Міністерство освіти і науки України

ДЕРЖАВНИЙ БІОТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Навчально-науковий інститут «Кібепорт»

Кафедра автоматизації та комп'ютерно-інтегрованих технологій

МОДЕЛЮВАННЯ ЗАСОБІВ АВТОМАТИЗАЦІЇ

Методичні вказівки
до виконання практичних робіт

для здобувачів денної та заочної форм здобуття освіти першого (бакалаврського) рівня вищої освіти,

спеціальності

151 Автоматизація та комп'ютерно-інтегровані технології

(174 Автоматизація.комп'ютерно-інтегровані технології та робототехніка)

Харків
2023

Міністерство освіти і науки України

ДЕРЖАВНИЙ БІОТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

Навчально-науковий інститут «Кібепорт»

Кафедра автоматизації та комп’ютерно-інтегрованих технологій

МОДЕЛЮВАННЯ ЗАСОБІВ АВТОМАТИЗАЦІЇ

Методичні вказівки до виконання
практичних робіт

для здобувачів денної та заочної форм здобуття освіти першого (бакалаврського) рівня вищої освіти, спеціальності
151 Автоматизація та комп’ютерно-інтегровані технології
(174 Автоматизація.комп’ютерно-інтегровані технології та робототехніка)

Затверджено рішенням Науково-технічної ради ННІ «Кіберпорт»
Протокол № 2 від 24.10.2023 р.

Харків
2023

УДК 510:621.9

Т 41

Схвалено на засіданні кафедри

автоматизації та комп'ютерно-інтегрованих технологій Протокол № 1 від 30.08.2023 р.

Моделювання засобів автоматизації: метод. вказівки до виконання практ. робіт для здобувачів першого (бакалаврського) рівня вищої освіти денної та заоч. форм здобуття освіти спец. 151 АКІТ (174 АКІТР); Харків. Державний біотехнологічний університет ; уклад.: Ю.А.Нечитайло, – Харків : [б. в.], 2023.– 112 с.

Методичні вказівки включають 8 практичних робіт. Матеріал розкриває сутність моделювання засобів автоматизації. Майбутні фахівці повинні володіти навичками моделювання засобів автоматизації в середовищі Simulink пакета MATLAB.

Видання призначене здобувачам першого (бакалаврського) рівня вищої освіти денної та заочної форм здобуття освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології (174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка).

Рецензенти:

О.А. Лєвтєров, д-р техн. наук, с.н.с., доцент кафедри управління та організації діяльності у сфері цивільного захисту Національного університету цівільного захисту України (НУЦЗУ);

Т.В. Плугіна, канд. техн. наук, доц. кафедри автоматизації та комп'ютерно-інтегрованих технологій Харківського національного автомобільно-дорожнього університету (ХНАДУ).

Відповідальна за випуск (зав. каф.)**: К.В.Демченко,** канд.техн. наук

ЗМІСТ

ВСТУП

Моделювання засобів автоматизації – це предмет, що вивчає методи, прийоми й засоби автоматизації процесів моделювання та їх застосування у практичній діяльності людини.

Предмет вивчення навчальної дисципліни - формування теоретичних знань та практичних навичок з моделювання засобів автоматизації.

Моделювання - це метод вивчення явища, об'єкта чи дії шляхом створення його уявної чи реалістичної моделі.

Математичне моделювання є одним із засобів дослідження поведінки реальних об’єктів та систем керування.

Дисципліна передбачає ознайомлення з загальними поняттями моделювання засобів автоматизації, класифікацією засобів, способами їхнього представлення.

Мета дисципліни:

- вивчення методів опису, аналізу та побудови математичних моделей систем автоматизації;

- надбання знань з алгоритмізації та програмної реалізації математичних моделей;

- набуття навичок використання сучасних прикладних пакетів та програм моделювання складних систем.

В результаті вивчення дисципліни здобувачи повинні:

знати:

- сучасні програмні продукти, призначені для моделювання складних систем;

- головні принципи моделювання складних систем;

- класичні моделі та методи моделювання;

- можливості, що пропонують новітні комп'ютерні та інформаційні технології для вирішення задач моделювання систем та процесів;

вміти:

- застосовувати набуті знання до моделювання складних систем;

- проводити імітаційні експерименти з моделями;

- працювати з сучасними програмними продуктами, призначеними для моделювання складних систем.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Моделювання (в широкому сенсі) - основний метод досліджень у всіх областях знань і науково обгрунтований метод оцінок характеристик складних систем, що використовується для прийняття рішень в різних сферах інженерної діяльності.

Моделлю (лат. Modulus - міра) називається об'єкт-заступник, який в певних умовах може заміняти об'єкт-оригінал, відтворюючи властивості оригіналу, які цікавлять дослідника.

Заміщення одного об'єкта іншим з метою отримання інформації про найважливіші властивості об'єкта-оригіналу за допомогою об'єкта-моделі називається **моделюванням.**

Моделювання - процес дослідження реальної системи, що включає:

- побудову моделі,

- вивчення властивостей моделі,

- перенесення отриманих відомостей на модельовану систему.

По відношенню до моделі дослідник є експериментатором (експеримент проводиться не з реальним об'єктом, а з його моделлю).

Функції моделювання - опис, пояснення і прогнозування поведінки реальної системи.

Типові цілі моделювання:

- пошук оптимальних або близьких до оптимальних рішень,

- оцінка ефективності рішень,

- визначення властивостей системи (чутливості до зміни значень характеристик та ін.),

- встановлення взаємозв'язків між характеристиками системи та ін.

Модель є цільовим відображенням оригіналу (створюється під поставлену задачу і повинна відображати властивості об'єкта, що цікавлять дослідника з точки зору вирішення цього завдання). Один і той же об'єкт-оригінал може мати безліч моделей, побудованих у відповідності з різними цілями дослідження.

Модель називається адекватною об'єкту, якщо результати моделювання підтверджуються і можуть служити основою для прогнозування процесів, що протікають в досліджуваних об'єктах. Адекватність моделі залежить від мети моделювання і прийнятих критеріїв.

Системний підхід в моделюванні систем

Класичний (індуктивний) підхід

- розглядає систему шляхом переходу від часткового до загального;

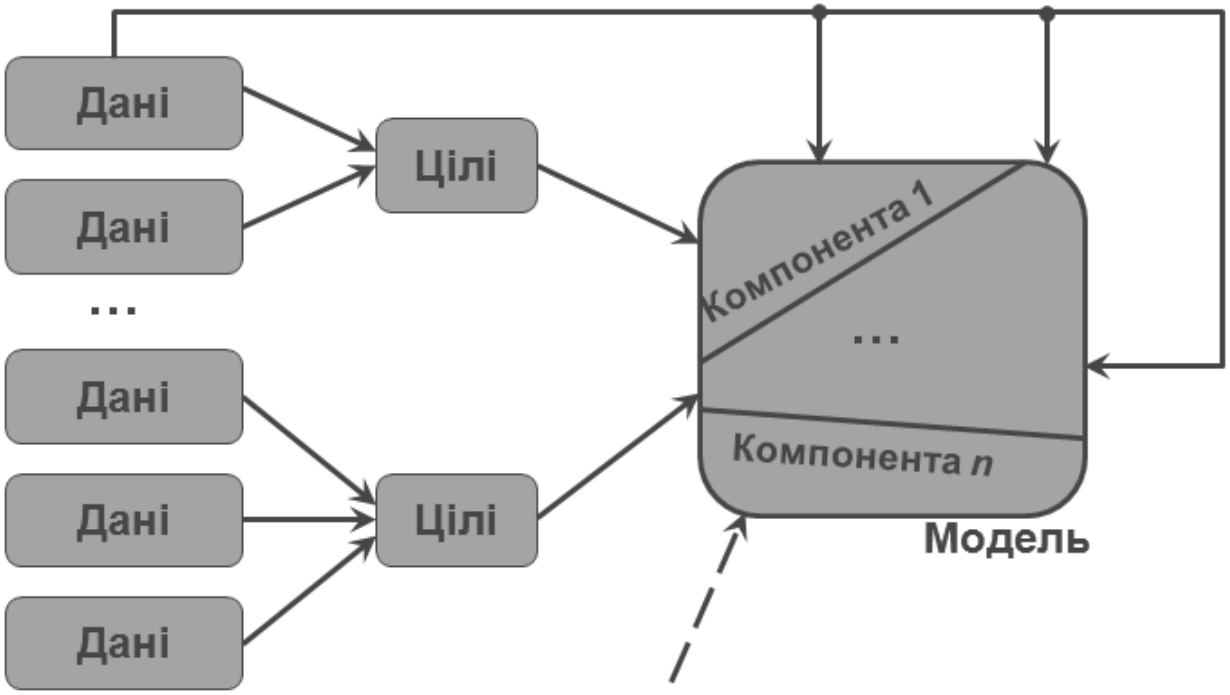
- синтезує (конструює) систему шляхом злиття її компонентів, що розробляються окремо.

Системний підхід передбачає послідовний перехід від загального до конкретного:

- в основі розгляду лежить мета;

- досліджуваний об'єкт виділяється з навколишнього середовища.

Синтез моделі на основі класичного (індуктивного) підходу



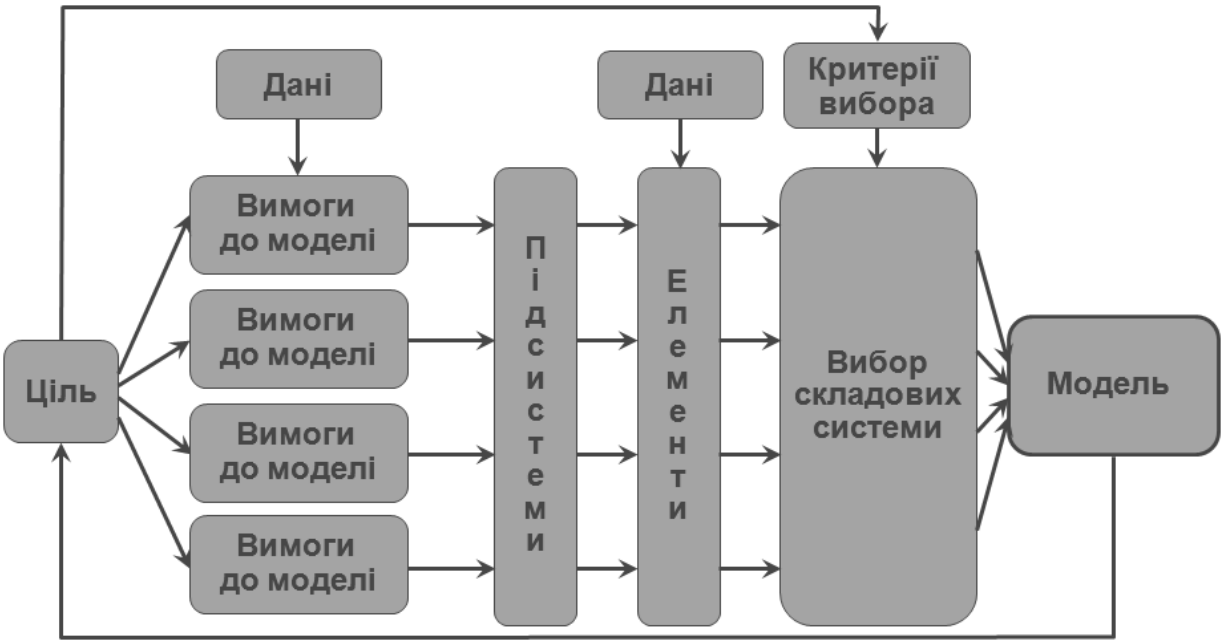
Реальний об'єкт, що підлягає моделюванню, розбивається на окремі підсистеми. За окремою сукупністю вихідних даних ставиться мета моделювання окремої сторони функціонування системи; на базі цієї мети формується деяка компонента майбутньої моделі.

Сукупність компонент об'єднується в модель.

Окремі компоненти підсумовуються в єдину модель; кожна з компонент вирішує свої власні завдання й ізольована від інших частин моделі.

Такий підхід можна використовувати для реалізації порівняно простих моделей, в яких можливо взаємно незалежний розгляд окремих сторін функціонування реального об'єкта.

Синтез моделі на основі системного підходу



На основі вихідних даних (з аналізу реальної системи), обмежень (накладаються зверху або виходячи з можливості реалізації) і цілі функціонування - вихідні вимоги до моделі системи.

На базі вимог формуються орієнтовно деякі підсистеми, елементи і здійснюється вибір складових системи (з використанням спеціальних критеріїв вибору).

При системному підході до моделювання систем:

- чітке визначення мети моделювання;

- важливо визначення структури системи (сукупності зв'язків між елементами системи, що відображають їх взаємодію).

При моделюванні необхідно забезпечити максимальну ефективність моделі системи. Ефективність, зазвичай, визначається як деяка різниця між якимись показниками цінності результатів, отриманих у результаті експлуатації моделі, і тими витратами, які були вкладені у її розробку і створення.

Стадії розробки моделей. На базі системного підходу може бути запропонована і деяка послідовність розробки моделей, коли виділяють дві основні стадії проектування: макропроекування і мікропроектування.

На стадії макропроекування на основі даних про реальну систему **S** і зовнішнє середовище **E** будується модель зовнішнього середовища, виявляються ресурси та обмеження для побудови моделі чи системи, вибирається модель системи й критерії, що дозволяють оцінити адекватність моделі **M** реальної системи **S**. Побудувавши модель системи та модель зовнішнього середовища, на основі критерію ефективності функціонування системи в процесі моделювання вибирають оптимальну стратегію управління, що дозволяє реалізувати можливості моделі з відтворення окремих сторін функціонування реальної системи **S.**

Стадія мікропроектування в значній мірі залежить від конкретного типу обраної моделі. У разі імітаційної моделі необхідно забезпечити створення інформаційного, математичного, технічного та програмного забезпечення системи моделювання. На цій стадії можна встановити основні характеристики створеної моделі, оцінити час роботи з нею і витрати ресурсів для отримання заданої якості відповідності моделі процесу функціонування системи **S.**

Незалежно від типу моделі **M** при її побудові необхідно керуватися принципами системного підходу: 1) пропорційно-послідовне просування по етапам і напрямкам створення моделі; 2) узгодження інформаційних, ресурсних, надійностних та інших характеристик; 3) правильне співвідношення окремих рівнів ієрархії в системі моделювання; 4) цілісність окремих відокремлених стадій побудови моделі.

Модель **M** повинна відповідати заданій меті її створення, тому окремі частини повинні компонуватися взаємно, виходячи з єдиної системної задачі. Мета може бути сформульована якісно, тоді вона буде володіти більшою змістовністю і триваліший час може відображати об'єктивні можливості даної системи моделювання. При кількісному формулюванні мети виникає цільова функція, яка точно відображає найбільш істотні фактори, що впливають на досягнення мети.

Побудова моделі відноситься до числа системних задач, при вирішенні яких синтезують розв’язок на базі величезного числа вихідних даних, на основі пропозицій великих колективів фахівців. Використання системного підходу в цих умовах дозволяє не тільки побудувати модель реального об'єкта, але і на базі цієї моделі вибрати необхідну кількість керуючої інформації в реальній системі, оцінити показники її функціонування і тим самим на базі моделювання знайти найбільш ефективний варіант побудови і вигідний режим функціонування реальної системи S.

ПРАКТИЧНЕ ЗАНЯТТЯ № 1

Загальні принципи моделювання в програмі *Simulink* пакета *MATLAB*

Мета роботи: ознайомитися із загальними принципами моделювання в програмі *Simulink* пакета *MATLAB.*

1.1 Опис програми

Програма *Simulink* є додатком до пакету *MATLAB*. При моделюванні з використанням *Simulink* реалізується принцип візуального програмування, відповідно до якого, користувач на екрані з бібліотеки стандартних блоків створює модель пристрою і здійснює розрахунки. При цьому, на відміну від класичних способів моделювання, користувачеві не потрібно досконально вивчати мову програмування і чисельні методи математики, а достатньо загальних знань потрібних при роботі на комп'ютері і, природно, знань тієї предметної області в якій він працює.

Simulink є достатньо самостійним інструментом *MATLAB* і при роботі з ним зовсім не потрібно знати сам *MATLAB* та інші його додатки. З іншого боку доступ до функцій *MATLAB* та іншим його інструментам залишається відкритим і їх можна використовувати в *Simulink*. Частина входять до складу пакетів має інструменти, вбудовувані в *Simulink* (наприклад, *LTI-Viewer* додатки *Control System Toolbox* – пакета для розробки систем управління). Є також додаткові бібліотеки блоків для різних областей застосування (наприклад, *Power System Blockset* – моделювання електротехнічних пристроїв, *Digital Signal Processing Blockset* – набір блоків для розробки цифрових пристроїв і т.ін.).

При роботі з *Simulink* користувач має можливість модернізувати бібліотечні блоки, створювати свої власні, а також складати нові бібліотеки блоків.

При моделюванні користувач може вибирати метод рішення диференціальних рівнянь, а також спосіб зміни модельного часу (з фіксованим або змінним кроком). У ході моделювання є можливість стежити за процесами, що відбуваються в системі. Для цього використовуються спеціальні пристрої спостереження, входять до складу бібліотеки *Simulink*. Результати моделювання можуть бути представлені у вигляді графіків або таблиць.

Перевага *Simulink* полягає також у тому, що він дозволяє поповнювати бібліотеки блоків за допомогою підпрограм написаних як на мові *MATLAB*, так і на мовах *C* + +, *Fortran* та *Ada*.

1.2 Порядок запуску програми *Simulink*

Для запуску програми необхідно попередньо запустити пакет *MATLAB*. Основне вікно пакета *MATLAB* показано на рисунку 1.1. Там же показана підказка, з'являється у вікні при наведенні покажчика миші на ярлик *Simulink* в панелі інструментів.

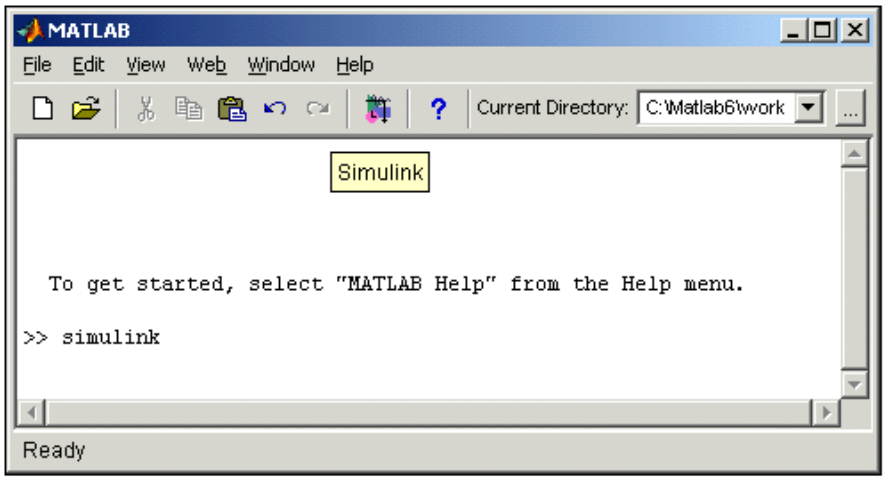


Рисунок 1.1 – Основне вікно програми *MATLAB*

Після відкриття основного вікна програми *MATLAB* потрібно запустити програму *Simulink*. Це можна зробити одним із трьох способів:

– натиснути кнопку (*Simulink*) на панелі інструментів командного вікна *MATLAB*;

– в командному рядку головного вікна *MATLAB* надрукувати *Simulink* і натиснути клавішу *Enter* на клавіатурі;

– виконати команду *Open…* в меню *File* і відкрити файл моделі (*mdl* – файл).

Останній варіант зручно використовувати для запуску вже готової і налагодженої моделі, коли потрібно лише провести розрахунки і не потрібно додавати нові блоки в модель. Використання першого і другого способів призводить до відкриття вікна оглядача розділів бібліотеки *Simulink* (рисунок 1.2).

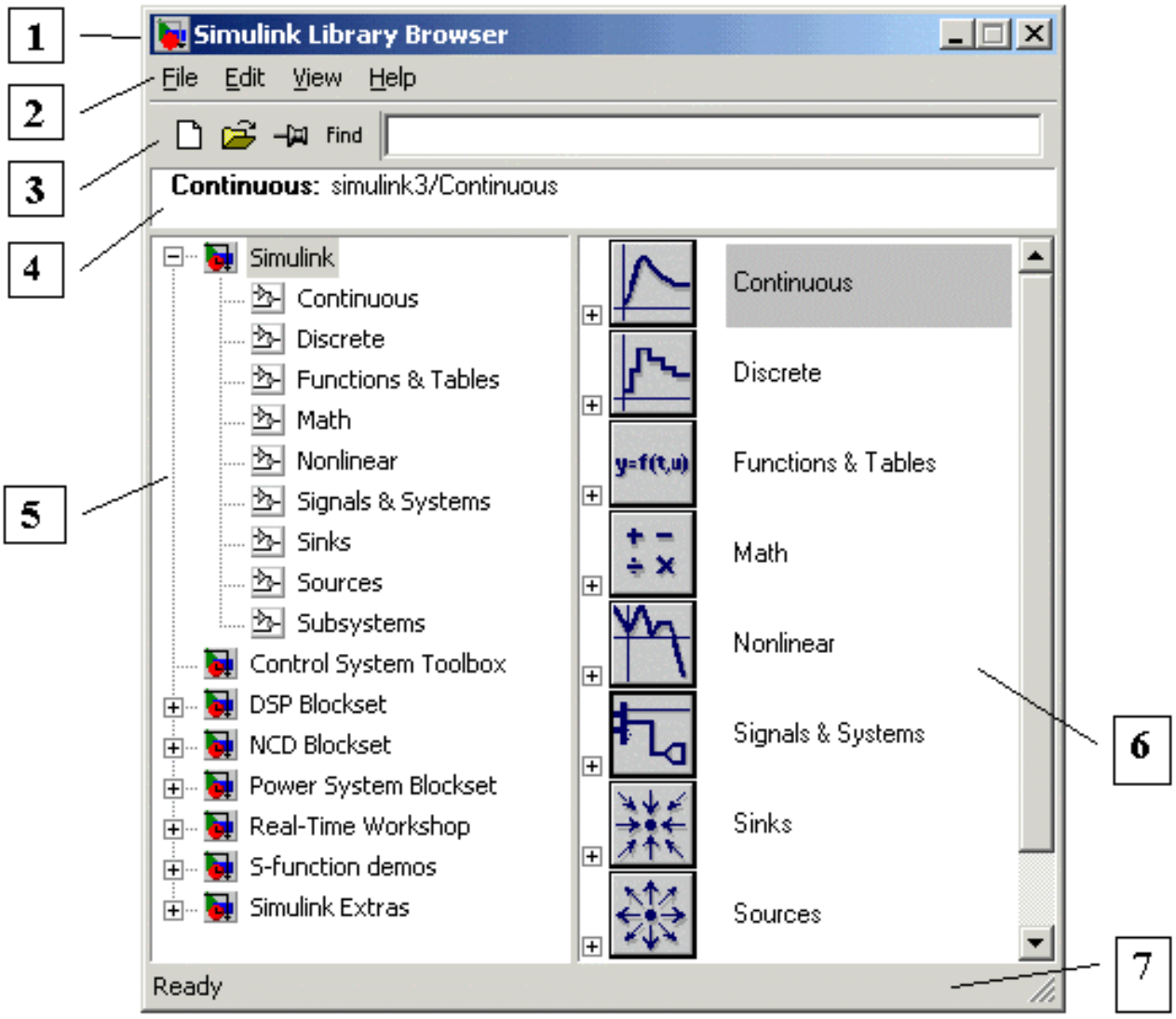


Рисунок 1.2 – Вікно оглядача розділів бібліотеки *Simulink*

1.3 Огляд основних розділів бібліотеки *Simulink*

Вікно оглядача бібліотеки блоків містить наступні елементи (рисунок 1.2):

– заголовок, з назвою вікна – *Simulink Library Browser*;

– меню, з командами *File, Edit, View, Help*;

– панель інструментів, з ярликами найбільш часто використовуваних команд;

– вікно коментаря для виведення пояснюючого повідомлення про вибраний блоці;

– список розділів бібліотеки, реалізований у вигляді дерева;

– вікно вмісту розділу бібліотеки (список вкладених розділів бібліотеки або блоків);

– рядок стану, що містить підказку по виконуваному дії.

На рисунку 1.2 виділена основна бібліотека *Simulink* (в лівій частині вікна) і показані її розділи (в правій частині вікна).

Бібліотека *Simulink* містить наступні основні розділи:

– *Continuous* – лінійні блоки;

– *Discrete* – дискретні блоки;

– *Functions & Tables* – функції та таблиці;

– *Math* – блоки математичних операцій;

– *Nonlinear* – нелінійні блоки;

– *Signals & Systems* – сигнали та системи;

– *Sinks* – реєструючі пристрої;

– Sources – джерела сигналів і впливів;

– Subsystems – блоки підсистем.

Список розділів бібліотеки *Simulink* представлений у вигляді дерева, і правила роботи з ним є загальними для списків такого виду:

– піктограма згорнутого вузла дерева містить символ "+", а піктограма розгорнутого містить символ "-";

– для того щоб розгорнути або згорнути вузол дерева, досить клацнути на його піктограмі лівою клавішею миші (ЛКМ).

При виборі відповідного розділу бібліотеки в правій частині вікна відображається його вміст (рисунок 1.3).

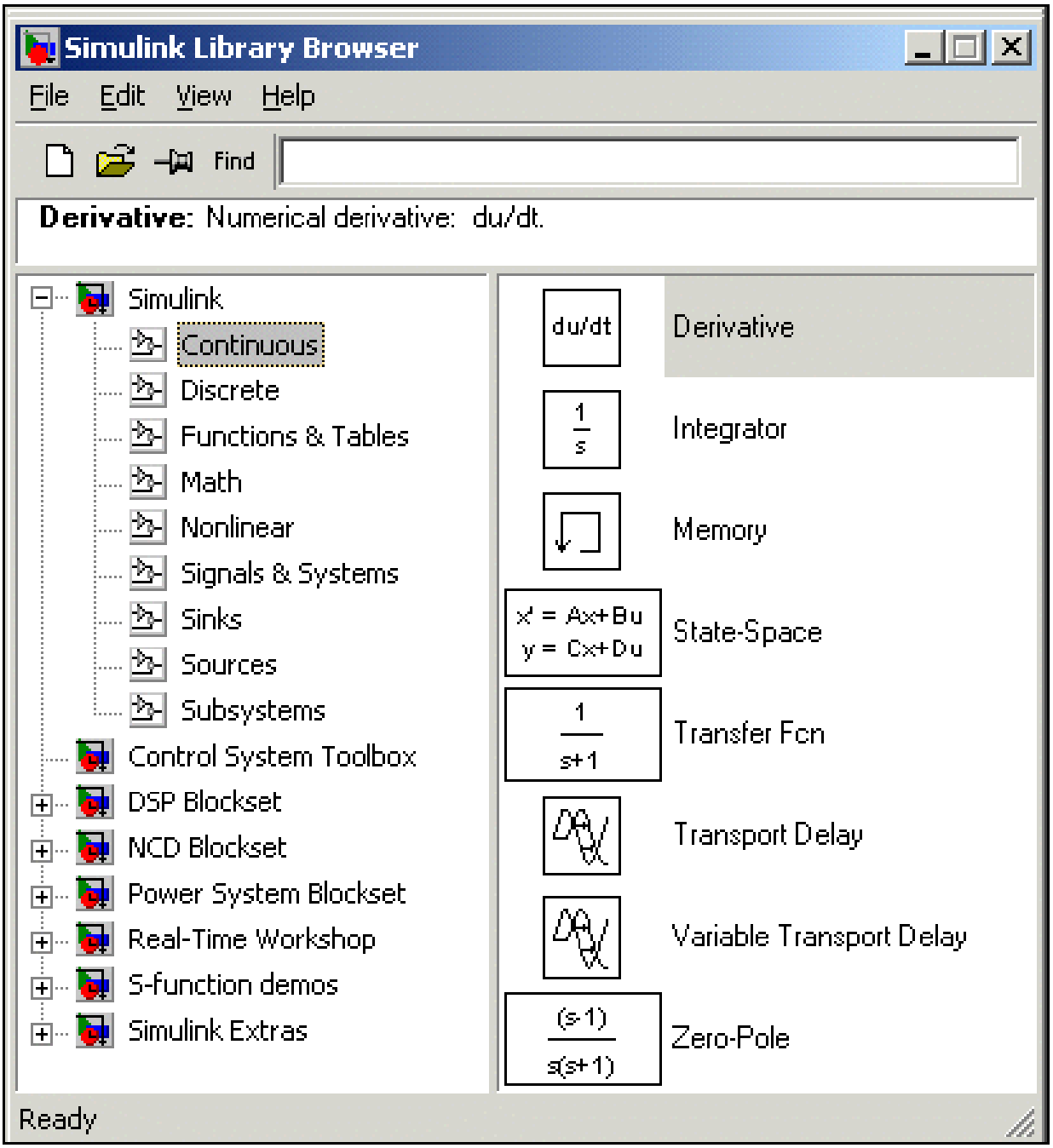


Рисунок 1.3 – Вікно оглядача з набором блоків розділу бібліотеки

Для роботи з вікном використовуються команди зібрані в меню. Меню оглядача бібліотек містить наступні пункти:

– *File* (Файл) – робота з файлами бібліотек;

– *Edit* (Редагування) – додавання блоків і їх пошук (по назві);

– *View* (Вид) – управління показом елементів інтерфейсу;

– *Help* (Довідка) – висновок вікна довідки по оглядачеві бібліотек.

Для роботи з оглядачем можна також використовувати кнопки на панелі інструментів (рисунок 1.4).

Рисунок 1.4 – Панель інструментів оглядача розділів бібліотек

Кнопки панелі інструментів мають наступне призначення:

– 1– створити нову *S*-модель (відкрити нове вікно моделі);

– 2– відкрити одну з існуючих *S*-моделей;

– 3– змінити властивості вікна оглядача. Дана кнопка дозволяє встановити режим відображення вікна оглядача "поверх всіх вікон". Повторне натискання скасовує такий режим;

– 4– пошук блоку за назвою (по перших символах назви). Після того як блок буде знайдений, у вікні оглядача відкриється відповідний розділ бібліотеки, а блок буде виділений. Якщо ж блок з такою назвою відсутня, то у вікні коментаря буде виведено повідомлення *Not found* < ім'я блоку> (Блок не знайдений).

Для створення моделі в середовищі *SIMULINK* необхідно послідовно виконати ряд дій.

Створити новий файл моделі за допомогою команди *File/New/Model*, або використовуючи кнопку на панелі інструментів (тут і далі, за допомогою символу “/”, зазначені пункти меню програми, які необхідно послідовно вибрати для виконання зазначеної дії). Новостворене вікно моделі показано на рисунку 1.5.

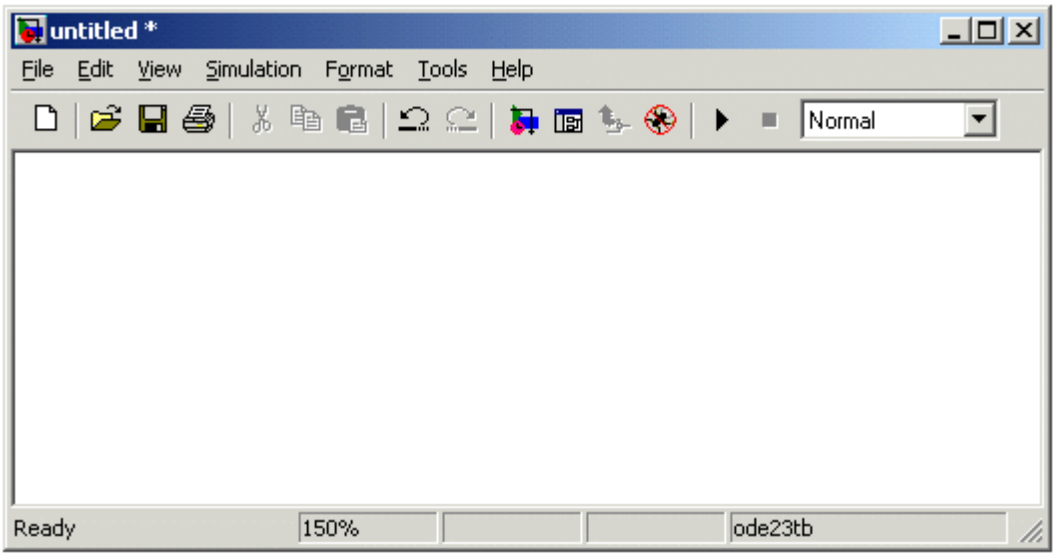


Рисунок 1.5 – Порожнє вікно моделі

Розташувати блоки у вікні моделі. Для цього необхідно відкрити відповідний розділ бібліотеки (Наприклад, *Sources* – Источники). Далі, вказавши курсором на необхідний блок і, натиснувши на ліву клавішу "миші", "перетягнути" блок у створене вікно. Клавішу миші потрібно тримати натиснутою. На рисунку 1.6 показано вікно моделі, що містить блоки.

Для видалення блоку необхідно вибрати блок (вказати курсором на його зображення і натиснути ліву клавішу "миші"), а потім натиснути клавішу *Delete* на клавіатурі. Для зміни розмірів блоку потрібно вибрати блок, встановити курсор в один з кутів блоку і, натиснувши ліву клавішу "миші", змінити розмір блоку (курсор при цьому перетвориться в двосторонню стрілку).

