Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Автоматні шаблони в технології проектування систем реального часу
(тема)

Виконав: студент 2 курсу, групи СКСм-18-1
Кучеренко І.О.
(прізвище, ініціали)

Спеціальність 123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп'ютерні системи
(повна назва освітньої програми)

Керівник доц. Кулак Е.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерна інженерія та управління

Кафедра Автоматизація проектування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп'ютерна інженерія

Освітня програма Спеціалізовані комп'ютерні системи

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 2019 р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ (ПРОЕКТ)

студентові Кучеренко Ірині Олександрівні
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Автоматні шаблони в технології проектування систем реального часу

затверджена наказом по університету від 04 11 2019 р. № №1624ст

2. Термін подання студентом роботи (проекту) 09 12 2019 р.

3. Вихідні дані до роботи (проекту)

САПР XILINX ISE

Мова опису апаратури VHDL

Системи реального часу

Темпоральні графи

Автоматне програмування

Автомат мура

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити)

Аналіз предметної області

Постановка задачі

Оцінка апаратних витрат по звіту

Дослідження поведінки однопроцесного та двопроцесного шаблону до імплементації

Дослідження поведінки шаблонів після імплементації

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

17 слайдів

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ п./п.	Назва етапів проекту (роботи)	Термін виконання етапів роботи (проекту)	Примітка
1	Отримання завдання	02.01.2019 – 03.01.2019	
2	Аналіз предметної області	03.01.2019-27.01.2019	
3	Аналіз джерел з проблемної галузі	27.01.2019 – 15.03.2019	
4	Оцінка апаратних витрат по звіту	15.03.2019 – 03.05.2019	
5	Дослідження поведінки однопроцесного та	03.05.2019 – 10.06.2019	
6	Дослідження поведінки однопроцесного та двопроцесного шаблону після імплементації	10.06.2019 – 11.07.2019	
7	Порівняння результатів дослідження	11.07.2019 – 07.09.2019	
8	Оформлення пояснювальної записки	07.09.2019 – 19.11.2019	
9	Оформлення графічного матеріалу	19.11.2019 – 06.12.2019	
10	Перевірка виконаного проекту керівником	06.12.2019 – 11.12.2019	

Дата видачі завдання 02.01.2019р.

Студент _____
(підпис)

Керівник роботи _____ _____ Доц.Кулак Е.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить у собі 76 сторінки, 22 рисунка, 18 посилань, 10 додатків.

ПАТЕРНИ ПРОГРАМУВАННЯ, СИСТЕМИ РЕАЛЬНОГО ЧАСУ, ТЕМПОРАЛЬНІ ГРАФИ, АВТОМАТНЕ ПРОГРАМУВАННЯ, АВТОМАТ МІЛІ, АВТОМАТ МУРА

Метою проекту є підвищення ефективності процесу розробки кінцевих керуючих автоматів з часовим логічним (не мікропрограмним) керуванням на мовах програмування та опису апаратури.

У ході виконання проекту були розглянуті шаблони програмування, автоматне програмування та темпоральні графи переходів. Розглянута поведінка однопроцесного та двупроцесного шаблону до та після імплементації.

ABSTRACT

Explanatory report includes 76 pages, 22 figures, 18 references, 10 supplements.

PATTERNS, REAL TIME SYSTEMS, TEMPORAL GRAPHICS, AUTOMATIC PROGRAMMING, MEALY MACHINE, MOORE MACHINE

The purpose of the project improves the development process of final control with a time machine logic (no firmware) running on programming languages and hardware description.

During the project implementation, programming templates, automatic programming, and temporal transition graphs were considered. The behavior of one- and two-process template before and after implementation is considered.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БІС – велика інтегральна схема

ГСА – граф-схема алгоритма

ДНФ – диз'юнктивна нормальна форма

МІС – мала інтегральна схема

МК – мікроконтроллер

НВІС – надвелика інтегральна схема

ОЗУ – оперативна пам'ять

ПЗУ – постійний запам'ятовувальний пристрій

ПЛІС – програмована логічна інтегральна схема

САПР – система автоматизованого проектування

ПК – пристрій керування

ASIC – application-specific integrated circuit (інтегральна схема для специфічного застосування)

CPLD – complex programmable logic device (програмовані логічні інтегральні схеми)

FPGA – field-programmable gate array (програмована користувачем вентильна матриця)

HDL – hardware description language (мова опису апаратури)

RTL – register transfer level (рівень регістрових передач)

VHDL – hardware description language (мова опису апаратури інтегральних схем)

ВСТУП

Системи автоматизованого управління зазвичай побудовані так, що в них можна виділити системи управління та керовані об’єкти. Дотримуючись цієї концепції, системи управління на основі кінцевих автоматів можна розділити на дві частини: керуючу частину, відповідальну за логіку поведінки та керовану частину, відповідальну за виконання дій, обраних для виконання керуючою частиною.

Серед усіх керуючих пристроїв можна виділити пристрої логічного управління, у яких управляючі впливи представляються у двійковому алфавіті. Оскільки для реалізації керуючої частини в таких пристроях, як правило, використовуються кінцеві автомати, вони називаються керуючі автомати. Подібні пристрої широко застосовуються в системах Internet of Things.

Будь-який цифровий пристрій, котрий реалізує алгоритм обробки інформації та її управління може бути реалізовано апаратним або програмно-апаратним способом.

При написанні алгоритму функціонування цифрових пристроїв використовуються автоматні програми, у яких розділяється написання логіки програми та описання семантики. Автоматні програми мають тільки три вида функцій: функції переходів, функції виходів та функції реалізації затримок і переходів у новий стан. Вони мають шаблон та використовують оператори вибору, умовні оператори та функції таймеру або фронту.

Сучасні засоби САПР дозволяють в наукових дослідженнях піти від неефективних ручних розрахунків і синтезу цифрових схем. Після синтезу в XILINX формується звіт, на підставі якого можна оцінити апаратурні витрати з точки зору використання (заповнення) ресурсів ПЛІС, часові та інші характеристики.

Необхідно з'ясувати, наскільки було б адекватним використання оцінки витрат апаратури, що надаються в звіті у вигляді кількості слайсів, тригерів і

LUT замість оцінки числа входів вентилів (по Квайну) по RTL-схемі при дослідженні витрат апаратури різних варіантів схем на прикладі мікропрограмних автоматів Мілі і Мура.

Об’єкт дослідження: кінцеві часові керуючі (не мікропрограмні) автомати і оцінка апартурних витрат за допомогою САПР.

Предмет дослідження: 1) шаблони опису кінцевих часових керуючих автоматів на мові опису апаратури VHDL, придатних для автоматизовнного проектуання, 2) кореляція між апаратними витратами, що надаються в звіті САПР XILINX ISE і апаратурними витратами по Квайну, які оцінюються по RTL-схемі після синтезу.

Ціль роботи: підвищення ефективності процесу розробки кінцевих керуючих автоматів з часовим логічним керуванням на мові опису апаратури VHDL шляхом використання чіткого шаблону.

1 АНАЛІЗ ПРОБЛЕМИ. ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз проблеми. Огляд автоматних шаблонів та технологій проектування систем реального часу

Пристрої логічного управління, побудовані на основі кінцевих автоматів, функціонують одразу у двох вимірах: у автоматному часі та у реальному часі. Автоматний час вимірюється в автоматних тактах. Автоматний такт – дискретних відрізок часу за який автомат переходить з одного стану в інший. Тривалість такого такту визначається частотою синхросигналу Clk. Реальний час визначається часовими параметрами алгоритму функціонування пристрою. Виникає протиріччя, яке потрібно вирішувати. Одним з рішенням даної проблеми може бути використання темпоральних графів.

1.1.1 Шаблони (pattern) в проектуванні

Як показано в [1], проектування об'єктно-орієнтованих програм − нелегка справа, а якщо їх потрібно використовувати повторно, то все стає ще складніше. Необхідно підібрати відповідні об'єкти, віднести їх до різних класів, дотримуючись розумної ступені деталізації, визначити інтерфейси класів і ієрархію наслідування і встановити суттєві відносини між класами.

У багатьох об'єктно-орієнтованих системах можна зустріти повторювані патерни, що складаються з класів і взаємодіючих об'єктів. З їх допомогою вирішуються конкретні завдання проектування, в результаті чого об'єктно-орієнтований дизайн стає більш гнучким, елегантним, і їм можна скористатися повторно.

Патерни проектування спрощують повторне використання вдалих проектних і архітектурних рішень. За допомогою патернів можна поліпшити якість документації і супровід існуючих систем, дозволяючи явно описати

взаємодії класів і об'єктів, а також причини, за якими система була побудована так, а не інакше. Простіше кажучи, патерни проектування дають розробнику можливість швидше знайти «правильний» шлях.

1.1.1.1 Патерни проектування

У загальному випадку патерн складається з чотирьох основних елементів:

- Ім'я. Пославшись на нього, ми можемо відразу описати проблему проектування; її рішення і їх наслідки. Присвоєння паттернам імен дозволяє проектувати на більш високому рівні абстракції.
- Завдання. Опис того, коли слід застосовувати патерн. Необхідно сформулювати завдання і його контекст. Може описуватися конкретна проблема проектування, наприклад спосіб представлення алгоритмів у вигляді об'єктів. Іноді відзначається, які структури класів або об'єктів свідчать про негнучкий дизайн. Також може включатися перелік умов, при виконанні яких має сенс застосовувати даний патерн.
- Рішення. Опис елементів дизайну, відносин між ними, функцій кожного елемента. Конкретний дизайн або реалізація не маються на увазі, оскільки патерн − це шаблон, який можна застосовувати в самих різних ситуаціях. Просто дається абстрактний опис завдання проектування і того, як він може бути вирішений за допомогою якогось вельми узагальненого поєднання елементів (в нашому випадку класів і об'єктів).
- Результати − це наслідки застосування патерну і різного роду компроміси. Хоча при описі проектних рішень про наслідки часто не згадують, знати про них необхідно, щоб можна було вибрати між різними варіантами і оцінити переваги і недоліки даного патерну.

Патерн проектування іменує, абстрагує і ідентифікує ключові аспекти структури спільного рішення, які і дозволяють застосувати його для створення повторно використовуваного дизайну. У виокремлюванні беруть участь класи і екземпляри, їх роль і відносини, а також функції. При описі кожного патерну

увага акцентується на конкретному завданні об'єктно-орієнтованого проектування. Аналізується, коли слід застосовувати патерн, чи можна його використовувати з урахуванням інших проектних обмежень, які будуть наслідки застосування методу.

1.1.1.2 Патерн State

Призначення: дозволяє об'єкту варіювати свою поведінку в залежності від внутрішнього стану. Ззовні створюється враження, що змінився клас об'єкту.

Розглянемо клас TCPConnection (рисунок 1.1), за допомогою якого представлено мережеве з'єднання. Об'єкт цього класу може перебувати в одному з декількох станів: Established (встановлено), Listening (прослуховування), Closed (закрито). Коли об'єкт TCPConnection отримує запити від інших об'єктів, то в залежності від поточного стану він відповідає по різному.

Наприклад, відповідь на запит Open (відкрити) залежить від того, чи знаходиться з'єднання в стані Closed або Established. Патерн State описує, яким чином об'єкт TCPConnection може вести себе по різному, перебуваючи в різних станах.

Основна ідея цього патерну полягає в тому, щоб ввести абстрактний клас TCPState для представлення різних станів з’єднання. Цей клас оголошує інтерфейс, загальний для всіх класів, що описують різні робочі стани. У підкласах TCPState реалізовано поведінку, специфічну для конконкретного стану. Наприклад, в класах TCPEstablished і TCPClosed реализована поведінка, характерна для станів Established і Closed відповідно.

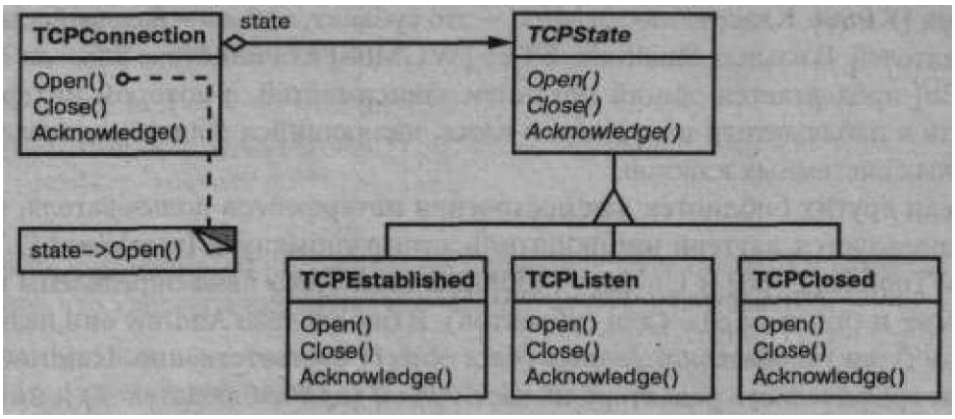


Рисунок 1.1 – Інтерфейс класів

Клас TCPConnection зберігає у себе об'єкт стану (екземпляр деякого підкласу TCPState), що представляє поточний стан з'єднання, і делегує всі залежні від стану запити цьому об'єкту. TCPConnection використовує свій екземпляр підкласу TCPState для виконання операцій, присутніх тільки в даному стану з'єднання.

При кожній зміні стану з'єднання TCPConnection змінює свій об'єкт-стан. Наприклад, коли встановлене з'єднання закривається, TCPConnection замінює екземпляр класу TCPEstablished екземпляром TCPCIosed.

Потрібно використовувати патерн State в наступних випадках:

– коли поведінка об'єкта залежить від його стану і має змінюватися під час виконання;

– коли в коді операцій зустрічаються умовні оператори, які складаються з багатьох гілок, в яких вибір гілки залежить від стану.

Зазвичай в такому випадку стан представлено переліком констант. Часто одна і та ж структура умовного оператора повторюється в кількох операціях. Патерн State пропонує помістити кожну гілку в окремий клас (рис. 1.2). Це дозволяє трактувати стан об'єкта як самостійний об'єкт, який може змінюватися незалежно від інших.

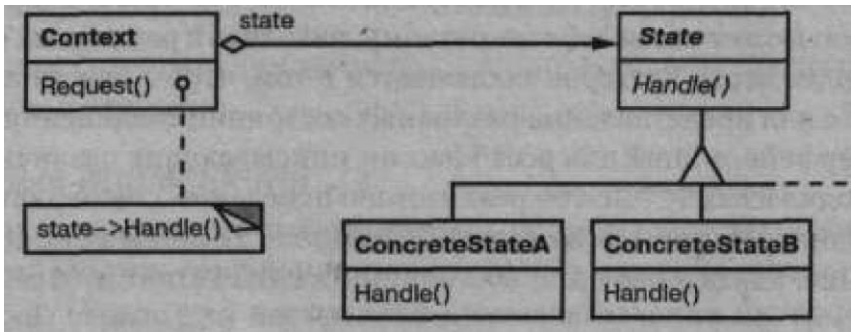


Рисунок 1.2 – Структура класів

Context (TCPConnection) − контекст:

− визначає інтерфейс, який представляє інтерес для клієнтів;

− зберігає екземпляр підкласу ConcreteState, яким визначається поточний стан.

State (TCPState) − визначає інтерфейс для інкапсуляції поведінки, асоційованого з конкретним станом контексту Context.

