Форма №Н-6.01

Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота за освітнім ступенем “магістр” на тему: Системи автоматизованого проектування напівпровідникових структур

Виконав: магістрант 2 курсу
групи М-КН-21
спеціальності 122 “Комп’ютерні науки”
Самолюк Віталій Петрович
Керівник: кандидат фізико-математичних наук, доцент Ігор Петрович Мороз

Рівне-2024

ЗМІСТ

ВСТУП

Сучасний розвиток напівпровідникової технології надає неабияке значення впровадженню ефективних та високотехнологічних систем автоматизованого проектування напівпровідникових структур. У цьому контексті виникає ряд актуальних питань, що визначають основні напрямки наукового дослідження в даній області.

Актуальність роботи визначається необхідністю оптимізації процесів проектування напівпровідникових структур в умовах стрімкого технологічного розвитку. Зокрема, виникає питання, як впровадження систем автоматизованого проектування може відповісти на виклики сучасної напівпровідникової технології. Отже, дана робота націлена на вирішення важливих питань у сфері автоматизованого проектування напівпровідникових структур, що визначає її актуальність та важливість в контексті сучасних технологічних викликів.

Метою дослідження є системний аналіз програмних комплексів, призначених для проектування та оптимізації характеристик напівпровідникових структур. Досягнення цієї мети сприятиме подальшому розвитку напівпровідникової технології та прискорить впровадження нових рішень у виробництво.

Об'єктом дослідження є програмні комплекси автоматизованого проектування виступають напівпровідникові структури, які визначають основні характеристики та параметри для автоматизованого проектування. Акцент робиться на тих аспектах, які мають ключове значення для високоефективного проектування.

Предмет дослідження – виявлення принципів побудови відповідних програмних комплексів**.** Дана робота спрямована на аналіз та вдосконалення систем автоматизованого проектування, зокрема, враховуючи конкретні аспекти, які впливають на точність та швидкість проектування

напівпровідникових структур. Основний фокус робиться на параметрах та властивостях, які є критичними для успішного впровадження систем.

Методи дослідження: методи системного аналізу виявлення функцій складу та структури програмного комплексу для проектування технічних систем.

Наукова новизна роботи полягає в виявленні ключових підходів та інноваційних методів у сфері автоматизованого проектування технічних систем напівпровідникових структур. Це відображається в удосконаленні існуючих систем та введенні нових концепцій, що розширюють можливості проектування.

Практична значущість. Отримані результати нашого дослідження можуть бути використані в індустрії напівпровідників для покращення якості та швидкості проектування, що в свою чергу призведе до прискорення виробництва та введення нових продуктів на ринок. Це має потенціал значно вплинути на розвиток галузі та забезпечити конкурентні переваги.

Ключові тренди:

1. Нанотехнології стосуються проектування, розробки та виготовлення структур і пристроїв на нанорозмірі в термінах кількох нанометрів.
2. Високотехнологічного обладнання.
3. Сучасні напівпровідникові чіпи виготовляються шляхом поєднання різних технологій, що робить їх потужними, ефективними та інтелектуальними. Кілька технологій відіграють важливу роль у розробці мікросхем, ось кілька прикладів останніх тенденцій:

- Штучний інтелект і машинне навчання: інтеграція технологій штучного інтелекту та машинного навчання, таких як прискорювачі штучного інтелекту та блоки нейронної обробки (NPU), у сучасну цифрову епоху допомагає підвищити ефективність та інтелект напівпровідникових пристроїв і систем.
- IoT і периферійні обчислення: Інтернет речей (IoT) і периферійні обчислення дозволяють нам розробляти системи в режимі реального

часу з високою обробкою даних із високошвидкісним підключенням до Інтернету та низькою затримкою передачі даних.

- Інтеграція 5G: П’яте покоління Інтернет-світу (5G) приносить прибуток завдяки високій швидкості, низькій затримці та передачі даних у реальному часі у великій мережі підключених пристроїв. Він включає мікрохвильові мікросхеми, антени та ефективні мережі.
- Квантові обчислення: це майбутнє обчислень із високою обчислювальною потужністю та енергоефективністю. Але він все ще знаходиться в стадії розробки та експериментів.

4. У зв’язку зі збільшенням кількості кібератак і витоків даних компанії-виробники напівпровідників розробляють апаратні засоби безпеки для захисту даних і інформації від кіберзагроз і витоку даних. Вони постійно розробляють ефективні рішення для захисту пристрою від несанкціонованого доступу.

5. Дослідники, науковці та інженери роблять усе можливе, щоб зменшити труднощі, складність, проектування та виробництво.

6. Автоматизоване проектування (САПР): ознайомтеся з інструментами та програмним забезпеченням САПР, наприклад SPICE. Вивчіть основи проектування схем, їх моделювання та аналізу за допомогою програмного забезпечення САПР.

7. Програмування та кодування: вивчайте основні поняття програмування та кодування. Наприклад, структури даних і алгоритми, оператори, цикли, умовні оператори, як-от if-else тощо [41].

РОЗДІЛ 1. ВСТУП ДО АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

1.1. Поняття інженерного проектування. Системний підхід до проектування.

Проектування технічного об'єкта — це процес створення (або представлення) образу майбутнього об'єкта, що проектується, у встановленій формі [1]. Цей процес розпочинається з визначення суспільної потреби у нових технічних засобах. Початковий опис об'єкта здійснюється в основному документі під назвою «технічне завдання» (ТЗ). Кінцевим результатом проектування є повний комплект технічної документації, що стосується об'єкта.

Процес проектування включає розроблення технічної пропозиції та/або технічного завдання. У ході перетворення початкового опису на завершений проект створюються проміжні описи, які використовуються для ухвалення проектних рішень.

Проектування, у якому рішення або їх частини приймаються в результаті взаємодії людини та ЕОМ, називається автоматизованим. Воно відрізняється від ручного (де ЕОМ не використовується) або автоматичного (де участь людини відсутня на проміжних етапах). Система, що забезпечує автоматизоване проектування, отримала назву САПР (у міжнародній практиці — CAD System, Computer Aided Design System).

У сучасному світі об'єктами проектування зазвичай є складні системи. Їхня складність може полягати у великій кількості компонентів, складних механізмах взаємодії між ними або використанні математичних моделей високого рівня. Проектування таких систем базується на використанні методів і принципів, які описані у ряді теоретичних концепцій. Найбільш універсальним є системний підхід [2–4], принципи якого впроваджуються у різних методиках проектування складних об'єктів.

1.1.1 Принципи системного підходу

Основи проектування складних систем (зокрема спеціалізованих САПР) закладені в системному підході. Для інженера-системотехніка ці принципи є зрозумілими й природними, але їх застосування може супроводжуватися труднощами, що виникають через специфіку проектування.

Ключовий принцип системного підходу полягає у розгляді складної системи з урахуванням взаємодії її компонентів. Цей підхід охоплює визначення структури системи, типізацію зв'язків, атрибутивний аналіз та оцінку впливу зовнішнього середовища.

Системний підхід розглядається як один із напрямків наукового пізнання. Теорія систем формалізує положення цього підходу, зосереджуючись на дослідженні та проектуванні складних, іноді недостатньо структурованих систем.

У технічній галузі аналогом теорії систем є системотехніка, яка займається організацією процесу створення, використання та вдосконалення технічних систем, а також методами й принципами їх проектування.

Системи автоматизованого проектування та управління належать до найскладніших штучно створених систем. Їхнє проектування неможливе без використання системного підходу.

1.1.2. Основні поняття системотехніки

У теорії систем і системотехніці було введено низку термінів, серед яких ключовими є такі поняття:

Система — сукупність елементів, що взаємодіють між собою через встановлені зв’язки.

Елемент — складова частина системи, яку недоцільно подрібнювати на менші частини в межах проектування.

Складна система — система, що має велику кількість елементів та значну кількість зв’язків між ними. Її складність визначається також такими

властивостями, як цілеспрямованість, цілісність, розчленованість, ієрархічність та багатоаспектність.

Підсистема — частина системи, яка сама має властивості системи.

Надсистема — система, в яку включено розглядувану як підсистему.

Структура — узагальнене представлення елементів системи та їх взаємозв'язків. Відмінність структури від системи полягає у відсутності конкретизації параметрів елементів і зв'язків.

Параметр — величина, що описує властивості системи, її частин або середовища, у якому вона функціонує. У моделях систем параметри є постійними протягом дослідження. Параметри поділяють на зовнішні, внутрішні та вихідні, які виражають відповідно властивості середовища, елементів і самої системи. Вектори параметрів визначаються як $X = (x_1, x_2, ..., x_n)$, $Y = (y_1, y_2, ..., y_m)$, $Q = (q_1, q_2, ..., q_k)$.

Фазова змінна — величина, що відображає енергетичний або інформаційний стан елемента чи підсистеми.

Стан — набір значень фазових змінних у певний момент часу в процесі функціонування.

Поведінка системи (динаміка) — зміни стану системи в часі.

Система без наслідків (без післядії) — система, стан якої у час $t > t_0$ залежить лише від стану у момент t_0 та вектору зовнішніх впливів $Q(t)$. У системах з наслідками також враховується історія станів до t_0.

Вектор змінних стану — мінімальний набір фазових змінних, який дозволяє визначити подальшу поведінку автономної системи без післядії.

Простір станів — множина всіх можливих значень змінних стану.

Фазова траєкторія — графічне представлення процесу змін стану системи у просторі станів.

До характеристик складних систем належать:

- Цілеспрямованість — здатність штучної системи виконувати певну функцію, яка слугує основою для оцінки її ефективності.

- Цілісність — залежність між вихідними параметрами системи та параметрами її елементів, що формує нову якість, відмінну від простої суми параметрів.
- Ієрархічність — можливість представлення системи у вигляді кількох рівнів, між якими існують відносини «ціле—частина».

Системотехніка охоплює:

- проектування ієрархічних структур систем;
- аналіз і моделювання систем;
- синтез і оптимізацію систем.

Моделювання охоплює:

1. Створення моделей складних систем (modeling).
2. Аналіз властивостей систем через їх моделі (simulation).

Синтез розподіляється на:

1. Розробку структури системи (структурний синтез).
2. Вибір параметрів елементів (параметричний синтез).

Процеси моделювання та оптимізації враховують статистичну природу систем. Детерміновані моделі є окремим випадком. У проектуванні часто присутні невизначеність вихідних даних та умов прийняття рішень. Статистичне моделювання базується на методі Монте-Карло, а для прийняття рішень використовують нечіткі множини, експертні системи та еволюційні алгоритми.

1.2. Структура процесу проектування

1.2.1. Ієрархічна структура проектних специфікацій та ієрархічні рівні проектування

Під час блочно-ієрархічного підходу до проектування система розділяється на ієрархічні рівні. На верхньому рівні використовується загальне представлення системи, тоді як деталізація зростає на наступних рівнях.

Розрізняють системний, макро- та мікро- рівні проектування для розв'язання загальних задач та окремих їх деталей відповідно. Розрізняють низхідне, висхідне і змішане проектування в залежності від послідовності розв'язання завдань на різних ієрархічних рівнях. Використовують розділення уявлень про об'єкти на аспекти, такі як функціональний, інформаційний, структурний та поведінковий.

Невизначеність та нечіткість вихідних даних обумовлюють потребу в прогнозуванні та ітеративності під час проектування.

Визначається, що список ієрархічних рівнів може бути специфічним для різних застосувань, і наводяться приклади різних галузей, де використовують різні терміни для рівнів. Виділення страт може бути багатозначним, і розглядаються різні підходи до класифікації страт, такі як функціональне, конструкторське, алгоритмічне та технологічне проектування.

1.2.2. Етапи проектування

Етапи проектування можна розглядати як ключові складові процесу створення технічного об'єкта, що реалізуються послідовно у часі. До основних етапів належать:

- Науково-дослідницькі роботи (НДР), також відомі як передпроектні дослідження чи етап технічної пропозиції.
- Ескізний проект або дослідно-конструкторські роботи.
- Технічний проект.
- Робочий проект.

- Випробування експериментальних зразків або експериментальних партій.

На кожному наступному етапі деталізація проекту та рівень його опрацювання підвищуються. Завершальний етап, робочий проект, має забезпечити повну готовність для виготовлення експериментальних або серійних зразків.

Проектні процедури — це складові частини кожного етапу проектування. Вони включають такі дії, як підготовка деталізованих креслень, кінематичний аналіз, моделювання перехідних процесів, оптимізація параметрів та інші спеціалізовані задачі. Кожна процедура, у свою чергу, може бути розділена на окремі елементи, відомі як проектні операції. Наприклад, під час аналізу міцності методом кінцевих елементів проектні операції можуть включати побудову сітки, вибір або розрахунок зовнішніх впливів, виконання моделювання напруг та деформацій, а також представлення результатів у текстовому чи графічному форматах.

Процес проектування, таким чином, являє собою виконання послідовності проектних процедур, що формують так звані маршрути проектування.

Іноді проектування поділяють на:

- Зовнішнє проектування, яке включає розробку технічного завдання (ТЗ).

- Внутрішнє проектування, що передбачає реалізацію вимог ТЗ.

Такий підхід допомагає чітко структурувати етапи розробки і забезпечує систематичність у виконанні задач.

1.3. Розробка автоматизованих систем проектування (САПР)

Розробка автоматизованих систем проектування (САПР) — це область інженерії, яка використовує комп'ютерні технології для полегшення та оптимізації процесів проектування в різних галузях, таких як

машинобудування, архітектура, електроніка, електротехніка і т.д. Основні аспекти розробки САПР включають:

1. Моделювання та Проектування:

САПР надає інструменти для створення віртуальних моделей об'єктів або систем, які можуть бути у подальшому використані для аналізу та оптимізації.

2. Графічний Інтерфейс:

Розробка зручного інтерфейсу для введення та взаємодії з користувачем, щоб спростити процес введення даних та взаємодії з програмою.

3. База Даних та Управління Даними:

САПР зазвичай включає у себе системи управління базами даних для ефективного зберігання та організації інформації про проект.

4. Аналіз та Симуляція:

Включення інструментів для проведення аналізу та симуляцій різних аспектів проекту, таких як стрес-тестування, теплові аналізи, електричні симуляції тощо.

5. Генерація Документації:

Автоматизована генерація технічної документації, креслень, звітів та іншої необхідної документації для проекту.

6. Інтеграція з Іншими Інструментами:

Можливість інтеграції САПР з іншими інженерними інструментами та системами, що використовуються в організації.

7. Автоматизація Завдань:

САПР може автоматизувати рутинні завдання, що сприяє збільшенню ефективності проектування та зменшенню ймовірності помилок.

8. Керування Версіями та Конфігурацією:

Системи контролю версій для відстеження змін та управління конфігураціями проектів.

9. Багатокористувацька Підтримка:

Можливість спільної роботи над проектами для команд інженерів.

10. Безпека та Захист Даних:

Забезпечення безпеки інформації, особливо важливо для конфіденційних проектів.

При розробці САПР важливо враховувати специфіку галузі та вимоги конкретного проекту. Використання САПР може значно покращити якість та ефективність інженерних проектів, скоротити час розробки та вартість виготовлення.

Забезпечення САПР включає: теорію процесів, що відбуваються в схемах і конструкціях; методи аналізу та синтезу конструкцій, систем та їх складових частин, їх математичні моделі; математичні методи та алгоритми чисельного розв'язання систем рівнянь, що описують конструкції. Зазначені компоненти становлять ядро САПР. У забезпечення САПР входять також алгоритмічні спеціальні мови програмування, термінологія, нормативи, стандарти та інші дані. Розробка комплексу забезпечення САПР потребує спеціальних знань у сферах застосування САПР. Отже, розробка забезпечення САПР – прерогатива фахівців у предметній галузі.

