# МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

# СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

# ІМЕНІ ВОЛОДИМИРА ДАЛЯ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ

КАФЕДРА ПРОГРАМУВАННЯ ТА МАТЕМАТИКИ

## Пояснювальна записка

**до дипломної роботи**

**\_\_\_\_\_\_\_\_** **бакалавр \_\_\_\_\_\_\_\_**

(освітньо-кваліфікаційний рівень)

**на тему «Програмна реалізація конструктора логічних схем»**

Виконав: студент 4 курсу, групи ІТ-651

напряму підготовки 6.040302 „ Програмна інженерія ”

\_\_\_\_\_\_\_\_\_\_\_\_ Замковець Д.М.

 (підпис)

Керівник,

доцент, к. т.н.\_\_\_\_\_\_\_\_\_ \_ Іванов В.Г.

 (підпис)Рецензент,

Ст. викладач \_\_\_\_\_\_\_\_\_\_\_ Батурін О.І.

 (підпис)

СЄВЄРОДОНЕЦЬК

2019 року

# СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

# ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет інформаційних технологій та електроніки

Кафедра програмування та математики

Освітньо-кваліфікаційний рівень бакалавр

Напрям підготовки 6. 050103 „ Програмна інженерія ”

# Спеціальність 7.050103 „ Програмна інженерія ”

|  |
| --- |
| ЗАТВЕРДЖУЮЗавідувач кафедри ПМ, д.т.н., доцент\_\_\_\_\_\_\_\_\_\_\_\_\_Лифар В.О.«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2019 р. |

## З А В Д А Н Н Я

### НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

### ЗАМКОВЕЦЬ ДАНИЛО МИХАЙЛОВИЧ

1. Тема роботи Програмна реалізація конструктора логічних схем

керівник роботи доцент Іванов Віталій Геннадьович

2. Строк подання студентом роботи 06 червня 2019 р.

3. Вихідні дані до роботи

Об'єктом даної роботи є програмна реалізація конструктора логічних схем.

3.1 Літературні джерела:

Проектування НВІС/ Ватамабе М., Асада К., Канн К., Оцуки. Т. - М .: Світ, 1988. - 304 с .

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

 4.1 Вступ

4.2 Аналіз предметної галузі (огляд літератури), з висвітленням наступних питань:

Конструктор логічних схем

 4.3 Основна частина, в якої висвітлити:

Інформаційна модель об’єкту.

Програмну реалізацію моделі.

 4.4 Висновки

 4.5 Перелік використаних джерел

5. Перелік графічного матеріалу немає

6. Дата видачі завдання 02 лютого 2019 року.

#### КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| №з/п | Назва етапів дипломної роботи | Строк виконання етапів роботи | Примітка |
| 1 | Одержання завдання на виконання роботи | 01.02.19 |  |
| 2 | Укладання і погодження з керівником плану і етапів виконання роботи | 20.02.19 |  |
| 3 | Узагальнення даних літературних джерел, укладання розділу «Аналіз предметної галузі» | 1.03.19 |  |
| 4 | Аналіз шляхів виконання завдання. Вибір і погодження з керівником оптимального шляху  | 10.03.19 |  |
| 5 | Проектування інфологічної моделі задачі що реалізується. | 01.04.19 |  |
| 6 | Укладання та тестування програмного продукту | 20.04.19 |  |
| 7 | Укладання, оформлення та погодження пояснювальної записки з керівником | 15.05.19 |  |
| 8 | Здача готової пояснювальної записки на кафедру | 06.06.19 |  |
| 9 | Укладання доповіді і презентації | 10.06.19 |  |

Студент \_\_\_\_\_\_\_\_\_\_\_\_ Замковець Д.М.

(підпис) м

Керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_ Іванов В.Г.

(підпис)

ЛИСТ ПОГОДЖЕННЯ І ОЦІНЮВАННЯ

дипломної роботи студента гр. ІТ-651 Замковець Д.М.

Науковий керівник

Доцент, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_ Іванов В.Г.

Оцінка наукового керівника: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рецензент Ст. викладач каф. ПМ СНУ ім.В.Даля Батурін О.І.

 місто роботи, посада, ПІБ

Оцінка рецензента: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кінцева оцінка за результатами захисту:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Голова ЕК

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лифар В.О.

 підпис

РЕФЕРАТ

 Текст – 86, рис. – 11, літературних джерел – 10

У ході виконання даної дипломної роботи було проведено розробка спеціалізованого програмного забезпечення, збір необхідних початкових даних, використовуваних в програмі;

В процесі виконання роботи була проведена наступна робота:

1. Освоєння принципів розробки алгоритмів побудови логічних схем;

2. Ознайомлення з різними сферами застосування методів побудови логічних схем

3. Розроблена програма яка дозволяє побудувати логічні схеми

 Ключові слова: ЛОГІЧНІ СХЕМИ, DELPHI

ЗМІСТ:

[ВСТУП 13](#_Toc11322949)

[РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД 16](#_Toc11322950)

[1.1 Застосування засобів алгебри логіки для опису функціонування пристроїв комп'ютера 16](#_Toc11322951)

[1.2 Логічні схеми 19](#_Toc11322952)

[1.3 Схемотехнічне моделювання аналого-цифрових пристроїв 24](#_Toc11322953)

[1.4 Синтез логічних схем 30](#_Toc11322954)

[1.5 Проектування друкованих плат 37](#_Toc11322955)

[РОЗДІЛ 2 ПОСТАНОВКА ЗАДАЧІ 42](#_Toc11322956)

[2.1 Характеристика задачі 42](#_Toc11322957)

[2.3.Логические элементы в модели 53](#_Toc11322958)

[РОЗДІЛ 3 ОПИС ПРОГРАМИ 61](#_Toc11322968)

[3.1 Вимоги до программи 61](#_Toc11322969)

[3.2 Проектування інтерфейсу програми 62](#_Toc11322970)

[3.3 Опис застосування програми 67](#_Toc11322971)

[СПИСОК ЛІТЕРАТУРИ 73](#_Toc11322972)

[ДОДАТОК А 74](#_Toc11322973)

# ВСТУП

Актуальність досліджень. Сучасна електронна обчислювальна машина - це складний комплекс пристроїв, що захоплює своїм технологічним досконалістю і різноманітністю фізичних принципів роботи. Різні пристрої машини представляють собою комбінацію 0великого числа елементів-цеглинок. Елементи комп'ютера виконують зберігання, перетворення і передачу двійкових змінних, а також ряд допоміжних функцій: затримку сигналу в часі, формування сигналу з певними фізичними характеристиками і т. П. Для подання та обробки інформації використовуються різні фізичні явища і процеси, наприклад електричний струм або магнітний потік. Наявність або відсутність електричного струму, рівня напруги різної величини або полярності, величини магнітного потоку і т. Д. Розглядаються як сигнали. У технічному відношенні передача і перетворення дискретної інформації реалізується найбільш просто, якщо розрізняти тільки два рівня сигналу, відповідні символам 0 і 1, або поняттям істинності ТАК чи НІ. Різні конструкції елементів можуть давати лише різні технічні реалізації, що гарантують розпізнавання двох таких станів.

Якою б складною не була структура конкретної системи, її можна вивчати і описувати по частинах, виявляючи в ній порівняно невелике число стандартних елементів, що становлять стандартні вузли і блоки. Пристрої машини представляють собою комбінації великого числа елементів-цеглинок. Різноманітність їх, однак, не настільки велика. Кожна машина складається з обмеженої кількості типів елементів, які виконують певні логічні функції, що виражають залежність сигналів на виходах від сигналів на входах цих елементів.

Побудова і аналіз логічних елементів і схем ґрунтуються на застосуванні формального методу математики до області логіки. Подібно до того, як в математиці для вираження відносин використовується мова формул, в математичній логіці та сама мова формул використовується для вираження логічних зв'язків, які існують між судженнями, поняттями, висловлюваннями..

Так само як і для математичної задачі, рішення будь-якої логічної задачі має свою "технологію". Вона складається з найпростіших операцій, схожих на додавання і множення. Прийнято вважати, що значення істинності висловлювання дорівнює одиниці, якщо воно істинне, і дорівнює нулю, якщо воно помилкове. Якщо умовно позначити різні висловлювання буквами A, B і т. Д., То в загальному вигляді запис A = 1 означає, що висловлювання А істинно, а B = 0 означає, що висловлювання B помилково. Те, що значення істинності може приймати два значення - 0 і 1, добре узгоджується з двійковій системою числення і з роботою двохпозиційних елементів, які використовуються в ЕОМ. Наприклад, в ЕОМ істинність висловлювання може бути передана сигналом позитивного електричної напруги, що умовно відповідає сигналу 1, а хибність - сигналом негативного напруги або відсутністю взагалі будь-якого сигналу. Це умовно відповідає сигналу 0. Висловлювання можуть бути простими і складними. З даних висловлювань за допомогою логічних операцій можна отримати нові висловлювання.

 Об’єкт досліджень: Реалізація можливостей збереження логічних схем і використання їх як елементів в інших схемах.

 Предмет досліджень: Побудова демонстраційної програми за допомогою будь-якого середовища візуального програмування, який був би доступний та легкий у використанні користувачу-початківцю.

 Мета дослідження:
Вивчення принципів роботи логічних елементів і їх схем. Створення програми, що моделює роботу логічних елементів.

Завдання дослідження:

а) скласти інформаційну модель;

б) розкрити алгоритми;

в) скласти програму;

 Методологічна та теоретична основа дослідження: структура та методи побудови логічних схем.

 Методи дослідження: програмування за допомогою візуального середовища.

# РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД

# 1.1 Застосування засобів алгебри логіки для опису функціонування пристроїв комп'ютера

Для опису того, як функціонують апаратні засоби комп'ютера дуже зручний математичний апарат алгебри логіки, оскільки основною системою числення в комп'ютері є двійкова, в якій використовуються цифри «1» і «0».

Одні і ті ж пристрої комп'ютера можуть застосовуватися для обробки і зберігання числової інформації, представленої в двійковій системі числення, логічних змінних. На етапі конструювання апаратних засобів алгебра логіки дозволяє значно спростити логічні функції, що описують функціонування схем комп'ютера, і, отже, зменшити число елементарних логічних елементів, з десятків тисяч яких складаються основні вузли комп'ютера.

Дані і команди в пам'яті комп'ютера і в регістрах процесора представляються у вигляді двійкових послідовностей різної структури і довжини.

Існують різні фізичні способи кодування двійкової інформації, але частіше за все одиниця кодується більш високим рівнем напруги, ніж нуль.

У логічній схемі комп'ютера виділяють логічні елементи. Логічний елемент комп'ютера - це частина електронної логічної схеми, яка реалізує елементарну логічну формулу.

Логічними елементами комп'ютерів є електронні схеми «І», «АБО», «НЕ», «І-НЕ», «АБО-НЕ» та інші (звані також вентилями), а також тригер. За допомогою цих схем можна реалізувати будь-яку логічну формулу, яка описує роботу пристроїв комп'ютера. Кожен логічний елемент має своє умовне позначення, яке виражає його логічну формулу, але не вказує на те, яка саме електронна схема в ньому реалізована. Це спрощує запис і розуміння складних логічних схем. Схема «І»реалізує кон'юнкцію двох або більше логічних значень. Умовне позначення на структурних схемах схеми «І» з двома входами представлено на рис. 1.1.



Рис. 1.1. схема «І»

На виході схеми «І» значення «1» буде тоді і тільки тоді, коли на всіх входах будуть «1». Коли хоча б на одному вході буде «0», на виході також буде «0».

Операція кон'юнкції на функціональних схемах позначається знаком «&» (читається як «амперсенд»), що є скороченим записом англійського слова and.

Схема «АБО» реалізує диз'юнкцію двох або більше логічних значень. Умовне позначення схеми «АБО» представлено на рис. 1.2.



Рис. 1.2. Схема «АБО»

Значення диз'юнкції одно «1», якщо сума значень операндів більше або дорівнює «1». Коли хоча б на одному вході схеми «АБО» буде «1», на її виході також буде «1». Операція диз'юнкції на функціональних схемах позначається знаком «1».

Схема «НЕ»(Інвертор) реалізує операцію заперечення. Умовне позначення схеми НЕ представлено на рис. 1.3.



Рис. 1.3. Схема «НЕ»

Якщо на вході схеми - «0», то на виході буде «1». Коли на вході - «1», на виході буде «0».

Схема «І-НЕ» складається з елемента «І» і інвертора і здійснює заперечення результату схеми «І». Умовне позначення схеми «І-НЕ» представлено на рисунку 1.4. Таблиця істинності схеми «І-НЕ» - це таблиця 1.1.



Рис. 1.4. Схема «І-НЕ»

Таблиця істинності схеми «І-НЕ»

Таблиця 1.1

|  |  |  |
| --- | --- | --- |
| х | у |  |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Схема «АБО-НЕ» складається з елемента «АБО» і інвертора і здійснює заперечення результату схеми «АБО». Умовне позначення схеми «АБО-НЕ» представлено на рис. 1.5, а таблиця істинності схеми АБО-НЕ - це табл. 1.2.



