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)

Моделі проектування спеціалізованих обчислювачів
дробово-ірраціональних функцій на основі ПЛІС.
.
(тема)

Виконав: студент 2 курсу, групи СКСм-18-2
Селезньова Є.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 к.т.н. доц. Ларченко Л.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ _____
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Автоматизації проектування обчислювальної техніки
Рівень вищої освіти другий (магістерський)
Спеціальність 123 – Комп'ютерна інженерія
Тип програми Освітньо-професійна
Освітня програма Спеціалізовані комп'ютерні системи

(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Селезньовій Євгенії Олегівні
(прізвище, ім'я, по батькові)

1. Тема роботи _____
Моделі проектування спеціалізованих обчислювачів дробово-ірраціональних функцій на основі ПЛІС
затверджена наказом по університету від 04 11 2019 р. № 1624Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____ 20__ р.
3. Вихідні дані до роботи _____
FPGA кристал сімейства Xilinx Spartan-3E серії XC3S500E
САПР Active-HDL
Мова опису апаратури VHDL
4. Перелік питань, що потрібно опрацювати в роботі _____
1 Аналіз предметної області та постановка завдання дослідження
2 Математична модель біт-потокового обчислювача дробово-ірраціональних функцій
3 Структурний синтез спроектованого обчислювача
4 Апаратна реалізація спроектованої моделі обчислювача
5 Верифікація, тестування та імплементація обчислювача в платформу ПЛІС

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

18 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

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання	03.09.2019 - 04.09.2019	
2	Аналіз предметної області	05.09.2019 - 15.09.2019	
3	Аналіз джерел з проблемної галузі	16.09.2019 - 01.10.2019	
4	Розробка математичної моделі обчислювача	02.10.2019 - 15.10.2019	
5	Структурний синтез проект. обчислювача	16.10.2019 - 30.10.2019	
6	Розробка апаратної реалізації обчислювача	01.11.2019 - 14.11.2019	
7	Перевірка правильності роботи обчислювача	15.11.2019 - 18.11.2019	
8	Оформлення пояснювальної записки	19.11.2019 - 29.11.2019	
9	Оформлення графічного матеріалу	30.11.2019 - 04.12.2019	
10	Перевірка виконаного проекту керівником	05.12.2019 - 10.12.2019	

Дата видачі завдання 03.09.2019

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка атестаційної роботи: 71 сторінку, 11 рисунків, 15 таблиць, 12 джерел за переліком посилань.

ПЛІС, САПР, ІМПУЛЬСНИЙ ПОТІК, АПРОКСИМАЦІЯ, БІТ-ПОТОКОВИЙ ОБЧИСЛЮВАЧ, ДРОБНО-РАЦІОНАЛЬНИЙ МОДУЛЬ, СУМАТОР, ГРАФ–СХЕМА АЛГОРИТМУ, VHDL–МОДЕЛЬ, ВЕРИФІКАЦІЯ

В атестаційній роботі досліджено та розроблено біт-потоковий апаратний обчислювач дробово-ірраціональних функцій з вхідними і вихідними імпульсними потоками даних. Розроблено математичну модель обчислювач дробово-ірраціональних функцій другого степеню з використанням відомого методу ступінчастої апроксимації висхідних неперервних функції. Розроблено структуру обчислювача заданої функції, що представляє собою синтез двох блоків: блоку обчислення дробово-раціональних функцій та блоку обчислення функції добування кореня. Використано принцип побудови біт-потокових апаратних обчислювачів на основі конвеєрних архітектур.

Здійснено апаратну реалізацію спроектованого пристрою заданої функції, розроблено граф-схему алгоритму його роботи, яка закодована для синтезу автомата Мура, розроблено граф переходів керуючого автомату Мура. За графами переходів з використанням стандартних шаблонів кода розроблено модель пристрою на мові опису апаратури VHDL.

Розроблено та використано тест-бенч для верифікації отриманого рішення. Результати моделювання співпадають з теоретичними розрахунками. Модель синтезована в ПЛІС Xilinx Spartan.

THE ABSTRACT

Explanatory note of the diploma work contains 71 pages, 11 illustrations, 15 tables, 12 sources.

FPGA, CAD, IMPULSE FLOW, APPROXIMATION, BIT-FLOW COMPUTER, FRACTIONAL-RATIONAL MODULE, ADDER, GRAPH DIAGRAM OF THE ALGORITHM, VHDL–MODEL, VERIFICATION

The bit-flow computer of fractional-irrational functions with input and output impulse data streams is investigated and developed in the diploma work. A mathematical model for calculating second-degree fractional-irrational functions using the well-known method of stepwise approximation of increasing continuous functions is developed. The structure of the computer of a given function representing the synthesis of two parts is developed. The first part is computer of fractional-rational functions and the second part is computer of calculating the function of root extraction. The principle of construction of bit-flow hardware based on pipeline architectures is used. The principle is implemented by the functional conversion with deployment type.

The hardware implementation of the designed device of the defined function is developed. The graph diagram of the algorithm of his work is developed. The diagram is encoded for Moore machine synthesis. Moore's control automaton transition graph was developed. Transition graphs using standard code templates have developed a device model in the VHDL description language. The hardware model was developed with VHDL by transition graphs using standard code templates.

Test bench developed and applied to test the developed solution. The simulation results coincide with the theoretical calculations. The model is synthesized in FPGA Xilinx Spartan.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 8
ВСТУП 9
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ 11
1.1 Галузі застосування біт-потокових обчислювачів математичних функцій 11
1.2 Особливості проектування з використанням ПЛІС 14
1.3 Постановка завдання дослідження 17
2 МАТЕМАТИЧНА МОДЕЛЬ БІТ-ПОТОКОВОГО ОБЧИСЛЮВАЧА ДРОБОВО-ІРРАЦІОНАЛЬНИХ ФУНКЦІЙ 19
2.1 Особливості функціонального перетворення імпульсних потоків в апаратних обчислювачах відтворення математичних функцій 19
2.2 Метод ступінчастої апроксимації неперервних висхідних функцій 22
2.3 Математичне обґрунтування обчислення дробово–ірраціональних функцій 25
2.3.1 Математичні основи реалізації дробово-ірраціонального перетворення імпульсних потоків 26
2.3.2 Математична модель апаратного обчислювача дробово-раціональних функцій 31
2.3.3 Математична модель обчислювача добування квадратного кореня 32
2.3.4 Алгоритм обчислення поліноміальних функцій 33
3 СТРУКТУРНИЙ СИНТЕЗ СПРОЕКТОВАНОГО ОБЧИСЛЮВАЧА 36
3.1 Принцип побудови біт-потокових апаратних обчислювачів на основі конвеєрних архітектур 36
3.2 Структурно-функціональна модель досліджуваного обчислювача ірраціональних функцій 37

3.2.1 Дробово-раціональний модуль. 39
3.2.2 Пристрій для добування кореня 42
3.3 Вибір елементної бази 44
4 АПАРАТНА РЕАЛІЗАЦІЯ СПРОЕКТОВАНОЇ МОДЕЛІ ОБЧИСЛЮВАЧА 47
4.1 Специфікація досліджуваного обчислювача 47
4.2 Обчислювальний процес в компонентах досліджуваного пристрою 48
4.3 Автоматна модель спеціалізованого обчислювача 52
4.4 Структурно-блокова схема досліджуваного обчислювача 55
4.4.1 Опис блоків проекту на МОА 59
4.4.2 Опис проекту на МОА 60
4.5 Верифікація та тестування роботи пристрою 63
4.6 Імплементація моделі обчислювача дробово-ірраціональної функції 64
ВИСНОВКИ 68
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 70
ДОДАТОК А ..... **ОШИБКА! ЗАКЛАДКА НЕ ОПРЕДЕЛЕНА.**
ДОДАТОК Б ..... **ОШИБКА! ЗАКЛАДКА НЕ ОПРЕДЕЛЕНА.**
ДОДАТОК В ..... **ОШИБКА! ЗАКЛАДКА НЕ ОПРЕДЕЛЕНА.**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КСАУ – комп’ютерні системи автоматичного управління

ПЛІС – програмована логічна інтегральна схема

САПР – система автоматизації проектування

ЕОМ – електронно-обчислювальна машина

ГСА – граф-схема алгоритму

ПДК – пристрій для добування кореня квадратного

МОА – мова опису апаратури

КМОН – комплементарна структура метал-оксид-напівпровідник

ОЗП – оперативний запам'ятовуючий пристрій

FPGA (Field Programmable Gate Arrays) – програмована користувачем вентильна матриця

МОА – мова опису апаратури

ВСТУП

В даний час широке застосування знаходить завдання створення систем, що можуть обробляти данні від багатьох різноорієнтованих сенсорів, а також розробка архітектур потокових процесорів, що містять типове процесорне ядро і декілька зовнішніх модулів обробки потоків. Основна мета розвитку даних архітектур – це спрощення взаємодії між компонентами, шляхом використання імпульсний (бітових) потоків для послідовної передачі даних. Такі біт-потокові пристрої, є складовою зовнішніх апаратних модулів децентралізованих систем, працюючих з потоковими формами.

Завдання узгодження сенсорів з системами збору і обробки даних має місце при розгляді питання удосконалення інтерфейсів, призначених для організації взаємодії «людина-компьютер», а також при розробці систем, що орієнтуються на наскрізні технології інтелектуальних сенсорів, інтернет-речей.

Дані, одержувані від різнорідних сенсорів, обробляються в паралельно працюючих каналах і далі передаються в обчислювальне ядро. При цьому застосовуються потокові форми представлення даних [1]. Інформаційні потоки формуються з елементарних інформаційних одиниць, які в залежності від базової реалізації можуть мати різну матеріальну сутність. В якості базової сутності фізичного носія біт-потокових даних можуть виступати електричні, оптичні, пневматичні, біологічні та інші сигнали. В електричних схемах в якості носія виступають електричні імпульси, на основі яких формуються потоки одиничних імпульсів і потоки широтно-імпульсно-модульованих сигналів [1].

При проведенні математичної обробки первинної вимірювальної інформації, що отримують з вимірювальних сенсорів, поряд з арифметичними, алгебраїчними та іншими операціями часто потрібне

виконання різних нелінійних (функціональних) перетворень імпульсних потоків.

При безперервному аналізі інформаційних процесів, що відбуваються в природі та технічних системах необхідне здійснення безперервного прийому потоку даних, безперервної обробки інформаційних елементів потоку у міру їх надходження і безперервне формування результату в процесі обробки. Для вирішення таких завдань необхідно створення елементів і пристроїв, орієнтованих на потокову обробку інформації. Реалізація потокового методу обчислень полягає в розгортці кодової інформації в часі з одночасним паралельно-послідовним виконанням перетворень над бітами і отриманими потоками час-імпульсних сигналів відповідно до необхідної функції [2] .

У більшості випадків чутливі елементи сенсорів формують аналогові вихідні сигнали, що призводить до необхідності їх перетворення в цифрову форму. Використання цифрових функціональних перетворювачів та обчислювальних пристроїв, імплементованих в платформу ПЛІС, забезпечує реалізацію досить складних алгоритмів (законів) управління, високі швидкодію та точність обчислень. Крім того, цифрові функціональні пристрої дозволяють спростити КСАУ шляхом застосування простих і надійних модулів. Таким чином, актуальним і важливим науково-технічним завданням є розробка моделей спеціалізованих обчислювачів на базі ПЛІС з використанням САПР на основі мов опису апаратури.

Метою атестаційної роботі є дослідження та розробка моделей і методів автоматизованого проектування біт-потокового обчислювача дробово-ірраціональних функцій на основі ПЛІС. Біт-потоковий обчислювач може бути застосований в якості функціональних перетворювачів імпульсних потоків в системах управління і контролю, в децентралізованих системах при узгодженні сенсорів з цифровими системами збору і обробки інформації в якості зовнішніх апаратних модулів потокової обробки даних в архітектурах потокових процесорів, що складаються з типового процесорного ядра і зовнішніх апаратних модулів потокової обробки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

В розділі розглянуто галузі застосування біт-потокових обчислювачів математичних функцій, особливості проектування пристроїв на основі ПЛІС, що виконується із застосуванням спеціалізованих САПР, сформульовано мету, об’єкт, предмет і постановку завдання дослідження.

1.1 Галузі застосування біт-потокових обчислювачів математичних функцій

Реалії сьогодення потребують постійного удосконалення систем, які отримують дані від великої кількості різноманітних сенсорів. Цей попит якнайкраще задовольняє підхід, що передбачає розробку архітектур потокових процесорів, складовими яких є типове процесорне ядро і зовнішні апаратні модулі потокової обробки.

Отримані від багатьох різнорідних датчиків дані, можуть паралельно оброблятися в декількох каналах і далі передаватися в обчислювальне ядро. Інформаційні потоки формуються з елементарних інформаційних одиниць, які в залежності від базової реалізації, можуть мати різну матеріальну сутність. В якості базової сутності фізичного носія даних у вигляді імпульсного потоку можуть виступати електричні, оптичні, пневматичні, біологічні та інші сигнали. В електричних схемах в якості носія виступають електричні імпульси, на основі яких формуються потоки одиничних імпульсів і потоки широтно-імпульсно-модульованих сигналів [1]. Це передбачає, що інформація передається у вигляді часових характеристик прямокутних одиничних імпульсів в системах з час-імпульсним перетворенням, що представляє собою процес перетворення певних часових інтервалів в цифрову величину, який здійснюється за допомогою заповнення цих інтервалів імпульсами опорної частоти.

Подібні потоки імпульсів будуються на базі одиничної системи числення. Вона передбачає, що кожне ціле число має бути представлено відповідним до цього значення числом імпульсів. Це непозиційна система, оскільки всі її імпульси рівноважні.

Очевидною перевагою даного принципу є його висока завадостійкість, але це також передбачає низьку швидкість передачі даних. Саме через цей недолік використання імпульсних потоків неможливе в ЕОМ загального призначення, проте вони знайшли своє місце в вирішенні завдань побудови широкого класу обчислювальних модулів з час-імпульсним перетворенням, які необхідні в системах автоматичного управління і регулювання, що працюють в комплексі з ЕОМ [2].

Це зумовлено тим, що перетворення імпульсного потоку може бути реалізовано найпростішими засобами при забезпеченні заданої точності обчислень. Окрім того, такі перетворювачі спроможні виконувати деякі обчислення, знімаючи навантаження з ЕОМ загального призначення, а також спрощуючи програмування роботи,одночасно пришвидшуючи всю системи в загалом. До того ж, використання імпульсного потоку дозволяє здійснювати розгортуюче функціональне перетворення, яке передбачає, що кожне наступне значення функції буде обчислено з урахуванням попереднього. Це дає змогу здійснювати функціональне перетворення в реальному часі [3].

Для аналізу безперервних природних або технологічних процесів необхідно приймати потік інформації, здійснювати її обробку і формувати результати безперервно, в процесі надходження даних. При вирішенні таких завдань використовують пристрої, що базуються на принципах потокової обробки інформації.

