабель монітора до відеоплати зі зворотного боку системного блока. Підключити кабель живлення до електромережі;

4) увімкнути комп'ютер і монітор;

5) установити програмні драйвери. Під час цієї процедури за допомогою технології автоматичного налагодження Plug and Play слід виконувати вказівки, що з'являються на екрані;

6) скоригувати роздільну здатність та інші параметри. Для цього в області «Панель управління» («Control Panel») необхідно обрати значок «Екран» («Display»). Викликати вкладку «Параметри» («Settings»). Як кольорову палітру обрати True Bit. В області «Робочий стіл» («Desktop Area») установити максимальну роздільну здатність, яку можуть підтримувати графічна плата і монітор;

7) скоригувати інші параметри за бажанням користувача. Для цього користуються вікнами налагоджень «Властивості» («Properties»): «Екран» («Display») з «Панелі керування» («Control Panel») чи клацають правою кнопкою миші будь-де на робочому столі та обирають пункт «Властивості» («Properties»);

8) провести тестування монітора. Для цього необхідно:

— порухати значки, намагаючись змінити розміри вікон, запустити деякі додатки. Визначити, чи досить чіткі кольори і наскільки вони правильні, чи змінюється яскравість ближче до країв зображення;

— запустити графічну програму і вивести на екран коло. Перевірити, чи воно спотворюється;

— завантажити текстовий процесор і надрукувати кілька речень. Зменшити розмір шрифта до 8-ми пунктів. Перевірити, чи можна легко прочитати текст, чи він чіткий і не розмитий;

— змінити яскравість від мінімуму до максимуму, дивлячись на краї екрана. Визначити, чи має картинка на екрані будь-які характерні спотворення;

— розмістити яскравий текст на чорному тлі в центрі, а потім — по краях екрана. Упевнитись, чи можна прочитати всі символи, особливо ті, що легко розмиваються, наприклад «е»;

— визначити, чи білі лінії на чорному тлі справді білого кольору;

— створити суцільні блоки зеленого, синього і червоного кольорів і перевірити їх однорідність у межах блока. При цьому слід змінювати розміри блоків;

— вивести на екран білий колір. Він повинен бути рівномірним і без плям;

— подивитись на екран монітора з тієї відстані, на якій ви будете працювати з ним. При цьому зображення на ньому повинно бути яскравим і чітким без мерехтіння.

Після виконання останньої операції тестування монітора вважають закінченим, таким чином завершують і саме його установлення.

Процес установлення клавіатури

Більшість користувачів рідко модернізують клавіатуру. Однак це може поліпшити її роботу, що досягається завдяки ергономічнішій (грец. ergon — робота, nomos — закон) її конструкції.

Процес модернізації клавіатури досить простий. Насамперед необхідно упевнитись, чи є в комп'ютері відповідний порт, з допомогою якого можна це зробити. У сучасних комп'ютерах є порт PS/2. Він має 6 штирків і призначений спеціально для підключення миші. Завдяки цьому вивільняють послідовний порт для підключення інших пристроїв. Клавіатура має 5-ти або 6-штирковий роз'єм. Однак це не обмежує підключення конкретного типу клавіатури або миші. Його можна зробити через відповідний їм адаптер.

Причиною заміни клавіатури може бути придбання ергономічнішого приладу. Крім стандартизованих моделей клавіатур, є багато видів спеціалізованих, які задовільняють специфічні вимоги користувачів (клавіатури, орієнтовані на певні операційні системи, клавіатури для дітей фірм «Comfy», «Kidtech», «KidBoard», мультимедіаклавіатури з гучномовцями і мікрофонами, безпровідні клавіатури та ін.).

Установлюючи нову клавіатуру, слід виконати такі операції:

— вимкнути комп'ютер;

— від'єднати стару клавіатуру, витягнувши її шнур з роз'єму порту;

— визначити тип порту клавіатури, який встановлений в системному блоці комп'ютера, і відповідний йому роз'єм (PS/2, DIN-роз'єм);

— визначити роз'єм нової клавіатури, щоб встановити, чи відповідає він роз'єму порту, установленому в ком-

п'ютері. При невідповідності слід використати спеціальний провід-адаптер, що, як правило, входить до комплекту постачання;

— увімкнути комп'ютер. За наявності технології автоматичного налагодження Plug and Play операційна система виявить нову клавіатуру і завантажить її драйвер. Якщо встановлення пристрою потребує інсталяції відповідного драйвера, то слід завантажити програмне забезпечення із завантажувальних дисків Windows;

— перезапустити комп'ютер з метою установлення нових параметрів налагодження, якщо це автоматично не зробила програма.

На цьому процес установлення клавіатури вважають закінченим.

Послідовність установлення миші

Стандартну мишу часто замінюють на альтернативну — трекбол, сенсорну подушку або цифрове перо-дигитайзер. Така заміна мотивована наявністю в них роликів прокручування або програмованих клавіш.

Установлення миші є досить простою процедурою, для чого необхідно здійснити такі дії:

— вимкнути комп'ютер;

— від'єднати установлену раніше мишу від порту, до якого вона під'єднувалась;

— з'ясувати, який роз'єм на кінці дроту миші, що установлюють, і чи потрібен перехідник для підключення роз'єму миші до порту;

— за необхідності задіяти перехідник під'єднати один його кінець до роз'єму миші, а інший — до роз'єму порту і зафіксувати його;

— підключивши мишу до порту, ввімкнути комп'ютер;

— завантажити драйвер миші.

За наявності технології автоматичного налагодження Plug and Play операційна система Windows 95/98 виявить нове обладнання і автоматично завантажить необхідний драйвер.

Альтернативні миші, зокрема пристрої із запрограмованими клавішами і ролики прокручування, мають власне програмне забезпечення. Для роликів прокручування необхідно встановити програмне забезпечення INTELLIPOINT фірми «Microsoft», а, приміром, для миші фірми «Kensington Internet» — MOUSEWORKS. Воно забезпечує виконання функцій автопрокручування. Якщо миша потребує інсталяції драйвера або інших керуючих програм, то вони мають бути в комплекті її постачання.

Установивши нову мишу й відповідне їй програмне забезпечення (драйвери, керуючі програми), необхідно перезапустити комп'ютер з метою установлення нових параметрів налагодження.

Особливості установлення принтера

Принтери для комп'ютерних систем постійно удосконалюють. Звичайні матричні принтери вже майже не використовують. Вони поступилися струминним і лазерним. Перші дають змогу отримати високоякісне кольорове зображення, другі — забезпечити хорошу якість друку в поліграфії.

При установленні принтера слід дотримуватись таких вимог: вимкнути комп'ютер і принтер з електромережі; упевнитися в наявності необхідних кабелів для підключення принтера (вони можуть не входити до комплекту постачання принтера); обов'язково дотримуватись інструкцій, що містить документація на принтер.

Для установлення принтера необхідно виконати операції:

— підготувати місце для розташування принтера. Навколо нього має бути природна циркуляція повітря. Він повинен міститися недалеко від комп'ютера, щоб довжини кабелів було достатньо для підключення пристрою до системного блока;

— ознайомитися з документацією на принтер;

— підготувати його для підключення до комп'ютера. Для цього з'єднати кабелем принтер і системний блок, під'єднати кабель живлення принтера до електромережі й установити в принтер картридж з чорнилами або з тонером (залежно від типу принтера — струминний чи лазерний);

— зібрати і установити лоток для подавання паперу;

— користуючись документацією на комп'ютер, упевнитись, чи можна під'єднати кабель до відповідного порту. В більшості комп'ютерів роз'єми портів спеціально позначені. Паралельні порти, до яких підключають принтери, мають, як правило, 25-контактні роз'єми;

— користуючись документацією, визначити послідовність установлення картриджа в принтер;

— вимкнути комп'ютер, витягнути старий картридж, уставити новий і зафіксувати його в принтері. При виконанні цієї операції дотримуватися правил, наведених у документації;

— увімкнути комп'ютер і принтер в електричну мережу;

— установити драйвер принтера. Ця операція залежить від типу установленої в комп'ютері операційної системи.

Установлення драйвера завершує процес установлення нового принтера.

Послідовність дій при установленні сканера

Сканер дає змогу доповнювати документи на екрані комп'ютера фотографіями. Популярність цих пристроїв зростає. Принцип роботи всіх сканерів однаковий. Вони перетворюють зображення документів на цифрову форму. Файл редагують за допомогою певних графічних програм, як правило, наявних у пристроях. Майже в усіх уніфікованих сканерах використовують типову програму TWAIN-драйвер для керування скануванням. Вона дає змогу графічним програмам розпізнавати текст, керувати сканером та обробляти відскановане зображення, не зберігаючи його в програмі сканування. Тобто з одним і тим самим сканером можуть працювати різні графічні програми і програми розпізнавання тексту.

Найпоширенішими є два способи під'єднання сканера до комп'ютера: під'єднання через паралельний порт; під'єднання через SCSI-інтерфейс. При цьому плату SCSI-інтерфейсу установлюють у комп'ютер додатково. Це роблять для складніших і дорожчих сканерів. Підключаючи сканер до паралельного порту, немає потреби відкривати комп'ютер. Третій спосіб під'єднання сканера до комп'ютера передбачає наявність на його системній платі шини USB. У такому разі сканер під'єднують до комп'ютера як USB-пристрій.

Підключаючи сканер через паралельний порт, слід здійснити такі операції:

— визначити, чи потрібно спеціально конфігурувати порт, а саме, чи потрібний для сканера порт EPP (Enhaced Parallel Port — розширений паралельний порт), ECP (Enhaced Capabilieties Port — порт з розширеними можливостями) або комбінований EPP/ECP-порт;

— за необхідності під'єднати до установленого паралельного порту комп'ютер слід увійти в процедуру SETUP і проглянути параметри портів, що зберігаються у BIOS. Перейти в пункт меню, в якому перераховані параметри периферійних пристроїв, і змінити уставки порту. Зберегти внесені зміни і перезавантажити ПК;

— якщо паралельний порт є частиною плати введення-виведення і на ній встановлено також COM-Port (COMmunication Port — послідовний порт), ігровий порт або інші порти, необхідно перевірити установки плати й упевнитися, чи вони відповідають необхідним параметрам. За потреби слід змінити їх. Це роблять, переставляючи перемички на самій платі.

Під'єднуючи сканер до комп'ютера, здійснюють такі процедури:

— вимикають комп'ютер і принтер. Від'єднують кабель від порту принтера (LPT-порт має роз'єм з 25-ма контактами) системного блока комп'ютера;

— під'єднують кабель даних до сканера. Якщо його можна під'єднати і до паралельного порту, і до SCSI-порту, необхідно обрати найоптимальніший варіант;

— підключають кабель даних сканера до порту принтера;

— під'єднують кабель принтера, що, як правило, підключений до паралельного порту, до додаткового роз'єму на сканері;

— перевіряють, чи можна сканувати і друкувати через один порт;

— установлюють програмне забезпечення на сканер перед тим, як ПК виявить пристрій. У цьому разі сканер і принтер не вмикають до того часу, поки не буде установлено відповідне програмне забезпечення. Якщо сканер виявляється автоматично завдяки технології автоматичного налагодження Plug and Play, то драйвер можна встановити після того, як пристрій буде автоматично виявлено;

— вмикають сканер, принтер і комп'ютер в електромережу. Після завантаження ПК подальші дії виконують відповідно до повідомлень операційної системи.

Щоб установити сканер із SCSI-інтерфейсом, насамперед необхідно упевнитись, що в комп'ютері наявна плата SCSI-інтерфейсу. Вона дає змогу одночасно керувати сімома пристроями, з'єднаними в ланцюжок. Тому її широко

використовують. До SCSI-інтерфейсу підключають дисководи CD-R, швидкісні CD-ROM, деякі стрічкові накопичувачі та ін.

У процесі підключення сканера на базі операційної системи Windows 95/98 необхідними є такі операції:

— клацнути на значку «Мій комп'ютер» («My Computer»), потім на значку «Панель керування» («Control Panel») і обрати значок «Система» («Systems»);

— проглянути перелік пристроїв, установлених у системі, на вкладці «Пристрої» («Device»);

— проглянути пункт «Контролери SCSI» («SCSI Controllers»). Якщо там є тільки пристрій «Iomega Parallel Port Zip», то в системі установлений Zip-пристрій, підключений до паралельного порту. Його для під'єднання сканера не використовують. Тому слід пошукати в переліку іншу інтерфейсну плату, наприклад фірми «Adaptec». Якщо вона установлена, то сканер підключають до неї.

Для установлення плати SCSI-інтерфейсу на базі операційної системи Windows 95/98 необхідно виконати такі операції:

— у вкладці «Пристрої» («Device») двічі клацнути на значку відповідного пристрою. У вікні, що з'явилося, обрати вкладку «Налагодження» («Setting»);

— перевірити уставки переривань (IRQ), каналів прямого доступу до пам'яті (DMA), портів введення-виведення (I/0) і адреси ділянок пам'яті, що виділені для плати SCSI-адаптера;

— якщо плата відповідає технології автоматичного налагодження Plug and Play, то інсталяція проводиться автоматично. Інколи для підключення Plug and Play необхідно переставити перемички;

— вимкнути комп'ютер і від'єднати його від електромережі;

— зняти кришку і знайти слот такого самого типу, як і роз'єм SCSI-плати. Більшість SCSI-плат — це ISA-плати, розраховані для установлення у подвійні слоти. Установити SCSI-плату у відповідний роз'єм. Сумістити планку з роз'ємом із щілиною в корпусі, прикріпити її до корпусу гвинтом;

— увімкнути комп'ютер і визначити, чи операційна система виявила SCSI-плату. Якщо це так, необхідно вставити дискету або компакт-диск з драйвером у комп'ютер і виконати процедуру установлення. Якщо ні, слід скористатися майстром «Установлення обладнання» («Add New Hardware Wizard») для виявлення і установлення SCSI-плати;

— упевнитись, що SCSI-плата з'явилася в переліку пристроїв у вікні «Панель керування» («Control Panel») і поряд з її значком немає жовтого символу «!». Він указує на наявність проблем, зокрема, що необхідно встановити нові драйвери або змінити апаратні установки.

