

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ  
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»  
Теплоенергетичний факультет

**ПРОГРАМУВАННЯ В АВТОМАТИЗОВАНИХ СИСТЕМАХ  
УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ**

**МЕТОДИЧНІ ВКАЗІВКИ**

**до виконання лабораторної роботи  
«Імітаційне моделювання системи керування»**

*для студентів напрямку підготовки 6.050202 «Автоматизація та комп'ютерно-інтегровані технології»*

*програм професійного спрямування  
6.05020201 «Автоматизоване управління технологічними процесами»,  
6.05020202 «Комп'ютерно-інтегровані технологічні процеси та виробництва»*

*Рекомендовано Вченою радою теплоенергетичного факультету*

Київ  
НТУУ «КПІ»  
2016

**Програмування в автоматизованих системах управління технологічними процесами: Методичні вказівки до виконання лабораторної роботи «Імітаційне моделювання системи керування»** студентів НТУУ «КПІ» напряму підготовки 6.050202 «Автоматизація та комп'ютерно-інтегровані технології» програм професійного спрямування 6.05020201 «Автоматизоване управління технологічними процесами», 6.05020202 «Комп'ютерно-інтегровані технологічні процеси і виробництва» / Укл.: О.В. Степанець, С.Г. Батюк — К.: КПІ, 2016. — 28 с.

*Гриф надано Вченою радою теплоенергетичного  
факультету  
(протокол № 10 від 30 травня 2016 р.)*

Електронне навчальне видання

**ПРОГРАМУВАННЯ В АВТОМАТИЗОВАНИХ СИСТЕМАХ  
УПРАВЛІННЯ ТЕХНОЛОГІЧНИМИ ПРОЦЕСАМИ**

**МЕТОДИЧНІ ВКАЗІВКИ**  
до виконання лабораторної роботи  
«Імітаційне моделювання системи керування»

*для студентів напряму підготовки 6.050202 «Автоматизація та комп'ютерно-інтегровані технології»*

*програм професійного спрямування  
6.05020201 «Автоматизоване управління технологічними процесами»,  
6.05020202 «Комп'ютерно-інтегровані технологічні процеси та виробництва»*

Укладачі: Степанець Олександр Васильович, к.т.н., доцент  
Батюк Сергій Георгійович, к.т.н., доцент

Відповідальний редактор: Ковриго Ю.М., професор, к.т.н., зав.каф.

Рецензент: Гагарін О.О., доц., к.т.н., доцент каф. АПЕПС

*За редакцією укладачів*

## ЗМІСТ

Вступ.....	4
1 Теоретичні відомості.....	5
1.1 Загальні дані .....	5
1.2 Загальний порядок розробки .....	7
1.3 Розробка програми контролера .....	8
1.4 Налаштування ОРС-сервера .....	15
1.5 Розробка проекту ОРС-клієнта.....	17
1.6 Робота системи імітаційного моделювання.....	23
2 Порядок виконання роботи .....	25
3 Правила виконання роботи та вимоги до оформлення протоколу .....	26
4 Контрольні запитання .....	27
5 Перелік використаної та рекомендованої літератури.....	28

## ВСТУП

Важливим етапом створення системи автоматичного керування є експериментальна перевірка якості роботи її алгоритмів та підсистем. Через значну вартість натурних досліджень значне поширення набув підхід імітаційного моделювання, коли компоненти системи представляються їх математичними моделями, а попередні досліди виконують в режимі імітації динаміки системи керування цими моделями.

**Мета роботи:** ознайомитись з OPC-технологією зв'язку програмно-апаратних компонентів АСУТП; отримати навички роботи з OPC-сервером, налаштування інформаційного обміну між компонентами; отримати базові знання з моделювання системи керування технологічним процесом з використанням промислово-наукових програмних засобів.

# 1 ТЕОРЕТИЧНІ ВІДОМОСТІ

## 1.1 Загальні дані

Технологія зв'язування та впровадження об'єктів для систем промислової автоматизації OPC (OLE for Process Control. OLE - Object Linking and Embedding — зв'язування і вбудовування об'єктів) призначена для забезпечення універсального механізму обміну даними між датчиками, виконавчими механізмами, контролерами, пристроями зв'язку з об'єктом і системами представлення технологічної інформації, оперативного диспетчерського управління, а також системами управління базами даних. Розроблена міжнародною організацією OPC Foundation як промисловий стандарт і опублікована у вигляді специфікацій.

Основний стандарт - OPC DA (Data Access) — описує інтерфейс обміну даними з пристроями в реальному часі.

OPC-сервер – це програма, яка отримує дані у внутрішньому форматі пристрою або системи і перетворює ці дані в формат OPC. OPC-сервер є джерелом даних для OPC-клієнтів. За своєю суттю OPC-сервер - це якийсь універсальний драйвер фізичного обладнання, що забезпечує взаємодію з будь-яким OPC-клієнтом.

OPC-клієнт - програма, яка приймає від OPC-серверів дані у форматі OPC. Прикладом таких програм є системи диспетчеризації SCADA.

Будь-який OPC-клієнт може обмінюватися даними з будь-яким OPC-сервером незалежно від специфіки пристроїв, з якими взаємодіє конкретний OPC-сервер. Таким чином, OPC-технологія забезпечує незалежність споживачів від наявності або відсутності драйверів або протоколів, що дозволяє вибирати обладнання та програмне забезпечення, що найбільш повно відповідає реальним потребам.

OPC заснована на моделі розподілених компонентних об'єктів Microsoft DCOM і встановлює вимоги до класів об'єктів доступу до даних і їх спеціалізованим (custom) інтерфейсам для використання розробниками клієнтських і серверних додатків.

Структура взаємодії між апаратними засобами, додатками-клієнтами та серверами OPC різних виробників показана на рис. 1.1.

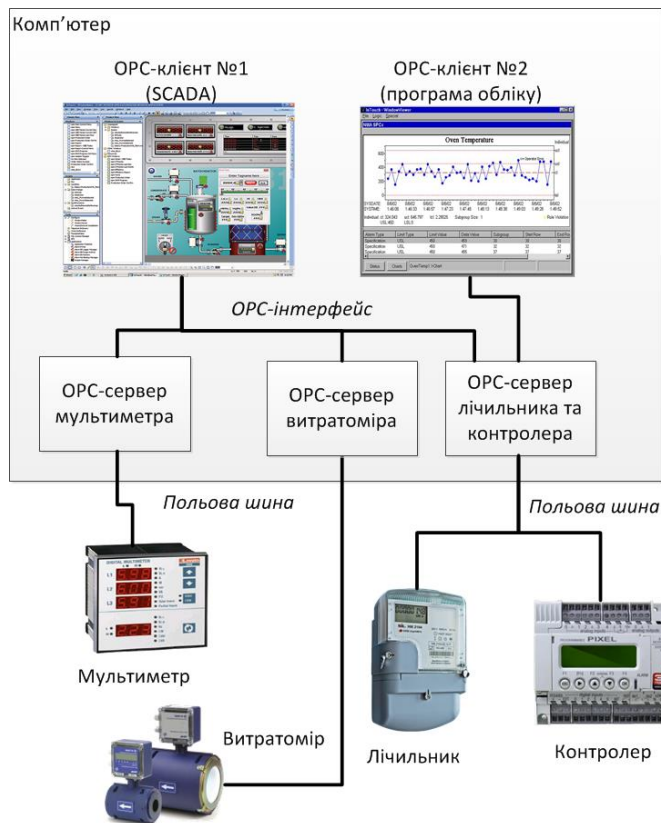


Рис. 1.1. Структура взаємодії OPC-серверів та клієнтів

Спираючись на об'єктну технологію COM / DCOM, стандарт OPC фіксує певну модель взаємодії між клієнтом і сервером.

Базовим поняттям цієї моделі є елемент даних (Item). Кожен елемент даних має значення, час останнього поновлення (timestamp) і ознаку якості, що визначає ступінь достовірності значення. Значення може бути практично будь-якого скалярного типу - булеве, ціле, плаваюче з точкою і т.п. або рядком (так званий OLE VARIANT). Час представляється з 100-наносекундною точністю (FILETIME Win32 API). Реальна точність вимірювання часу зазвичай буває гіршою і, в загальному випадку, залежить від реалізації сервера і апаратури. Якість - це код, що містить у собі грубу оцінку - UNCERTAIN, GOOD та BAD (не визначено, гарна і погана), а на випадок поганої - ще й розшифровку, наприклад QUAL\_SENSOR\_FAILURE - несправність датчика.

Наступним по ієрархії є поняття групи елементів (OPC Group). Група створюється OPC-сервером на вимогу клієнта, який потім може додавати в групу елементи (Item). Для групи клієнтом задається частота оновлення даних, і всі дані в групі сервер намагається оновлювати і передавати клієнту із заданою частотою. Клієнт може створити для себе на сервері кілька груп, що відрізняються необхідною частотою оновлення. Для кожного клієнта завжди створюється своя група (крім так званих публічних груп), навіть якщо склад елементів і частота оновлення збігаються. Від'єднання клієнта призводить до знищення групи.

Елементи в групі — це свого роду клієнтські посилання на якісь реальні змінні (теги), що знаходяться на сервері або на фізичному пристрої. Поняття тега специфікацією OPC не визначається, але мається на увазі неявно. Елементи в групу клієнт додає по імені, і ці імена є іменами відповідних тегів. Клієнт може або знати потрібні імена заздалегідь, або запросити список імен тегів у сервера. Для запити імен тегів служить інтерфейс `IOPCBrowseServerAddressSpace`, за допомогою якого сервер описує клієнтові своє "простір імен", організований в загальному випадку ієрархічно. Приклад повного імені тега: `Пристрій1.Модуль5.АналоговийВхід3`. При додаванні елемента в групу клієнт завжди вказує це повне ім'я. При цьому групи, створювані клієнтом, не зобов'язані збігатися (і, як правило, не збігаються) з підрозділами простору імен сервера, елементи в групу додаються в «різної». Єдине, що їх об'єднує — це загальна частота оновлення і синхронність відправки клієнтові.

Нарешті, на верхній сходинці ієрархії понять знаходиться сам OPC-сервер. З усіх перерахованих (OPC-група, OPC-елемент) він єдиний є COM-об'єктом, всі інші об'єкти доступні через його інтерфейси, які він надає клієнтові.

Якщо клієнт та сервер знаходяться на одному комп'ютері, то установча програма автоматично проводить реєстрацію сервера (запис відповідної інформації в системні реєстр). Програми-клієнти, як правило, мають відповідний користувальницький інтерфейс, що дозволяє вибрати зі списку зареєстрованих серверів потрібний (при цьому, якщо сервер не активний, він автоматично запуститься), переглянути його адресний простір і підписатися на необхідні теги. Таким чином, для забезпечення з'єднання на одній машині яких-небудь спеціальних налаштувань (крім установки сервера та клієнтського програмного забезпечення, наприклад SCADA-системи) проводити не потрібно.