Підкласи ConcreteState (TCPEstablished, TCPListen, TCPClosed) – конкретний стан. Кожен підклас реалізує поведінку, асоційовану з деяким станом контексту Context.

Відносини:

− клас Context делегує залежні від стану запити поточному об'єкту ConcreteState;

− контекст може передати себе в якості аргументу об'єкту State, котрий буде обробляти запит. Це дає можливість об'єкту-стану при необхідності отримати доступ до контексту;

− Context − це основний інтерфейс для клієнтів. Клієнти можуть конфігурувати контекст об'єктами стану State. Один раз сконфігурував контекст, клієнти вже не повинні безпосередньо зв'язуватися з об'єктами стану.

Результати використання патерну State.

Локалізує залежну від стану поведінку і ділить його на частини, відповідні станам. Патерн State поміщує всю поведінку, ассоціірована з

конкретним станом, в окремий об'єкт, оскільки залежний від стану код цілком знаходиться в одному з підкласів класу State, то додавати нові стани і переходи можна просто шляхом порождення нових підкласів. Замість цього можна було б використовувати дані члени для визначення внутрішніх станів, тоді операції об'єкта Context перевіряли б ці дані. Але в такому випадку схожі умовні оператори або умовні оператори були б розкидані по всьому коду класу Context. При цьому додавання нового стану зажадало б від зміни декількох операцій, що ускладнило б супровід. Патерн State дозволяє вирішити цю проблему, але одночасно породжує іншу, оскільки поведінка для різних станів виявляється розподіленою між декількома підкласами State. Це збільшує число класів. Звичайно, один клас компактніше, але якщо станів багато, то такий розподіл ефективніше, тому що в противному випадку довелося б мати справу з громіздкими умовними операторами. Наявність громіздких умовних операторів небажано, так само як і довгих процедур. Вони занадто монолітні, ось чому модифікація і розширення коду стає проблемою. Патерн стану пропонує більш вдалий спосіб структурування, який залежить від стану коду, логіка, що описує переходи між станами, більше не укладена в монолітні оператори if або switch, а розподілена між підклассами State. При інкапсуляції кожного переходу і дії клас стану стає повноцінним об'єктом. Це покращує структуру коду і проясняет його призначення.

Робить явними переходи між станами. Якщо об'єкт визначає свій поточний стан виключно в термінах внутрішніх даних, то переходи між станами не мають явного подання; вони проявляються лише як привласнення деяких змінних. Введення окремих об'єктів для різних станів робить переходи більш явними. Крім того, об'єкти State можуть захистити контекст Context від неузгодженості внутрішніх змінних, оскільки переходи з точки зору − це атомарні дії. Для здійснення переходу треба змінити значення тільки однієї змінної (об'єктної змінної State в класі Context), а не кількох.

1.1.2 Системи логічного управління керування автоматами. Візуальне представлення графом переходів

В даний час при програмній реалізації алгоритмів логічного керування технологічними процесами поряд з програмованими логічними контролерами все частіше використовуються комп'ютери, призначені для роботи в виробничих умовах [2].

Розглянемо автоматні граф-схеми, в яких в операційних вершинах формуються або зберігаються лише поодинокі і нульові значення змінних. Автоматні граф-схеми розіб'ємо на два класи: граф-схеми алгоритмів і граф-схеми програм [3].

При цьому будемо розрізняти граф-схеми алгоритмів (ГСА) за такими основними ознаками:

- наявності внутрішніх зворотних зв'язків;
- використовуваним типами змінних;
- наявності дешифратора станів;
- наявності замовчувань і неоднозначних змінних;
- наявності змінних, які можуть неодноразово змінюватися за один прохід граф-схеми;
- місця видачі значень вихідних змінних.

Використовуючи деякі з цих ознак, виділимо п'ять підкласів граф-схем алгоритмів:

- граф-схеми алгоритмів з внутрішніми зворотніми зв'язками і видачею значень вихідних змінних в будь-якій операторній вершині;
- граф-схеми алгоритмів без внутрішніх зворотних зв'язків і дешифратора станів, в яких видача значень вихідних змінних виконується в кінці «тіла» граф-схеми;
- граф-схеми алгоритмів без внутрішніх зворотних зв'язків, що враховують особливості використовуваних керуючих конструкцій мови програмування;

– граф-схеми алгоритмів без внутрішніх зворотних зв'язків, але з дешифратором станів і видачею значень вихідних змінних в кінці «тіла граф-схеми, які не містять вихідних змінних, неодноразово змінюються за один прохід граф-схеми;

– граф-схеми алгоритмів без внутрішніх зворотних зв'язків, але з дешифратором станів і видачею значень вихідних змінних в кінці «тіла граф-схеми, що містять вихідні змінні, які можуть неодноразово змінюватися за один прохід граф-схеми.

Графи переходів мають свою класифікацію [4].

Граф переходів, в якому у кожній вершині явно вказані значення кожної вихідної змінної, назвемо графом переходів автомата Мура з явним завданням значень всіх вихідних змінних. У цьому випадку значення виходів відповідають номеру вершини і не залежать від передісторії. З цієї причини такий граф переходів просто розуміється і в нього легко вносяться зміни.

Граф переходів, в якому у вершинах, крім явно визначених значень одних змінних, застосовуються також неявно, але однозначно визначені значення інших змінних, назвемо графом переходів автомата Мура з неявним завданням значень вихідних змінних. Неявне завдання змінної в вершині відображається прочерком, який в даному випадку може бути замінений тільки одним значенням булевої змінної. Ця зміна в розглянутій вершині зберігає те значення, яке присвоюється їй у всіх «суміжних» з нею вершинах.

Графи переходів цього типу читаються дещо гірше, ніж графи переходів автоматів Мура першого типу, проте їх доцільно використовувати для зменшення обсягу пам'яті програм.

Граф переходів, в якому у вершинах, крім явно визначених значень одних змінних, використовуються також неявно і неоднозначно задані значення інших змінних, назвемо графом переходів автомата Мура з неоднозначним завданням значень вихідних змінних.

Графи переходів цього класу можуть містити для однієї і тієї ж задачі менше число вершин, ніж графи переходів автоматів Мура двох інших

зазначених вище різновидів, так як в цьому випадку в вершині замість одного значення змінної можуть формуватися різні значення тієї ж змінної, що породжує в цій вершині різні набори значень вихідних змінних і забезпечує еквівалентність однієї такої вершини декільком, повністю визначеним.

Ця перевага з точки зору компактності опису і скорочення довжини програми породжує одночасно і головний недолік графів переходів цього − труднощі їх читання і розуміння. Саме з цієї причини такі графи переходів не рекомендується використовувати в якості мови спілкування.

Специфікацію завдань доцільно проводити за допомогою двох перших моделей графів переходів автоматів Мура.

Аналогічна класифікація може бути запропонована для графів переходів автоматів Мілі, в яких значення вихідних змінних формуються не в вершинах графів переходів, а на дугах. У багатьох задачах логічного управління перехід від моделі автомата Мура до моделі автомата Мілі не зменшує числа станів (вершин графа переходів).

Універсальний метод побудови структурованих граф-схем алгоритмів був запропонований Ашкрофтом і Манною, який по неструктурованій ГСА1 або ГСА2 будує структуровану ГСА4 або ГСА5 з багатозначним кодуванням станів [5].

Однак, цей метод має низку недоліків, що не дозволяють отримувати зрозумілу граф-схему алгоритмів:

− в явному вигляді не використовується поняття «стан»;

− не робиться розходження за типами автоматів і типами використовуваних змінних;

− використовується тільки один тип кодування фрагментів неструктурованого граф-схеми алгоритмів при об'єднанні в канонічну структуру;

− через зв'язок фрагментів структурованих граф-схем алгоритмів за даними візуально дуже складно виявити генеруючі «контури»;

− глибина структурованих граф-схем алгоритмів не обмежена одним переходом, що може не дозволити використовувати значення деяких внутрішніх змінних в інших компонентах алгоритмів управління;

− при застосуванні в якості вихідної ГСА1 в ній не усувається неоднозначність значень виходів;

− при використанні в якості вихідної ГСА2, в ній не тільки не усувається неоднозначність значень виходів, але крім того вона може містити велику кількість бітових проміжних і прапорів змінних (в тому числі з замовчуваними значеннями), які не зникають при структурі і до яких додається ще одна багатозначна змінна;

− людина звикає і розуміє вихідну форму подання алгоритму управління і зазвичай психологічно не готова до роботи з іншими формами представлення алгоритму;

− запропонований метод орієнтований на використання мов високого рівня.

Так як зазначений метод призначений для побудови структурованих граф-схем алгоритмів, то з її змісту випливає, що якщо вихідна граф-схема вже структурована, то до неї цей метод застосовуватися не повинен. З цієї причини, якщо, наприклад, в якості вихідної задана ГСА2, то виходячи з викладеного вона не повинна піддаватися подальшим перетворенням.

Таким чином, в цілому ідея, запропонована Ашкрофтом і Манною порочна, так як не має з чого спочатку будувати неструктуровану граф-схему алгоритму, щоб потім її структурувати. Слід або відразу будувати структуровану граф-схему з дешифратором станів для прийнятого варіанта кодування, або, що більш доцільно, проводити початковий опис у вигляді графа переходів для обраного типу автомата, в якому, по-можливості, повинні бути відсутніми замовчувані значення змінних, і який ізоморфно реалізується, наприклад, за допомогою керуючої конструкції switch мови С, що виробляє одночасно структурування і забезпечує доступ до будь-якого значення

багатозначної змінної. Підхід застосовний і для мов низького рівня, наприклад, мов інструкцій програмованих логічних контролерів.

У одержувану програму легко вносяться зміни. Вони досить просто перевіряються (в якості тесту використовується граф переходів) і з огляду на ізоморфізм з первинним описом легко верифіцуються.

Графи переходів по формі зображення є в загальному випадку «істотно площинними», що дозволяє відображати алгоритми управління в більш природній формі, ніж у вигляді граф-схеми або діаграми Графсет, які за стандартами зазвичай зображуються у формі, що наближається до «лінійної», в напрямку зверху вниз. Така форма подання відповідає послідовному книжковому зображенню і читання текстів, але не відповідає площинному (паралельному) зображенню картин, більш зручних для сприйняття людиною. При цьому, природно, що графи переходів повинні бути в максимальному ступені планарними.

При цьому відзначимо також, що графи переходів на відміну від таблиць переходів [6], що містять зазвичай велике число порожніх клітин, більш доступні для огляду і практично застосовні для задач великої розмірності. Основна перевага графів переходів, що дозволяє описувати завдання такої розмірності, полягає в тому, що якщо граф ортогоналізован, то перехід між двома вершинами визначається не всіма вхідними змінними, як це передбачено в позначках всіх дуг графа, як це має місце при використанні таблиць переходів, а тільки тими з вхідних змінних, які позначають дугу між зазначеної парою вершин.

При усуненні суперечності не ортогоналізацією, а розстановкою приорітетів, замовчування позначень деяких вхідних змінних зменшує читаємість графів переходів, так як для дуг з меншими пріоритетами, що виходять з деякої вершини, не вдається відразу визначити (прочитати) перелік всіх змінних, від яких залежить кожен перехід по цим дугам. Для визначення зазначеного переліку в цьому випадку доводиться аналізувати позначки всіх дуг, що виходять з даної вершини.

1.1.3 Автоматне програмування. Автоматні моделі

В даний час запропоновані, досліджені і з успіхом застосовуються різні математичні моделі, в назвах яких використовується слово «автомат». Вони мають багато спільного в своїх основах, але розрізняються, часто істотно, в деталях [7].

Причина різноманітності автоматних моделей пояснюється широтою області їх застосування. Автомати є незамінним інструментом в таких далеких один від одного областях як, наприклад:

- математична лінгвістика;
- логічне управління;
- моделювання поведінки людини;
- комунікаційні протоколи;
- теорія формальних мов, обчислюваності і обчислювальної складності.

Виявити загальне в різних автоматних моделях можна, якщо розглянути автомати як окремий випадок динамічних систем. Зазвичай терміном «динамічна система» в техніці, природі, життя і т. д. позначають систему, процеси якої розвиваються в часі. Стан системи в кожен момент часу характеризують безліччю узагальнених координат. Процеси в динамічній системі описуються рівняннями різних типів щодо узагальнених координат.

Динамічні системи можна поділити на декілька класів в залежності від наступних факторів.

- Модель часу. Час може вважатися поточним безперервно або дискретно. У першому випадку час змінюється на континуумі, у другому − на рахунковій множині, елементи якого називаються тактами.
- Розмірність системи. Число узагальнених координат може бути кінцевим або рахунковим.
- Потужність множин координат. Кожна з узагальнених координат може набувати значень з кінцевої, рахункової або континуальної множини.

У фізиці і техніці часто використовуються системи, в яких безперервні і узагальнені координати також змінюються на континуумі. Для опису процесів в таких системах використовуються диференціальні рівняння.

Якщо ж час дискретний, але узагальнені координати приймають значення з континуальних множин, то для опису процесів використовуються різницеві рівняння. Ті системи, в яких час дискретний, число узагальнених координат може приймати значення з кінцевої множини, називають кінцевими динамічними системами. Кінцеві автомати належать до класу кінцевих динамічних систем.

Системи, які відрізняються від кінцевих тим, що число узагальнених координат або ж безліч значень координат може бути нескінченними, утворюють більш загальний клас. До цього класу належать, наприклад, стрічки Тьюринга.

Кінцева динамічна система з дискретним часом, стан якої в кожен такт t характеризується кінцевим числом узагальнених координат Y(t) (|Y| =n).

На вхід системи подається кінцеве число вхідних впливів X(t) (|X| =m).

Така кінцева динамічна система називається кінцевим автоматом, якщо стан системи в кожен такт однозначно визначається станом системи в попередній такт і значеннями вхідних впливів або в поточний, або в попередній такт:

$$Y(t) = f\,(X\,(t), Y(t-1)); \tag{1.1}$$

$$Y(t) = f\,(X\,(t-1), Y(t-1)) \tag{1.2}$$

Таким чином, для опису поведінки кінцевих автоматів використовуються рекурентні співвідношення певного виду. Якщо використовується рекуррентное співвідношення (1.1), то автомат називається автоматом першого роду, а якщо співвідношення (1.2) − автоматом другого роду.

Можна показати, що поняття «кінцевий автомат» охоплює і ті кінцеві системи, стани яких визначаються передісторією будь-якої наперед заданої кінцевої довжини. Однак воно не охоплює системи, в яких стан визначається статистично або ж залежить від всієї передісторії. Таким чином, клас кінцевих динамічних систем, які «пам'ятають» кінцеве число попередніх тактів не ширше класу систем, які «пам'ятають» лише один такт.

У теорії формальних мов вивчаються, так звані, абстрактні автомати (Звані також абстрактними машинами або абстрактними обчислювачами).

Внутрішня структура абстрактних автоматів не розкривається. Інтерес при їх вивченні представляє тільки обчислювальна потужність − клас мов, які може розпізнавати машина.

У логічному управлінні розглядаються структурні автомати. Вони виступають в ролі керуючих пристроїв в системах управління. Інтерес тут представляє число паралельних входів і виходів автомата, його зв'язку з об'єктом управління та іншими елементами системи, простота реалізації.

1.1.4 Системи реального часу

Система називається системою реального часу [11], якщо правильність її функціонування залежить не тільки від логічної коректності обчислень, але і від часу, протягом якого це вираховується. Тобто для подій, що відбуваються в такій системі, то, коли ці події відбуваються, так само важливо, як логічна коректність самих подій.