Рис.1.5. Схема «АБО-НЕ»

Таблиця істинності схеми «АБО-НЕ»

Таблиця 1.2

|  |  |  |
| --- | --- | --- |
| х | у |  |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

#  Логічні схеми

Логічна схема - це схематичне зображення деякого пристрою, що складається з перемикачів і з'єднують їх провідників, а також з входів і виходів, на які подається і з яких знімається електричний сигнал.

Кожен перемикач має тільки два стани: замкнутий і розімкнуте. Перемикача Х поставимо у відповідність логічну змінну х, яка приймає значення 1 в тому і тільки в тому випадку, коли перемикач Х замкнутий і схема проводить струм; якщо ж перемикач розімкнути, то х дорівнює нулю.

Дві схеми називаються рівносильними, якщо через одну з них проходить струм тоді і тільки тоді, коли він проходить через іншу (при одному і тому ж вхідному сигналі). З двох рівносильних схем простіший вважається та схема, функція провідності якої містить меншу кількість логічних операцій або перемикачів. При розгляді перемикальних схем виникають дві основні задачі: синтез і аналіз схеми.

Синтез схеми за заданими умовами її роботи зводиться до наступних трьох етапах:

1. складання функції провідності по таблиці істинності, що відбиває ці умови;
2. спрощення цієї функції;
3. побудови відповідної схеми.

Аналіз схеми зводиться до визначення значень її функції провідності при всіх можливих наборах вхідних в цю функцію змінних та отриманню спрощеної формули.

Логічні схеми обгрунтування вибірки .Пропонується логічна схема для обґрунтування обсягу і складу вибірки. Будь-яке наукове дослідження припускає наявність визна-ленній логіки його побудови, яка призводить до того чи іншого результату. Якісні соціологічні дослідження тут не виняток. З логікою конкретного дослідження можна познакомитися, вивчивши його програму і архів, в якому зібрані протокол з описом процедур збору первинної інформації, аналітичні схеми, польові щоденники, транскрипти, інтерв'ю та ін. Через порівняно низького ступеня формалізації більшості процедур - опис, що і навіщо досліджується, чому обрано даний інструмент, на чому ґрунтуються гіпотези і висновки - зазвичай дається авторами якісного дослідження досить скупо. Особлива лаконічність авторів відзначається в описі логічної схеми побудови моделі вибірки і обґрунтування причин відбору пред-ставлених в якісному дослідженні респондентів. Тому часто буває абсолютно незрозуміло, чому автор зупинився на 5, 10 або 30 респондентів, чому вибрав саме цих інфор-мантів, чому вирішив, що саме їх випадок або їх експертиза буде релевантна цілям і завданням дослідження.

В дослідженнях, де застосовується біографічний метод або метод кейс-стаді, число респондентів і їх характеристики можна обґрунтувати виходячи з логіки унікальності випадку або біографії респондента, одиничних за своєю природою. Наприклад, всі до-ступні для інтерв'ю свідки розкуркулення в даному селі. Процедура рекрутування учасників фокус-груп визначена формальними вимогами до числа учасників, відсутністю у них досвіду участі в групових інтерв'ю, знайомством один другом і т.п. Однак є дослідження, де ставиться завдання відбору типових випадків для вивчення соціального явища. Часто такий типовий випадок для інтерв'ю треба виділити з невизначеного безлічі аналогічних випадків. Наприклад, не-обходимо логічно обґрунтувати, чому вибрано по 30 інтерв'ю з чоловіками і жінками - учасниками протестного мітингу, на якому були присутні кілька тисяч чоловік, або 10 учасників міжетнічного конфлікту, число яких також за експертними оцінками не одна сотня.

Усі, хто знайомий зі специфікою якісних методів, як і приймають, що статистичні прийоми організації вибірки респондентов, які використовуються при масових опитуваннях, тут неприйнятні. Це пояснюється тим, що, з одного боку, вибір якісного підходу диктується як раз недостатнім представ-ленням про об'єкт дослідження і його повсякденних практиках, наприклад, про таких як маргінальні або латентні групи кри-номінальних спільнот: споживачі наркотиків, представники неформальної економіки та ін ., де репрезентативну вибірку зробити неможливо. З іншого боку, дослідник тут біль-ше стурбований тим, що представляють собою дані респонденти та їх практична діяльність, ніж тим, як часто вони зустрічаються в суспільстві.

Тож не дивно, що дослідники, які вирішили викорис-товувати якісні методи, наполягають, що тут немає, не було і не може бути загальних правил відбору респондентів. Моделі поис-ка інформантів в якісних дослідженнях, побудовані на етапі їхнього задуму, часто носять умоглядний характер, тільки заплутують і відводять в сторону від «реальності», обмежують рамками усталених уявлень про предмет дослідження.

Створюється враження, що в якісному «поле» все визна-ділячи конкретні обставини і удача, а респондентів для інтерв'ю вибирають, як це сформулював Дж. MакКрекен, за власними «емпіричним правилам». Ці правила визна-чає сам дослідник, коли вирішує, наприклад, повинні респонденти мати експертне знання або буденне, відрізнятися за статтю та віком, знати один одного.

Говорячи про відсутність правил вибірки в якісних дослінях, автори тим не менше постійно згадують їх у вигляді рад, повчальних прикладів, притч, байок і розлогих роздумів про важливість володіння своїм професійним ремеслом - антрополога, етнографа, соціолога. Наприклад, дається пряма вказівка, що краще б «до поля» мати уявлення про необхідність ведення польового щоденника, прочитати Джексона або Емерсона1і потренуватися. Більш того, виявляється, що багато проблем можна було вирішити ще «до поля», якщо поговорити з експертами, прочитати спеціальну ли-тературе по темі або більш чітко прояснити для себе ключове питання (основна ідея, задум дослідження).[1]

Окремо слід виділити рекомендації розвивати і фіксувати рефлексію з приводу відбору респондентів, довіряти своїм відчуттям, «що я спілкуюся не з тими, хто мені потрібен».

Навіть якщо прийняти існування цих «емпіричних правил» за очевидний факт, то в будь-якому випадку вони повинні бути прописані і занесені в протокол дослідження, так як по ним можна буде судити про логіку планування майбутньої роботи, зокрема, про логіку створення вибірки респондентів. Однак на практиці ми в кращому випадку зустрічаємося з описом відбору респондентів в дусі антропологічних експедитора-ций, де дослідник поступово занурюється в «реальність» свого поля, намагається адаптуватися в незнайомому середовищі і знайти «провідників» в «чужому світі». Досить непогано розвинений жанр опису «історії успіху», коли пощастило і вдалося знайти «удач-ного інформанта», або «історії облому», коли «не пощастило з информантом» і респондент виявився, наприклад, некомпетентний-ним, закритим, брехливим, агресивним , неадекватним, балакучим, мовчазним, лякливим, - людиною, яка нічого не дав для дослідження.

Безумовно, ця рефлексія має величезне значення для раз-витку методології якісних досліджень і підготовки їх авторів, але так чи так уже соціолог в поле відданий напризволяще в порівнянні з кількісними дослідженнями з їх формальними правилами і щодо жорсткими алгоритмами проведення етапів роботи?

Тут буде запропоновано авторський підхід до вирішення завдання з відбору респондентів в якісному дослідженні. Мета автора - показати читачеві можливості формалізації вибіркової процедури для дослідницького інтерв'ю за допомогою реалізації послідовної логічної схеми для побудови моделі квотної вибірки до виходу в «поле». Сама ідея «формалізації», «Квотування» вибірки в якісному дослідженні передставляється досить спірною. Однак групова методична.

Це завдання передбачає не академічний огляд всіх суще-ствующих типологій вибірки, а лише короткий опис найбільш поширених їх видів в польових дослідженнях, в яких автор брав участь, або описаних в літературі.

# 1.3 Схемотехнічне моделювання аналого-цифрових пристроїв

Це найбільш поширена задача проектування. Всі сучасні продукти передбачають введення проекту в редакторі принципових схем, після чого генерується список з'єднань, необхідний для роботи програми моделювання.

Світовим лідером в області автоматизації схемотехнічного проектування можна вважати програму SPICE. Вона була розроблена в університеті Берклі, США в 1972 р Різні версії цього алгоритму були в різний час запозичені виробниками програмного забезпечення для використання в своїх продуктах і в даний час під цією назвою відомий ряд програм різних фірм. В силу використання єдиного обчислювального алгоритму програми різних виробників є за великим рахунком всього лише графічні оболонки, що надають користувачем доступ до функцій програми SPICE, а також деякі додаткові можливості обралення отриманих даних. Перша версія цієї програми для персональних комп'ютерів PSpice розроблена в 1985 р типовий набір методів аналізу входять статичний, динамічний і частотний види аналізу, змішане логіко-аналогове моделювання (mixed-signal simulation), температурний (з індивідуальної значеннями температури за приладами) і шумовий аналізи, розрахунок на найгірший випадок і статистичний за методом Монте-Карло, спектральний аналіз, максимізація швидкодії (Оптимізується до 8 параметрів). У логічної частини реалізовано подієвий моделювання, виявивляють ризики збою, розраховуються залежні від навантаження затримки. Програма характеризується набором математичних моделей компонентів, включаючи моделі магнітних елементів з урахуванням насичення і гістерезису, довгих ліній з урахуванням затримок, відображень, втрат і перехресних перешкод, взаємодії аналогової і цифрової частин і ін. Бібліотека відкрита для включення моделей поль-зователя, є відповідні інструментальні засоби поповнення бібліотеки CircuitMaker 2000 (Www.altium.com, www. Circuit-maker.com) - найпростіша система моделювання фірми Microcode Engineering (нині належить компанії Protel), в якій прийняті моделі компо-нентов стандарту SPICE. Програма містить велику бібліотеку моделей промислових виробів електронних компонентів з можливістю оперативного перегляду їх основних характеристик (на-приклад, для транзистора - це тип корпусу, максимальна напруга, струм, частота, фірма-виробник і ін.). Програма дозволяє досить оперативно готувати електричні схеми аналогових, цифрових або змішаних аналого-цифрових пристроїв і проводити їх моделювання з отриманням результатів у вигляді осцилограм сигналів і графікі частотних характеристик; можливе отримання точних відліків за допомогою вертикальних і горизонтальних візирних ліній; дозволяє контролювати режим по постійному струму в обраних точках схеми. Відмінною особливістю програми є наявність анімаційних компонентів (запуск ракети, старт автомобілів), покликаний-них імітувати кінцевий результат роботи схеми, а також наявність навчального посібника в демонстраці-онном режимі. Вона інтегрується з програмою розробки друкованих плат TraxMaker. Electronics Workbench 5(Www.interactiv.com) - розробка фірми Interactive Image Technologies. Як і зволяет моделювати аналогові, прості цифрові і аналого -Цифрові схеми електронних пристроїв, пристроїв автоматики, електромеханічних пристроїв, а також схем, що складаються з функцио-нальних блоків із заданими передавальними функціями. Є велика бібліотека моделей ана-логів і цифрових пристроїв, включаючи моделі інтегральних схем і польових транзисторів. Дозволяє створювати і редагувати принципову схему, підключати вимірювальні прилади і виконати зазначені нижче види аналізу. Можливість підключення в схему вимірювальних приладів, по зовніш-нього вигляду і характеристикам наближаються до промислових зразків, є однією з відрізни-них особливостей програми, Як контрольно-вимірювальних приладів можна вико-ти мультиметр, осцилограф, вимірювач АЧХ і ФЧХ, логічний аналізатор, логічний преобразо-ватель, генератор слів, функціональний генератор. Програма дозволяє імітувати відмови компонентів у вигляді розриву, короткого замикання, наявності перехідного опору. Реалізовано види аналізу[2]:

1. DC-аналіз (режими роботи по постійному струму);
2. АС-аналіз (АЧХ і ФЧХ);
3. аналіз перехідних процесів;
4. багатоваріантний аналіз по постійному струму;
5. Фур'є-аналіз і аналіз спектра внутрішніх шумів;
6. аналіз нелінійних і інтермодуляціонних спотворень;
7. аналіз при варіації параметрів схеми і температури;
8. розрахунок нулів і полюсів передавальних функцій;
9. розрахунок відносної чутливості характеристик схеми до зміни параметрів вибрати-го елемента в режимах DC і АС;
10. розрахунок на найгірший випадок;
11. статистичний аналіз методом Монте-Карло.

Користувачеві програми дається можливість налаштування параметрів аналізу - вибір методу чисельні-ного інтегрування; завдання величин відносної і абсолютної похибки обчислень; завдання мінімальної провідності гілок; зміна величини кроку і числа ітерації при статичному аналізі.