У забезпеченні САПР виділяються наступні відокремлені блоки:

Математичне забезпечення (МЗ) включає набір математичних методів, моделей та алгоритмів, які використовуються для проектування, представлених у конкретній формі. При автоматизованому проектуванні МЗ не застосовується безпосередньо, а використовуються компоненти, що базуються на його основі, зокрема програмне забезпечення. Однак розробка МЗ є одним з найскладніших етапів створення САПР, оскільки саме від нього залежить продуктивність та ефективність роботи системи при однакових технічних засобах.

Математичне забезпечення САПР поділяється на дві основні частини, залежно від їх функціонального призначення та способів реалізації. Перша частина включає математичні методи та побудовані на їх основі моделі, що описують об'єкти проектування або обчислюють необхідні властивості та параметри цих об'єктів. Друга частина полягає у формалізованому описі технології автоматизованого проектування.

У САПР обидві частини МЗ повинні органічно взаємодіяти між собою. Способи реалізації першої частини є найменш стандартними і залежать від специфіки проектного процесу. Постійно вдосконалюються методи цієї частини, особливо щодо оптимізації проектування.

Друга частина МЗ — це формалізація всієї технології автоматизованого проектування. Це складніше завдання, ніж алгоритмізація окремих етапів проектування, оскільки потребує формалізації всієї логіки проектного процесу, зокрема взаємодії між проектувальниками та використанням автоматизації. Такі проблеми вирішуються, здебільшого, методом експериментів і проб.

Таким чином, математичне забезпечення САПР має забезпечувати опис взаємозв'язку між об'єктами проектування, процесами та засобами автоматизації.

Технічне забезпечення (ТЗ) САПР включає комплекс технічних засобів, які взаємодіють для забезпечення її функціонування. До технічного забезпечення належать обчислювальні пристрої, організаційна техніка, засоби передачі даних, вимірювальні пристрої, а також засоби підготовки даних і організації архівів. Більшість сучасних САПР базується на локальних обчислювальних мережах.

Програмне забезпечення (ПЗ) САПР складається з машинних програм, необхідних для виконання процесу проектування, і включає системне та прикладне ПЗ. САПР містить:

- загальносистемне ПЗ (операційні системи та моніторинг САПР);
- пакети прикладних програм, орієнтовані на вирішення конкретних завдань у певних галузях;
- системи програмування, що включають інструменти для написання, трансляції та налагодження програм.

Інформаційне забезпечення (ІЗ) САПР охоплює необхідну для проектування інформацію, включаючи систему управління базами даних (СУБД), бази даних і бази знань. Вимоги до ІЗ:

1. Адекватність відомостей про стан предметної області.
2. Масовість доступу (колективна робота).
3. Швидкодія (швидкість відповіді на запит).
4. Продуктивність (кількість оброблених запитів за одиницю часу).
5. Розширюваність.
6. Надійність та захист інформації.

ІЗ САПР включає опис стандартних процедур проектування, типових рішень, елементів, комплектуючих виробів та їх моделей, числових значень параметрів та інших даних. Вся інформація зберігається у закодованій формі на машинних носіях. Також у ІЗ містяться правила та норми проектування, що визначаються нормативно-технічною документацією.

У базі даних (БД) можна виділити ключові частини, які мають різні функції у процесі проектування: довідники, архіви тощо. Довідники містять інформацію про ГОСТи, нормативи, уніфіковані елементи та раніше виконані проекти. Проектні дані поповнюються в міру розвитку проекту на різних етапах.

Лінгвістичне забезпечення (ЛЗ) САПР охоплює мови проектування, терміни, визначення, правила формалізації природних мов та методи обробки текстів. ЛЗ поділяється на мови програмування, проектування та управління.
- Мови програмування призначені для розробки системного та прикладного ПЗ. Вони ґрунтуються на алгоритмічних мовах і використовуються для формулювання алгоритмів.
- Мови проектування є спеціалізованими для обміну інформацією між користувачем і комп'ютером щодо об'єктів і процесів проектування.
- Мови управління служать для створення команд, які керують технічними засобами та периферійними пристроями.

Методичне забезпечення (МетЗ) складається з документів, що визначають правила використання засобів САПР. Організаційне забезпечення (ОЗ) включає документи, що встановлюють структуру проектної організації,

взаємозв'язки між підрозділами та порядок подання та розгляду проектної документації.

Для нормального функціонування САПР необхідна взаємодія всіх компонентів забезпечення. Технічне та програмне забезпечення в сукупності утворюють інструмент для проектування, який часто позначають як програмно-методичний комплекс (ПЗ + МетЗ) та програмно-технічний комплекс (ПЗ + ТЗ).

1.4. Зміст технічних завдань на проектування

У технічних завданнях на проектування об'єкта повинні бути зазначені, принаймні, такі дані:

1. Призначення об'єкта.

2. Умови експлуатації. Окрім якісних характеристик, які описуються в текстовій формі, необхідно вказати числові параметри, що відомі як зовнішні параметри, для яких визначено допустимі межі значень. Прикладами зовнішніх параметрів є температура навколишнього середовища, зовнішні сили, електричні напруги, навантаження тощо.

3. Вимоги до вихідних параметрів, тобто величин, що визначають характеристики об'єкта, важливі для споживача. Ці вимоги сформульовані у вигляді умов функціональності:

$$y_i R T_i,$$

де y_i - і-й вихідний параметр; $R \in \{=, <, >, \leq, \geq\}$ - вид відношення; T_i - норма і-го вихідного параметра. У випадку R = (дорівнює), необхідно визначити потрібну точність виконання рівності.

1.5. Класифікація моделей та параметрів, що використовуються при автоматизованому проектуванні

У процесі автоматизованого проектування, коли об'єкт ще не існує в реальності, робота проводиться з певним квазіоб'єктом-моделлю, що

відображає властивості об'єкта, які є актуальними для дослідження. Така модель може бути у вигляді фізичного об'єкта (макет, стенд) або специфікації. Відмінність моделей-специфікацій може бути у вигляді функціональних, поведінкових, інформаційних та структурних моделей (описів). Якщо ці моделі формалізовані математичними методами, вони також можуть називатися математичними.

Математична функціональна модель, в загальному випадку, є алгоритмом для обчислення вектора вихідних параметрів Y на основі заданих векторів параметрів елементів X та зовнішніх параметрів Q.

Математичні моделі поділяються на символічні та числові. Символічні моделі працюють з позначеннями (ідентифікаторами), а числові можуть бути як аналітичними, так і алгоритмічними.

Існує кілька способів класифікації математичних моделей, залежно від їх ієрархічного рівня, використовуваного математичного апарату, виду фазових змінних та інших ознак. Наприклад, на системному рівні застосовують моделі масового обслуговування або мережі Петрі, а на функціонально-логічному рівні — автоматні моделі через функціональні передавальні функції або скінчені автомати.

Моделі об'єктів можуть бути різними: лінгвістичними, теоретико-множинними, абстрактно-алгебраїчними, нечіткими, автоматними тощо.

Особливо важливими є поняття повних і макромоделей, статичних і динамічних моделей, детермінованих і стохастичних, аналогових і дискретних моделей. Зокрема, розглядаються характеристики вихідних параметрів, які можуть бути функціональними або граничними.

Повна модель об'єкта, на відміну від макромоделі, охоплює не лише процеси на зовнішніх виходах, але й внутрішні процеси об'єкта.

Статичні моделі описують стани, які не змінюються в часі, і не враховують час як незалежну змінну. Динамічні моделі відображають зміну стану системи з урахуванням часу.

Моделі поділяються на стохастичні та детерміновані в залежності від того, чи враховують вони випадкові змінні.

Аналогові моделі працюють з неперервними величинами фазових змінних, тоді як у дискретних моделях ці змінні мають дискретні значення. Деякі з дискретних моделей можуть бути логічними (булевими), де стан системи описується за допомогою булевих значень.

Інформаційні моделі належать до інформаційного рівня автоматизованих систем і застосовуються при проектуванні баз даних для опису зв'язків між одиницями інформації.

Однією з найбільших проблем є моделювання слабоструктурованих систем, що є типовим для системного рівня проектування. Тут активно використовуються експертні методи. В теорії систем є загальні рекомендації щодо вибору експертів для розробки моделей, організації експертизи та обробки отриманих результатів. Загальний підхід до побудови моделей складних слабоструктурованих систем часто виражається через методики IDEF.

В імітаційних моделях зазвичай використовуються фазові змінні. На макрорівні імітаційні моделі можна уявити як системи алгебро-диференціальних рівнянь:

$$\mathbf{\Phi}(dV/dt, V, t) = 0 \text{ при } t = 0, V = V0,$$

де V — вектор фазових змінних; t — час; V0 — вектор початкових умов. Прикладом фазових змінних можуть бути струми та напруги в електричних системах, сили та швидкість у механічних системах, тиск та витрати у гідравлічних.

Вихідні параметри системи поділяються на два типи: параметри-функціонали (як V(t), що є рішеннями диференціальних рівнянь моделі) і параметри, що визначають здатність об'єкта функціонувати за певних зовнішніх умов. Ці параметри визначають граничні значення змінних зовнішніх параметрів, за яких об'єкт зберігає свою функціональність.

1.6. Типові проектні процедури

Процес створення проекту об'єкта (виробу чи процесу) включає вибір структури об'єкта, визначення значень усіх його параметрів та подання результатів у встановленому форматі.

Задача структурного синтезу у системотехніці формулюється як задача прийняття рішень (ЗПР). Постановка цієї задачі полягає у визначенні мети, множини можливих рішень і обмежень, які накладаються на ці рішення. Реальні проектні задачі, зазвичай, мають кілька критеріїв для оцінки рішень. Однією з основних труднощів при формулюванні багатокритеріальних задач є встановлення правил для вибору оптимальних варіантів.

При синтезі структури автоматизованої системи завдання включає наступні вихідні дані:

- множина функцій, які система повинна виконувати (це набір реалізацій алгоритмів, кожен з яких може складатися з кількох операцій); в цій множині може бути часткове упорядкування функцій, яке можна подати у вигляді орієнтованого графа, де вершини представляють функції, а дуги — відношення порядку;

- типи допустимих серверів (машин), які виконуватимуть функції системи;

- множина зовнішніх джерел та споживачів інформації;

- у деяких випадках задається початкова структура системи у вигляді взаємопов'язаних серверів певних типів, яка може виступати як початковий варіант;

- різноманітні обмеження, зокрема на витрати ресурсів і (або) на час виконання функцій.

Задача полягає в синтезі (або корекції) структури, визначенні типів серверів (апаратно-програмних засобів), розподілі функцій між серверами таким чином, щоб досягти оптимуму цільової функції за умови виконання заданих обмежень.

Наступним етапом після синтезу є процедура аналізу. Метою аналізу є отримання інформації про характер функціонування та значення вихідних параметрів Y при заданій структурі об'єкта, відомих значеннях зовнішніх параметрів Q та параметрів елементів X. Якщо значення параметрів X і Q зафіксовані, то здійснюється процедура одно-варіантного аналізу, що зводиться до розв’язання рівнянь математичної моделі та обчислення вектора вихідних параметрів Y.

Структура САПР

Як і будь-яка складна система, САПР складається з різних підсистем. Розрізняють підсистеми, що безпосередньо займаються проектуванням, та підсистеми, що здійснюють обслуговування.

Підсистеми, що виконують проектування, здійснюють основні проектні операції. Прикладами таких підсистем можуть бути: підсистема тривимірного геометричного моделювання об'єктів, підсистема створення конструкторської документації, підсистема схемотехнічного аналізу, підсистема трасування з'єднань на друкованих платах та інші.

Підсистеми, що забезпечують обслуговування, підтримують функціонування проектувальних підсистем. Їх сукупність зазвичай називають системним середовищем (або оболонкою) САПР. Типовими обслуговуючими підсистемами є підсистеми управління проектними даними, підсистеми розробки та супроводу програмного забезпечення CASE (Computer Aided Software Engineering), які допомагають користувачам освоювати технології, реалізовані в САПР.

Структурування САПР з різних точок зору призводить до виділення видів забезпечення САПР. Зазвичай розрізняють сім основних видів забезпечення САПР:

- Технічне забезпечення (ТЗ), що включає різноманітне апаратне забезпечення (ЕОМ, периферійні пристрої, мережеве обладнання, лінії зв'язку, вимірювальні прилади);

- Математичне забезпечення (МЗ), яке об'єднує математичні методи, моделі та алгоритми для проектування;

- Програмне забезпечення, що надається програмами САПР;

- Інформаційне забезпечення, що включає бази даних, СУБД та інші дані, які використовуються в процесі проектування (вся сукупність таких даних складає інформаційний фонд САПР, а база даних разом із СУБД створює банк даних);

- Лінгвістичне забезпечення, основою якого є мови спілкування між проектувальниками та ЕОМ, мови програмування та мови обміну даними між технічними засобами САПР;

- Методичне забезпечення, яке включає різноманітні методики проектування, інколи також включаючи математичне забезпечення;

- Організаційне забезпечення, яке представлено штатними розкладами, посадовими інструкціями та іншими документами, що регламентують роботу проектного підприємства.

Різновиди САПР

Класифікація САПР проводиться за різними критеріями, такими як цільове призначення, масштаби (складність вирішуваних завдань), характер базової підсистеми – ядра САПР тощо.

Згідно з додатками, найбільш розповсюдженими та використовуваними є такі групи САПР:

1. САПР для загального машинобудування, часто звані машинобудівними САПР або системами MCAD (Mechanical CAD).
2. САПР для радіоелектроніки, які включають системи ECAD (Electronic CAD) або EDA (Electronic Design Automation).

3. САПР у галузі архітектури та будівництва.

За цільовим призначенням розрізняють САПР або підсистеми САПР, що забезпечують різні аспекти проектування.

За масштабами розрізняють окремі програмно-методичні комплекси (ПМК) САПР, такі як:

- Комплекси аналізу міцності механічних виробів за методом скінченних елементів (МСЕ).
- Комплекси для аналізу електронних схем.
- Системи з унікальними архітектурами як для програмного забезпечення (software), так і для технічного (hardware) забезпечення.

За характером базової підсистеми САПР можна виділити такі різновиди:

1. САПР на основі підсистеми машинної графіки та геометричного моделювання. Ці системи орієнтовані на додатки, де основною процедурою проектування є створення просторових форм та взаємного розташування об'єктів. До цієї групи відносяться більшість САПР у машинобудуванні, що використовують графічні ядра.
2. САПР, що включають СУБД. Ці системи використовуються в додатках, де великі обсяги даних обробляються при порівняно простих математичних розрахунках. Такі САПР часто зустрічаються в техніко-економічних додатках, таких як проектування бізнес-планів, а також у проектуванні об'єктів, наприклад, для щитів управління в автоматичних системах.
3. САПР, що використовують конкретні прикладні пакети. Це автономно використовувані ПМК, такі як імітаційне моделювання виробничих процесів, розрахунки міцності за методом скінчених елементів (МСЕ), синтез та аналіз систем автоматичного керування тощо. Прикладами є програми логічного проектування з використанням мови VHDL, математичні пакети типу MathCAD.
4. Комплексні (інтегровані) САПР, що складаються із сукупності підсистем попередніх видів. Яскравими прикладами таких систем є CAE/CAD/CAM-системи в машинобудуванні або САПР для проектування великих

інтегральних схем (ВІС). САПР ВІС включає СУБД та підсистеми для проектування компонентів, принципових і логічних схем, топології кристалів, а також тестування на придатність виробів. Для управління такими складними системами використовують спеціалізовані системні середовища.