Також, кажуть, що система працює у реальному часі, якщо її швидкодія адекватна швидкості протікання фізичних процесів на об'єктах контролю або управління. Так як оточуючий нас світ дуже різноманітний, тут доречно додати, що маються на увазі саме ті процеси, які безпосередньо пов'язані з функціями, виконуваними конкретної системою реального часу. Тобто система управління повинна зібрати дані, провести їх обробку відповідно до заданих алгоритмів і видати керуючі впливи за такий проміжок часу, який

забезпечує успішне вирішення поставлених перед системою завдань. З наведених визначень випливає кілька цікавих висновків.

По-перше, практично всі системи промислової автоматизації є системами реального часу.

По-друге, приналежність системи до класу систем реального часу ніяк не пов'язана з її швидкодією. Наприклад, якщо ваша система призначена для контролю рівня грунтових вод, то, навіть виконуючи вимірювання з періодичністю один раз за півгодини, вона буде працювати у реальному часі.

Вихідні вимоги до часу реакції системи і інших часових параметрів визначаються або технічним завданням на систему, або просто логікою її функціонування. Наприклад, шахова програма, думаюча над кожним ходом більше року, працює явно не в реальному часі, так як шахіст швидше за все не доживе до кінця партії.

Прийнято розрізняти системи «жорсткого» та «м'якого» реального часу. Системою «жорсткого» реального часу називається система, де нездатність забезпечити реакцію на будь-які події в заданий час є відмовою і веде до неможливості вирішення поставленого завдання.

Наслідки таких відмов можуть бути різні, від протоки дорогоцінної вологи на лінії по розливу алкогольних напоїв до більших неприємностей, якщо, наприклад, вчасно не спрацювала система аварійних блокувань атомного реактора. Однозначної думки про те, який час реакції властиво «жорстким» системам, немає. Більш того, зі збільшенням швидкодії мікропроцесорів цей час має тенденцію до зменшення, і якщо раніше в якості кордону називалося значення 1 мс, то зараз, як правило, називається час близько 100 мкс.

Точного визначення для «м'якого» реального часу не існує, тому будемо вважати, що сюди відносяться всі системи реального часу, але вони не попадають в категорію «жорстких». Так як система «м'якого» реального часу може не встигати все робити завжди в заданий час, виникає проблема визначення критерії успішності (нормальності) її функціонування.

1.1.5 Темпоральні графи переходів

Системи автоматизованого управління зазвичай побудовані так, що в них можна виділити системи управління та керовані об’єкти [12]. Дотримуючись цієї концепції, системи управління на основі кінцевих автоматів можна розділити на дві частини: керуючу частину, відповідальну за логіку поведінки та керовану частину, відповідальну за виконання дій, обраних для виконання керуючою частиною.

Серед усіх керуючих пристроїв можна виділити пристрої логічного управління, у яких управляючі впливи представляються у двійковому алфавіті. Оскільки для реалізації керуючої частини в таких пристроях, як правило, використовуються кінцеві автомати, вони називаються керуючі автомати. Подібні пристрої широко застосовуються в системах Internet of Things. Будь-який цифровий пристрій, котрий реалізує алгоритм обробки інформації та її управління може бути реалізовано апаратним або програмно-апаратним способом.

При програмно-апаратному способі алгоритм реалізується на апаратно-орієнтованій мові програмування. Найпопулярніша мова для цього – C зі спеціальними бібліотеками. Апаратна сторона реалізується, як правило, на різноманітних мікроконтролерах.

При апаратному способі алгоритм описується на мові описання апаратури HDL, синтезується інструментальними засобами систем автоматизованого проектування (САПР) та імплементується у ПЛІС або ASIC.

При написанні алгоритму функціонування цифрових пристроїв використовуються автоматні програми, у яких розділяється написання логіки програми та описання семантики. Автоматні програми мають тільки три вида функцій: функції переходів, функції виходів та функції реалізації затримок і переходів у новий стан. Вони мають шаблон та використовують оператори вибору, умовні оператори та функції таймеру або фронта.

Пристрої логічного управління, побудовані на основі кінцевих автоматів, функціонують одразу у двох вимірах: в автоматному часі та у реальному часі. Автоматний час вимірюється в автоматних тактах.

Автоматний такт – дискретних відрізок часу за який автомат переходить з одного стану в інший. Тривалість такого такту визначається частотою синхросигналу Clk. Реальний час визначається часовими параметрами алгоритму функціонування пристрою. Для усунення протиріччя пропонується використання темпорального графа переходів, який описується розширеною функцією переходів (1.3).

$$Z(t+1) = f(X(t), Z(t), T) \qquad (1.3)$$

У такому графі кожному стану становиться у відповідність затримка T, яка визначається числом автоматних тактів, протягом яких автомат знаходиться у даному стані. При цьому темпоральний граф переходів є не тільки візуальним відображенням алгоритму функціонування кінцевого керуючого автомата, а й його повною математичною моделлю. Затримка в кожній вершині темпорального графа переходів реалізується через петлю, умовами для якої є підрахунок числа тактів Clk, що схемно реалізується лічильником в ПЛІС або таймером з перериванням в МК. Це означає, що темпоральний граф ідеально підходить для розробки шаблону опису алгоритмів функціонування кінцевих автоматів у стилі автоматного програмування.

1.2 Апаратні витрати по звіту

Сучасні засоби САПР дозволяють в наукових дослідженнях піти від неефективних ручних розрахунків і синтезу цифрових схем. Так система XILINX дозволяє виконувати автоматизований синтез схем, описаної на мові VHDL або Verilog.

Після синтезу формується звіт, на підставі якого можна оцінити апаратурні витрати з точки зору використання (заповнення) ресурсів ПЛІС, часові та інші характеристики.

Крім того, система дозволяє переглядати графічний варіант внутрішньої синтезується моделі (netlist).

Існує два способи графічного відображення внутрішньої моделі: технологічний, де схема представлена тригерами і LUT, як неподільними елементами на нижчому рівні перегляду; і RTL (рівень регістрових передач), в якій тригери представлені цілісними неподільними елементами, а комбінаційна частина представляється на нижньому рівні перегляду в базисі І АБО НЕ.

Для багатьох досліджень RTL уявлення є дуже корисним і в плані структури та вмісту синтезованої схеми, і в плані оцінки витрат апаратури.

Оцінка витрат апаратури по Квайну, наприклад, по RTL схемі прийнятна, якщо схема містить не більше кількох десятків вентилів, в разі схем більшого розміру це стає неприйнятним.

Необхідно з'ясувати, наскільки було б адекватним використання оцінки витрат апаратури, що надаються в звіті у вигляді кількості слайсів, тригерів і LUT замість оцінки числа входів вентилів (по Квайну) по RTL схемі при дослідженні витрат апаратури різних варіантів схем на прикладі мікропрограмних автоматів Мілі і Мура.

1.3 Постановка задачі

Після аналізу таких тем як патерни програмування, системи логічного управління автоматами, графи переходів, автоматне програмування, автоматні шаблони, автомати на мовах опису апаратури, автоматні шаблони на HDL, системи реального часу, часові автомати, темпоральні графи переходів було з'ясовано, що для досягнення цілі потрібно вирішити наступні задачі:

1) Розробити двопроцесний шаблон опису кінцевих часових керуючих автоматів на мові опису апаратури VHDL, придатного для автоматизованного проектування.

2) Описати тестовий приклад на однопроцесному і двопроцесному шаблонах.

3) Просинтезувати і проімплементувати два шаблона в САПР, дослідити поведінку однопроцесного та двопроцесного шаблону до імплементації та після неї. Порівняти результати та скласти висновки дослідження.

4) Виконати оцінку апаратних витрат модулей по RTL-схемі із пакета XILINX ISE 10.1.01, плата Spartan 3E, мікросхема FPGA XC3S500E, Package FG 320 (xc3s500e-4fg320). У тому числі дослідити яким чином оцінювати витрати на один тригер.

5) Виконати оцінку апаратних витрат по звіту після синтезу, оцінити загальні витрати на LUT по Квайну.

6) Виконати порівняльний аналіз оцінок витрат по звіту, по RTL-схемі і по технологічному рівню.

7) Виконати аналогічну оцінку апаратних витрат модулей по звіту, по RTL-схемі і по технологічному рівню із пакета XILINX ISE 10.1.01, плата Virtex 5, мікросхема FPGA XC5VLX30, Package FF324 (xc5vlx30-3ff324).

2 ОЦІНКА АПАРАТНИХ ВИТРАТ ПО ЗВІТУ

2.1 Оцінка витрат апаратури модулів по RTL-схемі, плата Spartan 3E

Оцінка витрат апаратури модулів проводиться по RTL-схемі з пакета XILINX ISE 10.1.01, плата Spartan 3E, мікросхема FPGA XC 3 S 500 E, Package FG 320 (xc3s500e-4fg320).

Для дослідження взято приклади автоматів Мура, описаних в [13] і автоматів Мілі, описаних [12], а також два модуля з [16] − автомат Мілі і пристрій діагностування.

Таблиця 2.1 − Апаратурні витрати модулів по RTL-схема з пакета XILINX

№	Тип автомату	Модуль Xilinx	Модуль Active-HDL	Число тригерів	Витрати по Квайну комбінаційної части	Загальні витрати по Квайну
1	Мура	el6	FSM	3	59	98
2	Мура	el1	FSM_MX	3	119	158
3	Мура	el4	FSM_MX_unit	5	178	243
4	Мура	el2	FSM_Sh	3	89	128
5	Міли	el5	FSM_Mealy	3	93	132
6	Міли	el3	FSM_Mealy_Sh	3	128	167
7	Міли	el10	GAS_FSM	4	157	209
8	ПД	el11	GAS_Diagnostic_Device	23	96	395

Витрати по Квайну визначаються як сумарне число входів всіх вентилів у схемі, оскільки число входів вентиля пропорційно числу транзисторів в ньому [12].

Складність схеми оцінюється кількістю обладнання, що становить схему. При розробці схем на основі конкретної елементної бази (МІС - схеми) кількість устаткування зазвичай вимірюється числом корпусів (модулів) інтегральних мікросхем, використовуваних у схемі. У теоретичних розробках орієнтуються на довільну елементну базу і тому для оцінки витрат обладнання використовується оцінка складності схем по Квайну.

Складність (ціна) за Квайном визначається сумарним числом входів логічних елементів в складі схеми. При такій оцінці одиниця складності − один вхід логічного елемента. Ціна інверсного входу зазвичай приймається рівною двом. Такий підхід до оцінки складності виправданий з наступних причин:

− складність схеми легко обчислюється по булевим функціям, на основі яких будується схема: для ДНФ складність схеми дорівнює сумі кількості букв, (букві зі знаком заперечення відповідає ціна, і кількості знаків диз'юнкції, збільшеного на 1 для кожного диз'юнктивного вираження;

− всі класичні методи мінімізації булевих функцій забезпечують мінімальність схеми саме в сенсі ціни по Квайну.

Практика показує, що схема з мінімальною ціною по Квайну зазвичай реалізується найменшим числом конструктивних елементів − корпусів інтегральних мікросхем МІС. Для схем БІС і НВІС апаратурні витрати усередині кристалів немає сенсу оцінювати числом корпусів, тому їх оцінюють за кількістю входів вентилів.

Витрати по Квайну на реалізацію динамічно керованого D тригера (табл. 2.1) були умовно визначені як N(D1) = 13, виходячи зі схеми класичного динамічного тригера, представленого на рис. 2.1. Апаратурні витрати коммбінаціонної частини оцінювалися по Квайну безпосередньо по RTL-схемі після синтезу.

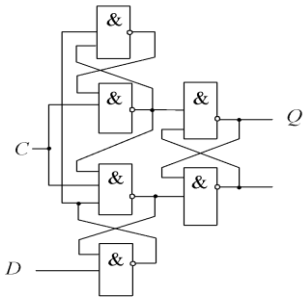


Рисунок 2.1 − Внутрішня структура динамічно керованого D тригера

Точно визначити витрати по Квайну на реалізацію динамічно керованого D тригера FPGA XC 3 S 500 E, Package FG 320, не представляється можливим через відсутність в інструкції по сімейству Spartan 3E (Data Sheet) внутрішньої структур і тригера.

З точки зору реалізації схеми в ПЛІС і заповнення її ресурсів інформація про структуру тригера абсолютно не потрібна, але, якщо оцінювати апаратурні витрати досліджуваних схем, залучаючи кошти XILINX для автоматизації рішення дослідницьких завдань, інформація про витрати на послідовних частинах може бути істотна, тоді можна йти двома шляхами.

Перший шлях: взяти умовне значення витрат по Квайну на тригер, наприклад, як вище згадувалося, рівним 13 (мінімальне значення), або по максимуму для реалізації всіх можливих функцій, визначених виробником ПЛІС (див. табл. 2.3) або щось середнє (рис. 2.2, 2.3 і табл. 2.2).

Так по опису функціонування тригера з слайса FPGA (рис. 2.2 і табл. 2.2) можна за основу для оціночних розрахунків умовно прийняти структуру тригера, представленого на рис. 2.3. Витрати по Квайну для такого тригера дорівнюватимуть N(D2) = 21.

Тригер в FPGA насправді складніший, ніж на рис. 2.1, в залежності від функцій, описаних у вихідному VHDL-коді (скидання, установка синхронні або асинхронні, дозволу синхронізації, тригер статично керований або динамічно), синтезується з різним набором входів (рис. 2.2) і видається цілісним об'єктом.

Другий шлях: після синтезу робити оцінку по набору входів тригерів, синтезованих системою (як, наприклад, на рис. 2.4). Цей варіант видається більш затратнім за часом, але, можливо, є більш точним.

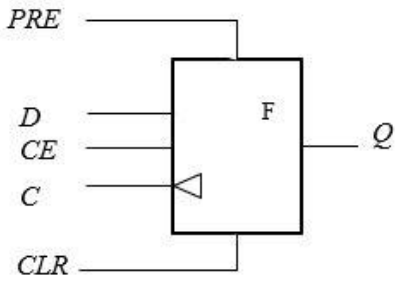


Рисунок 2.2 – Динамічно керований D-тригер FPGA XC 3 S 500 E

Таблиця 2.2 − Опис сигналів D-тригер а FPGA XC 3 S 500 E

Сигнал	Опис сигналу
D	Інформаційний вхід
Q	Інформаційний вихід
S, R (или CLR, PRE)	Синхронні входи установки, скидання(або асинхронне скидання, установка)
C	Синхровхід
CE	Вхід дозволу синхронізації

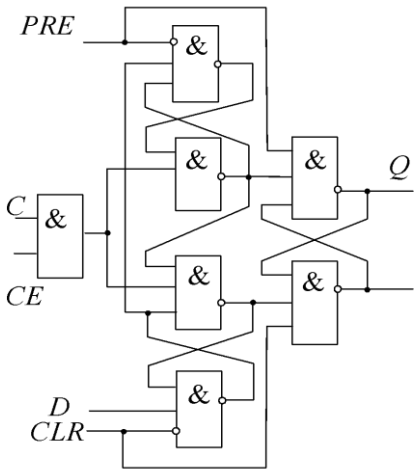


Рисунок 2.3 − Внутрішня структура динамічно керованого D тригера з асинхронною установкою, скиданням і дозволом синхронізації

У синтезованих схемах тригери можуть мати інші імена входів, що визначаються розробником FPGA XC 3 S 500 E, детально описані в інструкції [14].

