Кафедра організації та технічного забезпечення
аварійно-рятувальних робіт
Національного університету цивільного захисту України

Методичні вказівки до виконання контрольної роботи

з дисципліни
Автоматизовані системи управління технологічного процесу в хімічних виробництвах

«Розроблення прикладного програмного забезпечення для систем управління на мові програмування CFC в середовищі CoDeSys»

для здобувачів вищої освіти заочної форми навчання;
освітньо-кваліфікаційного рівня «магістр»
за спеціальністю 161 Хімічні технології та інженерія,
спеціалізацією «Радіаційний та хімічний захист»

Харків
2023

Підготовлено до друку за рішенням
засідання кафедри організації та
технічного забезпечення аварійно-
рятувальних робіт НУЦЗ України
Протокол від 28.08. 2023 р. № 1

.

Укладачі: Л.В. Борисова

Рецензенти**:** І.Г. Лисаченко, доцент кафедри автоматизації технологічних систем та екологічного моніторингу Національного технічного університету (ХПІ) кандидат технічних наук, доцент;
Л.М. Трифілова, професор кафедри спеціальної хімії та хімічної технології Національного університету цивільного захисту України, д.ф.-м.н. професор.

Розроблення прикладного програмного забезпечення для систем управління на мові програмування CFC в середовищі CoDeSys: методичні вказівки до виконання контрольної роботи. Для здобувачів вищої освіти заочної форми навчання / Л.В. Борисова. Х.: НУЦЗУ, 2023. – 82 с.

Зміст

Методичні вказівки для виконання контрольної роботи

Метою контрольної роботи є закріплення теоретичних знань та отримання практичних навичок програмування ПЛК при створенні автоматичних систем управління в хімічних виробництвах.

Основними завданням є отримання базових знань з розроблення програмного прикладного забезпечення для автоматизованих систем управління технологічними процесами в хімічних виробництвах на базі мікропроцесорних промислових контролерів вільного програмування на прикладі ОВЕН ПЛК із застосовуванням технологічної мови програмування CFC, використовуючи спеціалізоване програмне забезпечення для програмування та налагодження програми користувача (комплекс програмування CoDeSys).

Під час виконання контрольної роботи здобувач вищої освіти повинен на основі аналізу алгоритму функціонування об'єкта розробити алгоритм управління ним, обґрунтувати вибір моделей і модифікацій мікропроцесорних контролерів та розробити для них прикладне програмне забезпечення.

Завдання для виконання контрольної роботи видається у вигляді опису алгоритму управління об'єктом з переліком основних режимів роботи та функціональних вимог для забезпечення нормальної роботи об'єкта управління. Для одного і того ж об'єкта управління можуть існувати різні експлуатаційні режими: запуск, забезпечення нормального функціонування та реагування на виникнення різних нештатних та аварійних ситуацій.

Однією із складових завдання для виконання контрольної роботи є визначення процесів та алгоритмів, що піддаються імітаційному моделюванню під час розроблення прикладного програмного забезпечення для мікропроцесорних контролерів.

Вказівки до виконання контрольної роботи:

– алгоритмічна структура алгоритму управління складається для систематизації та структурування обробки інформації, а також вироблення необхідної послідовності управляючих дій для забезпечення нормального функціонування об'єкта;

– алгоритмічна структура управління є універсальним носієм інформації і не залежить від типу та моделі контролера, на якому цей алгоритм буде реалізований.

Для вибору моделі контролера і його конфігурації необхідно визначитись переліком технічних засобів автоматизації, які будуть використовуватись для реалізації алгоритму управління, а саме знати параметри сигналів від датчиків і виконавчі механізми.

Контрольна робота складається з двох частин:

1. Створення проекту. Входи і виходи. Логіка.
2. Обчислення для скомпонованого УОК імовірності безвідмовної роботи або середній час напрацювання до відмови.

Частина 1. Створення проекту. Входи і виходи. Логіка.

CoDeSys – це сучасний інструмент для програмування контролерів (CoDeSys утворюється від слів Controllers Development System).

CoDeSys – це зручне середовище для програмування контролерів на язиках стандарту МЕК 61131-3.

Процедура програмування ПЛК включає наступні етапи:

1) Попередній етап: установка операційної системи і ПО (середовища програмування) CoDeSys (Controller Development System).

2) Вибір контролера. Установка необхідного файлу налаштувань цільової платформи (target-файлу).

3) Створення та налагодження проекту.

4) Встановлення зв'язку з контролером. При установці зв'язку ПО CoDeSys автоматично компілює проект і пропонує завантаження скомпільованого коду в ОЗУ контролера.

5) Запуск виконання проекту (користувальницької програми ПЛК), перевірка її працездатності і, при необхідності, налагодження.

6) У разі коректної роботи проекту (користувальницької програми ПЛК) – збереження її в незалежній Flash-пам'яті контролера для подальшого завантаження і виконання при включенні живлення ПЛК.

У разі некоректної роботи проекту – повернення на етап 5 (у процесі налагодження проекту перераховані вище операції можуть виконуватися багаторазово).

1. Створення проекту

1.1 Створення нового проекту, знайомство з інтерфейсом

Після закінчення запуску установки програми на робочому столі комп'ютера з'явиться ярлик з трьома різнокольоровими шестикутниками  для запуску CoDeSys. Можна скористатися ним або вибрати в меню «Пуск» наступний шлях «Все программы - 3S Software - CoDeSys 2.3».

Після запуску CoDeSys з'явиться сіре вікно з рядком меню зверху.

Для ознайомлення з зовнішнім виглядом системи програмування створимо проект.

У меню «Файл» необхідно вибрати пункт «Создать» (рис. 1.1), або трохи нижче знайти іконку  і скористатися нею. У вікні «Настройка целевой платформы» (рис. 1.2) поки нічого не змінюємо, натискаємо кнопку «ОК». У наступному вікні (рис. 1.3) система програмування пропонує вибрати мову реалізації.

Вибираємо, наприклад, мову функціональних блоків CFC згідно з малюнком і натискаємо «ОК».

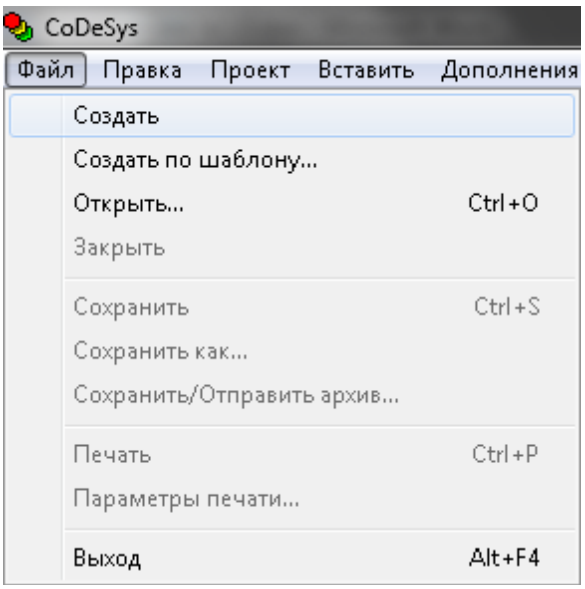


Рис. 1.1

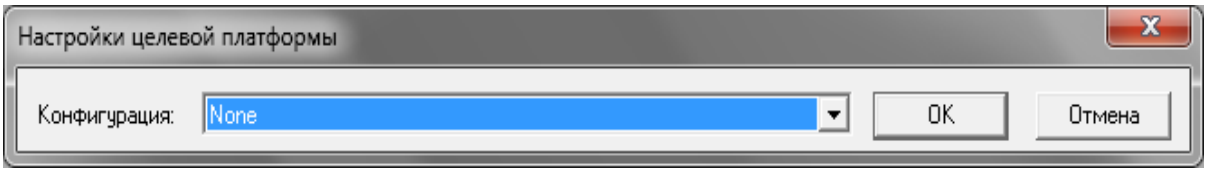


Рис.1.2

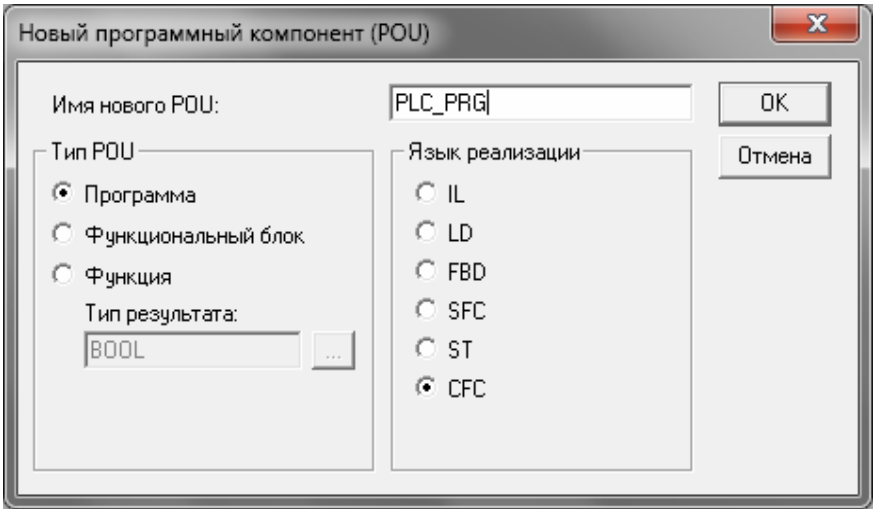


Рис. 1.3

Після цих операцій CoDeSys відкриває основну робочу область (рис. 1.4). *Головне меню* у верхній частині містить пункти «Файл», «Правка» і т.д.

Панель швидкого доступу, що розташовується нижче головного меню, дозволяє виконувати найбільш часто використовувані операції одним натисканням на відповідну іконку. Склад іконок буде змінюватися в процесі роботи над різними компонентами проекту (в більшості випадків все необхідне винесено в цю область).

