ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ, ПРАВА ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

Пояснювальна записка

до кваліфікаційної роботи на здобуття ступеня вищої освіти магістр

на тему: **«Розроблення та дослідження моделей логічних функцій дискретного перетворення інформації засобами комп'ютерної математики»**

Виконав: здобувач вищої освіти за освітньо-професійною програмою Інформаційні управляючі системи та технології спеціальності
126 Інформаційні системи
та технології
ступеня вищої освіти магістр групи
126ІСТ_мд_22
Раскін О. А.
Керівник: Одарущенко О. М.
Рецензент: Яхін С.В.

Полтава – 2023 року

ВСТУП

Актуальність теми дослідження. Для подальшого розвитку науки, техніки і виробництва необхідно вирішувати все більш складні завдання управління різноманітними динамічними процесами в галузях механіки, електротехніки, радіоелектроніки, а також в екологічних і соціальних системах. Зростання продуктивності, швидкості рухів, розмірів і потужності машин, підвищення точності та ускладнення характеру виробничого процесу або наукового експерименту роблять важким або навіть неможливим для людини швидке та точне управління рухом машин або ходом виробничого процесу. Автоматизація управління машинами і виробничими процесами вимагає, перш за все, глибокого теоретичного і експериментального вивчення самих процесів управління, а потім створення на основі отриманих знань широкого спектру надійних і ефективних приладів, пристроїв і систем автоматики. У сучасний період багато з наведених проблем успішно вирішуються за допомогою персональних комп'ютерів, які обладнані спеціальним програмним забезпеченням. Використання цих засобів дозволяє ефективно реалізовувати аналітичні розрахунки та чисельні методи, а також створювати візуальні графічні моделі, що імітують роботу вивчуваної системи або відтворюють структурні зміни складного приладу чи пристрою в часі. Ці методи дослідження відомі як імітаційне та ситуативне моделювання. Серед сучасних програмних систем комп'ютерної математики особливо вирізняється матрична математична система MATLAB, розроблена корпорацією MathWorks Inc. Ця система вважається ідеальним інструментом для реалізації різних видів моделювання, таких як аналітичне, чисельне, імітаційне і ситуаційне. MATLAB має потужні інструменти для взаємодії, візуалізації та графіки, а також численні програмні пакети для розширення функціоналу, такі як символічне диференціювання та інтегрування, ідентифікація систем, робота з нейронними мережами, обробка сигналів і зображень, розв'язання звичайних диференціальних рівнянь і інше. Одним із таких розширень є Simulink, пакет візуального імітаційного і ситуаційного моделювання, що дозволяє досліджувати різноманітні лінійні і нелінійні блокові динамічні системи та пристрої різного призначення.

Модель створюється за допомогою стандартних графічних блоків, які постійно розширюються. Параметри блоків можна задавати через зручні діалогові панелі. Результати роботи можуть бути візуалізовані у вигляді графіків або представлені у цифровій формі для подальшого використання. Користувач має можливість створювати звіти у форматі HTML, що включають структурну схему моделі, перелік блоків, таблиці параметрів блоків і записи реєструючих пристроїв у вигляді відповідних графіків і діаграм [1, 3-4].

Метою даної роботи є створення блоків математичних операцій для візуалізації результатів моделювання на основі програмного пакета Matlab Simulink.

Об'єктом дослідження є процеси розроблення блоків математичних операцій моделювання на основі програмного пакетів комп'ютерної математики.

Предметом дослідження є сукупність теоретичних, практичних основ розробки блоків математичних операцій на основі програмного пакета Matlab Simulink.

Інформаційна база кваліфікаційної роботи сформована з наукових фахових статей, наукових монографій, аналітичних звітів державних та міжнародних експертних груп щодо експлуатації критичних технічних систем, виконаних науково-дослідних та дисертаційних робот заданою тематикою.

Елементи наукової новизни роботи полягають в тому, що вперше розроблено моделі блоків математичних операцій для візуалізації результатів моделювання на основі програмного пакета Matlab Simulink.

Практична значущість роботи полягає в тому, що застосування математичної системи MATLAB, пакета імітаційного моделювання Simulink та розроблених моделей блоків дискретного перетворення іфнормації дозволяє швидко, ефективно і надійно вирішувати складні інженерні завдання при проектуванні систем автоматичного управління для реальних динамічних процесів з найрізноманітніших предметних областей.

Апробація результатів дослідження відбувалася шляхом оприлюднення доповідей на міжнародній та студентській конференціях.

Публікації. За результатами проведеного дослідження опубліковані тези: «Верифікаційна модель конфігураційної пам'яті програмовної логічної інтегральної схеми», матеріали Щорічного міждисциплінарного семінару «Студентські роботи за науковою тематикою кафедри інформаційних систем та технологій ННІ ЕУПІТ ПДАУ», що присвячений основним питанням студентської наукової конференції Полтавського державного аграрного університету, 29 листопада 2023 р. Полтава: ПДАУ, 2023.

Зв'язок роботи з науковими програмами, планами, темами: дослідження, проведені в кваліфікаційній роботі виконувалися в рамках/інтересах науково-дослідної роботи «Розвиток підприємництва: управлінські, економічні, інноваційні та правові аспекти» відповідно до договору №9 від 15.05.2023 р. між ТОВ «ПАФ Гарант» та Полтавським державним аграрним університетом (розділ «Обґрунтування показників оцінювання гарантоздатності розподілених інформаційних систем»).

Структура та обсяг роботи. Робота складається з вступу, трьох розділів, висновків, списку використаних джерел та додатків. Загальний обсяг роботи становить 76 сторінок; робота містить 42 рисунки; 4 таблиць; список використаних джерел, що включає 50 найменувань.

РОЗДІЛ 1
МОДЕЛЮВАННЯ ПРОЦЕСІВ І СИСТЕМ

1.1 Поняття моделі і математичного моделювання

В останні роки поняття «модель» використовується широко в самих різних областях науки, техніки, природознавства, в гуманітарних областях знання, в мистецтві і художній літературі. Скористаємося більш обмеженим поняттям математичної моделі – опис досліджуваного об'єкта на формальній мові, з допомогою чисел, рівнянь різного виду: кінцевих, диференційних, інтегральних, а також нерівностей або логічних співвідношень. Під терміном «математичне моделювання» часто розуміється деяке спрощення і вельми наближене математичний опис складної системи. Слово «модель» в цьому випадку протиставляється закону науки, щодо якого передбачається, що він описує явище природи деяким «безумовним» чином. Одна і та ж складна система може описуватися різними моделями, кожна з яких відображає тільки якусь сторону досліджуваної системи. У цьому випадку, природно, не виникає задача дискримінації – різні моделі можуть мати право на одночасне існування. Модель в цьому розумінні поводиться в якомусь сенсі так само, як і описувана нею система, а в іншому сенсі – інакше, бо модель не повністю ідентична (адекватна) описуваної системі або комплексу.

Математичне моделювання включає наступні кроки (етапи моделювання):

1) вибір розрахункової схеми і визначення необхідної деталізації;

2) математичний опис (складання системи рівнянь);

3) вибір методу рішення;

4) приведення моделі (що включає рівняння, метод, вихідні дані та початкові умови);

5) складання програми для ЕОМ;

6) проведення розрахунків (моделювання);

7) при необхідності слід повторити кроки 3-6;

8) аналіз отриманих результатів.

Важливим завданням є обробка результатів обчислень. На цьому етапі використовуються методи, які добре зарекомендували себе при експериментах з реальними об'єктами. Результати, отримані на математичних моделях, повинні бути порівняні з результатами натурного експерименту. Перші персональні комп'ютери в основному полегшували етап оформлення результатів моделювання. Тут використовувалися текстові редактори, графічні редактори, програми побудови графіків. Для побудови графіків-результатів використовувався відомий багатьом пакет GRAPHER, перші версії якого працювали ще під MS DOS. Сучасні пакети підготовки друкованої продукції включають засоби оформлення тексту, підготовки математичних формул, графіків, схем, таблиць. Сучасні технології дозволяють підготувати документ, що включає як об'єкти-документи інших типів або гіперпосилання на інші документи, так і програми обробки. В даний час найбільше застосування в задачах моделювання як етапу проектування отримали персональні комп'ютери. Початково широке їх використання визначалося не їх швидкодією, а можливістю гармонійно налаштувати робоче місце дослідника, організувати передачу даних між завданнями, отримати закінчений звіт. Сучасні програми чисельного моделювання систем і процесів стають все більш автоматизованими, полегшуючи користувачеві процес постановки і вирішення широкого класу складних задач. Ще більший ефект дають сучасні можливості якісного візуального представлення результатів. Серед таких програм, безумовно, одне з лідируючих місць займає система MATLAB + Simulink компанії MathWorks, на основі якої розроблено велику кількість професійних додатків (так званих Toolbox) для використання в особливих умовах. Ці додатки, що зібрали досягнення чисельного моделювання певного кола завдань, є не просто набором методів і команд, а, без перебільшення, останнім словом в даному напрямку досліджень. Професійне оволодіння спеціалізованим Toolbox дозволить розробнику піднятися на рівень світових досягнень і на рівних конкурувати з лідерами в цій галузі. Для проектування систем регулювання і управління, цифрової обробки сигналів, комунікаційних систем широко використовується блок інструментів Simulink, що дозволяє

моделювати динамічні системи, оцінювати їх роботу, модифікувати проект за допомогою графічних блок-діаграм. Simulink є інтерактивним середовищем для моделювання і аналізу різноманітних динамічних систем. Завдяки тісній інтеграції з MATLAB, у Simulink є прямий доступ до різноманітних інструментів для проектування і аналізу. У порівнянні з традиційним методом проектування систем, який передбачає створення прототипу, його всебічне тестування та внесення відповідних змін, Simulink пропонує ефективну і широко вживану альтернативу - імітаційне моделювання. Це дозволяє уникнути значних тимчасових та фінансових витрат, забезпечуючи при цьому точний та ефективний аналіз системи.Simulink представляє собою потужний інструмент для моделювання, що дозволяє швидко створювати та тестувати віртуальні прототипи з доступом до будь-якого рівня деталізації проєкту при мінімальних зусиллях. Використовуючи Simulink для ітеративного виправлення проекту до побудови прототипу, інженер може ефективно та швидко розробляти проєкти [2].

1.2 Математичне і імітаційне моделювання

Математичне та імітаційне моделювання – ключовий інструмент у розробці та фундаментальних дослідженнях складних систем.

Математичне моделювання (ММ) – процес встановлення відповідності реальної системи деякої математичної моделі і дослідження цієї моделі, що дозволяє отримати характеристики цієї системи.

Імітаційне моделювання (ІМ) – метод конструювання моделі реальної системи і постановки експериментів на цій моделі з метою дослідити її поведінку або оцінити різні стратегії, що забезпечують функціонування даної системи. Досягнення і прогрес теперішнього часу в області обчислювальних систем дозволяє ефективно проводити ІМ систем будь-якої складності, використовуючи продуктивні комп'ютери і спеціалізоване програмне забезпечення. Matlab і Simulink(програмні продукти компанії Mathworks) в комплексі представляють собою засіб математичного та імітаційного моделювання, що забезпечує

проведення досліджень практично у всіх областях науки і техніки. Дані програмні продукти були ефективно використані для розробки і побудови моделей, а також проведення ІМ, перебудованих обчислювальних середовищ (ПОС), спеціалізованих для високошвидкісного виконання алгоритмів цифрової обробки зображень.

ПОС – дискретна математична модель високопродуктивної обчислювальної системи, що складається з однакових і однаково з'єднаних один з одним елементарних обчислювачів (ЕО), програмно настроюються на виконання будь-якої функції з повного набору логічних функцій, пам'яті і будь-якого з'єднання зі своїми сусідами [3].

1.3 Засоби моделювання

Спробуємо розглянути програмне забезпечення персональних комп'ютерів, що використовується на різних етапах математичного моделювання. В останні роки в розвитку програмного забезпечення для персональних ЕОМ простежується тенденція застосування інтегрованих пакетів прикладних програм, що включають поряд зі спеціалізованими програмами і програми підготовки звітів, і багато ін.

Модульний підхід до моделювання досить глибоко реалізується в сучасних інтегрованих середовищах (пакетах прикладних програм). Одним з них є інтегроване середовище (в певному сенсі енциклопедія технічних застосувань при проектуванні та моделюванні) MATLAB фірми «The MathWorks Inc» (USA), яка, по суті, перемістилася з «великих» машин на персональні комп'ютери в кінці 80-х років, а широке застосування отримала в 90-х роках у зв'язку з істотним «поліпшенням» характеристик персональних комп'ютерів. Система MATLAB призначена для виконання інженерних та наукових розрахунків і високоякісної візуалізації отриманих результатів. Ця система застосовується в математиці, обчислювальному експерименті, імітаційному моделюванні, фінансових розрахунках і ряді інших областей. У пакет входить безліч добре перевірених чисельних методів, оператори графічного представлення результатів, засоби

створення діалогів. Відмінною особливістю MATLAB в порівнянні з рядом інших пакетів є матричне подання даних і великі можливості матричних операцій над даними. Використовуючи пакет MATLAB, можна, як з кубиків, побудувати досить складну математичну модель, або написати свою програму. А можна, використовуючи SIMULINK і технологію візуального моделювання, скласти імітаційну модель системи або комплексу та багато ін. Мова MATLAB дає можливість інженерам і вченим легко реалізовувати свої ідеї. Потужні чисельні методи і графічні можливості дозволяють перевіряти припущення і нові ідеї, а інтегроване середовище дає можливість швидко отримувати практичні результати. Сьогодні MATLAB використовується в безлічі областей, серед яких обробка сигналів і зображень, проектування систем управління, фінансові розрахунки і медичні дослідження. Його відкрита архітектура робить можливим використання MATLAB і супутніх продуктів для дослідження даних і створення власних інструментів, які використовують функціональні можливості MATLAB. Джерела MATLAB пов'язані з вирішенням завдань лінійної алгебри та підвищенням ефективності використання математичних пакетів LINPACK і EISPACK, призначених для роботи з матрицями. Якщо згадати, що джерелом математичних моделей лінійних систем з постійними параметрами служать лінійні диференціальні або різницеві рівняння з постійними коефіцієнтами, то стає зрозумілим, що теоретична основа цих рівнянь – лінійна алгебра. Тому ядро інтегрованої системи MATLAB становить основу для більшості процедур пакетів Control System Toolbox, систем Identification Toolbox, Frequency Domain Identification і ряду ін.

У цій системі згадані алгоритми інтегровані і є зручною для використання формі, що дозволяє назвати програму MATLAB симфонією алгоритмів.

Застосування поєднання програм MATLAB і Simulink призвело до розробки широкого класу професійних інструментальних додатків (toolboxes – набори інструментів) для створення, аналізу і оптимізації систем.

Toolbox – більше, ніж набір корисних функцій. Без перебільшення можна сказати, що вони являють собою останнє слово в розробці (дослідження) в таких областях, як управління, обробка сигналів, ідентифікація систем і багатьох ін.

Тому, освоївши і застосовуючи в MATLAB toolbox, можна досягти рівня розробників (дослідників) світового рівня. Деякі важливі характеристики toolbox: кожен toolbox побудований на програмах, надійність і точність яких перевірена багаторічним досвідом; всі toolbox сумісні і легко інтегруються не тільки з MATLAB, але і з Simulink і будь-яким іншим toolbox, який вже встановлений; всі toolbox написані в коді відкритої архітектури MATLAB, можна прочитати всі m-файли, зробити до них свої додавання або використовувати їх як шаблони при створення власних функцій; кожен toolbox може функціонувати на будь-якій комп'ютерної платформі, на якій працює MATLAB. MATLAB – дуже зручна система для розробки додатків, дозволяючи ефективно створювати програми, які виконуються як в середовищі MATLAB, так і незалежно. Крім багатьох можливостей мови MATLAB для створення алгоритмів, в ньому містяться також широкі можливості для проектування користувальницького інтерфейсу. У версії 5.x і 6.x для посилення цих можливостей включений модуль Guide (Graphical user interface design), що містить елементи візуального програмування. У Guide входять наступні взаємоузгоджені інструменти для створення та оформлення призначеного для користувача інтерфейсу: редактор властивостей для завдання і зміни властивостей складових частин інтерфейсу; панель керуючих елементів, що дозволяє включати в інтерфейс і вибирати розташування різних кнопок управління, вікна тексту і списків, рамки, засоби вертикальної і горизонтальної прокрутки; редактор команд, що дозволяє для кожного керованого елемента інтерфейсу записати їм програму; пристрій для вирівнювання об’єктів інтерфейсу; редактор меню для створення рядка меню інтерфейсу.

Операції, що виконуються при створенні користувальницького інтерфейсу в середовищі MATLAB, аналогічні за простотою і наочності операціями візуального програмування, виконуваних в середовищі програми Visual Basic [2, ст. 11-12].

1.4 Моделювання в АСУ

У процесі розробки нової автоматичної системи керування технологічними процесами (АСУ ТП) основним завданням є правильний вибір системи автоматичного регулювання (САР), подальший розрахунок і аналіз її характеристик. Уже на етапі розробки структури системи автоматичного регулювання розробники стикаються з проблемою вибору керуючих алгоритмів. Ключем до успішного вирішення даної проблеми є наявність вичерпної інформації про об’єкт управління (ОУ), необхідної для створення його математичної моделі. Наявність математичної моделі ОУ дозволить розробнику більш точно підходити до написання керуючих алгоритмів і визначення параметрів регуляторів, що входять до складу САР. Однак застосування математичної моделі об’єкта управління, що дозволяє проводити подібні розрахунки, підводить нас до наступної проблеми – проблемі реалізації математичної моделі в середовищах розробки АСУ ТП. Причиною тому є той факт, що реалізація математичної моделі не завжди є можливою в системах розробки програм АСУ для програмованих логічних контролерів (ПЛК) через обмеження в математичному апараті, також в цьому випадку істотно зростають тимчасові витрати на проектування. Перенесення програми управління в середовище, розроблену для математичного моделювання (наприклад, Matlab Simulink) також не є коректним, оскільки виконання алгоритму програми в ПЛК та персональному комп'ютері (ПК) будуть відрізнятися. Рішенням такого завдання є створення зв'язку, що об'єднує дві незалежно працюючі середовища: середовище налагодження керуючої логіки ПЛК та середовища моделювання з створеної математичної моделлю ОУ. Так, наприклад, в останніх версіях програмного забезпечення компанії MathSoft (Matlab v7.0 і вище) з’явився ряд нових компонентів, помітно спрощують запропоновану процедуру. Дані програми дозволяють отримувати доступ до даних контролера в режимі реального часу, як для їх читання, так і з можливістю запису в пам'ять контролера. Варто зазначити, що даний процес справедливий також і для випадку, коли замість реального ПЛК

використовується його симуляція в середовищі розробки. Процес обміну даними між ПЛК і ПК здійснюється за допомогою технології OPC (рис.1.1)

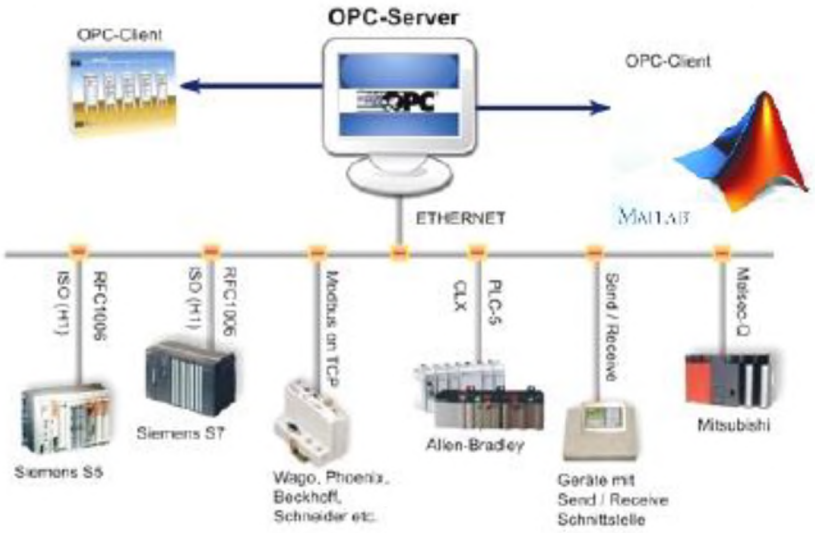

Рисунок 1.1 – Структурна схема АСУ ТП, заснованої на технології OPC

OPC (OLE для управління процесом) являє собою універсальний про-програмний інтерфейс, який може використовуватися на пристроях, що поставляються різними виробниками. OPC (OLE for Process Control) є промисловим стандартом, що описує обмін даними між різними додатками в умовах промислового виробництва. Цей стандарт дозволяє користувачам спостерігати, викликати та обробляти дані та події, які виникають в системах автоматизації, працюючи на своєму персональному комп'ютері. OPC-інтерфейс входить до складу програмного забезпечення на ПК і є платформою для систем операторського управління і візуалізації, а також інших додатків. Цей інтерфейс ґрунтується на моделі «клієнт/сервер», де один компонент (сервер) надає свої сервіси для іншого компонента (клієнта) через визначені інтерфейси.. Схематично зв'язок між програмованою логічною контроллером (ПЛК) та додатком Matlab на ПК можна зобразити на рисунку 1.2.