Залежно від функцій тригера можуть бути реалізовані входи синхронного скидання / установки R / S, входи асинхронного скидання/установки (CLR / PRE) і т.д. (рис. 2.4, табл. 2.3).

Таблиця 2.3 − Опис сигналів D-тригера FPGA XC 3 S 500 E з інструкції

Сигнал	Опис
D	Вхідні дані. Для фліп-флоп даних вхід D завантажується, коли R і S (або CLR і PRE) низькі, а CE - високий під час переходу з низького на високий. Для засувки Q відображає вхід D, тоді як вхід затвору (G) і включення затвору (GE) високі, а R і S (або CLR і PRE) низькі. Дані про вхід D під час переходу між воротами "Висока на низьку" зберігаються у засувці. Дані про вихід Q засувки залишаються незмінними до тих пір, поки G або GE залишаються низькими.
Q	Вихідні дані. Перемикається після переходу Clock від низького до високого для фліп-флопів та негайно для засувки.
C	Clock для фліп-флопів.
G	Ворота для рівне-чутливих засувок.
CE	Clock увімкнено для тригерів.
GE	Ворота увімкнено для фліп-флопів .
S	Синхронний Set (Q = високий). Коли вхід S високий, а R низький, під час переходу з низького на високий тактовий (C) перемикач встановлюється, виводиться "Високий". Одразу встановлюється засувка, вихід Високий.
R	Синхронний Reset (Q = низький); має перевагу над Set.
PRE	Асинхронний Preset (Q = високий). Якщо вхід PRE - високий, а CLR - низький, під час переходу з низького на високий

	тактовий (C) перемикач встановлюється, виводиться "високий". Одразу встановлюється засувка, вихід високий.
CLR	Асинхронний Clear (Q = низький); має перевагу перед Preset для скидання вихідного рівня Q
SR	Вхід CLB для R, S, CLR або PRE
REV	Вхід CLB для протилежного SR. Повинно бути асинхронним або синхронним, щоб відповідати SR.

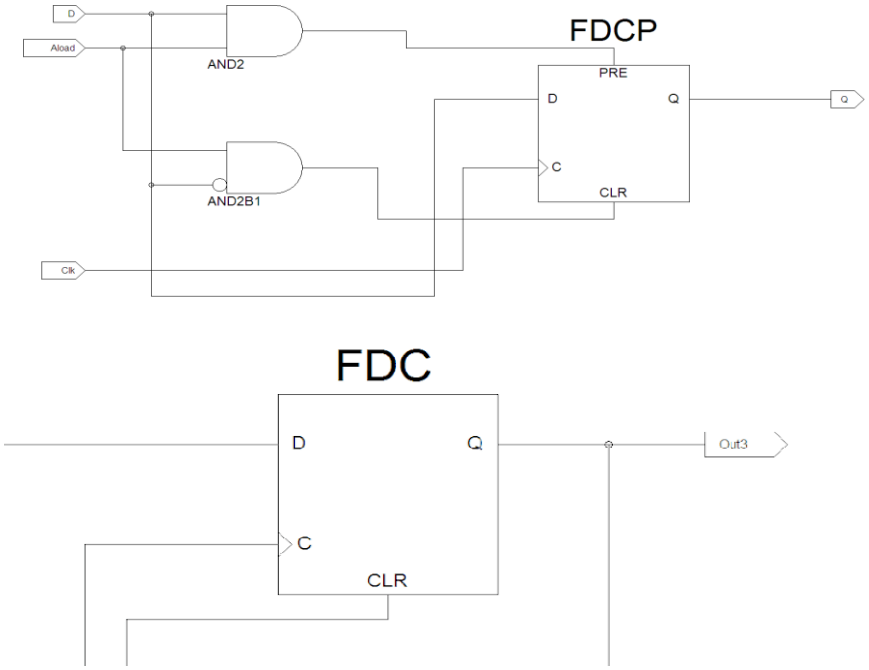


Рисунок 2.4 − Приклади синтезованих тригерів RTL-рівня в системі XILINX ISE

2.2 Оцінка витрат апаратури за звітами після синтезу в системі XILINX ISE

Основною структурною одиницею в FPGA сімейства Spartan 3E є осередок − SLICE, яка включає в себе 2 LUT для реалізації логічної функції (комбінаційна схема), 2 елементи пам'яті, а також додаткова логіка (рис. 2.5).

Використовуються два типи SLICE − SLICEM і SLICEL, обидві ці осередки мають такі елементи для реалізації логічних, арифметичних функцій і ПЗУ (ROM) функцій:

- два 4-входові LUT (F і G);
- два запам'ятовуючих елемента (Register);
- два мультиплексора широкого призначення (F 5 MUX і FiMUX);
- перенесення (Carry) і арифметичну логіку (Arithmetic Logic).

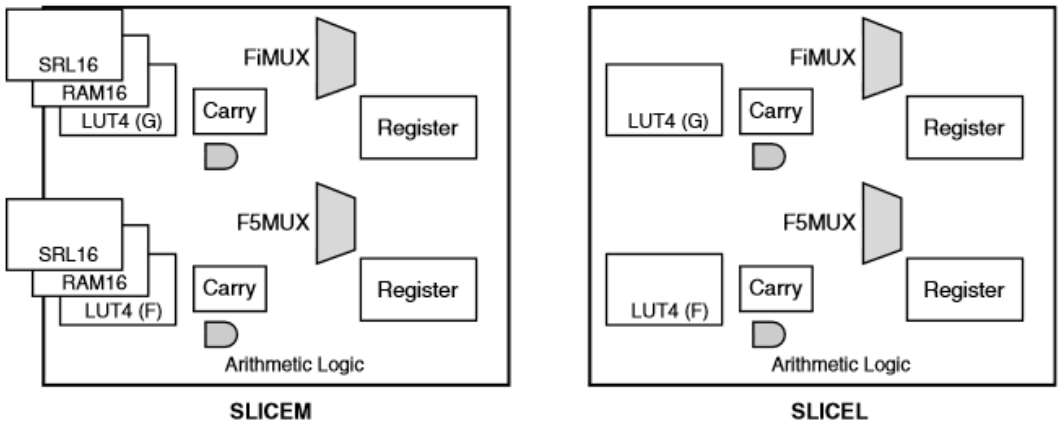


Рисунок 2.5 − Ресурси в Slice

SLICEM також підтримує дві додаткові функції:

- два 16 х 1 розподілених блоків ОЗУ (RAM 16);
- два 16- розрядних зсувних регістра (SRL 16).

Для аналізу буде використовуватися частина ресурсів SLICE: LUT і тригери (рис. 2.6).

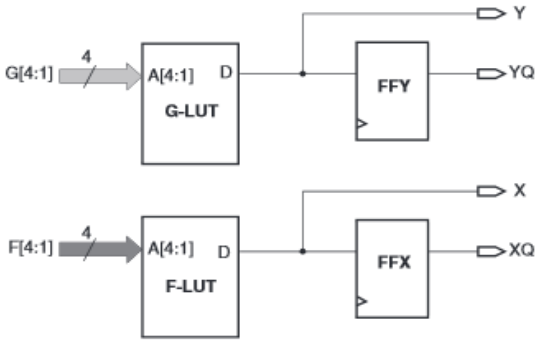


Рисунок 2.6 − LUT і тригери в Slice

LUT має пірамідальну структуру мультиплексорів з 2 в 1, для реалізації комбінаційних схем логічних функцій (рис. 2.7). Кожен рівень піраміди містить число мультиплексорів, кратне ступеня двійки.

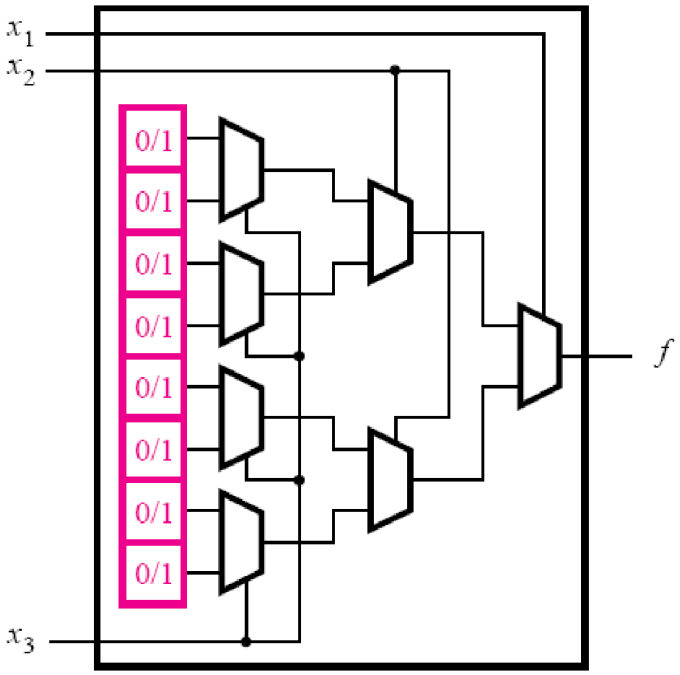


Рисунок 2.7 – Пірамідальна структура мультиплексорів для реалізації функції трьох змінних

LUT на 4 входи містить 15 таких мультиплексорів, кожен з яких має апаратурні витрати по Квайну, рівні 7, відповідно до внутрішньої структури мультиплексора, представленого на рис. 2.8.

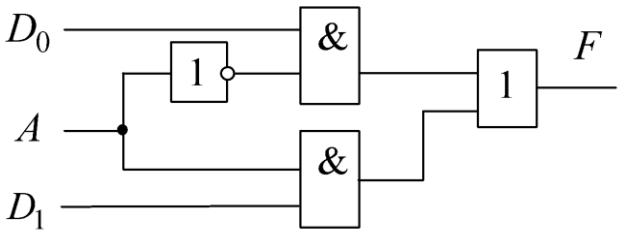


Рисунок 2.8 – Внутрішня структура мультиплексора з 2 в 1

Загальні витрати на один LUT по Квайну будуть визначатися як N (LUT) = 15 * 7 = 105.

Поза цією системою XILINX ISE, де можна в одному вікні побачити повний детальний звіт після синтезу, в проекті на диску можна відкрити два файли:

1) * .syr, де наведено докладний звіт, який повністю співпадає зі звітом в системі XILINX ISE .

2) * summary.html, де наведено короткий підсумковий звіт, який є частиною вмісту файлу * .syr.

* Означає ім'я VHDL-модуля.

У таблицях 2.4 – 2.9 наведені звіти по модулях.

Таблиця 2.4 – Звіт XILINX по модулю el6

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	4	4656	0%
Кількість Slice Flip Flops	3	9312	0%
Кількість 4 входових LUTs	7	9312	0%
Кількість зв’язаних IOBs	9	232	3%
Кількість GCLKs	1	24	4%

Таблиця 2.5 – Звіт XILINX по модулю el1

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	8	4656	0%
Кількість Slice Flip Flops	3	9312	0%
Кількість 4 входових LUTs	15	9312	0%
Кількість зв’язаних IOBs	11	232	4%
Кількість GCLKs	1	24	4%

Таблица 2.6 – Звіт XILINX по модулю el4

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	12	4656	0%
Кількість Slice Flip Flops	5	9312	0%
Кількість 4 входових LUTs	22	9312	0%
Кількість зв’язаних IOBs	10	232	4%
Кількість GCLKs	1	24	4%

Таблица 2.7 – Звіт XILINX по модулю el2

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	4	4656	0%
Кількість Slice Flip Flops	3	9312	0%
Кількість 4 входових LUTs	8	9312	0%
Кількість зв’язаних IOBs	10	232	4%
Кількість GCLKs	1	24	4%

Таблица 2.8 – Звіт XILINX по модулю el5

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	5	4656	0%
Кількість Slice Flip Flops	3	9312	0%
Кількість 4 входових LUTs	9	9312	0%
Кількість зв’язаних IOBs	10	232	4%
Кількість GCLKs	1	24	4%

Таблица 2.9 – Звіт XILINX по модулю el3

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Доступно	Вживання
Кількість Slices	6	4656	0%

Кількість Slice Flip Flops	3	9312	0%
Кількість 4 входових LUTs	11	9312	0%
Кількість зв’язаних IOBs	11	232	4%
Кількість GCLKs	1	24	4%

У табл. 2.10, 2.11 представлені апаратурні витрати модулів по RTL-схемі з пакета XILINX ISE 10.1.0, суміщені з даними зі звіту для оцінок з табл. 2.1, а також витрати на технологічному рівні. Число тригерів на RTL-схемі, на технологічному рівні і в звіті одне й теж.

Витрати зі звіту по Квайну визначається твором числа LUT і 105. Витрати по триггерам визначаються добутком кількості тригерів і 13 в табл. 2.10 і 21 в табл. 2.11.

Таблиця 2.10 − Апаратурні витрати модулів по RTL-схемі з пакета XILINX і звіту XILINX з урахуванням N (D1) = 13 (xc3s500e-4fg320)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	Число LUT зі звіту в шуках (По Квайну)	Загальні витрати зі звіту за Квайну	Витрати по Квайну комбінаційної частини по RTL - Схема	Загальні витрати по Квайну по RTL - Схема	Витрати по Квайну комбінаційної частини за схемою тенх. рівня	Загальні витрати по Квайну за схемою тенх. рівня
1	el6	3 (39)	7 (735)	774	60	99	55	94
2	el1	3 (39)	15 (1575)	1614	118	157	133	172
3	el4	5 (65)	22 (2310)	2375	204	269	186	241
4	el2	3 (39)	8 (840)	879	94	133	72	111
5	el5	3 (39)	9 (945)	984	84	123	93	132
6	el3	3 (39)	11 (1155)	1194	133	172	112	151
7	el10	4(52)	21 (2205)	2257	176	228	154	206

8	el11	23(299)	9 (945)	1244	151	450	56	355

Таблиця 2.11 − Апаратурні витрати модулів по RTL-схеми з пакета XILINX і звіту XILINX з урахуванням N (D2) = 21 (xc3s500e-4fg320)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	Число LUT зі звіту в шуках (По Квайну)	Загальні витрати зі звіту за Квайну	Витрати по Квайну комбінаційної частини по RTL-схемі	Загальні витрати по Квайну по RTL-схемі	Витрати по Квайну комбінаційної частини за схемою тенх. рівня	Загальні витрати по Квайну по схемі техн. рівня
1	el6	3 (63)	7 (735)	798	60	123	55	118
2	el1	3 (63)	15 (1575)	1638	118	181	133	196
3	el4	5 (105)	22 (2310)	2415	204	309	186	291
4	el2	3 (63)	8 (840)	903	94	157	72	135
5	el5	3 (63)	9 (945)	1008	84	147	93	156
6	el3	3 (63)	11 (1218)	1194	133	196	112	175
7	el10	4 (84)	21 (2205)	2257	176	260	154	238
8	el11	23(483)	9 (945)	1244	151	634	56	539

З таблиць 2.10 і 2.11 видно, що загальні витрати по Квайну відрізняються. Причому відмінність між RTL і технологічним рівнем незначні, це пов'язано з тим, що на RTL рівні вентильна схема реалізована на багатовхідних вентилях, а на технологічному рівні вентильна схема реалізована на двовходових вентилях, що збільшує витрати, але незначно.

Коли ж оцінка проводиться за кількістю тригерів і LUT зі звіту, враховуються всі вентилі LUT, що не завжди відповідає дійсності. Якщо подивитися на схему на технологічному рівні, можна виявити безліч LUT, у

яких, скажімо, з 4 входів задіяно 2 або 3 і, отже, вентильний еквівалент на технологічному рівні буде менше за витратами, ніж весь LUT .

