

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

П. В. Галкін, І. І. Ключник

Програмування ПЛК в CODESYS

Навчальний посібник

544010-TEMPUS-1-2013-1-DE-TEMPUS-JPHES
Тренінги по автоматизованим технологіям України (ТАТУ)

Харків 2019

УДК 001.8 (075.8)
Г16

*Рекомендовано Вченою радою Харківського національного університету
радіоелектроніки, протокол 12/14-4 від 29.06.2017 року*

Головний координатор професор **Reinhard Langmann**, Університет прикладних наук Дюссельдорфа, Німеччина, Competence Center Automation Düsseldorf (CCAD)

Рецензенти **Воропасва В. Я.**, канд. техн. наук, доц., проректор з науково-педагогічної роботи Донецького національного технічного університету (м. Покровськ);

Шапорін Р. О., канд. техн. наук, доц., завідувач кафедри комп'ютерних інтелектуальних систем та мереж Одеського національного політехнічного університету.

Галкін П. В., Ключник І. І.

Г16 Програмування ПЛК в CODESYS : навчальний посібник. Харків :
ФОП Панов А. М., 2019. 92 с.
ISBN 978-617-7722-62-4

Програмування контролерів орієнтовано на використання лабораторних стендів, що були розроблені провідними спеціалістами декількох університетів у рамках міжнародного проекту TEMPUS 544010-TEMPUS-1-2013-1-DE-TEMPUS-JPHES – “Trainings in Avtomation Technologies vor Ukraine” (TATU). TATU Smart Lab – мобільний набір пристроїв, що гнучко налаштовується, для навчання сучасним технологіям автоматизації. Лабораторні стенди містять обладнання різних виробників і розроблені у рамках концепції Industry 4.0 – четвертого етапу промислової революції. У якості інструмента програмування ПЛК, що відповідає міжнародному промислового стандарту IEC 61131-3, розглянуто один з найбільш поширених інструментальних засобів – CODESYS, який є універсальним середовищем, що дозволяє програмувати контролери різних типів.

Призначено для підготовки бакалаврів, фахівців та магістрів у багатьох галузях знань, включаючи інформаційні технології, автоматизацію і приладобудування, електроніку і телекомунікації, виробництво і технології, транспорт та ін., а також для післядипломної освіти інженерів, що забезпечують проектування та експлуатацію систем автоматизації різних рівнів складності (у тому числі інтелектуальних електронних апаратів, авіоніки тощо).

Зміст даної книги відображає точку зору авторів і Європейська Комісія не несе відповідальності за використання інформації, що в ній міститься (The content of this book reflects the views of the authors and the European Commission is not responsible for the use of information contained therein.)

УДК 001.8 (075.8)

ISBN 978-617-7722-62-4

© П. В. Галкін, І. І. Ключник, 2019

ЗМІСТ

Вступ	4
Розділ 1	
Учебний комплекс TSL – TATU SMART LAB	6
1.1. Апаратний модуль «Програмовані контролери і PROFINET»	6
1.2. Апаратний модуль «PROFIBUS»	23
1.3. Апаратний модуль «Моделювання процесів»	27
1.4. Комутація апаратних модулів	30
Розділ 2	
Програмне забезпечення контролера BERGHOF	33
2.1 Середовище CODESYS	33
2.2 Апаратні вимоги до середовища	33
2.3 Інсталяція середовища	33
Розділ 3	
Перші кроки роботи в середовищі CODESYS	35
3.1. Створення першого проекту	35
3.2. Створення першої програми	37
3.3 Налаштування ресурсів CODESYS для запуску і керування додатком в ПЛК	40
3.4 Компіляція і завантаження додатка в ПЛК	42
3.5 Запуск додатка і моніторинг змінних	42
3.6 Установка кроків і точок зупинки	45
3.7 Створення проекту для контролера BergHof	45
3.8 Налаштування підключення контролера BergHof з комп'ютером	48
3.9 Оновлення прошивки контролера Berghof	50
3.10 Установка з'єднання з контролером Berghof	50
3.11 Налаштування терміналу Berghof ET1007	51
Розділ 4	
Приклади проектів	53
4.1 Створення простої панелі керування	53
4.2 Приклад програми для роботи світлофора	60
Розділ 5	
Лабораторний практикум	66
5.1 Знайомство с середою розробки програмного забезпечення промислових контролерів	66
5.2 Розробка інтерфейсу оператора для ПЗ ПЛК	74
5.3 Застосування бібліотечних компонентів	82
5.4 Розробка кодового замка	86
Висновок	88
Література	89
Список скорочень	90

Вступ

ПЛК – програмований логічний контролер (англ. PLC – Programmable Logic Controller) є електронною складовою промислового контролера, що являє собою основу сучасних засобів автоматизації.

Застосування ПЛК в якості спеціалізованих комп'ютеризованих засобів автоматизації передбачає їхнє тривале автономне використання практично без обслуговування і втручання людини, частіше всього, в складних умовах експлуатації.

ПЛК належать до пристроїв, призначених для роботи в системах реального часу і мають декілька суттєвих відмінностей від інших подібних електронних пристроїв: мікроконтролерів, вбудованих систем, комп'ютерів. На відміну від мікроконтролерів – однокристальних комп'ютерів, що реалізуються у вигляді окремої мікросхеми та використовуються для керування електронними пристроями, сферою застосування ПЛК є автоматизовані технологічні процеси в промисловості, енергетиці, на транспорті і таке інше.

Порівняно з мікропроцесорами з жорсткою логікою ПЛК прийнятніші у разі одиничного і дрібносерійного виготовлення систем керування, особливо при необхідності їх адаптації до об'єктів керування.

ПЛК виготовляються як самостійні вироби, що відрізняє їх від вбудовуваних систем. На відміну від комп'ютерів, орієнтованих на ухвалення рішень і керування оператором, ПЛК, переважно, працюють з датчиками і виконавчими механізмами.

Таким чином, ПЛК – це елементна база сучасних систем автоматизації. Структура ПЛК, окрім процесора, містить пристрої пам'яті (ОЗП, ПЗП), порти входів/виводів (I/O), інтерфейси зв'язку, таймери, системний годинник і периферійні пристрої, що забезпечують роботу і взаємодію усіх складових частин і зовнішніх пристроїв ПЛК за допомогою спеціальних мікропрограм, що зберігаються в його внутрішній пам'яті. ПЛК також може містити наступні інтерфейси: RS-232, RS-485, Modbus, CC-Link, Profibus, Device Net, CAN, AS-interface, промисловий Ethernet.

В теперішній час ПЛК числові операції реалізуються нарівні з логічними. Сучасні ПЛК є вільно програмованими. Усі мови програмування ПЛК мають легкий доступ до маніпулювання бітами в машинних словах,

на відміну від більшості високорівневих мов програмування сучасних комп'ютерів.

Інструменти програмування ПЛК на мовах міжнародного промислового стандарту ІЕС 61131-3 можуть бути спеціалізованими (орієнтованими на окреме сімейство ПЛК) або універсальними, такими, що працюють з декількома типами контролерів. Найбільш використовуваними інструментальними засобами є: CODESYS, ISaGRAF, Beremiz, KLogic, ІСР «Кругол».

На світовому ринку основними постачальниками ПЛК є компанії: Allen-Bradley, Berghoff, Siemens, Schneider Electric, Phoenix Contact, Mitsubishi Electric, Advantech, Delta, ViPA, WAGO I/O, Segnetics, Овен, АТ «КОНСТАР» та ін.

Безперервний ріст рівня автоматизації систем керування процесами і об'єктами в найрізноманітніших областях економіки стимулює потребу ринку праці в кваліфікованих фахівцях, що забезпечують проектування, конструювання, експлуатацію і ремонт сучасних засобів і систем автоматизації.

У зв'язку з цим усе більш актуальною є підготовка фахівців, що мають достатній рівень компетентності в області програмування контролерів, організації передачі даних в цифрових мережах керування, у тому числі і бездротових.

Моніторинг і аналіз попиту ринку постійних і потенційних користувачів освітніх послуг університетів, що мають визнання як на національному так і на міжнародному рівнях, підтверджує необхідність підготовки вказаних фахівців у багатьох галузях знань, включаючи інформаційні технології, автоматизацію і приладобудування, електроніку і телекомунікації, виробництво і технології, транспорт та ін.

Розділ 1

УЧБОВИЙ КОМПЛЕКС TSL – TATU SMART LAB

TATU Smart Lab (TSL) – мобільний набір пристроїв, що гнучко налаштовується, для навчання сучасним технологіям автоматизації. Містить обладнання різних виробників і розроблений у рамках концепції Industry 4.0 (четвертий етап промислової революції). Це німецька приватно-державна програма, у рамках якої великі німецькі компанії-розробники в області інформаційних технологій створюють повністю автоматизовані виробництва, в яких окремі пристрої і їх вузли можуть взаємодіяти один з одним і споживачами з використанням бездротових технологій передачі даних.

TSL дозволяє вивчити низку сучасних технологій побудови систем автоматизації. У рамках проекту TATU ці технології розділені на наступні учбові модулі.

1. Програмування контролерів в інструментальному середовищі розробки проектів PC Worx і незалежному від апаратного забезпечення середовищі розробки CODESYS.
2. Використання стандартів Profinet і Modbus TCP і інтеграція систем автоматизації з мережами PROFIBUS.
3. Бездротові технології передачі даних.
4. Керування процесами реального часу.
5. Введення в стандарт обміну даними в реальному часі OPC DA.

TSL складається з трьох апаратних модулів, кожен з яких розміщений в окремому переносному ящику (валізі), розроблений для вивчення певних тем і може бути використаний незалежно від інших. Модулі TSL, їх взаємодія і можливе кабельне підключення показані на рис. 1.1.

1.1. Апаратний модуль «Програмовані контролери і PROFINET»

Перший апаратний модуль (рис. 1.2) називається «Програмовані контролери і PROFINET», оскільки містить два програмовані контролери (AXC 3050 і ILC151GSM/GPRS), пристрої введення/виведення PROFINET, керований мережевий комутатор і бездротову точку доступу для організації бездротової локальної мережі.

Цей апаратний модуль може бути використаний для вивчення широкого діапазону тем, починаючи від базового програмування контролерів в PC Worx до роботи з технологіями передачі даних, розроблених на базі технології Ethernet і

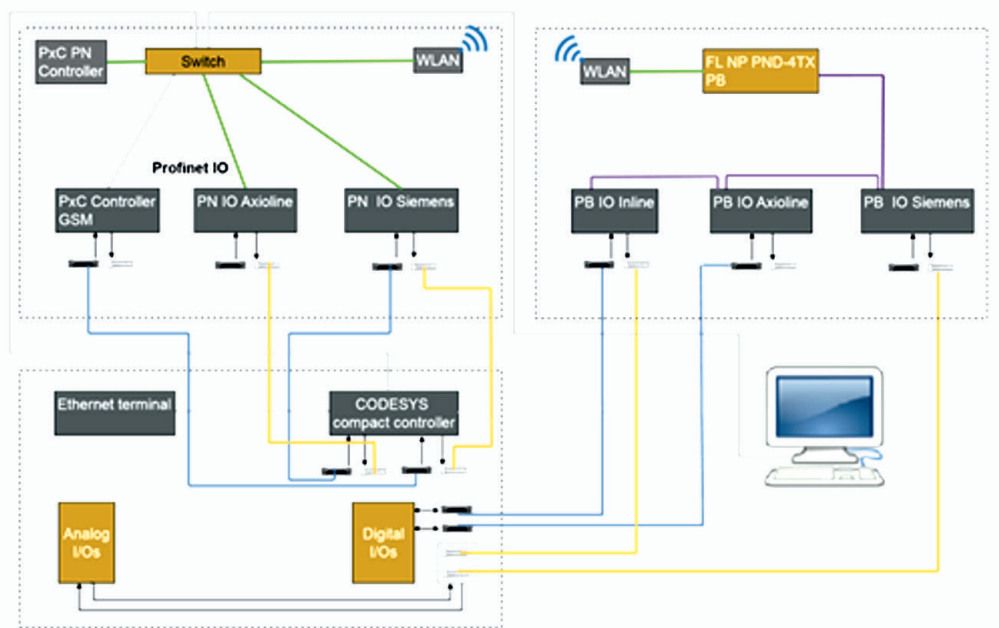


Рис. 1.1 – Можливе з'єднання трьох TSL стендів та ПК

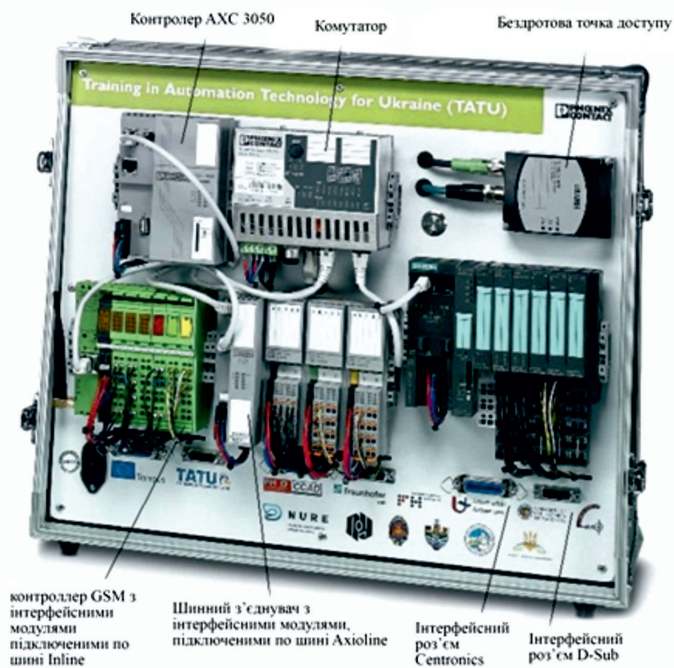


Рис. 1.2 – Зовнішній вигляд першого апаратного модуля TSL «Програмовані контролери і PROFINET»

бездротових технологій передачі даних. Два програмовані контролери можуть взаємодіяти, використовуючи протоколи передачі даних TCP/IP або Modbus TCP. Пристрої введення/виведення PROFINET можуть бути підключені тільки до контролера АХС 3050, а контролер ILC151 GSM/GPRS Inline не має функціональності PROFINET. Таким чином, реалізовані в цьому модулі пристрої і технології дозволяють вивчити матеріал учбових модулів 1, 2 і 3.

На рис. 1.3 показана схема з'єднання пристроїв в першому апаратному модулі. Програмований контролер АХС3050 (рис. 1.4) має можливості роботи з мережами сімейства Ethernet і локальною шиною AxiolineF, яка підтримує будь-які протоколи передачі даних на базі Ethernet. Станція Axioline може бути створена підключенням модулів Axioline до контролера. Локальна шина AxiolineF, детальніше описана далі, може бути задіяна при послідовній установці різних модулів (пристроїв) одного впритул до іншого.

На рис. 5 показано підключення напруги живлення. Контролер АХС3050 може бути повністю сконфігурований і запрограмований на одній з п'яти мов

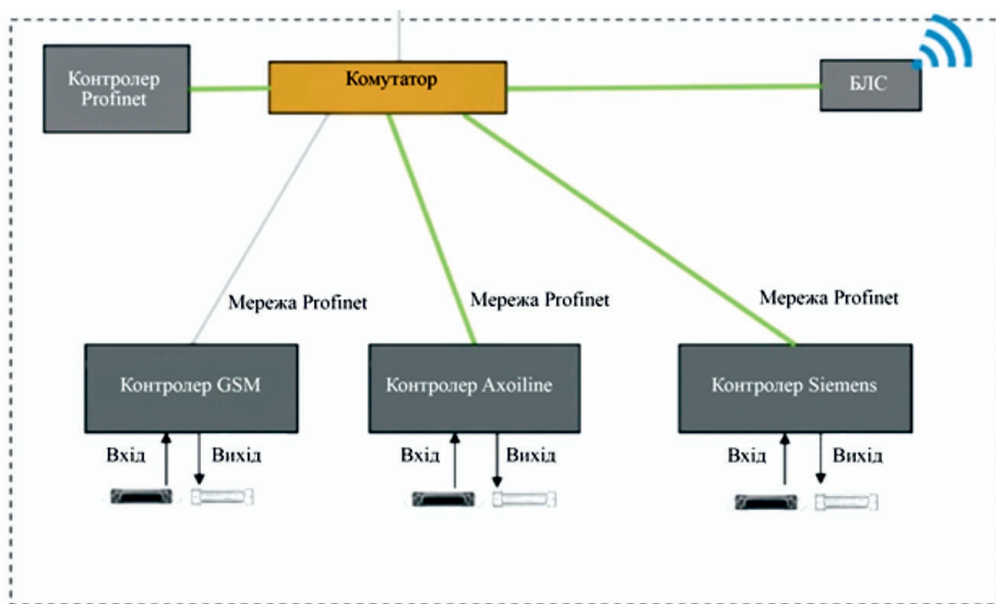


Рис. 1.3 – Підключення пристроїв в першому апаратному модулі TSL «Програмовані контролери і PROFINET»

програмування відповідно до стандарту МЭК 61131-3 за допомогою PC Worx при підключенні по мережі Ethernet.

У контролері АХС 3050 вбудовані інтерфейси підключення пристроїв по мережі Ethernet. По ній можна отримати доступ до контролера з використанням протоколів передачі даних TCP/IP або UDP (User Datagram Protocol – протокол призначених для користувача датаграм).

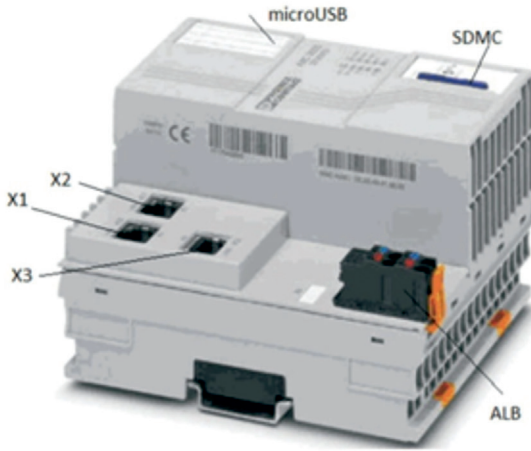


Рис. 1.4. – Контролер АХС3050:
X1, X2, X3 – інтерфейси для підключення до мереж сімейства Ethernet (восьмиконтактні роз'єми RJ-45);
SDMC (Secure Digital Memory Card – слот для підключення флеш-карток типу Secure Digital);
microUSB – роз'єм для підключення USB-пристроїв, прихований під паперовою біркою для інформаційних написів;
ALB – шина AxioLine

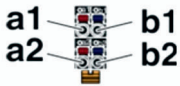


Рис. 1.5 – Цоколювання контактів на виносній панелі:
a1, a2 (червоні клеми) – напруга живлення 24 В постійного струму;
b1, b2 (сині клеми) – заземлення

Контролер має три інтегровані порти Ethernet X1, X2, X3 (див. рис. 1.4). Використовуючи в PC Worx функціональні блоки IP_USEND (відправка призначених для користувача даних по протоколу TCP/IP) і IP_URCV (отримання призначених для користувача даних по протоколу TCP/IP), можна організувати обмін даними (тобто значеннями змінних, що відповідають вимірюваним технологічним параметрам і фізичним величинам) між контролерами.

Такий підхід дозволяє реалізувати розподілені рішення, що конфігуруються, з автоматизації. При використанні сервера AX OPC (Object Linking and Embedding for Process Control, зв'язування і впровадження об'єктів для керування процесами, – сімейство програмних технологій, що надають єдиний інтерфейс для керування об'єктами автоматизації і технологічними процесами) контролер доступний по мережі Ethernet і може бути використаний в програмних пакетах візуалізації.

Технологія PROFINET може бути реалізована підключенням до інтерфейсів Ethernet контролера АХС 3050. Контролер PROFINET постійно доступний при підключенні через восьмиконтактний роз'єм RJ45 інтерфейсу X3. Функціональність PROFINET може бути активована на Ethernet-інтерфейсах X1, X2, X3 (див. рис. 1.4). За умовчанням ця функція відключена і може бути активована в PC Worx.

Технологія Modbus TCP також може бути реалізована підключенням до інтерфейсів Ethernet контролера АХС3050. Цей контролер може виступати в ролі клієнта Modbus, а при використанні його відповідних функціональних блоків може бути конфігурований і як сервер MODBUS TCP.

У нижній частині контролера АХС3050 є інтерфейс зв'язку з локальною шиною АхіолінеF (див. рис. 1.4) для підключення різних модулів.

До контролера можна підключити до 63 пристроїв. Реальне число пристроїв залежить від загального споживання струму усіма пристроями, яке не повинне перевищувати максимальний струм, яким контролер забезпечує локальну шину. Завдяки Web-інтерфейсу керування, інтегрованому в контролер, користувач може візуалізувати статусну і діагностичну інформацію від контролера у браузері.

Контролер АХС3050 обладнаний двома інтерфейсами USB (див. рис. 1.4).

Контролер АХС 3050 має внутрішню пам'ять. Вона може бути використана для зберігання програм і конфігурацій призначеного для користувача проєкту. Якщо внутрішня пам'ять недостатня для створеного застосування, АХС 3050 може працювати із зовнішньою пам'яттю у вигляді карти флеш-пам'яті формату SD (Secure Digital) або USB-накопичувачем. Контролер має 4 Мбайт внутрішньої пам'яті для зберігання програм, і 8 Мбайт пам'яті для зберігання даних; 128 кбайт використовується для зберігання даних після виключення живлення. Мінімальний час циклу контролера – 1 мс, одночасно виконуване число завдань керування – 16.

Контролер ILC 151GSM/GPRS (рис. 1.6) – невеликий, масштабований, модульний контролер, що має інтегровані порти для підключення мереж Ethernet і Interbus і вбудований чотирихдіапазонний модем.

Контролер можна конфігурувати і програмувати в PC Worx, використовуючи підключення по мережі Ethernet, на усіх п'яти мовах програмування від-

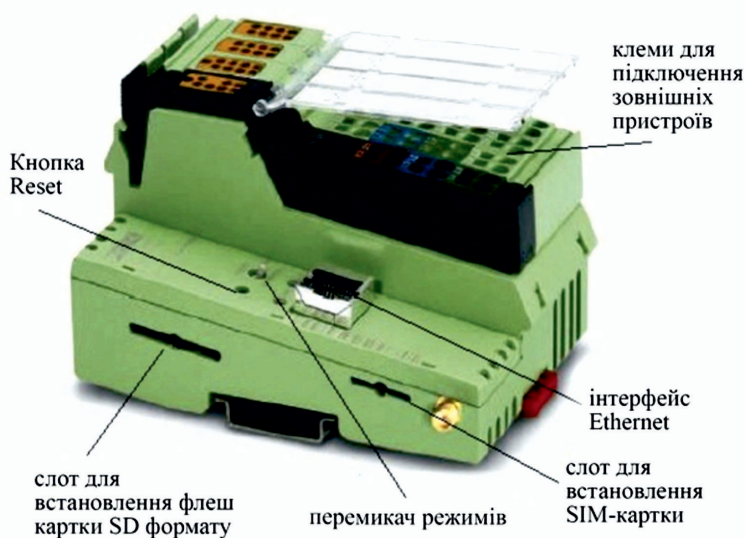


Рис. 1.6 – Контролер ILC151GSM/GPRS

повідно до стандарту МЭК 61131-3. Підключення до мережі Ethernet виконується по кабелю «вита пара», а для доступу до контролера використовуються протоколи передачі даних TCP/IP і UDP/IP.

Вбудовані комунікаційні функції дозволяють організувати обмін даними по мережі Ethernet. Використовуючи в PC Worx функціональні блоки IP_USEND (відправка призначених для користувача даних по протоколу TCP/IP) і IP_URCV (отримання призначених для користувача даних по протоколу TCP/IP), можна організувати обмін даними (тобто значеннями змінних, що відповідають вимірюваним технологічним параметрам і фізичним величинам) між контролерами. Такий підхід дозволяє реалізувати розподілені рішення, що конфігуруються, з автоматизації.

При використанні сервера AX OPC контролер доступний по мережі Ethernet і може бути використаний в різних пакетах візуалізації.

Комунікаційний протокол Modbus TCP може бути задіяний через інтерфейси Ethernet контролера ILC 151GSM/GPRS. Контролер може виступати в ролі клієнта Modbus.

Локальна шина Inline і віддалена шина Interbus задіюються через відповідне підключення. В цьому випадку розробник може створити повноцінну Interbus-систему (максимум чотири рівні віддаленої шини).

Рівень входів/виходів підключений до цього контролера по шині Interbus.

Контролер може працювати із зовнішньою пам'яттю у вигляді карти флеш-пам'яті формату SD. Вона може використовуватися для зберігання програм і конфігурацій призначеного для користувача проекту. Ця зовнішня пам'ять необов'язкова і не вимагається для нормальної роботи контролера.

GSM-модем, інтегрований в цей контролер, дозволяє виконувати функції:

- відправка і отримання SMS;
- віддалене керування контролером по протоколах GPRS (General Packet Radio Service, пакетний радіозв'язок загального користування) або CSD (Circuit Switched Data, передача даних при комутації каналів);
- постійне підключення по GPRS для роботи з цим протоколом без виконання програм.

З'єднання TCP/IP в призначеному для користувача проекті повинне використати відповідні функціональні блоки. PC Worx має вбудовані комунікаційні функціональні блоки MOBILE_CONNECT, SMS_SEND і SMS_RECEIVE, щоб реалізувати зв'язок шляхом передачі SMS по мережі GSM з використанням контролера ILC151GSM/GPRS.

PC Worx має стандартний функціональний блок GPRS_CONNECT для встановлення GPRS-з'єднання з контролером ILC 151GSM/GPRS. Блоки TCP/IP дозволяють організувати передачу даних по GPRS-з'єднанню засобами протоколу TCP/IP.

Контролер ILC 151GSM/GPRS має тільки два аналогові виходи.

Для розширення можливостей контролера можуть бути використані наступні додаткові апаратні модулі.

Модуль IBILDO 4 – ME (рис. 1.7) має 4 цифрові виходи. У маркіровці модуля використані скорочення: IB – InterBus, IL – InLine, DO – Digital Output (цифровий вихід), ME – Machine Edition (розроблено для завдань машинобудування, позиціонування і керування переміщенням).

Модуль має 4 виходи 24 В, 500 мА постійного струму. Він розроблений для виведення цифрових сигналів і має наступні характеристики:

- підключення для 4 цифрових виконавчих механізмів;
- підключення виконавчих механізмів по двох- і трипровідних технологіях;
- номінальний вихідний струм 500 мА;
- загальний струм модуля 2 А;
- виходи захищені від короткого замикання і перевантаження;
- є діагностичний і статусний індикатор.

Модуль IBILA14/U - PAC (рис. 1.8) має 4 аналогові входи. У маркіровці модуля використані скорочення: IB – InterBus, IL – InLine, AI – Analogue Input (аналоговий вхід).