Наприкінці 1998 р. програма Electronics Workbench 5.3 доповнена найпростішої програмою розробки друкованих плат EWB Layout (Multiboard), причому інтерфейс з іншими відомими САПР друкованих плат не передбачений, і потім серією програм Multisim 2001, Ultiboard 2001, Ultiroute і Commsim 2001.

Перша з них дозволяє здійснювати SPICE, VНDL, Verilog і змішане моделювання, дві інші програми виконують розміщення і трасування з'єднань на платах, що мають до 32 шарів. Multisim 2001 надходить до замовника з повною базою компонентів. Кожен з 16000 компонентів вклю-чає імітаційну модель, схематичний символ, електричні параметри і макет для розведення. Для програми створено інформаційний ресурс Інтернет з більш 12 мільйонами компонентів (edaParts.com), що дає користувачам швидкий і простий доступ до центру конструювання (Design Center), найбільшої в світі бази даних з електронних компонентів. Програма має модуль для редагування, імпорту або створення нових

Micro-Cap 7/8 - розробка фірми Spectrum Software. Надає практично всі види аналізу аналогових і цифрових схем, а також засоби синтезу пасивних і активних аналогових фільтрів;

1. інтерфейс з програмами розробки друкованих плат OrCAD, P-CAD, Protel і ін .;
2. режим анімації при аналізі;
3. аналіз S-параметрів лінійних 4-полюснік і кругові діаграми (діаграми Сміта) для мо-делірованія високочастотних пристроїв;
4. редактор впливів Stimulus Editor;
5. редактор компонентів Component Editor.
6. Інші достоїнства Micro-Cap:
7. вичерпна вбудована допомогу;
8. можливість завдання функціональних залежностей параметрів схеми (як функцій часу, струмів гілок і вузлових потенціалів), причому ці можливості навіть трохи ширше, ніж в Or-CAD - в них можна зробити параметри пасивних компонентів залежними тільки від струмів або напруг інших гілок схеми, в цих цілях не потрібно застосовувати керовані джерела струмів або напруг;
9. багатосторінковий графічний редактор принципових схем, що підтримує ієрархічні структури;
10. поведінковий моделювання аналогових і цифрових компонентів, можливість опису цифрових компонентів за допомогою логічних виразів, що дозволяє моделювати динамічн-ські системи, задані не тільки принциповими, але і функціональними схемами;
11. велика бібліотека компонентів;
12. макромоделі компонентів можуть бути представлені у вигляді принципових електричних схем або в текстовому вигляді;
13. графіки результатів виводяться в процесі моделювання або після його закінчення за вибором користувача, є сервісні можливості обробки графіків;
14. багатоваріантний аналіз при варіації параметрів і статистичний аналіз за методом Монте-Карло;

Є спеціальна програма для розрахунку параметрів математичних моделей аналогових компонентів за довідковими або експериментальними даними.

Програма Micro-Cap дуже зручна для початкового освоєння схемотехнічного моделювання електронних схем.

OrCAD 9.2 / 10 - інтегрований програмний комплекс для наскрізного проектування-вання аналогових, цифрових і змішаних аналого-цифрових пристроїв, синтезу пристроїв про-грамміруемой логіки і аналогових фільтрів. Випущений корпорацією Cadence Design Systems (колишня DesignLab), в яку в 1999 р влилися колишні розробники програми - компанії MicroSim і Or-CAD. Проектування починається з введення принципової схеми, її моделювання та оптимізації і закінчується створенням керуючих файлів для програматорів, розробкою друкованої плати і ви-водом керуючих файлів для фотоплоттерів і свердлильних верстатів. Основні модулі системи: OrCad Capture - керуючий модуль. За допомогою цього модуля створюються принципові схеми проектів різного типу. При синтезі ПЛІС і моделюванні цифрових пристроїв цей модуль працює модулем OrCad Express. При моделюванні аналогових або аналого-цифрових пристроїв він працює модулем PSPICE, при параметричної оптимізації - спільно з модулем PSPICE Optimizer, при розробці друкованих плат - з модулем OrCad Layout. OrCad Capture Cis (Component Information System) - модуль для створення принципових схем з під-Держко Internet. Зареєстрований користувач отримує доступ до каталогу з 200 000 компонентів провідних фірм-виробників.

OrCAD PSpise - модуль моделювання аналогових і цифрових пристроїв. Дозволяє розраховувати режими по постійному струму, чутливість характеристик до варіації параметрів компонентів, передавальні функції, частотні, фазові і шумові характеристики; перехідні процеси, прово-дить спектральний аналіз, статистичні випробування за методом Монте-Карло, багатоваріантний аналіз, параметричну оптимізацію.

OrCad Layout - модуль описаний в розділі «Проектування друкованих плат». Protel DXP - інтенсивно розвивається система наскрізного проектування аналого-вих і цифрових електронних пристроїв фірми Protel. Дозволяє проводити всі види аналізу з стан-дартного набору, пропонованого будь-якими програмами на базі ядра SPICE. За своїми функціональними можливостями наближається до системи OrCAD. Дозволяє виконувати проектування аналогових та аналого-цифрових пристроїв, в тому числі задаються бакшишниками принциповими схемами ієрархічної структури і схемами, що містять найсучасніші ПЛІС, проводити размеще-ня (компоновку / трасування) друкованих плат із застосуванням різних програм автотрасування, проводити аналіз цілісності сигналів.

View Analog - спочатку був розроблений компанією ViewLogic, а згодом увійшов до складу пакета eProduct Designer компанії Innoveda. Модуль має стандартний набір функцій моделювання змішаних аналого-цифрових пристроїв на базі алгоритму SPICE, але він настільки тес-но інтегрується з продуктами сімейства Fusion, що дозволяє моделювати поведінку программи-руемой логічної схеми, описаної на мовах VHDL, Verilog, Abel в оточенні аналогових компонентов.

Прикладом вітчизняних програм схемотехнічного аналізу можуть служити версії програм ПА: версія ПА7, в якій поряд з видами аналізу, звичайними для програм аналізу електронних схем, реалізовано моделювання механічних, гідравлічних, теплових процесів, і остання версія ПА9, написана на мові Java і орієнтована на використання в розподілених системах проектування.

# 1.4 Синтез логічних схем

Важливим завданням проектування є синтез логічної схеми для подальшої її реалізації на програмованих логічних інтегральних схемах (PLD, CPLD, FPGA). Як правило, в продуктах, призначених для цих цілей, використовується завдання функціонування роботи схеми, зроблене на одній з мов опису апаратури (HDL), наприклад, VHDL або Verilog. Для простих пристроїв проект може бути заданий у вигляді принципової схеми.

MAX + PLUS II фірми Altera пропонує повний спектр можливостей логічного дизайну: різноманітні засоби опису проектів з ієрархічною структурою, потужний логічний синтез, компіляцію з заданими часовими параметрами, поділ на частини, функціональне і тимчасової тестування (симуляцію), тестування декількох пов'язаних пристроїв , аналіз часів-них параметрів системи, автоматичну локалізацію помилок, програмування і верифікацію пристроїв.

Програма PeakFPGA (www.peakfpga.com) була спочатку розроблена компанією Accolade Design Automation, яка згодом увійшла до складу Protel. Програма призначена для побудови про-тів, описаних на мові VHDL, і упаковки їх в конкретні мікросхеми різних виробників. Потужні засоби моделювання і верифікації дозволяють проводити швидкий пошук помилок за різними критеріями і налагодження розроблюваного пристрою. Для опису проектів використовується мова CUPL, що дозволяє описувати логічні схеми довільної комбінацією трьох методів: буле-вимі виразами, таблицями істинності і методом кінцевих автоматів. Є можливість про-хизування окремих частин проектів в редакторі принципових схем з використанням спеціальних бібліотек. Синтезоване пристрій упаковується в конкретну мікросхему,

Більш потужним рішенням служить програма FPGA Studio, яку компанія Cadence початку предла-гать своїм користувачам замість програми OrCAD Express. Ця програма забезпечує розширений-ні можливості синтезу і моделювання логічних схем з подальшою упаковкою їх в мікро-схеми провідних виробників. Вона перевершує по функціональності продукти, пропоновані когось паніей Protel, проте варто при цьому майже на порядок дорожче.

Найбільш потужним рішенням в даній області є сімейство продуктів Fusion компанії Innoveda, що входить до складу інтегрованого продукту eProduct Designer. Програма Fusion / Speed-Wave дозволяє працювати на мові VHDL, Fusion / VSCi - на мові Verilog, ViewPLD - на мові ABEL, Fusion / ViewSim - на вентильному рівні. Всі ці програми об'єднуються в єдине ціле під управ-ленням модуля IntelliFlow, а в комбінації зі згаданим модулем ViewAnalog пропонують пользовате-лям неперевершені можливості по розробці складних ієрархічних проектів, окремі частини яких використовують різні технології. Найближчим конкурентом продуктів Innoveda є стоїть майже вдвічі дорожче пакет DSP Designer від Agilent Technologies.

Окремо слід згадати пакет SystemView компанії Elanix також дозволяє моделювати логічні схеми і упаковувати їх в ПЛІС фірми Хiliпх. Однак уро-вень моделювання проекту тут принципово інший - ця програма призначена для моделювання систем на рівні структурних схем. Тут використовуються поведінкові моделі, що дозволяють оцінити працездатність проекту на вентильному, а не на схемотехническом рівні. До переваг пакета можна віднести наявність потужного модуля синтезу цифрових фільтрів.

Проектування СВЧ-пристроїв. Принципово інший рівень складності завдань вирішують системи проектування НВЧ-пристроїв. Як правило, для отримання характеристик об'ємних структур тут потрібно рішення рівнянь Мак-Свелл, а для моделювання лінійних і нелінійних схем звичних моделей і методів, вико-мих ядром SPICE, недостатньо.

Microwave Office 2003- програма фірми Applied Wave Research дозволяє мо-деліровать СВЧ-пристрої, задані як в вигляді принципових, так і функціональних схем. Обсягів по-диня в собі модулі аналізу лінійних і нелінійних схем, проектування топології, електромагнітного-нітних моделювання планарних багатошарових структур (2.5-мірне моделювання), моделювання систем зв'язку на рівні структурних схем. Нелінійний аналіз тут виконується методом гармо-ного балансу і рядів Вольтерра. Електромагнітне моделювання планарних СВЧ-пристроїв ви-виконується методом моментів Гальоркіна. Модуль моделювання структурних схем, спочатку роз-бота компанією ICUCOM прекрасно інтегрований в середу і має найбільший набір бібліотек моделей. Редактор топологій є не просто графічне середовище промальовування топологій СВЧ-пристроїв,

Багатофункціональна автоматизована система SERENADE фірми Ansoft має модульну структуру. Деякі модулі системи підтвердили свою ефективність протягом тривалого періоду експлуатації. Модуль HSFF призначений для аналізу тривимірних електромагнітних-нітних полів, модуль Harmonica забезпечує проектування при використанні лінійних моделей таких пристроїв, як погоджують СВЧ-ланцюга, ланцюга зв'язку СВЧ-діапазону, фільтри, а також нелінійні пристрої НВЧ-діапазону (підсилювачі потужності, змішувачі, генератори , перемикачі), модуль Trilines призначений для розрахунку ліній передачі, модуль Synthesis - для синтезу фільтрів, модуль Super-Spice виконує моделювання НВЧ-пристроїв в тимчасовій області за допомогою системи SPICE, модуль Microwave Success моделюючи системи радіотелефонії.

Сімейство програмних продуктів MMICAD, вироблених фірмою Optotek, має у своєму складі 2-лінійний аналізатор; модуль, за допомогою якого здійснюється введення принципи-альних схем; модуль, за допомогою якого здійснюється компонування / трасування; систему, об'єд-няющих модуль, за допомогою якого здійснюється введення принципових схем, модуль компонуються-ки / трасування, а також лінійний аналізатор; нелінійний аналізатор в тимчасовій області; модуль синтезу фільтрів; модуль збору інформації та малосигнальних аналізу схем з польовими транзистора-ми; модуль збору інформації та моделювання схем з польовими транзисторами в режимах великого і малого сигналів.

Пакет RF Design System Suite - (скорочено RFDS) фірми HP EEsof / HPHighFreqency Design позволя-ет проектувати радіочастотні ланцюги стільникових радіотелефонів, інших персональних рухомих комунікаційних систем, універсальні процесори, ланцюги радіомовних супутників, відеосіс-теми і бездротові локальні обчислювальні мережі. Програмне забезпечення включає аналізатор лінійних і нелінійних ланцюгів, який має двосторонню зв'язок з векторним аналізатором ланцюгів, аналізатором спектру і цифровим осцилографом. RFDS Suite має інструменти для аналізу переходних процесів в ланцюгах і електромагнітного аналізу, системну бібліотеку, інструмент для когось поновкі. Крім того, можлива інтеграція з програмними продуктами інших фірм.