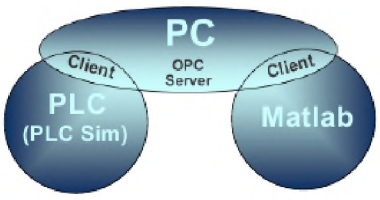

Рисунок 1.2 – Схема зв'язку Matlab і ПЛК за допомогою OPC

Розглянемо більш докладно процес спільної роботи ПЛК (або ж його симуляції) і математичної моделі в Matlab. Структура передачі і обробки інформації між персональним комп'ютером і контролером зображена на рис. 1.3.

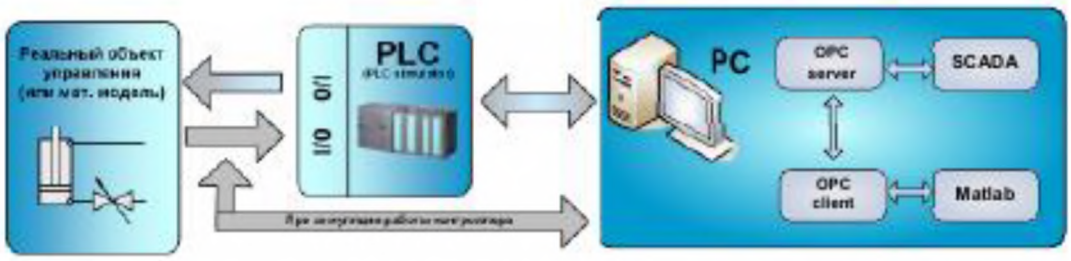

Рисунок 1.3 – Структурна схема передачі та обробки інформації між персональним комп'ютером і контролером

Для того щоб приступити до тестування і налагодження АСУ, необхідно виконати ряд кроків по створенню проекту тестування:

- Крок 1. Налаштування OPC сервера.

Необхідно створити теги обміну даними, які підміняють змінні, використовуються в програмі ПЛК для роботи з реальними вхідними/вихідними даними (датчики, сигнали зворотного зв'язку).

- Крок 2. Налаштування програми ПЛК.

У програмі ПЛК необхідно заблокувати звернення до реальних вхідним / вихідним каналам (датчики, сигнали зворотного зв'язку і т. д.). Далі OPC сервер буде виробляти читання / запис даних безпосередньо в програму ПЛК.

- Крок 3. Підготовка проекту в Matlab Simulink.

У Matlab Simulink необхідно створити математичну модель досліджуваного об'єкта. Також необхідно створити точки читання/запису даних по OPC, використовуючи стандартні інструменти з Matlab OPC Toolbox. Для мінімізації трудовитрат і помилок при моделюванні була розроблена спеціалізована бібліотека стандартних блоків спрощених математичних моделей різних об'єктів управління в Matlab Simulink. Кожен блок має додатковим зручним інтерфейсом введення параметрів моделі, що дозволяють налаштовувати модель під різні умови і вимоги системи.

В результаті виконання всіх кроків, описаних вище, отримуємо систему, що об'єднує в собі відразу кілька методів тестування і налагодження АСУ. Цикл виконання алгоритму даної системи можна розділити на наступні етапи:

1. Опитування та запис в область пам'яті ПЛК вхідних даних (стану об'єкта управління в даний момент часу - всілякі датчики, сигнали завдання і т. д.).

2. Обробка отриманих вхідних даних відповідно до керуючим алгоритмом в ПЛК.

3. Формування керуючих впливів на об'єкт управління (впливу на керуючі механізми).

4. Реакція об'єкта управління на вироблені АСУ керуючі впливи.

Для запуску алгоритму тестування необхідно виконати кроки в послідовності відповідно до рис. 1.4.

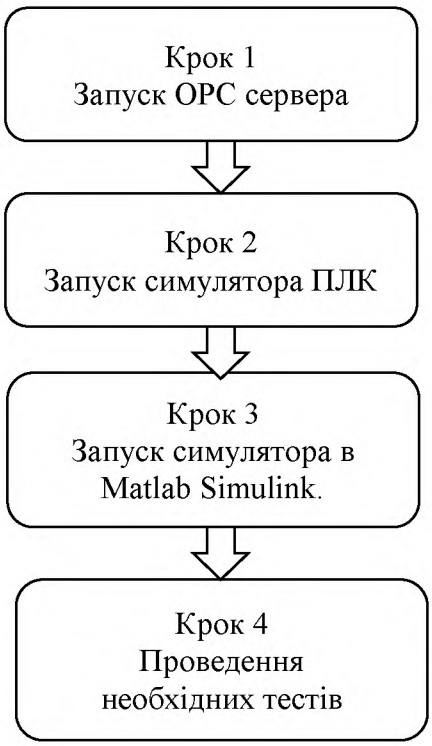

Рисунок 1.4 – Послідовність запуску симуляції

Процес виконання моделювання відбувається в режимі реального часу, тому користувач може спостерігати за всіма процесами, що відбуваються і виробляти керуючі впливу як всередині математичної моделі, так і в самій логікі контролера [4, ст. 1-5].

1.5 Використання програмного пакета OPC Toolbox

OPC Toolbox – це пакет, який розширює можливості середовища MATLAB і Simulink для взаємодії з серверами OLE for Process Control (OPC). Цей інструментарій дозволяє користувачам читати та записувати дані з пристроїв, які відповідають стандартам OPC. Крім того, він забезпечує взаємодію з системами розподіленого управління (SCADA), контролю диспетчерів, системами захоплення даних та програмованими логічними контролерами. Такий пакет робить можливим ефективне використання данних з різних джерел у середовищі MATLAB і Simulink для подальшого аналізу та моделювання динамічних систем.OPC Toolbox дозволяє MATLAB і Simulink реагувати на події OPC сервера:

1. Виключення;
2. Помилка;
3. Зміна параметрів.

OPC Toolbox дозволяє керувати різноманітними зовнішніми OPC-пристроями за допомогою єдиного простого синтаксису. Пакет також включає графічний інтерфейс OPC Tool для інтерактивної роботи та блоки Simulink, які дозволяють читати та записувати дані з OPC-серверів під час моделювання.. З використанням OPC Toolbox, інженерам у галузях електроенергетики, нафтохімії та інших галузях з безперервними процесами надається можливість керувати розподіленими системами керування. Пакет також дозволяє імпортувати промислові дані в середовище MATLAB для подальшого аналізу, візуалізації, імітаційного моделювання та тестування систем в зв'язці модель-пристрій. Такий підхід допомагає інженерам ефективно використовувати дані з промислових джерел у своїх дослідженнях та розробках [5].

Висновки з розділу 1

У першому розділі розглянуто теоретичні аспекти щодо побудови математичної моделі. Зроблено інформаційний огляд програми Matlab Simulink.

Розглянуто актуальність виконання моделювання в ході розроблення перспективних автоматизованих систем управління. Зроблено висновок про те, що у процесі розробки нової автоматичної системи керування технологічними процесами основним завданням є правильний вибір системи автоматичного регулювання (САР), подальший розрахунок і аналіз її характеристик.

РОЗДІЛ 2
SIMULINK ЯК ПІДСИСТЕМИ MATLAB

2.1 Загальні відомості

Комп'ютерна система MATLAB вважається однією з найбільш пристосованих для моделювання поведінки динамічних систем і широко використовується в інженерних та університетських колах через численні важливі переваги:

- простота опановування і доступність текстів практично всіх програмних засобів, окрім вбудованих: MATLAB визначається легкістю в освоєнні та доступністю текстів, що дозволяє швидко освоювати роботу з практично усіма його функціями;

- велика бібліотека доступних математичних програм: MATLAB має обширну бібліотеку, яка включає майже всі сучасні чисельні методи і функції, що полегшує використання різних математичних алгоритмів у дослідженнях;

- можливість утворювати власні програмні засоби: MATLAB дозволяє користувачам створювати свої власні програми і навіть коригувати існуючі, надаючи гнучкість індивідуалізації під конкретні вимоги;

- зручний і пристосований для практичних потреб інженерів і науковців апарат графічного оформлення результатів обчислень: MATLAB надає ефективний інтерфейс для візуалізації результатів обчислень, що робить його зручним для інженерів та науковців [6].

Так, MATLAB є потужним інструментом для числового аналізу, а також є мовою програмування, що використовується в даному пакеті.

Типове використання MATLAB включає:

- математичні обчислення: MATLAB надає розширений функціонал для виконання різноманітних математичних обчислень та оптимізації;

- створення алгоритмів: Мова програмування MATLAB дозволяє користувачам створювати складні алгоритми та реалізовувати різноманітні обчислювальні завдання;

- моделювання: MATLAB часто використовується для побудови і аналізу моделей, що дозволяє вивчати поведінку систем у різних умовах;

- аналіз даних, дослідження та візуалізація: Засоби MATLAB дозволяють виконувати аналіз і обробку даних, а також створювати візуалізації результатів;

- наукова і інженерна графіка: MATLAB забезпечує потужні можливості для створення наукових і інженерних графіків;

- розробка додатків, включаючи створення графічного інтерфейсу: MATLAB дозволяє розробляти додатки, включаючи ті, які мають графічний інтерфейс, що розширює можливості використання для кінцевого користувача.

Система MATLAB складається з п'яти основних частин, які можуть включати MATLAB, Simulink, Stateflow, MATLAB Compiler та інші інструменти залежно від конкретних потреб користувача.Мова MATLAB визначається як мова матриць і масивів високого рівня з міським управлінням потоками, функціями, структурами даних, уведенням-висновком і особливостями об’єктно-орієнтованого програмування. Вона надає зручний інструментарій для числового аналізу та розв'язання різноманітних завдань у наукових та інженерних областях. Середовище MATLAB включає в себе набір інструментів і пристосувань, з якими користувач чи програміст працює. Це включає в себе засоби для управління перемінними у робочому просторі MATLAB, введенням і виведенням даних, а також для створення, контролю та налагодження M-файлів і додатків MATLAB. Керуюча графіка – це графічна система MATLAB, яка має команди високого рівня для візуалізації дво- і тривимірних даних, обробки зображень, анімації і ілюстрованої графіки. Вона також включає команди низького рівня, що дозволяють повністю редагувати зовнішній вигляд графіки, а також створювати графічні користувацькі інтерфейси (GUI) для MATLAB додатків. Бібліотека математичних функцій в MATLAB є обширною колекцією обчислювальних алгоритмів, охоплюючи великий спектр від простих елементарних функцій, таких як сума, синус, косинус, комплексна арифметика, до складних операцій, таких як звернення матриць і обчислення власних значень. Програмний інтерфейс MATLAB дозволяє писати програми на мовах програмування C++ і Fortran, що

взаємодіють із самим MATLAB. Цей інтерфейс містить інструменти для виклику програм з MATLAB, реалізації динамічного зв'язку, а також для виклику MATLAB як обчислювального інструменту.

Також, цей інтерфейс надає можливість записувати дані у форматі MAT-файлів, що розширює можливості обміну даними між MATLAB та зовнішніми програмами, написаними на мовах C++ і Fortran.Simulink представляє собою інтерактивний інструмент для моделювання, імітації та аналізу динамічних систем, який забезпечує можливість будувати графічні блок-схеми, відтворювати роботу систем, аналізувати їх ефективність та вдосконалювати проекти.

Повна інтеграція з MATLAB дає користувачу безпосередній доступ до широкого спектру інструментів для аналізу і проектування. Окрім того, Simulink співпрацює з Stateflow для моделювання поведінки, активованої подіями.

При використанні Simulink втілюється концепція візуального програмування, що дозволяє користувачеві створювати моделі пристроїв, не маючи глибоких знань мов програмування чи чисельних методів. Цей інструмент є самостійним відносно MATLAB, хоча доступ до функцій MATLAB залишається відкритим. Вбудовані інструменти, такі як LTI-Viewer в Control System Toolbox, полегшують розробку систем управління.

Simulink дозволяє розширювати бібліотеку блоків, створювати власні блоки та складати нові бібліотеки. Користувач може вибирати методи розв'язання диференціальних рівнянь і способи зміни модельного часу. В процесі моделювання можна вести спостереження за системними процесами за допомогою спеціальних пристроїв з бібліотеки Simulink. Одна з переваг полягає в тому, що Simulink дозволяє розширювати функціонал, використовуючи підпрограми, написані на мовах Matlab, C++, Fortran та Ada.

2.2 Створення підсистеми Trsggered Subsystem

Підсистемою (subsystem) називають S-модель, створену з декількох блоків і представлену у вигляді одного блоку. Створення підсистем дозволяє вирішувати такі основні завдання:

1) компактного представлення S-моделей складних систем;

2) ієрархічного модульного підходу при розробці S-моделей складних систем;

3) компактного зберігання типових фрагментів, призначених для неоднократного використання;

4) розширення бібліотеки Simulink блоками користувача;

5) синхронізації паралельних процесів.

У Simulink розрізняють два різновиди підсистем:

1. Некеровані.

Підсистему називають некерованою, якщо в процесі моделювання її активізація не залежить від будь-якої умови. Розрізняють два види некерованих підсистем:

- Віртуальна (virtual) підсистема є «відкритою» – в процесі моделювання допускається зв'язок вхідних до її складу блоків з блоками S-моделі системи.

- Невіртуальна (nonvirtual) підсистема є «закритою» – в процесі моделювання вона функціонує як єдиний (неподільний) блок. Блок невіртуальної підсистеми виділяється жирною лінією.

2. Керовані.

Підсистему називають керованою, якщо в процесі моделювання її активізація залежить від деякого умови. Керовані підсистеми, за визначенням - невіртуальні, представлені двома родинами:

- умовні підсистеми (Conditional Subsystems);

- підсистеми, що моделюють логіку управління потоком (Modeling Control Flow Logic).

Для створення математичного блоку та функціонального блоку ми використовуємо керовану умовну тригерну підсистему.

В Т-підсистемі, для перегляду і редагування слід відкрити вікно підсистеми подвійним клацанням лівою кнопкою миші на піктограмі блоку Triggered Subsystem, керуючий вхід відображається наявністю блоку Trigger, який може розташовуватися на довільному місці (як це показано на рис. 2.1 та 2.2).

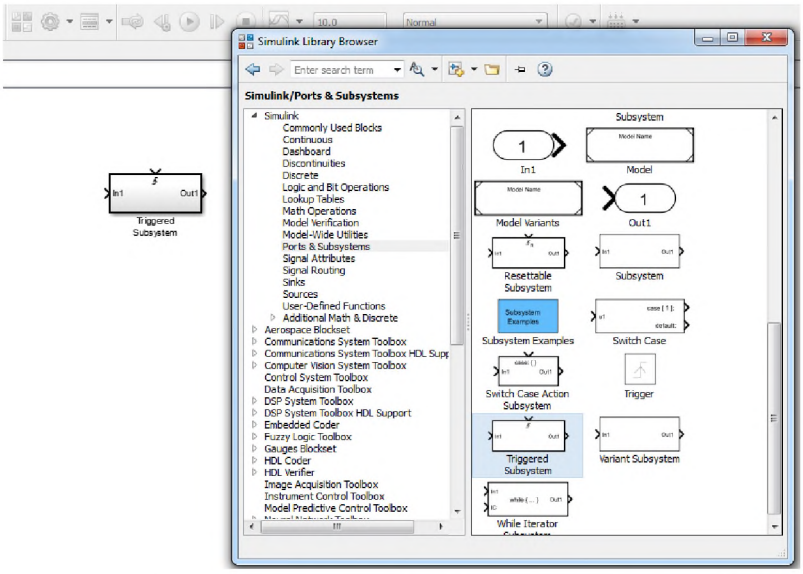

Рисунок 2.1 – Вікно S-моделей з блоком Triggered Subsystem

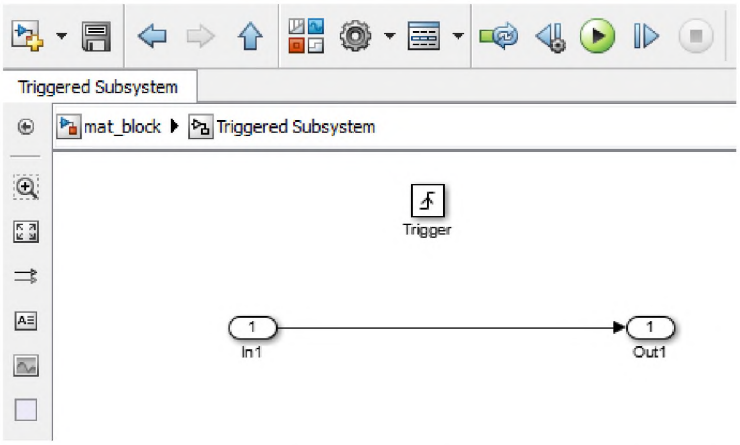

Рисунок 2.2 – Т-підсистемою, що містить, крім вхідного і вихідного портів, блок Trigger

Параметри блоку Trigger включають в себе (див. рисунок 2.3):

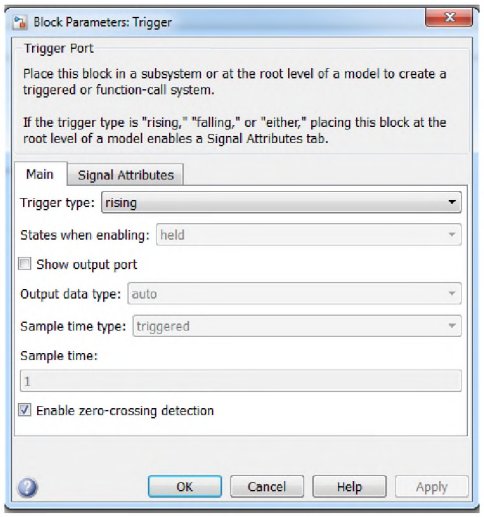

Рисунок 2.3 – Параметри блоку Trigger

1.Trigger type (Тип тригера) – запускає події на керуючому вході, активізує Т-підсистему:

- rising (зростання) - позитивний фронт – зростання сигналу від нульового або від'ємного значення до позитивного значення (або нульового при негативному початковому значенні);

- falling (спадання) - негативний фронт – спадання сигналу від позитивного або нульового значення до від'ємного значення (або нульового при позитивному початковому значенні);

- either (будь-який) – позитивний і негативний фронт (rising і falling).

States when enabling (Стан при вирішенні) – стан підсистеми при активізації:

- reset (скидання) – початкова (вихідне);

- held (утримання) – останнє, коли підсистема була активною.

Станом підсистеми називають значення внутрішніх змінних, які вона може зберігати до моменту активізації. Якщо такі змінні відсутні, то параметр States when enabling ігнорується.

2. Show output port (Показати вихідний порт) – флажок, при установці якого у блоку Trigger з'являється вихідний порт.

Через порт блоку Trigger всередину Т-підсистеми надходить той же сигнал, який є на її керуючому вході. Цей сигнал може бути використаний для управління всередині підсистем.

3. Output data type (Тип даних на виході) - типи даних на додатковому виході блоку Trigger ; параметр активізується при установці флажка Show output port; при виборі значень auto (за замовчуванням) тип даних успадковується від сигналу на виході блоку Trigger.

Інші параметри при зазначених значеннях параметра Trigger type не активовані [5].

Вихідні сигнали Т-підсистеми залежать від параметра Initial output вихідних портів Out, параметрів блоку Trigger, а також від флажка Latch input by delaying outside signal вхідних портів In.

2.3 Засоби взаємодії Matlab з Simulink

Моделювання процесів за допомогою S-моделей, хоча і має значні переваги, але також виявляє деякі суттєві недоліки. Серед переваг використання S-моделей можна відзначити наступне:

- швидкість і ефективність у створенні програм для моделювання складних динамічних систем. Це досягається шляхом складання блок-схем системи з готових стандартних блоків, що є візуальними представленнями відповідних математичних програм (візуальне програмування);

- зручні та наочні інструменти, які дозволяють перетворювати готову схему або отримувати додаткову інформацію щодо змін проміжних процесів;

- широкий набір ефективних програм-розв'язувачів (Solvers), які впроваджують методи чисельного інтегрування диференційних рівнянь з фіксованим кроком, інтегрування з автоматично змінюваним кроком, а також розв'язувачі для так званих жорстких систем диференційних рівнянь;

- відсутність потреби в спеціальній організації процесу чисельного інтегрування диференційних рівнянь;

- швидке та зручне отримання графічної інформації про зміну модельованих величин з часом, що сприяє легкій візуалізації та аналізу динамічних процесів. Однак використання S-моделей має свої недоліки:

- неякісне та незручне візуальне представлення сигналів у блок-оглядових вікнах, в порівнянні із засобами в середовищі Matlab;

- відсутність автоматизованої обробки результатів багаторазового моделювання S-моделей;

- обмежені можливості раціональної організації змінювання параметрів S-моделі та її блоків.