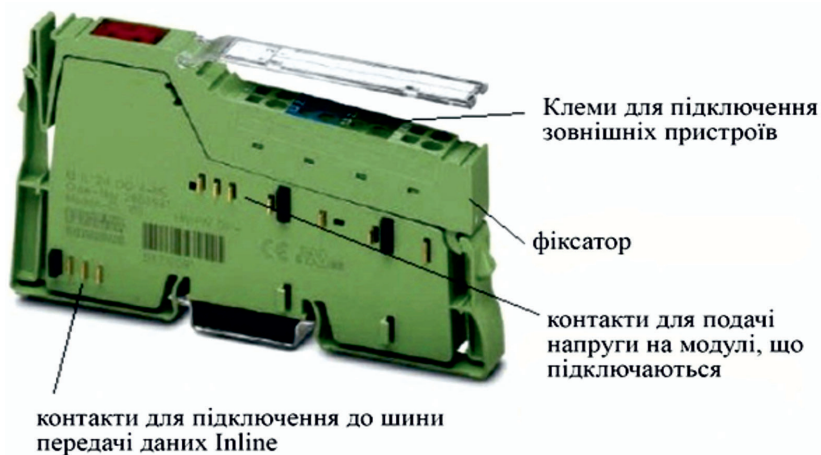


Рис. 1.7 – Зовнішній вигляд модуля IBILDO 4 - ME шини Interbus контролера ILC151GSM/GPRS

Модуль має 4 входи для підключення джерел з напругою, що міняється, або струмом. Його характеристики:

- 4 аналогових, біполярних вхідних каналів;
- підключення датчиків за двопровідною технологією;
- діапазони напруги від 0 до +10 В і від – 10 В до +10 В;
- генерація середньої величини значень на входах;
- оновлення даних на входах кожні 250 мс;

- діагностичні і статусні індикатори.

Модуль IBILA02/UI - PAC (рис. 1.9) має 2 виходи для виведення аналогових сигналів із струмами, що змінюються, і напругою. Його характеристики:

- підключення виконавчих механізмів за двопровідною схемою;

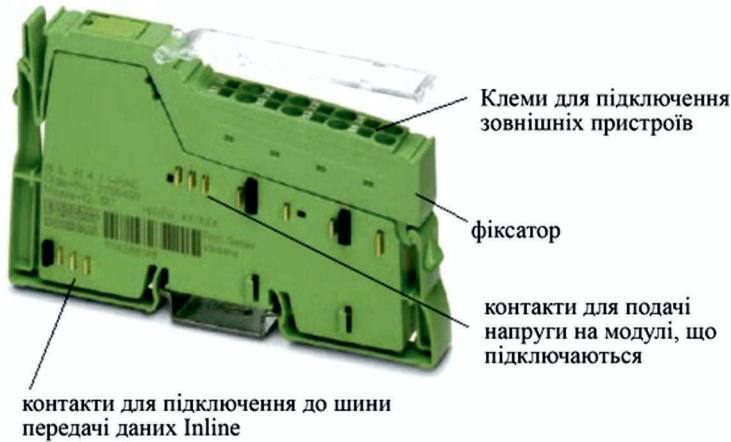


Рис. 1.8 – Зовнішній вигляд модуля IBILA02/UI - PAC шини Interbus контролера ILC151 GSM/GPRS

- діапазони зміни струмів від 0 до +20 мА, від +4 до +20 мА і від – 20 до +20 мА;
- діапазони зміни напруги від 0 до +10 В і від – 10 до +10 В;
- діагностичні і статусні індикатори.

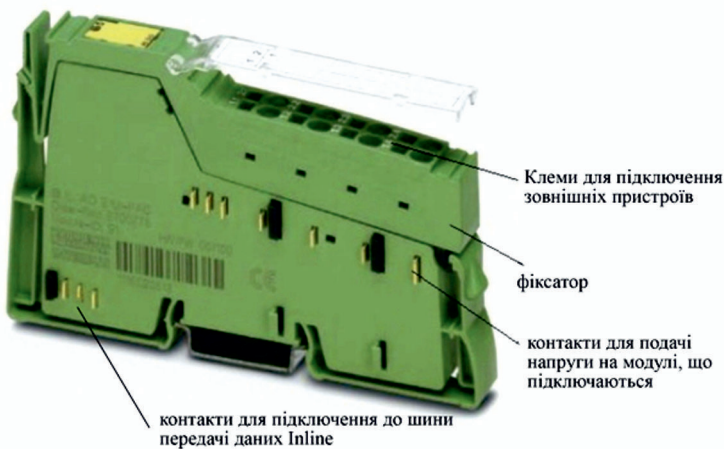


Рис. 1.9 – Зовнішній вигляд модуля IBILA02/UI - PAC шини Interbus контролера ILC151 GSM/GPRS

Шинний з'єднувач AXLFBKPN AxiolineF (рис. 1.10) об'єднує мережу PROFINET з системою AxiolineF. З його допомогою можна підключити до 63 пристроїв AxiolineF до існуючої мережі PROFINET. Його характеристики:

- 2 порту Ethernet і інтегрований комутатор;
- типовий час циклу для локальної шини AxiolineF 10 мкс;
- підтримка PROFINET RT (Real Time, реальний час) і IRT (Isochronous Real Time, ізохронний реальний час);
- підтримка стандарту PROFIsafe (розширення технологій Profibus і PROFINET, за допомогою якого можна створювати вільно програмовані функції безпеки і обмінюватися необхідними для цього входними і вихідними даними з безпечними пристроями входів-виходів);
- мінімальний час циклу PROFINET для RT і IRT складає 250 мкс;
- часом виконання в шинному з'єднувачі можна знехтувати;
- прошивка програмного забезпечення (firmware) може бути оновлена;
- є MRP-клієнт (Media Redundancy Protocol, протокол забезпечення надмірності середовища передачі даних), що забезпечує надійну роботу мережі незалежно від топології і навіть при виході з ладу деяких кабелів;
- керування налаштуваннями виконується у браузері;
- функції I & M (Identification and Maintenance, ідентифікація і обслуговування);
- доступ різних користувачів до пристрою по мережі;
- наявність діагностичних і статусних індикаторів.

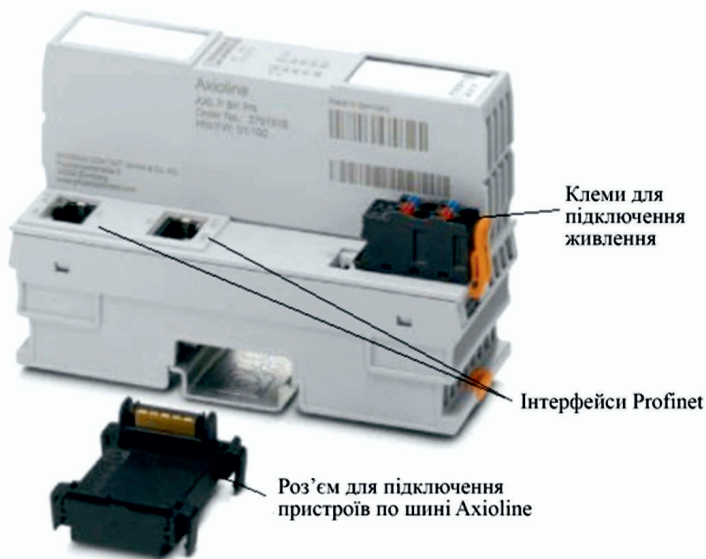


Рис. 1.10 – Шинний з'єднувач AXLFBKPN AxiolineF для PROFINET

Версія PC Worx 6.30 дозволяє налаштовувати пристрої PROFINET і містить доступний по мережі опис функцій пристроїв з технічними даними і конфігураційним файлом (якщо доступні декілька версій конфігураційних файлів, треба переконатися в тому, що версія файлу відповідає версії апаратного забезпечення і програмної прошивки).

Сучасні модулі входів-виходів виконують багато функцій, які раніше виконувалися тільки контролерами. Для виконання цих функцій пристрої мають бути відповідним чином налаштовані при інсталяції системи, обслуговуванні і параметризації. Тому необхідно мати точні і повні відомості про пристрої: тип виконуваних функцій, кількість входів/виходів, діапазон зміни змінних, одиниці виміру, значення за умовчанням, пристрої, що ідентифікують параметри, і т. д. Profibus пропонує декілька підходів для уніфікованого опису пристроїв. Найпростіший – застосування текстових файлів GSD, що містять загальну і специфічну для конкретного пристрою інформацію. GSD-файл завантажується в засіб конфігурації системи Profibus Configurator і використовується при її інсталяції.

Пристрій AXLFBKPN Axioline F має веб-сервер, який створює необхідні сторінки для веб-орієнтованого керування і при необхідності завантажує їх у браузер. Веб-орієнтоване керування можна використати для доступу до статичної інформації (технічні дані, MAC-адреси) або інформації (IP -адреси, статусна інформація), що динамічно міняється.

Доступ до веб-сервера пристрою можна отримати за допомогою IP-адреси, якщо він відповідним чином конфігурований. Домашня (основна) веб-сторінка пристрою завантажується після введення адреси у форматі: <http://IPадреса> (наприклад, <http://192.36.133.58>).

Шинний з'єднувач підтримує протокол SNMP (Simple Network Management Protocol, простий протокол мережевого керування) для керування пристроями в IP-мережах, що використовують протоколи TCP і UDP. SNMP підтримують маршрутизатори, комутатори, сервери, робочі станції, принтери, модемні стійки та ін.

Для розширення можливостей контролера АХС 3050 при побудові складних систем автоматизації до шинного з'єднувача AXLF BKPN AxiolineF можуть бути підключені додаткові модулі. Модулі, що входять в комплект TSL, описані нижче.

Модуль AXLFDI8/DO8/1 IH (рис. 1.11) використовується для цифрового введення і виводу, містить 8 входів (24 В постійного струму) і 8 виходів (24 В постійного струму), максимальний струм 500 мА, технологію підключення по одному провідникові.

Модуль розроблений для роботи у складі станції Axioline F і використовується для отримання і виведення цифрових сигналів. Тимчасові фільтри на входах можуть бути налагоджені так, щоб збільшити несприйнятність до

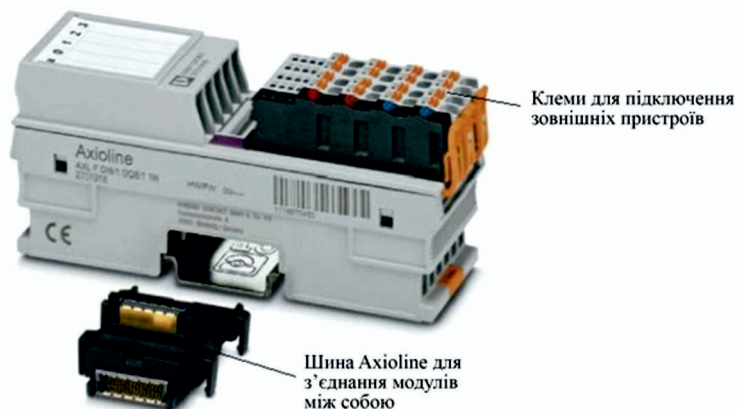


Рис. 1.11 – Зовнішній вигляд модуля AXLFDI8/1DO8/1 IH

шуму. Часовий фільтр в 100 мкс дозволяє реалізувати в додатку функцію лічильника з максимальною частотою в 5 кГц. Виходи захищені від короткого замикання і перевантаження.

Характеристики підсистеми введення :

- 8 вхідних сигналів;
- 24 В, 2,4 мА постійного струму;
- підключення датчиків за однопровідною технологією;
- тимчасові фільтри можуть бути налаштовані на три можливі величини приросту менше 100 мкс, 1000 мкс і 3000 мкс;
- максимальна частота входів 5 кГц.

Характеристики підсистеми виводу:

- 8 вихідних сигналів, 24 В, 500 мА постійного струму;
- підключення виконавчих механізмів за однопровідною схемою.

Мінімальний час оновлення модуля менше 100 мкс. Є діагностичні і статусні індикатори.

Модуль аналогового введення AXLFAI4U1H призначений для застосування усередині станції Axioline F, служить для введення аналогових сигналів напруги і має інтегрований блок живлення датчиків. Характеристики модуля:

- 4 аналогових біполярних каналу введення для підключення сигналів напруги;
- двох-, трьох- і чотирипровідна схема підключення датчиків;
- діапазони вхідної напруги від 0 до 10 В, від – 10 до +10 В, від 0 до 5 В і від –5 до +5 В;
- одночасне опитування усіх каналів за допомогою функції одночасної вибірки;
- високий коефіцієнт перехідного згасання між каналами завдяки роздільним сигнальним ланцюгам;
- висока стійкість до перешкод, електромагнітним випромінюванням, що викликається;

- є індикатори стану і діагностики.

Модуль аналогового виведення AXLFA041H призначений для застосування усередині станції Axioline F. Він служить для виведення аналогових сигналів напруги і струму. Характеристики модуля:

- 4 аналогових вихідних канала для підключення сигналів напруги або струму;
- біполярні виходи напруги, однополярні виходи струму;
- двопровідна схема підключення виконавчих елементів;
- діапазони напруги від 0 до 10 В, від -10 до +10 В, від 0 до 5 В і від -5 до +5 В;
- діапазони струму від 0 до 20 мА і від 4 до 20 мА;
- виходи захищені від коротких замикань;
- збереження специфікації у власній пам'яті пристрою;
- наявність індикаторів стану і діагностики.

Siemens ET 200S – периферійний пристрій введення/виводу, працюючий по протоколу PROFINET IO. Він складається з інтерфейсного модуля IM151-3PN, модуля живлення, модуля цифрового введення, двох модулів цифрового виводу, вставлених в термінальний модуль.

Розподілена система введення/виведення ET 200S – модульна, гнучка в налаштуванні система DP-slave для підключення до обробки сигналів на центральному контролері або в мережі PROFINET IO. ET 200S має клас захисту IP 20.

Інтерфейсний модуль IM151-3 PN (рис. 1.12) з інтегрованим двопортовим комутатором підключає ET 200S до мережі PROFINET IO. Ім'я пристрою і резервна копія даних можуть зберігатися на флеш-картці. Підтримує сервіси Ethernet, такі як ping, arp (дві стандартні утиліти командного рядка операційної системи Windows), Net diagnostics (SNMP)/ MIB-2, LLDP (Link Layer Discovery Protocol) і має вбудовані світлодіоди для діагностики стану пристрою. Топологія мережі, в якій використовується LLDP, може бути отримана з комп'ютера, що управляє, послідовним обходом і опитуванням кожного пристрою.

До IM151-3 PN можна підключити максимум 63 модулів. Можлива організація зв'язку в ізохронному режимі реального часу (IRT). Мінімальний час актуалізації складає 250 мкс в режимі IRT. Максимальна довжина з'єднання по шині – 2 м.

Інтерфейсний модуль IM151-PN HF дозволяє оновлювати прошивки фірмового ПО з використанням флеш-картки або по шині PROFINET IO. Під час роботи допускається використання наступних переривань:



Рис. 1.12 – Шинний з'єднувач Siemens IM151 - 3 PN HF PROFINET

- діагностичні;
- робочого процесу;
- для додавання/видалення модуля;
- для технічного обслуговування.

Максимальний адресний простір для даних введення/виводу складає 256 байт.

Модуль підключається до шини із задньої панелі, при цьому максимальна довжина шини – 2 м.

Термінальні модулі TM-P15S23-A1 і TM-E15S26-A1 (рис. 1.13) реалізують електричний і механічний зв'язок модулів введення/виводу з інтерфейсним модулем і кінцевим модулем.

Встановлений модуль введення/виводу виводить сигнали на клеми 1–16, A3, A4, A7, A8, A11, A12, A15, A16.

Необхідний термінальний модуль вибирається залежно від напруги, необхідної для додатка. У термінальні модулі інтегрована допоміжна шина AUX1 (на неї можна подавати допоміжну напругу живлення до ~230 В або використати її в якості шини захисного заземлення PE). Можна використати допоміжну шину як направляючу захисну планку або для додаткової напруги.

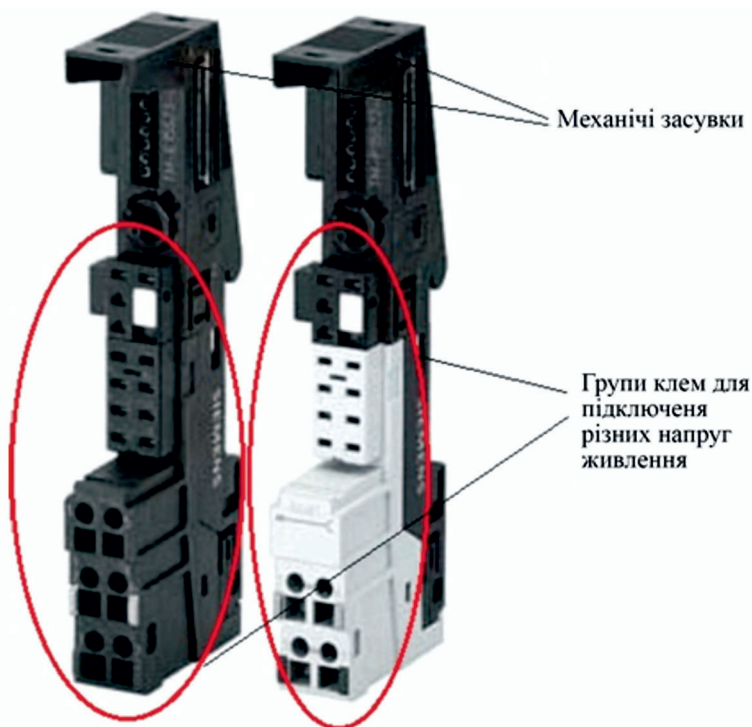


Рис. 1.13 – Зовнішній вигляд термінальних модулів ТМ - P15S23 - A1 і ТМ - E15S26 - A1

Характеристики ТМ - P15S23 - A1:

- подання нової групи напруги для наступного термінального модуля ТМ - Р;
- поставляється в трьох варіантах
- пригвинчення гвинтами, утримування за допомогою пружин і швидке підключення «fast connect» без демонтажу;
- суцільна надійна шина AUX1 з електричним підключенням до наступної групи напруги ліворуч;
- доступ до напруги шини AUX1 на клеммах А4 і А8.

Характеристики ТМ - E15S26 - A1:

- універсальний термінальний модуль для усіх електронних модулів, що мають ширину 15 мм;
- поставляється в трьох варіантах;
- пригвинчення гвинтами, утримування за допомогою пружин і швидке підключення «fast connect» без демонтажу;
- суцільна надійна шина AUX1 з електричним підключенням до наступної групи напруги ліворуч;
- доступ до напруги шини AUX1 на клеммах А4, А8 і А3, А7.

Розподілений **модуль введення/виведення живлення ET 200S PM - E DC24V** контролює напругу живлення для усіх електронних модулів. Напруга живлення подається засобами термінального модуля ТМ - Р.

Можна використати електронні модулі, виключаючи 2DI AC120V ST, 2DI AC230V ST, 2DO AC24.230V/1A, в групі напруги модуля подання живлення РМ - E DC 24V. Поточний статус модуля подання живлення зберігається в статусному байті в спеціальній області пам'яті (Process Image of the Inputs, PI), бо прямий доступ до модулів введення/виводу неможливий. Він оновлюється незалежно або тоді, коли включена діагностика відсутності напруги. Модуль подання живлення РМ - E DC 24V може бути використаний для модулів, що самовідключаються (захищених від збоїв). При вертикальній установці забезпечується розширений температурний діапазон: 0 – 55 °С.

Характеристики модулів, підключених до ET 200S PM - E DC24V, перераховані нижче.

Модуль 8DI DC 24V має 8 цифрових входів; номінальна вхідна напруга складає 24 В постійного струму; дозволяє підключати двопровідні датчики; підтримуються ізохронні операції.

Модуль 8 DO DC 24V/0,5A має 8 цифрових виходів; вихідний струм складає 0,5 А на кожен вихід (канал), загальний струм 4 А; номінальна напруга навантаження складає 24 В постійного струму; вбудований захист від короткого замикання; підтримуються ізохронні операції. Модуль сумісний з соленоїдними клапанами, контакторами постійного струму, системами індикації.

Модуль 2 AI STANDARD U має 2 виходи для виміру напруги з наступними вхідними діапазонами: ± 10 В з роздільною здатністю 13 біт + знаковий біт, ± 5 В з роздільною здатністю 13 біт + знаковий біт, 1 – 5 В з роздільною здатністю 13 біт. Виконана ізоляція від напруги навантаження L+. Допускається напруга до 5 В змінного струму, м'який пуск (Soft Start). Для вертикальної установки температурний діапазон складає 0 – 50 °С.

Модуль 2 AO U має 2 виходи для виведення напруги з вихідним діапазоном ± 10 В з розрядністю 13 біт + знак, 1 – 5 В з розрядністю 12 біт. Виконана ізоляція від напруги навантаження L+. Для вертикальної установки температурний діапазон складає 0 – 50 °С.

Комутатор FL SMCS 8TX - PN (Smart Managed Compact Switch) – інтелектуальний керований компактний комутатор стандарту промисловий Ethernet (рис. 1.15). Пристрій не використовує вентилятор, має зменшене енергоспоживання, відповідає усім промисловим стандартам в термінах електромагнітної сумісності, температурних режимів, механічного навантаження, гарантуючи максимально можливий рівень доступності. Надмірність топології мережі може бути створена за допомогою протоколів STP (Spanning Tree Protocol, протокол побудови основного дерева), RSTP (Rapid STP, модернізована, прискорена реалізація протоколу STP), MRP. На пристрої є спеціальні наклейки для написання коментарів до кожного порту.

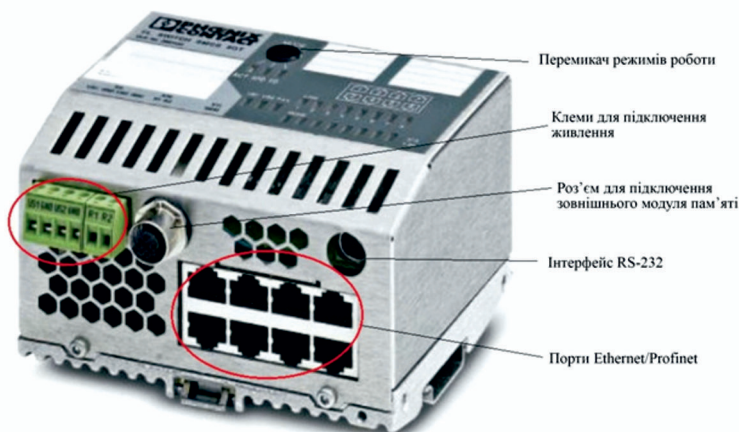


Рис. 1.14 – Комутатор FL SMCS 8TX - PN 26

Веб-сервер і SNMP-агент передбачені для діагностики, обслуговування і конфігурації по мережі. Термінальна точка доступу може бути використана для передачі даних на невеликі відстані.

Зеркалювання портів може бути використане для відстежування трафіку на мережевих з'єднаннях або як важлива сервісна функція.

Особливості і сфери застосування комутатора:

- максимальна продуктивність усіх портів до 1 Гбіт/с;
- збільшена продуктивність мережі шляхом фільтрування трафіку (локальний трафік не передається в зовнішню мережу і зменшується об'єм даних в мережевих сегментах);
- нескладне розширення мережі і її конфігурація;
- об'єднання кабельних сегментів мережі з різними швидкостями передачі даних;
- автовизначення швидкостей передачі даних в 10, 100 або 1000 Мбіт/с з автокресуванням портів RJ45;
- гнучке використання волоконно-оптичних модулів в слотах SFP (Small Form - factor Pluggable – промисловий стандарт модульних компактних приймачів (трансиверів) для передачі даних). Модулі SFP використовуються для підключення плати мережевого пристрою (комутатора, маршрутизатора) до оптичного волокна або неекранованої витієї пари;
- підвищена доступність у зв'язку з використанням надмірних шляхів передачі даних з мінімальним часом перемикання і використанням протоколу RSTP, а також визначенням режиму швидких оновлень fast ring (швидке визначення наявності кільцевої топології, розширення протоколу RSTP);
- підтримка різних топологій і комірчастих структур, а також кільцевих топологій з екстремним визначенням наявності кільця;
- конфігурація комутатора у веб-системі керування по протоколу SNMP або локально по інтерфейсу RS232;

- зеркалювання портів;
- визначення топології і використання протоколу LLDP;
- розподіл адрес з використанням протоколів BootP (Bootstrap Protocol – мережевий протокол для автоматичного отримання клієнтом IP-адреса), DCP (Discovery and Configuration Protocol, використовується для конфігурації імен пристроїв і IP-адрес в невеликих і середніх застосуваннях без установки DHCP-сервера в мережах PROFINET) або статично;
- підтримка протоколу MRP в якості клієнта (таким чином кільце MRP може бути створене з використанням будь-яких портів комутатора);
- комутатор може бути використаний в середовищі PROFINET;
- налаштування можуть бути легко змінені шляхом використання інтелектуального (Smart) режиму.

Бездротова точка доступу FL WLAN EPA (EPA – Ethernet Port Adapter – адаптер портів Ethernet, WLAN – Wireless Local Area Network – бездротова локальна мережа) є високопродуктивним інтерфейсом між кабельними і бездротовими мережами для мереж Ethernet і Profinet. Прозорий протокол використовується для передачі даних на другий рівень (L2) моделі OSI (Open System Interconnection, семирівнева модель взаємодії відкритих систем, розроблена для стандартизації пристроїв і засобів передачі цих різних виробників), що забезпечує просту інтеграцію і з мережами, які базуються на промисловому Ethernet (Profinet і Modbus/TCP). FL WLAN EPA відповідає вимогам Profinet по відповідності класу A і має профайл PROFI-safe для виключення збоїв при передачі даних.

FL WLAN EPA сертифікована на сумісність з бездротовими локальними мережами стандарту IEEE 802.11 b/g/n, працюючими на частоті 2,4 ГГц. За цим стандартом працюють стаціонарні комп'ютери і ноутбуки, мобільні ношені пристрої, сканери штрих-кодів, зчитувачі міток RFID, системи для зважування.

WLAN (IEEE 802.11 b/g/n/a) – стандартна бездротова технологія, що забезпечує надійну передачу даних в просторі з металевими предметами, і в середовищі з високим рівнем перешкод. В мережах автоматизації WLAN став стандартом бездротової передачі даних, які управляють. Бездротові локальні мережі серії Factoryline оптимізовані у рамках стандарту для автоматизації виробництв.

1.2. Апаратний модуль «PROFIBUS»

Другий апаратний модуль містить три Profibus-пристрої введення/виводу, Profinet-проксі для введення/виводу для технології Profibus DP з інтегрованим комутатором і бездротовою точкою доступу (рис. 1.15).



Рис. 1.15 – Другий апаратний модуль TSL «Profibus»

Цей модуль може бути використаний для вивчення технології Profibus. Його можна задіяти в комбінації з першим апаратним модулем або безпосередньо з контролером, працюючим по технологіях Profinet або Profibus. На рис. 1.16 показана схема з'єднання пристроїв в другому апаратному модулі.

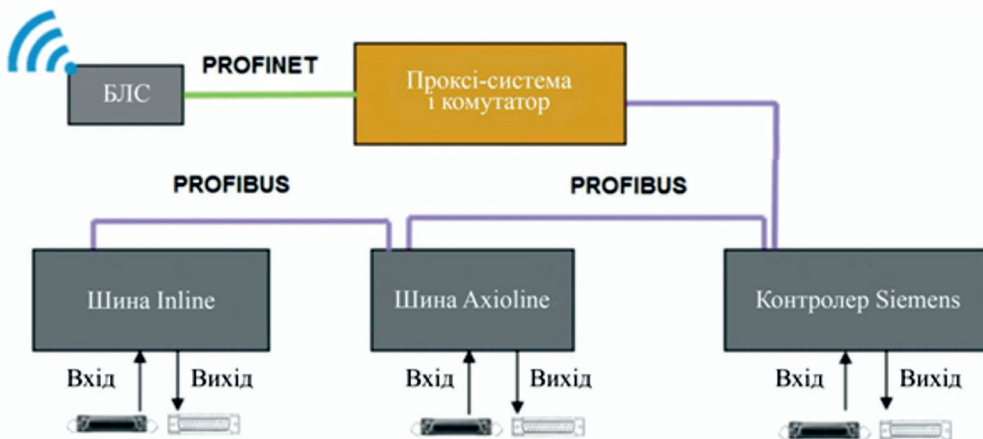


Рис. 1.16 – Принципіальна схема підключення TSL Box 2 - PROFIBUS

У другому апаратному модулі встановлені наступні пристрої.

