О.М.РОЇК, Т.О.САВЧУК, О.М.ТКАЧЕНКО

P-CAD 2000

ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СКЛАДНИХ ОБ'ЄКТІВ І СИСТЕМ

ACCEL EDA 15.0

Dr.Spice
SPECCTRA
Cross Talk
Signal Integrity

Міністерство освіти і науки України
Вінницький державний технічний університет

О.М.РОЇК, Т.О.САВЧУК, О.М.ТКАЧЕНКО

ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СКЛАДНИХ ОБ'ЄКТІВ І СИСТЕМ

Затверджено Ученою радою Вінницького державного технічного університету як навчальний посібник для студентів спеціальності "Інтелектуальні системи прийняття рішень" денної та заочної форм навчання, протокол № 2 від 28 вересня 2000р.

Вінниця ВДТУ 2001

УДК 681.31
Р 65

Рецензенти:

А.М.Пєтух, доктор технічних наук
Л.В.Крупельницький, кандидат технічних наук
В.В.Мотигін, кандидат технічних наук

Рекомендовано до видання Ученою радою Вінницького державного технічного університету Міністерства освіти і науки України

О.М.Роїк, Т.О.Савчук, О.М.Ткаченко
Р 65 Основи автоматизованого проектування складних об'єктів і систем. Навчальний посібник. - Вінниця: ВДТУ, 2001. - 111 с.

В навчальному посібнику викладено основні підходи щодо автоматизації проектування складних об'єктів і систем, розглянуто теоретичні та практичні матеріали з дисципліни "Основи автоматизованого проектування складних об'єктів і систем", де основна увага приділяється проблемам автоматизації процесу аналізу та синтезу об'єктів і систем зазначеного класу, застосуванню теорії масового обслуговування, мереж Петрі, моделюванню пристроїв РЕА. В навчальному посібнику наведені приклади і рекомендації з їх засвоєння, приведені рекомендації по розширеному опануванню. Приведені контрольні запитання, тематика лабораторних робіт, що можуть бути використані при проведенні як практичних, так і лабораторних занять.

Навчальний посібник призначено для студентів спеціальності "Інтелектуальні системи прийняття рішень" денної та заочної форм навчання.

УДК 681.31

ЗМІСТ

ВСТУП

В процесі розвитку науки і техніки системи, пристрої й споруди, що створюються людиною, стають дедалі складнішими. Одночасно вимоги, які висуваються щодо термінів проектування нових виробів, стають більш жорсткими. За цих умов неавтоматизовані методи проектування стають неефективними. Створення та широке застосування систем автоматизованого проектування (САПР) стало вкрай необхідним. Для розв'язання цієї важливої задачі вимагається відповідна підготовка фахівців. Випускникам вузів необхідно вміти працювати як користувачами САПР, крім користувачів потрібні також фахівці з розробки самих САПР.

Епізодичне розв'язання окремих інженерних задач на ЕОМ почалось одразу з появою самих обчислювальних машин. Однак історія САПР починається з початку 70-х років, коли створення єдиних програмно-методичних комплексів для проектування ЕОМ та їх елементів визначило появу перших систем автоматизованого проектування. В середині 70-х років промисловість почала серійний випуск програмно-технічних комплексів САПР, які отримали назву автоматизованих робочих місць. На початку 80-х років сформувались концепції багаторівневих САПР, що здійснюють повний цикл проектування. Дев'яності роки збагатили теорію та практику проектування методами штучного інтелекту.

Не зважаючи на появу нових цікавих напрямків в галузі науки і техніки, автоматизоване проектування і зараз лишається невід'ємною складовою частиною науково-технічного прогресу. До того ж за роки існування САПР фахівцями було висунуто багато нових ідей, розроблено велику кількість сучасних методів та алгоритмів. Знайомство з ними, без сумніву, має підвищити рівень фахівців в галузі інформаційних технологій, збагатити інтелектуальну скарбничку сучасного інженера-програміста.

1 ЗАГАЛЬНІ ВІДОМОСТІ ПРО АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ

1.1 Характеристика об'єктів проектування

Під проектуванням технічних систем розуміють комплекс робіт з дослідження, розрахунків та конструювання, які мають за мету отримання технічної документації, необхідної для створення нових пристроїв або реалізації нових процесів, що задовольняють задані вимоги.

Згідно з ДЕСТ 22487-77, проектування – це процес складання опису, необхідного для створення об'єкта, що ще не існує, який здійснюється перетворенням первинного опису (технічного завдання), оптимізацією заданих характеристик об'єкта та алгоритму його функціонування, усуненням протиріч первинного опису та послідовним поданням деталізованих описів об'єкта на різних мовах для різних етапів проектування [17].

Зазначимо, що наведене визначення підходить і для розробки програмних засобів.

Дане визначення потребує відповідей що найменше на два питання.

1. Що є об'єктом проектування (ОП)?
2. Які вимоги висуваються до ОП?

При цьому, на системному рівні розуміється проектування складних систем, наприклад, систем технічного контролю, медичних діагностичних комплексів або цифрових телефонних станцій. Найбільш детально обговорюється проектування цифрових та гібридних радіоелектронних пристроїв (плат з розташованними на них елементами). Прикладами ОП можуть бути вироби чи системи (вузли ЕОМ, модеми, пульти управління) або процеси (технологічні, обчислювальні).

Відповідь на друге питання більш складна. Перед усім слід визначити, чим характеризуються електронні (аналого-цифрові) пристрої.

Всі характеристики електронних пристроїв можна віднести до однієї з чотирьох груп: точность, часу (швидкодії), надійності, вартості.

Точність характеризується похибкою. В залежності від розмірності значення похибки поділяють на абсолютні та відносні. Під *абсолютною похибкою* Δ розуміють різницю між реальним значенням вихідного сигналу Y та його можливим ідеальним значенням Y_0:

$$\Delta = Y - Y_0. \qquad (1.1)$$

Відносна похибка не має розмірності і визначається як частка від ділення абсолютної похибки на реальне значення вихідного сигналу:

$$\delta = \Delta/Y = (Y - Y_0)/Y. \qquad (1.2)$$

Іноді відносну похибку подають у відсотках. Загалом точність має особливо важливе значення для аналогових або аналого-цифрових пристроїв.

Часові характеристики визначають динамічні властивості пристрою, тобто швидкість реалізації ним необхідних функцій. Для аналого-цифрових пристроїв це:

1. *Тривалість функціонування* $T_ф$ - затримка між моментом подачі вхідного сигналу та появою вихідного сигналу.
2. *Період функціонування* $T_ц$ - інтервал часу між двома послідовними перетвореннями вхідного сигналу.

Наприклад, для модему швидкодія буде визначатися його часовою діаграмою, часом встановлення та поновлення зв'язку, швидкістю передачі, протоколами зв'язку, тощо.

Надійність визначає здатність пристрою виконувати покладені на нього функції, зберігаючи свої експлуатаційні характеристики в заданих

межах протягом певного проміжку часу. Кількісною характеристикою надійності є *ймовірність безвідмовної роботи*. На практиці звичайно застосовується середній час безвідмовної роботи, тобто середній час функціонування аналого-цифрового пристрою до появи першої відмови.

Вартість визначає в узагальненому вигляді всі витрати, пов'язані з практичною реалізацією пристрою. Сюди відносять собівартість пристрою, а також масу та габаритні розміри.

Всі наведені характеристики описують окремі аспекти функціонування систем або пристроїв. Тому робилися спроби отримати узагальнені характеристики [19, 20]. Проте на практиці такі критерії звичайно не застосовують, а задаються якоюсь однією характеристикою (як правило собівартістю С) та її оптимізують і при цьому дивляться, щоб інші характеристики знаходились в певних межах.

Проектування, що здіснюється людиною у взаємодії з ЕОМ, називають *автоматизованим*. Ступінь автоматизації може бути різною і оцінюється часткою δ проектних робіт, які виконує ЕОМ без участі людини. Якщо $\delta=0$ проектування називають ручним або *неавтоматизованим*, якщо $\delta=1$ – *автоматичним*. Найчастіше степінь автоматизації коливається в межах 0.5 – 0.7.

Автоматизоване проектування РЕА здійснюється в рамках САПР.

1.2 Етапи проектування

Процес проектування складних систем або виробів звичайно складається з таких етапів.

Етап науково-дослідних робіт (НДР) – етап попереднього проектування. Як правило, нова складна система певного призначення або не має аналогів, або повинна перевершити їх за тими чи іншими показниками. І в першому, і в другому випадку для досягнення поставленої мети необхідні

наукові дослідження, пов'язані з пошуком принципових можливостей створення системи, нових підходів до розв'язання проблем, побудови структур та технічних засобів, тощо. Результатом виконання цього етапу є технічна пропозиція.

Етап ескізного проектування або **дослідно-конструкторських робіт (ДКР)** – етап створення ескізного проекту, в якому відображені результати детальної проробки можливостей побудови системи.

Етап технічного (робочого) проектування – етап детальної проробки всіх схемних, конструкторських та технологічних рішень, що фіксуються в технічному проекті системи.

Крім названих етапів при проектуванні виробів, що призначені для серійного випуску, виділяють також **етапи виготовлення та випробувань серійного зразка** або пробної серії та корегування проекту за результатами випробувань.

Основне призначення САПР – розв'язання задач ескізного та технічного проектування.

На будь-якій стадії або етапі проектування можуть виявитися помилки або неоптимальність прийнятих рішень, і, таким чином, необхідність або доцільність їх перегляду. Такі ситуації є характерними для проектування та зумовлюють його ітераційний характер. Може бути також виявлена необхідність корегування технічного завдання. В цьому випадку відбувається чергування процедур зовнішнього та внутрішнього проектування, що особливо характерно для ранніх стадій (НДР, ДКР). При цьому до зовнішніх процедур проектування відносять процедури формування або корегування технічного завдання, а до внутрішніх – процедури реалізації сформованого завдання.

В залежності від порядку, в якому виконуються етапи проектування, розрізняють висхідне та низхідне проектування. Висхідне проектування (проектування знизу догори) характеризується розв'язанням задач більш

низьких ієрархічних рівнів перед розв'язанням задач більш високих рівнів. Протилежна стратегія (просування згори донизу) приводить до низхідного проектування. При проектуванні електронної апаратури частіше застосовують низхідне проектування. В той же час при створенні програмних засобів, особливо із застосуванням об'єктно-орієнтованого підходу, все більше використовують висхідне проектування. Стадія конструкторського проектування також частіше виявляється висхідною.

1.3 Рівні проектування

Практика проектування електронної та обчислювальної техніки привела до виділення горизонтальних та вертикальних рівнів проектування, що наведені в таблиці 1.1 [13].

Горизонтальні рівні виділяють згідно з врахуванням характеру властивостей об'єкта. Основними горизонтальними рівнями є: функціональний, алгоритмічний та конструкторський.

Функціональне проектування пов'язано з розробкою структурних, функціональних та принципових схем.

Алгоритмічне проектування пов'язано з розробкою алгоритмів функціонування електронних пристроїв та з створенням математичного (в тому числі програмного) забезпечення.

Конструкторське проектування охоплює коло питань, пов'язаних з конструкторською реалізацією результатів функціонального проектування, тобто питань вибору форми та матеріалів оригінальних деталей, вибір типорозміру уніфікованих деталей, розташування складових частин, тощо.

Рівні проектування можна також виділяти згідно зі степенем деталізації подробиць, з якими відображаються властивості об'єкта, що проектується (так звані вертикальні, або ієрархічні рівні).

Таблиця 1.1 - **Горизонтальні рівні**

Функціональний	**Алгоритмічний**	**Конструкторський**
-	Проектування схем алгоритмів	-
Системний (структурний)	Архітектурний (проектування модулів та їх взаємодії)	Стійка, панель
Функціонально-логічний	Програмний, мікропрограмний	Модуль, плата
Схемотехнічний (компонентний)	-	Кристал ІС (програма для ПЛІС)

На цьому рівні ведеться узагальнений розгляд об'єкта в цілому, розподіл його на окремі частини.

На функціонально-логічному рівні проектуються функціональні та принципові схеми пристроїв. Звичайно при цьому виконується також розробка тестів для перевірки правильності роботи пристрою.

На схемотехнічному рівні проектуються принципові схеми ІС або фрагментів БІС. Зараз, коли широко застосовуються програмовані логічні інтегральні схеми (ПЛІС) сюди можна включити і створення ІС оригінальної структури на спеціалізованій мові.

На компонентному рівні розробляються окремі компоненти ІС з точністю до електронних дискретних елементів – резисторів, транзисторів, конденсаторів.

Алгоритмічне проектування починається з розробки схем алгоритмів функціонування системи в цілому. Тут бажано провести імітаційні експе-

рименти з метою виявлення можливих неприємних ситуацій при граничному навантаженні системи.

На архітектурному рівні програмне забезпечення розбивають на окремі модулі, обговорюють специфікації на них та стикування модулів між собою. Безпосередня розробка програмного забезпечення виконується на наступному рівні.

Слід зазначити, що наведена схема розробки складних програмних комплексів є дещо узагальненою. Деталізується та уточнюється вона в рамках методології об'єктно-орієнтованого проектування.

Наступне зауваження стосується розробки апаратно-програмних пристроїв (наприклад, модемів), які реалізуються на однокристальних ЕОМ або цифрових сигнальних процесорах. Для них мікропрограми створюються на спеціалізованих мовах програмування, а сам процес проектування лежить на стику між розробкою апаратних та програмних засобів. Цей сектор ринку в останні роки розвивається дуже динамічно і тому заслуговує пильного вивчення з боку як наукових робітників, так і майбутніх фахівців.

1.4 Склад САПР

Компонентами САПР є різні види забезпечення – технічне, математичне, програмне, лінгвістичне, інформаційне, методичне та організаційне [12].

Технічне забезпечення – сукупність технічних (апаратних) засобів, що використовуються в САПР для переробки, зберігання, передачі інформації, організації спілкування людини з ЕОМ, виготовлення проектної документації. Основу технічного забезпечення складають ЕОМ з різноманітними периферійними пристроями. До технічного забезпечення САПР відносять також засоби оргтехніки, вимірювальне обладнання для отримання даних, які використовують в процесі проектування.

Математичне забезпечення - сукупність математичних моделей, методів, алгоритмів для розв'язання задач автоматизованого проектування. Математичне забезпечення реалізується в програмному забезпеченні САПР.

Програмне забезпечення – сукупність програм разом з необхідною документацією, яка призначена для використання в САПР.

Лінгвістичне забезпечення – сукупність мов, що використовують в САПР для відображення інформації про об'єкти, які проектуються, процес та засоби проектування.

Інформаційне забезпечення – документи, що містять описи стандартних проектних процедур, типових проектних рішень, комплектуючих та інші дані, а також файли на магнітних носіях з електронними версіями зазначених документів.

Методичне забезпечення – документи, в яких відображені склад, правила відбору та експлуатації засобів автоматизованого проектування. Іноді поняття методичного забезпечення розширяють, включаючи в нього лінгвістичне та математичне забезпечення.

Організаційне забезпечення – положення, інструкції, накази, штатні розклади, кваліфікаційні вимоги та інші документи, які регламентують організаційну структуру підрозділів проектного підприємства та їх взаємодію з комплексом засобів автоматизованого проектування.

Сучасні САПР створюються за такими принципами:

- САПР – система, в якій взаємодіють людина та машина. Колектив розробників є складовою частиною системи проектування, виконуючи проектні роботи в діалозі з ЕОМ.
- Комплексна автоматизація всіх рівнів проектування. Введення автоматизованого проектування на деяких рівнях, за умов збереження застарілих ручних форм проектування на інших рівнях, менш ефективне, ніж комплексна автоматизація всього процесу.

- Інформаційна узгодженість підсистем та програм проектування. Програми, які входять в пакет прикладних програм деякої підсистеми САПР, повинні бути інформаційно узгодженими, тобто допускати можливість спільного виконання заданої проектної процедури без втручання людини в процес переходу від однієї програми до іншої.
- Відкритість САПР. Властивість відкритості означає можливість внесення змін в систему під час її експлуатації.
- Поєднання традиційного та автоматизованого проектування. Цей принцип має значення в тих випадках, коли автоматизоване проектування впроваджується вже на діючому підприємстві з установленою структурою, певними взаємовідносинами між підрозділами, тощо.

Контрольні запитання

1. Що розуміють під об'єктом проектування?
2. Поясніть різницю між ручним, автоматизованим та автоматичним проектуванням.
3. Поясніть різницю між рівнями та етапами проектування.
4. Дайте характеристику горизонтальним та вертикальним рівням проектування.
5. Наведіть приклад ієрархічної структури технічного пристрою.
6. Виконайте порівняльний аналіз висхідного й низхідного проектування.
7. Які види запезпечення входять в склад САПР?
8. Поясніть, в чому полягають основні принципи створення сучасних САПР.

2 АВТОМАТИЗАЦІЯ СИСТЕМНОГО ПРОЕКТУВАННЯ

2.1 Основні задачі системного проектування

Задачі системного проектування складних систем поділяються на задачі аналізу та синтезу. Як правило, на цій стадії доцільно розглядати процедури проектування апаратних та програмних засобів спільно, оскільки головна мета даного етапу – спроектувати систему в цілому так, щоб вона задовольняла поставлені вимоги, не вдаючись до розгляду деталей.

Проектні процедури синтезу майже не автоматизовані, їх автоматичне розв'язання на ЕОМ вдається отримати лише в окремих випадках. Процедури аналізу, як правило, автоматизовані. Однак отримання математичних моделей на системному рівні вимагає від користувача значно більших зусиль та витрат часу, ніж на інших ієрархічних рівнях. Тому процес проектування переважно відбувається в діалоговому режимі: користувач пропонує можливі варіанти структур, а ЕОМ їх оцінює (виконує верифікацію). Генерацію варіантів можна частково перекласти на ЕОМ на підставі одного з підходів до структурного синтезу [10, 11]. На підставі отриманих результатів користувач вносить зміни у варіанти, приймає рішення, коригує математичні моделі, програмні засоби, тощо.

2.2 Формалізація процедур синтезу

Процедури системного синтезу класифікуються за рядом ознак [12].

За цілями синтезу та змістом результатів, що отримуються виділяють такі процедури структурного синтезу:

- вибір принципів побудови та функціонування технічних об'єктів;
- вибір технічного рішення;
- синтез технічної документації.

Вибір принципів побудови та функціонування технічних об'єктів здійснюється на стадіях передпроектних досліджень та науково-дослідних

робіт. Його мета – встановлення фізичних, інформаційних, організаційних принципів, тощо. При проектуванні обчислювальних систем змістом цієї процедури є вибір архітектурних рішень та побудова структурних схем.

Вибір технічного рішення виконується переважно на стадіях дослідно-конструкторських робіт в рамках раніше встановлених принципів функціонування і має за мету отримання функціональних, принципових, кінематичних схем, конструктивних рішень, технологічних маршрутів виготовлення деталей, тощо.

Синтез технічної документації відноситься до стадій технічного та робочого проектування і полягає в автоматичному перетворенні даних про схеми та конструкції, що знаходяться у внутрішньому форматі САПР, в текстову та креслярську документацію, оформлену згідно з правилами Єдиної системи конструкторської документації.

За рівнем складності формалізації процедур синтезу виділяють п'ять рівнів.

Задачі *першого рівня* складності з'являються там, де структура об'єкта, що проектується, визначена наперед результатами раніше виконаних досліджень і синтез зводиться до вибору числових значень параметрів для заданої структури. Задачі першого рівня – це задачі параметричного синтезу.

Задачі *другого рівня* складності полягають у виборі структури з кінцевої множини варіантів за таких умов:

1) всі варіанти заздалегідь відомі або їх можна легко отримати;
2) потужність множини варіантів настільки мала, що можливий повний перебір при їх порівнянні між собою.

Задачі *третього рівня* складності також зводяться до вибору варіантів в кінцевій множині, але потужність множини достатньо велика, щоб зробити неможливим повний перебір.

Задачі *четвертого рівня* складності характеризуються вибором варіанта структури в множинах, потужність яких апріорно невідома, та не виключена можливість, що вона необмежена.

Задачі *п'ятого рівня* складності пов'язані з пошуком рішень, основаних на нових, раніше невідомих або на таких ідеях та принципах, що не використовувалися. В задачах попередніх рівнів існування рішень не викликало сумнівів та вимагалось знайти найкраще або прийнятне рішення. В задачах п'ятого рівня складності отримання рішення рівноцінне отриманню принципово нового типу технічних об'єктів.

За типом структур, що синтезуються, розрізняють процедури одновимірного, схемного та геометричного синтезу.

Одновимірний синтез полягає в побудові одновимірних послідовностей з елементів деякої (переважно однакової) природи. Приклади таких послідовностей – опис технологічних процесів в формі маршрутної або операційної технології, обчислювальних процесів – у вигляді алгоритмів та програм для ЕОМ.

Схемний синтез пов'язаний з розробкою різних схем – функціональних, структурних, кінематичних, принципових, тощо, які відображують результати проектування об'єктів до конкретизації їх геометричних форм.

Геометричний синтез виконується при конструюванні виробів та пов'язаний з визначенням їх геометричних форм (синтез форми) і з розташуванням об'єкта або його окремих частин в просторі відносно заданих орієнтирів (задачі позиціювання).

На системному рівні найбільш розповсюджені два підходи до структурного синтезу:

1) зведення задачі синтезу до задачі дискретного математичного програмування та її розв'язання методами скороченого перебору;
2) використання експертних систем, які містять знання про відомі структури та елементи систем і способи генерації нових структур.