У таблиці 2.12 наведені впорядковані по зростанню витрати апаратури послідовності номерів модулів для оцінок апаратурних витрат по числу LUT зі звіту, по RTL-схемі і за схемою технологічного рівня. Оцінки по Квайну для тригерів N (D1) = 13 і N (D2) = 21 на послідовності в даному випадку не вплинули.

Таблиця 2.12 − впорядковані по зростанню загальних витрат апаратури послідовності номерів модулів для xc3s500e-4fg320

Номера модулей по зростанню загальних витрат								LUT		RTL		Техн. уровень	
								N(D1)	N(D2)	N(D1)	N(D2)	N(D1)	N(D2)
1	4	5	6	8	2	7	3	1	1				
1	5	4	2	6	7	3	8			1	1		
1	4	5	6	2	7	3	8					1	1

З таблиці видно, що між оцінками по RTL і за технологічним рівнем відмінностей менше (і вони спостерігаються для близьких значень витрат апаратури), ніж між цими оцінками і оцінками по LUT зі звіту .

2.3 Оцінка витрат апаратури модулів по RTL-схемі, плата Virtex 5

Оцінка витрат апаратури модулів проводиться по RTL-схемі з пакета XILINX ISE 10.1.01, плата Virtex 5, мікросхема FPGA XC5VLX30, Package FF324 (xc5vlx30-3ff324).

Основною структурною одиницею в FPGA сімейства Virtex 5 є осередок − SLICE, який включає в себе 4 6-входових LUT для реалізації логічної функції (комбінаційної схеми), 4 елемента пам'яті, а також додаткову логіку. Більш детально в інструкції [15].

LUT на 6 вході містить 63 мультиплексорів, кожен з яких має апаратурні витрати по Квайну, рівні 7. Загальні витрати на один LUT по Квайну будуть визначатися як N (LUT) = 63 * 7 = 441.

Розглядаємо також дві оцінки для тригерів N (D1) = 13 і N (D2) = 21.

Таблиця 2.13 − Звіт XILINX по модулю el 6

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	3	19200	0%
Кількість Slice LUTs	6	19200	0%
Кількість повністю використовуваних LUT-FF пар	0	9	0%
Кількість зв’язаних IOBs	9	220	4%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.14 – Звіт XILINX по модулю el1

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	3	19200	0%
Кількість Slice LUTs	12	19200	0%
Кількість повністю використовуваних LUT-FF пар	3	12	25%
Кількість зв’язаних IOBs	11	220	5%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.15 – Звіт XILINX по модулю el4

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано

Кількість зареєстрованих Slice	5	19200	0%
Кількість Slice LUTs	14	19200	0%
Кількість повністю використовуваних LUT-FF пар	5	14	35%
Кількість зв’язаних IOBs	10	220	4%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.16 – Звіт XILINX по модулю el2

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	3	19200	0%
Кількість Slice LUTs	7	19200	0%
Кількість повністю використовуваних LUT-FF пар	0	10	0%
Кількість зв’язаних IOBs	10	220	4%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.17 – Звіт XILINX по модулю el5

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	3	19200	0%
Кількість Slice LUTs	7	19200	0%
Кількість повністю використовуваних LUT-FF пар	0	10	0%
Кількість зв’язаних IOBs	10	220	4%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.18 – Звіт XILINX по модулю el3

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	3	19200	0%
Кількість Slice LUTs	8	19200	0%
Кількість повністю використовуваних LUT-FF пар	0	11	0%
Кількість зв’язаних IOBs	11	220	5%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.19 – Звіт XILINX по модулю el10

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	4	19200	0%
Кількість Slice LUTs	17	19200	0%
Кількість повністю використовуваних LUT-FF пар	4	17	23%
Кількість зв’язаних IOBs	24	220	10%
Кількість BUFG/BUFGCTRLs	1	32	3%

Таблица 2.20 – Звіт XILINX по модулю el11

Підсумок використання пристрою (орієнтовні значення)			[-]
Використана логіка	Використано	Використана логіка	Використано
Кількість зареєстрованих Slice	23	19200	0%
Кількість Slice LUTs	13	19200	0%
Кількість повністю використовуваних LUT-FF пар	10	26	38%
Кількість зв’язаних IOBs	10	220	4%

Кількість BUFG/BUFGCTRLs	1	32	3%

У табл. 2.21, 2.22 представлені апаратурні витрати модулів по RTL-схемі з пакета XILINX ISE 10.1.01, суміщені з даними зі звіту для оцінок з табл. 2.1, а також витрати на технологічному рівні. Число тригерів на RTL-схемі, на технологічному рівні і в звіті один і той же.

Витрати зі звіту по Квайну визначається твором числа LUT і N (LUT) = 441. Витрати по триггерам визначаються твором кількості тригерів і 13 в табл. 2.21 і 21 в табл. 2.22.

Таблиця 2.21 − Апаратурні витрати модулів по RTL-схемі з пакета XILINX і звіту XILINX з урахуванням N (D1) = 13 (xc5vlx30-3ff324)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	Число LUT зі звіту в шуках (По Квайну)	Загальні витрати зі звіту за Квайном	Витрати по Квайну комбінаційної частини по RTL-схемі	Загальні витрати по Квайну по RTL-схемі	Витрати по Квайну комбінаційної частини за схемою тенх. рівня	Загальні витрати по Квайну за схемою техн. рівня
1	el6	3 (39)	6 (2646)	2685	73	112	52	87
2	el1	3 (39)	12 (5292)	5331	118	157	170	209
3	el4	5 (65)	14 (6174)	6239	204	269	220	285
4	el2	3 (39)	7 (3087)	3126	94	133	91	130
5	el5	3 (39)	7 (3087)	3126	92	131	76	115
6	el3	3 (39)	8 (3528)	3567	133	172	144	183
7	el10	4 (52)	17 (7497)	7549	176	228	193	245
8	el11	23(299)	13 (5733)	6032	151	450	205	504

Таблиця 2.22 − Апаратурні витрати модулів по RTL-схемі з пакета XILINX і звіту XILINX з урахуванням N (D2) = 21 (xc5vlx30-3ff324)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	Число LUT зі звіту в шуках (По Квайну)	Загальні витрати зі звіту за Квайном	Витрати по Квайну комбінаційної частини по RTL-схемі	Загальні витрати по Квайну по RTL-схемі	Витрати по Квайну комбінаційної частини за схемою тенх. рівня	Загальні витрати по Квайну за схемою техн. рівня
1	el6	3 (63)	6 (2646)	2709	73	136	52	115
2	el1	3 (63)	12 (5292)	5355	118	181	170	233
3	el4	5 (105)	14 (6174)	6279	204	309	220	325
4	el2	3 (63)	7 (3087)	3150	94	157	91	154
5	el5	3 (63)	7 (3087)	3150	92	155	76	139
6	el3	3 (63)	8 (3528)	3591	133	196	144	207
7	el10	4 (84)	17 (7497)	7581	176	260	193	277
8	el11	23(483)	13 (5733)	6216	151	634	205	688

Таблиця 2.23 − Впорядковані по зростанню загальних витрат апаратури послідовності номерів модулів для xc5vlx30-3ff324

Номера модулів по зростанню загальних витрат								LUT		RTL		Техн. уровень	
								N(D1)	N(D2)	N(D1)	N(D2)	N(D1)	N(D2)
1	4=5	4=5	6	2	8	3	7	1	1				
1	5	4	2	6	7	3	8			1	1		
1	5	4	6	2	7	3	8					1	1

4 = 5 позначає, що у 4 і 5 однакові значення витрат.

2.4 Оцінка витрат апаратури по «частинам» LUT

Якщо подивитися на схеми на технологічному рівні і на повний звіт після синтезу, можна відзначити, що в обох випадках використовуються не всі входи LUT, як у фрагменті звіту на рис. 2.9.

```
=========================================================
* Final Report *
=========================================================
Final Results
RTL Top Level Output File Name: FSM.ngr
Top Level Output File Name: FSM
Output Format: NGC
Optimization Goal: Speed
Keep Hierarchy: NO

Design Statistics
# IOs: 9

Cell Usage:
# BELS: 7
# LUT2: 3
# LUT3_L: 1
# LUT4: 3
# FlipFlops / Latches: 3
# FDC: 3
# Clock Buffers: 1
# BUFGP: 1
# IO Buffers: 8
#      IBUF                          : 4
#      OBUF                          : 4
================================================ ===========
```

Рисунок 2.9 – Фрагмент звіту по модулю el6 (xc3s500e-4fg320)

І якщо враховувати LUT з двома використовуваними входами не як чотиривходових LUT, а як двухвходових LUT, очевидно, що оцінка повинна бути точнішою, і можливо більш корелюватися з оцінками на RTL і на технологічному рівнях.

Потрібно це перевірити.

Спосіб оцінювання 1. «Частини» LUT будемо оцінювати за кількістю використовуваних входів LUT. Залежність апаратурних витрат по Квайну для LUT з різним числом входів представлена в табл. 2.24.

Таблиця 2.24 − Залежність витрат апаратури по Квайну для LUT з різним числом входів

Число входів LUT	6	5	4	3	2	1
Оцінка по Квайну	441	217	105	49	21	7
Число мультиплексорів в LUT	63	31	15	7	3	1

Спосіб оцінювання 2. «Частини» LUT будемо оцінювати за кількістю використовуваних входів LUT, але не по таблиці, а за формулою N (LUT) * N вх / N вх $_{LUT}$ * H $_{LUT}$, де N (LUT) − оцінка по Квайну для LUT, N вх − число використаних входів в LUT, N вх $_{LUT}$ − максимальне число входів LUT, H $_{LUT}$ − кількість таких LUT.

У табл. 2.25 і 2.26 представлені оцінки витрат апаратури по Квайну по «частинам» LUT. Витрати по тригерам визначаються добутком кількості тригерів і 13 в табл. 2.25 і 21 в табл. 2.26.

Таблиця 2.25 − Апаратурні по Квайну по «частинах» LUT зі звіту XILINX з урахуванням N (D1) = 13 (xc3s500e-4fg320)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	«Частини» LUT зі звіту	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT	Витрати комбінаційної частини по Квайнах по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT
				Спосіб оцінювання 1		Спосіб оцінювання 2	
1	el6	3 (39)	LUT2 : 3 LUT3_L : 1 LUT4 : 3	427	466	552	591
2	el1	3 (39)	LUT2 : 2 LUT3 : 4 LUT4 : 7 LUT4_L : 2	1183	1222	1365	1404
3	el4	5 (65)	LUT2 : 3 LUT3 : 6 LUT3_D : 1 LUT4 : 7 LUT4_L : 5	1666	1731	1968	2033
4	el2	3 (39)	LUT2 : 4	504	543	814	853

			LUT4 : 4				
5	el5	3 (39)	LUT3 : 2 LUT3_L : 1 LUT4 : 6	777	816	867	906
6	el3	3 (39)	LUT2 : 3 LUT4 : 8	903	942	997	1036
7	el10	4 (52)	LUT2 : 1 LUT3 : 4 LUT3_L : 2 LUT4 : 14	1785	1837	1996	2048
8	el11	23(299)	LUT3 : 5 LUT4 : 4	665	964	814	1113

Таблиця 2.26 − Апаратурні по Квайну по «частинах» LUT зі звіту XILINX з урахуванням N (D2) = 21 (xc3s500e-4fg320)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	«Частини» LUT зі звіту	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT	Витрати комбінаційної частини по Квайнах по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT
				Спосіб оцінювання1		Спосіб оцінювання2	
1	el6	3 (63)	LUT2 : 3 LUT3_L : 1 LUT4 : 3	427	490	552	585
2	el1	3 (63)	LUT2 : 2 LUT3 : 4 LUT4 : 7 LUT4_L : 2	1183	1246	1365	1428
3	el4	5 (105)	LUT2 : 3 LUT3 : 6 LUT3_D : 1 LUT4 : 7 LUT4_L : 5	1666	1771	1968	2073
4	el2	3 (63)	LUT2 : 4 LUT4 : 4	504	567	814	877
5	el5	3 (63)	LUT3 : 2 LUT3_L : 1 LUT4 : 6	777	840	867	930
6	el3	3 (63)	LUT2 : 3 LUT4 : 8	903	966	997	1060
7	el10	4 (84)	LUT2 : 1 LUT3 : 4 LUT3_L : 2 LUT4 : 14	1785	1869	1996	2080
8	el11	23(483)	LUT3 : 5 LUT4 : 4	665	1148	814	1297

У табл. 2.27 наведені впорядковані по зростанню витрат апаратури послідовності номерів модулів для оцінок загальних витрат апаратури по Квайну з додаванням оцінок по «частинах» LUT для (xc3s500e-4fg320) (розширена табл. 2.12).

Таблиця 2.27 − Впорядковані по зростанню загальних витрат апаратури послідовності номерів модулів для оцінок витрат апаратури для xc3s500e-4fg320

Номера модулів по зростанню загальних витрат								LUT		RTL		Техн. рівень		по «частинам» LUT спосіб 1		по «частинам» LUT спосіб 2	
								13	21	13	21	13	21	13	21	13	21
1	4	5	6	8	2	7	3	1	1								
1	5	4	2	6	7	3	8			1	1						
1	4	5	6	2	7	3	8					1	1				
1	4	5	6	8	2	3	7							1	1	1	1

Для Virtex 5 в табл. 2.28 і 2.29 представлені оцінки витрат апаратури по Квайну по «частинах» LUT. Витрати по тригерам визначаються добутком кількості тригерів і 13 в табл. 2.28 і 21 в табл. 2.29.

Таблиця 2.28 − Апаратурні по Квайну по «частинах» LUT зі звіту XILINX з урахуванням N (D1) = 13 (xc5vlx30-3ff324)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	«Частини» LUT зі звіту	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT
				Спосіб оцінювання 1		Спосіб оцінювання 2	
1	el6	3 (39)	LUT2 : 2 LUT3 : 2	574	613	1470	1509

			LUT5 : 2				
2	el1	3 (39)	LUT3 : 5 LUT5 : 4 LUT6 : 3	2436	2475	3896	3935
3	el4	5 (65)	LUT2 : 1 LUT4 : 2 LUT5 : 7 LUT6 : 4	3514	3579	5072	5137
4	el2	3 (39)	LUT2 : 4 LUT5 : 2 LUT6 : 1	959	998	1764	1803
5	el5	3 (39)	LUT2 : 1 LUT3 : 1 LUT4 : 2 LUT5 : 3	931	970	2059	2098
6	el3	3 (39)	LUT2 : 1 LUT4 : 1 LUT5 : 4 LUT6 : 2	1876	1915	3822	3861
7	el10	4 (52)	LUT3 : 2 LUT4 : 10 LUT5 : 3 LUT6 : 2	2681	2733	5366	5418
8	el11	23(299)	LUT4 : 2 LUT5 : 6 LUT6 : 5	3717	4016	4998	5297

Таблиця 2.29 − Апаратурні по Квайну по «частинах» LUT зі звіту XILINX з урахуванням N (D2) = 21 (xc5vlx30-3ff324)

№	Модуль Xilinx	Число тригерів зі звіту в шуках (По Квайну)	«Частини» LUT зі звіту	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT	Витрати комбінаційної частини по Квайну по «частинах» LUT	Загальні витрати по Квайну по «частинах» LUT
				Спосіб оцінювання 1		Спосіб оцінювання 2	
1	el6	3 (63)	LUT2 : 2 LUT3 : 2 LUT5 : 2	574	610	1470	1533
2	el1	3 (63)	LUT3 : 5 LUT5 : 4 LUT6 : 3	2436	2499	3896	3959
3	el4	5 (105)	LUT2 : 1 LUT4 : 2 LUT5 : 7	3514	3619	5072	5177

			LUT6 : 4				
4	el2	3 (63)	LUT2 : 4 LUT5 : 2 LUT6 : 1	959	1022	1764	1827
5	el5	3 (63)	LUT2 : 1 LUT3 : 1 LUT4 : 2 LUT5 : 3	931	994	2059	2122
6	el3	3 (63)	LUT2 : 1 LUT4 : 1 LUT5 : 4 LUT6 : 2	1876	1939	3822	3885
7	el10	4 (84)	LUT3 : 2 LUT4 : 10 LUT5 : 3 LUT6 : 2	2681	2765	5366	5450
8	el11	23(483)	LUT4 : 2 LUT5 : 6 LUT6 : 5	3717	4200	4998	5481

У табл. 2.30 наведені впорядковані по зростанню витрат апаратури послідовності номерів модулів для оцінок загальних витрат апаратури по Квайну з додаванням оцінок по «частинах» LUT для (xc5vlx30-3ff324) (розширена табл. 2.23).