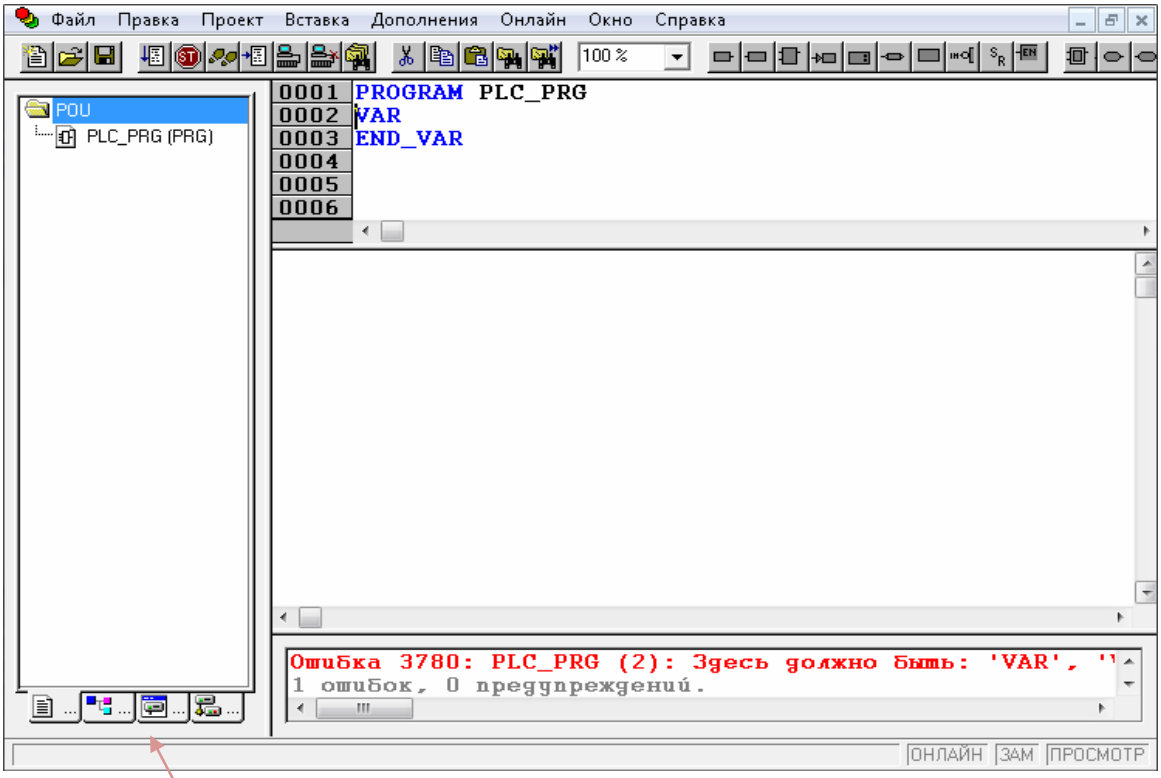


Рис. 1.4

Область, яка займає ліву сторону екрану – це *Менеджер об'єктів*. У її нижній частині можна побачити чотири вкладки.

Рис.1.5

Перемикаючись між ними за допомогою миші, вибираємо різні компоненти (об'єкти) проекту, дивимося їх вміст, щось видаляємо або додаємо. Якщо вибрати крайню ліву вкладку 

, лівою кнопкою миші (далі приймемо скорочення ЛКМ для позначення натискання на ліву кнопку миші) і потім двічі натиснути ЛКМ на написи 

, то в центральній частині екрана буде видна *робоча область головної програми PLC_PRG*. Аналогічним чином, перемикаючись між вкладками в менеджері об'єктів, викликаємо на екран вміст того чи іншого компонента для перегляду або редагування. Верхня частина робочого поля програми називається *областю визначення змінних*. Незалежно від вибраної мови програмування ця область завжди буде спочатку мати вигляд, представлений на рис. 1.6.

Рис. 1.6

При програмуванні в CoDeSys усі дані, динамічно змінюються в процесі роботи (вимірювання на входах, уставки, стан вихідних змінних і т.п.), прийнято задавати через різні змінні. Важливо запам'ятати, що саме *в області визначення задаються ті самі змінні, значення яких планується використовувати в поточній програмі*.

Нижче області визначення знаходиться *область програмування*, в якій надалі прописуються ті або інші алгоритми або в текстовому, або в графічному вигляді, в залежності від вибраної мови реалізації.

Нижче області програмування розташовується *вікно повідомлень*. Воно з'являється не відразу при створенні проекту, а в момент, коли система перевіряє результат роботи, наприклад, перед завантаженням алгоритму в ОВЕН ПЛК.

У даному вікні система наводить дані про проект, наприклад, розмір використаної пам'яті, кількість задіяних змінних і т.п.

Найбільш важливою інформацією тут є повідомлення про явні помилки, виявлені системою. У вікні наводиться загальна кількість виявлених помилок, і повідомлення про кожну з них. Такі повідомлення (рис. 1.7) виводяться червоним шрифтом. При цьому *система вказує, в якому місці проекту виявлена помилка і яка це помилка.*

Подвійне натискання ЛКМ на відповідній сходинці з повідомленням переносить нас в ту частину проекту, яка, на думку CoDeSys, містить помилку. До тих пір, *поки всі помилки не будуть виправлені, система не дозволить завантажити проект в ПЛК* або запустити його в режимі емуляції.

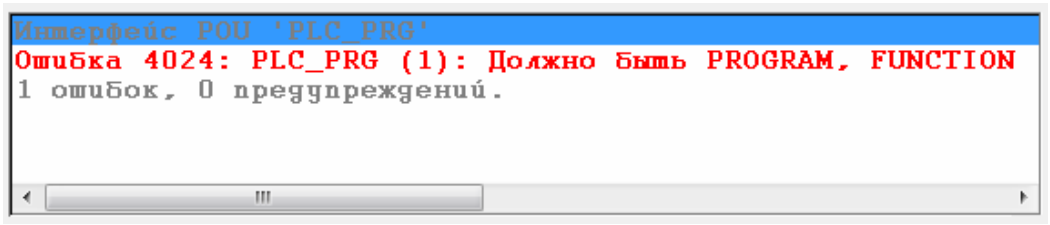


Рис.1.7

Ще нижче, в правому нижньому кутку екрану знаходиться невеликий за розміром, але важливий елемент. Це рядок статусу (рис. 1.8), тобто в якому режимі зараз знаходиться CoDeSys. Найочевидніше – чи можна зараз редагувати проект, або система перебуває на зв'язку з ОВЕН ПЛК. Типи режимів, їх особливості та індикацію вивчимо в міру просування по матеріалу.

рис. 1.8

Після створення проекту корисно зберегти його під яким-небудь ім'ям. Для цього можна скористатися меню «Файл» і в списку, команд вибрати

ЛКМ пункт «Сохранить» (рис.1.9). З тією ж метою можна скористатися поєднанням клавіш Ctrl + S або натиснути ЛКМ на іконку  .

У вікні «Сохранить как» (рис.1.10) визначається місце на комп'ютері, куди планується зберегти файл проекту. Доцільно створити окрему папку для збереження всіх проектів.

Потім в полі «Имя файла» записується довільне ім'я, наприклад, pro1. Мова не є принциповою: можна використовувати як кириличні, так і латинські символи.

Перевірте, що в поле «Тип файла» встановлено «CoDeSys проект (*.pro)» і натискайте «Сохранить».

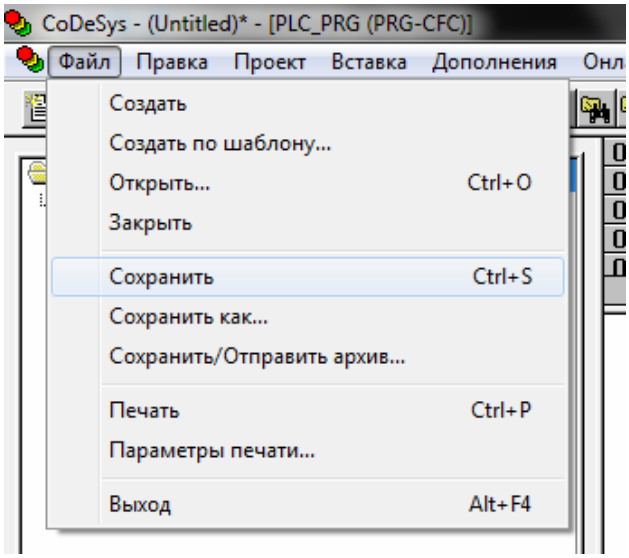


Рис.1.9

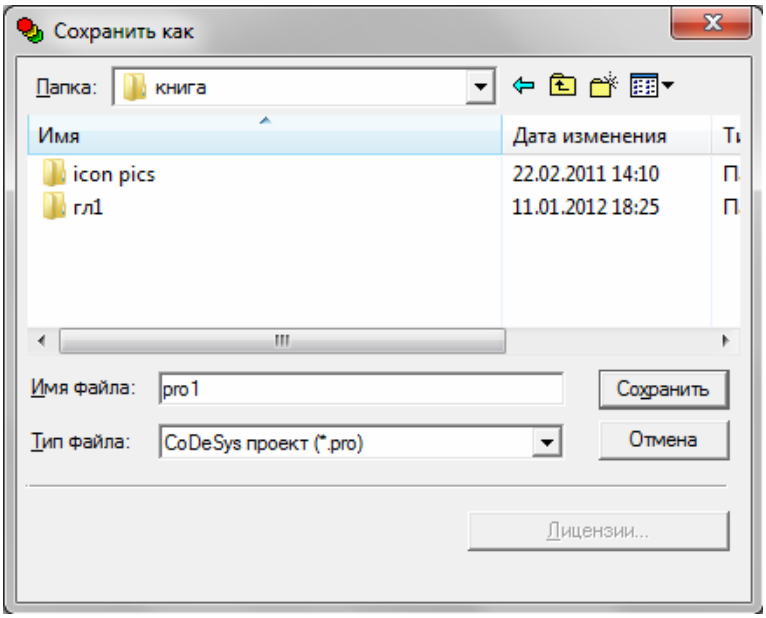


Рис. 1.10

Після збереження в лівому верхньому кутку вікна системи напис (Untitled) * змінився на присвоєне ім'я файлу (pro1) з розширенням .pro.