Щоб підключити сканер до SCSI-плати, слід виконати такі дії:

— вимкнути комп'ютер і сканер;

— під'єднати один кінець кабелю до роз'єму плати, інший — до роз'єму сканера;

— присвоїти сканеру логічний номер пристрою (LUN), оскільки навіть просте SCSI-з'єднання є, по суті, мережею на платі. Для цього необхідно ручку або повзун з номером від 0 до 7, що міститься на задній панелі сканера, поставити на позицію від 0 до 6, яка ще не задіяна іншими пристроями;

— якщо на задній панелі сканера є два роз'єми, то до другого роз'єму під'єднують термінатор за умови, що сканер — єдиний SCSI-пристрій в комп'ютерній системі. В інших сканерах для їх підключення до системи є перемикач «увімкнути-вимкнути»;

— якщо сканер підключають до системи, в якій вже встановлена SCSI-плата, то його під'єднують до кінця ланцюжка пристроїв;

— вимкнути термінатор, підключений до останнього пристрою в ланцюжку SCSI-пристроїв, і під'єднати до останнього пристрою сканер. Перевірити логічні номери підключених пристроїв для впевненості, що сканер під'єднаний до невикористовуваного порту;

— упевнитисЬ, чи установлена програма SCSI INTERROGATOR. Запустити її, щоб визначити, які логічні номери вже використовуються у цій системі, а які — вільні;

— увімкнути сканер та інші SCSI-пристрої, через кілька секунд — комп'ютер. Звернути увагу на повідомлення, чи виявлено сканер. Виконати подальші вказівки, що видає комп'ютер, щоб завершити установлення драйвера.

Після виконання останньої операції сканер готовий до використання за призначенням.

6.4. Модернізація програмного забезпечення ПК

Визначальна особливість комп'ютерних систем та, що вони є сукупністю апаратних і програмних засобів. У процесі експлуатації вони з різних причин, зокрема заміни більш ефективними, зміни складу ПК, заміни периферійних пристроїв, потребують оновлення. Воно є складнішим процесом, порівняно із заміною апаратних засобів, і часто призводить до помилок. Тому модернізації програмного забезпечення ПК слід приділяти належну увагу.

Оновлення BIOS

BIOS розміщена в постійному запам'ятовуючому пристрої типу ROM (CMOS-zip). Вона керує взаємодією обладнання ПК. При вмиканні процесора програма, наявна в BIOS, ідентифікує апаратні засоби системи, складаючи їх «опис», і готує установлені компоненти ПК до роботи.

Якщо комп'ютерну систему доукомплектовують пристроями, які не входили раніше до її складу, BIOS необхідно обновити. В іншому разі вони не будуть ідентифіковані системою і залишаться заблокованими. Іншою причиною оновлення BIOS є встановлення технології автоматичного налагодження Plug and Play з метою автоматизації процесу розпізнавання і настроювання периферійних пристроїв. Перед початком оновлення BIOS необхідно виконати такі дії:

— перезавантажити комп'ютер і ввійти в програму SETUP. З її допомогою задають базові характеристики системи. Вона повідомляє послідовність перегляду пристроїв початкового завантаження, зокрема імена пристроїв, за якими базова система шукає операційну систему, адреси послідовних і паралельних портів тощо. В комп'ютерах з AMI BIOS під час запуску необхідно натиснути клавішу DELETE або F1. На комп'ютерах з PHOENIX BIOS — клавіші CTRL+ALT+ESC;

— перед оновленням BIOS увійти в програму SETUP і записати всю інформацію з CMOS, відмічаючи параметри установленого обладнання. Це роблять, щоб запобігти

змінам або стиранню настроювання параметрів. В іншому разі комп'ютерна система стає недієздатною. В програму SETUP входять, перезавантаживши комп'ютер і виконавши певні дії діалогу. На екрані з'явиться повідомлення типу «Натисніть клавішу DELETE, якщо хочете запустити програму установлення SETUP» («Press delete key, if you want to run setup»);

— виконайте цю команду для того, щоб увійти в програму SETUP;

— користуючись командою ALT+P, слід запам'ятати параметри настроювання на сторінках після появи фрази «Сторінка 1 із 2» («Page of 2»). Якщо існує друга сторінка, то записати і додаткові параметри настроювання;

— знову перезавантажити систему. Щоб задокументувати CMOS-параметри настроювання комп'ютера, переходячи від однієї сторінки даних CMOS до іншої, використати клавішу PRINT SCREEN. За цією командою на принтері друкується копія поточного вмісту екрана монітора. Параметри настроювання містяться на головній сторінці й розширеній, якщо така є, і відповідно роздруковуються;

— перенести дані BIOS на завантажувальну системну дискету чи CD-RW. Залежно від поточного номеру версії BIOS може виникнути необхідність виконати не одне, а два оновлення BIOS. Окремі фірми («Gateway», «Dell» та ін.) здійснюють безкоштовне оновлення BIOS.

Суть процесу оновлення BIOS полягає у використанні броузера для того, щоб потрапити на Web-сайт фірми і зайти в область технічної підтримки. Для одержання інформації про BIOS та інші компоненти комп'ютера, зокрема таблиці посилань зі зв'язками змін BIOS і номерами системних плат, треба клацнути на опцію «Documentation link» з іменем «Motherboard Referens».

Якщо відомий номер оновлення BIOS або системної плати, необхідно клацнути на одному з них. Цим забезпечують зв'язок між інформацією про системну плату та областю, де запам'ятаються останні версії відповідної флеш-BIOS. Для оновлення BIOS системної плати слід здійснити такі операції:

— точно визначити модель системної плати, оскільки для різних моделей системних плат можуть використовуватись різні оновлення BIOS. Назву фірми, BIOS та її версію вказують на системній платі й у технічній документації на неї;

— завантажити файл з наступною версією BIOS із сайта фірми-виробника системної плати в Інтернеті (чи одержати її іншим шляхом);

— розпакувати одержаний файл з оновленим програмним кодом BIOS. Якщо файл з розширенням EXE, то, щоб його розкрити, достатньо натиснути клавішу ENTER. Процедура проводиться автоматично і файл матиме розширення BIN;

— вимкнути захист модифікації програмного коду BIOS у BIOS SETUP, функцію FLASH BIOS PROTECTION в меню SEE & CHIPSET SETUP і установити значення DISABLED;

— завантажити DOS без резидентних програм із системою і файлом COMMAND.COM. При використанні Windows 9x виконувати завантаження у режимі емуляції MS-DOS, пропустивши файли AUTOEXEC.BAT і CONFIG.SYS;

— запустити утиліту оновлення флеш-пам'яті, що містить BIOS. Файл оновлення повинен бути у тому ж каталозі, що і програма оновлення. Точно запам'ятати ім'я файла, в якому є нова версія BIOS.

Програми оновлення BIOS, як правило, є діалоговими. Під час роботи вони запитують повне ім'я файла (з розширенням) з новою версією BIOS, повне ім'я для зберігання поточної версії (OLDBIOS.BIN) та підтвердження на оновлення (Y/N).

Після закінчення процесу оновлення треба перезавантажити комп'ютер і перевірити, чи коректно й усталено працює комп'ютерна система. Це здійснюють, вмикаючи її апаратну і програмну частини. Детальний кількісний аналіз зміни, зокрема продуктивності окремих підсистем і комп'ютера в цілому, можна провести за допомогою програм WINBENCH 99, WINBENCH 2000 та ін. У табл. 6.2. наведено Web-адреси окремих виробників BIOS.

Оновлення BIOS є однією з найважливіших операцій під час модернізації ПК.

Таблиця 6.2

Web-адреси виробників BIOS

№ п/п	Фірма	Адреса
1	«Award Software»	www.award.com www.award.com.tw
2	«American Megatrends Inc.» («AMI»)	www.megatrends.com

Перехід від однієї операційної системи до іншої як спосіб оновлення операційної системи

Головна причина оновлення операційної системи комп'ютера полягає в необхідності удосконалення її роботи й підвищення продуктивності. Перед установленням операційної системи новий жорсткий диск розділяють на розділи і відформатовують. Цю процедуру виконують за допомогою програми FDISK або PARTITIONMAGIC. Найчастіше оновлення операційної системи відбувається як перехід від MS DOS до Windows 98, від Windows 3.1 до Windows 98, від Windows 98 до Windows 2000.

Інколи виникає необхідність установити кілька операційних систем на одному комп'ютері. Це стосується найпопулярніших операційних систем MS DOS, Windows 3.x, Windows 95, Windows NT та Windows 2000.

Перехід від MS DOS до Windows 98. Windows 98 є 32-розрядною операційною системою. Вона дає змогу використовувати всі переваги архітектури IA-32 (Pentium PC). Насамперед Windows 98 забезпечує простіший доступ до Інтернету. Набір підпрограм DIAL-UP NETWORKING, які інтегровані в програму INTERNET EXPLORER WEB, найефективніше допомагає здійснювати цю процедуру.

Інші удосконалення Windows 98 стосуються підтримки апаратних новинок, зокрема універсальної послідовної шини USB, стандартного паралельного порту IEEE 1394, цифрового відеодиска (DVD) та ін. Операційна система Windows 98 підтримує за допомогою програми SYSTEM INFORMATION UTILITY такі операції, як резервне копіювання системи, очищення, конфігурування, діагностика. Вона має також засоби для проведення автоматичної модернізації. Всі ці зручності роблять операційну систему Windows 98 однією з найвикористовуваніших.

Щодо мінімальних ресурсів, то для Windows 98 потрібен комп'ютер на базі процесора Pentium з тактовою частотою ядра 100 МГц, 16 Мбайт оперативної пам'яті, дисковод CD-ROM і 120—300 Мбайт об'єму пам'яті на жорсткому диску (залежно від додаткових засобів, обраних при установленні). При підготовці до заміни MS DOS операційною системою Windows 98 необхідно провести такі дії:

— зберегти в резервній копії всі важливі дані;

— упевнитись, що в комп'ютері установлена BIOS останньої модифікації. За необхідності установити нову версію BIOS її слід переписати з Web-серверу продавця;

— створити завантажувальний диск MS DOS. Для цього порожню дискету поміщають у дисковод і набирають рядок copy C:/DOS/FDISK.EXE A:, замінюючи ім'я FDISK.EXE іменами файлів, які необхідно скопіювати (FDISK.EXE, FORMAT.COM і XCOPY.EXE з каталогу DOS);

— перезапустити систему, установивши перед цим у дисковод завантажувальний диск MS DOS. У рядку запрошення A:\> набрати команду FDISK і натиснути ENTER. У головному меню FDISK набрати «Create DOS Partitional» («Створити розділ DOS») або «Logical DOS Drive» («Логічний диск DOS»). Як DOS-розділ, необхідно використати весь жорсткий диск і зробити його активним;

— після створення розділу вийти з меню FDISK, натиснувши клавішу ESC. У рядку запрошення A:\> набрати команду FORMAT C:/S і знову натиснути клавішу ENTER. Завдяки цьому жорсткий диск буде відформатовано і на нього установлено систему. Після цього витягнути дискету з дисковода і перезапустити комп'ютер;

— установити драйвери дисковода CD-ROM, використовуючи диск з комплекту поставки CD-ROM. Помістити його в дисковод і в рядку запрошення A:\> набрати команду INSTALL або SETUP. Драйвери будуть установлені автоматично.

Наступний комплекс операцій переходу від MS DOS до Windows 98 — установлення операційної системи Windows 98. Для цього слід виконати такі дії:

— помістити установчий диск Windows 98 у дисковод CD-ROM;

— в рядку запрошення C:\> набрати символ, що відповідає дисководу CD-ROM, і натиснути клавішу ENTER. Потім набрати команду SETUP і знову натиснути ENTER. Після цього виконувати команди, що з'являються у різних вікнах «Майстер установлення» («Setup Wizard»);

— за бажанням скопіювати весь вміст каталогу Windows 98 з диска CD-ROM (D:\win 98) у кореневий каталог жорсткого диска (C:\). Це дасть змогу встановлювати Windows 98 з жорсткого диска, крім того зберігати всі інсталяційні файли і CAB-файли для майбутніх інсталяцій і модифікацій в каталозі C:\win 98.

На цьому перехід від операційної системи MS DOS до Windows 98 вважають закінченим.

Перехід від Windows 3.1 до Windows 98. Ця процедура дає змогу працювати у 32-розрядній операційній системі, зберігаючи підтримку 16-розрядних прикладних програм DOS і Windows. Переваги Windows 98 полягають

у тому, що підключення до Інтернету здійснюють безпосередньо на робочому столі. Також існує чимало версій останніх модифікацій комп'ютерних систем і прикладних програм.

Підготовка до переходу від Windows 3.1 до Windows 98 складається з таких етапів:

— упевнитись, що в ПК установлено BIOS останньої модифікації;

— створити завантажувальний диск MS DOS;

— перезапустити систему, установивши завантажувальний диск MS DOS. Як DOS-розділ, використати весь жорсткий диск, зробивши його активним;

— вийти з меню FDISK. Набрати команду FORMAT C:/S і запустити її, натиснувши клавішу ENTER. Завдяки цьому жорсткий диск відформатується і на нього буде установлено систему;

— вийняти дискету з дисковода і перезапустити ПК;

— установити драйвери дисковода CD-ROM. Для цього використати команду INSTALL або SETUP. Процедура здійснюється автоматично.

Для установлення операційної системи Windows 98 необхідно провести операції:

— помістити установчий диск Windows 98 у дисковод CD-ROM;

— вибрати «Виконати» («Run») у меню «Файл» («File») «Менеджера програм» («Program Maneger»);

— у діалоговому вікні «Run» набрати D:\WIN 98\SETUP.EXE, де D: — позначення CD-ROM. Виконати вказівки, що з'являються у вікні «Майстер установлення»;

— скопіювати весь вміст каталогу Windows 98 з диска CD-ROM (D:\win 98) у кореневий каталог жорсткого диска. Це дасть змогу установлювати Windows 98 з жорсткого диска і зберігати всі інсталяційні файли цієї системи для майбутніх інсталяцій і модифікацій в каталозі C:\win 98.

Ця операція завершує перехід від Windows 3.1 до Windows 98.