Таким чином, використання S-моделей може бути ефективним, але їхні обмеження можуть бути подолані за допомогою додаткової програмної реалізації, також враховуючи переваги візуального програмування та чисельного інтегрування.

Висновок полягає в тому, що програмна реалізація моделювання та саме моделювання за допомогою S-моделей можуть ефективно доповнювати один одного, надаючи додаткові можливості та переваги. Такий підхід може виявитися оптимальним, оскільки програмна реалізація дозволяє ефективно вирішувати завдання автоматизації та обробки результатів, в той час як S-моделі надають інтуїтивно зрозуміле та візуальне представлення для швидкого створення складних систем. За такого підходу можна максимально використати переваги обох методів, забезпечуючи комплексний та ефективний процес моделювання динамічних систем.

2.3.1S-моделі в Matlab

Для успішного поєднання програми MATLAB і S-моделі, потрібно мати засоби, які надають конкретні можливості:

Засоби передавання даних:

- забезпечення ефективного обміну даними між MATLAB і S-моделлю, включаючи передачу вхідних параметрів та отримання результатів моделювання;

- розробка механізмів для забезпечення двосторонньої комунікації, дозволяючи обміну даними не лише в одному напрямку (MATLAB → S-модель), але і у зворотньому напрямку (S-модель → MATLAB);

Засоби запуску та змінювання параметрів моделювання:

- розробка інтерфейсу, що дозволяє ініціювати процес моделювання S-моделі безпосередньо з середовища MATLAB;

- можливість динамічно змінювати параметри моделювання та параметри блоків S-моделі в реальному часі;

Засоби виклику MATLAB-програм:

- розробка механізму для виклику MATLAB-програм з S-моделі, щоб реалізувати специфічні функції або обчислення в середовищі MATLAB.

Засоби створення S-блоків з MATLAB:

- розробка інструментів, які дозволяють створювати блоки S-моделі з MATLAB, використовуючи програми, написані на M-мові;

- забезпечення можливості інтеграції програм, написаних в MATLAB, в структуру S-моделі.

Такі засоби можуть бути реалізовані за допомогою спеціалізованих API або інших інтерфейсів, які дозволяють взаємодіяти між MATLAB і S-моделлю. Це може включати створення зручного інтерфейсу користувача для взаємодії між обома середовищами та реалізацію механізмів обміну даними для забезпечення ефективної комунікації.

2.3.2 Моделювання у Simulink

Властивості кожного блоку S-моделі можуть бути описані наступним чином (див. рисунок 2.4):

вектор вхідних величин u визначає вхідні сигнали або величини, які впливають на поведінку блоку. Це може бути вектор значень, що подаються на вхід, і використовується для обчислення вихідних величин або змінних стану.

Вектор вихідних величин y. Представляє собою вихідні сигнали або величини, які генерує блок як результат внутрішньої обробки вхідних сигналів. Вихідний вектор може містити значення, які подаються на виході блоку для подальшого використання в інших частинах системи.

Вектор змінних стану x. Це вектор змінних, які внутрішньо утримують інформацію про стан блоку. Змінні стану можуть бути використані для збереження попередніх значень або для внутрішньої динаміки блоку. Вони можуть допомагати враховувати попередні стани системи при кожному кроці часу.

Ці внутрішні характеристики кожного блоку S-моделі визначають, як блок реагує на вхідні сигнали, які величини він генерує на виході та які параметри внутрішньої динаміки враховуються в процесі моделювання. Такий формат дозволяє ефективно моделювати різноманітні динамічні системи з використанням блоків, які можуть бути з'єднані у складні конфігурації для аналізу та синтезу систем управління.

Вектор змінних стану в S-моделі може включати в себе різні типи змінних, такі як змінні неперервних станів, змінні дискретних станів або їх комбінації.

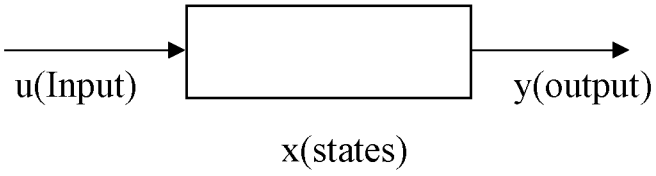

Рисунок 2.4 – Взаємодія величин, що визначають поточний стан блоку

Зв’язки між цими величинами можуть бути математично виражені за допомогою різних рівнянь. Нижче представлені можливі математичні зв'язки для вектора змінних стану:

- для змінних неперервних станів (continuous states):

$$X'_c = f_c(x_c,u,t),$$

де X'_c - похідна за часом змінних неперервних станів, f_c - функція, що визначає зміну цих змінних відносно часу, x_c - вектор змінних неперервних станів, u - вектор вхідних сигналів, t – час$

- для змінних дискретних станів (discrete states):

$$X_d(k+1)=f_d(x_c,x_d,u,t),$$

де $X_d(k+1)$ - значення змінних дискретних станів на момент часуk+1, f_d - функція, що визначає зміну цих змінних при переході від одного дискретного часу до наступного;

- для комбінації неперервних і дискретних станів:

$$X'_c = f_c\ (x_c,x_d,u,t)$$

$$x_d(k+1)=f_d(x_c,x_d,u,t)$$

Комбінує рівняння для неперервних та дискретних станів. Ці рівняння визначають, як змінюються стани системи в залежності від вхідних сигналів, часу та поточних значень станів.

Фаза моделювання у S-моделях складається з двох етапів: ініціалізації та основної фази. У фазі ініціалізації виконуються наступні етапи:

- оцінка параметрів блоків: Параметри блоків передаються в середовище Matlab для обчислення або оцінювання. Результати числових операцій стають

фактичними параметрами блоків згладжування ієрархії моделі: Кожна підсистема, яка не виконується умовно, замінюється блоками, з яких вона складається. Це допомагає врахувати всі необхідні компоненти для подальшого моделювання;

- сортування блоків: Блоки сортуються в порядку, в якому їх потрібно змінювати. Цей алгоритм забезпечує такий порядок, що будь-який блок з прямим підключенням залишається незмінним, доки змінюються блоки, що визначають вхідні величини. На цьому етапі виявляються алгебраїчні цикли;

- перевірка зв’язків між блоками: Перевіряються зв'язки між блоками, зокрема, забезпечується збіжність довжини вектора вихідних величин кожного блоку з очікуваною довжиною векторів вхідних величин, керованих ними блоків. Це включає перевірку правильності зв'язків для забезпечення коректної взаємодії між блоками під час основної фази моделювання.

Ці етапи ініціалізації є важливими для визначення початкових умов і забезпечення коректного виконання основної фази моделювання.

У процесі фази основного моделювання використовується чисельне інтегрування за допомогою методів розв'язання диференціальних рівнянь. Кожен метод інтегрування залежить від того, наскільки модель здатна визначати похідні її неперервних станів. Розрахунок цих похідних відбувається у два етапи:

- обчислення вихідних величин: Спочатку кожна вихідна величина блоку обчислюється в порядку, визначеному під час сортування блоків. Це включає в себе обчислення значень вихідних величин з урахуванням вхідних змінних і поточних значень змінних стану;

- обчислення похідних: На другому етапі обчислюються похідні кожного блоку для поточного моменту часу, враховуючи вхідні змінні і змінні стану. Отриманий вектор похідних використовується для обчислення нового вектора змінних стану у наступний момент часу. Після завершення обчислення нового вектора змінних стану, блоки даних і блоки-оглядові вікна оновлюються. Цей цикл повторюється для кожного кроку часу, забезпечуючи чисельне інтегрування та оновлення моделі протягом тривалості моделювання.

Властивості і параметри розв'язувача в пакеті Simulink можна налаштовувати у вікні «Configuration Parameters», яке викликається через опцію «Simulation > Configuration Parameters» у меню блок-схеми.

У верхній частині цього вікна розташовані поля для введення «Start time» (Початковий час) і «Stop time» (Кінцевий час), де можна встановити відповідно початкове і кінцеве значення часу. В області «Solver options» (Параметри розв'язувача) в списку «Type» (Тип) вибирається тип розв'язувачів, а в спадному списку «Solver» (Розв'язувач) праворуч від нього вибирається конкретний розв'язувач.

Якщо обраний тип розв'язувачів «Fixed-step» (з фіксованим кроком), то праворуч від цього параметру ви знайдете набір конкретних розв'язувачів, які можна вибрати. Описані параметри дозволяють користувачеві налаштовувати процес розв'язування рівнянь під час симуляції, вибираючи підхід, який найкраще відповідає характеру моделі та поставленим завданням.

Наявні типи розв'язувачів у розділі «Solver options» (Параметри розв'язувача) пакету Simulink включають:

- discrete (no continuous states) – дискретний (без неперервних станів);
- ode5 (Dormand-Prince) – метод Дормана-Принса (п'ятого порядку);
- ode4 (Runge-Kutta) – метод Рунге-Кутта (четвертого порядку);
- ode3 (Bogacki-Shampine) – метод Богацького-Шампена (третього порядку);
- ode2 (Heun) – метод Хойна (другого порядку);
- ode1 (Euler) – метод Ейлера (першого порядку);
- ode14x (extrapolation) - екстраполяція.

Ці розв'язувачі надають користувачу можливість вибирати оптимальний метод чисельного інтегрування для конкретної моделі залежно від її характеристик і вимог до точності розв'язку.

У вікні «Configuration Parameters» (Параметри конфігурації) Simulink, в розділі «Solver options» (Параметри розв'язувача), доступні різні параметри для вибору типу інтегратора та режиму роботи.

Ці параметри включають:

Якщо обраний тип розв'язувачів Fixed-step (з фіксованим кроком): Fixed step size (Розмір фіксованого кроку): Користувач вводить значення кроку інтегрування.

Режими роботи для періодичних часів вибіркових величин:

- Auto (автоматичний): автоматичний вибір режиму відповідно до характеристик моделі;

- SingleTasking (однозадачний): режим однозадачності;

- MultiTasking (багатозадачний): режим багатозадачності.

Якщо обраний тип розв'язувачів Variable-step (зі змінним кроком): Variable-step size (Розмір змінного кроку): користувач вводить значення для змінного кроку інтегрування.

Додаткові параметри для розв'язувачів Variable-step:

discrete (no continuous states) – дискретний (без неперервних станів);

ode45 (Dormand-Prince) – метод Дормана-Принса;

ode23 (Bogacki-Shampine) – метод Богацького-Шампена;

ode113 (Adams)– метод Адамса;

ode15s (stiff\NDF) – метод NDF для жорстких систем;

ode23s (stiff\Mod. Rozenbrock) – модифікація Розенброка для жорстких систем;

ode23t (Mod. stiff\Trapezoidal) – метод трапецій, модифікація для жорстких систем;

ode23tb(stiff\TR-BDF2) – метод TR-BDF2 для жорстких систем.

У розділі «Solver options» вікна «Configuration Parameters» у Simulink, якщо обраний тип розв'язувачів Variable-step (зі змінним кроком), виникають додаткові поля введення, які визначають параметри інтегрування. Нижче перераховані ці поля та їх значення за замовчуванням:

- Max step size (Максимальний розмір кроку): Автоматично встановлено значення «auto». Максимальний розмір кроку інтегрування;

- Min step size (Мінімальний розмір кроку): Автоматично встановлено значення «auto». Мінімальний розмір кроку інтегрування;

- Initial step size (Початковий розмір кроку): Автоматично встановлено значення «auto». Початковий розмір кроку інтегрування.

Праворуч від цих полів знаходяться:

- Relative tolerance (Відносна точність): Автоматично встановлено значення «auto», і за замовчуванням рівне 1e-3. Відносна точність, яка використовується для контролю точності чисельного інтегрування;

- Absolute tolerance (Абсолютна точність): Автоматично встановлено значення «auto», і за замовчуванням рівне 1e-6. Абсолютна точність, яка використовується для контролю точності чисельного інтегрування.

Зазвичай, користувач може залишити ці параметри автоматичними, якщо не має конкретних вимог до їх значень. Однак, у випадках, коли точність інтегрування важлива, користувач може встановити конкретні значення для цих параметрів.

2.3.3 Обмін даними між середовищем Matlab і S-моделлю

Робочий простір середовища MATLAB може використовуватися у S-моделі для передачі значень параметрів. Це означає, що при використанні імен змінних у вікнах налаштування блоків S-моделі, які заздалегідь визначені в робочому просторі, відповідні значення автоматично передаються блокам S-моделі. Таким чином, для комфортної зміни параметрів блоків S-моделі, наприклад, у діалоговому режимі, достатньо виконати наступні кроки:

- вказати у вікнах налаштування блоків S-моделі ідентифікатори (імена) замість конкретних числових значень;

- організувати, використовуючи інструменти середовища MATLAB, присвоєння числових значень цим ідентифікаторам та, за необхідності, їх зміну у діалоговому режимі;

- після присвоєння числових значень всім ідентифікаторам (наприклад, за допомогою запуску відповідної MATLAB-програми) провести запуск S-моделі для моделювання.

Було розглянуто деякі засоби обміну даними, такі як блоки From Workspace і To Workspace стандартної бібліотеки Simulink. Перший блок використовується для введення сигналів, які передбачено (обчислені) та записані у робочому просторі MATLAB (наприклад, під час обчислень в середовищі MATLAB), у процес моделювання S-моделі. Другий блок забезпечує можливість запису результатів, одержаних під час моделювання з використанням S-моделі, у робочий простір середовища MATLAB. Щоб записати результати моделювання у робочий простір, блок To Workspace додається до схеми S-моделі. На вхід блока подається сигнал, який потрібно записати, і вказується ім'я змінної у полі Variable name вікна налаштувань блоку. Моменти модельного часу при цьому не записуються. Для використання у S-моделі процесу, що використовує дані, записані у робочому просторі, використовується блок From Workspace. Вихід цього блоку з'єднується з одним з входів інших блоків, а вікно налаштувань блоку вказує, який вектор буде використаний для обчислення процесу в S-моделі. Це може бути, наприклад, масив, який складається з двох імен - масиву значень аргументів (моменти часу) і масиву значень процесу, призначеного для цих значень аргументів (наприклад, [T, D]). Якщо реальні значення моментів часу при моделюванні не збігаються з записаними у масиві T, відбувається лінійна інтерполяція значень масиву D відповідно до попереднього і наступного значень моментів часу у масиві T. Але, можна використовувати більш простий метод для виконання цих операцій, не застосовуючи вказані блоки. Щоб інтегрувати визначений у програмі Matlab процес у S-модель як вхідний, скористайтеся механізмом додавання та підключення портів введення і виведення. Для цього виконайте такі кроки.

В S-моделі відкрийте вікно «Ports & Subsystems» та оберіть «Inports» для входів або «Outports» для виходів. Виберіть опцію «Add & Connect» та оберіть «Inport» або «Outport» з меню. Введіть ім'я для порту та виберіть опцію «Connect to Simulink signal» для встановлення з’єднання. Вкажіть змінну MATLAB, яку ви хочете включити, у полі «MATLAB variable» та збережіть налаштування. Цей

метод дозволяє прямо включати результати з MATLAB у ваші симуляції Simulink, спрощуючи обмін даними між середовищами.

Щоб передати деякі сигнали, які формуються в S-моделі, до робочого простору Matlab, слід виконати наступні кроки:

- вставте блоки виходів (Out) у блок-схему моделі та підключіть до них потрібні вихідні величини інших блоків;

- у вікні S-моделі викличте команду Simulation > Configuration Parameters, щоб відкрити вікно Configuration Parameters. Оберіть команду Data Import/Export (Імпорт/експорт даних);

- в області Save to Workspace (Зберегти у робочому просторі) відкритого вікна встановіть прапорці Time і Output. У відповідних полях справа введіть імена, під якими значення часу і вихідні величини, що подаються на вихідні порти, будуть записані у робочий простір. За замовчуванням ці імена - tout (для модельного часу) та yout (для даних з вихідних портів).

У даному випадку значення модельного часу будуть зберігатися в робочому просторі у вигляді масиву під назвою «tout», а відповідні значення вихідних сигналів при цих моментах часу будуть заноситися у стовпець матриці «yout». У першому стовпці матриці буде інформація про процес, що подається на перший вихідний порт «Out1», у другому - процес, що подається на другий порт «Out2» і так далі. Після активації прапорця «Initial state» в розділі «Load from workspace» можна ввести початкові значення змінних стану системи в S-модель. Якщо встановити прапорець «States» в розділі «Save to workspace», то поточні значення змінних стану системи можна буде зберегти в робочому просторі під назвою «xout» (або під іншою обраною назвою, яку можна вказати у відповідному полі праворуч від прапорця «State»). Нарешті, при встановленні прапорця «Final state», можна записати кінцеві значення змінних стану у вектор «xFinal».

2.3.4 Використання моделювання S-моделі в середовищі Matlab

Розглянемо можливості запуску процесу моделювання S-моделей з використанням програми Matlab. Для запуску S-моделі можна використовувати

процедуру sim, яка має наступний формат: sim: [t,x,y1,y2,...,yn] = sim(model, timespan,options,ut).

У цьому виразі: model - це символьний рядок, що містить ім'я MDL-файлу, в якому зберігається відповідна S-модель; timespan - вектор, який складається з двох елементів, визначаючи початковий та кінцевий моменти часу моделювання; options - вектор значень параметрів інтегрування, який встановлюється за допомогою процедури: simset options = simset ('Властивість1',Значення1,'Властивість2',Значення2...);

Функція sim повертає наступні значення: t - вектор моментів модельного часу; x - масив або вектор змінних стану системи; y1, y2, ..., yn - стовпці матриці вихідних змінних системи, які подаються до вихідних портів. В Matlab можна змінювати параметри розв'язувача та процесу інтегрування за допомогою функції simset, як показано вище. Цим способом можна задавати значення різних властивостей розв'язувача.

У контексті використання функції simset в Matlab, можна визначити різні параметри розв'язувача та процесу інтегрування. Ось опис кількох таких параметрів:

'Solver' – назва розв’язувача; значення може бути одним із наступних: 'ode45', 'ode23', 'ode1b', 'ode15s', 'ode23s' для інтегрування з автоматично змінюваним кроком; або 'ode5', 'ode4', 'ode3', 'ode2', 'ode1' для інтегрування з фіксованим кроком;

'RelTol' – відносна припустима похибка; значення може бути додатним скаляром; за замовчуванням встановлюється 1e-3;

'AbsTol' – абсолютна припустима похибка; значення може бути додатним скаляром; за замовчуванням встановлюється 1e-6;

'FixedStep' – фіксований крок (додатний скаляр);

'MaxOrder' – максимальний порядок методу (застосовується лише для методу 'ode15'); може бути одним з цілих чисел 1, 2, 3, 4; за замовчуванням дорівнює 5;

'MaxRows' – максимальна кількість рядків у вихідному векторі; невід'ємне ціле;

'InitialState' – вектор початкових значень змінних стану;

'FinalStateName' – ім'я вектора, в який буде записуватися кінцеве значення вектора змінних стану моделі;

'OutputVariables' – вихідні змінні системи; за замовчуванням має значення {txy}; можливі варіанти tx, ty, xy, t, x, y; усі вони неявно вказують, які саме вихідні змінні не будуть виводитися.

2.3.5 Створення S-блоків

В системі Matlab передбачено використання механізму S-функцій для перетворення процедур, написаних мовами високого рівня, у блоки S-моделі в середовищі Simulink. S-функція представляє собою програму, яка написана користувачем мовою Matlab або C і візуально представляється у вигляді блока Simulink. Використання S-функцій дозволяє вирішити такі задачі:

- утворення нових (користувацьких) блоків: S-функції дозволяють створювати нові блоки, які можуть бути включені у бібліотеку Simulink, розширюючи можливості пакета.

- використання опису модельованої системи у виді системи математичних рівнянь: S-функції надають можливість використовувати опис модельованої системи у вигляді математичних рівнянь, що спрощує процес моделювання та аналізу систем;

- виключення раніше створених програм: Можна включати раніше написані програми, які були реалізовані на M-мові або мові C, у S-модель, роблячи їх доступними для використання в середовищі Simulink.

Цей підхід дозволяє зручно інтегрувати існуючий код та алгоритми, написані на різних мовах програмування, у симуляційні моделі систем в Simulink, що розширює можливості моделювання та аналізу систем.

В загальному випадку, заголовок S-функції, яка формується на основі M-мови, має наступний вид. Структура коду S-функції може бути визначена у файлі SfinTMPL.m, який розташований у папці TOOLBOX\SIMULINK\BLOCKS.

Заголовок S-функції зазвичай виглядає наступним чином:function [sys,x_0,str,ts] = <Ім’я S-функції>(t, x, u, flag{,<Параметри>})

Стандартні аргументи S-функції включають:

t: поточне значення аргументу, яке представляє собою модельний час;

x: поточне значення вектора змінних стану системи;

u: поточне значення вектора вхідних величин, які подаються на вхід S-функції;

flag: цілочислова змінна, яка вказує на етап виклику S-функції. Зазвичай, це може бути 0 для ініціалізації, 1 для обчислення виводів, 2 для обчислення похідних, 3 для обчислення і виводів і похідних, і 9 для завершення;

<Параметри>: це додаткові ідентифікатори, які характеризують модельовану систему. Їх може бути будь-яка кількість і вони використовуються у визначенні S-функції. Їхня наявність не є обов'язковою, і це залежить від конкретних потреб моделі.