Багато задач формування оптимальної структури та вибору складу обладнання зводяться до задачі дискретного математичного програмування:

$$\min \mathbf{F}(\mathbf{X}),\ \mathbf{X} \in \mathbf{X}_D. \qquad (2.1)$$

де $\mathbf{X} = (x_1, x_2, \ldots, x_n)$, x_i – кількість одиниць обладнання і-го типу в системі, що проектується;

$\mathbf{X}_D$ – область допустимих значень, яка визначається обмеженнями виду

$$y_j\,(\mathbf{X}) \leq y_{Tj}, \qquad (2.2)$$

де $\mathbf{Y} = (y_1(X), y_2(X), \ldots, y_m(X))$, $y_j(X)$ – j-й вихідний параметр системи (середній час розв'язання задачі, пропускна здатність, ймовірність відмови, тощо); $\mathbf{Y}_T = (y_{T1}, y_{T2}, \ldots, y_{Tm})$ – вектор заданих технічних вимог; $F(\mathbf{X})$ – цільова функція, яка формується згідно з одним із способів постановки екстремальних задач [12].

Задача (2.1) є комбінаторною, оскільки на кожен x_i накладено певні обмеження. В загальному випадку вона відноситься до класу NP-повних задач, а це означає, що з ростом розмірності x_i кількість варіантів зростає експоненціально. Ефективні наближені методи розв'язання вдається отримати лише для окремих випадків загального формулювання.

В експертних системах знання про структуру систем та мереж частіше всього подаються у вигляді І-АБО графів як різновиду семантичних мереж [16].

Наприклад, вершина І відображує комп'ютерну мережу. Вона може мати радіальну, кільцеву або розподілену структуру. Оскільки тут існує багато альтернатив, це будуть вершини типу АБО і так далі.

Розробка семантичної мережі у вигляді графа І-АБО не викликає принципових труднощів. Складніше формалізувати правила вибору опти-

мального маршруту в графі згідно з вимогами технічного завдання. Розробка його здійснюється згідно з системними правилами (продукціями) і складає сутність структурного синтезу з використанням експертних систем.

2.3 Формалізація процедур аналізу

На системному рівні функціонування систем розглядається з інформаційної точки зору при повному абстрагуванні від фізичної суті процесів, які відбуваються в системі. В САПР за допомогою математичних методів моделюються процеси введення, обробки та виведення даних. При цьому застосовуються як аналітичні, так і імітаційні моделі. Можливості отримання аналітичних моделей для складних систем обмежені, тому аналіз найчастіше виконується методом імітаційного моделювання (як найбільш універсальним).

Слід зазначити, що системний аналіз неможливо виконати на прикладі окремої задачі. Необхідно отримати описи, узагальнені на весь клас задач. Узагальнення відбувається завдяки врахуванню статистичних закономірностей приходу та обробки даних. Тому на системному рівні найчастіше для аналізу використовують моделі систем масового обслуговування (СМО) та апарат мереж Петрі.

2.3.1 Системи масового обслуговування

Теорія масового обслуговування – розділ прикладної математики, який вивчає явища та процеси, пов'язані з задоволенням якогось попиту. СМО складається з статичних об'єктів, які називають обслуговувальними апаратами (ОА), та динамічних об'єктів, що носять назву заявок або транзактів [4].

Звичайні приклади СМО – телефонні станції, де роль ОА виконує комутаційне обладнання, а заявками є телефонні виклики; виробничі лінії (ОА – устаткування, заявки – деталі, які приходять на обробку).

Заявки, що надходять на обслуговування можуть утворювати черги (в протилежному випадку СМО називають недоступними). За структурою СМО можуть бути одно- чи багатоканальними (рис.2.1), одно- чи багатофазними (рис.2.2).

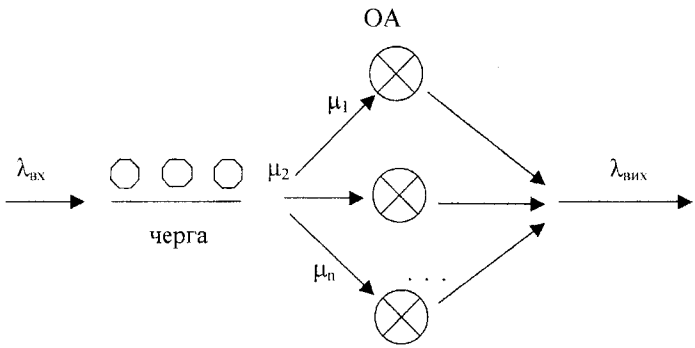


Рисунок 2.1 - Багатоканальна СМО

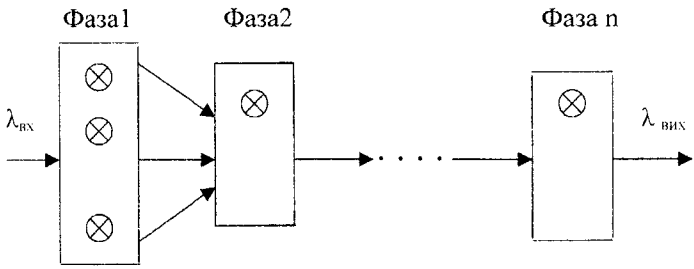


Рисунок 2.2 - Багатофазна СМО

Під $\lambda_{вх}$ розуміють інтенсивність вхідного потоку, тобто кількість заявок, що приходять на СМО за одиницю часу.

Під μ_i розуміють інтенсивність обслуговування заявки і-м автоматом, тобто кількість заявок, які можна обслужити за одиницю часу.

Типовим прикладом багатоканальної однофазної СМО може служити обслуговування викликів телефонними станціями, одноканальної багатофазної СМО – конвейєрна лінія з обробки деталей.

Функціонування СМО виглядає як процес проходження заявок через систему. Коли заявка надходить для обслуговування на деякий ОА, змінна u, що характеризує стан заявки, приймає значення "обслуговування" (1), а змінна w, що характеризує стан ОА – значення "зайнято" (1). Якщо ОА не зайнятий обслуговуванням, то змінна w має значення "вільний" (0). Якщо заявка надходить на вхід ОА, зайнятого обслуговуванням іншої заявки, то утворюється черга на вході ОА. Змінна u, що характеризує стан заявки, яка знаходиться у черзі, має значення "очікування" (0). Таким чином змінні u, w приймають одне з двох можливих значень, тобто є булевими змінними. Стан ОА характеризується також довжиною черги r на вході, яка дорівнює кількості заявок.

Правило, згідно з яким заявки надходять на обслуговування з черги, називається дисципліною обслуговування. Величина, яка виражає переважне право на обслуговування, називається пріоритетом.

Якщо всі заявки мають рівні пріоритети, дисципліна обслуговування називається безпріоритетною. Відомі правила FIFO та LIFO відносяться до безпріоритетних дисциплін обслуговування.

Потоки заявок. Прихід заявок на обслуговування може відбуватись через рівні, нерівні, але визначені, та випадкові проміжки часу. Вхідний потік вимог також може бути однорідним та неоднорідним. Однорідний потік утворюється заявками, на обслуговування яких в середньому витра-

чається однаковий час. Відповідно неоднорідний потік утворюється вимогами, середній час обслуговування яких істотно відрізняється.

Потік заявок можна розглядати як потік певних подій φ(t) (рис.2.3):

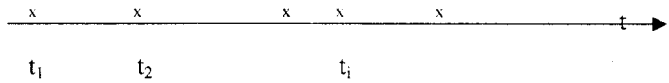


Рисунок 2.3 - Потік заявок

Потік заявок може описуватись різними законами розподілу: рівномірним, нормальним, пуасонівським, ерлангівським.

Простіший потік характеризується трьома властивостями:

- стаціонарність;
- ординарність;
- відсутність післядії.

Вхідний потік заявок називається стаціонарним, якщо ймовірність $p_n(\tau)$ того, що за час τ відбудеться n подій, не залежить від розташування τ на осі часу, а залежить лише від довжини τ та інтенсивності вхідного потоку λ (рис.2.4).

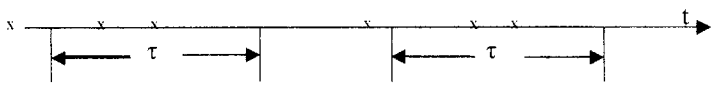


Рисунок.2.4 - Стаціонарний потік заявок

Ординарність означає, що ймовірність появи в один і той самий момент часу двох і більше заявок є незкінченно малою величиною:

$$\lim_{\Delta t \to 0} p_{>1}(\Delta t)=0 \qquad (2.3)$$

Відсутність післядії полягає в тому, що ймовірність приходу за відрізок часу τ вимог у кількості n не залежить від того, скільки вимог вже надійшло в систему.

За цих умов потік заявок описується законом Пуасона:

$$p_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}, \qquad \lambda = const. \tag{2.4}$$

Аналітичні моделі СМО можливо отримати тільки для систем, що мають вказані властивості. Оскільки більшість систем на практиці не підпадають під визначення "простіших", для їх аналізу використовують методи імітаційного моделювання.

2.3.2 Мережі Петрі

Мережею Петрі є четвірка

$$C = (P, T, I, O), \tag{2.5}$$

де $P = \{p_1, p_2, \ldots, p_n\}$ – кінцева множина позицій;

$T = \{t_1, t_2, \ldots, t_n\}$ – кінцева множина переходів;

$I: T \rightarrow P^{\infty}$ є вхідною функцією відображення з переходів в комплекти позицій;

$O: T \rightarrow P^{\infty}$ є вихідною функцією відображення з переходів в комплекти позицій;

p_i є вхідною позицією переходу t_j, якщо $p_i \in I(t_j)$;

p_i є вихідною позицією переходу t_j, якщо $p_i \in O(t_j)$ [15].

Входи і виходи переходів являють собою комплекти позицій. Комплект є узагальнення множини, в яке включаються елементи, що можуть багаторазово повторюватись. Використання комплектів, а не множин для

входів і виходів переходу дозволяє позиції бути кратним входом або кратним виходом переходу.

Теоретична робота по мережам Петрі базується на формальному визначенні. Але для ілюстрації понять більш зручним є графічне подання мережі Петрі.

Приклад графічного подання мережі Петрі наведений на рис.2.5.

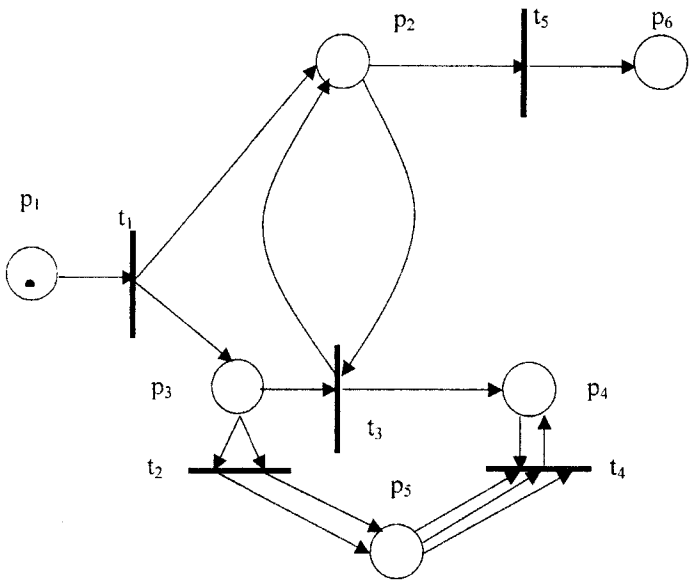


Рисунок 2.5 - Приклад мережі Петрі

Аналітичне подання мережі Петрі, яке зображене графічно на рис.2.5, виглядає так:

$C = (P, T, I, O);$

$P = \{p_1, p_2, p_3, p_4, p_5, p_6\};$

$T = \{t_1, t_2, t_3, t_4, t_5\};$

$I(t_1) = \{p_1\};$

$I(t_2) = \{p_3\}$;

$I(t_3) = \{p_2, p_3\}$;

$I(t_4) = \{p_4, p_5, p_5\}$;

$I(t_5) = \{p_2\}$;

$O(t_1) = \{p_2, p_3\}$;

$O(t_2) = \{p_3, p_5, p_5\}$;

$O(t_3) = \{p_2, p_4\}$;

$O(t_4) = \{p_4\}$;

$O(t_5) = \{p_6\}$.

Як видно з рис.2.5, теоретико-графовим поданням мережі Петрі є дводольний орієнтований мультиграф.

Структура мережі Петрі - це сукупність позицій та переходів. Відповідно до цього граф мережі Петрі має два типи вузлів. Коло є позицією, а планка – переходом. Орієнтовані дуги (стрілки) з'єднують позиції та переходи, при цьому деякі дуги спрямовані від позицій до переходів, а інші – від переходів до позицій.

Мережа Петрі є мультиграфом, тому що допускається існування кратних дуг від однієї вершини до іншої. Оскільки дуги є спрямованими, то це орієнтований мультиграф. Крім того, вершини графа можна поділити на дві множини (P і T), таким чином, що кожна дуга буде спрямована від елемента однієї множини до елемента іншої множини; такий граф є дводольним орієнтованим мультиграфом.

За допомогою мереж Петрі моделюються процеси, що подаються у вигляді послідовності подій. Прийнято вважати, що події відбуваються миттєво і в різні моменти часу. Кожній з можливих подій відповідає перехід. Подія відбувається, якщо виконані деякі умови. Кожній умові в мережі Петрі відповідає певна позиція. Виконання умов відображується за допомогою маркерів (фішок), які розташовують у вигляді крапок всередині відповідних позицій. Кількість станів мережі Петрі визначається кількістю

можливих маркувань. Маркуванням називається розподіл фішок по позиціях. Так, в мережі Петрі, що приведена на рис.2.5, маркування визначиться так: $\mu = \{1, 0, 0, 0, 0, 0\}$.

Моделювання процесів за допомогою мережі Петрі відбувається шляхом пересування маркерів (фішок) між позиціями. Пересування відбувається через запуск переходів. Перехід запускається видаленням фішок з його вхідних позицій і утворенням нових фішок у вихідних позиціях даного переходу.

Перехід може запускатися, якщо він є *дозволеним*. Перехід називається дозволеним, якщо в кожній його вхідній позиції кількість фішок принаймні дорівнює кількості дуг з позицій в перехід. Кратні фішки необхідні для кратних вхідних дуг.

Просте подання системи мережею Петрі базується на двох основних поняттях: подіях та умовах. Події – це дії, які мають місце в системі. Виникненням подій управляє стан системи. Стан системи може бути описаний множиною умов. Умова – це предикат або логічний опис стану системи. Умова може приймати значення "істина", або значення "хибно".

Природно, що події в системі можуть відбуватись. Для того, щоб подія відбулась, необхідно виконання відповідних умов. Ці умови називають передумовами події. Виникнення події може викликати порушення передумов та привести до виникнення інших умов, *післяумов*.

Як приклад розглянемо задачу моделювання роботи станка-автомата. Станок-автомат знаходиться в стані очікування до тих пір, поки не з'явиться деталь, яку він має обробити та послати на доставку. Умовами такої системи є:

а) станок-автомат очікує;

б) деталь прийшла та очікує;

в) станок-автомат обробляє деталь;

г) деталь оброблена.

Подіями будуть:

1. Деталь надійшла.
2. Станок-автомат починає обробляти деталь.
3. Станок-автомат закінчує обробляти деталь.
4. Деталь відсилається на доставку.

Може виникнути запитання: чи не доцільно об'єднати події 2 та 3 в одну подію: оброблення деталі. Таке рішення є логічним, проте, входить в протиріччя з одним з головних принципів моделювання мережами Петрі – події мають відбуватись миттєво. Щоб обійти це обмеження, процес оброблення (поштучно) виглядає як дві події – початок та кінець оброблення, між якими знаходиться умова: деталь обробляється.

Мережа Петрі на рис.2.6 ілюструє модель наведеного вище станка-автомата.

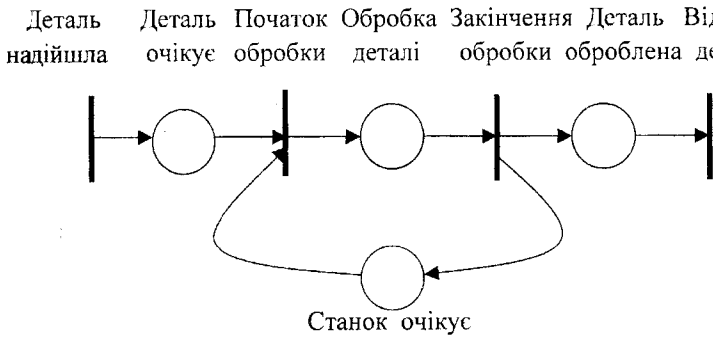


Рисунок 2.6 - Мережа Петрі для станка-автомата

Як бачимо, моделювання процесів за допомогою мереж Петрі виконується порівняно просто. Проте саме моделювання малокорисне. Необхідно провести аналіз системи, що проектується. Цей аналіз, можна сподіватись, приведе до більш глибокого розуміння функціонування системи. Після

проведення такого аналізу досліджені властивості мереж Петрі можуть бути перенесені на об'єкт фізичного світу.

Таким чином, ми підійшли до необхідності розгляду методів аналізу мереж Петрі. Але перед цим треба вияснити, які властивості мереж Петрі є найбільш важливими.

Обмеженість означає, що кількість фішок в будь-якій позиції не може перевищувати деякого числа K (K - обмеженість).

Безпека визначається умовою K=1, тобто кількість фішок в безпечній мережі Петрі не перевищує одиниці.

Досяженість передбачає можливість досягнення певних маркувань. Аналіз досяженості дозволяє визначити множину маркувань, в які можливі переходи в процесі функціонування мережі. Слід зазначити, що до аналізу досяженості певних маркувань зводиться багато задач, зокрема проектування пристроїв для виконання певних функцій.

Збережність – це властивість мережі зберігати (не збільшувати і не зменшувати) кількість фішок в усіх позиціях разом. Для строго збережної мережі Петрі

$$\Sigma\ \mu\ (p_i) = \Sigma\ \mu_0\ (p_i), \qquad (2.6)$$

де μ_0 – початкове маркування.

Збережність важлива для моделювання збереження ресурсів.

Живучість (активність) означає, що з будь-якого стану, який може бути досяженим з початкового, можливий перехід в будь-який інший стан. Аналіз живучості дозволяє виявити наявність тупиків, зациклювань, блокувань, тощо, наприклад, в процесі передачі повідомлень в обчислювальній мережі або при виконанні паралельних програм в багатопроцесорній системі.

Існують два основних методи аналізу вказаних властивостей мереж Петрі. Перший з них носить назву *дерева досяжності*, другий пов'язаний з матричними рівняннями. Ми розглянемо лише перший з них. Детальний опис матричного підходу можна знайти в [15].

Дерево досяжності являє собою множину досяжності мережі Петрі. Розглянемо марковану мережу Петрі, що подана на рис.2.7. Її початкове маркування – (1, 0, 0). В цьому початковому маркуванні дозволені два переходи: t_1 і t_2. Оскільки ми хочемо отримати всі множини досяжності, визначимо нові вершини в дереві, які утворяться в результаті запуска кожного переходу. Дуга, помічена переходом, що запускається, приводить з початкового маркування до кожного з нових маркувань (рис.2.8).

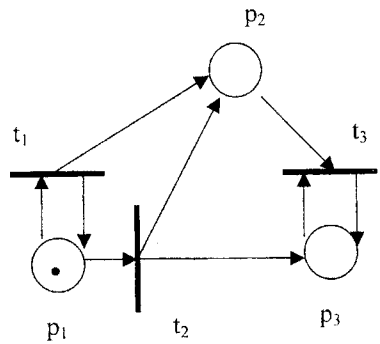


Рисунок 2.7 - Маркована мережа Петрі

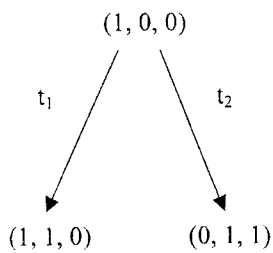


Рисунок 2.8 - Перший крок побудови дерева досяжності

Тепер необхідно розглянути всі маркування, досяжні з нових маркувань. З маркування (1, 1, 0) можна знову запустити t_1, отримуючи (1, 2, 0) та знову запустити t_2, отримуючи (0, 2, 1); з (0, 1, 1) можна запустити перехід t_3, отримуючи (0, 0, 1). Починаючи з трьох нових маркувань, необхідно повторити процес, утворюючи нові маркування, які потрібно ввести в дерево, як показано на рис.2.9.

Відмітимо, що маркування (0, 0, 1) пасивне; ніякий перехід в ньому не є дозволеним, тому ніякі нові маркування з цього пасивного маркування в дереві досяжності утворюватись не будуть. Крім того, необхідно відзначити, що маркування (0, 1, 1), яке утворюється після запуску t_3, вже було утворене безпосередньо з початкового маркування запуском переходу t_2.