1.2 Настройка проекту

Зробимо кілька налаштувань системи, які дозволять в подальшому зробити роботу більш зручною. Для цього ми переходимо в розділ «Проект» головного меню, а потім ЛКМ вибираємо пункт «Опции» (рис. 1.11).

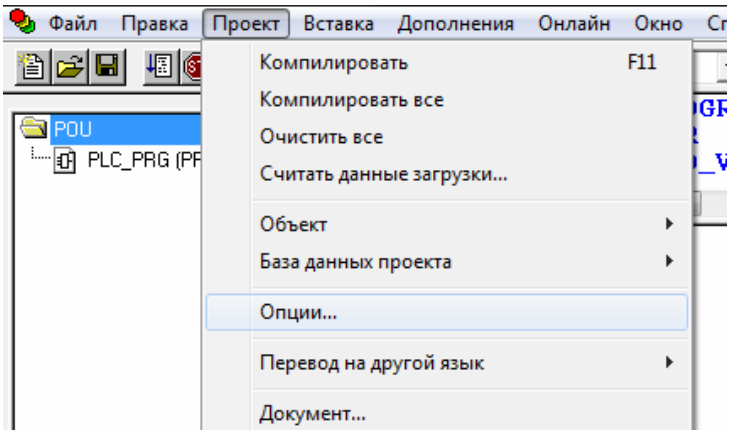


Рис. 1.11

У вікні «Настройки» в лівій частині вибираємо категорію «Сохранение». Вид вікна представлений на рис. 1.12.

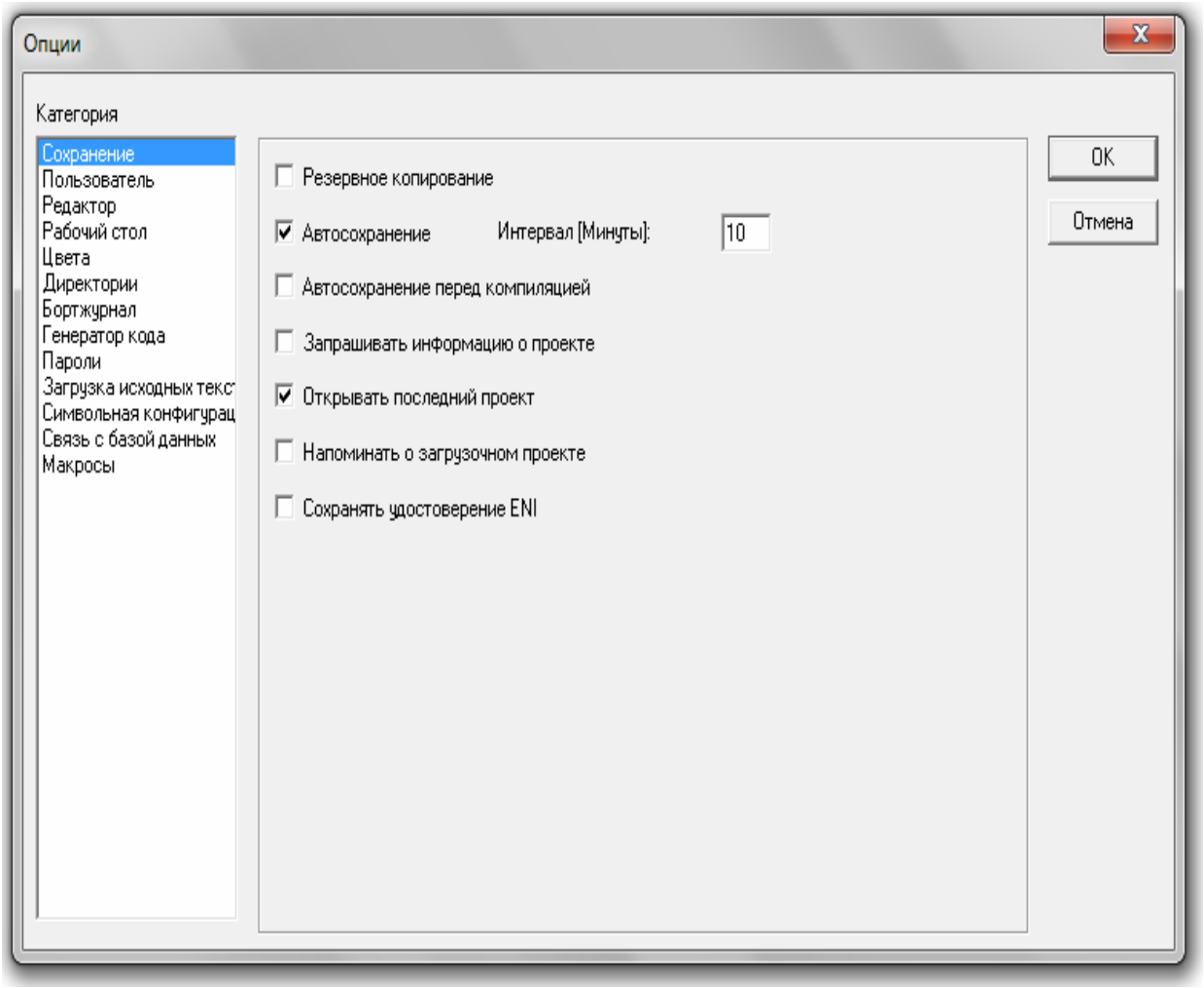


Рис. 1.12

Серед опцій, що відкрилися, корисно звернути увагу на пункт «Автосохранение».

Щоб зберігати проміжні результати, у процесі програмування ОВЕН ПЛК, корисно в опціях проекту (рис. 1.12) встановити галочку в пункті «Автосохранение» і лівіше – певні інтервалами створення резервних копій. Необхідність час від часу зберігати проект вручну залишається.

Опція «Автосохранение» включається тільки після того, як файлу проекту присвоїли якесь ім'я. *Запам'ятати: файл проекту корисно зберегти відразу після його створення.*

У категорії «Сохранение» (рис. 1.12) можна поставити галочку «Открыть последний проект»: при поновленні роботи з CoDeSys система автоматично буде завантажувати останній проект, над яким працювали.

Категорія «Редактор» (рис. 1.12) дозволяє змінити шрифт (кнопка «Шрифт»), який використовується в процесі програмування.

У процесі створення безпосередньо алгоритму в CoDeSys, як правило, використовуються латинські символи. Доцільно залишати коментарі до тих чи інших частин проекту, пояснюючи на полях найбільш важливі для розуміння речі. Натискаємо кнопку «ОК» у вікні вибору шрифту переходимо в категорію «Рабочий стол».