Проксі-система FL NP PND - 4TX PB зі вбудованим комутатором на 4 порти RJ - 45 (рис.1.17) має наступні характеристики:

- підтримка пристроїв введення/виведення PROFINET;
- інтерфейс Ethernet 10/100Base - T(X) для витой пари;
- вбудований PROFINET IO проксі для PROFIBUS;
- підтримка узгодження з PROFINET класу B;
- вбудований керований чотирьохпортовий комутатор;
- майстер-пристрій Profibus DP класу 1;
- Profibus DP майстер-підключення зі швидкістю до 12 Мбіт/с (підключення по мідному кабелю і стандарту RS - 485);
- повна конфігурація з використанням PC Worx;
- підключення системи Profibus DP до контролера PROFINET IO;
- використання в невеликих пристроях керування для простої інтеграції існуючих рішень Profibus DP в мережі PROFINET.

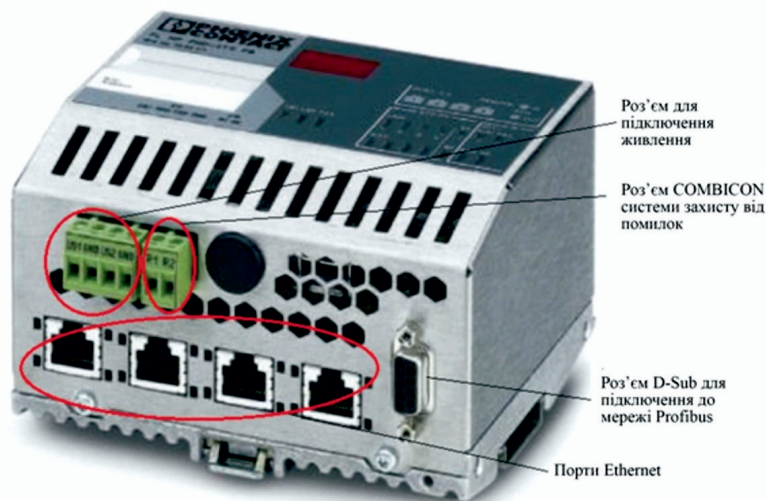


Рис. 1.17 – Проксі-система FL NP PND

Проксі-система Profibus DP може бути конфігурована за допомогою PC Worx.

Пам'ять, що параметризується, спрощує заміну пристроїв, якщо контролер PROFINET IO не підтримує автоматичне визначення пристроїв, засноване на топології. Зеркалювання портів можна використати для аналізу пакетів.

У другому апаратному модулі встановлена бездротова точка доступу FL WLAN EPA, така ж як і в першому.

Шинний з'єднувач AXL F BK PB AxiolineF зв'язує мережу PROFIBUS і шину Axioline F. З його допомогою можна підключити до 63 пристроїв Axioline F в існуючу мережу PROFIBUS.

Основні характеристики шинного з'єднувача:

- підключення виконується за допомогою дев'ятиконтактних роз'ємів D - SUB Female використовується фізичний інтерфейс RS - 485;
- при роботі майстер-пристроїв класів 1 і 2 використовується режим Profibus DP - V1 (підстандарт DP - V0 використовується для циклічного обміну даними і діагностики; DP - V1 використовується для ациклічного і циклічного обміну даними і обробки сигналів тривоги; DP - V2 використовується для синхронізації режиму і обміну даними між підпорядкованими одиницями устаткування);
- виконується автовизначення швидкості передачі даних в діапазоні 9,6 кбіт/с – 12 Мбіт/с;
- застосовуються поворотні кодуючі перемикачі для установки адрес PROFIBUS;
- реалізується підтримка динамічної конфігурації;
- виконана підтримка адрес PROFIBUS в діапазоні 0 – 126;
- опис пристрою виконаний у файлі GSD;
- функції I & M (Identification and Maintenance, ідентифікація і обслуговування).

Шинний з'єднувач AxiolineF для Profibus DP має ті ж модулі, що підключаються, що і шинний з'єднувач AXLFBK PN для Profinet в першому апаратному модулі.

Шинний з'єднувач IL PB BK DI8 DO4/EF - PAC для Profibus DP має 8 цифрових входів і 4 цифрові виходи. Він потрібний для зв'язування пристроїв з інстальованою системою Inline. Крім того, він використовується для збору і виведення цифрових сигналів. Шинний з'єднувач дозволяє підключити 61 Inline-пристрій до будь-якої точки існуючої мережі Profibus DP. Шинний з'єднувач і пристрої серії Inline формують одну станцію з максимальним числом в 63 пристрої, підключених до локальної шини. Вхідні і вихідні пристрої шинного з'єднувача є першими і другими пристроями локальної шини.

Характеристики шинного з'єднувача:

- підключення до PROFIBUS виконується за допомогою дев'ятиконтактних роз'ємів D - SUB Female;
- застосовується фізичний інтерфейс RS - 485 для PROFIBUS;
- можливе підключення до 61 додаткового Inline-пристрою;
- можливе підключення до 16 пристроїв PCP (Port Control Protocol – мережевий протокол, який в IP-мережах управляє перенаправленням пакетів даних, що приходять, в маршрутизаторі. Маршрутизатор фільтрує пакети даних або працює відповідно до механізму NAT (Network Address Translation, перетворення мережевих адрес), що дозволяє в IP-мережах перетворювати IP-адреса транзитних пакетів). Це дозволяє спростити керування мережевим трафіком і конфігурацією, щоб системи, розміщені за пристроями NAT або мережевими фільтрами (frewalls), були

- досяжні з Інтернет (тобто вони можуть також працювати як сервери), що необхідно для багатьох додатків);
- можливе застосування майстер-пристрій Profibus DP/V1 класів 1 і 2;
- виконується автовизначення швидкості передачі даних в діапазоні 9,6 кбіт/с – 12 Мбіт/с;
- застосовані поворотні кодуючі перемикачі для установки адрес PROFIBUS;
- реалізована підтримка адрес PROFIBUS в діапазоні 0 – 126;
- опис пристрою приведений у файлі GSD;
- реалізовані функції ідентифікації і обслуговування I & M;
- є 8 цифрових входів, 4 цифрові виходи;
- автовизначення швидкості роботи локальної шини (500 кбіт/с або 2 Мбіт/с);
- виклик IO - Link версії програмної прошивки 2.0 і вище (IO - Link – перша стандартизована технологія введення/виводу, що дозволяє обмінюватися даними на усіх рівнях, від системи керування до найнижчого рівня автоматизації).

Пристрій схвалений для використання в зоні двох потенційно вибухонебезпечних зон. Шинний з'єднувач PROFIBUS конфігурує станцію і управляє обміном даних з мастер-пристроєм PROFIBUS. Він також підтримує подання живлення підключеним терміналам Inline. Можуть бути також використані модулі PROFIsafe I/O.

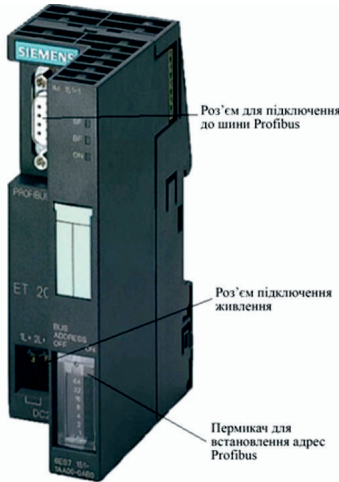
Шинний з'єднувач може працювати з тими ж модулями, що підключаються, що і контролер ILC 151GSM/GPRS.

Шинний з'єднувач IL PB BK DI8 DO4/EF - PAC Inline має тільки 2 аналогові виходи, які підключені до роз'єму.

Інтерфейсний модуль IM151 - 1 HIGH FEATURE (рис. 1.18) має характеристики:

- сполучає периферійний пристрій введення/виведення ET200S з мережею Profibus DP по інтерфейсу RS 485;
- ліквідовані обмеження Simatic 7 для максимальної довжини параметра (звичайно це 244 біта);
- максимальний адресний простір – 244 байти для входів і 244 байти для виходів;
- робота в якості веденого пристрою DPV0 або DPV1;
- максимальне число обслуговуваних модулів – 63;
- максимальна довжина шини – 2 м;
- підтримка опції обслуговування і наявність статусного байта для силових модулів;
- може синхронізуватися з циклами Profibus DP;
- фірмове ПО може бути оновлено по Profibus DP з використанням утиліти Siemens Simatic Step 7 HW Config;

- обмін даними виконується відповідно до технології Safetyrelated I-slave-I-slave компанії Siemens по Profibus DP;
- робота в якості DPV1 за технологією Y-switching;
- використання модулів ProfSafe.



**Рис. 1.18 – Інтерфейсний модуль
IM151 - 1 HIGH FEATURE**

Цей інтерфейсний модуль працює з тими ж модулями, що підключаються, що і розглянутий в п. 1.2 шинний з'єднувач Siemens Profinet IM151 - 3 PN HF.

1.3 Апаратний модуль «Моделювання процесів»

Третій апаратний модуль (рис. 1.19) призначений для моделювання технологічних процесів. Моделі повинні розроблятися в CODESYS версій 3.x і завантажуватися в контролер EtherCAT EC2250. Графічний інтерфейс можна побачити у браузері, ввівши посилання

<http://xxx.xxx.xxx.xxx: 8080/webvisu.htm>,

де xxx.xxx.xxx.xxx – IP-адреса контролера EtherCAT EC2250.

Для візуалізації можна також використати вбудований графічний термінал Ethernet ET1007 WT. Цей модуль також дозволяє вивчати програмування контролерів в CODESYS версій 3.x.

Третій апаратний модуль має різні аналогові і цифрові входи і виходи, а також кнопки, які можна використати для тестування і імітації роботи системи керування високої складності. Аналогові входи/виходи розташовані ліворуч і підключені до роз'єму X7. Кожна кнопка має вбудований світлодіод, підключений до цифрового виходу. Кнопки без фіксації в рядах SA і SB підключені до роз'єму X3. Кнопки в рядах SC і SD – звичайні кнопки з фіксованим станом, підключені до роз'єму X4. Таким чином, реалізовані в цьому модулі пристрої і технології дозволяють вивчити матеріал учбових модулів 1, 4 і 5.



Рис. 1.19 – Третій апаратний модуль TSL «Моделювання процесів»

На рис.1.20 показана схема з'єднання пристроїв в третьому апаратному модулі.

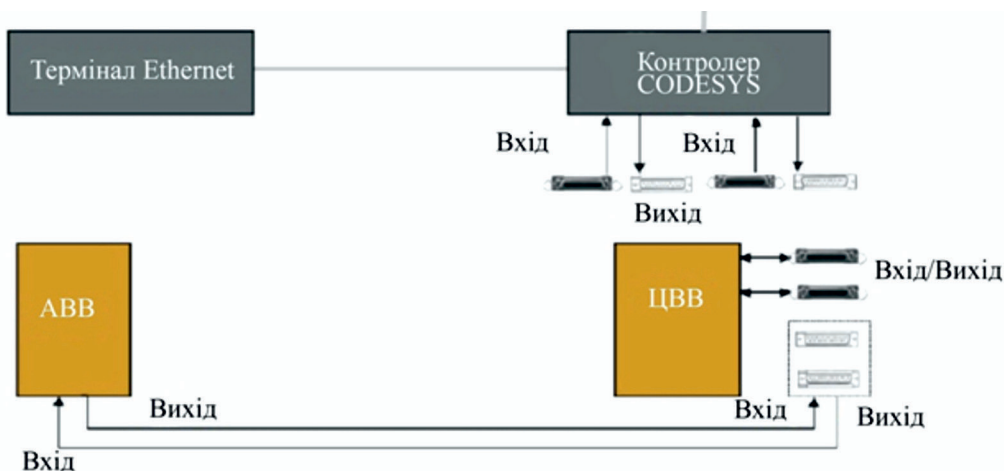


Рис. 1.20 – Схема з'єднання пристроїв в третьому апаратному модулі TSL «Моделювання процесів»

У третьому апаратному модулі встановлені наступні пристрої.

Компактний контролер EC2250 EtherCAT (рис.1.29) призначений для систем жорсткого реального часу і відрізняється малим часом виконання циклу. У контролері використаний високопродуктивний масштабований процесор ARM з ядром Cortex TM-A9 і частотою 800 МГц.

EtherCAT-шина на базі Ethernet, стандартизована асоціаціями SEMI (Semiconductor Equipment and Materials International, міжнародна асоціація по виробництву напівпровідникового устаткування і матеріалів), IEC, ISO. Принцип роботи EtherCAT істотно відрізняється від інших рішень на базі Ethernet. У мережі EtherCAT єдиний пакет Ethernet включає вхідні і вихідні дані багатьох пристроїв. Реальне використання смуги пропускання може досягати більше 90 %. EtherCAT значно швидше за традиційні шини і промислові рішення, що базуються на Ethernet. Типовий час циклу EtherCAT складає 50...250 мкс, тоді як в традиційних шинах на кожне оновлення потрібно 5...15 мс.

Безліч цифрових і аналогових входів і виходів розміщені в самому контролері, як і вбудовані мови програмування для роботи з CODESYS версії 3. Спільно з пакетом CODESYS SoftMotion можуть експлуатуватися складні додатки для керування багатоосьовими приводами. У пристрій інтегровані комунікаційні інтерфейси Ethernet, EtherCAT, CAN, RS 232, RS 485. Доступні протоколи Profinet, BACnet і Modbus. Для візуалізації EC2250 використовує CODESYS, Web Visu і відповідні термінали Ethernet.

CODESYS SoftMotion – вбудований в середовище програмування і систему виконання CODESYS функціональний набір засобів керування рухом від простих переміщень по одній осі до складної багатовимірної інтерполяції сучасних систем з числовим програмним керуванням.

Контролер EC2250 має 32 цифрові входи/виходу і 18 аналогових входів/виходів і дозволяє:

- виконувати програмування, візуалізацію, зв'язок, керування рухом в CODESYS і SoftMotion;
- використати майстер-режими в технологіях EtherCAT і CANopen;
- використати послідовні інтерфейси передачі даних;
- використати карти розширення.

У третьому апаратному модулі контролер EC2250 призначений головним чином для моделювання процесів.

Ethernet-терминал ET1007 WT розроблений спеціально для візуалізації в CODESYS, має просте керування і високу швидкість роботи. Може працювати з різним устаткуванням, виконувати веб-візуалізацію CODESYS або target-візуалізацію (цільову візуалізацію) CODESYS по протоколу VNC (Virtual Network Computing, віртуальні мережеві обчислення, – система віддаленого доступу до ресурсів (наприклад, до робочого столу) комп'ютера, що використовує протокол RFB (Remote Frame Buffer, віддалений кадровий буфер)).

Не має значення, приходять дані для візуалізації від контролера компанії Berghof або від будь-якого іншого контролера, працюючого з CODESYS. Продуктивність і об'єм пам'яті терміналу дозволяють швидко міняти сторінки проекту візуалізації.

Термінал отримує дані по інтерфейсу Ethernet зі швидкостями 10/100 Мбіт/с. Введення IP-адреса і HTTP-адреса гарантує просте і швидке підключення до контролера CODESYS (серверу візуалізації). Ethernet-термінал візуалізує файл, запущений на контролері EC2250.

Термінал має високу яскравість (більше 400 Кд/м.кв.).

Для роботи цільової візуалізації в налаштуваннях системи має бути активована опція «Target-Visualization», розташована на вкладці «General» системи CODESYS. Якщо це дозволено в цільовому файлі, то ця опція може включатися або відключатися користувачем.

1.4 Комутація апаратних модулів

Для забезпечення фізичних підключень пристроїв апаратних модулів використовується роз'єм Centronics IEEE 488, що має 24 виводи. Кожен набір з 8 цифрових входів і виходів підключений до виводів роз'єму відповідно до вимог IEEE 488.

Також застосований 15-контактний роз'єм D-Sub.

До роз'єму D-Sub підключаються до 6 аналогових сигналів. Аналого-цифрове перетворення виконується з розрядністю 12 біт. Частота вибірки складає 0,5 кГц.

За допомогою стандартних кабелів тільки третій апаратний модуль може бути підключений до інших модулів, бо їх входи і виходи встановлені симетрично. На рис.1.23 показаний приклад використання роз'ємів Centronics для з'єднання апаратних модулів. На рис.1.24 показаний приклад використання роз'ємів D-Sub для з'єднання апаратних модулів.

Апаратні модулі підключаються один до одного за допомогою стандартних кабелів, що знаходяться в комплекті TSL відповідно до схеми, представленій на рис.1.1. Ці кабелі можуть бути використані і для підключення інших зовнішніх пристроїв, що не мають IP-адрес. Пристрої, що мають IP-адресу, підключаються до портів RJ-45 комутаторів Ethernet за допомогою стандартного восьмижильного кабелю типу «вита пара» категорії 5 і вище.

Для відпрацювання завдань у рамках цього навчального посібника використовуються два штатні кабелі, за допомогою яких з'єднується перший і третій апаратні модулі («Програмовані контролери і Profinet» і «Моделювання процесів»).

У першому апаратному модулі (див. рис.1.2) використовуються гнізда, розташовані під контролером ILC 151 або під контролером АХС 3050, а в третьому апаратному модулі (див. рис.1.22) гнізда Х4 і Х7.

Після комутації модулів стануть доступні елементи керування, розташовані в третьому апаратному модулі.

Апаратні модулі 1 і 2 (8 цифрових входів,
8 цифрових виходів)

Апаратний модуль (8 цифрових входів, 8
цифрових виходів)

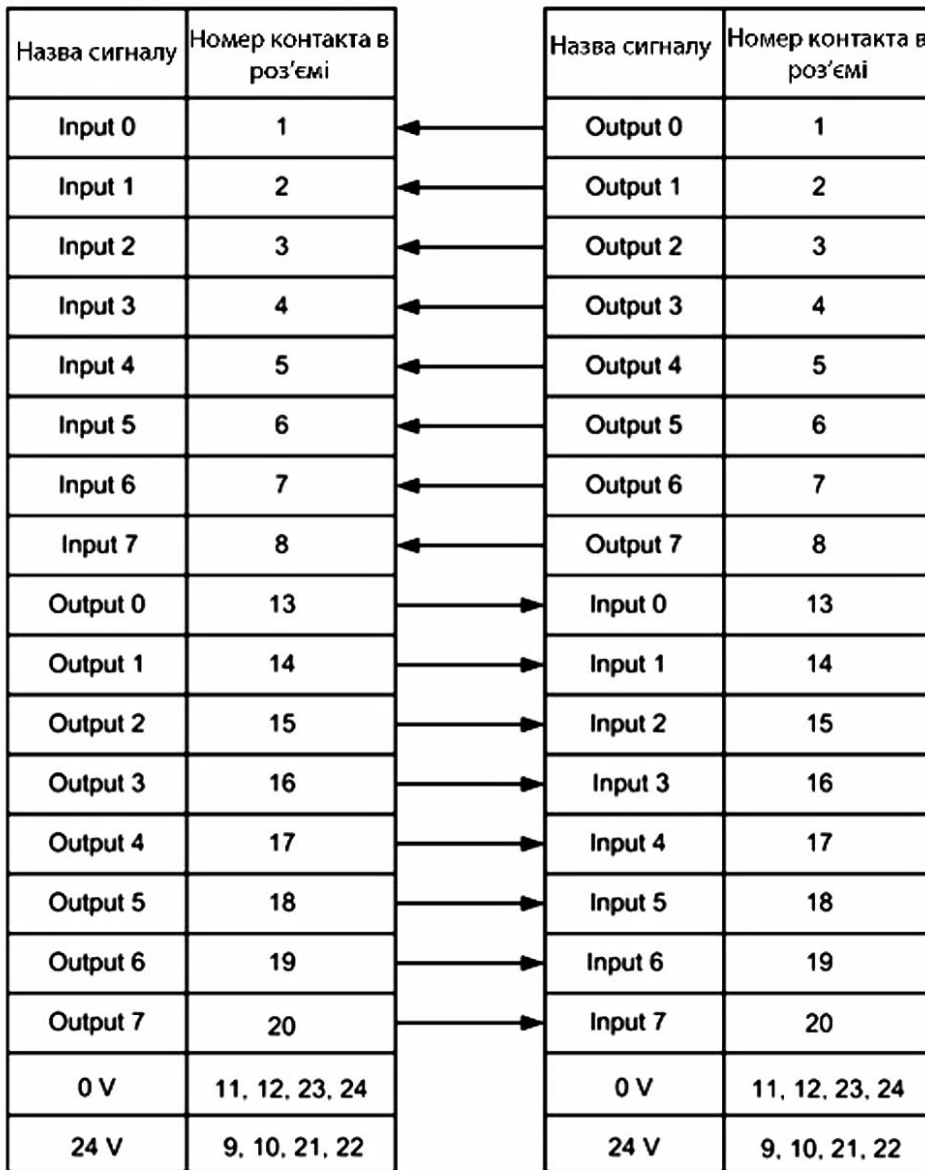


Рис. 1.21 – Цоколювання контактів при з'єднанні апаратних модулів за допомогою роз'ємів Centronics

Апаратні модулі 1 і 2 (2 аналогових входи, 4 аналогових виходи)

Назва сигналу	Номер контакта в роз'ємі
Input 0	1
Input 1	2
0 V	3
Не використовується	4
Не використовується	5
0 V	6
Output 0	7
Output 1	8
Не використовується	9
Не використовується	10
Не використовується	11
Не використовується	12
Не використовується	13
Output 2	14
Output 3	15

Апаратний модуль 3 (2 аналогових входи, 4 аналогових виходи)

Назва сигналу	Номер контакта в роз'ємі
Output 0	1
Output 1	2
0 V	3
Не використовується	4
Не використовується	5
0 V	6
Input 0	7
Input 1	8
Не використовується	9
Не використовується	10
Не використовується	11
Не використовується	12
Не використовується	13
Input 2	14
Input 3	15

Рис. 1.22 – Цоколівка контактів при з'єднанні апаратних модулів за допомогою роз'ємів D-Sub

Розділ 2

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОНТРОЛЕРА BERGHOF

2.1 Середовище CODESYS

CODESYS – це сучасний інструмент для програмування контролерів (CODESYS утворюється від слів Controllers Development System) від компанії 3S – Smart Software Solutions GmbH. CODESYS є пристроєнезалежним середовищем, що надає користувачеві можливість створення програм на мовах стандарту IEC 61131-3.

2.2 Апаратні вимоги до середовища

Для середовища CODESYS потрібне наступне апаратне забезпечення:

- процесор – мінімум Pentium 4/Celeron 1,8 ГГц;
- оперативна пам'ять – не менше 1 Гбайт;
- дозволи монітора – не менше 1024x768;
- операційна система – Windows 7 і вище.

2.3 Інсталяція середовища

Середовище CODESYS можна завантажити і встановити з офіційного сайту компанії 3S – Smart Software Solutions – www.CODESYS.com. Послідовність завантаження середовища буде наступна:

1. На головній сторінці компанії перейти на вкладку «Download».
2. На сторінці «Download» необхідно перейти по посиланню «CODESYS V3.5», яка знаходиться в полі «CODESYS download area».
3. Потім слід натиснути кнопку «Download», що знаходиться під назвою продукту.

Зверніть увагу, щоб завантажити це середовище необхідно бути зареєстрованим користувачем на порталі store.CODESYS.com. Якщо натиснути на кнопку «Download» незареєстрованим користувачем, з'явиться застережливе вікно, яке запропонує увійти або зареєструватися.

Отриманий цим чином файл є інсталяційний. Для того, щоб приступити до інсталяції середовища CODESYS слід запустити цей файл. У першому вікні майстер інсталяції перевіряє наявність на комп'ютері необхідних компонентів і у разі відсутності пропонує їх інсталяцію.

По закінченню інсталяції додаткових компонентів майстер установки переходить безпосередньо до інсталяції середовища CODESYS, пропонуючи ознайомитися з умовами ліцензійної угоди і прийняти її.

У разі потреби є можливість змінити місцезнаходження цього ПО, натиснувши кнопку «Browse».

На наступному етапі установки ПО обираються список встановлюваних додатків і теки в програмному меню. Після цього почнеться процес копіювання і установки файлів на комп'ютері. По завершенню процесу майстер інсталяції запропонує ознайомитися і погодитися з важливою інформацією.

Завершальне вікно майстра інсталяції інформує про закінчення установки. В результаті установки в меню пуск буде створено теку «3 S CODESYS», що містить посилання на ПО CODESYS, що входить до складу середовища.

Розділ 3

ПЕРШІ КРОКИ РОБОТИ В СЕРЕДОВИЩІ CODESYS

3.1. Створення першого проекту

При першому запуску середовища розробки CODESYS відкриється діалогове вікно вибору профілю оточення за замовчанням (рис. 3.1).

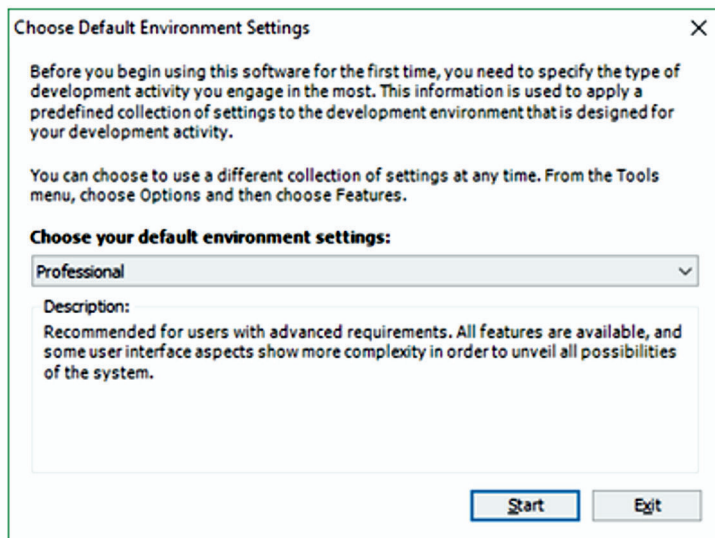


Рис. 3.1 – Діалогове вікно вибору оточення

Слід вибрати налаштування «Professional». У будь-який момент можна перемкнутися на налаштування «Standard» (Tools → Options → Features → Predefined features sets...), якщо це буде необхідно.

В оточенні «Standard» призначений для користувача інтерфейс підлаштований під ефективне використання, і деякі рідко використовувані функції прибрані. В оточенні «Professional» в призначеному для користувача інтерфейсі доступні усі функції і деякі аспекти інтерфейсу представлені у більше розгорнутому виді.

При наступному включенні система програмування автоматично відкриється з профілем, вибраним за замовчанням. Надалі, коли на комп'ютері з'являться різні версії компонентів CODESYS, буде необхідно обрати потрібний профіль системи при старті.