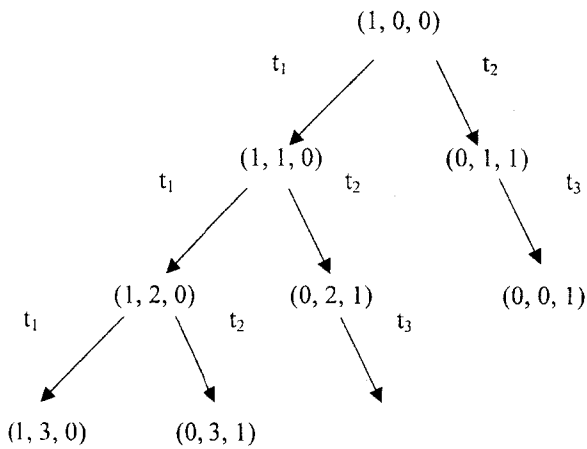


Рисунок 2.9 - Третій крок побудови дерева досяжності

Якщо наведену процедуру повторювати багато разів, кожне досяжне маркування буде утворене. Проте отримане дерево досяжності може виявитися нескінченним. Для того, щоб дерево стало корисним інструментом аналізу, необхідно знайти спосіб обмеження його до кінцевого розміру. Для цього треба обмежити введення нових маркувань (які називають гра-

ничними вершинами) на кожному кроці. Тут можуть допомогти пасивні маркування, тобто такі, в яких немає дозволених переходів. Ці пасивні маркування називають термінальними вершинами. Інший клас маркувань – це маркування, які раніше зустрічалися в дереві. Такі дублювальні маркування називають дублювальними вершинами; ніякі інші маркування розглядати немає потреби – всі вони будуть утворені з місця першої появи дублювального маркування.

Нарешті, можлива ситуація, коли при повторенні деякої послідовності запусків переходів σ в позиції p_i може утворитись довільно велика кількість фішок (наприклад, з кожним запуском t_1 в позиції p_2 буде на одну фішку більше). Для позначення нескінченної кількості фішок в дерево досяжності вводиться спеціальний символ ω, який означає "нескінченність".

Тепер можна сформулювати алгоритм побудови дерева досяжності. Кожна вершина *i* дерева пов'язується з розширеним маркуванням μ[i]; в розширеному маркуванні кількість фішок в позиції може бути або невід'ємним цілим числом, або ω. Кожна вершина класифікується або як гранична вершина, термінальна вершина, дублювальна вершина, або як внутрішня вершина (тобто така, від якої проведені всі дуги в дереві досяжності). Граничними є вершини, які ще не оброблені алгоритмом; алгоритм перетворить їх в термінальні, дублювальні або внутрішні. Кінцевий вигляд дерева наведений на рис.2.10.

Дерево досяжності можна використовувати для розв'язання таких задач, як безпека, обмеженість та збереженість. На жаль, в загальному випадку його неможливо використати для розв'язання задач досяжності та активності, а також для визначення можливої послідовності запусків. Розв'язання цих задач обмежено існуванням символу ω. Символ ω означає втрату інформації: конкретні числа фішок відкидаються, враховується лише існування їх великої кількості. Разом з тим в окремих випадках за допомогою дерева досяжності можливе розв'язання і задач досяжності та ак-

тивності. Так, мережа, дерево досяжності якої містить термінальну вершину, не активна (оскільки деяке досяжне маркування не має наступних маркувань). Аналогічно, якщо деяке маркування μ зустрічається в дереві досяжності, то воно є досяжне.

Теорія класичних мереж Петрі динамічно розвивається. В той же час з'являються нові різновиди мереж Петрі: часові (в яких перехід спрацьовує не миттєво, а протягом певного проміжку часу), стохастичні (з ймовірностями запуску переходів), кольорові (спроба ввести семантику в мережі Петрі), інгібіторні (з забороняючими гілками), пріоритетні (з пріоритетами дуг).

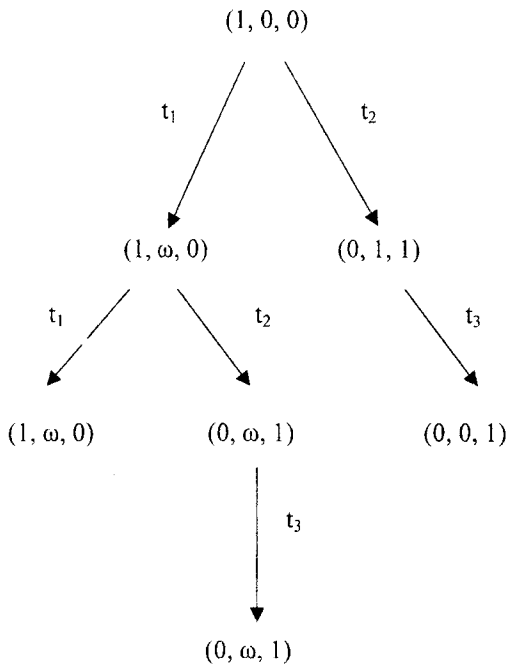


Рисунок 2.10 - Дерево досяжності для мережі Петрі

Контрольні запитання

1. Чому формальні методи синтезу не знайшли широкого застосування на практиці?
2. За якими ознаками здійснюється класифікація процедур структурного синтезу?
3. Охарактеризувати п'ять рівнів складності розв'язання задач структурного синтезу.
4. Дати порівняльну характеристику найбільш розповсюдженим методам синтезу.
5. Що таке СМО та які основні структури СМО існують?
6. Чому аналітичні моделі СМО мають обмежене застосування?
7. Дайте визначення мережі Петрі.
8. Як відбувається виконання мережі Петрі?
9. Дайте характеристику основним властивостям мереж Петрі та методам їх аналізу.
10. Як за допомогою мереж Петрі моделюють процеси в складних системах?
11. З чим пов'язане виникнення великої кількості різновидів мереж Петрі?
12. В чому полягає алгоритм побудови дерева досяженості?
13. Для вирішення яких задач застосовують дерево досяженості?
14. Які різновиди мереж Петрі Вам відомі? Для вирішення яких задач їх можна використовувати?

3 ФУНКЦІОНАЛЬНО-ЛОГІЧНЕ ПРОЕКТУВАННЯ

3.1 Основні задачі етапу функціонально-логічного проектування

Основними задачами функціонально-логічного проектування є розробка функціональних та принципових схем пристроїв. Етап функціонально-логічного проектування наслідує етап системного проектування.

Розробка функціональних та принципових схем вимагає розв'язання задач синтезу та аналізу.

Задачу синтезу в спрощеному вигляді можна сформулювати так: задані вихідні функції пристрою (наприклад, у вигляді часових діаграм), необхідно отримати принципову схему в якомусь базисі елементів (наприклад, І-НІ). Слід визнати, що задачі синтезу в автоматичному режимі вирішити, як правило, не вдається, оскільки: по-перше, не всі процедури синтезу формалізовані (наприклад, відомо, як отримати комбінаційну схему в базисі І-НІ, а у випадку наявності в схемі елементів пам'яті, зворотних зв'язків сучасні алгоритми синтезу вже не працюють). По-друге, обсяг обчислень з ростом розмірності задачі росте не лінійно, а експоненціально (тобто в багатьох випадках не існує ефективних обчислювальних алгоритмів для розв'язання таких задач).

Виходячи з цього, можна стверджувати, що більша частина задач по створенню функціональних та принципових схем виконується людиною, та лише в окремих випадках для нескладних схем вдається отримати результати в автоматичному режимі. Також слід враховувати, що людина в цьому аспекті поки працює краще за комп'ютер, тобто схеми, які отримані людиною, в 2-3 рази менші за обсягом та працюють швидше від комп'ютерних.

Тому перевага на стадії функціонально-логічного проектування з точки зору автоматизації процесу віддається задачам аналізу. Головна мета аналізу – оцінка якості запропонованого варіанту схеми. Така оцінка вико-

нується машинним моделюванням роботи схеми. Це моделювання може бути багатоступеневим. Спочатку доцільно перевірити схему на відповідність заданим функціям без врахування затримок елементів, елементної бази, тощо. Таке моделювання виконується значно швидше. Після цього, можливо, виконується моделювання більш детально, з врахуванням затримок, з метою виявлення ризиків збою, критичних змагань сигналів, тощо.

Крім того, на етапі функціонально-логічного проектування вирішується задача синтезу тестів для контролю та діагностування радіоелектронної апаратури в процесі її виготовлення та експлуатації.

3.2 Моделювання цифрових пристроїв

Моделювання комбінаційних схем (схем без зворотних зв'язків) є порівняно простим. В процесі моделювання відбувається послідовне розповсюдження сигналів від входів до виходів схеми. При цьому активізуються моделі елементів – спеціальні процедури, які зберігаються в бібліотеках (базі даних). Моделювання схем зі зворотними зв'язками є більш складним процесом, оскільки на входах деяких елементів можуть змінюватись значення сигналів внаслідок наявності ланцюжків зворотних зв'язків. Тому моделювання таких схем – процес ітераційний, який може привести до виникнення генерації в схемі.

3.2.1 Ранжування функціональних схем

При моделюванні схеми необхідно визначити порядок реалізації рівнянь моделі згідно з розповсюдженням сигналів в реальній схемі. Для встановлення такого порядку треба вирішити задачу ранжування схем. Тут діють такі правила.

Для комбінаційних схем:

- 1-й ранг надається таким елементам, всі входи яких підключені до вхідних ланцюгів схеми;
- (k+1)-й ранг надається елементам, якщо хоча б один його вхід з'єднується з виходом елемента k-го рангу, а інші входи – з виходами елементів не вище k-го рангу.

Щоб вирішити задачу ранжування для схем зі зворотними зв'язками, потрібно визначити контури. В результаті розриву ланцюгів зворотних зв'язків отримуємо комбінаційну схему, що впорядковується, як було вказано вище (обрив дуг приводить до перетворення схеми в комбінаційну).

Таким чином, можна запропонувати такий алгоритм ранжування (ранги надаються не тільки елементам, а й ланцюгам зв'язку) [1]:

Перший етап. Вхідним ланцюгам схеми надається 1-й ранг.

Другий етап. Послідовно розглядаються всі елементи, яким ранг ще не присвоєно. Якщо хоча б один вхідний ланцюг елемента не має рангу, визначити його ранг неможливо. В іншому випадку

$$\mathbf{r}(\mathbf{d}_i) = \max_{\mathbf{k}_j \in \mathbf{A}} \mathbf{r}(\mathbf{k}_j),$$

де A – множина входів елемента.

Вихідним ланцюгам елемента d_i надається ранг $(r(d_i) + 1)$, якщо $r(d_i) \neq 0$.

Для схем з шинною структурою можлива ситуація, коли ланцюг містить виводи вихідних сигналів декількох елементів. В такому випадку присвоїти ранг елементам ланцюга можна лише тоді, коли визначені ранги всіх елементів, виходи яких належать даному ланцюгу.

Якщо елемент не має вхідних ланцюгів (це може статися після розриву зворотних зв'язків), йому надається 1-й ранг.

Другий етап повторюється до виконання однієї з умов:

а) всім елементам були присвоєні ранги;

б) жодному елементу неможливо присвоїти ранг. В цьому випадку переходимо до 3-го етапу.

Третій етап. В схемі виділяється контур і визначається ланцюг зворотного зв'язку. Для цього будується ланцюг елементів, яким ще не присвоєно ранг. Знаходимо елемент, що не має рангу. Серед його вхідних ланцюгів є такі, що не мають рангу (інакше елементу було б присвоєно ранг). Обираємо один з них і також включаємо в ланцюг. Далі знаходимо, вихід якого елемента з'єднаний з цим входом; його також включаємо в ланцюг. Цей процес продовжуємо, поки не зустрінемо елемент, який вже включено в ланцюг. Це вказує на існування контура. Щоб знайти сам контур, треба визначити, де в ланцюгу цей елемент зустрівся вперше. Всі елементи, які розташовані в ланцюгу праворуч від цього місця, утворюють контур.

Припустимо, що складено ланцюг елементів

$$d_{i1}, d_{i2}, \ldots, d_k, d_{ik+1}, \ldots, d_{is},$$

ранги яким ще не присвоєні. Якщо при дослідженні елемента d_{is} наступним визначається елемент d_{ik}, який вже включено в ланцюг, то отримуємо контур елементів

$$d_{ik}, d_{ik+1}, \ldots, d_{is-1}, d_{is}.$$

Вихідний ланцюг будь-якого з цих елементів можна вважати ланцюгом зворотного зв'язку. Після визначення контура переходимо до четвертого етапу.

Четвертий етап. Обираємо ланцюг та розриваємо зворотний зв'язок. Обирається вихідний ланцюг елемента контура, який є вхідним ланцюгом більш ніж одного елемента схеми. Після розриву зворотного зв'язку вихідний ланцюг елемента d_{ik+1} вилучається з списку вхідних ланцюгів d_{ik}. Після розриву ланцюга зворотного зв'язку переходимо до другого етапу ранжування схеми. Робота алгоритму закінчується, коли всім елементам схеми буде присвосно ранг. Алгоритм є швидкодійним та просто реалізується на

ЕОМ, хоча й не гарантує, що кількість зворотних зв'язків буде мінімальною.

Розглянемо алгоритм ранжування на прикладі схеми, що подана на рис.3.1.

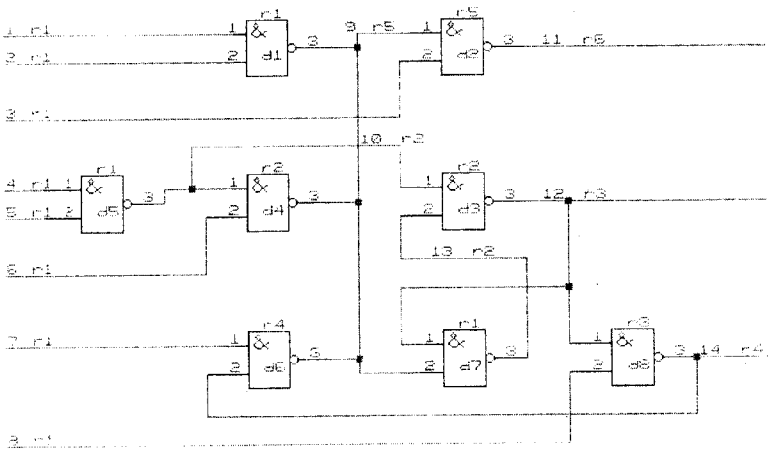


Рисунок 3.1 – Приклад схеми електричної

Перший етап. Вхідним ланцюгам надається перший ранг:

$r(k_1) = r(k_2) = r(k_3) = r(k_4) = r(k_5) = r(k_6) = r(k_7) = r(k_8) = 1.$

Другий етап.

1-а ітерація. Ранг елемента d_1 дорівнює максимальному рангу вхідних ланцюгів елемента:

$$r(d_1) = \max [r(k_1), r(k_2)] = 1.$$

Оскільки ранги елементів d_4 та d_6 не визначені, неможливо присвоїти ранг вихідному ланцюгу k_9. Також не вдається присвоїти ранги елементам d_2, d_3, d_6, тому що невідомі ранги ланцюгів k_9 та k_{10}.

Визначаємо ранг елемента d_5 та його вихідного ланцюга:

$$r(d_5) = \max [r(k_4), r(k_5)] = 1;$$

$$r(k_{10}) = r(d_5) + 1 = 2.$$

2-а ітерація.

$$r(d_4) = \max [r(k_{10}), r(k_6)] = 2.$$

3-я ітерація. Під час третьої ітерації не присвоєно ранг жодному новому елементу, тому переходимо до третього етапу.

Третій етап. Визначається елемент з неприсвоєним рангом. Припустимо, це елемент d_2. Включаємо його в ланцюг як початковий. Вхідному ланцюгу k_9 елемента d_2 не присвоєно ранг, тобто $r(k_9) = 0$. Ланцюг k_9 містить виходи декількох елементів. Елемент d_6 ($r(d_6) = 0$) включаємо в ланцюг $[d_2,d_6]$. Вхідному ланцюгу k_{14} елемента d_6 не присвоєно ранг, тобто $r(k_{14})=0$. В ланцюг $[d_2, d_6, d_8]$ включаємо елемент d_8, вихід якого належить ланцюгу k_{14}. Далі, включаючи в ланцюг елементи d_3 та d_7, отримуємо $[d_2, d_6, d_8, d_3, d_7]$.

Для елемента d_7 маємо $r(k_9) = 0$ та $r(k_{12}) = 0$. Обираємо один ланцюг, припустимо k_{12}. Переходимо до елемента d_3, який вже зустрічався в ланцюгу. Відповідно елементи d_3 та d_7 об'єднані в контур. Переходимо до четвертого етапу.

Четвертий етап. Обирається вихідний ланцюг k_{12} елемента d_3 як ланцюг зворотного зв'язку. Він є входом для двох елементів d_7 та d_8. Ланцюг k_{12} вилучається зі списку вхідних ланцюгів елемента d_7. Переходимо до другого етапу.

Другий етап. Під час другого етапу жодному новому елементу ранг присвоїти не вдається. Переходимо до третього етапу.

Третій етап. Таким же чином складається ланцюг, починаючи з елемента d_2: $[d_2, d_6, d_8, d_3, d_7]$.

В цьому випадку для елемента d_7 лише $r(k_9) = 0$. Ланцюг k_{12} з списку вхідних ланцюгів елемента d_7 вилучений. Ланцюг k_9 містить вихід елемента d_6, для якого $r(d_6) = 0$.

Переходимо до елемента d_6, який вже включено в ланцюг. Відповідно елементи d_6, d_8, d_3, d_7 об'єднані в контур. Переходимо до четвертого етапу.

Четвертий етап. Обираємо вихідний ланцюг k_9 елемента d_6 як ланцюг зворотного зв'язку. Він є вхідним ланцюгом для двох елементів d_2 та d_7. Ланцюг k_9 вилучається зі списку вхідних ланцюгів елемента d_7. Переходимо до другого етапу.

Другий етап.

1-а ітерація. Елемент d_7 не має вхідних ланцюгів і йому надається перший ранг:

$$r(d_7) = 1; \quad r(k_{13}) = 2.$$

2-а ітерація.

$$r(d_3) = \max [r(k_{10}), r(k_{13})] = 2;$$

$$r(k_{12}) = r(d_3) + 1 = 3.$$

3-я ітерація.

$$r(d_8) = \max [r(k_{12}), r(k_8)] = 3;$$

$$r(k_{14}) = r(d_8) + 1 = 4.$$

4-а ітерація.

$$r(d_6) = \max [r(k_{14}), r(k_7) = 4.$$

Тепер можна визначити ранг ланцюга k_9, оскільки визначені ранги всіх елементів, виходи яких належать цьому ланцюгу:

$$r(k_9) = \max [r(d_6), r(d_4), r(d_2)] + 1 = 5.$$

5-а ітерація.

$$r(d_2) = \max [r(k_9), r(k_3)] = 5;$$

$$r(k_{11}) = r(d_2) + 1 = 6.$$

Таким чином, всім елементам і ланцюгам схеми присвоєно ранги. Ранжування завершено.

3.2.2 Математичні моделі функціональних схем

Моделюванню функціональних схем цифрової радіоелектронної апаратури притаманні такі особливості [13]:

1. Стан елементів схеми характеризується фазовими змінними одного типу, якими позначається інформація; фізична природа змінної (струм, напруга) не конкретизується.

2. Фазові змінні доцільно подавати в дискретній формі, оскільки інформація має цифрову форму. Звичайно фазові змінні можуть приймати два значення – 0 та 1, тоді моделювання називають двозначним. У ряді випадків аналізу функціональних схем зручно використовувати трьох- та п'ятизначні змінні.

3. Аналіз функціональних схем відбувається в дискретному часі. Вісь часу розділяється на такти тривалістю Т. Зміна значення хоча б одної фазової змінної називається подією; в більшості випадків вважається, що події відбуваються миттєво. Якщо подія відбулася між моментами часу (t+nT) та (t+(n+1)T), то в моделі цю подію відносять до моменту часу (t+(n+1)T). Часто використовується відносний час, який виражають в кількості тактів (відносний час – частка від ділення абсолютного часу на Т).

Модель елемента функціональної схеми в загальному випадку задається системою рівнянь виду

$$\begin{cases} \mathbf{Y} = \psi(\mathbf{X}, \mathbf{A}); \\ \mathbf{A}' = \varphi(\mathbf{X}, \mathbf{A}), \end{cases} \qquad (3.1)$$

де X – вектор вхідних змінних елемента; A та A′ - вектори внутрішніх змінних, що характеризують стан елементів; Y – вектор вихідних змінних.

Якщо елементи векторів X та A відносяться до моменту відносного часу t, то елементи y_j вектора Y – до моменту часу $(t + k_j)$, причому затримки k_j можуть бути різними. Також затриманими в часі можуть бути елементи вектора A′.

В окремому випадку одновихідного комбінаційного елемента модель (3.1) приймає вигляд

$$\mathbf{y(t + k) = \psi\ (X(t)),} \qquad (3.2)$$

де ψ -- логічна функція.

Математична модель функціональної схеми – сукупність математичних моделей елементів, в яких виконано ототожнення фазових змінних, що відносяться до виводів елементів, з'єднаних між собою.

Для запису математичної моделі функціональної схеми в загальному випадку зручно ввести такі позначення: U – вектор вхідних для схеми величин; V та V′ - вектори вихідних та внутрішніх змінних всіх елементів схеми для поточного t та майбутнього t′ часу відповідно. Для елемента v_j' вектора **V′** момент майбутнього часу t′ визначається як $(t+k_j)$, тобто значення затримок k_j можуть бути різними для різних змінних.

Вихідні змінні функціональної схеми є вихідними змінними деяких елементів і, відповідно, входять в вектори V та V′. Тоді математична модель функціональної схеми - це система рівнянь

$$\mathbf{V' = F(V, U)} \qquad (3.3)$$

з дискретними змінними V′, V, U, в якій F - це оператор перетворення дискретних змінних.

Модель (3.3) називається асинхронною моделлю функціональної схеми.