Система Sonnet Software Suite of 3D Planar High-Frequence Electromagnetic Software (коротко - Sonnet Suite) служить для дослідження високочастотних електромагнітних полів сучасних 3D-nnaHapHbix ланцюгів і антен. Для APLAC - призначена для проектування і моделювання електричних схем і систем в тимчасовій і частотній областях. До їх складу можуть входити як цифрові, так і аналогові компоненти, в тому числі пристрої діапазону СВЧ. Виконуються наступні види розрахунків: режим по постійному струму, частотні характеристики, спектральна щільність і коефіцієнт шуму, чувст-вітельность і параметричну оптимізація, перехідні процеси, спектри сигналів, аналіз періодичних-чеських режимів, статистичний аналіз за методом Монте-Карло. Крім того, до складу APLAC входить підпрограма розрахунку тривимірних електромагнітних полів мікросмужкових конструкцій і інших пристроїв діапазону СВЧ. Останнє, на що слід звернути увагу, це можливість введення резуль-татів вимірювань і виведення керуючих сигналів за допомогою інтерфейсних плат стандарту IEEE-488 (GPIB, HP-IB),дослідження резонансу оболонки включено дослідження поля шести-вугільного короба.

Пакет AppCad (повна назва AppCad Design Software) фірми Hewlett-Packard призначений для б-стрих інженерних розрахунків виробів ВЧ і СВЧ-діапазонів, від схем з дискретними діодами і транзи-сторі до СВЧ-інтегральних схем. Працює в діалоговому режимі і дозволяє розраховувати: зміщений-ні струмом і напругою інтегральні схеми; схеми з діодами Шотткі в режимах великого і малого сигналів; коефіцієнт посилення підсилювачів; коефіцієнти передачі та відображення ліній; десять типів ліній; ланцюга зміщення біполярних транзисторів, емітерний зворотний зв'язок і подільники напруги; надійність і тепловий режим.

Програми CST Microwave Studio німецької компанії CST і QuickWave-3D польської QWED вирішують задачу повного тривимірного електромагнітного моделювання об'єк-приймальних СВЧ-пристроїв. Остання програма більше відома під своїм комерційним назвою CONCERTO. Обидві програми використовують метод кінцевих різниць (FDTD), доповнений методом конформних перетворень. Головна відмінність програм полягає в завершеності інтерфейсу: німець-кий продукт є закінченою графічної середовищем для постановки задачі, польський ж крім промальовування структури вимагає від користувача написання програмного коду. Обидві програми мають засоби оптимізації і обидві показують кращі результати моделювання, ніж однойменні продук-ти HFSS від Agilent і Ansoft.

Більш просте і дешеве рішення пропонує інша німецька компанія – IMST. Її продукт EMPIRE використовує класичну реалізацію методу FDTD, тому для отримання точних ре-зультатів для об'ємних структур довільної форми, утворених криволінійними поверхнями, потрібно більше часу і обчислювальної потужності. Ще тут можна отримувати різні частотні характеристики НВЧ-пристроїв, а також діаграми спрямованості антен.

Поведінковий (імітаційне) моделювання на рівні структурних схем. Ця категорія продуктів націлена на моделювання електронних систем на поведінковому рівні. Тут від програми потрібна велика гнучкість, наявність точних бібліотек функціональних модулів, можливість сполучення з іншими системами моделювання.[3]

Програмний комплекс MATLAB 6.5 фірми The MathWorks, Inc. відноситься до систем комп'ютерної математики. Система підтримує виконання операцій з мат-ріцамі, полиномами, рішення лінійних, нелінійних і диференціальних рівнянь, оптимізацію, інтерполяцію, побудова графіків, тривимірних фігур та багато іншого. MATLAB - інтерактивна система, основним об'єктом якої є масив, для якого не потрібно вказувати розмір-ність явно. Це дозволяє вирішувати багато обчислювальні завдання, пов'язані з векторно-матричними формулюваннями.

Головним розширенням системи MATLAB є програмний пакет Simulink. Уже в силу своєї назви Simulink виконує як би симуляцію роботи модельованих систем і пристроїв, яку також називають імітаційним моделюванням. Для побудови функціональної блок-схеми Simulink має велику бібліотеку блокових компонентів і зручний редактор блок-схем. Simulink автомати-зірует наступний, найбільш трудомісткий етап моделювання: він становить і вирішує складні системи алгебраїчних і диференціальних рівнянь, що описують задану функціональну схему, забезпечуючи зручний і наочний візуальний контроль за поведінкою створеного користувачем виртуального пристрою. Цінність Simulink полягає і у великій, відкритій для вивчення і модифікації бібліотеці когось тами. Вона включає джерела сигналів з практично будь-якими часовими залежностями, мас-штабірующіе, лінійні і нелінійні перетворювачі з різноманітними формами передавальних ха-рактерістік, квант пристрій, що інтегрують і диференційні блоки і т. Д. У бібліотеці є цілий набір віртуальних реєструючих пристроїв - від простих вимірників типу вольт-метра або амперметра до універсальних осцилографів, що дозволяють переглядати тимчасові зави - симости вихідних параметрів модельованих систем - струмів, напруг, переме ений, тисків і т. п. Є графічний пристрій для створення фігур в полярній системі координат, наприклад фігур Ліссажу і фазових портретів коливань. Simulink має засоби анімації і звукового супроводоня.System View 5.0 - програма компанії Elanix є конструктор, з по - міццю якого зі стандартних "кубиків" будується функціональна схема досліджуваної електронної системи. Продукт призначений для моделювання будь-яких динамічних систем і дозволяє провести моделювання в тимчасовій області будь-якої системи, будь то логічна або аналогова схема або ж зовсім якась математична абстракція - головне, щоб для цього була необхідна бібліотека. Розраховуються перетворення Фур'є графіків, кореляційні і взаємно кореляційні функ-ції, виконуються арифметичні та тригонометричні операції, статистична обробка даних і багато іншого. З пакетом поставляється великий набір бібліотек функціональних блоків, є можливість створення користувацьких бібліотек будь-якої складності.

Пакет Microwave Office описаний в розділі «Проектування СВЧ-пристроїв».

Пакет Visual System Simulator фірми Applied Wave Research є програмою моделювання цифрових пристроїв систем зв'язку на системному рівні. Пакет має більш моделей функціональних блоків, прикладні бібліотеки, що підтримують всі поширені в даний час стандарти зв'язку, такі як: GSM, CDMA, 3G, GPS, DVB, HDTV і багато інших. Так як пакет Visual System Simulator орієнтований на моделювання саме телекомунікаційних систем, тут є великий набір моделей каналів. Сюди входять моделі таких ефектів, як за-МІРАННА, многолучевость, блокування і імпульсні перешкоди, що дозволяють досить точно оцінювати ра-боту систем в реальних умовах експлуатації.

# 1.5 Проектування друкованих плат

Будь-яка система проектування друкованих плат є складним комплексом програм, забезпе-чувати наскрізний цикл, починаючи з промальовування принципової схеми і закінчуючи генерацією керуючих файлів для обладнання виготовлення фотошаблонів, свердління отворів, складання і електроконтролю.

Найкращих результатів досягла компанія Mentor Graphics. Маючи власну систему проектування друкованих плат Mentor BoardStation, компанія поглинула двох своїх конкурентів, компанії Verybest і Innoveda, і зараз продовжує розвивати лінії продуктів Expedition PCB і PADS PowerPCB. Ключем до успіху компанії стала орієнтація на сучасні інтегровані середовища проектування для Widows.

Пакет Expedition PCB представляє зараз найбільш потужне рішення в області проектування плат. Основу системи складають середовище AutoActive, що дозволяє реалізувати такі функції, як предтопо-логічний аналіз цілісності сигналів, інтерактивна і автоматичне трасування з урахуванням требо-ваний високочастотних плат і спеціальних технологічних обмежень, що накладаються вико-ристанням сучасної елементної бази. Єдине середовище дозволяє моделювати наведення в провідниках безпосередньо під час прокладання траси або шини і контролювати перевищення ними заданого рівня.

Інший продукт компанії Mentor, система PADS PowerPCB пропонує більш дешеве рішення. Ця система може похвалитися кращим Автотрасувальник BlaseRouter, що підтримує всі необхідні при трасуванні високочастотних плат функції. Пакет має модулі предтопологі-чекого і посттопологіческого аналізу, тісно взаємодіють з системою контролю обмежень.

Далі по потужності пропонованих рішень йде компанія Cadence. Для верхнього рівня проектуванння пропонується пакет PCB Design Studio. В якості редактора друкованих плат тут використовується програма Allegro, що дозволяє розробляти багатошарові і високошвидкісні плати з високою щільністю розміщення компонентів. В якості штатного модуля авторозміщення і автотрасування тут використовується програма SPECCTRA, керована обширним набором правил проектування і технологічних обмежень. Виконується аналіз електромагнітних-нітних сумісності. Інший продукт компанії Cadence, пакет OrCAD рекомендується як більш легке і дешеве рішення для проектування друкованих плат. Даний пакет розглядається фірмою Cadence як пріоритетна система введення проектів і моделювання: модулі Capture CIS і PSpice зараз постав-ляють в складі пакету PCB Design Studio. Редактор друкованих плат OrCAD Layout має три різні конфігурації з різними функціональними можливостями. У проекті плати тут може присутність-вать до 30 шарів, 16 з яких можуть бути сигнальними. Є вбудовані засоби авторазмеще - ня і автотрасування, а також інтерфейс з програмою SPECCTRA. Однак головним модулем тут є не редактор друкованих плат, а редактор принципових схем OrCAD Capture CIS, оснащений єдиною в своєму роді системою управління базами даних компонентів. Система CIS (Component Information System) була розроблена для забезпечення всім користувачам OrCAD доступу через Ін-тернет до централізованих баз даних компонентів на сайті www.spincircuit.com. Гнучкість системи CIS дозволяє організувати корпоративні бази дозволених до застосування компонентів і працювати в локальних мережах, а також використовувати процедури автоматизованого нормоконтролю.

Третім виробником САПР друкованих плат можна назвати австралійську компанію Altium Technologies. Завдяки вмілій інвестиційній політиці ця фірма змогла звести до мінімуму втрати, пов'язані зі спадом ринку високих технологій в 2002 році. У серпні 2002 року компанія випустила в світ пакет Protel DXP, що представляє собою продовження власної оригінальної ліній продуктів Protel. Цей пакет забезпечує наскрізний цикл проектування-вання змішаних аналого-цифрових друкованих плат з використанням програмованої логіки фірм Xilinx і Altera. Весь інструментарій реалізований на базі інтегрованого середовища проектування, рабо-тане під управлінням Windows XP. До були раніше засобам посттопологіческого аналізу це-лостності сигналів додалася можливість виконувати предтопологіческій аналіз.

На тлі повної мобілізації зусиль на розробку пакета Protel DXP компанія Altium продовжує раз-Віва свій другий пакет проектування друкованих плат P- CAD 2002. Ця система є досить популярною, що обумовлено, з одного боку, хорошою функціональностю програми і, з іншого боку, по асоціації з поширеними тут старими версіями P-CAD 4.5 - 8.7. У 1996 р фірма ACCEL Technologies вперше представила версію широко відомої сис-теми розробки друкованих плат P-CAD на платформі Windows. Оновлений продукт отримав нову назву ACCEL EDA. З цього моменту продукт ACCEL EDA набув широкої популярності серед розробників електронних пристроїв. У вересні 1999 р вийшла остання 15 версія продукту. 17 ян-варя 2000 р відбулося злиття двох провідних розробників систем САПР друкованих плат - фірм Protel International і ACCEL Technologies, які об'єднали свої спільні зусилля під торговою маркою Protel (нині Altium). З березня 2000 р продукт ACCEL EDA змінив свою назву на P-CAD. Система P-CAD 2002 виконує повний цикл проектування друкованих плат, а саме:

1. графічний введення електричних схем;
2. змішане аналого-цифрове моделювання на основі ядра SPICE3;
3. упаковку схеми на друковану плату;
4. інтерактивне розміщення компонентів;
5. інтерактивну і автоматичне трасування провідників;
6. контроль помилок
7. в схемі і друкованій платі; випуск документації;
8. аналіз цілісності сигналів і перехресних спотворень;
9. підготовку файлів Gerber і NC Drill для виробництва друкованих плат;
10. підготовку бібліотек символів, топологічних посадкових місць і моделей компонентів. Основні можливості P-CAD 2002: Додати
11. Зручний інтерфейс для Windows.
12. Зберігання проектної інформації в бінарних і текстових файлах.
13. Зручна довідкова система.
14. Проект схеми може містити 999 аркушів, проект плати - до 999 шарів (11 з них стандарт-них).
15. Число ланцюгів в проекті - до 64000.
16. Число вентилів в компоненті - до 5000.
17. Максимальне число висновків у компонента - 10000.
18. Максимальні розміри листа схеми або креслення друкованої плати 60х60 дюймів.
19. Підтримка дюймової і метричною систем заходів.
20. Граничне дозвіл 0.0001 дюйма (0.1 мила) або 0.01 мм (10 мікрон).
21. Мінімальний кут повороту компонентів на платі - 0.1 град.
22. Довжина імен компонентів - до 30 символів, максимальний обсяг текстових написів і Атрібі-тов - до 20000 символів.
23. Механізм переносу змін друкованої плати на схему і навпаки.
24. Бібліотеки компонентів, що містять більше 27000 елементів і сертифіковані за стандартом ISO 9001.