Етапи проектування автоматизованих систем

Етапи проектування

Проектування автоматизованих систем (АС) є багатогранним процесом, що включає два основних напрями:

1. Проектування АС для конкретних підприємств на базі готових компонентів (програмних і апаратних), що здійснюється за допомогою спеціальних інструментальних засобів розробки.
2. Розробка компонентів АС (програмних та апаратних засобів), орієнтованих на багаторазове використання при створенні різних АС.

У першому напрямку головна увага приділяється системній інтеграції, де розробник повинен бути висококваліфікованим спеціалістом, знати міжнародні стандарти, тенденції розвитку технологій і володіти CASE-засобами для розробки додатків. Це дозволяє інтегрувати різні технології та компоненти в єдину автоматизовану систему.

Другий напрямок стосується розробки методів, алгоритмів та програм, орієнтованих на широке застосування. Важливими є технології компонентно-орієнтованого програмування, які дозволяють створювати програми, здатні інтегруватися в різні АС.

Для проектування АС використовуються національні (ГОСТ 34.601-90) [14] та міжнародні (ISO/IEC/IEEE 12207:2018) [6] стандарти, які регламентують створення АС та їх життєвий цикл.

Існують різні стилі проектування:

- Низхідне (Top-down Design) — проектування через деталі, що поступово деталізуються.

- Висхідне (Bottom-up Design) — проектування через формування базових елементів.
- Еволюційне (Middle-of-Design) — поступове оновлення і вдосконалення проекту в процесі розробки.

Етапи проектування (за ГОСТ 34.601-90):

1. Концептуальне проектування:
   - Передпроектні дослідження: аналіз діяльності підприємства, виявлення слабких місць у поточних технологіях.
   - Моделі "As Is" та "To Be": створення моделей для відображення поточного стану та бажаного результату.
   - Технічне завдання (ТЗ): включає вимоги до інформаційних потоків, захисту, продуктивності тощо.
2. Ескізний проект:
   - Проектування архітектури системи, вибір базових апаратних та програмних засобів.
   - Врахування розвитку підприємства та можливості розширення.
3. Прототипування:
   - Створення прототипу для демонстрації майбутньої системи та корекції проекту за відгуками користувачів.
4. Робоче проектування:
   - Уточнення компонентів, програмних продуктів, побудова баз даних та розробка оригінального програмного забезпечення.
   - Виконання детального інфологічного проектування та створення робочої документації.
5. Впровадження та експлуатація:
   - Закупівля і установка програмних і апаратних засобів.
   - Введення в експлуатацію системи [14].

Проектування АС часто передбачає створення корпоративних мереж, що включають локальні підмережі та зв'язки між ними. Це важливо для

забезпечення надійного функціонування системи в різних географічних точках.

Цей процес вимагає детального підходу до вибору мережевого обладнання, програмного забезпечення та методів безпеки. Важливим є також використання інтернет-ресурсів для зв'язку між віддаленими точками, з урахуванням можливих проблем з безпекою та надійністю доставки даних.

РОЗДІЛ 2.

АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

2.1. Структура апаратного забезпечення

Апаратне забезпечення систем автоматизованого проектування (САП) включає в себе різноманітні компоненти, які спрямовані на ефективне виконання завдань проектування та автоматизації робочих процесів. Структура апаратного забезпечення таких систем може включати наступні елементи:

1. Робочі станції:

- Комп'ютери: Сучасні персональні комп'ютери використовуються для виконання програм проектування.
- Монітори: Великі екрани для зручного відображення графічної інформації.
- Клавіатури та миші: Введення команд та взаємодія з програмами.

2. Сервери:

- Центральні сервери: Великі сервери, які забезпечують централізоване збереження та обробку даних.
- Файлові сервери: Забезпечують доступ до спільних ресурсів та зберігання проектних файлів.
- Бази даних: Для збереження та організації інформації про проекти.

3. Спеціалізоване обладнання:

- Графічні пристрої: Високоефективні відеокарти для обробки великих графічних файлів та 3D-моделей.
- Принтери та плоттери: Для друку проектної документації та схем.
- Сканери: Для цифрового введення фізичних документів у систему.

4. Мережеве обладнання:

- Мережеві комутатори та маршрутизатори: Забезпечують зв'язок між робочими станціями та серверами.

- Кабельна і бездротова інфраструктура: Засоби зв'язку для обміну даними між різними частинами системи.

5. Засоби зберігання даних:

- Жорсткі диски: Для зберігання програм, проектів та інших даних.
- Мережеві сховища (NAS): Забезпечують централізоване зберігання та обмін даними.

6. Засоби безпеки:

- Брандмауери та антивірусне обладнання: Захист від зовнішніх загроз та вірусів.
- Системи резервного копіювання: Для забезпечення безпеки даних та можливості відновлення після випадкового видалення чи поломки.

2.2. Типи мереж

Методи розділення лінії передачі даних, такі як часовий мультиплексинг (TDM) і частотний розділ (FDM), є важливими при проектуванні систем зв'язку, зокрема для мереж, що підтримують САПР.

1. Часовий мультиплексинг (TDM) дозволяє організувати кілька каналів в одному фізичному середовищі шляхом виділення кожному каналу певного проміжку часу. Це підходить для випадків, коли кілька сигналів передаються по одному каналу, чергуючись у часі. Кожен сигнал має виділений слот часу, що дозволяє зменшити перехресні інтерференції між каналами.

2. Частотний розподіл (FDM) передбачає розділення каналу передачі на кілька частин, кожна з яких працює в окремому діапазоні частот. Це дає змогу одночасно передавати декілька сигналів через один фізичний канал без взаємного втручання. Метод підходить для випадків, коли потрібно одночасно передавати різні види даних по різних частотних каналах.

Що стосується локальних обчислювальних мереж (ЛОМ, LAN), вони зазвичай використовуються в маленьких проектних організаціях для забезпечення зв'язку між комп'ютерами, розташованими на невеликих відстанях один від одного. Топологія ЛОМ може бути шинною, кільцевою, або

зірковою, що визначає, як з'єднані комп'ютери в мережі. ЛОМ має обмежену протяжність і кількість підключених вузлів.

Для більших організацій та корпоративних мереж (LAN) зазвичай використовують більш складну структуру, що включає кілька ЛОМ, з'єднаних між собою через комутаційні сервери. Такі сервери відповідають за взаємодію між підмережами, а також за забезпечення зв'язку з іншими мережами (наприклад, з Інтернетом). Вони можуть бути об'єднані через опорну мережу, що забезпечує централізовану передачу даних між підмережами.

З розвитком віртуальних підприємств і технологій CALS, виникає потреба в територіальних мережах, таких як Інтернет, для забезпечення зв'язку між віддаленими співробітниками. Однак використання Інтернету для передачі важливих даних вимагає посилених заходів для забезпечення інформаційної безпеки, оскільки відкриті мережі піддаються численним загрозам.

2.3. Обладнання робочих місць в автоматизованих системах проектування та управління

Автоматизовані системи проектування та управління (САПР) використовують робочі станції, сервери та персональні комп'ютери для обробки даних, уникаючи використання дорогих великих ЕОМ, таких як суперкомп'ютери. Автоматизовані робочі місця (АРМ) базуються на робочих станціях чи ПК і включають в себе обладнання, таке як ЕОМ, пристрої введення-виведення, монітори та інші периферійні пристрої.

АРМ мають ієрархічну структуру пам'яті з кеш-пам'яттю, операційною пам'яттю та зовнішньою пам'яттю. Для зв'язку пристроїв використовується системна шина, а материнська плата має розширювальні шини для підключення різних пристроїв.

Графічні робочі станції використовуються в САПР для машинобудування і вимагають високої продуктивності процесора для виконання графічних операцій. Усі ці елементи забезпечують ефективне

функціонування системи, дозволяючи забезпечувати високий рівень продуктивності за доступною ціною.

РОЗДІЛ 3.

ОРГАНІЗАЦІЯ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

3.1. Методи розпаралелювання обчислювальних задач

1. Паралельні обчислення: використання паралельних обчислень може значно збільшити швидкість проектування. Розподілення задач між різними обчислювальними вузлами дозволяє використовувати їхні ресурси паралельно.

2. Кластерні обчислення: створення кластера обчислювальних вузлів для вирішення великих задач, таких як оптимізація параметрів напівпровідникових структур або чисельне моделювання.

3. Хмарні обчислення: використання хмарних сервісів для швидкого доступу до обчислювальних ресурсів та масштабування їх вгору або вниз в залежності від потреб.

4. Методи MapReduce для обробки великих обсягів даних: якщо дослідження включає аналіз великих обсягів даних, можна використовувати методи MapReduce для розподіленого обчислення.

5. Системи контейнерів та оркестрація: використання технологій контейнерів (наприклад, Docker) та оркестраторів (наприклад, Kubernetes) для розподілення обчислювальних завдань між різними контейнерами на різних вузлах.

6. Обчислення на багатоядерних архітектурах: враховуючи те, що багатоядерні процесори стають стандартом, важливо оптимізувати алгоритми та програми для використання паралельних обчислень.

7. Методи розподіленої оптимізації: якщо задачі проектування включають в себе оптимізацію параметрів, можна використовувати методи розподіленої оптимізації для швидкого знаходження оптимальних рішень.

3.2 Інтеграція розпаралелювальних систем у програмні комплекси

Інтеграція розподілених систем у програмних комплексах для автоматизованого проектування напівпровідникових структур може включати в себе кілька аспектів:

1. Паралельні обчислення:
   - використання розподілених обчислень для паралельної обробки чисельних розрахунків та симуляцій напівпровідникових структур;
   - інтеграція розподілених обчислень для оптимізації швидкості та ефективності обчислень.
2. Системи керування версіями та спільна робота:
   - використання систем керування версіями для координації та збереження змін у програмному коді та моделях;
   - створення інструментів для спільної роботи та обміну даними між учасниками проекту.
3. Інтеграція з системами відстеження завдань:
   - використання систем відстеження завдань для контролю та організації завдань, пов'язаних із проектуванням структур;
   - реалізація автоматизованих сповіщень та оновлень для сприяння співпраці між командою.
4. Розподілена обробка великих обсягів даних:
   - використання розподілених систем для ефективної обробки та аналізу великих обсягів експериментальних даних та результатів моделювання;
   - інтеграція інструментів великих даних для виявлення закономірностей у результатах чисельного моделювання.
5. Системи автоматизованого тестування:
   - розробка систем автоматизованого тестування для перевірки правильності та стабільності програмних модулів у розподіленому середовищі.
6. Кластери обчислень та хмарні рішення:
   - використання кластерів обчислень та хмарних сервісів для швидкого масштабування обчислювальних ресурсів залежно від потреб.

7. Безпека та автентифікація:

- розробка засобів безпеки та автентифікації для забезпечення конфіденційності та цілісності даних при обміні між розподіленими системами.

8. Інтеграція з графічним інтерфейсом:

- розробка інтерфейсу користувача, який інтегрує різні функції та ресурси для зручного керування та моніторингу процесу проектування.

3.3. Оптимізація роботи розпаралелювальних систем

Оптимізація роботи розподілених систем у контексті даної дипломної роботи є ключовим аспектом для забезпечення ефективності та продуктивності процесів. Нижче подано кілька напрямків оптимізації:

1. Розподілення завдань:

- визначення оптимального розподілу завдань між різними вузлами системи;
- використання алгоритмів розподілення, що враховують обсяг та складність кожного завдання.

2. Балансування навантаження:

- моніторинг навантаження на кожному вузлі та розподіл завдань так, щоб уникнути перевантаження окремих ресурсів;
- застосування алгоритмів балансування навантаження для автоматичної корекції нерівномірності розподілу.

3. Ефективне використання мережі:

- мінімізація комунікаційного навантаження між вузлами;
- використання локальної кеш-пам'яті для зберігання проміжних результатів та зменшення необхідності в частих мережевих взаємодіях.

4. Кешування та передача даних:

- використання механізмів кешування для збереження проміжних результатів та уникнення повторних обчислень;
- передача лише необхідної інформації між вузлами для зменшення обсягу даних, що передаються.

5. Оптимізація алгоритмів:

- аналіз та оптимізація алгоритмів, які використовуються для чисельного моделювання та оптимізації структур;
- підбір алгоритмів, які оптимально працюють у розподілених середовищах.

6. Адаптація до змін:

- розробка механізмів для автоматичної адаптації системи до змін у ресурсах чи навантаженні;
- застосування алгоритмів, які дозволяють динамічно змінювати розподіл завдань під час роботи системи.

7. Управління помилками та відновлення:

- розробка механізмів управління помилками та відновлення роботи системи після виникнення непередбачених ситуацій;
- застосування алгоритмів детектування та корекції помилок.

8. Масштабованість:

- розробка системи, яка може легко масштабуватися вгору або вниз в залежності від обсягу та складності проектів;
- використання технологій, які дозволяють автоматично масштабувати ресурси.

Оптимізація роботи розподілених систем у дипломній роботі буде спрямована на підвищення ефективності та продуктивності процесів автоматизованого проектування напівпровідникових структур.

РОЗДІЛ 4.

МЕХАНІЗМИ РОЗПАРАЛЕЛЮВАННЯ (ДЕКОМПОЗИЦІЇ) АЛГОРИТМІВ

4.1. Роль розпаралелювання алгоритмів у проектуванні напівпровідникових структур

Розпаралелювання алгоритмів відіграє важливу роль у проектуванні напівпровідникових структур, особливо з урахуванням зростаючої складності та обсягу обчислень, що вимагаються для їх створення та аналізу. Ось деякі ключові аспекти ролі розпаралелювання алгоритмів у цьому контексті:

1. Підвищення продуктивності: Напівпровідникові структури стають все складнішими з розвитком технологій. Розпаралелювання алгоритмів дозволяє розподілити обчислювальні завдання між кількома обчислювальними вузлами, що призводить до значного підвищення швидкості проектування та оптимізації ресурсів.

2. Масштабування: Завдяки розпаралелюванню можна ефективно масштабувати обчислювальні процеси для вирішення більших завдань. Це особливо важливо в контексті росту розмірності та складності напівпровідникових структур, що вимагають більших обсягів обчислень.

3. Зменшення часу проектування: Розпаралелювання алгоритмів дозволяє розділити обчислювальні завдання на декілька частин, які можуть виконуватися паралельно. Це допомагає скоротити час, необхідний для завершення проектування, забезпечуючи швидший процес розробки та тестування.

4. Оптимізація ресурсів: Розпаралелювання дозволяє ефективніше використовувати обчислювальні ресурси, такі як багатоядерні процесори або обчислювальні кластери. Це дозволяє забезпечити краще використовувати обладнання та підвищити продуктивність процесу проектування.

5. Можливість використання нових технологій: Розпаралелювання алгоритмів робить можливим використання сучасних технологій, таких як

GPU обчислення або обчислення у хмарних сервісах, для прискорення процесу проектування та аналізу напівпровідникових структур.

Отже, розпаралелювання алгоритмів грає ключову роль у забезпеченні швидкості, ефективності та масштабованості процесу проектування напівпровідникових структур. Його використання дозволяє вдосконалити процес та забезпечити високу якість результата.