Для роботи в мережі (клієнт і сервер на різних комп'ютерах) необхідна присутність в мережі хоча б однієї станції із встановленою Windows NT (Server або Workstation). Станція з Windows NT використовується в якості сервера авторизації та аутентифікації, при цьому сам OPC-сервер може розташовуватися як на ній, так і на іншій мережевій станції. Перед встановленням з'єднання між додатком-клієнтом і віддаленим сервером слід провести налаштування системних компонентів DCOM.

## 1.2 Загальний порядок розробки

Загальна структура взаємодії компонентів системи (рис. 1.2) включає в себе:

1. об'єкт керування, представлений математичною моделлю в середовищі Matlab;
2. систему керування, яка реалізується на контролері, що працює під управлінням CoDeSys;
3. зв'язок між Matlab та контролером з CoDeSys, реалізований за допомогою OPC-технології.

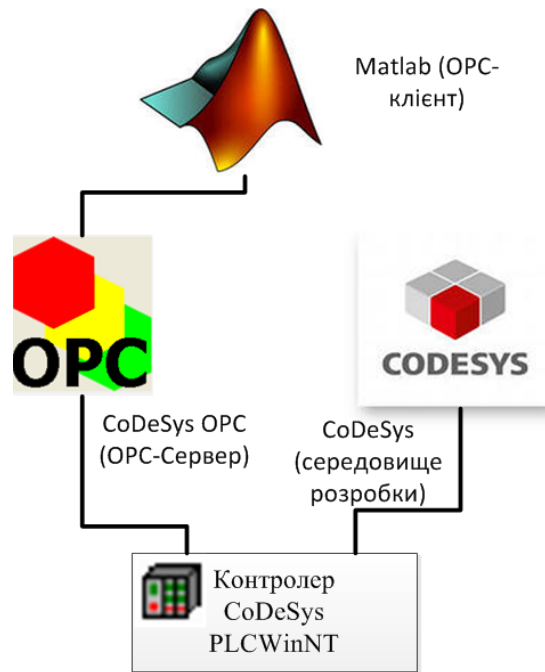


Рис. 1.2. Структура системи імітаційного моделювання

### 1.3 Розробка програми контролера

Для того, щоб побудувати інформаційну систему між рівнями АСУТП, кожен компонент такої системи повинен мати можливості передавати та отримувати необхідні йому для роботи дані. У випадку програмованих логічних контролерів (ПЛК) забезпечення такої можливості лягає на плечі програміста.

На нашому ПЛК буде реалізований ПІД-регулятор, що керуватиме деяким технологічним процесом.

У якості платформи для прикладної програми використаємо CoDeSys V2.3 та віртуальний контролер, що входить до його складу: 3S CoDeSys SP PLCWinNT V2.4.

Для цього створюємо новий проект та обираємо в списку доступних таргетів (конфігурацій контролерів) 3S CoDeSys SP PLCWinNT V2.4 (рис. 1.3):



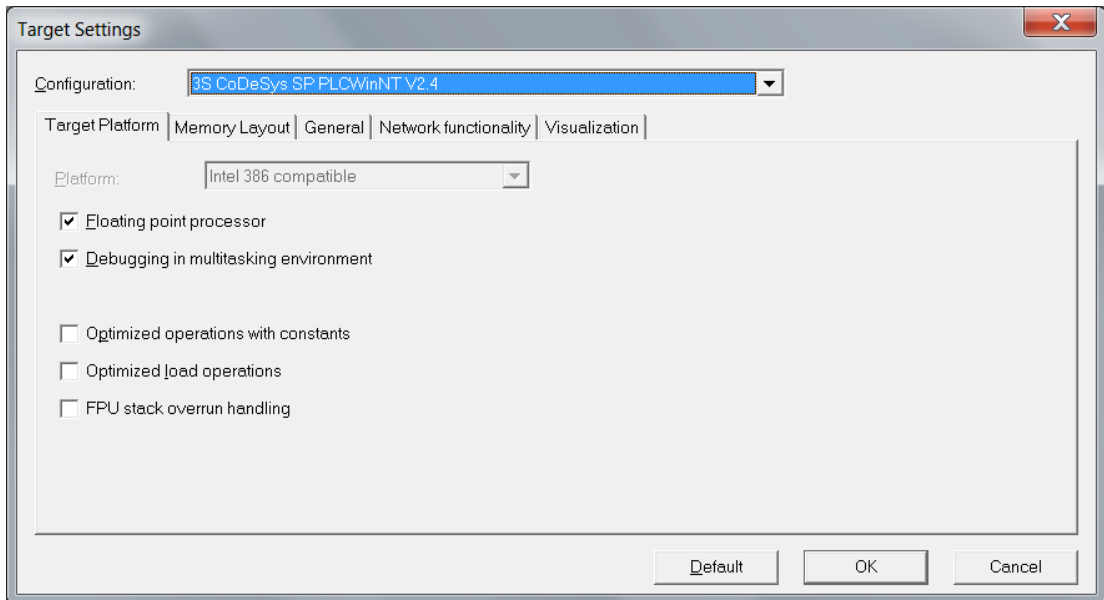


Рис. 1.3. Вікно вибору таргета контролера

Прикладну програму створимо на мові FBD (рис. 2.4):

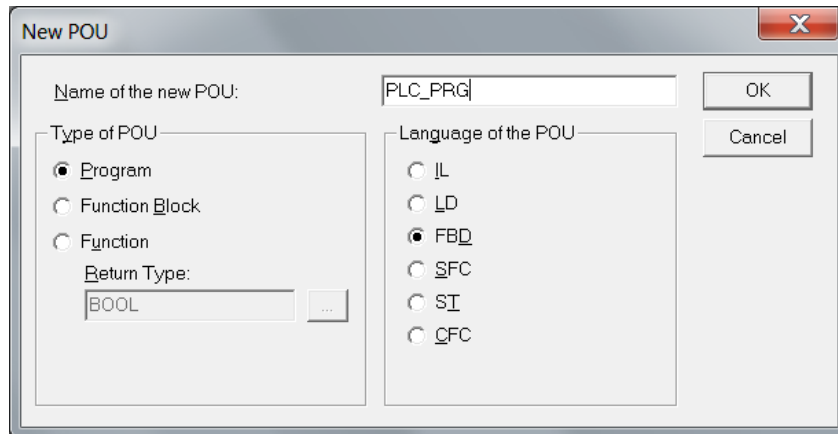


Рис. 1.4. Вікно вибору мови програмування

Вона містить лише один функціональний блок PID (рис. 1.5), що входить до стандартної бібліотеки `Utils.lib`. Він, власне, і реалуватиме ПІД-закон керування технологічним процесом.

Зв'яжемо входи та виходи блоку зі змінними, які надалі будемо використовувати у програмі.

Для навчального прикладу ці змінні зробимо глобальними, хоча в реальних умовах рішення про створення глобальної змінної повинне бути добре обгрунтованим. Це допоможе уникнути випадку з можливим неочікуваним доступом до даних змінної в непотрібний момент.

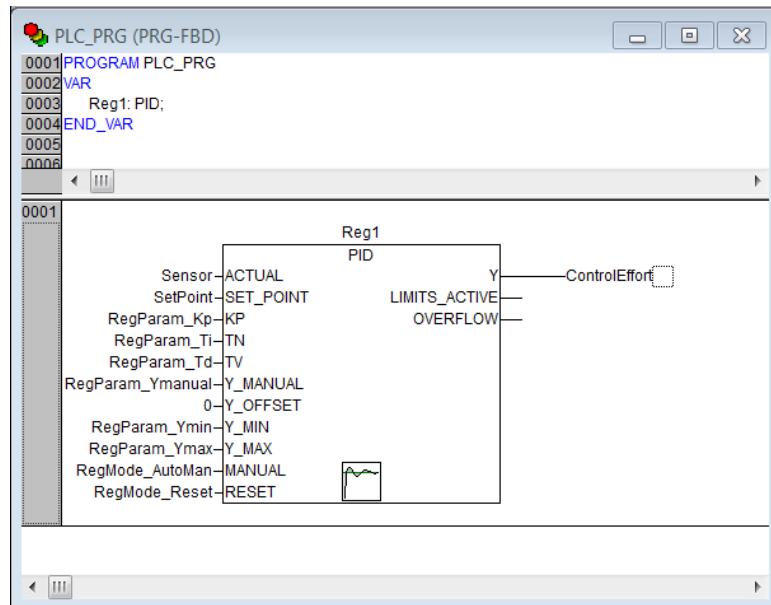


Рис. 1.5. Прикладна програма в CoDeSys.

Перелік глобальних змінних:

#### VAR\_GLOBAL

ControlEffort: REAL; (\*вихід регулятора\*)  
 Sensor: REAL; (\*покази давача — зворотний зв'язок від об'єкта\*)  
 SetPoint: REAL; (\*бажане значення технологічної змінної — уставка\*)  
 RegParam\_Kp: REAL; (\*параметр налаштування Kp\*)  
 RegParam\_Ti: REAL; (\*параметр налаштування Ti\*)  
 RegParam\_Td: REAL; (\*параметр налаштування Td\*)  
 RegParam\_Ymanual: REAL; (\*вихід регулятора у випадку, коли він знаходиться в ручному режимі\*)  
 RegParam\_Ymin: REAL; (\*мінімальне значення виходу регулятора\*)  
 RegParam\_Ymax: REAL; (\*максимальне значення виходу регулятора\*)  
 RegMode\_AutoMan: BOOL; (\*вибір режиму роботи регулятора. Значення TRUE відповідає ручному режиму\*)  
 RegMode\_Reset: BOOL; (\*вимкнення регулятора — якщо значення = TRUE\*)  
 END\_VAR

Змінна Sensor буде надходити від OPC-клієнта, який імітувати об'єкт керування. Керуючий сигнал, значення якого міститься в змінній ControlEffort, також буде передаватися до OPC-клієнта.

Окрім коду, що виконуватиметься контролером, створимо ще й візуалізацію (рис. 1.6), яка містить параметри налаштування системи і графік

зміни параметрів. У тренді відобразяться уставка, значення технологічної змінної (давач) та вихід регулятора.