Не можна не згадати ще один майже невідомий, але досить потужний і популярний в світі продукт - Visula компанії ZUKEN. Продукти цієї компанії забезпечують наскрізний цикл проектування і пропонують потужні засоби моделювання та синтезу про-грамміруемой логіки з подальшою розробкою друкованої плати. Тут є стандартний набір інструментарію, а також власні кошти авторозміщення і автотрасування. Слід зазначити, що компанія ZUKEN також пропонує користувачам інтегровані засоби тривимірного твердо-ного моделювання розроблювальних пристроїв.[4]

CircuitMaker, розроблений фірмою MicroCode Engineering, після злиття її з компанією Protel став пропонуватися як найдешевше рішення для проектування нескладних друкованих плат. Стандартна версія дозволяє розробляти плати, що містять до б сигнальних шарів і до двох шарів металліза-ції. Цей продукт має зручний і гнучкий редактор схем, а також програму моделювання. Автоматичне розміщення і трасування реалізуються і в ряді інших систем проектування друк - них плат, зокрема, у вітчизняній САПР RELIEF з оригінальним алгоритмом швидкої щільної упаковки різногабаритних елементів. Алгоритм заснований на багаторазовому дихотомічному розподілі безлічі розміщуваних елементів

# РОЗДІЛ 2 ПОСТАНОВКА ЗАДАЧІ

В бакалаврській роботі проектується система для побудови логічних схем.

В даному проекті поставлена проблема створення програми проектування логічних схем, а також їх моделювання. Тобто в даній програмі користувач може не тільки зібрати логічну схему, а й побачити, як дана логічна схема функціонує.

Було вивчено теоретичну частину засновану на булевої алгебри і провів аналіз програм-аналогів.

. Було почато зі створення логічної частини програми і розробив алгоритм роботи логічних схем на основі двох моделей: контакту і примітивного елемента. Це дозволило на рівні мови програмування збирати логічні схеми і моделювати їх роботу. Важливою особливістю є також і те, що логічну схему можна визначити як елемент і в подальшому використовувати в складі більшої логічної схеми.

Далі перед автором постало питання графічного редагування логічних схем. Було самостійно розроблено на її базі такі компоненти, як: кнопки, вкладки, панелі, рядок стану, що перетягуються об'єкти, діалоги і поля введення.

# 2.1 Характеристика задачі

Чому необхідно вміти будувати логічні схеми? Справа в тому, що з вентилів складають більш складні схеми, які дозволяють виконати арифметичні операції і зберігати інформацію. Причому схему, що виконує певні функції, можна побудувати з різних за поєднанням і кількістю вентилів. Тому значення формального уявлення логічної схеми надзвичайно велике. Воно необхідне для того, щоб розробник мав можливість вибрати найбільш підходящий йому варіант побудови схеми з вентилів. Процес розробки загальної логічної схеми пристрою (в тому числі і комп'ютера в цілому) таким чином стає ієрархічним, причому на кожному наступному рівні в якості «цеглинок» використовуються логічні схеми, створені на попередньому етапі.

Алгебра логіки дала в руки конструкторам потужний засіб розробки, аналізу та вдосконалення логічних схем. Справді, набагато простіше, швидше і дешевше вивчати властивості і доводити правильність роботи схеми за допомогою виражає її формули, ніж створювати реальний технічний пристрій. Саме в цьому полягає сенс будь-якого математичного моделювання.

Логічні схеми необхідно будувати з мінімально можливої ​​кількості елементів, що в свою чергу, забезпечує більшу швидкість роботи і збільшує надійність пристрою.

Алгоритм побудови логічних схем:

1) Визначити число логічних змінних.

2) Визначити кількість базових логічних операцій і їх порядок.

3) Зобразити для кожної логічної операції відповідний їй вентиль.

4) З'єднати вентилі в порядку виконання логічних операцій.

2.2 Логічні елементи

Будь-яке цифровий пристрій - персональний комп'ютер, або сучасна система автоматики складається з цифрових інтегральних мікросхем (ІМС), які виконують певні складні функції. Але для виконання однієї складної функції необхідно виконати кілька найпростіших функцій. Наприклад, додавання двох двійкових чисел розміром в один байт відбувається всередині цифрової мікросхеми званої «процесор» і виконується в кілька етапів великою кількістю логічних елементів знаходяться всередині процесора. Двійкові числа спочатку запам'ятовуються в буферній пам'яті процесора, потім переписуються в спеціальні «головні» регістри процесора, після виконується їх складання, запам'ятовування результату в іншому регістрі, і лише після результат складання виводиться через буферну пам'ять з процесора на інші пристрої комп'ютера.

Процесор складається з функціональних вузлів: інтерфейсів вводу-виводу, осередків пам'яті - буферних регістрів і «акумуляторів», суматорів, регістрів зсуву і т.д. Ці функціональні вузли складаються з найпростіших логічних елементів, які, в свою чергу складаються з напівпровідникових транзисторів, діодів і резисторів. При конструюванні простих тригерних і інших електронних імпульсних схем, складні процесори не застосувати, а використовувати транзисторні каскади - «минуле століття». Тут і приходять на допомогу - логічні елементи.

Логічні елементи, це найпростіші «кубики», складові частини цифрової мікросхеми, що виконують певні логічні функції. При цьому, цифрова мікросхема може містити в собі від одного, до декількох одиниць, десятків, ... і до кількох сотень тисяч логічних елементів в залежності від ступеня інтеграції. Одиниця цифрової інформації це «один біт». Він може приймати два логічних стану - логічний нуль "0", коли напруга дорівнює нулю (низький рівень), і стан логічної одиниці «1», коли напруга дорівнює напрузі живлення мікросхеми (високий рівень).[5]

Оскільки найпростіший логічний елемент це електронний пристрій, то це означає, що у нього є входи (вхідні висновки) і виходи (вихідні висновки). І входів і виходів може бути один, а може бути і більше.

Для того, щоб зрозуміти принципи роботи найпростіших логічних елементів використовується «таблиця істинності». Крім того, для розуміння принципів роботи логічних елементів, входи, в залежності від їх кількості позначають: Х1, Х2, ... ХN, а виходи: Y1, Y2, ... YN.

Функції, що виконуються найпростішими логічними елементами, мають назви. Як правило, попереду функції ставиться цифра, що позначає кількість входів. Найпростіші логічні елементи завжди мають лише один вихід.

Розглянемо найпростіші логічні елементи

- «НЕ» (NOT) - функція заперечення (інверсії сигналу). Тому його частіше називають - «інвертор». Графічно, інверсія позначається порожнім кружечком навколо виведення елемента (мікросхеми). Зазвичай гурток інверсії ставиться біля виходу, але в більш складних логічних елементах, він може стояти і на вході. Графічне позначення елемента «НЕ» і його таблиця істинності представлені на малюнку зліва.

У елемента «НЕ» завжди один вхід і один вихід. По таблиці істинності випливає, що при наявності на вході елемента логічного нуля, на виході буде логічна одиниця. І навпаки, при наявності на вході логічного одиниці, на виході буде логічний нуль. Цифра «1» всередині прямокутника позначає функцію «АБО», її прийнято малювати і всередині прямокутника елемента «НЕ», але це зовсім нічого абсолютно не означає.

Позначення D1.1 означає, що D - цифровий логічний елемент, 1 (перша) - номер мікросхеми в загальній схемі, 1 (друга) - номер елемента в мікросхемі. Точно також розшифровуються і інші логічні елементи.

Часто, щоб відрізнити цифрові мікросхеми від аналогових мікросхем, застосовують позначення з двох літер: DD - цифрова мікросхема, DA - аналогова мікросхема. В подальшому, ми не будемо загострювати увагу на це позначення, а повернемося лише тоді, коли це буде необхідним.

Найпоширенішою мікросхемою «транзисторних-транзисторної логіки» (ТТЛ), яка виконує функцію «НЕ», є інтегральна мікросхема (ІМС) К155ЛН1, всередині якої є шість елементів «НЕ». Нумерація висновків цієї мікросхеми показана праворуч.

- «І» (AND) - функція складання (якщо на всіх входах одиниця, то на виході буде одиниця, в іншому випадку, якщо хоча б на одному вході нуль, то і на виході завжди буде нуль). В алгебрі-логіці елемент «І» називають «кон'юнктор». Графічне позначення елемента «2И» і його таблиця істинності представлені зліва.

Назва елемента «2И» позначає, що у нього два входи, і він виконує функцію «І». На схемі всередині прямокутника мікросхеми малюється значок «&», що на англійській мові означає «AND» (в перекладі на російську - І).

По таблиці істинності випливає, що на виході елемента «І» буде логічна одиниця тільки в одному випадку - коли на обох входах буде логічна одиниця. Якщо хоча б на одному вході нуль, то і на виході буде нуль.

Найпоширенішою мікросхемою «транзисторних-транзисторної логіки» (ТТЛ), яка виконує функцію «2І», є інтегральна мікросхема (ІМС) К155ЛИ1, всередині якої є чотири елементи «2І». Нумерація висновків цієї мікросхеми показана праворуч.

Для того, щоб вам було зрозуміліше що таке «2І», «3І», «4І», і т.д., приведу графічне позначення і таблицю істинності елемента «3І».

По таблиці істинності випливає, що на виході елемента «3І» буде логічна одиниця тільки в тому випадку - коли на всіх трьох входах буде логічна одиниця. Якщо хоча б на одному вході буде логічний нуль, то і на виході елемента також буде логічний нуль. Найпоширенішою мікросхемою ТТЛ, що виконує функцію «3І», є мікросхема К555ЛІ3, всередині якої є три елементи «3І».- «І-НЕ» (NAND) - функція складання з запереченням (якщо на всіх входах одиниця, то на виході буде нуль, в іншому випадку на виході завжди буде одиниця). Графічне позначення елемента «2І-НЕ» і його таблиця істинності наведені зліва.

По таблиці істинності випливає, що на виході елемента «2І-НЕ» буде логічний нуль тільки в тому випадку, якщо на обох входах буде логічна одиниця. Якщо хоча б на одному вході нуль, то на виході буде одиниця.

Найпоширенішою мікросхемою ТТЛ, що виконує функцію «2 І-НЕ», є ІМС К155ЛА3, а мікросхемами КМОП (комплементарний металооксидних напівпровідник) - ІМС К561ЛА7 і К176ЛА7, всередині яких є чотири елементи «2 І-НЕ». Нумерація висновків цих мікросхем показана праворуч.

Порівнявши таблиці істинності елемента «2І-НЕ» і елемента «2І» можна здогадатися про еквівалентність схем:



Додавши до елементу «2І» елемент «НЕ» ми отримали елемент «2 І-НЕ». Так можна зібрати схему, якщо нам необхідний елемент «2 І-НЕ», а у нас в розпорядженні є тільки елементи «2І» і «НЕ». І навпаки:



Додавши до елементу «2 І-НЕ» елемент «НЕ» ми отримали елемент «2И». Так можна зібрати схему, якщо нам необхідний елемент «2І», а у нас в розпорядженні є тільки елементи «2 І-НЕ» і «НЕ».

Аналогічним чином, шляхом з'єднання входів елемента «2І-НЕ» ми можемо отримати елемент «НЕ»:



Зверніть увагу, що було введено нове в позначенні елементів - дефіс, що розділяє праву і ліву частину в назві «2 І-НЕ». Цей дефіс неодмінний атрибут при інверсії на виході (функції «НЕ»).

- «АБО» (OR) - функція вибору (якщо хоча б на одному з входів - одиниця, то на виході - одиниця, в іншому випадку на виході завжди буде нуль). В алгебрі-логіці, елемент «АБО» називають «діз'юнктор». Графічне позначення елемента «2АБО» і його таблиця істинності наведені зліва.

Найпоширенішою мікросхемою ТТЛ, що виконує функцію «2АБО», є ІМС К155ЛЛ1, всередині якої є чотири елементи «2АБО». Нумерація висновків цієї мікросхеми показана праворуч.

Припустимо, що нам в схемі необхідний елемент, що виконує функцію «2АБО», але у нас є в розпорядженні тільки елементи «НЕ» і «2 І-НЕ», тоді можна зібрати схему, яка буде виконувати функцію «2АБО»:



- «АБО-НЕ» (NOR) - функція вибору (якщо хоча б на одному з входів - одиниця, то на виході - нуль, в іншому випадку на виході завжди буде одиниця). Як ви зрозуміли, елемент «АБО-НЕ» виконує функцію «АБО», а потім інвертує його функцією «НЕ».

Графічне позначення елемента «2АБО-НЕ» і його таблиця істинності наведена зліва.