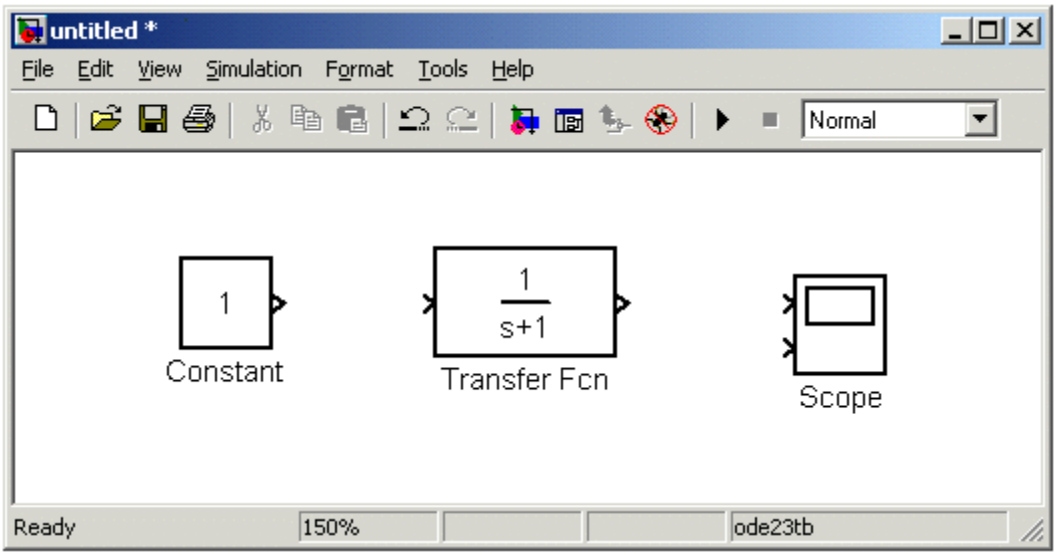


Рисунок 1.6 – Вікно моделі, що містить блоки

Далі, якщо це потрібно, потрібно змінити параметри блоку, встановлені програмою "за замовчуванням". Для цього необхідно двічі клацнути лівою клавішею "миші", вказавши курсором на зображення блоку. Відкриється вікно редагування параметрів даного блоку. При завданні чисельних параметрів слід мати на увазі, що в якості десяткового роздільника повинна використовуватися точка, а не кома. Після внесення змін потрібно закрити вікно кнопкою *OK*. На рисунку 1.7 як приклад показані блок, моделюючий передатну функцію і вікно редагування параметрів даного блоку.

Після установки на схемі всіх блоків із потрібних бібліотек потрібно виконати з'єднання елементів схеми. Для з'єднання блоків необхідно вказати курсором на "вихід" блоку, а потім, натиснути і, не відпускаючи ліву клавішу "миші", провести лінію до входу іншого блоку. Після чого відпустити клавішу. У разі правильного з'єднання зображення стрілки на вході блоку змінює колір. Для створення точки розгалуження в сполучній лінії потрібно підвести курсор до передбачуваного вузла і, натиснувши праву клавішу "миші", протягнути лінію.

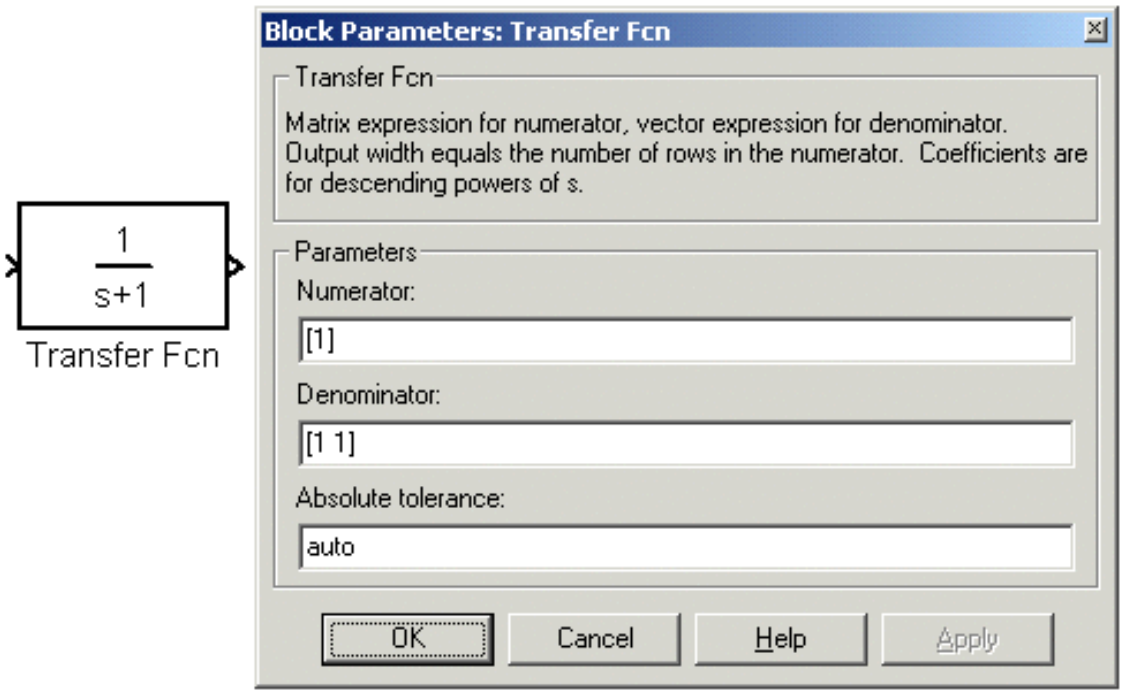


Рисунок 1.7 – Блок, що моделює передатну функцію і вікно редагування параметрів блоку

Для видалення лінії потрібно вибрати лінію (так само, як це виконується для блоку), а потім натиснути клавішу *Delete* на клавіатурі. Схема моделі, в якій виконані з'єднання між блоками, показана на рисунку 1.8.

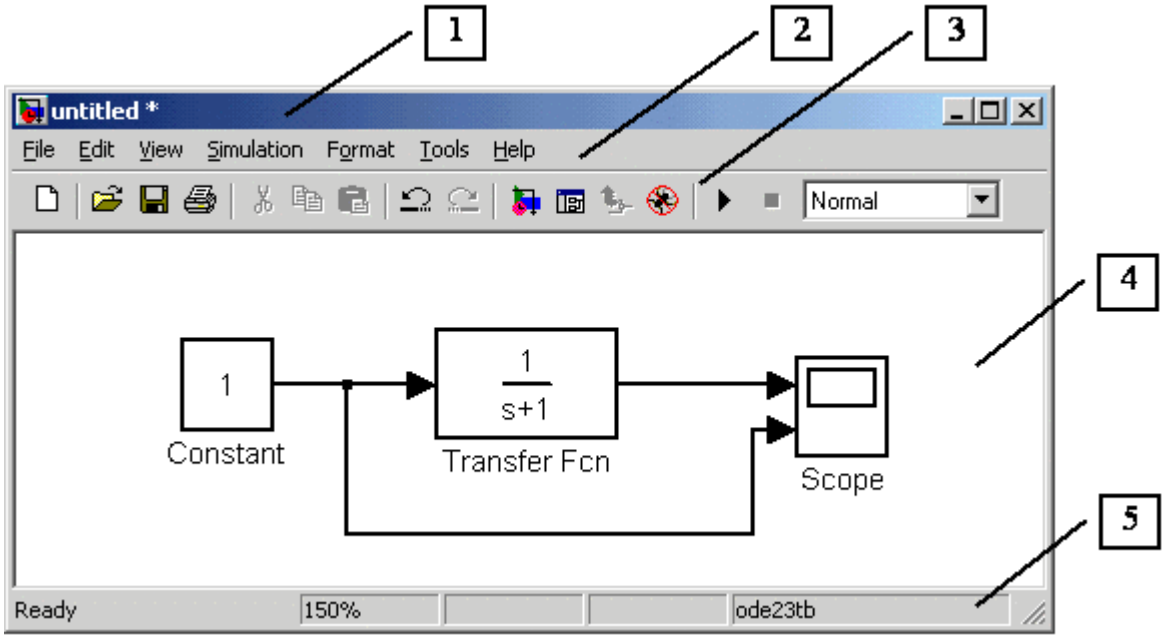


Рисунок 1.8 – Схема моделі

Після складання розрахункової схеми необхідно зберегти її у вигляді файлу на диску, вибравши пункт меню *File/Save As...* у вікні схеми і вказавши папку та ім'я файлу. Слід мати на увазі, що ім'я файлу не повинен перевищувати 32 символів, має починатися з букви і не може містити символи кирилиці та спецсимволи. Ця ж вимога ставиться і до шляху файлу (до тих папках, в яких зберігається файл). При подальшому редагуванні схеми можна користуватися пунктом меню *Fille/Save*. При повторних запусках програми *SIMULINK* завантаження схеми здійснюється за допомогою меню *File/Open...* у вікні оглядача бібліотеки або з основного вікна *MATLAB*.

ПРАКТИЧНЕ ЗАНЯТТЯ № 2

Дослідження методів підготовки і редагування моделі

Мета роботи: дослідити методи підготовки та редагування моделі в програмі *Simulink* пакета *MATLAB.*

2.1 Складові вікна моделі

Вікно моделі містить наступні елементи:

– заголовок, з назвою вікна. Новоствореному вікна присвоюється ім'я *Untitled* з відповідним номером;

– меню с командами *File, Edit, View* и т.д.;

– панель іінструментів;

– вікно для створення схеми моделі;

– рядок стану, що містить інформацію про поточний стан моделі.

Меню вікна містить команди для редагування моделі, її налаштування і управління процесом розрахунку, роботи файлами і т.п.:

– *File* (Файл) – робота з файлами моделей;

– *Edit* (Редагування) – зміна моделі та пошук блоків;

– *View* (Вид) – управління показом елементів інтерфейсу;

– *Simulation* (Моделювання) – завдання налаштувань для моделювання і керування процесом розрахунку;

– *Format* (Форматування) – зміна зовнішнього вигляду блоків і моделі в цілому;

– *Tools* (Інструментальні засоби) – застосування спеціальних засобів для роботи з моделлю (відладчик, лінійний аналіз і т.п.);

– *Help* (Довідка) – висновок вікон довідкової системи.

Для роботи з моделлю можна також використовувати кнопки на панелі інструментів (рисунок 2.1).

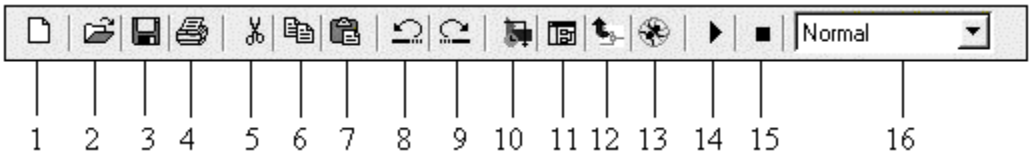


Рисунок 2.1 – Панель інструментів вікна моделі

Кнопки панелі інструментів мають наступне призначення:

– *New Model* – відкрити нове (порожнє) вікно моделі;

– *Open Model* – відкрити існуючий *mdl*-файл;

– *Save Model* – зберегти *mdl*-файл на диску;

– *Print Model* – вивід на друк блок-діаграми моделі;

– *Cut* – вирізати виділену частину моделі в буфер проміжного зберігання;

– *Copy* – скопіювати виділену частину моделі в буфер проміжного зберігання;

– *Paste* – вставити у вікно моделі вміст буфера проміжного зберігання;

– *Undo* – скасувати попередню операцію редагування;

– *Redo* – відновити результат скасованої операції редагування;

– *Library Browser* – відкрити вікно оглядача бібліотек;

– *Toggle Model Browser* – відкрити вікно оглядача моделі;

– *Go to parent system* – перехід з підсистеми в систему вищого рівня ієрархії ("батьківську систему"). Команда доступна тільки, якщо відкрита підсистема;

– *Debug* – запуск відладчика моделі;

– *Start/Pause/Continue Simulation* – запуск моделі на виконання (команда Start); після запуску моделі на зображенні кнопки виводиться символ  , и їй відповідає вже команда *Pause* (Призупинити моделювання); для відновлення моделювання слід клацнути по тій же кнопці, оскільки в режимі паузи їй відповідає команда *Continue* (Продовжити);

– *Stop* – закінчити моделювання. Кнопка стає доступною після початку моделювання, а також після виконання команди *Pause*;

– *Normal/Accelerator* – звичайний / прискорений режим розрахунку. Інструмент доступний, якщо встановлено додаток *Simulink Performance Tool*.

У нижній частині вікна моделі знаходиться рядок стану, в якій відображаються короткі коментарі до кнопок панелі інструментів, а також до пунктів меню, коли покажчик миші знаходиться над відповідним елементом

інтерфейсу. Це ж текстове поле використовується і для індикації стану *Simulink*: *Ready* (Готовbq) або *Running* (Виконання). У рядку стану відображаються також:

– масштаб відображення блок-діаграми (у відсотках, вихідне значення дорівнює 100%);

– індикатор ступеня завершеності сеансу моделювання (з'являється після запуску моделі);

– поточне значення модельного часу (виводиться також тільки після запуску моделі);

– використовуваний алгоритм розрахунку станів моделі (метод рішення).

2.2 Основні прийоми підготовки і редагування моделі

2.2.1 Додавання текстових написів

Для підвищення наочності моделі зручно використовувати текстові написи. Для створення напису потрібно вказати мишею місце напису і двічі клацнути лівою клавішею миші. Після цього з'явиться прямокутна рамка з курсором вводу. Аналогічним чином можна змінити і підписи до блоками моделей. На рисунку 2.2 показані текстова напис і зміна написи в блоці передавальної функції. Слід мати на увазі, що розглянута версія програми (*Simulink* 4) не адаптована до використання кириличних шрифтів, і застосування їх може мати самі різні наслідки: – відображення написів в нечитабельним вигляді, обрізання написів, повідомлення про помилки, а також неможливість відкрити модель після її збереження. Тому застосування написів російською мовою для поточної версії *Simulink* не рекомендується.

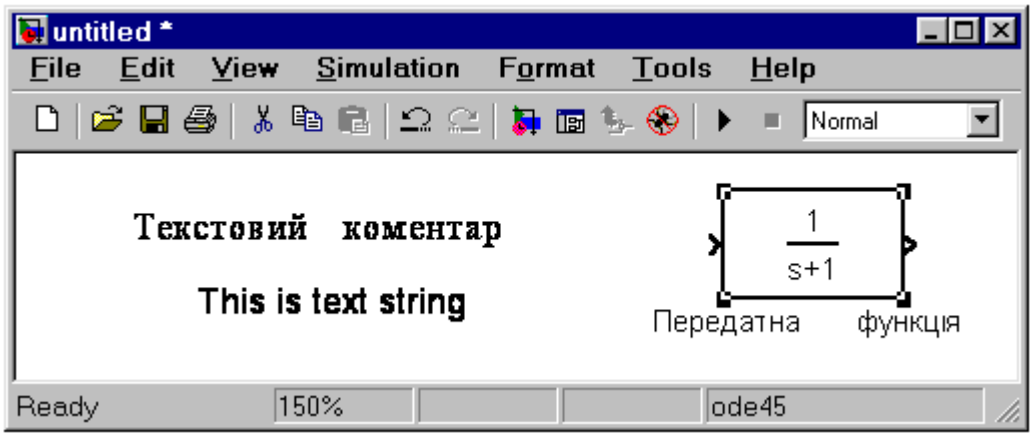


Рисунок 2.2 – Текстовий надпис і зміна надпису в *Transfer Function*

2.2.2 Виділення об'єктів

Для виконання будь-якої дії з елементом моделі (блоком, сполучної лінією, написом) цей елемент необхідно спочатку виділити.

Виділення об'єктів найпростіше здійснюється мишею. Для цього необхідно встановити курсор миші на потрібному об'єкті і клацнути лівою клавішею миші. Відбудеться виділення об'єкта. Про це будуть свідчити маркери по кутах об'єкта (див. рисунок 2.2). Можна також виділити кілька об'єктів. Для цього треба встановити курсор миші поблизу групи об'єктів, натиснути ліву клавішу миші і, не відпускаючи її, почати переміщати мишу. З'явиться пунктирна рамка, розміри якої будуть змінюватися при переміщенні миші. Всі охоплені рамкою об'єкти стають виділеними. Виділити всі об'єкти також можна, використовуючи команду *Edit/Select All*. Після виділення об'єкта його можна копіювати або переміщати в буфер проміжного зберігання, витягувати з буфера, а також видаляти, використовуючи стандартні прийоми роботи в *Windows*-програмах.

2.2.3 Копіювання і переміщення об'єктів в буфер проміжного зберігання

Для копіювання об'єкта в буфер його необхідно попередньо виділити, а потім виконати команду *Edit/Copy* або скористатися інструментом  на панелі інструментів.

Для вирізування об'єкту в буфер його необхідно попередньо виділити, а потім виконати команду *Edit/Cut* або скористатися інструментом  на панелі інструментів. При виконанні даних операцій слід мати на увазі, що об'єкти поміщаються у власний буфер *MATLAB* і недоступні з інших додатків. Використання команди *Edit/Copy model to Clipboard* дозволяє помістити графічне зображення моделі в буфер *Windows* і, відповідно, робить його доступним для решти програм.

Копіювання можна виконати і таким чином: натиснути праву клавішу миші, і не відпускаючи її, перемістити об'єкт. При цьому буде створено копію об'єкта, яку можна перемістити у необхідне місце.

2.2.4 Вставка об'єктів з буфера проміжного зберігання

Для вставки об'єкта з буфера необхідно попередньо вказати місце вставки, клацнувши лівою клавішею миші в передбачуваному місці вставки, а потім виконати команду *Edit/Paste* або скористатися інструментом  на панелі інструментів.

2.2.5 Видалення об'єктів

Для видалення об'єкта його необхідно попередньо виділити, а потім виконати команду *Edit/Clear* або скористатися клавішею *Delete* на клавіатурі. Слід врахувати, що команда *Clear* видаляє блок без приміщення його в буфер обміну. Однак цю операцію можна скасувати командою меню *File/Undo*.

2.2.6 З'єднання блоків

Для з'єднання блоків необхідно спочатку встановити курсор миші на вихідний порт одного з блоків. Курсор при цьому перетвориться на великий хрест з тонких ліній (рисунок 2.3). Тримаючи натиснутою ліву кнопку миші, потрібно перемістити курсор до вхідного порту потрібного блоку.

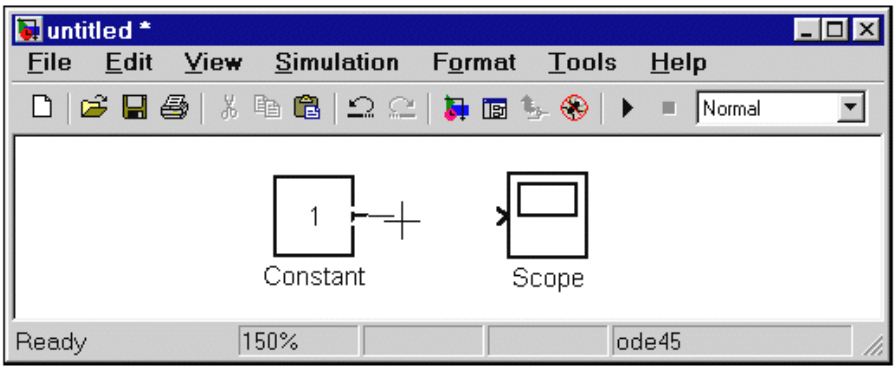


Рисунок 2.3 – Початок створення сполуки

Курсор миші прийме вигляд хреста з тонких здвоєних ліній (рисунок 2.4). Після створення лінії необхідно відпустити ліву клавішу миші. Свідченням того, що з'єднання створено, буде жирна стрілка біля вхідного порту блоку. Виділення лінії виробляється точно так, і виділення блоку – одинарним клацанням лівої клавіші миші. Чорні маркери, розташовані у вузлах сполучної лінії, будуть говорити про те, що лінія виділена.

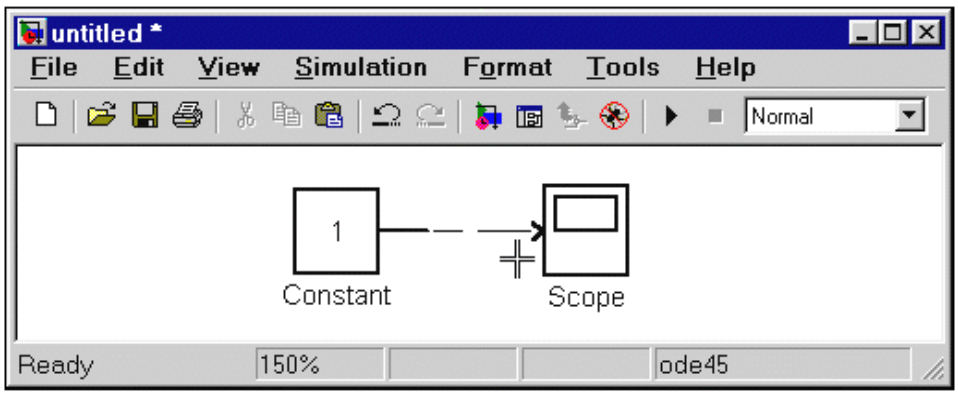


Рисунок 2.4 – Завершення створення сполуки

Створення петлі лінії з'єднання виконується також як переміщення блоку. Лінія з'єднання виділяється, і потім потрібна частина лінії переміщається. Рисунок 2.5 пояснює цей процес. Видалення з'єднань виконується також як і будь-яких інших об'єктів.

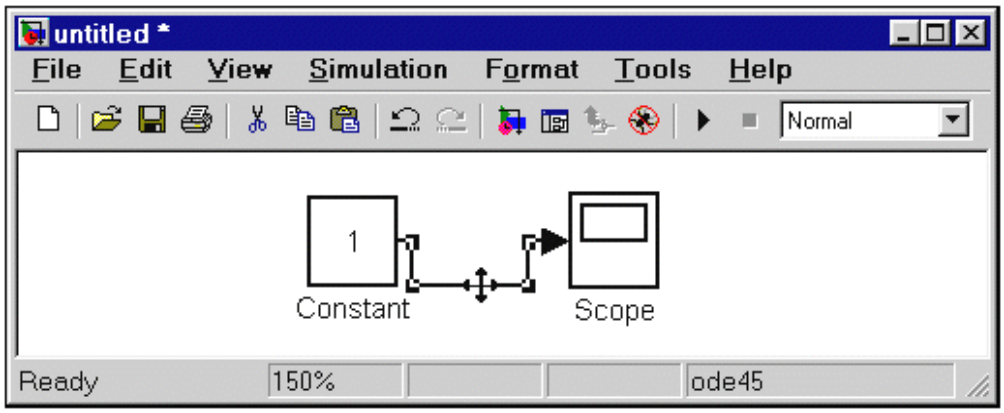


Рисунок 2.5 – Створення петлі в сполучній лінії

2.2.7 Зміна розмірів блоків

Для зміни розміру блоку він виділяється, після чого курсор миші треба встановити на один із маркерів по кутах блоку. Після перетворення курсору на двосторонню стрілку, необхідно натиснути ліву клавішу миші і розтягнути (або стиснути) зображення блоку. На рисунку 2.6 показаний цей процес. Розміри написів блоку при цьому не змінюються.

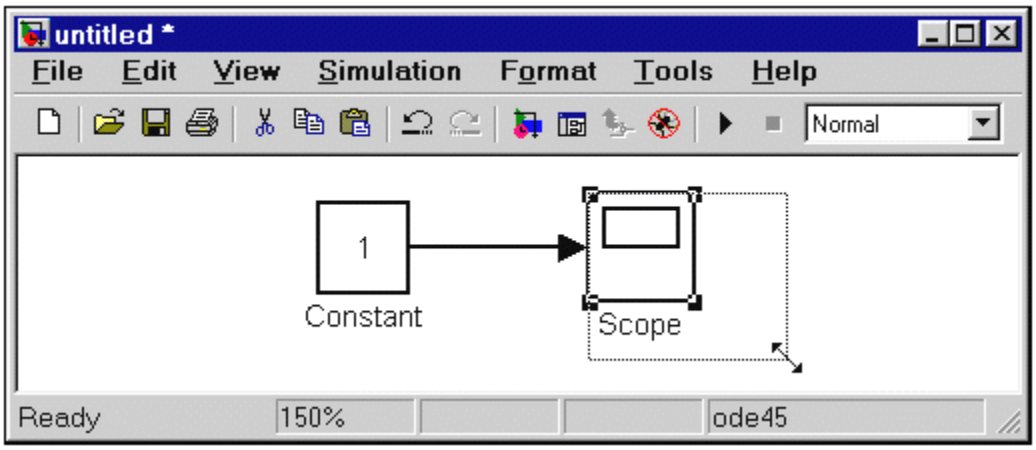


Рисунок 2.6 – Зміна розміру блоку

2.2.8 Переміщення блоків

Будь блок моделі можна перемістити, виділивши його, і пересунувши, тримаючи натиснутою ліву клавішу миші. Якщо до входів і виходів блоку

підведені сполучні лінії, то вони не розриваються, а лише скорочуються або збільшуються в довжині. В з'єднання можна також вставити блок, що має один вхід і один вихід. Для цього його потрібно розташувати в необхідному місці сполучної лінії.

2.2.9 Використання команд *Undo* та *Redo*

У процесі освоєння програми користувач може вчиняти дії здаються йому незворотними (наприклад, випадкове видалення частини моделі, копіювання і т.д.). У цьому випадку слід скористатися командою *Undo* – скасування останньої операції. Команду можна викликати за допомогою кнопки в панелі інструментів вікна моделі або з меню *Edit*. Для відновлення скасованої операції служить команда *Redo* (інструмент ).

2.2.10 Форматування об'єктів

У меню Format (також як і в контекстному меню, що викликається натисканням правої клавіші миші на об'єкті) знаходиться набір команд форматування блоків. Команди форматування поділяються на кілька груп.

Зміна відображення написів:

– *Font* – форматування шрифту надписів і текстових блоків;

– *Text alignment* – вирівнювання тексту в текстових надписах;

– *Flip name* – переміщення підпису блоку;

– *Show/Hide name* – відображення чи приховування підпису блоку.

Зміна кольорів відображення блоків:

– *Foreground color* – вибір кольору ліній для виділених блоків;

– *Background color* – вибір кольору фону виділених блоків;

– *Screen color* – вибір кольору фону для всього вікна моделі.

Зміна положення блоку і його види:

– *Flip block* – дзеркальне відображення щодо вертикальної осі симетрії;

– *Rotate block* – поворот блоку на 90^0 за годинниковою стрілкою;

– *Show drop shadow* – показ тіні від блоку;

– *Show port labels* – показ міток портів.

Інші установки:

– *Library link display* – показ зв'язків з бібліотеками;

– *Sample time colors* – вибір кольору блоку індикації часу;

– *Wide nonscalar lines* – збільшення / зменшення ширини нескалярних ліній;

– *Signal dimensions* – показ розмірності сигналів;

– *Port data types* – показ даних про тип портів;

– *Storage class* – клас пам'яті. Параметр, установлюваний при роботі *Real-Time Workshop*;

– *Execution order* – висновок порядкового номера блоку в послідовності виконання.

ПРАКТИЧНЕ ЗАНЯТТЯ № 3

Дослідження методів установки параметрів і виконання розрахунку моделі

Мета роботи: дослідити методи установки параметрів і виконання розрахунку моделі в програмі *Simulink* пакета *MATLAB*.

Перед виконанням розрахунків необхідно попередньо задати параметри розрахунку. Завдання параметрів розрахунку виконується в панелі управління меню *Simulation/Parameters*. Вид панелі управління наведений на рисунку 3.1.

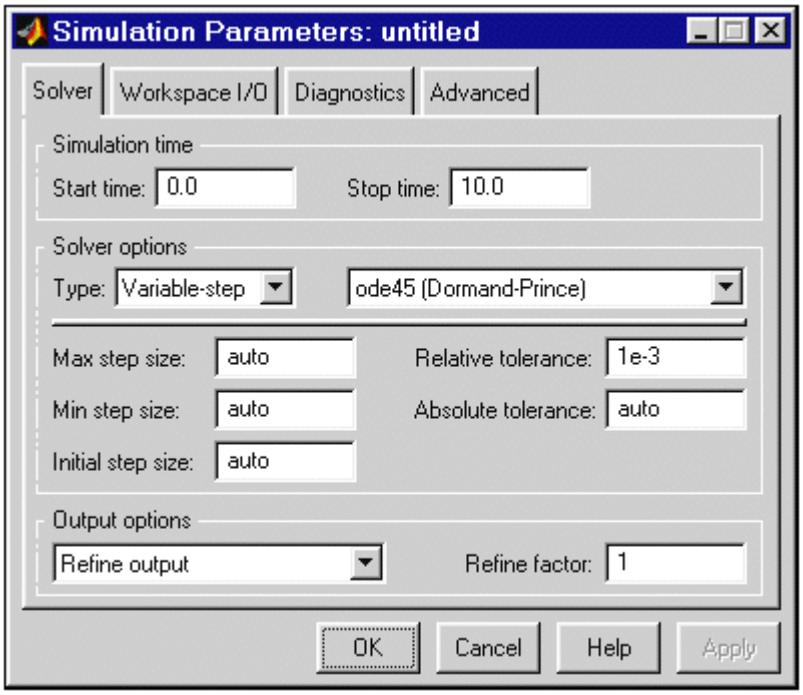


Рисунок 3.1 – Панель управління

Вікно налаштування параметрів розрахунку має 4 вкладки:

– *Solver* (Расчет) – установка параметрів розрахунку моделі;

– *Workspace* I/O (Введення / висновок даних у робочу область) – установка параметрів обміну даними з робочою областю *MATLAB*;

– *Diagnostics* (Диагностика) – вибір параметрів діагностичного режиму;

– *Advanced* (Дополнительно) – установка додаткових параметрів.

Установка параметрів розрахунку моделі виконується за допомогою елементів управління, розміщених на вкладці *Solver*. Ці елементи розділені на

три групи (рисунок 3.1): *Simulation time* (Інтервал моделювання або, іншими словами, час розрахунку), *Solver options* (Параметри розрахунку), *Output options* (Параметри виводу).

3.1 Установка параметрів розрахунку моделі

3.1.1 Інтервал моделювання або час розрахунку (*Simulation time*)

Час розрахунку задається вказівкою початкового (*Start time*) і кінцевого (*Stop time*) позначень часу розрахунку. Початковий час, як правило, задається рівним нулю. Величина кінцевого часу задається користувачем виходячи з умов розв'язуваної задачі.

3.1.2 Параметри розрахунку (*Solver options*)

При виборі параметрів розрахунку необхідно вказати спосіб моделювання (*Type*) і метод розрахунку нового стану системи. Для параметра *Type* доступні два варіанти – з фіксованим (*Fixed-step*) або з перемінним (*Variable-step*) кроком. Як правило, *Variable-step* використовується для моделювання безперервних систем, а *Fixed-step* – для дискретних.

Список методів розрахунку нового стану системи містить кілька варіантів. перший варіант (*discrete*) використовується для розрахунку дискретних систем. Решта методи використовуються для розрахунку безперервних систем. Ці методи різні для змінного (*Variable-step*) і для фіксованого (*Fixed-step*) кроку часу, але, по суті, являють собою процедури розв'язання систем диференціальних рівнянь. Детальний опис кожного з методів розрахунку станів системи приведено у вбудованій довідковій системі *MATLAB*. Нижче двох списків *Type*, що розкриваються, знаходиться область, вміст якої змінюється залежно від вибраного способу зміни модельного часу. при виборі *Fixed-step* в даній області з'являється текстове поле *Fixed-step size* (величина фіксованого кроку) дозволяє вказувати величину кроку моделювання (рисунок 3.2). Величина кроку моделювання за замовчуванням встановлюється системою автоматично (*auto*). Необхідна величина кроку може бути введена замість значення *auto* або у формі числа, або у вигляді обчислюваного виразу (те ж саме відноситься і до всіх параметрів встановлюються системою автоматично).

При виборі *Fixed-step* необхідно також задати режим розрахунку (*Mode*). Для параметра *Mode* доступні три варіанти:

– *MultiTasking* (Багатозадачний) – необхідно використовувати, якщо в моделі присутні паралельно працюючі підсистеми, і результат роботи моделі

залежить від часових параметрів цих підсистем. Режим дозволяє виявити невідповідність швидкості і дискретності сигналів, що пересилаються блоками один одному;

– *SingleTasking* (Однозадачний) – використовується для тих моделей, у яких недостатньо сувора синхронізація роботи окремих складових не впливає на кінцевий результат моделювання;

– *Auto* (Автоматичний вибір режиму) – дозволяє *Simulink* автоматично встановлювати режим *MultiTasking* для тих моделей, у яких використовуються блоки з різними швидкостями передачі сигналів і режим *SingleTasking* для моделей, в яких містяться блоки, що оперують однаковими швидкостями.

При виборі *Variable-step* в області з'являються поля для установки трьох параметрів:

– *Max step size* – максимальний крок розрахунку. За замовчуванням він встановлюється автоматично (*auto*) і його значення в цьому випадку дорівнює (*SfopTime* – *StartTime*)/50. Досить часто це значення виявляється занадто великим, і спостережувані графіки являють собою ламані лінії. У цьому випадку величину максимального кроку розрахунку необхідно задавати явно;

– *Min step size* – мінімальний крок розрахунку;

– *Initial step size* – початкове значення кроку моделювання.

При моделюванні безперервних систем з використанням змінного кроку необхідно вказати точність обчислень: відносну (*Relative tolerance*) і абсолютну (*Absolute tolerance*). За замовчуванням вони дорівнюють відповідно 10^{-3} і *auto*.