Якщо всі затримки $k_j \neq 0$, то асинхронна модель (3.3) є сукупністю рекурентних співвідношень, що дозволяють при відомих значеннях змінних V та заданому законі змінення U(t) обчислити значення вектора V′ по всьому часовому діапазону, тобто побудувати часові діаграми роботи пристрою. Асинхронну модель можна вважати дискретним аналогом динамічної моделі.

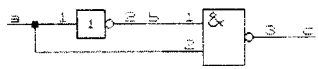
Асинхронні моделі є універсальними, оскільки можуть застосовуватись для аналіза як асинхронних, так і синхронних схем (тобто таких, в яких всі затримки $k_j = 0$), дослідження перехідних процесів та дослідження усталених станів схеми. Проте використання асинхронних моделей супроводжується великими витратами машинного часу. Скоротити ці витрати можливо при аналізі синхронних схем, в яких передача інформації між регістрами дозволяється лише при наявності спеціальних синхронізувальних імпульсів. Для таких схем достатньо виконати аналіз лише в усталеному режимі. При $k_j = 0$ маємо $V' = V$ і синхронну модель отримуємо з асинхронної моделі (3.3) в формі системи рівнянь, що називають логічними рівняннями

$$\mathbf{V = F(V, U)}, \tag{3.4}$$

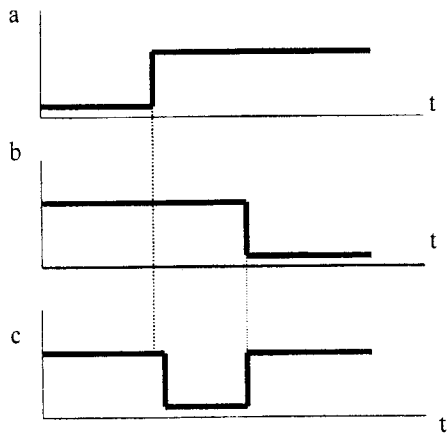
в яких час не фігурує. Отримана система рівнянь є дискретним аналогом неперервних статичних моделей. Синхронні моделі використовують для перевірки коректності з'єднань елементів в функціональній схемі, для виявлення ризиків збою, при побудові тестів. Якщо в синхронній моделі використовуються булеві змінні, то модель називається двозначною моделлю. Двозначні моделі найбільш економічні, моделювання в двійковому алфавіті найбільш швидке, проте за їх допомогою можливо вирішувати лише частину задач, наприклад, виявлення грубих помилок при побудові функціональної схеми. Але вони не можуть адекватно вирішувати ряд інших задач, наприклад, виявлення ризиків збою.

Ризиком збою називають можливість появи хибних сигналів. Якщо сигнали на виході схеми для двох сусідніх наборів вхідних сигналів A та B залишаються однаковими, але під час перехідного процесу можлива поява хибного сигнала протилежного значення, то така ситуація називається *статичним ризиком збою.*

Статичний ризик збою ілюструється на рис.3.2. Внаслідок затримки сигналу в точці b відносно точки a, зміна сигналу в точці *a* з 0 на 1 приводить до появи на виході *c* хибного сигналу: сигнал *a* змінюється (0→1) раніше сигналу *b* (1→0).



а)



б)

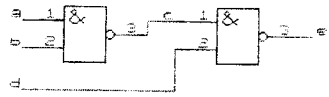
Рисунок 3.2 – Статичний ризик збою: а) схема; б) часові діаграми

Динамічний ризик збою припускає можливість багатократної зміни значень сигналу на виході при переході від вхідного набору A до набору B, коли вихідний сигнал схеми змінюється на протилежний.

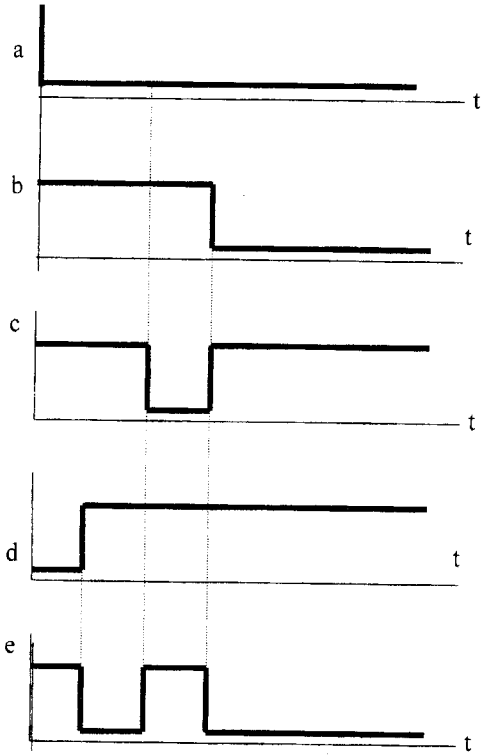
Зміна 0→1 сигналу *d* раніше зміни 0→1 та 1→0 сигналів *a* та *b* при переході від вхідного набору A= *abd* = 010 до набору B= *abd*=101 приводить до динамічного ризику збою (рис.3.3).

З часової діаграми роботи елементів схеми бачимо, що динамічний ризик збою є наслідком статичного ризику збою. При аналізі схем для ви-

явлення ризику збою необхідно враховувати часове неузгодження вхідних сигналів елементів.



а)



б)

Рисунок.3.3 - Динамічний ризик збою: а) схема; б) часові діаграми

При аналізі логічних схем часто використовують різні види моделювання, які дозволяють відтворити поведінку схеми або її окремих елементів при подачі на схему набору вхідних сигналів. Методи моделювання но-

сять універсальний характер і застосовуються в тому випадку, коли прямі методи перевірки не розроблені або виявляються занадто складними.

При синхронному моделюванні імітується лише логіка роботи схеми на основі ідеалізованої моделі схеми. Моделювання схеми зводиться до розв'язання системи рівнянь, кожне з яких моделює логіку перемикання одного елемента. Процес перемикання елементів в схемі при моделюванні синхронізується упорядкуванням елементів за рівнями запуску (згідно з алгоритмом ранжування). Моделювання починається з завдання сигналів на незалежних входах схеми, далі моделюються елементи 1-го, 2-го та наступних рангів. Для схем зі зворотними зв'язками моделювання повторюється декілька разів, поки схема не перейде в стійкий стан або не виникнуть коливання сигналів.

Для виявлення статичного ризику збою в комбінаційних схемах використовується трійкове моделювання: вводиться значення сигналу під час перехідного процесу, яке позначається *x*. Правила реалізації функцій І, АБО, НІ наведені в табл. 3.1.

Таблиця 3.1 - **Правила реалізації функцій І, АБО, НІ при трійковому моделюванні**

І			АБО			НІ	
A	B	Y	A	B	Y	a	Y
0	0	0	0	0	0	0	1
0	X	0	0	X	x	x	X
0	1	0	0	1	1	1	1
X	0	0	X	0	x	-	-
X	X	X	X	X	x	-	-
X	1	X	X	1	1	-	-
1	0	0	1	0	1	-	-
1	X	X	1	X	1	-	-
1	1	1	1	1	1	-	-

При трійковому моделюванні схеми крім вхідних наборів A та B використовується набір A/B. що визначає стан схеми під час перехідного процесу.

Якщо значення вихідних сигналів логічного елемента для вхідних наборів A та B збігаються, а для перехідного набору A/B зафіксовано невизначений стан *x*, на виході елемента може появитися хибний сигнал і схема містить статичний ризик збою.

Для схеми, що приведена на рис.3.3(а), значення сигналів при трійковому моделюванні наведені в табл.3.2.

Таблиця 3.2 - **Значення сигналів при трійковому моделюванні**

	a	b	D	c	e
A	0	1	0	1	1
A/B	x	x	x	x	x
B	1	0	1	1	0

Розглянемо точку *c*. Оскільки для вхідних наборів A та B значення сигналу не змінюється, а для перехідного набору A/B змінюється, в схемі існує статичний ризик збою. Таким чином, трійкове моделювання дозволяє виявити статичний ризик збою. Проте наявність в схемі динамічного ризику збою трійкове моделювання виявити не може. З метою виявлення динамічного ризику збою застосовують п'ятизначне моделювання. Правила реалізації функцій І, АБО, НІ та особливості моделювання в п'ятизначному алфавіті описані в [1].

Внутрішній стан схеми з пам'яттю визначається сигналами зворотного зв'язку. Вхідний набір може змінити значення кількох зворотних зв'язків. Затримки в схемі викликають змагання сигналів, які можуть виявитися критичними. В залежності від послідовності зміни значень сигналів зворотного зв'язку схема переходить в різні стійкі стани. Критичні зма-

гання сигналів можуть бути виявлені за допомогою трійкового моделювання.

Для схем з пам'яттю, на відміну від комбінаційних, трійкове моделювання триває до тих пір, поки значення сигналів в схемі не перестануть змінюватись. Моделювання починається від деякого заданного стійкого стану, який визначається сигналами зворотного зв'язку Y_A та вхідним набором А. Багатократним розв'язанням систем рівнянь цифрова схема моделюється для перехідного набору А/В та набору В. Якщо в результаті хоча б один сигнал зворотного зв'язку приймає значення *x*, то в схемі мають місце критичні змагання сигналів.

Під час моделювання перехідного набору А/В значення *x* з'являються на виводах лише тих елементів, сигнали яких можуть змінитися. Якщо значення x приймає ланцюг зворотного зв'язку, і наступний набір В не змінює значення зворотного зв'язку, а лише підтримує його, має місце неоднозначна поведінка схеми. Саме така ситуація і носить назву **критичних змагань сигналів**. Проте трійкове моделювання не враховує конкретних значень затримок в схемі, припускаючи найгіршу комбінацію затримок. При проектуванні схем звичайно враховують конкретні затримки. Тому деякі спроектовані людиною схеми з пам'яттю функціонують надійно, хоча трійкове моделювання попереджає про наявність критичних змагань.

Для схеми, наведеної на рис.3.4, трійкове моделювання показує наявність статичного ризику збою та критичних змагань.

Маємо вхідний набір А = *abcd* = 1111 і стан схеми, який визначається сигналом ланцюга зворотного зв'язку *h* = 1. При трійковому моделюванні маємо перехідний набір А/В = 11*x*1. Ланцюг зворотного зв'язку приймає значення *x*, що свідчить про наявність критичних змагань. Насправді хибний сигнал на виході елемента d2 не з'являється, і ланцюг зворотного зв'язку утримує те ж саме значення, оскільки сигнал *c* змінюється раніше сигналу *e* (рис.3.5).

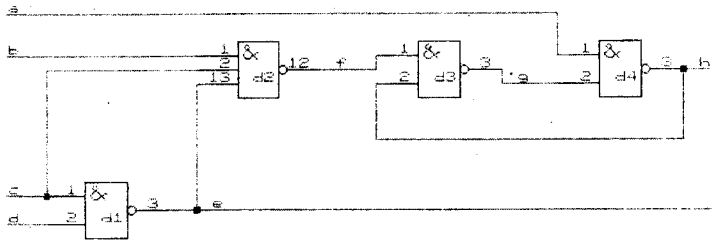


Рисунок 3.4 - Приклад цифрової схеми

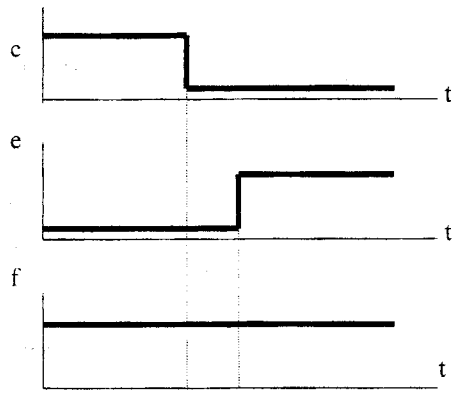


Рисунок 3.5 - Часові діаграми роботи схеми, наведеної на рис.3.4

Найбільш точно поведінку логічних схем відображує асинхронне моделювання. Асинхронна модель будується на припущенні, що кожний елемент схеми затримує сигнал на час $\Delta\tau$. Для аналізу перехідного процесу схему необхідно розглядати як асинхронну в межах одного такта її роботи. Це означає, що елементи переключаються асинхронно зі зміненням сигналів на їх входах, але тривалість перехідного процесу не перевищує тривалості такту роботи схеми ΔT, який повинен бути кратним абстрактній затримці $\Delta\tau$ і не повинен бути меншим за максимальну затримку сигналу в

схемі. Оскільки тривалість вхідних сигналів дорівнює такту роботи схеми, то до моменту зміни сигналів на входах схеми перехідний процес закінчується.

Моделювання станів схеми здійснюється в дискретні моменти часу з кроком, який дорівнює абстрактній затримці $\Delta\tau$. За кожний такт моделювання сигнали в схемі просуваються на час, рівний $\Delta\tau$. Для кожного елемента схеми вводиться поняття збудження та реакції. Будь-яка дія на входи елемента приводить до збудження, яке досягає виходу елемента через час $\Delta t = k\Delta\tau$. Елемент знаходиться в стійкому стані, якщо його реакція не протирічить збудженню. Якщо всі елементи знаходяться в стійкому стані, то і схема перебуває в стійкому стані.

Асинхронні моделі дозволяють відтворити часову послідовність усіх подій для пристрою, що досліджується, тобто обчислити та побудувати часові діаграми для різних вхідних наборів. Завдяки їм, зокрема, можна встановити, чи є в схемі критичні змагання сигналів.

Але більша точність та універсальність асинхронних моделей досягається за рахунок збільшення трудомісткості обчислень при аналізі. Якщо при синхронному моделюванні обчислення рівнянь виконується лише один або два рази, то при використанні асинхронних моделей розв'язання системи рівнянь звичайно виконується $\Delta T/\Delta\tau > 1$ раз.

Асинхронне моделювання, так само як і синхронне, може виконуватись в двозначному, тризначному і п'ятизначному алфавіті сигналів.

Алгоритм аналізу функціональної схеми, яка описується асинхронною моделлю (3.4), потребує виконання кількох кроків. Обсяг обчислень в простіших алгоритмах визначається кількістю виразів, що входять в (3.4). При аналізі схем, які складаються з тисяч та мільйонів логічних елементів, цей обсяг може виявитися занадто великим. Тому в програмах аналізу функціональних схем частіше використовуються алгоритми, які реалізують подійний метод.

Основна ідея подійного методу полягає в виконанні обчислень лише за рівняннями тільки активних елементів, тобто елементів, у яких хоча б на одному вході відбулася подія (змінилась вхідна змінна). На кожній ітерації обчислювального процесу існує своя група активних елементів. Тому в алгоритмі подійного методу на кожному кроці встановлюється, які елементи є активними і виконуються звертання лише до їх моделей. В складних функціональних схемах реальної РЕА на кожному такті, як правило, активізується не більше кількох відсотків від загальної чисельності елементів. Тому використання подійного методу дозволяє істотно скоротити витрати машинного часу при аналізі схем.

Для розв'язання систем логічних рівнянь $\mathbf{V} = \mathbf{F}(\mathbf{V}, \mathbf{U})$ застосовуються ітераційні методи: метод простої ітерації, метод Зейделя без ранжування та з ранжуванням. Перед розв'язанням системи в кожному з цих методів повинні бути задані вектор вхідних сигналів **U** та вектор початкового наближення $\mathbf{V}_0$ для вектора **V**. Розглянемо зазначені методи.

Метод простої ітерації. Цей метод передбачає виконання ітерацій за формулою

$$\mathbf{V}_i = \mathbf{F}(\mathbf{V}_{i-1}, \mathbf{U}), \qquad (3.5)$$

де $\mathbf{V}_i$ – значення вектора **V** на і-й ітерації.

В результаті, якщо $\mathbf{V}_i = \mathbf{V}_{i-1}$, то розв'язок знайдено; якщо $\mathbf{V}_i \neq \mathbf{V}_{i-1}$, то виконується нова ітерація; якщо ітераційний процес не сходиться, то це свідчить про помилки проектування схеми пристрою. На практиці вважають, що процес не сходиться, якщо умова $\mathbf{V}_i = \mathbf{V}_{i-1}$ не досягається при заданій кількості ітерацій.

Алгоритм методу простої ітерації аналогічний алгоритму асинхронного аналізу за умов, коли всі елементи схеми мають однакові затримки. Такий збіг не випадковий: за методом простої ітерації визначається усталене

значення **V**; таке ж саме усталене значення **V** буде знайдено при асинхронному моделюванні до моменту, коли перехідні процеси в схемі закінчаться.

Як зазначено вище, асинхронний аналіз має високу трудомісткість; така ж висока трудомісткіть притаманна методу простої ітерації. Зменшити обсяг обчислень вдається, якщо побудувати ітераційний процес за **методом Зейделя**. Особливість ітерації за методом Зейделя полягає в тому, що при обчисленні чергового елемента вектора $\mathbf{V}_i$ в праву частину (3.6) там, де це можливо, підставляються не елементи вектора $\mathbf{V}_{i-1}$, а ті елементи вектора $\mathbf{V}_i$, які вже обчислені до цього моменту. Кількість ітерацій в методі Зейделя істотно залежить від порядка, в якому реалізуються рівняння моделі.

В **методі Зейделя без ранжування** рівняння моделі перераховуються в довільному порядку.

Метод Зейделя з ранжуванням. Тут рівняння розташовуються в тому порядку, який відповідає проходженню сигналів. Тоді для аналізу схем без зворотних зв'язків потрібна буде лише одна ітерація. В схемах зі зворотними зв'язками метод Зейделя з ранжуванням вимагає кількох ітерацій, але їх число істотно менше, ніж в методі простої ітерації.

Як бачимо, метод Зейделя без ранжування за трудомісткістю обіймає проміжне становище між методом простої ітерації та методом Зейделя з ранжуванням. Подальше скорочення обсягу обчислень досягається за допомогою подійного методу, в якому скорочується трудомісткіть кожної ітерації. Подійний метод можна застосовувати в межах методів простої ітерації та Зейделя при синхронному та асинхронному моделюванні.

3.3 Генерація тестів для цифрових схем

Перевірка цифрових вузлів РЕА відбувається шляхом подачі на входи схеми деяких наборів з множини наборів теста і порівняння результатів експерименту з реакцією справної схеми. За допомогою тестів контролю

визначається наявність або відсутність несправностей в схемі. Будь-яка несправність з класу несправностей, для пошуку яких будується тест, має проявитись хоча б на одному наборі.

За допомогою діагностичного тесту локалізується тип та місце несправності в схемі. Діагностичний словник містить інформацію про всі несправності в схемі та відповідні реакції на виходах схеми.

Основними характеристиками тестів є:

1. **Повнота тесту**. Повний тест – це тест, реалізація якого дозволяє виявити всі несправності із заданого класу несправностей. Кількісною оцінкою повноти контролю є відношення кількості виявлених несправностей до загального числа можливих несправностей заданого класу.

2. **Мала трудомісткість контролю**. Трудомісткіть контролю безпосередньо пов'язана з кількістю наборів в тесті. Тому для оцінки трудомісткості звичайно використовують довжину тесту.

3. Для діагностичних тестів важливою характеристикою також є **ступінь локалізації несправностей** з множини всіх можливих несправностей схеми. Звичайно тест дозволяє локалізувати місце несправності з точністю до виводу елемента, з точністю до елемента, до групи елементів, до функціонального блока, тощо.

При побудові тестів до уваги береться обмежений клас несправностей, які підлягають виявленню та локалізації. Як правило, в цей клас включають несправності, що мають такі наслідки, як і обриви та короткі замикання виводів. Ці несправності можуть бути такими:

- *константний нуль*, коли на виході якогось елемента присутнє постійне значення логічного нуля "0", що не залежить від значень сигналів на входах елемента;
- *константна одиниця* – те ж саме, але при наявності на виході значення логічної одиниці "1".

Звичайно тести розробляються для пошуку поодиноких несправностей. Задача побудови тестів для пошуку кратних несправностей, з одного боку вимагає великих витрат машинного часу, а з іншого – не має практичного сенсу, оскільки доведено, що тести, які виявляють поодинокі несправності, дозволяють знайти також і кратні.

Останнім часом для побудови тестів використовуються такі методи:

- ймовірностні процедури;
- d-алгоритм;
- булева різниця;
- еквівалентна нормальна форма.

В усіх названих методах утворюється активізований шлях до деякого виходу схеми. Відрізняється лише спосіб обчислення вхідних сигналів для побудови активізованого шляху.

Найбільш універсальним є **ймовірнісний метод**, який передбачає підбір тестів на основі псевдовипадкових вхідних послідовностей, які спочатку проходять перевірку коректності, а потім повноти або діагностичних властивостей. Процедура закінчується, якщо отримано тест заданої повноти або ефективність нових наборів стає занадто малою. Ефективність виражається кількістю несправностей, які перевіряються даною послідовністю.

Універсальним способом визначення характеристик тесту є моделювання справної схеми S_0 та усіх її модифікацій $S_1, S_2, \ldots, S_k$, в яких містяться несправності 1, 2, ..., k з наступним порівнянням реакції схеми S_j з реакцією схеми S_0. Несправність перевіряється тестом, якщо реакції схем S_j та S_0 відрізняються між собою. Несправність вводиться в модель зміною відповідних констант моделі.

В цьому випадку треба моделювати роботу схеми $(1 + | F |)$ разів, де F – множина несправностей, для яких будується тест. Таким чином можна

отримати тести як для комбінаційних схем, так і для схем з пам'яттю. Побудова тестів за допомогою ймовірнісних процедур описана в [23].