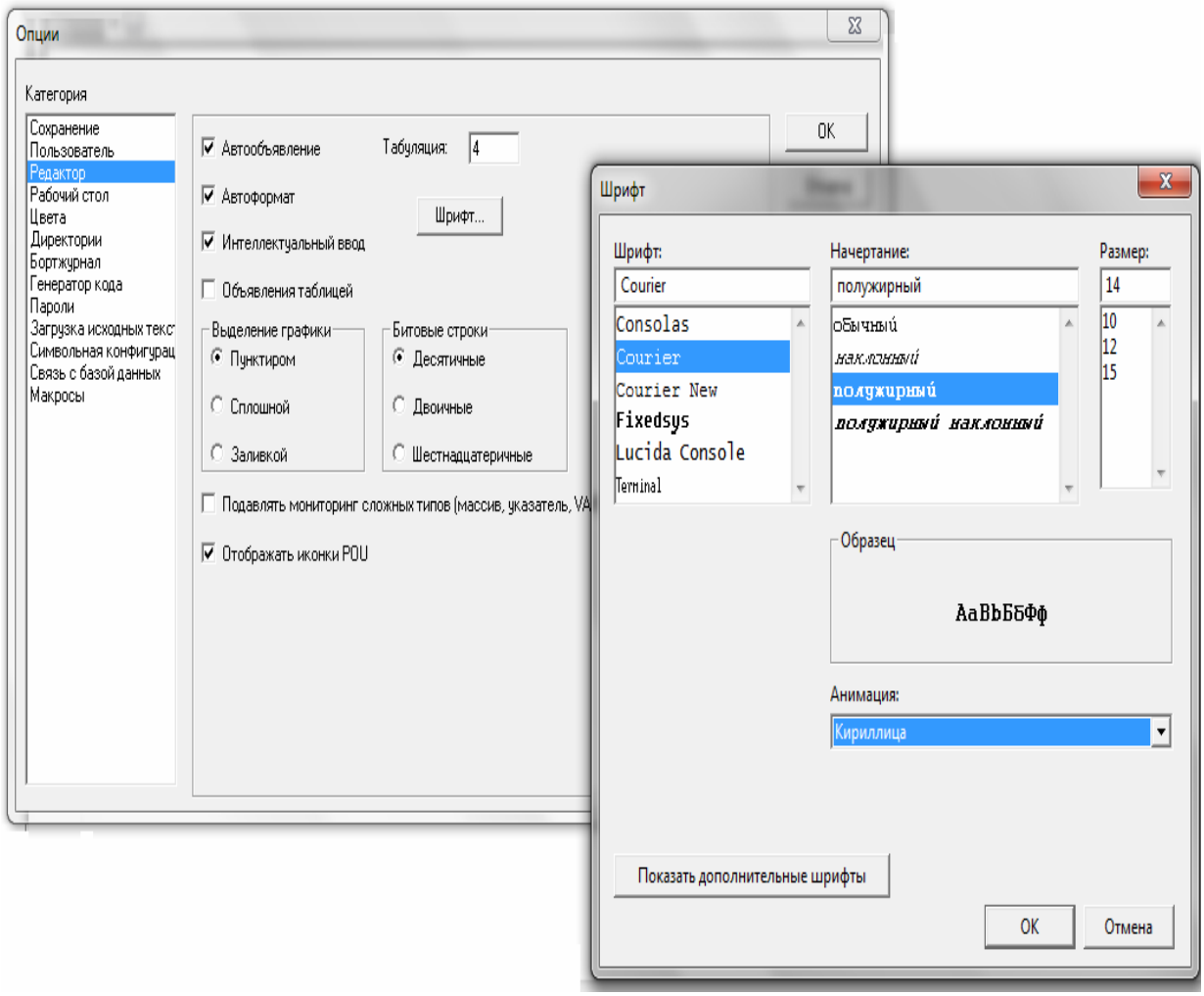


Рис. 1.13

Версія системи програмування CoDeSys 2.3.9.22, як і версії, що з'явилися пізніше (більш високий порядковий номер) мають особливість: можливість перейти до більш звичної мови на вкладці «Рабочий стол» (рис. 1.13). Потім натиснути кнопку «ОК», прийнявши всі зроблені у вікні «Наастройки» зміни. Після цього корисно зберегти проект, наприклад, натиснувши поєднання клавіш Ctrl + S.

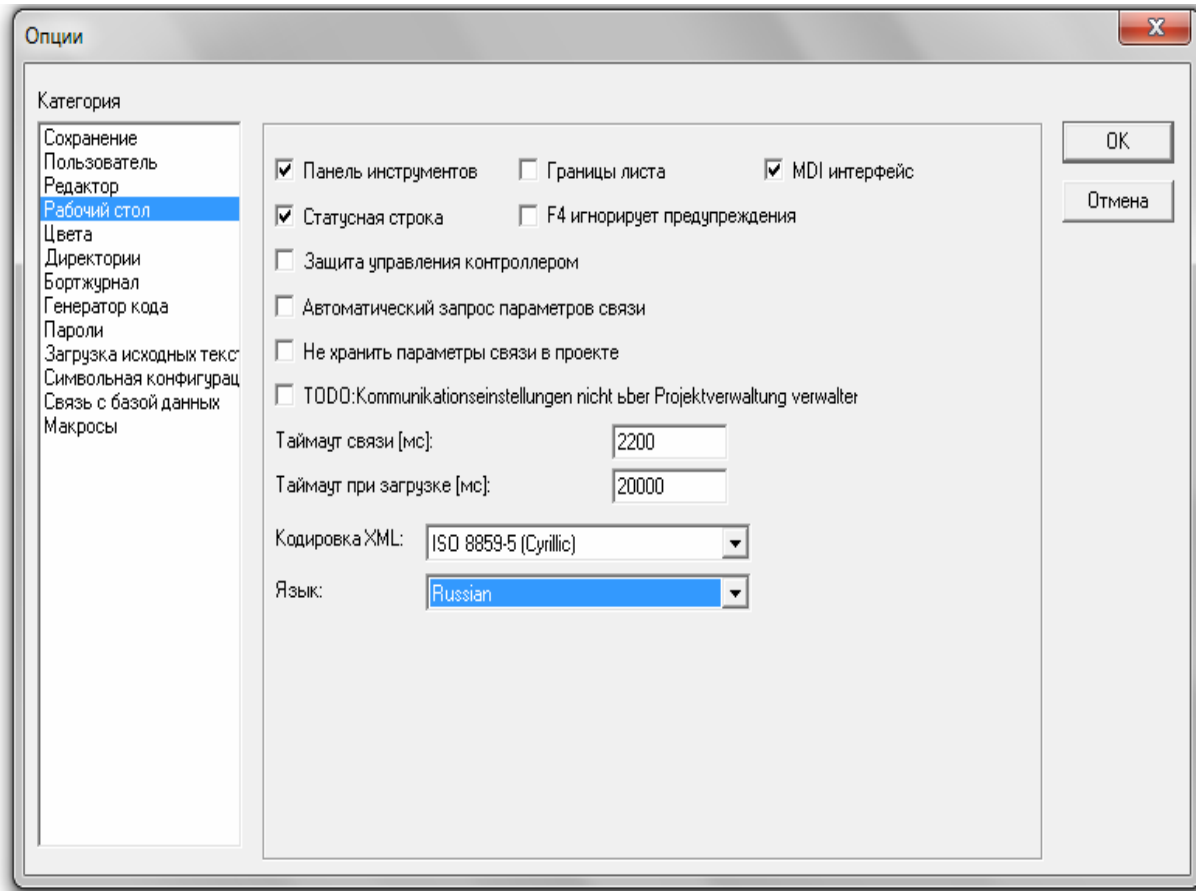


Рис. 1.14

1.3 Задача: режим редагування і режим виконання

Необхідно реалізувати на мові функціональних блоків CFC вираз ***k = k + 1****; тут* ***k*** *– цілочисельна змінна, до якої за умовою необхідно додавати одиницю.*

У проекті pro1.pro необхідно відкрити робочу область головної програми PLC_PRG. Для цього необхідно ЛКМ вибрати в Менеджері об'єктів (це область в лівій частині екрана) крайню ліву вкладку POU і потім двічі натиснути ЛКМ на написи PLC_PRG (PRG). На екрані з'явиться робоча область, поки порожня, а над нею область визначення змінних.

Починати реалізацію будь-якого завдання корисно з визначення тих змінних, які будуть задіяні. Більшість даних, які використовуються в алгоритмі, зазвичай існують у проекті в якості значень певних змінних. На практиці це означає, що якщо беруть значення зі змінної ***k*** або записують в неї отриманий результат – змінна ***k*** повинна існувати в проекті, іншими

словами, повинна бути оголошена. Під *оголошенням* розуміється пояснення системі, що під ім'ям ***k*** приймається деяке значення певного типу. У нашому випадку, за умовою ***k*** – ціле число, тому вибирається цілочисельний тип змінної. На рис. 1.15. необхідно ЛКМ поставити курсор в області визначення після ключового слова VAR, натиснути кнопку *Enter* на клавіатурі, звільняючи таким чином вільний рядок. Потім набираємо рядок: ***k: INT;***

Необхідно правильно надрукувати всі зазначені знаки пунктуації!

Важливо, щоб зроблений напис розташовувалася між ключовими словами VAR і END_VAR!

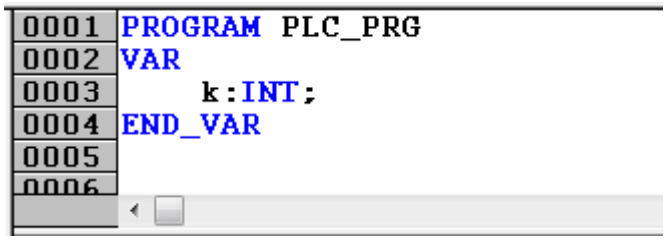


Рис. 1.15

Після натискання клавіші *Enter* в кінці рядка слово *INT* автоматично забарвиться в *синій колір*. Зроблений напис повідомляє CoDeSys, що буде використано ім'я ***k***, і пояснює, що в цієї змінної необхідно зберігати цілі значення (значення типу *INT*).

Якщо звернути увагу на панель швидкого доступу (вона розташована під головним меню), то в правій її частині є кілька іконок (рис. 1.16). Вони характерні для програмування на мові функціональних блоків CFC (нагадаємо, саме така мова вибрана при створенні проекту – Рис. 1.2).

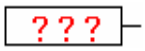
Рис. 1.16

Першими трьома іконками нам належить скористатися. Познайомимося з ними:

 – «вхід» – дозволяє отримати поточне значення змінної, щоб використовувати його в алгоритмі.

 – «вихід» – дозволяє записати якесь отримане значення в змінну.

 – «елемент» – дозволяє додати потрібний оператор на робочу область.

Почнемо з «входу». Натиснемо на іконці  ЛКМ, пересунемо курсор на робочу область і ще раз натиснемо ЛКМ в потрібному місці, щоб додати виклик значення в нашу програму. У робочій області з'являється зображення 

. Аналогічну операцію виконуємо, якщо в робочій області

натиснути правою кнопкою миші (далі ПКМ) і в контекстному меню (Рис.1.17) вибрати пункт «Вход». Потім значок ???, що з'явився, встановлюємо в потрібне місце робочої області за допомогою ЛКМ.