4.1.1. Декомпозиція як метод пониження рівня складності задач

Декомпозиція є важливим підходом для вирішення складних задач, зокрема в області автоматизованого проектування напівпровідникових структур. Її суть полягає в розбитті вихідної задачі на більш дрібні, прості складові, що значно полегшує їх вирішення. Завдяки цьому забезпечується ефективне використання обчислювальних ресурсів, знижується ризик помилок і спрощуються процеси проектування.

У застосуванні декомпозиції особливу роль відіграє її ієрархічність. На першому етапі система розбивається на великі функціональні блоки, які відповідають основним завданням. Далі кожен із цих блоків деталізується до рівня елементарних компонентів. Такий підхід дозволяє організувати структуру завдання таким чином, щоб кожну частину можна було опрацьовувати незалежно, не зачіпаючи інші.

Безперечною перевагою декомпозиції є її здатність підвищувати гнучкість дизайну. Наприклад, у процесі роботи над проектом можна модифікувати лише один із модулів, зберігаючи загальну стабільність системи. Крім того, така модульна структура полегшує повторне використання вже створених компонентів у нових проектах, що скорочує витрати часу та ресурсів.

Декомпозиція також відкриває можливості для розпаралелення процесів. Поділ завдання на автономні частини дозволяє одночасно виконувати кілька етапів з використанням різних обчислювальних вузлів. Це

особливо актуально у великих проектах, які потребують значної обчислювальної потужності.

В автоматизованому проектуванні декомпозицію можна застосовувати на різних рівнях. На мікрорівні він охоплює деталізацію окремих транзисторів, або елементів схеми, тоді як на мезорівні розглядаються взаємодії між підсистемами, такими як блоки пам’яті або арифметичні операції. На макрорівні увага приділяється загальній структурі системи з урахуванням її глобальних взаємозв’язків.

Застосування методу декомпозиції в системах автоматизованого проектування дозволяє ефективно знизити трудомісткість завдань, зберігаючи при цьому точність і якість рішень. Завдяки такому підходу забезпечується масштабованість, адаптивність і стабільність програмних рішень, що є основою успішного проектування в сучасній технологічній сфері.

4.2. Методи розпаралелювання

Під час розробки програмного забезпечення для автоматизованого проектування напівпровідникових структур важливо враховувати можливості розпаралелювання алгоритмів для підвищення продуктивності та ефективності обчислень. Нижче наведено основні методи розпаралелювання, які можуть бути використані в цьому контексті:

1. Розподіл завдань (Task Parallelism)

Метод полягає в розділенні обчислювального завдання на незалежні частини, які можуть бути виконані паралельно. Кожен обчислювальний вузол отримує свою частину завдань для обробки.

2. Паралельність даних (Data Parallelism)

У цьому методі дані розбиваються на частини, і кожен обчислювальний вузол обробляє свою частину даних. Це особливо корисно, коли операції, які потрібно виконати над кожним елементом даних, є однаковими.

3. Функційна паралельність (Functional Parallelism)

Цей метод використовується, коли функції алгоритму можна розділити на окремі частини, які можуть виконуватися паралельно. Кожен обчислювальний вузол виконує свою частину функції, а потім результати об'єднуються.

4. Паралельність задач (Task-level Parallelism)

Метод передбачає розділення алгоритму на окремі задачі, які можуть виконуватися паралельно. Кожна задача може мати свої власні вимоги до обчислювальних ресурсів та часу виконання.

5. Декомпозиція даних (Data Decomposition)

У цьому методі великі обсяги даних розбиваються на менші частини, які можуть оброблятися паралельно. Цей підхід широко використовується у задачах, де дані можуть бути легко розділені на незалежні частини.

Кожен з цих методів має свої переваги та недоліки, і вибір конкретного методу залежить від характеру алгоритму, доступних обчислювальних ресурсів та вимог до продуктивності проектування напівпровідникових структур.

4.3. Оптимізація взаємодії між обчислювальними вузлами

Оптимізація взаємодії між обчислювальними вузлами є ключовим аспектом розпаралелювання алгоритмів у системах проектування напівпровідникових структур. Ефективна комунікація та координація між вузлами дозволяють забезпечити оптимальне використання ресурсів і мінімізувати затримки у взаємодії. Ось деякі методи оптимізації взаємодії між обчислювальними вузлами:

1. Мінімізація затримок передачі даних: Великі затримки при передачі даних між обчислювальними вузлами можуть суттєво уповільнити процес розпаралелювання. Одним з способів оптимізації є використання ефективних протоколів комунікації та мережевих технологій, таких як InfiniBand або RDMA (Remote Direct Memory Access), які забезпечують швидку передачу даних без значних затримок.

2. Балансування навантаження: Для максимальної продуктивності всі обчислювальні вузли повинні працювати приблизно однаково. Балансування навантаження може включати розподіл завдань таким чином, щоб кожен вузол мав приблизно однакове навантаження, або автоматичне перерозподілення завдань під час виконання в залежності від навантаження на вузли.

3. Уникання збоїв та переповнень: При роботі в розподіленому середовищі важливо передбачити можливі збої та переповнення. Для цього можна використовувати механізми контролю та управління, такі як механізми блокування, обробка помилок та автоматичне відновлення роботи.

4. Оптимізація комунікаційних патернів: Використання ефективних комунікаційних патернів, таких як розподілені обчислення, патерни збору та розсилання, може допомогти оптимізувати взаємодію між вузлами та зменшити накладні витрати на комунікацію.

5. Використання кешування та локальних обчислень: Використання кешування результатів обчислень на локальних вузлах може допомогти зменшити потребу у передачі даних між вузлами. Також можна спробувати виконувати обчислення локально на кожному вузлі, якщо це можливо, замість передачі даних для централізованого обчислення.

Ці методи дозволяють оптимізувати взаємодію між обчислювальними вузлами та підвищити ефективність розпаралелювання алгоритмів у системах проектування напівпровідникових структур.

4.4. Практичні аспекти впровадження механізмів розпаралелювання

Практичне впровадження механізмів розпаралелювання в системах проектування напівпровідникових структур передбачає ретельне планування та виконання кількох ключових кроків.

Спочатку, важливо провести аналіз алгоритмів та даних, які будуть оброблятися в системі. Цей аналіз допоможе визначити, які частини

алгоритмів можуть бути розпаралелені та які дані можуть бути розділені для обробки на різних обчислювальних вузлах.

Після цього необхідно вибрати підходящі технології та інструменти для розпаралелювання. Це можуть бути MPI, OpenMP, або інші бібліотеки та фреймворки для розподілених обчислень, які найкраще відповідають потребам системи.

Далі необхідно розробити паралельні версії алгоритмів та коду, що включає розподіл обчислювальних завдань між вузлами, синхронізацію даних та керування виконанням.

Після розробки паралельного коду необхідно провести тестування та налагодження для переконання у правильності та ефективності роботи. Це може включати тестування на різних наборах даних та в різних умовах використання.

Далі важливо здійснити моніторинг продуктивності та оптимізацію роботи системи. Це може включати масштабування рішення для роботи на більшій кількості вузлів, а також вдосконалення паралельних алгоритмів для підвищення швидкодії.

Управління ресурсами та обмеженнями також важливий аспект впровадження механізмів розпаралелювання. Керування кількістю доступних обчислювальних вузлів, обсягом доступної пам'яті та іншими факторами допомагає забезпечити ефективну роботу системи.

4.5. Виклики та перспективи

Виклики та перспективи впровадження механізмів розпаралелювання в системи проектування напівпровідникових структур є важливою темою для обговорення. Ось кілька ключових аспектів:

1. Складність алгоритмів і обробки даних: Багато алгоритмів, що використовуються у проектуванні напівпровідникових структур, можуть бути дуже складними та вимагати значних обчислювальних ресурсів.

Розпаралелювання таких алгоритмів може стати складною задачею через їх високу ступінь взаємозалежності та об'єм оброблюваних даних.

2. Потреба у великих обчислювальних ресурсах: Розпаралелювання алгоритмів у проектуванні напівпровідникових структур може вимагати значних обчислювальних ресурсів, таких як потужні комп'ютери або обчислювальні кластери. Це може стати викликом для деяких організацій або дослідницьких груп із обмеженими бюджетами.

3. Синхронізація та керування даними: Ефективне управління та синхронізація даними між різними обчислювальними вузлами може бути складною задачею. Потрібно розробляти ефективні механізми для передачі даних, уникнення конфліктів та забезпечення правильного порядку виконання операцій.

4. Підтримка та обслуговування інфраструктури: Успішне впровадження розпаралелювання вимагає підтримки та обслуговування складної інфраструктури, включаючи обчислювальні кластери, мережі та програмне забезпечення. Це може вимагати додаткових зусиль та ресурсів для забезпечення надійності та ефективності системи.

Незважаючи на ці виклики, впровадження механізмів розпаралелювання в системи проектування напівпровідникових структур відкриває перед нами значні перспективи:

1. Збільшення продуктивності: Розпаралелювання дозволяє використовувати обчислювальні ресурси більш ефективно, що може призвести до значного збільшення продуктивності проектування.

2. Зменшення часу розробки: Швидше виконання обчислень завдяки розпаралелюванню може сприяти зменшенню часу розробки нових напівпровідникових структур та прискорити процес введення їх на ринок.

3. Можливість використання великих обсягів даних: Розпаралелювання дозволяє обробляти великі обсяги даних швидше та ефективніше, що може бути корисним для аналізу та моделювання складних систем напівпровідникових структур.

В цілому, впровадження механізмів розпаралелювання в системи проектування напівпровідникових структур має значний потенціал для підвищення продуктивності та ефективності процесу розробки та дослідження.

РОЗДІЛ 5
ХМАРНІ ТЕХНОЛОГІЇ

Розділ "Хмарні технології" висвітлює можливості та використання хмарних обчислень у сфері автоматизованого проектування напівпровідникових структур. З розвитком хмарних технологій важливо визначити, як ці інновації можуть бути застосовані для покращення ефективності та доступності інструментів проектування в даній області.

5.1. Основні принципи хмарних обчислень

Основні принципи хмарних обчислень визначають способи створення, розгортання та використання обчислювальних ресурсів через Інтернет. Нижче подано детальний опис цих принципів:

1. Віртуалізація ресурсів: Цей принцип передбачає використання технологій віртуалізації для створення віртуальних екземплярів обчислювальних ресурсів, таких як сервери, мережі та сховища даних. Віртуалізація дозволяє забезпечити ізоляцію та розділення ресурсів між різними користувачами, що забезпечує безпеку та ефективне використання обчислювальних потужностей.

2. Масштабованість: Хмарні системи можуть масштабуватися вгору або вниз, щоб забезпечити потрібну кількість обчислювальних ресурсів в залежності від потреб користувачів. Це означає, що користувачі можуть легко збільшувати або зменшувати кількість ресурсів у відповідь на зміни навантаження або вимоги їх додатків.

3. Самообслуговування: Цей принцип дозволяє користувачам самостійно вибирати, налаштовувати та керувати обчислювальними ресурсами через веб-інтерфейси або API, без необхідності звертатися до адміністраторів системи. Користувачі можуть легко створювати віртуальні машини, налаштовувати мережеві налаштування та керувати доступом до даних.

4. Платіж за використання (Pay-as-You-Go): Цей принцип передбачає, що користувачі сплачують лише за фактично використані обчислювальні ресурси та послуги, без необхідності витрачати гроші на покупку, налаштування та утримання власних серверів. Це дозволяє оптимізувати витрати на обчислювальні потреби та знижувати загальні витрати на IT.

5. Резервування ресурсів за запитом: Цей принцип передбачає можливість користувачів швидко отримувати доступ до додаткових обчислювальних ресурсів за запитом. Користувачі можуть легко резервувати обчислювальні ресурси, коли вони потрібні для виконання певних завдань або обробки великих обсягів даних.

Ці принципи забезпечують гнучкість, ефективність та економічність хмарних обчислень, що робить їх популярними у багатьох галузях, включаючи інженерію та проектування.

5.2. Інтеграція хмарних сервісів у системи проектування

Інтеграція хмарних сервісів у системи проектування напівпровідникових структур відкриває широкі можливості для поліпшення ефективності, зручності та доступності робочих процесів. Ось деякі ключові аспекти інтеграції хмарних сервісів у системи проектування:

1. Сховище даних в хмарі: Використання хмарних сховищ даних, таких як Google Drive, Dropbox або AWS S3, дозволяє зручно зберігати та спільно використовувати проектні файли та документацію. Це забезпечує доступність даних з будь-якого пристрою та місця з підтримкою синхронізації.

2. Хмарні обчислення: Використання хмарних обчислень дозволяє розширювати обчислювальні можливості без необхідності володіння та підтримки власної обладнаної інфраструктури. Зокрема, можна використовувати віртуальні машини у хмарі для виконання складних обчислень та моделювання.

3. Спільна робота та колаборація: Багато хмарних сервісів мають інструменти для спільної роботи та колаборації, такі як Google Docs або

Microsoft Office 365. Це дозволяє різним учасникам проекту працювати над спільними документами та файлами в реальному часі, що підвищує продуктивність команди та спрощує обмін інформацією.

4. Хмарний доступ до програмних засобів: Деякі виробники програмного забезпечення пропонують версії своїх продуктів у хмарі, що дозволяє користувачам отримати доступ до програм без необхідності встановлення та оновлення на локальних комп'ютерах. Це особливо корисно для доступу до спеціалізованих програм для проектування напівпровідникових структур без необхідності встановлення їх на кожному робочому місці.

5. Автоматизація та інтеграція з іншими системами: Хмарні сервіси часто мають API та інтеграційні можливості, що дозволяють автоматизувати процеси роботи та інтегрувати їх з іншими системами, такими як системи управління версіями коду, системи управління завданнями або CAD/CAM системи.

Інтеграція хмарних сервісів у системи проектування дозволяє покращити ефективність, забезпечити зручний доступ до обчислювальних ресурсів та спростити спільну роботу у команді.

5.3. Вплив хмарних технологій на ефективність проектування

Вплив хмарних технологій на ефективність проектування напівпровідникових структур є значний і має декілька ключових аспектів. Враховуючи потужності та можливості, які пропонують хмарні сервіси, можна виокремити наступні позитивні впливи:

1. Збільшена доступність ресурсів: Хмарні технології дозволяють отримувати доступ до обчислювальних ресурсів та програмних інструментів з будь-якого місця та пристрою, що підключений до мережі Інтернет. Це забезпечує безперервну доступність для інженерів та дозволяє їм працювати з будь-якої точки світу.

2. Підвищена гнучкість та масштабованість: Хмарні технології надають можливість швидко розгортати та масштабувати обчислювальні ресурси згідно з потребами проекту. Це дозволяє ефективно використовувати ресурси у відповідності з обсягом та складністю завдань.

3. Зниження витрат на обладнання та програмне забезпечення: Використання хмарних сервісів дозволяє уникнути значних витрат на придбання та підтримку власних серверів та програмного забезпечення. Крім того, користувачі платять лише за фактично використані ресурси, що дозволяє ефективно використовувати фінансові ресурси.

4. Підвищення продуктивності та спільної роботи: Хмарні сервіси забезпечують зручні інструменти для спільної роботи та колаборації, що підвищує ефективність командної роботи. Різні учасники проекту можуть одночасно працювати над спільними документами та файлами, обмінюватися інформацією та вносити зміни в реальному часі.