За допомогою ймовірнісного методу звичайно отримують тести, повнота яких значно менша за 100%. Для пошука наборів, що покривають несправності, які не охоплені випадковим тестом, застосовують детерміновані методи синтезу. Це пояснюється тим, що таких несправностей залишається небагато, а використанню детермінованих методів віддають перевагу при синтезі тестів для порівняно нескладних пристроїв.

Більшість детермінованих методів базується на таких положеннях:

1. Вхідні набори тесту підбирають по черзі для всіх несправностей з заданного списку.
2. Для кожної несправності визначають шляхи, які складаються з елементів схеми і з'єднують несправний елемент з виходами схеми.
3. Для виявлення несправності необхідно забезпечити проходження неправильного сигналу від несправного елемента до виходів схеми, тобто подати на інші входи елементів, що входять в шлях, такі значення сигналів, які забезпечать транспортування несправності до виходу. Шлях, приведений в такий стан, називається активізованим.
4. Для виявлення несправностей та активізації шляхів підбирають відповідний вектор вхідних змінних, який і складає черговий набір тесту.

Найбільш наочно процес побудови тестів представляє активізація одновимірних шляхів. Рот [1] формалізував послідовність дій з активізації шляхів у вигляді d-алгоритму, який став одним з фундаментальних методів синтезу тестів.

d-алгоритм синтезує тест для деякої конкретної несправності, що існує в схемі. Цей алгоритм по суті являє собою неявну процедуру перебору

при обчисленні зворотного шляху. Для кожного елемента схеми можна обрати вхідні сигнали, які перевіряють всі константні несправності елемента. Основна проблема полягає в тому, що ми не можемо безпосередньо подавати сигнали на входи елемента і спостерігати реакцію елемента на його виході, оскільки він "захований" в середині схеми. А щоб судити про його реакцію на виходах схеми, треба визначити активізований шлях від елемента E_0, що містить несправність, через елементи E_1, E_2, ..., E_{m-1} до елемента E_m, вихід якого є одночасно виходом схеми.

Значення вихідного сигналу елемента E_i повинно залежати лише від значення вихідного сигналу тільки елемента E_{i-1} (E_i та E_{i-1} лежать на активному шляху). Інші значення вхідних сигналів елемента E_i обираються таким чином, щоб змусити елемент E_i "нести повну відповідальність" за значення вихідного сигналу елемента E_{i-1}. Узагальнюючи це положення, можна досягти такої ситуації, при якій про значення вихідного сигналу елемента E_0 можна було судити, спостерігаючи за виходами елемента E_m. Цей процес носить назву фази просування вперед або прямої фази.

Визначивши активний шлях, ми повинні знайти деякий первинний вхідний набір, який забезпечить всі можливі вхідні значення сигналів елементів E_0, E_1, ..., E_m. Для цього прокладається зворотний шлях по схемі від входів елементів E_0, E_1, ..., E_m до первинних входів схеми. Цей процес носить назву зворотної фази або фази просування назад.

При активізації одновимірного шляху може випадково активізуватися і інший шлях, який приховує несправності основного активізованного шляху. Такого роду явище може виникати кожен раз, коли два або більше шляхи, що розгалужуються від місця несправності, потім знову сходяться в одній точці, та парність кількості інверсій вздовж шляхів неоднакова. Слабе місце одновимірного методу полягає в тому, що кожен раз дозволяється активізувати лише один шлях, навіть при розгалуженнях, які сходяться.

Основна ідея алгоритмічного методу полягає в одночасній активізації всіх можливих шляхів від місця несправності до всіх виходів схеми. Ця ідея дозволяє позбутися основної вади одновимірного методу та приводить до d-алгоритму. Спочатку обирається тест для несправності, що разглядається, на мові його входів. Далі генеруються всі можливі шляхи від місця несправності до всіх виходів схеми. На кожному кроці відбувається відмова від тих шляхів, які неможливо активізувати через розгалуження, що сходяться. Цей крок є узагальненням прямої фази одновимірного методу. Нарешті робиться спроба побудувати первинний набір, який реалізує всі умови, розроблені під час виконання прямої фази.

При підборі значень вхідних сигналів елемента можливий випадок, коли невдалий вибір пізніше приведе до несумісності обраних значень сигналів. Якщо це відбудеться, алгоритм повинен "дати зворотний шлях", змінюючи вибір до тих пір, поки несумісність не буде усунена. В такому випадку виконується пошук по дереву з усіма добре відомими труднощами, які виникають при цьому.

d-алгоритм можна використовувати, щоб отримати один тест для кожної несправності та об'єднати всі ці тести. Такий підхід не дозволяє в повній мірі скористатися потужністю d-алгоритму, оскільки тест, побудований для певної несправності, звичайно, так же добре знаходить і багато інших несправностей. Для цього треба дослідити отриманий активізований шлях. Будь-яка несправність, яка лежить на такому шляху і викликає зміну значень сигналів вздовж шляху, також буде виявлена. Можна визначити множини поодиноких та кратних несправностей, що виявляються даним тестом, і для кожного елемента схеми обчислити, при яких несправностях змінюється значення вихідного сигналу елемента. Обчислюється функція, що вказує, які несправності впливають на значення вихідного сигналу елемента.

При відповідній модифікації d-алгоритм дозволяє знаходити тести виявлення декількох несправностей, розрізняти дві несправності, а також виявляти будь-які несправності, що відносяться до класу можливих несправностей.

Характеристику методів булевої різниці та еквівалентної нормальної форми можна знайти в [13].

Контрольні запитання

1. Які задачі розв'язуються на стадії функціонально-логічного проектування?
2. Чому на етапі функціонально-логічного проектування перевага віддається методам аналізу?
3. Як відбувається процес моделювання роботи цифрових пристроїв?
4. З яких етапів складається процес ранжування функціональних схем?
5. Дайте порівняльну характеристику синхронного та асинхронного моделювання.
6. Дайте порівняльну характеристику двійкового та трійкового моделювання.
7. Поясніть за допомогою часових діаграм, як проявляється статичний та динамічний ризик збою.
8. Що таке "критичні змагання сигналів" та як їх можна виявити при моделюванні?
9. Які методи застосовують для прискорення процесу моделювання?
10. Назвіть основні характеристики, які застосовують для оцінки якості тестів цифрових схем.
11. Дайте характеристику основних методів побудови тестів.
12. Як метод активізації шляху реалізується в рамках d-алгоритму?

4 КОНСТРУКТОРСЬКЕ ПРОЕКТУВАННЯ

4.1 Основні задачі етапу конструкторського проектування

Конструкторське проектування – один з найважливіших аспектів проектування складних пристроїв та систем. Початковими даними для конструкторського проектування є результати структурного та функціонального проектування, тобто структурні, функціональні та принципові схеми пристроїв. В свою чергу, результати конструкторського проектування (конструкторська документація і носії для станков з ЧПУ та автоматів для виготовлення фотошаблонів), служать основою технологічного проектування, тобто застосовуються для розробки технологічного процесу виготовлення готових виробів.

Для розробки конструкцій складних електронних систем в цілому характерне висхідне проектування: спочатку розробляються плати, далі модулі, блоки та пристрої. В свою чергу, на кожному з цих рівнів проектування послідовно розв'язуються такі задачі:

- компонування елементів конструкції у вузли даного ієрархічного рівня;
- розташування елементів конструкції по конкретних установлювальних місцях;
- трасування з'єднань між елементами.

Ця група задач відноситься до комутаційно-монтажного проектування і найбільше відображена в САПР.

Проте розв'язання задач комутаційно-монтажного проектування не завершує етапу конструкторського проектування. Інша важлива група задач – задачі аналізу розроблених варіантів конструкції, які містять аналіз теплових режимів в ній; аналіз завадостійкості при конструюванні елементів, вузлів та пристроїв; аналіз механічних характеристик конструкції. Слід відзначити, що ці види аналізу здійснюються за допомогою окремих при-

кладних програм, які не об'єднані в комплекс та не мають гармонійного зв'язку з програмами комутаційно-монтажного проектування.

Самостійна задача конструкторського проектування – виготовлення пакету конструкторської документації.

4.2 Компонування конструктивних вузлів

Функціональний поділ складного пристрою має вигляд ієрархічного дерева. Конструктивно пристрій побудовано за модульним принципом, згідно з яким він поділяється на декілька констуктивних одиниць, які, в свою чергу, поділяються на конструктивні одиниці більш низького рангу і т.д. В процесі компонування визначається однозначна відповідність між функціональним та конструктивним поділом пристрою. Технічне проектування відбувається у висхідному порядку: модуль – стійка – блок – комірка, тобто у зворотному порядку відносно функціонального проектування.

Модуль (плата з елементами) є мінімальною конструктивною одиницею, яку в процесі експлуатації можна замінити. Він звичайно складається з декількох функціональних або базових елементів.

При реалізації компонування повинні виконуватись вимоги:

1. Максимальне використання обладнання. При цьому найчастіше використовується мінімізація числа вузлів m.
2. Типізація, тобто мінімальна кількість різних типів вузлів.
3. Мінімальне число міжвузлових з'єднань f або мінімальне число зовнішніх виводів всіх вузлів f^*.

Звичайно вимоги третьої групи є головними, оскільки мінімізація числа міжвузлових з'єднань f веде до скорочення сумарної довжини з'єднань при розв'язанні всієї задачі технічного проектування. Компонування повинно сприяти полегшенню виконання наступних етапів розташування елементів та трасування з'єднань. Зменшенню кількості ланцюгів між елемен-

тами різних вузлів при постійній кількості міжвузлових з'єднінь f сприяє мінімізація сумарного числа зовнішніх виводів всіх вузлів f^*. Це наочно показано на рис.4.1, де наведені два різних варіанти компонування дев'яти елементів в три вузли. Компонування еквівалентні відносно критерія f, але не еквівалентні відносно критерія f^*. Неважко помітити, що при зменшенні f^* має місце тенденція до зменшення f та навпаки.

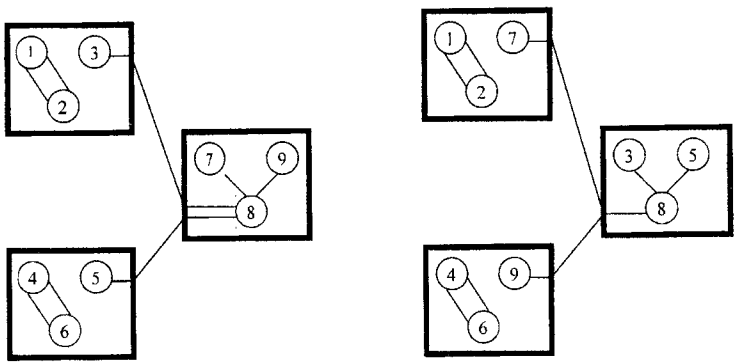


Рисунок 4.1 - Варіанти компонування конструктивних вузлів для а) f = 2, f^* = 4; б) f = 2, f^* = 3

4. Дотримання конструктивних обмежень:

- кількість елементів n_q вузла q не може первищувати граничного значення n:

$$\mathbf{n_q \leq n}, \tag{4.1}$$

де q = 1, 2, …, m.

- кількість зовнішніх виводів f^*_q вузла q не може перевищувати граничного значення M:

$$\mathbf{f^*_q \leq M}, \tag{4.2}$$

де q = 1, 2, …, m.

5. Нерозривність функціонального призначення вузлів.

6. Мінімальна зв'язність вузлів за назвами.

Якщо позначити число вузлів, з якими вузол q має зовнішні зв'язки, через S_q, то зв'язність за назвами можна виразити таким чином:

$$S = 1 / m \sum_{q=1}^{m} S_q , \qquad (4.3)$$

Мінімізація S сприяє полегшенню виконання етапу розташування елементів.

Всі відомі наближені алгоритми компонування можна віднести до однієї з двох груп:

1. Алгоритми послідовного заповнення вузлів.

2. Ітераційні алгоритми послідовного поліпшення наближень.

Спільним для всіх алгоритмів першої групи є вибір на кожному кроці для компонування одного елемента з максимальним значенням обраної норми S^a. Норма S^a в загальному випадку є функцією від деякого числа факторів:

$$S^a = f(S^a_1, S^a_2, \ldots, S^a_n). \qquad (4.4)$$

В алгоритмах використовують такі фактори.

Фактор кон'юнкції S^a_1, значення якого дорівнює числу з'єднань між досліджуваним елементом *a* та підмножиною елементів A_q, що призначені вузлу.

Фактор діз'юнкції S^a_2, значення якого дорівнює числу з'єднань, інцидентних досліджуваному елементу *a*, але не інцидентних підмножині A_q елементів, призначених вузлу, що формується.

Сутність алгоритмів послідовного поліпшення наближень полягає в тому, що деякі елементи міняються місцями з іншими елементами або пе-

реносяться на вільні місця незаповнених вузлів. Перед виконанням ітераційного алгоритму повинно бути задане початкове компонування, яке визначається довільно або після виконання іншого алгоритму. Зміна елементів відбувається для зменшення обраного критерію.

4.3 Розташування елементів

Розташування – це визначення положення модулів, комірок, блоків, стійок в системі, що проектується. Надалі розглянемо узагальнену задачу розташування елементів на платі, тобто в двокоординатному просторі.

Початковими даними для розташування є:

- схема з'єднань конструктивних елементів деякого вузла, отримана за результатами компонування;
- конструктивні параметри елементів (форма, розміри);
- параметри монтажного простору плати.

Найбільш привабливою метою розташування слід вважати максимальне зниження спотворень логічних сигналів та максимальне полегшення наступного трасування з'єднань. Точний розрахунок всіх затримок та відображень сигналів є навряд чи можливим внаслідок складності розрахунку електромагнітних полів, що виникають при взаємодії великої кількості джерел сигналів. Спотворення логічних сигналів так чи інакше пов'язане з довжиною з'єднань, яка, в свою чергу, залежить від розташування елементів плати. В зв'язку з цим можна виділити такі практичні критерії розташування [1]:

- мінімізація сумарної довжини L всіх з'єднань плати;
- мінімізація довжини найбільш довгого з'єднання плати M (критерій усереднення довжини);
- мінімізація кількості перетинів з'єднань, що відносяться до різних сигналів;

- максимізація кількості ланцюгів з можливо простою конфігурацією (звичайно – кількості лінійних ланцюгів).

За останні двадцять років вийшло багато робіт, де описані різні наближені алгоритми розташування [2, 3, 21]. Всі відомі алгоритми можна поділити на три групи :

1. Алгоритми, що використовують силові функції.
2. Ітераційні алгоритми послідовного поліпшення розташування, які використовують перестановки елементів.
3. Алгоритми послідовного розташування елементів.

Алгоритми, що використовують силові функції. Вперше метод силових функцій був запропонований в роботі Мілє [1] для розташування додаткових верхівок при розв'язанні однієї з модифікацій задачі Штейнера. При реалізації методу між розміщуваними вершинами вводяться сили притягання, пропорційні числу з'єднань і відстані між ними, і після розв'язку відповідної системи нелінійних диференційних рівнянь визначаються координати вершин, при яких система знаходиться в рівновазі. Відома також електрична інтерпретація алгоритму, вперше запропонована в [14].

Ітераційні алгоритми послідовного поліпшення розташування. Вперше ітераційний метод був запропонований в роботі Глейзера. Початкове розташування елементів на платі задається певним чином, і в подальшому використовуються попарні перестановки з метою зменшення сумарної довжини з'єднань. Існує багато робіт, де запропоновані різні модифікації ітераційного алгоритму. Зокрема в роботі [9] показана можливість застосування циклічних перестановок для додаткового поліпшення розташування, коли попарні перестановки не приводять до бажаного результату. Слід зазначити, що ефективність ітераційних алгоритмів в значній мірі залежить від початкового розташування, оскільки отриманий розв'язок приводить до локального мінімуму функції, що оптимізується.

Алгоритми послідовного розташування елементів. В цих алгоритмах початкову множину елементів розташовують на платі за визначену кількість кроків, меншу за n, оскільки до початку розташування частина елементів та роз'єми отримують фіксовані позиції. На кожному кроці обирається черговий нерозташований елемент x_i та встановлюється в незайняту позицію p_i. Надалі елемент не пересувається.

Правила вибору та установки чергового елемента визначають конкретні алгоритми. В найпростішому випадку вибір елемента базується на оцінці числа зв'язків елемента x_i з розташованими (з множини $\mathbf{X}^p$) та нерозташованими (з множини $\mathbf{X}^н$) елементами:

$$\mathbf{F(x_i)} = \sum_{\mathbf{x_j \in X^p}} \mathbf{a_{ij}} - \sum_{\mathbf{x_j \in X^н}} \mathbf{a_{ij}}. \qquad (4.5)$$

З усіх можливих елементів обирається той, у якого значення $F(x_i)$ максимальне. В більш складних випадках функція $F(x_i)$ враховує параметри електричних кіл, що пов'язують модулі, тоді схема з'єднань елементів подається дводольним графом. При наявності декількох елементів з однаковим значенням функції $F(x_i)$ застосовують правила "уточнення вибору": випереджувальний перегляд, обчислення вторинної функції або довільний вибір.

Найпростіше правило установки використовує оцінку сумарної довжини з'єднань елемента x_i з розташованими елементами з множини $\mathbf{X}^p$:

$$\mathbf{F(p_i)} = \sum_{\mathbf{x_j \in X^p}} \mathbf{a_{ij}}\, \mathbf{d(x_i, x_j)} \qquad (4.6)$$

Обирається саме та з позицій p_i, для якої значення $F(p_i)$ мінімальне.

Послідовні алгоритми приводять до менших витрат часу та пам'яті ЕОМ, ніж ітераційні.

4.3 Трасування з'єднань

Початковими даними для задачі трасування є параметри комутаційного поля, координати контактів мікросхем, перелік контактів елементів, що входять в конкретний ланцюг (для кожного ланцюга). Під час розв'язання задачі трасування треба з'єднати між собою контакти кожного комплексу (роз'єма, мікросхеми), оптимізуючи деяку функцію якості монтажа при дотриманні технологічних обмежень.

Основними обмеженнями задачі трасування є неприпустимість перетину в одному шарі проводників різних ланцюгів та метричні обмеження, пов'язані з монтажним простором та розмірами трас.

Функція якості монтажу (критерій оптимальності) в загальному випадку повинна враховувати такі параметри:

- сумарну довжину з'єднань;
- кількість додаткових міжшарових переходів;
- кількіть кутів в з'єднаннях;
- взаємні наводки трас різних ланцюгів;
- кількість шарів друкованого монтажу.

Результат розв'язку задачі трасування залежить не тільки від застосованого алгоритму прокладки трас, а ще й від розв'язання низки пов'язаних між собою задач. Серед них можна виділити чотири базові задачі [1].

1. Складання списку з'єднань.
2. Розподіл з'єднань по шарах.
3. Визначення порядку трасування в кожному шарі.
4. Проведення з'єднань або власне трасування.

Отже, для визначення трас з'єднань маємо отримати відповіді на питання: *що? де? коли?* та *як?*

При складанні списку з'єднань визначається, *що* повинно з'єднуватись, тобто формуються списки координат контактів комплексів,

які підлягають з'єднанню. При цьому бажано використовувати властивість інваріантності входів логічних схем.

Під час розподілу з'єднань по шарах вказується, *де* (в якому шарі) бажано проводити кожне з'єднання.

Для визначення, *коли* проводити трасу кожного з'єднання, списки з'єднань впорядковуються згідно з деякими параметрами.

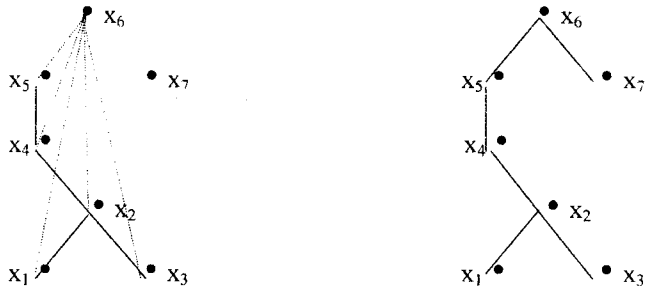
Як проводити трасу чергового з'єднання визначає алгоритм прокладки трас.

На першому етапі необхідно вирішити, в якій послідовності треба з'єднувати контакти одного ланцюга (тобто встановити перелік проводів одного ланцюга для кожної пари контактів), щоб сумарна довжина всіх з'єднань була мінімальна. Ця задача зводиться до задачі побудови мінімального зв'язувального дерева.

Найбільш розповсюджений розв'язок цієї задачі базується на **алгоритмі Прима**. На першому кроці алгоритму для довільного контакту знаходиться найближча вершина та з'єднується з ребром. На наступних (n-2) кроках з множини непідключених контактів обирається той, що знаходиться найближче до групи вже зв'язаних контактів та з'єднується ребром. На рис.4.2(а) показано фрагмент дерева (x_1, x_2, x_3, x_4, x_5) після четвертого кроку алгоритму, найближчий контакт x_6 та найкоротше ребро (x_5, x_6), що має мінімальну відстань d_{ij} серед всіх можливих ребер. В результаті перелік проводів для трасування ланцюга на рис.4.2(б) буде таким:

$$[(x_1, x_2)\ (x_2, x_3)\ (x_2, x_4)\ (x_4, x_5)\ (x_5, x_6)\ (x_6, x_7)].$$

Алгоритм Прима застосовується для визначення кількості перетинів проводів, оцінки якості розв'язку задачі розташування, а також трасування з'єднань. В останньому випадку обмежується кількість ребер, інцидентних одному контакту, через обмеження кількості пайок m на один контакт, тому алгоритм модифікується. При m=2 задача зводиться до відомої теоретичної задачі про комівояжера.