Такі завдання ставляться при вимірюванні температури і тиску в нафтових свердловинах, статичного і диференціального тиску в трубопроводах, швидкостей, переміщень, прискорень і тисків в авіаційній техніці [3].

Особливо широке застосування функціональні перетворювачі знайшли

при реалізації обчислювальних пристроїв, що використовуються, як основний вузол деяких цифрових вимірювальних приладів, та реалізують функціональні залежності між вхідними та вихідними величинами. Зокрема, до них відносяться аналізатори спектра частотно-модульованих сигналів, які визначають модулі комплексних спектральних складових, цифрові амперметри і вольтметри, що визначають діючі показники несинусоїдних періодичних струмів і напруг, а також цифрові прилади для вимірювання віброшвидкості та віброприскорення, які призначені для визначення параметрів вібрації і знаходять використання в обчислювальній техніці в області систем контролю, управління і регулювання.

Функціональне перетворення застосовується в задачах лінеаризації і масштабування отриманих від датчиків сигналів, при вирішенні завдань:

– непрямого вимірювання;

– отримання коригувальних сигналів в системах управління;

– знаходження нелінійних математичних залежностей вихідних сигналів;

– здійснення статистичної обробки вимірюваних даних.

Інколи, функціональне перетворення в обчислювальних системах і пристроях може реалізуватися автономно. Зокрема, при обчисленні множення-ділення, тригонометричних перетворень та деяких інших математичних операцій. Під час здійснення математичної обробки первинних вимірювальних даних, поряд з арифметичними, алгебраїчними та іншими операціями часто потрібне виконання різних нелінійних (функціональних) перетворень імпульсних потоків.

Виконання обчислювальних операцій в таких перетворювачах зводиться зазвичай до вирішення завдання отримання нової послідовності імпульсів частоти, пов'язаної певної функціональною залежністю з частотами вихідних імпульсних послідовностей, які отримують за допомогою елементів частотозадаючих ланцюгів генераторів, що змінюють свої параметри під впливом зовнішніх дій [4].

1.2 Особливості проектування з використанням ПЛІС

Проектування цифрових пристроїв являє собою ітераційний процес, заснований на принципах функціональної декомпозиції. Проектування традиційно поділяють на етапи: системний, структурно-алгоритмічний, функціонально-логічний, конструкторсько-технологічний [4].

На системному етапі весь проект розбивається на частини, визначаються їх призначення і взаємозв'язок, приймається рішення про способи реалізації частин. Рішення про використання ПЛІС, прийняте на системному етапі, дозволяє виконувати конструкторсько-технологічне проектування модуля верхнього рівня паралельно з виконанням інших етапів. Проектування пристроїв на основі ПЛІС виконується із застосуванням спеціалізованих САПР, що у великій мірі диктують методику і засоби розробки, а також елементну базу. При цьому використовуються як візуальні засоби проектування, що мають добру наочність, так і засоби командного управління процесом проектування, що сприяють більшій автоматизації. Проектування за допомогою таких САПР полягає в послідовному використанні наданих програмних засобів. У термінології САПР такий процес називається маршрутом проектування.

Структурно-алгоритмічний і функціонально-логічний етапи проектування пристроїв на основі ПЛІС базуються на ітераційному введенні і верифікації описів паралельно функціонуючих процесів, кожен з яких реалізує заданий алгоритм.

Сучасні САПР підтримують кілька способів опису пристрою:

– з використанням мов опису апаратних засобів (VHDL, Verilog, AHDL та ін.) і спеціалізованого текстового редактора;

– схемотехнічний спосіб опису за допомогою програми візуального проектування, що дозволяє розробнику поміщати на робочу область функціональні блоки і здійснювати їх з'єднання. Після закінчення візуального проектування схема перетворюється в мовний опис;

– графічне представлення цифрових автоматів в спеціалізованому редакторі, що забезпечує перетворення отриманого графічного представлення в мовний опис;

– опис комбінаційної логіки за допомогою таблиць істинності, карт Карно, функцій алгебри логіки.

Конструкторсько-технологічний етап проектування пристроїв з використанням ПЛІС розділяється на пов'язані завдання, схожі з завданнями, які розв'язуються при схемно-топологічному проектуванні на основі замовних ІС. Даний етап включає наступні завдання [4]:

– синтез (Synthesis) – відображення схеми в базис логічних ресурсів ПЛІС. Мета синтезу – перетворення вихідного схемотехнічного або високорівневого опису пристрою в опис, що оптимально реалізується на обраній ПЛІС, а також придатний для подальшого розміщення і трасування. На стадії синтезу і після неї використовуються різні методи оптимізації опису, спрямовані на досягнення найкращих результатів з точки зору мінімуму необхідних ресурсів кристала, максимуму частоти синхросигналу, мінімуму споживаної потужності;

– глобальне розміщення (Mapping) – призначення частин схеми макроділянок ПЛІС, що представляють собою групи сусідніх логічних блоків, макрокомірок і блоків введення/виводу. Мета глобального розміщення – створення найкращих умов для локального розміщення і трасування. Для досягнення цієї мети використовується інформація про відповідність пристрою зовнішнім сигналам. Як правило, призначення логічних ресурсів кристала макроділянкам здійснюють з надмірністю, що полегшує подальше трасування;

– локальне розміщення (Placement) – детальне призначення логічних ресурсів макроділянок, обраних на стадії глобального розміщення, частинам схеми. При цьому переслідуються цілі рівномірного заповнення макроділянок елементами і трасами, мінімізації сумарної довжини ліній зв'язку та ін. Основна мета локального розміщення – створення найкращих

умов для трасування;

– трасування (Routing) - визначення зв'язків між логічними блоками, макрокомірками і блоками вводу/виводу у вигляді комутованих ділянок трас. На даній стадії переслідуються цілі вибору трас, що забезпечують заданий час поширення сигналу; мінімізації сумарної кількості програмованих точок зв'язку; мінімізації часу поширення сигналу по найдовшій лінії зв'язку. За результатами трасування можуть бути визначені часові параметри отриманого варіанту пристрою і виконано їх порівняння з заданими обмеженнями.

Процес проектування є ітераційним. Після виконання кожного завдання проводиться верифікація отриманого опису, для чого застосовуються різні засоби моделювання та аналізу. У сучасних САПР для моделювання використовуються наступні види описів:

– вихідний поведінковий опис;

– опис рівня реєстрових передач (RTL-опис);

– технологічний опис після синтезу;

– опис на вентильному рівні;

– технологічний опис з урахуванням результатів розміщення;

– технологічний опис з урахуванням результатів трасування.

Особливу увагу розробників приділяється опису тестових впливів, так як грамотно складений тест дозволяє виявити більшість помилок. Опис тестів може бути виконано кількома способами, в тому числі наступними:

– опис генератора тестових впливів, що не підлягає подальшому синтезу. Для моделювання цим способом використовується пара взаємопов'язаних описів: опис генератора і опис цільового пристрою. На вихідних портах цільового пристрою в процесі моделювання визначається оклик на тестовий вплив. Цей спосіб є універсальним, описаний мовою VHLD або Verilog, тестуючий пристрій-генератор може успішно застосовуватися в будь-якому моделируючому пакеті. Крім того, при описі тесту на мовах VHDL, Verilog можуть бути використані широкі алгоритмічні

можливості мов: цикли, очікування, пастки, повідомлення, генерації;

– опис тестових впливів в спеціальному графічному редакторі. Такий спосіб підходить для простих тестових впливів.

Після виконання трасування і верифікації результатів автоматично може бути згенеровано файл с конфігураційною послідовністю, що містить інформацію про комутацію та функціональність всіх ресурсів кристала. На заключному етапі маршруту проектування виконуються програмування ПЛІС і подальша внутрішньосхемна верифікація пристрою (перевірка працездатності на макетної ПЛІС). При цьому розробник може використовувати додаткове обладнання (осцилографи, логічні аналізатори, генератори сигналів) або скористатися спеціалізованими, що вбудовуються в ПЛІС, логічними аналізаторами.

1.3 Постановка завдання дослідження

Метою атестаційної роботи є розробка та дослідження апаратного біт-потокового обчислювача дробово-ірраціональних функцій на технологічній платформі ПЛІС з використанням інструментальних засобів САПР на основі мов опису апаратури для підвищення ефективності проектування спеціалізованого пристрою.

Об’єкт дослідження – спеціалізовані апаратні обчислювачі дробово-ірраціональних функцій перетворення імпульсних потоків з зовнішніх сенсорів фізичних величин.

Предмет дослідження – математичні, структурні моделі біт-потокових обчислювачів дробово-ірраціональної, дробово-раціональної функції і функції добування кореня із заданою абсолютною похибкою обчислень, засоби побудови структури біт-потокових обчислювачів на основі конвеєрних архітектур, апаратні моделі обчислювачів на основі цифрових автоматів.

Поставлена мета визначила наступні завдання дослідження:

– аналіз методу ступінчастої апроксимації відтворення неперервних висхідних функцій, аргумент яких представлений імпульсним потоком;

– розробка математичної моделі біт-потокового обчислювача дробово-ірраціональної функції, що є декомпозицією математичних моделей обчислювачів дробово-раціональної функцій другого степеню і функції добування квадратного кореня із заданою абсолютною похибкою обчислення функцій;

– аналіз способу побудови конвеєрних архітектур апаратних обчислювачів поліноміальних функцій;

– структурний синтез моделі обчислювача заданої функції на основі відомих структурних рішень біт-потокових обчислювачів дробово-раціональних функцій та функцій добування кореня, що входять до складу спроектованого пристрою;

– розробка апаратної реалізації пристрою на основі кінцевого автомата Мура;

– верифікація, тестування та імплементація апаратної моделі біт-потокового обчислювача в технічну платформу ПЛІС.

2 МАТЕМАТИЧНА МОДЕЛЬ БІТ-ПОТОКОВОГО ОБЧИСЛЮВАЧА ДРОБОВО-ІРРАЦІОНАЛЬНИХ ФУНКЦІЙ

В розділі розглядаються особливості функціонального перетворення імпульсних потоків в апаратних обчислювачах, математичне обґрунтування застосування методу ступінчастої апроксимації відтворення функцій визначеного класу. Розглянуто математичні основи реалізації дробово-ірраціонального перетворення імпульсних потоків, а також розроблено математичні моделі обчислювачів дробово-раціональної функції і функції добування кореня, що входять складовими в декомпозицію математичної моделі дробово-ірраціональної функції.

2.1 Особливості функціонального перетворення імпульсних потоків в апаратних обчислювачах відтворення математичних функцій

В [1] зазначено, що в даний час знаходять застосування спеціалізовані апаратні біт-потокові обчислювачі для відтворення безперервних висхідних функцій $y^* = f(x^*)$ методом ступінчастої апроксимації, обмеженнями яких є умови

$$x^* \in [0, 1], \quad y^* \in [0, 1],$$

$$\frac{dy^*}{dx^*} > 0, \tag{2.1}$$

де функція y*=f(x*) має зворотну $x^* = \psi(y^*)$.

При таких обмеженнях функції даного класу є монотонно зростаючими, що умовно розподіляються на два підкласи: першому з них

належать функції, що знаходяться нижче бісектриси, а другому – вище бісектриси першого координатного кута.

Особливістю біт-потокових апаратних обчислювачів відтворення математичних функцій є те, що вхідний x і вихідний y інформаційні сигнали являють собою два імпульсних потоки, періодичність проходження імпульсів першого з яких визначається способом квантування відтворюваної функції по аргументу, а другого – алгоритмом функціонування обчислювача.

У разі, коли функція y^* змінюється монотонно, на вхід пристрою подають періодичну послідовність імпульсів прямокутної форми, що являє собою імпульсний потік забезпечуючи рівномірне квантування аргументу цілочисельними значеннями x = 1,2,3, ... При синтезі таких пристроїв, мінімізують час і похибку обчислення функцій.

Якщо аргумент x і значення функції y представлені імпульсними потоками, мінімально можливий час обчислення буде забезпечено, якщо за час введення в обчислювач x одиничних імпульсів вхідного імпульсного потоку на його виході формується значення функції y, що являє собою вихідний імпульсний потік.

Оптимальним режимом за точністю обчислення пристрою можна вважати, якщо для всіх цілочисельних значень забезпечується граничне значення абсолютної похибки обчислення $|\delta_{max}|$, що не перевищує половини одиниці молодшого біту аргументу x.

В результаті реалізація оптимального методу формування ступінчастих функцій в обчислювачах, що розглядаються, з точки зору точності та часу обчислення (відтворення) має включати основні етапи:

Етап 1. Вибір апроксимуючої ступінчастої функції y для заданої безперервної $y^* = f(x^*)$ та граничного значення абсолютної похибки $|\delta_{max}|$ її обчислення в цілочисельних точках.

Етап 2. Встановлення функціонального зв'язку у вигляді аналітичної залежності між номерами вихідних імпульсів y=1, 2, 3, ... пристрою та

відповідними їм вхідними імпульсами $x_y = x_1, x_2, x_3$ ...., що обираються з вхідного імпульсного потоку x.

Етап 3. Створення технічного пристрою, що забезпечує на виході обчислювача формування вихідних імпульсів потоку y=1, 2, 3, … в моменти надходження на його вхід певних x_1, x_2, x_3,… імпульсів вхідного потоку x.

Безперервне формування приростів функції на виході пристрою можна забезпечити шляхом послідовної вибірки певних імпульсів з вхідного імпульсного потоку по мірі їх надходження на вхід обчислювача, так як існує обмеження $y \le x$. Наприклад, при x=100, значення функції $y = \sqrt{x}$ дорівнює 10. Це означає, що структура обчислювача, яка реалізує дану функцію, має забезпечити вибірку десяти імпульсів зі ста, що надійшли на його вхід.

У більшості випадків число одиничних імпульсів в імпульсному потоці аргументу є величина випадкова і задані лише його граничні значення x_{min} та x_{max}, при цьому x_{max} має порядок $10^6 \ldots 10^9$ та більше.

Процес обчислення цілочисельних значень функції в обчислювачі пов'язаний з операцією округлення дробових значень ґратчастої функції до цілих чисел в цілочисельних точках. В залежності від граничного значення абсолютної похибки обчислення $|\delta_{max}|$, обираються певні номера біт з вхідного імпульсного потоку x і подаються на вихід пристрою.

Отже, існує певна функціональна залежність між видом відтворюваної функції y, похибкою $|\delta_{max}|$, номерами y = 1,2,3, ... вихідних імпульсів пристрою та відповідними їм вхідними імпульсами x_y послідовності x.

Встановлення цього зв'язку дає можливість отримати формулу загального члена x_y числової послідовності x_1, x_2, x_3, .., подальший аналіз якої і дозволить перейти до синтезу технічного пристрою, практично здійснюючого в пристрої обчислення членів числового ряду.