Перехід від Windows 95 до Windows 98. Підготовка до переходу від операційної системи Windows 95 до Windows 98 передбачає перевірку жорсткого диска, на якому мають установлювати цю систему, і запуск антивірусного програмного забезпечення. Також слід упевнитись, що жорсткий диск не стиснутий.

Перед оновленням необхідно скопіювати важливі файли на диск, використавши «Провідник» («Explorer»), зокрема файли AUTOEXEC.BAT, CONFIG.SYS, WIN.INI,

SYSTEM.INI, USER.DAT, SYSTEM.DAT та ін. Також створити новий диск початкового завантаження Windows 95. Для цього на «Панелі керування» («Control Panel») зайти в «Установлення і видалення програм» («Add/Remove Programs»). Потім обрати закладку «Системний диск» («Startup Disk»), вставити дискету в дисковод і клацнути на кнопці «Створити диск» («Create Disk»). Слід перевірити, чи увімкнуті будь-які програми спостереження і поточного контролю. Якщо так, то їх необхідно вимкнути, оскільки вони заважатимуть установленню Windows 98. Видалити всі прикладні програми з папки «Системний диск» («Startup Disk»), використавши «Провідник» («Explorer»).

Щоб почати установлення, насамперед закривають Windows, відключають комп'ютерну систему і перезапускають ПК. Для інсталяції Windows 98 слід здійснити такі операції:

— вставити інсталяційний диск у дисковод CD-ROM. Процедура здійснюється автоматично. Якщо ні, то необхідно клацнути на кнопці «Пуск» («Start») і обрати команду «Виконати» («Run»). У діалоговому вікні набрати D:\WIN 98\SETUP.EXE, де D:;

— для виклику програми «Майстер установлення» («Setup Wizard»), клацнути на кнопці «Продовжити» («Continue»). Коли з'явиться вікно ліцензійної угоди Microsoft, то клацнути на кнопці «Приймаю» («I accept»), а потім — на кнопці «Далі» («Next»);

— ввести у вікні реєстрування реєстраційний код програми і клацнути на кнопці «Далі» («Next»). Потім ввести інформацію, що записується;

— за необхідності зберегти системні файли Windows 95 на жорсткому диску дати на запит перед запусканням програми установлення відповідь «Так» («Yes»);

— при запиті на створення завантажувального диска дати відповідь «Так» («Yes»). Windows 98 сама визначає на відміну від попередніх операційних систем, яке установлення потрібне і найкраще підходить для комп'ютерної системи: «Типова» («Typical»), «Переносна» («Portable»), «Компактна» («Compact») або «Вибіркова» («Custom»).

Операційна система, використовуючи технологію автоматичного налагодження Plug and Play, може видавати дати і перезапускатися кілька разів. За невдалої спроби встановити Windows 98 необхідно перевантажити цю операційну систему з Windows 95 і спробувати установити знову.

За першого успішного завантаження Windows 98 з'явиться повідомлення «Ласкаво просимо в» («Welcome to») Windows 98. Далі слід клацнути на кнопці «Maintain Your Computer». Запуститься новий «Майстер супроводу», який видалить непотрібні файли, перевірить жорсткий диск на наявність помилок і прискорить виконання деяких прикладних програм.

Якщо комп'ютер підключений до Інтернету, необхідно клацнути на кнопці «Пуск» («Start») і обрати опцію «Windows Update» для звертання до фірми «Microsoft». Це роблять для того, щоб отримати і установити нові системні програми і оновлення для установленої операційної системи.

Установлення Windows NT4.0 і Windows 2000

Ці операційні системи на відміну від DOS, Windows 3.1 та Windows 9x розробляли на зовсім інших засадах. Вони несумісні зі згаданими версіями операційних систем. Тому Windows 9x ближча до DOS, ніж до Windows NT і Windows 2000. Через це Windows NT порівняно з Windows 9x впроваджують досить повільно. Це стосується також і нової версії — Windows NT4.0. Раніше основною причиною повільного впровадження Windows NT була недостатня кількість 32-розрядних додатків (16-розрядні додатки на основі цієї системи не впроваджують). У наш час ситуація змінюється на користь Windows NT.

Windows NT і Windows 2000 відрізняються від DOS тим, що вони не є оболонкою в традиційному розумінні цього слова. Ці операційні системи швидше емулюють DOS і таким чином дають змогу всім бажаючим попрацювати зі звичним інтерфейсом командного рядка. Проте більшість DOS-програм у сеансі DOS не працює у Windows NT. Немає тут і символьного режиму екрана, який у Windows 9x установлюють перед завантаженням графічної оболонки.

Для завантаження драйверів і зберігання параметрів операційних систем Windows NT і Windows 2000, як і Windows 9x, використовують системний реєстр, а файлів CONFIG.SYS, AUTOEXEC.BAT та .INI тут немає.

Модернізувати Windows 9x до Windows NT неможливо, оскільки при установленні цієї операційної системи всі додатки необхідно установити і налагодити заново.

Windows NT і Windows 2000 використовують файлову систему FAT. Таким чином, комп'ютер можна завантажи-

ти з DOS-диска і мати повний доступ до всіх файлів. Однак більшість нових можливостей цих операційних систем забезпечує їх власна файлова система NTFS (NT FILE SYSTEM). Вона дає змогу створювати на диску розділи до 2 Гбайт. Крім того, в ній вбудовані функції стискання файлів, безпеки й аудиту, що необхідні при роботі у мережному середовищі. У Windows 2000 реалізують підтримку файлової системи FAT 32.

Установлення Windows NT4.0. починають на диску FAT, але наприкінці процедури дані на диску можна конвертувати у формат NTFS. Це роблять, користуючись утилітою CONVERT.EXE, яку додають до операційної системи. Розділ диска, перетворений під систему NTFS, стає недоступним для інших операційних систем. Щоб повернутися до DOS або Windows 9x, розділ NTFS видаляють, а замість нього створюють розділ FAT. Windows 2000 можна установлювати на диску з файловою системою FAT 32 і NTFS.

Завантаження операційної системи Windows NT або Windows 2000 аналогічне завантаженню інших систем до моменту зчитування завантажувального запису активного розділу. У Windows NT чи Windows 2000 замість файлів IO.SYS і MSDOS.SYS запускають завантажувач NTLDR, який визначає обладнання і дає змогу обрати систему для завантаження (файл BOOT.INI).

Обравши її, запускають файл NT DETECT.COM і визначають обладнання ПК. Після цього в його пам'ять завантажують ядро Windows NT4.0. (NTOSKRNL.EXE) і рівень апаратних абстракцій (HAL.DLL). Ядро системи ініціалізує більшу частину операційної системи, включаючи драйвери пристроїв, підсистему NT, служби, систему керування пам'яттю. На цьому автоматичне завантаження системи припиняють до того часу, поки не буде натиснута комбінація клавіш CTRL + ALT + DEL і користувач не зареєструється у системі.

Установлення кількох операційних систем на одному ПК

Для того щоб установити на ПК більше ніж одну операційну систему, необхідно сконфігурувати процедуру завантаження системи як багатоопераційної системи з необхідними параметрами. При цьому користувач працює в певний момент тільки з однією операційною системою. За

необхідності перейти до іншої слід закрити цю систему і завантажити наступну. Найпопулярнішими операційними системами є MS DOS, Windows 95, Windows 98, Windows NT і Windows 2000.

Підготовка до їх установлення полягає у створенні резервної копії жорсткого диска, що дає змогу відновити працездатність системи при невдалому установленні, та завантажувальної дискети для завантаження в разі необхідності установленої раніше системи. Конфігурація кількох операційних систем має свої особливості.

Установлення Windows 95 і Windows 98. Ця операція дещо складніша, ніж установлення кожної із цих систем зокрема. Налагоджувати ПК на подвійне завантаження Windows 95 і Windows 98 треба до переходу від Windows 95 до Windows 98. Для цього необхідно виконати такі дії:

— під установленням Windows 95 відформатувати дискету 3.5″ на 1,44 Мбайт;

— клацнути на кнопці «Закрити» («Close») у корінному каталозі (C:\) ПК, скопіювати з нього файл MSDOS.SYS на дискету і використати її як дискету початкового завантаження Windows 95;

— вставити інсталяційний диск Windows 98 у дисковод CD-ROM, тримаючи натиснутою клавішу SHIFT. Ця дія відключає режим CD AUTOPLAY;

— у «Провіднику» («Explorer») Windows необхідно помістити покажчик миші на установчому диску в папці «\Win 98» і перенести CAB-файли з диску Windows 98 на жорсткий диск;

— у меню «Пуск» («Start») обрати «Завершення роботи» («Shutdown»), помітити «Перезавантажити комп'ютер у режимі емуляції MS DOS» («Restart computer in MS DOS») і клацнути на кнопці ОК;

— після перезапуску системи в DOS-режимі перейти у каталог Windows 98 (cd\Win 98) і набрати команду SETUP;

— після запиту визначити каталог, у якому буде установлюватись Windows 98, обрати підпрограму OTHER DIRECTORY. Клацнути на кнопці «Далі» («Next») і набрати ім'я нового каталогу, наприклад C:\W 98 чи C:\Windows 98. Програма установлення системи розмістить усі необхідні файли Windows 98 у створеному каталозі і перезапише файл MSDOS.SYS, щоб проінформувати систему про місцезнахо-дження робочого каталогу Windows;

— систему перезапускають, потім вона завантажує операційну систему Windows 98. Прикладні програми у новій операційній системі установлюють заново.

За необхідності виконати завантаження у Windows 95 слід увімкнути комп'ютер з дискети початкового завантаження Windows 95. Вона має копію файла MSDOS.SYS, який посилають на Windows у каталозі Windows 95 (C:\Windows або C:\W 95). Завантаження системи із цієї дискети перешле її в оригінальний каталог Windows 95. Завдяки цьому операційна система Windows 95 завантажиться.

Установлення Windows 3.x, Windows 95 і Windows NT4 можливе за наявності жорсткого диска об'ємом не менше 2 Гбайт з файловою структурою FAT. Це єдина файлова система сумісна з усіма трьома операційними системами. Послідовність установлення така: спочатку ставлять Windows 3.x, потім Windows 95 і Windows NT4. Після цього необхідно упевнитись, чи установлена кожна система у свій окремий каталог: C:\Win 3.1, C:\Win 95, C:\Win nt.

Потім, завантажуючи комп'ютер, з меню початкового завантаження Windows NT4 обирають операційну систему. За потреби використати одну прикладну програму в кожній операційній системі необхідно це зробити в кожній системі. Якщо вона має різні версії для кожної операційної системи, то слід установити кожну прикладну програму у власному каталозі під кожну операційну систему. Це запобігатиме конфліктам між прикладною системою і різними операційними системами.

Оновлення драйверів ПК

Під час модернізації обладнання ПК необхідно оновлювати і драйвери. Більшість із них входить до складу операційної системи. Однак для нових пристроїв, що увійдуть до комп'ютерної системи, слід завантажити також і драйвери. Їх оновлення підвищує ефективність ПК або надає йому нові функції, відсутні у перших версіях драйверів. Тому модернізація цих пристроїв має певний сенс.

Для оновлення драйверів насамперед використовують можливості Інтернету, в якому є багато інформації про ці пристрої. Кілька відомих Web-сайтів типу WIN DRIVERS.COM містять вичерпну інформацію про них. Є Web-сайти, які містять широкий асортимент драйверів окремих пристроїв, що легко переписується на ПК користувача.

У процесі виробництва окремих компонентів ПК, зокрема жорстких дисків, CD-ROM, сканерів, фірма «Adaptec» розробила відповідні драйвери. Широко використовують універсальний драйвер ASPI. Він підтримує периферійні пристрої й операційні системи, зокрема OS/2, UNIX, NOVELL та ін. Фірми «Futura Domain» і «NCR» розробили універсальний драйвер: CAM (Common Assess Method — метод загального доступу).

Серед драйверів пристроїв введення інформації на увагу заслуговують особливості драйверів клавіатури і миші.

Драйвер клавіатури. Він є складовою частиною будь-якої операційної системи. В операційній системі MS DOS він має назву KEYB.COM. Після установлення операційної системи драйвер, як правило, міститься в директорії DOS. Ініціалізацію цього пристрою здійснюють після набору командного рядка: KEYB.RU на запрошення DOS. Його краще помістити у файл AUTOEXEC.BAT, тоді пристрій щоразу завантажуватиметься автоматично. Драйвер клавіатури завантажують у пам'ять ПК резидентно, тобто у процесі роботи комп'ютера він постійно міститься у його пам'яті. Інші функції драйвера (погодження алфавітів країн, зміна швидкості повторення окремих клавіш за допомогою команд DOS та ін.) описані в документації на DOS.

Драйвери, які розробляють інші фірми (наприклад, KBD.COM), займають набагато менше місця в пам'яті, ніж драйвери DOS. Це альтернатива звичайним драйверам клавіатури.

Драйвер миші. Пов'язує її з операційною системою. Якщо його немає, ПК не одержуватиме інформацію через мишу. Без програмного забезпечення працюють лише миша з шинним інтерфейсом (BUS MOUSE) — її підключають до власної плати розширення, або миша, яку підключають до порту в режимі PS/2.

Стандартом для PC-комп'ютерів є так звана Microsoft-сумісна миша (MS-Mouse). Драйвер для неї дає змогу керувати всіма сумісними мишами. Для установлення пристрою в режим сумісності з MS-Mouse знаходять перемикач на корпусі миші й установлюють його в цей режим.

Слід зважати, щоб миша за можливості функціювала в режимі Microsoft. Це зумовлено двома причинами:

1) драйвери, які поставляють разом з мишею, не гарантують стовідсоткової сумісності з апаратним і програмним

забезпеченням ПК, крім того, займають порівняно великий об'єм пам'яті. Оскільки драйвер миші — це резидентна програма (миша повинна бути доступна завжди), то краще їх не використовувати;

2) миша має узгоджуватись з програмним забезпеченням. Його здебільшого розробляють під стандарт MS-Mouse. Несумісна миша з додатками Windows заважатиме ефективній роботі користувача ПК.