Спосіб обробки цих аргументів та виведення необхідних значень залежить від конкретного призначення S-функції і її ролі в моделі.

Наведені процедури є стандартними функціями, які визначають різні аспекти роботи S-функції в середовищі Simulink. Давайте коротко оглянемо кожну з них:

- mdlInitializeSizes: Ця функція встановлює розміри змінних S-функції і надає початкові значення змінних стану. Це важливо для належної ініціалізації S-функції перед початком симуляції;

- mdlDerivatives: Якщо система описується диференційними рівняннями, ця функція обчислює поточні значення правих частин диференційних рівнянь системи;

- mdlUpdate: Ця функція відповідає за оновлення значень змінних стану на наступному інтервалі дискретного часу, якщо змінні стану об'явлені як дискретні;

- mdlOutputs: Ця функція обчислює значення вектора вихідних змінних блоку S-функції. Вона грає ключову роль у визначенні виводів системи;

- mdlGetTimeOfNextVarHit: Це внутрішня функція, яка визначає момент часу, коли конкретна змінна перетинає заданий рівень;

- mdlTerminate: Функція, яка викликається при завершенні роботи S-функції. Вона може використовуватися для звільнення ресурсів та завершення додаткових дій.

Ці функції надають розширені можливості для визначення поведінки S-функції в різних контекстах, забезпечуючи більш гнучке та налаштоване моделювання систем. Їх використання залежить від конкретних вимог та особливостей моделі, яку ви розробляєте [9].

2.4 Складові елементи математичного блоку

Для початку розглянемо типи даних з якими будем працювати. Існує нескінченна кількість цілих чисел, проте ми завжди можемо вибрати таку кількість біт, щоб ефективно представити будь-яке ціле число, яке виникає при розв'язанні конкретної задачі. З іншого боку, дійсні числа не тільки є нескінченними, але також безперервними. Тому, незалежно від того, скільки біт ми вибрали, ми неминуче стикаємося з числами, для яких немає точного представлення.

Використання чисел з плаваючою комою – це один із способів представлення дійсних чисел, який є компромісом між точністю та діапазоном прийнятних значень. Число з плаваючою комою складається з окремих розрядів, розділених на знак, експонент, порядок і мантису. Експонент і мантиса - цілі числа, які, разом із знаком, визначають представлення числа з плаваючою комою. Математично це виражається так:

$$(-1)^s \times M \times B^E \qquad (2.4)$$

де s - знак,

B-основа, E - порядок, M - мантиса.

Основа числення (B) визначає систему числення для розрядів. Математично було доведено, що числа з плаваючою комою з базою B = 2 (двійкове представлення) є найбільш стійкими до помилок округлення. Таким чином, на практиці найчастіше використовуються лише бази 2, і рідше - 10. Ми завжди будемо вважати, що B = 2, і формула числа з плаваючою комою матиме вигляд:

$$(-1)^s \times M \times 2^E \qquad (2.5)$$

Мантиса – це ціле число фіксованої довжини, яке представляє старші розряди дійсного числа.

Існує розрізнення між нормалізованими та денормалізованими числами. У нормалізованих числах мантиса завжди є позитивною. Якщо мантиса не є нульовою, число вважається ненульовим, і його порядок визначається, при цьому неявний старший біт мантиси розглядається як рівний нулю. Такі числа отримали назву «денормалізовані» [9].

Тип даних single – це числові масиви з числами одинарної точності.

Перейдемо до опису основних блоків, які використовуються в даній моделі.

Блок Add - виконує матричне або поелементне складання. Параметри блоку задаються на двох вкладках - Main і Signal Attributes (див. рисунок 2.1)

На вкладці Main визначаються параметри:

1. Icon shape (Форма піктограми) – вид піктограми:

- прямокутник (rectangular);

- окружність (round).

2. List of signs (Список знаків) – послідовність знаків доданків – плюс (+) і мінус (-), між якими, при бажанні, можна ставити роздільник (|);

3. Sample time (Інтервал часу) – інтервал часу між сусідніми значеннями сигналу в процесі моделювання.

На вкладці Signal Attributes визначаються параметри:

1.Require all inputs to have the same data type (Потребувати, щоб усі вхідні сигнали належали до однакового типу даних) – флажок, при установці якого вхідні сигнали (складові) повинні належати до однакового типу даних, в іншому випадку буде видане повідомлення про помилку;

2.Accumulator data type - тип даних для суми (див. табл. 2.1)

Таблиця 2.1 – Тип даних на виході

Тип даних на виході	Спосіб формування
Inherit from '…'	Успадковується від зазначеного в апострофа параметра блоку
Inherit as input	Успадковується від сигналу на вході
Inherit as first input	Успадковується від сигналу на першому вході
Inherit as second input	Успадковується від сигналу на другому вході
Inherit via back propagation	Успадковується від сигналу на виході попереднього блоку
Inherit via internal rule	Успадковується відповідно до внутрішнього правилом без переповнень з максимальною точністю
double, single, int8, uint8, int16, uint16, int32, uint32, boolean	Вказується явно
<data type expression>	Вказується символічне ім'я типу даних в апострофа для константи в поле введення Constant value, представленої виразом, наприклад, 'single' і т.п.
fixdt	Формуються за допомогою об'єкта fixdt для даних з ФТ

3. Output minimum – мінімальне значення даних на виході, за замовчуванням дорівнює [ ], що еквівалентно -Inf; Output maximum - максимальне значення даних на виході, за замовчуванням дорівнює [ ], що еквівалентно +Inf.

Параметри Output minimum і Output maximum використовуються для контролю діапазону значень даних в процесі моделювання, а також для автоматичного масштабування даних з фіксованою точкою.

4. Output data type (Тип даних на виході) – тип даних на виході (див. табл. 2.1).

5. Lock output data type setting against changes by the fixed-point tools (Блокування типу вихідних даних від змін засобами фіксованої точки) – флажок, при установці якого блокується автоматичне масштабування даних типу fixdt. Тобто, цей флажок активний для всіх типів даних, крім fixdt.

6. Integer rounding mode (Режим цілочисельного округлення)

- Floor (округлення в напрямку -∞) - округлення до найближчого цілого в сторону зменшення.

- Selling (округлення в напрямку ∞) - округлення до найближчого цілого в сторону збільшення.

- Convergent - округлення до найближчого парного цілого. При дробової частині числа, що дорівнює 0.5, - в сторону найближчого парного числа.

- Nearest - округлення до найближчого цілого. При дробової частини числа, що дорівнює 0.5, - в сторону збільшення.

- Round - округлення до найближчого цілого. При дробової частини, що дорівнює 0.5, - в сторону збільшення модуля числа.

- Simplest - автоматичний вибір оптимального режиму округлення за допомогою комбінації режимів округлення для даних з фіксованою точкою.

- Zero - округлення в напрямку нуля - відсікання дробової частини.

7. Saturate on integer overflow (Насичення до цілого при переповненні) - флажок, при установці якого використовується арифметика насичення для даних з фіксованою точкою: при переповненні результат автоматично замінюється максимально можливим (за модулем) для формату слово.

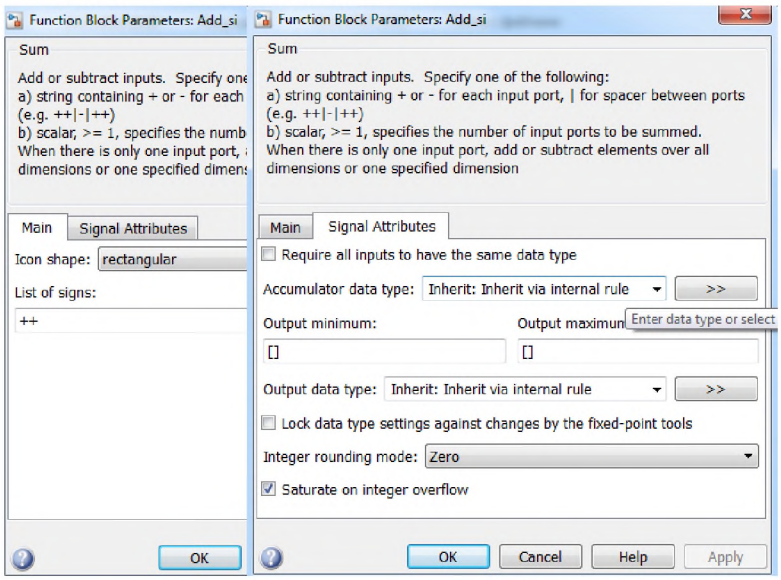

Рисунок 2.5 – Параметри блоку Add

Аналогічні параметри були встановлені у блоках Subtract, Product.

Блок Divide виконує матричне або поелементне ділення. Параметри блоку задаються на двох вкладках – Main і Signal Attributes (див. рис. 2.6).

На вкладці Main визначаються параметри:

1. Number inputs (Кількість входів) - кількість входів блоку, при виконанні операції ділення з двома операндами задається (за замовчуванням), як "* /";

2. Multiplication (Множення) - спосіб ділення (множення): поелементне – Element-wise (. *) і матричне - Matrix (*);

На вкладці Signal Attributes параметри подібні параметрам блоку Add [5].

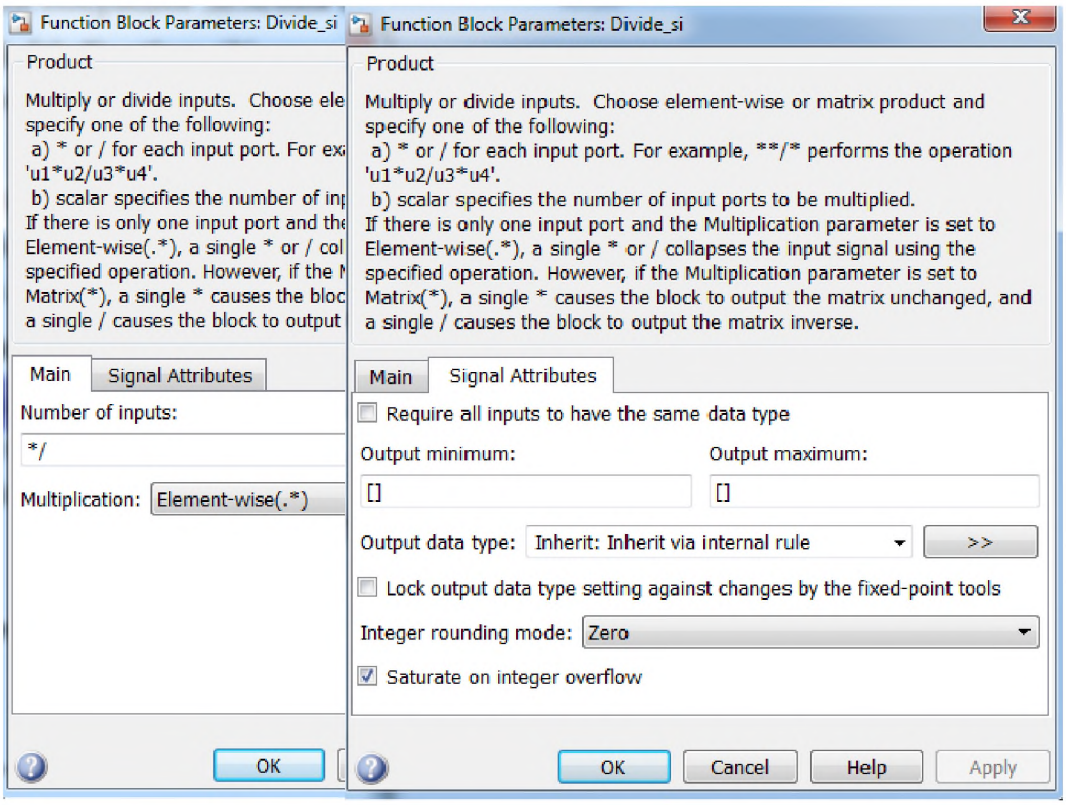

Рисунок 2.6 – Параметри блоку Divide

Для реалізації операцій Add, Subtract, Product, Divide типу даних Signed 32 bits нам знадобляться наступні блоки: Data Type Conversion, Constant, Relational Operator, Compare To Constant, Logic Operator.

Блок Data Type Conversion - перетворює тип даних вхідного сигналу. Для блоку визначені наступні параметри (див. рис. 2.7): аналогічні блоку Add, але додається параметр Input and Output to have equal (Вхідні і вихідні дані повинні бути рівні) - варіанти представлення даних на виході:

1. Real World Value (RWV) - десяткове число;

2. Stored Integer (SI) - десяткове число, розраховане за допомогою бінарного еквіваленту, отриманому після округлення десяткового числа до цілого відповідно до обраного режиму Integer rounding mode [5].

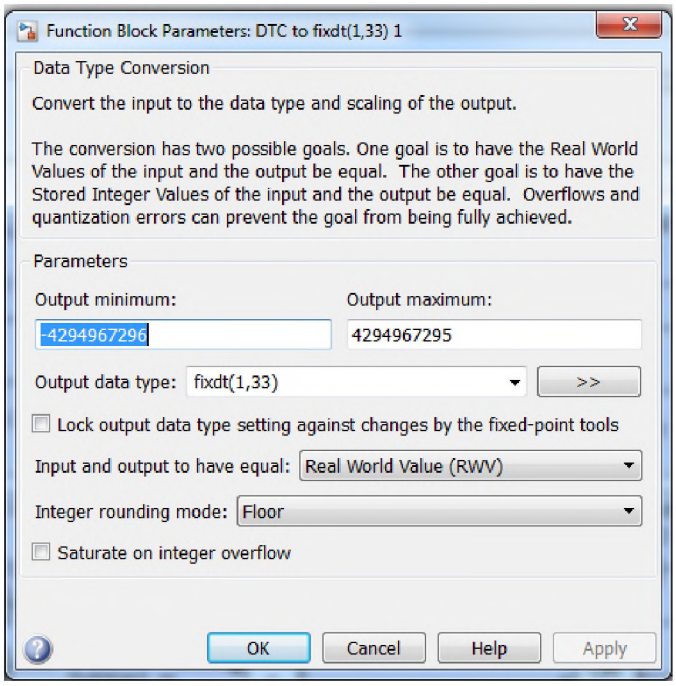

Рисунок 2.7 – Параметри блока Data Type Conversion to fixdt (1,33)

Блок Constant – видає сигнал у вигляді даних – численних або логічних.

Параметри блоку задаються на двох вкладках – Main (Головна) і Signal Attributes (Атрибути сигналу) (див. рис. 2.8):

1. Constant value – Постійна величина.

2. Interpret vector parameters as 1-D. Далі необхідно розглянути векторні параметри як одновимірні при установці відповідного прапорця. Цей параметр є характерним для більшості блоків у бібліотеці Simulink. Значення цієї константи може бути реальним або комплексним числом, яке обчислюється виразом, вектором або матрицею.

3. Sample time (Інтервал часу) [7]

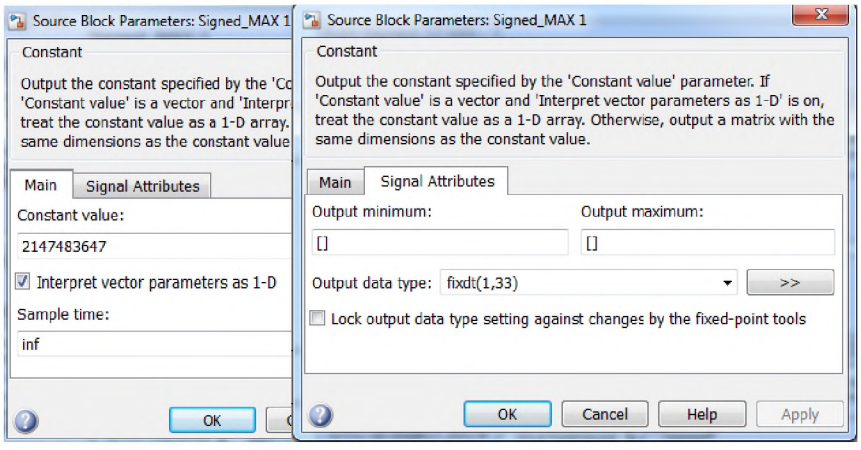

Рисунок 2.8 – Параметри блока Constant

Блок Relational Operator - виконує поелементне порівняння двох операндів.

Параметри блоку задаються на двох вкладках - Main і Data Type (див. рисунок 2.5). На вкладці *Main* визначаються параметри:

1.Relational Operator (Оператор відношення) - символ операції відношення (див. таблицю 2.2). Блок має два входи: верхній - для операнда зліва від операції відношення, і нижній - справа;

Таблиця 2.2 - Символи операцій відношення

Символ операції	Операція
==	Рівно
~=	Не рівно
<	Менше
>	Більше
<=	Менше або рівно
>=	Більше або рівно

2. Enable zero crossing detection (доступно виявлення перетину нуля) - флажок контролю монотонності сигналу на кожному кроці моделювання. Установка флажка (за замовчуванням) дозволяє визначати моменти стрибків сигналу (різкої зміни значення) і автоматично зменшувати крок моделювання при виборі типу зі змінним кроком (Variable-step);

При відкритій вкладці Data type параметри подібні параметрам блоку Add.

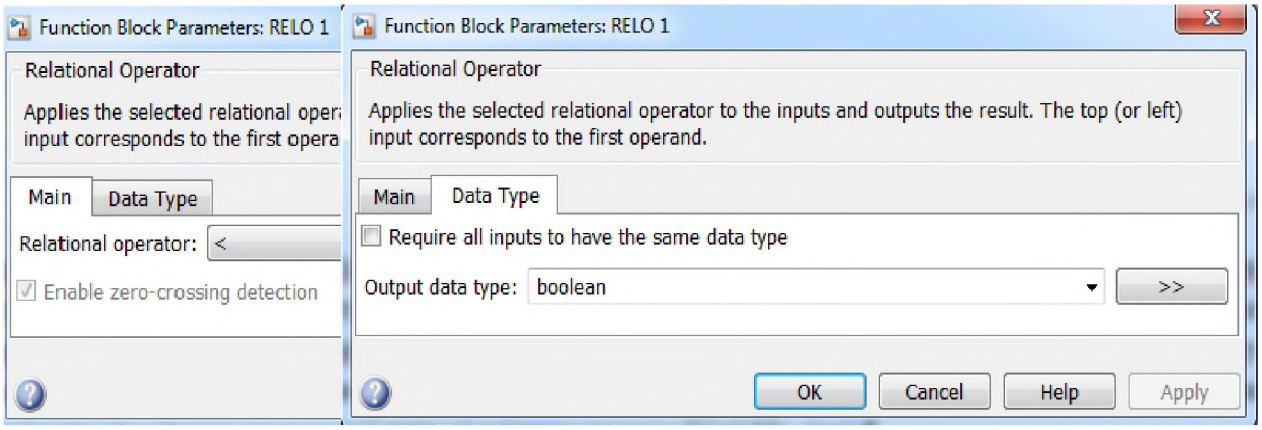

Рисунок 2.9 – Параметри блока Relational Operator

Блок Compare To Constant - виконує порівняння з константою. Блок має наступні параметри (див. рис. 2.10):

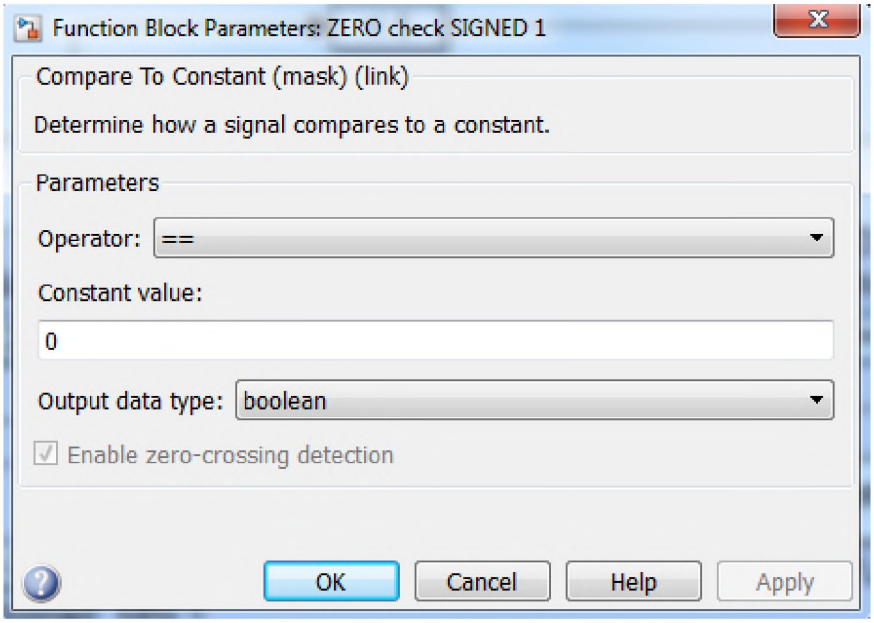

Рисунок 2.10 – Параметри блоку Compare To Constant

1. Operator (Оператор) - символ операції відношення (див. табл. 2.2);

2. Constant Value (Константа) - константа (скаляр, вектор або матриця), з якої порівнюються значення вхідного сигналу і за результатами порівняння видається 1, якщо умова виконується, і 0, якщо не виконується. Якщо константа задана вектором або матрицею, то вхідний сигнал повинен збігатися з нею по вимірності і розміром;

3. Output data type mode (Режим типу даних на виході) - тип даних на виході: uint8 або boolean;

4. Enable zero crossing detection (доступно виявлення перетину нуля).

Блок Logic Operator - виконує логічні операції. Параметри блоку задаються на двох вкладках - Main і Data Type (див. рис. 2.11).