2.2 Метод ступінчастої апроксимації неперервних висхідних функцій

В [1] розглянуто метод ступінчастої апроксимації неперервних висхідних функцій, що може бути використаний при описі математичної моделі апаратного обчислювача добування квадратного кореня, що входить складовим блоком в пристрій дробово-ірраціонального перетворення імпульсних потоків.

Насамперед зауважимо, що при даному підході цілочисельні значення функції y, що обчислюються із заданою похибкою $|\delta_{max}|$, можуть бути відтворені на виході обчислювача ступінчастою функцією

$$y = [f(x) + |\delta_{max}|] \qquad (2.2)$$

де x=1,2,3,...; $|\delta_{max}|$ – задане граничне значення абсолютної похибки відтворення відповідних неперервних функцій. В (2.2) квадратні дужки позначають цілу частину числа.

Перейдемо до пошуку значень $x_y = x_1, x_2, x_3$ - незалежної змінної x, які відповідають моментам початку формування кожної y = 1,2,3, ... сходинки апроксимуючої вихідної функції пристрою.

Для функції (2.2) з урахуванням обмеження $y \le x$ для будь-якого рівня $y - |\delta_{max}|$, де y = 1,2,3, ..., можна вказати пару сусідніх цілочисельних значень аргументу $x_y - 1$ та x_y, для яких має місце система нерівностей (рисунок 2.1)

$$\begin{cases} f(x_y - 1) < y - |\delta_{max}| \\ f(x_y) \ge y - |\delta_{max}|, \end{cases} \qquad (2.4)$$

де $f(x_y - 1)$ – значення функції в точці x_y-1, $f(x_y)$ – значення функції в точці x_y.

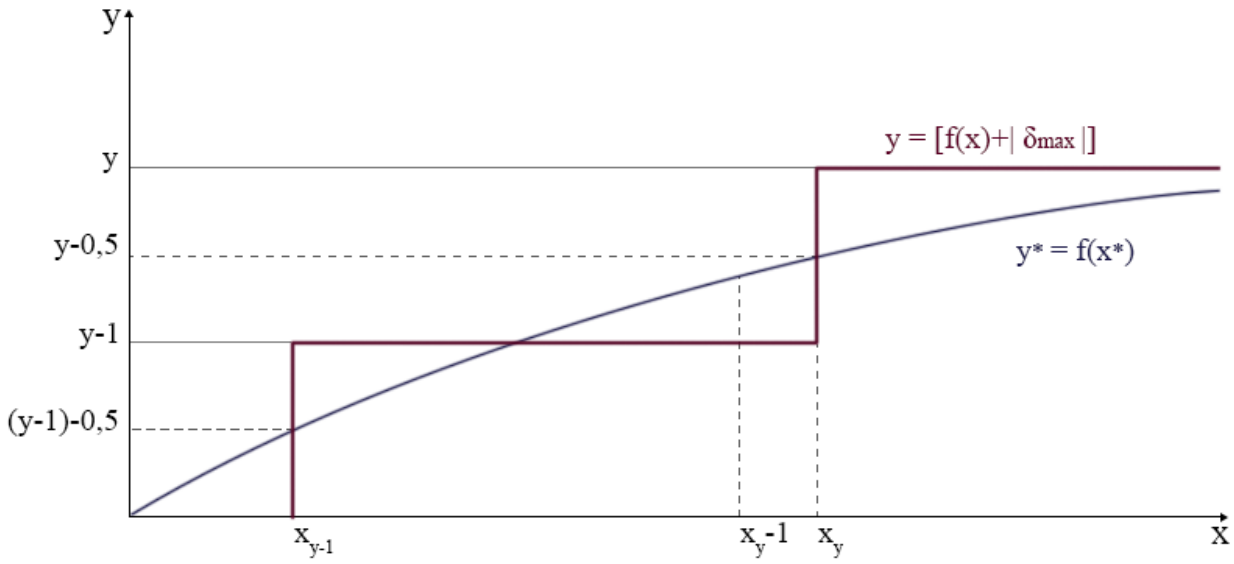

Рисунок 2.1 – Безперервна і апроксимуюча функції

Визначаючи з системи (2.4) значення x_y, отримаємо формулу загального члена числової послідовності x_1, x_2, x_3,.., відповідну вузлам апроксимації вихідної функції y

$$\varphi(y-\delta_{max}) \le x \le \varphi(y+\delta_{max}), \qquad (2.5)$$

де $\varphi(y-\delta_{max})$ функція, зворотна $f(x)$.

Значення x_y можуть бути знайдені шляхом послідовної підстановки y=1,2,3, ... в нерівність (2.5), обчисленні лівої його частини і округленні одержуваних значень в більшу сторону до найближчого цілого числа, або обчисленні правої частині і округленні в меншу сторону [1].

Так як ліва і права частини нерівності (2.5) відрізняються на одиницю, кожне таке округлення дозволяє отримати єдине значення x_y.

З огляду на це, нерівність (2.5) можна замінити рівністю

$$x_y = [\varphi(y-\delta_{max})]. \qquad (2.6)$$

При значеннях y, яким відповідають цілочисельні значення $\Psi(y+|\delta_{max}|)$, нерівність (2.5) трансформується в рівність

$$x_y=\Psi(y+|\delta_{max}|). \quad (2.7)$$

В окремому випадку, при $|\delta_{max}|=0{,}5$, буде забезпечена мінімальна похибка обчислення функції в цілочисельних точках аргументу. При цьому розрахункові формули (2.5), (2.6), (2.7) приймуть вигляд

$$\Psi(y-0{,}5)<x_y<\Psi(y+0{,}5), \quad (2.8)$$

$$x_y=[\Psi(y-0{,}5)]+1, \quad (2.9)$$

$$x_y=\Psi(y-0{,}5) \quad (2.10)$$

і будуть відповідати оптимальній апроксимації з точки зору точності обчислення значень функції y.

Метод апроксимації забезпечує безперервний процес відтворення функції (2.2) в реальному часі в темпі надходження бітів аргументу x.

Як приклад, використаємо розглянуту вище методику обчислення x_y для функції $y=[\sqrt[3]{x}+0{,}5]$, яка апроксимує безперервну $y^*=\sqrt[3]{x^*}$ з абсолютною похибкою обчислення $|\delta_{max}|=0{,}5$.

У цьому випадку нерівність (2.5) набуває вигляду

$$(y-0{,}5)^3<x_y<(y+0{,}5)^3. \quad (2.11)$$

Підставляючи в (2.11) $y=1, 2, 3, \ldots$, отримаємо значення вибірок x_y=1, 4, 16, 43, …Ця послідовність формує на виході пристрою функцію

$y=[\sqrt[3]{x}+0,5]$, оптимальну з точки зору точності відтворення гратчастої функції $y=\sqrt[3]{x}$ в цілих числах.

Наведений приклад показує перевагу запропонованого методу. Особливістю є те, що число вихідних одиничних імпульсів у завжди менше числа вхідних одиничних імпульсів х.

Очевидно, для функцій, обмежених умовою $y > x$, число вихідних біт обчислювача на інтервалі відтворення завжди більше числа вхідних х.

З математичної точки зору це означає, що для таких функцій одному і тому ж значенню x_y в (2.5) або в (2.8) відповідає деяка підмножина значень вихідної функції у.

2.3 Математичне обґрунтування обчислення дробово–ірраціональних функцій

Відповідно до мети дослідження апаратний обчислювач дробово–ірраціональних функцій, має обчислювати функцію

$$y=[\sqrt{[\frac{\sum_{i=0}^{2} a_i x^i}{m}+|\delta_{1max}|]}+|\delta_{2max}|]. \qquad (2.12)$$

де х – аргумент функції, що представляє собою імпульсний потік;

$|\delta_{1max}|$ – граничне значення абсолютної похибки ділення полінома $\sum_{i=0}^{2} a_i x^i$ на m;

$|\delta_{2max}|$ – граничне значення абсолютної похибки добування кореня квадратного.

При обчисленні поліномів з цілими коефіцієнтами похибка обчислення відсутня, тому похибка обчислення дробово-раціональної функції виникає

при поділі полінома на число m та при добуванні кореня і може бути забезпечена 0,5 одиниці молодшого біту числа x.

При абсолютній похибці $\delta_{max} \le \delta_{max} \le 0$ забезпечується оптимальна мінімально можлива похибка відтворення заданої функції, так як і результат ділення, і результат добування кореня округляється до найближчого цілого числа [5].

Вхідним інформаційним сигналом обчислювача є поданий на його вхід імпульсний потік x. На виході пристрою формується імпульсний потік y, що відтворює задану безперервну функцію і має похибку, яка не перевищує половину одиниці молодшого біту аргументу.

З огляду на вищесказане, обчислювач дробово-ірраціональних функцій має реалізувати функцію

$$y = [\sqrt{[\frac{\sum_{i=0}^{2} a_i x^i}{m} + 0{,}5]} + 0{,}5] \qquad (2.13)$$

Обчислення (2.13) полягає в послідовному виконанні наступних дій над імпульсним потоком x, що надходить на вхід обчислювача:

– операції визначення полінома другого степеню $\sum_{i=0}^{2} a_i x^i$ та ділення отриманого результату на ціле число m, що поєднує в собі блок обчислення дробово-раціональної функції;

– операцію визначення квадратного кореня з цілого числа, отриманого в результаті обчислення дробово-раціональної функції.

2.3.1 Математичні основи реалізації дробово-ірраціонального перетворення імпульсних потоків

Розглянемо математичні основи реалізації дробово-ірраціональних функцій, що базуються на обчисленні дробово-раціональних функцій на

першому етапі, та обчисленні функції добування кореня – на другому.

В [5] розглянуто математичні моделі апаратних дробово-раціональних пристроїв перетворення імпульсних потоків.

Дробово-раціональні пристрої в загальному випадку призначені для відтворення неперервних функцій виду

$$y^* = \frac{\sum_{i=0}^{n} a_i x^{*i}}{\sum_{j=0}^{l} b_j x^{*j}}, \qquad (2.14)$$

Якщо вихідна функція задовольняє обмеженням (2.1), вона може бути відтворена функцією (2.2).

Тоді з урахуванням (2.2) і $|\delta_{max}| = 1$ апроксимуюча функція набуде вигляду

$$y = [\frac{\sum_{i=0}^{n} a_i x^{i}}{\sum_{j=0}^{l} b_j x^{j}}] \qquad (2.15)$$

Досліджуваний обчислювач в підкореневому виразі має відтворювати неперервну дробово-раціональну функцію

$$y^* = \frac{\sum_{i=0}^{n} a_i x^{*i}}{m} \qquad (2.16)$$

де i, a_i, m– цілі додатні числа.

В (2.16) в чисельнику маємо поліном n-го ступеня, а поліном в знаменнику є константою m (a_0=m), так як коефіцієнти $a_l = a_{l-1} = a_{l-2} = ... = a_1 = 0$

Апроксимуюча дробово-раціональна функція, що відтворює неперервну (2.16)

$$y = [\frac{\sum_{i=0}^{n} a_i x^i}{m} + 0{,}5] \ . \qquad (2.17)$$

Задана абсолютна похибка відтворення, що забезпечує обчислення функції $|\delta_{max}| = 0{,}5$, є оптимальною з точки зору точності обчислення.

Так як функція (2.17) не має аналітичного виразу зворотної їй функції, і обчислення x_y є неможливим, опустивши квадратні дужки можна перейти до нерівності

$$\frac{\sum_{i=0}^{n} a_i x^i}{m} \geq (y - \frac{1}{2}) . \qquad (2.18)$$

Після перетворень, отримаємо

$$2\sum_{i=0}^{n} a_i x^i \geq m(2y-1) \qquad (2.19)$$

З нерівності (2.19) випливає, що першому (y=1) біту вихідного імпульсного потоку обчислювача буде відповідати той біт x_1 вхідного імпульсного потоку x=1, 2, 3,…, при якому буде виконана умова

$$2\sum_{i=0}^{n} a_i x_1^i \geq m \qquad (2.20)$$

Аналогічно, другий (y=2) і наступні біти вихідного імпульсного потоку обчислювача визначаються системою нерівностей

$$2\sum_{i=0}^{n} a_i x_2^i \geq 3m$$

$$2\sum_{i=0}^{n} a_i x_3^i \geq 5m \qquad (2.21)$$

. . . . . .

Дану систему нерівностей можна представити у вигляді різниць з урахуванням рівностей

$$\sum_{i=0}^{n} a_i x_y^i = \sum_{i=0}^{n} a_i x_y^i - \sum_{i=0}^{n} a_i x_{y-1}^i + \sum_{i=0}^{n} a_i x_{y-1}^i \qquad (2.22)$$

$$m(2y-1) = m(2y-1) - m(2y-1) - 1) + m(2y-1) - 1).$$

Отже, в результаті було отримано нерівність, що може бути використана при розробці математичної моделі обчислювача дробово-раціональної функції (4) в різницях

$$2(\sum_{i=0}^{n} a_i x_y^i - \sum_{i=0}^{n} a_i x_{y-1}^i) + \Delta_{y-1} \geq 2m \qquad (2.23)$$

В (2.23) Δ_{y-1} визначається, як

$$\Delta_{y-1} = 2(\sum_{i=0}^{n} a_i x_y^i - \sum_{i=0}^{n} a_i x_{y-1}^i) + \Delta_{y-2} - 2m\,. \qquad (2.24)$$

З (2.23) випливає, що визначення вибірок x_y може бути зведено до обчислення приростів ґратчастої функції $2\sum_{i=0}^{n} a_i x^i$ змінної x на кожному з інтервалів $(x_{y-1}; x_y]$ і їх порівнянні з приростами функції $m(2y-1)$ з урахуванням їх різниці Δ_{y-1}, отриманої на попередньому кроці інтервалу $(x_{y-2}; x_{y-1}]$ в точці x_{y-1}.

При обчисленні функції (2.13) на другому етапі необхідно добувати корень k-го степеню з цілого числа, отриманого в результаті обчислення дробово-раціональної функції.

На основі методу ступінчастої апроксимації функцій, розглянутого в [1]отримаємо математичну модель пристрою добування кореня, що входить до складу обчислювача дробово-ірраціональних функцій.

Апаратні біт-потокові обчислювачі добування кореня довільного степеню відтворюють неперервну функцію

$$y^* = \sqrt[k]{x^*} \tag{2.25}$$

Функція, що апроксимує безперервну функцію (2.25) і абсолютною похибкою $|\delta_{max}| = 0{,}5$ має вигляд

$$y = [\sqrt[k]{x} + 0,5] \tag{2.26}$$

Для функції (2.26), що задовольняє обмеженням $x, y \geq 0, y_i > y_{i-1}, y \leq x$ визначимо зворотну функцію і запишемо нерівність

$$(2y-1)^k \leq 2^k x_y < (2y-1)^k + 1 \tag{2.27}$$

Закон вибірки x_y при цьому визначається необхідною похибкою обчислення функції. Номери x_y вхідного імпульсного потоку x можуть бути визначені за формулою (2.27) при підстановці y = 1, 2, 3,… При цьому буде забезпечена абсолютна похибка обчислень, що не перевищує 0,5 одиниці молодшого біту аргументу.