Підключення драйвера миші аналогічне підключенню драйвера клавіатури. Для автоматичної ініціалізації його включають в один зі стартових файлів AUTOEXEC.BAT або CONFIG.SYS. У стартовий файл AUTOEXEC.BAT вводять таку команду: C:\WINDWS\MOUSE. Підключення драйвера CONFIG.SYS здійснюють за допомогою рядка: DEVICE = C:\WINDWS\MOUSE.SYS.

Конкретніші вказівки щодо послідовності підключення драйвера миші подані в керівництві DOS або в документації на цей пристрій. Спеціальні миші мають свої оригінальні драйвери. Більшість мишей підключають через послідовний порт і поставляють з 9-контактним роз'ємом Sub-D. Інший варіант передбачає підключення через 6-контактний роз'єм. У цьому випадку йдеться про PS/2-сумісну мишу.

Щодо драйверів пристроїв виведення інформації, наприклад принтерів, то для них не так багато норм. Їх визначають провідні виробники. Тому при встановленні і оновленні драйверів для цих пристроїв слід користуватися інформацією з Інтернету і конкретною документацією на пристрої виведення і спеціальні драйвери.

З точки зору складності модернізація програмного забезпечення є чи не найскладнішою операцією у процесі модернізації ПК.

6.5. Експлуатаційне обслуговування

Надійність фунціювання ПК і комп'ютерних систем значною мірою залежить від їх експлуатаційного обслуговування. Особливу увагу слід звертати не тільки на обсяг виконаних робіт, а й на періодичність і правильність виконання технологічних операцій експлуатаційного обслуговування. Важливу роль тут відіграє правильний підбір матеріалів, використовуваний під час цього процесу.

Сутність, завдання, основні параметри експлуатаційного обслуговування

Безвідмовність є однією з найважливіших особливостей системи.

Безвідмовність *— властивість функційного модуля (системи) виконувати належні функції в певних умовах протягом заданого інтервалу часу або напрацювання.*

Важливим показником при цьому є середнє напрацювання між відмовами (на відмову). Якщо відомий інтервал часу, протягом якого використовують модуль або систему.

Середнє напрацювання між відмовами *— відношення сумарного напрацювання відновлювального об'єкта до математичного сподівання кількості його відмов протягом цього напрацювання.*

Відновлювальну систему, якщо вона справна, характеризує поняття готовності.

Готовність *— властивість об'єкта бути здатним виконувати потрібні функції в заданих умовах у будь-який час або протягом заданого інтервалу часу за умови забезпечення необхідними зовнішніми ресурсами.*

Якщо система потребує відновлення, то її характеризує поняття неготовності.

Неготовність *— стан об'єкта, в якому він нездатний виконувати потрібну функцію з будь-якої причини.*

Для того щоб систему відновити, вона повинна бути ремонтопридатною.

Ремонтопридатність *— властивість об'єкта бути пристосованим до підтримання і відновлення стану, в якому він здатний виконувати належні йому функції за допомогою технічного обслуговування і ремонту.*

На відновлення витрачається певний час. Параметром, що характеризує тривалість відновлення, є середній час відновлення.

Середній час відновлення *— середній час, потрібний для відновлення працездатності для відомого інтервалу часу використання функційного модуля (системи).*

Обслуговування розглядають як технічне обслуговування (системи) або просто обслуговування.

Технічне обслуговування системи — будь-яка діяльність, призначена для збереження або відновлення працездатності функційного модуля (системи).

Обслуговування поділяють на відновлювальне, профілактичне та періодичне (ремонт).

Відновлювальне обслуговування — обслуговування, яке проводять для відновлення працездатного стану і ресурсів функційного модуля (системи).

Профілактичне обслуговування — обслуговування, яке здійснюють для попередження відмов функційного модуля (системи).

Періодичне обслуговування — обслуговування, яке виконують через задані інтервали часу щодо встановленого часового графіка.

В експлуатаційній документації на комп'ютерні системи типу PC використовують кілька спеціальних термінів, зокрема поняття MTBF (Mean Time between Failure — середній час роботи елемента між несправностями). Розмірність цього параметра — година. Інше поняття MTTR (Mean Time to Repair — середній час ремонту).

ПК є складною комп'ютерною частиною, що складається з модулів і пристроїв, а вони, у свою чергу, — з друкованих плат, в які вмонтовані інтегральні схеми підвищеного ступеня інтеграції. Чим вищий ступінь інтеграції, тим більша ймовірність виходу мікросхем з ладу. З іншого боку, будь-який технічний пристрій, у тому числі й ПК, працює значно довше, якщо його своєчасно обслуговують.

Експлуатаційне обслуговування компонентів ПК

Для організації і здійснення експлуатаційного обслуговування слід спочатку визначити, які складові входять до ПК, оскільки окремі модулі й пристрої не обслуговують або це роблять у спеціальних сервісних лабораторіях. Однак частину пристроїв ПК обслуговують безпосередньо на робочому місці. При цьому слід перед видаленням пилу в пристроях комп'ютера відімкнути його від електромережі.

Обслуговування корпусу системного блока. Для відведення тепла у системний блок вбудований вентилятор, що

подає в корпус достатню для охолодження кількість повітря. Проте разом з ним до пристрою потрапляють пилинки, які осідають на системну плату ПК та інтегральні схеми, що призводить до збоїв і відмов.

Щоб очистити внутрішню частину системного блока, необхідно зняти кришку і м'яким пензликом видалити найбільші накопичення пилу на материнській платі, у місцях установлення елементів RAM (Random Accebs Memory — оперативна пам'ять з довільним доступом) і на блоці живлення. У добре доступних місцях пил збирають пилососом. Повітродувкою видаляють бруд у віддалених місцях за допомогою довгого і тонкого шланга. Найбільше пилу збирається на корпусі блока живлення, оскільки біля нього розташований вентилятор, що видуває повітря з протилежного боку корпусу. Блок живлення очищають від пилу, не знімаючи кришки системного блока. Бруд у його внутрішній частині видаляють за допомогою пилососа.

Недоцільне проникнення всередину блока живлення небезпечне для некваліфікованого користувача, який не є спеціалістом з технічного обслуговування ПК, зокрема блоків живлення.

За необхідності зняти кришку системного блока слід перевірити правильність і надійність підключення відповідних кабелів до системної плати та інших пристроїв. Також необхідно звернути увагу на непід'єднані кабелі, щоб визначити, чи вони не можуть спричинити коротке замикання. Їх треба ізолювати. Для усунення безладу при під'єднанні кабелів застосовують джгутування.

Щоб очистити поверхню корпусу системного блока, використовують звичайний домашній очищувач, але перед цим можна спробувати скористатись просто м'яким сукном. Не слід використовувати для очищення поверхні блока шампунь, засоби, що містять ацетон, оскільки ці речовини руйнують пластмасу передньої панелі корпусу і приводів.

Накопичувачі на жорстких дисках не потребують обслуговування. Розбирання вінчестера призводить до виходу його з ладу.

Очищення дисковода. Здійснюють за допомогою пилососа. Якщо значна кількість пилу осідає на головки зчитування-запису, то на екрані дисплея з'являється повідомлення «Can't read disk in drive A:». Однак перед прийняттям рішення про несправність дисковода слід перевірити

дискети. Якщо вони читаються на інших дисководах, то повинні читатися і на тому, який перевіряють. Після цього роблять висновок про справність або несправність дисковода.

Очищення головок зчитування-запису, як правило, виконують спеціалісти. Для загального очищення використовують пилосос, а для детального слід приготувати паличку з ватою або м'якою тканиною. Краще використати спеціальні дискети, що чистять, або аерозоль для чищення. Головки не можна зачіпати і згинати. При чищенні необхідно вологою паличкою провести 5—6 разів по кожній головці, щоб видалити частинки пилу. Після цього дисковод працюватиме знову, якщо причиною збоїв було забруднення головок.

Обслуговування CD-ROM. Воно теж зводиться насамперед до видалення пилу, зокрема з оптики приводу CD-ROM. Її очищають, коли корпус приводу відкритий. Поміщений у закритий корпус CD-ROM очищувати немає сенсу.

Для очищення CD-ROM не можна використовувати тканину, змочену в спирті. Для цього слід послуговуватись тільки аерозольним очищувачем. Оптику CD-ROM очищують потоком невологого повітря, оскільки вона вкрита спеціальним шаром речовини, що псується при вологому очищенні.

Обслуговування клавіатури. Під час обслуговування клавіатури головне не допускати потрапляння рідини на пристрій. Інколи клавіатуру накривають пластиковою кришкою. Клавіатуру часто очищують за допомогою пилососа. Великі частинки видаляють мокрою ганчіркою або пензликом, уникаючи надлишку вологи.

Коли натиснення клавіші клавіатури не відображається на моніторі або западають окремі клавіші, а зовнішнього забруднення не видно, то необхідно розкрити пристрій, викрутивши гвинти зі зворотного боку корпусу. Знявши корпус, слід оглянути плату клавіатури. Якщо на ній є сторонні предмети, то видалити їх. Якщо візуально причина несправності не визначена, то, можливо, її спричинив поганий контакт. Щоб усунути таку несправність, плату спочатку очищують сухою ганчіркою, а потім обробляють спеціальним аерозолем. Рекомендовано використовувати аерозолі «Контакт-60» або «Контакт-WL».

Якщо це не допомагає, слід вимити корпус і плату клавіатури теплою водою без мила, потім висушити протягом 1—2 діб, після цього знову скласти пристрій. До цього можна вдаватися тільки тоді, коли інші заходи неможливі або неефективні.

Обслуговування миші. Вона теж з часом забруднюється на робочому місці. Внаслідок цього маркер миші смикається на екрані монітора, не можна її зсунути ні в одному з напрямів.

Пристрій легко розбирають, видаляючи утримувач, який переміщують у вказаному напрямі, і виймаючи кульку, що там міститься. У виїмці є рухливі ролики. Якщо поглянути на кульку або ролики, то видно, де вони забруднені. Чистити звичайну мишу необхідно регулярно, а не тільки після її відмови.

Оптичні миші такого обслуговування не потребують. У них чистять лінзи на нижньому боці миші та килимок.

Обслуговування монітора. Процедура полягає в очищенні екрана і корпусу. Повітряні отвори монітора чистять за допомогою пилососа. Якщо це роблять за допомогою вологої тканини, не слід допускати попадання вологи через отвори всередину монітора, щоб запобігти короткому замиканню. Неспеціалісту відкривати монітор небезпечно, оскільки він працює під високою напругою.

Скляні поверхні моніторів чистять звичайними очищувальними засобами. Якщо поверхня монітора зі спеціальним захисним покриттям, то миючі засоби розбавляють. У будь-якому разі перед тим слід ознайомитись з рекомендаціями щодо очищення поверхні монітора.

Обслуговування принтера. Серед принтерів виокремлюють голчасті, струминні й лазерні.

Голчастий принтер. Його технічне обслуговування полягає у вчасній і систематичній заміні фарбувальної стрічки. Крім того, необхідно регулярно очищати напрямну, якою переміщається головка принтера. Інтервал між процедурами очищення напрямної залежить від тривалості використання пристрою.

Потребує обслуговування і головка принтера. На її торцях залишається фарба, яка з часом потрапляє в канали, якими рухаються голки, і засихає там. Тому при робочому русі голки блокуються, а внаслідок деформації — ламаються. Для очищення головки необхідно вийняти з принтера картридж з фарбувальною стрічкою. Пересунути головку напрямною на середину, щоб мати доступ до неї. Можна і зовсім вийняти її, але робити це слід відповідно до документації. Далі необхідно обробити головку спеціальним аерозолем («Контакт-60» або інший), зробити її рухо-

мою і видалити частинки фарби. Потім увімкнути принтер і надрукувати будь-який текст на чистому листку паперу. Повторювати це необхідно до того часу, поки головка принтера не перестане залишати на папері сліди фарби. Можна також обробити її паличкою з ватою, змоченою у спирті, і так видалити залишки фарби. При цьому не слід сильно тиснути на головку, щоб не погнути голки.

Струминний принтер. Головка цього пристрою на відміну від голчастого не потребує такої уваги при обслуговуванні. Чистити необхідно тільки вузли, вказані в документації. Більшість струминних принтерів мають вбудовані засоби очищення. Щоб запобігти висиханню чорнила і закупорюванню капілярів, слід поміщати резервуар для чорнила або всю головку в герметичний футляр.

Лазерний принтер. У ньому необхідно очищати тільки вузли, вказані в документації. Для захисту користувача від шкідливої дії озону в принтері є спеціальний озоновий фільтр. Його треба регулярно замінювати, приблизно через 2—3 місяці.

Слід також систематично видаляти з принтера залишки паперу. Щоб очистити його внутрішню частину, не слід використовувати пилосос, тому що можна пошкодити чутливі деталі. До принтера додають пристосування для очищення. За їх відсутності необхідно використати паличку з тампоном або пензлик. Для видалення розсипаного тонера використовують паличку з вологим тампоном або вологу тканину.

Профілактичні заходи при обслуговуванні ПК

Їх проведення полягає в періодичному контролі технічного стану і технічного діагностування. Кожну процедуру здійснюють через встановлені інтервали часу. Причому чим довший інтервал часу між повтореннями процесу обслуговування, тим більший обсяг робіт необхідно виконати і тим більша тривалість процесу обслуговування. Крім того, тривалість його залежить від моделі та складу ПК. Метою профілактичного обслуговування є забезпечення експлуатації ПК з коефіцієнтом використання, зазначеним у технічній документації.

Суть профілактичного технічного обслуговування полягає в перевірці компонентів і вузлів, не охоплених системою автоматичного контролю ПК, визначенні та заміні тих

компонентів і модулів, параметри яких внаслідок зношування знаходяться на межі допустимих або вийшли за неї.