На вкладці Main визначаються параметри:

1. Operator (Оператор) - символ логічної операції (див. табл. 2.3). Операндами можуть бути скаляри, вектори однакової довжини і матриці однакового розміру, для яких логічні операції виконуються поелементно;

2. Number of input ports (Кількість вхідних портів) - кількість входів;

3. Icon shape (Форма піктограми) - вид піктограми: rectangular (прямокутник) або round (коло) .

Таблиця 2.3 – Логічні операції блока Logic Operator

Символ операції	Операція
AND	І - істина, якщо обидва операнди не рівні нулю
OR	АБО - істина, якщо хоча б один з операндів не дорівнює нулю
NAND	І-НЕ - істина, якщо обидва операнди дорівнюють нулю
NOR	АБО-НЕ - істина, якщо хоча б один з операндів дорівнює нулю
XOR	Або - істина, якщо тільки один з операндів дорівнює нулю
NOT	НЕ - всі операнди, нерівні нулю, замінюються нулями, а все рівні нулю - одиницями

На вкладці Data Type параметри подібні параметрам для блоку Add [5].

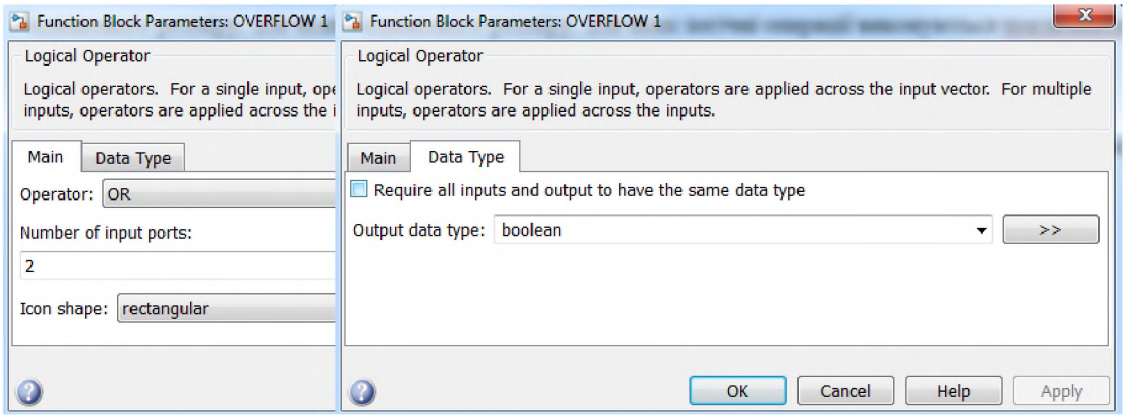

Рисунок 2.11 – Параметри блоку Logic Operator

У загальному вигляді маємо:

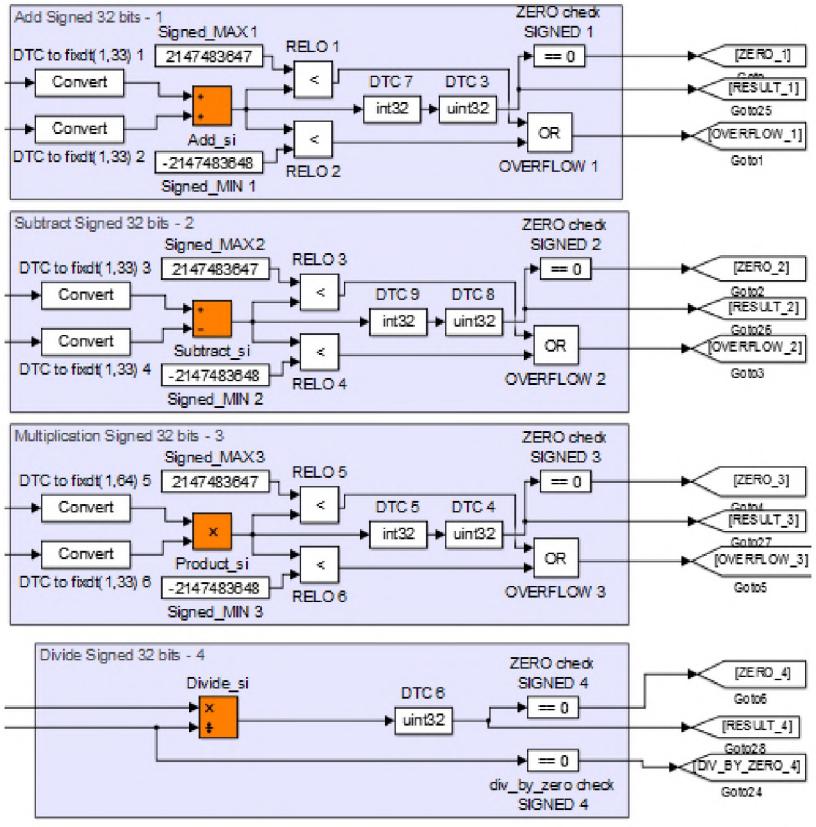

Рисунок 2.12 – Блоки реалізації операцій для типу даних Signed 32 bits

Для реалізації операцій Add, Subtract, Product, Divide типу даних Floating Point 32 bits нам знадобляться такі ж самі блоки, як для типу Signed 32 bits,але крім цього додаються блоки Bit Slice та single_to_32bit.

Блок Bit Slice - повертає поле послідовних бітів з вхідного сигналу (див. рис.2.13).

Параметри:

1. MSB Position - задає позицію біта (починаючи з нуля) найстаршого біта (MSB) поля для вилучення. Довжина слова вихідного сигналу обчислюється як (MSB LSB позиції - Position) + 1.

2. LSB Position - задає позицію біта (починаючи з нуля) найменшого значущого біта (LSB) поля для вилучення [8].

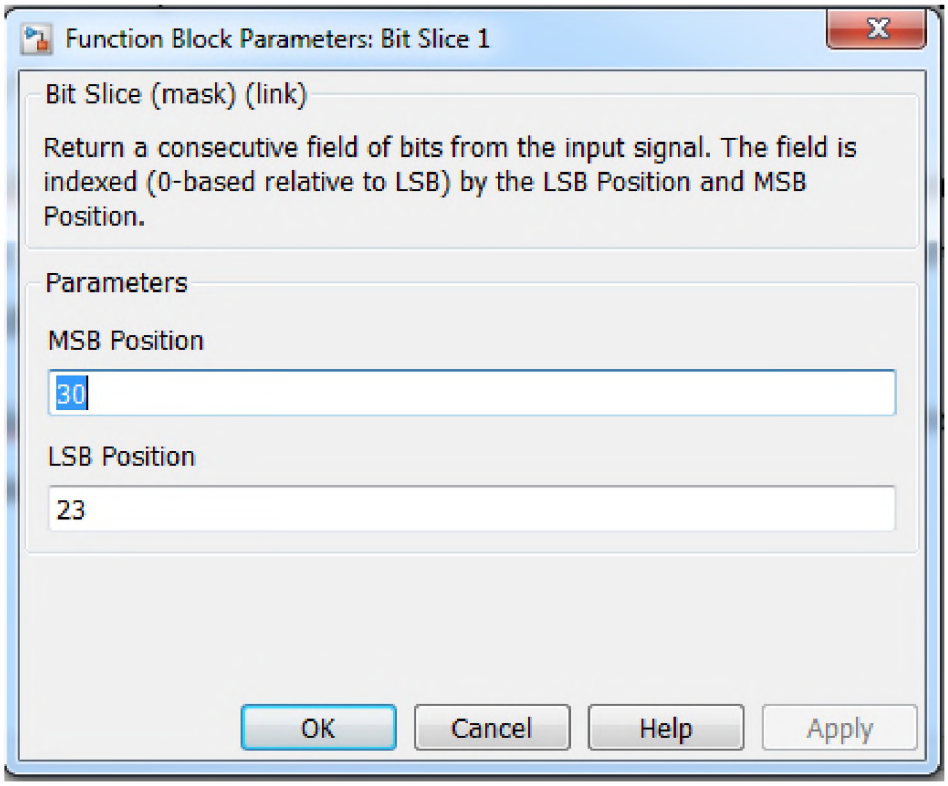

Рисунок 2.13 – Параметри блоку Bit Slice

Блок Single_to_32bit – конвертує число типу single в 32-бітну шину.

Реалізація блоку Add Floating Point показано на рис. 2.14 а реалізація блоку Divide Floating Point 32 на рис. 2.15.

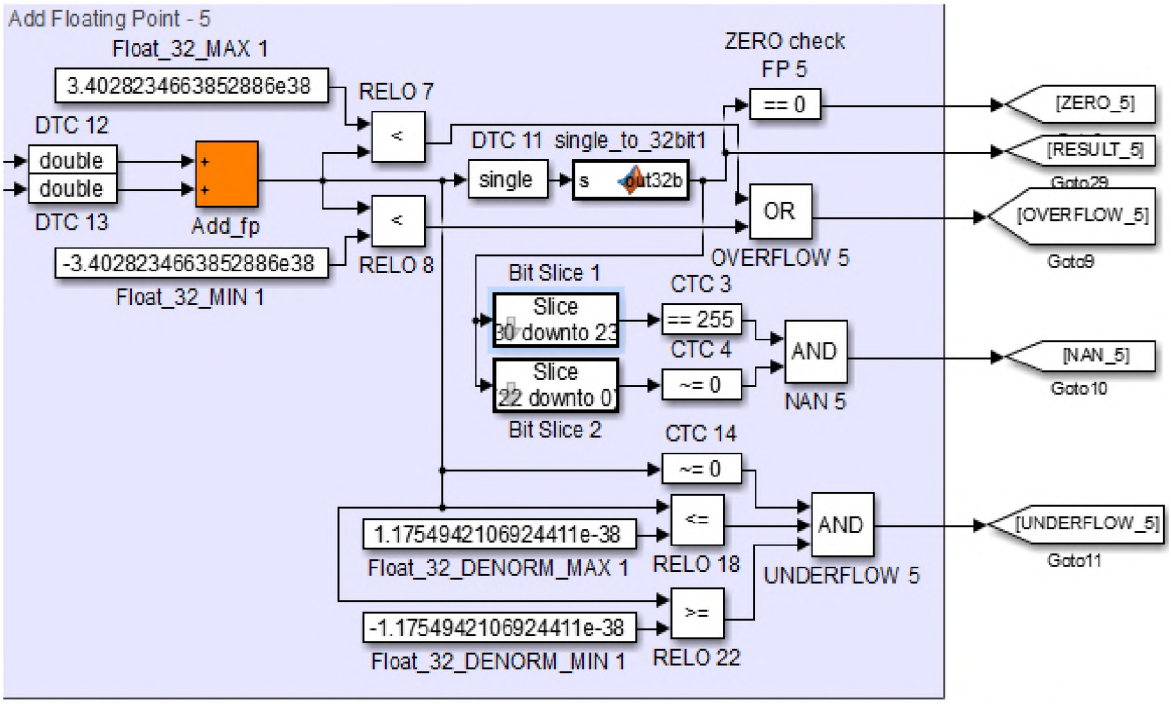

Рисунок 2.14 – Блок Add для типу даних Floating Point 32 bits

Блоки Subtract, Product аналогічні.

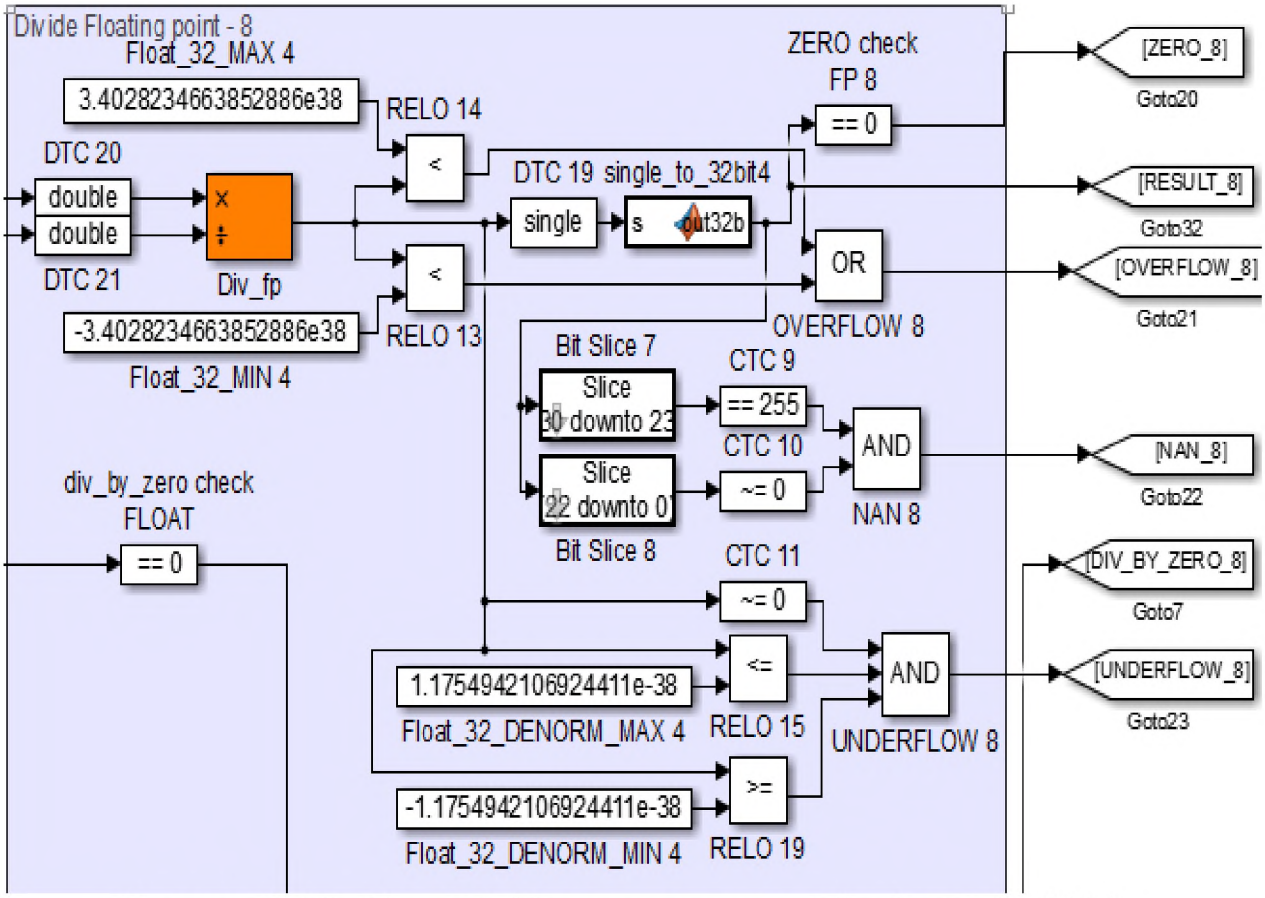

Рисунок 2.15 – Блок Divide для типу даних Floating Point 32 bits

2.5 Складові елементи функціонального блоку

Функціональний блок призначений для обчислення операцій типу даних Floating Point 32 bits. Реалізація даних операцій є аналогічною до реалізації операцій математичного блоку. Єдина відмінність в тому, що на вхід подається лише одне число а також задані інші константи. Перша операція – квадратний корінь числа (див. рис. 2.16).

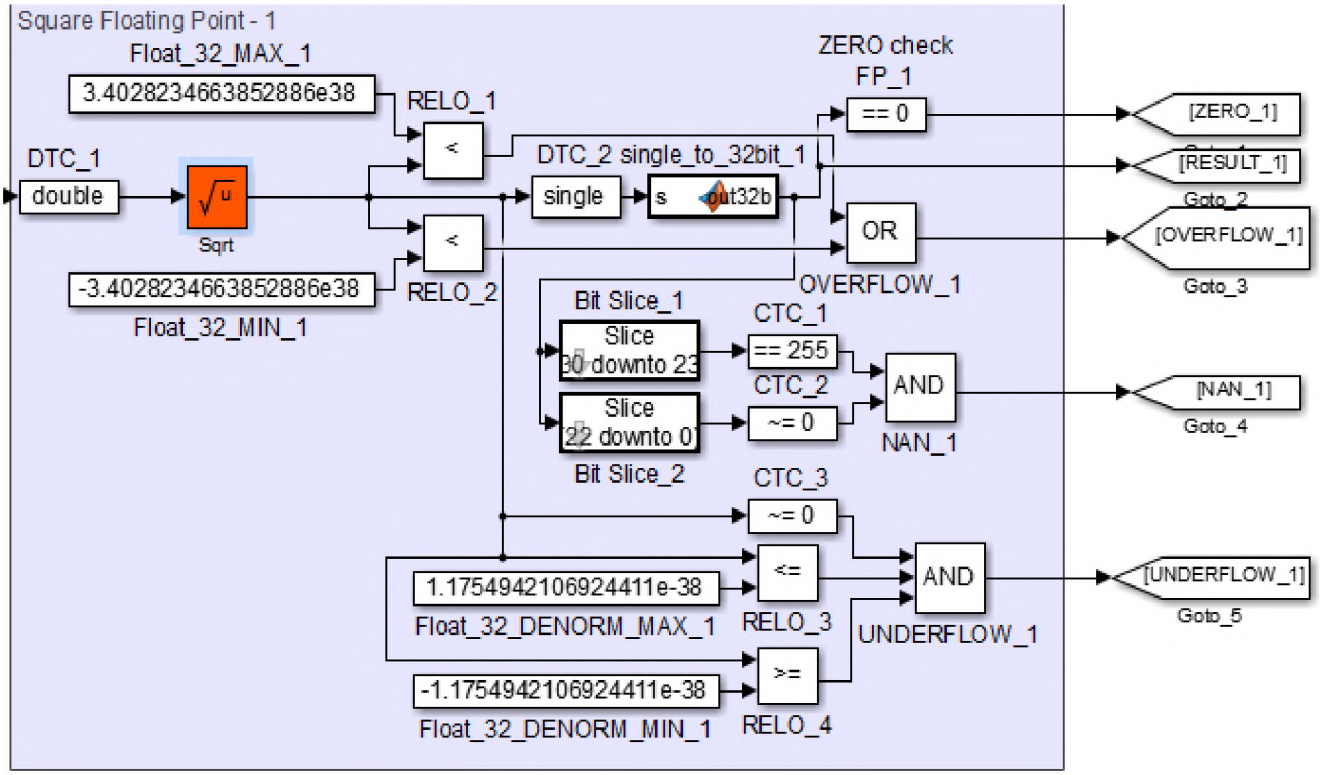

Рисунок 2.16 – Блок Square Floating Point

Блок Abs - обчислює абсолютне значення (модуль) вхідного сигналу. Параметри блоку задаються на двох вкладках - Main і Signal Attributes (див. рис. 2.17).

На вкладці Main визначаються параметри:

1. Enable zero crossing detection - доступно виявлення перетину нуля.

2. Sample time – інтервал часу.

При відкритій вкладці Signal Attributes параметри подібні параметрам блоку Add [5].