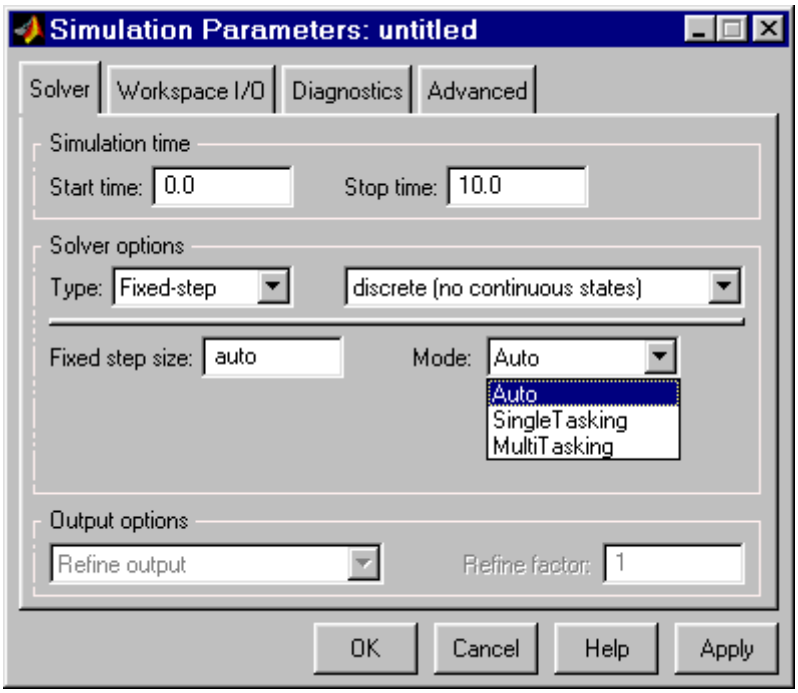


Рисунок 3.2 – Вкладка *Solver* при виборі фіксованого кроку розрахунку

3.1.3 Параметри виводу (*Output options*)

У нижній частині вкладки *Solver* задаються налаштування параметрів виводу вихідних сигналів модельованої системи (*Output options*). Для даного параметра можливий вибір одного з трьох варіантів:

– *Refine output* (Скоригований висновок) – дозволяє змінювати дискретність реєстрації модельного часу і тих сигналів, які зберігаються в робочій області *MATLAB* за допомогою блоку *To Workspace*. Установка величини дискретності виконується в рядку редагування *Refine factor*, розташованій праворуч. За умовчанням значення *Refine factor* дорівнює 1, це означає, що реєстрація проводиться з кроком *D t* = 1 (тобто для кожного значення модельного часу). Якщо задати *Refine factor* равеным 2, це означає, що буде реєструватися кожне друге значення сигналів, 3 – кожне третє і т. д. Параметр *Refine factor* може приймати тільки цілі позитивні значення;

– *Produce additional output* (Додатковий висновок) – забезпечує додаткову реєстрацію параметрів моделі в задані моменти часу; їх значення вводяться в рядку редагування (в цьому випадку вона називається *Output times*) у вигляді списку, укладеного в квадратні дужки. При використанні цього варіанта базовий крок реєстрації (*D t*) дорівнює 1. Значення часу в списку *Output times* можуть бути дробовими числами і мати будь-яку точність;

– *Produce specified output only* (Формувати тільки заданий висновок) – встановлює висновок параметрів моделі тільки в задані моменти часу, які вказуються в полі *Output times* (Моменти часу виводу).

3.2 Установка параметрів обміну з робочою областю

Елементи, що дозволяють управляти введенням і виведенням в робочу область *MATLAB* проміжних даних і результатів моделювання, розташовані на вкладці *Workspace* I/O (рисунок 3.3).

Елементи вкладки розділені на 3 поля:

– *Load from workspace* (Завантажити з робочої області). Якщо прапорець *Input* (Вхідні дані) встановлений, то в розташованому праворуч текстовому полі можна ввести формат даних, які будуть зчитуватися з робочої області *MATLAB*. Установка прапорця *Initial State* (Початковий стан) дозволяє ввести в пов'язаному з ним текстовому полі ім'я змінної, що містить параметри початкового стану моделі. Дані, зазначені в полях *Input* и *Initial State*, передаються у виконувану модель за допомогою одного або більше блоків In (з розділу бібліотеки *Sources*);

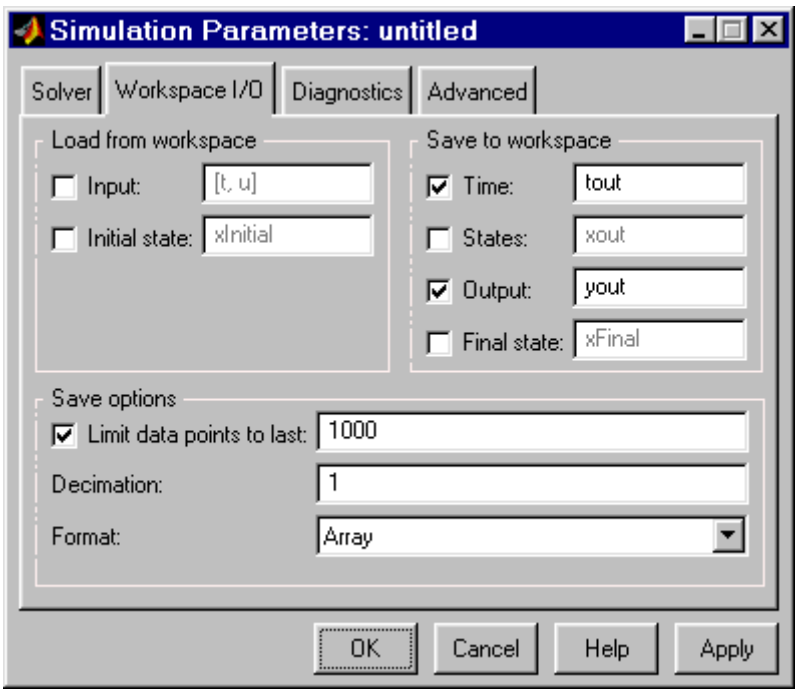


Рисунок 3.3 – Вкладка *Workspace* I/O діалогового вікна установки параметрів моделювання

– *Save to workspace* (Записати в робочу область) – дозволяє встановити режим виводу значень сигналів в робочу область *MATLAB* і задати їх імена;

– *Save options* (Параметри запису) – задає кількість рядків при передачі змінних в робочу область. Якщо прапорець *Limit rows to last* встановлений, то в полі введення можна вказати кількість переданих рядків (відлік рядків проводиться від моменту завершення розрахунку). Якщо прапорець не встановлений, то передаються всі дані. параметр *Decimation* (Виняток) задає крок запису змінних в робочу область (аналогічно параметру *Refine factor* вкладки *Solver*). Параметр *Format* (формат даних) задає формат переданих в робочу область даних. доступні формати *Array* (Масив), *Structure* (Структура), *Structure With Time* (Структура з додатковим полем – ”час”).

3.3 Установка параметрів діагностування моделі

Вкладка *Diagnostics* (рисунок 3.4) дозволяє змінювати перелік діагностичних повідомлень, що виводяться *Simulink* в командному вікні *MATLAB*, а також встановлювати додаткові параметри діагностики моделі.

Повідомлення про помилки або проблемних ситуаціях, виявлених *Simulink* в ході моделювання і вимагають втручання розробника виводяться в командному вікні *MATLAB*. Вихідний перелік таких ситуацій і вид реакції на них наведено в списку на вкладці *Diagnostics*. Розробник може вказати вид

реакції на кожне з них, використовуючи групу перемикачів у полі *Action* (вони стають доступні, якщо в списку обрано одне з подій):

– *None* – ігнорувати;

– *Warning* – видати попередження і продовжити моделювання;

– *Error* – видати повідомлення про помилку і зупинити сеанс моделювання.

Обраний вид реакції відображається в списку поряд з найменуванням події.

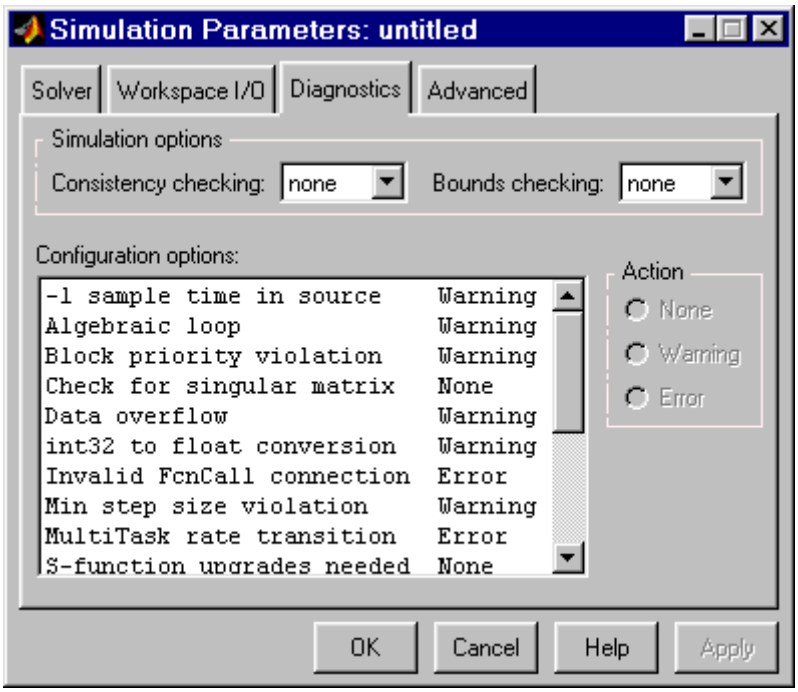


Рисунок 3.4 – Вкладка *Diagnostics* вікна установки параметрів моделювання

3.4 виконання розрахунку

Запуск розрахунку виконується за допомогою вибору пункту меню *Simulation/Start* або інструмента ▶ на панелі інструментів. Процес розрахунку можна завершити достроково, вибравши пункт меню *Simulation/Stop* або інструмент ■. Розрахунок також можна зупинити (*Simulation/Pause*) і потім продовжити (*Simulation/Continue*).

Для завершення роботи необхідно зберегти модель у файлі, закрити вікно моделі, вікно оглядача бібліотек, а також основне вікно пакета *MATLAB*.

ПРАКТИЧНЕ ЗАНЯТТЯ №4

Дослідження блоків джерел сигналів бібліотеки *Simulink* і їх параметрів

Мета роботи: дослідити блоки джерел сигналів *Sources* бібліотеки *Simulink* пакета *MATLAB*, їх призначення, параметри.

4.1 Джерело постійного сигналу *Constant*

Призначення:

Задає постійний за рівнем сигнал.

Параметри:

– *Constant value* – постійна величина;

– *Interpret vector parameters as 1-D* – інтерпретувати вектор параметрів як одновимірний (при встановленому прапорці). Даний параметр зустрічається у більшості блоків бібліотеки *Simulink*.

Значення константи може бути дійсним або комплексним числом, обчислюваним виразом, вектором або матрицею.

Рисунок 4.1 ілюструє застосування цього джерела і вимір його вихідного сигналу за допомогою цифрового індикатора *Display*.

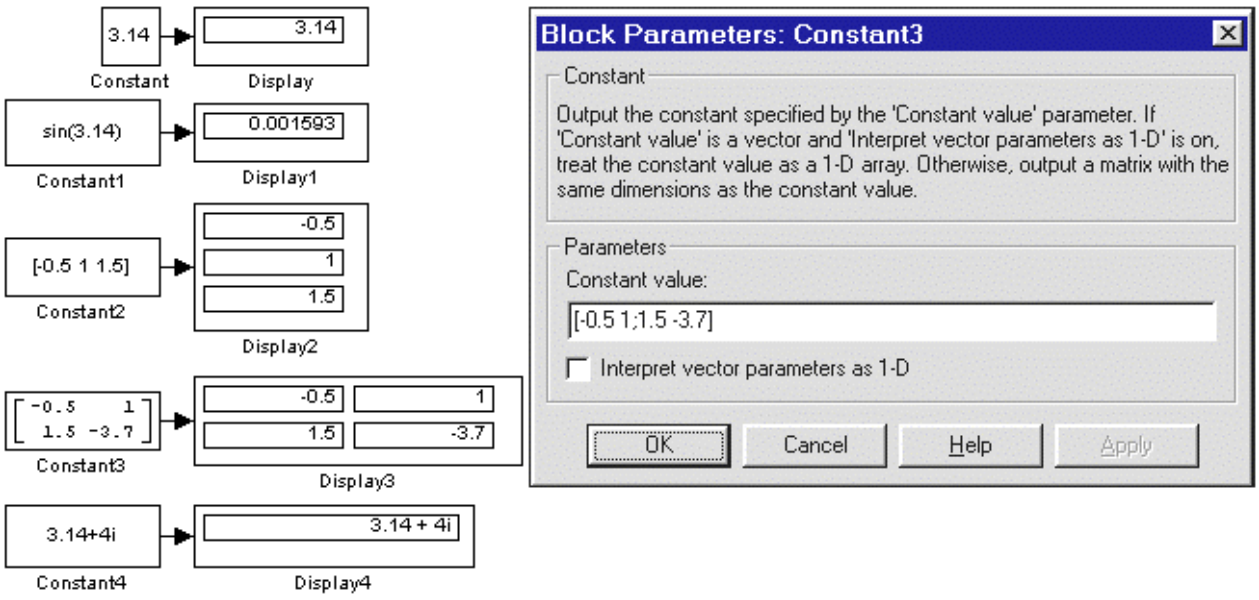


Рисунок 4.1 – Джерело постійного впливу *Constant*

4.2 Джерело синусоїдального сигналу *Sine Wave*

Призначення:

Формує синусоїдальний сигнал із заданою частотою, амплітудою, фазою і зміщенням.

Для формування вихідного сигналу блоком можуть використовуватися два алгоритми. Вид алгоритму визначається параметром *Sine Type* (спосіб формування сигналу):

– *Time-based* – По поточному часу;

– *Sample-based* – За величиною кроку модельного часу.

4.2.1 Формування вихідного сигналу за поточним значенням часу для безперервних систем

Вихідний сигнал джерела в цьому режимі відповідає виразу:

$$y = Amplitude * sin(frequency * time + phase) + bias.$$

Параметри:

– *Amplitude* – амплітуда;

– *Bias* – постійна складова сигналу;

– *Frequency* (*rads/sec*) – частота (рад/с);

– *Phase* (*rads*) – початкова фаза (рад);

– *Sample time* – крок модельного часу. Використовується для узгодження роботи джерела і інших компонентів моделі у часі. Параметр може приймати такі значення:

0 (за замовчуванням) – використовується при моделюванні безперервних систем;

> 0 (позитивне значення) – задається при моделюванні дискретних систем. У цьому випадку крок модельного часу можна інтерпретувати як крок квантування за часом вихідного сигналу;

–1 – крок модельного часу встановлюється таким же, як і в попередньому блоці, тобто блоці, звідки приходить сигнал в даний блок.

Цей параметр може задаватися для більшості блоків бібліотеки *Simulink*.

При розрахунках для дуже великих значень часу точність розрахунку вихідних значень сигналу падає внаслідок значної помилки округлення.

4.2.2 Формування вихідного сигналу за поточним значенням часу для дискретних систем

Алгоритм визначення значення вихідного сигналу джерела для кожного наступного кроку розрахунку визначається виразом (в матричній формі):

$$\begin{bmatrix} \sin(t+\Delta t) \\ \cos(t+\Delta t) \end{bmatrix} = \begin{bmatrix} \cos(\Delta t) & \sin(\Delta t) \\ -\sin(\Delta t) & \cos(\Delta t) \end{bmatrix} \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix},$$

де Δt – постійна величина, рівна значенню *Sample time*.

У даному режимі помилка округлення для великих значень часу також зменшує точність розрахунку.

4.2.3 Формування вихідного сигналу за величиною модельного часу і кількості розрахункових кроків на один період

Вихідний сигнал джерела в цьому режимі відповідає виразу:

$$y = Amplitude * \sin[(k + Number\ of\ offset\ samples) / Samples\ per\ period] + bias\ ,$$

де k – номер поточного кроку розрахунку.

Параметри:

– *Amplitude* – амплітуда;

– *Bias* – постійна складова сигналу;

– *Samples per period* – кількість розрахункових кроків на один період синусоїдальної сигналу:

$$Samples\ per\ period = 2p / (frequency * Sample\ time)$$

– *Number of offset samples* – початкова фаза сигналу. Здається кількістю кроків модельного часу:

Number of offset samples = Phase Samples per period/ (2p)*

– *Sample time* – крок модельного часу.

У даному режимі помилка округлення не накопичується, оскільки *Simulink* починає відлік номера поточного кроку з нуля для кожного періоду.

На рисунку 4.2 показано застосування блоку з різними значеннями кроку модельного часу (*Sample time* = 0 для блока *Sine Wave* = 1 і *Sample time* = 0.1 для блока *Sine Wave* = 2). Для відображення графіків вихідних сигналів в моделі використаний віртуальний осцилограф (*Scope*).

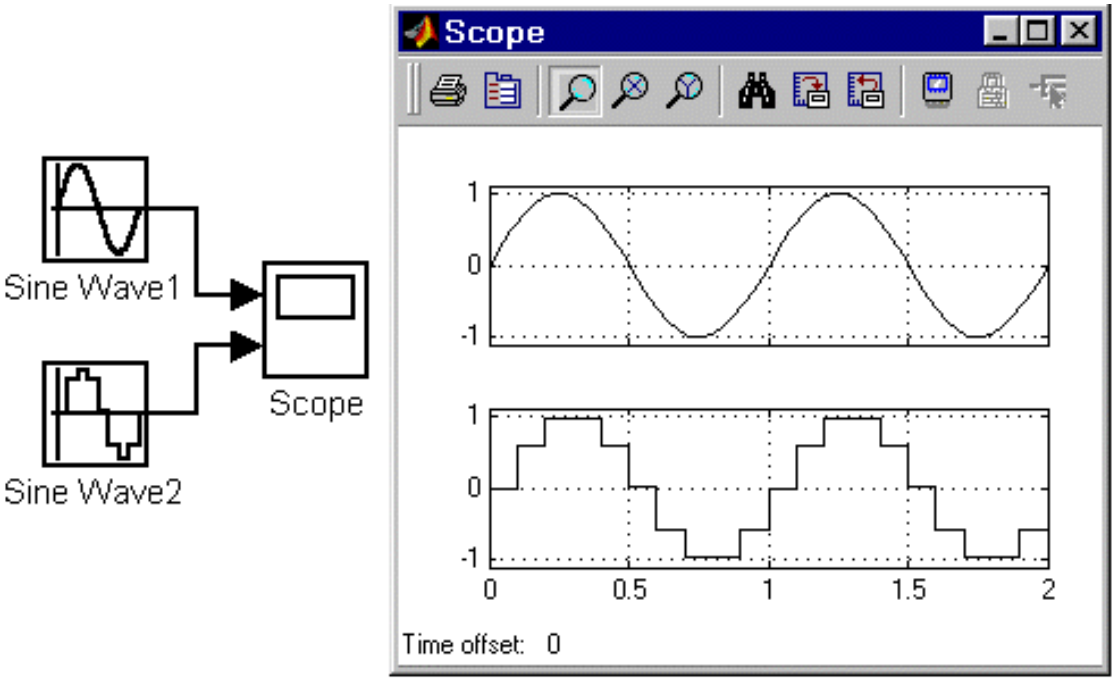


Рисунок 4.2 – Блок *Sine Wave*

4.3 Джерело лінійно мінливого впливу *Ramp*

Призначення:

Формує лінійний сигнал виду

y = Slope time + Initial value.*

Параметри:

– *Slope* – швидкість зміни вихідного сигналу;

– *Start time* – час початку формування сигналу;

– *Initial value* – початковий рівень сигналу на виході блоку.

На рисунку 4.3 показано використання даного блоку.

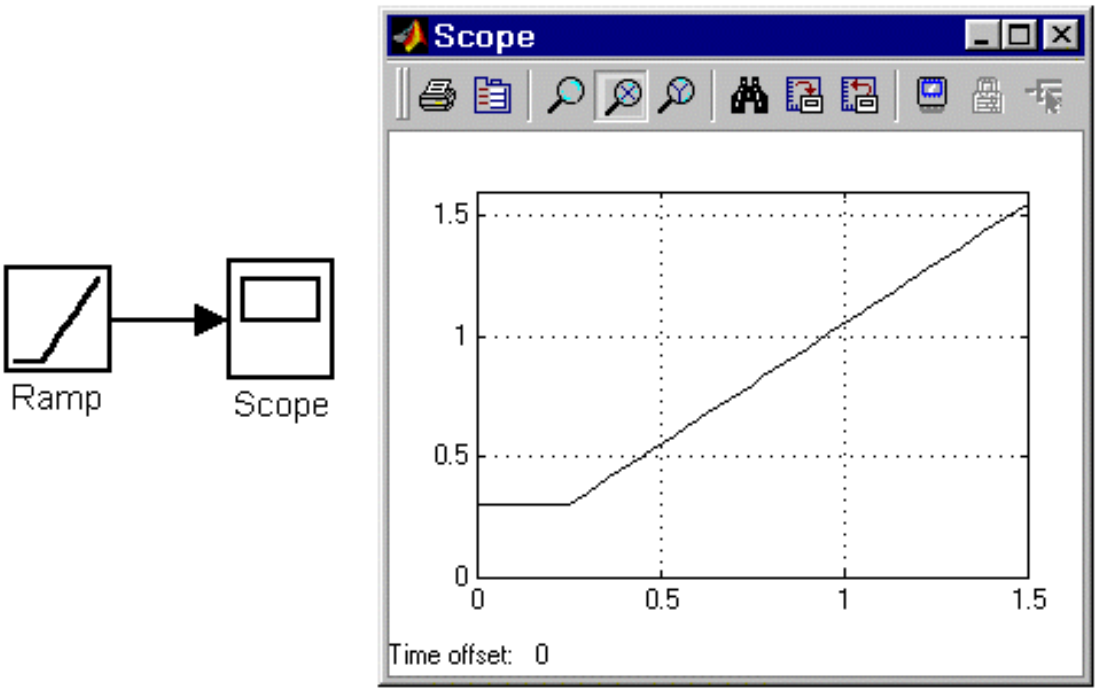


Рисунок 4.3 – Блок *Ramp*

4.4 Генератор ступінчастого сигналу *Step*

Призначення:

Формує ступінчастий сигнал.

Параметри:

– *Step time* – час настання перепаду сигналу (с);

– *Initial value* – початкове значення сигналу;

– *Final value* – кінцеве значення сигналу.

Перепад може бути як у більшу сторону (кінцеве значення більше ніж початкове), так і в меншу (кінцеве значення менше ніж початкове). Значення початкового та кінцевого рівнів можуть бути не тільки позитивними, але й негативними (наприклад, зміна сигналу з рівня –5 до рівня –3).

На рисунку 4.4 показано використання генератора ступінчастого сигналу.

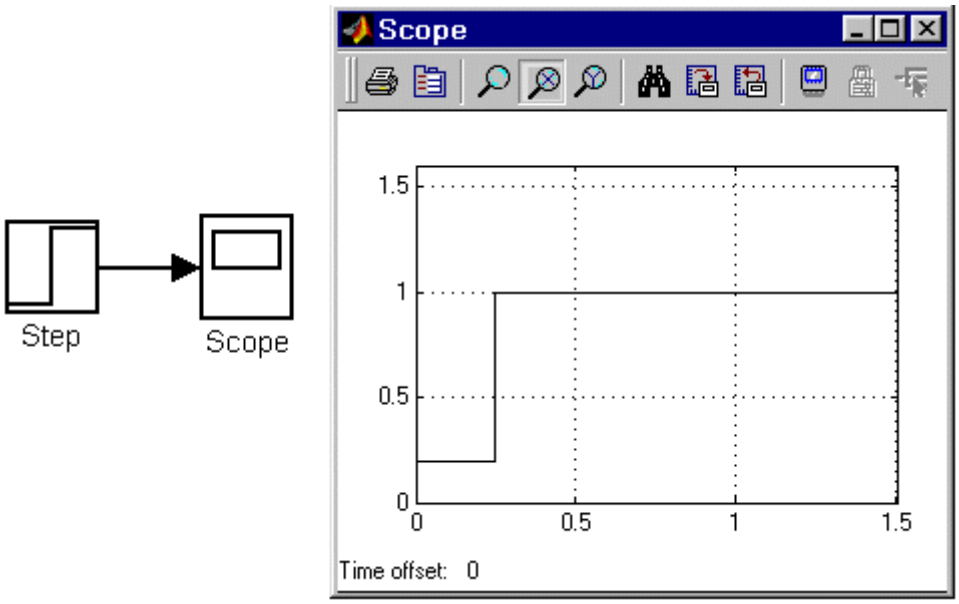


Рисунок 4.4 – Блок *Step*

4.5 Генератор сигналів *Signal Generator*

Призначення:

Формує один з чотирьох видів періодичних сигналів:

– *sine* – синусоїдальний сигнал;

– *square* – прямокутний сигнал;

– *sawtooth* – пилообразний сигнал;

– *random* – випадковий сигнал.

Параметри:

– *Wave form* – вид сигнала;

– *Amplitude* – амплітуда сигнала;

– *Frequency* – частота (рад/с);

– *Units* – одиниці вимірювання частоти. Може приймати два значення:

– *Hertz* – Гц;

– *rad/sec* – рад/с.

На рисунку 4.5 показано застосування цього джерела при моделюванні прямокутного сигналу.

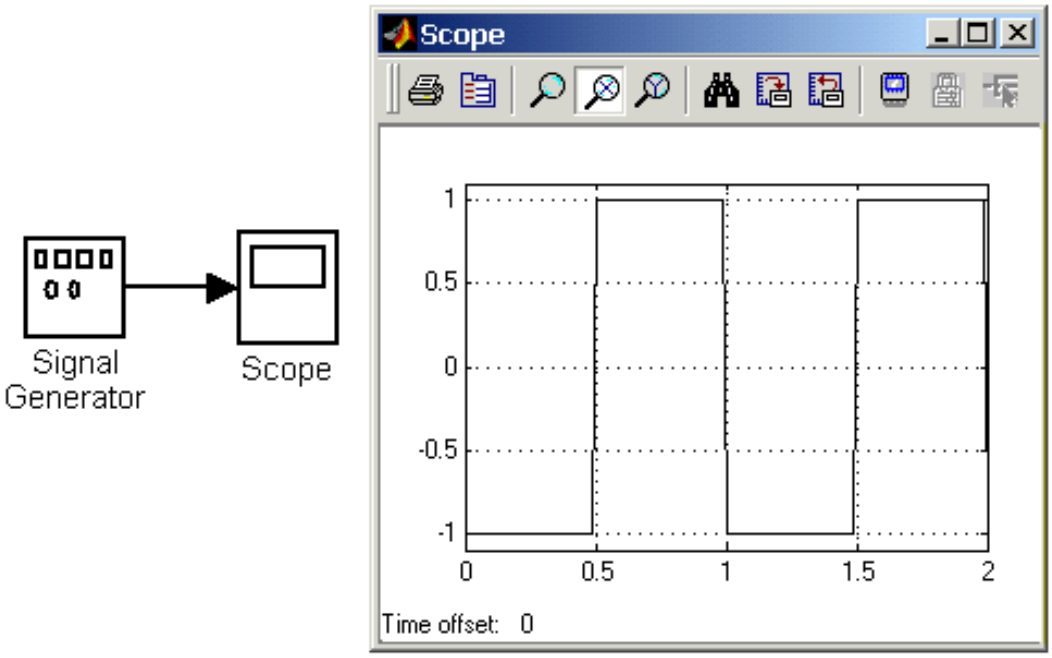


Рисунок 4.5 – Блок генератора сигналів

4.6 Джерело випадкового сигналу з рівномірним розподілом *Uniform Random Number*

Приклад використання блоку і графік його вихідного сигналу представлений на рисунку 4.6.

Призначення:

Формування випадкового сигналу з рівномірним розподілом.

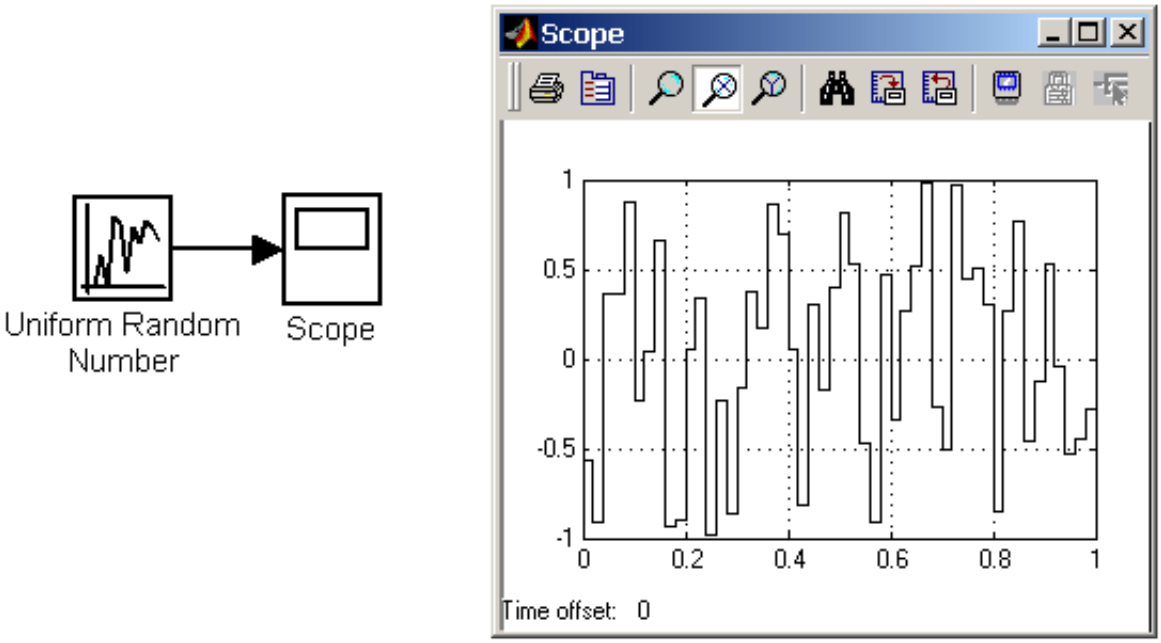


Рисунок 4.6 – Джерело випадкового сигналу з рівномірним розподілом

Параметри :

– *Minimum* – мінімальний рівень сигнала;

– *Maximum* – максимальний рівень сигнала;

– *Initial seed* – початкове значення.

4.7 Джерело випадкового сигналу з нормальним розподілом *Random Number*

Призначення:

Формування випадкового сигналу з нормальним розподілом рівня сигналу.

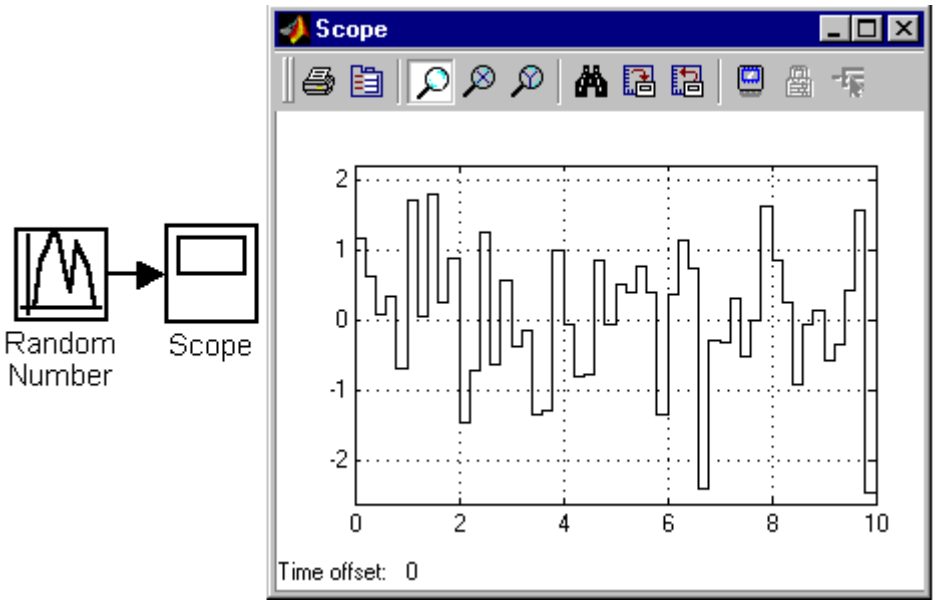


Рисунок 4.7 – Джерело випадкового сигналу з нормальним розподілом

Параметри:

– *Mean* – середнє значення сигнала;

– *Variance* – дисперсія (середньоквадратичне відхилення);

– *Initial seed* – початкове значення.

4.8 Джерело імпульсного сигнала *Pulse Generator*

Приклад використання *Pulse Generator* наведений на рисунку 4.8.

Призначення:

Формування прямокутних імпульсів.

Параметри:

– *Pulse Type* – спосіб формування сигнала. Може приймати два значення:

– *Time-based* – по поточному часу;

– *Sample-based* – за величиною модельного часу і кількості розрахункових кроків;

– *Amplitude* – амплітуда;

– *Period* – період. Здається в секундах для *Time-based Pulse Type* або в кроках модельного часу для *Sample-based Pulse Type*;

– *Pulse width* – ширина імпульсів. Здається в % по відношенню до періоду для *Time-based Pulse Type* або в кроках модельного часу для *Sample-based Pulse Type*;

– *Phase delay* – фазова затримка. Здається в секундах для *Time-based Pulse Type* або в кроках модельного часу для *Sample-based Pulse Type;*

– *Sample time* – крок модельного часу. Задається для *Sample-based Pulse Type.*

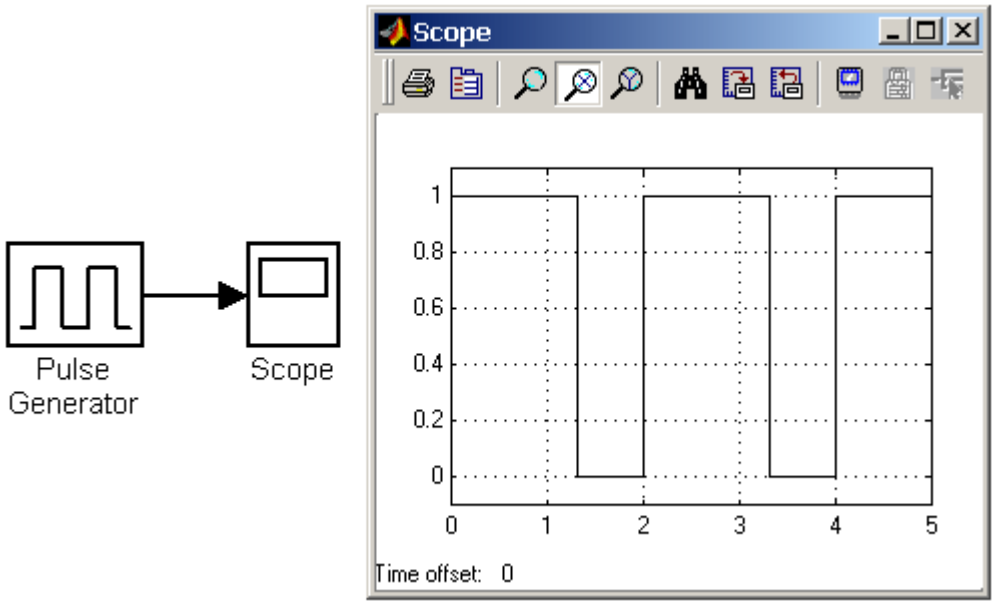


Рисунок 4.8 – Джерело прямокутних імпульсів

4.9 Генератор лінійно-змінюваної частоти *Chirp Generator*

Приклад використання блоку показаний на рисунку 4.9.