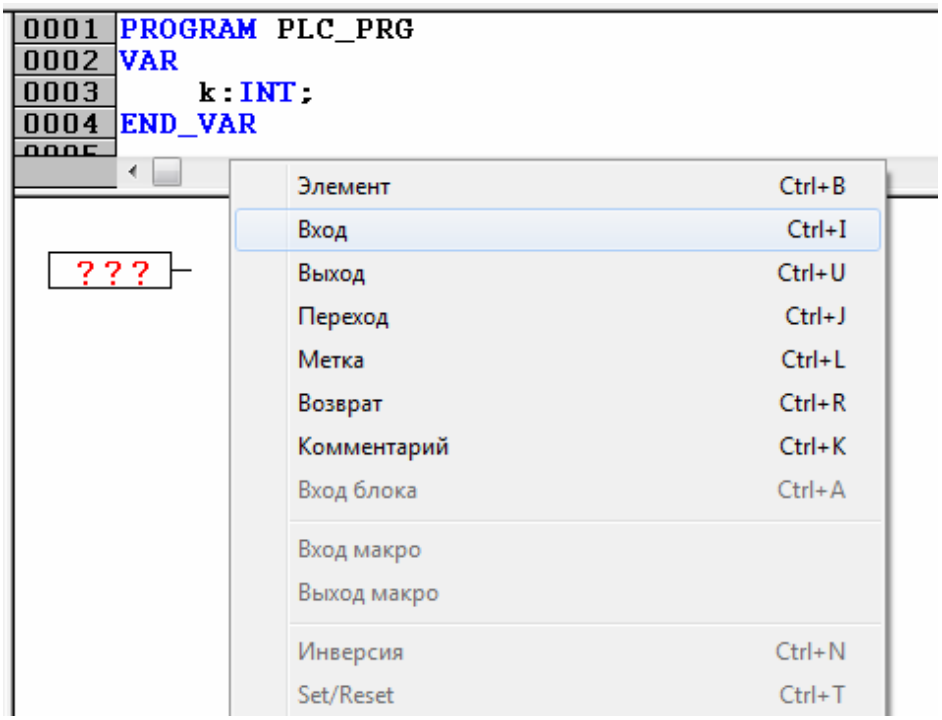


Рис.1.17.

Додавання двох «входів» на робочу область.

Якщо елемент необхідно пересунути, він виділяється в робочій області. Для цього натискається ЛКМ і пунктирна рамка, яка з'явилася на екрані, розтягується, захоплюючи ті елементи, які необхідно виділити (рис.1.18). Потім ЛКМ необхідно затиснути виділену область в лівій частині потрібного елемента (рис.1.19), перетягнути його на потрібне місце і відпустити ЛКМ.

Виділений елемент можна видалити з робочої області, наприклад, натиснувши кнопку Del на клавіатурі. Для того, щоб зняти виділення, досить натиснути ЛКМ в будь-якому вільному місці робочої області.

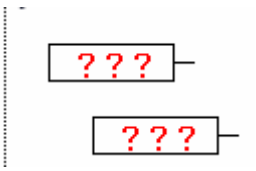


Рис.1.18

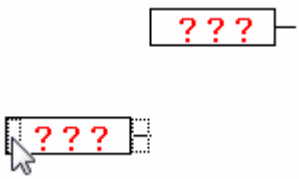


Рис.1.19

У двох зображеннях, що з'явилися в робочій області, присутні знаки питання червоного кольору. Таким чином система підказує, що замість знаків питання необхідно вказати значення або ту змінну, з якої таке значення потрібно взяти. У завданні необхідно використовувати значення цілої змінної ***k***, а також використовувати значення 1.

Натискаємо ЛКМ на знаках питання в будь-якому з доданих елементів. Напис виділяється ???, його можна стерти і написати ім'я змінної ***k***, а потім натиснути Enter. Аналогічно виділяючи текст всередині другого «входу» записуємо туди значення 1 (Рис.1.20).

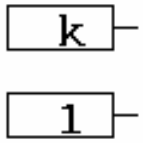


Рис.1.20

Далі додати на робочу область «Вихід», тобто операцію запису значення в змінну. За аналогією зі «Вход», натискають ЛКМ на іконці  і встановлюють «Выход» в потрібне місце робочої області, або використовують контекстне меню (Рис.1.17), в якому вибирають пункт «Выход». Перетягування цього зображення проводиться аналогічно «Вход», однак область, за яку треба «вхопитися» курсором, розташована у «Выход» праворуч (рис.1.21).

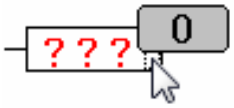


Рис.1.21

Поки не звертайте уваги на цифру 0 поруч з «Выход». З нею визначимося пізніше. За умовою задачі необхідно результат роботи програми записати назад в змінну k. Тому замість знаків питання в «Выход» записуємо ім'я змінної і натискаємо Enter. Отриманий результат видно на рис. 1.22.

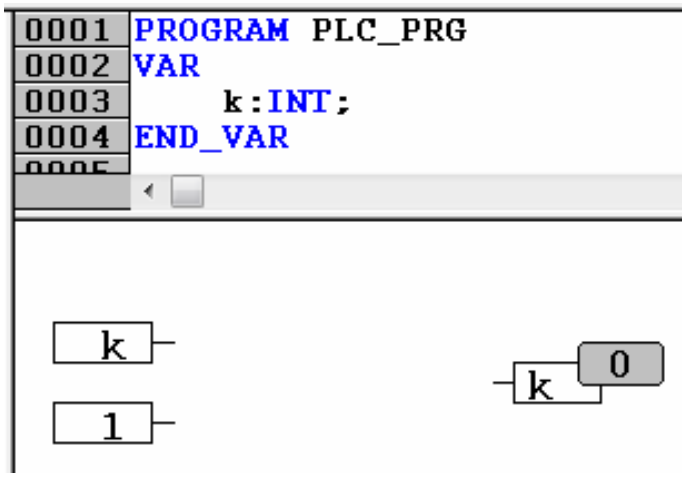


Рис.1.21

Для реалізації операції складання, результат якої необхідно передати на «Выход», тобто записати в змінну ***k*** використовуємо «Элемент», натиснувши ЛКМ на іконці.

Однойменний пункт є в контекстному меню (Рис.1.17), яке можна викликати, натиснувши на вільному місці робочої області ПКМ. «Элемент» розміщуємо між входами і виходом, як показано на рис. 1.22.

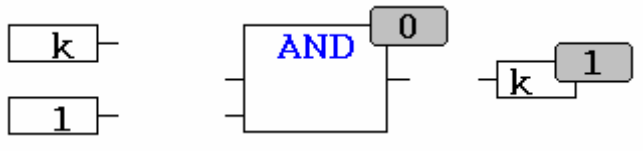


Рис.1.22

«Элемент», який називається блоком, це найпростіша операція. Зліва до нього підводяться вихідні дані, праворуч можна отримати результат і по лініях зв'язку передати в іншу частину алгоритму, наприклад, як вихідні дані для іншого блоку. Тема елемента визначає, яку операцію цей блок буде виконувати.

Практично всі алгоритми на мові CFC виглядають, як набір блоків, входів і виходів. Алгоритми можуть містити й інші елементи. Однак без трьох, розібраних вище, зазвичай обійтися не вдається. Вони складають основу програмування на мові функціональних блоків CFC.

Операція складання, якою нам необхідно скористатися, має в CoDeSys заголовок ADD. За замовчуванням в «Элемент» підставляється інший заголовок – AND – логічне «І». Натиснувши ЛКМ на заголовку блоку, надрукувавши потрібне нам ADD. Потім необхідно зазначити для операції додавання вихідні дані.

Іншими словами, підвести значення змінної ***k*** і одиницю, використовуючи лінії зв'язку.

Для цього ми натискаємо ЛКМ на «хвостику» входу ***k*** (рис. 1.23). Утримуючи ЛКМ, акуратно перетягуємо лінію зв'язку до «хвостику» зліва від операції ADD (рис. 1.24, 1.25) і потім відпускаємо ЛКМ. Тепер змінну ***k*** і блок ADD з'єднує лінія зв'язку (рис. 1.26).

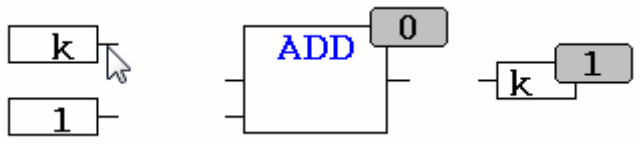


Рис. 1.23

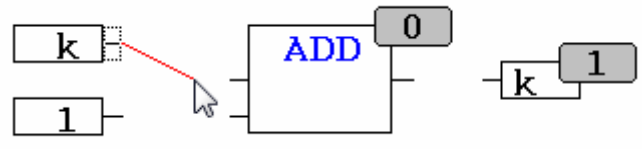


Рис. 1.24

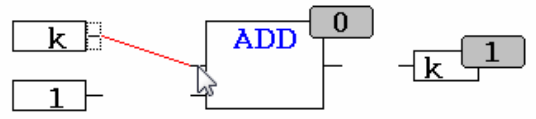


Рис. 1.25

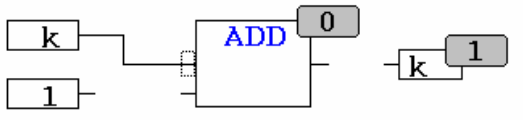


Рис. 1.26

Таким же чином з'єднують другий вхід і вихід з відповідними «хвостами» ADD (1.27), а саме входами і виходами блоку, маючи на увазі, що на входи подаються вихідні дані, а з виходу можна забрати результат операції.

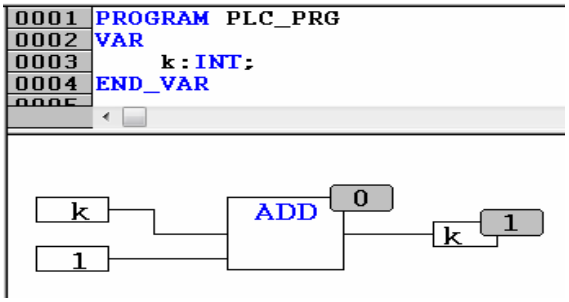


Рис. 1.27

На рис. 1.27 програма набула практично закінчений вид.

Щодо нумерації, яка з'явилася при додаванні на робочу область виходу і блоку ADD. У даному випадку число 0 або 1 в правому верхньому куті блоку (операції) визначає порядок, в якому система буде обробляти дії програми. Виходячи з нумерації на рис. 1.27 спочатку система складе значення змінної ***k*** і одиницю, і тільки потім отриманий результат буде знову поміщений в змінну. Така послідовність є цілком логічною.