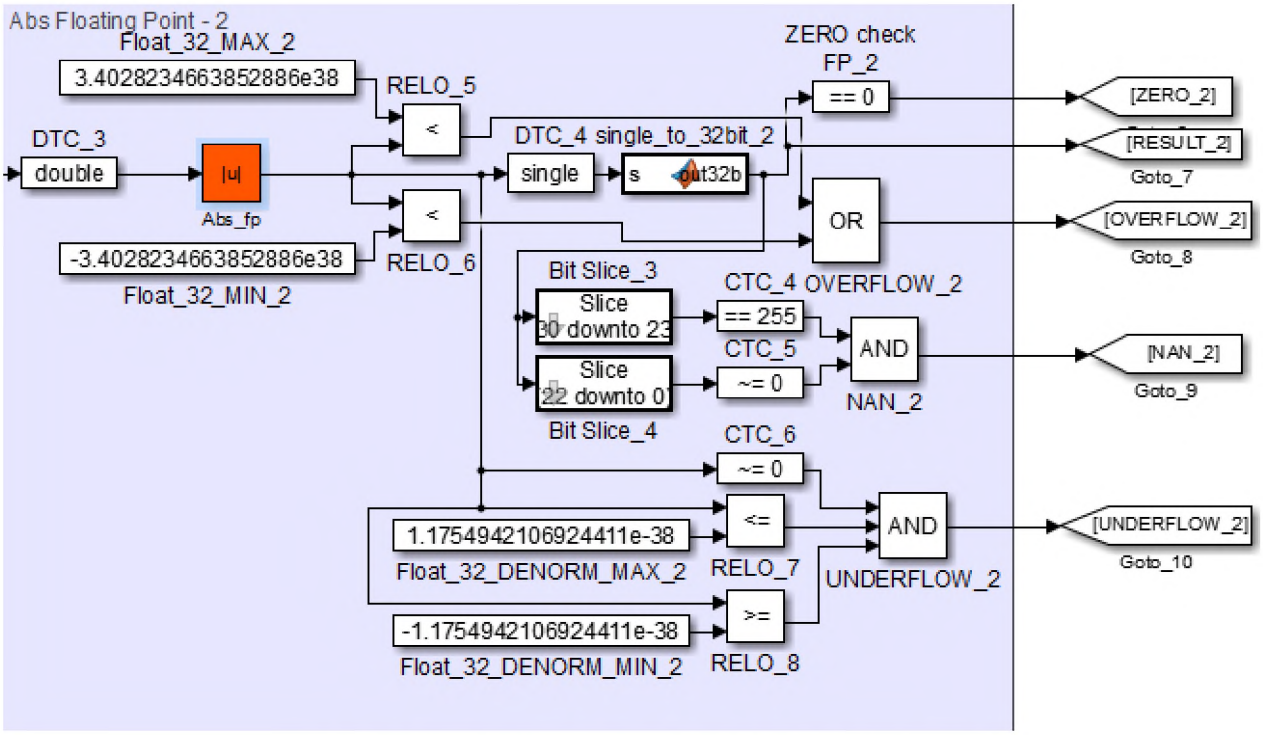

Рисунок 2.17 – Блок Abs Floating Point

Функції sin та cos знаходяться у блоці Trigonometric Function - обчислює значення вхідного сигналу відповідно до зазначеної тригонометричної функції (див. рис. 2.18, 2.19).

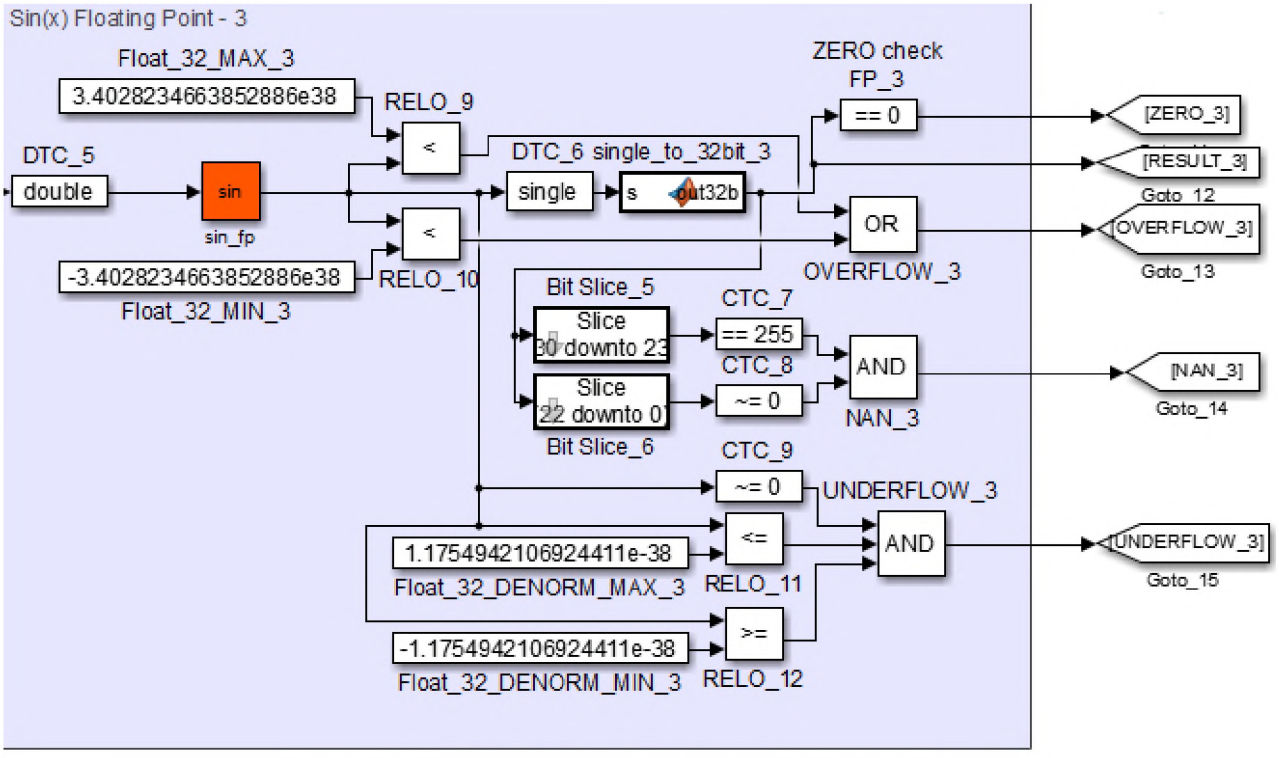

Рисунок 2.18 – Блок sin(x) Floating Point

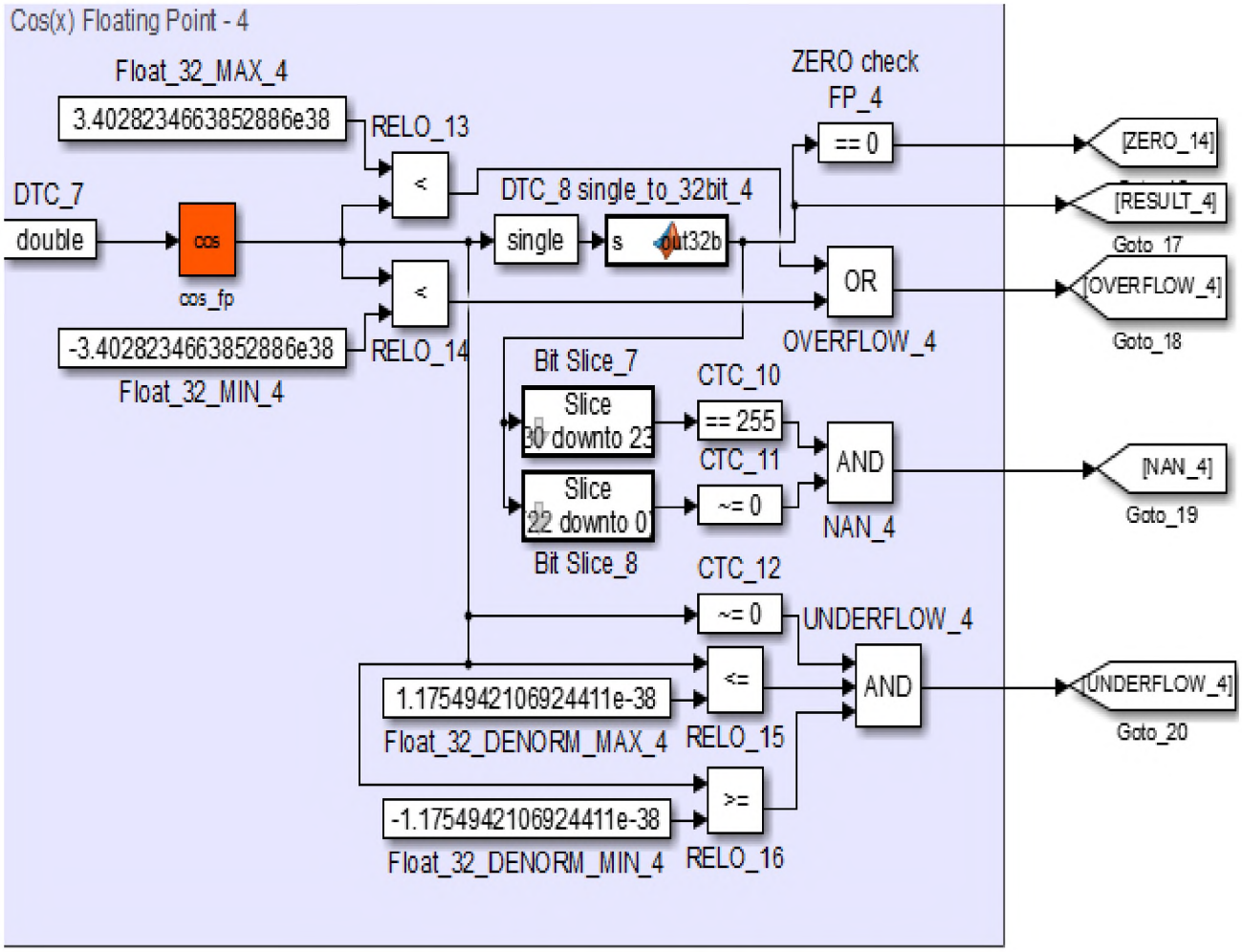

Рисунок 2.19 – Блок cos(x) Floating Point

Реалізація блоків $\ln(x)$, e^x, $\frac{1}{x}$ аналогічно попереднім.

2.6 Опис вихідних портів

Для виконання операцій необхідно так підбирати вхідні дані, щоб усі виняткові ситуації спрацьовували, а саме переповнення (overflow), недозавантаження (underflow), нуль (zero), нечислове значення (nan) і ділення на нуль (div_by_zero).

Переповнення (overflow), коли встановлений прапорець обмеження сигналів цілого типу, повинно відбуватися коректно. Це означає, що результат не повинен виходити за певний діапазон. Якщо результат виходить за цей діапазон, отримуємо значення «+∞» або «-∞» (див. рис. 2.20).

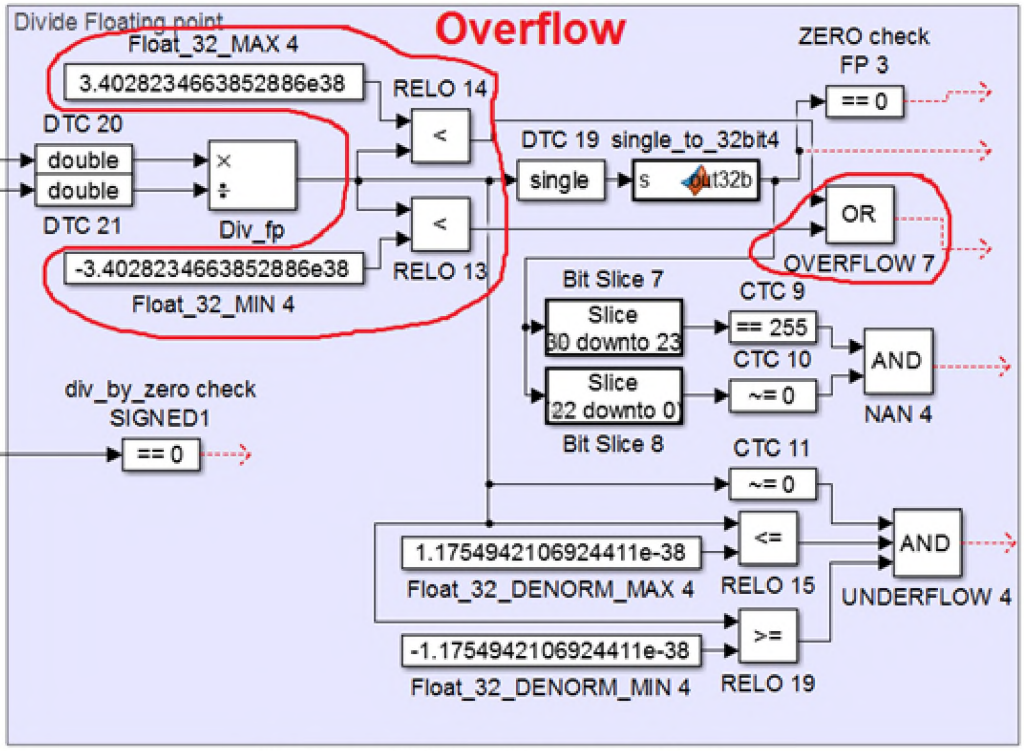

Рисунок 2.20 – Flag «overflow»

Underflow – результат повинен попасти в діапазон денормалізованих чисел.

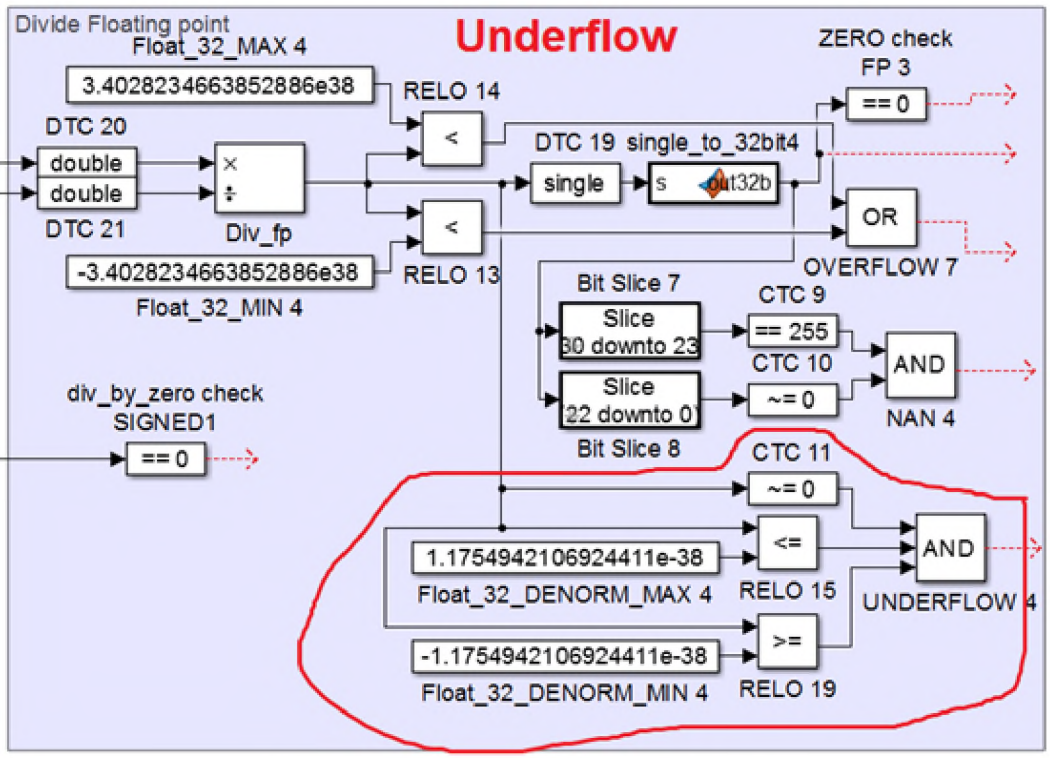

Рисунок 2.21 – Flag «underflow»

Zero -перевіряєм чи дорівнює результат нулю (див. рис. 2.22).

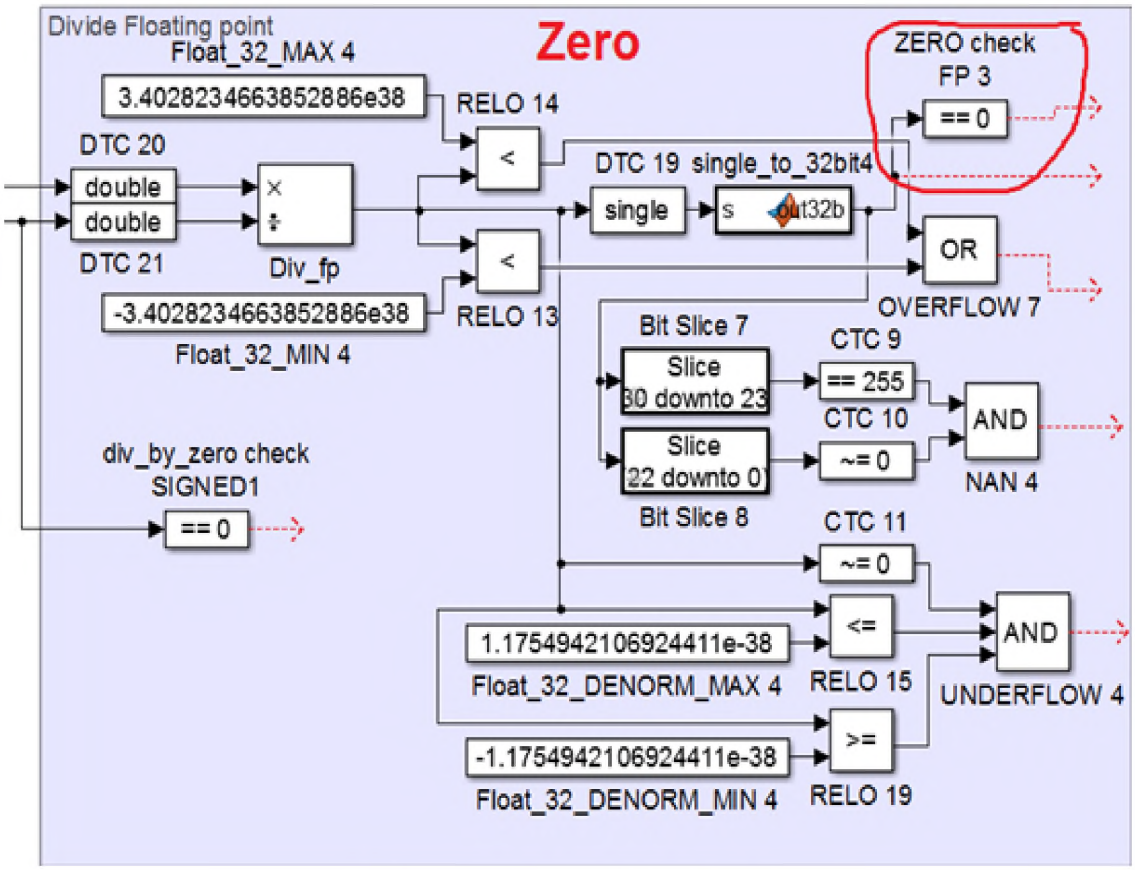

Рисунок 2.22 – Flag «zero»

NaN – представлення числа, придумане для того, щоб арифметична операція могла завжди повернути якесь значення (див. рис. 2.23). У IEEE 754 NaN представлений як число, в якому $E = E_{\max} + 1$, а мантиса не нульова. Будь-яка операція з NaN повертає NaN. При бажанні в мантису можна записувати інформацію, яку програма зможе інтерпретувати. Стандартом це не обумовлено і мантиса найчастіше ігнорується. Як можна отримати NaN? Одним із таких способів:

1. $\infty + (-\infty)$
2. $0 \times \infty$
3. $0/0$, ∞/∞
4. sqrt(x), де x<0

За визначенням NaN ≠ NaN, тому, для перевірки значення змінної потрібно просто порівняти її з собою.

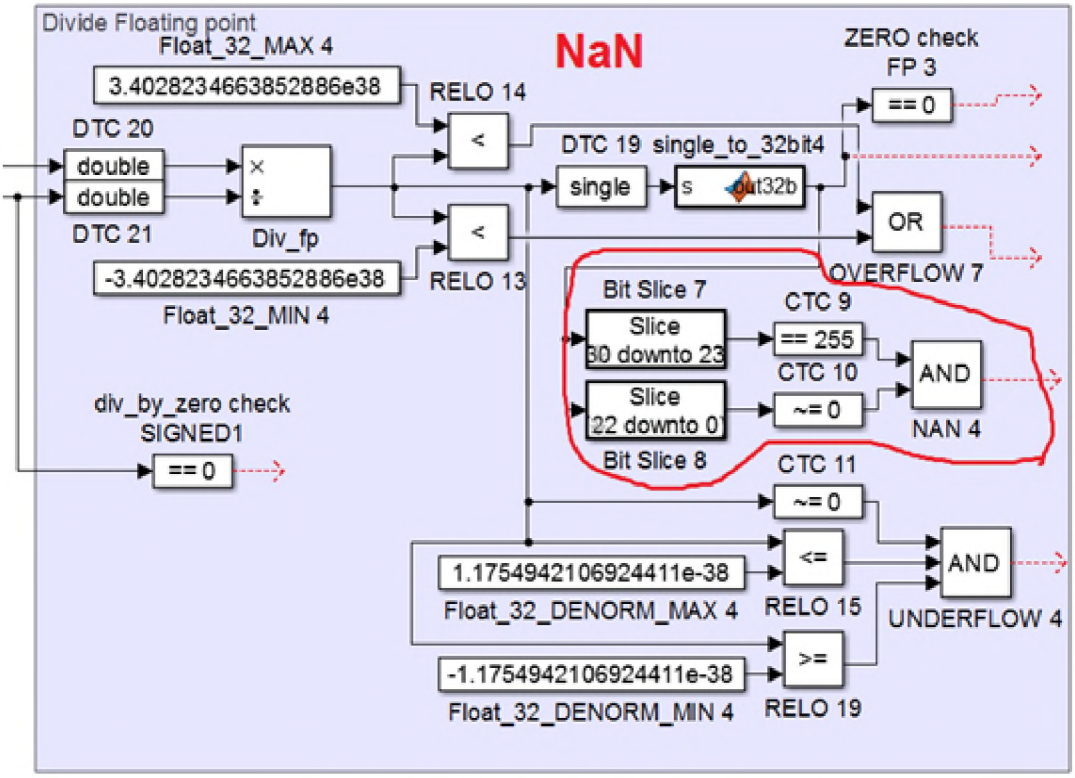

Рисунок 2.23 – Flag «NaN»

Div_by_zero -перевіряєм ділення на нуль (див. рис.2.24). Результатом буде “+ ∞”.