Створення нового проекту розпочинається з вибору в меню File команди New project. У правій частині вікна New project (рис. 3.2) в полі Templates необхідно вибрати Standard project, ввести ім'я проекту, а також задати місце

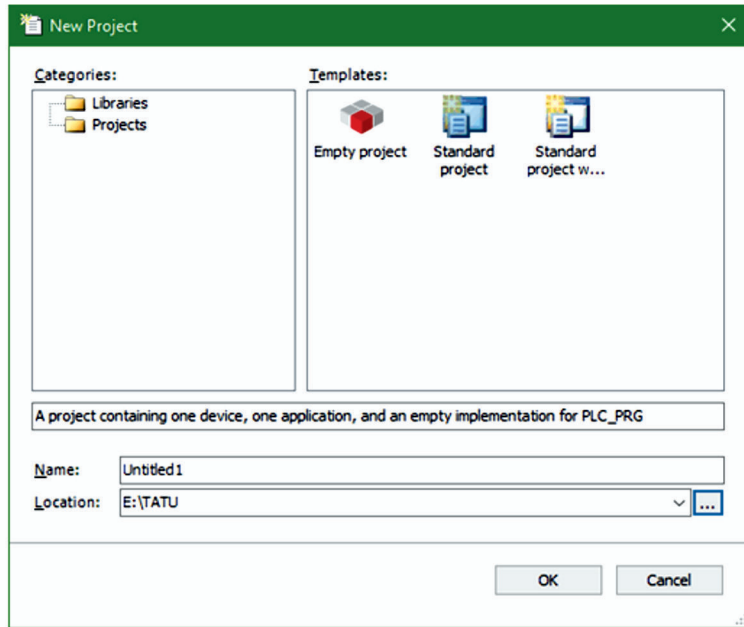


Рис. 3.2 – Діалогове вікно створення проекту

його розташування на комп'ютері. На завершення слід натиснути кнопку ОК. Після цього відкривається діалогове вікно майстра налаштування проекту (рис. 3.3).

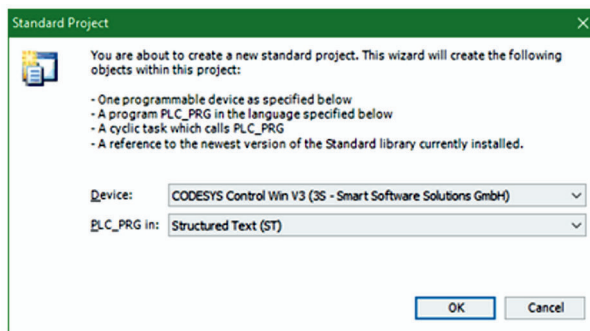


Рис. 3.3 – Діалогове вікно майстра налаштування проекту

У вікні Standard project слід вибрати пристрій CODESYS Control Win V3 і мову програмування Structured Text. Після натиснення на ОК відкривається новий проект. Ім'я проекту з'явиться в назві головного вікна CODESYS і в назві кореневого вузла у вікні «Devices».

У вікні «Devices» виводиться дерево проекту з пристроєм «Device (CODESYS Control Win V3)» і прикріпленим до нього додатком, який включає програму «PLC_PRG» і Конфігуратор завдань (Task Configuration) із завданням «Main Task». Це завдання якраз управляє виконанням «PLC_PRG».

Додаток, крім того, включає Менеджер бібліотек (Library Manager), який за умовчанням складається з бібліотеки «iostandard.library», що забезпечує конфігурацію входів/виходів, і Стандартної бібліотеки (Standard library), що об'єднує усі необхідні функції і функціональні блоки.

У разі потреби пристрій може бути перейменований. Для цього слід вибрати пристрій, натиснути <Пропуск> і в полі, що з'явилося, ввести його нове ім'я.

3.2. Створення першої програми

Створення програми насамперед розпочинається з оголошення змінних, які використовуватимуться при роботі додатка.

Для оголошення змінних в CODESYS використовується спеціальне вікно-редактор. Воно відкривається шляхом вибору POU «PLC_PRG», який стандартно розташовується у вікні POU.

Вікно «ST language editor» для редагування «PLC_PRG» відкривається в центральній частині вікна CODESYS. Редактор включає два розділи: у верхній частині розділ оголошень, в нижній розділ коду.

У лівій частині розділу оголошень виведені номери рядків, тип POU (в даному випадку програма «PROGRAM PLC_PRG»), а також ключові слова «VAR» і «END_VAR» для оголошення змінних. Спочатку текст програми відсутній за винятком одного рядка, який відображається.

Далі курсор слід переістити у розділ оголошень, поставити після слова VAR і натиснути <Введення>. У порожньому рядку, що з'явився, необхідно оголосити змінні *ivar* і *erg* типу INT і *fbinst* типу FB1:

```
PROGRAM PLC_PRG
VAR
  ivar: INT;
  fbinst: FB1;
  erg: INT;
END_VAR
```

Слід зазначити, що в середовищі CODESYS немає необхідності усі змінні оголошувати заздалегідь. У програмі їх можна використати без оголошення. В цьому випадку визначення змінних відбувається автоматично – «Auto Declaration» (див. нижче).

Наступним кроком в написанні програми є опис самого тіла програми. Для цього в нижній частині вікна-редактора в розділі коду PLC_PRG слід ввести наступне:

```

ivar:= ivar+1; //лічильник
fbinst(in:=I1, out=>erg); //
виклик экз. функц. блоку FB1 з параметрами erg:=
fbinst.out; //запис результату в «erg»

```

Замість такого запису параметрів і коментарів можливе використання функції автооголошення (Auto Declaration): ввести рядок коду і натиснути <Enter>. Для усіх ще не оголошених змінних відкриватимуться діалогові вікна «Auto Declare», в якому кожній змінній можуть бути задані необхідні параметри (рис. 3.4).

Поля «Name», «Scope» і «Object» заповняться автоматично.

Необхідно лише вибрати тип змінної і в графі «Comment» додати коментар. Слід зауважити, що в рядку оголошення ці коментарі не будуть показані, як це було наведено вище після «//». Коментар буде збережений у вигляді xml-опису і може використовуватися при складанні документації проекту.

Вікно «Auto Declare» закривається шляхом натиснення ОК. Після чого змінна *ivar* з'явиться в розділі оголошень змінних POU.

The image shows a dialog box titled "Auto Declare" with a green header bar. It contains several input fields and checkboxes. The "Scope" dropdown is set to "VAR". The "Name" text field contains "ivar". The "Type" dropdown is set to "INT". The "Object" dropdown is set to "PLC_PRG [Application]". The "Initialization" field is empty with a "... " button. The "Address" field is empty. Under "Flags", there are three unchecked checkboxes: "CONSTANT", "RETAIN", and "PERSISTENT". There is a large text area for "Comment". At the bottom right, there are "OK" and "Cancel" buttons.

Рис. 3.4 – Діалогове вікно Auto Declare

Тепер, для прикладу, у проект можна додати новий функціональний блок. У цьому прикладі він реалізований на мові ST і виконуватиме функцію складання значення входу *in* та змінної *ivar*, яка дорівнюватиме двом. Результуючі дані передаватимуться на вихід *out*.

З метою створення нового функціонального блоку POU потрібно в меню «Project» або в контекстному меню «Application» виділити підпункт «Add object» і вибрати в ньому «POU».

На екрані з'явиться вікно створення нового об'єкту POU.

У вікні «Add POU» у відповідному рядку необхідно ввести ім'я нового POU (у цьому прикладі – «FB1»), в полі «Type» вказати тип нового POU – програма, функціональний блок або функція (у прикладі – функціональний блок).

Мова програмування вибирається в полі «Implementation language» (у цьому прикладі був вибраний Structured Text). Збереження налаштувань підтверджується кнопкою Add.

Після появи вікна редактора нового функціонального блоку POU (рис. 3.5) слід задати параметри змінних тим же способом, що і в PLC_PRG:

```
FUNCTION_BLOCK FB1
VAR_INPUT
in: INT;
END_VAR
VAR_OUTPUT
out: INT;
END_VAR
VAR
ivar: INT:=2;
END_VAR
```

У розділі коду ввести наступний запис:

```
out:=in+ivar;
```

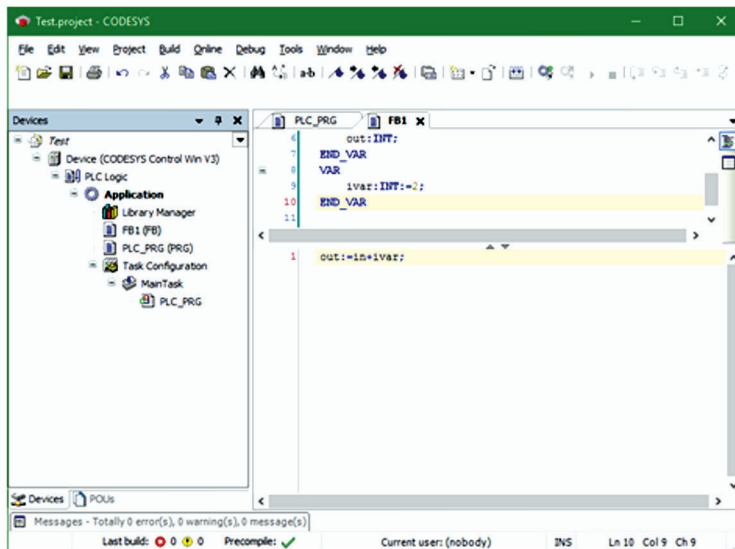


Рис. 3.5 – Вікно редактора для нового функціонального блоку

3.3 Налаштування ресурсів CODESYS для запуску і керування додатком в ПЛК

Середовище CODESYS може емулювати роботу справжнього ПЛК. Для цього в її складі є програми емуляції сервера – Gateway Server і PLC – CODESYS Control Win V3.

Gateway Server є службовою програмою і автоматично запускається спільно з операційною системою. Для керування і контролю за його роботою служить програма Gateway SysTray. З її допомогою можна вручну запускати і зупиняти роботу сервера. Також, варто враховувати, що меню цієї програми містить команду Exit Gateway Control, яка відключає програму Gateway SysTray, але не сам Gateway Server.

Підтвердженням запуску сервера є червоний колір іконки на панелі завдань. Чорний колір іконки вказує на відключений стан сервера.

CODESYS Control Win V3 також запускається автоматично з операційною системою. Його керуванням і контролем займається програма CODESYS Control Win SysTray. Ця службова програма надає функції включення і виключення ПЛК. За умовчанням, після запуску програми ПЛК вимкнений.

Це пов'язано з міркуваннями безпеки, і, починаючи з версії V3.5 SP2, запуск ПЛК не здійснюється при старті програми. При включеному ПЛК іконка має інший стан.

Слід запустити ПЛК вибравши команду Start PLC з контекстного меню програми CODESYS Control Win SysTray.

Далі слід задати активний додаток, тобто який виконуватиметься в даний момент.

По подвійному кліку по «Main Task» у вікні «Device» проекту відкриється редактор конфігурації завдань (рис. 3.6).

У вікні «Device» ім'я «Application» буде виділено жирним текстом. Це означає те, що це застосування є активним. Тому усі команди і дії стосовно установки зв'язку з ПЛК будуть відноситись тільки до цього застосування. Для того, щоб зробити додаток активним (якщо їх 2 і більше) слід вибрати Set Active Application в контекстному меню вибраного застосування у вікні «Devices». Після цього можна настроїти зв'язок між середовищем і віртуальним ПЛК.

Для початку налаштування слід відкрити вікно «Communication». Для цього треба клікнути двічі на ПЛК (у цьому прикладі «MyPLC») у вікні «Devices». У цьому вікні «Communication» необхідно встановити з'єднання між ПЛК (пристроєм, цільовою платформою) і середовищем програмування.

Насамперед треба настроїти Gateway сервер. Gateway сервер включений в інсталяційний пакет CODESYS. Для його додавання необхідно натиснути Add gateway, щоб відкрити відповідне діалогове вікно.

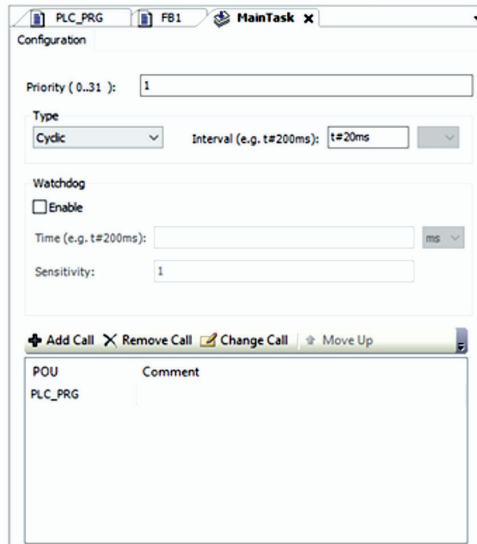


Рис. 3.6 – Редактор конфігурації завдань

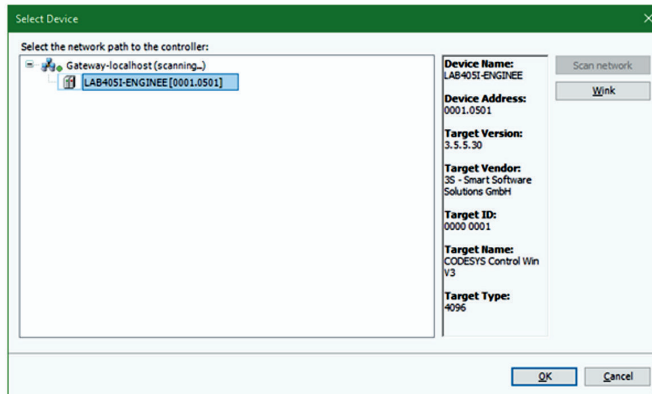


Рис. 3.7 – Вікно сканування підключених пристроїв

Тут пропонується ввести символічне ім'я (Name) для Gateway, визначити тип драйвера «TCP/IP» і ввести IP-адресу (у цьому прикладі «localhost»). Поле Port слід редагувати відповідно до вибраної IP-адреси. Після заповнення можна закрити вікно, натиснувши ОК.

Назва сервера буде поміщена в центральній частині вікна, а ім'я Gateway буде додано в список під написом «Gateway». Якщо сервер працює, поруч горітиме зелений індикатор, інакше він буде червоним.

Тепер треба визначити канал зв'язку, через який пристрій буде підключений через Gateway. Для цього слід натиснути кнопку Scan network. Таким чином буде здійснено пошук пристроїв, доступних цьому серверу в локальній мережі.

3.4 Компіляція і завантаження додатка в ПЛК

Первинним кроком для завантаження проекту в пристрій являється його компіляція і побудова. Це здійснюється за допомогою команди «Build» з меню Build або простим натисненням <F11>. Програма буде перевірена на наявність помилок. У наведеному прикладі їх немає, так що повідомлень про помилки бути не повинно (рис. 3.8).

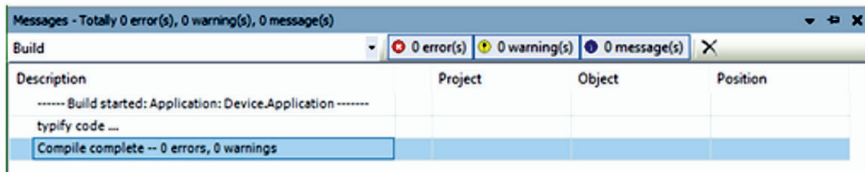


Рис. 3.8 – Успішна компіляція проекту

Наступним кроком є безпосереднє завантаження проекту в пристрій. Для цього можна використати команду Login з меню Online. Якщо усі налаштування зв'язку виконані відповідно до вказівок з пункту 3.3, виведеться повідомлення, показане на рис. 3.9 (інакше буде необхідно перевірити налаштування).

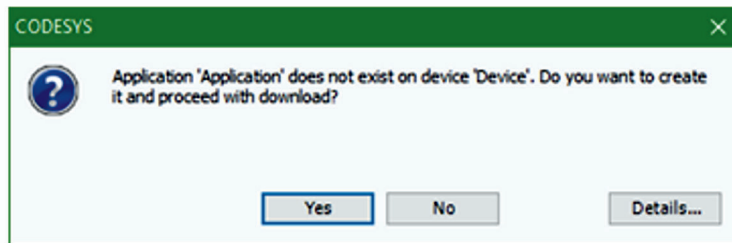


Рис. 3.9 – Діалогове вікно створення завантаження додатка в ПЛК

Після натиснення Yes додаток буде завантажений в ПЛК.

3.5 Запуск додатка і моніторинг змінних

Створивши проект і завантаживши відлагоджений додаток «Application», як описано вище, його можна запустити на пристрої CODESYS Control Win V3.

Запуск додатка в ПЛК здійснюється за допомогою команди Start, яка за умовчанням знаходиться в контекстному меню вибраного застосування або через меню «Debug». Програма почне виконуватися.

Далі можна відстежувати хід виконання цієї програми на цьому контролері, тобто виконувати його моніторинг. Найчастіше відстежують зміну деяких змінних і їх значень.

Є декілька способів моніторингу змінних застосовної програми:

- перегляд змінних, заздалегідь визначених в watch list;
- запис і фіксація змінних;
- перегляд значень змінних певного POU, у вікні редактора.

3.5.1. Перегляд значень змінних певного POU

POU може мати декілька екземплярів, дані яких різні, тому при моніторингу змінних POU, треба мати можливість вибору необхідного екземпляра.

Для цього у вікні «POUs» необхідно клікнути двічі на програмі «PLC_PRG» або вибрати команду «Edit object» з контекстного меню цього рядка. У діалоговому вікні, що відкрилося, відобразяться усі екземпляри PLC_PRG (у нашому прикладі тільки один тому це вікно не відкриється).

Відкриється вікно PLC_PRG (рис. 3.10), нижня частина якого є тілом програми, де можна безпосередньо простежити за тим, як виконується PLC_PRG; поряд з кожною змінною є маленьке віконце з поточним значенням цієї змінної. Верхня частина є таблицею, в якій перераховані елементи PLC_PRG; тут показані поточні значення відповідних змінних додатка.

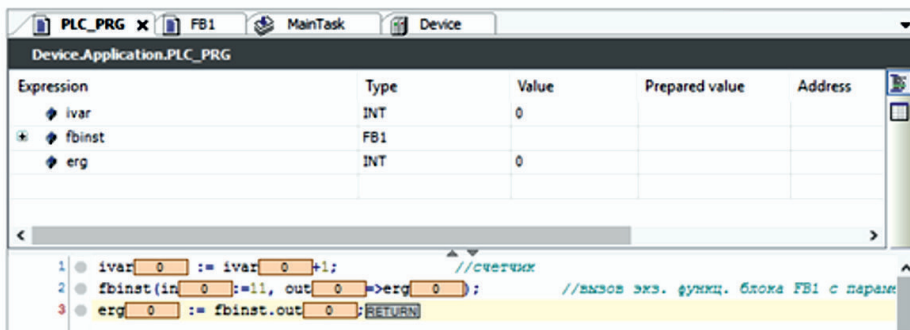


Рис. 3.10 – Вікно PLC_PRG в режимі моніторингу

3.5.2. Запис і фіксація змінних

Можна задати одноразово або зафіксувати заготовлене значення (Prepared value) до будь-якої змінної в PLC, наприклад *ivar*. Це означає, що змінною привласнюватиметься це значення на початку кожного наступного програмного циклу.

Для цього слід клікнути двічі в полі колонки Prepared value, ввести бажане ціле значення і закрити поле, натиснувши <Введення> чи клікнувши мишкою поза полем. Виконавши команду «Write values» або «Force values», значення запишеться або зафіксується в PLC. Результат буде негайно відображений в колонці Value.

3.5.3. Використання списків змінних

Вікна Watch view можна використати для створення наборів об'єктів, моніторинг яких необхідно виконати при відлагоджуванні.

Для цього в меню View треба вибрати команду «Watch» -> «Watch1». Відкриється вікно для введення і редагування списку змінних.

У колонці Expression слід клікнути по першому рядку таблиці, щоб активувати поле введення. Тут потрібно ввести повний шлях для змінної моніторинг якої необхідно виконати. В цьому прикладі: «MyPlc.Application.PLC_PRG.ivar». Потім натиснути <Введення>, щоб підтвердити введення. Тип змінної автоматично з'явиться в колонці Type. Зручно використати кнопку при введенні імені змінною. Після її натиснення відкриється діалогове вікно вибору змінної (рис. 3.11).

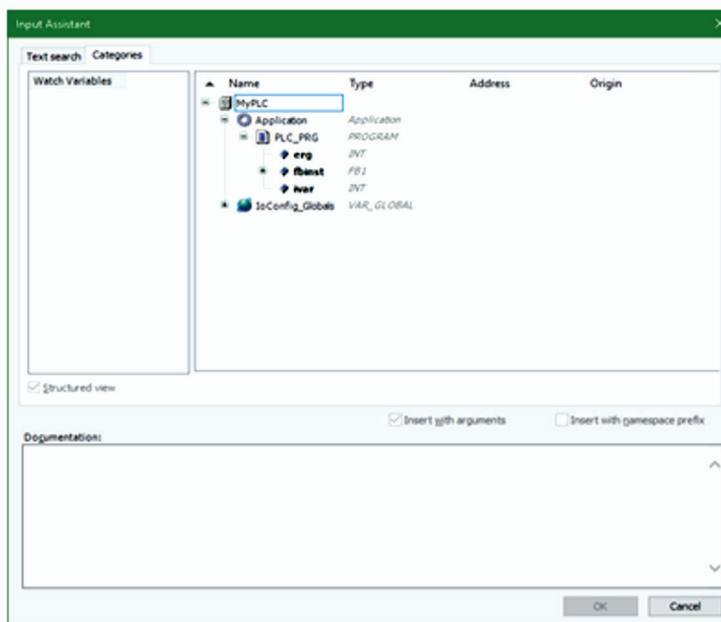


Рис. 3.11 – Вікно вибору змінної

Так само додаються і інші змінні. Також слід зауважити, що в цей список можуть додаватися змінні не лише з одного елементу програми, але і інших об'єктів ROU. Повторіть операцію для інших змінних. Запис і фіксація значень змінних в списках виконуються так само як було описано вище (п. 3.5.2).

Щоб розірвати з'єднання з ПЛК, доречно використати команду «Logout» з меню Online.

3.6 Установка кроків і точок зупинки

У режимі онлайн можливо встановити точки зупинки (Breakpoints), в яких виконання програми буде призупинено.

Якщо програма досягла точки зупинки, можна виконати її по кроках. У кожній такій точці доступні поточні значення змінних.

Для додавання точки зупинки спершу слід виділити рядок 1 в PLC_PRG, а потім натиснути <F9> чи команду «Toggle Breakpoint» з меню Debug. Буде встановлена точка останову. Якщо в даний момент додаток знаходиться в стані STOP, то відобразиться додана точка відобразиться як на рис. 3.12.

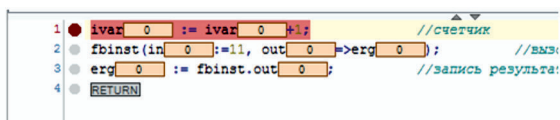


Рис. 3.12 – Точка зупинки в стані STOP

Якщо ж воно запущене, то останов буде виконаний у вказаному рядку (рис.3.13).

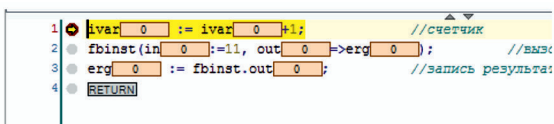


Рис. 3.13 – Зупинка додатка в точці зупинки

Тепер можна виконати програму по кроках клавішею <F8>, чи використувуючи команду Step Into з меню Debug. Для виконання програми по кроках без переходу у функціональний блок слід виконати команду Step Over (<F10>).

Також можна викликати діалогове вікно Breakpoints за допомогою команди «Breakpoints» з меню View. Тут можна переглядати і змінювати поточні точки зупинки, а також встановити нові.

Пам'ятайте, що ці точки зберігаються в пам'яті, коли ви відключаєтеся від ПЛК. Вони будуть помічені червоними кружками, що потьмяніли.

3.7 Створення проекту для контролера BergHof

Для початку роботи з контролером фірми BergHof необхідно насамперед інсталиувати в CODESYS необхідний пакет підтримки. Для цього слід відкрити Package Manager через меню Tools.

У цьому вікні слід натиснути на кнопку Install і в діалоговому вікні, що відкрилося, вибрати шлях до пакету для контролера BergHof (рис. 3.14). Потім відкриється нове діалогове вікно, в якому слід вибрати Typical setup і чекати доки усе встановиться.

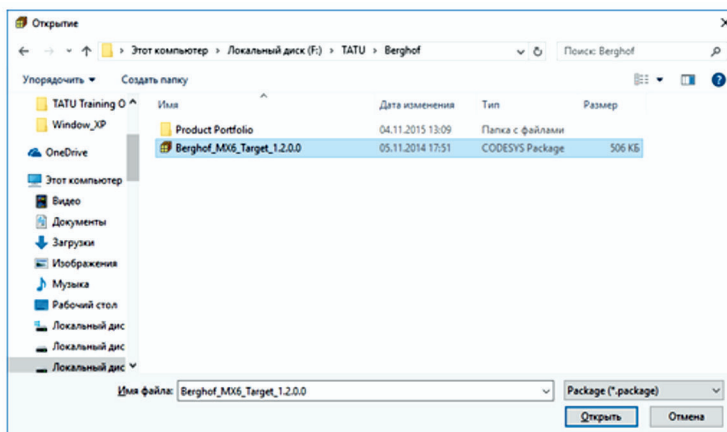


Рис. 3.14 – Диалогове вікно вибору пакету

Після установки пакет повинен відображатися в Package Manager.

Далі треба створити новий проект як в розділі 3.1, тільки при цьому слід вибрати пристрій Berghof MX6 Control (Berghof Automation GmbH) і зручну мову програмування (рис 3.15). Після усіх операцій відкриється вікно нового проекту (рис. 3.16).

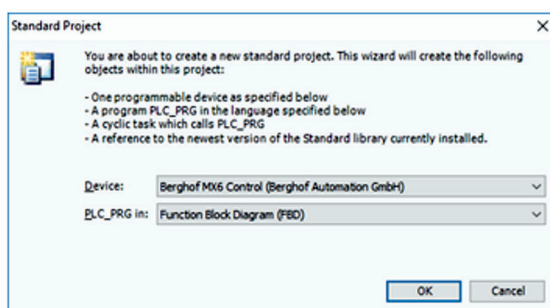


Рис. 3.15 – Вікно вибору пристрою

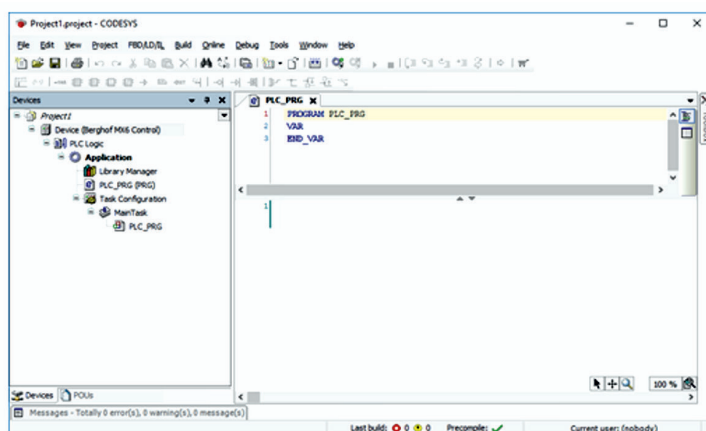


Рис. 3.16 – Вікно нового проекту

Потім слід додати розширювач фізичних входів. Для цього треба відкрити діалогове вікно Add Device через контекстне меню Device (Berghof MX6 Control). У цьому вікні в списку Miscellaneous вимагається вибрати пристрій IO Slot і натиснути кнопку Add Device. Після цього закриваємо діалогове вікно Add Device (рис. 3.17).