Таблиця 2.30 − Впорядковані по зростанню загальних витрат апаратури послідовності номерів модулів для оцінок витрат апаратури для xc5vlx30-3ff324

Номера модулів по зростанню загальних витрат								LUT		RTL		Техн. рівень		По «частинам» LUT спосіб 1		По «частинам» LUT спосіб 2	
								13	21	13	21	13	21	13	21	13	21
1	4 = 5	4 = 5	6	2	8	3	7	1	1								
1	5	4	2	6	7	3	8			1	1						
1	5	4	6	2	7	3	8					1	1	1	1		
1	4	5	6	2	3	8	7									1	
1	4	5	6	2	3	7	8										1

У табл. 2.31 та 2.32 наведені впорядковані по зростанню витрат апаратури послідовності номерів модулів для оцінок апаратурних витрат по Квайну з додаванням оцінок по «частинах» LUT тільки для комбінаційних частин для (xc3s500e-4fg320) і для (xc5vlx30-3ff324) відповідно.

Таблиця 2.31 − Впорядковані по зростанню витрат апаратури послідовності номерів модулів (xc3s500e-4fg320)

Номера модулів по зростанню загальних витрат тільки для комбінаційних частин								LUT	RTL	Техн. рівень	По «частинам» LUT спосіб 1	По «частинам» LUT спосіб 2
1	4	5=8	5=8	6	2	7	3	1				
1	5	4	2	6	8	7	3		1			
1	8	4	5	6	2	7	3			1		
1	4	8	5	6	2	3	7				1	
1	4=8	4=8	5	6	2	3	7					1

Таблиця 2.32 − Впорядковані по зростанню загальних витрат апаратури послідовності номерів модулів для оцінок витрат апаратури для (xc5vlx30-3ff324)

Номера модулів по зростанню загальних витрат тільки для комбінаційних частин								LUT	RTL	Техн. рівень	По «частинам» LUT спосіб 1	По «частинам» LUT спосіб 2
1	4=5	4=5	6	2	8	3	7	1				
1	5	4	2	6	8	7	3		1			
1	5	4	6	2	7	8	3			1		
1	5	4	6	2	7	3	8				1	
1	4	5	6	2	8	3	7					1

З цих даних можна зрозуміти, що кореляції між оцінками витрат апаратури модулів по звіту (по LUT), по RTL-схемі і по технологічному рівні з пакета XILINX ISE 10.1.01 немає. Це стосується і загальних витрат апаратури, і витрат тільки для комбінаційної частини.

Тому оцінку витрат необхідно робити в прив'язці до конкретної плати і мікросхемі за звітом (по LUT). При цьому більш точною оцінкою буде оцінка по «частинам LUT» по способу 1.

Витрати на тригери можна оцінювати, як 13 або 21, на один тригер, особливої ролі це не грає.

Для різних мікросхем ПЛІС результати синтезу однієї і тієї ж схеми на RTL-рівні можуть іноді сильно відрізнятися. Також можуть бути відмінності і на технологічному рівні і за звітом.

3 ДОСЛІДЖЕННЯ ПОВЕДІНКИ ОДНОПРОЦЕСНОГО ТА ДВУПРОЦЕСНОГО ШАБЛОНУ ДО ІМПЛЕМЕНТАЦІЇ

3.1 Опис ПК за допомогою однопроцесного шаблону

У програмуванні при аналізі програм, що описують автоматні системи реального часу, широко застосовуються часові автомати. У реалізації часового автомата кожному стану ставиться у відповідність годинник, який визначає інтервали метричного (безперервного) часу. Переходи зі стану в стан в часовому автоматі відбуваються миттєво, але автомат не переходить в новий стан, поки годинник не покаже можливість переходу (закінчиться заданий інтервал метричного часу). Дана модель відповідає кінцевому автомату Мура.

Для внесення реального часу в опис структурного автомата пропонується використовувати розширену функцію переходів [18]

$$Z(t+1) = f(X(t), Z(t), T), \tag{3.1}$$

в якій аргументом виступає реальний час. При такому підході аргумент T відповідає показанням годин часового автомата. Граф переходів автомата, де враховується параметр часу, будемо називати темпоральним графом переходів (temporal state diagram). У цьому графі кожному стану ставиться у відповідність затримка Ti, яка визначається числом автоматних тактів, протягом яких автомат знаходиться в даному стані. Вихідні сигнали, відповідні даному стану автомата, протягом цього часу не змінюються (для автомата Мура). Проблема перерахунку метричного часу в автоматні такти є досить важливим завданням, але через обмеженість обсягу, в даній роботі детально не розглядається.

Як уже згадувалося вище, граф переходів кінцевого автомата є візуальним представленням моделі абстрактного автомата. При переході до

моделі структурного автомата, яка буде використовуватися в САПР, необхідно вказати наступні параметри: спосіб кодування станів, розрядність вхідних і вихідних змінних, вид синхронізації і, для часового автомата, затримку в кожному стані. Такий граф переходів, за аналогією з змістовної граф-схемою алгоритму, будемо називати змістовним графом переходів кінцевого автомата. Таким чином, змістовний темпоральний граф переходів є не тільки візуальним поданням алгоритму роботи структурного автомата, а й його повною математичною моделлю. Змістовний граф переходів в САПР описується шаблонами спеціального виду (patterns) на мовах програмування або описання апаратури.

Затримка в кожній вершині темпорального графа переходів реалізується через петлю, умовами для якої є підрахунок числа тактів Clk, що схемно реалізується лічильником в ПЛІС або таймером з перериванням в МК. На рис. 3.1 наведено відповідність умовного позначення затримок в вершинах темпорального графа переходів автомата Мура без петель і з петлями.

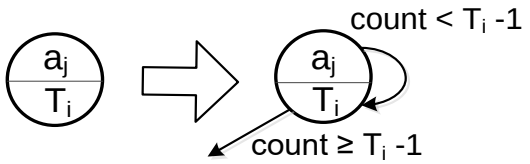


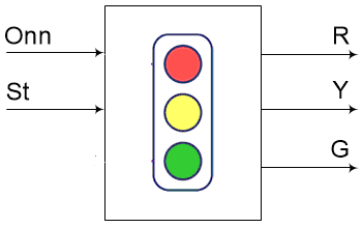
Рисунок 3.1 − Реалізація затримок в темпоральном графі переходів автомата Мура

Затримка Ti здійснюється багаторазовим переходом зі стану в цей же стан, при цьому число переходів визначається числом тактів затримки. Значення лічильника count порівнюється з Ti-1, оскільки при переході в стан aj, автомат один такт знаходиться в ньому до перевірки count, і щоб затримка була точно Ti тактів, необхідно ще Ti-1 тактів повторення.

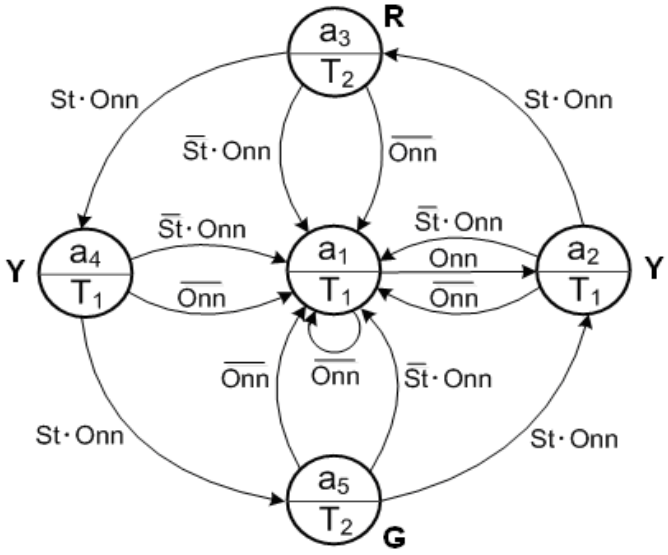
Як приклад пристрою логічного управління розглянемо пристрій управління дорожнім світлофором, який є частиною автоматизованої системи

управління дорожнім рухом. В якості керуючого автомата використовується модель Мура.

На рис. 3.2 (а) показаний інтерфейс даного керуючого автомата. Безліч вхідних сигналів X = { Onn , St }, де Onn = {0, 1} − сигнал включення світлофора, St = {0, 1} − сигнал запуску стандартного циклу роботи світлофора. Безліч вихідних сигналів Y = {R, Y, G}, де R = { Red } − сигнал включення червоного світла світлофора, Y = { Yellow } − сигнал включення жовтого світла світлофора, G = { Green } − сигнал включення зеленого світла світлофора.



а)



б)

Рисунок 3.2 − Інтерфейс і граф переходів автомата Мура для пристрою управління світлофором

Цей пристрій реалізує два циклу роботи світлофора:

− нічний (аварійний) цикл − миготіння жовтого світла світлофора з інтервалом (затримкою) T1;

− стандартний цикл роботи світлофора {Y - R - Y - G -Y}, при цьому затримка R, G буде T2 (5 тактів), а затримка Y буде T1 (один такт).

На рис. 3.2 (б) представлений темпоральний граф переходів даного пристрою управління. На розглянутому графі всі сигнали є однорозрядні, тому їх розрядність не вказується, спосіб кодування станів визначається в налаштуваннях САПР, синхронізація здійснюється по передньому фронту.

При автоматизованому проектуванні цифрових пристроїв на ПЛІС для опису алгоритму функціонування, як правило, використовуються мови опису апаратури (HDL).

Для опису VHDL-моделі розглянутого ПК був обраний однопроцесний та двопроцесний шаблон. Розглянемо однопроцесний шаблон.

Поза процесом p1 оголошуються константні значення затримок T1 і T2, що задаються в періодах Clk. Також оголошується сигнал count для реалізації лічильника періодів Clk. При активному сигналі скидання Reset = 1 автомат встановлюється в початковий стан a1, а лічильник періодів Clk обнуляється. При описі переходу зі стану ai з затримкою Ti з кожним новим тактом автомат перекладається в вихідний ai стан, і на цьому ж переході лічильник збільшується на одиницю. Це повинно відбуватися при роботі автомата до тих пір, поки значення лічильника count не стане рівним Ti-1. При переході зі стану ai в кожне наступне стан aj ≠ ai, лічильник повинен обнулятися, щоб при попаданні в наступний стан, в якому необхідний новий цикл рахунку, він був в початковому стані.

Якщо автомат потрапляє в несанкціонований стан, він повинен повернутися в початковий стан a1. Керуючі (вихідні) сигнали G, Y і R формуються поза процесом за допомогою паралельного оператора умовного призначення, що дозволяє отримати при синтезі комбінаційну функцію виходів і значно скорочує апаратурні витрати.

У додатку Б представлений лістинг VHDL-моделі, відповідної темпоральність графу переходів, представленому на рис. 3.2 (б).

Випробувальний стенд для світлофора для деяких режимів роботи для моделювання z в XILINX ISE 14.7 знаходиться у додатку В.

На рис. 3.3 приведена часова діаграма (waveform) результатів моделювання роботи розглянутого ПК в системі XILINX ISE 14.7.

Установка автомата в початковий стан здійснюється протягом першого автоматного такту, далі наведено різні варіанти розвитку подій. Видно, що вихідний керуючий сигнал формується відразу, як тільки автомат переходить в новий стан, відмінний від a1. Тривалість сигналу Y становить 2 такта, сигналів G і R − 5 тактів.

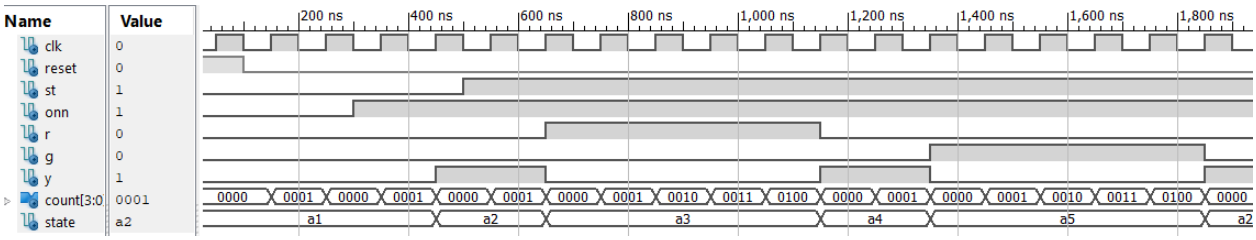


Рисунок 3.3 − Результати моделювання вихідної моделі пристрою управління світлофором

Для забезпечення необхідної тривалості автоматного такту слід використовувати стандартний дільник частоти (лічильник). В даному прикладі тривалість одного автоматного такту t буде дорівнювати одній хвилині (t = 60 с). Період синхросигналу з виходу генератора тактової частоти визначається як

$$Tclk = 1/Hclk \text{ (с)}, \qquad (3.2)$$

де Hclk − частота синхросигналу (Гц). Для того, щоб визначити у скільки разів необхідно знизити частоту (Hclk) або збільшити період (Tclk), необхідно знайти відношення

$$t/Tclk\,, \tag{3.3}$$

яке визначить значення коефіцієнта перерахунку лічильника, і розрахувати n з виразу

$$2^{n-1} < \frac{t}{Tclk} \le 2^{n}\,, \tag{3.4}$$

для визначення розрядності лічильника.

У таблиці 3.1 наведені результати синтезу даного ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою CPLD на платі AntBoard Xilinx XC9572XL -10-TQ100**.**

Таблиця 3.1 − Результати синтезу, сумарний обсяг використовуваних ресурсів

Використано макроячеек	Використано птермів	Використано регістрів	Використано пінів	Використано входів функціональних блоків
8/72 (12%)	28/360 (8%)	7/72 (10%)	7/72 (10%)	33/216 (16%)

Таблиця 3.2 − Результати синтезу, кінцеві часові характеристики

Підсумок роботи	
Мінімальний Clock період	11.000 ns.
Максимальна Clock частота	90.909 MHz.
Обмежений час циклу для Clk	
Clock до налаштування	11.000 ns.
Налаштування Clock до Pad	7.500 ns.
Clock Pad до вихіної Pad затримки	13.500 ns.