В окремому випадку, коли для визначення зворотної функції використовується $|\delta_{max}| = 0{,}5$, вираз (2.27) набуватиме вигляду

$$x_y = [(y - 0,5)^k] + 1, \tag{2.28}$$

Вираз (2.28) можна використати при розробці математичної моделі пристрою добування квадратного кореня.

2.3.2 Математична модель апаратного обчислювача дробово-раціональних функцій

Згідно з метою дослідження, дробово-раціональна функція в підкореневому виразі набуває вигляду

$$y = [\frac{\sum_{i=0}^{2} a_i x^i}{m} + 0,5] \tag{2.29}$$

Скористаємось (2.22), (2.23) і (2.24) отримаємо математичну модель дробово-раціональної функції досліджуваного обчислювача, що включає поліном другого степеню.

Одержимо математичну модель обчислювача дробово-раціональної функції в різницях

$$\begin{aligned}
&2\sum_{i=0}^{2} a_i x_1^i \geq m \\
&2(\sum_{i=0}^{2} a_i x_2^i - \sum_{i=0}^{2} a_i x_1^i) + \Delta_1 \geq 2m \\
&2(\sum_{i=0}^{2} a_i x_3^i - \sum_{i=0}^{2} a_i x_2^i) + \Delta_2 \geq 2m \\
&\quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \\
&2(\sum_{i=0}^{2} a_i x_y^i - \sum_{i=0}^{2} a_i x_{y-1}^i) + \Delta_{y-1} \geq 2m ,
\end{aligned} \tag{2.30}$$

де $\Delta_{y-1} = 2(\sum_{i=0}^{2} a_i x_y^i - \sum_{i=0}^{2} a_i x_{y-1}^i) + \Delta_{y-2} - 2m$.

З системи нерівностей (2.30) можна стверджувати, що в цьому випадку визначення x_y може бути зведено до обчислення приростів ґратчастої функції $2\sum_{i=0}^{2} a_i x^i$ для змінної x на кожному з інтервалів $(x_{y-1}; x_y]$ та їх порівнянні з приростами функції $m(2y - 1)$ з урахуванням їх різниці Δ_{y-1}, отриманої на попередньому кроці інтервалу $(x_{y-2}; x_{y-1}]$ в точці x_{y-1}.

2.3.3 Математична модель обчислювача добування квадратного кореня

Для обчислення ірраціональної функції (2.13), після знаходження значення полінома і ділення отриманого результату на число m, необхідно видобути корінь квадратній з попередньо отриманого числа. Математична модель пристрою добування квадратного кореня, що входить до складу досліджуваного обчислювача, наведено в [5]. Для її розгляду скористаємося методом апроксимації функцій, який було представлено параграфі 2.2.

Функція $y=[\sqrt{x}+0,5]$ задовольняє обмеженням (2.1), отже для неї можна записати нерівність

$$[\quad\quad] \qquad (2.31)$$

Зауважимо, що при цьому необхідною похибкою обчислення функції визначається закон вибірки x_y.

Номери x_y вхідного імпульсного потоку x обчислимо за формулою (2.8), що забезпечить абсолютну похибку обчислень, не перевищуючу половину одиниці молодшого розряду.

В окремому випадку, коли для визначення зворотної функції використовується $|\delta_{max}|=0,5$, вираз (2.8) набуватиме вигляду

$$x_y = [(y-0{,}5)^2] + 1, \qquad (2.32)$$

звідки

$$x_y = y^2 - y + 1. \qquad (2.33)$$

При підстановці у вираз (2.33) $y = 1, 2, 3 \ldots$ утворюється числова послідовність x_y: 1, 3, 7, 13, 21, 31, ..., – арифметичний ряд 2-го порядку. Для даної послідовності арифметичні ряди різниць 1-го і 2-го порядків приймають наступний вигляд

Δ 2, 4, 6, 8, 10, ...

Δ^2 2, 2, 2, 2, ...

Отриманий арифметичний ряд 2-го порядку, а також ряди різниць 1-го і 2-го порядків можуть бути використані для синтезу обчислювача, який має виконувати добування квадратного кореня. Саме він необхідний нам в якості складової частини досліджуваного спеціалізованого обчислювача ірраціональних функцій.

2.3.4 Алгоритм обчислення поліноміальних функцій

При обчисленні дробово-раціональних функції (2.17), що мають вигляд

$$y = [\frac{\sum_{i=0}^{n} a_i x^i}{m} + 0{,}5] \quad , \qquad (2.35)$$

необхідно забезпечити обчислення полінома n-го степеню, що записано в чисельнику (2.35).

В основі конвеєрних біт-потокових обчислювачів поліномів n-го степеню лежить відомий алгоритм обчислення поліноміальних функцій. Розглянемо алгоритм обчислення поліномів виду

$$y = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \ldots + a_0, \qquad (2.34)$$

де n – ціле число, a_{n-1}, a_{n-2},..., a_0 – коефіцієнти.

Характерна ознака поліномів з цілочисельними коефіцієнтами a_i це відповідність послідовності його цілочисельних значень функції $y_0, y_1, y_2, y_3, \ldots y_i$ значенням $x_i = 0, 1, 2, 3, ..., i$. Підставляючи в вираз (2.34) $x_i = 0, 1, 2, 3, ..., i$ отримаємо числову послідовність y_i. При цьому утворений ряд значень функції $y_0, y_1, y_2, y_3, \ldots y_i$ називається арифметичним рядом n – го порядку. [6]

Питання синтезу поліноміального обчислювача вирішується шляхом зниження порядку різниць.

Для функції (2.34) можна записати значення функції y_i визначаються за виразом

$$y_i = f(i+1) - f(i), \tag{2.35}$$

де $i = 0, 1, 2, 3, ...$, $f(i+1)$ – значення функції в точці x_{i+1}, $f(i)$ – значення функції в точці x_i відповідно.

Різниці першого порядку для значення і аргументу визначаються за формулою

$$\Delta_i = y_{i+1} - y_i, \tag{2.36}$$

різниці другого порядку

$$\Delta_i^2 = \Delta_{i+1} - \Delta_i, \tag{2.37}$$

різниці n-го порядку

$$\Delta_i^n = \Delta_{i+1}^{n-1} - \Delta_i^{n-1} \quad . \tag{2.38}$$

В результаті отримаємо математичну модель обчислення поліноміальної функції (2.34), представлену в різницевих рівняннях

$$\Delta_i = y_{i+1} - y_i,$$

$$\begin{aligned}
\Delta_i^2 &= \Delta_{i+1} - \Delta_i, \\
\Delta_i^3 &= \Delta_{i+1}^2 - \Delta_i^2, \\
&\cdots\cdots\cdots\cdots \\
\Delta_i^n &= \Delta_{i+1}^{n-1} - \Delta_i^{n-1}.
\end{aligned} \tag{2.39}$$

де $i = 0, 1, 2, 3, \ldots$.

Вищеозначена методика може бути використана при проектуванні конвеєрних архітектур біт-потокових спеціалізованих обчислювачів.

При побудові поліноміальних обчислювачів компоненти конвеєрної архітектури потребують встановлення початкових умов, тобто компоненти мають бути ініціалізовані початковими значеннями $y_0, \Delta_0, \Delta_0^2, \ldots, \Delta_0^n$, а саме, першими членами арифметичного ряду n-го порядку та арифметичними рядами різниць 1-го, 2-го і n-го порядку відповідно.

3 СТРУКТУРНИЙ СИНТЕЗ СПРОЕКТОВАНОГО ОБЧИСЛЮВАЧА

В розділі розглянуто принцип побудови біт-потокових апаратних обчислювачів поліноміальних функцій на основі конвеєрних архітектур, структурний синтез моделі обчислювача заданої функції на основі відомих структурних рішень біт-потокових обчислювачів дробово-раціональних функцій та функцій добування кореня, що входять складовими блоками в спроектований пристрій.

3.1 Принцип побудови біт-потокових апаратних обчислювачів на основі конвеєрних архітектур

Конвеєризація (або конвеєрна обробка) в загальному випадку заснована на розділі функції, що має виконуватися на менші частини, звані ступенями, і виділенні для кожної з них окремого блоку апаратури. Так обробку будь-якого обчислення можна розділити на кілька етапів, організувавши передачу даних від одного етапу до наступного. При цьому конвеєрну обробку можна використовувати для поєднання етапів виконання різних обчислень. Продуктивність при цьому зростає завдяки тому, що одночасно на різних каскадах конвеєра виконуються декілька обчислення. Конвеєрна обробка такого роду широко застосовується швидкодіючих апаратних процесорах.

При побудові конвеєрних архітектур біт-потокових обчислювачів, конвеєр створюється при реалізації системи різницевих рівнянь (2.39), отриманої в параграфі 2.3.4, а саме при знаходженні арифметичних рядів різниць порядків і реалізується функціональним перетворенням розгортуючого типу.

Перетворення, в основі якого лежить послідовне обчислення значень функцій, виконаних для сусідніх значень аргумента, називається цифровим функціональним перетворенням розгортуючого типу. Такі перетворення

можуть враховувати попередню історію процесу, адже інформація, взята з кожного обчислення буде враховуватися при обчисленні наступного значення аргументу. Слід зауважити, що при цьому перше обчислення буде здійснено з урахуванням заданої інформації, наприклад, початкових умов, а саме, ініціалізації компонентів пристрою [6].

Функціональне перетворення розгортуючого типу може бути здійснено за допомогою біт-потокового пристрою обчислення поліноміальних функцій.

При побудові конвеєрних архітектур спеціалізованих біт-потокових обчислювачів основними компонентами є суматори, регістри, лічильники. Перевагами подібних архітектур з одного боку є простота побудови архітектури, швидкодія та завадостійкість, а з другого – можливість виконувати обчислення біт-потокових даних в реальному масштабі часу, при застосуванні біт-потокових обчислювачів в системах з час-імпульсним перетворенням вхідних сигналів.

Поліноміальні біт-потокові обчислювачі, що базуються на реалізації математичної моделі (2.39), в загальному випадку, містять у собі n паралельних накопичуючих суматорів SUM1, …, SUMn ; n груп елементів &1, …,&n ; n-1 елементів затримки DE1,…, DEn і регістр пам’яті константи, що утворюється в ряду різниць n-го порядку RG.

Саме на вищеозначеному принципі базується структура досліджуваного обчислювача.

3.2 Структурно-функціональна модель досліджуваного обчислювача ірраціональних функцій

Досліджуваний обчислювач має реалізувати функцію

$$y = [\sqrt{[\frac{\sum_{i=0}^{2} a_i x^i}{m} + 0{,}5]} + 0{,}5] \ . \qquad (3.1)$$

Перепишемо вираз (3.1) в іншому вигляді

$$y = [\sqrt{[\frac{ax^2 + bx + c}{m} + 0{,}5]} + 0{,}5] \ . \qquad (3.2)$$

При цьому, згідно з наведеними в попередньому розділі алгоритмами обчислення, перший етап знаходження значень функції зводиться до обчислення значень полінома другого степеню, що наведено в чисельнику (3.2), математична модель якого представлена системою різницевих рівнянь (2.40) та ділення отриманого результату на число m. Ці операції зручно реалізувати модулем для обчислення дробово-раціональних функцій.

Другий етап полягає у видобуванні квадратного кореня з цілого числа, отриманого після виконання 1-го етапу обчислень.

Виходячи с цього, можна стверджувати, що досліджуваний обчислювач реалізується двома блоками.

1. Дробово-раціональний модуль. Він включає суматори SUM1, SUM2, регістри RG1, RG2, а також елементи затримки DE1, DE2, і групи елементів &1, &2, &3.

2. ПДК, тобто пристрій для добування квадратного кореня. Він включає підсумовуючий лічильник CNT_{sqrt}, лічильник результату CNT_{res}, а також елемент затримки DE3 і групу логічних елементів &4 [7].

На рисунку 3.1 наочно представлено структурно-функціональну модель досліджуваного обчислювача дробово-ірраціональних функцій.

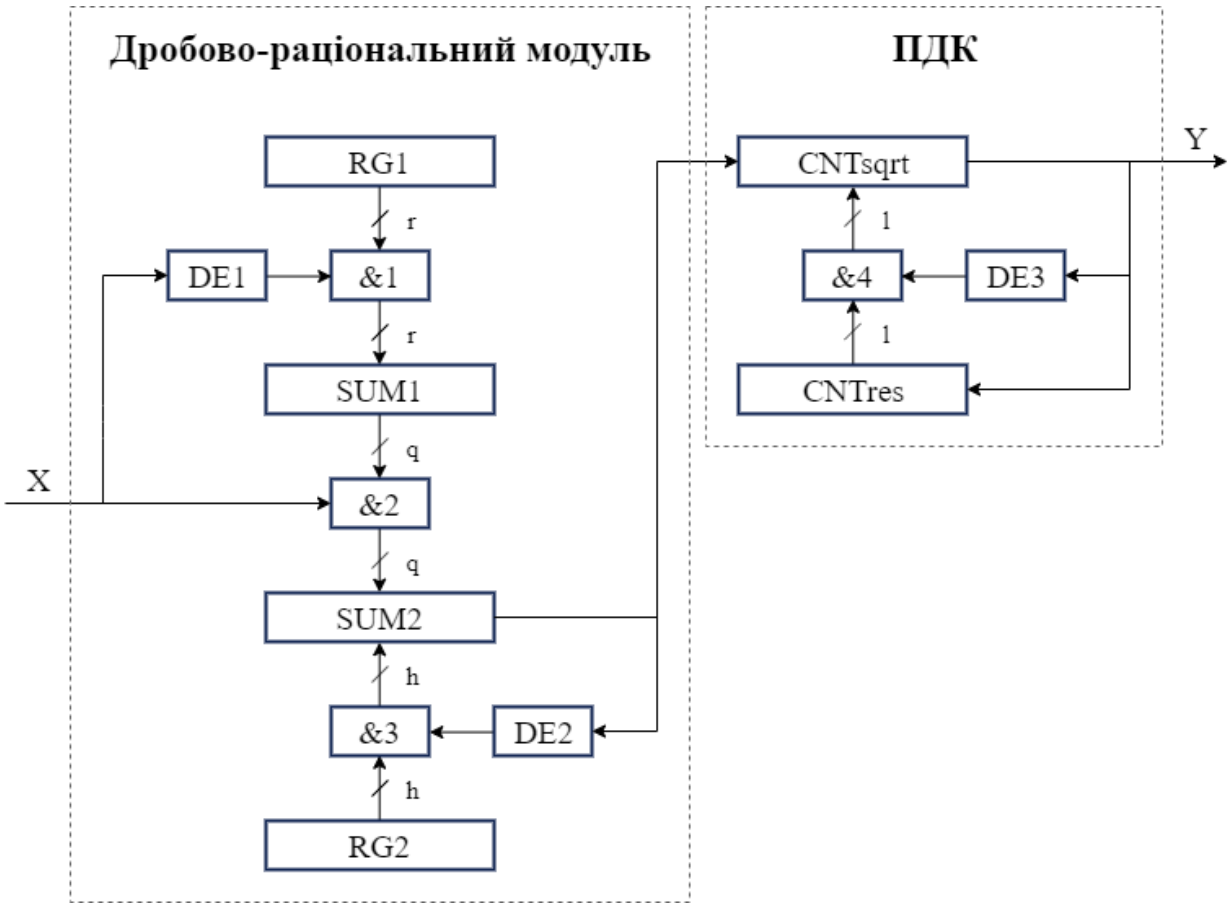

Рисунок 3.1 – Структурно-функціональна модель досліджуваного обчислювача

Розглянемо згадані блоки більш докладно.