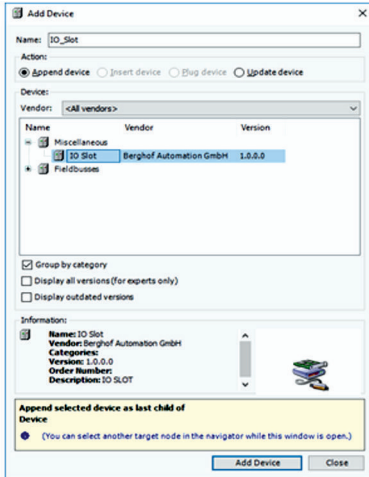


Рис. 3.17 – Діалогове вікно Add Device

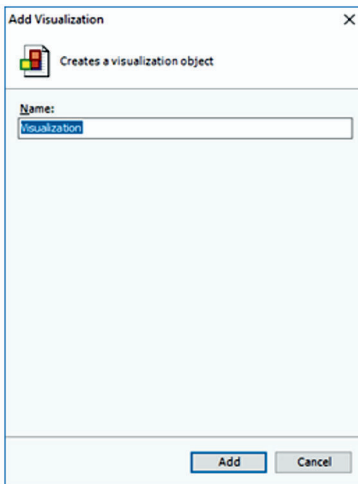


Рис. 3.18 – Створення візуалізації

Потім слід підключити пристрій. Через контекстне меню порожнього елемента (<Empty>) пристрій «IO_Slot(IO Slot)» треба вибрати пункт Plug Device. У діалоговому вікні, що з'явився, вимагається вибрати пристрій Berghof IO і натиснути кнопку Plug Device. Після цього пристрій повинен відобразитися в дереві проекту.

Наступним кроком буде додавання візуалізації в проект. Для цього в контекстному меню Application необхідно вибрати Add Object → Visualization.. Також це можливо здійснити через меню Project → Add Object → Visualization. У діалоговому вікні введіть ім'я об'єкту візуалізації (рис. 3.18).

На цьому початкове створення проекту закінчене. Тепер проект виглядатиме як показано на рис. 3.19.

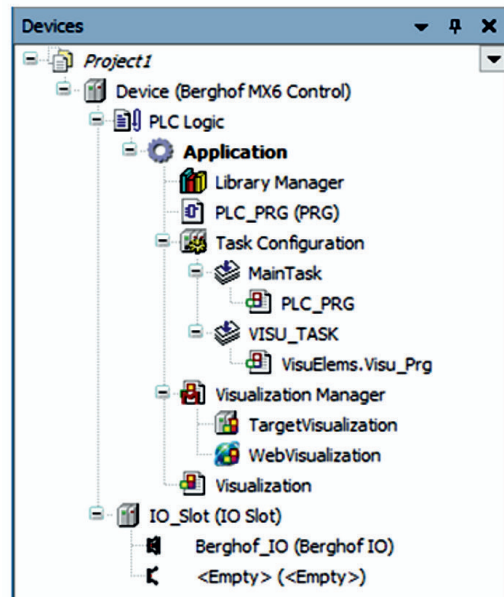


Рис. 3.19 – Вид готового проекту

3.8 Налаштування підключення контролера BergHof з комп'ютером

Перед тим, як почати налаштування підключення, слід визначити мережеву адресу контролера. Якщо контролер конфігурується перший раз, то він матиме стандартні мережеві налаштування, описані нижче. Якщо ж він вже налаштовувався, слід упізнати його мережеві налаштування. На випадок неможливості визначити мережеві налаштування контролера можна скинути їх до стандартних. Для цього слід перезавантажити контролер, затиснувши при цьому кнопку S1 (рис. 3.20) до того моменту, поки світлодіоди не почнуть блимати з періодичністю у 2 с. Це означатиме, що контролер повернувся до заводських налаштувань.

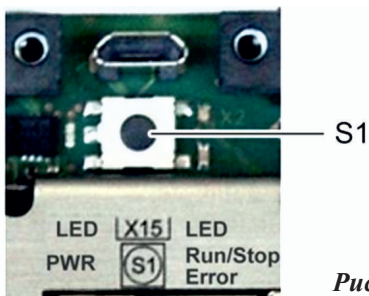


Рис. 3.20 – Зовнішній вигляд кнопки контролера

Нижче будуть використані налаштування, коли контролер знаходиться в первинному заводському стані.

Щоб комп'ютер зміг підключитися до контролера слід присвоїти комп'ютеру мережеву адресу:

IP – 169.254.255.1

Subnet mask – 255.255.255.0

Тепер можна підключити контролер через мережевий кабель до комп'ютера. Мережева адреса контролера виглядатиме таким чином:

IP – 169.254.255.XX,

де XX – це останні дві цифри серійного номера контролера, який можна знайти на наклейці контролера (рис. 3.21).

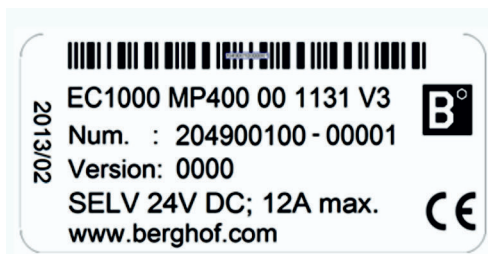


Рис. 3.21 – Наклейка на контролері з серійним номером

Далі необхідно відкрити будь-який браузер і ввести в адресному рядку мережеву адресу контролера. Якщо усе встановлено правильно, то з'явиться сторінка входу – веб-інтерфейс контролера.

Для входу необхідно використати наступне Ім'я і Пароль:

Name: admin

Password: admin

Після входу відкриється сторінка конфігурацій (рис. 3.22)



Рис. 3.22 – Сторінка конфігурації

Насамперед слід встановити необхідну мережеву адресу контролера. Для цього на сторінці конфігурацій слід відкрити посилання «Network» в списку Configuration. Відкриється сторінка мережевих налаштувань.

Полі ETH0 відповідає за входи X10...X12, а ETH1 – за X13. Налаштуйте параметри мережі згідно необхідним. По закінченню налаштувань необхідно натиснути кнопку Save.

Після цього слід перезавантажити контролер. Це можна здійснити двома способами: відключити живлення від контролера, а після цього підключити його назад, або зайти на сторінку Reboot в полі System і натиснути кнопку «Reboot Module».

Потім слід змінити налаштування мережевої адреси комп'ютера відповідно до вибраних і через браузер, вже за новою адресою, яка була прописана, зайти на сторінку конфігурації контролера.

Якщо усе зроблено правильно, то з'явиться сторінка входу. На цьому мережева конфігурація контролера вважається завершеною.

3.9 Оновлення прошивки контролера Berghof

Цей крок потрібний для правильної роботи середовища CODESYS з контролером Berghof EC2250. Прошивка має бути версії 1.2.0. Для перевірки поточної версії прошивки (Firmware Version) необхідно перейти на сторінку Info, що знаходиться в полі System.

Для оновлення прошивки необхідно спершу зайти на сторінку Update що знаходиться в полі System конфігурації контролера. Потім вибрати файл «firmware_mx6 - plc_1.2.0.tgz» в діалоговому вікні, що з'явилось, і натиснути кнопку «Sending Request». Після цього почнеться завантаження прошивки в контролер і по закінченню станеться перезавантаження пристрою. На сторінці Info повинна з'явитися оновлена версія прошивки.

3.10 Установка з'єднання з контролером Berghof

Для установки з'єднання між контролером і комп'ютером в програмі CODESYS необхідно відкрити вікно «Communication», клікнувши двічі на «Device (Berghof MX6 Control)» у вікні «Devices».

Насамперед необхідно настроїти Gateway сервер. Для цього треба натиснути кнопку Add gateway, щоб відкрити відповідне діалогове вікно (рис. 3.23).

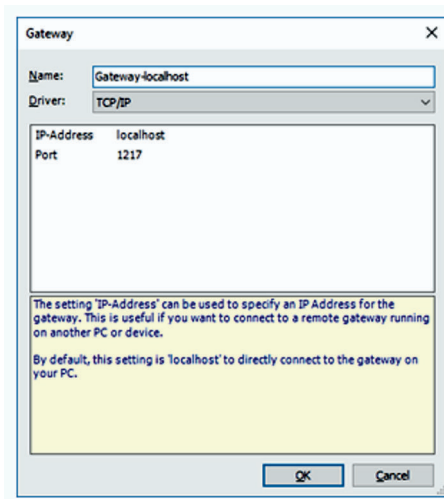


Рис. 3.23 – Діалогове вікно Add Gateway

У цьому діалоговому вікні необхідно ввести бажане символічне ім'я (Name) для Gateway, і визначити тип драйвера «TCP/IP». Якщо контролер підключений безпосередньо до комп'ютера, то поле IP-Address повинне містити «localhost» (поле можна редагувати після подвійного кліку). У іншому випадку необхідно ввести адресу комп'ютера, до якого підключений контр-

олер. Поле Port може залишитися незмінним. Закрити вікно і зберегти налаштування можна натиснувши ОК.

Назва сервера буде поміщена в центральній частині вікна, а ім'я Gateway буде додано в список під написом «Gateway». Якщо сервер працює, поруч горітиме зелений індикатор, інакше він буде червоним (рис. 3.24).

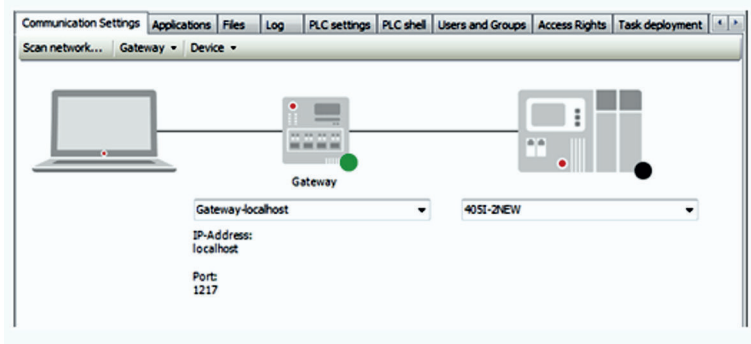


Рис. 3.24 – Приклад працюючого серверу Gateway

Тепер необхідно визначити канал зв'язку, через який пристрій буде підключений через Gateway. Це робиться натисненням кнопки Scan network. У цьому діалоговому вікні будуть відображені підключені контролери до даного серверу. Для пошуку доступних цьому серверу пристроїв в локальній мережі необхідно натиснути кнопку Scan network.

Після цього повинен визначитися хоч би один ПЛК. Якщо цього не сталося, слід перевірити підключення контролера з цим сервером (локальним комп'ютером). Визначивши підключені пристрої, необхідно вибрати ПЛК Berghof і натиснути кнопку ОК. Таким чином активується канал зв'язку з цим контролером.

3.11 Налаштування терміналу Berghof ET1007

Ethernet термінал Berghof ET1007 (рис. 1.19) спеціально призначений для відображення CODESYS візуалізації. Він може відображати як і web візуалізації для CODESYS контролерів, так і target візуалізації через VNC протокол.

Для того, щоб термінал міг показувати завантажену на контролер візуалізацію, необхідно його настроїти. Налаштування полягатиме у вказівці потрібної адреси терміналу і шляху підключення.

Після включення термінал налагоджений по замовчуванню і намагатиметься підключитися до контролера по тих налаштуваннях, що у нього виставлені. По спливанню невеликого проміжку часу, за умови що термінал не зміг підключитися до контролера, на екрані буде доступна кнопка Configuration. Натиснувши на неї відкриється вікно налаштування терміналу.

Перша сторінка налаштування, основна інформація про контролер. Для переходу на наступну сторінку необхідно натиснути кнопку Next.

На другій сторінці терміналу можна буде настроїти мережеву адресу терміналу. Редагування полів здійснюється після натиснення на кнопку Edit. Встановивши потрібну мережеву адресу (для прикладу, IP – 192.168.1.11, Mask – 255.255.255.0) можна перейти на наступну сторінку налаштувань.

Третя сторінка є налаштуванням VNC сервера, до якого підключатиметься термінал. У полі Server необхідно ввести мережеву адресу контролера, у якому здійснюватиметься підключення, а потім слід натиснути кнопку Expert, щоб відкрити вікно додаткових налаштувань VNC сервера. На цій сторінці слід змінити тип підключення. У полі Lifeguard виставити значення Ping LG. Потім можна перейти на наступну сторінку конфігурацій.

Далі будуть сторінки налаштування звуку, яскравості екрану і калібрування екрану. Їх налаштування не впливатиме на підключення, тому змінювати їх необов'язково.

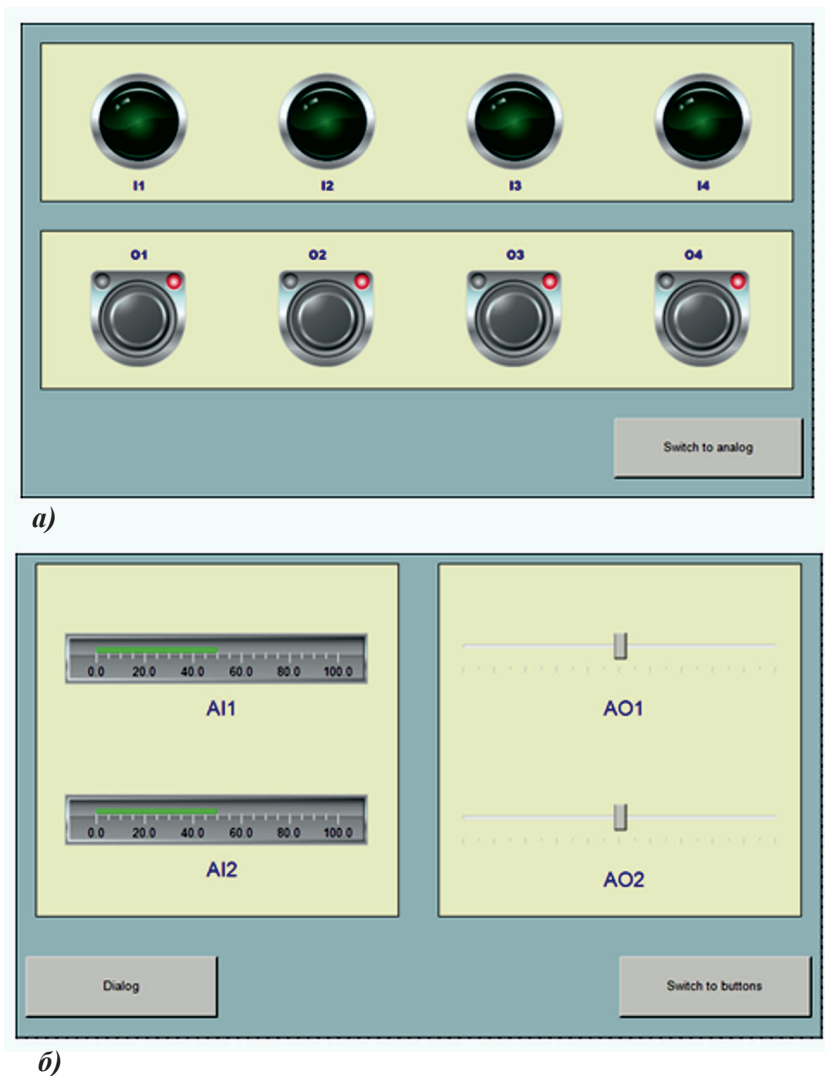
Після усіх налаштувань на останню сторінку будуть висвітлені усі внесені зміни і прохання підтвердження їх. Після підтвердження усі налаштування будуть збережені і термінал буде готовий до роботи.

Розділ 4

ПРИКЛАДИ ПРОЕКТІВ

4.1 Створення простої панелі керування

Приклад панелі керування зображено на рис. 4.1.

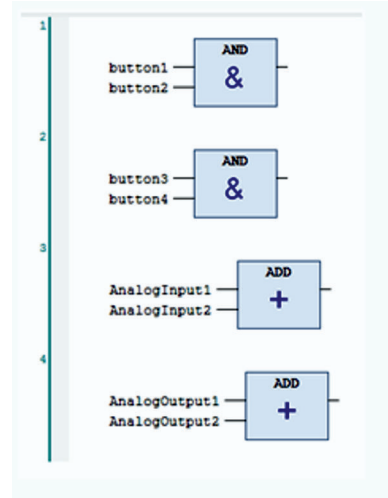


*Рис. 4.1 – Панель керування:
а) панель кнопок і індикаторів; б) панель аналогових регуляторів*

Створення основної програми

Головна програма міститиме оголошення змінних, представлених нижче, а також код на мові FBD.

```
PROGRAM PLC_PRG
VAR
    button1: BOOL;
    button2: BOOL;
    button3: BOOL;
    button4: BOOL;
    lamp1: BOOL;
    lamp2: BOOL;
    lamp3: BOOL;
    lamp4: BOOL;
    AnalogInput1: REAL;
    AnalogInput2: REAL;
    AnalogOutput1: REAL;
    AnalogOutput2: REAL;
END_VAR
```



Створення візуалізації

Спершу потрібно створити 3 вікна візуалізації з назвами BUTTON_VISU, ANALOG_VISU, DIALOG. Услід за цим треба змінити розміри робочої області візуалізації. У вікні «Properties», яке можна викликати з контекстного меню візуалізації, слід вибрати вкладку «Visualization». На цій вкладці необхідно поставити галочку навпроти «Use specified visualization size» і в полі «Visualization size» ввести розміри. Для візуалізації BUTTON_VISU і ANALOG_VISU розміри мають бути такі: Width – 800, Height – 480. Для візуалізації DIALOG слід виставити розміри: Width – 400, Height – 200.

Наповнення візуалізації відбувається за допомогою базових елементів, які доступні в CODESYS за умовчанням з правого боку вікна у вкладці «Toolbox».

Для зручності використання усі елементи розділені на декілька категорій за функціональними особливостями.

Щоб створити прямокутник треба вибрати зі списку «Basic» вкладки «Toolbox» вибрати «Rectangle» і після цього натисненням і утриманням лівої кнопки миші в області візуалізації створити прямокутник потрібного розміру. Також можна просто натиснути лівою кнопкою миші. Створиться прямокутник стандартного розміру, розміри якого можна змінювати також

показчиком миші або ввести за допомогою клавіатури.

Для введення розмірів з клавіатури треба по створеному прямокутнику знову натиснути лівою кнопкою миші. Автоматично відкривається вкладка «Properties» (рис. 4.2), в якій в списку «Position» необхідно змінити поля Width і Height.

Після цього прямокутник слід перемістити в початок координат, що можна здійснити або за допомогою миші, або просто обнуливши поля X і Y в списку «Position».

Для зміну його кольору слід відкрити список Colors, а потім підсписок Normal State, де є поля Frame color – колір окантовки, і Fill color – колір заповнення. Колір для цих полів можна вибрати зі списку або ж визначити власний колір.

Далі слід створити ще 2 прямокутники вже меншого розміру і іншого кольору, в яких будуть знаходитися кнопки і лампи.

Тепер можна перейти до створення ламп (Lamps). Вони доступні в списку Lamps/Switches/Bitmaps. Обравши в списку їх можна створити як і прямокутники. Задати розмір, затиснувши лівою кнопкою миші на робочій області або клікнувши лівою кнопкою миші по робочій області, і надалі змінювати розміри мишкою або в текстовому форматі.

Після створення лампи її необхідно прив'язати до змінної. Для цього у вкладці «Properties» лампи є поле Variable. У нього слід вписати ім'я змінної, що підключається, або вибрати його зі списку за допомогою Input Assistant. У цьому прикладі до ламп прив'язуються змінні lamp1, lamp2, lamp3 і lamp4.

Так само створюються і кнопки (Push Switch LED), які також знаходяться в списку Lamps/Switches/Bitmaps. До кнопок прив'язуються змінні button1, button2, button3 і button4.

Для наочного відображення яка кнопка або лампа за яку функцію відповідає, їх можна підписати. Проста текстова мітка (Label) доступна в списку Common controls, і створюється так само, як і вище описані елементи. Для зміни тексту мітки існує поле Text в списку Texts вкладки «Properties». Для зміни шрифту або кольору тексту є список Text properties, в якому знаходяться потрібні поля.

Щоб перемкнутися між візуалізацією необхідно створити активний елемент, при натисненні на який здійснюватиметься перехід. Активним

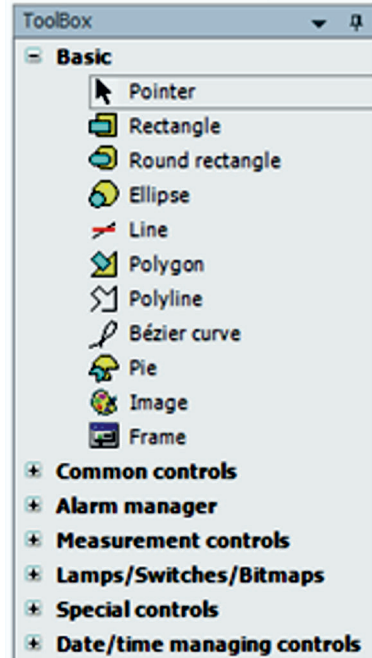


Рис. 4.2 – Вкладка Toolbox

може служити майже будь-який створений стандартний елемент, для цього необхідно прив'язати до нього потрібну дію. У цьому прикладі таким елементом є кнопка.

Кнопка (Button) знаходиться в списку Common controls і створюється так само, як і попередні елементи. Прив'язка кнопки (чи іншого елемента) до якої-небудь реакції на дію користувача можна здійснити в списку Inputconfiguration вкладки Properties (рис. 4.3).

Цей список містить різні дії, які може вчинити користувач. Натиснути кнопкою миші (чи пальцем на сенсорному екрані), затиснути кнопку, відсунути покажчик та ін. В прикладі потрібна буде дія OnMouseClicked, яка викли-

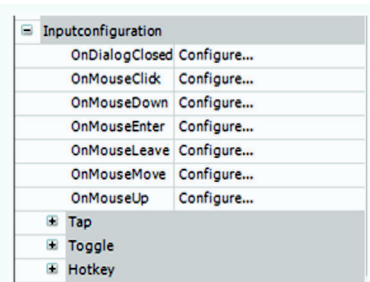


Рис.4.3 – Список Inputconfiguration

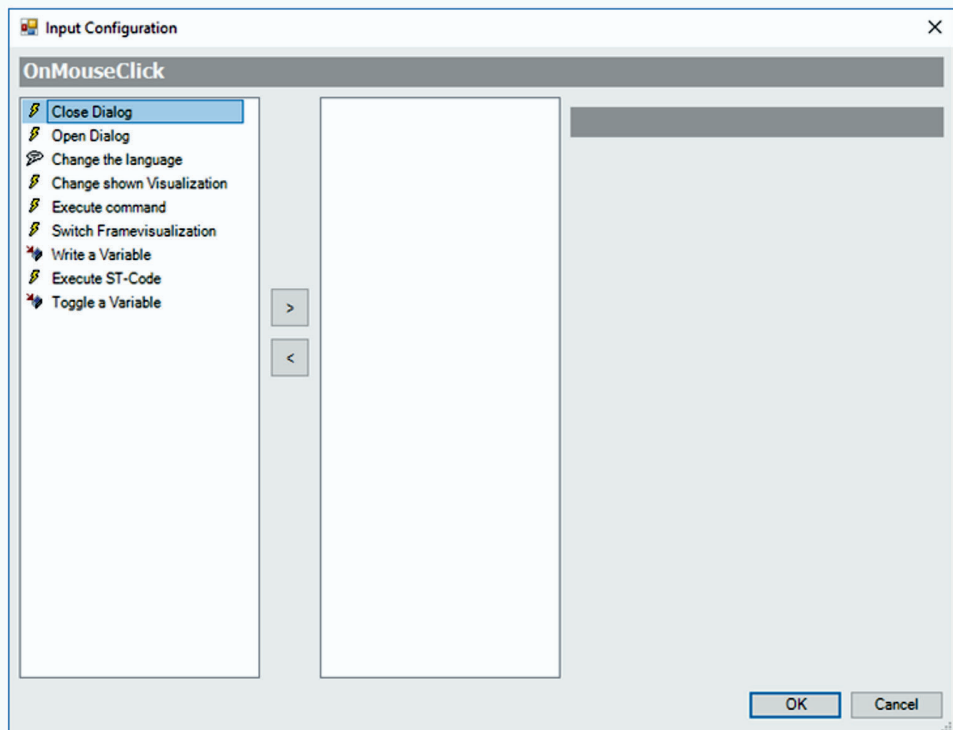


Рис. 4.4 – Вікно Input Configuration

кається кліком по полю Configure. Навпроти поля з дією, з'являється вікно Input Configuration (рис. 4.5).

У цьому вікні в лівій колонці запропоновані різні дії. Для перемикаання візуалізації слід вибрати дію Change shown Visualization і натиснути кнопку із стрілкою управо, після чого вона з'явиться в правій колонці, а правіше будуть відображені параметри переходу.

У полі Assign вимагається вибрати ту візуалізацію, на яку здійснюватиметься перехід, в даному випадку це ANALOG_VISU, і натиснути кнопку OK.

Розставивши елементи відповідно до прикладу, створення візуалізації BUTTON_VISU можна вважати завершеним.

Тепер можна перейти до наступної візуалізації ANALOG_VISU (рис. 4.16).

Первинні налаштування такі самі, як і з попередньою візуалізацією, аж до створення прямокутників для оформлення фону і кнопки перемикаання візуалізації.

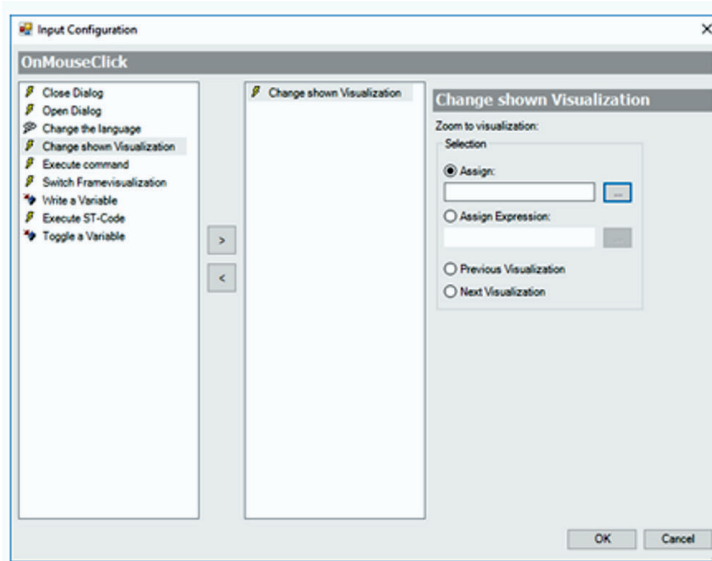


Рис. 4.5 – Активна дія і її параметри переходу

Новими елементами в цій візуалізації є Slider і Bar Display.

Slider доступний в списку Common controls. Він призначений для установки вихідної величини за допомогою слайдера (повзунка) від початкової до кінцевої точки. Його стандартний вид після створення представлений на рис. 4.6.



Рис. 4.6 – Зовнішній вигляд елементу Slider

Для зміни його зовнішнього вигляду, який представлений в прикладі (рис. 4.16), слід виконати два кроки:

- 1) змінити його розмір в довжину, що можливо здійснити за допомогою миші або редагування полів Width і Height в списку «Position»;
- 2) поставити галочку в полі Show scale списку Scale, щоб відобразити шкалу.

Також слід прив'язати змінну. У цьому прикладі прив'язуються змінні AnalogOutput1 і AnalogOutput2.

Наступний елемент Bar Display, який знаходиться в списку Measurement controls. Цей елемент призначений для відображення інформації у формі кольорової смуги (рис. 4.7), яка змінюється залежно від прикріпленої до нього змінної.

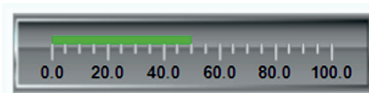


Рис. 4.7 – Елемент Bar Display

Створення Bar Display проходить також, як і елементу Slider, тобто створюється елемент, змінюється його розмір і прив'язується змінна. До цих елементів слід прив'язати змінні AnalogInput1 і AnalogInput2.

У візуалізації ANALOG_VISU також є ще одна додаткова кнопка Dialog, яка викликає діалог.