Найважливішими є такі види профілактичного обслуговування комп'ютерних систем:

— добове технічне обслуговування. Охоплює зовнішній огляд технічних засобів ПК, очищення пристроїв, перевірку температури і вологості в приміщенні, роботу добового регламенту окремих пристроїв комп'ютерної системи, автоматичний контроль функціювання ПК за допомогою вбудованих програмних засобів, контроль функціювання ПК за допомогою засобів операційної системи;

— місячне технічне обслуговування. Крім програми добового обслуговування, його доповнюють огляд кріплення роз'ємів та виконанням робіт тижневого регламенту (якщо такий передбачено) окремих пристроїв ПК. Передбачає перевірку роботи пристроїв керування і сигналізації, функціювання ПК при зміні напруги вторинних джерел живлення, комплексну програму перевірки роботи комп'ютера при номінальній напрузі;

— річне технічне обслуговування. Крім програми місячного обслуговування, передбачає огляд і очищення плат, блока живлення, кабелів і джгутів, перевірку заземлення шляхом вимірювання перехідного опору, виконання робіт місячного регламенту окремих пристроїв (якщо таке передбачено), резервне копіювання системи, перевірку захисту системи електроживлення і сигналізації на випадок перегрівання процесора чи інших компонентів ПК, відпрацювання комплексу програм контролю функціювання і контролю продуктивності комп'ютера.

Обслуговування вважають закінченим, якщо не було збоїв при виконанні комплексу програм контролю функціювання комп'ютерної системи.

Серед профілактичних заходів виокремлюють активні і пасивні.

Активні профілактичні заходи — операції, які мають на меті продовжити час безвідмовної роботи ПК. Вони стосуються періодичного очищення як усієї системи, так і окремих її модулів і блоків. Активне профілактичне обслуговування охоплює операції очищення та змащення всіх основних елементів і компонентів, переустановлення мікросхем і роз'ємів (за необхідності), переформатування жорстких дисків, резервне копіювання тощо.

Пасивні профілактичні заходи — роботи, спрямовані на захист комп'ютера від зовнішніх шкідливих дій. Це — захист комп'ютерних пристроїв у мережі електроживлен-

ня, заходи підтримання чистоти і необхідної температури в приміщенні, де встановлено ПК, зменшення рівня вібрації та ін. Пасивні профілактичні заходи покликані гарантувати безпеку комп'ютерної системи.

Заходи активного профілактичного обслуговування ПК. Таке обслуговування ПК здійснюють з огляду на стан оточуючого середовища і якість компонентів системи. Якщо, наприклад, комп'ютер установлений в заводському цеху, його доведеться очищувати щомісячно. Якщо ж він в офісі, то його чистять раз на 1—2 роки. Виявивши під час профілактики багато пилу всередині комп'ютера, строки між проведенням таких робіт слід скоротити.

Серед основних заходів активного профілактичного обслуговування виокремлюють декілька.

Резервне копіювання. Цю процедуру здійснюють з метою поновлення працездатності системи після серйозних збоїв або відмов. Для резервного копіювання використовують високонадійні та високоємні пристрої зберігання. Це — магнітна стрічка, CD-R, CD-RW, ZIP або JAZZ та ін.

Регулярне очищення пристроїв комп'ютера від пилу. Одна з найважливіших робіт профілактичного обслуговування. Пил — причина багатьох несправностей. Він є теплоізолятором, а тому погіршує охолодження комп'ютерної системи. В результаті скорочується строк служби компонентів ПК. Більшість системних блоків ПК охолоджують за рахунок пониження тиску в їх корпусах. При цьому пил і різні хімічні речовини з оточуючого середовища потрапляють усередину і осідають там. З часом накопичений бруд може спричинити небажані наслідки.

Деякі компоненти ПК виділяють багато тепла, яке потрібно відводити, а пил, що осів на них, цьому перешкоджає. Крім того, він містить речовини, які проводять електричний струм. Це породжує додаткові електричні кола, що проводять сигнали. Речовини, які містить пил, прискорюють процес окислення контактів роз'ємів плат і кабелів та виводів компонентів, що встановлюють у гнізда. Все це призводить до збоїв і відмов комп'ютера.

— Очищення дисководів проводять часто, оскільки вони найбільше забруднюються. В них накопичується багато пилу і шкідливих хімічних речовин. На противагу їм жорсткі диски мають герметичну конструкцію, тому їх очищення полягає у видаленні пилу із зовнішніх поверхонь корпусу.

— Серед засобів видалення пилу — пилососи, щітки і тампони. Існують пилососи, спеціально створені для обслуговування електронних пристроїв. Вони мінімізують електричний розряд, що виникає. Перед використанням приладу пил можна частково зняти щіткою. Ними найкраще чистити корпуси блоків, крильця вентиляторів, решітки повітрозабірних отворів, клавіатуру. При цьому користувачу слід одягати антистатичний браслет із заземленням.

— Очищаючи механізми фіксації дискет у накопичувачах, напрямні, якими переміщуються блоки головок у дисководах, напрямні друкуючої головки принтера, використовують силіконові мастила. Їх перевага в тому, що вони не загусають і на них не осідає пил. Кількість мастил, що наносять на поверхню, повинна бути мінімальною. Категорично заборонено змащувати місця, які не передбачені для цієї процедури. Щоб нанести мастила на відповідні поверхні, слід користуватися губчатим тампоном, а для точкового нанесення — пластмасовою паличкою. Виконуючи операції технічного обслуговування, слід заземляти пристрої і себе, тому що можна вивести з ладу мікросхеми на платах.

— При очищенні плат необхідні тампони і очисні розчини. Спочатку очищають від пилу і бруду плату, а потім — її роз'єми. Краще це робити за допомогою спеціального пилососа або балончика зі стиснутим газом. Він особливо ефективний при здуванні пилу з плати, на якій установлено багато компонентів.

— Очищувати контакти роз'ємів необхідно для того, щоб з'єднання між компонентами і пристроями комп'ютерної системи були надійними. Чистять усі роз'єми, в тому числі й друковані. Це роблять за допомогою тампона, змоченого в очищувальному розчині. Очищення необхідно починати з позолочених контактів роз'ємів, а потім переходити до інших. Особливо ретельно протирають контакти роз'ємів плат, які вставляють у роз'єми на системній платі (адаптери). Для цього використовують засіб з добавленим струмопровідним мастилом. Це полегшує встановлення плати адаптера у слот і захищає контакти роз'єму від окислення. Роз'єми плоских кабелів протирають тим самим розчином.

— Очищення клавіатури проводять за допомогою пилососа. Для цього її перевертають клавішами донизу і продувають струменем стиснутого повітря. При залипанні будь-якої клавіші знімають її ковпачок і бризкають трохи очищувача безпосередньо на перемикач, потім капають у її контактний вузол. Для цього немає потреби розбирати

клавіатуру. Проблем із залипанням клавіш і поганими контактами буде менше, коли систематично очищувати клавіатуру за допомогою пилососа чи балончика зі стиснутим газом.

— Щоб почистити мишу, достатньо відкрити кришку, яка закриває відсік з кулькою, і витягнути її з гнізда. Потім кульку протирають очищувальним розчином. Не слід використовувати для цього очищувач з мастилом, тому що вона буде сковзати, а не котитися по столі. Після цього щіткою або змоченим у розчині тампоном чистять ролики, яких торкається кулька всередині корпусу миші. Під час роботи на ПК миша повинна перебувати на спеціальному килимку, тоді вона не так забруднюється.

Періодичне установлення мікросхем на місце. Оскільки при вмиканні та вимиканні ПК нагрівається й остигає, то відповідно розширяються і стискаються його компоненти. Мікросхеми, установлені в сокетах, з часом перестають щільно триматися там. Це стосується мікросхем пам'яті, які установлюють в сокети чи входять до складу модулів SIMM або DIMM (Dual In-Line Memory Module — модуль пам'яті з двосторонніми друкованими виводами (ОЗП)). Їх фіксують у роз'ємах за допомогою спеціальних фіксаторів. Однак у модулів SIMM їх немає. Тому нерідко ці компоненти не щільно тримаються у своїх слотах.

У сокети установлюють також мікропроцесор, співпроцесор, мікросхеми ROM. Мікропроцесори у багатьох комп'ютерах поміщають у гнізда ZIF — з нульовим зусиллям вставляння. У них є важіль, за допомогою якого затискують або вивільняють одразу всі виводи установленої мікросхеми. В таких сокетах мікросхеми, як правило, тримаються щільно.

Для того щоб вставити мікросхему в сокет, необхідно натиснути на неї, обов'язково притримуючи при цьому плату зі зворотного боку. Все це слід робити обережно. Великі мікросхеми (процесор, співпроцесор) установлюють, легко натискуючи спочатку з одного боку, а потім з іншого, поки вони не стануть на своє місце. Плати при цьому виймають із роз'ємів або корпусу. Перевіряти, чи щільно тримаються мікросхеми на своїх місцях і установлювати їх за потреби слід не менше одного разу на рік.

Заходи страхування від втрати даних. Вони належать до робіт з профілактичного обслуговування жорстких дисків. Для цього існують програми створення резервних копій критичних зон жорсткого диска, які за необхідності можна відтворити. Критичними в даному разі є зони, при пошкодженні яких доступ до файлів стає неможливим.

З метою збереження інформації проводять раз на тиждень дефрагментацію жорсткого диска. Також це роблять після контактної операції резервного копіювання. Програми дефрагментації виконують такі функції:

— дефрагментацію файлів;

— ущільнення файлів з упорядкуванням вільного місця;

— сортування файлів.

До складу операційних систем входять різні програми дефрагментації. Зокрема, Windows 9x містить програму, що працює з файловими системами FAT16 і FAT32. Програми дефрагментації для цих систем несумісні. Тому не слід запускати у Windows 9x програми SCANDISK for DOS або NORTON DISK DOCTOR. До складу операційної системи Windows 98 входить програма «Майстер обслуговування» («Maintenance Wizard»), яка автоматизує виконання певних процесів під час профілактичного обслуговування. Запустивши цю програму і обравши в ній необхідні підпрограми, файли і розклад обслуговування, користувач, не відволікаючись, може займатися основною роботою, оскільки ці функції виконуватимуться автоматично.

До активних профілактичних заходів належать і систематичне виявлення та знищення вірусів. Їх здійснюють перед кожною операцією резервного копіювання жорсткого диска.

Заходи пасивного профілактичного обслуговування ПК. Пасивні профілактичні заходи зумовлені фізичним впливом температури оточуючого середовища, пилом, перепадом температур при вмиканні і вимиканні ПК, вібрацією та механічним ударом. Крім того, на ПК діють електричні впливи, зокрема електростатичні розряди, завади в колах живлення, радіочастотні завади. Головна мета будь-яких профілактичних робіт — зберегти комп'ютерну систему й експлуатувати її якомога довше. Для цього слід дотримуватись декількох вимог.

Звертати увагу на організацію робочого місця й оточуючого середовища. Передусім там не має бути пилу. Не слід ставити комп'ютер біля вікна, радіопередавальних пристроїв та інших джерел випромінювання, оскільки на нього впливають сонячне світло і перепади температури. Розетки для вмикання комп'ютера повинні бути заземленими, напруга в електромережі стабільна.

Температуру в приміщенні, де установлений комп'ютер, слід підтримувати постійну. Існує допустимий діапазон тем-

ператур при експлуатації та зберіганні ПК. Для комп'ютерів фірми «IBM» цей діапазон при експлуатації — від +15 до +32°С, при зберіганні — від +10 до +43°С. Якщо комп'ютер заносять з морозу або повітря, температура яких нижче вказаного діапазону, то спочатку необхідно дати йому прогрітися до кімнатної температури, а потім вмикати.

Під час експлуатації ПК його слід якомога рідше вмикати і вимикати. Постійно увімкнутий комп'ютер матиме довший строк експлуатації. Однак доведеться враховувати вартість електроенергії, пожежну безпеку і т. д. З огляду на ці причини не можна залишати комп'ютер увімкнутим на ніч і на вихідні дні. Його вмикають один раз на день, але не частіше. Якщо комп'ютер залишається довго увімкнутим і на ньому не працюють, то краще вимкнути монітор. Коли на екран ПК довго виводять статичне зображення, то вигорає люмінофор кінескопа. Сучасні монітори підтримують функцію енергозбереження і за командою системи автоматично переходять у режим очікування.

Заземлювати комп'ютерну систему. Для цього мережний шнур електроживлення системи слід під'єднати до розетки з трьома гніздами. Це роблять, тому що серйозну загрозу для компонентів комп'ютера, насамперед для інтегральних схем, становлять електростатичні заряди. Вони особливо небезпечні за низької вологості повітря, а також у районах із сухим кліматом. Тому в таких умовах необхідно дотримуватись заходів безпеки при роботі з ПК. Одним з них є надійне заземлення комп'ютера.

Крім того, необхідно бути особливо обережним, відкриваючи системний блок або працюючи з окремими платами, вийнятими з комп'ютера. Щоразу, виймаючи з корпусу окремі плати, для вирівнювання електростатичного потенціалу слід братися за ділянки, наприклад за кронштейн, з'єднані із загальним дротом.

Не слід підключати комп'ютер до тієї самої мережі, до якої підключені пристрої великої потужності. Це допоможе запобігти впливу завад у мережі електроживлення. Перепади напруги, що виникають у результаті вмикання і вимикання ПК, одразу впливають на його роботу. Перехідні процеси, зокрема сплеск напруги з амплітудою до 1000 В і вище, можуть вивести з ладу блок живлення. Обираючи місце і спосіб підключення ПК до електромережі, слід враховувати такі особливості і вимоги:

— за можливості підключати комп'ютери до окремої лінії електроживлення зі своїми запобіжниками (автоматичними);

— перевіряти опір шини заземлення, який повинен перебувати у допустимих межах;

— під'єднувати пристрої і блоки комп'ютера до електромережі за допомогою триштиркових вилок, не користуючись перехідниками з двома гніздами, оскільки в такому разі пристрої і блоки будуть незаземленими;

— не користуватись без потреби подовжувачами. За потреби необхідно їх обирати з урахуванням потужності споживачів електроенергії;

— для під'єднання пристроїв, що не стосуються комп'ютерної системи, краще скористатись іншою розеткою, а не тією, до якої підключений комп'ютер.

Не слід розетки електроживлення кількох комп'ютерів з'єднувати послідовно. Це призводить до того, що в комп'ютерах, підключених останніми в цьому послідовному колі, падає напруга.