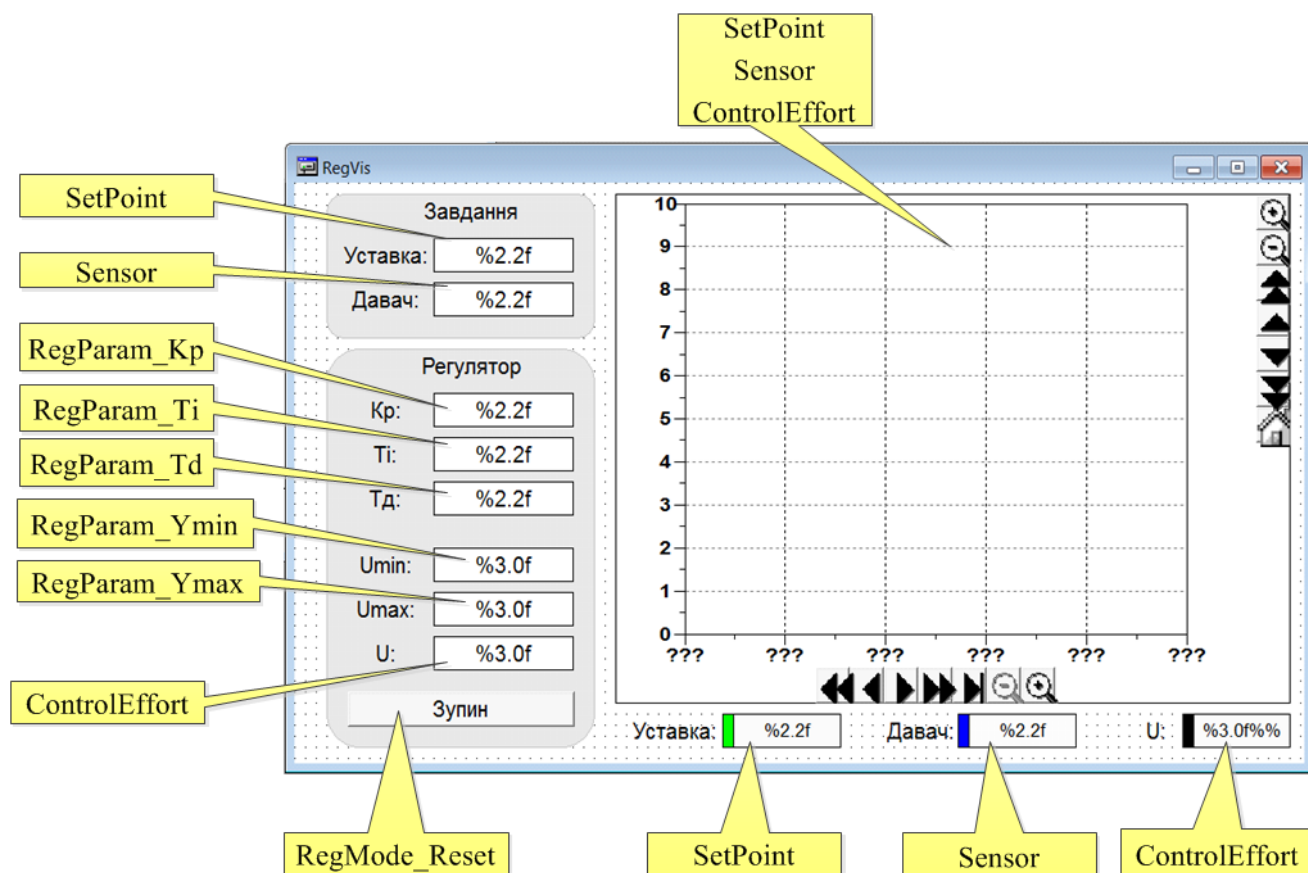


Рис. 1.6. Візуалізація прикладного програмного забезпечення

Для обміну даними з використанням технології OPC необхідно активувати опцію «Створювати опис» (Dump symbol entries), щоб символний файл із змінними генерувався автоматично. У цьому файлі міститиметься опис всіх змінних, які ми хочемо показати зовні. Для цього вибираємо команду меню Project – Options... У вікні, яке з'явилося (рис. 1.7), вибираємо розділ «Symbol configuration» («Символьна конфігурація») і ставимо галочку навпроти напису «Dump symbol entries».

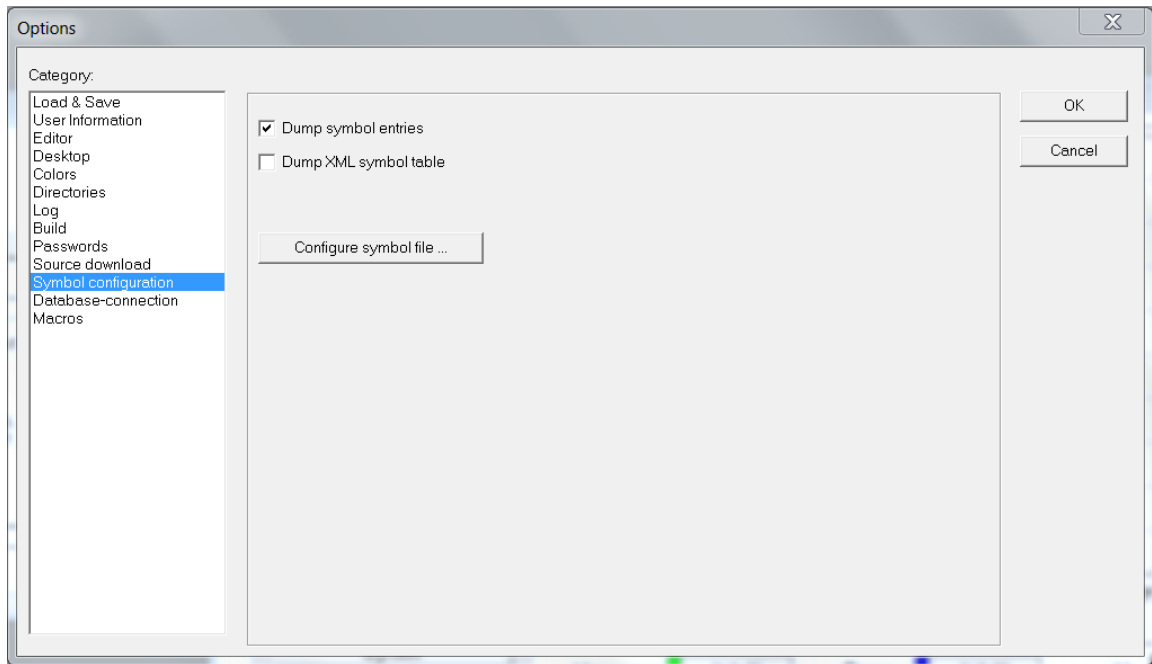


Рис. 1.7. Налаштування проекту

Для вибору власне змінних для обміну даними потрібно натиснути кнопку «Configure symbol file...». У вікні, яке відкрилось (рис. 1.8), можна визначити, до яких змінних буде доступ ззовні, а також характер цього доступу (лише для читання, лише для запису чи для читання та запису одночасно). Характер доступу визначається атрибутами об'єкта.

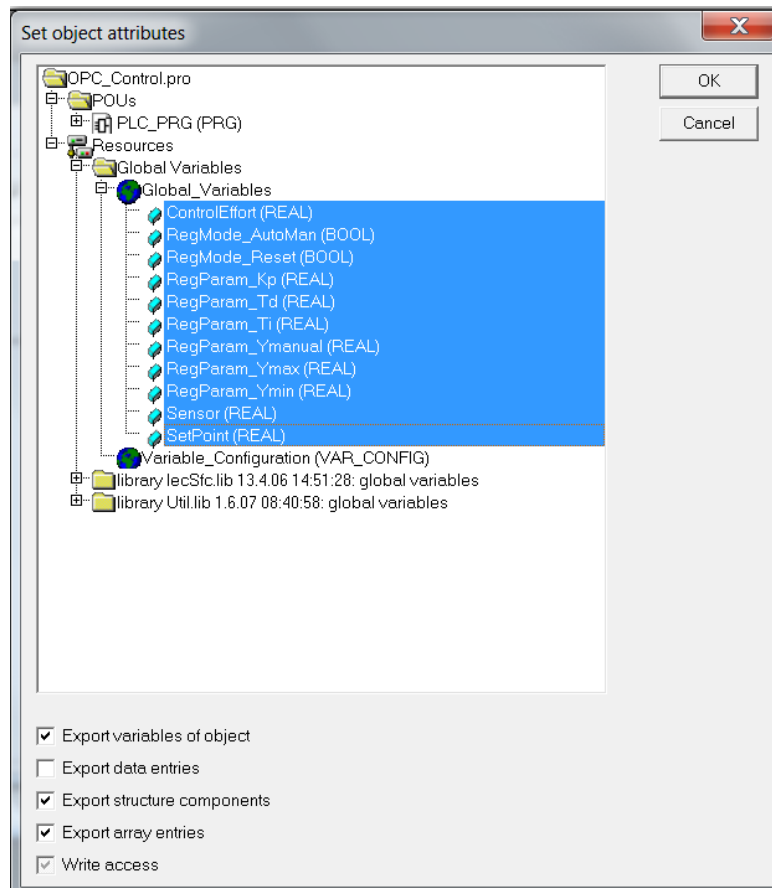


Рис. 1.8. Вибір змінних для символного файлу

Нас цікавлять глобальні змінні, які ми визначили в проекті контролера.

Перелік атрибутів, наведений в нижній частині вікна, стосується обраних на даний момент елементів.

- Export variables of object (Експорт змінних проекта) — якщо опція встановлена, то змінні об'єкта будуть передаватися в символічний файл;
- Export data entries (Експорт даних) — якщо опція встановлена, то будуть генеруватися описи структур та масивів;
- Export structure components (Експорт структур) — якщо опція встановлена, то для кожного елемента структури буде генеруватися власний опис;
- Export array entries (Експорт масивів) — якщо опція встановлена, то для кожного елемента масива буде генеруватися власний опис;
- Write access (Доступ до запису) — якщо опція встановлена, то змінна буде доступна для запису.

Встановимо всім змінним, окрім ControlEffort, атрибут «Write access». Це дозволить змінювати значення з OPC-клієнта. Змінна ControlEffort буде доступна лише для читання, так як за її неї відповідає сам ПІД-алгоритм і не потрібно втручатися в його роботу, підмінюючи його значення.