Найпоширенішою мікросхемою ТТЛ, що виконує функцію «2АБО-НЕ», є ІМС К155ЛЕ1, а мікросхемами КМОП - К561ЛЕ5 і К176ЛЕ5, всередині яких є чотири елементи «2АБО-НЕ». Нумерація висновків цих мікросхем показана праворуч.[6]

Припустимо, що нам в схемі необхідний елемент, що виконує функцію «2АБО-НЕ», але у нас є в розпорядженні тільки елементи «НЕ» і «2 І-НЕ», тоді можна зібрати наступну схему, яка буде виконувати функцію «2АБО-НЕ» :



За аналогією з елементом «2 І-НЕ», шляхом з'єднання входів елемента «2АБО-НЕ» ми можемо отримати елемент «НЕ»:



- «Що виключає АБО» (XOR) - функція нерівності двох входів (якщо на обох входах елемента однакові сигнали, то на виході - нуль, в іншому випадку на виході завжди буде одиниця). Операція, яку він виконує, часто називають «складання по модулю 2».

Графічне позначення елемента «виключає Або" і його таблиця істинності наведені зліва.

Найпоширенішою мікросхемою ТТЛ, що виконує функцію «Що виключає АБО», є ІМС К155ЛП5, а мікросхемами КМОП - К561ЛП2 і К176ЛП2, всередині яких є чотири елементи «виключає Або». Нумерація висновків цих мікросхем показана праворуч.

Припустимо, що нам в схемі необхідний елемент, що виконує функцію «Що виключає АБО», але у нас є в розпорядженні тільки елементи «2 І-НЕ», тоді можна зібрати наступну схему, яка буде виконувати функцію «Що виключає АБО»:



У цифровій схемотехніці процесорів головна функція - «Підсумовування двійкових чисел», тому складний логічний елемент - «Акумулятор» є невід'ємною частиною арифметико-логічного пристрою будь-якого, без винятку процесора. Складовою частиною суматора є набір логічних елементів, що виконують функцію «Що виключає Або з перенесенням залишку». Що це таке? Відповідно до наукою «Інформатика», результатом складання двох двійкових чисел, дві одиниці одного розряду дають нуль, при цьому формується «одиниця переносу» в наступний старший розряд, який бере участь в операції підсумовування в старшому розряді. Для цього в схему додається ще один висновок «перенесення» - «Р».

Графічне позначення елемента «виключає Або з перенесенням» і його таблиця істинності представлена ​​зліва.

Така функція складання однорозрядних чисел у простих пристроях зазвичай не використовується, і як правило, інтегрована до складу однієї мікросхеми - сумматора, з мінімальною кількістю розрядів - чотири, для складання четирехбітних чисел. Через слабкий попит, промисловість таких логічних елементів не випускає. Тому, в разі необхідності, функцію «Що виключає Або з перенесенням» можна зібрати за такою схемою з елементів «2 І-НЕ» і «2АБО-НЕ», яка активно застосовується як всередині простих сумматоров, так і у всіх складних процесорах (в тому числі Pentium, Intel-Core, AMD і інших, які з'являться в майбутньому):



Перераховані вище логічні елементи виконують статичні функції, а на основі них будуються більш складні статичні і динамічні елементи (пристрої): тригери, регістри, лічильники, шифратори, дешифратори, суматори, мультиплексори.

# 2.3.Логические элементы в модели

Істинності значення нових висловлювань визначаються при цьому тільки істиннісними значеннями входять до них висловлювань. Побудова з даних висловлювань (або з даного висловлювання) нового висловлювання називається логічною операцією. Знаки логічних операцій називаються логічними зв'язками. Логічні зв'язки бувають одномісними (унарними), двомісними (бінарні), тримісні (тернарного) і т.д. Наприклад:

* З висловлювань «х> 2», «х <3» за допомогою зв'язки «і» можна отримати висловлювання «x> 2 і х <3»;
* з висловлювань «у> 10», «х <3» за допомогою зв'язки «або» можна отримати висловлювання «у> 10 або х <3»;
* з висловлювань «х> 2», «у <3» за допомогою зв'язки «якщо ..., то ...» можна отримати висловлювання «якщо x> 2, то у <3».

Істинність або хибність одержуваних таким чином висловлювань залежить від істинності і хибності вихідних висловлювань і відповідного трактування в'язок як операцій над висловлюваннями.

В алгебрі логіки логічні операції найчастіше описуються за допомогою таблиць істинності. У таблиці 2.1 наведено таблицю істинності для операції «заперечення» ( «інверсія»).

Таблиця 2.1

|  |  |
| --- | --- |
| А | НЕ А |
| 0 | 1 |
| 1 | 0 |

У таблиці 2.2 наведені основні бінарні логічні операції і зв'язки.

Таблиця 2.2

|  |  |  |  |
| --- | --- | --- | --- |
| Позначення логічної операції | інші позначеннялогічної операції | Назва логічної операції і зв'язки | Примітка (читається) |
| А1∧ А2 | А1 & А2А1⋅ А2А1А2 | кон'юнкція, логічне множення,логічне «і» | А1 і А2 |
| А1∨ А2 | А1 + А2 | диз'юнкція, роз'єднання,логічне «або» | А1 або А2 |
| А1→ А2 | А1⊇ А2А1⇒ А2 | імплікація, логічне слідування | якщо А1, то А2; А1 имплицирует А2; А1 тягне А2 |
| А1⊕ А2 | А1 + А2А1∨ А2А1Δ А2 | сума по модулю 2, розділова диз'юнкція, розділову «або» | А1 плюс А2; або А1, або А2 |
| А1 ~ А2 | А1≡ А2А1↔ А2А1⇔ А2 | еквіваленція, еквівалентність, рівнозначність, тотожність | А1 тоді і тільки тоді, коли А2; А1 еквівалентно А2 |
| А1⏐ А2 |  | штрих Шеффера,антікон'юнкція | невірно, що А1 і А2; А1 штрих Шеффера А2 |
| А1↓ А2 | А1°А2А1 А2 | стрілка Пірса,антідіз'юнкція, функція Вебба,функція Даггера | ні А1, ні А2; А1 стрілка Пірса А2 |

Примітка: А1 і А2 є висловлюваннями.

Зв'язки і частка «не» розглядаються в алгебрі логіки як операції над величинами, що набувають значення 0 (брехня / false) і 1 (істина / true), і результатом застосування цих операцій також є числа 0 або 1. У таблиці 3 представлені всі набори значень змінних А1 і А2 і значення функцій на цих наборах.

Таблиця 2.3

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А1 | А2 | ∧ | ∨ | → | ⊕ | ~ | ⏐ | ↓ |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |

### Інверсія

Заперечення висловлювання А (тобто не А) позначається, або , або  і часто читається: «заперечення А", "не А» або «А з межею».

Наприклад. Висловлення А = <Київ-столиця Франції>, тоді складне висловлювання НЕ А означає: не вірно, що А, тобто не вірно, що «Київ-столиця Франції>.[7]

### Кон'юнкція

Результатом операції кон'юнкції для висловлювання А∧ У буде істинна тільки тоді, коли істинні одночасно обидва висловлювання.

Наприклад. Висловлювання А = «Москва - столиця Росії» і В = «Рим - столиця Італії». Складне висловлювання А∧ В (А & В) істинно, так як істинні обидва висловлювання.

### Диз'юнкція

Результатом операції диз'юнкції для висловлювання А∨ У буде істинна тоді, коли істинно хоча б один вислів, що входить в нього.

Наприклад. Висловлювання А = «2 + 3 = 5» і В = «3 + 3 = 5». Складне висловлювання: А∨ В (А + В) істинно, так як істинно висловлювання А.

### Еквіваленція (рівнозначність)

Результатом операції еквіваленціі для висловлювання А ~ В буде істинна тоді, коли істинні або помилкові одночасно обидва висловлювання. Відмінність еквіваленціі від кон'юнкції полягає в тому, що незалежно від змісту, рівнозначними є як справжні, так і помилкові висловлювання.

Наприклад. Висловлювання А = «2 + 2 = 7» і В = «1 - 8 = 5». Складне висловлювання А≡ В (А ~ В) істинно, так як обидва висловлювання помилкові.

### Імплікація

Результатом операції імплікації для висловлювання А→У буде брехня тільки тоді, коли перше висловлювання (А) істинно, а друге (В) помилково. При цьому А - передумова, а В - наслідок.

Наприклад. Висловлювання А = «2 + 2 = 4» і В = «1 - 8 = 5». Складне висловлювання А→ В (А ⇒ В) помилково, так як висловлювання А істинно, а В - хибне.

### Антікон'юнкція

Результатом операції антікон'юнкціі для висловлювання А ⏐ У буде брехня тільки тоді, коли обидва висловлювання істинними.

Наприклад. Висловлювання А = «Москва - столиця Росії» і В = «Рим - столиця Італії». Складне висловлювання А⏐ У ложно, так як істинні обидва висловлювання.

### Антідіз'юнкція

Результатом операції антідіз'юнкціі для висловлювання А ↓ У буде істинна тільки тоді, коли обидва висловлювання помилкові.

Наприклад. Висловлювання А = «Рим - столиця Росії» і В = «Москва - столиця Італії». Складне висловлювання А↓ В істинно, так як помилкові обидва висловлювання.

Основними символамиалгебри логіки є:

* пропозіціональние змінні;
* унарна зв'язка ¬ і бінарні зв'язки ∧, ∨, →, ~;
* дужки ().

Змінна, значеннями якої є висловлювання, називається пропозіціональной змінної.

Далі индуктивно вводиться поняття формули, що є формалізацією поняття «складного» висловлювання. До формули алгебри логіки відносять:

* вираз, що складається тільки з пропозіціональной змінної (А1, В, С);
* вираження, що складаються з пропозіціональних формул з'єднаних зв'язками (¬ З, (А1 ∧ А2), (Н1 → Н2)).

Правила скорочення записів в пропозіціональних формулах:

* замість¬ А пишуть ;
* замість А1∧ А2 пишуть А1А2;
* пріоритет застосування зв'язок зростає в наступному порядку

~ → ∨ ∧ ¬

* зовнішні дужки опускаються.
* ;
* .

Для перетворень формул в рівні формули важливу роль в алгебрі логіки грають такі рівності:

1.  (Закон коммутативности).
2.  (Закон асоціативності).
3.  (Закон поглинання).
4.  (Закон дистрибутивности).
5.  (Закон протиріччя).
6.  (Закон виключеного третього);
7.  (Закон зняття подвійного заперечення);
8.  (Закон склеювання);
9.  (Закон де Моргана);
10.  (Закон згортки).

Ці рівності дозволяють істотно спростити запис формул звільненням від зайвих дужок.

Елементарної кон'юнкція називається кон'юнкція декількох змінних, взятих з запереченням або без заперечення, причому серед змінних можуть бути однакові.

Елементарної диз'юнкція називається диз'юнкція кількох змінних, взятих з запереченням або без заперечення, причому серед змінних можуть бути однакові.

### Кон'юнктивна нормальна форма

Будь-яка кон'юнкція елементарних диз'юнкцій називається кон'юнктівной нормальною формою, тобто КНФ.

Досконалої КНФ (СКНФ) називається КНФ, в якій немає рівних елементарних диз'юнкцій і всі вони містять одні й ті ж змінні, причому кожну змінну тільки один раз (можливо з запереченням).

### Диз'юнктивна нормальна форма

Будь-яка диз'юнкція елементарних кон'юнкція називається диз'юнктивній нормальною формою, тобто ДНФ. [8]

Досконалої ДНФ (СДНФ) називається ДНФ, в якій немає рівних елементарних кон'юнкція і всі вони містять одні й ті ж змінні, причому кожну змінну тільки один раз (можливо з запереченням).

# РОЗДІЛ 3 ОПИС ПРОГРАМИ

# 3.1 Вимоги до программи

Потрібно написати програму, яка буде демонструвати роботу побудови логічних схем. Данна программа повинна забезпечувати:

Відкриття збереженних файлів

Збереження у форматі Bitmap

Звичайне збереження

Розроблювати конструкторську модель та редагувати ії

Робити переадресацію у Paint

Очищення

Окрім вищеперелічених можливостей програма повинна мати також зручний інтерфейс для створення користувачеві комфортних умов роботи з програмою.

Вхідна оперативна інформація

дані, що зберігаються в текстовому файлі;

запити користувача.

Вихідна інформація

Інформація, що зберігається в текстовий файл;

Вибирання програмних засобів, середовища реалізації і операційної системи

Для написання даного програмного продукту вибрана мова програмування Delphi пакет Borland Delphi 7 Studio

Інтегроване середовище розробки є многооконной системою. Вона включає в себе все необхідне для швидкої розробки Windows-додатків, і може гнучко налаштовуватися. Результуючі програми добре оптимізовані за швидкістю виконань і витратам пам'яті. Дизайнер форм, інспектор об'єктів і інші засоби залишаються доступними під час роботи програми, тому вносити зміни можна в процесі відладки.