Призначення:

Формування синусоїдальних коливань, частота яких лінійно змінюється.

Параметри:

– *Initial frequence* – початкова частота (Гц);

– *Target time* – час зміни частоти (с);

– *Frequence at target time* – кінцеве значення частоти (Гц).

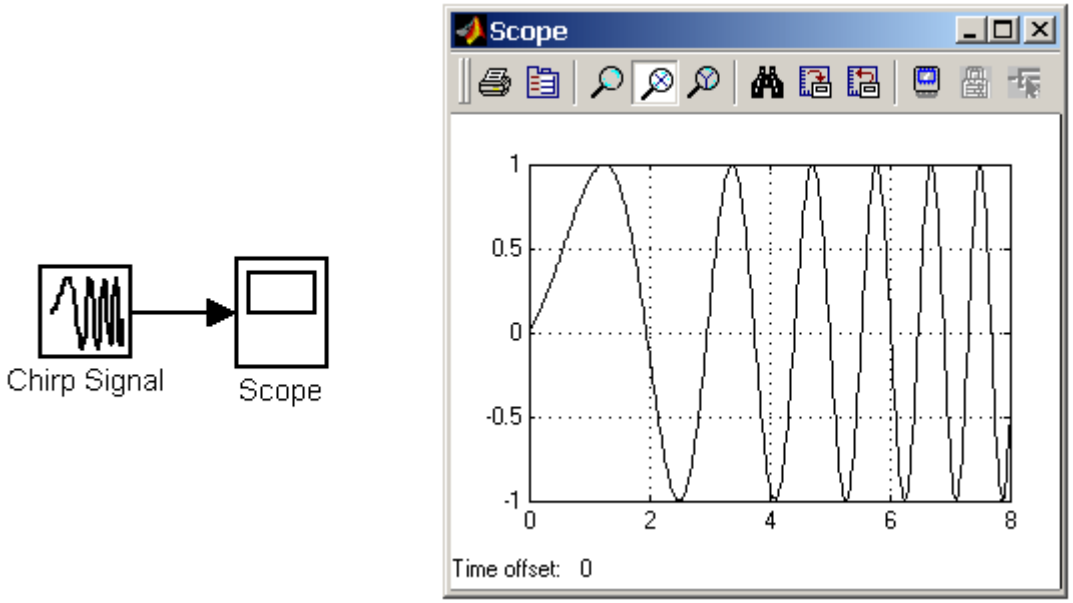


Рисунок 4.9 – Генератор лінійно-змінюваної частоти

4.10 Генератор білого шуму *Band-Limited White Noice*

Рисунок 4.10 показує роботу цього генератора.

Призначення:

Створює сигнал заданої потужності, рівномірно розподіленим по частоті.

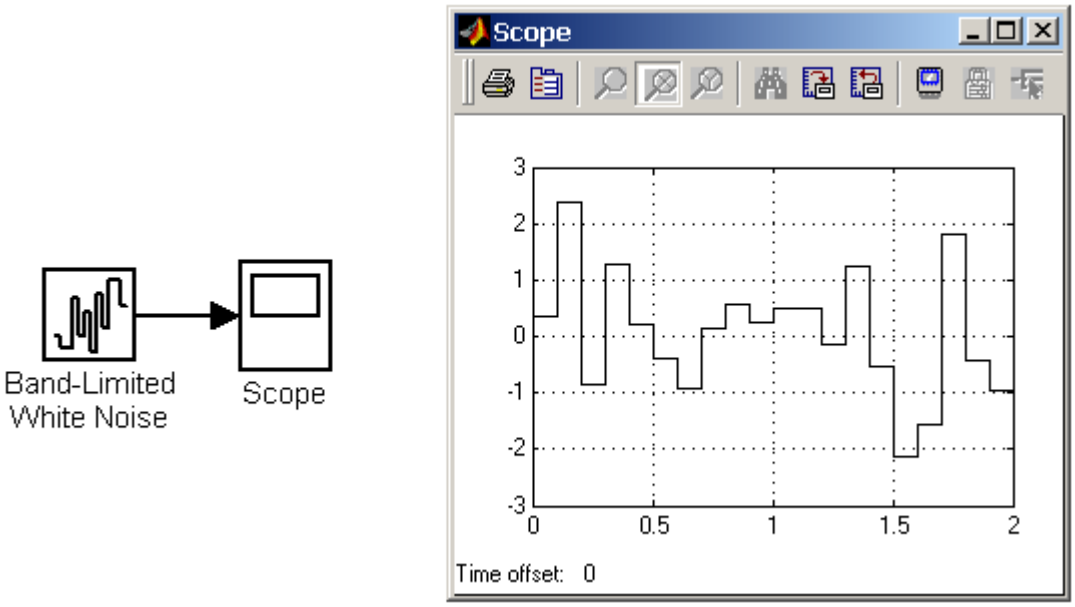


Рисунок 4.10 – Генератор білого шуму

Параметри:

– *Noice Power* – потужність шуму;

– *Sample Time* – модельний час;

– *Seed* – число, необхідне для ініціалізації генератора випадкових чисел.

4.11 Джерело тимчасового сигналу *Clock*

Призначення:

Формує сигнал, величина якого на кожному кроці розрахунку дорівнює поточному часу моделювання.

Параметри:

– *Decimation* – крок, з яким оновлюються показання часу на зображенні джерела (в тому випадку, якщо встановлений прапорець параметра *Display time*). Параметр задається як кількість кроків розрахунку. Наприклад, якщо крок розрахунку моделі у вікні діалогу *Simulation parameters* встановлений рівним 0.01 с, а параметр *Decimation* блока *Clock* заданий рівним 1000, то оновлення показань часу буде проводитися кожні 10 з модельного часу;

– *Display time* – відображення значення часу в блоці джерела.

На рисунку 4.11 показаний приклад роботи даного джерела.

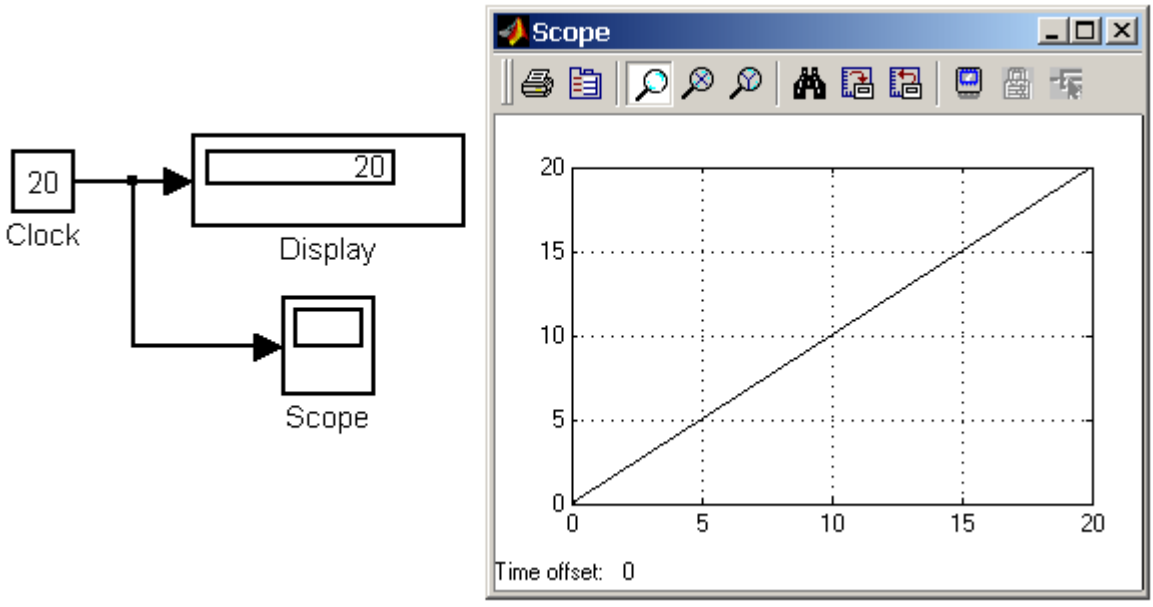


Рисунок 4.11 – Джерело тимчасового сигналу

4.12 Цифрове джерело часу *Digital Clock*

Призначення:

Формує дискретний часовий сигнал.

Параметр:

– *Sample time* – крок модельного часу (с).

На рисунку 4.12 показана работа джерел *Digital Clock*.

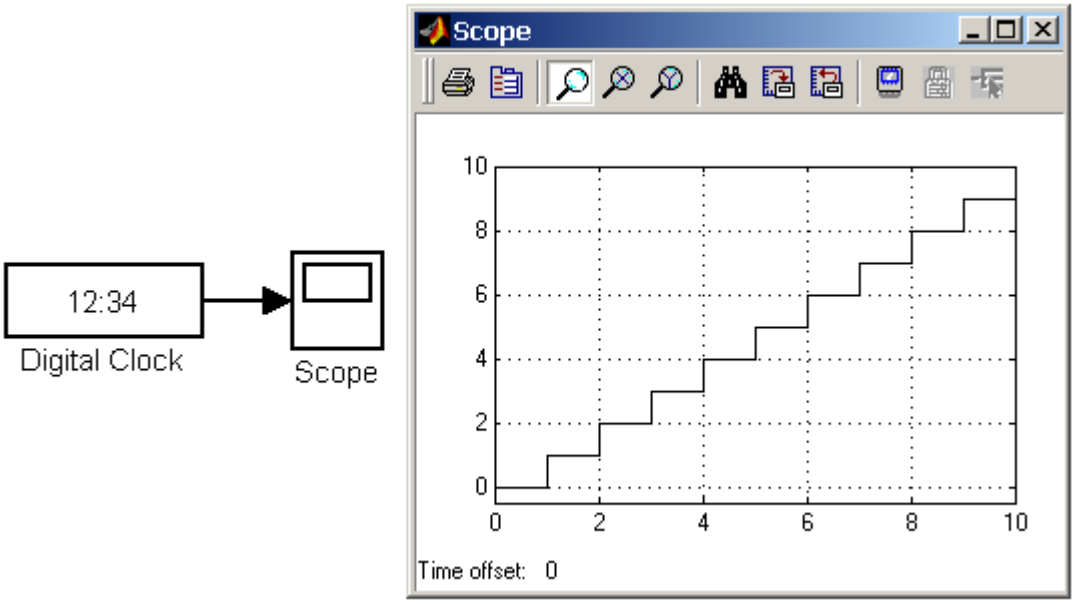


Рисунок 4.12 – Цифрове джерело тимчасового сигналу

4.13 Блок зчитування даних з файлу *From File*

Призначення:

Отримання даних із зовнішнього файлу.

Параметри:

File Name – ім'я файлу з даними.

Sample time – крок зміни вихідного сигналу блоку.

Дані у файлі повинні бути представлені у вигляді матриці:

$$\begin{bmatrix} t_1 & t_2 & \dots t_{final} \\ u1_1 & u1_2 & \dots u1_{final} \\ \dots & & \\ un_1 & un_2 & \dots un_{final} \end{bmatrix}.$$

Матриця повинна складатися, як мінімум, з двох рядків. Значення часу записані в першому рядку матриці, а в інших рядках знаходяться значення сигналів, що відповідають даними моментів часу. Значення часу повинні бути записані в зростаючому порядку. Вихідний сигнал блоку містить лише значення сигналів, а значення часу в ньому відсутні. Якщо крок розрахунку

поточної моделі не збігається з відліками часу у файлі даних, то *Simulink* виконує лінійну інтерполяцію даних.

Файл даних (*mat*-файл), з якого зчитуються значення, не є текстовим. Структура файлу докладно описана в довідковій системі *MATLAB*. Користувачам *Simulink* найзручніше створювати *макет*- файл за допомогою блоку *To File* (бібліотека *Sinks*). На рисунку 4.13 показаний приклад використання даного блоку. З файлу *data.mat* зчитуються значення синусоїдального сигналу.

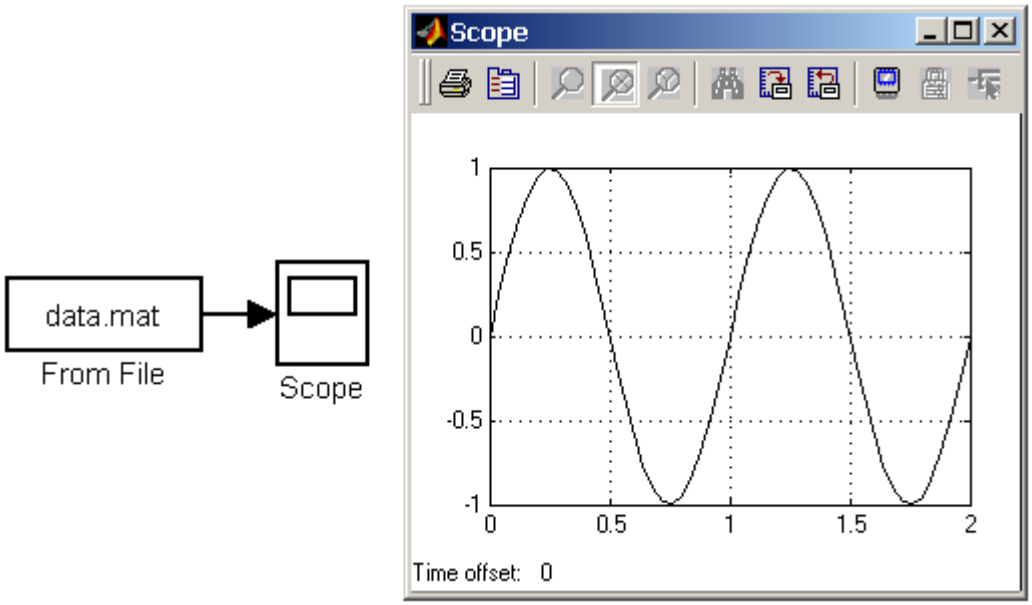


Рисунок 4.13 – Блок *From File*

4.14 Блок зчитування даних з робочого простору *From Workspace*

Призначення:

Отримання даних з робочого простору *MATLAB*.

Параметри:

– *Data* – ім'я змінної (матриці або структури) містить дані;

– *Sample time* – крок зміни вихідного сигналу блоку;

– *Interpolate data* – інтерполяція даних для значень модельного часу не збігаються зі значеннями у змінній *Data*;

– *Form output after final data value by* – вид вихідного сигналу по закінченні значень часу в змінної *Data*:

– *Extrapolate* – лінійна екстраполяція сигналів;

– *SettingToZero* – нульові значення сигналів;

– *HoldingFinalValue* – вихідні значення сигналів дорівнюють останнім значенням;

– *CyclicRepetition* – циклічне повторення значень сигналів. Даний варіант може використовуватися, тільки якщо змінна *Data* має формат *Structure without time*.

На рисунку 4.14 показаний приклад використання даного блоку. Дані в змінну *simin* робочої області *MATLAB* завантажуються з файлу за допомогою блоку *Read data*.

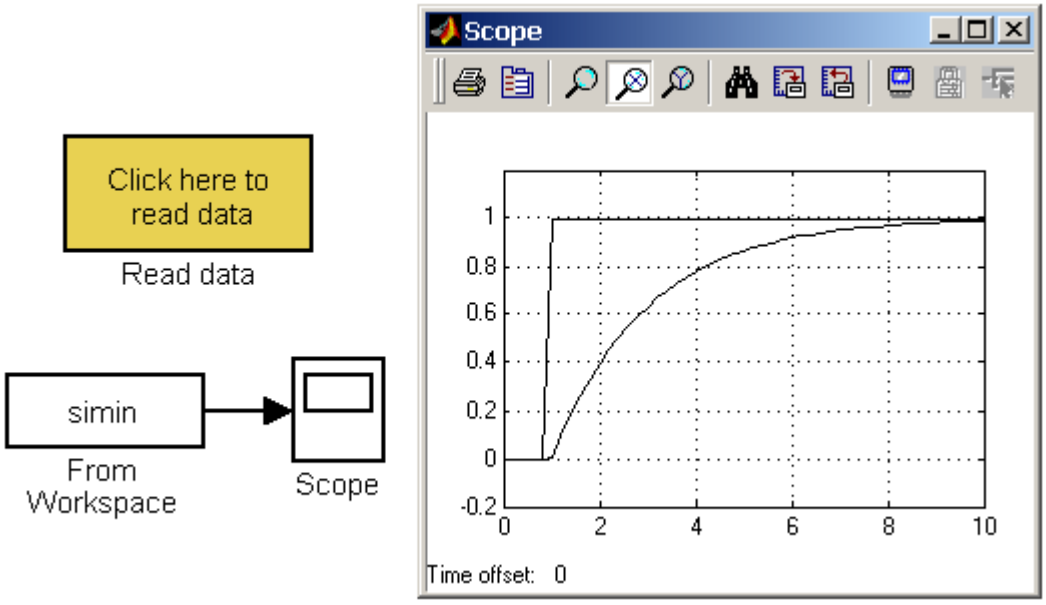


Рисунок 4.14 – Блок *From File*

4.15 Блок сигналу нульового рівня *Ground*

Призначення:

Формування сигналу нульового рівня.

Параметри:

Немеє.

Якщо який-небудь вхід блоку в моделі не приєднаний, то при виконанні моделювання в головному вікні *MATLAB* з'являється попереджувальне повідомлення. Для усунення цього на непідключеними вхід блоку можна подати сигнал з блоку *Ground*.

На рисунку 4.15 наведено приклади використання блоку. У першому випадку сигнал з блоку *Ground* надходить на один з входів суматора, а в другому на один з входів блоку множення. показання блоків *Display* підтверджують, що виробляється блоком *Ground* сигнал має нульове значення.

З рисунка також видно, що тип вихідного сигналу блоку встановлюється автоматично, відповідно до типів сигналів, що подаються на інші входи блоків (в даному випадку – на входи блоків *Sum* і *Product*).

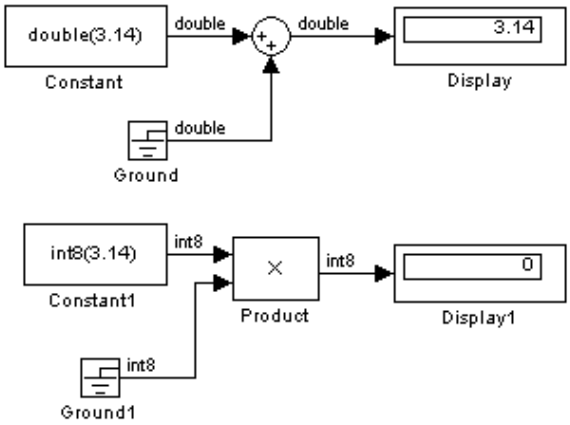


Рисунок 4.15 – Застосування блоку *Ground*

4.16 Блок періодичного сигналу *Repeating Sequence*

Призначення:

Формування періодичного сигналу.

Параметри:

Time values – вектор значень модельного часу.

Output values – вектор значень сигналу для моментів часу заданих вектором *Time values*.

Блок виконує лінійну інтерполяцію вихідного сигналу для моментів часу не збігаються зі значеннями заданими вектором *Time values*. На рисунку 4.16 показаний приклад використання блоку для формування пилкоподібної сигналу. Значення модельного часу задані вектором [0 3], а значення вихідного сигналу вектором [0 2].

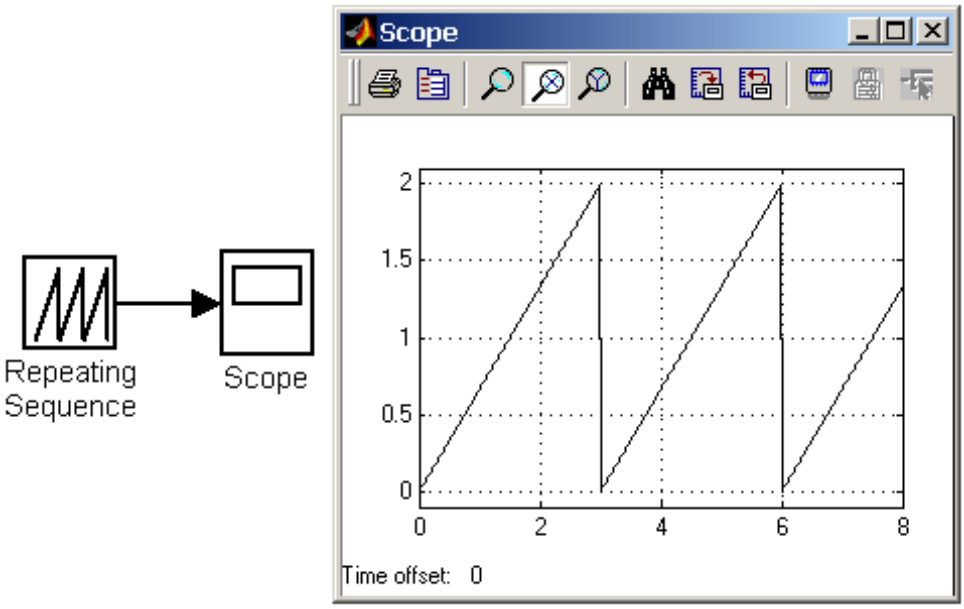


Рисунок 4.16 – Використання блоку *Repeating Sequence*

4.17 Блок вхідного порту *Inport*

Призначення:

Створює вхідний порт для підсистеми або моделі верхнього рівня ієрархії.

Параметри:

– *Port number* – номер порта;

– *Port dimensions* – розмірність вхідного сигналу. Якщо цей параметр дорівнює –1, то розмірність вхідного сигналу буде визначатися автоматично;

– *Sample time* –крок модельного часу.

– *Data type* – тип даних вхідного сигналу: *auto, double, single, int*8, *uint*8, *int*16, *uint*16, *int*32, *uint*32 або *Boolean;*

– *Signal type* – Тип вхідного сигналу:

– *auto* – автоматичне визначення типу;

– *real* – дійсний сигнал;

– *complex* – комплексний сигнал;

– *Interpolate data* (прапорець) – інтерполювати вхідний сигнал. У разі, якщо тимчасові відліки вхідного сигналу зчитуваного з робочої області *MATLAB* не збігаються з модельним часом, то блок буде виконувати інтерполяцію вхідного сигналу. При використанні блока *Inport* в підсистемі даний параметр не доступний.

4.17.1 Використання блоку *Inport* в підсистемах

Блоки Inport підсистеми є її входами. Сигнал, що подається на вхідний порт підсистеми через блок *Inport*, передається всередину підсистеми. Назва вхідного порту буде показано на зображенні підсистеми як мітка порту.

При створенні підсистем і додаванні блоку *Inport* в підсистему *Simulink* використовує наступні правила:

При створенні підсистеми за допомогою команди *Edit/Create subsystem* вхідні порти створюються і нумеруються автоматично починаючи з 1.

Якщо в підсистему додається новий блок *Inport*, то йому присвоюється наступний по порядку номер.

Якщо який або блок *Inport* видаляється, то інші порти перейменовуються таким чином, щоб послідовність номерів портів була безперервною.

Якщо в послідовності номерів портів є розрив, то при виконанні моделювання *Simulink* видасть повідомлення про помилку і зупинить розрахунок. У цьому випадку необхідно вручну перейменувати порти таким чином, щоб послідовність номерів портів не порушувалася.

На рисунку 4.17 показана модель, що використовує підсистему і схема цієї підсистеми.

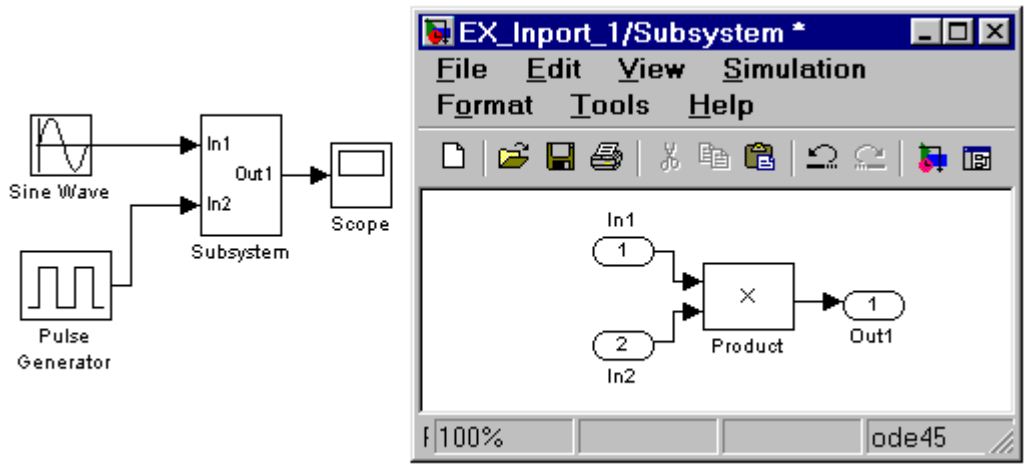


Рисунок 4.17 – Використання блоку *Inport* в підсистемі

4.17.2 Використання блоку *Inport* в моделі верхнього рівня

Вхідний порт в системі верхнього рівня використовується для передачі сигналу з робочої області *MATLAB* в модель.

Для передачі сигналу з робочого простору *MATLAB* потрібно не тільки встановити в моделі вхідний порт, але і виконати установку параметрів

введення на вкладці *Workspace* I/O вікна діалогу *Simulation parameters…* (має бути встановлений прапорець для параметра *Input* і задано ім'я змінної, яка містить вхідні дані). Тип даних, що вводяться: *Array* (масив), *Structure* (структура) або *Structure with time* (структура з полем "час") задається на цій же вкладці.

На рисунку 4.18 показана модель, що зчитує вхідний сигнал з робочого простору *MATLAB*. Підсистема *Load Data* забезпечує введення даних з файлу в робочу область *MATLAB.*

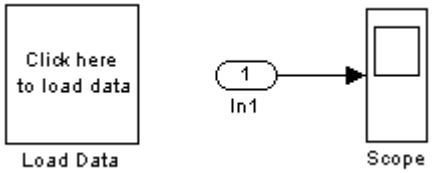


Рисунок 4.18 – Модель, що зчитує вхідний сигнал з робочого простору *MATLAB* за допомогою блоку *Input*

ПРАКТИЧНЕ ЗАНЯТТЯ №5

Дослідження блоків приймачів сигналів бібліотеки *Simulink* та їх параметрів

Мета роботи: дослідити блоки приймачів сигналів *Sources* бібліотеки *Simulink* пакета *MATLAB*, їх призначення, параметри.

5.1 Осцилограф *Scope*

Призначення:

Будує графіки досліджуваних сигналів в функції часу. Дозволяє спостерігати за змінами сигналів в процесі моделювання.

Зображення блоку і вікно для перегляду графіків показані на рисунку 5.1.

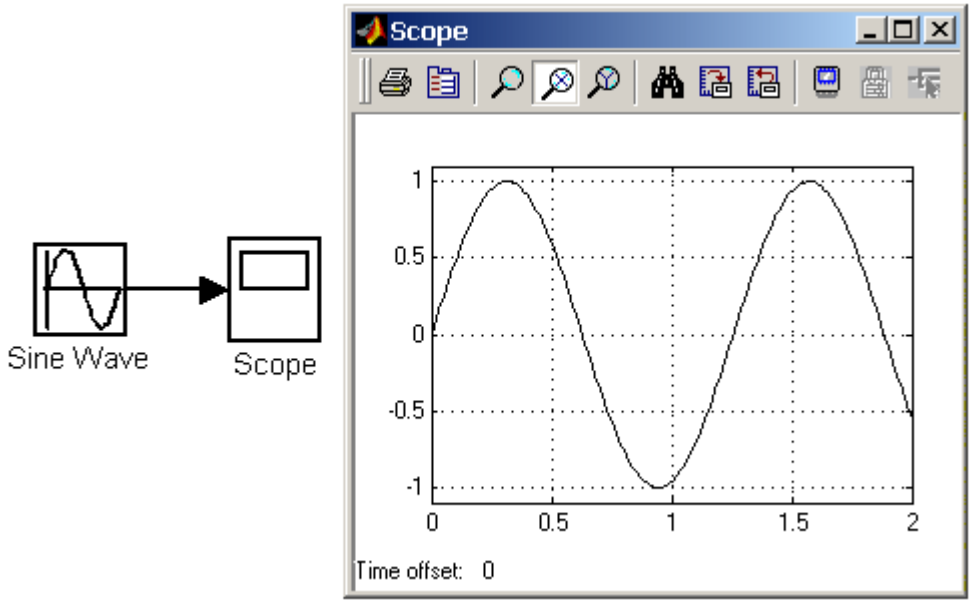


Рисунок 5.1 – Осцилограф *Scope*

Для того, щоб відкрити вікно перегляду сигналів необхідно виконати подвійне клацання лівою клавішею "миші" на зображенні блоку. Це можна зробити на будь-якому етапі розрахунку (як до початку розрахунку, так і після нього, а також під час розрахунку). У тому випадку, якщо на вхід блоку надходить векторний сигнал, то крива для кожного елемента вектора будується окремим кольором.

Налаштування вікна осцилографа виконується за допомогою панелей інструментів (рисунок 5.2).

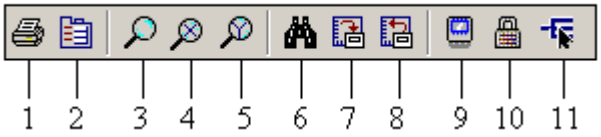


Рисунок 5.2 – Панель інструментів блоку *Scope*

Панель інструментів містить 11 кнопок:

– *Print* – друк вмісту вікна осцилографа

– *Parameters* – доступ до вікна налаштування параметрів;

– *Zoom* – збільшення масштабу по обох осях;

– *Zoom X-axis* – збільшення масштабу по горизонтальній осі;

– *Zoom Y-axis* – збільшення масштабу по вертикальній осі;

– *Autoscale* – автоматична установка масштабів по обох осях;

– *Save current axes settings* – сохранение текущих настроек окна;

– *Restore saved axes settings* – установка раніше збережених налаштувань вікна;

– *Floating scope* – переклад осцилографа в "вільний" режим;

– *Lock/Unlock axes selection* – закріпити / розірвати зв'язок між поточною координатною системою вікна і відображуваним сигналом. Інструмент доступний, якщо включений режим *Floating scope*;

– *Signal selection* – вибір сигналів для відображення. Інструмент доступний, якщо включений режим *Floating scope*.

Зміна масштабів відображуваних графіків можна виконувати кількома способами:

1. Натиснути відповідну кнопку (, або) і клацнути один раз лівою клавішею "миші" в потрібному місці графіка. Відбудеться 2,5 кратне збільшення масштабу.

2. Натиснути відповідну кнопку (, або) і, натиснувши ліву клавішу "миші", за допомогою динамічної рамки або відрізка вказати область графіка для збільшеного зображення. рисунок 5.3 пояснює цей процес.

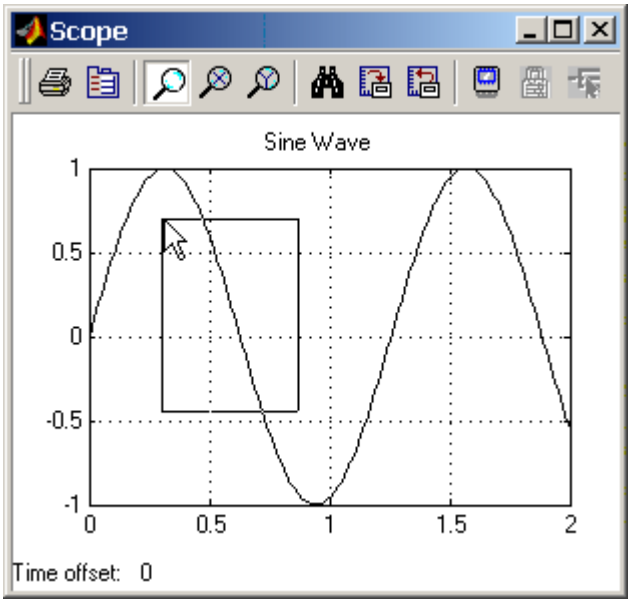


Рисунок 5.3 – Збільшення масштабу графіка

3. Клацнути правою клавішею "миші" у вікні графіків і, вибрати команду *Axes properties…* в контекстному меню. Відкриється вікно властивостей графіка, в якому за допомогою параметрів *Y-min* і *Y-max* можна вказати граничні значення вертикальної осі. У цьому ж вікні можна вказати заголовок графіка (*Title*), замінивши вираз %<*SignalLabel*> в рядку введення. Вікно властивостей показано на рисунку 5.4.

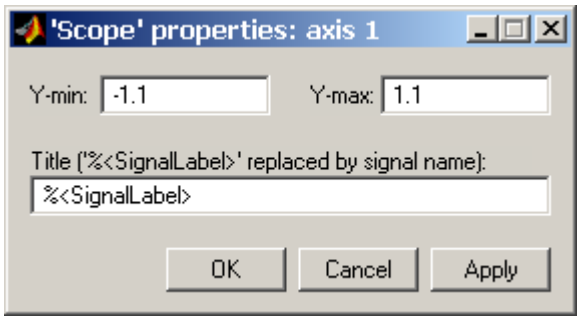


Рисунок 5.4 – Вікно властивостей графіка

Параметри:

Параметри блоку встановлюються у вікні ‘*Scope*’ parameters, яке відкривається за допомогою інструменту  (*Parameters*) панелі інструментів. Вікно параметрів має дві вкладки:

– *General* – загальні параметри;

– *Data history* – параметри збереження сигналів в робочій області *MATLAB*.