Нижче наведено фрагмент звіту після синтезу.

```
=========================================================================
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors                    : 1
 4-bit adder                           : 1
# Registers                             : 1
 4-bit register                        : 1
# Comparators                           : 2
 4-bit comparator less                 : 2

=========================================================================
*                  Advanced HDL Synthesis                  *
=========================================================================

Analyzing FSM <FSM_0> for best encoding.
Optimizing FSM <State/FSM> on signal <State[1:3]> with user encoding.
-------------------
 State | Encoding
-------------------
 a1    | 000
 a2    | 001
 a3    | 010
 a4    | 011
 a5    | 100
-------------------

=========================================================================
```

У додатку Ґ зображено результат синтезу вихідної моделі пристрою управління світлофором, схема світлофора для CPLD.

Далі представлені результати дослідження шаблону на пристрої FPGA на платі Spartan 3E, мікросхема XC3S500E -4fg320.

При синтезі опису, представленого в лістингу 3.1 система зробила попередження про розряд count _3, який завжди дорівнює нулю, тому в цьому лістингу довелося скорегувати наступні оголошення:

Лістинг 3.1 – Скореговані оголошення

```
signal count: std_logic_vector ( 2 downto 0); - periods CLK counter
constant T1: std_logic_vector ( 2 downto 0): = "010"; - delay = 2 periods CLK
constant T2: std_logic_vector ( 2 downto 0): = "101"; - delay = 5 periods CLK
```

У таблиці 3.3 наведені результати синтезу даного ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою FPGA.

Таблиця 3.3 − Результати синтезу, сумарний обсяг використовуваних ресурсів

Підсумок використання пристрою (орієнтовні значення)		
Використана логіка	Використано	Доступно
Кількість Slices	8	4656
Кількість Slice Flip Flops	6	9312
Кількість 4х входових LUTs	15	9312
Кількість зв’язаних IOBs	7	232
Кількість GCLKs	1	24

Сумарний звіт.

Timing Summary:

Speed Grade: -5

Minimum period: 4.131ns (Maximum Frequency: 242.072MHz)
Minimum input arrival time before clock: 4.674ns
Maximum output required time after clock: 5.501ns
Maximum combinational path delay: No path found

Нижче представлений фрагмент протоколу синтезу даного пристрою:

Analyzing FSM <FSM_0> for best encoding.
Optimizing FSM <State/FSM> on signal <State[1:3]> with user encoding.

State | Encoding
---
a1 | 000
a2 | 001
a3 | 010
a4 | 011
a5 | 100
==========
Advanced HDL Synthesis Report
Macro Statistics
FSMs : 1

```
# Adders/Subtractors                    : 1
 3-bit adder                           : 1
# Registers                             : 3
 Flip-Flops                            : 3
# Comparators                           : 2
 3-bit comparator less                 : 1
 4-bit comparator less                 : 1
```

У додатку Д представлені результати синтезу вихідної моделі пристрою управління світлофором, схема світлофора на FPGA.

3.2 Опис ПК за допомогою двопроцесного шаблону

Розглянемо тепер двопроцесний шаблон. У ньому два процеси як і в звичайному не часовому автоматі, які описують, власне, сам автомат (процес, що описує комбінаційну частину, і процес, що описує послідовну частину) і ще окремий процес, що описує лічильник. Таким чином, він фактично трьороцесним, але далі будемо називати його двопроцесним.

У додатку В представлено лістинг двопроцесного шаблону.

На рис. 3.4 наведена часова діаграма (waveform) результатів моделювання роботи розглянутого ПК в системі XILINX ISE 14.7.

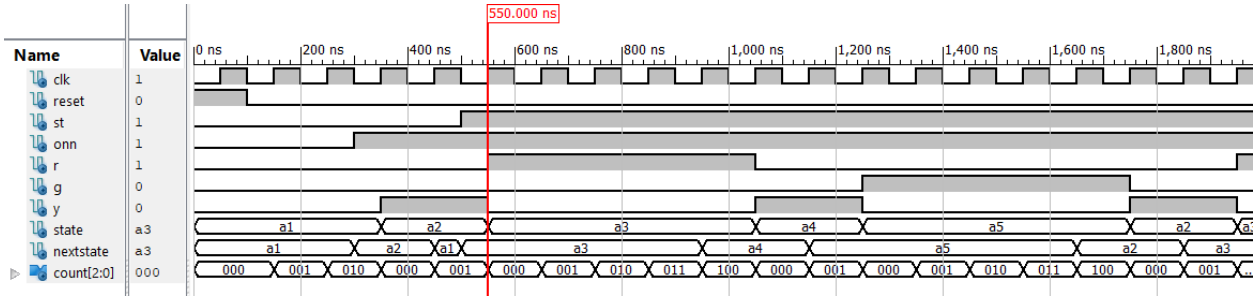


Рисунок 3.4 – Результати моделювання двопроцесної моделі пристрою управління світлофором

У таблиці 3.4 наведені результати синтезу даного ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою CPLD на платі AntBoard Xilinx XC9572XL -10-TQ100.

Таблиця 3.4 − Результати синтезу, сумарний обсяг використовуваних ресурсів

Використано макроячеек	Використано птермів	Використано регістрів	Використано пінів	Використано входів функціональних блоків
7/72 (10%)	31/360 (9%)	6/72 (9%)	7/72 (10%)	30/216 (14%)

Таблиця 3.5 − Результати синтезу, підсумкові часові характеристики

Підсумок роботи	
Мінімальний Clock період	11.000 ns.
Максимальна Clock частота	90.909 MHz.
Обмежений час циклу для Clk	
Clock до налаштування	11.000 ns.
Налаштування Clock до Pad	7.500 ns.
Clock Pad до вихіної Pad затримки	13.500 ns.

Нижче наведено фрагмент звіту після синтезу

```
HDL Synthesis Report
Macro Statistics
# Counters                              : 1
 3-bit up counter                      : 1
# Comparators                           : 3
 3-bit comparator less                 : 1
 3-bit comparator not equal            : 1
 4-bit comparator less                 : 1
=========================================================================
*                  Advanced HDL Synthesis                  *
=========================================================================
Analyzing FSM <FSM_0> for best encoding.
Optimizing FSM <State/FSM> on signal <State[1:3]> with user encoding.
-------------------
 State | Encoding
-------------------
 a1    | 000
 a2    | 001
```

```
a3   | 010
a4   | 011
a5   | 100
```

У додатку Е зображено результат синтезу вихідної моделі пристрою управління світлофором, схема світлофора для CPLD.

Далі представлені результати дослідження шаблону на пристрої FPGA на платі Spartan 3E, мікросхема XC3S500E -4fg320.

Таблиця 3.5 − Результати синтезу, сумарний обсяг використовуваних ресурсів

Підсумок використання пристрою (орієнтовні значення)		
Використана логіка	Використано	Доступно
Кількість Slices	9	4656
Кількість Slice Flip Flops	6	9312
Кількість 4х входових LUTs	18	9312
Кількість зв’язаних IOBs	7	232
Кількість GCLKs	1	24

Результати синтезу, підсумкові часові характеристики:

```
Timing Summary:
Speed Grade: -5

   Minimum period: 4.215ns (Maximum Frequency: 237.245MHz)
   Minimum input arrival time before clock: 4.417ns
   Maximum output required time after clock: 5.597ns
   Maximum combinational path delay: No path found
```

Нижче наведено фрагмент звіту після синтезу:

```
=========================================================================
HDL Synthesis Report
Macro Statistics
# Counters                             : 1
 3-bit up counter                      : 1
# Comparators                          : 3
 3-bit comparator less                 : 1
 4-bit comparator less                 : 1
```

```
 5-bit comparator not equal                    : 1
*                  Advanced HDL Synthesis                  *
Analyzing FSM <FSM_0> for best encoding.
Optimizing FSM <State/FSM> on signal <State[1:3]> with user encoding.
-------------------
 State | Encoding
a1    | 000
 a2    | 001
 a3    | 010
 a4    | 011
 a5    | 100
```

У додатку Є представлено результати синтезу вихідної моделі пристрою управління світлофором, схема світлофора на FPGA.

4 ДОСЛІДЖЕННЯ ПОВЕДІНКИ ШАБЛОНІВ ПІСЛЯ ІМПЛЕМЕНТАЦІЇ

4.1 Дослідження поведінки однопроцесного шаблону після імплементації

ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою CPLD на платі AntBoard Xilinx XC9572XL -10-TQ100**.**

У процесі синтезу САПР оцінила мінімальний період сигналу clk = 11 нс. Нижче наведена часова діаграма на цьому періоді clk = 12 нс (рис. 4.1).

З даного рисунка видно, що логіка роботи не спотворювана, але жовтий починає формуватися в перший раз на 5, 5 тактів пізніше і перехід від жовтого до червоного запізнюється (червона мітка) на 7.7 нс (64% від тривалості сигналу clk), при переході від червоного до жовтого затримується також на 7.7 нс. Затримка закінчення червоного від переднього фронту clk дорівнює 13.5 (112.5%). Зміна жовтого і зеленого відбувається без здвигів (синя мітка).

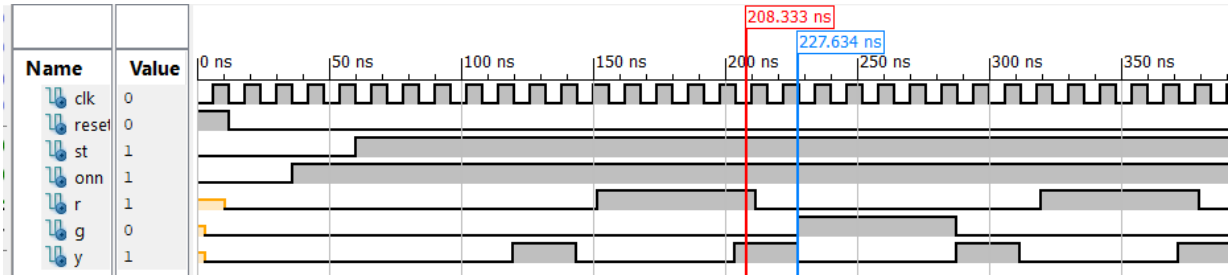


Рисунок 4.1 – Часова діаграма у періоді clk = 12 нс

Нижче наведена часова діаграма на періоді clk = 22 нс (рис. 4.2).

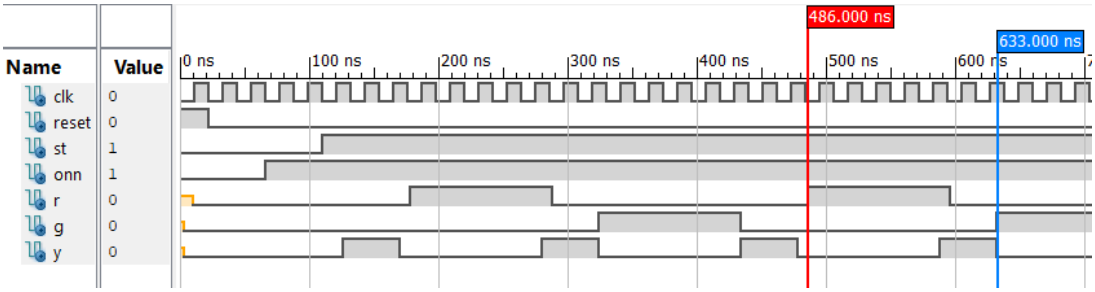


Рисунок 4.2 – Часова діаграма у періоді clk = 22 нс

З даного рисунка видно, що логіка роботи не спотворена, але жовтий починає формуватися в перший раз на такт пізніше і перехід від жовтого до червоного запізнюється (червона мітка) на 7.7 нс (на 35%), при переході від червоного до жовтого також затримується на 7.7 нс. Затримка закінчення червоного від переднього фронту clk дорівнює 13.5 (на 61.3%). Зміна жовтого і зеленого відбувається без здвигів (синя мітка).

Нижче наведена часова діаграма на періоді clk = 100 нс (рис. 4.3).

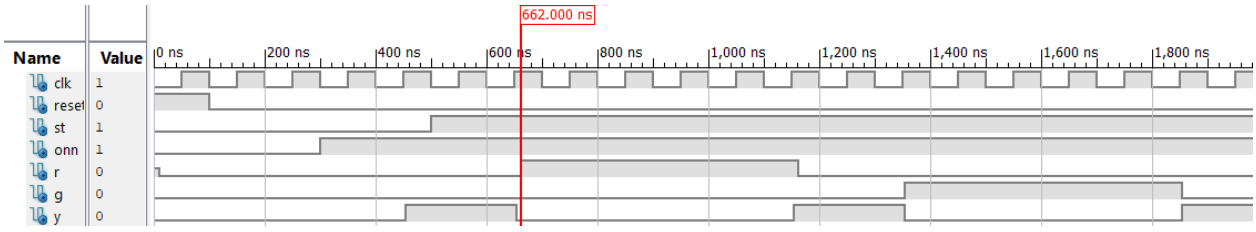


Рисунок 4.3 − Часова діаграма у періоді clk = 100 нс

З даного рисунка видно, що логіка роботи не спотворена, жовтий починає формуватися в перший раз без зсуву як і при моделювання вихідного опису і перехід від жовтого до червоного запізнюється на 6.7 нс (на 6.7%) (червона мітка), а при переході від червоного до жовтого затримується на 7.7 нс (на 7.7%). Затримка закінчення червоного від переднього фронту clk дорівнює 8.5 (на 8.5%). Зміна жовтого і зеленого відбувається без здвигу.

Таким чином, чим більше період clk, тим ближче результати моделювання після імплементації до результатів моделювання вихідного опису. До того ж якщо тривалість кольорів світлофора обчислюється хвилинами, частота роботи пристрою не суттєва, а значить вибираємо більший період для досягнення більшої точності в роботі пристрою.

ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою FPGA на платі Spartan 3 E Starter kit Xilinx XC 3 S 500 E -5 fg 320 Мінімальний період для FPGA clk = 4.131 нс.

Нижче наведена часова діаграма на цьому періоді clk = 12 нс, щоб було зручніше порівняти з CPLD (рис. 4.4).

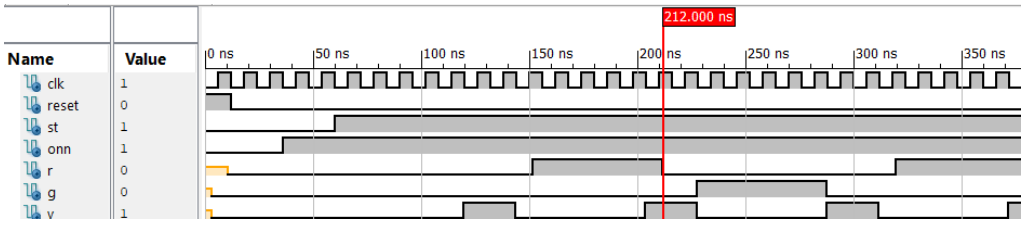


Рисунок 4.4 – Часова діаграма у періоді clk = 12 нс

Нижче наведена часова діаграма на цьому періоді clk = 12 нс (рис. 4.5).