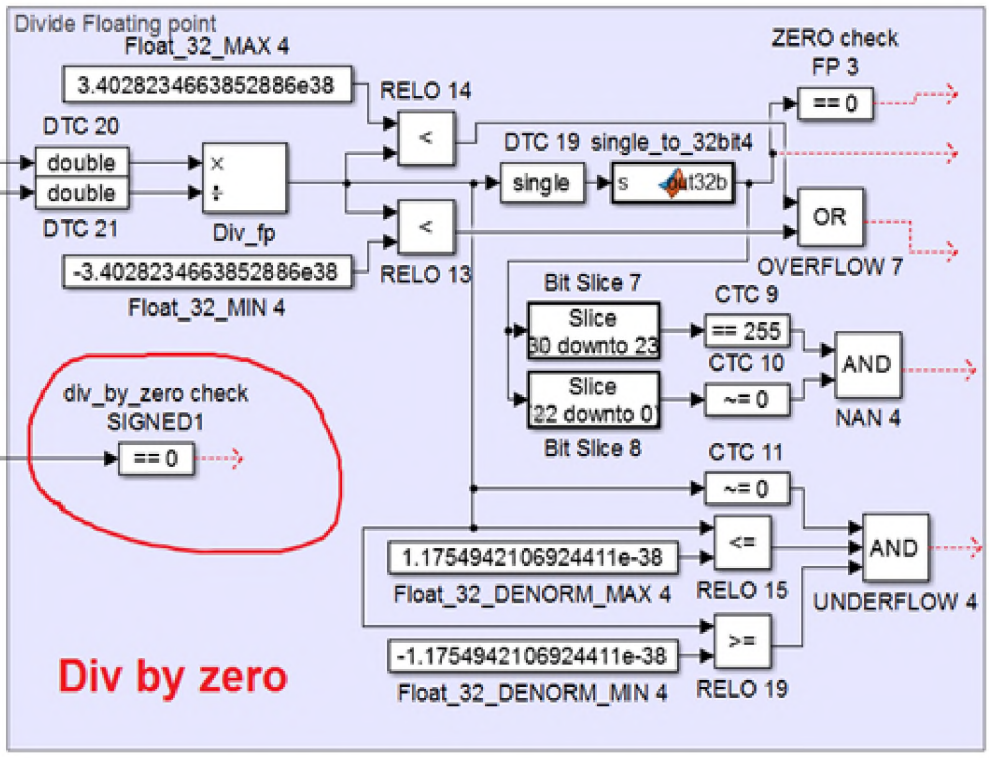

Рисунок 2.24 – Flag «div_by_zero»

Щоб передати сигнал від одного блоку до іншого, достатньо просто з'єднати вихід і вхід за допомогою стрілки (натискання та утримування лівої кнопки миші). Зокрема, сигнал з результатом і флажками діагностики мають бути передані до блоків «Goto». Для кожного такого сигналу призначте унікальне ім'я, яке відповідає порядковому номеру операції, наприклад, «RESULT_1», «ZERO_1», «OVERFLOW_1» – для операції Add_si. За допомогою блоків «From» приєднати відповідні сигнали в блоки типу «Multiport Switch». Представлені в групі блоків Signal Routing (Маршрутизація сигналу)

Блок Goto – безпровідний передатчик сигнала на вхід блока From. У блоку задаються наступні параметри (див. рис. 2.25):

1. Goto Tag – мітка блоку Goto

2. Tag Visibility (область видимості мітки) – список компонентів S-моделі системи, для яких доступний блок з даної міткою:

- local (локальна) – локальна підсистема;

- scoped (всередині) – локальна підсистема і підсистеми нижчого рівня;

- global (глобальна) – будь-яка компонента S-моделі системи.

Для компонентів local і scoped мітка всередині піктограми автоматично у квадратних дужках, а для компонента global відображається без дужок. Одному блоку Goto може відповідати кілька блоків From, список міток яких виводиться в поле Corresponding From blocks (Відповідні блоки From). Посилання refresh (оновити) дозволяє оновити список після вибору іншого мітки блоку Goto у вікні параметрів блоку From.

3. Icon Display (Текст всередині піктограми) - варіанти відображення тексту всередині піктограми:

- Tag – мітка блоку Goto

- Signal name – ім'я сигналу

- Tag and Signal name – мітка блоку Goto і ім'я сигналу.

Блок From – це бездпровідний приймач сигнала з виходу блока Goto. У блоці задаються наступні параметри (див. рис.2.25):

1.Goto Tag - мітка блоку Goto, обрана зі списку. Якщо в списку вказані мітки не всіх блоків Goto, то для його розширення слід вибрати значення <More Tags>. Кнопка Update Tags (Оновити список) дозволяє оновити список при зміні міток блоків Goto.

2. Icon Display - текст всередині піктограми [5].

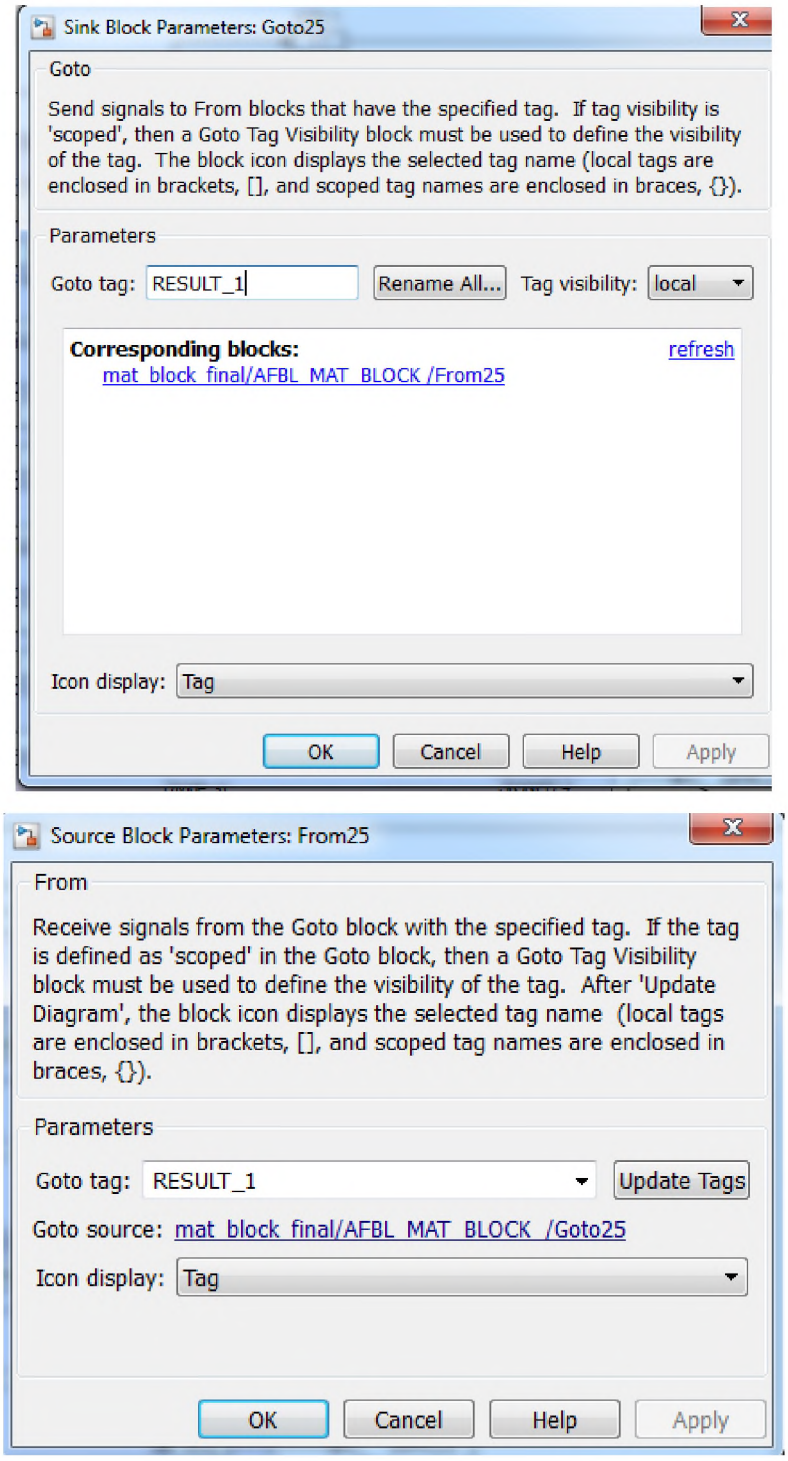

Рисунок 2.25 – Параметри блоків Goto і From і приклад їх спільного використання

Блок Multiport switch - виконує перемикання вхідного сигнала по сигналу керування, що задає номер активного вхідного порту (див. рис. 2.26).

Number of inputs – Кількість входів. Блок багатоходового перемикача Multiport Switch, пропускає на вихід сигнал з того вхідного порту, номер якого дорівнює поточному значенням керуючого сигналу.

Якщо керуючий сигнал не є сигналом цілого типу, то блок Multiport Switch виробляє відкидання дробової частини числа, при цьому в командному вікні Matlab з'являється попередження [5].

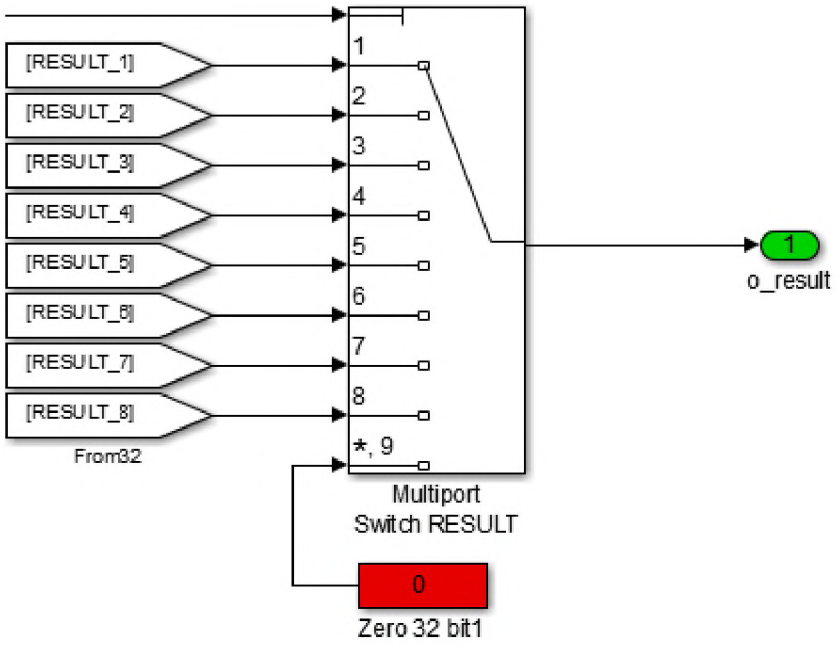

Рисунок 2.26 – Приклад Multiport switch

Блок Zero 32 bit – константа, яка містить 32-бітне число рівне нулю.

Висновки з розділу 2

У другому розділі розглянуто опис моделі з чітким визначенням ролі кожного застосовуваного елемента в блок-схемі. Було проведено аналіз опису вихідних портів. Для успішного поєднання програми Matlab із S-моделлю необхідно мати засоби, які забезпечують: передачу даних між середовищем Matlab та S-моделлю в обидві сторони; запуск процесу моделювання S-моделі з середовища Matlab, а також можливість динамічної зміни параметрів

моделювання і параметрів S-блоків з середовища Matlab; можливість виклику програм Matlab з контексту S-моделі; можливість створення S-блоків, як з використанням готових блоків, так і шляхом використання програм, написаних на мові Matlab (M-мова).

У процесі проектування були створені наступні блоки для дискретного перетворення інформації: «Блок Square Floating Point», а також блок «Abs», який обчислює абсолютне значення (модуль) вхідного сигналу.

РОЗДІЛ 3
ПРИНЦИП РОБОТИ БЛОКІВ

3.1 Опис роботи математичного блоку

Для перевірки роботи кожної операції на вході подаються два операнди: i_1_oprd та i_2_oprd. Ураховуючи розгляд двох типів даних, створюється константа type_of_data, яка призначена для ідентифікації вхідних даних: float=1; signed=0. Порт i_conf – це ціле додатнє число з 5 біт. Параметр конфігурації призначений для вибору потрібної операції. Іншими словами, ми повинні підібрати такі два числа, щоб спрацьовував спочатку один флажок, потім наступні два числа для спрацювання флажка overflow і так далі.

З метою спрощення роботи генеруються тест-кейси. Важливо враховувати, що для кожної операції не всі флажки діагностики перевіряються (див. рис. 3.1). Для типу даних «single» флажок o_underflow і o_NaN не перевіряються, оскільки вони застосовуються до чисел з плаваючою комою. Результат будь-якої операції завжди буде рівний нулю, тому флажок o_zero спрацює для кожної операції. Флажок o_div_by_zero перевіряється тільки для ділення двох чисел. Переповнення можливе для операцій додавання, віднімання та множення.

Формування тест - кейсів

Таблиця очікуваного результату

№ тест - кейса	Набір вхідних даних			Проміжний результат	Отриманий результат							
	i_conf	i_1_oprd	i_2_oprd		o_result	o_mat_edi	o_overflow	o_underflow	o_zero	o_nan	o_div_by_zero	Тип даних
1.	0	x*	x*	0	0	1	0	0	0	0	0	1
2.	10	x*	x*	0	0	1	0	0	0	0	0	0
3.	1	2	2	4	4	0	0	0	0	0	0	0
4.	1	2000000000	2000000000	4000000000	2147483647	0	1	0	0	0	0	0
5.	1	-3	3	0	0	0	0	0	1	0	0	0
6.	2	6	2	4	4	0	0	0	0	0	0	0
7.	2	-2125897000	2106975000	-4232872000	-2147483648	0	1	0	0	0	0	0
8.	2	-203789218	-203789218	0	0	0	0	0	1	0	0	0
9.	3	2	2	4	4	0	0	0	0	0	0	0
10.	3	2118203740	2097558029	4,44305526189 4828e+18	2147483647	0	1	0	0	0	0	0
11.	3	2	0	0	0	0	0	0	1	0	0	0
12.	4	2	2	1	1	0	0	0	0	0	0	0
13.	4	0	1	0	0	0	0	0	1	0	0	0
14.	4	1	0	2147483647	2147483647	0	0	0	0	0	1	0

Рисунок 3.1 – Таблиця тест – кейс для типу даних single 32 bits

Для типу даних floating point ми перевіряєм усі флажки діагностики, маємо:

15.	5	1.5	3.5	5.0	5.0	0	0	0	0	0	0	1
16.	5	3.0e38	3.0e38	6.0e38	Inf	0	1	0	0	0	0	1
17.	5	-1.1e-37	1.10003e-37	-2.993E-42	-2.993E-42	0	0	1	0	0	0	1
18.	5	-25.6	25.6	0	0	0	0	0	1	0	0	1
19.	5	NaN	25.6	NaN	NaN	0	0	0	0	1	0	1
20.	6	3,3	1,3	2,0	2,0	0	0	0	0	0	0	1
21.	6	-3,2e38	3,1e38	-6,3e38	-Inf	0	1	0	0	0	0	1
22.	6	-1,11e-37	-1,0098e-37	-1,00e-38	-1,00e-38	0	0	1	0	0	0	1
23.	6	-2,5	-2,5	0	0	0	0	0	1	0	0	1
24.	6	NaN	1,5	NaN	NaN	0	0	0	0	1	0	1
25.	7	1,1	2,6	2,86	2,86	0	0	0	0	0	0	1
26.	7	2,6e38	3,1e38	8,06e38	Inf	0	1	0	0	0	0	1
27.	7	-1,11e-37	1,00e-0,2	-1,11e-39	-1,11e-39	0	0	1	0	0	0	1
28.	7	0	3,1	0	0	0	0	0	1	0	0	1
29.	7	NaN	2,45	NaN	NaN	0	0	0	0	1	0	1
30.	8	2,6	2,6	1	1	0	0	0	0	0	0	1
31.	8	3,4e+37	1,0e-5	3,4e+42	Inf	0	1	0	0	0	0	1
32.	8	-1,11e-37	1,00e+0,2	-1,11e-39	-1,11e-39	0	0	1	0	0	0	1
33.	8	0	1,4	0	0	0	0	0	1	0	0	1
34.	8	NaN	3,2	NaN	NaN	0	0	0	0	1	0	1
35.	8	1,89	0	+inf	+inf	0	0	0	0	0	1	1

Рисунок 3.2 – Таблиця тест – кейс для типу даних floating point 32 bits

Розглянемо роботу деяких блоків.

Блок Switch - являє собою автоматичний перемикач сигналів. Блок має один вихід і три входи для сигналів u1 (верхній), u2 (середній) і u3 (нижній). Управління між сигналами u1 і u3 здійснюється керуючим сигналом u2: якщо заданий для нього умова виконується, то на вихід надходить сигнал u1, в іншому випадку - сигнал u3. Параметри блоку задаються на двох вкладках – Main і Signal Attributes (див. рис. 3.3).

На вкладці Main визначаються параметри:

1. Criteria for the first input (Критерій для першого входу) - умова для керуючого сигналу u2 на середньому вході.

2. Threshold (Поріг) - граничне значення в умови Criteria for the first input.

3. Enable zero crossing detection - доступно виявлення перетину нуля.

На вкладці Signal Attributes параметри подібні параметрам блоку Add [5].

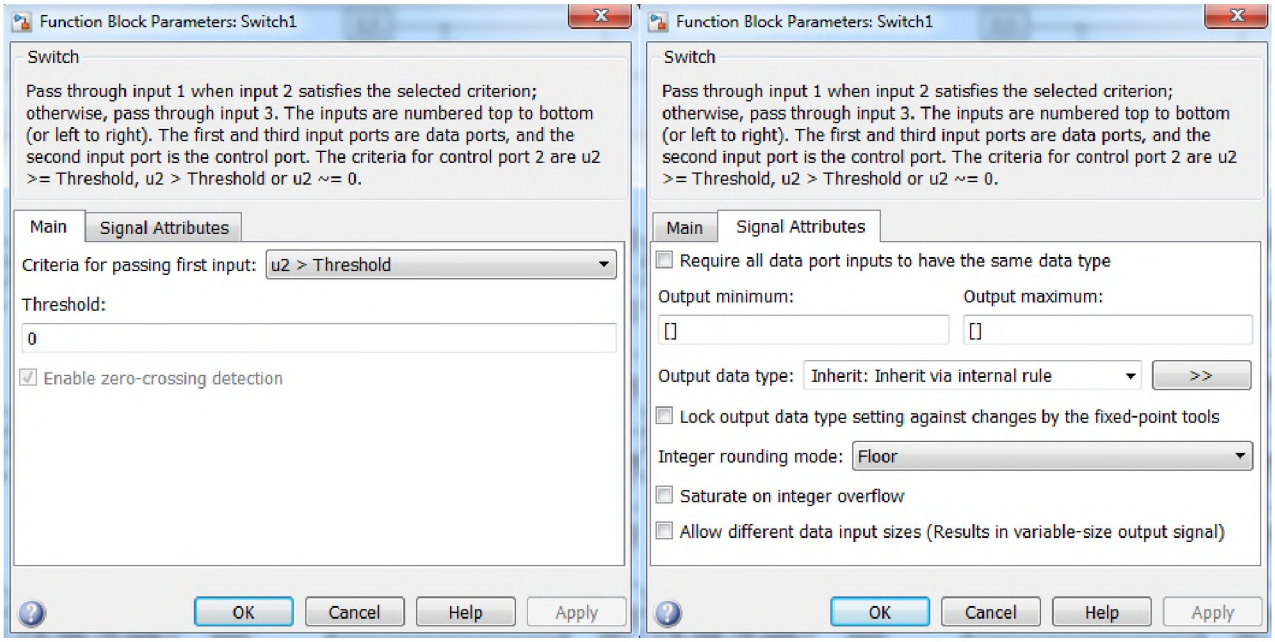

Рисунок 3.3 – Параметри блоку Switch

На рис. 3.4 наведено приклад управління сигналами u1 і u3, відповідно на виходах блоків single_to_32 bit і DTC з параметрами, заданими за замовчуванням. Керуючим є сигнал u2.