Вкладка загальних параметрів показана на рисунок 5.5.

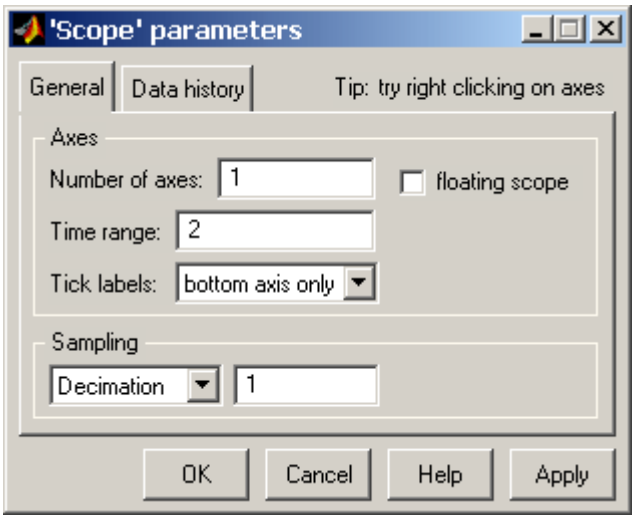


Рисунок 5.5 – Вкладка загальних параметрів *General*

На вкладці *General* задаються наступні параметри:

– *Number of axes* – число входів (систем координат) осцилографа. При зміні цього параметра на зображенні блоку з'являються додаткові вхідні порти;

– *Time range* – величина часового інтервалу для якого відображаються графіки. Якщо час розрахунку моделі перевищує задане параметром *Time range*, то висновок графіка проводиться порціями, при цьому інтервал відображення кожної порції графіка дорівнює заданому значенню *Time range*;

– *Tick labels* – висновок / приховування осей та міток осей. Може приймати три значення (вибираються зі списку):

– *all* – підписи для всіх осей;

– *none* – відсутність усіх осей і підписів до них;

– *bottom axis only* – підписи горизонтальній осі тільки для нижнього графіка;

– *Sampling* – установка параметрів виводу графіків у вікні. Задає режим виводу розрахункових точок на екран. При виборі Decimation кратність виведення встановлюється числом, що задає крок виводяться розрахункових точок. На рисунках 5.6 і 5.7. показані графіки синусоїдальних сигналів розрахованих з фіксованим кроком 0.1 с. На рисунку 5.6 у вікні блоку *Scope* виводиться кожна розрахункова точка (параметр *Decimation* дорівнює 1). На рисунку 5.7 показаний висновок кожного другого значення (параметр

Decimation дорівнює 2). Маркерами на графіках відзначені розрахункові точки. У тому випадку, якщо режим виводу розрахункових точок задається як *Sample time*, то його числове значення визначає інтервал квантування при відображенні сигналу. На рисунку 5.8 показаний графік синусоїдального сигналу, для випадку, коли значення параметра *Sample time* дорівнює 0.1;

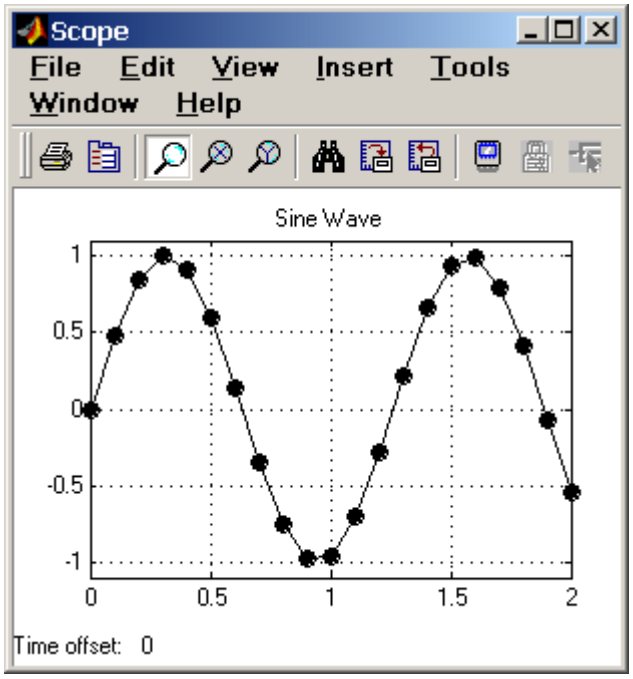


Рисунок 5.6 – Відображення синусоїдального сигналу (*Decimation*=1)

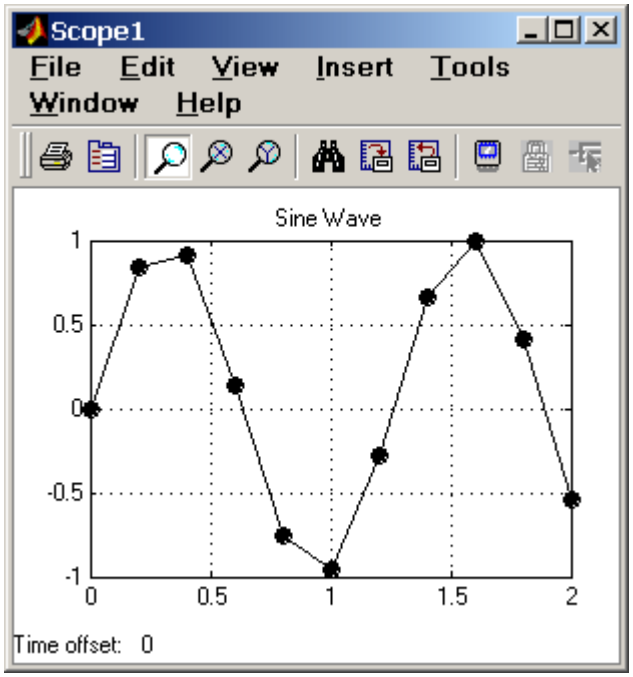


Рисунок 5.7 – Відображення синусоїдального сигналу (*Decimation*=2)

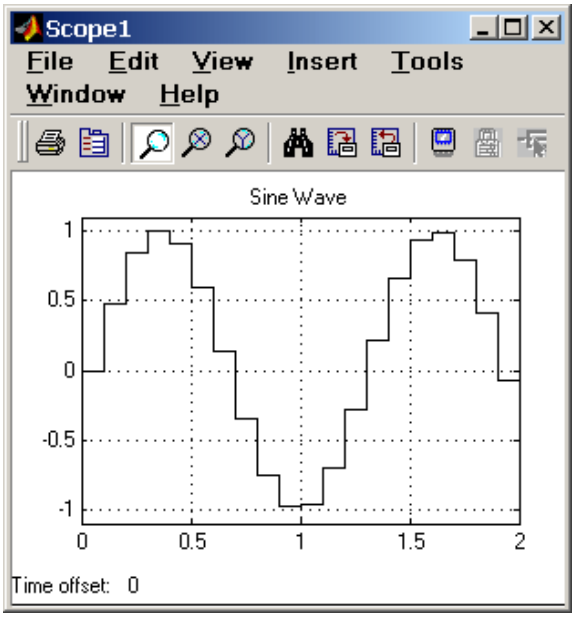


Рисунок 5.8 – Відображення синусоїдального сигналу (*Sample time*=0.1).

– *floating scope* – переклад осцилографа в "вільний" режим (при встановленому прапорці).

На вкладці *Data history* (рисунок 5.9) задаються наступні параметри:

– *Limit data points to last* – максимальна кількість відображуваних розрахункових точок графіка. При перевищенні цього числа початкова частина графіка обрізається. У тому випадку, якщо прапорець параметра *Limit data points to last* не встановлений, то *Simulink* автоматично збільшить значення цього параметра для відображення всіх розрахункових точок;

– *Save data to workspace* – збереження значень сигналів в робочій області *MATLAB*;

– *Variable name* – ім'я змінної для збереження сигналів в робочій області *MATLAB*;

– *Format* – формат даних при збереженні в робочій області *MATLAB*. Може приймати значення:

– *Array* – масив;

– *Structure* – структура;

– *Structure with time* – структура з додатковим полем ”час”.

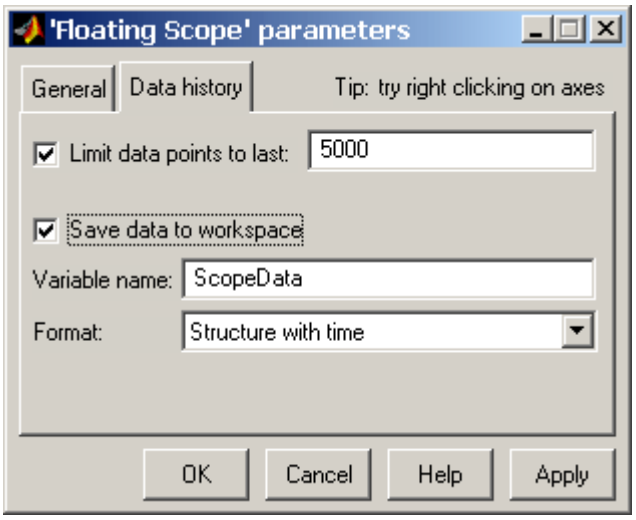


Рисунок 5.9 – Вкладка *Data history*

5.2 Осцилограф *Floating Scope*

Осцилограф *Floating Scope*, по суті, є звичайний осцилограф *Scope*, переведений в "вільний" режим. Приклад моделі з осцилографом *Floating Scope* показаний на рисунку 5.10.

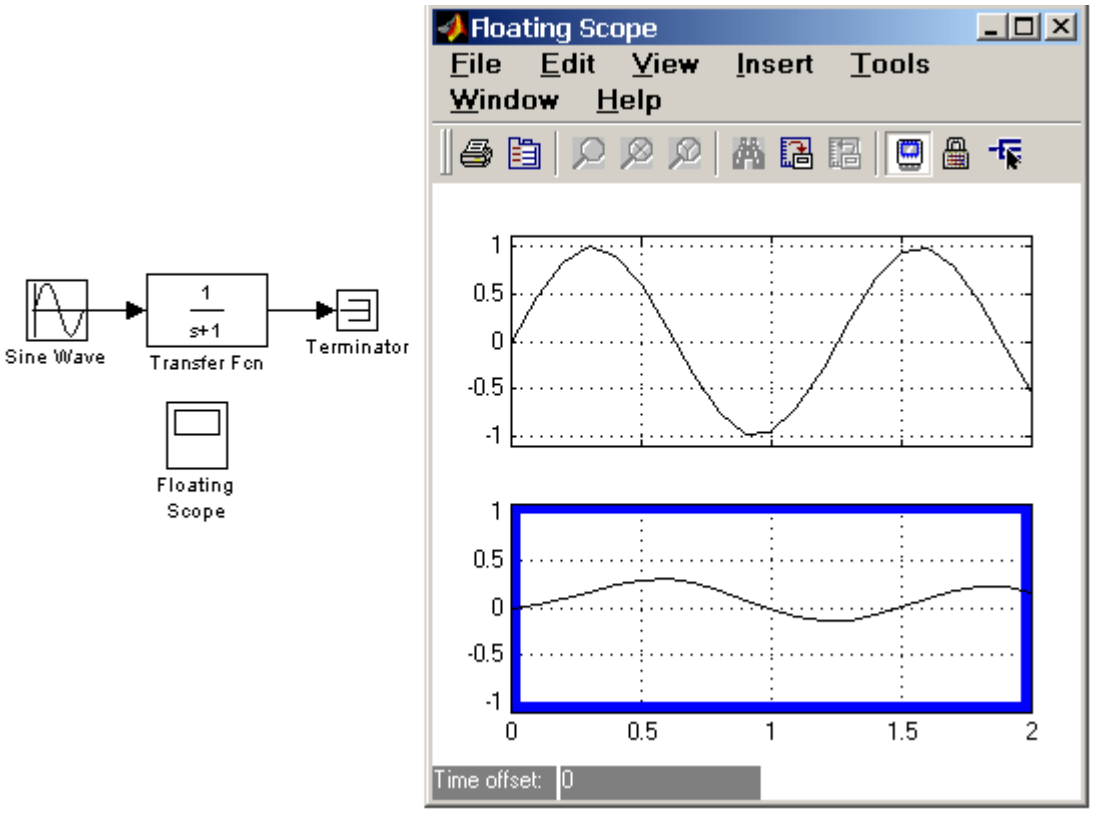


Рисунок 5.10 – Приклад моделі з осцилографом *Floating Scope*

У цьому режимі блок осцилографа не має входів, а вибрати весь сигналу здійснюється за допомогою інструменту (*Signal selection*) панелі інструментів. Для вибору сигналів необхідно виконати наступні дії:

1. Виділити систему координат, в якій буде відображатися графік. Це досягається за допомогою одиночного клацання лівою клавішею "миші" всередині потрібної системи. Обрана система координат буде підсвічена по периметру синім кольором.

2. За допомогою інструмента відкрити вікно діалогу *Signal Selector* (рисунок 5.11).

3. Відзначити прапорцем імена блоків, сигнали з виходу яких потрібно досліджувати.

Після виконання розрахунку у вікні блоку *Floating Scope* будуть відображені обрані сигнали.

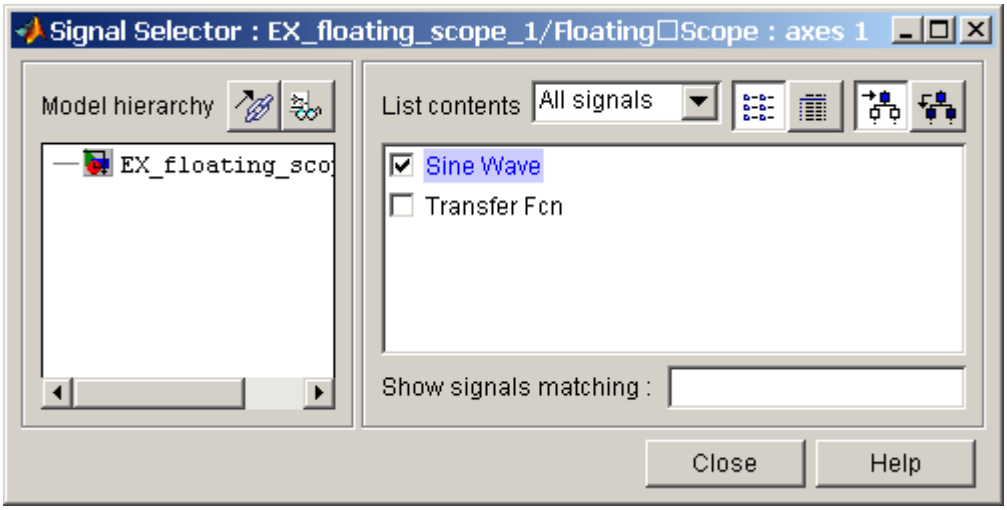


Рисунок 5.11 – вікно діалогу *Signal Selector*

5.3. Графобудівник *XY Graph*

Призначення:

Будує графік одного сигналу у функції іншого (графік виду *Y*(*X*)).

Параметри:

– *x-min* – Мінімальне значення сигналу по осі *X;*

– *x-max* – Максимальне значення сигналу по осі *X;*

– *y-min* – Мінімальне значення сигналу по осі *Y;*

– *y-max* – Максимальне значення сигналу по осі *Y;*

– *Sample time* – крок модельного часу.

Блок має два входи. Верхній вхід призначений для подачі сигналу, який є аргументом (*X*), нижній – для подачі значень функції (*Y*).

На рисунку 5.12, в якості прикладу використання графопостроителя, показано побудова фазової траєкторії коливального ланки.

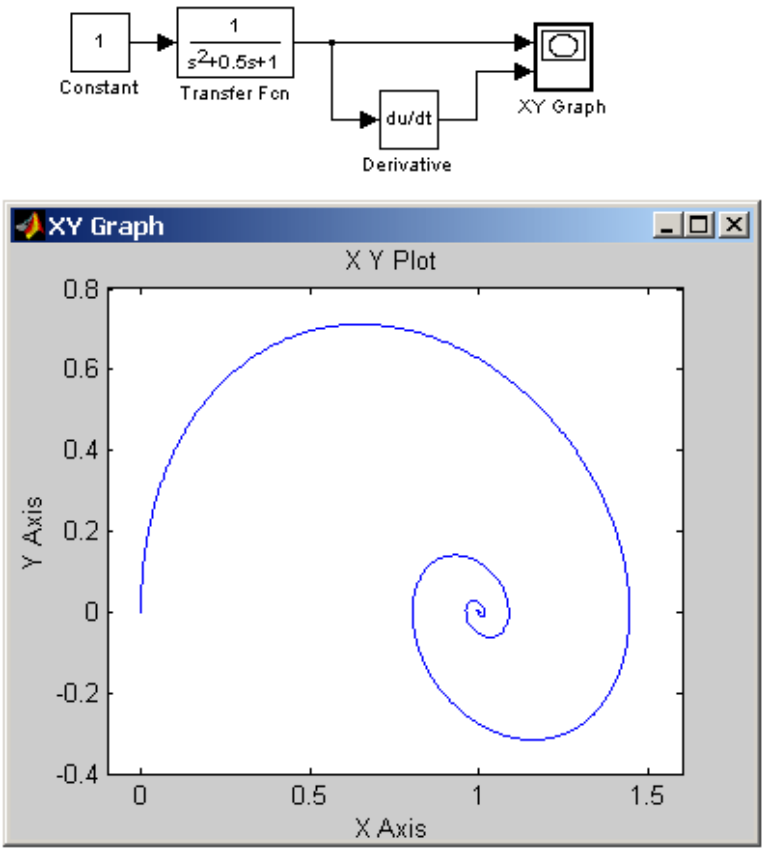


Рисунок 5.12 – Приклад використання графобудівника *XY Graph*

Графобудівник можна використовувати і для побудови тимчасових залежностей. Для цього на перший вхід слід подати тимчасової сигнал з виходу блоку *Clock*. Приклад такого використання графобудівника показаний на рисунку 5.13.

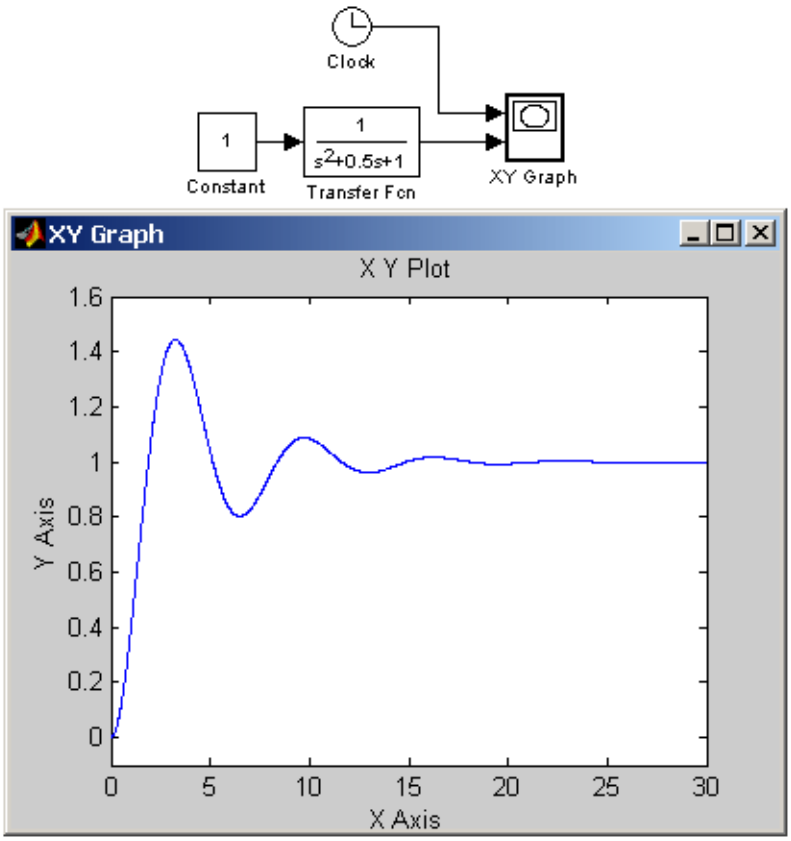


Рисунок 5.13 – Приклад використання блоку *XY Graph* для відображення тимчасових залежностей

5.4 Цифровий дисплей *Display*

Призначення:

Відображає значення сигналу у вигляді числа.

Параметри:

– *Format* – формат відображення даних. Параметр *Format* може приймати наступні значення:

– *short* – 5 значущих десяткових цифр;.

– *long* – 15 значущих десяткових цифр;

– *short_e* –5 значущих десяткових цифр і 3 символу ступеня десяти;

– *long_e* – 15 значущих десяткових цифр і 3 символу ступеня десяти;

– *bank* – "грошовий" формат. Формат з фіксованою точкою і двома десятковими цифрами в дробовій частині числа;

– *Decimation* – кратність відображення вхідного сигналу. При *Decimation* = 1 відображається кожне значення вхідного сигналу, при *Decimation* = 2 кожне

друге значення вхідного сигналу, при *Decimation* = 3 –кожне третє значення і т.д.;

– *Sample time* – крок модельного часу. Визначає дискретність відображення даних;

– *Floating display* (флажок) – перевод блоку в "вільний" режим. У даному режимі вхідний порт блоку відсутній, а вибір сигналу для відображення виконується клацанням лівої клавіші "миші" на відповідній линії зв'язку. У цьому режимі для параметра розрахунку *Signal storage reuse* повинно бути встановлено значення *off* (вкладка *Advanced* у вікні діалогу *Simulation parameters*…).

На рисунку 5.14 показано застосування блоку *Display* з використанням різних варіантів параметра *Format*.

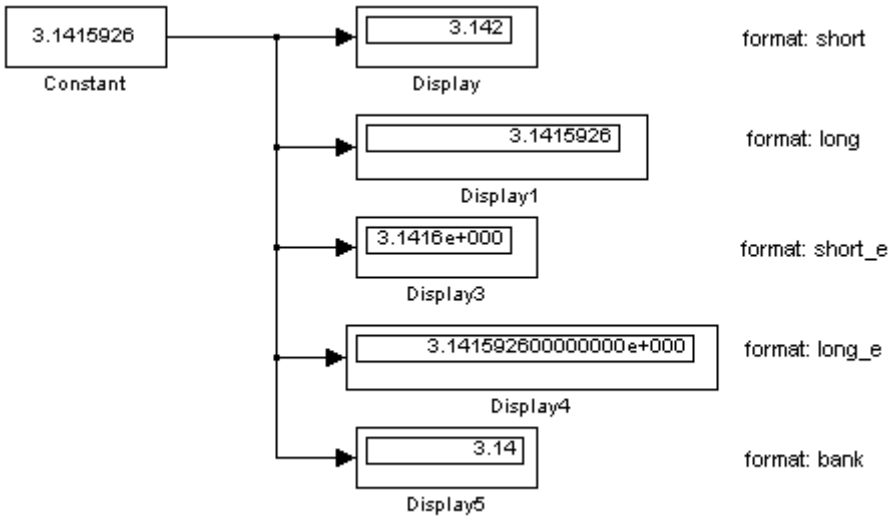


Рисунок 5.14 – Застосування блоку *Display* з використанням різних варіантів параметра *Format*

Блок *Display* може використовуватися для відображення не тільки скалярних сигналів, але також векторних, матричних і комплексних. рисунок 5.15 ілюструє це. Якщо всі відображувані значення не можуть поміститися у вікні блоку, то в правому нижньому кутку блоку з'являється символ, який вказує на необхідність збільшити розміри блоку (див.блок *Display* 4 на рисунку 5.15).

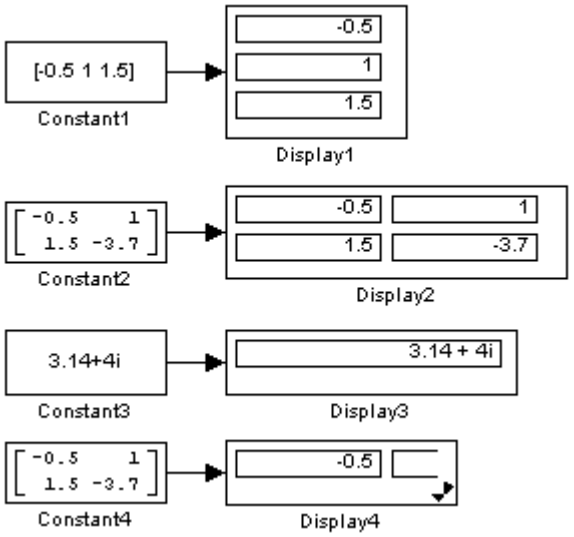


Рисунок 5.15 – Застосування блоку *Display* для відображення векторних, матричних і комплексних сигналів

5.5 Блок зупинення моделювання *Stop Simulation*

Призначення:

Забезпечує завершення розрахунку, якщо вхідний сигнал блоку стає не рівним нулю.

Параметри:

Немає.

При подачі на вхід блоку ненульового сигналу *Simulink* виконує поточний крок розрахунку, а потім зупиняє моделювання. Якщо на вхід блоку поданий векторний сигнал, то для зупинення розрахунку достатньо, щоб один елемент вектора став ненульовим. На рисунку 5.16 показаний приклад використання даного блоку. У прикладі зупинка розрахунку відбувається, якщо вихідний сигнал блоку *Transfer Function* стає більшим або рівним 0.99.

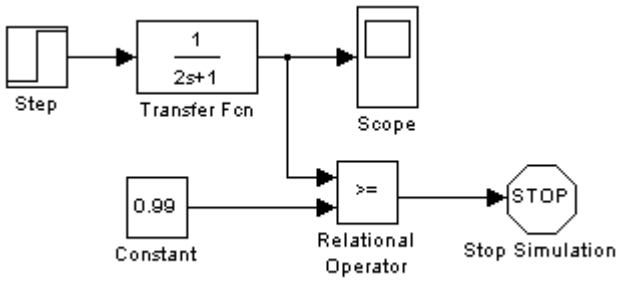


Рисунок 5.16 – Застосування блоку *Stop Simulation*

5.6 Блок збереження даних у файлі *To File*

Призначення:

Блок записує дані, що надходять на його вхід, в файл.

Параметри:

– *Filename* – ім'я файлу для запису. За замовчуванням файл має ім'я *untitled.mat*. Якщо не зазначений повний шлях до файлу, то файл зберігається у поточній робочій папці;

– *Variable name* – ім'я змінної, що містить записувані дані;

– *Decimation* – кратність запису в файл вхідного сигналу. При *Decimation* = 1 записується кожне значення вхідного сигналу, при *Decimation* = 2 записується кожне друге значення, при *Decimation* = 3 – кожне третє значення і т.д.;

– *Sample time* – крок модельного часу. Визначає дискретність запису даних.

Дані у файлі зберігаються у вигляді матриці:

$$\begin{bmatrix} t_1 & t_2 & \dots t_{final} \\ u1_1 & u1_2 & \dots u1_{final} \\ \dots & & \\ un_1 & un_2 & \dots un_{final} \end{bmatrix}.$$

Значення часу записуються в першому рядку матриці, а в інших рядках будуть знаходитися значення сигналів, відповідних даним моментів часу.

Файл даних (*mat*-файл), в який записуються дані, не є текстовим. Структура файлу докладно описана в довідковій системі *MATLAB.* Користувачам *Simulink* найзручніше зчитувати дані з mat-файлу за допомогою блоку *From File* (бібліотека *Sources*).

На рисунку 5.17 показаний приклад використання даного блоку. Результати розрахунку зберігаються у файлі *result.mat.*

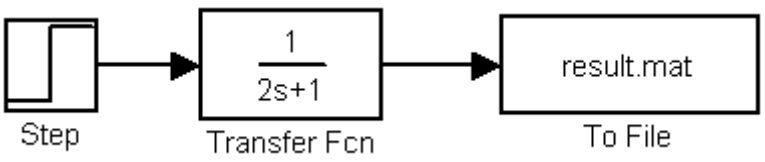


Рисунок 5.17 – Застосування блоку *To File*

5.7 Блок збереження даних у робочій області *To Workspace*

Призначення:

Блок записує дані, що надходять на його вхід, в робочу область *MATLAB*.

Параметри:

– *Variable name* – ім'я змінної, що містить записувані дані;

– *Limit data points to last* – максимальна кількість збережених розрахункових точок по часу (відлік ведеться від моменту завершення моделювання). У тому випадку, якщо значення параметра *Limit data points to last* задано як *inf*, то в робочій області будуть збережені всі дані;

– *Decimation* – кратність запису даних у робочу область;

– *Sample time* – крок модельного часу. Визначає дискретність запису даних;

– *Save format* – формат збереження даних. Може приймати значення:

– *Matrix* – матриця. Дані зберігаються як масив, в якому число рядків визначається числом розрахункових точок по часу, а число стовпців – розмірністю вектора подається на вхід блоку. Якщо на вхід подається скалярний сигнал, то матриця буде містити лише один стовпець;

– *Structure* – структура. Дані зберігаються у вигляді структури, що має три поля: *time* – час, *signals* – зберігаються значення сигналів, *blockName* – ім'я моделі та блоку *To Workspace*. Поле *time* для даного формату залишається не заповненим;

– *Structure with Time* – структура з додатковим полем (час). Для даного формату, на відміну від попереднього, поле *time* заповнюється значеннями часу.

На рисунку 5.18 показаний приклад використання даного блоку. Результати розрахунку зберігаються в змінній *simout*.

Для зчитування даних збережених в робочій області *MATLAB* можна використовувати блок *From Workspace* (бібліотека *Sources*).

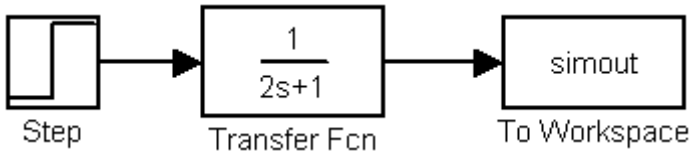


Рисунок 5.18 – Застосування блоку *To Workspace*

5.8 Кінцевий приймач *Terminator*

Призначення:

Блок використовується для подачі сигналу з невикористаного виходу іншого блоку.

Параметри:

Немає.

У тому випадку, якщо вихід блоку виявляється не підключеним до входу іншого блоку, *Simulink* видає попереджувала повідомлення в командному вікні *MATLAB*. Для виключення цього необхідно використовувати блок *Terminator*. На рисунку 5.19 показаний приклад використання кінцевого приймача. Витягуваний, за допомогою блоку *Demux*, з матриці другий елемент не ніяк не використовується, тому він подається на вхід блоку *Terminator*.

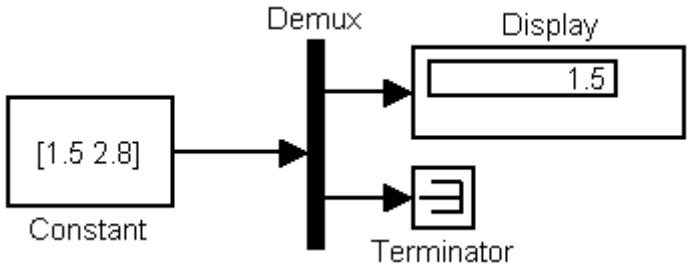


Рисунок 5.19 – Застосування блоку *Terminator*

5.9 Блок вихідного порту *Outport*

Призначення:

Створює вихідний порт для підсистеми або для моделі верхнього рівня ієрархії.

Параметри:

– *Port number* – номер порта;

– *Output when disabled* – вид сигналу на виході підсистеми, у разі якщо підсистема вимкнена. Використовується для керованих підсистем. Може приймати значення (вибираються зі списку):

– *held* – вихідний сигнал підсистеми дорівнює останньому розрахованому значенню;

– *reset* – вихідний сигнал підсистеми дорівнює значенню задаваемому параметром *Initial output*;

– *Initial output* – значення сигналу на виході підсистеми до початку її роботи і в разі, якщо підсистема вимкнена. Використовується для керованих підсистем.

5.4 Використання блоку *Outport* в підсистемах

Блоки *Outport* підсистеми є її виходами. Сигнал, що подається в блок *Outport* всередині підсистеми, передається в модель (або підсистему) верхнього рівня. Назва вихідного порту буде показано на зображенні підсистеми як мітка порту.

На рисунку 5.19 показана модель, що використовує підсистему і схема цієї підсистеми.

При створенні підсистем і додаванні блоку *Outport* в підсистему *Simulink* використовує наступні правила:

При створенні підсистеми за допомогою команди *Edit / Create subsystem* вихідні порти створюються і нумеруються автоматично починаючи з 1.

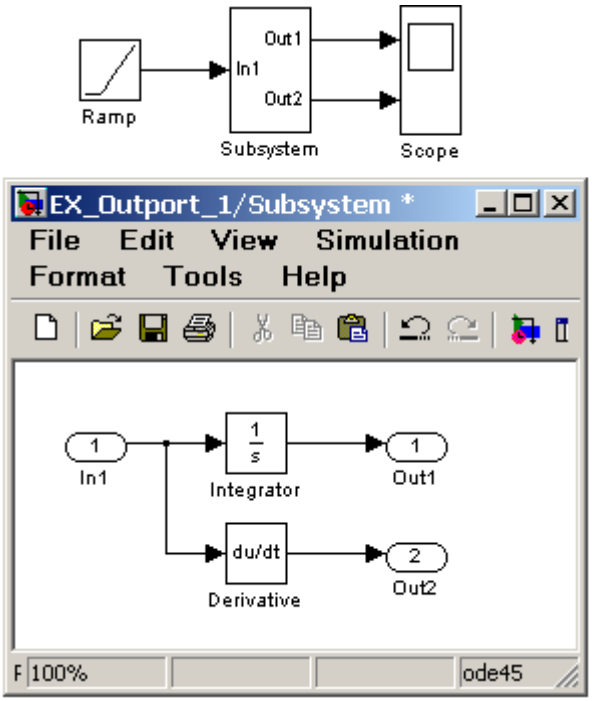


Рисунок 5.19. – Застосування блоку *Outport* в підсистемі

Якщо в підсистему додається новий блок *Outport*, то йому присвоюється наступний по порядку номер.

Якщо який або блок *Outport* віддаляється, то інші порти перейменовуються таким чином, щоб послідовність номерів портів була безперервною.