Отже, при дотриманні правил і рекомендацій ПК буде служити довго і надійно, а необхідність розбирати його виникатиме тільки під час планового профілактичного обслуговування.

Запитання. Завдання

1. Обгрунтуйте поняття «модернізація ПК».
2. На яких засадах приймається рішення про модернізацію комп'ютера?
3. Яка послідовність дій при заміні системної плати?
4. Опишіть процес установлення нових модулів оперативної пам'яті.
5. Охарактеризуйте дії при модернізації системного блока ПК.
6. Дайте загальну характеристику процесу установлення портів і плат розширення при модернізації ПК.
7. Яка послідовність нарощування відеопам'яті графічного адаптера і які можливі конфігурації при цьому?
8. Які особливості установлення модемів?
9. Назвіть дії, які необхідно виконати для установлення другого IDE-вінчестера.
10. Опишіть установлення нового 3,5"-дисковода.
11. Назвіть і охарактеризуйте типи накопичувачів CD-ROM. Чим відрізняється процес їх установлення?
12. Яка послідовність дій при установленні DVD-накопичувача?
13. Назвіть операції, які необхідно здійснити для установлення нової графічної плати.
14. Опишіть процедуру установлення нового монітора.
15. У чому полягає процес установлення в ПК нової клавіатури?

16. Охарактеризуйте послідовність дій при заміні миші в ПК.
17. Яких правил слід дотримуватись при установленні принтера в комп'ютерній системі?
18. Які особливості установлення сканера в ПК?
19. У чому суть оновлення BIOS?
20. Опишіть процес переходу від операційної системи MS DOS до Windows 98.
21. Як здійснити перехід від Windows 3.1 до Windows 98?
22. Назвіть послідовність дій при установленні на ПК Windows NT4.
23. Охарактеризуйте операції установлення на ПК Windows 2000.
24. Як установити на один ПК дві операційні системи, зокрема Windows 95 і Windows 98?
25. У чому суть поняття «експлуатаційне обслуговування ПК»?
26. Які особливості обслуговування накопичувачів на жорстких дисках?
27. Яка послідовність дій при обслуговуванні клавіатури?
28. Назвіть операції, які здійснюють при обслуговуванні миші.
29. Яким чином обслуговують монітор?
30. Які операції проводять при обслуговуванні голчастого, струминного, лазерного принтерів?
31. Назвіть і охарактеризуйте види профілактичного обслуговування ПК.

Додаток

Поширені програмні помилки користувальницького інтерфейсу

Тип помилки	Ознаки прояву помилки
1	2
1. Функційна помилка:	За допомогою програми неможливо або незручно виконати те, чого очікує користувач:
— пропущена функція;	у програмі відсутня функція, що задана в специфікації;
— функція неправильно працює;	виконує іншу функцію, а не ту, що вказана в специфікації;
— програма не виконує те, чого чекає від неї користувач;	програма не виконує тих функцій, які очікує від неї користувач;
— виконання надлишкових функцій;	функції програми, які рідко використовують, ускладнюють задіяння її базових можливостей, тобто рівень функційності програми виходить з-під контролю;
— неадекватність реалізації базових функцій	одна з базових функцій реалізована дуже вузько, до неї не можна звернутися або програма працює дуже повільно, її не можна реально експлуатувати

Продовження

1	2
2. Помилки взаємодії програми і користувача:	Помилки взаємодії інтерактивної програми або пакетних програм, які видають певну інформацію про помилки, з користувачем:
— відсутня інструкція на екрані;	на екрані немає інструкцій про назву програми, як з неї вийти, отримати довідникову інформацію, ознайомитися з переліком команд командної мови, яку використовують у програмі;
— тривала відсутність активності;	користувач не бачить, що програма працює, а не зупинилась. При цьому бажано, щоб індикатор відображав, яку частину завдання вже виконано;
— нёмає курсору;	немає курсору (текстового або вказівника миші) у будь-якій інтерактивній програмі;
— програма не розпізнає введення інформації;	інтерактивна програма не реагує на введення інформації користувачем, крім випадків, коли вона перебуває в процесі переходу з одного стану в інший, ігнорує певні дії користувача (дано команду про заборону відображення введення інформації, а вводиться пароль чи інша секретна інформація);
— неможливо визначити, чи дана програмі команда виконана;	видалені дані залишаються на екрані доти, доки користувач не вийде з поточного режиму;
— неможливість визначити, чи один і той же документ відкритий кілька разів;	програма не перевіряє кожний документ, що відкривається, і не повідомляє користувачеві, якщо він відкрився повторно;
— стан, з якого важко знайти вихід;	складно з'ясувати, як вийти з небажаного стану програми або перервати такий процес;
— проста помилка	після того як користувач обрав певний режим або команду, на екрані залишається інформація попереднього режиму, яка зайва і не стосується справи

Продовження

1	2
3. Неправильна або сумнівна з точки зору користувача інформація:	Незначні з першого погляду помилки або повідомлення програми, через які користувач може зробити неправильні узагальнення або висновки:
— проста помилка;	сплутування дійсного стану програми з інформацією попереднього режиму, що залишилася на екрані і не стосується режиму або команди, яку розглядають;
— синтаксичні помилки;	невиправлені через неуважність програмістів помилки (помилки правопису);
— невдала зовнішня структура програми;	користувачу важко розібратися у призначенні компонентів програми. Це збільшує час на її вивчення. Для виконання певного завдання він повинен знати про програму якнайменше;
— кілька назв однієї й тієї самої функції;	одна й та сама функція має в програмі кілька назв;
— неточні назви команд і функцій	назвам команд і функцій не відповідають закріплені за ними значення українською (англійською) мовою
4. Помилки, пов'язані з повідомленнями про помилки:	Не звертається належна увага на правильність повідомлень про помилки:
— невідповідність і високий рівень складності тексту повідомлень про помилки;	текст інтерактивної довідникової системи і повідомлень про помилки складніший за текст друкованого керівництва;
— фактичні помилки;	у документації і повідомленнях програми наведено неправильні приклади про те, як правильно виконати ті чи інші дії;
— контекстові помилки;	підсистеми оброблення помилок і контекстово-залежні довідникові системи не перевіряють, що робить програма в момент їх виклику. Їхні повідомлення, меню, рекомендації та інше не залежать від ситуації;
— неправильне визначення джерела помилки;	у повідомленні про помилку не сказано, яка саме помилка виникла, не приведені помилкові дані, не вказана причина і спосіб виходу з помилкової ситуації;
— відмова у наданні ресурсів без пояснення причини;	не повідомляються причини відмови програмі використати ресурси комп'ютерної системи;

Продовження

1	2
— повідомлення про хибні помилки;	програма повідомляє про хибні помилки або в повідомленнях про помилки є інформація про рядові події;
— на екрані дисплея є два курсори;	така ситуація — відображення переплутаних операторів у програмному коді, невідомо, який з двох курсорів активний;
— зникнення курсору;	поверх курсору користувач відображає певну інформацію, змінивши його координати і не перемалювавши курсор;
— курсор відображається не на тому місці;	курсор зображено в одному місці екрана монітора, а дані, що вводить користувач, відображаються на іншому;
— курсор міститься поза областю введення даних;	курсор міститься поза областю введення даних;
— відображення введення даних не на тому місці екрана;	курсор відображається правильно, а дані, що вводять, відображаються не на тому місці екрана;
— відображений неправильний або неповний рядок;	повідомлення, що відображається на екрані, має вади, наприклад фрагмент відображеного повідомлення довший або коротший за саме повідомлення;
— повідомлення на екрані залишається дуже довго або зникає дуже швидко.	повідомлення на екрані залишається задовго або зникає швидше, ніж користувач встиг ознайомитися з ним. Причиною цього можуть бути «змагання» сигналів
5. Помилки організації екрана:	Екран не виглядає належним чином:
— невдала організація меню;	спосіб вибору елемента меню не є очевидним і не вказаний на екрані, команди меню не є незалежними, подібні або концептуально пов'язані команди меню не об'єднані в окремі групи і чітко не відділяються одна від одної;
— помилки організації діалогових вікон;	діалогові вікна не мають стандартизованого інтефейсу, зокрема не виводять інформацію в одному і тому самому місці екрана, текст не відображають одним і тим самим шрифтом та інше, зв'язані елементи діалогового вікна не розташовані поруч, а їх групи нечітко відділені одна від одної;

Продовження

1	2
— неможливість звільнитися від надлишкової інформації на екрані	неможливо усунути відображення на екрані меню програми і панелі інструментів у разі необхідності
6. Помилки організації команд і способів їх введення:	Помилки проявляються при організації команд або їх введенні:
— непослідовні правила завершення введення команд;	незручні і нелогічні для користувача правила повідомлення програмі про завершення введення даних у полі;
— невідповідність опцій;	опція, що має однаковий зміст для кількох команд, недоступна для них усіх;
— подібні назви команд;	назви двох команд схожі, їх легко сплутати;
— неоднакове розташування команди в меню;	одна й та сама команда трапляється в кількох меню або підменю, але не розміщується в одній і тій самій позиції;
— непослідовність правил опрацювання помилок;	поведінка конкретної програми не досить послідовна і передбачувана;
— непослідовність правил редагування;	для зміни будь-яких введених раніше даних повинні застосовуватись одні й ті самі клавіші та команди;
— непослідовність правил зберігання даних	програма не завжди і не всюди зберігає дані однаково, зокрема з одними й тими самими інтервалами часу і командами
7. Помилки організації меню:	Помилки, які полягають у неповній інформативності назв команд:
— неможливість переходу між певними станами;	після виконання користувачем певних дій програма не дає йому змоги вийти з того режиму, в якому він працював;
— зв'язані команди містяться в різних меню;	зв'язані команди містяться в різних меню;
— не розпізнаються повні імена команд;	скорочені імена команд розпізнають, а повні — ні;

Продовження

1	2
8. Нестандартне використання клавіатури:	за клавішами стандартної клавіатури не закріплено певні традиційні команди:
— неможливість використання клавіш керування курсором, функційних клавіш і клавіш редагування;	не працює одна або декілька зі згаданих клавіш;
— нестандартне використання клавіш керування курсором і редагування;	клавіші не працюють так, як звикли користувачі тих комп'ютерів, для яких призначена програма, яку в даний час відпрацьовують;
— відсутність індикаторів стану клавіатури;	не можна в будь-який момент визначити стан клавіатури, використовуючи індикатори на клавіатурі та підказки на екрані;
— відсутність реакції на керуючі клавіші	неможливо перервати поточну операцію, програма не розпізнає стандартні системні клавіші
9. Пропущені команди:	Відсутність у деяких програмах необхідних команд і функцій:
— переходи між станами;	не сприймається повідомлення користувача про те, що він хоче повернутися до попереднього стану;
— неможливо вийти з підпрограми;	неможливо перервати поточне завдання, яке виконує програма, і повернути дані до вихідного стану;
— неможливо перервати виконання команди;	неможливо перервати поточну команду, що виконує програма;
— неможливо призупинити роботу програми;	неможливо повідомити програмі про необхідність перервати роботу і через деякий час продовжити її з того самого місця;
— немає команди відміни;	неможливо відмінити останні зміни чи виконану останню команду;
— немає можливості періодично зберігати дані;	при збоях частина даних, що періодично зберігалася, не зберігається;
— не передбачено умови для користувачів, щоб перевірити дані, які вводять;	неможливо вказати, які дані допустимі в кожному полі;

Продовження

1	2
— важко виправити допущену помилку;	виправлення помилки, яка допущена користувачем, не повинно викликати труднощів;
— немає засобів відображення зв'язків між змінними;	немає засобів візуального аналізу зв'язків між змінними;
— надлишок захисту;	засоби захисту програми спрацьовують без потреби, наприклад необхідно без кінця вводити пароль;
— неможливо заховати меню	неможливо прибрати з екрана елементи меню
10. Помилки обчислень:	Програма здійснює обчислення та одержує неправильний результат:
— застарілі константи;	константи в нових версіях програм змінені тільки в певних місцях, а не у всій програмі, що є джерелом безлічі помилок;
— помилки обчислень;	замість одних функцій виконуються інші або переплутуються їх параметри;
— неправильно поставлені дужки;	неправильно поставлені дужки у виразах з багатьма дужками;
— неправильна послідовність операторів;	програма виконує обчислення не в тій послідовності, в якій очікує програміст;
— переповнення і втрата значущих розрядів;	втрата значущих розрядів або перетворення результату на нуль;
— помилки відтинання і округлення;	виконані обчислення є неточними;
— неправильне перетворення даних з одного формату в інший	виникнення помилок у фрагментах програмного коду, в яких виконують перетворення формату даних
11. Помилки, джерела яких — апаратне забезпечення:	Програма не розпізнає непрацездатність пристрою або системи і не вживає відповідних заходів:
— неправильний пристрій;	програма вводить дані на один пристрій замість іншого;
— неправильна адреса пристрою;	програма записує дані не за тією адресою, яку задають;
— використання пристрою заборонено певному користувачеві або програмі;	програма не вміє обробляти відмову в наданні певного пристрою;

Закінчення

1	2
— даний рівень привілеїв не дає змоги одержати доступ до пристрою;	програма не вміє обробляти відмову, визначену рівнем привілеїв;
— помилки, пов'язані з «шумами»;	спотворення сигналів, що передаються комутаційними каналами, через електричні наводки та ін.;
— переривання зв'язку;	переривання зв'язку при роботі одного комп'ютерного пристрою з іншим;
— неправильний накопичувач;	неможливо знайти необхідні файли на відповідних накопичувачах і повідомити про це користувачу;
— не перевіряється вміст поточного диска;	непомітність заміни дисків у дисководі, копіювання каталогу диска в оперативну пам'ять і не зчитування його до того часу, поки не ініціює користувач;
— незакритий файл;	псування файла при вимкиенні комп'ютера;
— несподіваний кінець файла;	програма проігнорувала маркер кінця файла і продовжує зчитування файла;
— помилки, пов'язані з довжиною файлів і дисковими секторами;	останній символ кожного сектору чи файла скопійований неправильно або двічі, втрачений або програма псує весь файл, що міститься після нього на диску;
— неправильний код операції або команди;	програма не знає, з яким пристроєм вона працює, і передає йому неправильні команди;
— неправильно інтерпретований код стану або повернення;	програма ігнорує коди помилок або шукає їх у старому чи неправильному списку при виконанні команди або повідомленні про неможливість її виконання;
— помилка протоколу обміну з пристроєм;	неправильний формат даних, пристрої відправляють дані чи відповідають не за тією адресою;
— ігнорування або неправильне використання механізму сторінкового кодування пам'яттю;	програма неправильно переключається між сторінками або звертається до них, звертається до сторінки, якої немає, витирає її вміст у пам'яті, не зберігаючи його на диску;
— помилки, що пов'язані з програмованими функційними клавішами	програмованим клавішам не відповідають правильні послідовності кодів і відповідні функції

Література

А.с. 1705782 СССР, МКИ G01 R 31/28. Автоматизированная система тестового контроля и диагностирования цифровых микропроцессорных блоков / Локазюк В. Н. // Открытия. Изобретения. — 1992. — №2. — С. 180.