5. Забезпечення безпеки та захисту даних: Багато хмарних сервісів надають розширені засоби захисту даних, включаючи шифрування, автоматичні резервні копії та захист від вторгнень. Це дозволяє забезпечити безпеку та конфіденційність даних проекту.

Отже, враховуючи всі переваги, використання хмарних технологій у системах проектування напівпровідникових структур є важливим кроком у напрямку покращення ефективності, зручності та безпеки робочих процесів.

РОЗДІЛ 6. МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ В АВТОМАТИЗОВАНОМУ ПРОЕКТУВАННІ

6.1. Математичні моделі пристроїв напівпровідникової електроніки

У галузі надвисокочастотної електроніки значного поширення набули напівпровідникові p-i-n діоди (їх ще називають плазмовими діодами) [42]. Ці пристрої створюються шляхом формування контактів між матеріалами з різними типами провідності: шар напівпровідника з власною провідністю (матеріал і-типу) розміщується між напівпровідниками p- (діркового) та n-типу (електронного). Робота p-i-n діода базується на принципі керування провідністю плазми з електронів та дірок, яка виникає в активній зоні пристрою (і-області).

Згідно з основами феноменологічної теорії електронно-діркових струмів у напівпровідникових p-i-n-діодах, математична модель роботи діода в стаціонарному режимі описується наступною системою диференціальних рівнянь [42, 51, 52, 53, 49]:

$$
\begin{aligned}
&-\frac{1}{e}\vec{\nabla}\cdot\vec{j}_p - R_p + G_p = 0,\ \frac{1}{e}\vec{\nabla}\cdot\vec{j}_n - R_n + G_n = 0,\\
&\vec{j}_p = -e\mu_p p\vec{\nabla}\varphi - eD_p\vec{\nabla}p,\ \vec{j}_n = -e\mu_n n\vec{\nabla}\varphi + eD_n\vec{\nabla}n,\\
&div\left(\vec{\nabla}\varphi\right) = -e\left(p - n + N_d\right),
\end{aligned}
\qquad (6.1)
$$

(1)

де n, p, φ – функції, які визначають концентрації електронів, дірок і потенціалу відповідно; $\vec{j}_p$, $\vec{j}_n$ – густини струмів дірок і електронів, що включають дрейфову та дифузійну компоненти; N_d – задана функція розподілу легуючих домішок у напівпровіднику (донорних або акцепторних); e – заряд електрона; D_p, D_n – коефіцієнти дифузії для дірок і електронів; μ_p, μ_n – рухомості носіїв заряду; R_p, R_n, G_p, G_n – швидкості рекомбінації та генерації носіїв заряду. Система рівнянь (6.1) доповнюється відповідними граничними умовами для меж активної області, а також n- і p-областей. Подібні постановки задач для

моделювання характеристик p-i-n-структур описані, зокрема, у роботах [49, 53].

Для аналізу електропровідності активної області p-i-n діодів часто використовують дифузійно-дрейфову модель. Вона спирається на рівняння неперервності електронного і діркового струмів, а також рівняння Пуассона з відповідними граничними умовами. У разі одновимірної стаціонарної задачі, що є актуальною через особливості геометрії пристрою та режими його роботи, формулювання задачі набуває вигляду [43-45]:

$$\begin{cases} \mu^2 \dfrac{\partial E}{\partial x} = \left(p - n + N_d \right), \\ \dfrac{\partial^2 n}{\partial x^2} + \dfrac{\partial n}{\partial x} E + n \dfrac{\partial E}{\partial x} - A_n n = 0, \\ \dfrac{\partial^2 p}{\partial x^2} - \dfrac{\partial p}{\partial x} E - p \dfrac{\partial E}{\partial x} - A_p p = 0, \end{cases} \tag{6.2}$$

$$\left(\frac{\partial n}{\partial x} - \gamma_n w n \right)\Big|_{x=0} = \frac{J}{eD_n} \frac{w}{N_i}, \quad \left(-\frac{\partial p}{\partial x} - \gamma_p w p \right)\Big|_{x=0} = 0, \tag{6.3}$$

$$\left(\frac{\partial n}{\partial x} - \gamma_n w n \right)\Big|_{x=1} = 0, \qquad \left(-\frac{\partial p}{\partial x} - \gamma_p w p \right)\Big|_{x=1} = \frac{J}{eD_p} \frac{w}{N_i},$$

$$E\big|_{x=0} = 0, \quad E\big|_{x=1} = 0, \quad \int_0^1 E dx = U, \tag{6.4}$$

де $E(x)$, $n(x)$, $p(x)$ – функції, що визначають напруженість електричного поля, а також розподіли концентрацій електронів і дірок в активній зоні діода (після нормування: $\tilde{x} = \frac{x}{w}$, $\tilde{E} = \frac{Eew}{kT}$, $\tilde{n} = \frac{n}{N_i}$, $\tilde{p} = \frac{p}{N_i}$); $\mu^2 = \frac{\varepsilon \varepsilon_0 kT}{e^2 w^2 N_i}$ – малий параметр $\mu \sim 10^{-6} \div 10^{-8}$; ε – відносна діелектрична стала; ε_0 – діелектрична стала вакууму; w – характерний розмір активної області; e – заряд електрона; N_i – концентрація носіїв заряду у власному напівпровіднику; k – стала Больцмана; T – температура (°К); $A_n = \frac{w^2}{D_n \tau_n^*}$, $A_p = \frac{w^2}{D_p \tau_p^*}$ – сталі; D_n, D_p – коефіцієнти дифузії для електронів і дірок відповідно; τ_n^*, τ_p^* – часи рекомбінації в об'ємі активної

області; N_d – профіль легування; J – густина струму інжекції; U – різниця потенціалів на активній зоні; γ_n, γ_n – коефіцієнти рекомбінації на контактах. У подальших розрахунках знак "~" не використовується.

6.2. Елементи теорії збурень в аналізі математичних моделей пристроїв напівпровідникової електроніки

Сформульована задача (1)-(3) є сингулярно збуреною. Її розв'язок, як зазначено у [46, 47], шукають у вигляді асимптотичних рядів за малим параметром:

$$\begin{pmatrix} N(x)+\underline{N}(\underline{\xi})+\overline{N}(\overline{\xi}) \\ P(x)+\underline{P}(\underline{\xi})+\overline{P}(\overline{\xi}) \\ \%(x)+\underline{E}(\underline{\xi})+\overline{E}(\overline{\xi}) \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^{s}\mu^i n_i(x,t) \\ \sum_{i=0}^{s}\mu^i p_i(x,t) \\ \sum_{i=0}^{s}\mu^i E_i(x,t) \end{pmatrix} + \begin{pmatrix} \sum_{i=0}^{s}\mu^i \underline{N}_i(\underline{\xi}) \\ \sum_{i=0}^{s}\mu^i \underline{P}_i(\underline{\xi}) \\ \sum_{i=-1}^{s}\mu^i \underline{E}_i(\underline{\xi}) \end{pmatrix} + \begin{pmatrix} \sum_{i=0}^{s}\mu^i \overline{N}_i(\overline{\xi}) \\ \sum_{i=0}^{s}\mu^i \overline{P}_i(\overline{\xi}) \\ \sum_{i=-1}^{s}\mu^i \overline{E}_i(\overline{\xi}) \end{pmatrix} + \begin{pmatrix} R_{n(s)}(x,\mu) \\ R_{p(s)}(x,\mu) \\ R_{E(s)}(x,\mu) \end{pmatrix}, \quad (6.5)$$

де $N(x), P(x), \%(x)$ – регулярні складові розв'язку, які виражають через ряди за степенями малого параметра; $\underline{N}(\underline{\xi})$, $\underline{P}(\underline{\xi})$, $\underline{E}(\underline{\xi})$, $\overline{N}(\overline{\xi}), \overline{P}(\overline{\xi}), \overline{E}(\overline{\xi})$ – примежові поправки, які будуються в околах *x=0* та *x=1*, які також подаємо у вигляді відповідних асимптотичних рядів ($\underline{\xi} = \frac{x}{\mu}, \overline{\xi} = \frac{1-x}{\mu}$ - регуляризуючі розтяги); $R_{E(s)}(x,\mu)$, $R_{n(s)}(x,\mu)$, $R_{p(s)}(x,\mu)$ - залишкові члени.

Шляхом виконання стандартної процедури декомпозиції вихідної задачі (1)-(3) приходимо до постановок рекурентної послідовності задач для систем звичайних диференціальних рівнянь з відповідними граничними умовами (за аналогією до [48]).

Наведемо приклади такої послідовності задач (приклад запозичено з роботи [48]):

$$\begin{cases} p_0(x) - n_0(x) = 0, \\ \dfrac{d^2 n_0(x)}{dx^2} - A_n n_0(x) = -\dfrac{d}{dx}\left(n_0(x) E_0(x)\right), \\ \dfrac{d^2 p_0(x)}{dx^2} - A_p p_0(x) = \dfrac{d}{dx}\left(p_0(x) E_0(x)\right), \end{cases} \tag{6.5.1}$$

$$\frac{\partial n_0(x)}{\partial x} - \gamma_n w n_0(x)\Big|_{x=0} = \frac{J}{eD_n}\frac{w}{N_i},$$

$$-\frac{\partial p_0(x)}{\partial x} - \gamma_p w p_0(x)\Big|_{x=1} = \frac{J}{eD_p}\frac{w}{N_i};$$

$$\begin{cases} \dfrac{\partial \underline{E}_{-1}(\underline{\xi})}{\partial \underline{\xi}} = \underline{P}_0(\underline{\xi}) - \underline{N}_0(\underline{\xi}), \\ \dfrac{\partial^2 \underline{N}_0(\underline{\xi})}{\partial \underline{\xi}^2} = -\dfrac{d}{d\underline{\xi}}\left(\left(n_0(0) + \underline{N}_0(\underline{\xi})\right)\underline{E}_{-1}(\underline{\xi})\right), \\ \dfrac{\partial^2 \underline{P}_0(\underline{\xi})}{\partial \underline{\xi}^2} = \dfrac{d}{d\underline{\xi}}\left(\left(p_0(0) + \underline{P}_0(\underline{\xi})\right)\underline{E}_{-1}(\underline{\xi})\right), \end{cases}$$

$$\lim_{\underline{\xi}\to\infty} \underline{E}_{-1}(\underline{\xi}) = E^*(U),\ \frac{\partial \underline{N}_0(\underline{\xi})}{\partial \underline{\xi}}\Big|_{\underline{\xi}=0} = 0,\ \lim_{\underline{\xi}\to\infty} \underline{N}_0(\underline{\xi}) = 0, \tag{6.5.2}$$

$$\frac{\partial \underline{P}_0(\underline{\xi})}{\partial \underline{\xi}}\Big|_{\underline{\xi}=0} = 0,\ \lim_{\underline{\xi}\to\infty} \underline{P}_0(\underline{\xi}) = 0;$$

$$\begin{cases} -\dfrac{\partial \overline{E}_{-1}(\overline{\xi})}{\partial \overline{\xi}} = \overline{P}_0(\overline{\xi}) - \overline{N}_0(\overline{\xi}), \\ \dfrac{\partial^2 \overline{N}_0(\overline{\xi})}{\partial \overline{\xi}^2} = \dfrac{d}{d\overline{\xi}}\left(\left(n_0(1) + \overline{N}_0(\overline{\xi})\right)\overline{E}_{-1}(\overline{\xi})\right), \\ \dfrac{\partial^2 \overline{P}_0(\overline{\xi})}{\partial \overline{\xi}^2} = -\dfrac{d}{d\overline{\xi}}\left(\left(p_0(1) + \overline{P}_0(\overline{\xi})\right)\overline{E}_{-1}(\overline{\xi})\right), \end{cases} \tag{6.5.3}$$

$$\lim_{\overline{\xi}\to\infty} \overline{E}_{-1}(\overline{\xi}) = E^*(U),$$

$$\frac{d\overline{N}_0(\overline{\xi})}{d\overline{\xi}}\Big|_{\overline{\xi}=1} = 0,\ \lim_{\overline{\xi}\to\infty} \overline{N}_0(\overline{\xi}) = 0,\ \frac{d\overline{P}_0(\overline{\xi})}{d\overline{\xi}}\Big|_{\overline{\xi}=1} = 0,\ \lim_{\overline{\xi}\to\infty} \overline{P}_0(\overline{\xi}) = 0;$$

$$\begin{cases} p_1(x)-n_1(x)=0, \\ \dfrac{d^2 n_1(x)}{dx^2}-A_n n_1(x)=-\dfrac{d}{dx}\Big(n_1(x)E_{\ 0}(x)+n_0(x)E_{\ 1}(x)\Big), \\ \dfrac{d^2 p_1(x)}{dx^2}-A_p p_1(x)=\dfrac{d}{dx}\Big(p_1(x)E_{\ 0}(x)+p_0(x)E_{\ 1}(x)\Big), \end{cases} \tag{6.6.1}$$

$$\frac{\partial n_1(x)}{\partial x}-\gamma_n w n_1(x)\Big|_{x=0}=0,\ -\frac{\partial p_1(x)}{\partial x}-\gamma_p w p_1(x)\Big|_{x=1}=0;$$

$$\begin{cases} \dfrac{\partial \underline{E}_0(\underline{\xi})}{\partial \underline{\xi}}=\underline{P}_1(\underline{\xi})-\underline{N}_1(\underline{\xi}), \\ \dfrac{\partial^2 \underline{N}_1(\underline{\xi})}{\partial \underline{\xi}^2}=-\dfrac{d}{d\underline{\xi}}\Big(E_0(0)\underline{N}_0(\underline{\xi})+\underline{E}_0(\underline{\xi})\Big(n_0(0)+\underline{N}_0(\underline{\xi})\Big)+ \\ \qquad +\underline{E}_{-1}(\underline{\xi})\Big(n_1(0)+\underline{\xi}n_0'(0)\Big)+\underline{N}_1(\underline{\xi})\underline{E}_{-1}(\underline{\xi})\Big), \\ \dfrac{\partial^2 \underline{P}_1(\underline{\xi})}{\partial \underline{\xi}^2}=\dfrac{d}{d\underline{\xi}}\Big(E_0(0)\underline{P}_0(\underline{\xi})+\underline{E}_0(\underline{\xi})\Big(p_0(0)+\underline{P}_0(\underline{\xi})\Big)+ \\ \qquad +\underline{E}_{-1}(\underline{\xi})\Big(p_1(0)+\underline{\xi}p_0'(0)\Big)+\underline{P}_1(\underline{\xi})\underline{E}_{-1}(\underline{\xi})\Big) \end{cases} \tag{6.6.2}$$

$$\Big(E_0(\mu\underline{\xi})+\underline{E}_0(\underline{\xi})\Big)\Big|_{\underline{\xi}=0}=0,\ \lim_{\underline{\xi}\to\infty}\underline{E}_0(\underline{\xi})=0,$$

$$\frac{\partial \underline{N}_1(\underline{\xi})}{\partial \underline{\xi}}-\gamma_n w\underline{N}_0(\underline{\xi})\Big|_{\underline{\xi}=0}=0,\ \lim_{\underline{\xi}\to\infty}\underline{N}_1(\underline{\xi})=0,$$

$$-\frac{\partial \underline{P}_1(\underline{\xi})}{\partial \underline{\xi}}-\gamma_p w\underline{P}_0(\underline{\xi})\Big|_{\underline{\xi}=0}=0,\ \lim_{\underline{\xi}\to\infty}\underline{P}_1(\underline{\xi})=0;$$