# 3.2 Проектування інтерфейсу програми

Спроектуємо інтерфейс програми. Borland Delphi 7 це робиться за допомогою форм. Форма є екранним об'єктом, зазвичай прямокутної форми, який можна застосовувати для надання інформації користувачеві і для обробки введення інформації від користувача. Саме на ній ми і розташовуватимемо решту всіх елементів інтерфейсу програми.

Логічні операції реалізуються з використанням електричних схем, які називаються логічними елементами. Вони виготовляються у вигляді інтегральних мікросхем в більшості своїй на базі діодів і транзисторів, або тільки транзисторів. У зв'язку з цим розрізняють два типи логічних елементів: ДТЛ (діод-транзисторна логіка) і ТТЛ (транзисторних-транзисторна логіка). На рис. 3.1 представлені схемні позначення елементів, що виконують найпростіші логічні операції. При цьому елементи "АБО" і "І" - двухвходових, тобто виконують операції з двома вхідними змінними. Однак промисловістю випускаються логічні елементи "АБО" і "І" з великим числом вхідних клем.



Рис. 3.1. Схемне позначення логічних елементів

На рис. 3.2 приведена схема на діодах, за допомогою якої реалізується операція логічного додавання. Дійсно, при відсутності на входах цієї схеми сигналів. Все діоди будуть знаходитися в закритому стані і на резисторі падіння напруги дорівнює нулю (F = 0). При подачі хоча б на один вхід схеми сигналу позитивної полярності діод, встановлений на цьому вході, перейде у відкритий стан. Через нього і послідовно включений з ним резистор R буде протікати струм, падіння напруги на резисторі буде відрізнятися від нуля. F = 1. [9]



Рис. 3.2. Рисунок 3.3.

Операція "НЕ" може бути реалізована на біполярному транзисторі типу npn, включеного за паралельним варіанту побудови ключової . При реалізації логічної операції «НЕ» через наявність залишкового напруги на відкритому транзисторі вихідна напруга також відрізняється від нуля. Таким чином, в реальних логічних елементах логічним нулем, фактично, є не нульове, а мала величина напруги. У зв'язку з цим в довідкових даних зазвичай замість «0» вказується «н» (низький рівень напруги) і, відповідно, замість «1» вказується «в» (високий рівень напруги).

Логічні елементи випускаються промисловістю у вигляді серій. Елементи кожної серії, що виконують різні операції, можуть сполучатися в складну схему логічного пристрою. Елементи серії створюються на базі основного елемента, який зазвичай виконує операції "АБО-НІ" або "І-НЕ". Математична форма подання цих операцій має вигляд:

Схеми, що реалізують операції "АБО-НІ" і "І-НЕ", можуть бути отримані при послідовному з'єднанні двох схем, що виконують операції відповідно "АБО" - "НЕ" і "І" - "НЕ", що показано на рис. 3.4, а і 3.5., а для випадку двухвходових елементів. Схемні позначення елементів "АБО-НІ" і "І-НЕ" представлені на рис. 3.4, б і 3.5, б.



Рис. 3.4. Логічна операція «АБО-НЕ»:

а - принцип реалізації операції,

б - схемне позначення логічного елемента



Рис. 3.5. Логічна операція «І-НЕ»:

а - принцип реалізації операції,

б - схемне позначення логічного елемента



Рис. 3.6. Реалізація логічної операції «І-НЕ»

на транзисторах

На рис. 3.6 представлена ​​схема на транзисторах, що реалізує логічну операцію "І-НЕ". Особливістю цієї схеми є використання многоеміттерного транзистора. Яким замінюється діодний частина елемента "І-НЕ", що виконує операцію "І". У многоеміттерного транзисторі емітерний шар складається з декількох областей при загальних базовому і коллекторном шарах. Принцип роботи схеми рис. 3.6 наступний. Якщо на всі еміттерние входи транзистора буде подано позитивне напруга, рівне Е, все емітерний переходи транзистора будуть закриті, а його ток бази буде протікати через відкритий колекторний перехід, обумовлюючи протікання колекторного струму, який одночасно є базовим струмом транзистора Т. транзистор Т, Який виконує операцію логічного заперечення, в цьому випадку буде перебувати у відкритому стані, і на його виході напруга буде практично відсутнім (F = 0). [10] При відсутності хоча б на одному емітерний вході транзистора Т сигналу позитивної полярності, струм, що протікає через резистор R і через відповідний відкритий емітерний перехід, буде замикатися по ланцюгу цього емітерного входу. Базовий струм транзистора Т дорівнюватиме нулю, оскільки сумарний опір колекторного переходу транзистора Т і базового шару транзистора Т виявляється досить великим у порівнянні з опором відкритого емітерного переходу транзистора Т. транзистор Т тому буде перебувати в закритому стані, і на його виході напруга буде близько до величини + Е, Тобто F = 1.



Рис. 3.7. Схеми реалізації логічних операцій на елементах «І-НЕ»

а - операції «НЕ», б - операції «І», в - операції «АБО»

Логічні елементи «АБО-НЕ» і «І-НЕ» називають базовими, оскільки схеми, складені тільки з цих елементів одного типу, дозволяють реалізовувати найпростіші логічні операції. Це неважко довести з використанням формул алгебри логіки, наприклад, на схемах, складених з елементів «І-НЕ», які наведені на рис. 3.7.

Лістинг програми у додатоку А.

# Опис застосування програми

Програмний продукт, написаний в середовищі Borland delphi 7 використанням візуальних компонентів не вимогливий до ресурсів системи. На рисунку 3.8. зоображений основне середовище для побудови логічних схем.

****

Рис. 3.8 Середовище програми

У лівому кутку програми представлені два меню(рис. 3.9)



Рис. 3.9 Меню

Меню «Файл» забезпечує функції, які зображені на рис. 3.10, а саме відкриття файлу, збереження, друк та вихід з програми.



Рис. 3.10 Меню «Файл»

Меню «Конструктор» забезпечує функції, які зображені на рис. 3.11, а саме Редактор, Контроль, Очищення, Налаштування та пере адресація у Paint.



Рис. 3.11 Меню «Конструктор»

Також у лівому кутку програми (рис 3.12) зображений перелік кнопок, які допомагають побудувати логічні схеми, тобто виконують основну задачу.



Рис. 3.12 Клавіши для побудови програми

Перелік доступних клавіш:

1. Перемикач(Вхід)
2. Табло(Вихід)
3. Диз'юнкція(АБО)
4. Кон'юнкція(І)
5. Стрілка Пірса
6. Штрих Шеффера
7. Сума Жегалкина
8. Заперечення(НЕ)
9. Еквівалентність
10. Імплікація
11. Перетин
12. Вузол
13. Поворот
14. Відрізок
15. Гумка

На рис. 3.13. зоображена функціональність програми, тобто побудова логічної схеми.



Рис. 3.13 Приклад побудови логічних схем



Рис. 3.14 Збереження файлу



Рис. 3.15 Збереження файлу у форматі Bitmaps

ВИСНОВКИ

При виконанні проекту були пройдені всі етапи розробки спеціалізованого програмного забезпечення:

формалізація завдання;

збір необхідних початкових даних, використовуваних в програмі;

В процесі виконання роботи була проведена наступна робота:

1. Освоєння принципів розробки алгоритмів побудови логічних схем;

2. Ознайомлення з різними сферами застосування методів побудови логічних схем

3. Розроблена програма яка дозволяє побудувати логічні схеми.

# СПИСОК ЛІТЕРАТУРИ

* + - 1. М.Г. Мнея "Фізичні принципи роботи ЕОМ", 1987 р
			2. www.elcp.ru
			3. www.eltm.ru
			4. rk6.bmstu. ru
			5. Проектування НВІС: Пер. з япон. / Ватамабе М., Асада К., Канн К., Оцуки. Т. - М .: Світ, 1988. - 304 с .;
			6. Автоматизоване проектування цифрових пристроїв / С. С. Барулин, Ю. М. Барнаул, В. Л. Бердишев та ін-М .: Радио и связь, 1981. - 240 с .;
			7. Мілн Б. Удосконалені САПР електронних схем на базі ПЛІС // Електроніка. - 1989, - № 8. - С. 61 - 67;
			8. Micro–Cap III. Third–generation interactive circuit analysis//Byte. – 1989.–Vol. 14. – № 4. – P. 81.;
			9. Разевіг В. Д. Моделювання аналогових електронних пристроїв на персональних ЕВМ.-М .: Изд-во МЕІ, 1992. - 162 с .;
			10. «CAD / CAM / CAE Observer», 2010. -15с.

# ДОДАТОК А

Листинг програми

unit Unit1;

interface

uses

 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

 Dialogs, ExtCtrls, Menus, ImgList, ComCtrls, ToolWin, inifiles, Grids,ShellApi,

 StdCtrls, Buttons, ExtDlgs;

type

 TForm1 = class(TForm)

 ToolbarImages: TImageList;

 MainMenu1: TMainMenu;

 N1: TMenuItem;

 MenuOpen: TMenuItem;

 MenuSaveAs: TMenuItem;

 BMP1: TMenuItem;

 N6: TMenuItem;

 MenuPrint: TMenuItem;

 N9: TMenuItem;

 MenuExit: TMenuItem;

 N3: TMenuItem;

 MenuRed: TMenuItem;

 MenuControl: TMenuItem;

 N2: TMenuItem;

 N4: TMenuItem;

 ImageList1: TImageList;

 paint1: TMenuItem;

 PaintBox1: TPaintBox;

 N8: TMenuItem;

 OpenDialog1: TOpenDialog;

 SaveDialog1: TSaveDialog;

 SavePictureDialog1: TSavePictureDialog;

 Panel1: TPanel;

 SpeedButton1: TSpeedButton;

 SpeedButton2: TSpeedButton;

…

 SpeedButton21: TSpeedButton;

 procedure TForm1.paint1Click(Sender: TObject);

var cod:byte;

begin

 cod:=WinExec(Pchar('mspaint.exe'), SW\_SHOWNORMAL);

 case cod of

 0: ShowMessage('Не хватает ресурсов');

 2: ShowMessage('Файл не найден');

 3: ShowMessage('Путь не найден');

 end;

end;

procedure recurs(i,j,ruld: Integer);

begin

 if (ruld = 1) then begin

 a[i,j].l:= a[i,j-1].r;

 if((a[i,j].inf = 3) or (a[i,j].inf = 4)) then

 begin

 if (a[i,j].l = 0) then a[i,j].inf:= 4;

 if (a[i,j].l = 1) then a[i,j].inf:= 3;

 end;

 if (a[i,j].inf = 9) then

 begin

 if (a[i,j].l = 0) then a[i,j].r:= 1;

 if (a[i,j].l = 1) then a[i,j].r:= 0;

 if (j+1 < n) then recurs(i, j+1, 1);

 end;

 if(a[i,j].inf=13) or (a[i,j].inf=14) or (a[i,j].inf=15)or (a[i,j].inf=16)then begin

 if(a[i,j].l=0) then

 a[i,j].r:=0;

 if(a[i,j].l=1)then

 a[i,j].r:=1;

 if((a[i,j].l=1) or (a[i,j].r=1))and((a[i,j].u=1) or (a[i,j].d=1)) then

 a[i,j].inf:=16;

 if((a[i,j].l=0) or (a[i,j].r=0))and((a[i,j].u=1) or (a[i,j].d=1)) then

 a[i,j].inf:=15;

 if((a[i,j].l=1) or (a[i,j].r=1))and((a[i,j].u=0) or (a[i,j].d=0))then

 a[i,j].inf:=14;

 if((a[i,j].l=0) or (a[i,j].r=0))and((a[i,j].u=0) or (a[i,j].d=0))then

 a[i,j].inf:=13;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n)) then

 recurs(i,j+1,1);

 end;

 if((a[i,j].inf=21) or (a[i,j].inf=22)) then begin//право-вниз работает

 a[i,j].d:=a[i,j].l;

 if(a[i,j].l=0) then

 a[i,j].inf:=21;

 if(a[i,j].l=1) then

 a[i,j].inf:=22;

 if((a[i+1,j].u<>a[i,j].d) and (i+1<n))then

 recurs(i+1,j,4);

 end;

 if((a[i,j].inf=23) or (a[i,j].inf=24))then begin//право-вверх работает

 a[i,j].u:=a[i,j].l;

 if(a[i,j].l=0)then

 a[i,j].inf:=23;

 if(a[i,j].l=1)then

 a[i,j].inf:=24;

 if((a[i-1][j].d<>a[i,j].u) and (i-1>=0)) then

 recurs(i-1,j,2);

 end;

 if((a[i,j].inf=25) or (a[i,j].inf=26))then begin //верт-гориз работает

 a[i,j].u:=a[i,j].l;

 a[i,j].r:=a[i,j].l;

 a[i,j].d:=a[i,j].l;

 if(a[i,j].l=0) then

 a[i,j].inf:=25;