3.2.1 Дробово-раціональний модуль.

Дробово-раціональний модуль призначений для обчислення функції $y_{int} = \frac{ax^2 + bx + c}{m} + 0{,}5$ і реалізує систему нерівностей (2.30).

Модуль складається з двох блоків:

1. Квадратор, який призначений для реалізації функції $ax^2 + bx + c$. Він включає наступні компоненти:

– суматор SUM1;

– групи логічних елементів &1 та &2;

– суматор SUM2;

– елемент затримки DE1;

– регістр RG1.

2. Дільник чисел, що призначений для ділення на число m з урахуванням абсолютної похибки відтворення функції $|\delta_{max}|=0,5$. Він включає наступні компоненти:

– суматор SUM2;

– група логічних елементів &3;

– елемент затримки DE2;

– регістр RG2.

Наведемо ініціалізацію компонентів дробово-раціонального модуля.

SUM2 має ініціалізуватися значенням 2^i–m+c (i–розрядність суматора), при цьому у квадраторі в SUM2 необхідно записати число c в прямому коді, оскільки при значенні x = 0, квадратичний поліном набуває значення y = c. А у дільнику чисел в Sum2 необхідно записати 2^i–m, а саме число m в додатковому коді.

Внаслідок підсумовування початкових значень компонентів, в SUM2 і виникає число 2^i–m+c.

SUM1 має ініціалізуватися значенням $a + b$, при чому необхідно урахувати знак числа b.

RG1 має ініціалізуватися значенням 2a, при цьому кожний вхідний імпульс потоку x, через групи елементів &1 буде переносити число, що відповідає значенню константи в ряду різниць другого порядку, тобто 2a, із RG1в SUM1.

RG2 має ініціалізуватися значенням $2^i - 2m$.

Регістр RG1 з'єднує вихід DE1 з групою елементів & 2, завдяки цьому при надходженні кожного імпульсу вхідного потоку x, в SUM1 записується число 2a.

Через групу елементів &2, числа, що надходять з SUM1 в SUM2 у прямому коді, переносяться в SUM2 із одиничним зсувом в бік старших

розрядів. Отже, під час надходження в обчислювач x імпульсів з SUM1 в SUM2 надійде значення $2(ax^2+bx+c)$.

Через групу елементів &3, кожен вхідний імпульс переносить значення 2^i-2m, де i – розрядність RG2, із RG 2 в SUM2.

Для забезпечення абсолютної похибки $|\delta_{max}|=0{,}5$ при діленні полінома на число m, необхідно в додатковому коді переносити подвоєне значення знаменника функції з RG2 в SUM2 в зворотному зв'язку, відповідно до (2.30).

Отже, при умові, що в початковому стані в SUM2 має значення $2^i-m+c,$ в процесі роботи обчислювача, на його виході буде формуватися шуканий результат знаходження значень підкореневого виразу, тобто $y_{int}=[\frac{ax^2+bx+c}{m}+0,5]$, у вигляді імпульсного потоку, що далі має надходити на вхід ПДК.

Розглянемо процес роботи дробово-раціональний модуля. З приходом першого імпульсу потоку x на вхід модуля, будуть здійснені наступні перетворення.

1.Через групу елементів &2 прямий бінарний код значення $a+b$, переноситься із SUM1 в SUM2 зі одиничним зсувом в бік старших розрядів і додається до його поточного вмісту. Внаслідок виконання данного перетворення, SUM2 прийме значення $(2^i-m+c)+2(a+b)=\Delta_1$. При умові, що значення, яке в прямому коді надходить в SUM2 більше від'ємного значення SUM2, на виході SUM2 буде сформовано імпульс переповнення і SUM2 збереже значення Δ_1.

2. Через групу елементів &1 прямий бінарний код значення 2a, переноситься із RG1в SUM1 і додається до його поточного вмісту. Внаслідок SUM1 набуде значення $a+b+2a$. Так в SUM1 утворюється значення першого члену арифметичного ряду різниць 1-го порядку. В подальшому,

при надходженні вхідних імпульсів потоку x, SUM1 буде набувати значень членів арифметичного ряду різниць 1-го порядку.

SUM2 сформує перший імпульс переповнення при надходженні на вхід пристрою імпульсу з номером x_1, що призведе до виконання першої нерівності з системи (2.30) $2\sum_{i=1}^{2} a_i x_1^i \geq m$.

Імпульс переповнення:

– надходить на вихід y з виходу SUM2;

– після проходження елементу затримки DE2 відкриває групу елементів &3. При цьому в SUM2 із регістру RG2 надходить значення 2^i–2m, і додається до поточного вмісту SUM2. Таким чином, SUM2 набуває значення $2^i - 2m + \Delta_1$, оскільки $\Delta_1 < 2m$ імпульс переповнення в SUM2 не буде сформовано.

В подальшому, при надходженні наступних вхідних імпульсів аргументу, вищеописані процеси будуть циклічно повторюватися.

SUM2 сформує другий імпульс переповнення при надходженні на вхід пристрою імпульсу з номером x_2, що призведе до виконання другої нерівності з системи (2.30) $2(\sum_{i=1}^{2} a_i x_2^i - \sum_{i=1}^{2} a_i x_1^i) + \Delta_1 \geq 2m$, тощо.

Таким чином, під час роботи обчислювача на виході SUM2 буде формовано шуканий результат обчислення функції $y_{int} = \frac{ax^2 + bx + c}{m} + 0{,}5$.

3.2.2 Пристрій для добування кореня

Розглянемо блок ПДК (рисунок 3.1). На вхід ПДК надходить імпульсний потік $x = y_{int}$, що є вихідним імпульсним потоком блоку Дробово-раціональний модуль, у якому $y_{int} := f(x)$. Результат обчислення у процесі роботи блоку ПДК представляється у формі імпульсного потоку на виході CNTsqrt, а в CNTres у формі паралельного коду.

В ПДК через групу елементів &4 переноситься значення CNTres в додатковому коді із одиничним зсувом в бік старших розрядів, в підсумовуючий лічильник CNTsqrt. Отже, CNTsqrt набуває значення $2^i - 2N$ в додатковому коді, де N – це число, що містить лічильник CNTres.

Приведемо ініціалізацію компонентів:

CNTsqrt набуває значення $2^i - 1$,

CNTres набуває значення 0.

Детально розглянемо роботу модуля.

З приходом 1-го імпульсу вхідного потоку ПДК на вхід лічильника CNTsqrt, він формує вихідний сигнал переповнення та приймає значення 0.

Даний сигнал переповнення протрапляє на вихід пристрою в канал у, також надходить на вхід CNTres (при цьому CNTres приймає значення 1), і далі, пройшовши елемент затримки DE3, подається на групу елементів &4, відкриває групу елементів &4 і в CNTsqrt записує значення $2^i - 2$, тобто додатковий код бінарного числа CNTres зі зсувом на один розряд у бік старших розрядів.

З приходом на вхід ПДК чергових двох імпульсів потоку x, CNTsqrt знову формує сигнал переповнення і приймає значення 0.

Як і в попередньому кроці, другий сигнал переповнення CNTsqrt потрапляє на вихід пристрою в канал у, формує в CNTres значення 2 і пройшовши елемент затримки DE3, через групу елементів &4 в CNTsqrt записує значення $2^i - 4$.

В подальшому описані процеси будуть циклічно повторюватися з приходом чергових імпульсів потоку x.

У CNTsqrt послідовно надходять значення ряду різниць першого порядку в додатковому коді, а на вихід у, що є виходом всього біт-потокового обчислювача дробово–ірраціональних функцій – імпульсні сигнали з номерами x_y вхідного імпульсного потоку x обчислювача.

З приходом в ПДК останнього вхідного імпульсу, лічильник результату CNTres набуде значення результату обчислення досліджуваної функції $y = [\sqrt{[\frac{ax^2 + bx + c}{m} + 0{,}5]} + 0,5]$.

Заміна в ПДК лічильника CNTsqrt паралельним суматором, дозволить знаходити корінь квадратний і з чисел у вигляді паралельних бінарних кодів.

3.3 Вибір елементної бази

При виборі САПР, необхідно спиратися на досить багато факторів, які вплинуть на кінцевий результат.

Обрана САПР має бути досить поширеною і відносно недорогою, при цьому, вона має підтримувати необхідну елементну базу; охоплювати і ефективно виконувати якомога більше етапів проектування; мати широку бібліотечну підтримку стандартизованих рішень; а також просто і зручно стикуватися з іншими САПР. До того ж досить важливий критерій – це зручність роботи, та дружність САПР, легкість її вивчення, та, при необхідності, можливість здійснення корпоративної взаємодії з обраною системою.

Спираючись на актуальні тенденції розвитку архітектур програмованої логіки пакети CPLD і FPGA є поширеною основою елементної бази цифрових систем. Особливими перевагами є використання програмованої логіки в цифрових системах в задачах швидкї розробки дослідних зразків виробів, при можливих частих коригуваннях проекту під час розробки або при необхідності реконфігурування структури пристрів в процесі функціонування.

У сьогоденні обсяг ПЛІС складає понад 1млн. вентилів типу 2 I-HI. Сучасні ПЛІС – це ISP прилади, тобто прилади, програмовані безпосередньо в системі. Час їх затримки не перевищує до 0,5нс, а системні частоти складають 200 МГц.

Перші FPGA (Field Programmable Gate Arrays), тобто програмовані користувачем вентильні матриці, були розроблені фірмою Xilinx в 1985р. В FPGA елементи налаштування складають програмовані мультиплексори. Кожного разу, перед початком роботи з FPGA, необхідно проводити її налаштування на потрібний режим функціонування. Необхідна програма налаштування заздалегідь записується в ПЗП (ОЗП). Відразу після подання живлення, завантажується інформація з ПЗП і FPGA автоматично ініціалізується (вона містить необхідні для цього логічні схеми).

Для реалізації обчислювача було обрано FPGA кристал сімейства Spartan-3E серії XC3S500E фірми Xilinx. Його технічні характеристики наочно представлені в таблиці 3.1.

Таблиця 3.1 – Параметри кристала

Device	Logic Cells	System Gates (Logic and RAM)	CLB Array (C*R)	Total CLBs	Maximum Available User I/O	Total Distributed RAM Bits	Total Block RAM Bits
XC3S500E	2160	100000	16*22	240	108	15K	360K

Програмована користувачем вентильна матриця Spartan-3E охоплює конфігуровані логічні блоки (configurable logic blocks - CLBs) і блоки введення – виведення (IOBs). CLB блоки служать для створення функціональних логічних елементів, а блоки I/O створюють інтерфейс між контактами мікросхеми і CLB блоками. Логічна комірка є базовим будівельним елементом CLB блоку. Вона включає чотиривходовий функціональний генератор, логіку прискореного перенесення і запам'ятовуючий елемент. Вихід кожного функціонального генератора в кожній логічній комірці приєднаний до вихідної лінії CLB-блоку і до D-входу тригера. Кожен CLB-блок в серії Spartan-3E містить чотири логічних комірки, організовані у вигляді двох однакових секторів (Slice). На додаток

до чотирьох базових логічних комірок, CLB-блок серії Spartan-3E містить логіку, яка дозволяє комбінувати ресурси функціональних генераторів для реалізації функцій від п'яти або шести входів. Функціональні генератори реалізовані у вигляді 4-х входових функціональних таблиць (LUT). Крім використання в якості функціональних генераторів, кожний LUT-елемент може бути також використаний як синхронна пам'ять типу RAM розмірністю 16x1 біт.

4 АПАРАТНА РЕАЛІЗАЦІЯ СПРОЕКТОВАНОЇ МОДЕЛІ ОБЧИСЛЮВАЧА

В результаті розробки та аналізу структурно-функціональної моделі біт-потокового обчислювача, необхідно здійснити опис проекту для введення в САПР. У даному розділі досліджено обчислювальний процес в компонентах пристрою, розроблено граф-схему алгоритму обчислювача заданої функції, на підставі якої отримано граф переходів автомата Мура, розроблена апаратна модель на мові VHDL, виконана верифікація отриманого рішення за допомогою тест-бенч та імплементація моделі в ПЛІС Xilinx Spartan 3E.

4.1 Специфікація досліджуваного обчислювача

Для автоматизації досліджуваної моделі була створена експериментальна апаратна реалізація спеціалізованого обчислювача дробово-ірраціональної функції з імпульсними потоками даних

$$y = [\sqrt{[\frac{\sum_{i=0}^{2} a_i x^i}{m} + 0{,}5]} + 0{,}5]. \quad (4.1)$$

Вираз (4.1) перепишемо, як

$$y = [\sqrt{[\frac{ax^2 + bx + c}{m} + 0{,}5]} + 0{,}5]. \quad (4.2)$$

Для наочної демонстрації прикладу роботи досліджуваної моделі можна перевірити її працездатність.

Оберемо значення цілочисельних коефіцієнтів a, b, c, m підкореневого

виразу функції y та x_{max} вхідного імпульсного потоку

a=2, b =3, c=1, m=10, x_{max} = 10.

Лістинг інтерфейсу обчислювача наведено в 4.1.

Лістинг 4.1 – Опис інтерфейсу обчислювача

```
entity aprox_toplevel is
      generic(
      width: natural := 8;
            m: integer := 10);
      port(
      reset_i: in std_logic;
      clock_i: in std_logic;
      x_i: in std_logic;
      y_o: out std_logic;
      count: out std_logic_vector(width-1 downto 0));
end aprox_toplevel;
```

4.2 Обчислювальний процес в компонентах досліджуваного пристрою

Детально проілюструємо процес обчислення досліджуваної функції з певними заданими параметрами в компонентах пристрою.

З урахуванням обраних значень цілочисельних компонентів, функція набуває вигляду $y = [\sqrt{[\frac{2x^2+3x+1}{10}+0{,}5]}+0{,}5]$, при чому максимальна кількість імпульсів, що потрапляє на вхід обчислювача x_{max}=10.