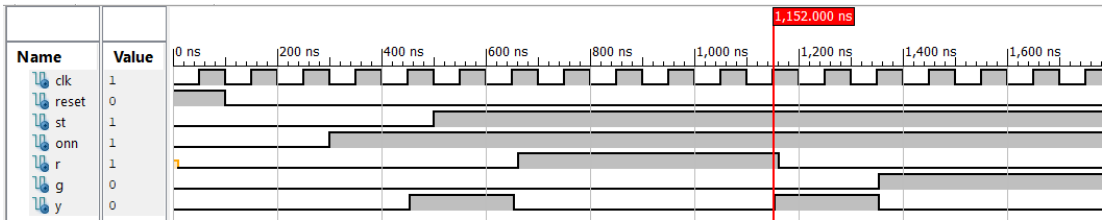


Рисунок 4.5 – Часова діаграма у періоді clk = 12 нс

Результати моделювання однопроцесного шаблону ідентичні результатам для CPLD.

4.2 Дослідження поведінки двопроцесного шаблону після імплементації

ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою CPLD на платі AntBoard Xilinx XC 9572 XL -10- TQ 100.

Мінімальний період синхросигналу для CPLD clk = 11 нс. На рис. 4.6 наведена часова діаграма на періоді clk = 12 нс.

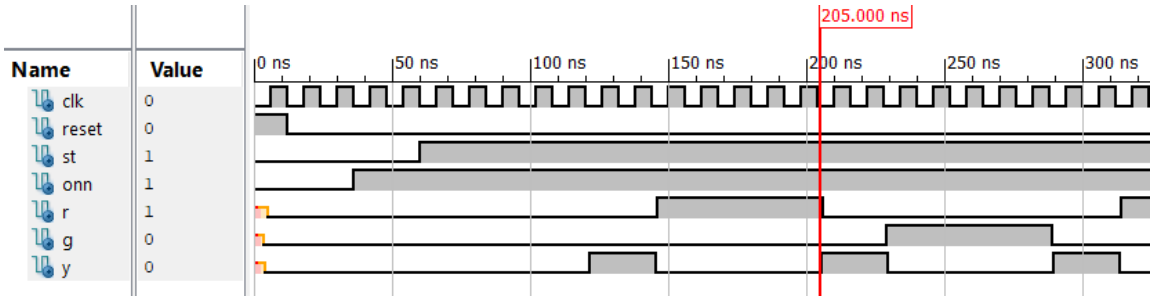


Рисунок 4.6 – Часова діаграма на періоді clk = 12 нс

З даного рисунка видно, що логіка роботи не спотворюється, але жовтий починає формуватися в перший раз на 5,5 тактів пізніше і перехід від жовтого до червоного запізнюється (червона мітка) на 0,796 нс (6,5 % від тривалості сигналу clk), при переході від червоного до жовтого затримується також на 0,796 нс. Зміна жовтого і зеленого (в обидві сторони) відбувається з невеликим зрушенням 0,47 нс (3,9%).

На рис. 4.7 наведена часова діаграма на періоді clk = 100 нс.

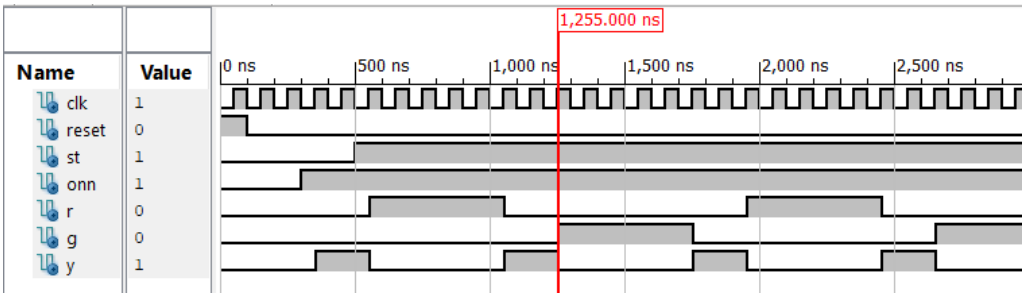


Рисунок 4.7 – Часова діаграма на періоді clk = 100 нс

В даному випадку часова діаграма повністю збігається з часовою діаграмою для вихідного опису.

ПК в ПЛІС в САПР XILINX ISE 14.7 для пристрою FPGA на платі Spartan 3 E Starter kit Xilinx XC 3 S 500 E -5 fg 320.

Мінімальний період синхросигналу для FPGA clk = 4.215 нс.

Нижче наведена часова діаграма на періоді clk = 12 нс, щоб було зручніше порівняти з CPLD.

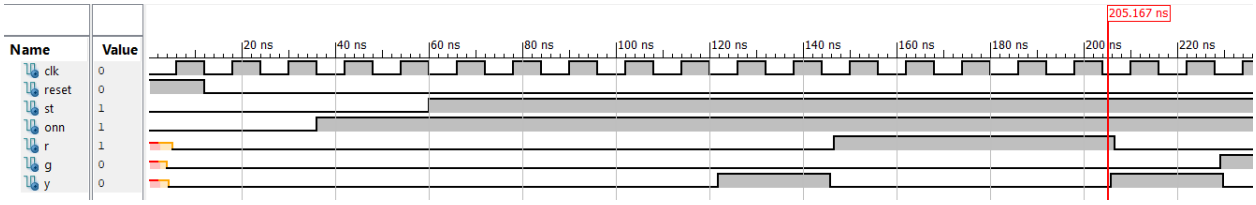


Рисунок 4.8 – Часова діаграма на періоді clk = 12 нс

Результати моделювання повністю збіглися з CPLD.

Нижче наведена часова діаграма на періоді clk = 100 нс.

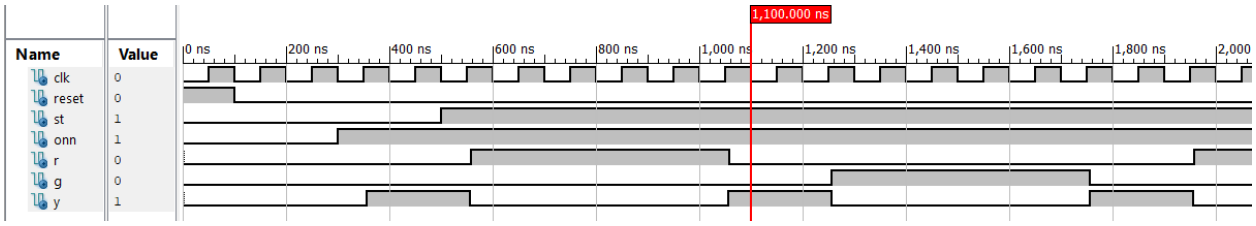


Рисунок 4.9 − Часова діаграма на періоді clk = 100 нс

В даному випадку часова діаграма повністю збігається з часовою діаграмою для вихідного опису.

У таблиці 4.1 представлені порівняльні характеристики результатів моделювання після імплементації.

Таблиця 4.1 − Порівняльні характеристики результатів моделювання після імплементації

мікросхема / Період clk , нс	Число процесів в шаблоні	Затримки перемикання кольорів світлофора в% від періоду clk			
		Y → G	G → Y	Y → R	R → Y
CPLD/ 12	1	0	0	64	64
CPLD/ 22	1	0	0	35	35
CPLD/ 100	1	0	0	6,7	7,7
FPGA/ 12	1	0	0	64	64
FPGA/ 100	1	0	0	6,7	7,7
CPLD/ 12	2	3,9	3,9	6,5	6,5
CPLD/ 100	2	0	0	0	0
FPGA/ 12	2	3,9	3,9	6,5	6,5
FPGA/ 100	2	0	0	0	0

З таблиці видно, що найкращий результат за випадковим збігом часових діаграмм до синтезу і після імплементації дав двопроцессний шаблон з максимальним періодом clk.

ВИСНОВКИ

У ході науково-дослідницької роботи були оглянуті автоматні шаблони в технології проектування систем реального часу.

Для цього було проведено огляд літератури по даній тематиці. Також виконано синтез та оцінка апаратних витрат по звіту, у тому числі і оцінку витрат апаратури модулів по RTL-схемі з пакета XILINX ISE 10.1.01, плата Spartan 3E та плата Virtex 5. Оцінку витрат апаратури за звітами після синтезу в системі XILINX.

1) Описан керуючий автомат за допомогою однопроцесного шаблону та створеного двопроцесного шаблону на прикладі світлофора.

2) Досліджено поведінку однопроцесного шаблону до імплементації та після неї.

3) Створено двопроцесний шаблон, та перевірено його на працездатність. Досліджена його поведінка до імплементації та після.

4) Для цих двух шаблонів описан керуючий автомат за допомогою однопроцесного шаблону та двупроцесного шаблону на прикладі світлофора.

5) У результаті порівняння до імплементації та після імплементації однопроцесного шаблону було виявлено, що чим більше період clk, тим ближче результати моделювання після імплементації до результатів моделювання вихідного опису. До того ж якщо тривалість кольорів світлофора обчислюється хвилинами, частота роботи пристрою не суттєва, а значить вибираємо більший період для досягнення більшої точності в роботі пристрою. А для забезпечення необхідної тривалості автоматного такту слід використовувати стандартний дільник частоти (лічильник).

6) Двопроцесний шаблон виявився точнішим з точки зору забезпечення збігу результатів моделювання вихідного модуля і опису синтезованого автомата після імплементації.

7) Кореляції між оцінками витрат апаратури модулів за звітом (по LUT), по RTL-схемі і за технологічним рівнем з пакета XILINX ISE 10.1 .01 немає. Це стосується і загальних витрат апаратури, і витрат тільки для комбінаційної частини. Тому оцінку витрат необхідно робити в прив'язці до конкретної плати і мікросхеми за звітом (по LUT). При цьому більш точною оцінкою буде оцінка по «частинам LUT» по способу 1.

8) Витрати на тригери можна оцінювати, як 13 або 21, на один тригер, особливої ролі це не грає.

9) Для різних мікросхем ПЛІС результати синтезу однієї і тієї ж схеми на RTL-рівні можуть іноді сильно відрізнятися. Також можуть бути відмінності і на технологічному рівні і за звітом.

Наукова новизна даного досліду – розроблено двопроцесний шаблон кінцевих керуючих автоматів Мура з часовим логічним управлінням, досліджена його придатність для автоматизованого синтезу, встановлено, що він дає результати моделювання після імплементації більш точні в порівнянні з однопроцесной шаблоном з точки зору забезпечення збігу результатів моделювання вихідного модуля і опису синтезованого автомата після імплементації. Запропонований базовий шаблон може бути розширено на автомат Мілі або С, які часто зручні при використанні в пристроях управління об'єктами в енергосистемах. Також досліджені можливості оцінювати апаратурні витрати для різних реалізацій схем з використанням САПР.

Практична значимість: шаблони опису алгоритмів функціонування кінцевих автоматів у системах логічного управління на мові VHDL можуть бути використані проектувальниками цифрових систем логічного управління і студентами університетів в навчальному процесі при вивченні теорії автоматів і створення власних проектів. а також можуть давати спрощення задачі верифікації.

Результати досліджень були опубліковані в тезах докладів на двух конференціях [17, 18].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Гамма Э. Приемы объектно−ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влассидес. − СПб.: Питер, 2001. – 368 с.

2. Harel D. Statechars: a Visual Formalism for complex systems / D. Harel // Science of Computer Programming. Vol. 8, 1987, – P. 231−274.

3. Шалыто А.А. Использование граф−схем и графов переходов при программной реализации алгоритмов логического управления. Том I. / А.А. Шалыто // Автоматика и телемеханика. – 1996. – №6. – С.148−158.

4. Шалыто А.А. Использование граф−схем и графов переходов при программной реализации алгоритмов логического управления. Том II. / А.А. Шалыто // Автоматика и телемеханика. – 1996. – №7. – С.144−169.

5. Шалыто А.А. SWITCH−технология — автоматный подход к созданию программного обеспечения «реактивных» систем / А.А. Шалыто, Н.И. Туккель // Программирование. – 2001. – №5. – С.45−62.

6. Шалыто А.А. Алгоритмизация и программирование для систем логического управления и «реактивных» систем. / А.А. Шалыто // Автоматика и телемеханика. – 2001. – №1. – С.3−39.

7. Шалыто А. А. Автоматное программирование / Н.И. Поликарпова, А.А. Шалыто. – Спб.: Питер, 2008. – 168 с.

8. Шкиль, А.С. Обнаружение ошибок проектирования в HDL−моделях конечных автоматов с использованием синхронизирующих последовательностей / А.С. Шкиль, М.А. Мирошник, Э.Н. Кулак А.С. Гребенюк, Д.Е. Кучеренко // Радиоэлектроника и информатика. – 2016. – № 3(74). – С. 39−46.

9. Бибило П.Н. Синтез логических схем с использованием языка VHDL / П.Н. Бибило. – М.: СОЛОН−Р, 2009. – 384 с.

10. Haskell R. Digital Design Using Digilent FPGA Boards − VHDL /

Active−HDL Edition / Richard E. Haskell, Darrin M. Hanna. – LBE Books Rochester Hills, MI, 2009.– 381 p.

11. Alur R. A theory of timed automata / R. Alur, D. L. Dill // Theoretical Computer Science. – 1994. – V.126. – N 2. – P. 183−235.

12. Shkil A.S. Design automation of easy−tested digital finite state machines / M.A. Miroschnyk, Y.V. Pakhomov, A.S. Shkil, E.N. Kulak, D.Y. Kucherenko // Radio Electronics, Computer Science, Control. – 2018. – №2. – P. 117−124.

13. Miroschnyk M. Design automation of testable finite state machines / M. Miroschnyk, Y. Pakhomov, E. German, A. Shkil, E. Kulak, D. Kucherenko // IEEE EWDTS, Novi Sad, Serbia, September 29 – October 2, 2017, p. 203-208.

14. Тарабрин Б.В. Справочник по интегральным микросхемам / Б.В. Тарабрин, С.В. Якубовский, Н.А. Барканов и др.; под ред. Б.В. Тарабрина. – 2-е изд., перераб. и дополн. – М.: Энергия, 1981. – 816 с.

15. Spartan-3E FPGA Family Data Sheet [Электронный ресурс] – Режим доступу: http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf. – Загл. з екрану.

16. Spartan-3E FPGA Family Data Sheet [Электронный ресурс] – Режим доступу: https://www.xilinx.com/support/documentation/data_sheets/ds100.pdf. – Загл. з екрану.

17. Кучеренко І.О Розробка шаблону опису кінцевих автоматів з використанням темпоральних графів / І.О. Кучеренко // Радіоелектроніка та молодь у XXI столітті. – 2019. – № 5. – С. 13−14.

18. Кучеренко І.О Використання темпоральних графів при розробці шаблону опису алгоритмів функціонування скінченних автоматів / І.О. Кучеренко // Комп’ютерна інженерія і кібербезпека: досягнення та інновації. – 2018. – С. 262 − 263.

ДОДАТОК А

Графічний матеріал атестаційної роботи

Атестаційна робота магістра

Автоматні шаблони в технології проектування систем реального часу

Виконав: студент групи СКСм-18-1
Кучеренко Ірина
Керівник: доц. Кулак Е.М.

Об’єкт, предмет, ціль роботи

Об’єкт дослідження: кінцеві часові керуючі (не мікропрограмні) автомати і оцінка апартурних витрат за допомогою САПР.

Предмет дослідження: 1) шаблони опису кінцевих часових керуючих автоматів на мові опису апаратури VHDL, придатних для автоматизованого проектування, 2) кореляція між апаратурними витратами, що надаються в звіті САПР XILINX ISE і апаратурними витратами по Квайну, які оцінюються по RTL-схемі після синтезу.

Ціль роботи: підвищення ефективності процесу розробки кінцевих керуючих автоматів з часовим логічним керуванням на мові опису апаратури VHDL шляхом використання чіткого шаблону.