Встановлюємо параметри зв'язку нашого контролера (Online – Communication parameters...) (рис. 1.9):

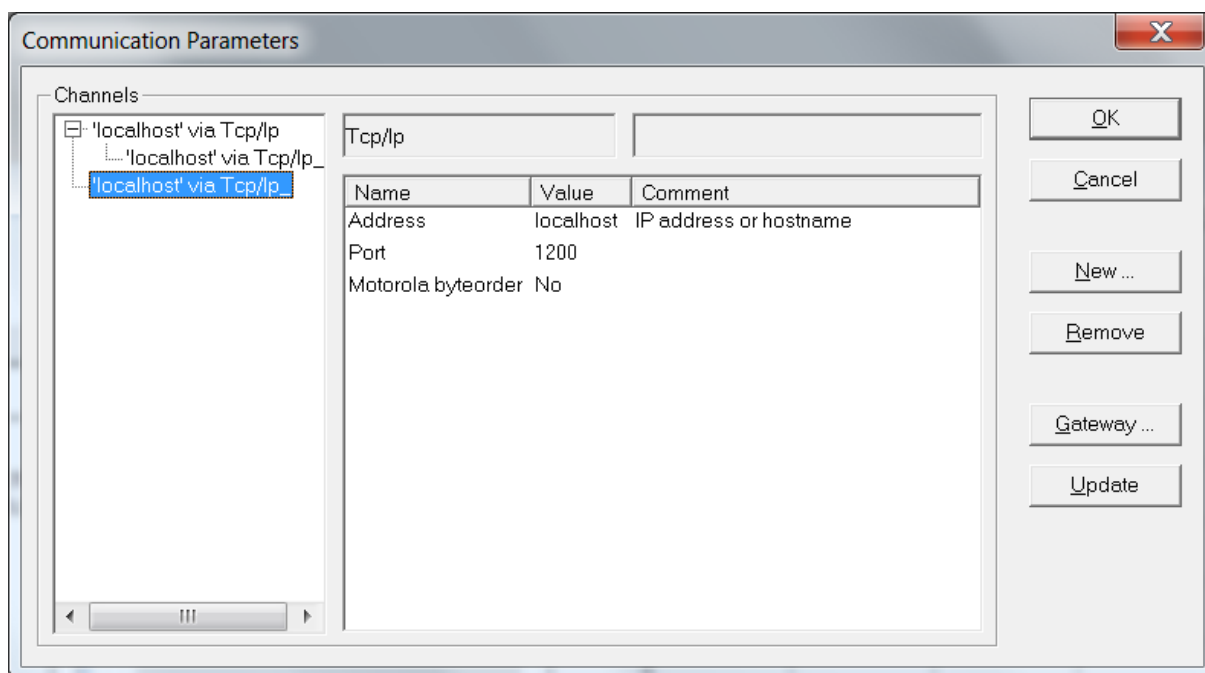


Рис. 1.9. Параметри зв'язку з контролером

Завантажуємо зроблений проект в цільовий контролер (CoDeSys SP PLCWinNT).

Для цього запускаємо віртуальний контролер (рис. 1.10), який можна знайти в директорії з встановленим середовищем CoDeSys  
*3S Software\CoDeSys SP PLCWinNT \ PLCWinNT24.exe*

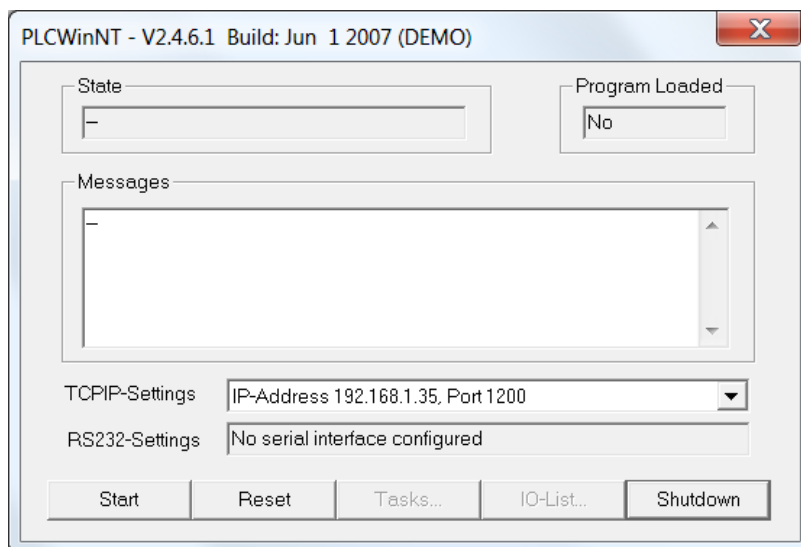


Рис. 1.10. Початкове вікно віртуального контролера CoDeSys SP PLCWinNT

Підключаємося до контролера з середовища CoDeSys (Online -- Login).

Режим симуляції (Simulation Mode) повинен бути вимкнений.

На запит системи про відсутність програми в контролері та рекомендації завантажити її до контролера відповідаємо схвально.

Запускаємо роботу програми, натиснувши кнопку «Start» у вікні контролера (рис 1.11).

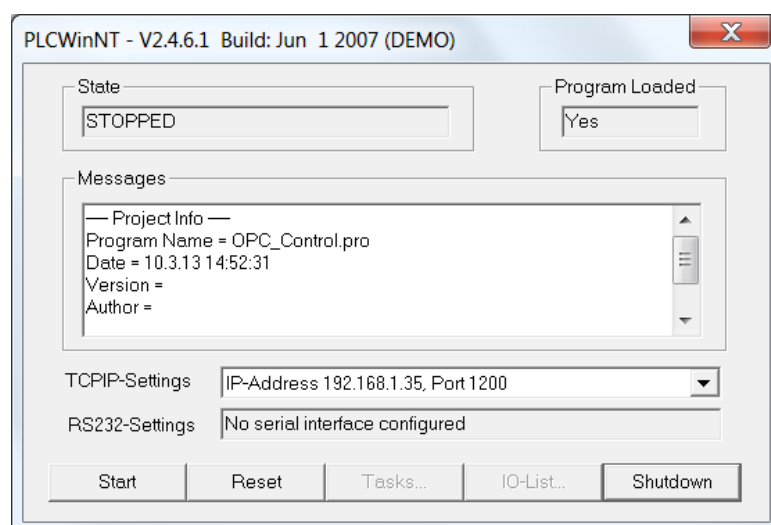


Рис. 1.11. Контролер CoDeSys SP PLCWinNT із завантаженою програмою.

У середовищі CoDeSys тепер можна спостерігати за роботою програми.

## 1.4 Налаштування OPC-сервера

До складу програмного комплексу CoDeSys входить власний OPC-сервер, призначений для роботи з платформою.

Встановивши його на комп'ютер (див. документацію за шляхом `\3S Software\ CoDeSysOPC \ OPC_20_how_to_use_E.pdf`), OPC-сервер потрібно налаштувати для роботи з конкретним контролером. Саме з тим, що виконуватиме створений проект.

Після запуску OPC-сервера (файл `OPCConfig.exe`) з'являється вікно (рис. 1.12):

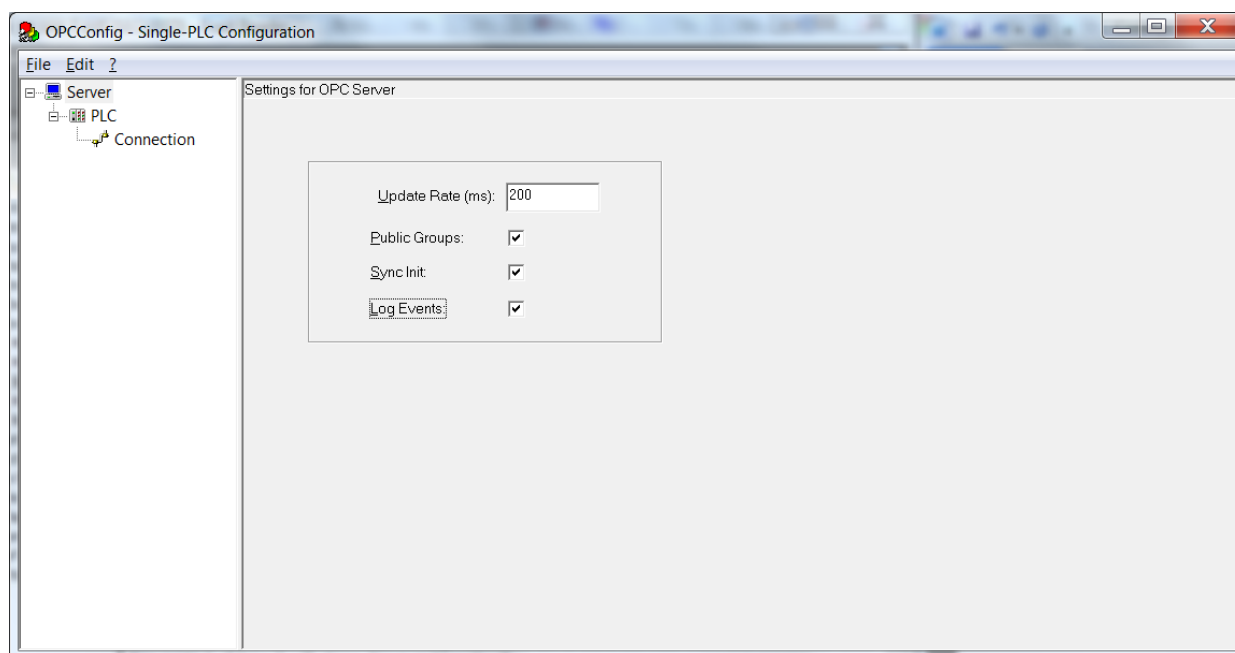


Рис. 1.12. Початкове вікно OPC-сервера

В меню `File` встановлюємо галочку навпроти пункту `Single PLC`. Тепер OPC-сервер працюватиме з одним контролером.

- `Update Rate, ms` (Період оновлення, мс) — період оновлення даних в OPC-сервері (тобто період, з яким він опитує контролери);
- `Public Groups` (Спільні групи) — якщо опція активована, то OPC-сервер використовує спільні групи для окремих ROU та глобальних змінних;
- `Sync Init` (Синхронна ініціалізація) — якщо опція встановлена, то OPC-сервер активується тільки після повного завантаження всіх символічних файлів;
- `Log Events` () — якщо опція активована, то дії OPC-сервера записуються в журнал.

Після цього потрібно встановити налаштування ПЛК (рис. 1.13).