а) б)

Рисунок 4.2 - Мінімальне зв'язувальне дерево після виконання: а) четвертого кроку алгоритму; б) останього кроку алгоритму

Розподіл проводів по шарах може бути виконаний за допомогою двох стратегій:

1. Послідовно проводять з'єднання до заповнення чергового шару, після чого переходять до заповнення наступного шару (при цьому отримують велику кількість шарів та їх нерівномірне заповнення).
2. Підраховують можливу кількість перетинів проводів, розташованих в одному шарі, після чого виконують розподіл по шарах.

Підрахунок можливої кількості перетинів проводів здійснюється для одного з двох варіантів з'єднання контактів:

- з'єднання подаються у вигляді двох прямолінійних відрізків та їх перетини визначаються з рівнянь прямих ліній (рис.4.3(а));
- з'єднання подаються в ортогональному просторі, а їх перетини визначаються з рівнянь прямих, паралельних осям координат (рис.4.3(б)).

Обидві оцінки дають завищену кількість можливих перетинів та мають приблизно однакову ефективність.

Визначення порядка трасування проводів в кожному шарі пов'язано з тим, що результат трасування чергового проводу істотно залежить від

конфігурації вже проведених трас. Для розв'язання цієї задачі розроблені різні евристичні алгоритми, оскільки задача не формалізується теоретичними методами.

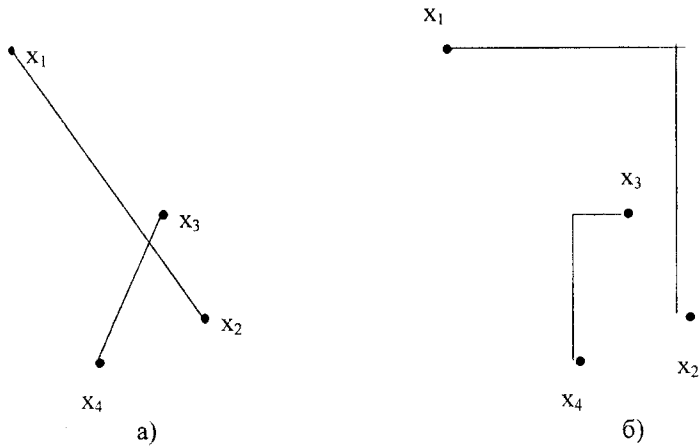


Рисунок.4.3 - Варіанти оцінки перетинів двох проводів

Найбільше розповсюдження отримали методи упорядкування, які базуються на оцінці довжини проводів. Можливі два різних підходи:

a) з'єднання проводів в порядку зростання довжини окремих проводів (за оцінку довжини приймається найкоротша відстань між з'єднуваними контактами);

b) з'єднання проводів в порядку зменшення довжини проводів, оскільки більш довгі проводи важче трасувати.

З точки зору мінімізації сумарної довжини з'єднань обидва підходи дають приблизно однакові результати.

Власне трасування з'єднань полягає в послідовній побудові трас в кожному шарі для всіх пар контактів з врахуванням заданих вимог та обмежень. Існуючі алгоритми проведення трас можна умовно поділити на групи, що базуються на ідеях хвильового та евристичного алгоритмів.

Хвильовий алгоритм або **алгоритм Лі** визначає шлях між двома комірками **A** та **B** дискретної решітки. Для оцінки якості шляху задається багатовимірна вагова функція $F = (f_1, f_2, ..., f_r)$, де f_i характеризує деякий параметр шляху (довжину, число перетинів, тощо). Під час визначення шляху деяким коміркам приписують масу $M = (m_1, m_2, ..., m_r)$. Вводиться поняття заборонених комірок, тобто таких, які не можуть бути включені в шлях.

В алгоритмі чітко виділяються дві частини: розповсюдження хвилі та проведення шляху.

При розповсюдженні хвилі, починаючи з комірки **A**, коміркам, сусіднім з вже розглянутими, крім заборонених, приписують значення маси, тобто формується фронт нової хвилі. Ця процедура триває до тих пір, поки комірці **B** не буде присвоєно значення маси. На зворотному шляху від комірки **B** до **A** з монотонно зменшуваними значеннями маси визначається шлях між **A** та **B** з мінімальним значенням функції F.

Для спрощення процедури проведення шляху при розповсюдженні хвилі коміркам надаються шляхові координати, тобто вказуються напрямки, протилежні розповсюдженню хвилі.

Геометрія шляху залежить від форми і відношення сусідства комірок. В роботі алгоритма Лі прийнята квадратна форма комірок і сусідніми вважаються квадрати з спільним ребром. Роботу алгоритму Лі ілюструє приклад, наведений на рис.4.4.

Алгоритми Лі визнають самими універсальними серед локально-оптимальних алгоритмів трасування. Їх недоліки – великий обсяг обчислень та необхідність використання великих об'ємів пам'яті. З метою скорочення витрат пам'яті та машинного часу запропоновано багато різних модифікацій хвильових алгоритмів.

Скорочення витрат пам'яті та часу ЕОМ можна досягти, застосовуючи променеві алгоритми, запропоновані Абрайтісом. Основна ідея променe-

вих алгоритмів полягає в дослідженні не всіх комірок, а тільки частини з них по деяких заданих напрямках [1].

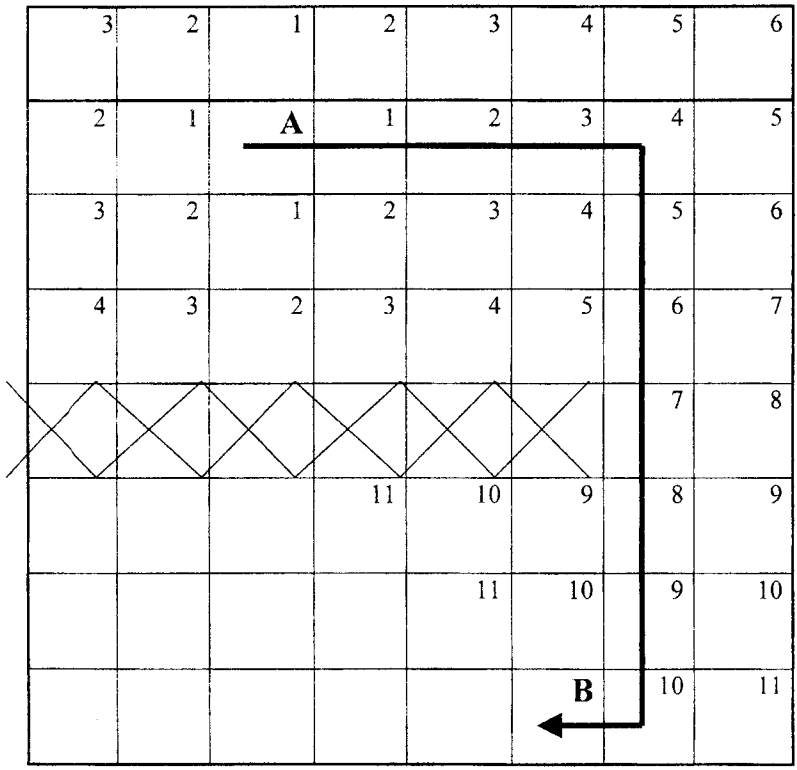


Рисунок 4.4 - Хвильовий алгоритм трасування

Розглянемо двопроменевий варіант алгоритму. В цьому варіанті від кожного контакту x_i та x_j розповсюджується по два промені, та кожний новий фронт містить не більше чотирьох елементів. Розповсюдження кожного променя припиняється, якщо всі сусідні квадрати, що обираються з числа можливих, зайняті або заборонені (блокування променя). Траса існує, якщо промені від різних контактів перетинаються в деякому квадраті.

В залежності від порядку контактів до початку алгоритму обчислюються пріоритетні напрямки розповсюдження променів.

Звичайно за допомогою променевого алгоритму вдається провести 80% трас, а більш складні конфігурації, нерозведені траси прокладають за допомогою хвильового алгоритму або вручну. Для реалізації всіх можливих променевих алгоритмів вводять спеціальні процедури виходу променів з тупиків.

Евристичні алгоритми трасування з'єднань є більш швидкодійними. На відміну від хвильового алгоритму в них не аналізуються всі можливі траси для вибору оптимальної, а одразу намагаються прокласти трасу по найкоротшому шляху.

Якщо на шляху зустрічаються перешкоди, то обминання здійснюється по першому вільному напрямку або згідно з правилами, що визначають порядок обминання перешкод. До початку роботи алгоритму задається пріоритетний порядок обминання перешкод.

Робота евристичного алгоритму ілюструється рис.4.5. Пріоритетні напрямки обминання перешкод при просуванні від **A** до **B** в цьому випадку будуть: донизу-праворуч та догори-праворуч. Загальний напрямок руху має проходити вздовж прямої, що з'єднує контакти **A** та **B**. Якщо немає перешкод, то обирається напрямок вздовж цієї прямої. Побудована за допомогою даного алгоритму траса не є оптимальною (оптимальна траса показана пунктиром), але вона отримана за малий час.

На користь евристичних алгоритмів говорять також такі дані: проведення вручну 1% трас, не прокладених хвильовим алгоритмом, може виявитися складнішим за проведення 20% з'єднань, що залишилися після евристичного алгоритму. Це пояснюється тим, що за допомогою евристичних алгоритмів звичайно проводяться тільки траси простіших конфігурацій без обминання перешкод, і на платі залишається достатньо місця для ручного проведення інших з'єднань.

Головною вимогою, що висувається при розв'язку задачі трасування є забезпечення проведення максимальної кількості з'єднань при обмежених

розмірах комутаційного поля. Розглянуті алгоритми розв'язання цієї задачі для багатошарових друкованих плат дозволяють здійстнити побудову в середньому 95 – 100% всіх з'єднань схеми. Кращі результати трасування забезпечують алгоритми ітераційного типу, але їх реалізація вимагає значних витрат часу та пам'яті ЕОМ.

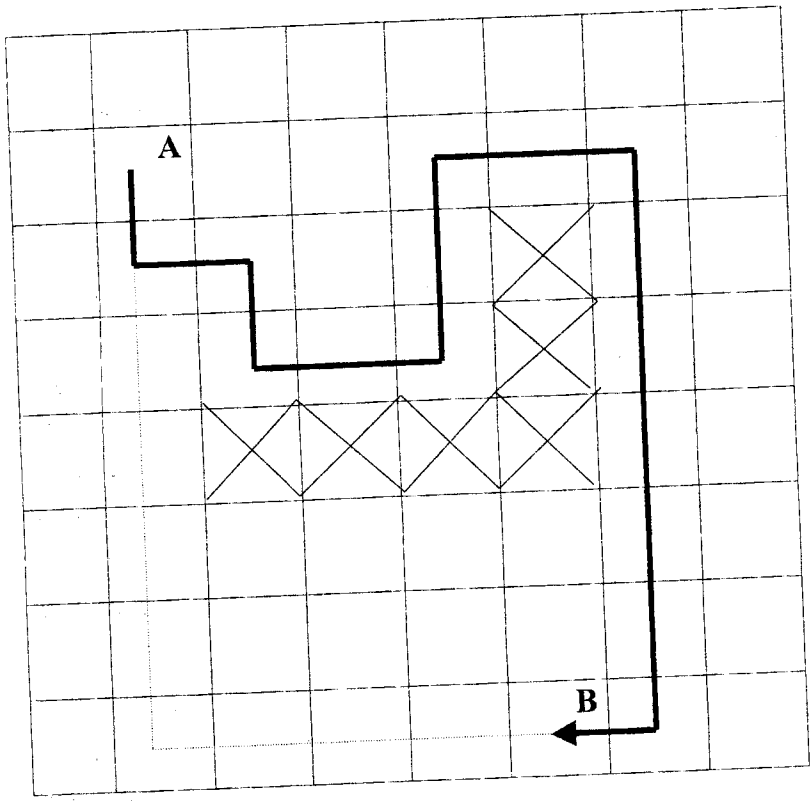


Рисунок 4.5 - Евристичний алгоритм трасування

В алгоритмах **ітераційного типу** на першому етапі всі траси прокладаються без врахування взаємного впливу. Після цього виконується аналіз взаємного розташування трас. Траси, які мають максимальну довжину, максимальну кількість перетинів та перегинань, вилучаються, після чого

здійснюється їх повторне трасування з врахуванням положення інших трас. Цей процес повторюється, поки не буде отримано потрібний розв'язок. Алгоритми ітераційного типу застосовують також для корегування різних варіантів трасування.

Слід відмітити, що ітераційні алгоритми не завжди можуть гарантувати 100% трасування. Тому з'єднання, що залишилися, має прокладати конструктор. Загалом трасування виконується в інтерактивному режимі, коли конструктор виконує доробку та корегування з'єднань за екраном монітора.

Контрольні запитання

1. Які основні задачі необхідно розв'язати при конструкторському проектуванні?
2. Які групи задач охоплює поняття комутаційно-монтажного проектування?
3. Назвіть основні вимоги до процесу компоновки конструктивних вузлів.
4. Які критерії оцінки якості розташування елементів набули практичного значення?
5. Дайте порівняльну характеристику основних алгоритмів розташування елементів.
6. За якими критеріями оцінюють якість виконання друкованого монтажу?
7. Назвіть основні задачі, що мають бути розв'язані в процесі проведення монтажних трас.
8. В чому полягає сутність алгоритму Прима?
9. Дайте характеристику хвильового алгоритму трасування з'єднань.
10. Дайте характеристику променевого алгоритму трасування з'єднань.
11. В чому полягають переваги та недоліки евристичного алгоритму трасування з'єднань?
12. Дайте характеристику ітераційнійним алгоритмам трасування з'єднань.

ЛАБОРАТОРНИЙ ПРАКТИКУМ

Лабораторні роботи з дисципліни «Основи автоматизованого проектування складних об'єктів і систем» мають орієнтацію на моделювання за допомогою сучасних пакетів прикладних програм функціонування елементів і схем обчислювальної техніки, що дає можливість освоїти сучасний інструментарій проектування. Звіт з лабораторної роботи повинен включати титульний лист (зразок приведено в додатку А), мету роботи, схеми, що досліджуються, аналіз тестових наборів для проведення досліджень, результати досліджень (у вигляді таблиць, часових діаграм і т.п.), аналіз результатів досліджень і висновки по роботі.

Лабораторна робота № 1

ФОРМУВАННЯ БІБЛІОТЕКИ БАЗОВИХ ЕЛЕМЕНТІВ

Мета роботи: ознайомлення з пакетом програм PCAD, вивчення програми (PCCAPS) графічного вводу і оволодіння методами формування бібліотеки базових елементів, отримання навиків в створенні найпростіших інтегральних елементів на персональному комп'ютері.

Порядок виконання роботи

1. Для нормального функціонування програми необхідно:

ввійти в систему PCAD, ввійти в свій розділ, вказаний викладачем, і загрузити програму PCCAPS (pccaps або pccaps-r). При цьому екран дисплея форматується, на ньому з'являється графічний курсор "Хрест", праворуч від головної зони показу з'являється меню команд.

2. Встановити режим графічного редактора SYMB. Команда SYMB встановлює режим введення/редагування графічних зображень компонентів (символів) принципової схеми; при цьому меню команд зафарбовано в червоний колір.

Зона меню команд містить список основних команд і підкоманд. Щоб вибрати команду з цього меню необхідно помітити курсором ім'я команди (в зоні меню команд курсор має форму прямокутників) і натиснути кнопку 1 маніпулятора "миша" або клавішу "пропуск" на клавіатурі. Якщо у вибраній команді є підкоманди, вони висвічуються жовтим кольором в середній частині лівої колонки зони меню команд. Після вибору підкоманди вона активізується, в рядку станів з'являються параметри, а в рядку повідомлень – інформація про подальші дії.

3. Встановити необхідні розміри зображення на екрані.

При створенні символів мікросхем зручно вибрати крок координатної сітки 5мм, що в метричній системі відповідає масштабу 50:50. При нанесенні текстових написів зручно вибирати більш мілкий масштаб, наприклад 10:10 (крок сітки 1мм). Для змінення масштабу активізується одна з команд DRAW, курсор переводиться в рядок станів, відмічається позиція масштабу X:Y і в відповідь запит

Enter X qrid size:

Enter Y qrid size:

вводяться нові значення масштабу по кожній з осей координат.

4. За допомогою команди VLYR активізувати наступні шари:

GATE ABL A

PRINNUM ABL

PRINNUM ABL

PINCOM ABL

REFDES ABL

ATTR ABL

DEVICE ABL

Інші шари повинні знаходитись в стані OFF (невидимі). Вийти з команди VLYR, вибравши команду QUIT.

5. Введення рисунка контуру елемента.

Він виконується за командою DRAW, що має такі підкоманди:

LINE – введення відрізків прямих ліній;

RECT – введення контура прямокутника;

FRECT – введення прямокутника, залитого фарбою;

CRIC – введення кола;

ARC – введення дуги кола з заданим центром і радіусом.

В рядку станів встановлюються параметри:

GATE – ім'я шару, в який заноситься контур компонента;

SOULD – проведення суцільної лінії;

ORTH – проведення перпендикулярних відрізків ліній (при необхідності проведення ліній під довільними кутами встановлюється параметр ANGL або 450D).

W – ширина лінії, кола, дуги, прямокутника і схеми;

S – вмикання сітки (символ S зафарбований в зелений колір);

G – параметр керування пересуванням курсору, служить для встановлення вільного або дискретного пересування курсору в залежності від його кольору (символ G зафарбований в зелений колір).

Нарисувати контур елемента, що заданий за варіантом, командою

DRAW/LINE (DRAW/RECT), а також вхідні і вихідні лінії.

На підказку системи:

Select start point .....

курсор встановлюється в початкову точку і натискається кнопка 1 маніпулятора "миша", після чого з'являється повідомлення про необхідність вибору наступної точки:

Select next point .....

При переміщенні курсору між ними і початковою точкою з'являється лінія білого кольору, що "розтягується". Після вибору наступної точки натисканням кнопки 1 колір лінії змінюється і можна провести наступний відрізок лінії аналогічним чином. Для усунення тільки що намальованого відрізка (поки він зображений білим кольором) потрібно курсор помістити вздовж тієї ділянки, яка усувається і натиснути кнопку 2. Викреслювання ломаної лінії завершується натисканням кнопки 2 маніпулятора "миша".

DRAW/CIRC – рисувати коло (позначення інверсії) на підказку системи:

Circuit point.....

вибрати центр кола, на підказку системи:

Point on circumference.....

вибрати точку на прямокутнику і нарисувати необхідним радіусом коло.

Для виправлення або усунення контура, цифр і інших позначень використовується команда DEL.

Команди DEL знищують об'єкт або групу об'єктів в схемі.

Це наступні команди:

IDEN – усунути з підтвердженням;

WIN – усунути кола, що вміщенні в "вікно";

UNDO – відновить останній об'єкт, усунутий командою DEL (без використання підменю). Ця команда неефективна після команд DEL/IDEN і DEL/WIN.

6. Введення текстових позначень.

Воно втілюється по команді DRAW/TEXT, перед виконанням якої в рядку станів потрібно встановити параметри:

DEVICE – ім'я шару;

SIZ:25 – висота тексту (можна змінити на будь-яку);

LCF – параметри розміщення тексту;

M – червоний;

S,G – зелений:

На підказку системи:

Select text location. (Attributes OK?).....

курсор підводиться до вибраної точки на полі креслення і натискається кнопка 1 маніпулятора "миша", після чого в відповідь на запит

Tipe in text.....

набирається на клавіатурі потрібний текст і на завершення натискається кнопка 1 або <Enter>.

Текст автоматично розміститься на вказаному місці.

7. Позначення контактів компонентів.

По команді ENTR/PIN в рядку станів встановлюється шар PICON і для кожного компонента вказується його тип (IN – вхід, OUT – вихід, 10 – двонаправлений контакт і т.ін.). Місце розташування контактів відмічається на кресленні в відповідь на підказку:

Select pin locatin.....

Курсор підводиться до кінця виводу і натискається кнопка 1, після чого вибрана для контакту точка позначається на екрані хрестиком синього кольору (колір шару PICON). Порядок, в якому позначаються контакти, має значення для компонентів, якщо припускається моделювання пристроїв за допомогою програми PLOGS або PSPICE. Після утворення контакту з'являється наступна підказка:

Select pin name location.(Attrib. OK).....

Вибравши місце для імені контакту, потрібно натиснути кнопку 1 або кнопку 2 і після запиту:

Entr pin name.....

набрати на клавіатурі ім'я контакту і потім натиснути кнопку 1. Якщо при відповіді на попередній запит була натиснута кнопка 2, то ім'я контакту запам'ятовується в базі даних без відображень його на екрані (звичайно ім'я контакту виводити на креслення не потрібно), а при натисканні кноп-

ки 1 – ім'я контакту на екрані відтворюється. Після введення імені контакту він позначається колом синього кольору з хрестиком всередині.

Таким же чином під'єднати і інші вхідні виводи, якщо вони є.

Змінити на лінії статусу (Pin Tipe:) тип виводу OUT і ввести, аналогічно вхідному виводу вхідні з відповідним ім'ям.

8. Призначити прив'язку (задання ключової точки).

В якості прив'язки призначається лівий ніжній вивід. За командою ENTR/ORG позначається ключова точка і на підказку системи:

Select the original.....

позначається вхідний нижній вивід. Після чого потрібно натиснути кнопку 1. Вона позначається на кресленні білим колом.

9. Ведення інформації про упаковку компонента.