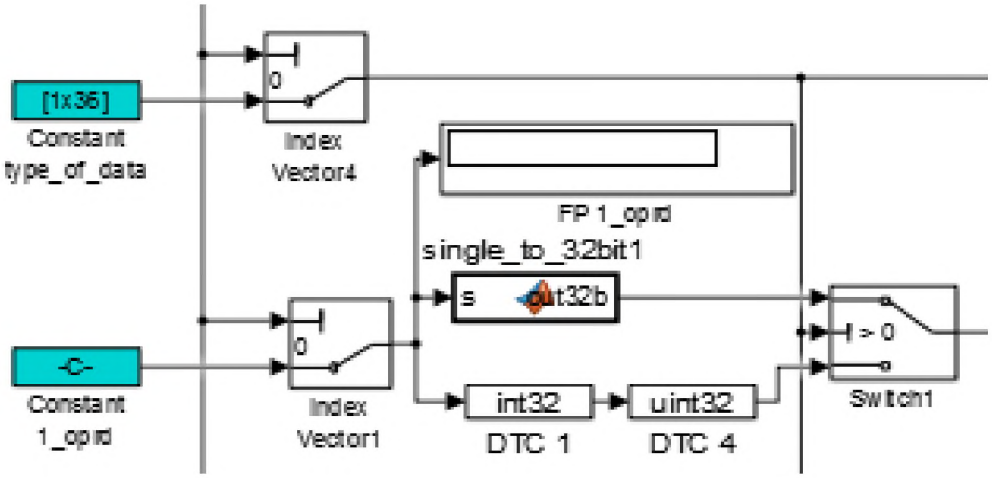

Рисунок 3.4 – Приклад управління сигналами Switch1

Блок Index Vector – вибирає з векторного сигнала елементи, індекси яких відповідають значенням на керуючому порту і відправляє їх на вихідних порт.

Для того, щоб користувач бачив дані, які вводяться і який результат видає програма використовуємо блок Display.

Блок Display – відображає чисельні значення. Блок має наступні параметри (див. рис. 3.5):

1. Format (Формат) – формат виведення чисельних значень (див. табл. 3.1), за замовчуванням – short.

Таблиця 3.1 – Формати виведення чисельних значень в блоці Display

Формат виведення	Призначення
short	Короткий формат з виведенням у звичайній формі
long	Довгий формат з виведенням у звичайній формі
short_e	Короткий формат E з виведенням у нормалізованому формі E
long_e	Довгий формат E з виведенням у нормалізованому формі E
bank	Банківський формат з виведенням у звичайній формі з двома значущими цифрами в дробовій частині
hex (Stored Integer)	Шістнадцятковий еквівалент чисел типу - int, uint або fixdt
binary (Stored Integer)	Двійковий еквівалент чисел типу int, uint або fixdt
decimal (Stored Integer)	Цілочисельний еквівалент чисел типу int, uint або fixed

2.Decimation – коефіцієнт проріджування при виведення чисельних значень сигналу. При Decimation: 1 виводяться всі значення сигналу без проріджування, при Decimation: 2 – кожне друге значення і т. д .

3.Floating display (Плаваючий дисплей) – перехід блоку в плаваючий стан. У плаваючому стані блок не має входу і підключається до виходу блоку клацанням лівої кнопки миші на відповідній лінії зв’язку.

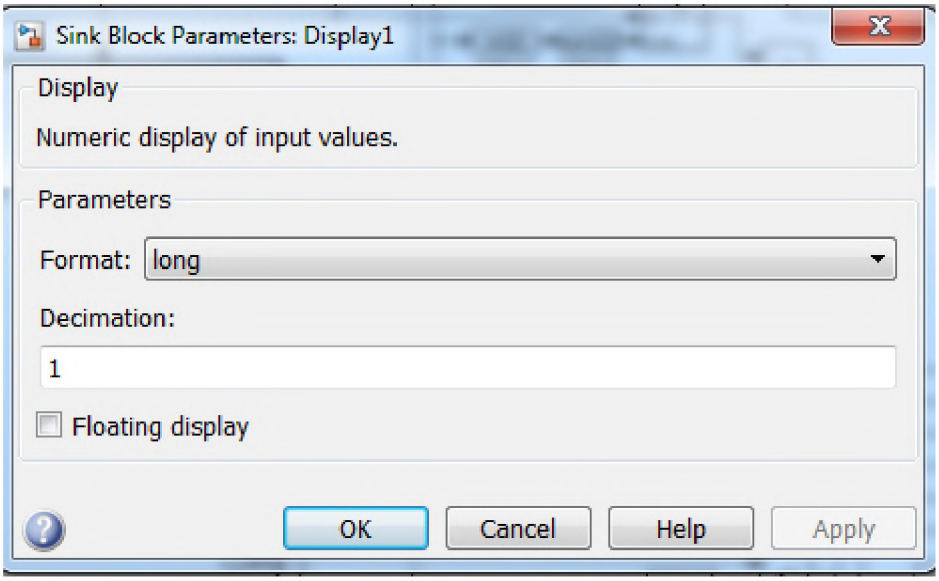

Рисунок 3.5 – Параметри блоку Display

Блок Digital Clock - видає дискретний сигнал поточного часу моделювання з періодом дискретизації (див. рис. 3.6). Єдиний параметр Period задає період дискретизації поточного дискретного часу моделювання.

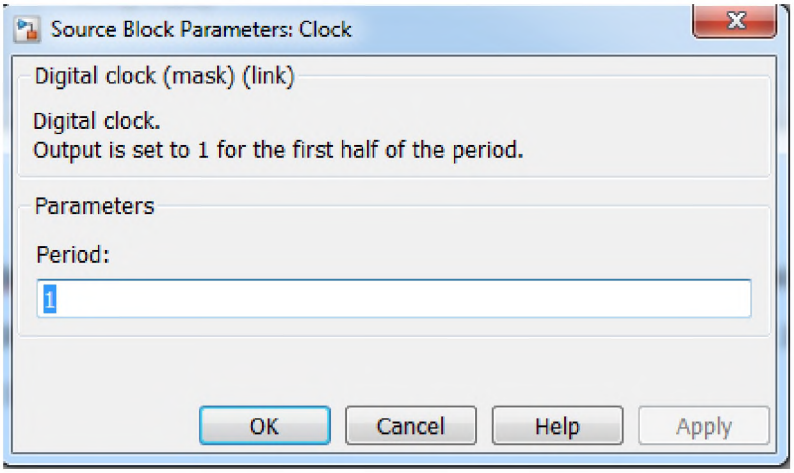

Рисунок 3.6 – Параметри блоку Digital Clock

Блок Counter Limited - формує сигнал на виході лічильника з довільним коефіцієнтом перерахунку і має наступні параметри (див. рис. 3.7):

1.Upper limit (Верхня межа) - максимальне значення N на виході лічильника; значення змінюються від 0 до N з інтервалом часу Sample time.

2 Sample time – інтервал часу [5].

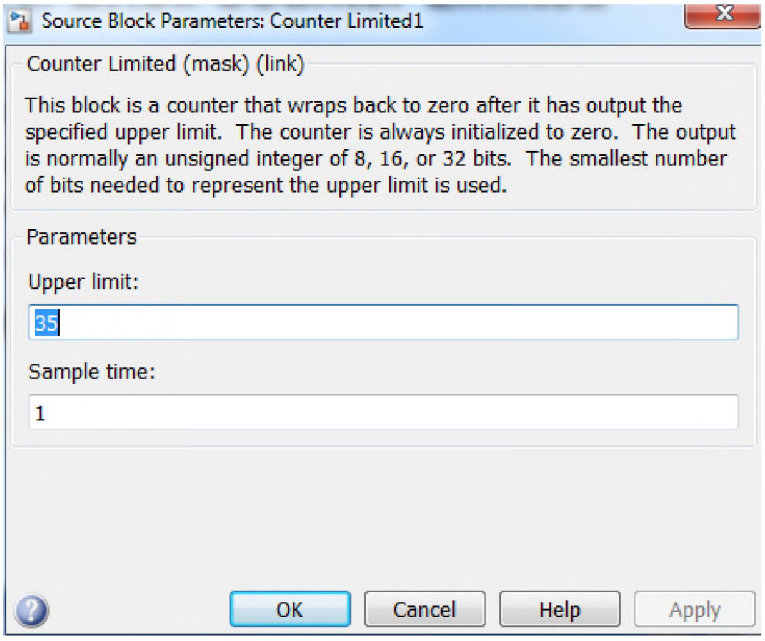

Рисунок 3.7 – Параметри блоку Counter Limited

Отже, для роботи даної програми необхідно подати два числа, де параметр конфігурації вибирає потрібну операцію і при цьому враховується тип даних. Далі, функція перевірки коректності результату порівнюється з певним числом.

Якщо очікуваний результат не збігається з отриманим, програма видає помилку, яка відображається на графіку, наприклад, коли результат потрапляє в одиницю. У випадку, коли результати збігаються, це свідчить про правильність обчислень, і графік відображається на нулі.

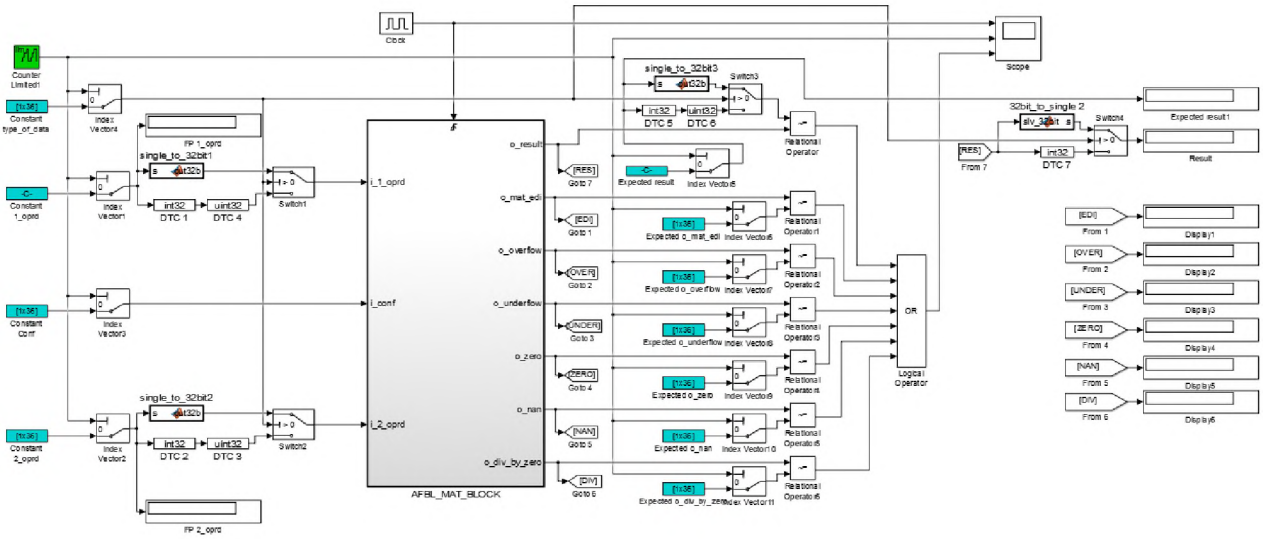

Рисунок 3.8 – Загальний вид програми

Результат програми видно з графіка блок Scope досліджуваних сигналів в процесі моделювання (див. рис. 3.9).

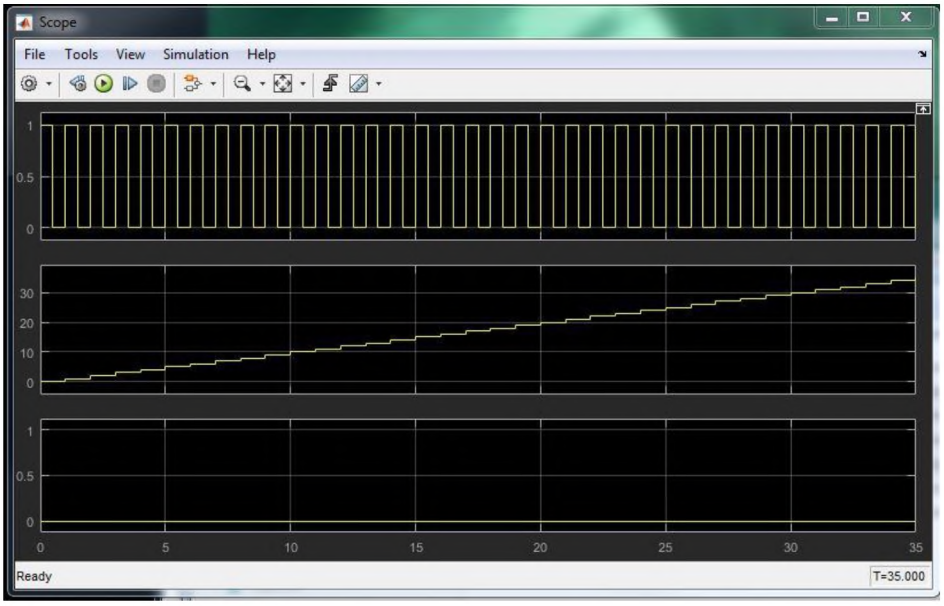

Рисунок 3.9 – Результат програми mat_block

Перший графік відображає тактові часи, визначаючи моменти виконання блоку.

Другий графік представляє номери прикладів, де 0 - початкове значення, і надалі до останнього прикладу.

Третій графік відтворює результат виконання програми. Для отримання позитивного результату графік повинен знаходитися на рівні нуля; в разі потрапляння в одиницю, це свідчить про невідповідність отриманого результату очікуваному. Оскільки вхідні дані були правильно підібрані і результат успішно перевірено на відповідність очікуваному, отримали позитивні результати (див. рис. 3.9).

3.2 Опис роботи функціонального блоку

Принцип роботи функціонального блоку є аналогічним до попередньої програми. Проте, маємо лише 2 вхідних порта: i_data та i_conf. Для кожної операції ми перевірятимем не всі флажки діагностики (див. рис. 3.10).

Формування тест - кейсів

Таблиця очікуваного результату

№ тест - кейса	Набір вхідних даних			Отриманий результат						
	i_conf	i_data	Проміжний результат	o_result	o_funct_edi	o_overflow	o_underflow	o_zero	o_nan	o_div_by_zero
1.	0	0	0	0	1	0	0	0	0	0
2.	15	0	0	0	1	0	0	0	0	0
3.	1	2.2	1.4832397699356	1.4832	0	0	0	0	0	0
4.	1	3.3e37	5.7445628763041e18	5.74456e18	0	0	0	0	0	0
5.	1	1.18e-38	1.0862780080821e-19	1.08628e-19	0	0	0	0	0	0
6.	1	0	0	0	0	0	0	1	0	0
7.	1	-1.2	Nan	Nan	0	0	0	0	1	0
8.	2	-3.3e38	3.3e38	3.3e38	0	0	0	0	0	0
9.	2	3.3e38	3.3e38	3.3e38	0	0	0	0	0	0
10.	2	-Inf	Inf	Inf	0	1	0	0	0	0
11.	2	1.15e-38	1.15e-38	1.15e-38	0	0	1	0	0	0
12.	2	0	0	0	0	0	0	1	0	0
13.	2	Nan	Nan	Nan	0	0	0	0	1	0
14.	3	1.1	0.8912073969841	0.8912073969841	0	0	0	0	0	0

Рисунок 3.10 – Приклад тест – кейсу для func_block

В загальному вигляді маємо:

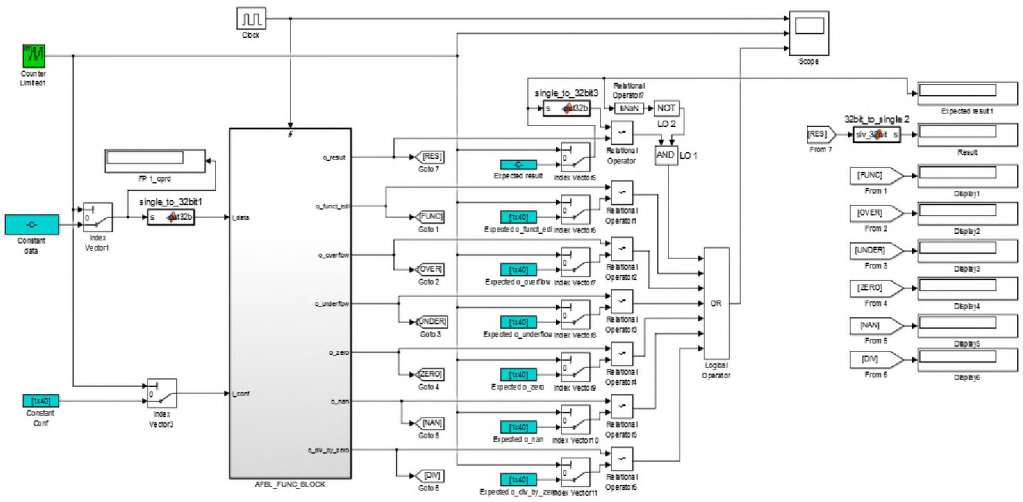

Рисунок 3.11 – Загальний вид програми

Результат також відображається у вигляді графіка, де був отриманий позитивний результат.

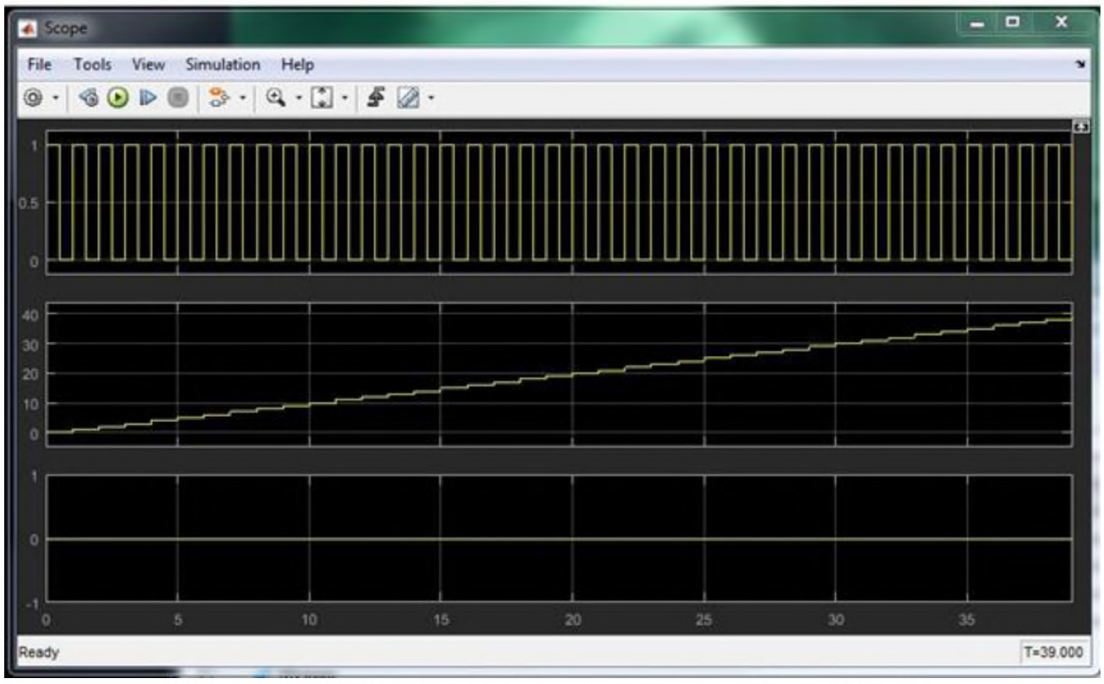

Рисунок 3.12 - Результат програми func _block

Висновки з розділу 3

В третьому розділі здійснено опис роботи блоків. Підібраний набір вхідних даних для перевірки коректності формування результату. Детально виконано опис функціонування розроблених блоків, наведено результати їх виконання.

ВИСНОВКИ

Використання математичної системи MATLAB та інструменту імітаційного моделювання Simulink дозволяє ефективно і швидко розв'язувати складні інженерні завдання при проектуванні систем автоматичного управління для реальних динамічних процесів у різних галузях. Залежно від мети проекту та характеристик об'єкта управління можна використовувати безперервні, нелінійні або дискретні блоки, і їх склад постійно розширюється. Також передбачена можливість створення власних бібліотек. Глибокі знання математичного аналізу, високий рівень інтуїції та вміння використовувати сучасні інструменти комп'ютерної математики гарантують успішне вирішення складних проблем управління, що виникають в сучасній епохі. У даній магістерській роботі були розроблені та проаналізовані два блоки для обчислення ряду операцій та функцій з вхідними та вихідними портами для типів даних signed 32 bits та floating point 32 bits , а саме:

РОЗДІЛ 1. Розглянуто теоретичні аспекти щодо побудови математичної моделі. Зроблено інформаційний огляд програми Matlab Simulink.

РОЗДІЛ 2. Розглянуто опис моделі із зазначенням ролі кожного застосовуваного елемента блок – схеми. Проаналізовано опис вихідних портів.

РОЗДІЛ 3. Здійснено опис роботи блоків. Підібраний набір вхідних даних для перевірки коректності формування результату.

На основі отриманих результатів було побудовано графік, на якому видно, що програма працює коректно.