Якщо в послідовності номерів портів є розрив, то при виконанні моделювання *Simulink* видасть повідомлення про помилку і зупинить

розрахунок. У цьому випадку необхідно вручну перейменувати порти таким чином, щоб послідовність номерів портів не порушувалася.

У тому випадку, якщо підсистема є керованою, то для її вихідних портів можна задати вид вихідного сигналу для тих часових інтервалів, коли підсистема заблокована. На рисунку 5.20 показана модель, що використовує керовану підсистему (схема підсистеми така ж, як і в попередньому прикладі). Для першого вихідного порту підсистеми параметр *Output when disabled* заданий як *held*, а для другого – як *reset*, причому величина початкового значення задана рівною нулю. Графіки сигналів показують, що коли підсистема заблокована, сигнал першого вихідного порту залишається незмінним, а сигнал другої стає рівним заданому початковому значенням (нулю).

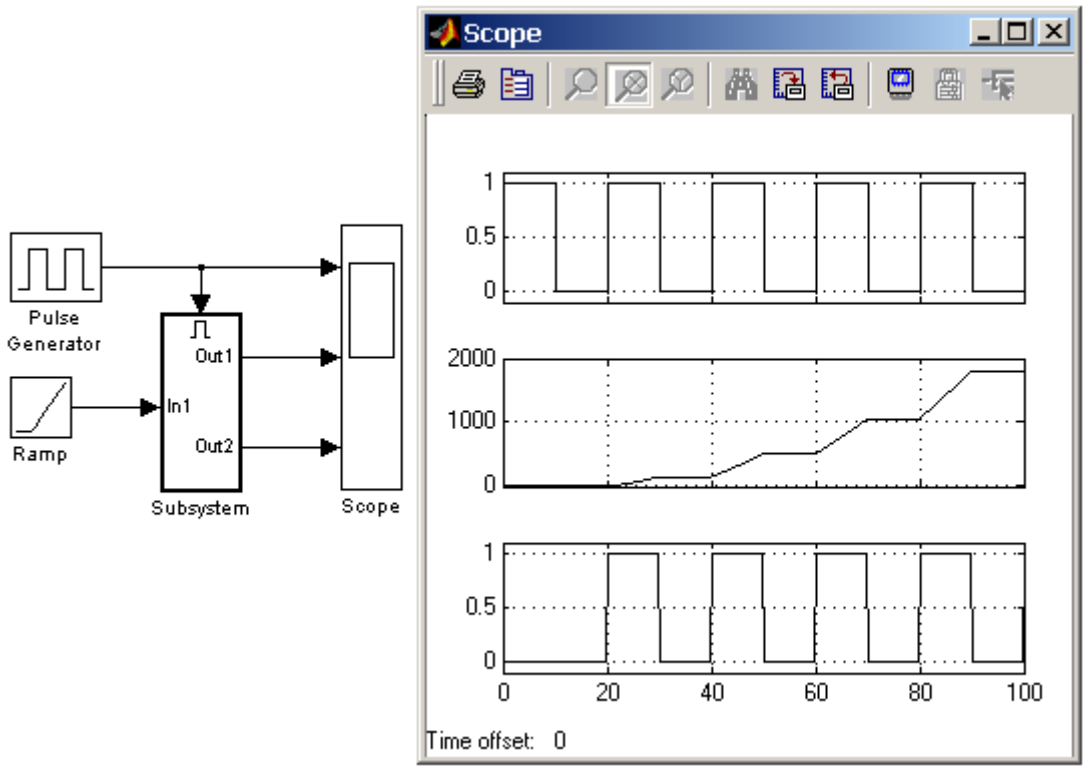


Рисунок 5.20 – Керована підсистема з різними настройками вихідних портів

5.5 Використання блоку *Outport* в моделі верхнього рівня

Вихідний порт в системі верхнього рівня використовується у двох випадках:

– для передачі сигналу в робочий простір *MATLAB*;

– для забезпечення зв'язку функцій аналізу з виходами моделі.

Для передачі сигналу в робочий простір *MATLAB* потрібно не тільки встановити в моделі вихідні порти, але і виконати установку параметрів виведення на вкладці *Workspace* I/O вікна діалогу *Simulation parameters*… (має бути встановлений прапорець для параметра *Output* і задано ім'я змінної для збереження даних). Тип даних, що зберігається – *Array* масив, *Structure* (структура) або *Structure with time* (структура з полем "час") задається на цій же вкладці.

На рисунку 5.21 показана модель, що передає сигнали в робочий простір *MATLAB*.

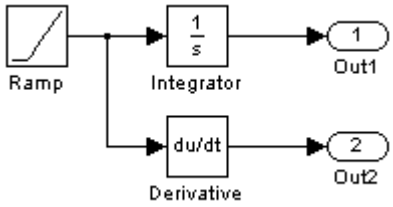


Рисунок 5.21 – Модель, що передає сигнали в робочий простір *MATLAB* за допомогою блоків *Outport*

Блок *Outport* може використовуватися також для зв'язку моделі з функціями аналізу, наприклад: *linmod* или *trim*.

ПРАКТИЧНЕ ЗАНЯТТЯ №6

Дослідження аналогових блоків *Continuous* бібліотеки *Simulink* та їх параметрів

Мета роботи: дослідити аналогові блоки *Continuous* бібліотеки *Simulink* пакета *MATLAB*, их призначення, параметри.

6.1 Блок обчислення похідної *Derivative*

Призначення:

Виконує чисельне диференціювання вхідного сигналу.

Параметри:

Немає.

Для обчислення похідної використовується наближена формула Ейлера:

$$\frac{\partial u}{\partial t} = \frac{\Delta u}{\Delta t},$$

де u – величина зміни вхідного сигналу за час t,

t – поточне значення кроку модельного часу.

Значення вхідного сигналу блоку до початку розрахунку вважається рівним нулю. Початкове значення вихідного сигналу також покладається рівним нулю.

Точність обчислення похідної істотно залежить від величини встановленого кроку розрахунку. Вибір меншого кроку розрахунку покращує точність обчислення похідної.

На рисунку 6.1 показаний приклад використання диференціюючого блоку для обчислення похідної прямокутного сигналу. У розглянутому прикладі, для підвищення наочності, крок розрахунку обраний досить великим.

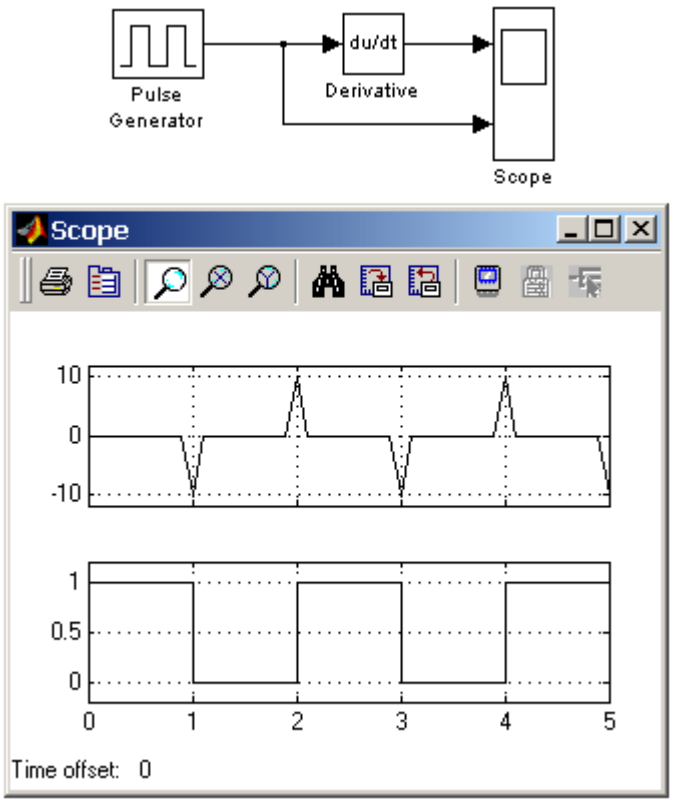


Рисунок 6.1 – Використання блоку *Derivative*
для диференціювання сигналу

Даний блок використовується для диференціювання аналогових сигналів. При диференціюванні дискретного сигналу за допомогою блоку *Derivative* його вихідний сигнал буде являти собою послідовність імпульсів відповідних моментів часу стрибкоподібної зміни дискретного сигналу.

6.2 Інтегруючий блок *Integrator*

Призначення:

Виконує інтегрування вхідного сигналу.

Параметри:

– *External reset* – зовнішнє скидання. Тип зовнішнього керуючого сигналу, що забезпечує скидання інтегратора до початкового стану. Вибирається зі списку:

– *none* – немає (скидання не виконується);

– *rising* – наростаючий сигнал (передній фронт сигналу);

– *falling* – спадаючий сигнал (задній фронт сигналу);

– *either* – наростаючий або спадаючий сигнал;

– *level* – не нульовий сигнал (скид виконується якщо сигнал на керуючому вході стає не рівним нулю);

У тому випадку, якщо обраний небудь (але не *none*), тип керуючого сигналу, то на зображенні блоку з'являється додатковий керуючий вхід. Поруч з додатковим входом буде показано умовне позначення керуючого сигналу.

– *Initial condition source* – Джерело початкового значення вихідного сигналу. Вибирається зі списку:

– *internal* – внутрішній;

– *external* – зовнішній. У цьому випадку на зображенні блоку з'являється додатковий вхід, позначений x_0, на який необхідно подати сигнал задаючий Початкове значення вихідного сигналу інтегратора;

– *Initial condition* – Початкова умова. Установка початкового значення вихідного сигналу інтегратора. Параметр доступний, якщо обраний внутрішній джерело початкового значення вихідного сигналу;

– *Limit output* (прапорець) – Використання обмеження вихідного сигналу;

– *Upper saturation limit* – Верхній рівень обмеження вихідного сигналу. Може бути заданий як числом, так і символьної послідовністю *inf*, тобто + ;

– *Lower saturation limit* – Нижній рівень обмеження вихідного сигналу. Може бути заданий як числом, так і символьної послідовністю *inf*, тобто – ;

– *Show saturation port* – управляє відображенням порту, котрий виводить сигнал, що свідчить про вихід інтегратора на обмеження. Вихідний сигнал даного порту може приймати наступні значення:

Нуль, якщо інтегратор не знаходиться на обмеженні;

+1, якщо вихідний сигнал інтегратора досяг верхнього обмежує межі;

–1, якщо вихідний сигнал інтегратора досяг нижнього обмежує межі;

– *Show state port* (флажок) – Відобразити / приховати порт стану блоку. Даний порт використовується в тому випадку, якщо вихідний сигнал інтегратора потрібно подати в якості сигналу зворотного зв'язку цього ж інтегратора. На приклад, при установці початкових умов через зовнішній порт або при скиданні інтегратора через порт скидання. Вихідний сигнал з цього порту може використовуватися також для організації взаємодії з керованою підсистемою;

– *Absolute tolerance* – Абсолютна похибка.

На рисунку 6.2 показаний приклад роботи інтегратора при подачі на його вхід ступінчастого сигналу. Початкова умова прийнято рівним нулю.

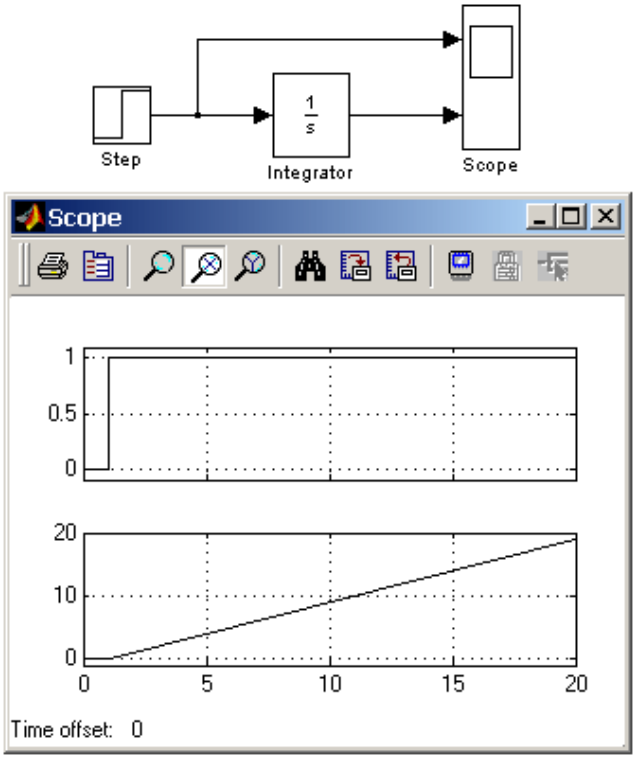


Рисунок 6.2 – Інтегрування ступінчастого сигналу

Приклад на рисунку 6.3 відрізняється від попереднього подачею початкового значення через зовнішній порт. Початкове значення вихідного сигналу в цьому прикладі задано рівним –10.

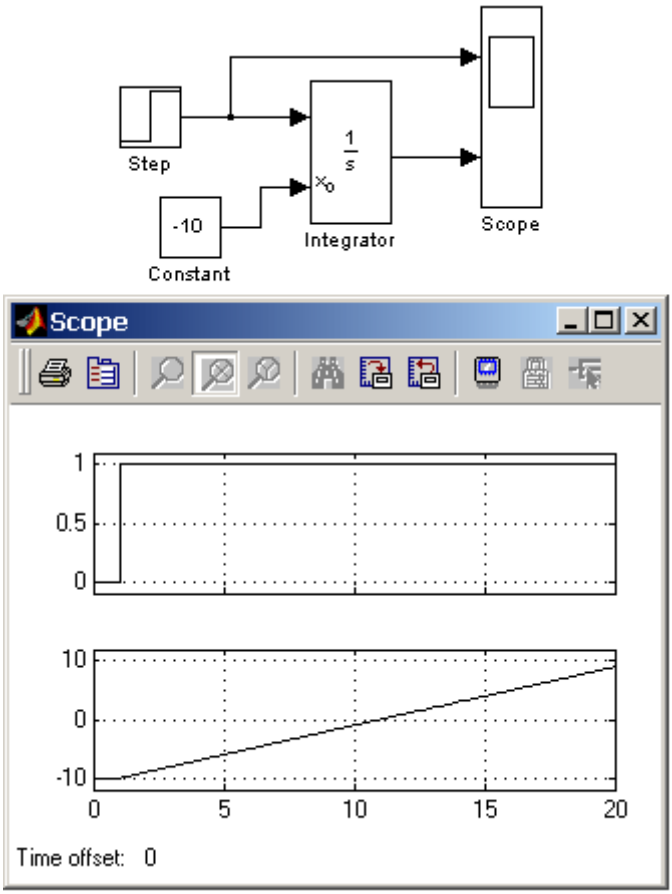


Рисунок 6.3 – Інтегрування ступінчастого сигналу з установкою початкового значення вихідного сигналу

Приклад на рисунку 6.4 демонструє використання вхідного порту для скидання вихідного сигналу і порту стану інтегратора з метою організації зворотного зв'язку. Схема працює в такий спосіб: вхідний постійний сигнал перетвориться інтегратором в лінійно-змінюється, по досягненні вихідним сигналом значення рівного 1 блок *Relational Operator* виробляє логічний сигнал, по передньому фронту якого відбувається скидання вихідного сигналу інтегратора до початкового значення рівного нулю. У результаті на виході інтегратора формується пилкоподібний сигнал, що змінюється від 0 до +1.

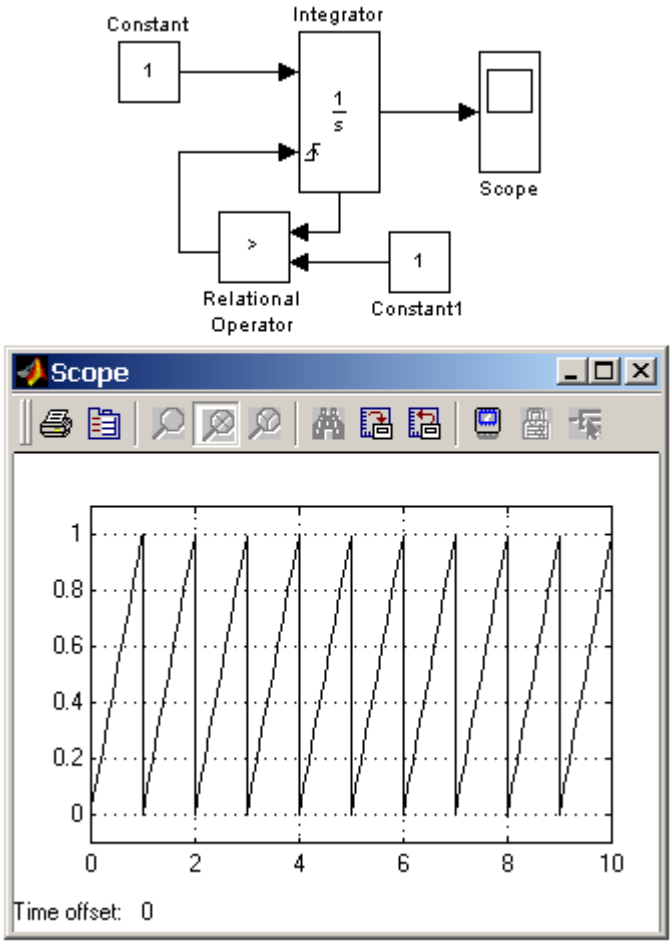


Рисунок 6.4 – Генератор пилоподібного сигналу на основі інтегратора

Наступна схема (рисунок 6.5) використовує установку початкового значення інтегратора за допомогою його вихідного сигналу. У перший момент часу Початкове значення вихідного сигналу інтегратора за допомогою блоку *IC (Initial Condition)* встановлюється рівним нулю. По досягненні вихідним сигналом значення рівного 1 блок *Relational Operator* подає сигнал скидання вихідного сигналу інтегратора на початковий рівень, при цьому сигналом, що задає початковий рівень, виявляється інвертований вихідний сигнал інтегратора (тобто –1). Далі цикл роботи схеми повторюється. На відміну від попередньої схеми вихідним сигналом генератора є двуполярний сигнал.

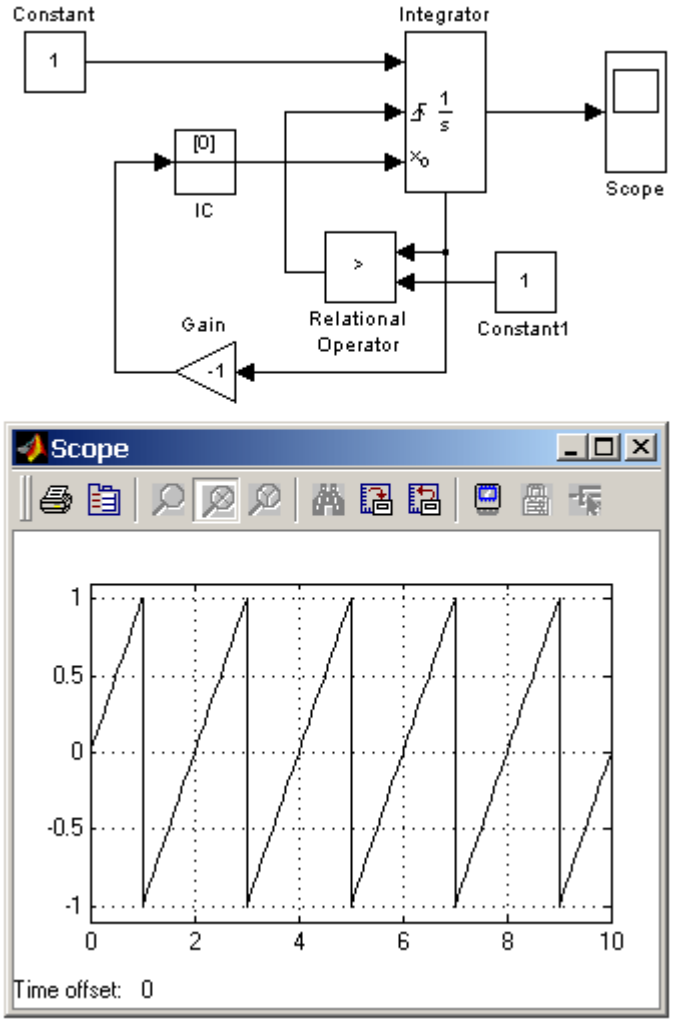


Рисунок 6.5 – Генератор двополярної пилоподібного сигналу на основі інтегратора

6.3 Блок *Memory*

Призначення:

Виконує затримку вхідного сигналу на один часовий такт.

Параметри:

– *Initial condition* – Початкове значення вихідного сигналу;

– *Inherit sample time* (флажок) – Успадковувати крок модельного годині. Якщо цей прапорець встановлений, то блок *Memory* використовує крок модельного годині (*Sample time*) такий же, як і в попередньому блоці.

На рисунку 6.6 показаний приклад використання блоку *Memory* для затримки дискретного сигналу на один часовий такт.

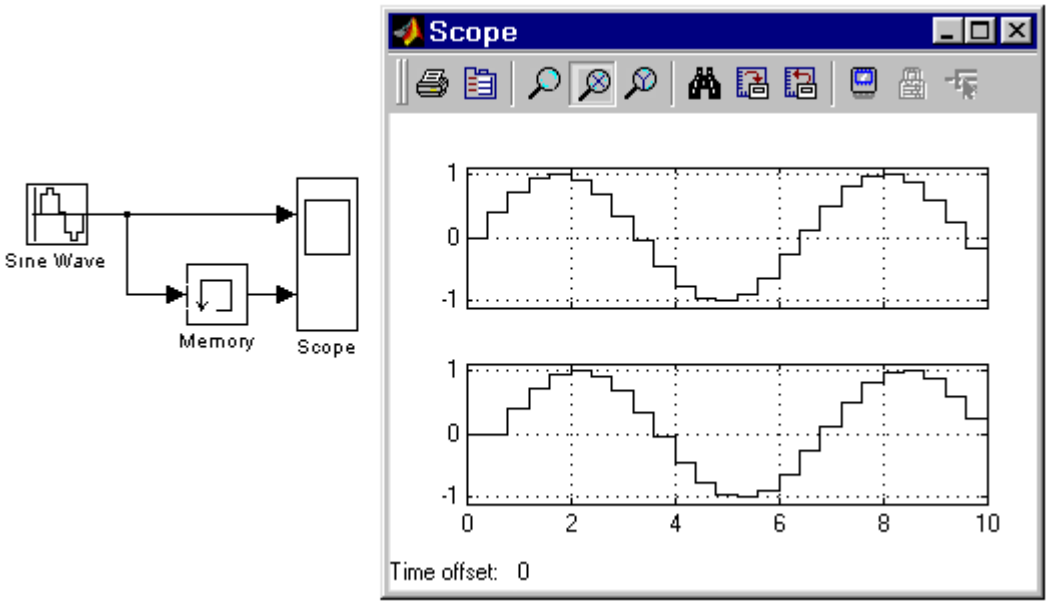


Рисунок 6.6 – Застосування блоку для затримки сигналу на один часовий такт

6.4 Блок фіксованої затримки сигналу *Transport Delay*

Призначення:

Забезпечує затримку вхідного сигналу на заданий час.

Параметри:

– *Time Delay* – Час затримки сигналу (не від'ємне значення);

– *Initial input* – Початкове значення вихідного сигналу;

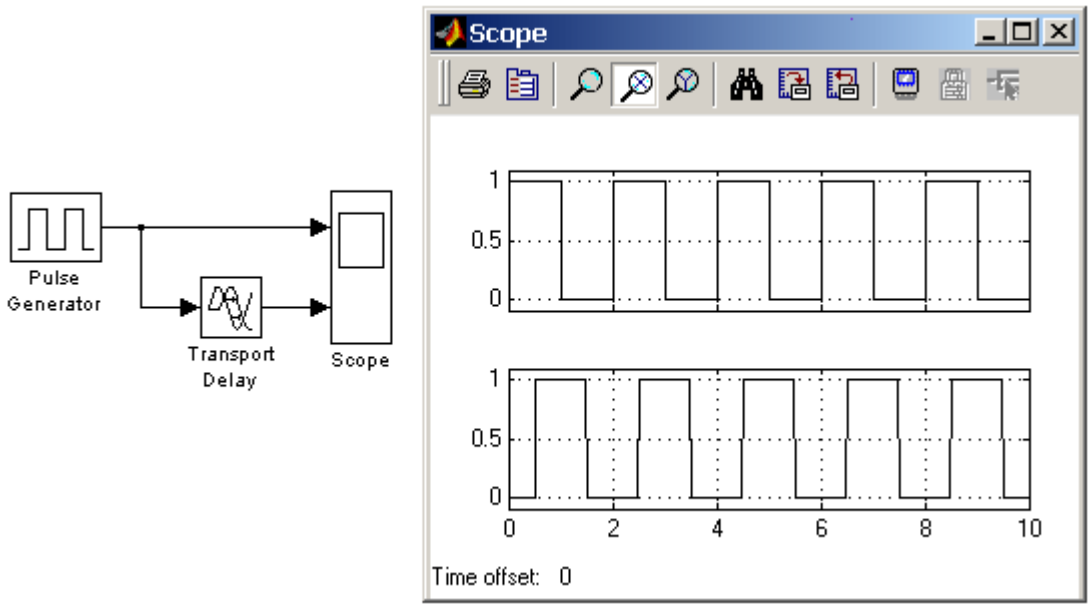
– *Buffer size* – Розмір пам'яті, що виділяється для зберігання затриманого сигналу. Здається в байтах числом, кратним 8 (за замовчуванням 1024);

– *Pade order* (*for linearization*) – Порядок ряду Паде, використовуваного при апроксимації вихідного сигналу. Здається цілим позитивним числом.

При виконанні моделювання значення сигналу і відповідне йому модельне час зберігаються у внутрішньому буфері блоку *Transport Delay*. Після закінчення часу затримки значення сигналу, витягується з буфера і передається на вихід блоку. У тому випадку, якщо кроки модельного часу не збігаються зі значеннями моментів часу для записаного в буфер сигналу, блок *Transport Delay* виконує апроксимацію вихідного сигналу.

У тому випадку, якщо початкового значення обсягу пам'яті буфера не вистачить для зберігання затриманого сигналу, *Simulink* автоматично виділить додаткову пам'ять. Після завершення моделювання в командному вікні *MATLAB* з'явиться повідомлення із зазначенням потрібного розміру буфера.

На рисунку 6.7 показаний приклад використання блоку *Transport Delay* для затримки прямокутного сигналу на 0.5 с.



Ррисунок 6.7 – Приклад використання блоку *Transport Delay* для затримки сигналу

6.5 Блок керованої затримки сигналу *Variable Transport Delay*

Призначення:

Виконує затримку вхідного сигналу, задану величиною сигналу управління.

Параметри:

– *Maximum delay* – Максимальне значення часу затримки сигналу (не від'ємне значення);

– *Initial input* – Початкове значення вихідного сигналу;

– *Buffer size* – Розмір пам'яті, що виділяється для зберігання затриманого сигналу. Здається в байтах числом, кратним 8 (за замовчуванням 1024);

– *Pade order* (*for linearization*) – Порядок ряду Паде, використовуваного при апроксимації вихідного сигналу. Здається цілим позитивним числом.

Блок керованої затримки *Variable Transport Delay* працює аналогічно блоку постійної затримки сигналу *Transport Delay*.

У тому випадку, якщо значення керуючого сигналу задає величину затримки перевищує значення, задане параметром *Maximum delay*, то затримка виконується на величину *Maximum delay*.

На рисунку 6.8 показаний приклад використання блоку *Variable Transport Delay*. Величина часу затримки сигналу змінюється від 0.5с до 1с в момент часу рівний 5с.

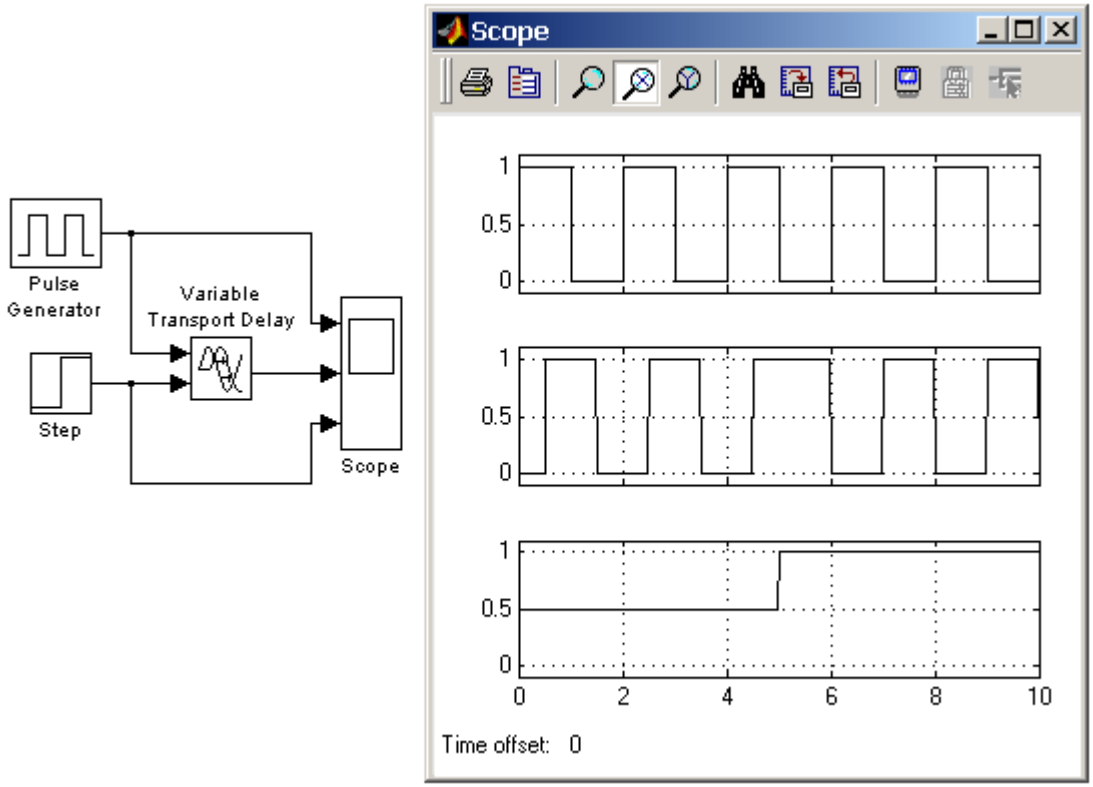


Рисунок 6.8 – Приклад використання блоку *Variable Transport Delay*

6.6 Блок передавальної функції *Transfer Fcn*

Призначення:

Блок передавальної характеристики *Transfer Fcn* задає передатну функцію у вигляді відношення поліномів:

$$H(s)=\frac{y(s)}{u(s)}=\frac{num(s)}{den(s)}=\frac{num(1)s^{nn-1}+num(2)s^{nn-2}+...+num(nn)}{den(1)s^{nd-1}+den(2)s^{nd-2}+...+den(nd)},$$

де *nn* і *nd* – порядок чисельника і знаменника передавальної функції,

num – вектор або матриця коефіцієнтів чисельника,

den – вектор коефіцієнтів знаменника.

Параметри:

– *Numerator* – вектор або матриця коефіцієнтів полінома чисельника;

– *Denominator* – вектор коефіцієнтів полінома знаменника;

– *Absolute tolerance* – абсолютна похибка.

Порядок чисельника не повинен перевищувати порядок знаменника.

Вхідний сигнал блоку повинен бути скалярним. У тому випадку, якщо коефіцієнти чисельника задані вектором, то вихідний сигнал блоку буде також скалярним (як і вхідний сигнал). На рисунку 6.8 показаний приклад моделювання коливальної ланки за допомогою блоку *Transfer Fcn*.

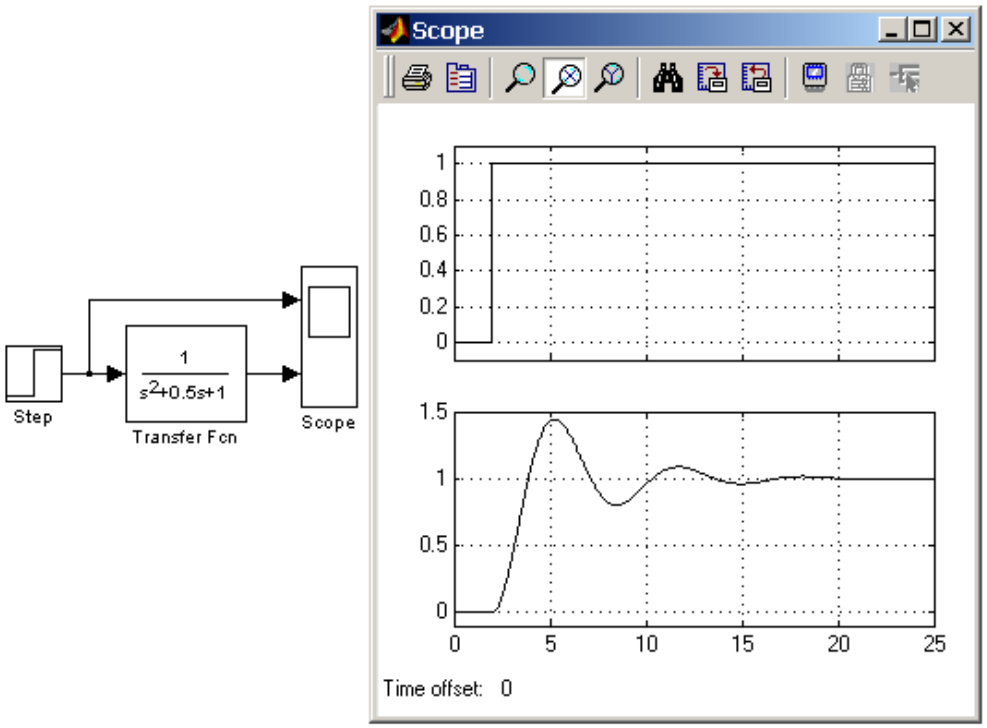


Рисунок 6.8 – Приклад моделювання коливального ланки

Якщо коефіцієнти чисельника задані матрицею, то блок *Transfer Fcn* моделює векторну передатну функцію, яку можна інтерпретувати як кілька передавальних функцій мають однакові поліноми знаменника, але різні поліноми чисельника. При цьому вихідний сигнал блоку є векторним і кількість рядків матриці чисельника задає розмірність вихідного сигналу.