А.с. 1762280 СССР, МКИ G01R 31/28. Устройство для контроля цифровых блоков / Локазюк В. Н. // Открытия. Изобретения. — 1992. — №34. — С. 184.

Байда Н. П., Кузьмин И. В., Шпилевой В. Г. Микропроцессорные системы поэлементного диагностирования РЭА. — М.: Радио и связь, 1987. — 256 с.

Байда Н. П., Месюра В. И., Роик А. М. Самообучающиеся анализаторы производственных дефектов РЭА. — М.: Радио и связь, 1991. — 256 с.

Безруков Н. Н. Компьютерная вирусология: Справ. рук-во. — К.: УРЕ, 1991.

Бэрри Нанс. Компьютерные сети / Пер. с англ. — М.: БИНОМ, 1995. — 400 с.

Вальков В. М. Контроль в ГАП. — Л.: Машиностроение. Ленинградское отделение, 1986. — 232 с.

Вентцель Е. С. Теория вероятностей. — М.: Наука, 1969. — 576 с.

Городецкий В. Э., Тыдыков В. П. Методы тестирования микропроцессорных схем // Зарубежная радиоэлектроника. — 1986. — № 10. — С. 20—34.

ГОСТ 23564-79. Техническая диагностика. Показатели диагностирования. — М.: Издательство стандартов, 1980.

Гуляев В. А., Кудряшов В. И. Автоматизация наладки и диагностирования микроУВК. — М.: Энергоатомиздат, 1992. — 256 с.

ДСТУ 2389-94. Технічне діагностування та контроль технічного стану. Терміни та визначення.

ДСТУ 2873-94. Системи оброблення інформації. Програмування. Терміни та визначення.

Жердев Н. К., Креденцер Б. П., Белоконь Р. Н. Контроль устройств на интегральных схемах.— К.: Техніка, 1986. — 160 с.

Зинглер К. Методы проектирования программных систем / Пер. с англ. — М.: Мир, 1985. — 328 с.

Иыду К. А. Надежность, контроль и диагностика вычислительных машин и систем: Учебн. пособие для вузов. — М.: Высш. шк., 1989. — 216 с.

Каган Б. М., Мкртумян И. Б. Основы эксплуатации ЭВМ: Учебн. пособие для вузов / Под ред. Б. М. Кагана. — М.: Энергоатомиздат, 1983. — 376 с.

Канер С. и др. Тестирование программного обеспечения / Пер. с англ. / Сэм Канер, Джек Фолк, Енг Кек Нгуен. — К.: Издательство «Диа Софт», 2000. — 544 с.

Колесниченко О. В., Шишигин И. В. Аппаратные средства PC. — 3-е изд., перераб. и доп. — СПб: БХВ — Санкт-Петербург, 1999. — 800 с.

Кондратьев В. В., Махалин Б. Н. Автоматизация контроля цифровых функциональных модулей. — М.: Радио и связь, 1990. — 152 с.

Кулаков А. Ф. Оценка качества программ ЭВМ. — К.: Техніка, 1984. — 167 с.

Левин В. И. Логико-динамический синтез цифровых устройств // Автоматика и вычислительная техника. — 1982.— №3.— С. 36—43.

Липаев В. В. Качество программного обеспечения. — М.: Финансы и статистика, 1983. — 263 с.

Липаев В. В. Надежность программного обеспечения АСУ. — М.: Энергоиздат, 1981. — 240 с.

Лихтциндер Б. Я. Внутрисхемное дигностирование узлов радиоэлектронной аппаратуры. — К.: Техника, 1988. — 168 с.

Локазюк В. М. Контроль і діагностування обчислювальних пристроїв та систем: Навч. посібник для вузів. — Хмельницький: ТУП, 1996. — 175 с.

Локазюк В. М. Мікропроцесори та мікроЕОМ у виробничих системах: Навч. посібник для вузів. — К.: «Академія» (Альма-матер), 2002. — 368 с.

Локазюк В. М., Поморова О. В., Домінов А. О. Інтелектуальне діагностування мікропроцесорних пристроїв та систем: Навч. посібник для вузів. — К.: «Такі справи», 2001. — 286 с.

Локазюк В. Н., Карякин В. А. Комбинированное диагностирование и надежность вычислительных устройств. — Хмельницкий: Поділля, 1994. — 128 с.

Локазюк В. Н., Огневой А. В., Хмельницкий Ю. В. Повышение достоверности и интеллектуализация диагностирования вычислительных структур на основе моделей нейронных сетей // Автоматизація виробничих процесів. — 1997. — №1. — С. 103—108.

Людино-машинні системи автоматизації управління якістю, безпекою і надійністю / Архангельський В. І., Богаєнко І. М., Грабовський Г. Г., Рюмшин М. О. — К.: НВК «КІА», 2000. — 296 с.

Майерс Г. Искусство тестирования программ / Пер. с англ. — М.: Финансы и статистика, 1982. — 156 с.

Майерс Г. Надежность программного обеспечения / Пер. с англ. — М.: Мир, 1980. — 360 с.

Марголис А. Поиск и устранение неисправностей в персональных компьютерах. — К.: Фирма «Диалектика», 1994. — 368 с.

Медведев А. М. Надежность и контроль качества печатного монтажа. — М.: Радио и связь, 1986. — 216 с.

Модернизация и обслуживание ПК. Базовый курс / Пер. с англ. — К.: ВЕК+, М.: ЭНТРОП, М.: Корона-Принт, 2000. — 592 с.

Молчанов А. А. Моделирование и проектирование сложных систем. — К.: Вища школа, 1988. — 359 с.

Мюллер С. Модернизация и ремонт ПК, 11-е издание / Пер. с англ.: Учебное пособие. — М.: Издательский дом «Вильямс», 2000. — 1136 с.

Мячев А. А. Интерфейсы средств вычислительной техники: Справочник. — М.: Радио и связь, 1993. — 352 с.

Надежность автоматизированных систем управления / Аповмян И. О., Вайрадян А. С., Руднев Ю. П. и др. / Под ред. Я. А. Хетагурова. — М.: Высшая школа, 1979. — 288 с.

Нейронные сети в системах автоматизации / В. И. Архангельский, И. Н. Богаенко, Г. Г. Грабовский, Н. А. Рюмшин. — К.: «Техніка», 1999. — 364 с.

Нортон П. Программно-аппаратная организация IBM PC / Под ред. В. Г. Абрамова. — М.: Радио и связь, 1992.

Оптимизация технико-экономических характеристик радиоаппаратуры / С. Е. Алексеев и др. / Под ред. В. К. Маригодова. — К.: Техніка, 1990. — 192 с.

Смирнов Н. И., Стручков А. А., Судовцев В. А. Диагностика неисправностей в цифровой радиоаппаратуре на БИС // Зарубежная радиоэлектроника.— 1979.— №1.— С. 53—60.

Справочник проектировщика АСУ ТП / Г. Л. Смилянский и др. — М.: Машиностроение, 1983. — 527 с.

Тейер Т., Липов М., Нельсон Э. Надежность программного обеспечения. — М.: Мир, 1981.

Тестовое диагностирование логических структур / В. А. Пелипейко, И. А. Анучин, В. К. Жуляков и др. / Под ред. В. А. Пелипейко. — Рига: Зинатне, 1986. — 282 с.

Харченко В. С., Литвиненко В. Г., Мельников В. А. Методы повышения отказоустойчивости СБИС бортовых цифровых вычислительных комплексов // Зарубежная радиоэлектроника. — 1990. — №12. — С. 56—76.

Шураков В. В. Надежность программного обеспечения систем обработки данных: Учебник для вузов. — М.: Статистика, 1981. — 216 с.

Щербаков Н. С. Достоверность работы цифровых устройств. — М.: Машиностроение, 1989. — 224 с.

A Universal Approach to the problems of testing electronic printed circuit board assamblies // Membrain Limited, 23. Cobhaw Road, Ferndown Industrial Estate, Wimborne, 1980. — P. 70.

Cheung R. C. A User-Oriented Software Reliable Model. — IEEE Transactions on Software Engineering, March 1980, vol. SE-6, №2, p. 118 — 125.

Diagnosing beyond the node with Membrain's FLO-TRACER. Application Report-1 // Membrain — Schlumberger.— 1981. — P. 6.

Mills H. D. On the Statistical Validation of Computer Programs, FSC — 72 — 6015, IBM Federal Systems Div., Gaithersburg, Md., 1972.

Richards F. R. Computer Software: Testing, Reliability, Models, and Quality Assurance, NRS-55RH74071A, Naval Postgraduate School, Monterey, Ca., 1974.

Rubey R. J., Dana J. A., Bichl P. W. Quantitative aspects software validation. — IEEE Trans, on Software Engineering, 1975, vol. SF-1, №2.

Saglietti F., Ehrenberder W. und Kersken M. Software — Daversitat fur Stenerungen mit Sicherheitsverantwortung // Bundesanstalt fur Arbeitersschutz. — Wirschaftverlag NW, 1992.

Короткий термінологічний словник

Автономний програмний монітор — сукупність прикладних програм операційної системи. Вони дають змогу фіксувати широкий спектр подій при трасуючому і вибірковому методах вимірювань. Призначені для видавання розширених відомостей про систему.

Активні профілактичні заходи — операції, які мають на меті продовжити час безвідмовної роботи ПК.

Безвідмовність — властивість функційного модуля (системи) виконувати належні функції в певних умовах протягом заданого інтервалу часу або напрацювання.

Вбудований програмний монітор — сукупність програм, що входять до складу операційної системи. Він має статус керуючої програми операційної системи і призначений для видавання мінімальних відомостей про систему.

Вектор помилки $E = (e_1, e_2, \ldots, e_n)$ — двійкова комбінація, яка містить одиниці в розрядах слова, що спотворені.

Відмова — подія, яка полягає в утраті функційним модулем (системою) здатності виконувати потрібну функцію.

Відновлювані об'єкти — об'єкти, які в період експлуатації в разі виникнення відмов можуть бути відремонтовані шляхом заміни несправних компонентів.

Відновлювальне обслуговування — обслуговування, яке проводять для відновлення працездатного стану і ресурсів функційного модуля (системи).

Відповідна реакція (вихідна реакція) — сукупність логічних контрольованих сигналів на вихідних контактах у заданому такті контролю чи діагностування, які є реакцією об'єкта на подані на його входи тестові впливи.

Внутрішній модем — модем, що знаходиться всередині конструкції системного блока.

Вторинна помилка — спотворення вихідних результатів при виконанні програми.

Глибина пошуку місця відмови (несправності) — характеристика, де вказують складову частину об'єкта, з точністю до якої визначають місце відмови.

Готовність — властивість об'єкта бути здатним виконувати потрібні функції в заданих умовах у будь-який час або протягом заданого інтервалу часу за умови забезпечення необхідними зовнішніми ресурсами.

Достовірність діагностування — ступінь об'єктивної відповідності результату контролю дійсному технічному стану об'єкта.

Достовірність помилкового функціювання — властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати помилковим результат роботи пристрою за наявності хибних сигналів помилок, що видаються засобами контролю.

Достовірність правильного функціювання — властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати правильним результат роботи пристрою за наявності пропуску помилок засобами контролю.

Достовірність функціювання — властивість обчислювального пристрою, що характеризує здатність засобів контролю визнати результат роботи пристрою правильним або помилковим за наявності пропуску помилок чи видачі хибних сигналів помилок засобами контролю.

Драйвер — програма, що імітує надходження інформації з попередніх модулів.

Експертні системи технічної діагностики — пакет програм, що класифікує ОД і несправності, які виникають у них, проводить їх аналіз, видає консультації і ставить діагноз.

Експлуатаційна ситуація — обставини, що зумовлюють вплив зовнішнього середовища, мету і режими функційного використання системи, запит на систему і результати її функціювання.

Еталонні реакції (реакції, що вимагаються) — сукупність логічних контрольованих сигналів на вихідних контактах, які відповідають розрахованим для справного ОК (ОД) відповідним реакціям у заданих тактах контролю (діагностування).

Ефективність (лат. effectivus — дійовий) **системи діагностування** — ступінь її пристосованості до процесу визначення технічного стану ОД і пошуку в ньому дефектів.

Жорстке ядро — ядро, склад якого постійний.

Збій — короткочасне порушення правильної роботи цифрового чи мікропроцесорного пристрою, після якого його працездатність відновлюється або її відновлює оператор без проведення ремонту.

Зовнішній модем — модем, що знаходиться за межами системного блока.

Інтеграція (лат. integratio — поповнення) **модулів** — злиття модулів у програму або програмну систему.

Інтенсивність відмов — інтегральний критерій l(t), що характеризує густину розподілу напрацювання до першої відмови, розраховану за умови, що до моменту часу, який розглядають, система пропрацювала безвідмовно.

Ймовірність безвідмовної роботи — ймовірність того, що при заданих умовах експлуатації протягом інтервалу часу t не виникне відмова.

Ймовірність безвідмовної роботи P(t) — функція розподілу ймовірностей безвідмовної роботи об'єкта, яка характеризує ймовірність того, що об'єкт за час t не втратить працездатності.

Ймовірність невиявленої відмови (несправності) в даному елементі (групі) — ймовірність того, що за наявності відмови в результаті діагностування приймається рішення про її відсутність у даному елементі (групі).