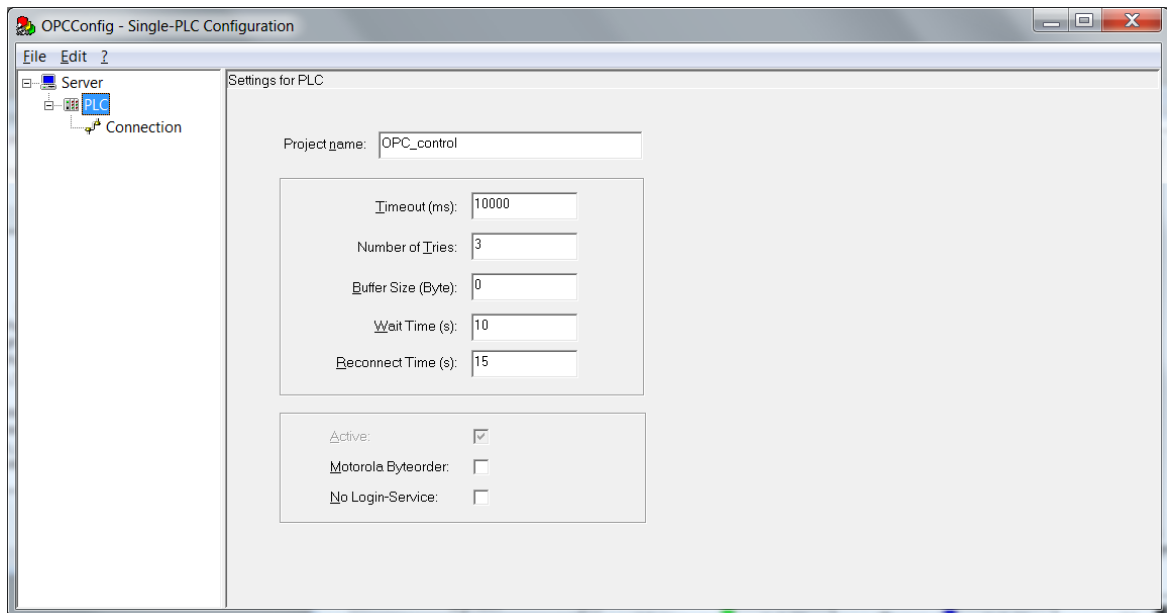


Рис. 1.13. Налаштування OPC-сервера

- Project name (Ім'я проекту) — ім'я завантаженого проекту. Поле не обов'язкове, якщо сервер працює з одним ПЛК, за виключенням випадку, коли потрібно виконати офлайнове завантаження символічного файлу. При цьому шукається симфольний файл спочатку в директорії проекту, і, якщо його там немає, в директорії завантаження Gateway, а самі симболи (змінні) будуть хоча б відображатися в клієнті. Працює з версією OPC-сервера не нижче 2.3.8.4 та Gateway-Server з версією 2.3.3.3 чи вище.
- Timeout (ms) (таймаут, мс) — якщо OPC-сервер не отримує відповіді від контролера впродовж вказаного часу, то автоматично вимикається.
- Number of Tries (кількість спроб) — кількість спроб, які будуть здійснені комунікаційним шлюзом, щоб правильно передати дані. Якщо всі спроби вичерпано, то згенерується повідомлення про помилку.
- Buffer Size (Byte) (розмір буфера, байт) — розмір комунікаційного буфера цільової системи. Якщо встановлене значення «0», буде здійснена спроба отримати цю інформацію від драйвера пристрою. Якщо такої інформації немає, то буфер вважається необмеженим.
- Wait Time (s) (час очікування, с) — час, впродовж якого OPC-сервер чекатиме на готовність контролера (важливо у випадку автоматичного включення контролера).
- Reconnect Time (s) (час перепідключення, с) — часовий інтервал, з яким OPC-сервер намагатиметься перепідключитися до контролера через шлюз.
- Active — індикатор включення контролера в список опитуваних (активно лише для конфігурації Multi PLC).
- Motorola Byteorder — не активовано. Актуально для 68К, 8051, Power PC.



- No Login-Service (без логіну) — опцію потрібно деактивувати для цільових систем (таргетів), які вимагають авторизації.

Тепер виконуємо налаштування з'єднання (рис. 1.14). Тобто як саме OPC-сервер знайде нашій контролер. Вони повинні бути такими ж, як встановлені в проекті контролера.

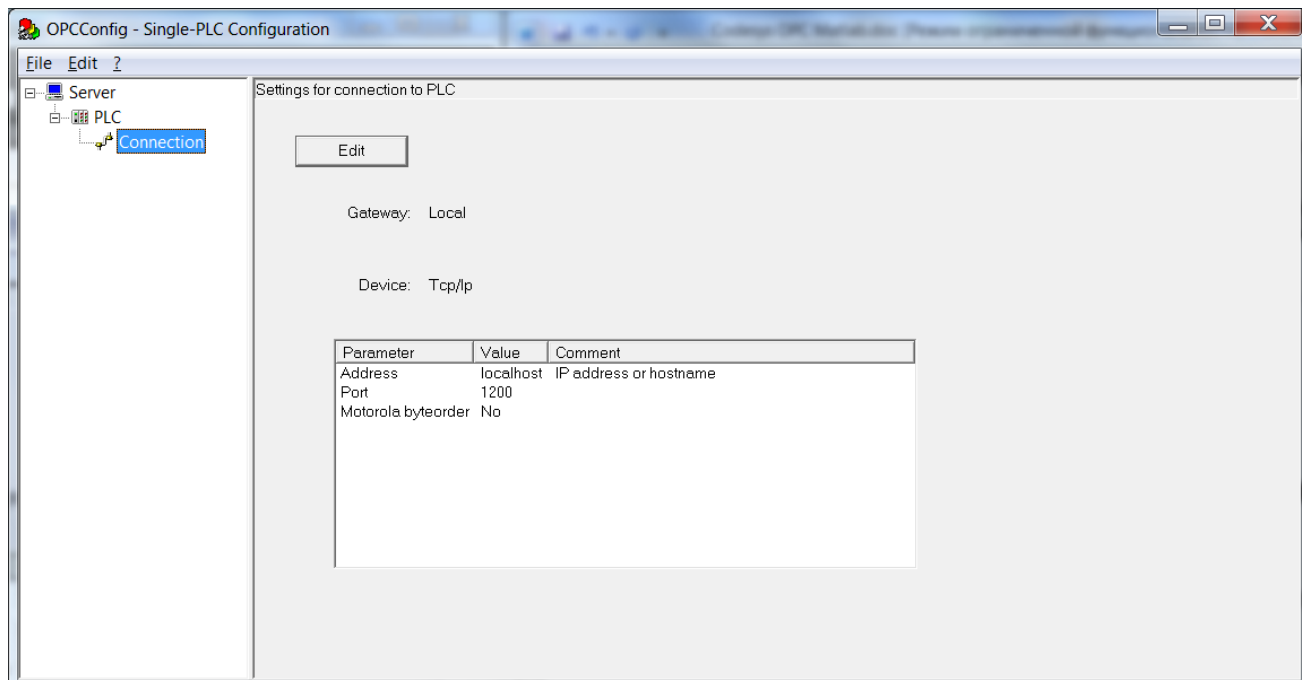


Рис. 1.14. Параметри зв'язку з контролером

Щоб змінити, за необхідності, параметри зв'язку, натискаємо кнопку «Edit» та встановлюємо необхідні значення

## 1.5 Розробка проекту OPC-клієнта

OPC-клієнтом виступатиме Matlab. В ньому, а точніше, в Simulink, відбуватиметься імітація роботи об'єкта керування. OPC-клієнт отримуватиме від OPC-сервера CoDeSys значення виходу ПІД-регулятора, яке розраховане в проекті контролера. Отримане значення поступатиме на вхід об'єкта, а його вихід передаватиме своє поточне значення назад до OPC-сервера. Таким чином контролер ніби отримуватиме значення від давача технологічної величини і по ньому розраховуватиме вихід регулятора по ПІД-алгоритму.

Для роботи з OPC-сервером в Simulink включена бібліотека OPC Toolbox (рис. 1.15).

В ній містяться функціональні блоки для конфігурації підключення OPC-сервера, отримання від нього даних та передачу даних йому, а також визначення статусу OPC-сервера.

**OPC Configuration** визначає OPC-клієнтів, які будуть використовуватися в моделі, налаштовує поведінку моделі в псевдо реальному часі, і визначає поведінку у випадку помилки OPC чи інших подій. Блок не має вхідних портів. Один додатковий вихідний порт відображає час, витрачений на очікування в кожному кроці симуляції для досягнення псевдо-реальної поведінки. В моделі може біти лише один блок конфігурації.

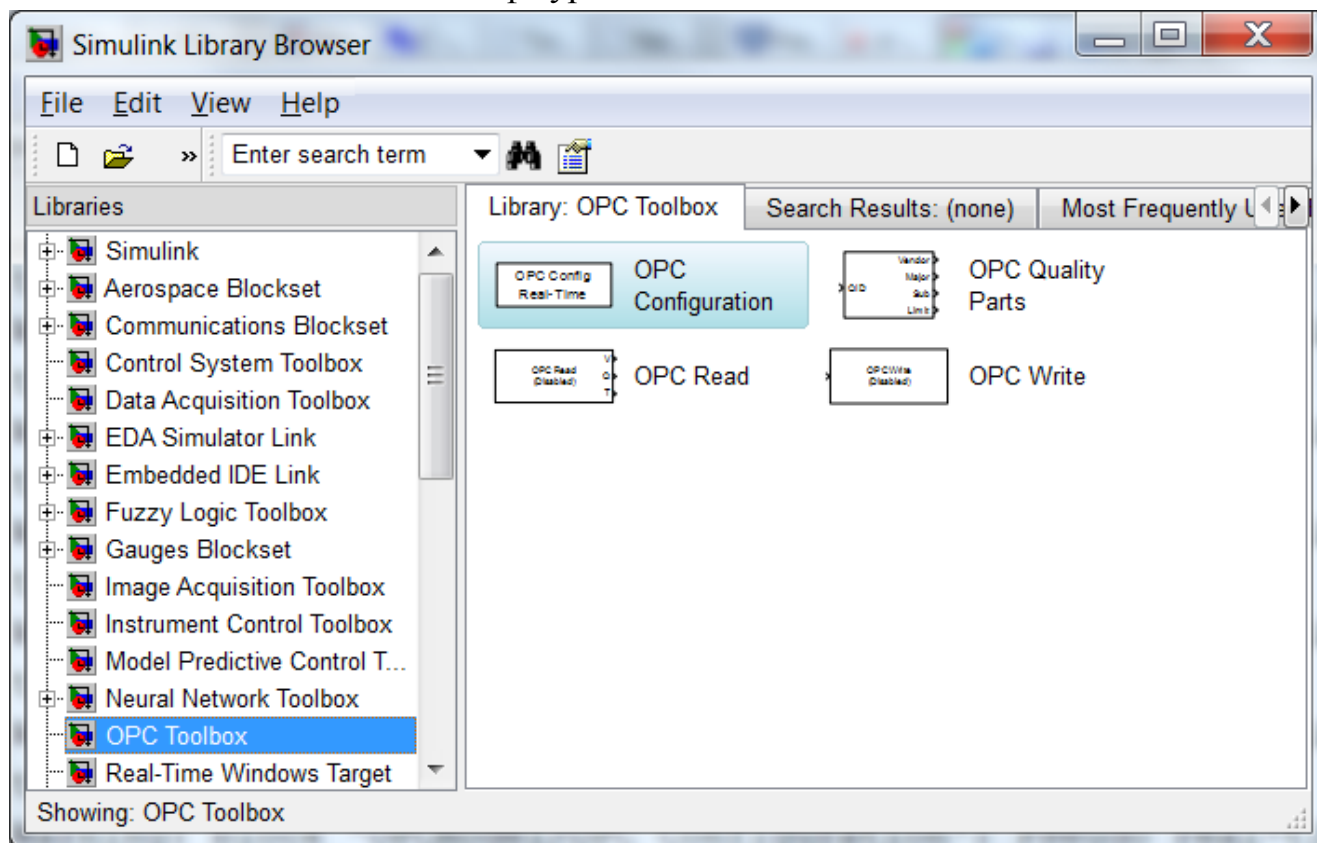


Рис. 1.15. Бібліотека Simulink OPC Toolbox

**OPC Read** зчитує дані з одного або декількох елементів OPC-сервера. Операція читання відбувається синхронно (з кеша або з пристрою) або асинхронно (з пристрою). Блок виводить значення (V) із запитаних елементів у перший вихід, і, у додаткових виходах, виводить якість ідентифікаторів (Q) і міток часу (T), пов'язаних з кожним значенням даних. Мітка часу може бути представлена як дата (реальний час), або як кількість секунд від початку моделювання (моделювання часу). Трійка V, Q, T відображає на вихідні порти останні відомі дані для кожного з елементів, прочитаних блоком. Для роботи блоку у моделі повинен бути блок OPC Configuration.