При проходженні вхідного імпульсного потоку x_{max}=10, функція прийматиме наступні значення з урахуванням округлення результату значення функції до найближчого цілого числа

x=1: $y=[\sqrt{[\frac{2\cdot 1^2+3\cdot 1+1}{10}+0{,}5]}+0{,}5]=[1{,}5]=1;$

x=2: $y=[\sqrt{[\frac{2\cdot 2^2+3\cdot 2+1}{10}+0{,}5]}+0{,}5]=[1{,}9]=1;$

x=3: $y=[\sqrt{[\frac{2\cdot 3^2+3\cdot 3+1}{10}+0{,}5]}+0{,}5]=[2{,}2]=2;$

x=4: $y=[\sqrt{[\frac{2\cdot 4^2+3\cdot 4+1}{10}+0{,}5]}+0{,}5]=[2{,}7]=2;$

x=5: $y=[\sqrt{[\frac{2\cdot 5^2+3\cdot 5+1}{10}+0{,}5]}+0{,}5]=[3{,}1]=3;$

x=6: $y=[\sqrt{[\frac{2\cdot 6^2+3\cdot 6+1}{10}+0{,}5]}+0{,}5]=[3{,}5]=3;$

x=7: $y=[\sqrt{[\frac{2\cdot 7^2+3\cdot 7+1}{10}+0{,}5]}+0{,}5]=[3{,}9]=3;$

x=8: $y=[\sqrt{[\frac{2\cdot 8^2+3\cdot 8+1}{10}+0{,}5]}+0{,}5]=[4{,}3]=4;$

x=9: $y=[\sqrt{[\frac{2\cdot 9^2+3\cdot 9+1}{10}+0{,}5]}+0{,}5]=[4{,}8]=4;$

x=10: $y=[\sqrt{[\frac{2\cdot 10^2+3\cdot 10+1}{10}+0{,}5]}+0{,}5]=[5{,}2]=5.$

Згідно з алгоритмом, необхідно у вираз $y = 2x^2 + 3x + 1$ підставити значення x = 1, 2, 3, …, 10, для отримання арифметичного ряду другого порядку, що містить послідовність значень функції y

y: 6, 15, 28, 45, 66, 91, 120, 153, 190, 231, ...

Обчислимо для цієї послідовності арифметичні ряди різниць 1-го і 2-го порядків. Вони приймають вигляд:

Δ: 9, 13, 17, 21, 25, 29, 33, 37, 41, ...

Δ^2: 4, 4, 4, 4, 4, 4, 4, 4, ...

Знайдемо числа, що ініціалізують компоненти обчислювача.

SUM1 = 5, SUM2 = -9, RG1 = 4, RG2 = -20, CNT_{sqrt} = -1, CNT_{res} = 0.

Розглянемо приклад обчислення в моделі досліджуваного спеціалізованого обчислювача.

В таблицях 4.1 – 4.10, покроково наведено деталі обчислювального процесу, що відбуваються в компонентах досліджуваного обчислювача дробово-ірраціональних функцій, при подачі на вхід бітового потоку довжиною 10 імпульсів (x_{max}=10).

Таблиця 4.1 – Обчислювальний процес для x=1

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-9 + 10 = 1	1	5+4=9	-1 + 1 = 0	1	1
1 – 20 = -19			0 – 2 = -2		

Таблиця 4.2 – Обчислювальний процес для x=2

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-19 + 18 = -1		9+4=13			

Таблиця 4.3 – Обчислювальний процес для x=3

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-1 + 26 = 25	1	13+4=17	-2 + 1 = -1		
25 – 20 = 5	1		-1 + 1 = 0	1	2
5 – 20 = -15			0 – 4 = -4		

Таблиця 4.4 – Обчислювальний процес для x=4

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-15 + 34 = 19	1	17+4=21	-4 + 1 = -3		
19 – 20 = -1					

Таблиця 4.5 – Обчислювальний процес для x=5

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-1 + 42 = 41	1	21+4=25	-3 + 1 = -2		
41 – 20 = 21	1		-2 +1 = -1		
21 - 20 = 1	1		-1 + 1 = 0	1	3
1- 20 = -19			0 – 6 = -6		

Таблиця 4.6 – Обчислювальний процес для x=6

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-19 + 50 = 31	1	25+4=29	-6 + 1 = -5		
31 – 20 = 11	1		-5 + 1 = -4		
11 – 20 = -9					

Таблиця 4.7 – Обчислювальний процес для x=7

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-9 + 58 = 49	1	29+4=33	-4 + 1 = -3		
49 – 20 = 29	1		-3 + 1 = -2		
29 – 20 = 9	1		-2 + 1 = -1		
9 – 20 = -11					

Таблиця 4.8 – Обчислювальний процес для x=8

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-11 + 66 = 55	1	33+4=37	-1 + 1 = 0	1	4
55 – 20 = 35	1		0 – 8 = -8		
35 – 20 = 15	1		-8 + 1 = -7		
15 – 20 = -5			-7 + 1 = -6		

Таблиця 4.9 – Обчислювальний процес для x=9

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-5 + 74 = 69	1	37+4=41	-6 + 1 = -5		
69 – 20 = 49	1		-5 + 1 = -4		
49 – 20 = 29	1		-4 + 1 = -3		
29 – 20 = 9	1		-3 + 1 = -2		
9 – 20 = -11					

Таблиця 4.10 – Обчислювальний процес для x=10

SUM2	Переповнення SUM2	SUM1	CNTsqrt	Переповнення CNTsqrt	CNTres
-11 + 82 = 93	1	41+4=45	-2 + 1 = -1		
93 – 20 = 73	1		-1 + 1 = 0	1	5
73 – 20 = 53	1		0 –10 = -10		
53 – 20 = 33	1		-10 + 1 = -9		
33 – 20 = 13	1		-9 + 1 = -8		
13 – 20 = -7			-8 + 1 = -7		

4.3 Автоматна модель спеціалізованого обчислювача

Автоматна модель досліджуваного спеціалізованого обчислювача представлена композицією управляючого і операційного автоматів.

Досліджувана модель працює за наступним алгоритмом:

1. При поданні сигналу скидання (reset = 1) регістри обчислювача набувають наступних значень:

SUM1 = 5,

SUM2 = -9,

CNTsqrt = -1,

CNTres=0,

RG1=4,

RG2=-20.

2. При надходженні кожного імпульсу:

– регістр SUM2 збільшує своє значення на подвоєне значення регістра SUM1;

– регістр SUM1 збільшує своє значення на 4.

3. Якщо регістр SUM2 набуває невід'ємного значення, то:

– на його виході генерується вихідний імпульс;

– значення CNTsqrt збільшується на 1;

– значення регістра SUM2 зменшується значення 2m.

4. Якщо регістр CNTsqrt приймає невід'ємне значення:

– збільшується значення регістра CNTres на 1;

– значення регістра CNTsqrt зменшується на подвійне значення регістра CNTres.

5. Третій та четвертий пункти повторюються доки регістр SUM2 або регістр CNTsqrt мають невід'ємне значення.

Представлена на рисунку 4.1 змістовна ГСА, роботи досліджуваного обчислювача була розроблена, спираючись на данні його структурно-функціональної моделі та її математичного опису.

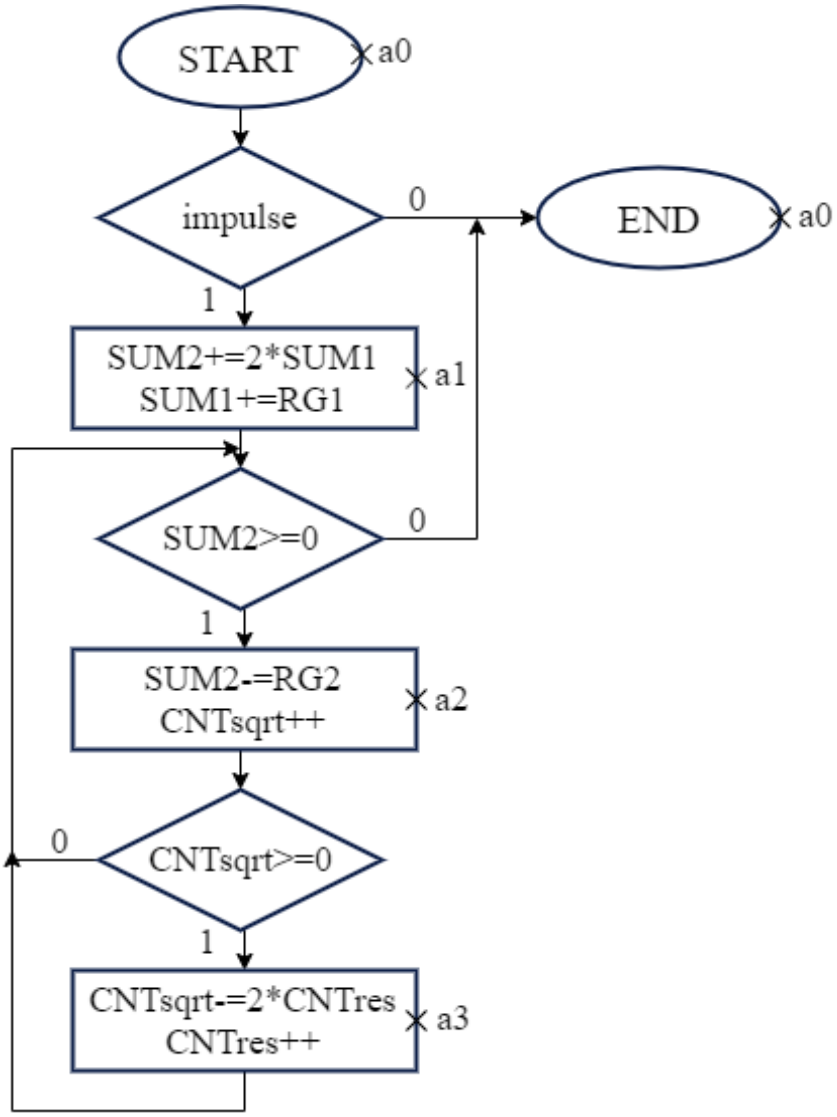

Рисунок 4.1 – Граф-схема алгоритму роботи обчислювача

На рисунку 4.2 представлений граф алгоритму у вигляді кінцевого автомату Мура, що відображає принцип роботи арифметичного блоку спеціалізованого обчислювача. Він спирається на принципи роботи, описані в попередньому розділі при розгляданні структурно-функціональної схеми (рисунок 3.1) досліджуваного обчислювача.

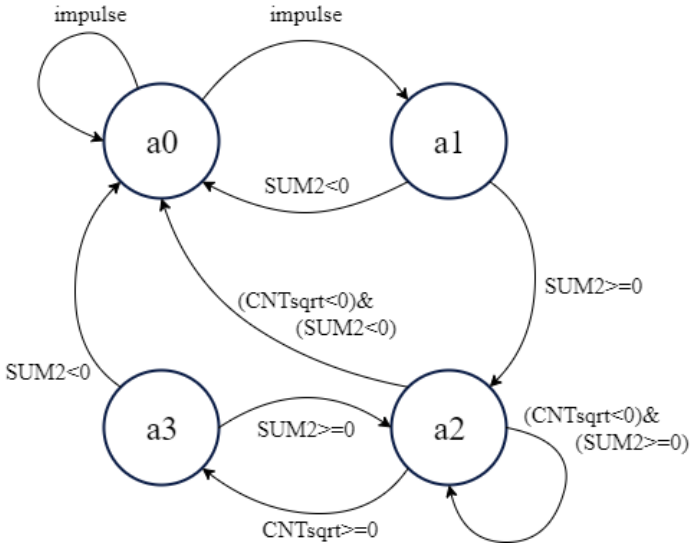

Рисунок 4.2 – Граф переходів автомата Мура арифметичного блоку

Управляючий і операційний автомат у композиції представляють арифметичний блок досліджуваного спеціалізованого обчислювача. Управляючий автомат описано отриманим в результаті аналізу ГСА для автомата Мура графом переходів. Він може приймати 4 стани: a0, a1, a2, a3.

Після подання імпульсу скидання reset = 1 даний автомат переходить в стан a0, тобто свій початковий стан. Він буде зберігати даний стан, доки із вхідного буфера не надійде сигнал impulse. Після цього автомат змінить свій стан на a1.

Стан a1 означає, що доки значення регістру SUM2 буде від'ємним, автомат буде додавати до суми SUM2 подвійне значення регістру SUM1 зі зсувом на одиницю в сторону старших розрядів, а також збільшувати значення регістру SUM1 на 4, оскільки, для обраних параметрів, значення RG1 = 4. Якщо регістр SUM2 набуде невід'ємного значення, автомат перейде у стан a2, інакше він перейде в початковий стан.

У стані a2 автомату, до значення регістру SUM2 додається значення регістру RG2 в додатковому коді, окрім того від значення регістру SUM2 віднімається значення регістру RG2, а підсумовуючий лічильник CNTsqrt збільшується на 1. В нашому випадку регістр RG2 приймає значення -20. Також, в стані a2 генерується сигнал для блоку отримання квадратного кореня. Коли регістр CNTsqrt набуває невід'ємного значення, автомат

переходить в стан a3, коли регістри CNTsqrt і SUM2 мають від’ємні значення, автомат повертається в стан a0.

В стані a3 до результату CNTres додається одиниця, а також віднімається подвоєне значення блоку отримання квадратного кореня, що надходить з лічильника результату CNTres в лічильник CNTsqrt. Окрім того, автомат видає вихідному буферу сигнал для формування імпульсу на виході обчислювача. Коли значення регістра SUM2 стає невід'ємним, автомат переходить в стан a2, інакше він набуває стану a0.

Обрано пакет програмного забезпечення Active-HDL 6.1, для моделювання і верифікації моделі досліджуваного обчислювача. Даний пакет має широкі можливості автоматизації процесу проектування, передбачає створення TestBench, крім того сприяє здійсненню більш ефективного аналізу проектних рішень. Програмний продукт Synplicity Synplify 7.0.2 було обрано для синтезу проекту, оскільки, порівняно з альтернативним програмним забезпеченням даного призначення він відзначається більш високою швидкістю синтезу, наочним, у вигляді нетліста, представленням результатів синтезу, а також він в включає бібліотеки під елементну базу світових виробників сучасності. Незважаючи на необхідність знання специфіки використовуваного мовного опису на МОА, а також порівняно більших часових витрат, було обрано саме цей мовний опис для ведення проекту, оскільки він є найбільш гнучким варіантом.

4.4 Структурно-блокова схема досліджуваного обчислювача

Структурно-блокова схема, зображена на рисунку 4.3, включає три блоки:

– вхідний буфер;

– спеціалізований обчислювач;

– вихідний буфер.