$$
\begin{cases}
-\dfrac{\partial \overline{E}_0\left(\overline{\xi}\right)}{\partial \overline{\xi}} = \overline{P}_1\left(\overline{\xi}\right) - \overline{N}_1\left(\overline{\xi}\right), \\
\dfrac{\partial^2 \overline{N}_1\left(\overline{\xi}\right)}{\partial \overline{\xi}^2} = \dfrac{d}{d\overline{\xi}}\Big(E_0(1)\overline{N}_0\left(\overline{\xi}\right) + \overline{E}_0\left(\overline{\xi}\right)\Big(n_0(1) + \overline{N}_0\left(\overline{\xi}\right)\Big) + \\
\quad + \overline{E}_{-1}\left(\overline{\xi}\right)\Big(n_1(1) - \overline{\xi} n_{0\mu}'(1)\Big) + \overline{N}_1\left(\overline{\xi}\right)\overline{E}_{-1}\left(\overline{\xi}\right)\Big), \\
\dfrac{\partial^2 \overline{P}_1\left(\overline{\xi}\right)}{\partial \overline{\xi}^2} = -\dfrac{d}{d\overline{\xi}}\Big(E_0(1)\overline{P}_0\left(\overline{\xi}\right) + \overline{E}_0\left(\overline{\xi}\right)\Big(p_0(1) + \overline{P}_0\left(\overline{\xi}\right)\Big) + \\
\quad + \overline{E}_{-1}\left(\overline{\xi}\right)\Big(p_1(1) - \overline{\xi} p_{0\mu}'(1)\Big) + \overline{P}_1\left(\overline{\xi}\right)\overline{E}_{-1}\left(\overline{\xi}\right)\Big),
\end{cases}
\tag{6.6.3}
$$

$$
\left(E_0\left(1-\mu\underline{\xi}\right) + \overline{E}_0\left(\underline{\xi}\right)\right)\Big|_{\underline{\xi}=0} = 0, \quad \lim_{\overline{\xi}\to\infty} \overline{E}_0\left(\overline{\xi}\right) = 0,
$$

$$
\frac{d\overline{N}_1\left(\overline{\xi}\right)}{d\overline{\xi}} - \gamma_n w \overline{N}_0\left(\overline{\xi}\right)\Big|_{\overline{\xi}=0} = 0, \ \lim_{\overline{\xi}\to\infty} \overline{N}_1\left(\overline{\xi}\right) = 0,
$$

$$
-\frac{d\overline{P}_1\left(\overline{\xi}\right)}{d\overline{\xi}} - \gamma_p w \overline{P}_0\left(\overline{\xi}\right)\Big|_{\overline{\xi}=0} = 0, \ \lim_{\overline{\xi}\to\infty} \overline{P}_1\left(\overline{\xi}\right) = 0,
$$

де $E_0(x)$, $n_0(x)$, $p_0(x)$ – головні члени регулярної частини асимптотики; $\underline{N}_0\left(\underline{\xi}\right)$, $\underline{P}_0\left(\underline{\xi}\right)$, $\underline{E}_{-1}\left(\underline{\xi}\right)$, $\overline{N}_0\left(\overline{\xi}\right), \overline{P}_0\left(\overline{\xi}\right), \overline{E}_{-1}\left(\overline{\xi}\right)$ – головні члени примежових поправок відповідно в околах точок $x=0$ та $x=1$; $E_1(x)$, $n_1(x)$, $p_1(x)$, $\underline{N}_1\left(\underline{\xi}\right)$, $\underline{P}_1\left(\underline{\xi}\right)$, $\underline{E}_0\left(\underline{\xi}\right)$, $\overline{N}_1\left(\overline{\xi}\right), \overline{P}_1\left(\overline{\xi}\right), \overline{E}_0\left(\overline{\xi}\right)$ – наступні члени асимптотики. Відмітимо, що для пошуку членів асимптотики з індексом i≥2 (i≥1 для примежових складових електростатичного поля) будуються послідовності задач, які аналогічні (5)-(6) [48].

Процес розв’язання передбачає декомпозицію вихідної задачі (1)-(3) на послідовність рекурентних підзадач для звичайних диференціальних рівнянь із відповідними граничними умовами. Такий підхід демонструє ефективність при застосуванні методу примежових поправок, хоча і вимагає значних обчислювальних ресурсів.

Модель задачі (1)-(3) є базовою для напівпровідникової електроніки.

У напівпровідникових матеріалах на поведінку носіїв заряду впливають процеси нагрівання кристала та електронно-діркової плазми, що виникають через виділення Джоулевого тепла в об’ємі p-i-n-діода, а також через вивільнення енергії в процесі рекомбінації носіїв заряду [49, 50]. Температурне поле описується рівнянням теплопровідності з відповідними граничними умовами, застосованими в одновимірному випадку.

6.3. Декомпозиція моделі і агрегація результатів

Декомпозиція моделі є невіддільною складовою математичного моделювання в автоматизованому проектуванні складних систем, зокрема напівпровідникових структур. Цей процес передбачає розбиття загальної задачі на простіші підзадачі, кожна з яких може бути вирішена окремо з використанням оптимізованих методів і алгоритмів. Такий підхід дозволяє зменшити складність розрахунків, уникнути перевантаження обчислювальних ресурсів і забезпечити ефективність моделювання.

Агрегування результатів полягає в об'єднанні розв’язків окремих підзадач у єдину модель, яка точно відображає загальну поведінку системи. Цей етап є критично важливим, адже від коректності інтеграції залежить точність кінцевого рішення. У напівпровідниковій електроніці результати агрегуються для створення єдиної картини поведінки пристрою в різних режимах роботи.

Проблеми декомпозиції та агрегації:

Попри очевидні переваги декомпозиції, процес може супроводжуватися низкою проблем. Одна з головних трудностей полягає в різних формах подання результатів окремих підзадач. Наприклад, одна частина моделі може бути виражена аналітично у вигляді математичної функції, тоді як інша – у формі таблиць числових значень, отриманих із експериментальних даних чи чисельного моделювання. Така різнорідність ускладнює процес об’єднання даних і потребує спеціальних методів для узгодження результатів між собою.

Крім того, під час агрегування виникає необхідність врахування впливу локальних параметрів однієї підзадачі на результати іншої. Особливо це стосується складних нелінійних систем, де навіть невелике відхилення в параметрах може істотно вплинути на кінцевий результат.

Результати декомпозиції та агрегування:

Отримані результати моделювання у процесі декомпозиції та агрегування подаються у формах, зручних для подальшого аналізу:

1. Таблиці:

- Використовуються для представлення числових даних, таких як концентрації носіїв заряду, температурні розподіли, потенціали або швидкості процесів.
- Дозволяють порівнювати отримані дані між різними підзадачами чи варіантами моделі.

2. Графіки:

- Візуалізують залежності між основними параметрами, наприклад, розподіл концентрації носіїв заряду вздовж активної області або динаміку зміни напруженості електричного поля.
- Є ефективним інструментом для перевірки узгодженості результатів та оцінки поведінки системи у різних режимах.

Використання таких форм подання результатів полегшує виявлення невідповідностей у моделі, допомагає оптимізувати її параметри та вдосконалювати процес автоматизованого проектування.

РОЗДІЛ 7. КОНЦЕПЦІЯ АРХІТЕКТУРИ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ ДОСЛІДЖЕННЯ ЗАДАЧ НАПІВПРОВІДНИКОВОЇ ЕЛЕКТРОНІКИ

7.1. Застосування DDD-підходу в задачах розробки архітектури ПЗ

1. Аналіз сучасних принципів та підходів до автоматизованого проектування напівпровідникових структур**:** DDD-підхід вимагає детального розуміння домену (в даному випадку — проектування напівпровідникових пристроїв). Першим етапом є *аналіз домену* через вивчення основних етапів проектування напівпровідникових структур, включаючи системний підхід. Оскільки DDD зосереджений на розумінні специфіки та мови домену, потрібно розробити *уточнені специфікації* для кожного етапу проектування, що можуть бути описані через класи, об'єкти та методи, що моделюють кожен етап проекту (наприклад, моделювання матеріалів, фізичних процесів, обчислювальних етапів).

2. Розробка концептуальної моделі архітектури автоматизованої системи проектування**:** У рамках DDD важливо розробити концептуальну модель архітектури, яка чітко відображатиме *контексти домену* проектування напівпровідникових пристроїв. Створення *моделей класів та компонентів*, що відповідають специфікаціям кожного етапу проектування, є важливою частиною. Концептуальна модель в DDD містить не лише структуру класів, але й механізми, які дозволяють *оптимізувати ієрархічний процес проектування*. Для цього можна застосувати принципи *bounded contexts* для чіткої ізоляції окремих частин системи та інтеграції їх через спеціальні інтерфейси та адаптери, які допомагають інтегрувати різні підсистеми в рамках загальної системи автоматизованого проектування.

3. Інтеграція алгоритмів розпаралелювання обчислювальних процесів**:** DDD також допомагає вирішувати проблеми, пов'язані з обчислювальними процесами, через *створення окремих доменних сервісів*, які можуть бути оптимізовані для роботи в розподілених обчислювальних середовищах.

Застосування принципів *сервісно-орієнтованої архітектури* (SOA) в DDD дозволяє ефективно розподіляти задачі між різними вузлами обчислень, що дає змогу *паралельно обробляти великі задачі*. Це дозволяє зменшити час на обчислення та підвищити швидкість проектування.

4. Вивчення можливостей інтеграції хмарних сервісів**:** DDD підхід також підтримує ідею *розподілу та інтеграції хмарних сервісів* для зберігання та обробки даних. Хмарні технології можуть бути інтегровані у вашу архітектуру через *чітко визначені інтерфейси* для зберігання проектних даних та результатів розрахунків, що дозволить забезпечити безпечне зберігання та доступність обчислювальних ресурсів. *Сервіси зберігання даних* можуть бути ізольовані в окремому контексті, що забезпечить легкість масштабування та надійність системи.

5. Оцінка рівня безпеки автоматизованої системи проектування**:** В DDD безпека є важливим аспектом при проектуванні складних систем, тому важливо *визначити рівень доступу* до компонентів системи та механізмів захисту. В рамках доменної моделі можна розробити спеціалізовані механізми захисту від несанкціонованого доступу, використовуючи принципи *separation of concerns* та *principle of least privilege* для кожного етапу проектування, що буде відповідати вимогам безпеки. Такі механізми можуть бути інтегровані на етапах, де обробляються критичні дані проекту.

6. Експериментальне впровадження та тестування компонентів системи**:** Тестування є важливим аспектом DDD, оскільки дозволяє перевірити, чи відповідають розроблені компоненти реальним вимогам домену. Для цього необхідно розробити *моделі тестування*, які будуть орієнтовані на перевірку всіх аспектів проектування напівпровідникових пристроїв. Це дозволяє виявити ефективність моделей та архітектурних рішень і забезпечити *високу точність результатів* у реальних умовах експлуатації.

7.2. Архітектура ПЗ для автоматизованого розв’язання сингулярно - збурених задач

1. *Розробка інтерфейсу* (читання вхідної інформації; виведення інформації в потрібних форматах (графіки, діаграми, таблиці); введення/виведення; керування системою; читання інформації з баз моделей; надання доступу до системи, організація колективної роботи розробників електронних пристроїв, передбачити ролі користувачів);

Вимоги до інтерфейсів:

- Функціональність: Інтерфейс повинен забезпечувати можливість введення/виведення даних у зручній формі: текстові файли, бази даних, графічне представлення результатів. Інтерактивні інструменти для аналізу та перевірки проектних даних [17].

-Візуалізація: Створення графіків і діаграм, що дозволяють наочно аналізувати складність структури, залежності параметрів тощо. Відображення структури проекту у вигляді дерева задач із можливістю переходу до підзадач [18].

- Забезпечення колективної роботи розробників відповідної технічної системи (розподіл ролей, робота в команді): Забезпечення функціоналу для роботи декількох користувачів із різними рівнями доступу: Адміністратор: налаштування системи, управління ролями користувачів. Розробник моделі: внесення змін до структури математичних моделей проекту (розробка проекту на низькому рівні), проведення експериментів по налагодженню алгоритмів, тестування. Користувач: розробник технічного проекту (розробка проекту на високому рівні), проведення машинних експериментів, доступ до результатів і звітів [19]. Реалізація чату для обміну інформацією між членами команди [20].

- Інтеграція з базами даних: Можливість доступу до бібліотек моделей, збереження результатів проектів та їх версій у базі даних [21].

- Технічні вимоги: Розробка веб-інтерфейсу (HTML5, CSS3, JavaScript/TypeScript) або настільного додатка (Qt, WPF, Electron). Використання RESTful API для комунікації клієнта з сервером [22]. Підтримка масштабування, щоб система залишалася стабільною за великих обсягів даних.

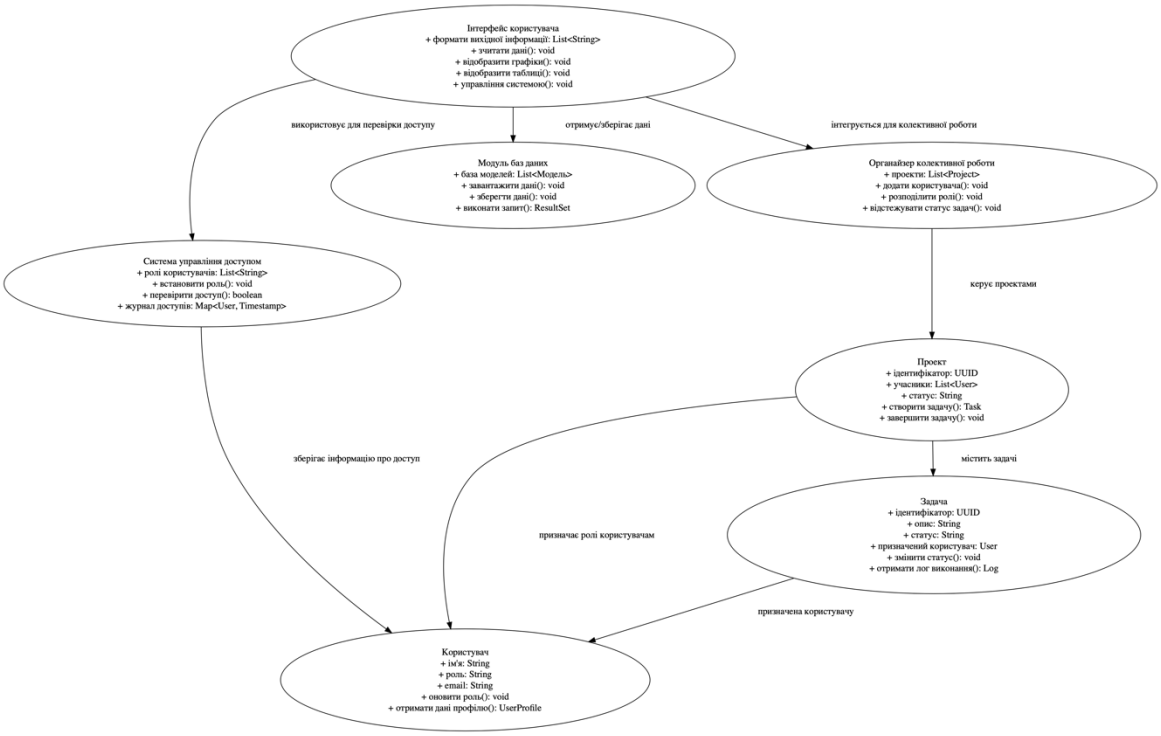

Рис. 7.2.1. Розробка інтерфейсу користувача

1. Інтерфейс користувача (UI):

- Відповідає за зчитування даних, відображення графіків, таблиць та управління системою.