При *написанні алгоритму CoDeSys автоматично розставляє порядок обробки дій в напрямку з лівого верхнього в правий нижній кут.* Іншими словами, якщо додається операція вище або правіше, то в програмі ця дія буде виконуватися раніше. Однак коли починають вносити виправлення, копіювати елементи і частини програми, перетягувати їх на нове місце, порядок може порушуватися. Причому, чим більше алгоритм, тим складніше відстежити ці невідповідності. А вони можуть викликати непрацездатність, здавалося б, правильно написаної програми. Дійсно, якщо спочатку використовувати результат складання ADD, а тільки потім його формування, то в змінну ***k*** весь час будуть записуватися неправильні значення. Таким чином, *необхідно стежити за дотриманням правильного порядку*. Особливо це важливо перед тим, як буде запускатися алгоритм для перевірки.

Відновити правильний порядок в CoDeSys не складно: для цього натискається ПКМ на вільному місці робочої області і в контекстному меню вибираємо пункт «Порядок», а потім підпункт «В соответствии с потоком данных» (рис. 1.28). Після цього система автоматично відновить порядок, взявши за початок лівий верхній елемент і від нього рухаючись по лініях зв'язку до кінця програми. Початкові операції повинні розташовуватися в робочій області вище і правіше. Після розстановки порядку, як і після інших важливих змін, корисно зберегти проект  .

Відзначають ще раз один важливий момент, важливий для мови CFC. Після внесення виправлень, нехай навіть мінімальних, перед тим, як запускати проект на перевірку, необхідно розставити порядок обробки операцій, використавши потрібний пункт контекстного меню (рис. 1.28).

Щоб розставляти порядок для кожної дії самостійно див. Пункти контекстного меню нижче напису «В соответствии с потоком данных». Набагато правильніше і набагато простіше і зручніше відразу розміщувати блоки в потрібному місці робочої області і потім кожен раз автоматично приводити все в порядок. Звичайно, в інших мовах програмування, наприклад в ST (Структурований текст), немає необхідності в розстановці порядку.

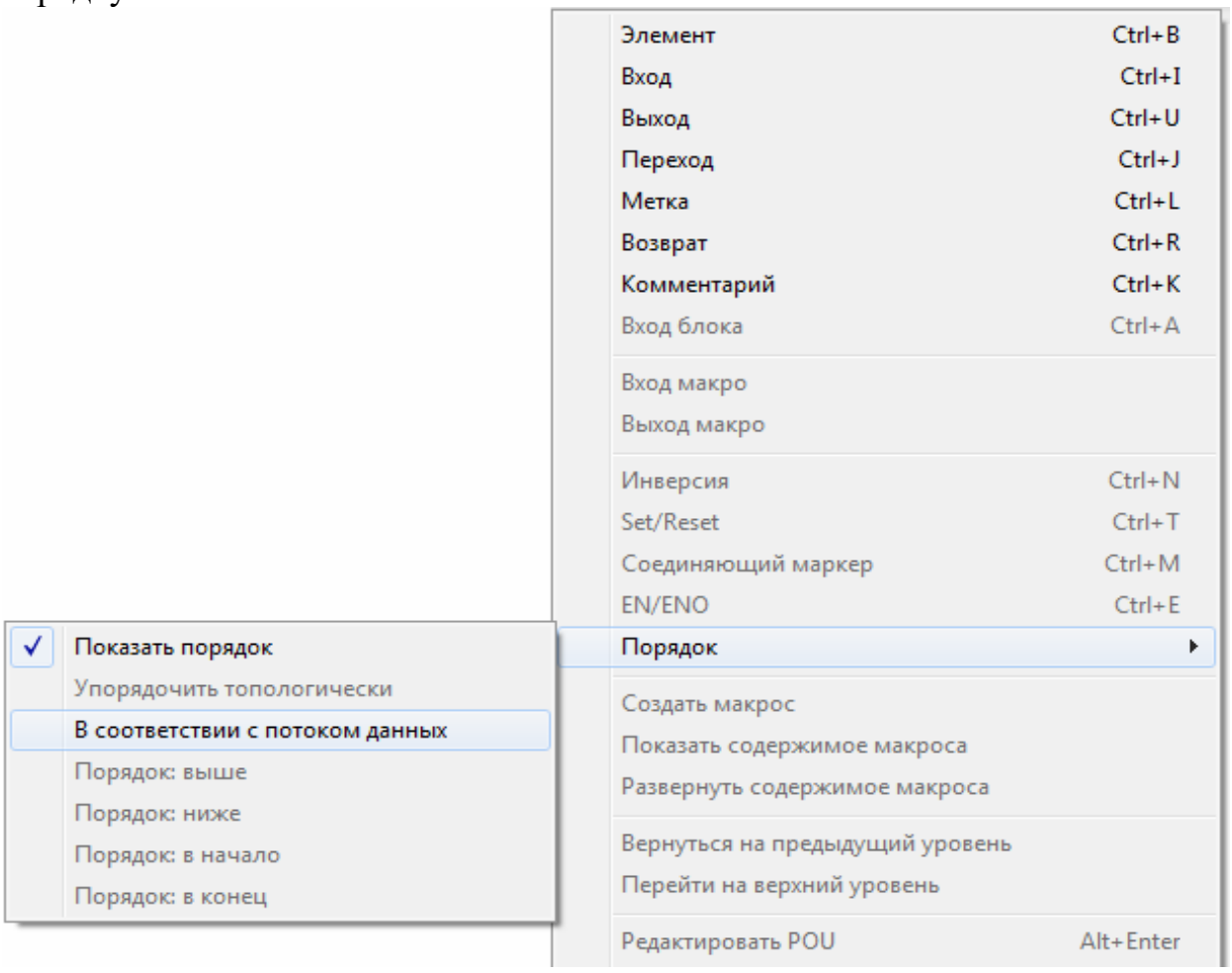


Рис. 1.28

1.4 Запуск проекту на виконання

Доцільно перевіряти виконану роботу. Для початку системі CoDeSys необхідно оцінити його на наявність помилок, критичних для виконання алгоритму. Крім того, системі необхідно скомпілювати проект або, з нашого графічного представлення алгоритму отримати машинний код, який в подальшому буде завантажуватися безпосередньо в пам'ять ОВЕН ПЛК.

Очевидно, що нулі і одиниці контролеру набагато зрозуміліше прямокутників. Для виконання перевірки і компіляції необхідно в головному меню CoDeSys вибрати вже знайомий розділ «Проект», і далі знайти в списку пункт «Компилировать все» (рис. 1.29).

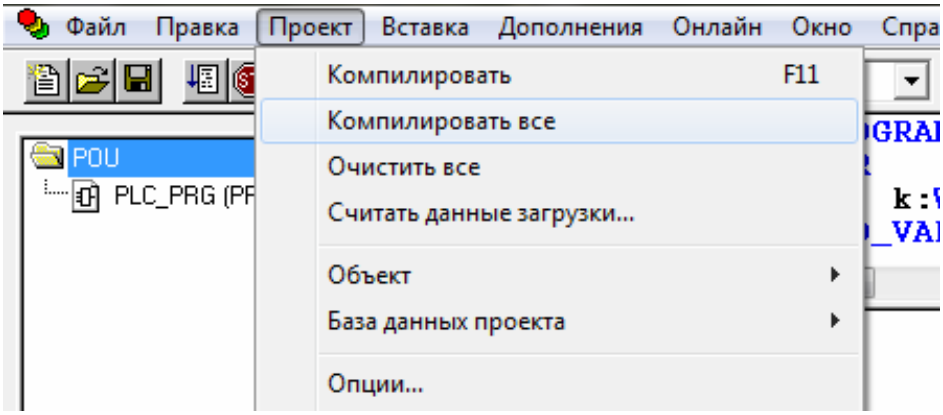


Рис. 1.29

Після цього система зробить необхідний аналіз і під робочою областю у вікні повідомлень побачите статистику проекту. Цифри можуть дещо відрізнятися від наведених на рис.1.30, це нормально. Важливо, щоб в самій нижній сходинці було зазначено «0 помилок». У цьому випадку можна запускати проект на виконання.

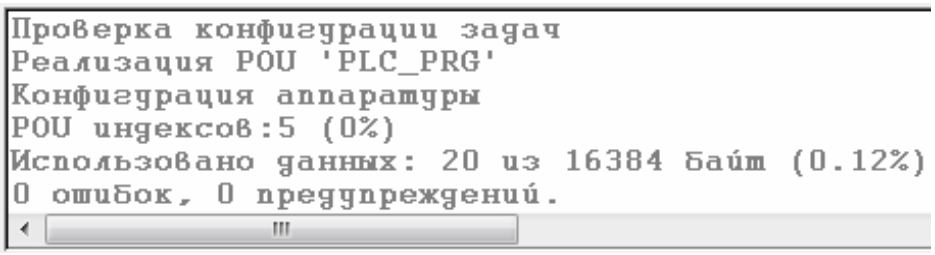


Рис. 1.30

Якщо система виявила помилки, треба переглянути їх список у вікні повідомлень, для цього двічі натиснути ЛКМ на підсвічені червоним рядки і оцінити, які елементи в робочому вікні або в області визначення виявляться виділеними.

Наприклад, помилка на рис. 1.31 говорить про те, що в області визначення в кінці оголошення змінної ***k*** не поставлена обов'язкова крапка з комою. Правильний варіант подивіться на рис.1.15.

Рис. 1.31

В іншому прикладі на рис. 1.32 програма містить виклик неправильної операції AND.

Необхідно використовувати блок ADD для реалізації складання. Тому в заголовок блоку необхідно внести виправлення. Після цього корисно ще раз скомпілювати проект, наприклад, використовуючи клавішу F11.