Інформація про упаковку компонента вводиться за командою SCMD/PNLC. Тут необхідно на початку виконання команди SCMD/PNLC встановити REFDES шар активним.

SIN:20 – висота літер;

LCF – параметри розміщення тексту.

На підказку системи:

Enter gates per package.....

ввести кількість екземплярів в корпусі.

Встановити параметри:

FINNUM – активний шар;

SIN:20 – розмір символів;

LBF – для лівих контактів;

RBF – для правих контактів.

На підказку системи:

Select loo for ref designstor.....

встановити квадрат з курсором на місце, де необхідно провести нумерацію ніжок мікросхем і натиснути кнопку 1.

На підказку системи:

Select loo for pin namber…..

вибрати місце для інших номерів виводів, кожний раз натискуючи кнопку 1 маніпулятора “миша”.

Потім проставляються номери контактів компонента, починаючи з першої секції. позначеної символом А:

Enter package pin number <ім’я контакту>:

Gate assigted to section A.

Вивід контакту, що нумерується, зафарбовується в білий колір. Після закінчення нумерації контактів першої секції аналогічно за запитами програми нумеруються контакти інших секцій, що позначені символами B, C, D, … (В режимі SYMB номери контактів, що набираються на клавіатурі, заносяться в базу даних, але на екрані не виводяться).

10. Ідентифікація типу ID компонента.

Тип компонента встановлюється командою SCMD/SCAT за запитом

Symbol Old type = 255. New type =

Для всіх компонентів, що не є примітивами, тип компонента повинен бути визначений рівним 256.

11. Введення атрибутів компонентів.

Виконується командою ATTR/ACOM за запитами:

Select location.(Text attributes OK ?) …

Type in attribute spec

Атрибут (додаткова інформація) складається з двох частин: ключового слова і значення, розділених знаком рівності “=”. Ключове слово повинно починатись з літери і мати довжину до 7 символів. Значення атрибута - це послідовність чисел або текстових змінних, розділених комами.

Цифрові схеми мають тип атрибута:

PCL = < текст атрибута>.

З допомогою атрибута PCL задаються затримки сигналів і навантажувальна властивість мікросхем, що використовуються при логічному моделюванні з допомогою програми PCLOGS, наприклад, PCL = 5,5. Всім компонентам стандартних бібліотек присвоєні необхідні атрибути.

12. Запис графічного зображення компонента в бібліотеку.

Запис на диск проводиться за командою FILE/SAVE. В відповідь на запит:

Enter file name

слід ввести ім'я файлу (без розширення .SYM, яке встановлюється по умовчанню в режимі .SYMB). Це файл заноситься в каталог, вказаний при настройці конфігурації програми PCCAPS. Для занесення файла в інший каталог слід ввести повне ім'я файлу, що містить шлях до цього каталога.

13. Вивести креслення мікросхем на принтер (див. додаток).

14. Закінчення роботи з графічним редактором .

Вибрати команду SYS/QUIT – вийти з меню команд в режим встановлення загальної програми PCCAPS. Знайти в каталозі і в бібліотеці свого розділу записані файли з необхідним розширенням.

15. Ввімкнути і підготувати до роботи принтер.

16. Викликати програму PCPRINT безпосередньо командою >Pcprint і натиснути клавішу Enter. Креслення мікросхеми зберегти для звіту з лабораторної роботи, а також для лабораторної роботи №2.

Зміст звіту

Звіт повинен містити:

1. Мету виконання лабораторної роботи.
2. Короткі відомості про систему проектування PCAD.
3. Структурну схему моделювання цифрових пристроїв.
4. Загальні принципи роботи з графічним редактором.
5. Функціональне позначення елемента згідно з заданим варіантом.

6. Порядок виконання лабораторної роботи і призначення основних етапів проектування.
7. Висновок.

Контрольні питання

1. Функціональні можливості системи проектування PCAD.
2. Розповісти, як відбувається обмін інформацією між окремими модулями в структурній схемі моделювання цифрових пристроїв.
3. На які зони розбивається екран дисплея і їх призначення?
4. Структура шарів креслення і їх призначення.
5. Команди рядка стану і для чого вони використовуються.
6. Що містить зона меню команд і підкоманд?
7. Які команди і підкоманди використовуються при введенні рисунка контура елемента?
8. Яка команда використовується при введенні текстових позначень і які параметри встановлюються в рядку станів?
9. Які команди використовуються при позначенні контактів компонентів і при введенні інформації про упаковку компонента?
10. Яка команда використовується при введенні атрибутів компонента і з яких частин вона складається?

Лабораторна робота №2

СТВОРЕННЯ ТЕКСТОВОГО ФАЙЛУ З ФУНКЦІОНАЛЬНИМ ОПИСОМ НЕСТАНДАРТНОЇ ЦИФРОВОЇ МІКРОСХЕМИ ТА ЇЇ МОДЕЛЮВАННЯ

Мета роботи: освоєння пакета програм PCAD для логічного моделювання, створення текстового файлу з функціональним описанням логічної моделі нестандартного компонента цифрового пристрою.

Порядок виконання роботи

1. Скласти текстовий файл з розширенням .PML і командний файл з розширенням .CMD в редакторі Norton або знайти в каталозі, скопіювати його і записати під ім'ям свого робочого файлу.

2. Викликати з свого розділу функціональні позначення схеми, створенні в лабораторній роботі №1 командою:

pccaps-r

3. Перейти в режим SYMB. Командою FILE/LOAD викликати компонент на екран.

4. Задати командою SCMD/SCAT тип компонента ID=100 (замість 256) і командою ATTR/ACOM атрибут:

MDL = <ім'я файлу>.MDL

5. Записати файл <ім'я файлу>.SYM. Командою FILE/SAVE вказати ім'я створеного символічного зображення мікросхеми (примітива). Очистити екран командою FILE/ZAP.

6. Перейти в режим DETL.

7. Командою VLYR встановити статус ABL таким шарам:

WIRES ABL A
CATE ABL
FINCOM ABL
REFDES ABL
ATTR ABL
SDOT ABL
NETNAM ABL
CMPNAM ABL

Іншим шарам встановити статус OFF. Вийти з режиму команди VLYR, вибравши QUIT. Командою ENTR/COMP вивести на екран зображення компонента з бібліотеки (ім'я бібліотеки відповідає вашому розділу).

8. З'єднати входи і виходи схеми ланцюга командою ENTR/WIRE.

Командою NAME/NET позначити виводи мікросхеми і присвоїти їм ті ж імена, які використані в текстовому файлі з її функціональним описом.

9. Записати файл <ім'я файлу>.SCH .

Подальша робота з програмами можлива в інтерактивному або пакетному режимі. В інтерактивному режимі користувач вводить по черзі команди та спостерігає результати їх виконання на екрані дисплея. Пакетний режим дозволяє послідовно завантажити декілька програм системи проектування PCAD. Для цього за допомогою текстового редактора створюється файл з розширенням .BAT, який заноситься командні строки виклику послідовності програм. Наприклад, файл пакетної обробки:

```
pcmodel %1.pml
pcnodes %1.scr
presim %1.nlt
pclogs %1.cmd
```

буде послідовно викликати програми PCMODEL, PCNODES, PRESIM, PCLOGS. В лабораторній роботі пакетний файл star0.bat.

10. Завантажити файл пакетної обробки та ім'я файлу мікросхеми командою:

<ім'я пакетного файлу> <ім'я файлу схеми>.

Ім'я файла мікросхеми брати без розширення.

11. Натиснути клавішу <Enter>. Йде виконання програм. Прослідкувати вірність виконування програм.

12. В програмі PCLOGS в нашому випадку результати представлені у вигляді часових діаграм. За часовими діаграмами прослідкувати вірність роботи вашої схеми.

13. Для виходу з програми необхідно подати команду EXIT.

14. Видати на друк часову діаграму (див. додаток).

15. Підготувати до роботи пристрій друкування.

16. Завантажити програму POSTSIM і файл plot.cmd командою POSTSIM plot.cmd і отримати часову діаграму. Зберегти її для звіту з лабораторної роботи.

17. Вийти з програми POSTSIM командою EXIT A.

Зміст звіту

1. Короткі теоретичні відомості, що використовують програми PCAD.
2. Алгоритм виконання лабораторної роботи.
3. Таблиця істинності і синтезу цифрової мікросхеми.
4. Текстовий файл.
5. Командний файл для програми PCLOGS.
6. Функціональне позначення мікросхеми.
7. Часові діаграми функціонування мікросхеми.
8. Аналіз одержаних результатів.
9. Висновки по роботі.

Контрольні питання

1. Пояснити принцип складання файлу функціонального опису цифрової мікросхеми.
2. Пояснити роботу мікросхеми за отриманими часовими діаграмами.
3. Яка інформація записується в кожний шар (згідно з завданням)?
4. Які команди використовуються при побудові зображення електричних зв'язків?
5. Призначення програм, що використовуються в файлі пакетної обробки.
6. Поясніть командний файл для програми.
7. Призначення програми POSTSIM.

ПОБУДОВА І ДОСЛІДЖЕННЯ ВБУДОВАНИХ МОДЕЛЕЙ ТИПОВИХ КОМПОНЕНТІВ В СИСТЕМІ PCAD

Мета роботи: Вивчення методів побудови типових компонентів і їх логічне моделювання.

Порядок виконання роботи

Необхідно побудувати і дослідити схему компонента, одержати часові діаграми, проаналізувати результат моделювання. Для цього:

1. Ввійти в систему PCAD, у свій розділ і завантажити програму PCCAPS.
2. Встановити режим графічного редактора SYMB.
3. Встановити необхідні розміри зображення компонента на екрані дисплея.
4. З допомогою команди VLIR активізувати необхідні шари.
5. Намалювати контур мікросхеми за варіантом згідно з табл.1.
6. Ввести текстові позначення командою DRAW/TEXT.
7. Позначити контакти компонента командою ENTR/PIN згідно з порядком слідування виводів. Вказаний вище порядок слідування виводів необхідно при створенні графічного образу компонента.
8. Присвоїти заданий код індентифікації ID своєму компоненту командою SCMD/SCAT.
9. Ввести атрибути компоненту командою ATTR/ACOM.
10. Записати графічне зображення компонента на диск (одночасно компонент записується у бібліотеку Вашого розділу).
11. Очистити екран командою FILE/ZAP.
12. Вийти в режим DETL.
13. Встановити необхідне середовище проектування командою VLYR.

14. З бібліотеки Вашого розділу визвати компонент і розмістити його на полі креслення.
15. Побудувати зображення електричних зв'язків, тобто з'єднати входи і виходи компонента ланцюга командою ENTR/WIRE і присвоїти їм ті ж імена, що використані у командному файлі для програми PCLOGS.
16. Задати необхідне ім'я компонента командою NAME/COMP (згідно з заданим варіантом, наприклад, NOR5).
17. Записати в файл <ім'я файлу>.sch.
18. Вийти з програми PCCAPS.
19. Записати командний файл для програми PCLOGS.
20. Викликати необхідний файл пакетної обробки та ім'я файлу мікросхеми командою:

<ім'я пакетного файлу> <ім'я файлу мікросхеми>.

21. За часовими діаграмами перевірити вірність функціонування Вашої мікросхеми.
22. Видати на друк часову діаграму і графічне зображення компонента та зберегти їх для звіту.

Зміст звіту

1. Мета роботи.
2. Короткі теоретичні відомості про побудову і дослідження вбудованих моделей типів компонентів.
3. Часові діаграми роботи компонента.
4. Необхідні файли, що використовуються при проектуванні.
5. Аналіз отриманих результатів.
6. Висновки по роботі.

Контрольні запитання

1. В чому полягає сутність побудови вбудованих моделей типових компонентів?

2. Характерні особливості побудови окремих компонентів.
3. Які команди використовуються при побудові графічного зображення компонента?
4. Які програми PCAD використовуються при виконанні лабораторної роботи та їх призначення?
5. Які засоби застосовуються при змінені затримки якого-небудь компонента?
6. Які засоби побудови компонентів можливі при дослідженні складних цифрових мікросхем?

Таблиця 1 - **Варіанти завдань**

Номер варіанту	Найменування мікросхеми
1	Логічне 5І-НІ
2	Логічне 5АБО-НІ
3	Логічне 6І
4	Логічне 6АБО
5	Логічне виключне АБО
6	Логічне виключне АБО-НІ
7	Мультиплексор 4х1
8	D – тригер типу DFFPLSR
9	D – тригер типу DFFPHSR
10	JK – тригер типу JKFFPLSR
11	JK – тригер типу JKFFPHSR
12	JK – тригер типу JKFFNLSR
13	JK – тригер типу JKFFNHSR
14	T – тригер типу TFFNLSR
15	T – тригер типу TFFNHSR

Лабораторна робота №4

ПОБУДОВА ПРИНЦИПОВИХ СХЕМ ЦИФРОВИХ ПРИСТРОЇВ ТА ЇХ МОДЕЛЮВАННЯ

Мета роботи: Вивчення основних методів побудови електричних схем цифрових пристроїв з використанням бібліотеки символів в підсистемі PCCAPS; складання заданих електричних схем і їх логічне моделювання.

Підготовка до роботи

1. Вивчити призначення основних програм PCAD, що використовуються в лабораторній роботі.
2. Визначити свій варіант функції перемикання. Для цього необхідно номер варіанта перевести в двійкову систему числення і записати шість його молодших розрядів у вигляді слова $\mathbf{a_6}$ $\mathbf{a_5}$ $\mathbf{a_4}$ $\mathbf{a_3}$ $\mathbf{a_2}$ $\mathbf{a_1}$. Визначивши значення $\mathbf{a_i}$, поставити їх в таблицю 1.

   Наприклад, якщо номер варіанта 5 (000101), то $\mathbf{a_6}$=0, $\mathbf{a_5}$=0, $\mathbf{a_4}$=0, $\mathbf{a_3}$=1, $\mathbf{a_2}$=0, $\mathbf{a_1}$=1.

   Для заданих функцій відомими методами отримайте логічне рівняння, що описує роботу схеми. Вибрати операторні схеми, які забезпечують отримання комбінаційної схеми з максимальною швидкодією і з мінімальним числом вузлових корпусів. Оператори подання функції повинні бути реалізовані на елементах, що знаходяться в бібліотеці <LIB>, які задані в табл.2.
3. Скласти принципову схему за отриманими логічними рівняннями.
4. Створити командний файл з розширенням .CMD.

Порядок виконання роботи.

1. Побудувати і дослідити отриману принципову схему в системі, отримати часові діаграми і табличне значення Вашої функції перемикання.

2. Проаналізувати результати моделювання. В випадку негативного результату повторити процедуру отримання логічного рівняння.

3. Записати командний файл. Для цього необхідно натиснути клавіші <Sift F4> і на запит системи:

Edit the file:

набрати з клавіатури <ім'я файлу>.cmd і натиснути клавішу <Enter>. Тут необхідно набрати зміст командного файлу (див. приклад командного файлу). Записати командний файл (натиснувши клавішу <F2>).

4. Ввійти в систему PCAD, в свій розділ в режимі DETL.

Таблиця 1 - **Варіанти функції перемикання**

X4	X3	X2	X1	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	$\mathbf{a_1}$
0	0	1	1	1
0	1	0	0	$\mathbf{a_2}$
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	$\mathbf{a_3}$
1	0	0	1	$\mathbf{a_4}$
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	$\mathbf{a_5}$
1	1	1	1	$\mathbf{a_6}$

Таблиця 2 – **Варіанти елементів**

Позначення мікросхеми	Тип елементів	Число елементів в корпусі	Час затримки сигналу
1551a1	4І-НІ	2	20
1551a2	8І-НІ	1	22
1551a3	2І-НІ	4	22
1551a4	3І-НІ	3	22
1551a6	4І-НІ	2	22
1551n1	НІ	6	22
1551i1	2І	4	27
1551i3	3І	3	20
1551e1	2АБО-НІ	4	22
1551e4	3АБО-НІ	3	15
1551l1	2АБО	4	22

5. Встановити середовище проектування командою.

Параметри шарів повинні бути:

WIRES ABL A

GATE ABL

PINCOM ABL

SDOT ABL

NETNAM ABL

CMPNAM ABL

ATTR ABL

REFDES ABL

Інші шари знаходяться в стані OFF.

6. З бібліотеки викликати необхідні компоненти і розмістити їх на полі креслення в відповідності з вашою принциповою схемою. На полі креслення компоненти розміщуються по команді ENTR/COMP в шарі GATE. Спочатку у відповідь на запит:

Comp file name G/T-scales (F1 for list)

необхідно набрати ім'я файлу, якій містить компоненти або натиснути клавішу <F1>. В меню висвічується:

Level 1
lib\

Далі потрібно натиснути ліву кнопку "миші" два рази. Вибрати курсором необхідний елемент і натиснути ліву кнопку "миші". На запит:

Select loc to place comp (Oriention OK?)

необхідно прямокутником, що визначає розмір компонента вказати розміщення компонента на екрані (лівою кнопкою "миші"). Зображення компонента можна помістити в декількох місцях на полі креслення, кожний раз підводячи його до нового місця і фіксуючи натисненням лівої кнопки "миші". Повторити вище вказані операції для інших компонентів. Для виходу з команди ENTR/COMP необхідно натиснути праву кнопку"миші" два рази.

6. Побудувати зображення електричних кіл. Для того, щоб з'єднати компоненти в відповідності з принциповою схемою, необхідно подати команду ENTR/WIRE. Після активізації цієї команди в рядку станів встановлюються параметри:

WIRES – шар кіл, що встановлюються автоматично;

ORTH:O – товщина ліній;

L – режим захвату (символ L зеленого кольору).

Далі за запитом системи:

Select start point …

курсор підводять до початкової точки кола, яка фіксується натисканням лівої кнопки "миші", після чого за запитом:

Select next point …

курсор підводять до наступної точки, натискають ліву кнопку "миші" і потім аналогічно проводять наступний відрізок провідника в тому ж напрямку або під кутом 90^0. Введення кола закінчується натисканням правої

кнопки "миші". При з'єднанні контактів компонентів незакінчене коло до натискання правої кнопки "миші" зафарбовано в білий колір, а після її натискання – в зелений.

При введенні на креслення зображення компонентів їх контакти позначені хрестиком; після підключення до них кіл хрестики зникають, що свідчить про наявність контакту.

В точці перетину кіл електричне з'єднання автоматично не відбувається. Для отримання електричного з'єднання спочатку зображується Т-подібний перетин, після чого у відповідь на запит системи про з'єднання двох відрізків кіл:

Merge the nets? Yes, No

слід вибрати відповідь Yes; місце з'єднання відмічається як точка пайки. При необхідності з точкою пайки можна з'єднати ще одне або декілька кіл.

Для редагування розташування компонентів і кіл призначенні різні команди. Паралельний зсув компонентів здійснюється командою MOVE; при цьому з'єднувальні кола розсовуються або стискаються, набуваючи зламів, але не рвуться. Кола редагуються командами EDIT.

8. Присвоїти імена колам та компонентам. Імена присвоюються командою NAME. Імена кіл необхідні для посилань при моделюванні, а також при складанні списку з'єднань схеми. Ім'ям кола може бути довільна послідовність латинських літер та цифр.

Ім'я кола вводиться командою NAME\NET(при цьому повинен бути активізований шар NATNAM).

На запит системи:

Select a net …

курсором вибрати коло, якому надаємо ім'я (воно яскраво висвічується).

На запит системи:

Enter net name:

набрати на клавіатурі ім'я кола і натиснути клавішу <Enter>.

На запит системи:

Name = <ім'я кола> Select location …

курсором помітити місце його розташування на кресленні. Для виходу натиснути праву кнопку "миші".

Ім'я компонентів вводяться командою NAME/COMP в шарі CMPNAM. Спочатку за запитом:

Select a component …

курсором вибрати компонент, який після цього яскраво висвічується. Далі на підказку системи:

Enter component name:

з клавіатури ввести ім'я компонента (наприклад D2.1). Місце розташування цього позначення на кресленні вказати за запитом:

Name = <ім'я> Select location …

Ім'я компонента на кресленні фіксується натисканням правої кнопки "миші".

9. Ввести атрибути схеми командою ATTR/ACOM і за запитом:

Select a component …

курсором вибрати компонент й натиснути ліву кнопку "миші". Даний компонент яскраво висвічується. На підказку системи:

Select location. (Text attributes OK?)…

знайти курсором місце (біля компонента) для задання атрибутів. Натиснути ліву кнопку "миші".

На підказку системи:

Tape in attribute spec.

з клавіатури ввести (наприклад):

PCL=5,5

і натиснути клавішу <Enter>.

Таким же чином ввести атрибути і для компонентів, що залишилися.

Якщо компонент введений з бібліотеки з позначенням MODEL, то необхідно подати команду ATTR/SCHG і на підказку:

Select one attribute ...

курсор перемістити до надпису MODEL та натиснути ліву кнопку "миші"; з'явиться:

PCL=MODEL.

На підказку:

Enter new value

набрати на клавіатурі часові затримки й натиснути клавішу <Enter>.

10. Записати графічне зображення схеми. Запис в файл з розширенням .SCH здійснюється командою FILE/SAVE. У відповідь на запит:

Enter file name

вводиться ім'я файлу (без розширення .SCH, яке встановлюється автоматично в режимі DETL). Цей файл заноситься в поточний підкаталог.

11. Створити вихідний файл для програми PCPRINT з розширенням .PLT для виведення принципової схеми на друк.

12. Вийти з програми PCCAPS.