Діалог – «Спеціальне вікно», яке відкривається поверх наявної візуалізації, і може чекати якої-небудь дії від користувача, або просто надає інформацію.

Для того, щоб кнопка викликала вікно діалогу у вікні Input Configuration для дії користувача OnMouseClicked слід їй прив'язати дію Open Dialog, де в полі Dialog вибрати заздалегідь створену візуалізацію діалогу.

Візуалізації DIALOG (рис. 4.8) відрізняється від двох попередньої візуалізації тим, що вона є діалогом. Щоб зробити її такою треба у вікні «Properties», яке можна викликати з контекстного меню візуалізації, слід вибрати вкладку «Visualization» і знайти поле Use Visualization as:, в якому поставити галочку навпроти пункту Dialog.

При створенні кнопок в цій візуалізації, хоч би для однієї з них слід призначити дію Close Dialog, щоб діалог можна було закрити.

На цьому створення візуалізації для цього прикладу може вважатися завершеним.

Компіляція і прошивка проекту

Після усіх кроків створення основної програми і візуалізації, наступним, що треба зробити являється прив'язка змінних до фізичних входів-виходів.

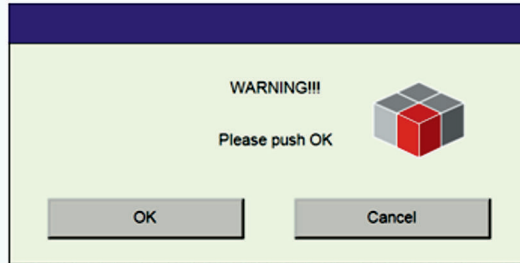


Рис. 4.8 – Візуалізація Dialog

У вікні Berghof_IO, яке можна викликати з дерева проекту, на вкладці BGH Slot I/O Mapping будуть доступні усі входи-виходи цього контролера (рис. 4.9).

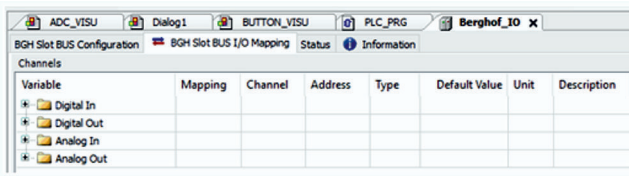


Рис. 4.9 – Теки з доступними входами-виходами

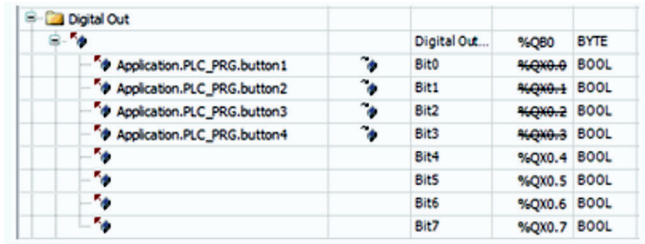
Підключення змінних здійснюється введенням їх імені в потрібне поле виводу або цілого порту. Лампи слід підключати до цифрових входів (Digital In) контролера (рис. 4.10). Кнопки слід підключити до цифрових виходів (Digital Out) контролера (рис. 4.11). Змінні AnalogInput1 і AnalogInput2 треба підключати до аналогових входів (Analog In) по напрузі (рис. 4.12). А змінні AnalogOutput1 і AnalogOutput2 – до аналогових виходів (Analog Out) контролера (рис. 4.13).

Після підключення усіх змінних програму слід відкомпілювати, упевнитися що помилок немає, а потім завантажити програму в контролер.

Успішно завантажена програма відразу ж чекатиме від користувача старту додатка.

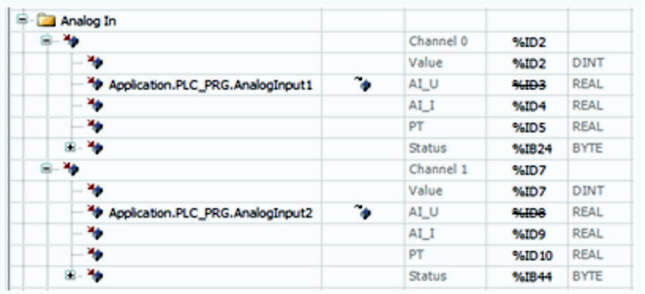
Variable	Mapping	Channel	Address	Type
- Digital In - Application.PLC_PRG.lamp1 - Application.PLC_PRG.lamp2 - Application.PLC_PRG.lamp3 - Application.PLC_PRG.lamp4		Digital Inp...	%IB0	BYTE
		Bit0	%IX0.0	BOOL
		Bit1	%IX0.1	BOOL
		Bit2	%IX0.2	BOOL
		Bit3	%IX0.3	BOOL
		Bit4	%IX0.4	BOOL
		Bit5	%IX0.5	BOOL
		Bit6	%IX0.6	BOOL
		Bit7	%IX0.7	BOOL
		Digital Inp...	%IB1	BYTE

Рис. 4.10 – Підключені змінні до цифрових входів контролера



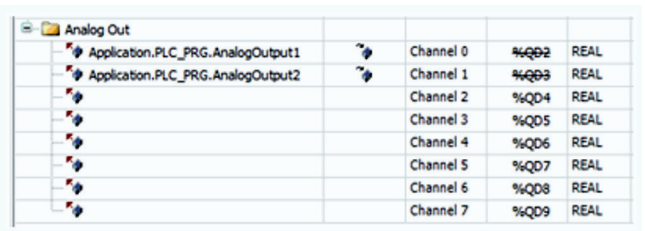
Digital Out...	%QB0	BYTE
Bit0	%QX0.0	BOOL
Bit1	%QX0.1	BOOL
Bit2	%QX0.2	BOOL
Bit3	%QX0.3	BOOL
Bit4	%QX0.4	BOOL
Bit5	%QX0.5	BOOL
Bit6	%QX0.6	BOOL
Bit7	%QX0.7	BOOL

Рис. 4.11 – Підключені змінні до цифрових виходів контролера



Channel 0	%ID2	
Value	%ID2	DINT
AI_U	%ID3	REAL
AI_I	%ID4	REAL
PT	%ID5	REAL
Status	%IB24	BYTE
Channel 1	%ID7	
Value	%ID7	DINT
AI_U	%ID8	REAL
AI_I	%ID9	REAL
PT	%ID10	REAL
Status	%IB44	BYTE

Рис. 4.12 – Підключені змінні до аналогових входів контролера



Channel 0	%QD2	REAL
Channel 1	%QD3	REAL
Channel 2	%QD4	REAL
Channel 3	%QD5	REAL
Channel 4	%QD6	REAL
Channel 5	%QD7	REAL
Channel 6	%QD8	REAL
Channel 7	%QD9	REAL

Рис. 4.13 – Підключені змінні до аналогових виходів контролера

4.2 Приклад програми для роботи світлофора

Цей приклад емулюватиме роботу світлофора для машин і пішоходів (рис. 4.14).

Кожній лампі у світлофорах відповідає фізичний вхід на контролері, а також один з виходів підключений до кнопки пішохода, тим самим надається можливість керування світлофором зовнішнім контролером.

Створення основної програми

Головна програма утримуватиме оголошення змінних, представлених нижче, а також код на мові FBD (рис 4.15)

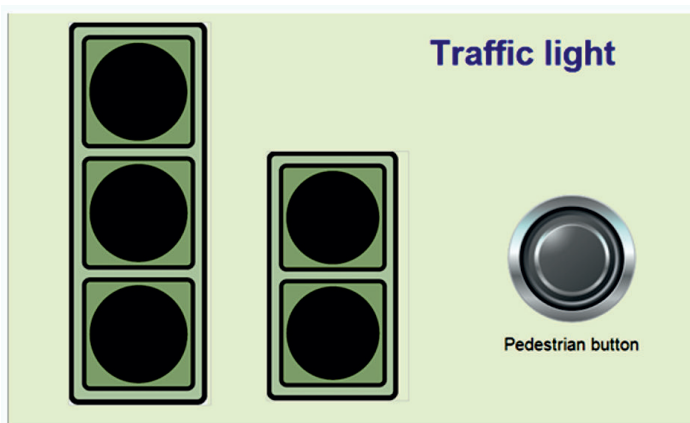


Рис. 4.14 – Зовнішній вигляд програми світлофора

```
PROGRAM PLC_PRG
```

```
VAR
```

```
    ImagePedestrian: STRING;
```

```
    ImageCar: STRING;
```

```
    Image_Switch_Ped0 : Image_Switch_Ped;
```

```
    Image_Switch_Car0 : Image_Switch_Car;
```

```
END_VAR
```

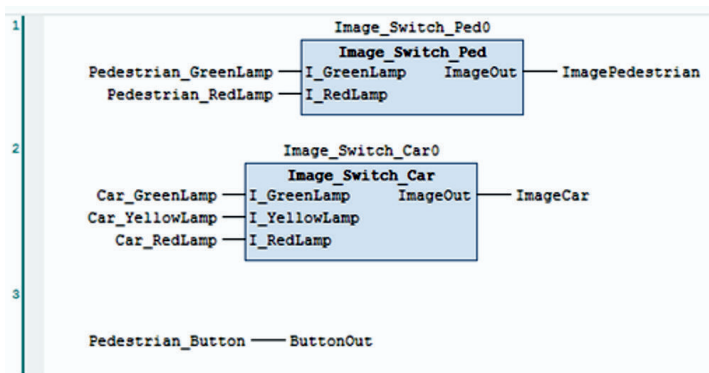


Рис. 4.15 – Програма світлофору

У програмі є присутніми два призначені для користувача функціональні блоки Image_Switch_Ped і Image_Switch_Car. Вони дають можливість змінювати зображення світлофорів, залежно від вхідних сигналів.

Функціональний блок Image_Switch_Ped призначений для визначення вхідного зображення пішохідного світлофора. Він реалізований на мові ST і містить код, представлений нижче.

Поле оголошення змінних:

```
FUNCTION_BLOCK Image_Switch_Ped
VAR_INPUT
    I_GreenLamp : BOOL;
    I_RedLamp : BOOL;
END_VAR
VAR_OUTPUT
    ImageOut: STRING;
END_VAR
VAR
END_VAR
```

Основний текст програми:

```
IF((I_GreenLamp = 1) AND (I_RedLamp = 1)) THEN
    ImageOut:= 'sp4';
ELSIF ((I_GreenLamp = 0) AND (I_RedLamp = 1)) THEN
    ImageOut:= 'sp2';
ELSIF ((I_GreenLamp = 1) AND (I_RedLamp = 0)) THEN
    ImageOut:= 'sp3';
ELSE
    ImageOut:= 'sp1';
END_IF
```

Цей функціональний блок за допомогою операторів IF...ELSIF...END_IF, залежно від вхідних сигналів I_GreenLamp і I_RedLamp, встановлює значення ImageOut, яке містить назву поточного малюнка з глобального списку зображень (див. нижче).

Функціональний блок Image_Switch_Car виконує ту ж функцію, що і Image_Switch_Ped, тільки для світлофора машин. Він також виконаний на мові ST, але у своїй структурі містить ще оператор SWITCH. Код функціонального блоку представлений нижче.

Код оголошення змінних:

```
FUNCTION_BLOCK Image_Switch_Car
VAR_INPUT
    I_GreenLamp : BOOL;
    I_YellowLamp : BOOL;
```

```

    I_RedLamp : BOOL;
END_VAR
VAR_OUTPUT
    ImageOut: STRING;
END_VAR
VAR
    TmpCounter: INT;
END_VAR

```

Основний текст функціонального блоку :

```

    TmpCounter:= 0;
    IF (I_GreenLamp = 1) THEN
        TmpCounter:= TmpCounter + 1;
    END_IF
    IF (I_YellowLamp = 1) THEN
        TmpCounter:= TmpCounter + 2;
    END_IF
    IF (I_RedLamp = 1) THEN
        TmpCounter:= TmpCounter + 4;
    END_IF
    CASE TmpCounter OF
    0:
        ImageOut:= 'sc1';
    1:
        ImageOut:= 'sc4';
    2:
        ImageOut:= 'sc3';
    3:
        ImageOut:= 'sc7';
    4:
        ImageOut:= 'sc2';
    5:
        ImageOut:= 'sc6';
    6:
        ImageOut:= 'sc5';
    7:
        ImageOut:= 'sc8';
    ELSE
        ImageOut:= 'sc1';
    END_CASE

```

Також вимагається створити глобальний список змінних, в якому має бути оголошено вхідні змінні. Його створення здійснюється за допомогою вибору Add object → Global Variable List. у контекстному меню Application або у меню Project.

У вікні, що з'явилося, слід оголосити наступні змінні:

```
VAR_GLOBAL
    Car_RedLamp : BOOL;
    Car_YellowLamp : BOOL;
    Car_GreenLamp : BOOL;
    Pedestrian_Button : BOOL;
    Pedestrian_RedLamp : BOOL;
    Pedestrian_GreenLamp : BOOL;
END_VAR
```

Створення візуалізації

У цьому прикладі буде тільки одна візуалізація, представлена на рис. 4.14.

Ця візуалізація заснована на використанні об'єкту Image. Цей елемент прив'язується до строкової змінної, що визначає назву малюнка з ImagePool, який повинен відображатися в даний момент.

Насамперед слід створити ImagePool, де зберігатимуться усі зображення для проекту. Для цього в контекстному меню Application або меню Project слід вибрати Add object → Image Pool. У вікні, що з'явилося, слід ввести назви цього елемента і натиснути ОК. Після чого відкриється порожній лістинг зображень.

Тут всього 3 поля для введення, з них:

- ID відповідає за назву цього зображення усередині проекту;
- File name містить в собі фізичний шлях до файлу і його реальну назву;
- Image показує мініатюру зображення.

Для додавання зображення в список слід записати шлях в полі File name, або вибрати його на комп'ютері за допомогою діалогового вікна. Також можна вибрати, чи буде файл скопійований в теку з проектом або ні.

Для цього проекту повинен бути список, представлений на рис. 4.16. Зображення для проекту можна буде знайти за адресою WWW.

Після створення ImagePool можна приступати до створенні самої візуалізації.

При створенні елементу Image, слід вибрати початкове зображення, яке відображатиметься за умовчанням. Потім у вкладці «Properties» в списку «Bitmap ID variable» полю Bitmap ID слід присвоїти ім'я змінної, яка міститиме назву поточного малюнка.

ID	File name	Image
sc1	sc1.png	
sc2	sc2.png	
sc3	sc3.png	
sc4	sc4.png	
sc5	sc5.png	
sc6	sc6.png	
sc7	sc7.png	
sc8	sc8.png	
sp1	s1.png	
sp2	s2.png	
sp3	s3.png	
sp4	s4.png	

Рис. 4.16 – Список зображень для проекту

Підключення фізичних входів-виходів

У вікні Berghof_IO, слід розставити виводи таким чином – лампи світлофорів слід підключати до цифрових входів (Digital In) контролера (рис. 4.17); кнопку слід підключити до цифрових виходів (Digital Out) контролера (рис. 4.18).

Variable	Mapping	Channel	Address	Type
Digital In		Digital In...	%IB0	BYTE
Application.Car_GreenLamp		Bit0	%IX0.0	BOOL
Application.Car_YellowLamp		Bit1	%IX0.1	BOOL
Application.Car_RedLamp		Bit2	%IX0.2	BOOL
Application.Pedestrian_GreenLamp		Bit3	%IX0.3	BOOL
Application.Pedestrian_RedLamp		Bit4	%IX0.4	BOOL
		Bit5	%IX0.5	BOOL
		Bit6	%IX0.6	BOOL
		Bit7	%IX0.7	BOOL
		Digital In...	%IB1	BYTE
		Digital In...	%IB2	BYTE
		Digital In...	%IB3	BYTE
		Digital In...	%IB4	BYTE

Рис. 4.17 – Підключені змінні до цифрових входів контролера

Variable	Mapping	Channel	Address	Type
Digital Out		Digital Out...	%QB0	BYTE
ButtonOut		Bit0	%QX0.0	BOOL
		Bit1	%QX0.1	BOOL
		Bit2	%QX0.2	BOOL
		Bit3	%QX0.3	BOOL
		Bit4	%QX0.4	BOOL
		Bit5	%QX0.5	BOOL
		Bit6	%QX0.6	BOOL
		Bit7	%QX0.7	BOOL

Рис. 4.18 – Підключені змінні до цифрових виходів контролера

Як можна помітити на рис. 4.18, змінна ButtonOut є змінною, створеною в цьому вікні, про що свідчить знак в полі Mapping. Таким чином також можна створювати змінні, не виводячи їх в глобальний список змінних.

Після підключення усіх змінних програму слід відкомпілювати, упевнитися, що помилок немає, а потім завантажити програму у контролер.

Успішно завантажена програма відразу ж чекатиме від користувача старту додатка.

Розділ 5

ЛАБОРАТОРНИЙ ПРАКТИКУМ

В цьому розділі будуть наведені та досконально розібрані базові завдання до лабораторних робіт.

5.1 Знайомство с середою розробки програмного забезпечення промислових контролерів

Мета роботи

Практичне закріплення знань про розробку ПЗ ПЛК у середовищі CODESYS. Створення та конфігурування проекту. Програмування на мовах FBD, IL, SFC та CFC, застосування стандартних та бібліотечних компонентів, взаємодії POU, налагоджування ПЗ.

Методичні вказівки з організації самостійної роботи студентів

Під час підготовки до цієї лабораторної роботи слід передивитися розділи конспекту лекцій, що присвячені мовам FBD, IL, SFC та CFC, застосуванню стандартних та бібліотечних компонентів.

Опис лабораторної установки

При проведенні лабораторної роботи використовується персональний комп'ютер з встановленим пакетом САПР ПЗ ПЛК CODESYS.

Порядок виконання роботи і методичні вказівки з її виконання

У цій роботі необхідно створити програму керування рухом на перехресті, що має світлофори для двох пересічних напрямків руху. Світлофори повинні мати два протилежних стани – червоний і зелений. Щоб уникнути нещасних випадків, слід додати загальноприйняті перехідні стадії: жовтий і жовто-червоний. Остання стадія повинна бути довше попередньої.

Виконуючи лабораторну роботу слід з'ясувати: як керовані часом процеси можна представити засобами мов стандарту МЭК 61131-3 та як можна комбінувати різні мови в CODESYS, а також ознайомитись із засобами моделювання в CODESYS.

Створення компонентів організації програм (program organization unit, далі – POU)

Запустити CODESYS і вибрати «File» «New».

У вікні діалогу визначимо перший POU. За замовчуванням він одержує найменування PLC_PRG. Не змінюйте його. Тип цього POU, безумовно, повинен бути – програма. Кожний проект повинен мати програму з таким ім'ям. Як мову програмування даного POU ми виберемо мову Continuous Function Chart (CFC).

Тепер створіть ще три об'єкти. Скористайтесь командою «Project» «Object Add» у системному або в контекстному (натисніть праву кнопку миші у вікні Object Organizer) меню. Створіть: програму на мові Sequential Function Chart (SFC) з ім'ям SEQUENCE, функціональний блок мовою Function Block Diagram (FBD) з ім'ям TRAFFICSIGNAL і ще один аналогічний блок – WAIT, що ми будемо описувати мовою Список Інструкції (IL).

В POU TRAFFICSIGNAL ми зіставимо певні стадії процесу відповідним кольорам. Тобто ми впевнимися, що червоне світло запалене в червоній стадії й у жовто-червоній стадії, жовте світло в жовтій і жовто-червоній стадії й т.д.

В WAIT ми створимо таймер, що на вхід одержує довжину стадії в мілісекундах і на виході видає стан ІСТИНА після закінчення заданого періоду часу.

В SEQUENCE усе буде об'єднано так, щоб потрібні вогні запалювалися в правильний час і на необхідний період часу.

В PLC_PRG уводиться вхідний сигнал включення, що дозволяє початок роботи світлофора й «колірні команди» кожної лампи пов'язані з відповідними виходами апаратури.

Оголошення «TRAFFICSIGNAL»

Повернемося тепер до POU TRAFFICSIGNAL. У редакторі оголошень визначте вхідну змінну (між ключовими словами VAR_INPUT і END_VAR) по імені STATUS типу INT. STATUS буде мати чотири можливих стани, що визначають відповідні стадії – зелена, жовта, жовто-червона й червона.

Оскільки наш POU TRAFFICSIGNAL має три виходи, потрібно визначити ще три змінних RED, YELLOW і GREEN. Тепер розділ оголошень блоку TRAFFICSIGNAL повинен виглядати згідно рисунку 5.1.

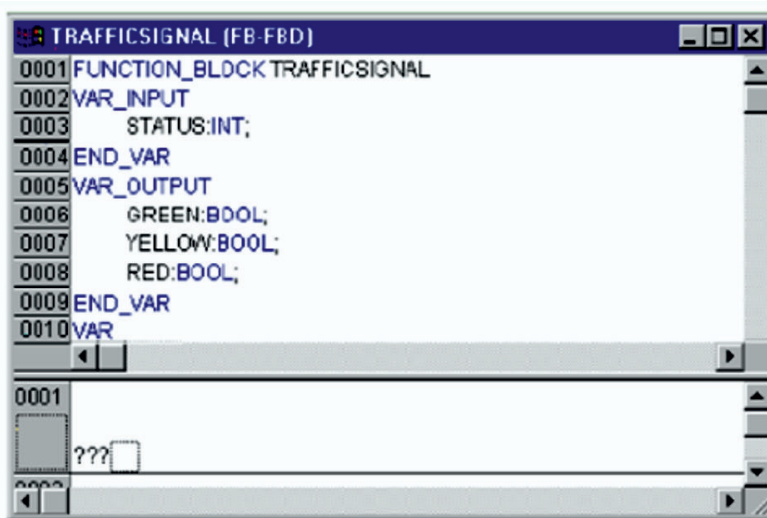
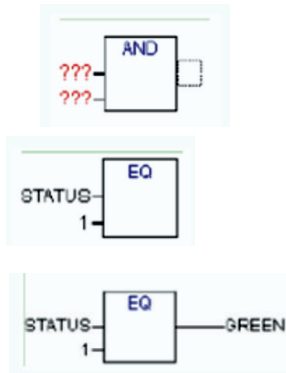


Рис. 5.1 – Розділ оголошень POU TRAFFICSIGNAL

Програмуємо «TRAFFICSIGNAL»

Тепер настав час описати, що зв'язує вхід STATUS з вихідними змінними. Для цього перейдіть в розділ реалізації POU. Клікніть на поле ліворуч від першого ланцюга (сіра область із номером 1). Ви обрали перший ланцюг. Тепер оберіть команду меню «Insert» «Operator».



У першому ланцюзі буде вставлений прямокутник з оператором AND і двома входами. Клікніть мишкою на тексті AND і замініть його на EQ.

Три знаки питання біля верхнього із двох входів змініть на ім'я змінної STATUS. Для нижнього входу замість трьох знаків питання потрібно поставити «1». У результаті Ви повинні одержати наступний елемент.

Клікніть тепер на місці за прямокутником EQ. Тепер обраний вихід EQ. Виконайте команду меню «Insert» «Assignment». Замініть три питання ??? на GREEN. Ви тепер створили ланцюг наступного виду.

STATUS зрівнюється з 1, результату привласнюється GREEN. Таким чином, GREEN буде увімкнено, коли STATUS дорівнюватиме 1.

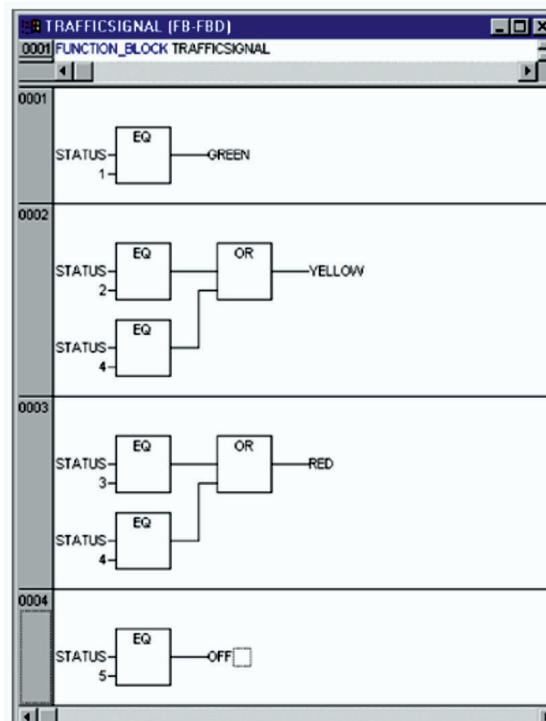


Рис. 5.2 – Функціональний блок TRAFFICSIGNAL

Для інших кольорів TRAFFICSIGNAL нам знадобляться ще два ланцюги. Створіть їх командою «Insert» «Network (after)». Закінчений POU повинен виглядати в такий спосіб.

Щоб вставити оператор перед входом іншого оператора, Ви повинні виділити сам вхід, а не текст (виділяється прямокутником). Далі використайте команду «Insert» «Operator».

Тепер наш перший POU завершено. Як і планувалося, TRAFFICSIGNAL буде управляти включенням виходів, керуючись значенням змінної STATUS.

Підключення standard.lib

Для створення таймера в POU WAIT нам знадобиться POU зі стандартної бібліотеки. Отже, відкрийте менеджер бібліотек командами «Window» «Library Manager». Оберіть «Insert» «Additional library». Повинне відкритися діалогове вікно вибору файлів. Оберіть standard.lib зі списку бібліотек.

Оголошення POU «WAIT»

Тепер давайте повернемося до POU WAIT. Як передбачалося, цей POU буде працювати таймером, що задає тривалість стадій TRAFFICSIGNAL. Наш POU повинен мати вхідну змінну TIME_IN типу TIME і генерувати на виході двійкову (Boolean) змінну, котру ми назвемо OK. Ця змінна повинна приймати значення TRUE, коли бажаний період часу закінчено. Попередньо ми встановлюємо цю змінну в FALSE наприкінці рядка оголошення (але до крапки з комою) «:= FALSE».

Тепер нам потрібен генератор часу POU TP. Він має два входи (IN, PT) і два виходи (Q, ET). TP робить наступне: Поки IN установлений в FALSE, ET буде 0 і Q буде FALSE. Як тільки IN перемкнеться в TRUE, вихід ET почне відраховувати час у мілісекундах. Коли ET досягне значення заданого PT, рахунок буде зупинений. Тим часом вихід Q дорівнює TRUE, поки ET менше PT. Як тільки ET досягне значення PT, вихід Q знову перемкнеться в FALSE.

Щоб використати TP в POU WAIT, ми повинні створити його локальний екземпляр. Для цього ми оголосимо локальну змінну ZAB (відлічений час) типу TP (між ключовими словами VAR, END_VAR).

Розділ оголошень WAIT тепер повинен виглядати як показано на рисунку 5.3

Програмуємо «WAIT»

Для створення бажаного таймера текст програми повинен бути як показано на рисунку 5.4

Спочатку перевіряється, чи встановлений Q в TRUE (можливо, відлік уже запущений), у цьому випадку ми не торкаємо установки ZAB, а викликаємо функціональний блок ZAB без вхідних змінних – щоб перевірити, чи завершено період часу. Інакше ми встановлюємо змінну IN ZAB в FALSE і одночасно ET в 0 і Q в FALSE. Таким чином, всі змінні встановлені в початковий стан. Тепер ми встановлюємо необхідний час TIME змінної PT і викликаємо ZAB з IN:=TRUE.