Часом система одну помилкову дію описує кількома повідомленнями. Тому нехай не дивує, що після одного виправлення пропадають кілька червоних рядків.

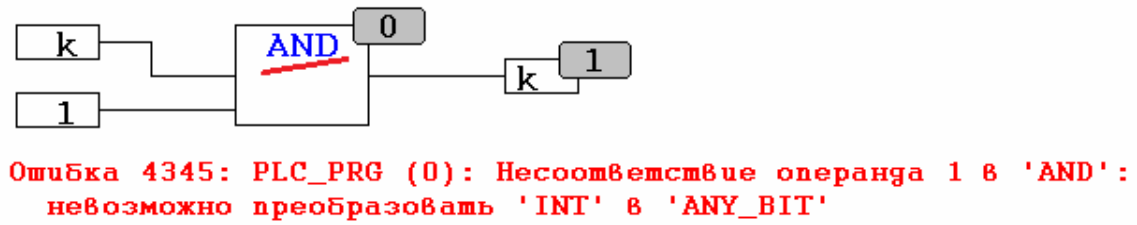


Рис. 1.32

Розберемо помилки, які виникають найчастіше. Якщо всі помилки виправлені або якщо вони не були допущені, після компіляції в вікні повідомлень ми маємо напис (рис.1.33):

Рис. 1.33

Для запуску проекту в меню «Онлайн» ми вибираємо пункт «Подключение» (рис. 1.34). Також можна використовувати поєднання клавіш Alt + F8 або іконку  .

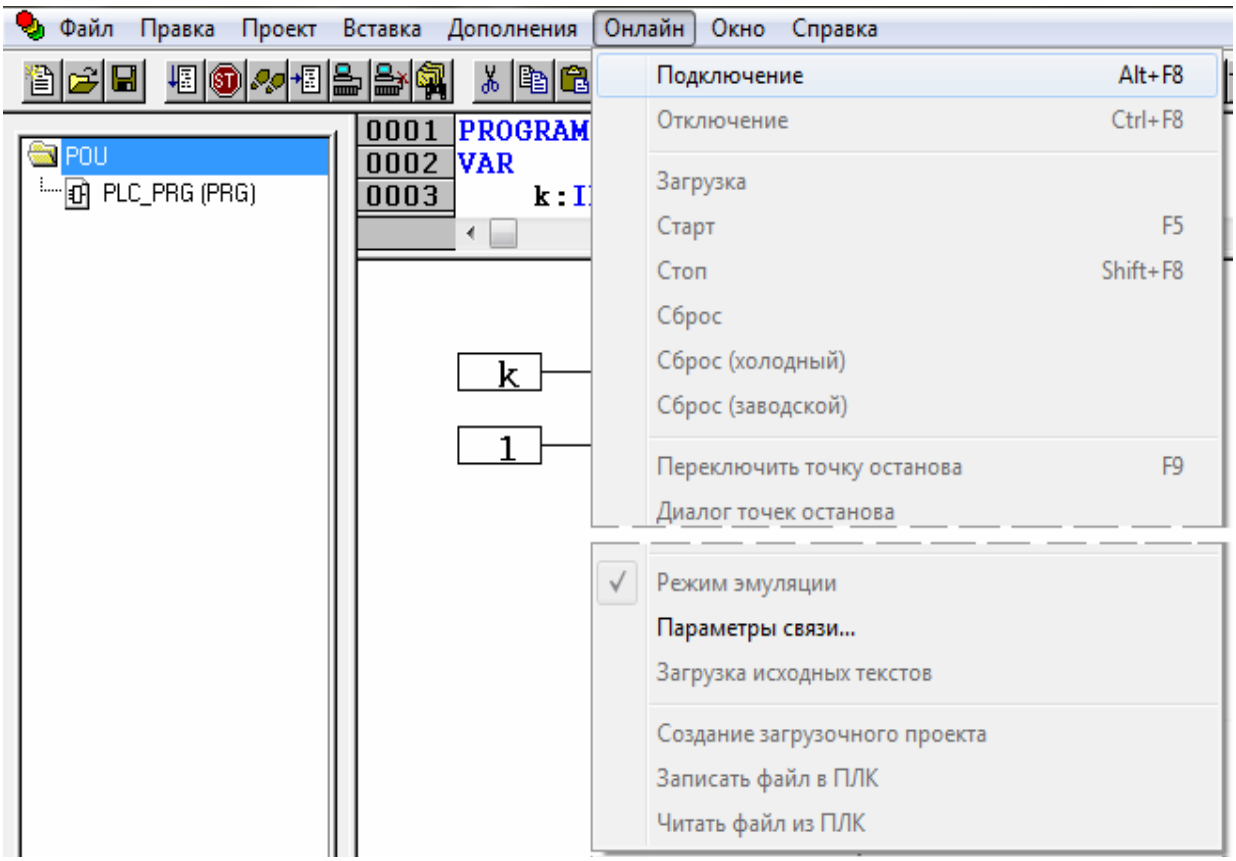


Рис. 1.34

Якщо з’явиться повідомлення з попередженням (рис. 1.35), то необхідно натиснути «ОК» і повернутися до виправлення помилок.

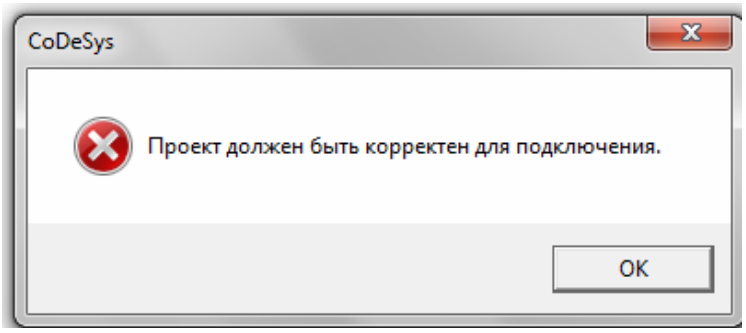


Рис. 1.35

Якщо все в порядку, то при запуску проекту на виконання зовнішній вигляд системи трохи змінюється (Рис. 1.36). В області визначення бачимо змінну і її поточне значення. Також значення змінної і результат роботи блоку видно на робочій області.

З цієї точки зору робота з мовою CFC є дуже зручною і наочною, тому що завжди зрозуміло які дані в якій частині програми формуються і куди передаються. Крім того, в програму додається операція RETURN. Система робить це автоматично для організації так званого циклу роботи ПЛК, про нього поговоримо трохи пізніше.

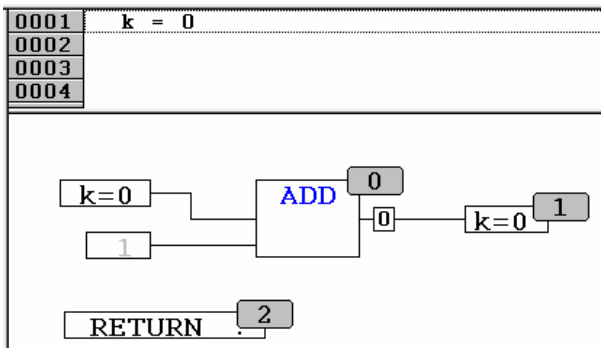


Рис. 1.36

До теперішнього моменту система програмування перебувала в режимі редагування: змінювалася програма, визначалися змінні. При цьому в рядку стану, в правому нижньому кутку екрану можна бачити напис «Онлайн». Поки редагується проект, цей напис має неактивний, сірий колір, що є підказкою. Після вибору пункту «Подключение» в меню «Онлайн» система перейшла в режим виконання. При цьому в рядку статусу напис «ОНЛАЙН» стала активною (Рис. 1.37).

Рис. 1.37

У режимі онлайн (інша назва режиму виконання) вже немає можливості вносити зміни по ходу процесу. Але можна перевіряти алгоритм на відповідність заявленим вимогам, задавати різні значення змінних, зупиняти і запускати процес у потрібні нам моменти.

Якщо до комп'ютера не підключено контролер, то в CoDeSys є можливість емулювати роботу алгоритму. Це відбувається безпосередньо на комп'ютері, без завантаження проекту в ОВЕН ПЛК. Емуляція дозволяє в достатній мірі вивчити самі принципи програмування. Однак без урахування особливостей тієї чи іншої модифікації ПЛК, без використання фізичних входів і виходів, підключення модулів розширення або операторських панелей побудувати реально працюючу систему важко.

Про те, що система запустила проект в емуляції можна судити по тому ж рядку статусу (Рис. 1.33). Якщо напис «ЭМУЛ.» активний, значить програма працює на вашому комп'ютері, а не завантажена в контролер.

Якщо не вказуються в системі дані про ПЛК, який буде використано, то CoDeSys за замовчуванням завантажить проект в емуляції. Надалі при роботі з реальним пристроєм можна перемикатися між емуляцією і цільової платформою (виконанням алгоритму в ПЛК). Для цього перед виконанням операції рівні 0. Для запуску роботи програми необхідно зайти в меню «Онлайн» і вибрати пункт «Старт» (рис. 1.38). Однак в роботі набагато зручніше використовувати клавішу F5 або іконку  . Після цього напис «Запущено» в рядку статусу стане активним. А в робочій області побіжать

цифри в змінної ***k***. «Подключение» необхідно в меню «Онлайн» поставити або прибрати галочку напроти пункту «Режим эмуляции» (Рис. 1.34).

Відразу після запуску проекту на виконання система ставить його на паузу. Робиться це для того, щоб користувач був готовий до того моменту, коли алгоритм почне роботу. Про постановку на паузу можна судити по рядку статусу. Якщо напис «Запущено» не активний, як, наприклад, на рис. 1.37, система очікує команди на запуск. При цьому значення змінної і виходу операції ADD незмінні і ПЛК, так само як і його емуляція в системі програмування, працюють циклічно. Це означає, що контролер повторює всі описані інструкції через певні проміжки часу. В емуляції цей час дорівнює приблизно 50 мс.