 if(a[i,j].l=1)then

 a[i,j].inf:=26;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n))then

 recurs(i,j+1,1);

 if((a[i+1,j].u<>a[i,j].d) and (i+1<n))then

 recurs(i+1,j,4);

 if((a[i-1,j].d<>a[i,j].u) and (i-1>=0))then

 recurs(i-1,j,2);

 end;

 if((a[i,j].inf=29) or (a[i,j].inf=30))then begin//гориз-вниз работает

 a[i,j].r:=a[i,j].l;

 a[i,j].d:=a[i,j].l;

 if(a[i,j].l=0)then

 a[i,j].inf:=29;

 if(a[i,j].l=1)then

 a[i,j].inf:=30;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n))then

 recurs(i,j+1,1);

 if((a[i+1,j].u<>a[i,j].d) and (i+1<n)) then

 recurs(i+1,j,4);

 end;

 if((a[i,j].inf=31) or (a[i,j].inf=32))then begin //верт-лево работает

 a[i,j].u:=a[i,j].l;

 a[i,j].d:=a[i,j].l;

 if(a[i,j].l=0)then

 a[i,j].inf:=31;

 if(a[i,j].l=1)then

 a[i,j].inf:=32;

 if((a[i+1,j].u<>a[i,j].d) and (i+1<n)) then

 recurs(i+1,j,4);

 if((a[i-1,j].d<>a[i,j].u) and (i-1>=0))then

 recurs(i-1,j,2);

 end;

 if((a[i,j].inf=33) or (a[i,j].inf=34)) then begin//гориз-вверх работает

 a[i,j].u:=a[i,j].l;

 a[i,j].r:=a[i,j].l;

 if(a[i,j].l=0)then

 a[i,j].inf:=33;

 if(a[i,j].l=1)then

 a[i,j].inf:=34;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n))then

 recurs(i,j+1,1);

 if((a[i-1,j].d<>a[i,j].u) and (i-1>=0))then

 recurs(i-1,j,2);

 end;

 if((a[i,j].inf=35) or (a[i,j].inf=36))then begin//горизонтальная работает

 a[i,j].r:=a[i,j].l;

 if(a[i,j].l=0)then

 a[i,j].inf:=35;

 if(a[i,j].l=1)then

 a[i,j].inf:=36;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n))then

 recurs(i,j+1,1);

 end;

 end;

 if (ruld=2) then begin

 a[i,j].d:=a[i+1,j].u;

 if(a[i,j].inf=5) then

 begin

 if(a[i,j].u=1) and (a[i,j].d=1)then

 a[i,j].r:=1

 else

 a[i,j].r:=0;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1))then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=6)then begin

 if((a[i,j].u=0) and (a[i,j].d=0))then

 a[i,j].r:=0

 else if(a[i,j].d<>-1) then

 a[i,j].r:=1;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1))then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=7) then begin

 if((a[i,j].u=1) and (a[i,j].d=0)) then

 a[i,j].r:=0

 else if(a[i,j].d<>-1) then

 a[i,j].r:=1;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1))then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=8)then begin

 if((a[i,j].u)=(a[i,j].d)) then

 a[i,j].r:=1

 else if(a[i,j].d<>-1) then

 a[i,j].r:=0;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1)) then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=10) then begin//стр Пирса

 if((a[i,j].u=0) and (a[i,j].d=0))then

 a[i,j].r:=1

 else if(a[i,j].d<>-1) then

 a[i,j].r:=0;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1)) then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=11) then begin

 if(a[i,j].u=a[i,j].d) then

 a[i,j].r:=0

 else if(a[i,j].d<>-1) then

 a[i,j].r:=1;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1)) then

 recurs(i,j+1,1);

 end;

 if(a[i,j].inf=12) then begin

 if((a[i,j].u=1) and (a[i,j].d=1)) then

 a[i,j].r:=0

 else if(a[i,j].d<>-1) then

 a[i,j].r:=1;

 if((a[i,j+1].l<>a[i,j].r) and (j+1<n) and (a[i,j].r<>-1)) then

 recurs(i,j+1,1);

 end;

 if((a[i,j].inf=13) or (a[i,j].inf=14) or (a[i,j].inf=15) or (a[i,j].inf=16)) then begin

 a[i,j].u:=a[i,j].d;

 if((a[i,j].l=1) or (a[i,j].r=1))and((a[i,j].u=1) or (a[i,j].d=1)) then

 a[i,j].inf:=16;

 if((a[i,j].l=0) or (a[i,j].r=0))and((a[i,j].u=1) or (a[i,j].d=1))then

 a[i,j].inf:=15;

 if((a[i,j].l=1) or (a[i,j].r=1)) and ((a[i,j].u=0) or (a[i,j].d=0)) then

 a[i,j].inf:=14;

 if((a[i,j].l=0) or (a[i,j].r=0)) and ((a[i,j].u=0) or (a[i,j].d=0)) then

 a[i,j].inf:=13;

 if((a[i-1,j].d<>a[i,j].u) and (i-1>=0))then

 recurs(i-1,j,2);

 end;

 if((a[i,j].inf=19) or (a[i,j].inf=20)) then begin

 a[i,j].r:=a[i,j].d;

 if(a[i,j].d=1) then

 a[i,j].inf:=20;

 if(a[i,j].d=0) then

 a[i,j].inf:=19;

 if((a[i,j].r<>a[i][j+1].l) and (j+1<n)) then

 recurs(i,j+1,1);

 end;

 if((a[i,j].inf=21) or (a[i,j].inf=22)) then begin//вверх-влево

 a[i,j].l:=a[i,j].d;

 if(a[i,j].d=1) then

 a[i,j].inf:=22;

 if(a[i,j].d=0) then

 a[i,j].inf:=21;

 if((a[i,j].l<>a[i,j-1].r) and (j-1>=0))then

 recurs(i,j-1,3);

 end;

 if((a[i,j].inf=25) or (a[i,j].inf=26)) then begin

 a[i,j].u:=a[i,j].d;

 a[i,j].l:=a[i,j].d;

 a[i,j].r:=a[i,j].d;

 if(a[i,j].d=1)then

 a[i,j].inf:=26;

 if(a[i,j].d=0)then

 a[i,j].inf:=25;

 if((a[i,j].l<>a[i,j-1].r)and (j-1>=0)) then

 recurs(i,j-1,3);

 if((a[i,j].r<>a[i,j+1].l) and (j+1<n))then

 recurs(i,j+1,1);

 if((a[i,j].u<>a[i-1,j].d) and (i-1>=0))then

 recurs(i-1,j,2);

 end;

…

procedure paint;

var

 i,j: Integer;

begin

 pic.Canvas.Brush.Color:=RGB(255,255,255);

 pic.Canvas.FillRect(Rect(0,0,pic.Width,pic.Height));

 pic.Canvas.Brush.Color:=RGB(0,0,0);

 for i:=0 to n-1 do

 begin

 for j:=0 to n-1 do

 begin

 if(proverka=false) then

 begin

 pic.Canvas.Pen.Color:=RGB(200,200,200);

 pic.Canvas.Brush.Color:=RGB(255,255,255);

 pic.Canvas.Rectangle(Rect(x0+j\*h0-1,y0+i\*h0-1,x0+(j+1)\*h0,y0+(i+1)\*h0));

 end;

 if ( (a[i,j].inf >0) and (a[i,j].inf <39 ) ) then

 pic.Canvas.Draw(x0+j\*h0,y0+i\*h0,element[a[i,j].inf].p);

 end;

 end;

 Form1.PaintBox1.Canvas.Draw(0,0,pic);

end;

procedure TForm1.PaintBox1MouseDown(Sender: TObject; Button: TMouseButton;

 Shift: TShiftState; X, Y: Integer);

var

 k, l: Integer;

begin

 x1:= X;//x1=X;

 y1:= Y;//y1=Y;

 if (Button = mbRight) then

 begin

 mdown:= True;

 PaintBox1.Cursor:= crHandPoint;

 end;

 if( (Button = mbLeft) and (proverka = False) ) then

 begin

 mdown2:= True;

 k:= (Y-y0)div h0;

 l:= (X-x0) div h0;

 a[k,l].inf:= pero;

 end;

procedure TForm1.FormPaint(Sender: TObject);

begin

 unit1.paint;

end;

procedure TForm1.SpeedButton19Click(Sender: TObject);

begin

 pero:=12;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton19.Hint;

end;

procedure TForm1.SpeedButton5Click(Sender: TObject);

begin

 pero:=8;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton5.Hint;

end;

procedure TForm1.SpeedButton3Click(Sender: TObject);

begin

 pero:=9;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton3.Hint;

end;

procedure TForm1.SpeedButton10Click(Sender: TObject);

begin

 pero:=11;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton10.Hint;

end;

procedure TForm1.SpeedButton4Click(Sender: TObject);

begin

 pero:=7;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton4.Hint;

end;

procedure TForm1.SpeedButton8Click(Sender: TObject);

begin

 pero:=13;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton8.Hint;

end;

procedure TForm1.SpeedButton15Click(Sender: TObject);

begin

 pero:=25;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton15.Hint;

end;

procedure TForm1.SpeedButton12Click(Sender: TObject);

begin

 pero:=19;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton12.Hint;

end;

procedure TForm1.SpeedButton14Click(Sender: TObject);

begin

 pero:=21;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton14.Hint;

end;

procedure TForm1.SpeedButton11Click(Sender: TObject);

begin

 pero:=17;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton11.Hint;

end;

procedure TForm1.SpeedButton13Click(Sender: TObject);

begin

 pero:=23;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton13.Hint;

end;

procedure TForm1.SpeedButton20Click(Sender: TObject);

begin

 pero:=33;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton20.Hint;

end;

procedure TForm1.SpeedButton16Click(Sender: TObject);

begin

 pero:=27;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton16.Hint;

end;

procedure TForm1.SpeedButton18Click(Sender: TObject);

begin

 pero:=31;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton18.Hint;

end;

procedure TForm1.SpeedButton17Click(Sender: TObject);

begin

 pero:=29;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton17.Hint;

end;

procedure TForm1.SpeedButton22Click(Sender: TObject);

begin

 pero:=35;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton22.Hint;

end;

procedure TForm1.SpeedButton23Click(Sender: TObject);

begin

 pero:=37;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton23.Hint;

end;

procedure TForm1.SpeedButton21Click(Sender: TObject);

begin

 pero:=0;

 statusbar1.Panels[1].Text:='Используемый элемент: '+SpeedButton21.Hint;

end;

procedure TForm1.MenuControlClick(Sender: TObject);

var i,j:integer;

begin

 proverka:=true;

 for i:=0 to n-1 do begin

 for j:=0 to n-1 do begin

 a[i,j].r:=0;

 a[i,j].u:=0;

 a[i,j].l:=0;

 a[i,j].d:=0;

 end;

 end;

 proverka1;

end;

procedure TForm1.MenuRedClick(Sender: TObject);

begin

 proverka:=false;

 paint();

end;

procedure TForm1.N8Click(Sender: TObject);

var i,j:integer;

begin

 for i:=0 to n-1 do begin

 for j:=0 to n-1 do begin

 a[i,j].inf:=0;

 a[i,j].r:=-1;

 a[i,j].u:=-1;

 a[i,j].l:=-1;

 a[i,j].d:=-1;

 end;

 end;

 paint();

end;

procedure TForm1.MenuSaveAsClick(Sender: TObject);

begin

 SaveDialog1.Execute;

end;

procedure TForm1.MenuOpenClick(Sender: TObject);

begin

 OpenDialog1.Execute;

end;

procedure TForm1.OpenDialog1CanClose(Sender: TObject;

 var CanClose: Boolean);

var

 str: String;

 i: Integer;

 j: Integer;

 InputFile: File of Integer;

begin

 str:= OpenDialog1.FileName;

 if str <> '' then

 begin

 AssignFile(InputFile,str);

 Reset(InputFile);

 for i := 0 to n - 1 do

 for j := 0 to n - 1 do

 begin

 a[i,j].inf:= 0;

 a[i,j].r:= 0;

 a[i,j].u:= 0;

 a[i,j].l:= 0;

 a[i,j].d:= 0;

 Read(InputFile, a[i,j].inf);

 end;

 CloseFile(InputFile);

 end;

 unit1.paint;

end;

procedure TForm1.Timer1Timer(Sender: TObject);

begin

 StatusBar1.Panels[2].Text:=DateToStr(now);

end;

procedure TForm1.SaveDialog1CanClose(Sender: TObject;

 var CanClose: Boolean);

var

 str: String;

 i: Integer;

 j: Integer;

 OutputFile: File of Integer;

begin

 str:= SaveDialog1.FileName;

 AssignFile(OutputFile, str);

 Rewrite(OutputFile);

 for I := 0 to n - 1 do

 for j := 0 to n - 1 do

 Write(OutputFile,a[i,j].inf);

 CloseFile(OutputFile);

end;

end.