- Взаємодіє з Системою управління доступом (Access) для перевірки доступу, з Модулем баз даних (DB) для зберігання і отримання даних, а також з Органайзером колективної роботи (Collab) для підтримки командної роботи [17].

2. Система управління доступом (Access):

- Керує ролями користувачів, перевіряє доступ до системи та зберігає інформацію про доступ.

- Взаємодіє з Користувачем (User) для управління доступом [19].

3. Модуль баз даних (DB):

- Відповідає за зберігання та завантаження даних із бази.

- Взаємодіє з Інтерфейсом користувача (UI) для зчитування та збереження даних [21].

4. Органайзер колективної роботи (Collab):

- Керує проектами та учасниками, розподіляє ролі та відстежує статус задач.

- Взаємодіє з Проектом (Project) та Задачею (Task) для керування проектами та завданнями [20].

5. Проект (Project):

- Має учасників, завдання та статус. Керує задачами та призначає ролі учасникам.

- Взаємодіє з Задачею (Task) та Користувачем (User) для управління проектами.

6. Задача (Task):

- Відповідає за опис, статус та призначеного користувача для виконання завдання.

- Взаємодіє з Користувачем (User) для призначення завдань та зміни їх статусу.

Захист інформації:

- Організаційні заходи: Визначення політик доступу: детальне розмежування прав доступу відповідно до ролей користувачів. Регулярні аудити безпеки для оцінки ризиків і усунення вразливостей [23].

Програмні механізми:

- Аутентифікація (авторизація): Підтримка багаторівневої авторизації користувачів. Використання протоколів OAuth2, SAML для інтеграції з корпоративними рішеннями.

- Шифрування:

- Використання сучасних криптографічних методів для збереження конфіденційності даних: Зашифроване зберігання даних на сервері (AES-256). Захищені з’єднання через протокол HTTPS.

- Контроль доступу: Використання системи журналів для моніторингу та аналізу дій користувачів. Захист від багаторазових невдалих спроб входу (наприклад, блокування IP).

- Захист від атак: Інтеграція фільтрів для запобігання SQL-ін'єкціям і XSS-атакам. Використання захисних систем для аналізу трафіку (IDS/IPS).[23]

2. *Виконання символьних перетворень* - проведення декомпозиції (синтаксичний аналіз введеної інформації; реалізація алгоритмів пов’язаних із підстановками рядкової інформації, виділенням частини рядка тощо) [25,26].

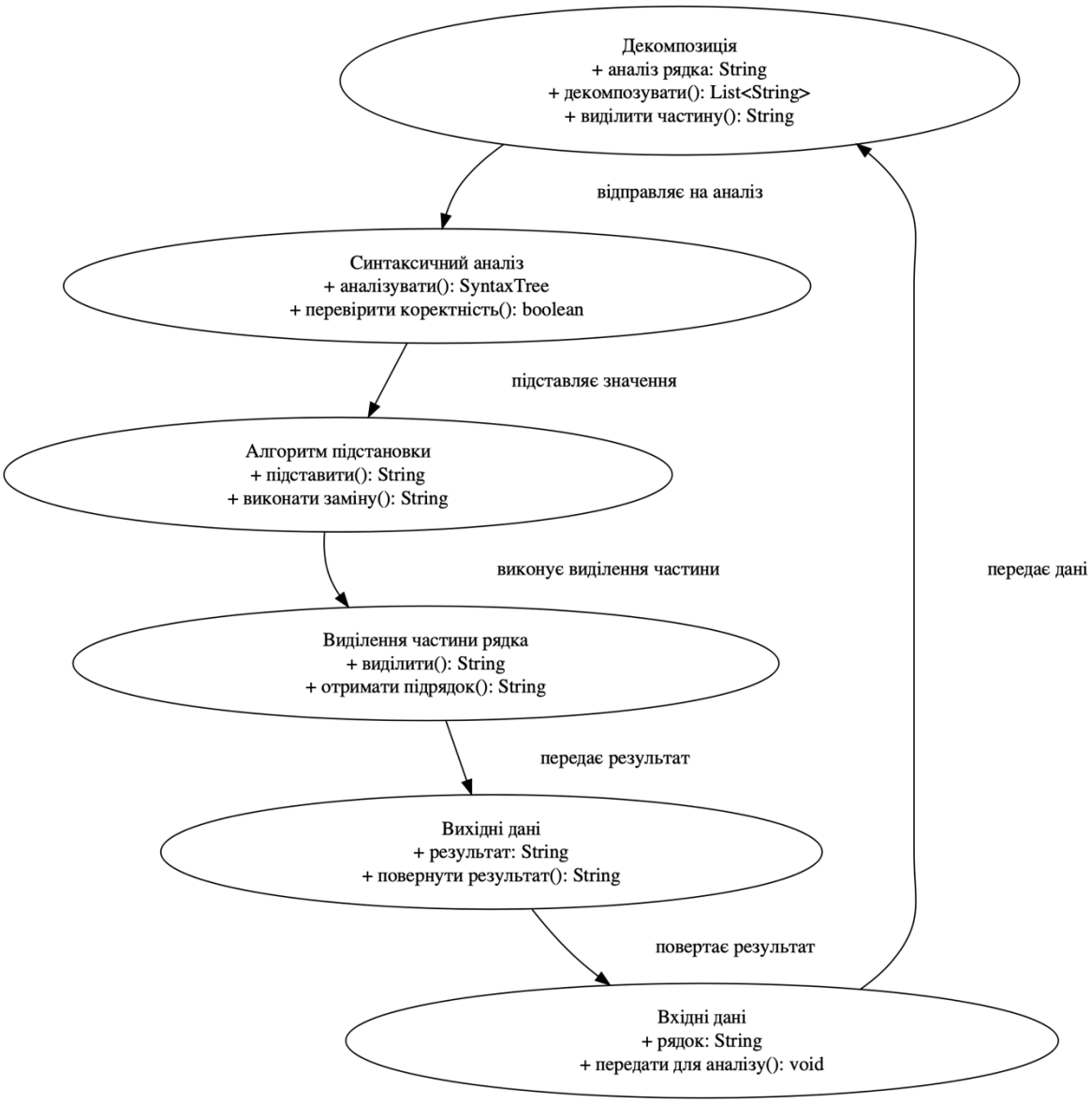

Рис. 7.2.2. Вимоги до символьних перетворень

Опис схеми:

1. Декомпозиція (Decomposition): клас, який відповідає за аналіз і декомпозицію рядка, виділення частини рядка.

- Має методи для декомпозиції і виділення частини рядка.

2. Синтаксичний аналіз (SyntaxAnalysis): клас для аналізу синтаксису введеного рядка.

- Метод для перевірки коректності і створення синтаксичного дерева [27].

3. Алгоритм підстановки (StringSubstitution): клас, який реалізує алгоритм підстановки значень у рядки.

- Має методи для виконання підстановок.

4. Виділення частини рядка (StringExtraction): клас для виділення частин рядка за допомогою алгоритмів виділення підрядків.

5. Вхідні дані (InputData): клас, який містить вхідний рядок та передає його для подальшої обробки.

6. Вихідні дані (OutputData): клас для зберігання результату обробки та повернення результату користувачу.

Взаємодія:

- Вхідні дані передаються до Декомпозиції, де рядок розбивається на частини.
- Потім рядок передається для Синтаксичного аналізу.
- Після аналізу використовуються Алгоритм підстановки та Виділення частини рядка для остаточного результату.
- Вихідні дані повертаються користувачу.

3. *Автоматизоване визначення типів отриманих підзадач* (використання штучного інтелекту; формування відповідних DataSet; тренування мережі тощо) [28,29].

Вимоги до ядра програмної системи:

3.1 Символьні перетворення

Основні функції:

- Синтаксичний аналіз: Аналіз текстових або структурних описів проекту для виявлення логічних і синтаксичних помилок. Підтримка стандартизованих мов опису структур (наприклад, VHDL, Verilog) [30].

- Робота з математичними виразами: Автоматичне приведення виразів до канонічної форми [31]. Можливість перетворення символьних рівнянь для чисельного розв’язання.

Інструменти та технології:

- Використання спеціалізованих бібліотек: SymPy (Python) — для роботи з математичними виразами [32]. Lex/Yacc або ANTLR — для синтаксичного аналізу [33].

- Декомпозиція задачі:

Реалізація алгоритмів, які автоматично виділяють компоненти задачі (підсистеми, вузли напівпровідникових структур). Система має визначати залежності між підзадачами та формувати дерево задач із позначенням їхнього статусу.

1.2. Елементи штучного інтелекту (ШІ) для розпізнавання типу математичної задачі.

Архітектура нейронної мережі:

- Типи мереж: Для класифікації задач: багатошаровий перцептрон (MLP) [34]. Для аналізу послідовностей (описів проектів): рекурентні нейронні мережі (RNN) або трансформери [35]. Для розпізнавання структур у графічних даних: згорткові мережі (CNN) [36].

- Особливості навчання:

Використання великих DataSet із даними про типи задач та їхні параметри [37]. Навчання моделей на GPU для підвищення швидкості (використання TensorFlow або PyTorch) [38].

Процес інтеграції:

- Формування наборів даних (DataSet): Збір та анотація даних із баз попередніх проектів. Стандартизація даних для коректного навчання моделі.

- Реалізація функцій нейронної мережі: Передбачення типу задачі на основі введеної інформації. Автоматична рекомендація методів розв’язання (чисельний, аналітичний) [39].

- Експлуатація моделей: Інтеграція моделей у систему через API (наприклад, Flask для Python або FastAPI). Постійне донавчання моделі на нових даних у фоновому режимі.

- Вимоги до продуктивності: Здатність обробляти вхідні дані за секунди, навіть за великої кількості параметрів. Оптимізація моделей для розгортання на сервері з обмеженими ресурсами [40].

3.3. Блок синтезу рішень (агрегація інформації необхідної для опису проекту технічної системи).

Зв'язок з інтерфейсною частиною, з відповідними базами даних, демонстрація схем, графіків тощо.

4. *Отримання розв'язків математичних задач* (три способи: 1. запозичення розв'язку з бази даних, 2. пропозиція аналітичного розв'язку (методу розв'язання), 3. використання чисельних методів розв'язання (вибір методу чисельного розв'язання)).

5. *Синтез розв'язку* (агрегація) (об'єднання елементів розв'язку задачі, які отримані різними методами).

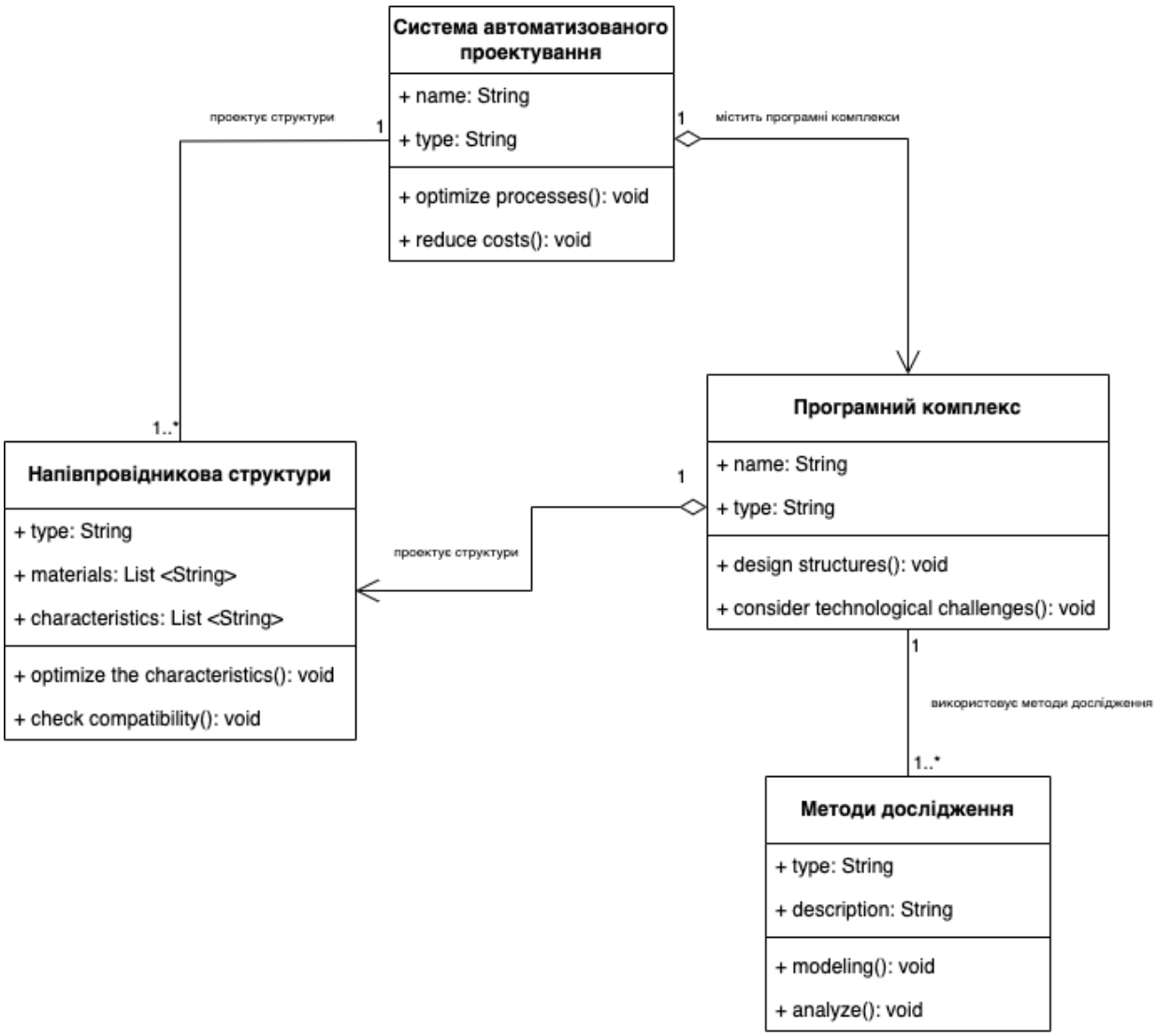

Рис.7.2.3 Діаграма класів використання

Ключові компоненти UML-схеми:

1. Клас "Система автоматизованого проектування"

- Атрибути:

  - назва (String)
  - тип (String)

- Методи:

  - оптимізувати процеси()
  - зменшити витрати()`

2. Клас "Напівпровідникова структура"

- Атрибути:

- тип структури (String)
- матеріали (List<String>)
- характеристики (List<String>)

- Методи:
- оптимізувати характеристики()
- перевірити сумісність()

3. Клас "Програмний комплекс"

- Атрибути:
- назва комплексу (String)
- тип алгоритмів (String)

- Методи:
- проектувати структури()
- враховувати технологічні виклики()

4. Клас "Методи дослідження"

- Атрибути:
- тип методу (String)
- опис методу (String)

- Методи:
- моделювати()
- аналізувати()

Взаємозв'язки:

- Асоціація між "Система автоматизованого проектування" і "Напівпровідникова структура" (система проектує структури).
- Асоціація між "Програмний комплекс" і "Методи дослідження" (програмні комплекси використовують методи дослідження для оптимізації проектування).
- Агрегація між "Система автоматизованого проектування" і "Програмний комплекс" (система містить програмні комплекси).