13. Провести моделювання принципової схеми та отримати часові діаграми її роботи. Для цього необхідно запустити пакетний файл з розширенням .BAT, в який заносяться командні рядки виклику послідовності програм. Наприклад, файл пакетної обробки START.BAT:

```
PONODES    %1.SCH
PRESIM     %1.NTL
PCLOGS     %1.CMD
```

Ввести команду start.bat <ім'я файлу>. Ім'я файлу принципової схеми необхідно вводити без розширення . SCH.

14. Провести аналіз роботи Вашої схеми за часовою діаграмою.

15. Видати на друк часову діаграму.

Зміст звіту

1. Мета роботи.
2. Алгоритм виконання лабораторної роботи.
3. Короткі теоретичні відомості, що необхідні для виконання завдання, всі схеми, таблиці, часові діаграми, що були отриманні при виконанні лабораторної роботи.
4. Короткий опис програм PCAD, що використовувались.
5. Необхідні файли, що використовувались при моделюванні схеми.
6. Аналіз отриманих результатів.
7. Висновки по роботі.

Контрольні запитання

1. В чому полягає сутність проблеми мінімізації функцій перемикання?
2. В чому суттєвість задач аналізу та синтезу комбінаційних схем?
3. Як визначають складність і швидкодію комбінаційних схем?
4. Пояснити роботу синтезованої схеми?
5. Яка інформація записується в кожному шарі з Вашої схеми?
6. Які команди використовуються при побудові електричних зв'язків?
7. Які команди використовуються при заданні атрибутів компонентів?
8. Які програми PCAD використовуються при виконанні лабораторної роботи і їх призначення?
9. Поясніть кожну команду файлу з розширенням .CMD для програми PCLOGS?
10. Поясніть кожну команду файлу з розширенням .CMD для програми POSTSIM?

Лабораторна робота №5

РЕАЛІЗАЦІЯ ЛОГІЧНИХ ФУНКЦІЙ З ВИКОРИСТАННЯМ ФУНКЦІОНАЛЬНИХ БЛОКІВ І МОДЕЛЮВАННЯ ЇХ В СИСТЕМІ PCAD

Мета роботи: Вивчити методи проектування комбінаційних функціональних вузлів. На основі булевого виразу сконструювати принципову схему на логічних елементах і мультиплексорі бібліотеки PCAD.

Підготовка до роботи

1. Виконати синтез логічного пристрою на основі мультиплексора 155КП2 згідно зі своїм варіантом і побудувати функціональну схему.
2. Для побудови схеми і отримання часових діаграм ознайомитися з основними командами PCAD.
3. Визначити функцію $f(X_1 \ldots X_5)$.
4. Підготувати командний файл для програми PCLOGS:

   <ім'я файла>.cmd.
5. Підготовити пакетний файл для виконання необхідних програм з розширенням .BAT.

Порядок виконання роботи

1. Ввійти в редактор PCCAPS.
2. Встановити необхідне середовище проектування в режимі DETL.
3. Командою ENTR/COMP з бібліотеки викликати компоненти схеми (155КП2К і інші логічні компоненти) та розмістити на полі форматування в відповідності з принциповою схемою. Кожному компоненту, введеному з бібліотеки, (за винятком 155КП2К) необхідно задати атрибут командою ATTR/ACOM.
4. Побудувати зображення електричних зв'язків.
5. Присвоїти імена електричним зв'язкам.

6. Записати принципову схему на диск.
7. Вивести копію принципової схеми на друк.
8. Вивести на дисплей часові діаграми роботи Вашої схеми, перевірити вірність її функціонування та роздрукувати її.
9. Вийти з програми POSTSIM.

Зміст звіту

1. Мета роботи.
2. Алгоритм виконання лабораторної роботи.
3. Короткі теоретичні відомості, що необхідні для виконання завдання, всі схеми, таблиці, часові діаграми, що були отриманні при виконанні лабораторної роботи.
4. Необхідні файли (командний файл та пакетний файл програм для виконання), що використовувались при моделюванні схеми.
5. Аналіз отриманих результатів.
6. Висновки по роботі.

Контрольні запитання

1. Застосування мультиплексора.
2. Які операції включає в себе синтез логічного пристрою?
3. Поясніть принцип роботи схеми, яку ви синтезуєте.
4. Яка інформація записується в кожному шарі з вашої програми?
5. Які команди використовуються при побудові зображень електричних зв'язків?
6. Які команди меню використовуються при створенні атрибутів елементів?
7. Які програми використовуються при виконанні лабораторної роботи?
8. Призначення програми PCLINK.

Лабораторна робота №6

ПРОЕКТУВАННЯ СХЕМ З ІЄРАРХІЄЮ І ЇХ ЛОГІЧНЕ МОДЕЛЮВАННЯ

Мета роботи: Вивчення основних методів проектування принципових схем з ієрархією програмами PCAD з використанням бібліотеки символів (вбудованих типових компонентів) і моделювання спроектованих схем.

Порядок виконання роботи

1. Створити схему макромоделі.

   Для цього необхідно:

   - ввійти в редактор PCCAPS;
   - встановити середовище проектування, т.т. режим, і встановити командою VLYR необхідний статус шарам;
   - згідно з заданою принциповою схемою макромоделі з бібліотеки вибрати компоненти (чи створити їх) і розмістити їх на полі монітора командою ENTR/COMP;
   - ввести з'єднання схеми командою ENTR/WARE;
   - іменувати компоненти схеми командою NAME/COMP;
   - іменувати коло командою NAME/NET;
   - ввести атрибути кожному компоненту командою ATTR/ACOM, т.т. інформацію про затримку сигналу;
   - запам'ятовувати і очищати робочий файл не потрібно! На цьому ж файлі виконати наступний пункт.

2. Створити символ для схеми макромоделі.

   Схемі необхідно поставити відповідний графічний символ, який надалі буде використаний в якості компонента для створення схем більш високого рівня.

Для створення символу схеми в тому ж робочому файлі необхідно:

- створити графіку умовного графічного визначення схеми (макромоделі) в режимі SYMB (див. лаб.1);
- ввести вхідні (IN) і вихідні (OUT) символи командою ENTR/PIN;
- присвоїти символу ім'я командою DRAW/TEXT;
- встановити тип ID компонента, рівний 256 командою SCMD/SCAT;
- записати файл командою FILE/SAVE: <ім'я файла>.SYM;
- вийти з редактора PCCAPS.

3. Програмою PCNODES добути з макромоделі список електричних зв'язків. Для цього необхідно викликати програму PCNODES і натиснути клавішу <Enter>. На запит

   Database filename: <ім'я файла>.SCH

   ввести з клавіатури <ім'я файла>.SYM і натиснути клавішу <Enter>. При успішному завершені роботи програма видасть повідомлення:

   Completed & save as <ім'я файла>.NLT

   т.т. сформується файл з розширенням .NLT для програми PCLINK. Для виходу з програми натиснути <Esc>.

4. Використати створений символ макромоделі для створення ієрархічної структури Вашої схеми. Для цього необхідно виконати наступне:
   - створити схему (див. варіанти завдань), використовуючи раніше створений символ макромоделі;
   - запам'ятати файл під іншим ім'ям;
   - вийти з програми PCCAPS.
5. Створити командний файл для вашої схеми з розширенням .CMD.
6. Провести моделювання принципової схеми, використовуючи пакетний файл програм:

   pcnodes %1.sch

pclink

presim %1.xnl

pclogs %1.cmd

При виконанні програми PCLINK (після заставки PC-LINK) на підказку ввести інформацію:

Netlist Extension: nlt <Enter>

Work Directory: Current <Enter>

Library Path: Current <Enter>

Netlist Filename: <ім'я файла> <Enter>

Expanded Netlist: <ім'я файла>.xnl <Enter>

Для виходу з програми PCLINK натиснути клавішу <Esc>.

7. Провести аналіз отриманих результатів моделювання. При отриманні негативних результатів роботи схеми її необхідно відредагувати. При вірному функціонуванні схеми модуля перейти до наступного кроку виконання лабораторної роботи.
8. Видати на друк часову діаграму, а також принципову схему ієрархічної структури.

Зміст звіту

1. Мета роботи.
2. Алгоритм виконання лабораторної роботи.
3. Короткі теоретичні відомості, що необхідні для виконання завдання, всі схеми, таблиці, часові діаграми, що були отриманні при виконанні лабораторної роботи.
4. Необхідні файли (командний файл та пакетний файл програм для виконання), що були отримані в процесі експериментального дослідження схем.
5. Аналіз отриманих результатів.
6. Висновки по роботі.

Контрольні запитання

1. Для чого використовується метод ієрархічного проектування принципових схем?
2. Методика проектування принципових схем з ієрархією.
3. Поясніть роботу схеми вашого варіанта.
4. Принцип роботи програми PCLINK.
5. Які команди використовуються при редагуванні принципової схеми?
6. Призначення ідентифікатора ID.
7. Призначення атрибутів схеми.
8. Які існують режими графічного редактора PCCAPS і їх призначення?
9. Які команди використовуються для переходу по рівням ієрархії?

Лабораторна робота №7

ПРОЕКТУВАННЯ ТА ДОСЛІДЖЕННЯ ІНТЕГРАЛЬНИХ ЗАПАМ'ЯТОВУВАЛЬНИХ ПРИСТРОЇВ У СИСТЕМІ PCAD ТА ЇХ МОДЕЛЮВАННЯ

Мета роботи: Вивчення головних методів побудови запам'ятовувальних пристроїв (ЗП), набуття навиків їх практичного застосування і логічне моделювання ОЗП та ПЗП у системі PCAD.

Підготовка до роботи

1. Вивчити принцип роботи ОЗП і ПЗП.
2. Вивчити особливості створення графічного образу ОЗП і ПЗП в системі PCAD.
3. Нарисувати функціональне позначення ОЗП і ПЗП згідно з заданим варіантом (див. табл.2).
4. Визначити об'єм пам'яті заданого ПЗП і ОЗП в бітах.

5. Скласти командний файл для PCLOGS з розширенням 'CMD' відповідно враховуючи завдання на моделювання. У вільні адреси для ОЗП записати і порахувати вхід за кодами D1 …DM. Наприклад:

```
LOAD LAB7.NET
CYCLE 16
MEMLOAD RAM1 HEX /00 05 02 03 04 06 09
GEN [0 0] D1 SO/24 (S1/32 SO/32)
GEN [0 0] D2 SO/24 (S1/16 SO/16)
. . . . . .
GEN [0 0] A1 (SO/32 S1/32)
. . . . . .
SIM 64
LOAD LAB7.NET
CYCLE 16
MEMLOAD ROM1 HEX /01 02 03 04 05 A B C /0A 01 A D
GEN [0 0] A1 (SO/4 S1/4)
. . . . . .
GEN [0 0] CS (SO/1 S1/2 SO/1)
. . . . . .
```

Порядок виконання роботи

1. Ввійти в систему PCAD і в свій розділ.
2. Створити графічний образ компонента (режим SYMB) згідно з таблицею 1.

   Встановити необхідні шари (див. лабораторну роботу №1), дотримати порядок проходження вхідних (IN), вихідних (OUT) виводів командою ENTR/PIN і код ID командами PCCAPS. Задати необхідний атрибут. Записати компонент в свій розділ <ім'я файлу>, за замовчуванням з розширенням "SYM".

3. Провести логічне моделювання свого пристрою. Для цього необхідно провести підготовчі операції. Ввійти в графічний редактор PCCAPS (в режим DETL) і командою ENTR/COMP вивести з свого розділу графічне зображення створеного компонента. За допомогою команди ULYR активізувати необхідні шари. Створити рисунок принципової схеми, т.т. з'єднати входи і виходи електричного ланцюга командою ENTR/WIRE та присвоїти імена ланцюгам і компоненту відповідно командами NAME/NET і NAME/COMP.
4. Записати графічне зображення схеми в свій розділ.
5. Записати командний файл для програми PCLOGS з розширенням "CMD".
6. Завантажити пакетний файл обробки з програмами PCNODES, PRESIM і PCLOGS.
7. За часовими діаграмами перевірити правильність функціонування створеної схеми ОЗП (ПЗП) в усіх режимах.
8. При отриманні вірогідних результатів роботи схем часові діаграми видати на друк і зберегти їх для звіту.
9. Вийти з системи програм PCAD.

Зміст звіту

1. Мета роботи.
2. Алгоритм виконання лабораторної роботи.
3. Короткі теоретичні відомості про ЗП, що необхідні для виконання завдання, всі схеми, таблиці, часові діаграми, що були отриманні при виконанні лабораторної роботи.
4. Необхідні файли, вхідний файл для програми PCLOGS.
5. Аналіз отриманих результатів.
6. Висновки по роботі.

Контрольні запитання

1. Основні поняття та класифікація ЗП.
2. Оперативні ЗП (ОЗП).
3. Постійні ЗП (ПЗП).
4. Поясніть принцип роботи ОЗП.
5. Поясніть принцип роботи ПЗП.
6. Як проводиться запис інформації в Вашій схемі?
7. Як проводиться читання інформації в Вашій схемі?
8. Характерні особливості при проектуванні та моделюванні ЗП в системі PCAD.
9. Як задаються початкові стани ПЗП та ОЗП?
10. Поясніть зміст команди ініціалізації початкових даних.

Таблиця 1 - **Варіанти завдань**

№ вар.	Кількість адресних входів (N)	Кількість адресних виходів (M)	Код початкового стану
ОЗП			
1	4	4	/00 03 04 05 06 /0A 07 04 08 09 01
2	4	4	/03 04 05 06 0A 0B /0C 07 09 03 02
3	5	4	/04 07 05 06 0C 0D /0A 01 05 08 06
4	5	5	/01 03 02 04 0A 0C /12 07 08 03 02
5	6	4	/03 04 05 06 07 08 /0D 01 02 09 07
6	4	4	/04 01 02 03 05 06 /0A 05 03 03 02
7	8	5	/0A 02 07 08 09 0A /15 07 08 03 01
8	7	5	/04 07 05 03 04 0C /16 04 05 06 02
9	5	5	/05 08 09 0A 0B 0C /18 07 04 05 03
10	6	5	/02 09 07 05 0C 0D /17 04 07 03 04
ПЗП			
1	8	4	/00 01 02 03 04 05 /0A 0A 0B 0C 05
2	5	4	/01 01 03 04 07 09 /11 0A 0B 0C 0D
3	6	4	/05 06 07 08 0C 0D /12 0F 0D 08 09
4	7	4	/02 01 0D 0E 02 07 /13 04 0A 08 01
5	4	4	/04 0A 0B 0C 0D 0F /0A 04 06 07 08
6	6	5	/06 01 0A 0B 02 07 /21 07 09 0A 0B
7	8	5	/02 0A 0C 0B 0D 0F /25 0A 01 05 07
8	6	5	/03 0F 0C 0F 0B 05 /26 09 08 07 0A
9	8	4	/01 0A 0B 0C 0D 0E /15 07 03 02 01
10	8	5	/02 01 0B 05 0F 03 /23 0A 05 0B 06

ДОДАТОК А - Зразок титульного листа звіту про виконання лабораторної роботи

Міністерство освіти та науки України

Вінницький державний технічний університет

Факультет ІТКІ

Кафедра ІС

ЗВІТ

про виконання лабораторної роботи №1

«ФОРМУВАННЯ БІБЛІОТЕКИ БАЗОВИХ ЕЛЕМЕНТІВ»

з дисципліни «Автоматизація проектування складних об'єктів і систем»

Виконав: ст. групи 1ІТП-96 Іванов І.І.

Перевірив

Вінниця – 2001

ВИСНОВКИ

Найбільш значні успіхи автоматизації проектування були отримані в галузях моделювання, аналізу та комп'ютерної графіки. Скромніші здобутки в галузі синтезу, хоча і тут можна відмітити результати в таких напрямках, як параметрична оптимізація об'єктів з неперервними моделями, комбінаторна оптимізація при проектуванні топології друкованих плат, синтез комбінаційних логічних схем.

Першочерговими задачами подальшого розвитку САПР є:

- розробка математичного забезпечення для підвищених розмірностей задач функціонально-логічного та топологічного проектування, що зумовлено ростом ступеня інтеграції мікросхем;
- застосування нових засобів технічного забезпечення, що з'явились внаслідок бурхливого розвитку в галузі обчислювальної техніки.

Розв'язання цих задач потребує зусиль фахівців за такими напрямками:

- створення програмно-методичних комплексів для синтезу структур об'єктів на базі штучного інтелекту;
- істотне підвищення ефективності моделювання й аналізу, оскільки тільки за таких умов можлива автоматизація проектування таких об'єктів та систем, в склад яких входять мільйони компонентів;
- розробка спеціалізованих обчислювальних пристроїв для апаратної реалізації найбільш трудомістких проектних процедур.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

1. *Абрайтис Л.Б., Шейнаускас Р.И., Жилевичус В.А.* Автоматизация проектирования ЭВМ. – М.: Советское радио, 1978. – 272 с.

2. Автоматизация конструирования больших интегральных микросхем/ *А.И.Петренко, П.П.Сыпчук, А.Я.Тетельбаум* и др. – Киев: Вища школа, 1983. – 312 с.

3. Автоматизация конструирования монтажных плат РЭА: Справочник специалиста/ *А.Т.Абрамов, В.Б.Артемов, В.П. Богданов* и др.; Под ред. Л.П.Рябова. – М.: Радио и связь., 1986. – 192 с.

4. Автоматизация проектирования вычислительных систем. Языки, моделирование и базы данных/Под ред. *М.Брейера.*– М.: Мир, 1979.– 464 с.

5. Автоматизация проектирования цифровых устройств/ *Баранов С.И., Майоров С.А., Сахаров Ю.П., Селютин В.А.* – Л.: Судостроение, 1979. – 261 с.

6. Автоматизированное проектирование цифровых устройств/ *Бадулин С.С.,Барнаулов Ю.М.,Бердышев В.А.* и др.; Под ред.*С.С. Бадулина.* – М.: Радио и связь, 1981. – 240 с.

7. *Арсеньев Ю.Н., Журавлев В.М.* Проектирование систем логического управления на микропроцессорных средствах: Учебное пособие для вузов. – М.: Высшая школа, 1991. – 318 с.

8. *Зиглер К.* Методы проектирования программных систем: Пер. с англ. - М.: Мир, 1985. – 328 с.

9. *Ильин В.Н., Коган В.Л.* Разработка и применение программ автоматизации схемотехнического проектирования. – М. Радио и связь, 1984. – 368 с.

10. Интеллектуальные системы автоматизированного проектирования больших и сверхбольших интегральных микросхем/ *В.А.Мищенко, Л.М.Городецкий, Л.И.Гурский* и др.; Под ред. *В.А.Мищенко.* – М.: Радио и связь, 1988. – 272 с.

11. *Карбери П.Р.* Персональные компьютеры в автоматизированном проектировании: Пер. с англ. – М.: Машиностроение, 1989. – 142 с.

12. *Норенков И.П., Маничев В.Б.* Основы теории и проектирования САПР: Учебник для втузов. – М.: Высшая школа, 1990. – 335 с.

13. *Норенков И.П., Маничев В.Б.* Системы автоматизированного проектирования электронной и вычислительной аппаратуры: Учебное пособие для вузов. – М.: Высшая школа, 1983. – 272 с.

14. *Петренко А.И., Тетельбаум А.Я.* Формальное конструирование электронно-вычислительной аппаратуры. – М.: Советское радио, 1979. – 256 с.

15. *Питерсон Дж.* Теория сетей Петри и моделирование систем: Пер. с англ. – М.: Мир, 1984. – 264 с.

16. Построение экспертных систем: Пер. с англ. / Под ред. *Ф. Хейеса-Рота, Д.Уотермана, Д.Лената.* – М.: Мир, 1987. – 441 с.

17. Проектирование автоматизированное. Термины и определения. ГОСТ 22487 – 77.

18. *Рафикузаман М.* Микропроцессоры и машинное проектирование микропроцессорных систем: В 2-х кн.: Пер. с англ. – М.: Мир. 1988.

19. Системы автоматизированного проектирования: Учебное пособие для втузов: В 9 кн./ Под ред. *И.П.Норенкова.* – М.: Высшая школа, 1986.

20. Системы автоматизированного проектирования в радиоэлектронике: Справочник / *Е.В.Авдеев, А.Т.Еремин, И.П.Норенков, М.И. Песков* / Под ред. *И.П.Норенкова.* – М.: Радио и связь, 1986. – 368 с.

21. *Соловьев В.В.* Проектирование функциональных узлов цифровых систем на программируемых логических устройствах: Учебное пособие для вузов. – Мн.: ПКООО Бестпринт. 1996. – 252 с.

22. Теория и методы автоматизации проектирования вычислительных систем / Под ред. *М.Брейера* – М.: Мир, 1977.–282 с.

23. *Чжен Г., Мэннинг Е., Метц Г.* Диагностика отказов цифровых вычислительных систем: Пер. с англ.- М.: Мир, 1972. – 486 с.

Навчальне видання

Роїк О.М., Савчук Т.О., Ткаченко О.М

ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СКЛАДНИХ ОБ'ЄКТІВ І СИСТЕМ

Навчальний посібник

Оригінал-макет підготовлено авторами

Редактор С.А.Малішевська

Підписано до друку 6.06.2001р
Формат 29,7х42 1/4 Гарнітура Times New Roman
Друк різографічний Ум.друк.арк. 5.11
Тираж 75 прим.
Зам.№ 2001-112

Віддруковано в комп'ютерному інформаційно-видавничому центрі Вінницького державного технічного університету
21021, м.Вінниця, Хмельницьке шосе, 95, ВДТУ, ГНК, 9-й поверх
Тел. **(0432) 44-01-59**