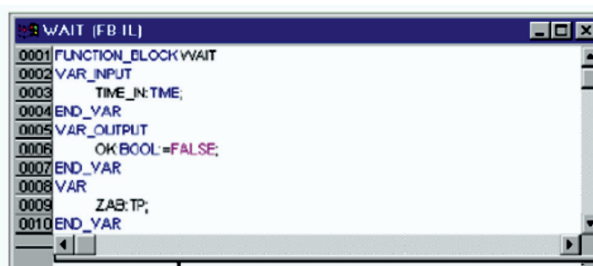


Рис. 5.3 – Розділ оголошень WAIT

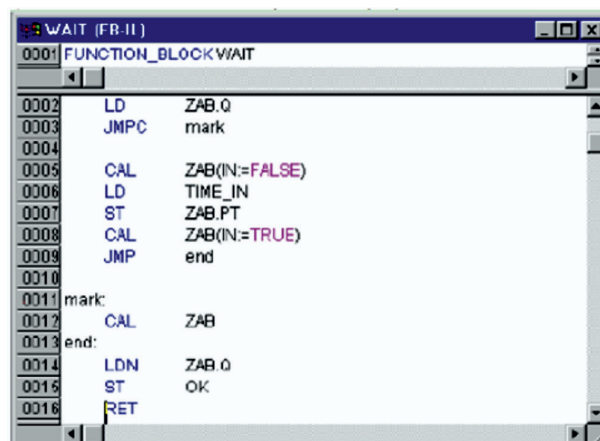


Рис. 5.4 – Створення таймера

Функціональний блок ZAB тепер буде працювати, доки не досягне значення TIME і не встановить FALSE. Інвертоване значення Q буде зберігатися в змінній OK після кожного виконання WAIT. Як тільки Q стане FALSE, OK прийме значення TRUE.

POU «SEQUENCE» версія 1

Спочатку оголосимо необхідні змінні. Це вхідна змінна START типу BOOL, дві вихідні змінні TRAFFICSIGNAL1 і TRAFFICSIGNAL2 типу INT і одна – DELAY типу WAIT. Програма SEQUENCE тепер виглядає як зображено на рисунку 5.5

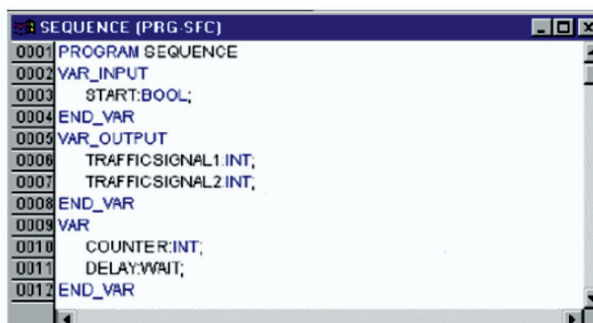


Рис. 5.5 – Програма SEQUENCE версія 1, розділ оголошень

Створюємо SFC діаграму

Спочатку SFC граф завжди складається з етапу «Init» переходу «Trans0» і повернення назад до Init, звісно, нам потрібно трохи доповнити його.

Перш ніж програмувати конкретні етапи й переходи, вибудуємо структуру графа. Спочатку нам знадобляться етапи для кожної стадії TRAFFICSIGNAL. Вставте їх, відзначаючи Trans0 і вибираючи команди «Insert» «Step transition (after)». Повторіть цю процедуру ще три рази.

Для редагування назви переходу або етапу потрібно просто клікнути мишкою на потрібному тексті.

Назвіть перший перехід після Init «START», а всі інші переходи «DELAY.OK».

Перший перехід дозволяється, коли START устанавлюється в TRUE, а інші – коли DELAY в OK стане TRUE, тобто коли заданий період закінчиться.

Етапи (зверху вниз) одержують імена Switch1, Green2, Switch2, Green1, ну й Init, звичайно, збереже своє ім'я. «Switch» повинен включати жовту фазу, в Green1 TRAFFICSIGNAL1 буде зеленим, в Green2 TRAFFICSIGNAL2 буде зеленим. Нарешті, замініть адресу повернення Init на Switch1. Якщо ви все зробили вірно, діаграма повинна виглядати як зображено на рисунку 5.6.

Тепер ми повинні запрограмувати етапи. Якщо ви зробите подвійного щиглика мишею на зображенні етапу, то повинен відкритися діалог визначення нової дії. У нашому випадку ми будемо використовувати мову IL (Список Інструкцій).

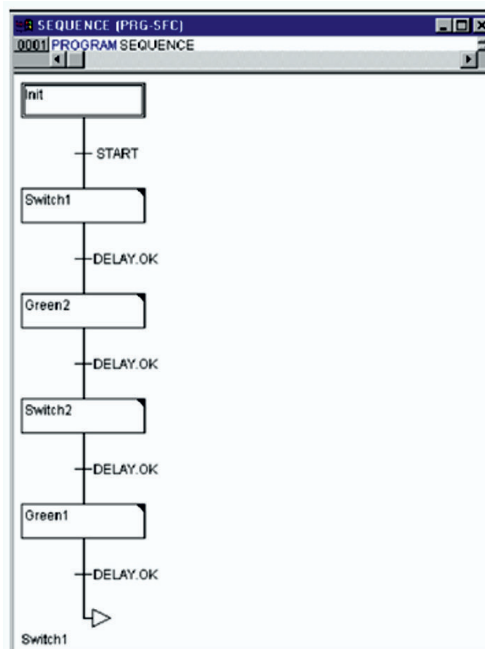


Рис. 5.6 – Програма SEQUENCE версія 1, розділ інструкцій

Під час дії етапу Init перевіряємо, активний сигнал включення START чи ні. Якщо сигнал не активний, то світлофор вимикається. Цього можна досягти, якщо записати в змінні TRAFFICSIGNAL1 і TRAFFICSIGNAL2 число 5.

Етап Init повинен виглядати як зображено на рисунку 5.7.

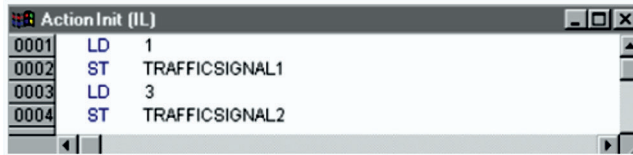


Рис. 5.7 – Етап Init

В Green1 TRAFFICSIGNAL1 буде зеленим (STATUS:=1), TRAFFICSIGNAL2 буде червоним.

(STATUS:=3), затримка в 5000 мілісекунд.

Етап Green1 повинен виглядати як зображено на рисунку 5.8.



Рис. 5.8 – Етап Green1

Switch1 змінює стан TRAFFICSIGNAL1 на 2 (жовте) і, відповідно, TRAFFICSIGNAL2 на 4 (жовто-червоне). Крім того, тепер встановлюється затримка в 2000 мілісекунд. Це виглядає так:

Етап Switch1 повинен виглядати як зображено на рисунку 5.9.



Рис. 5.9 – Етап Switch1

Green2 включає червоний в TRAFFICSIGNAL1 (STATUS:=3) і зелений в TRAFFICSIGNAL2 (STATUS:=1). Затримка встановлюється в 5000 мілісекунд.

Етап Green2 повинен виглядати як зображено на рисунку 5.10.

В Switch2 STATUS в TRAFFICSIGNAL1 змінюється на 4 (жовто-червоний), відповідно, TRAFFICSIGNAL2 буде 2 (жовтий). Затримка тепер повинна бути в 2000 мілісекунд.

Етап Switch2 повинен виглядати як зображено на рисунку 5.11.

Перша версія програми закінчена.



Рис. 5.10 – Eman Green2



Рис. 5.11 – Eman Switch2

Перевірка роботи ПЗ

Для того щоб перевірити роботу ПЗ в режимі емуляції, зробіть наступне. Відкрийте POU PLC_PRG. Кожний проект починає роботу з PLC_PRG. Вставте в нього компонент і замініть «AND» на «SEQUENCE». Входи й виходи залиште поки вільними.

Тепер ви можете відкомпілювати проект («Project» «Build») і перевірити відсутність помилок. У вікні повідомлень ви повинні побачити текст: «0 Errors, 0 Warnings».

Тепер увімкніть прапорець «Online» «Simulation» і запустіть команду «Online» «Login». Запустіть програму «Online» «Run».

Відкрийте програму SEQUENCE. Програма запущена, але не працює, оскільки змінна START повинна мати значення TRUE. Далі це буде робити PLC_PRG, але зараз ви можете змінити її вручну. Для цього клікніть двічі мишею по оголошенню цієї змінної. Її значення тепер виділено кольорами й дорівнює TRUE. Оберіть команду запису значень змінних («Online» «Write values»).

Тепер ви можете спостерігати за роботою програми. Активні кроки діаграми виділяються блакитним кольором.

Для продовження редагування програми закрийте режим онлайн командою «Online» «Logout».

Контрольні запитання і завдання

1. Надайте визначення ПЛК. Розкрийте суть наступних понять: входи-виходи, режим реального часу й обмеження на застосування ПЛК, умови роботи ПЛК, інтеграція ПЛК у систему керування підприємством.
2. Що таке робочий цикл, час реакції, будова ПЛК, системне й прикладне програмне забезпечення?
3. Охарактеризуйте поняття: типи даних, елементарні типи даних, цілочисельні типи, логічний тип, дійсні типи.

4. Як використовуються: інтервал часу, час доби й дата, рядка, ієрархія елементарних типів, призначені для користування типи даних, перетворення типів даних?
5. Що таке масиви, структури, перерахування, обмеження діапазону, псевдоніми типів, змінні, ідентифікатори?
6. В чому полягають: специфіка реалізації типів даних CODESYS, розподіл пам'яті змінних, пряма адресація, порозрядна адресація, перетворення типів, тонкощі обчислень?
7. Надайте визначення: компоненту організації програм (POU), оголошенню POU, формальним та актуальним параметрам, параметрам та змінним компонентів.
8. Що таке функції, функції зі змінним числом параметрів, функції в логічних виразах, оператори? Як здійснюється виклик функції з надання значень параметрам, присвоєння значень параметрам функції, обмеження можливостей функції, перевантаження функцій і операторів? Наведіть приклади функцій і операторів.
9. Дайте визначення функціональним блокам. Як реалізуються: створення екземпляра функціонального блоку, доступ до змінних екземпляра, виклик екземпляра блоку, ініціалізація даних екземпляра, тиражування екземплярів, особливості створення й застосування функціональних блоків, шаблонні змінні?
10. Що таке програми та як вони використовуються? Що таке компоненти в CODESYS?

5.2 Розробка інтерфейсу оператора для ПЗ ПЛК

Мета роботи

Практичне закріплення знань про програмування на мовах FBD, IL, SFC та CFC, взаємодії POU, налагоджування ПЗ, побудови інтерфейсу оператора (візуалізації).

Методичні вказівки з організації самостійної роботи студентів

Під час підготовки до цієї лабораторної роботи слід передивитися розділи конспекту лекцій, що присвячено мовам FBD, IL, SFC та CFC, застосуванню стандартних та бібліотечних компонентів, побудови інтерфейсу оператора (візуалізації).

Опис лабораторної установки

При проведенні лабораторної роботи використовується персональний комп'ютер з встановленим пакетом САПР ПЗ ПЛК CODESYS.

Порядок виконання роботи і методичні вказівки з її виконання

В цій роботі продовжуємо розробку проекту, що ми почали у попередній лабораторній роботі.

Зміни у SEQUENCE. Тепер трохи ускладнимо нашу програму. Доречним буде вимикати наші світлофори на ніч. Для цього слід створити в програмі

лічильник, що після деякого числа циклів TRAFFICSIGNAL вимкне пристрій.

Для початку нам потрібна нова змінна COUNTER типу INT. Оголосить її як звичайно в розділі оголошень SEQUENCE.

Тепер оберіть перехід після Switch1 і вставте ще один етап і перехід. Оберіть результуючий перехід і вставте альтернативну галузь праворуч. Після лівого переходу вставте додатковий етап і перехід. Після нового результуючого переходу вставте вилучений перехід (jump) на Init.

Назвіть нові частини так: верхній із двох нових етапів потрібно назвати «Count» і нижній – «Off». Переходи будуть називатися (зверху вниз, зліва направо) EXIT, TRUE і DELAY.OK. Тепер нові частини повинні виглядати як фрагмент, виділений рамкою.

Програма «SEQUENCE», розділ інструкцій повинен виглядати як зображено на рисунку 5.12.

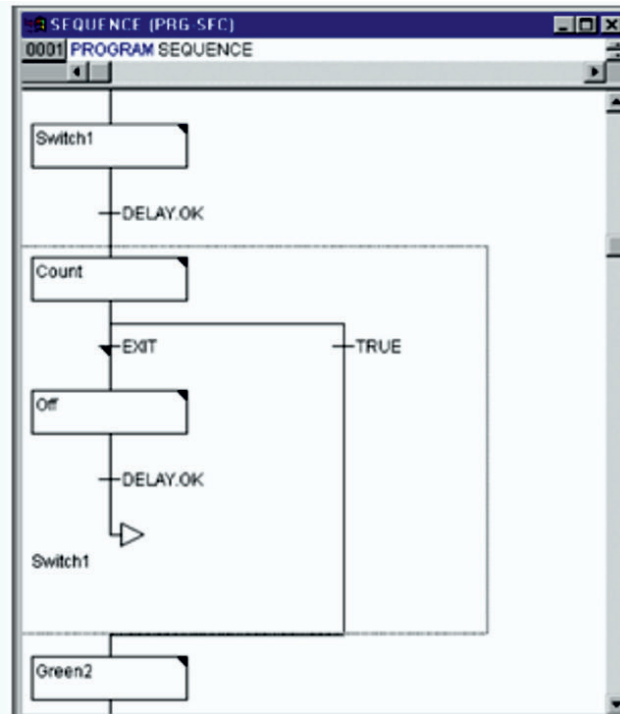


Рис. 5.12 – Розділ інструкцій програми «SEQUENCE»

Тепер два нових етапи й перехід необхідно наповнити змістом.

На етапі Count виконується тільки одна дія – COUNTER інкрементується (рис. 5.13).

На переході EXIT1 перевіряється досягнення лічильником заданого значення, наприклад 7 (рис. 5.14).

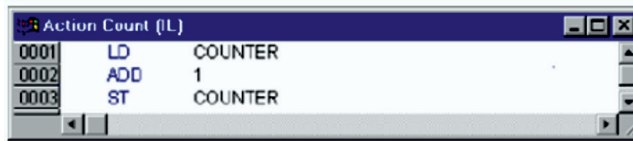


Рис. 5.13 – Eman Count

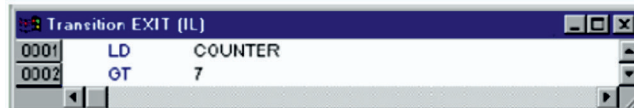


Рис. 5.14 – Перехід EXIT1

На етапі Off стан обох світлофорів установлюється в 5 (світлофор вимкнено), COUNTER скидається в 0 і встановлюється затримка часу в 10 секунд (рис. 5.15).

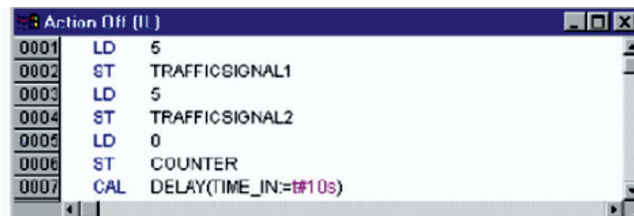


Рис. 5.15 – Eman Off

У нашій гіпотетичній ситуації ніч настає після семи циклів TRAFFICSIGNAL. Світлофори повністю вимикаються до світанку, і процес повторюється знову. При бажанні ви можете ще раз перевірити роботу програми в емуляторі, перш ніж продовжити її вдосконалення.

PLC_PRG. Ми визначили два строго корельовані у часі світлофора в блоці SEQUENCE. Тепер повністю закінчимо програму. Для цього необхідно розподілити вхідні й вихідні змінні в блоці PLC_PRG. Ми хочемо дати можливість запустити систему вимикачем IN і хочемо забезпечити перемикання всіх шести ламп (2 світлофори) шляхом передачі «команд перемикання» на кожному кроці SEQUENCE. Оголосимо тепер відповідні Boolean змінні для всіх шести виходів і одного входу, потім створимо програму й зіставимо змінні відповідного IEC адресам.

Наступний крок – це оголошення змінних LIGHT1 і LIGHT2 типу TRAFFICSIGNAL у розділі оголошень.

Розділ оголошень PLC_PRG повинен виглядати як зображено на рисунку 5.16.

Для подання шести ламп світлофорів потрібно 6 змінних типу Boolean. Однак ми не будемо повідомляти їх у розділі оголошень блоку PLC_PRG, замість цього використаємо глобальні змінні (Global Variables) з ресурсів (Resources). Двійкова вхідна змінна IN, необхідна для встановлення змінної START блоку SEQUENCE в TRUE, буде визначена в такий же спосіб. Виберіть вкладку Resources і відкрийте



Рис. 5.16 – Розділ оголошення глобальних змінних PLC_PRG

те список Global Variables. Розділ оголошення глобальних змінних PRG повинен виглядати згідно рисунку 5.17.

Закінчимо PLC_PRG. Для цього ми перейдемо у вікно редактора. Ми обрали редактор Continuous Function Chart, отже, нам доступна відповідна панель інструментів. Клікніть правою клавішею миші у вікні редактора й оберіть елемент Вох. Клікніть на тексті AND і напишіть «SEQUENCE». Елемент автоматично перетвориться в SEQUENCE із уже певними вхідними й вихідними змінними.

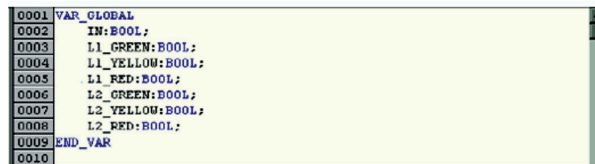


Рис. 5.17 – Розділ оголошення глобальних змінних PRG

Вставте далі два елементи й назвіть їх TRAFFICSIGNAL. TRAFFICSIGNAL – це функціональний блок, і, як звичайно, Ви одержите три червоних знаки питання, які потрібно замінити вже оголошеними локальними змінними LIGHT1 і LIGHT2.

Тепер створіть елемент типу Input, що одержить назву IN і шість елементів типу Output яким потрібно дати наступні імена: L1_green, L1_yellow, L1_red, L2_green, L2_yellow, L2_red.

Всі елементи програми тепер на місці і Ви можете з'єднувати входи й виходи. Для цього клікніть мишею на короткій лінії входу/виходу й тягніть її (не відпускаючи клавішу миші) до входу/виходу потрібного елемента. Програма повинна прийняти вигляд, показаний на рисунку 5.18.

Після цього наша програма повністю готова.

Перевірка роботи ПЗ

Тепер перевірте остаточно вашу програму в режимі емуляції. Переконайтеся в правильності її роботи, контролюючи послідовність виконання й значення змінних у вікнах редакторів CODESYS.

Побудова інтерфейсу оператора

За допомогою візуалізації можна швидко й легко поживити змінні проекту. Повний опис візуалізації дещо складніший, але зараз ми намалюємо два світло-

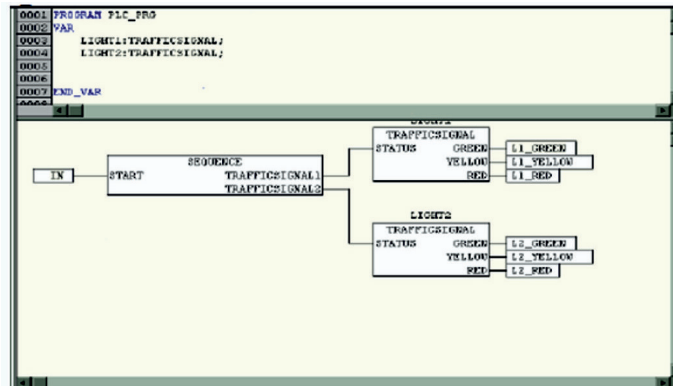


Рис. 5.18 – Робоча програма

фори і їхній вимикач, що дозволить нам вмикати й вимикати блок керування світлофором.

Для того щоб створити візуалізацію, оберіть вкладку Visualizations в організаторі об'єктів. Тепер виконайте команду «Project» «Object Add».

Діалог для створення нової візуалізації зображено на рисунку 5.19.



Рис. 5.19 – Створення нової візуалізації

Уведіть будь-яке ім'я для візуалізації, наприклад Lights. Коли Ви натиснете кнопку Ok, відкриється вікно, у якому ви будете створювати візуалізацію.

Для створення візуалізації світлофора слід виконати наступні дії.

Оберіть команду «Insert» «Ellipse» і намалюйте коло із діаметром близько 2 сантиметрів. Для цього клікніть мишею на робочому полі й, утримуючи ліву кнопку миші, розтягніть коло, що з'явилося, до необхідного розміру.

Двічі клікніть мишею на колі. З'явиться діалогове вікно для налаштування елемента візуалізації.

Оберіть категорію Variables і в поле Change color уведіть ім'я змінної L1_red. Уводити ім'я змінної зручно за допомогою Input Assistant (клавіша <F2>). Глобальна змінна L1_red буде управляти кольорами намальованого Вами кола. Діалог для завдання змінної зображено на рисунку 5.20.

Оберіть категорію Color. В області Color натисніть кнопку Inside і у вікні, що з'явилося, оберіть який завгодно нейтральний колір, наприклад, чорний.

Натисніть кнопку Inside в області Alarm Color і виберіть червоні кольори. Діалог для вибору кольору елемента візуалізації зображено на рис. 5.21.

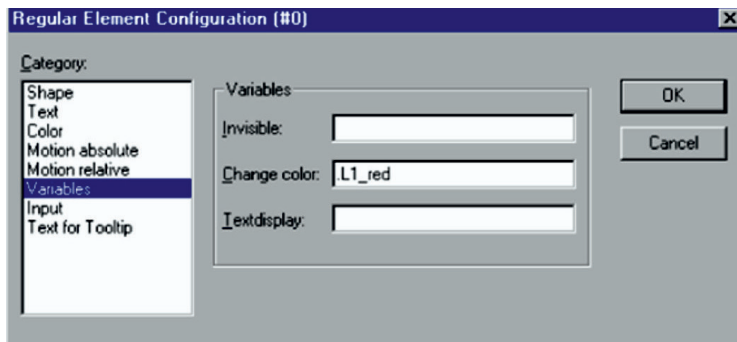


Рис. 5.20 – Діалогове вікно

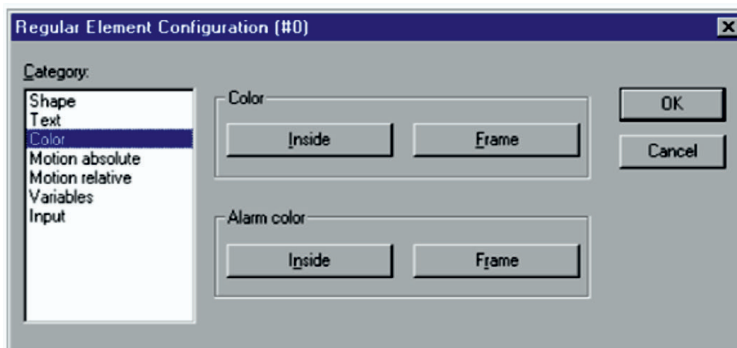


Рис. 5.21 – Вибір кольору елементів візуалізації

Отримане коло буде чорним, коли значення змінної «неправдиве», і червоним, коли змінна «істинна».

Таким чином, ми створили перший ліхтар першого світлофора.

Тепер викличте команду копіювання «Edit» «Copy» (<Ctrl>+<C>) і двічі виконайте команду вставки «Edit» «Paste» (<Ctrl>+<V>). Ви отримаєте два нових кола. Переміщати ці кола можна за допомогою мишки. Розташуйте їх так, щоб вони являли собою вертикальний ряд у лівій частині вікна редактора. Подвійний щиклик по колу приводить до відкриття вікна для налаштування властивостей елемента візуалізації. У поле Change Color вікон налаштування властивостей відповідних кіл уведіть наступні змінні:

- для середнього кола: .L1_yellow;
- для нижнього кола: .L1_green.

У категорії Color в області Alarm color установіть кольори кіл (жовтий і зелений).

Тепер викличте команду «Insert» «Rectangle» і вставте прямокутник так, щоб уведені раніше кола перебували всередині нього. Оберіть кольори прямокутника й потім виконайте команду «Extras» «Send to back», що перемістить його на задній план. Після цього кола знову буде видно.

Активізуйте режим емуляції, виконавши команду «Online» «Simulation» (режим емуляції активний, якщо перед пунктом меню «Online» «Simulation» стоїть галочка).

Запустіть програму шляхом виконання команд «Online» «Login» і «Online» «Run» і ви побачите, як будуть мінятися кольори світлофора.

Найпростіший спосіб створити другий світлофор – скопіювати всі елементи першого. Виділіть елементи першого світлофора й скопіюйте їх, виконавши команди «Edit» «Copy» і «Edit» «Paste». Замініть імена змінних, що керують кольорами (наприклад, .L1_red на .L2_red), і другий світлофор буде готовий.

Як описано вище, вставте прямокутник, установіть його кольори й уведіть змінну .ON у поле Change Color категорії Variables. У поле Content категорії Text уведіть ім'я «ON». Діалог для конфігурування елементу візуалізації, розділ текст, зображено на рисунку 5.22.

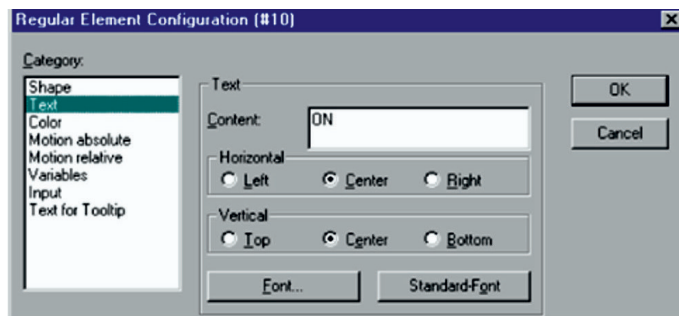


Рис. 5.22 – Конфігурування елементів візуалізації розділу тексту

Для того щоб змінна ON перемикалася при кліку мишкою на цьому елементі, у полі Toggle variable категорії Input уведіть змінну .ON. Створений нами перемикач буде увімкати/вимикати світлофори.

Відобразити включений стан можна кольорами, як і для світлофора. Впишіть змінну в поле Change Color.

Діалог для конфігурування елементу візуалізації, розділ ввід, зображено на рисунку 5.23.

Під світлофорами вставимо два прямокутники. У властивостях елемента в категорії Color кольори межі (frame) прямокутника задайте білим. У поле Contents (категорія Text) уведіть назви світлофорів «Light1», «Light2».

Кінцевий вигляд візуалізації проекту зображено на рисунку 5.24. Тепер потрібно перевірити роботу в режимі емуляції.