- Агрегація між "Програмний комплекс" і "Напівпровідникова структура" (комплекс проектує структури).

7.3. Роль елементів штучного інтелекту у системах автоматизованого проектування

Штучний інтелект (ШІ) все більше інтегрується в системи автоматизованого проектування (САП**Р**), оскільки ці технології дозволяють підвищити ефективність, точність та швидкість розробки складних технічних систем, зокрема в області проектування напівпровідникових пристроїв. У цьому підрозділі розглянуто основні напрямки застосування ШІ в САП**Р** та його роль у оптимізації процесів проектування.

Використання алгоритмів машинного навчання для оптимізації проектування

Машинне навчання (МН) є однією з основних складових штучного інтелекту, що активно застосовується для оптимізації різних етапів проектування. Основними напрямками є:

Аналіз великих даних. Процес проектування напівпровідникових пристроїв супроводжується величезними обсягами даних, таких як фізичні характеристики матеріалів, параметри електронних компонентів та вимоги до продуктивності. Алгоритми машинного навчання допомагають швидко обробляти ці дані, виділяючи закономірності та надаючи рекомендації щодо оптимальних параметрів пристроїв.

Прогнозування властивостей матеріалів:

ШІ може бути використано для прогнозування поведінки матеріалів при різних умовах експлуатації, що є критичним для проектування надійних напівпровідникових структур. Алгоритми глибокого навчання можуть моделювати фізичні процеси на мікрорівні, дозволяючи передбачити поведінку матеріалів без необхідності проведення дорогих експериментів.

Автоматизація задач проектування за допомогою еволюційних алгоритмів:

Еволюційні алгоритми (ЕА) і генетичні алгоритми є потужними інструментами, які можуть бути застосовані для автоматизації процесу проектування. Вони використовуються для оптимізації складних структур і параметрів пристроїв, що мають безліч взаємозв'язків між елементами:

Ітеративне вдосконалення проекту:

Еволюційні алгоритми дозволяють автоматично генерувати та тестувати варіанти проектів, поступово покращуючи їх характеристики за допомогою механізму "відбору". Це дозволяє знайти оптимальні рішення для проектування, враховуючи численні фактори та обмеження, що виникають на кожному етапі проектування.

Паралельне обчислення. Завдяки можливості паралельного виконання алгоритмів, еволюційні методи можуть обробляти одночасно велику кількість варіантів проектів, що значно скорочує час на прийняття рішень та дозволяє знаходити найбільш ефективні варіанти конструкцій.

Роль штучного інтелекту в автоматизованому аналізі та верифікації проектів

Одним із важливих аспектів проектування є перевірка правильності виконаних розрахунків та відповідності проекту заданим вимогам. ШІ дозволяє автоматизувати ці процеси, що значно зменшує ймовірність людських помилок:

Автоматичний синтаксичний та семантичний аналіз. Використання алгоритмів ШІ для аналізу проектної документації дозволяє автоматично виявляти помилки або невідповідності специфікаціям. Це дозволяє зменшити час на перевірку проекту і підвищити якість кінцевого продукту.

Інтелектуальні системи верифікації. На основі даних про параметри пристроїв ШІ може проводити автоматичну верифікацію проекту, виявляючи потенційні помилки та пропонуючи рішення для їх усунення. Такі системи можуть передбачити проблеми в роботі пристроїв, що зменшує необхідність у тестуванні прототипів.

Інтеграція ШІ в процеси прийняття рішень:

У складних проектах, де потрібно враховувати множинні фактори, включаючи економічні, фізичні та технічні обмеження, штучний інтелект може значно допомогти в *прийнятті оптимальних рішень*. Це досягається за допомогою наступних методів:

Рекомендаційні системи. Інтелектуальні системи, які пропонують оптимальні варіанти проектних рішень на основі аналізу попереднього досвіду або існуючих даних, допомагають проектувальникам швидше приймати правильні рішення. Це може бути особливо корисно при проектуванні нових, складних або нестандартних структур.

Інтелектуальні підтримувальні системи. ШІ може служити допоміжним інструментом для проектувальників, надаючи їм аналіз і рекомендації на кожному етапі проектування, що дозволяє значно підвищити продуктивність праці та якість проектів.

Майбутні напрямки розвитку ШІ в автоматизованому проектуванні:

У майбутньому роль штучного інтелекту в системах автоматизованого проектування продовжить зростати. Одним з найперспективніших напрямків є *застосування глибокого навчання та нейронних мереж* для автоматичного проектування пристроїв і компонентів, а також для інтеграції з іншими передовими технологіями, такими як *блокчейн* для забезпечення безпеки даних та *хмарні обчислення* для масштабованості і доступності.

ВИСНОВКИ

У процесі виконання дипломної роботи було розглянуто та вирішено низку завдань, що стосуються систем автоматизованого проектування напівпровідникових структур. Результати дослідження демонструють, що впровадження сучасних технологій, зокрема розпаралелювання обчислень та хмарних сервісів, здатне значно підвищити ефективність проектування та вдосконалити процес розробки складних електронних пристроїв.

Основні підсумки роботи включають такі положення:

1. Аналіз сучасних методів проектування:

Проведено глибинний аналіз існуючих підходів та інструментів автоматизованого проектування (САПР). Виявлено переваги та недоліки різних методів, що дозволило визначити основні напрямки вдосконалення систем проектування.

2. Розробка концептуальної архітектури програмної системи:

Запропоновано концептуальну модель архітектури програмного забезпечення для САПР, що поєднує в собі функції математичного моделювання, розпаралелювання обчислень та використання елементів штучного інтелекту. Ця модель дозволяє оптимізувати процес розробки та скоротити час на проектування напівпровідникових структур.

3. Інтеграція алгоритмів розпаралелювання:

Проведено аналіз методів декомпозиції та паралельної обробки даних. Запропоновано підходи до розпаралелювання обчислювальних задач, що забезпечують підвищення продуктивності та ефективності програмних комплексів.

4. Хмарні технології у САПР:

Розглянуто можливості інтеграції хмарних обчислень у процес проектування. Показано, що використання хмарних сервісів сприяє збільшенню масштабованості системи та спрощує спільну роботу над проектами.

5. Математичне моделювання:

Розроблено підхід до побудови математичних моделей напівпровідникових структур, що враховують специфіку сингулярно-збурених задач. Запропоновані моделі дозволяють підвищити точність аналізу параметрів пристроїв.

6. Практична апробація:

Реалізовано прототип програмної системи для вирішення окремих завдань автоматизованого проектування. Проведено тестування розроблених компонентів, яке підтвердило доцільність використання запропонованих рішень у промислових умовах.

Результати дипломної роботи підтверджують актуальність і практичну значущість автоматизованих систем проектування для напівпровідникових структур. Запропоновані рішення відкривають нові можливості для подальшого розвитку САПР, зокрема через інтеграцію сучасних технологій та підвищення рівня автоматизації.

СПИСОК ЛІТЕРАТУРИ

1. ДСТУ 3008-95 „Документація. Звіти у сфері науки і техніки. Структура і правила оформлення”.
2. ДСТУ 3973-2000„Правила виконання науково-дослідних робіт. Загальні положення”.
3. ДСТУ 2873-94 Системи оброблення інформації. Програмування. Терміни та визначення.
4. ДСТУ 2941-94 Системи оброблення інформації. Розроблення систем. Терміни та визначення.
5. ДСТУ ISO/IEC 90003:2006 Програмна інженерія. Настанови щодо застосування ISO 9001:2000 до програмного забезпечення (ISO/IEC 9003:2004, IDT).
6. ДСТУ 3918-99 (ISO/IEC 12207:1995) Інформаційні технології. Процеси життєвого циклу програмного забезпечення.
7. ДСТУ 3919-99 (ISO/IEC 14102:1995) Інформаційні технології. Основні напрямки оцінювання та відбору CASE-інструментів.
8. ДСТУ 4302:2004 Інформаційні технології. Настанови щодо документування комп`ютерних програм (ISO/IEC 6592:2000, MOD)
9. ДСТУ ISO/IEC TR 12182:2004 Інформаційні технології. Класифікація програмних засобів (ISO/IEC TR 12182:1998, IDT) .
10. ДСТУ ISO/IEC 14764-2002 Інформаційні технології. Супровід програмного забезпечення (ISO/IEC 14764:1999, IDT).
11. ГОСТ 19781-90 Програмне забезпечення систем обробки інформації. Терміни та визначення
12. ГОСТ 34.003-90. Інформаційна технологія. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Терміни та визначення.

13. ГОСТ 34.201-89. Інформаційна технологія. Комплекс стандартів на автоматизовані системи. Види, комплектність і позначення документів при створенні автоматизованих систем;
14. ГОСТ 34.601-90. Інформаційна технологія. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення.
15. ГОСТ 34.602-89. Інформаційна технологія. Комплекс стандартів на автоматизовані системи. Технічне завдання на створення автоматизованої системи.
16. РД 50-682-89. Методичні вказівки. Інформаційна технологія. Комплекс стандартів і керівних документів на автоматизовані системи. Загальні положення.
17. Shneiderman, B. (2010). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley.
18. Tufte, E. R. (2006). *The Visual Display of Quantitative Information*. Graphics Press.
19. Dix, A., Finlay, J., Abowd, G., & Beale, R. (2004). *Human-Computer Interaction*. Prentice Hall.
20. Kerpen, D. (2015). *The Art of Social Media: Power Tips for Power Users*. Penguin.
21. Date, C. J. (2004). *An Introduction to Database Systems*. Addison-Wesley.
22. Richardson, L., & Ruby, S. (2007). *RESTful Web Services*. O'Reilly Media.
23. Schneier, B. (2015). *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. W. W. Norton & Company.
24. OAuth.net. (n.d.). Retrieved from https://oauth.net/.
25. Aho A.V., Lam M.S., Sethi R., Ullman J.D. *Compilers: Principles, Techniques, and Tools*. Pearson, 2006.
26. Chomsky N. *Syntactic Structures*. Mouton, 1957.

27. Knuth D. *The Art of Computer Programming, Vol. 1*. Addison-Wesley, 1997.
28. Goodfellow I., Bengio Y., Courville A. *Deep Learning*. MIT Press, 2016.
29. Bishop C.M. *Pattern Recognition and Machine Learning*. Springer, 2006.
30. Ashenden P. *The Designer's Guide to VHDL*. Morgan Kaufmann, 2008.
31. Zwillinger D. *CRC Standard Mathematical Tables and Formulae*. CRC Press, 2018.
32. Meurer A. et al. *SymPy: Symbolic Computing in Python*. *Computing in Science & Engineering*, 2017.
33. Parr T. *The Definitive ANTLR 4 Reference*. Pragmatic Bookshelf, 2013.
34. Rumelhart D.E., Hinton G.E., Williams R.J. *Learning Representations by Back-Propagating Errors*. Nature, 1986.
35. Vaswani A. et al. *Attention Is All You Need*. NeurIPS, 2017.
36. Krizhevsky A., Sutskever I., Hinton G.E. *ImageNet Classification with Deep Convolutional Neural Networks*. NeurIPS, 2012.
37. LeCun Y., Bengio Y., Hinton G. *Deep Learning*. Nature, 2015.
38. Abadi M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2016.
39. Chollet F. *Deep Learning with Python*. Manning Publications, 2017.
40. He K., Zhang X., Ren S., Sun J. *Deep Residual Learning for Image Recognition*. CVPR, 2016.
41. Datt Panchal Aug 9, 2023 The Semiconductor Industry: Latest Trends, Future, and Career Path
42. S. Sze and K., Physics of Semiconductor Devices, New York: Wiley-Interscience, pp. 815, 2006, doi: 10.1002/ 0470068329
43. E.I. Adirovich, P.M. Karageorgii-Alkalaev and A.Iu. Leiderman, Currents Double Injection in Semiconductors, Moscow: Sov. radio, pp. 320, 1978.

44. A.Ya. Bomba, I.P. Moroz and M.V. Bojchura, "The Optimization of the Shape and Size of the Injection Contacts of the Integrated P-I-N-structures on the Base of Using the Conformal Mapping Method," Radio Electronics, Computer Science, Control, no. 1, pp. 14-27, 2021.
45. Baliga B 'Fundamentals of power semiconductor devices," 233 Spring Street, New York, NY 10013, USA: Springer, 2010, 1086 p.
46. A.B. Vasil'eva, V.F. Butusov and L.V. Kalachev, The Boundary Function Method for Singular Perturbation Problems. SIAM, Philadelphia, 1995, doi: 10.1137/1.9781611970784.
47. A.Ya. Bomba, I.P. Moroz, "Analysis of Nonlinear Processes in the P-I-N Diodes Plasma by the Perturbation Theory Methods"
48. A.Ya. Bomba, I.P. Moroz, "Analysis of Nonlinear Processes in the P-I-N Diodes Plasma by the Perturbation Theory Methods"
49. Bomba, A., Moroz, I., Boichura, M. Development and analysis of a mathematical model of plasma characteristics in the active region of integrated P-I-N-structures by the methods of perturbation theory and conformal mappings. Східно-Європейський журнал передових технологій. – №5 (113). – 2021. – С. 51-61. https://doi.org/10.15587/1729-4061.2021.243097
50. Бомба А.Я., Мороз І.П. Математичне моделю-вання дифузійно-дрейфового процесу в активній області *p-i-n* діодів з врахуванням розігріву та рекомбінації методами теорії збурень // Журнал обчислюваль-ної та прикладної математики. – №1 (135). – 2021. – С. 29-35.
51. *Kwok K.* Complete Guide to Semiconductor Devices. – New York: Wiley-Interscience, 2002. – 740 p. URL: https://ieeexplore.ieee.org/book/5271197
52. *Koshevaya S. V., Kishenko Ya. I., Smoilovskii M. I., Trapezon V. A.* Fast wideband modulators on p–i–n structures // Vyssh. Uchebn. Zaved., Radioelektron.; Radioelectron. Commun. Syst. – 1989. – No. 10. – P. 14-23.
53. *Адирович Э. И., Карагеоргий-Алкалаев П. М., Лейдерман А. Ю.* Токи

двойной инжекции в полупроводниках. Под ред. Гальперина. – М.: Радио, 1978. – 320 с.

ДОДАТКИ

Додаток 1

Ключові слова:

1. Напівпровідникові пристрої
2. Автоматизоване проектування
3. CAD (Computer-Aided Design) для напівпровідникових пристроїв
4. Методи проектування напівпровідникових пристроїв
5. Сучасні тенденції в галузі напівпровідникових пристроїв електроніки
6. Симуляція напівпровідникових електронних пристроїв
7. САПР
8. Системний аналіз предметної області
9. Системний аналіз в області проектування напівпровідникових пристроїв
10. UML в проектуванні комп'ютерних систем

1. Semiconductor devices
2. Automated design
3. CAD (Computer-Aided Design) for semiconductor devices
4. Design methods of semiconductor devices
5. Modern trends in the field of electronic semiconductor devices
6. Simulation of semiconductor electronic devices
7. CAD
8. System analysis of the subject area
9. System analysis in the field of designing semiconductor devices
10. UML in the computer systems design

Додаток 2

Сертифікат учасника XVII Всеукраїнської науково-практичної конференції "Інформаційні технології у професійній діяльності»

« - »

СЕРТИФІКАТ №2024-208

VI -

“ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРОФЕСІЙНІЙ ДІЯЛЬНОСТІ”

5 2024 , .

Самолюк Віталій Петрович

, . . .