Ймовірність невиявленої відмови (несправності) при діагностуванні — умовна ймовірність того, що непрацездатний об'єкт у результаті діагностування визнають працездатним.

Ймовірність хибної відмови (несправності) в даному елементі (групі) — ймовірність того, що за відсутності відмови в результаті діагностування приймається рішення про її наявність у даному елементі (групі).

Ймовірність хибної відмови (несправності) під час діагностування — ймовірність того, що працездатний об'єкт у результаті діагностування виявиться непрацездатним.

Кодова відстань між двома словами d — кількість розрядів, за якими одне слово відрізняється від іншого.

Комбінований технологічний процес діагностування — послідовний ланцюжок технологічних операцій з паралельними розгалуженнями і (або) замкнутими ланцюжками операцій.

Комплексне тестування — пошук невідповідності системи своїй вихідній (початковій) меті.

Коректність програми — відповідність її специфікації. Оскільки специфікація може не відповідати фактичним вимогам до програми, то трапляються випадки, коли некоректна програма працює надійно або, навпаки, коректна — ненадійно.

Кратність помилок t — кількість розрядів, спотворених помилкою.

Лінійний технологічний процес діагностування — послідовний ланцюжок операцій.

Логічна динамічна несправність — стан, який призводить до спотворення логічних функцій, що реалізують елементи структури в динамічних режимах роботи.

Міра довіри C — ймовірність того, що модель правильно відхилятиме хибне припущення.

Модернізація (франц. modernisation — оновлення) — заміна в комп'ютері застарілих компонентів на нові з вищими параметрами, якщо це дає змогу зробити конструкція комп'ютера.

Модуль (лат. modulus — міра) — елемент (компонент) програми, стандартизований за формою запису і зовнішніми зв'язками.

Надійність — властивість об'єкта зберігати в часі у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах та умовах застосування, технічного обслуговування, зберігання і транспортування.

Надійність програмного забезпечення — властивість програми виконувати задані функції в заданих умовах роботи і на заданій ЕОМ.

Невідновлювані об'єкти — об'єкти, для яких ремонт із конструктивних причин або умов експлуатації є неможливим.

Неготовність — стан об'єкта, в якому він нездатний виконувати потрібну функцію з будь-якої причини.

Нелогічна динамічна несправність — стан, який призводить до появи неправильних значень сигналів на виходах структури в динамічних режимах роботи і не пов'язаний зі спотворенням елементами структури логічних функцій, що ними реалізуються.

Несправність ВІС або НВІС — стан, зумовлений дефектами одного чи кількох елементів внутрішньої структури кристала або провідників, що з'єднують його з виводами цієї ВІС (НВІС). Враховуючи це, несправністю цифрового чи мікропроцесорного пристрою вважають стан, зумовлений дефектами одного чи кількох компонентів (елементів) їх структури або провідників, що з'єднують компоненти в цю структуру.

Несправність Ц або МПП — формалізоване представлення факту прояву дефекту цих пристроїв у вигляді неправильних значень сигналів на входах і виходах.

Пасивні профілактичні заходи — роботи, спрямовані на захист комп'ютера від зовнішніх шкідливих дій.

Первинна помилка — спотворення в тексті програми.

Періодичне обслуговування — обслуговування, яке виконують через задані інтервали часу щодо встановленого часового графіка.

Питомі витрати — відношення об'єму буферної пам'яті до швидкості передавання тест-векторів.

Плаваюче ядро — ядро, склад якого непостійний.

Повнота діагностування — характеристика, яка визначає можливість виявлення відмов (несправностей) в об'єкті з використанням обраного методу його діагностування (контролю).

Показник ефективності використання діагностичних засобів — кількісна характеристика ступеня досягнення корисних результатів при використанні системи в конкретній експлуатаційній ситуації з урахуванням експлуатаційних витрат.

Показник якості — кількісна характеристика однієї або кількох властивостей системи, які складають її якість, що розглядається стосовно певних умов створення і споживання.

Породжуюча матриця Q кода — матриця, рядки якої в сукупності утворюють базис групи, що відповідає множині всіх кодових слів.

Правильний результат роботи обчислювального пристрою — результат, коли пристрій справді працює правильно, відсутні сигнали помилок, або результат, коли пристрій справді працює неправильно, про що свідчить сигнал помилки.

Пропуск помилки засобами контролю — результат неправильної роботи пристрою і відсутності сигналу помилки. Якщо пристрій працює правильно, а засоби контролю сигналізують про помил-

ку, то це свідчить, що їх контролюють не повністю або зовсім не контролюють.

Профілактичне обслуговування — обслуговування, яке здійснюють для попередження відмов функційного модуля (системи).

Резервування (франц. reserve — запас) — метод підвищення надійності шляхом включення резервних елементів при розробленні системи або в процесі її експлуатації.

Ремонтопридатність — властивість об'єкта бути пристосованим до підтримання і відновлення стану, в якому він здатний виконувати належні йому функції за допомогою технічного обслуговування і ремонту.

Розподілене ядро — ядро, що може бути організоване на будь-якому комп'ютері комп'ютерної системи (КС), навіть на неперевіреній.

Середній час відновлення — середній час, потрібний для відновлення працездатності для відомого інтервалу часу використання функційного модуля (системи).

Середнє напрацювання між відмовами — відношення сумарного напрацювання відновлювального об'єкта до математичного сподівання кількості його відмов протягом цього напрацювання.

Середній час напрацювання на відмову — відношення сумарного часу напрацювання відновлювального об'єкта (пристрою) до математичного сподівання кількості його відмов протягом цього часу.

Система технічного діагностування — сукупність засобів об'єкта і виконавців, необхідна для проведення діагностування (контролю) за правилами, встановленими технічною документацією. Як правило, вона охоплює зовнішні засоби технічного діагностування.

Системний підхід — сумісний аналіз і розроблення всіх елементів системи.

Тестовий (англ. test — випробування) **контроль** — контроль, під час якого на об'єкт контролю подають тестові впливи.

Технічна діагностика — галузь знань, що досліджує стани ОД і розробляє методи їх розпізнавання, визначає принципи проектування та організації пристроїв і систем технічного діагностування.

Технічне діагностування — процес визначення технічного стану об'єкта з означеною (заданою) точністю.

Технічне обслуговування системи — будь-яка діяльність, призначена для збереження або відновлення працездатності функційного модуля (системи).

Технічний контроль — процес, що забезпечує виявлення несправностей у роботі цифрових пристроїв, комп'ютерів, комп'ютерних систем і мереж, викликаних відмовою або збоями апаратурних засобів, помилками в програмах, помилкою оператора тощо.

Технічний ресурс виробу — прогнозована тривалість його експлуатації від початку до моменту, коли інтенсивність відмов збільшується до рівня, який робить подальшу експлуатацію неможливою або недоцільною з економічних міркувань.

Технічний стан об'єкта — стан, який характеризують у певний момент часу за певних умов зовнішнього середовища значення параметрів, установлених технічною документацією на об'єкт.

Технологічна діагностична операція — частина технологічного процесу діагностування, яку здійснюють над одним чи кількома ОД, на одному робочому місці, одним чи кількома спеціалістами-експертами або спеціалістами-розробниками.

Тривалість діагностування — інтервал часу, необхідний для проведення діагностування об'єкта.

Функції ненадійності — криві, отримані в результаті експерименту або на основі оброблення статистичних даних про кількість відмов.

Функційний (лат. functio — виконання) **(робочий) контроль** — контроль, під час якого на об'єкт контролю подають робочі впливи.

Централізоване ядро — ядро, що організують на одному заздалегідь визначеному справному комп'ютері.

Циклічний код — кільце, тобто замкнута сукупність елементів, для яких задані дві операції та зворотні до них.

Циклічний технологічний процес діагностування — замкнута послідовність операцій ланцюжка.

Ядро — сукупність апаратних і програмних засобів, що забезпечують реалізацію тестового діагностування системи.

Якість експлуатації — сукупність властивостей процесу експлуатації системи, від яких залежить відповідність цього процесу і його результатів встановленим вимогам.

Якість системи — сукупність властивостей системи, що дає змогу задовольняти певні потреби відповідно до її призначення.

Скорочення і умовні позначення

AGP (Accelerated Graphic Port) — покращений графічний порт.

AT (Advanced Technology) — передова технологія; ПК з процесорами I80286 і вище.

BIOS (Basic Input-Output System) — базова система введення-виведення (програма).

BIST (Built in Self-Test) — програма самоперевірки процесора при вмиканні живлення.

BSDL (Boundary Scan Description Language) — мова опису пристроїв для діагностування за допомогою інтерфейсу JTAG.

CAM (Common Assess Method) — метод загального доступу.

COM Port (COMmunication Port) — послідовний порт.

CD-R (Compact Disc-Recordable) — компакт-диск з можливістю запису.

CD-ROM (Compact Disc-Read Only Memory) — постійна пам'ять на компакт-дисках тільки для зчитування.

CD-RW (Compact Disc-ReWritable) — компакт-диск з можливістю багаторазового перезапису.

CMOS (Complimentary Metal Oxide Semiconductor) — комплементарна структура метал-оксид-напівпровідник (КМОН).

CPU (Central Processor Unit) — центральний процесор.

DIMM (Dual In-Line Memory Module) — модуль пам'яті з двосторонніми друкованими виводами (ОЗП).

DIN (Deutsch Industrie Norm) — малогабаритний круглий багатоконтактний роз'єм.

D-LED™ (Dual-Light Emitting Diode) — світлодіодний індикатор для визначення несправностей комп'ютера, розташований на системній платі.

DMA (Direct Memory Access) — передавання даних у режимі прямого доступу до пам'яті.

ECP (Enhaced Capabilieties Port) — порт з розширеними можливостями.

EISA (Extended ISA) — розширена шина ISA.

EPP (Enhaced Parallel Port) — розширений паралельний порт.

IDE (Integrated Drive Electronics) — інтерфейс для дисків (жорстких) з інтегрованим у нього контролером.

IEEE (Institute of Electrical and Electronic Engineers) — Інститут інженерів з електротехніки і електроніки, що встановлює численні стандарти.

IRQ (Interrupt Request) — лінія запиту на переривання.

ISA (Industry Standart Architecture) — тип системної шини.

JTAG (Boundary Scan Architecture; стандарт IEEE 1149.1) — інтерфейс для тестування складних логічних схем (процесорів).

HDD (Hard Disc Drive) — накопичувач на жорстких дисках (вінчестер).

LPT (Line Printer) — порт комп'ютера для підключення принтера.

MIDI (Musical Instrument Digital Interface) — цифровий інтерфейс звукової плати (музичних інструментів).

MPEG (Monitor Picture Experts Group) — плата декодера.

MTBF (Mean Time between Failure) — середній час роботи елемента між несправностями.

MTTR (Mean Time to Repair) — середній час ремонту.

PCI (Periphrial Component Interconnect bus) — шина взаємодії периферійних пристроїв.

POST (Power-On Self Test) — самотестування комп'ютера при вмиканні живлення.

RAID (Redundant Array of Inexpensive Disks) — матриця недорогих дисків з надлишковістю.

RAM (Random Access Memory) — оперативна пам'ять з довільним доступом.

ROM (Read Only Memory) — постійний запам'ятовуючий пристрій (ПЗП).

RTC (Real Time Clock) — годинник реального часу.

SCSI (Small Computer System Interface) — системний інтерфейс малих комп'ютерів.

SD RAM (Synchronous DRAM) — надшвидкодіюча синхронна динамічна пам'ять.

SIMM (Single In-Line Memory Modyle) — модуль пам'яті з однорядним розташуванням штирків (ОЗП).

TCK (Test Clock) — сигнал синхронізації послідовних даних.

TDI (Test Data Input) — вхідні дані в послідовному двійковому коді.

TDO (Test Data Output) — вихідні дані в послідовному двійковому коді.

TMS (Test Mode Select) — сигнал вибору тестового режиму.

USB (Universal Serial Bus) — універсальна послідовна шина.

VGA (Video Graphics Array) — графічний адаптер.

ZIF (Zero Insertion Forse) — гніздо для установлення мікропроцесорної мікросхеми.

Локазюк В. М., Савченко Ю. Г.

Л 73 Надійність, контроль, діагностика і модернізація ПК: Посібник. — К.: Видавничий центр «Академія», 2004. — 376 с. (Альма-матер)

ISBN 966—580—168—6

У навчальному посібнику розглянуто різноманітні аспекти забезпечення надійності обчислювальних пристроїв, персональних комп'ютерів (ПК) і комп'ютерних систем, методи контролю цифрових пристроїв. Особливу увагу приділено методам і засобам діагностування ПК, програмного забезпечення, від умілого використання яких значною мірою залежить надійність їх функціювання. Також висвітлено питання модернізації та експлуатаційного обслуговування ПК.

Адресований студентам вищих технічних навчальних закладів, спеціалістам, які займаються експлуатацією комп'ютерних систем і мереж та використовують при вирішенні виробничих і управлінських завдань комп'ютерні технології.

ББК 32.965

Навчальне видання

Серія «Альма-матер»
Заснована в 1999 році

ЛОКАЗЮК Віктор Миколайович
САВЧЕНКО Юлій Григорович

Надійність, контроль, діагностика і модернізація ПК

Посібник

Спільний проект із видавництвом «Академнидав»

Редактор М. П. Кордюмова
Технічний редактор Т. І. Семченко
Коректор Т. П. Шайнюк
Комп'ютерна верстка В. П. Богуславця

Підписано до друку
з оригінал-макета 14.11.2003.
Формат 84x108/32. Папір офс. № 1.
Гарнітура Шкільна. Друк офсетний.
Ум.-друк. арк. 19,7. Ум. фарбовідб. 20,12.
Обл.-вид. арк. 22,5. Зам. 3–444.

Видавничий центр «Академія»
04119, м. Київ-119, а/с 37.
Тел./факс: (044) 213-19-24; 456-84-63.
E-mail: academia-pc@svitonline.com
Свідоцтво: серія ДК № 555 від 03.08.2001 р.

ВАТ «Поліграфкнига».
03057, м. Київ, вул. Довженка, 3.