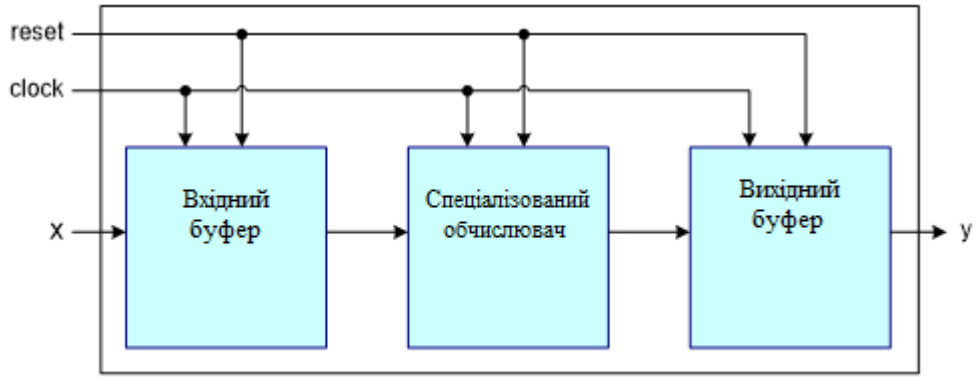

Рисунок 4.3 – Структурно-блокова схема обчислювача

Призначення блоків:

1. Вхідний буфер. Призначення блоку полягає у детектуванні імпульсного потоку, що приходить на вхід пристрою. Вхідний сигнал детектується за двома сусідніми тактами сигналу clock. Коли, на черговому такті, сигнал x набуває нульового значення, а на наступному змінює своє значення логічного нуля на логічну одиницю, на виході вхідного буферу з'являється імпульс. Він встановлює сигнал impulse =1 на виході. Саме цей сигнал отримає обчислювач для подальшої обробки.

На рисунку 4.4 можна побачити, що для виявлення вхідного імпульсу, період сигналу clock t_{clk} має бути в два рази меншим за період вхідного імпульсу(t_1). Коли t_1 менше періоду синхронізації, вхідний буфер не зможе задетектувати імпульси, для яких $t_1 < t_{clk}$.

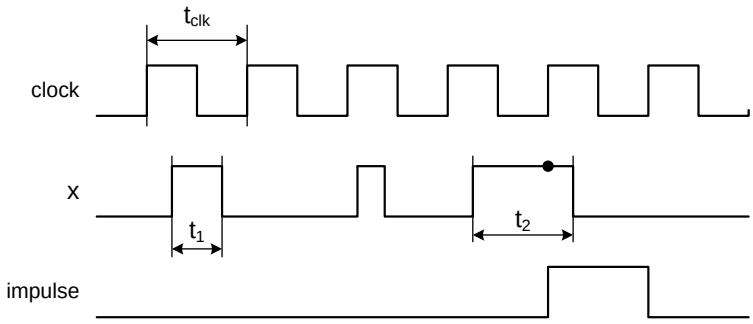

Рисунок 4.4 – Часова схема вхідного буфера

Наведемо вхідні и вихідні параметри для даного блоку в таблиці 4.11, а також лістинг 4.2, у якому відображено інтерфейс вхідного буферу.

Таблиця 4.11 – Вхідні і вихідні параметри вхідного буфера

Data	вхідний імпульсний поток	IN
Reset	скидання обчислювача в початковий стан	IN
Clock	сигнал синхронізації	IN
DataOut	вихідний імпульсний поток	OUT
Ready	сигнал готовності з арифметичного блоку	IN

Лістинг 4.2 – Інтерфейс блоку «Вхідний буфер»

```
entity aprox_inputbuffer is
      generic(
      width: natural := 8);
      port(
      x_i: in std_logic;
      clock_i: in std_logic;
      reset_i: in std_logic;
      ready_i: in std_logic;
      y_o: out std_logic);
end aprox_inputbuffer;
```

1. Спеціалізований обчислювач. Згідно алгоритму він включає два блоки:

– апроксиматор;

– ПДК.

Апроксиматор складається з квадратора і дільника чисел. Він призначений, відповідно, для відтворення квадратичного полінома $2\hat{x}^2+3x+1$, а також ділення обчисленого результату на m=10 з урахуванням похибки $|\delta_{1max}|=0{,}5$.

Наведемо вхідні и вихідні параметри для даного блоку в таблиці 4.12, а також лістинг 4.3, у якому відображено його інтерфейс.

Таблиця 4.3 – Вхідні і вихідні параметри для даного блоку

Data	вхідний імпульсний потік	IN
Reset	скидання обчислювача в початковий стан	IN
Clock	сигнал синхронізації	IN
DataOut	вихідний імпульсний потік, як результат відтворення функції $y_{int} = [\frac{ax^2 + bx + c}{m} + 0{,}5]$	OUT
Count	значення функції в двійковому N-розрядному коді на виході суматора	OUT

Лістинг 4.3 – Опис інтерфейсу блоку «Апроксиматор»

```
entity aproximator is
      generic(
      width: natural := 8;
      m: integer := 10
      );
      port(
      x_i: in std_logic;
      ready_o: out std_logic;
      clock_i: in std_logic;
      reset_i: in std_logic;
      y_o: out std_logic;
      sum_o: out std_logic_vector(width-1 downto 0));
end entity aproximator;
```

ПДК. Блок пристрою для добування квадратного кореня складається з підсумовуючого лічильника, а також лічильника результату. Він призначений для добування квадратного кореня з урахуванням похибки $|\delta_{2max}| = 0{,}5$. Отриманий результат буде округлений до найближчих цілих чисел.

Наведемо вхідні и вихідні параметри для даного блоку в таблиці 4.13, а також лістинг 4.4, у якому відображено його інтерфейс.

Таблиця 4.13 – Вхідні і вихідні параметри блоку ПДК

Data	вхідний імпульсний потік з попереднього блоку	IN
Clock	сигнал синхронізації	IN
Reset	скидання обчислювача в початковий стан	IN
DataOut	вихідний імпульсний потік, як результат відтворення функції $y = [\sqrt{[\frac{ax^2 + bx + c}{m} + 0{,}5]} + 0, 5]$	OUT
Count	значення функції в бінарному N-розрядному коді на виході суматора	OUT

Лістинг 4.4 – Опис інтерфейсу блоку «ПДК»

```
entity sqrt_toplevel is
      generic(
      width: natural := 8);
      port(
      reset_i: in std_logic;
      clock_i: in std_logic;
      x_i: in std_logic;
      y_o: out std_logic;
      count: out std_logic_vector(width-1 downto 0));
end sqrt_toplevel;
```

4. Вихідний буфер. Його призначення полягає у формуванні вихідного імпульсного потоку. Блок подає сигнал готовності обчислювача приймати для обробки наступний вхідний сигнал.

Після надходження сигналу Ready=1 на виході даного блоку, він генерує вищеозначений сигнал готовності. Результат його роботи – це сигнал DataOut, який представляє собою результат обчислення досліджуваної функції у вигляді імпульсного потоку.

Наведемо вхідні параметри для даного блоку в таблиці 4.14.

Таблиця 4.14 – Вхідні параметри для блоку «Вихідний буфер»

Enable	вхідний імпульсний потік з виходу ПДК	IN
Reset	скидання обчислювача в початковий стан	IN
Ready	сигнал готовності обчислювача приймати наступний імпульс	OUT
Clock	сигнал синхронізації	IN
DataOut	вихідний імпульсний потік, як результат відтворення функції $y = [\sqrt{[\frac{ax^2 + bx + c}{m} + 0{,}5]} + 0,5]$	OUT

4.4.1 Опис блоків проекту на МОА

Обрання VHDL, як МОА для опису зумовлено, його високоякісною адаптованістю відносно запитів проектувальника, а також присутністю зручного середовища моделювання (Active-HDL) і середовища синтезу, що оперує VHDL (Synplify).

Для спрощення опису проекту на VHDL представимо багатокомпонентний проект як ієрархічну структуру, збільшуючи його

деталізацію на нижчих рівнях структури. Дана ієрархія допоможе спростити написання VHDL-моделей незалежних компонентів, їх корекцію та налагодження.

Файлова ієрархія даного проекту була сформована в результаті аналізу структурно-функціональної схеми. Вона представлена на рисунку 4.5.

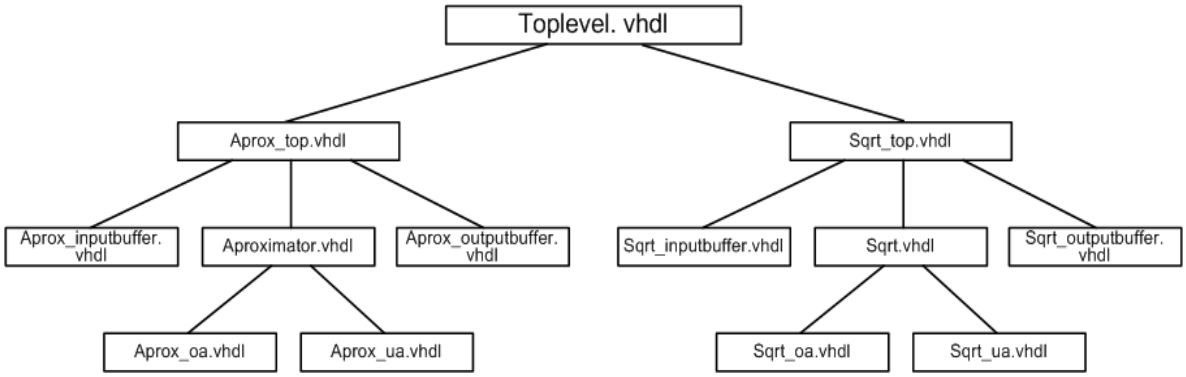

Рисунок 4.5 – Файлова ієрархія проекту

4.4.2 Опис проекту на МОА

Файл Toplevel.vhdl обіймає верхню ланку вищенаведеної ієрархії проекту. Він об’єднує в собі усі файли попередніх рівнів. Файл Toplevel.vhdl містить компоненти, які об'єднуються через оператори port map, які пов’язують ланки файлів нижньої ієрархії. Цей файл – реалізація на МОА специфікації обчислювача цілком. В entity описані всі вхідні і вихідні сигнали даного проекту (лістинг 4.5).

Лістинг 4.5 – Опис файлу «Toplevel»

```
entity toplevel is
      generic(
      width: natural := 8;
      m: natural := 10);
      port(
      reset_i: in std_logic;
      clock_i: in std_logic;
      x_i: in std_logic;
      y_o: out std_logic;
      count: out std_logic_vector(width-1 downto 0));
end toplevel;
```

Компоненти "aprox_toplevel(struct)" та "sqrt_toplevel(struct)" займають у вищенаведеній ієрархії другий рівень. Наведемо лістинг сполучення цих блоків (лістинг 4.6).

Лістинг 4.6 – Опис сполучених блоків "aprox_toplevel(struct)" і "sqrt_toplevel(struct)"

```
APROX_1: aprox_toplevel
      generic map (
      width => width,
      m => m)
      port map (
            reset_i => reset_aprox,
            clock_i => clock_slow,
            x_i => x_i,
            y_o => y_internal,
            count => fake_count);


            SQRT_1: sqrt_toplevel
            generic map(
            width => width)
            port map(
            reset_i => reset_i,
            clock_i => clock_i,
            x_i => y_internal,
            y_o => y_o,
            count => count);
end struct;
```

У лістингу 4.7 представлено фрагмент програми, яким описується робота операційного автомата апроксиматора.

Лістинг 4.7 – Операційний автомат апроксиматора.

```
process (clock_i, reset_i)
      begin
            if (reset_i = '1') then
                  sum <= CONV_STD_LOGIC_VECTOR(6 - m, width);
                  counter <= CONV_STD_LOGIC_VECTOR(5, width);
            else
                  if (falling_edge(clock_i)) then
                        if (sum_plus_a_i = '1') then
                              counter <= counter + 4;
                              sum <= sum + counter + counter;
                        else
                              if (sum_minus_b_i = '1') then
                                    sum <= sum - 2*m;
                              end if;
                        end if;
```

```
                    end if;
               end if;
          end process;

     end beh;
```

Фрагмент програми, що реалізує роботу операційного автомата пристрою ПДК для добування кореню представлено у лістингу 4.8.

Лістинг 4.8 – Операційний автомат ПДК

```
    process (clock_i, reset_i)
     begin
          if (reset_i = '1') then
               sum <= CONV_STD_LOGIC_VECTOR(-1, width);
               counter <= (others => '0');
          else
               if (falling_edge(clock_i)) then
                    if (sum_plus_xk_i = '1') then
                         sum <= sum + ('0' & xk);
                    else
                         if (sum_minus_counter_i = '1') then
                              sum <= sum - counter - counter - 2;
                              counter <= counter + 1;
                         end if;
                    end if;
               end if;
          end if;
     end process;
```

Програма-обгортка Wrapper.verilog написана мовою Verilog. Вона слугує для інтеграції з індикацією на платі. Дана програма представлена у лістингу 4.9

Лістинг 4.9 – Програма-обгортка Wrapper.verilog

```
     Wrapper.verilog
     module wrapper
             (input rst, St, clk, sel,
              output [7:0] LED);

          reg s1, s2, s3, s4;
          reg [17:0] count;
          wire y_o;

          toplevel    UUT  (.reset_i(rst),  .x_i(St_in),  .count(LED),
.clock_i(clk), .y_o(y_o) );
          //----   no bounce ---
          always @(posedge clk)
          if (count == 0)
```

```
begin
      s1<=St;
      s2<=s1;
      s3<=s2;
end

assign St_in = s1 & s2 & ~s3;

always @ (posedge clk)
      if (rst)
            count = 0;
      else
            count = count + 1;

endmodule
```

4.5 Верифікація та тестування роботи пристрою

Для верифікації даної апаратної реалізації, мовою VHDL було розроблено тестове оточення (test bench). Тест-бенч – це модуль на мові VHDL, для якого відсутній інтерфейс, проте він містить примірник тестового модуля, подає тестові впливи і аналізує реакції модуля, що тестується. На рисунку 4.6 продемонстровано взаємодію тест-бенча та тестованого модуля, а також їх ієрархічне відношення.

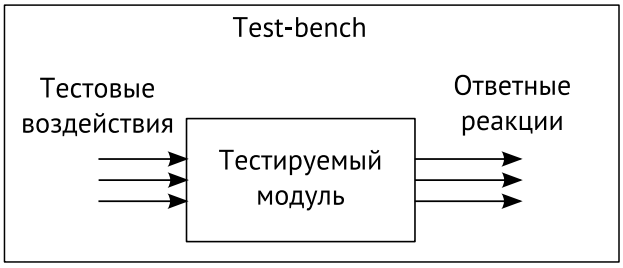

Рисунок 4.6 – Відношення тест-бенча і тестованого модуля

При цьому опис структури тестової програми на мові VHDL здійснюється так само, як самостійної цифрової системи.

На рисунку 4.7 наведені результати моделювання поведінкової моделі досліджуваного обчислювача. На ньому наочно продемонстровано, що результати тестування збігаються з результатами, що були отримані внаслідок здійснення попередніх обчислювальних процесів в компонентах

пристрою. Це слугує підтвердженням правильності роботи досліджуваного обчислювача.