На рисунку 6.9 показаний приклад блоку *Transfer Fcn* задаючий векторну передатну функцію. Там же показана модель повністю аналогічна розглянутій за своїми властивостями, але складається з окремих блоків *Transfer Fcn*.

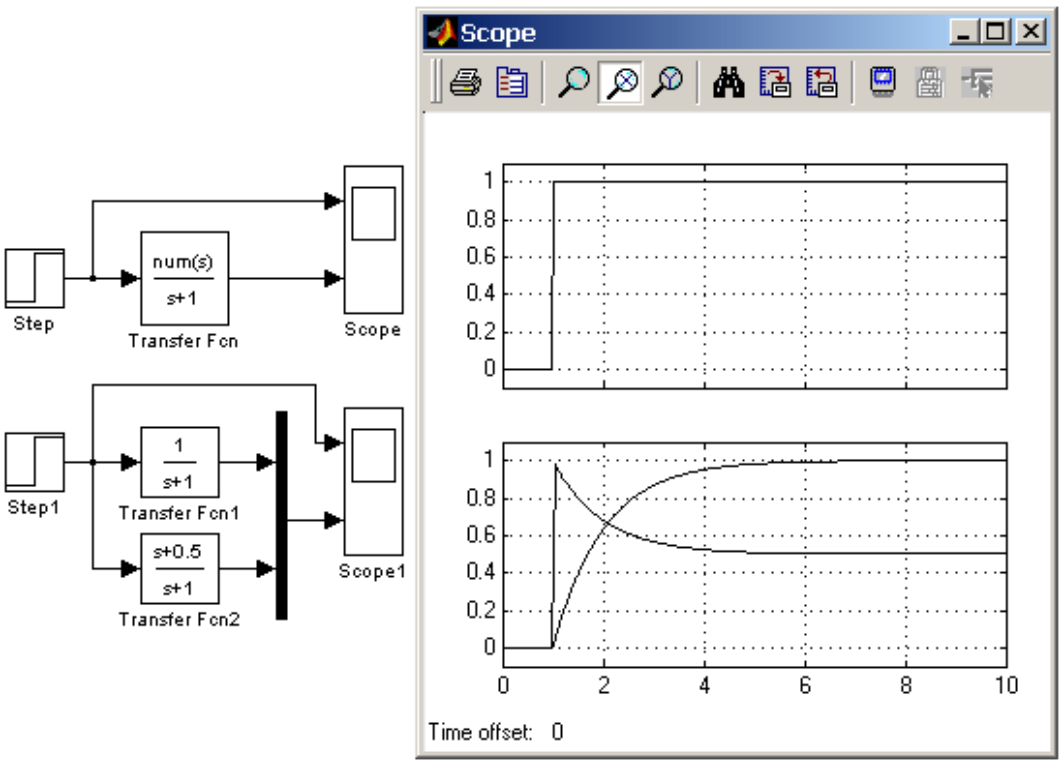


Рисунок 6.9 – Приклад моделювання векторної передавальної функції і її аналог

Початкові умови при використанні блоку *Transfer Fcn* покладаються нульовими. Якщо ж потрібно, щоб початкові умови не були нульовими, то необхідно за допомогою функції *tf2ss* (інструмент *Control System Toolbox)* перейти від передавальної функції до моделі в просторі станів і моделювати динамічний об'єкт за допомогою блоку *State-Space*.

6.7 Блок передавальної функції *Zero-Pole*

Призначення:

Блок *Zero-Pole* визначає передатну функцію із заданими полюсами і нулями:

$$H(s) = K\frac{Z(s)}{P(s)} = K\frac{(s-Z(1))(s-Z(2))...(s-Z(m))}{(s-P(1))(s-P(2))...(s-P(n))},$$

де Z – вектор або матриця нулів передавальної функції (коренів полінома чисельника),

P – вектор полюсів передавальної функції (коренів полінома знаменника),

K – коефіцієнт передавальної функції, або вектор коефіцієнтів, якщо нулі передавальної функції задані матрицею. При цьому розмірність вектора *K* визначається числом рядків матриці нулів.

Параметри:

– *Zeros* – вектор або матриця нулів;

– *Poles* – вектор полюсів;

– *Gain* – скалярний або векторний коефіцієнт передавальної функції;

– *Absolute tolerance* – абсолютна похибка.

Кількість нулів не повинна перевищувати число полюсів передавальної функції.

У тому випадку, якщо нулі передавальної функції задані матрицею, то блок *Zero-Pole* моделює векторну передатну функцію.

Нулі або полюси можуть бути задані комплексними числами. У цьому випадку нулі або полюса повинні бути задані комплексно-сполученими парами полюсів або нулів, відповідно.

Початкові умови при використанні блоку *Zero-Pole* покладаються нульовими. На рисунку 6.10 показаний приклад використання блоку *Zero-Pole*. У прикладі передавальна функція має один дійсний нуль і два комплексно-сполучених полюса.

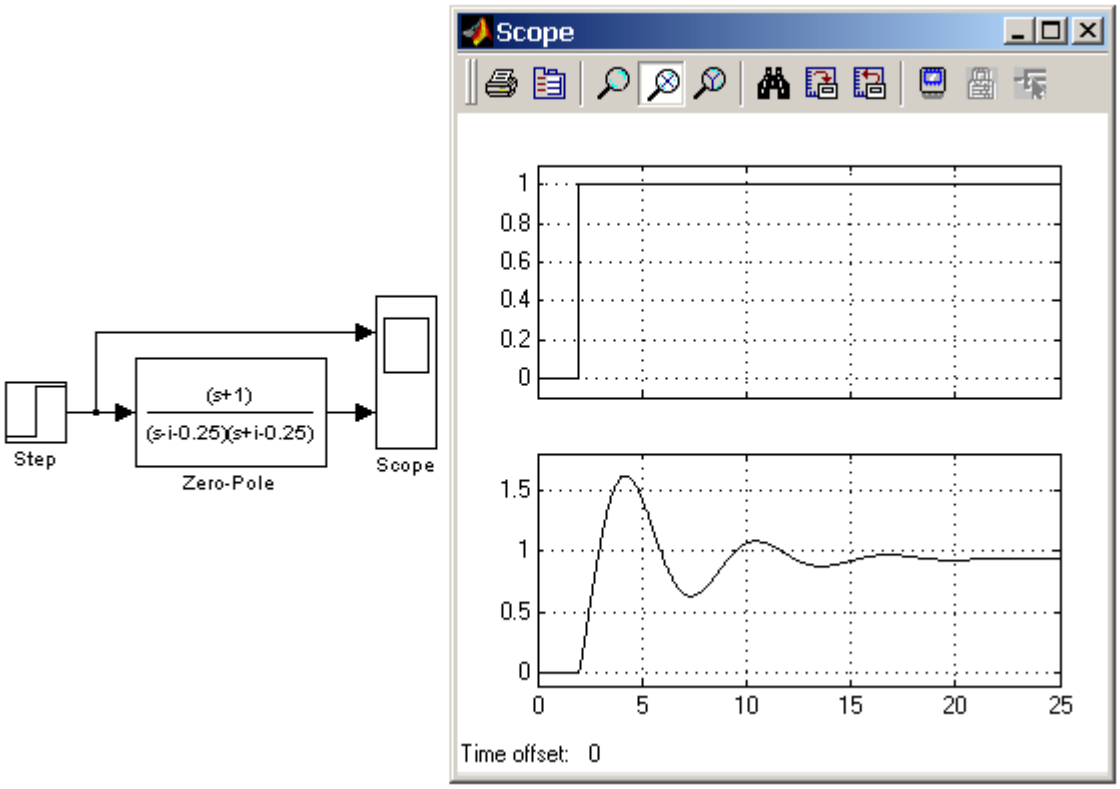


Рисунок 6.10 – Приклад використання блоку *Zero-Pole*

6.8 Блок моделі динамічного об'єкта *State-Space*

Призначення:

Блок створює динамічний об'єкт, описуваний рівняннями в просторі станів:

$$x = Ax + Bu\,,$$

$$y = Cx + Du\,,$$

де x – вектор стану;

u – вектор вхідних впливів;

y – вектор вихідних сигналів;

A, B, C, D – матриці: системи, входу, виходу і обходу, відповідно.

Розмірність матриць показана на рисунку 6.11 (*n* – кількість змінних стану, *m* – число вхідних сигналів, *r* – число вихідних сигналів).

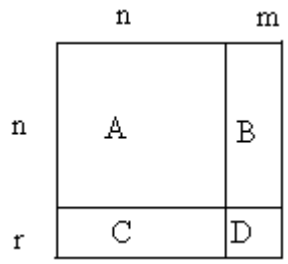


Рисунок 6.11 – Розмірність матриць блоку *State-Space*

Параметри:

– *A* –матриця системи;

– *B* – матриця входу;

– *C* – матриця виходу;

– *D* – матриця обходу;

– *Initial condition* – вектор початкових умов;

– *Absolute tolerance* – абсолютна похибка.

На рисунку 6.11 показаний приклад моделювання динамічного об'єкта за допомогою блоку *State-Space*. Матриці блоку мають такі значення:

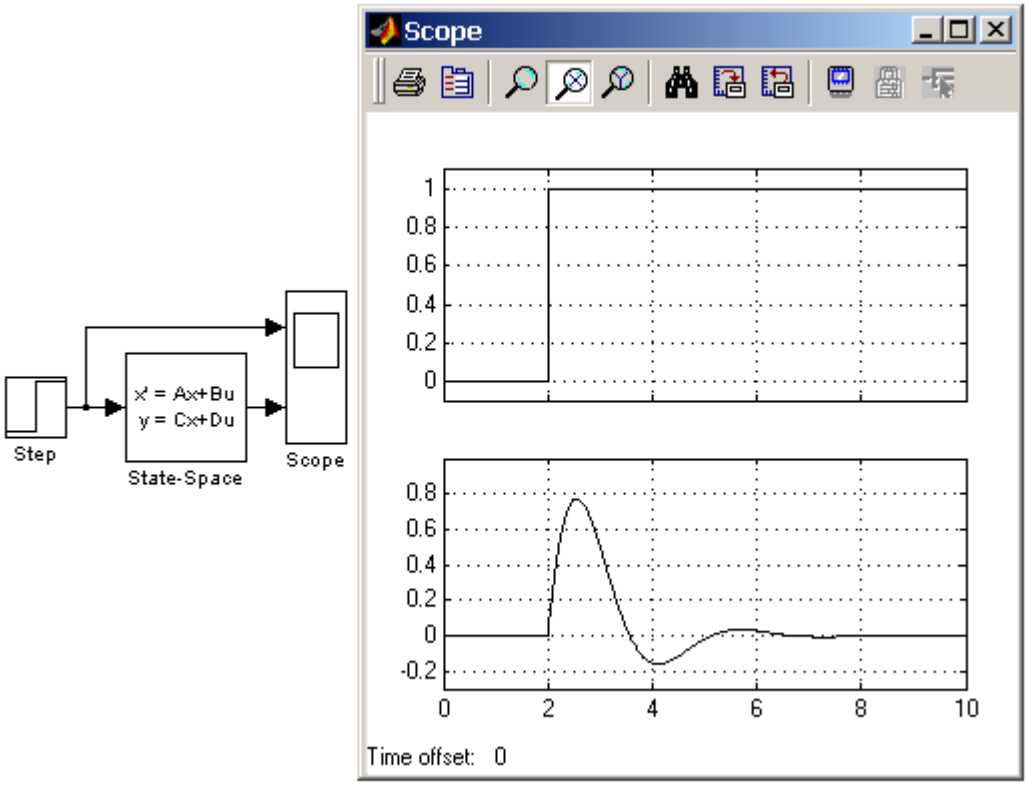


Рисунок 6.12 – Приклад використання блоку *State-Space*

ПРАКТИЧНЕ ЗАНЯТТЯ №7

Дослідження основних команд *MATLAB* для керування *Simulink* моделлю

Мета роботи: дослідити основні команди *MATLAB* для керування *Simulink* моделлю.

При розробці графічного інтерфейсу користувача, створенні S-функцій і т.п. задач що вимагають управління конфігурацією, параметрами і роботою *Simulink*-моделі допускається використовувати спеціальні команди (функції) мови *MATLAB*. За допомогою таких команд можна відкривати і закривати модель, запускати модель на розрахунок, додавати і прибирати блоки, змінювати параметри блоків і здійснювати інші операції з моделлю.

7.1 *add_block*

Призначення: Додавання нового блоку в модель

Синтаксис:

add_block ('src', 'dest')

Команда додає блок, повний шлях якого заданий параметром '*src*', в модель відповідно до шляхом призначення '*dest*'.

add_block ('src', 'dest', 'parameter1', value1, ...)

Команда додає блок, повний шлях якого заданий параметром '*src*', в модель відповідно до шляхом призначення '*dest*' і встановлює значення параметрів блоку.

Приклад 1:

Команда *add_block* ('*built-in*/*Gain*', '*EX_add_block* / *Gain*') додає в модель *EX_add_block.mdl* підсилювач *Gain* з вбудованої бібліотеки.

Приклад 2:

Команда *add_block* ('*EX_add_block* / *In*1', '*EX_add_block* / *My_Subsystem* / *In*1')

копіює блок вхідного порту *In*1 з моделі *EX_add_block.mdl* в підсистему *My_Subsystem* тієї ж самої моделі.

Приклад 3:

Команда *add_block* ('*built-in*/*Constant*', '*EX_add_block* / *Constant*', '*Value*', '150 ') додає в модель *EX_add_block.mdl* блок *Constant* і встановлює параметр *Value* цього блоку рівним 150.

7.2 *add_line*

Призначення: Додавання нової лінії зв'язку в модель

Синтаксис:

h = add_line ('sys', 'oport', 'iport')

Команда додає нову лінію зв'язку в моделі *sys* від вихідного порту *oport* до вхідного порту *iport*. Параметри oport і iport задають повні шляхи блоків.

h = add_line ('sys', 'oport', 'iport', 'autorouting', 'on')

Команда аналогічна попередньої. Додатковий параметр *autorouting* (автоматичне трасування), значення якого дорівнює *on*, забезпечує створення лінії без перетинання піктограм блоків. За "замовчуванням" значення цього параметра дорівнює *off*.

h = add_line ('sys', points)

Команда додає нову лінію зв'язку в моделі sys відповідно до координатами, заданими матрицею *points*. Початком системи координат вікна моделі вважається лівий верхній кут вікна.

Приклад 1:

Команда *add_line* ('*EX_add_line*', '*Step* / 1', '*Sum* / 2') додає нову лінію зв'язку в моделі *EX_add_line.mdl* між виходом блоку *Step* (блок має один вихід) і другим входом блоку *Sum*.

Приклад 2:

Команда *add_line* ('*EX_add_line*', '*Step*1 / 1', '*Sum* / 2', '*autorouting*', '*on*') додає нову лінію зв'язку в моделі *EX_add_line.mdl* між виходом блоку *Step*1 і другим входом блоку *Sum*, при включеному режимі автотрасування .

Приклад 3:

Команда *add_line* ('*EX_add_line*', [20 55; 40 10; 60 60]) додає нову лінію зв'язку в моделі *EX_add_line.mdl* відповідно до координатами, заданими матрицею [20 55; 40 10; 60 60]).

7.3 *add_param*

Призначення: Додавання нового параметра в модель.

Синтаксис:

Команда *add_param* ('*sys*',, *value*1,, *value*2, ...) додає в модель *sys* нові параметри *parameter*1, *parameter*2 ... і присвоює їм значення *value*1,, *value*2, ... Нові параметри доступні командам *get_param*, *set_param* і нічим не відрізняються від стандартних параметрів *Simulink*-моделі. Імена параметрів не чутливі до регістру символів. Значення параметрів чутливі до регістру символів.

Приклад 1:

Команда *add_param* ('*EX_add_param*', '*data*', '01 *december* 2002 ',' *time* ', '21 .00') додає в модель *EX_add_param.mdl* нові параметри *data* і *time* і присвоює їм значення '01 *december* 2002 "та '21 .00 ', відповідно.

7.4 *bdclose*

Призначення: Команда закриває *Simulink*-модель (всі моделі) без збереження змін.

Синтаксис:

bdclose

Команда закриває активну модель.

bdclose ('sys')

Команда закриває модель *sys*.

bdclose ('all')

Команда закриває всі відкриті моделі.

Приклад:

Команда *bdclose* ('*my_model*') закриває модель *my_model.mdl*.

7.5 *bdroot*

Призначення: Повертає ім'я моделі (підсистеми верхнього рівня).

Синтаксис:

bdroot

Команда повертає ім'я активної моделі.

bdroot ('obj')

Команда повертає ім'я моделі, що містить об'єкт *obj*.

Приклад:

Команда *bdroot* (*gcb*) повертає ім'я моделі, що містить виділений в даний момент блок.

7.6 *close_system*

Призначення: Команда закриває модель з можливістю збереження змін.

Синтаксис:

close_system

Команда закриває модель або підсистему. Якщо модель або підсистема були змінені, то на екран буде виведено вікно з питанням про збереження змін.

close_system ('sys')

Команда закриває модель або підсистему з вказаним ім'ям *sys*.

close_system ('sys', saveflag)

Команда закриває модель або підсистему і, в залежності від параметра *saveflag*, виконує або не виконує збереження змін. У випадку, якщо параметр *saveflag* дорівнює 0, зміни не зберігаються. Якщо ж значення даного параметра дорівнює 1, то внесені в модель або підсистему зміни зберігаються.

close_system ('sys', 'newname')

Команда зберігає модель sys під новим ім'ям newname.

close_system ('blk')

Команда закриває вікно діалогу блоку, повний шлях якого заданий параметром *blk*, або викликає *CloseFcn* функцію, якщо дана функція для блоку визначена.

Приклад 1:

Команда *close_system* ('*my_model*', '*new_model*') зберігає модель *my_model.mdl* під новим ім'ям *new_model.mdl.*

Приклад 2:

Команда *close*_system ('*my_model*', 1) зберігає модель *my_model.mdl* з усіма змінами.

7.7 *delete_block*

Призначення: Видалення блоку з моделі.

Синтаксис:

delete_block ('blk')

Команда видаляє блок у відповідності з параметром blk, що задає повний шлях до блоку.

Приклад 1:

Команда *delete_block* ('*EX_delete_block/My_Subsystem*/*In*1') видаляє вхідний порт *In*1 з підсистеми *My_Subsystem* моделі *EX_delete_block.mdl.*

7.8 *delete_line*

Призначення: Видалення лінії зв'язку

Синтаксис:

delete_line ('sys', 'oport', 'iport')

Команда видаляє лінію зв'язку в моделі *sys* від вихідного порту *oport* до вхідного порту *iport*. Параметри *oport* і *iport* задають повні шляхи блоків.

delete_line ('sys', [x y])

Команда видаляє лінію зв'язку, якій належить точка з координатами [*xy*].

Приклад 1:

Команда *delete_line* ('*EX_delete_line*', '*Step* / 1', '*Sum* / 2') видаляє лінію зв'язку в моделі *EX_delete_line.mdl* між виходом блоку *Step* (блок має один вихід) і другим входом блоку *Sum.*

Приклад 2:

Команда *delete_line* ('*EX_delete_line*', [20 55]) видаляє лінію зв'язку, якій належить точка з координатами [20 55].

7.9 *delete_param*

Призначення: Видалення параметра моделі, доданого командою *add_param*.

Синтаксис:

*delete_param ('sys', 'parameter*1*', 'parameter*2*', ...)*

Команда видаляє з моделі *sys* параметри *parameter*1, *parameter*2 ..., додані раніше командою *add_param*.

Приклад:

Команда *delete_param* ('*EX_delete_param*', '*data*', '*time*') видаляє з моделі *EX_delete_param.mdl* параметри *data* і *time*, додані раніше командою *add_param*.

7.10 *gcb*

Призначення: Отримання шляху поточного блоку.

Синтаксис:

gcb

Команда повертає повний шлях поточного блоку.

gcb ('sys')

Команда повертає повний шлях поточного блоку в моделі *sys*.

Під поточним блоком розуміється виділений у вікні моделі блок, блок який виконується в даний момент часу під управлінням *S*-функції, блок *callback*-функція якого виконується в даний момент часу або маскувати блок для якого виконується функція ініціалізації.

Команду зручно використовувати при отриманні шляху блоку для команд *get_param* і *set_param*.

Приклад:

Команда *get_param* (*gcb*, '*Gain*') для поточного блоку *Gain* повертає значення параметра *Gain*.

7.11 *gcs*

Призначення: Отримання шляху поточної моделі.

Синтаксис і правила використання команди аналогічні команді *gcb.*

7.12 *find_system*

Призначення: Пошук моделей (підсистем), блоків, ліній, портів і текстових описів.

Синтаксис:

find_system (*sys*, '*c*1', *cv*1, '*c*2', *cv*2, ... '*p*1', *v*1, '*p*2', *v*2, ...)

Команда виконує пошук моделей (підсистем), блоків, ліній, портів та їх описів, повний шлях яких заданий параметром *sys*, з використанням обмежень, заданих параметрами *c*1, *c*2 і мають значення параметрів *v*1, *v*2.

Види обмежень наведено в Таблиці 7.1.

Таблиця 7.1 – Види обмежень

обмеження	значення	опис
'SearchDepth'	*scalar*	Встановлює глибину пошуку (0 – тільки для відкритих систем, 1 – для блоків і підсистем верхнього рівня, 2 – для системи верхнього рівня і її дочірніх підсистем, і т.д.) Значення за замовчуванням *all* – всі рівні.
'LookUnderMasks'	*'none'*	Пропуск маскованих блоків.
	{'graphical'}	Пошук всередині маскованих блоків, що не мають вікон діалогу і робочої області маски. Цей параметр використовується "за замовчуванням".
	'functional'	Пошук всередині маскованих блоків, що не мають вікон діалогу.
	'all'	Пошук всередині всіх маскованих блоків.

'FollowLinks'	*'on'| {'off'}*	Якщо параметр має значення '*on*', то відстежуються зв'язку з бібліотечними блоками. Значення за замовчуванням '*off*'.
'FindAll'	*'on'| {'off'}*	Якщо параметр має значення '*on*', то пошук поширюється на лінії, порти і текстові описи в межах поточної моделі. Значення за замовчуванням '*off*'.
'CaseSensitive'	*{'on'}| 'off'*	Пошук з урахуванням регістру символів (при пошуку строкових параметрів). Значення за замовчуванням '*on*'.
'RegExp'	*'on'| {'off'}*	Якщо параметр має значення '*on*', то допускається проводити пошук з використанням шаблонів. Значення за замовчуванням '*off*'.

У таблиці значення використовуються "за замовчуванням" параметрів наведені у фігурних дужках.

Приклад 1:

Команда *find_system* повертає масив комірок містять імена всіх відкритих підсистем і блоків.

Приклад 2:

Команда *find_system* ('*type*', '*block_diagram*') повертає масив комірок містять імена всіх відкритих моделей.

Приклад 3:

Команда *find_system* ('*my_model*', '*SearchDepth*', 2, '*BlockType*', '*Product*') виконує пошук блоків множення *Product* у моделі *my_model.mdl* і в її вкладених підсистемах.

Приклад 4:

Команда *find_system* ('*my_model*', '*BlockType*', '*Constant*', '*Value*', '100 ') виконує пошук блоків *Constant* у яких значення параметра *Value* одно 100.

Для пошуку з використанням шаблонів можна застосовувати спеціальні символи наведені в таблиці Таблиця 7.2

Таблиця 7.2 – Спеціальні символи для пошуку з використанням шаблонів

Символ	Опис
.	Замінює будь-який символ. Наприклад, шаблоном '*a*.' Відповідають виразу '*aa*', '*ab*', '*ac*' і т.п.
*	Замінює будь-яку послідовність символів (включаючи порожню). Наприклад, шаблоном '*a* *' відповідають вираженія '*a* ',' *ab* ',' *abc* 'і т.п. Шаблоном '. *' Відповідає будь-який рядок, в тому числі і порожня.
+	Замінює будь-яку кількість попередніх символів. Наприклад, шаблоном '*ab* +' відповідають виразу '*ab*', '*abb*', '*abbb*' і т.п
^	Відзначає початок послідовності символів. Наприклад, шаблоном '^ *a*' відповідає будь-який рядок починається на символ '*a*'.
$	Зазначає останній символ рядка символів. Наприклад, шаблоном '$ *a*' відповідає будь-який рядок, що закінчується на символ '*a*'.
\	Наказує вважати наступний символ звичайним текстовим символом. Наприклад, шаблон '\ \' відповідає рядку містить символ '\'.
[ ]	Визначає набір символів у вираженні пошуку. Наприклад, шаблон '*f* [*oa*] *r*' відповідає виразами '*for*' і '*far* '. Символ (–) задає діапазон символів. Наприклад, шаблон '[*a*-z*A*-*Z*1-9]' відповідає будь-якому алфавітно-цифрового символу. Символ (^) визначає виключаються символи при пошуку. Наприклад, шаблон '*f* [^ i] *r*' відповідає рядкам '*far*' *and* '*for*', але не відповідає рядку '*fir*'.

\w	Задає пошук рядків, що містять тільки алфавітно-цифрові символи. Наприклад, шаблон '^ \ w' відповідає рядку 'mu', але не відповідає рядку '& mu'.
\d	Задає пошук рядків, що містять тільки цифрові символи. Наприклад, шаблон '\ d +' задає пошук будь-якого цілого числа.
\D	Задає пошук рядків, не містять цифрові символи (аналог шаблону [^ 0-9]).
\s	Задає пропуск у вираженні пошуку (аналог шаблону [\ t \ r \ n \ f]).
\S	Виключає прогалини з виразу пошуку (аналог шаблону [^ \ t \ r \ n \ f]).
\<WORD\>	Задає пошук слова (послідовності символів відокремлених з обох сторін пробілами). Наприклад, шаблоном '\ <to\>' відповідає слово 'to', але не відповідає слово'today '.

Приклад 5:

Команда *find_system* ('*my_model*', '*regexp*', '*on*', '*blocktype*', '*port*') задає пошук вхідних та вихідних портів в моделі *my_model.mdl*.

7.13 *get_param*

Призначення: Отримання значення параметрів моделі або блоку.

Синтаксис:

get_param ('obj', 'parameter')

Команда повертає значення параметра parameter, для об'єкта, повний шлях якого заданий виразом *obj*.

Приклад 1:

Команда *get_param* ('*EX_get_param* / *Constant*', '*Value*') визначає значення параметра *Value* блоку *Constant* моделі *EX_get_param.mdl*.

Приклад 2:

Команда *get_param* ('*EX_get_param* / *Constant*', '*ObjectParameters*') визначає всі атрибути блоку *Constant* моделі *EX_get_param.mdl*.

Приклад 3:

Команда *get_param* ('*EX_get_param* / *Constant*', '*DialogParameters*') визначає параметри задаються у вікні діалогу блоку *Constant* моделі *EX_get_param.mdl*.

Приклад 4:

Команда *get_param* ('*EX_get_param*', '*MaxStep*') визначає значення параметра *MaxStep* (максимальний крок розрахунку) моделі *EX_get_param.mdl*.

7.14 *new_system*

Призначення: Створення нової моделі.

Синтаксис:

new_system ('sys')

Команда створює нову модель *sys*. При цьому вікно моделі не відкривається. Для відкриття вікна слід використовувати команду *open_system ('sys')*.

Приклад:

Команда *new_system* ('*my_model*') створює модель *my_model.mdl*.

7.15 *open_system*

Призначення: Команда відкриває вікно моделі, підсистеми, вікно діалогу блоку.

Синтаксис:

open_system ('sys')

Команда відкриває модель *sys.mdl*.

open_system ('blk')

Команда відкриває вікно діалогу блоку *blk* моделі *sys.mdl*.

open_system ('sys / Subsystem', 'force')

Команда відкриває маскувати підсистему *Subsystem* моделі *sys.mdl*. Команда аналогічна пункту меню *Look Under Mask*.

Приклад 1:

Команда *open_system* ('*my_model*') відкриває модель *my_model.mdl.*

Приклад 2:

Команда *open_system* ('*my_model* / *Constant*') відкриває вікно діалогу блоку *Constant* моделі *my_model.mdl*.

Приклад 3:

Команда *open_system* ('*my_model* / *Subsystem*') відкриває вікно маскированной підсистеми *Subsystem* моделі *my_model.mdl*.

7.16 *replace_block*

Призначення: Команда виконує заміну одного блоку на інший.

Синтаксис:

replace_block ('*sys*', '*blk*1', '*blk*2', '*noprompt*')

Команда замінює всі блоки типу *blk*1 на блоки *blk*2 моделі sys без запиту на підтвердження операції. Якщо *blk*2 не є бібліотечним блоком, то потрібно вказати повний шлях до блоку.

replace_block ('*sys*', '*Parameter*', '*value*', '*blk*', ...)

Команда замінює всі блоки, параметр яких *Parameter* дорівнює *value* на блоки *blk* моделі *sys.*

Приклад 1:

Команда *replace_block* ('*EX_replace_block*', '*Step*', '*Inport*', '*noprompt*') замінює в моделі *EX_replace_block.mdl* блок *Step* на блок *Inport* без запиту на підтвердження операції.

Приклад 2:

Команда *replace_block* ('*EX_replace_block*', '*Value*', '100 ','*Gain*', '*noprompt*') замінює в моделі *EX_replace_block.mdl* блоки, параметр яких дорівнює 100 на блоки *Gain* без запиту на підтвердження операції.

7.17 *save_system*

Призначення: Збереження файлу моделі.

Синтаксис:

save_system

Збереження відкритої моделі під поточним ім'ям.

save_system ('sys')

Збереження моделі sys під поточним ім'ям.

save_system ('sys', 'newname')

Збереження моделі sys під новим ім'ям newname.

Приклад 1:

Команда *save_system* ('*my_model*') зберігає модель у файлі *my_model.mdl.*

Приклад 2:

Команда *save_system* (*'my_model*', *new_model*') зберігає модель у файлі *new_model.mdl*.

7.18 *set_param*

Призначення: Установка параметрів моделі або блоку.

Синтаксис:

set_param ('*obj*', '*parameter*1', *value*1, '*parameter*2', *value*2, ...)

Команда виконує присвоювання нових значень *value*1, *value*2 ... параметрам *parameter*1, *parameter*2 ... моделі (блоку) *obj*. Імена параметрів не чутливі до регістру символів. Значення параметрів чутливі до регістру символів.

Приклад 1:

Команда *set_param* ('*EX_set_param*', '*Solver*', '*ode*15*s*', '*StopTime*', '100 ') встановлює метод рішення (параметр *Solver*) *ode*15*s* і час закінчення розрахунку (параметр *StopTime*) 100 для моделі *EX_set_param.mdl.*

Приклад 2:

Команда *set_param* ('*EX_set_param / Step*', '*After*', '1 .5 ') встановлює параметр *Final Value* блоку *Step* моделі *EX_set_param.mdl* рівним 1.5.

Приклад 3:

Команда *set_param ('EX_set_param / Transfer Fcn', 'Numerator', '[5 7 9]', 'Denominator', '[2 3 0]')* встановлює параметри блоку *Transfer Fcn*, таким чином, щоб отримати передавальну функцію такого вигляду:

[Викачати приклад]

Команда може використовуватися для зміни параметрів моделі або блоку в процесі розрахунку. Однак не всі параметри блоків можуть бути змінені в цьому випадку. Наприклад, не можна змінити в процесі розрахунку розмірності вхідних і вихідних портів підсистеми або блоку. Параметри блоків бібліотеки *Power System Blockset* також не можна змінювати в процесі розрахунку. Слід мати на увазі ще й те, що іноді назва параметра, дане у вікні діалогу, відрізняється від фактичного назви параметра (імені змінної, якій присвоюється значення параметра). Так, наприклад, для блоку *Step*, фактичне ім'я параметра *Initial Value є Before*, а фактичне ім'я параметра *Final Value* є *After*. Для з'ясування фактичних імен параметрів можна відкрити файл моделі в будь-якому текстовому редакторі і переглянути секцію, в якій описаний даний блок. Нижче наведено приклад текстового опису блоку *Step* у файлі моделі:

```
Block                                                    {
BlockType                                                Step
Name                                                     "Step"
Position                [125,           75,          155,          105]
Time                                                     "0.1"
Before                                                   "10"
After                                                    "20"
SampleTime                                               "0"
VectorParams1D                                           on
} .
```

З наведеного фрагмента добре видно, які фактичні імена мають параметри даного блоку.

7.19 *simulink*

Призначення: Команда відкриває вікно бібліотеки блоків *simulink*.

синтаксис:

simulink

ПРАКТИЧНЕ ЗАНЯТТЯ №8

Дослідження графічного інтерфейсу відладчика *Simulink* моделей

Мета роботи: дослідити графічний інтерфейс та відладити типові *Simulink* моделі

Відладчик *Simulink* є інструментом для пошуку і діагностування помилок у моделях *Simulink*. Він дає можливість точно визначити проблеми, виконуючи моделювання поступово з відображенням значень вхідних та вихідних сигналів кожного з цікавлять блоків моделі. *Simulink*-відладчик має і графічний, і інтерфейс користувача командного рядка. Графічний інтерфейс дозволяє найбільш зручно використовувати основні можливості відладчика. Інтерфейс командного рядка дає спосіб звертатися до всіх можливостей відладчика. Користувач, як правило, працює з графічним інтерфейсом відладчика і звертається до інтерфейсу командного рядка по мірі необхідності.

Запуск графічного інтерфейсу відладчика можливий одним із двох способів:

1. За допомогою команди меню *Tools / Debugger* вікна *Simulink* моделі.

2. За допомогою кнопки панелі інструментів вікна *Simulink* моделі.

Після запуску відладчика на екрані з'явиться його вікно (див. рисунок 8.1).

Вікно відладчика містить наступні елементи:

– Панель інструментів;

– Список контрольних точок *Break / Display* points;

– Панель завдання точок переривання по умові Break *on conditions*;

– Головне вікно відладчика.