**OPC Write** записує дані в один або декілька елементів на OPC-сервера. Операції запису відбувається синхронно або асинхронно. Кожен елемент вхідного вектора записується у відповідний пункт у списку «Item IDs», визначений для блоку.

**OPC Quality Parts** перетворює ID-вектор якості у чотири частини: статус постачальника, основний статус якості, підстатус якості і стан обмежень. ID-вектор якості генерується портом Quality (Q) блоку OPC Read.

Створюємо нову модель Simulink (рис. 1.16) та встановлюємо на ній блоки з OPC Toolbox. Також з блоків Transfer Fcn та Transport Delay збираємо модель об'єкта керування вигляду

$$W(s) = \frac{Ke^{-\tau s}}{Ts + 1}$$

з параметрами  $K = 0.65, T = 22, \tau = 7,4$ .

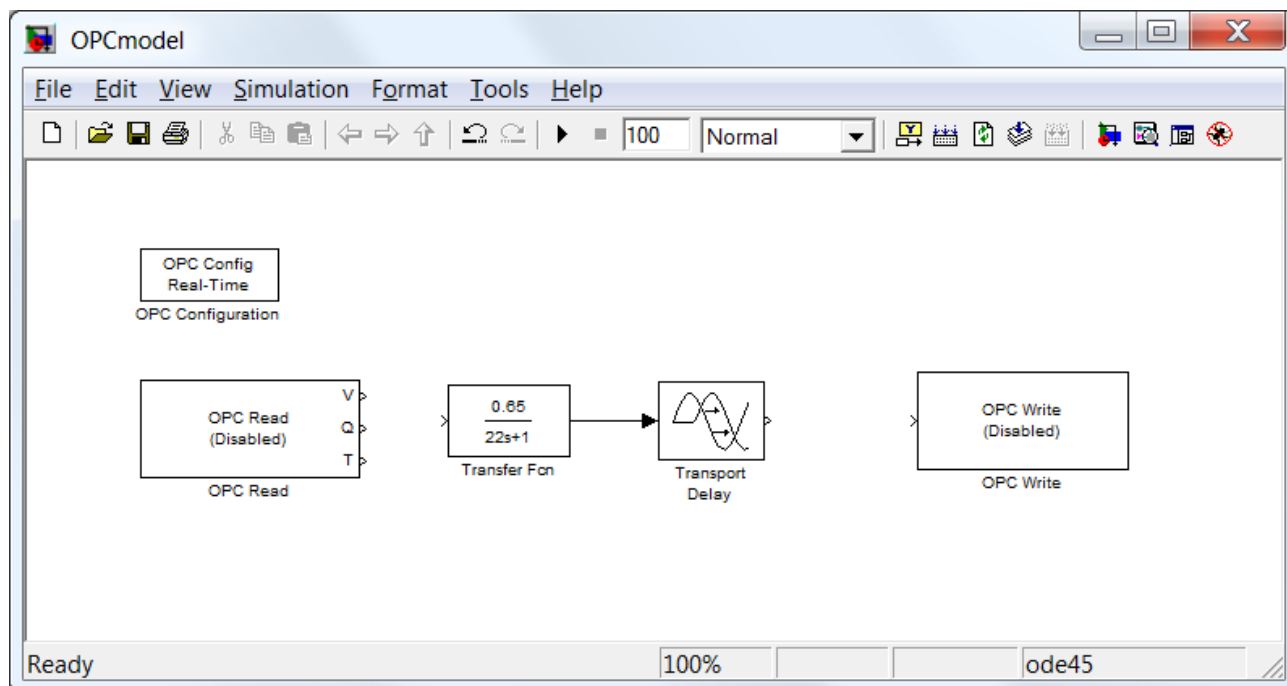


Рис. 1.16. Блоки моделі

Налаштовуємо блок OPC Configuration (рис. 1.17).

1. Відкриваємо налаштування блоку;
2. Переходимо до вибору OPC-сервера — натискаємо кнопку «Configure OPC Clients...»
3. У вікні, що відкрилося, додаємо потрібний OPC-сервер, натиснувши кнопку «Add...».
4. Так як OPC-сервер знаходиться на тому ж комп'ютері, що й Matlab, то не змінюємо адрес Host (там залишається значення localhost), а просто натискаємо кнопку «Select...» і вибираємо OPC-сервер зі списку. У нашому випадку це CoDeSys.OPC.02.

Після цього модель знатиме, до якого сервера буде звертатися за даними.

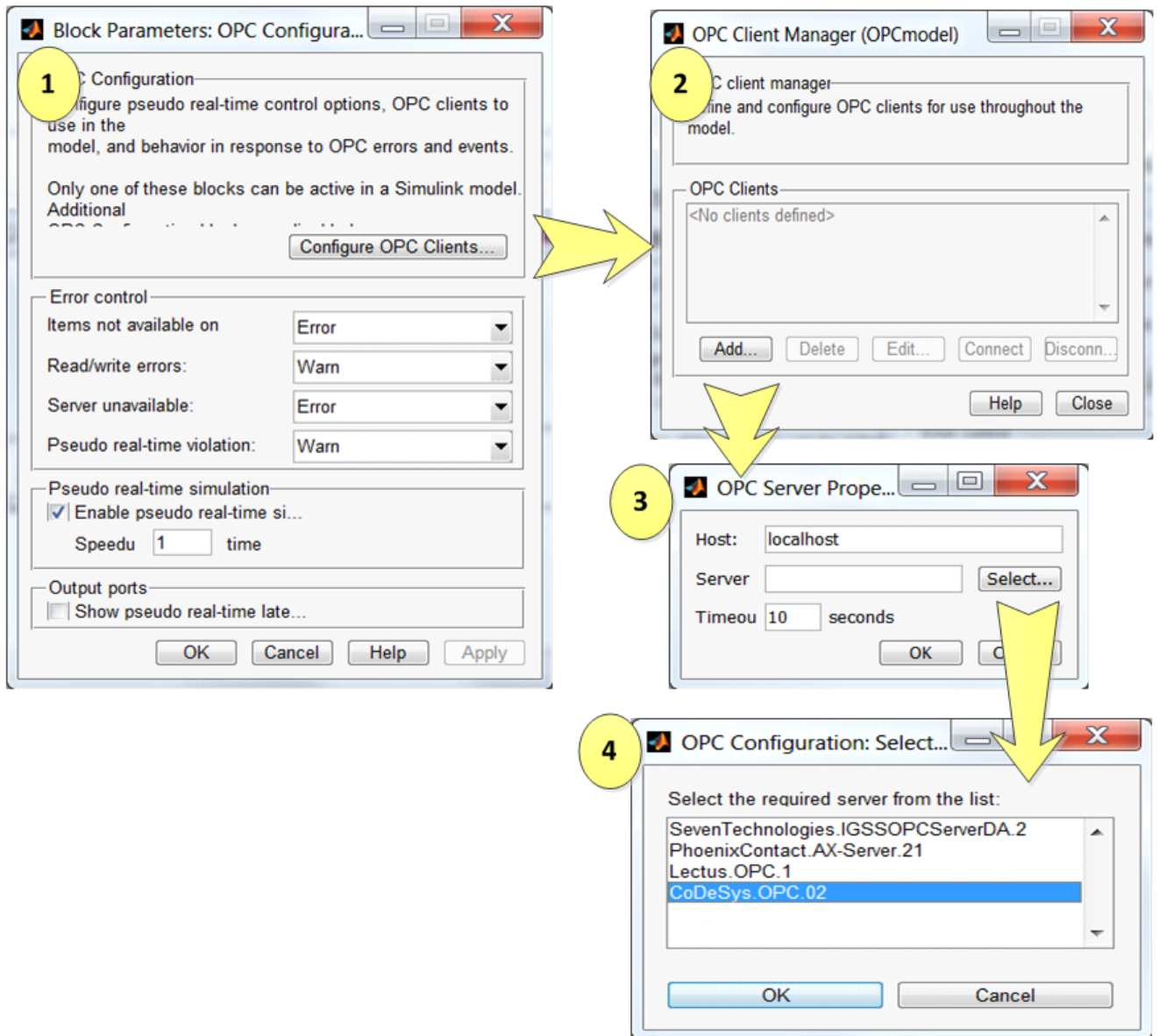


Рис. 1.17. Налаштування блоку OPC Configuration

Тепер налаштуємо зчитування необхідних даних з OPC-сервера (тобто, фактично, з контролера). Для роботи такої простої моделі достатньо зчитати вихід регулятора (ControlEffort), щоб подати його на вхід об'єкта керування (рис. 1.18).

1. Заходимо в налаштування блоку OPC Read. Додаємо елементи (теги OPC-сервера, які нам потрібні). Для цього натискаємо кнопку «Add Item...»;
2. у вікні, що з'явилося, вибираємо тег .ControlEffort
3. Переносимо обрану змінну до списку змінних, які будуть опитуватися, натисканням на кнопку « >> ».