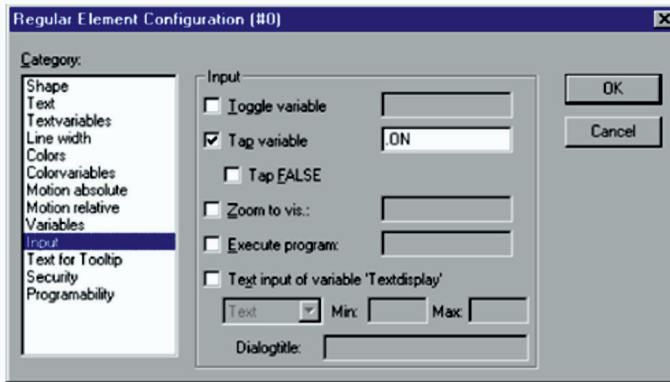


Рис. 5.23 – Конфігурування елементів візуалізації розділу вводу

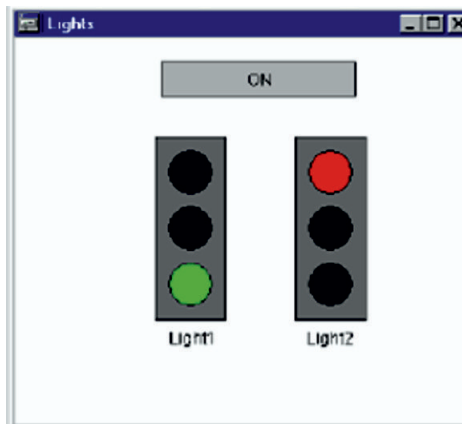


Рис. 5.24 – Вигляд робочої програми

Контрольні запитання і завдання

1. Що включає сімейство мов МЕК?
2. Поясніть наступні поняття: мова лінійних інструкцій (ІЛ), формат інструкції, акумулятор, перехід на мітку, модифікатори, оператори.
3. Дайте визначення поняттям: виклик функціональних блоків і програм, виклик функції, коментування тексту, ІЛ у режимі виконання.
4. Що таке структурований текст (ST), вираження, порядок обчислення виражень, порожнє вираження, оператор вибору IF, оператор множинного вибору CASE?
5. Як використовуються цикли WHILE і REPEAT, цикл FOR, переривання ітерацій операторами EXIT і RETURN, ітерації на базі робочого циклу ПЛК, оформлення тексту?
6. Поясніть, у чому полягає суть релейних діаграм (LD), реле із самофіксацією, порядку виконання й зворотнього зв'язку, керування порядком виконання, розширення можливостей LD, LD-діаграм в режимі виконання та особливостей реалізації LD в CODESYS.

5.3 Застосування бібліотечних компонентів

Мета роботи

Практичне закріплення знань про принципи побудови ПЗ ПЛК, програмування на мовах FBD та SFC, застосування стандартних та бібліотечних компонентів.

Методичні вказівки з організації самостійної роботи студентів

Під час підготовки до цієї лабораторної роботи слід передивитися розділи конспекту лекцій, що присвячено мовам FBD та SFC, застосуванню стандартних та бібліотечних компонентів.

Опис лабораторної установки

При проведенні лабораторної роботи використовується персональний комп'ютер з встановленим пакетом САПР ПЗ ПЛК CODESYS.

Порядок виконання роботи і методичні вказівки з її виконання

Завдання: Реалізувати контроль оператором руху деякого механізму. Оператор повинен періодично підтверджувати правильність функціонування механізму. У противному випадку, необхідно видати попередження, а потім зупинити роботу. Робочий орган машини здійснює циклічний рух по периметру прямокутника.

Для реалізації проекту виконайте наступні дії:

1. Запустіть CODESYS.
2. Створіть новий проект. Проект – машино-незалежний, тому при налаштуванні встановіть “None”.
3. Головна програма PLC_PRG POU буде написана на мові FBD.
4. Далі оголосимо змінну, яка буде відповідати за стан підтвердження. Вона буде міняти своє значення при підтвердженні оператором. Ця змінна повинна бути глобальною, мати тип Bool, ім'я – observer.
5. Для нас важливо щоб оператор натискав час від часу кнопку, а не спав з натиснутою клавішею, тому нам потрібно відслідковувати саме моменти зміни значення observer (з True в False і назад). Додайте до першого ланцюгу Box. Скористайтесь помічником вводу (клавіша <F2>), у діалозі оберіть категорію Standard Functional Blocks, з тригерів бібліотеки standard.lib оберіть R_TRIG. Цей елемент формує логічну «1» на виході в момент переходу з «0» до «1» на своєму вході. Задайте ім'я для нового екземпляру функціонального блоку R_TRIG – Trig1. Клас змінної – Var (локальні змінні). Виділіть вхід Trig1, скориставшись помічником вводу (клавіша <F2>), з'єднайте його зі змінною observer. Після цього програма мусить виглядати як зображено на рисунку 5.25.
6. Нам потрібен ще детектор заднього фронту. Його необхідно додати згідно п. 5, тип елемента – F_TRIG, ім'я – Trig2. Виходи Trig1 та Trig2 необхідно з'єднати зі входами оператора OR, який теж необхідно додати.
7. Після цього нам необхідно організувати контроль часу. Додамо екземп-

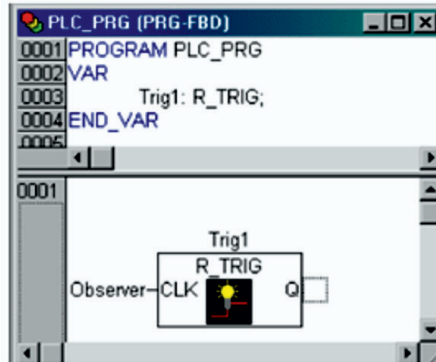


Рис. 5.25 – Фрагмент програми

ляр функціонального блоку TOF – таймер з затримкою відключення (бібліопатка standard.lib), його ім'я – Timer1. Його вхід IN з'єднати з виходом оператора OR. На вході PT повинна бути часова константа T#10s (10 секунд). Створіть глобальну змінну Warning, типу Bool. Вихід Q Таймера, скориставшись помічником вводу (клавіша <F2>), з'єднайте зі змінною Warning. У контекстному меню на виході таймеру додайте команду Negate, щоб інвертувати його значення. Після цих дій програма мусить виглядати як зображено на рисунку 5.26.

8. По іншому часовому інтервалу нам потрібно формувати сигнал зупинки руху. Для цього командою insert->network (after) додайте ще один лан-

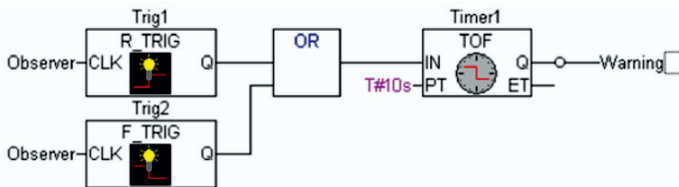


Рис. 5.26 – Вигляд програми з реалізацією функції контролю часу

цюг. В нього додайте екземпляр функціонального блоку TON – таймер з затримкою включення (бібліотека standard.lib), його ім'я – Timer2. Його вхід IN з'єднати зі змінною Warning. На вході PT повинна бути часова константа T#5s (5 секунд). Створіть глобальну змінну Stop, типу Bool. Вихід Q Таймера, скориставшись помічником вводу (клавіша <F2>), з'єднайте зі змінною Stop. Новий ланцюг повинен виглядати як зображено на рисунку 5.27

9. Далі нам потрібно створити POU керування механізмом – ім'я – Machine, тип – програма, мова – SFC.

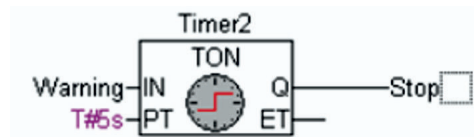


Рис. 5.27 – Фрагмент програми з реалізацією функції затримки включення

10. Кожній фазі роботи механізму відповідає крок SFC. Нам знадобляться 5 кроків – додайте їх. Перший (після Init) повинен обзиватись Go_Right, наступний – Go_Down, наступний – Go_Left, наступний – Go_up, наступний – Count.
11. Оголосимо змінні: X_Pos, Y_Pos, Counter. Тип змінних – INT, клас – локальні. Запрограмуємо кроки. Скористаємось мовою ST. На кожному кроці наш механізм повинен переміщатись на одну одиницю. Переходи повинні містити умови, що дозволяють перейти до наступного кроку. Остаточний вигляд POU Machine зображено на рисунку 5.28.
12. Поверніться до POU PLC_PRG. Додайте ще один ланцюг. На початку ланцюгу додайте змінну Stop, потім із контекстного меню додайте оператор Returne. Тобто, коли Stop – True, буде виконано вихід з програми.

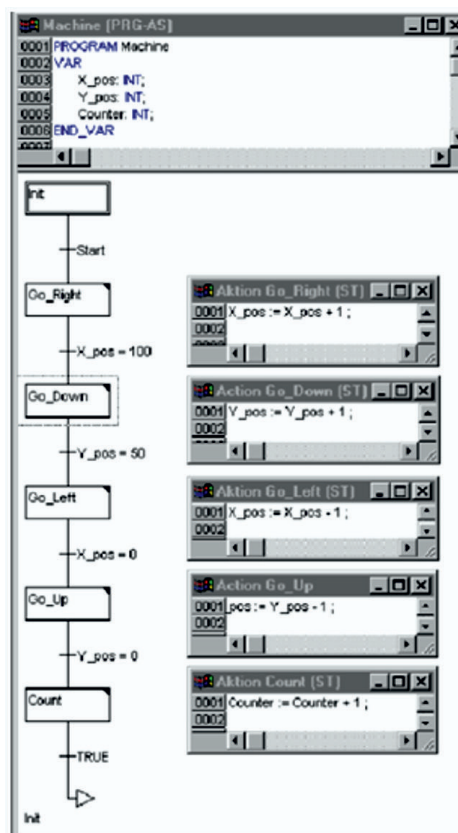


Рис. 5.28 – Остаточний вигляд POU Machine

13. Нам потрібно ще додати виклик POU Machine. Додайте ще один ланцюг. Скориставшись помічником вводу (клавіша <F2>), додайте туди POU Machine (категорія User defined Progtams).
14. Скомпілюйте проект. Виправте помилки, якщо потрібно.

15. Створіть візуалізацію, ім'я – Observation. Вигляд візуалізації зображено на рисунку 5.29.
16. Сконфігуруємо елементи візуалізації. Кнопка «ОК»: в категорії Text,

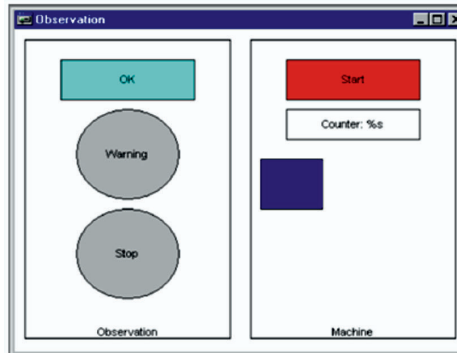


Рис. 5.29 – Вигляд візуалізації Observation

поле Contents – OK. В категорії Variables, поле Change Color – змінна Observer. В категорії Input, Toggle variable – змінна Observer. Коло «Warning»: категорії Text, поле Contents – Warning. В категорії Variables, поле Change Color – змінна Warning. Коло «Stop»: категорії Text, поле Contents – Stop. В категорії Variables, поле Change Color – змінна .Stop. Кнопка «Start»: в категорії Text, поле Contents – Start. В категорії Variables, поле Change Color – змінна .Start. В категорії Input, Toggle variable – змінна .Start. Напис «Counter»: в категорії Text, поле Contents – Counter : %s. В категорії Variables, поле Textdisplay – змінна Machine.Counter. В категорії Input, Toggle variable – змінна .Start. Прямокутник, що зображує механізм: в категорії Absolute movement, поле X-Offset – Machine.X_pos, в категорії Absolute movement, поле Y-Offset – Machine.Y_pos.

17. Запустіть проект, переконайтесь у його працездатності.

Контрольні запитання і завдання

1. Надайте визначення наступним поняттям: функціональні блокові діаграми (FBD), відображення ROU, сполучні лінії, порядок виконання FBD, інверсія логічних сигналів, з'єднувачі й зворотні зв'язки, мітки, переходи й повернення, перетворення ST в FBD.
2. Що таке: послідовні функціональні схеми (SFC), переходи, початковий крок, паралельні вітки, альтернативні вітки, перехід на довільний крок?
3. В чому полягає відмінність спрощеного SFC від стандартного SFC? Поясніть суть: класифікатора дій, дії – зміни, механізму керування дією, внутрішніх змінних кроку й дій, функціональних блоків й програм SFC, налагодження й контролю виконання SFC?
4. Як здійснюється створення об'єкта візуалізації, вставка елементів візуалізації, позиціонування елементів візуалізації, вибір елементів візуалізації, зміна режиму виділення й режиму вставки, копіювання елементів візуалізації, зміна елементів візуалізації, переміщення елементів візуалізації,

угруповання елементів, перенесення на передній план, перенесення на задній план, вирівнювання. Що таке список елементів, рядок стану у візуалізації?

- Охарактеризуйте наступні поняття: конфігурування елементів візуалізації, кут (Angle), форма (Shape), текст (Text), атрибути тексту (Textvariables), товщина ліній (Line width), колір (Color), кольорні змінні (Color Variables), абсолютне переміщення (Motion absolute), відносне переміщення (Motion Relative), змінні (Variables), введення даних (Input), спливаюча підказка (ToolTip), права доступу (Security), програмний доступ до властивостей елемента (Programmability), таблиця (Table), стрілочний індикатор (Meter), стовпчастий показчик (Bar Display), конфігурування гістограми (Histogram), таблиця тривоги (Alarm table), тренд (Trend), растровий малюнок (Bitmap), елемент візуалізації (Visualization).

5.4 Розробка кодового замка

Мета роботи

Практичне закріплення навичок розробки ПЗ ПЛК у CODESYS.

Методичні вказівки з організації самостійної роботи студентів

Під час підготовки до цієї лабораторної роботи слід передивитися розділи конспекту лекцій, що присвячені мовам IL, ST, LD, FBD, SFC.

Опис лабораторної установки

При проведенні лабораторної роботи використовується персональний комп'ютер з встановленим пакетом САПР ПЗ ПЛК CODESYS.

Порядок виконання роботи і методичні вказівки з її виконання

В цій роботі необхідно самостійно розробити ПЗ ПЛК, що керує роботою кодового дверного замка. На рисунку 5.31 наведена принципова електрична схема кодового замка на електромагнітних реле.

Для відкривання замка необхідно набрати код послідовним натисканням кнопок K2, K3, K4, K5. Першу кнопку коду K2 потрібно натискати одночасно із кнопкою дверного дзвінка K1. Всі проміжні реле P2 – P5 працюють із самофіксацією, одночасно звільняючи реле попереднього ланцюга. Помилкове натискання кнопок K6 – K9 або відкривання дверей (Кд) скидають замок у вихідний стан. Таким чином, «свій» відкриває замок з дуже коротким дзвінком, «чужий» змушений підняти дзвоніння при спробах підбора коду. Одночасне натискання всіх кнопок K2 – K9 викликає скидання у вихідний стан. Таємність цифр коду досягається розпаюванням K2 – K5 до різних клавіш складального поля. Силові ланцюги дзвінка голосного бою й електромагнітного плунжера на схемі не показані.

Згідно зі своїм варіантом необхідно реалізувати ПЗ на одній з мов стандарту МЕК 61131.

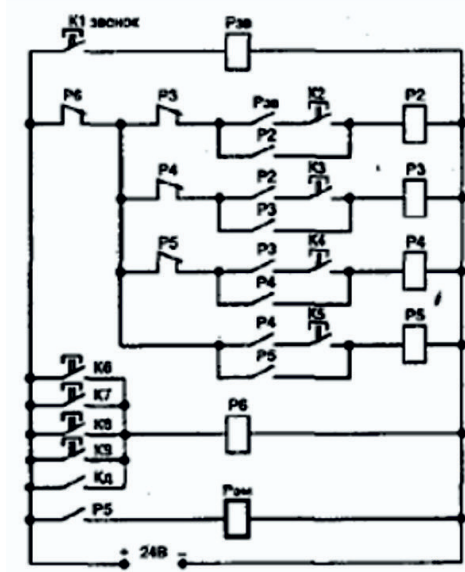


Рис. 5.31 – Принципова електрична схема кодового замка

Контрольні запитання і завдання

1. В чому полягає огляд ресурсів? Що таке глобальні і конфігураційні змінні, файл коментарів?
2. Надайте визначення конфігурації тривоги (Alarm Configuration), загальній інформації, термінології, класам, групам, записам тривоги.
3. Як використовуються менеджер бібліотек (Library Manager), бортжурнал (Log)?
4. Поясніть наступні поняття: конфігуратор ПЛК (PLC Configuration), огляд, робота в редакторі конфігуратора ПЛК, загальні параметри конфігурації ПЛК, конфігурація ПЛК у режимі Online.
5. Що таке: конфігуратор завдань (Task Configuration), огляд, робота у конфігураторі завдань, системні події, конфігуратор завдань у режимі online?
6. Для чого існують менеджер рецептів (Watch and Receipt Manager), трасування (Sampling Trace), робоча область (Workspace)?
7. В чому полягає суть наступних понять: менеджер параметрів (Parameter Manager), налаштування цільової платформи (Target Settings), ПЛК-браузер (PLC-Browser), інструменти (Tools).

Висновок

З декількох існуючих засобів розробки додатків для ПЛК, що відповідають міжнародному промисловому стандарту IEC 61131-3, CODESYS є однією з найпопулярніших.

У CODESYS реалізовано найбільш повний спектр розширень специфікації стандарту IEC 61131-3. Найістотнішим із них являється підтримка об'єктно-орієнтованого програмування.

CODESYS версії V3 дозволяє розвивати свій комплекс шляхом підключення власних плагінів.

Весь інструментарій в цілому динамічно розвивається, з'являються нові версії, удосконалюється процес переходу від програмування додатків до їх швидкого складання.

Тому при викладенні матеріалу в навчальному посібнику основна увага приділена концептуальним підходам, що служать основою для подальшого освоєння як можливостей нових версій CODESYS, так і інших інструментальних середовищ, що відповідають вимогам стандарту IEC 61131-3.

Подальша підтримка стандарту міжнародною електротехнічною комісією і розробниками промислових і мобільних систем керування гарантує довгостроковість використання CODESYS і інших подібних інструментальних середовищ.

Знання і навички, отримані при вивченні CODESYS, допоможуть без великих труднощів освоїти, при необхідності, будь-яке інше середовище, включаючи спеціалізовані, орієнтовані на інші окремі типи ПЛК.

ЛІТЕРАТУРА

1. Пупена О. М. Промислові мережі та інтеграційні технології в автоматизованих системах : навчальний посібник / О. М. Пупена, І. В. Ельперін, Н. М. Луцька, А. П. Ладанюк. – К.: Ліра, 2011. – 552 с.
2. Jochen Petry IEC 61131-3 mit CoDeSys V3: Ein Praxisbuch für SPS-Programmierr. – 2011. – 839 с.
3. Деменков Н. П. Языки программирования промышленных контроллеров : учеб. пособие / Под ред. К.А. Путкова, – М.: Изд-во МГТУ им. Н. Э. Баумана, 2004 – 172 с.
4. Hanssen H. Dag. Programmable Logic Controllers: A Practical Approach to IEC 61131-3 using CoDeSys, Wiley, 2015. – 408 с.
5. Горб С. И. Программирование контроллеров в инструментальной среде : учебное пособие / С. И. Горб, В. В. Никольский, В. Ф. Шапо, С. Г. Хнюнин. – Харьков: Издатель ФЛП Панов А. Н., 2017. – 172 с.
6. Langmann R. Workshop: The TATU Lab & smart education / R. Langmann, I. Klyuchnyk, P. Galkin et al. // Remote Engineering and Virtual Instrumentation (REV), 2016 13th International Conference on. – IEEE, 2016. – С. 400 – 402.
7. Makarova Y. Prototype of the modern hands-on smart lab for automation engineering / Y. Makarova, R. Langmann // Remote Engineering and Virtual Instrumentation (REV), 2016 13th International Conference on. – IEEE, 2016. – С. 254 – 259.
8. Ключник І. Програмування ПЛК в CoDeSys V3.0 / І. Ключник, П. Галкін, О. Шапоріна. – Одеса: ФОП Побута М. І., 2017. – 107 с.
9. Ключник І. Бездротові технології / І. Ключник, П. Галкін, О. Шапоріна. – Одеса: ФОП Побута М. І., 2017. – 44 с.
10. Iustinus Tim Avery Codesys Cel Publishing, 2012. – 96 с.
11. Галкин П. В. Исследование дальности и скорости передачи данных по витой паре в промышленных сетях RS-485 и PROFIBUS / П. В. Галкин, В. В. Гавриленко, А.И. Монько. Проблемы телекоммуникацій, 2016. – № 2 (19). – С. 94 – 110.
12. Birgit Vogel-Heuser Automation and Embedded Systems: Effizienzsteigerung Im Engineering, Kassel: Kassel university press GmbH, 2009. – 123 p.
13. Hugh J. Automating Manufacturing Systems with PLCs, GNU Free Documentation, 2008. – 868 с.
14. Бобров М. А. Специализированные интерфейсы в системах «умного дома» / М. А. Бобров, О. В. Шишов // Огарёв-Online, 2014. – №. 3 (17). – С. 2.
15. Galkin P. Interaction model design of ZigBee-gateway between wireless sensor network and industrial network //Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), 2017 4th International. – IEEE, 2017. – P. 501 – 504.

СПИСОК СКОРОЧЕНЬ

CFC (англ. Continuous Function Chart) – додаткова реалізація мови FBD із вільним розміщенням елементів мови, блоків і з'єднань.

CODESYS – інструментальне пристроєнезалежне середовище для розробки додатків для ПЛК.

DHCP (англ. Dynamic Host Configuration Protocol) – протокол динамічної конфігурації вузла) – це стандартний протокол прикладного рівня, який дозволяє комп'ютерам автоматично отримувати IP-адресу та інші параметри, необхідні для роботи в мережі.

Ethernet – найпопулярніший протокол кабельних комп'ютерних мереж, що працює на фізичному та канальному рівнях мережевої моделі OSI.

FBD (англ. Function Block Diagram) – мова функціональних блокових діаграм.

GPRS (англ. General Pocket Radio Service) – стандарт, який використовує не зайняту голосовим зв'язком смугу частот для передачі інформації.

GSM (англ. Global System for Mobile Communications) – міжнародний стандарт для мобільного цифрового стільникового зв'язку з розділенням каналу за принципом TDMA та високим рівнем безпеки за рахунок шифрування з відкритим ключем.

IL (англ. Instruction List) – мова лінійних інструкцій.

IP (англ. Internet Protocol) – найпоширеніша реалізація ієрархічної схеми мережевої адресації. LD (англ. Ladder diagram) – мова релейних діаграм.

Modbus – комунікаційний протокол, заснований на технології master-slave. Широко застосовується в промисловості для організації зв'язку між електронними пристроями.

OPC DA (англ. Data Access) – один з відкритих протоколів, що регламентують взаємодію між собою різних об'єктів автоматизації, таких як SCADA-системи, конкретно він забезпечує обмін даними з пристроями або програмними компонентами.

PC WORX – інструментальне середовище для програмування ПЛК.

PLC (англ. Programmable Logic Controller) – програмований логічний контролер.

POU (англ. Program Organization Unit) – компонент організації програм.

PROFIBUS – мережа інтеграції систем автоматизації.

PROFINET (англ. Process Field Network – мережа польового рівня) – відкритий промисловий стандарт для автоматизації від асоціації Ethernet PROFIBUS & PROFINET International (PI). PROFINET використовує TCP / IP і IT-стандарти, і режим реального часу Ethernet.

RFID (англ. Radio frequency identification) – радіочастотна ідентифікація. Радіочастотне розпізнавання здійснюється за допомогою закріплених за об'єктом спеціальних міток, що несуть ідентифікаційну та іншу інформацію.

SCADA (англ. Supervisory Control And Data Acquisition) – система управління, в якій обов'язково присутня людина (оператор, диспетчер).

SD (англ. Secure Digital) – формат карти флеш-пам'яті.

SFC (англ. Equential Function Chart) – мова послідовних функціональних схем.

SNMP (англ. Simple Network Management Protocol – простий протокол керування мережею) – це протокол керування мережами зв'язку на основі архітектури TCP/IP.

ST (англ. Structured Text) – мова структурованого тексту.

TDMA (англ. Time Division Multiple Access) – це метод часового поділу одного фізичного каналу зв'язку (зазвичай радіо).

TSL (TATU Smart Lab) – спеціалізована лабораторія, що створена в рамках Темпус проекту ТАТУ.

USB (англ. Universal Serial Bus) – універсальна послідовна шина, призначена для з'єднання периферійних пристроїв обчислювальної техніки.

WLAN (англ. Wireless Local Area Network) – бездротова локальна мережа. При такому способі побудови мереж передача даних здійснюється через радіоефір; об'єднання пристроїв у мережу відбувається без використання кабельних з'єднань. Найпоширенішими на сьогоднішній день способами побудови є Wi-Fi і WiMAX.

МЕК (IEC) – міжнародна електротехнічна комісія.

ПЗ – програмне забезпечення.

ПЛК – програмований логічний контролер.

САПР – система автоматизації проектних робіт.

Компоненти організації програм

ABS – числова функція обчислює абсолютне значення операнда, підключеного до IN1.

ADD – арифметична функція виконує додавання операндів, підключених до входів IN1 і IN2.

AND – побітова булева функція виконує логічну операцію AND на операндах, підключених до IN1 і IN2.

DIV – арифметична функція ділить операнд, з'єднаний з IN1 операндом, підключеним до IN2.

MUL – арифметична функція множить операнди, підключені до входів IN1 і IN2.

NOT – побітова булева функція виконує побітове заперечення на операнді, підключеному до IN1.

OR – побітова булева функція виконує логічну операцію АБО для операндів, підключених до IN1 та IN2.

SHL – функція біт-рядка виконує операцію покрокового зсуву лівого операнда на операнді, підключеному до входу IN. N вказує кількість бітів, які повинні бути зміщені. Позиції порожнього біта заповнюються нулями.

SHR – функція біт-рядка виконує операцію покрокового зсуву по операнду, підключеному до входу IN. N вказує кількість бітів, які повинні бути зміщені. Позиції порожнього біта заповнюються нулями.

TOF – функціональний блок таймера реалізує затримку встановлення значення TRUE на виході.

TON – функціональний блок таймера реалізує затримку встановлення значення FALSE на виході.

WORD TO BYTE – функція перетворення типу перетворює вхідне значення типу даних WORD у вихідне значення типу даних BYTE.

WORD_TO INT – функція перетворення типу перетворює вхідне значення типу даних WORD у вихідне значення типу даних INT.

XOR – побітова булева функція виконує логічну операцію XOR (виключає АБО) на операндах, підключених до IN1 і IN2.

Навчальне видання

Павло Вікторович Галкін

старший викладач кафедри проектування та експлуатації електронних апаратів Харківського національного університету радіоелектроніки;

Ігор Іванович Ключник

канд. техн. наук, проф. кафедри проектування та експлуатації електронних апаратів Харківського національного університету радіоелектроніки.

Програмування ПЛК в CODESYS

Навчальний посібник

Головний координатор професор Reinhard Langmann,

Університет прикладних наук Дюсельдорфа, Німеччина,
Competence Center Automation Dsseldorf (CCAD).

Видано в авторській редакції по проекту TEMPUS 544010-TEMPUS-1-2013-1-DE-TEMPUS-JPHES – «Trainings in Avtomation Technologies vor Ukraine» (TATU).

Зміст даної книги відображає точку зору авторів і Європейська Комісія не несе відповідальності за використання інформації, що в ній міститься (The content of this book reflects the views of the authors and the European Commission is not responsible for the use of information contained therein).

Дизайн обкладинки Трубіцин О. О.

Підписано до друку 19.02.19. Друк цифровий. Папір офсетний.
Гарнітура Times New Roman. Формат 70x100/16. Умов. друк. аркушів 5,62.
Наклад 120 прим. Замовлення № 13/02/19.

Видавець і виготовлювач: ФОП Панов А.М.

Свідоцтво про внесення до Державного реєстру видавців,
виготівників і розповсюджувачів видавничої продукції
серія ДК № 4847 від 06.02.2015 р.

м. Харків, вул. Жон Мироносиць, 10, оф. 6,
тел. +38(057)714-06-74, +38(050)976-32-87

copy@vlavke.com