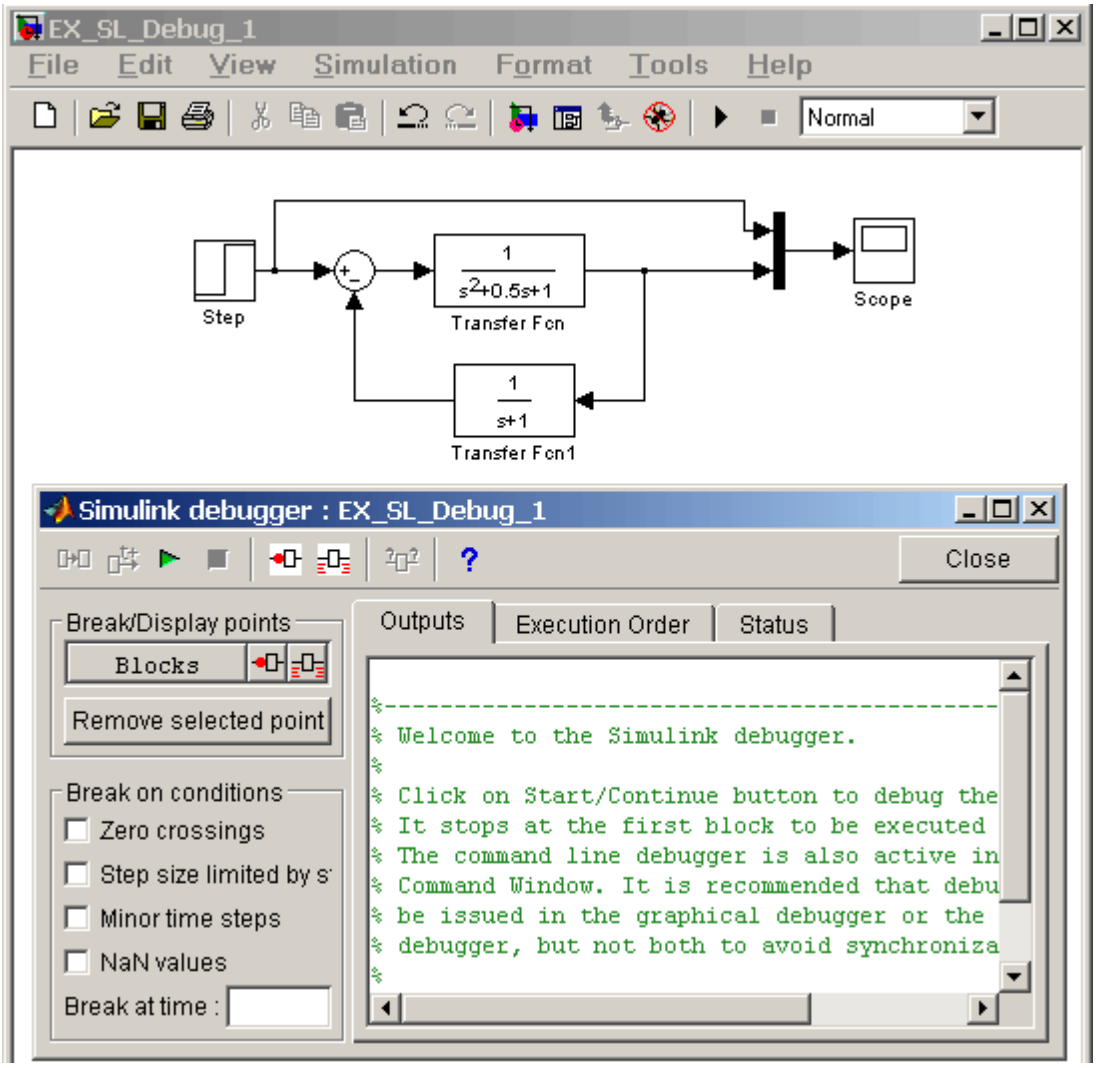


Рисунок 8.1 – Вікно відладчика *Simulink* моделі

8.1 Панель інструментів

Загальний вигляд панелі інструментів показаний на рисунку 8.2.

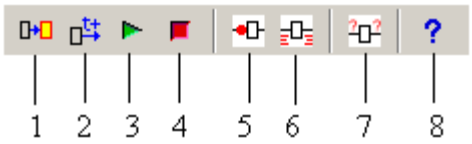


Рисунок 8.2 – Панель інструментів відладчика

Ця панель має наступні кнопки:

1. *Next Block* – Перехід до наступного блоку. За допомогою даної кнопки здійснюється режим відладки із зупинкою моделювання після кожного виконаного блоку.

2. *Next Time Step* – Перехід до наступного тимчасовому кроці. За допомогою даної кнопки виконується режим відладки із зупинкою моделювання після кожного виконаного тимчасового кроку.

3. *Start / Continue* – Початок / Продовження налагодження. Натискання даної кнопки після запуску відладчика приводить до початку процесу моделювання і зупинці його перед першим виконуваним блоком. Дана кнопка служить також для продовження процесу налагодження при встановлених точках переривання (зупинки). Якщо точки переривання встановлені в моделі, то натискання даної кнопки дозволяє продовжити моделювання і, потім, зупинити його в заданій точці переривання. Повторне натискання кнопки відновлює процес моделювання і викликає зупинку на наступній точці переривання. Якщо на поточному часовому кроці всі точки переривання пройдені, то відбувається перехід на наступний часовий крок і зупинка в першій точці переривання на новому часовому кроці.

4. *Stop* – Зупинка налагодження.

5. *Break before selected block* – Установка точки переривання перед виділеним блоком. Точка переривання не може бути встановлена для віртуального блоку, тобто такого блоку, чия функція є виключно графічної. На рисунку 8.1 таким блоком є мультиплексор *Mux*. Список Невіртуальна блоків можна подивитися на вкладці *Execution Order* вікна відладчика або вивести його командою slist при роботі з відладчиком в режимі командного рядка. При спробі встановити контрольну точку для віртуального блоку відладчик виведе на екран попереджувальне повідомлення.

Для установки точки переривання досить виділити блок у вікні моделі і, потім, натиснути цю кнопку. Назва блоку негайно з'явиться у вікні *Break / Display points* відладчика, як це показано на рисунку 8.3.

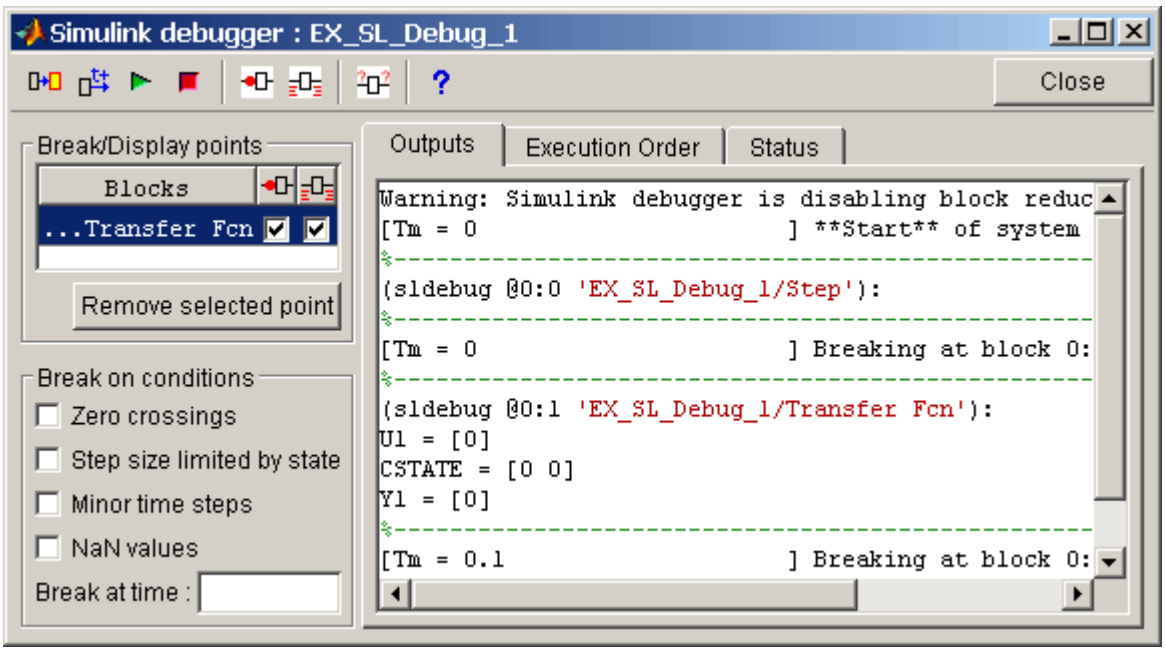


Рисунок 8.3 – Вікно відладчика з встановленою точкою переривання для блоку *Transfer Fcn*

6. *Display I / O of selected block with executed* – Відображати значення вхідних і вихідних сигналів. Дана кнопка дозволяє встановити режим перегляду вхідних і вихідних сигналів якогось Невіртуальна блоку. Режим може бути встановлений як для блоку, для якого встановлена точка переривання, так і для блоку для якого контрольна точка не встановлена. Значення вхідних і вихідних сигналів відображаються на вкладці *Outputs* вікна відладчика. Вхідні сигнали відображаються ідентифікаторами *U*1, *U*2, *U*3 і т.д., а вихідні – *Y*1, *Y*2, *Y*3 і т.д. Додатково тут же відображаються значення змінних стану – *CSTATE* (див. рисунок 8.3).

7. *Display current I / O of selected block* – Показати значення вхідних і вихідних сигналів для виділеного блоку. Даною кнопкою зручно користуватися, коли необхідно переглянути значення сигналів-якого блоку в момент зупинки. Наприклад, якщо для моделі на рисунку 8.1 встановлена точка переривання перед виконанням блоку *Transfer Fcn* і зупинка моделювання в цій точці відбулася, то переглянути значення сигналів суматора можна виділивши цей блок у вікні моделі і натиснувши кнопку відладчика.

8. *Help* – Виклик довідки по відладчику.

8.2 Список контрольних точок *Break / Display points*

Вікно списку контрольних точок (рисунок 8.4) містить список блоків, для яких встановлені контрольні точки (графа *Blocks*), а також властивості цих точок вказані за допомогою прапорців.

Користувач знімаючи або встановлюючи прапорці може змінювати властивості контрольної точки, а саме: ставити / забирати точку переривання на вході блоку (графа) або вмикати / вимикати режим відображення значень сигналів блоку (графа).

Видалити контрольну точку можна виділивши її в списку, і натиснувши кнопку *Remove selected point* (прибрати виділену точку).

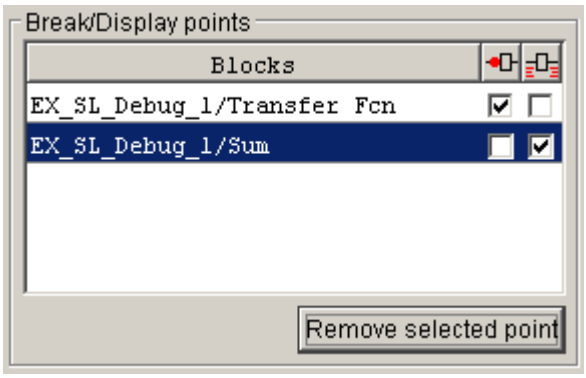


Рисунок 8.4 – Вікно списку контрольних точок *Break/Display points*

8.3 Панель завдання точок переривання по умові *Break on conditions*

Панель (рисунок 8.5) містить список умов, при настанні яких розрахунок повинен бути зупинений.

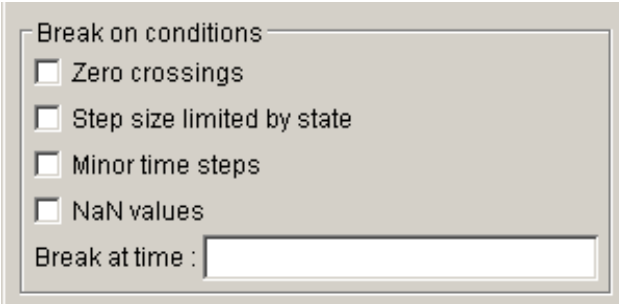


Рисунок 8.5 – Список умов переривання розрахунків

Список містить:

1. *Zero crossings* – Перехід сигналу через нульовий рівень при його стрибкоподібному зміну. На рисунку 8.6 показаний приклад моделі, в якій має місце така ситуація. На рисунку добре видно, що в момент часу $t = 5$ с відбувається стрибкоподібне зміна сигналу з перетином нульового рівня.

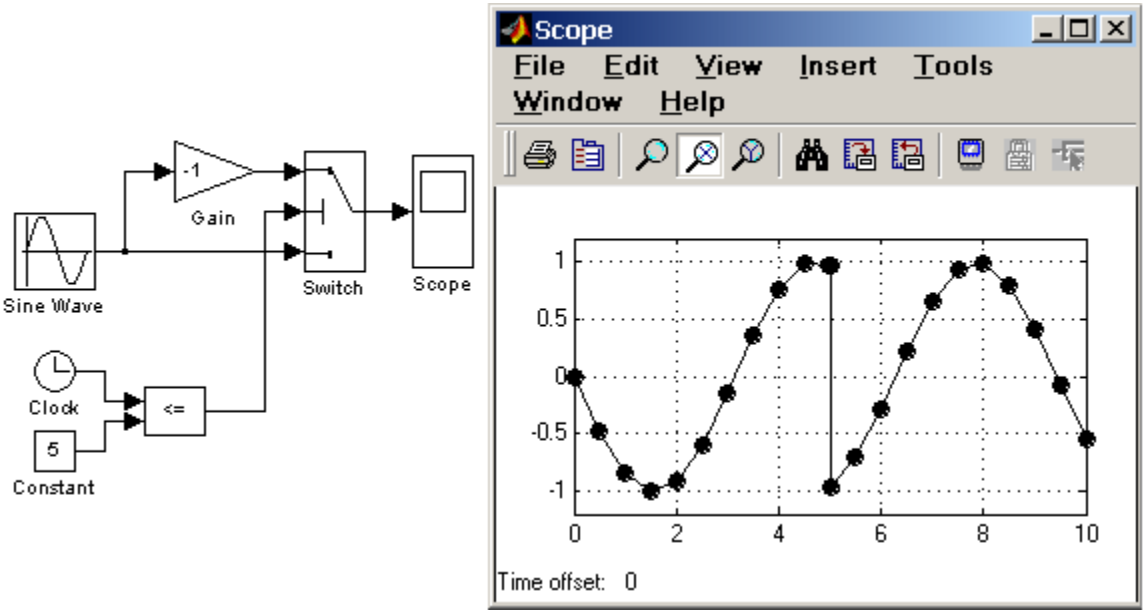


Рисунок 8.6 – Приклад моделі зі зміною полярності сигналу

2. *Step size limited by state* – Стан обмежує крок розрахунку. Опція змушує відладчик зупиняти моделювання, коли модель використовує вирішувач із змінним кроком і вирішувач стикається зі станом вимагає обмеження розміру кроку розрахунку. Ця опція корисна при налагодженні моделей, що вимагають, як здається, надмірно багато розрахункових кроків.

3. *Minor time steps* – режим відладки з використанням внутрішніх (малих) кроків. При виконанні розрахунків *Simulink* може зменшувати заданий крок розрахунку для досягнення потрібної точності. Для того, щоб побачити і ці малі (внутрішні) кроки необхідно встановити опцію *Minor time steps*.

4. *NaN values* – Не числове значення. Розрахунок буде перерваний, коли обчислена значення нескінченно або лежить поза діапазону значень, які можуть бути представлені комп'ютером, який виконує моделювання (занадто малі або занадто великі значення). Ця опція корисна для точного визначення обчислювальних похибок в моделі *Simulink*.

5. *Break at time* – Зупинка в заданий момент часу. Параметр дозволяє задати час до якого модель розраховується у звичайному режимі. Після досягнення заданого часу розрахунок буде зупинений. Далі розрахунок

необхідно відновити з використанням можливостей відладчика. Даний режим зручний, якщо помилка виникає не на початку інтервалу моделювання, а в якій-небудь більш пізній момент. У цьому випадку можна встановити час зупинки безпосередньо перед появою помилки, а потім продовжити розрахунок в покроковому режимі.

8.4 Головне вікно відладчика

Головне вікно містить три вкладки:

1. *Outputs* – Відображення результатів роботи в режимі налагодження. На даній вкладці (рисунок 8.7) відображається поточний модельне час *Tm* (або *Ti* для внутрішніх кроків), індекс контрольованого блоку у вигляді *@ s*: *b*, де *s* – номер моделі (підсистеми), *b* – номер блоку, а також ім'я блоку із зазначенням повного шляху до блоку.

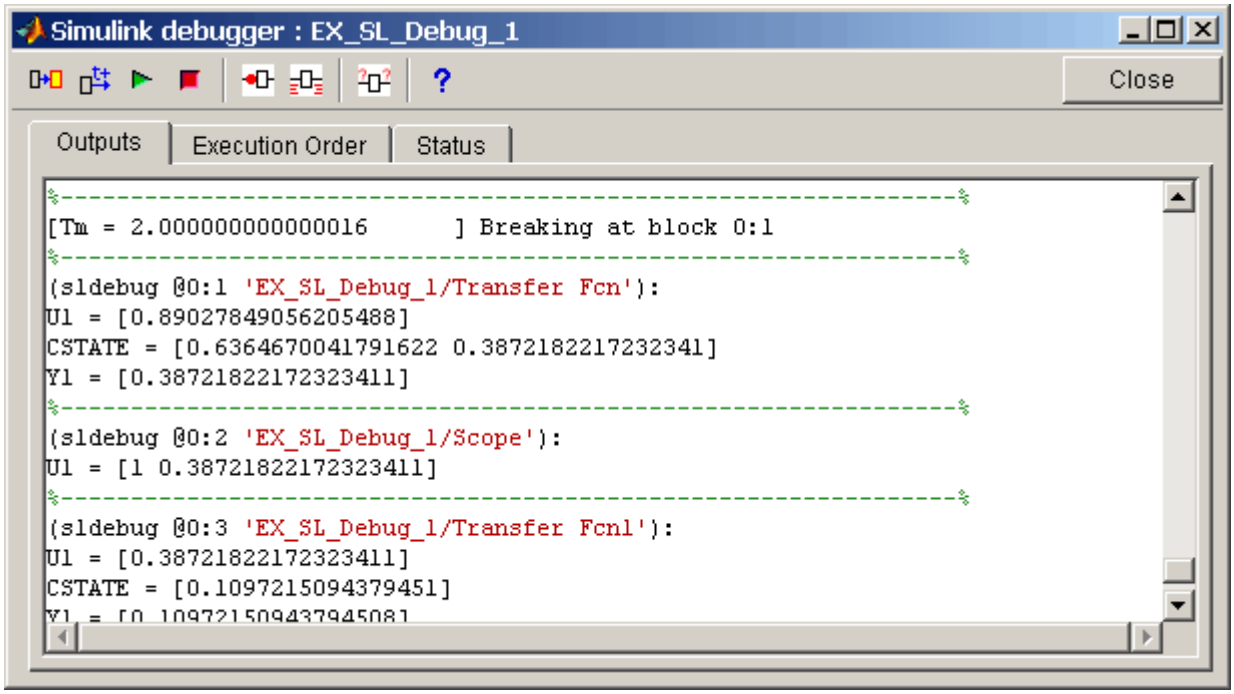


Рисунок 8.7 – Вкладка *Outputs* головного вікна відладчика

На цій же вкладці виводяться значення вхідних (*U*) і вихідних (*Y*) сигналів блоку, якщо відповідна опція встановлена.

2. *Execution Order* – Порядок виконання. На вкладці відображається список Невіртуальна блоків в порядку їх виконання. Блоки, розташовані на початку списку виконуються раніше, ніж блоки, розташовані у кінці списку. На рисунку 8.8 показаний приклад даної вкладки для моделі, зображеної на рисунку 8.1.

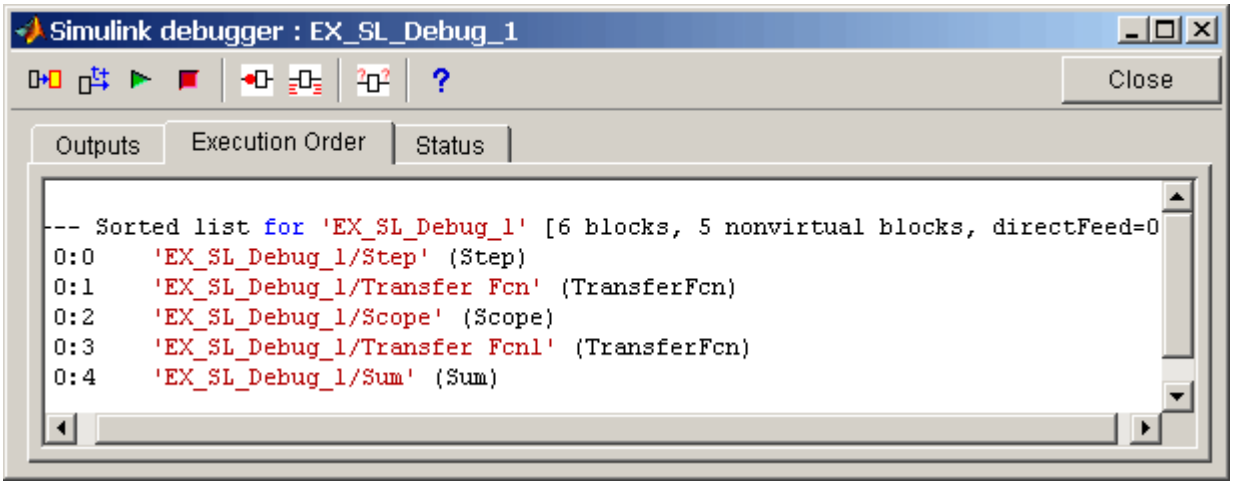


Рисунок 8.8 – Вкладка *Execution Order* головного вікна відладчика

3. *Status* – Статус відладчика. На даній вкладці (рисунок 8.9) відображається інформація про налаштування та поточний стан відладчика: значення поточного часового кроку, кількість точок переривання, інформація про установку точок переривання по умові і т.п.

Таким чином, запустивши відладчик в графічному режимі, користувач може провести покрокову (по блокам або за тимчасовими крокам) налагодження моделі, встановивши при необхідності потрібні контрольні точки.

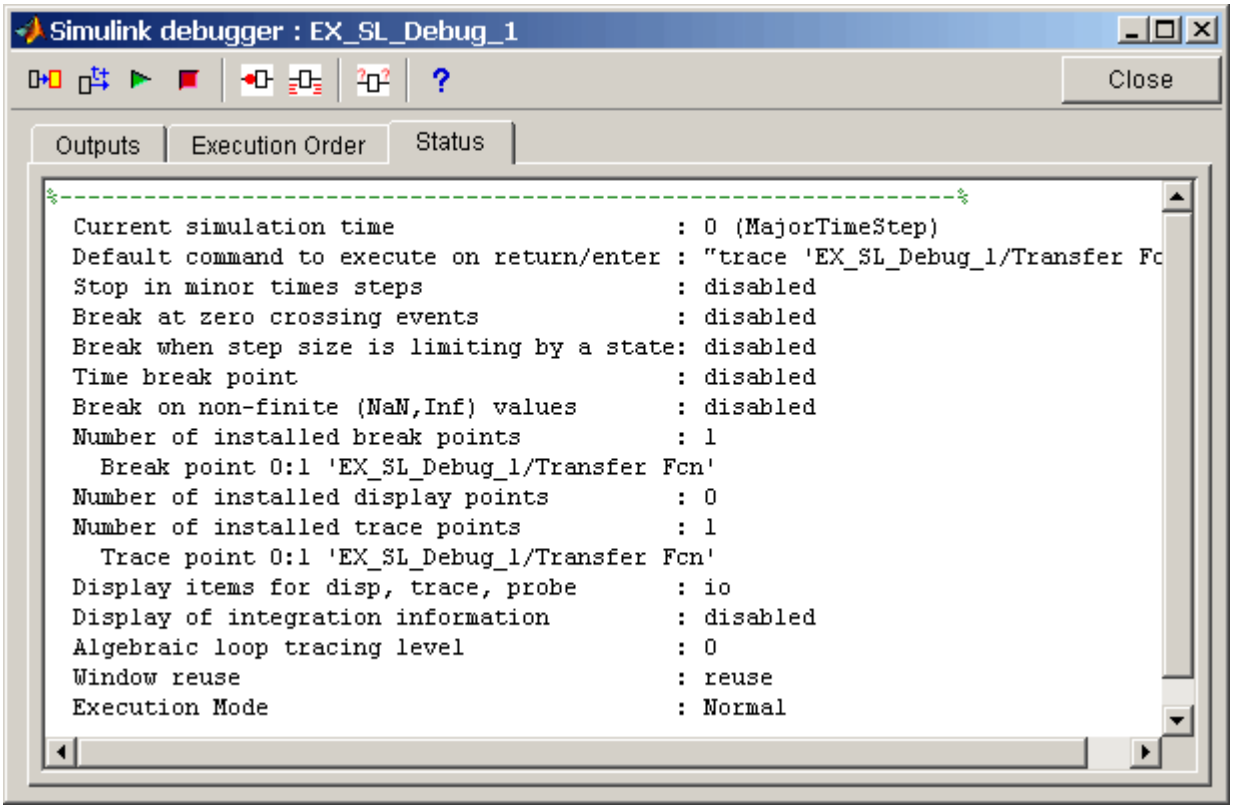


Рисунок 8.9 – Вкладка *Status* головного вікна відладчика

8.5 Інтерфейс командного рядка відладчика Simulink моделей

Інтерфейс командного рядка дає користувачеві доступ до всіх можливостей відладчика.

Запуск відладчика в режимі командного рядка можливий за допомогою команди, яка задається в робочому вікні *MATLAB*:

sldebug ('*My_model*'), де *My_model* – ім'я налагоджують моделі.

Для роботи з відладчиком потрібно вводити команди в головному вікні *MATLAB*. Список команд наведено в Таблиці 8.1.

Таблиця 8.1 – Список команд

Команда	Коротка форма	Повтор	Призначення
step	*s*	так	Перехід до наступного блоку
next	*n*	так	Перехід до наступного тимчасовому кроці

disp [s:b | gcb]	*d*	так	Показ вхідних і вихідних сигналів блоку при зупинці
undisp <s:b | gcb>	*und*	так	Видалення блоку зі списку відображуваних
trace <s:b | gcb>	*tr*	так	Показ вхідних і вихідних сигналів блоку під час виконання
untrace <s:b | gcb>	*unt*	так	Видалення блоку зі списку трасування
probe [s:b | gcb]	*p*	ні	Показ вхідних і вихідних сигналів зазначеного блоку
break <s:b | gcb>	*b*	ні	Вставка точки зупинки при вході в блок
bafter <s:b | gcb>	*ba*	ні	Вставка точки зупинки при виході з блоку
bshow s:b	*bs*	ні	Показ зазначеного за допомогою індексу блоку
clear <s:b | gcb>	*cl*	ні	Видалення точки останову
zcbreak	*zcb*	ні	Переривання при виявленні стрибкоподібного переходу сигналом нульового рівня (непередбачені перетин нуля)
zclist	*zcl*	ні	Список блоків дають непередбачені перетин нуля
xbreak	*x*	ні	Переривання при змінному кроці розрахунку в стані вимагає обмеження кроку розрахунку
tbreak [t]	*tb*	ні	Установка / видалення зупинки в

			зазначений момент часу
nanbreak	*na*	ні	Установка / видалення зупинки при виявленні не числового (*NaN, Inf*) значення
continue	*c*	так	Продовження моделювання
run	*r*	ні	Закінчення режиму налагодження та продовження розрахунку в звичайному режимі
stop	*sto*	ні	Зупинка моделювання
quit	*q*	ні	Переривання моделювання
status [all]	*stat*	ні	Показ параметрів відладчика
states	*state*	ні	Показ поточних значень змінних стану
systems	*sys*	ні	Відображення списку невіртуальних підсистем
slist	*sli*	ні	Список невіртуальних блоків
minor	*m*	ні	Режим налагодження з використанням внутрішніх (малих) кроків
ishow	*i*	ні	Включення / вимикання режиму показу інформації про інтегруючих блоках
emode	*e*	ні	Виведення інформації про поточний режим моделювання (звичайний або прискорений)
probe level {all} | io		ні	Встановити рівень подробиці показу сигналів блоків (всі або тільки вхідні і вихідні)

atrace level	*at*	ні	Установка рівня відображення інформації при трасуванні алгебраїчних контурів (0–нічого, 4 – усі)
ashow <gcb | s:b>	*as*	ні	Показ алгебраїчного контуру, що містить вказаний блок
ashow s#n	*as*	ні	Показ алгебраїчного контуру з номером *n* в підсистемі (моделі) *s*
ashow clear	*as*	ні	Скасувати показ алгебраїчного контуру

Частина команд наведених у таблиці вимагають вказівки індексу блоку (див. п. 8.4). При використанні таких команд замість імені блоку можна вказувати команду *gcb* (отримати шлях поточного блоку), попередньо виділивши потрібний блок у вікні моделі.

Приклад командного вікна *MATLAB* в процесі налагодження моделі показаний на рисунку 8.10.

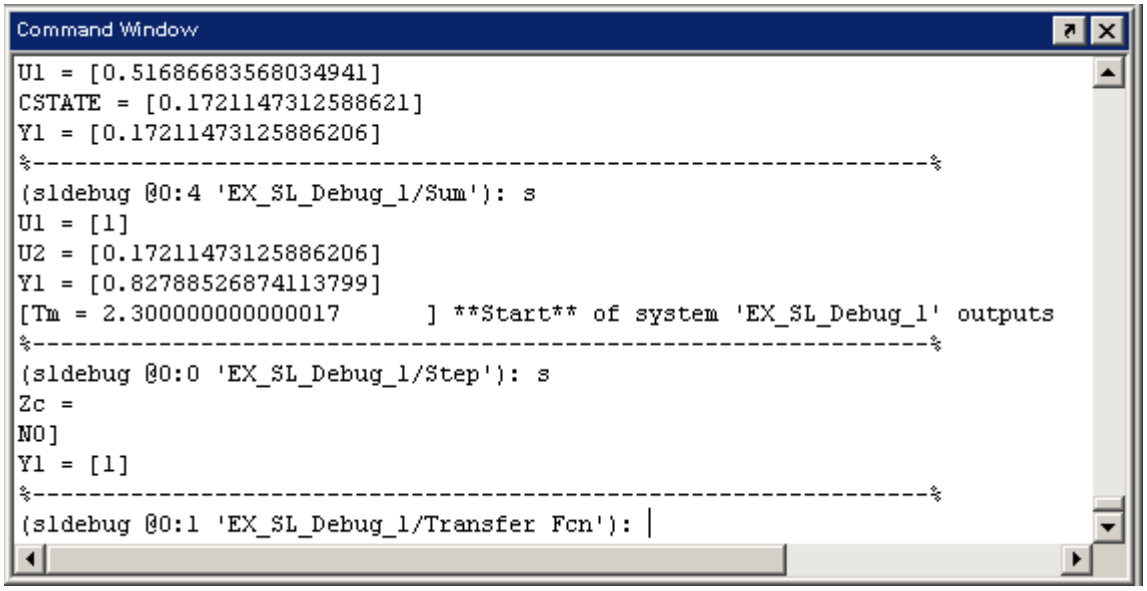


Рисунок 8.10 – Командне вікно *MATLAB* в процесі відладки моделі

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Арсеньєва С.І. Використання програмних засобів MATLAB для розв’язання типових задач аналогової автоматизації: навчальний посібник для здобувачів вищих навчальних закладів / С.І. Арсеньєва – Запоріжжя: НУ «Запорізька політехніка», 2019. – 118 с.
2. Василенко О.В. Комп’ютерне моделювання: Навчальний посібник / О.В.Василенко – Запоріжжя: НУ «Запорізька політехніка», 2020. – 175 с.
3. Васильев В.В. Математическое и компьютерное моделирование процессов и систем в среде Matlab/Simulink. Учебное пособие для здобувачов и аспирантов / В.В.Васильев, Л.А.Симак, А.М. Рибникова. К.: НАН Украины, 2008. – 91 с.
4. Гаєв Є.О., Нестеренко Б.М. Універсальний математичний пакет MATLAB і типові задачіобчислювальної математики. Навчальний посібник. – К.: НАУ, 2004. – 176 с.
5. Гоблик Н.М. MATLAB в інженерних розрахунках. Комп'ютерний практикум / Н.М. Гоблик, В.В. Гоблик. — Львів : Вид-во НУ "Львівська політехніка", 2010. — 120 с.
6. Жученко А.І. Динамічна оптимізація з використанням MATLAB та SIMULINK. А.І. Жученко, Л.Р. Ладієва, Р.М. Дубік. К.: НТУУ “КПІ”, 2010.–209 с.
7. Забара С.С. Моделювання систем у середовищі MATLAB / С.С. Забара, А.А. Гагарін, І.М. Кузьменко, Ю.Д. Щербашин – К.: Видавництво “Університету “Україна” , 2011. – 136 с.
8. Коржик М.В. Моделювання об’єктів та систем керування засобами MatLab: навч. посіб. для студ. вищ. навч. закл. / М.В. Коржик. − Київ : НТУУ “КПІ”, 2016. − 174 с. : іл.
9. Лазарєв Ю.Ф. Довідник з MATLAB / Електронний навчальний посібник з курсового і дипломного проектування. – К.: НТУУ "КПІ", 2013. – 132 с.

10. Лазарев Ю.Ф. MatLab 5.x. – К.: Видавнича група BHV, 2000. – 384 с.

Навчальне видання

МОДЕЛЮВАННЯ ЗАСОБІВ АВТОМАТИЗАЦІЇ

Методичні вказівки
до виконання практичних робіт

Укладач:

НЕЧИТАЙЛО Юлія Анатоліївна

Формат 60x84/16. Гарнітура Times New Roman
Папір для цифрового друку. Друк ризографічний.
Ум. друк. арк._.
Наклад ___пр.
Державний біотехнологічний університет
61002, м. Харків, вул. Алчевських, 44