Також, на рисунку 4.7 представлена часова діаграма, що детальніше розкриває попередньо виконану верифікацію режимів роботи досліджуваного обчислювача.

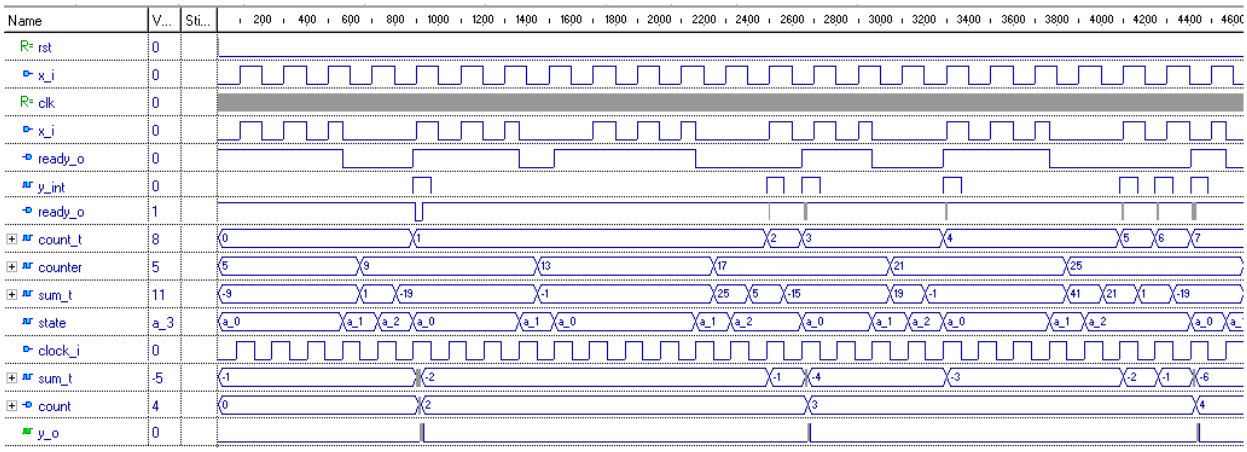

Рисунок 4.7 – Поведінкова модель досліджуваного обчислювача

Після перевірки режимів верифікації обчислювача необхідно перейти до наступного етапу проектування, а саме – імплементації моделі.

4.6 Імплементація моделі обчислювача дробово-ірраціональної функції

Для дослідження було використано кристал сімейства Xilinx Spartan 3E. Вироблений наприкінці 2005 року фірмою Xilinx, у рамках серійного виробництва сімейства ПЛІС з архітектурою FPGA, Spartan 3E поєднує в собі такі переваги, як високі технічні характеристики і відносно невисоку вартість. Після його створення було розпочато виготовлення різноцільових інструментальних моделей, тобто програмних засобів, які призначені для проектування, налагодження, синтезу пристроїв, що базуються на кристалах даного сімейства. Використання цих інструментальних засобів дозволяє значно зменшити загальний час, якого

потребує розробка досліджуваного обчислювача, а також уникнути великої частки можливих помилок, що неминуче виникають при виробництві друкованої плати і монтажі компонентів. Більше того, подібні інструментальні модулі полегшують дослідження нових методів, та програмних засобів розробки систем на базі ПЛІС, зокрема, систем що базуються на конфігурованих мікропроцесорних ядрах. Для дослідження процесів наскрізного проектування вбудованих мікропроцесорних систем, що базуються на ядрах сімейств PicoBlaze і MicroBlaze, найбільший потенціал має виконаний на основі ПЛІС XC3S500E інструментальний комплект Spartan 3E Starter Kit.

Крім того, фірмою Xilinx регулярно надається сучасне програмне забезпечення, для розробки і конфігурації кристалів. Так, повний перехід до нового покоління систем автоматизованого проектування – Integrated Synthesis Environment (ISE) було завершено на початку 2002 року. До цього моменту, їх застосовували як альтернативу попередній версії САПР Foundation Series.

Integrated Synthesis Environment допомагає суттєво зменшити час розробки, а також підвищити ефективність її результатів, оскільки він оперує більльш досконалими методами проектування, алгоритмами синтезу, розміщення та трасування проекту.

Суттєва відмінність між чотирма конфігураціями (Foundation ISE, BaseX ISE, Alliance ISE і WebPACK ISE) у яких випускаються засоби ISE, полягає в кількості кристалів, що підтримуються, а також переліку додаткових інструментів проектування.

В реалізації досліджуваного обчислювача ми скористаємося засобами, що надає модифікація WebPACK ISE, адже вона – єдина вільно розповсюджувана серед конфігурацій.

Програмні засоби WebPACK ISE – це система наскрізного проектування, що реалізує повний цикл розробки цифрових пристроїв на базі ПЛІС, включаючи всі етапи створення вихідних описів проекту,

синтезу, моделювання, розміщення і трасування, а також програмування кристала. До складу пакету входить модуль iMPACT, який призначений для конфігурації майже всіх кристалів, розроблених фірмою Xilinx.

Для синтезу проекту використано Synplify 7.0 Pro, розроблений фірмою Synplicity. Внаслідок синтезу було отримано RTL схему вентильного рівня, схему Technology View, а також файл-звіт.

Лістинг 4.10 містить попередньо створений для імплементації проекту UCF-файл.

Лістинг 4.10 – UCF- файл проекту

```
NET "rst" LOC = "H13" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "pack" LOC = "V4" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "St" LOC = "K17" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "LED<7>" LOC = "F9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<6>" LOC = "E9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;
```

Рисунок 4.8 демонструє RTL-схему, а рисунок 4.9 RTL схему нижнього рівня досліджуваного обчислювача.

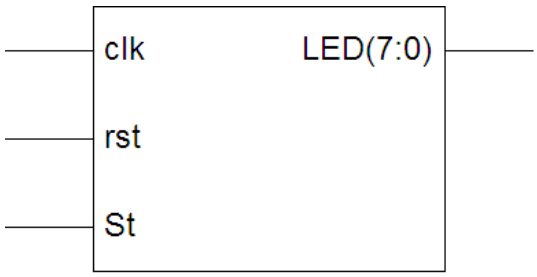

Рисунок 4.8 – RTL схема

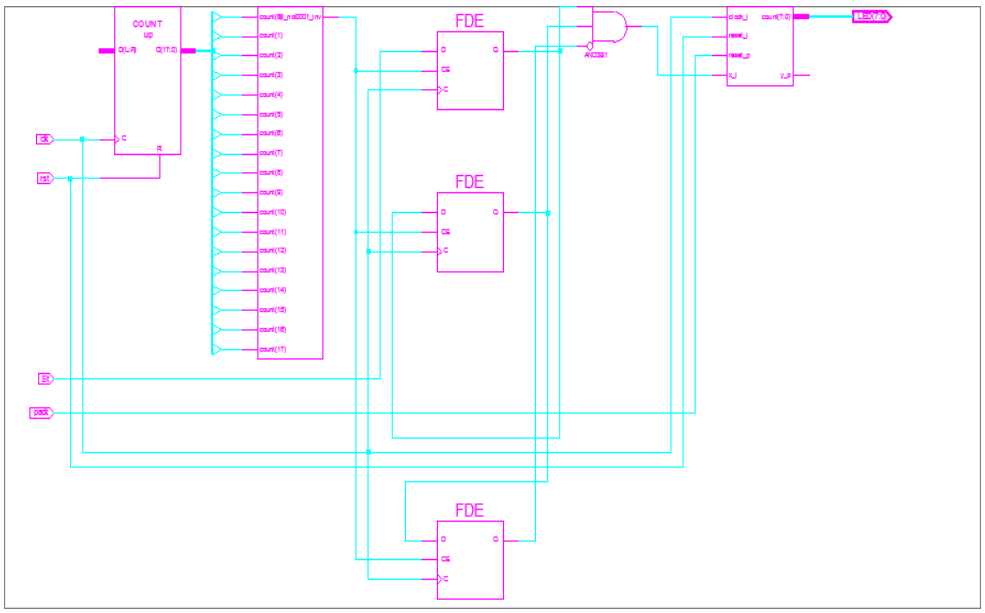

Рисунок 4.9 – RTL схема нижнього рівня

На У звіті на лістигну 4.11 наведено файл-звіт. На ньому наочно продемонстровано, що для синтезу досліджуваного обчислювача з використанням Xilinx SPARTAN 3E серії XC3S100E було задіяно приблизно 6% ресурсів, пов'язаних з логічними елементами. Отже, досліджуваний обчислювач можна назвати зберігаючим щодо використаних ресурсів, адже використана мікросхема могла б задіяти до 15 аналогічних пристроїв.

Лістинг 4.11 – Звіт про використання ресурсів кристала

```
Logic Utilization:
 Number of Slice Flip Flops:       79 out of  1,920   4%
 Number of 4 input LUTs:           64 out of  1,920   3%
Logic Distribution:
 Number of occupied Slices:        60 out of    960   6%
   Number of Slices containing only related logic:    60 out of    60 100%
   Number of Slices containing unrelated logic:       0 out of    60   0%
 Total Number of 4 input LUTs:     103 out of  1,920   5%
   Number used as logic:           64
   Number used as a route-thru:    39

 The Slice Logic Distribution report is not meaningful if the design is
 over-mapped for a non-slice resource or if Placement fails.

 Number of bonded IOBs:            20 out of    83   24%
 Number of BUFGMUXs:               1 out of    24   4%
```

ВИСНОВКИ

В атестаційній роботі розроблено і досліджено біт-потоковий апаратний обчислювач дробово-ірраціональних функцій, аргумент яких представлений імпульсним потоком.

Було розроблено математичну модель біт-потокового пристрою обчислення дробово-ірраціональних функцій, що є декомпозицією математичної моделі дробово-раціональної функції другого степеню та математичної моделі обчислювача добування квадратного кореня. Для отримання математичної моделі пристрою добування кореня був застосований метод ступінчастої апроксимації неперервних висхідних функції, що базується на визначенні обернених функцій з урахуванням абсолютної похибки обчислень. Мінімально можлива абсолютна похибка обчислення поточних значень дробово-ірраціональної функції для цілочисельних значень аргументу забезпечена на рівні половини одиниці молодшого біту вхідного імпульсного потоку. Застосування даного методу дозволяє підвищити точність і час обчислення функцій, що визначається часом введення аргументу в пристрій, тобто виконання обчислень в реальному масштабі часу.

Досліджуваний обчислювач реалізується двома послідовно з'єднаними блоками: дробово-раціональним модулем та пристроєм для добування квадратного кореня. Використано переваги принципу побудови біт-потокової конвеєрної архітектури обчислювача поліноміальних функцій, що входить складовою в структуру дробово-раціонального модуля спроектованого пристрою, яка реалізує функціональне перетворення розгортуючого типу на основі обчислення приростів відтворюваної функції, що сприяє зростанню продуктивності спроектованого пристрою, оскільки одночасно, на різних каскадах конвеєра, можуть виконуватися декілька паралельних обчислень.

З урахуванням вимог, пред’явлених до САПР, для реалізації обчислювача було обрано FPGA кристал сімейства Spartan-3E серії XC3S500E фірми Xilinx.

В результаті розробки та аналізу структурно-функціональної моделі біт-потокового обчислювача, було здійснено опис проекту для введення в САПР. Апаратна модель спроектованого обчислювача сформована на основі цифрового кінцевого автомата за типом Мура, що забезпечує надійність функціонування пристрою. Була розроблена змістовна граф-схема алгоритму його роботи, і, на підставі ГСА, отриманий граф переходів. За графами переходів з використанням стандартних шаблонів кода розроблено модель пристрою на мові опису апаратури VHDL.

Для верифікації апаратної реалізації, було розроблено тестове оточення (test bench). На отриманій поведінковій моделі досліджуваного обчислювача було наочно продемонстровано, що результати тестування збігаються з результатами попередніх обчислювальних процесів в компонентах пристрою.

Після перевірки режимів верифікації біт-потокового обчислювача було перейдено до наступного етапу проектування, а саме – імплементації моделі. Для дослідження було використано програмовану логічну інтегральну схему кристал сімейства Xilinx Spartan 3E.

Наукова новизна роботи полягає у реалізації біт-потокового обчислювача на новій технологічній платформі ПЛІС з використанням САПР на основі мов опису апаратури, для підвищення ефективності проектування цифрового пристрою.

Спроектований біт-потоковий обчислювач дробово-ірраціональних функцій може бути застосований в системах, орієнтованих на наскрізні технології інтелектуальних сенсорів, інтернет речей, удосконалення інтерфейсів при узгодженні сенсорів з цифровими системами збору і обробки інформації в якості зовнішніх апаратних модулів потокової обробки даних в архітектурах потокових процесорів, що складаються з типового процесорного ядра і зовнішніх апаратних модулів потокової обробки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Ларченко Л.В. Функціональне перетворення імпульсних потоків в апаратних обчислювачах математичних функцій / Л.В. Ларченко, Е.М. Кулак, Б.Д. Ларченко // Радіоелектроніка та інформатика. №3. – 2019. – С 27-34.

2. Буренева О.И. Бит-потоковое устройство извлечения квадратного корня / О.И. Буренева, О.А. Жирнова / Известия ЛЭТИ № 2, – 2019. С. 26 – 32.

3. Шахмейстер Л.Е. Цифро-частотные и время-импульсные преобразователи информации / Л.Е. Шахмейстер. – Москва: Книжный дом "Университет", 2008 – 252с.

4. Попов А.Ю. Проектирование цифровых устройств с использованием ПЛИС: Учебное пособие. / А.Ю. Попов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2009. – 80 с.

5. Ларченко Б.Д. Декомпозиція математичної моделі біт-потокового обчислювача ірраціональних функцій / Б.Д. Ларченко // Радіоелектроніка та інформатика. №4. – 2019. – С 34-39.

6. Стахів М.Ю. Автореф. дисертації. Цифрові функціональні перетворювачі розгортуючого типу з покращеними характеристиками // Поліграф. Центр Видавництва Національного університету "Львівська політехніка" – 2013. – 22 С.

7. Ларченко Л.В. Специализированный вычислитель для извлечения корня квадратного из суммы квадратов. / Л.В. Ларченко, А.В.Хаханова. // Радиоэлектроника и информатика. № 1(48). – 2010. – С.71 – 74.

8. Матвійків О.М. Інженерне проектування складних об'єктів і систем. Навчальний посібник./ О.М. Матвійків, С.П. Ткаченко, В.И. Хаханов. – Львів: Національний університет "Львівська політехніка", 2016 – 261с.

9. Норенков И.П Основы автоматизированого проектирования: учеб. Для вузов / И.П. Норенков. – 4-е изд., перераб. и доп. – М.: Изд-во МГТУ им.

Н. Э. Баумана. – 2009. – 430, [2] с.: ил. – («Информатика в техническом университете»).

10. Бибило П.Н. Основы языка VHDL. / П.Н. Бибило – М.: СОЛОН-Пресс, 2010. – 201с.