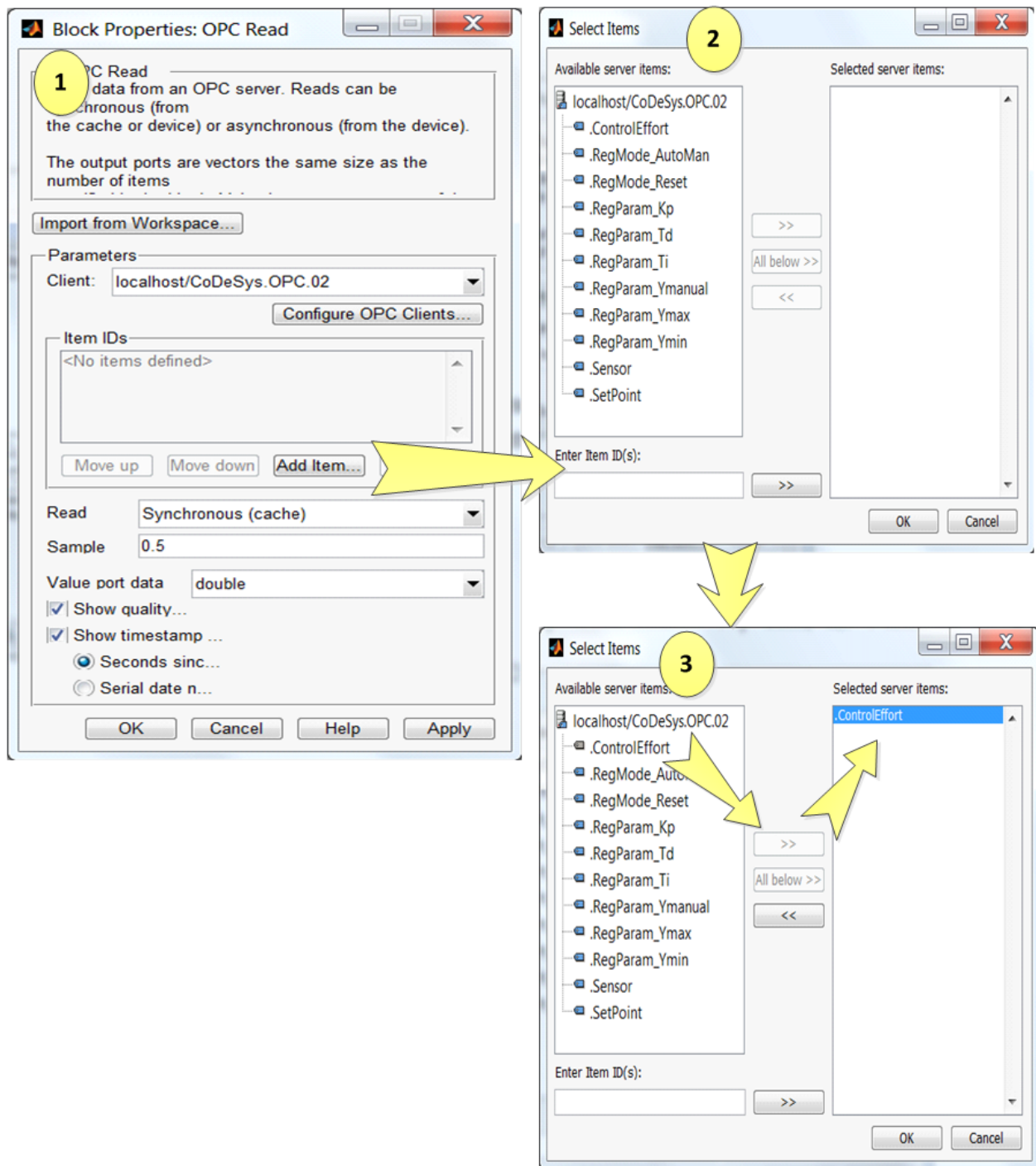


Рис. 1.18. Налаштування блоку OPC Read

Для того, щоб ПІД-регулятор нашого контролера працював вірно, йому потрібно знати покази давача. Давачем у нашій системі виступає вихід об'єкта керування. Саме його значення ми повинні передати контролеру (рис. 1.19).

Для цього налаштовуємо блок OPC Write.

1. Відкриваємо параметри налаштування блоку, натискаємо кнопку «Add item...»
2. Додаємо елемент-змінну, який відповідає давачу системи (.Sensor)

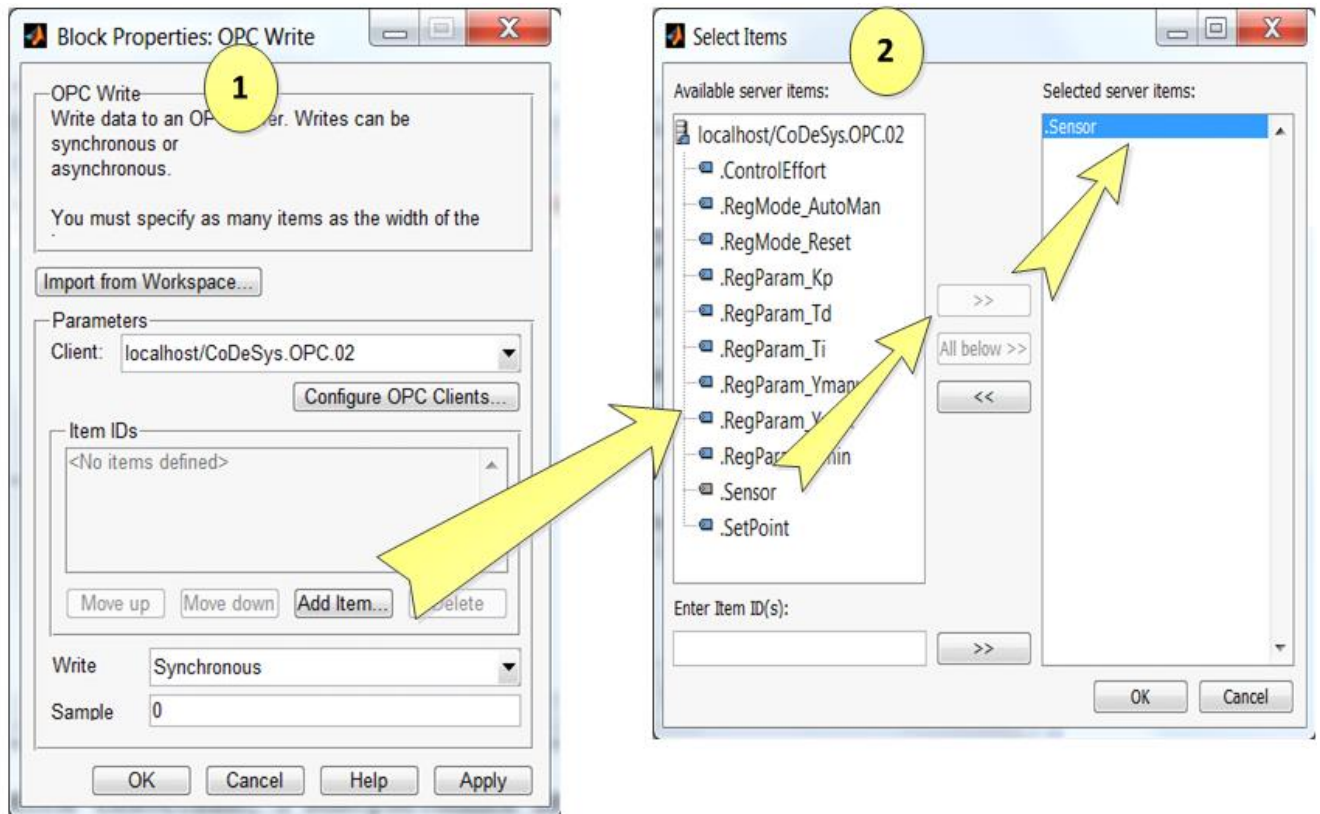


Рис. 1.19. Налаштування блоку OPC Write

Після виконаних процедур завершуємо створення моделі, поєднуючи блоки інформаційними сигналами (рис. 1.20).

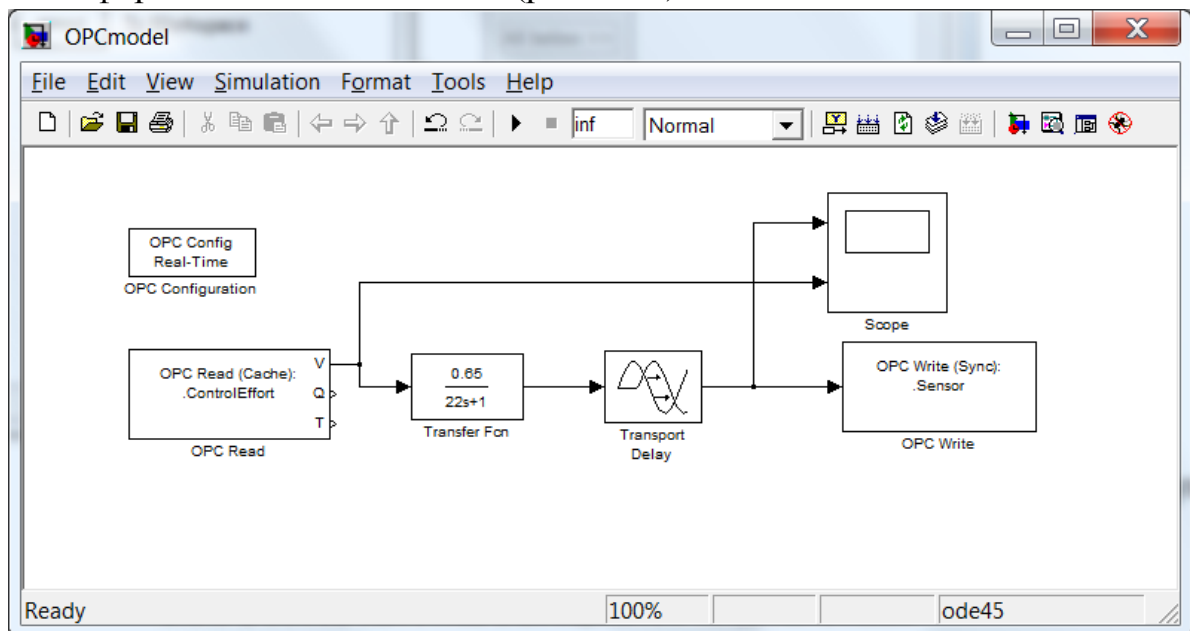


Рис. 1.20. Модель об'єкта керування в Simulink

Для того, щоб спостерігати за моделюванням в реальному часі і без обмежень, встановлюємо час моделювання рівним «inf» (нескінченний).



## 1.6 Робота системи імітаційного моделювання

Після описаних вище процедур система готова до роботи.

Перед запуском переконайтесь, що віртуальний контролер CoDeSys SP PLCWinNT запущено, а CoDeSys має зв'язок з ним.

Відразу в CoDeSys можна встановити потрібні для роботи ПІД-алгоритма значення, скориставшись розробленою візуалізацією:

SetPoint=3

RegParam\_Kp=3,2

RegParam\_Ti=15,4

RegParam\_Td=4

RegParam\_Ymanual=0

RegParam\_Ymin=0

RegParam\_Ymax=0

RegMode\_AutoMan=0

RegMode\_Reset=0

Після цього можна запустити моделювання в Simulink і спостерігати за перебігом імітаційного моделювання (рис. 1.21), використавши блок Scope.

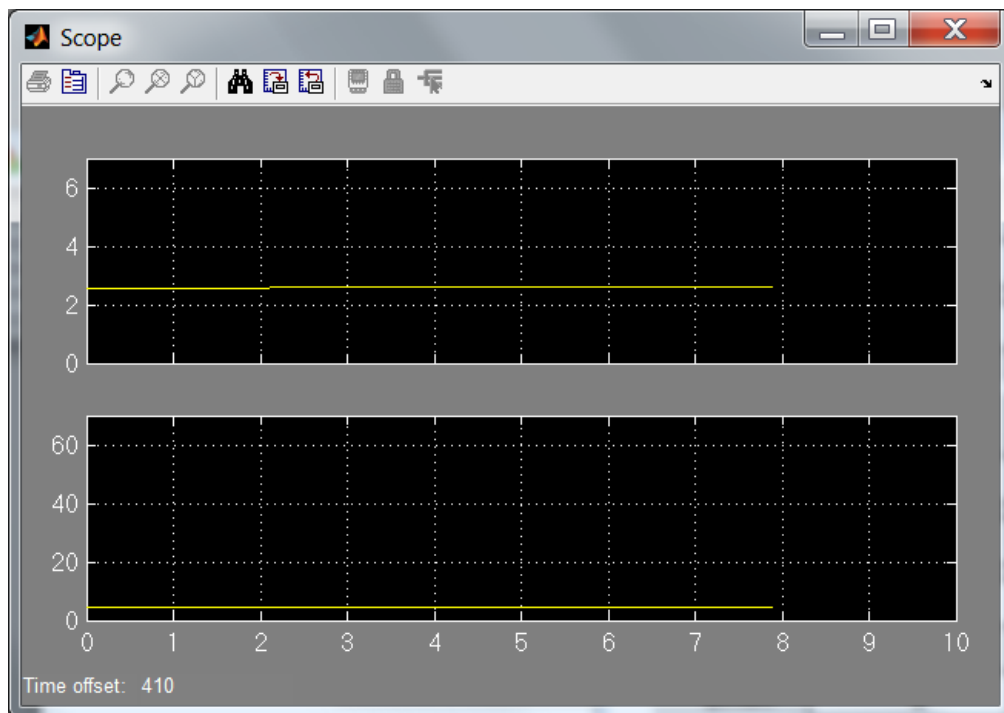


Рис. 1.21. Робота блоку Scope

Аналогічну картину можна спостерігати і у вікні візуалізації CoDeSys (рис. 1.22).

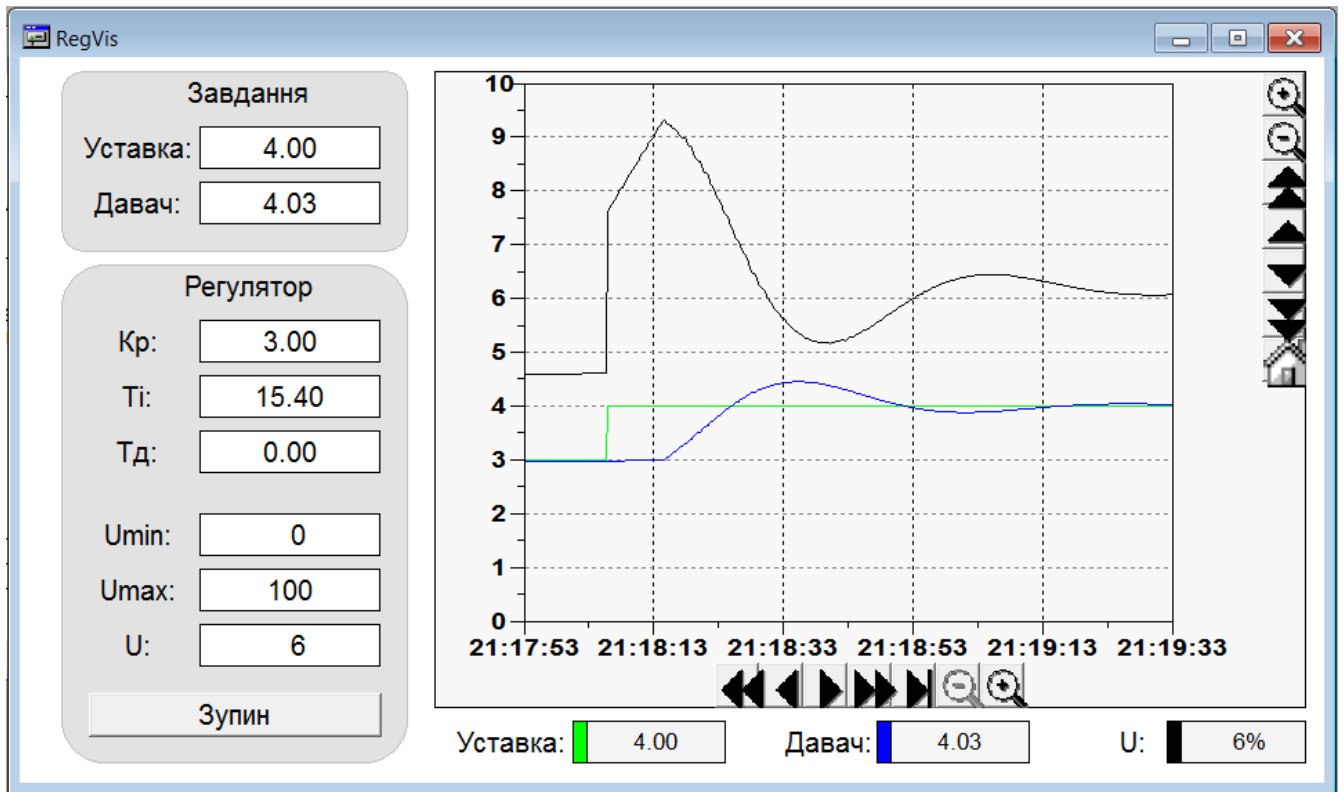


Рис. 1.22. Візуалізація CoDeSys в режимі роботи

Таким чином, побудована система імітаційного моделювання, де об'єкт керування імітується у Matlab, а регулятор — віртуальним контролером. Параметри налаштування регулятора доступні для зміни як із самого CoDeSys, та і із зовнішніх програм-клієнтів (Matlab, SCADA).



## 2 ПОРЯДОК ВИКОНАННЯ РОБОТИ

Складові виконання роботи:

1. Розробити проект регулятора в середовищі CoDeSys.
2. Налаштувати OPC-сервер для взаємодії контролера із OPC-клієнтом.
3. Розробити модель об'єкта керування в Matlab.
4. Дослідити роботу системи автоматичного керування в режимі імітаційного моделювання.

Завдання (згідно варіанту):

1. Розробити імітаційний проект, що реалізує одноконтурну систему керування ПД-регулятором та виконавчим механізмом з аналоговим керуванням у межах 0-100%. Об'єкт керування — аперіодична ланка першого порядку з коефіцієнтом передачі.
2. Розробити імітаційний проект, що реалізує каскадну систему керування з двома П-регуляторами та виконавчим механізмом з аналоговим керуванням у межах 0-100%. Об'єкт керування — аперіодичні ланки першого порядку з коефіцієнтом передачі.
3. Розробити імітаційний проект, що реалізує каскадну систему керування з двома П-регуляторами та трипозиційним виконавчим механізмом постійної швидкості. Об'єкт керування — аперіодичні ланки першого порядку з коефіцієнтом передачі.
4. Розробити імітаційний проект, що реалізує систему автоматичного керування з Під-регулятором та компенсацією збурень. Об'єкт керування — аперіодична ланка першого порядку з коефіцієнтом передачі.
5. Розробити імітаційний проект, яка реалізує двоконтурну систему керування з регулятором та диференціатором. Об'єкт керування — аперіодичні ланки першого порядку з коефіцієнтом передачі.

Розробити імітаційний проект, що реалізує систему керування з ПД-регулятором. Об'єкт керування — аперіодична ланка першого порядку з коефіцієнтом передачі. Завдання регулятора змінюється в залежності від зовнішнього сигналу (представляє собою кусково-задану функцію з трьох пар значень «зовнішній сигнал — завдання регулятору»).

### **3 ПРАВИЛА ВИКОНАННЯ РОБОТИ ТА ВИМОГИ ДО ОФОРМЛЕННЯ ПРОТОКОЛУ**

Робота виконується студентом самостійно, з використанням необхідної довідникової літератури та необхідного програмного забезпечення.

Результати виконання роботи студент оформлює у вигляді протоколу, що містить наступні розділи:

1. титульний аркуш;
2. зміст протоколу;
3. завдання до роботи;
4. короткі теоретичні відомості;
5. хід виконання роботи, що містить необхідні для розуміння та аналізу дані, включно з додатковими математичними залежностями, програмним кодом, блок-схемами, ілюстраціями тощо;
6. висновки.

Виконана робота захищається у вигляді співбесіди, під час якої студент демонструє елементи роботи, використання отриманих знань та навичок у відповідях на питання та виконанні практичних завдань.

Програми повинні супроводжуватися коментарями, словесним описом роботи алгоритму та ілюстрацією роботи алгоритму у вигляді блок-схеми.

## **4 КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Суть та призначення імітаційного моделювання.
2. Технологія OPC – призначення, особливості, функціональність.
3. Процедура розробки моделі об'єкта керування.
4. Процес розробки регулятора в середовищі CoDeSys.
5. Порядок налаштування OPC-сервера CoDeSys.
6. Бібліотека функціональних блоків OPC Toolbox.

## 5 ПЕРЕЛІК ВИКОРИСТАНОЇ ТА РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Петров И.В. Программируемые логические контроллеры [Текст] / И.В. Петров - М. : СОЛОН-Пресс, 2004. – 256 с.
2. Денисенко В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием [Текст] / В.В. Денисенко – М.: Горячая линия–Телеком, 2009. – 608 с.
3. Руководство пользователя по программированию ПЛК в CoDeSys 2.3 [Текст] – 3S - 455 с.
4. Кучерук В. Ю. Програмування логічних контролерів Schneider Electric: Навч. посібник для широкого кола інж.-техн. працівників [Текст] / В.Ю. Кучерук, В. О. Поджаренко, А. І. Кулаков — Вінниця : ВДТУ, 2002. — 131с.
5. Getting Started with OPC Toolbox [Електронний ресурс] / MathWorks — Режим доступа : \www/ URL: <http://www.mathworks.com/help/opc/getting-started-with-opc-toolbox.html> — 23.05.2016 г. — Загол. з екрану.