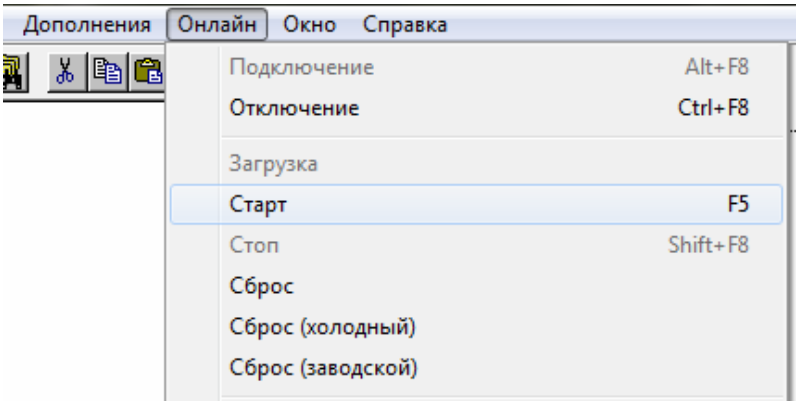


Рис. 1.38

У ОВЕН ПЛК в залежності від модифікації і складності алгоритму час циклу може становити до 1 мс. Таким чином, коли описують операцію складання і пропонувють системі записувати результат в змінну ***k***, то організується постійне зростання значення цієї змінної. Кожні 50 мс система забирає з ***k*** значення, додає до нього одиницю і знову записує в ***k***. На екрані це виглядає, як постійне збільшення числа. Цей процес можна зупинити, якщо поставити виконання алгоритму на паузу. Для цього можна вибрати в меню «Онлайн» пункт «Стоп» (рис. 1.39) або використовувати поєднання клавіш Shift + F8 або іконку  .

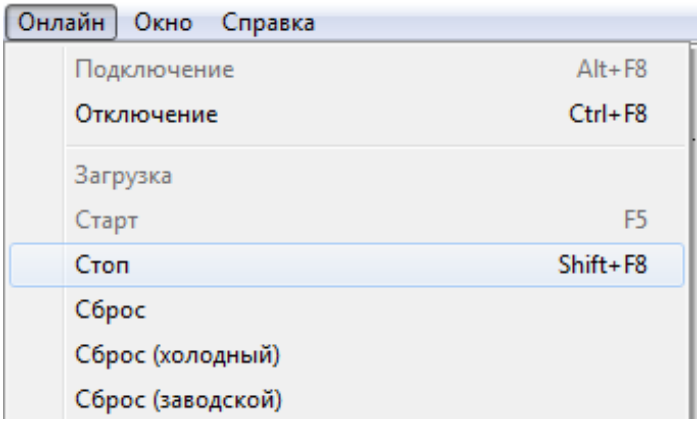


Рис. 1.39

Після натискання «СТОП» система зупиняє розрахунок і залишає на екрані останні отримані результати, тобто результати останнього відпрацьованого до кінця циклу.

Таким чином, з’являється можливість відстежити ті зміни, які відбулися під час роботи алгоритму, оцінити правильність його роботи. Наступний запуск проводиться, як і на початку роботи, через меню «Онлайн» і рядок «Пуск» або за допомогою кнопки F5. Часом, в процесі роботи може знадобитися вручну змінити ті чи інші значення, наприклад, можна обнулити змінну ***k***. Для цього необхідно в області визначення або на робочій області двічі натиснути ЛКМ на імені змінної і у вікні (Рис. 1.40) в нижньому полі ввести нове значення, а потім натиснути «ОК».

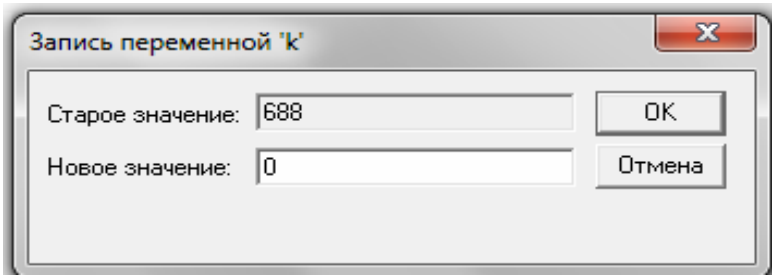


Рис. 1.40

Після цього в робочій області, праворуч від змінної блакитним кольором буде відображатися нове значення (рис. 1.41). Для введення нового значення замість поточного необхідно вибрати в меню «Онлайн» пункт «Записати значення» або натиснути клавіші Ctrl + F7 (рис. 1.42). Після цього значення змінної ***k*** обнулится.

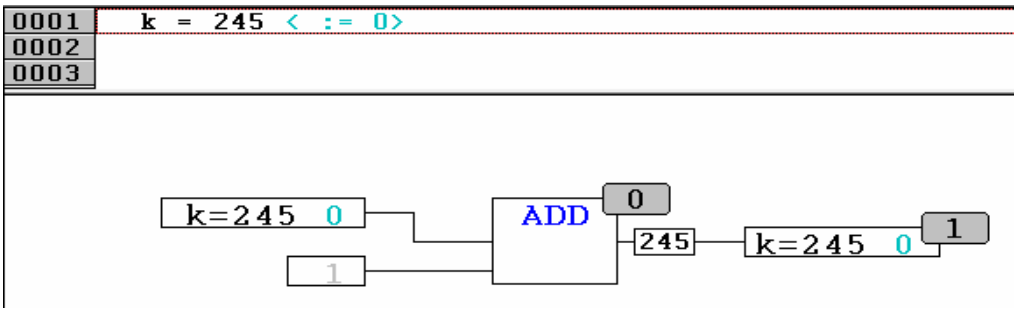


Рис. 1.41

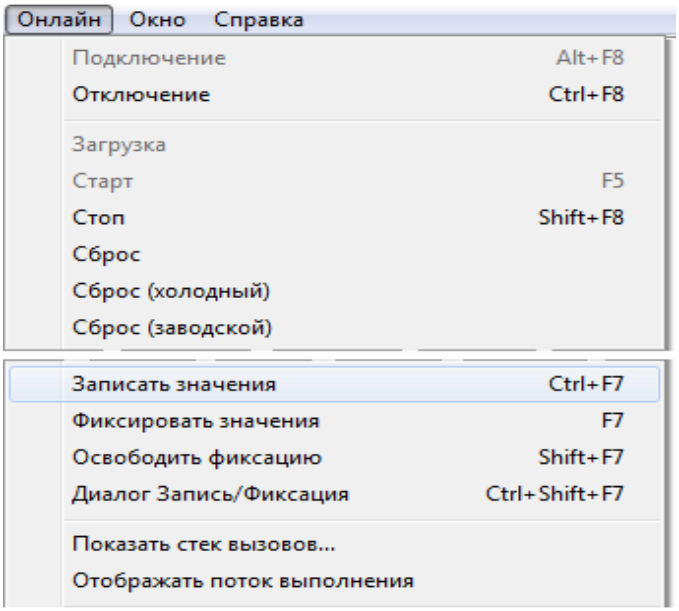


Рис. 1.42

Для того щоб одночасно поміняти кілька змінних, спочатку готуються потрібні значення. Ці значення блакитним кольором відображаються у робочій області. А вже потім натискається Ctrl + F7, і вибрані змінні одночасно змінюються в програмі. Якщо в процесі роботи з режимом виконання необхідно обнулити значення всіх змінних, тобто скинути їх в початковий стан, використовується пункт «Сброс» меню «Онлайн» (рис. 1.42). Після виконання цієї операції програма автоматично стає на паузу. Про це можна судити по неактивній написи «Запущено» в рядку статусу (Рис. 1.37). Далі, натискаючи кнопку F5, знову запускаємо роботу алгоритму з самого початку.

Якщо після запуску програми на виконання вона не працює, тобто значення змінних не змінюються, необхідно звернутися до рядка статусу.

Перевірте, чи запущена програма, і якщо вона на паузі, скористайтеся кнопкою F5. Після того, як написали, запустили і перевірили перший проект, необхідно повернутися в режим редагування, якщо є потреба внести доповнення або зміни в алгоритм. У процесі роботи програми в емуляції або на ПЛК зробити такі зміни не вдасться. Тому для повернення до редагування проекту ми вибираємо в меню «Онлайн» пункт «Відключення».

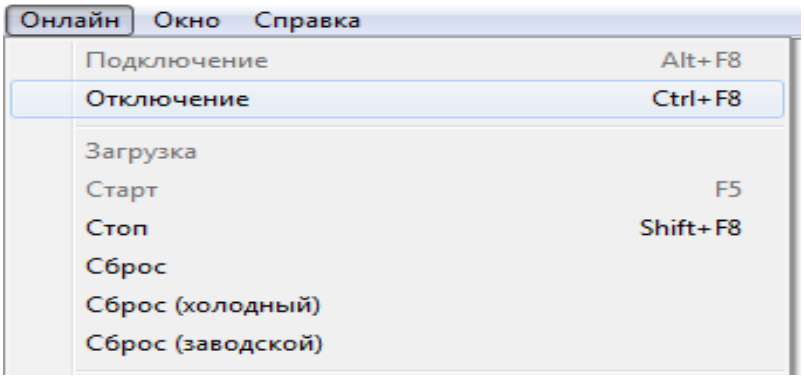


Рис. 1.43

Крім того, для операції відключення можна використовувати іконку  або поєднання клавіш Ctrl + F8. Вікно повернеться до колишнього вигляду, з яким працювали при написанні нашого першого прикладу.

2. Входи і виходи

Вивчимо організацію роботи з фізичними входами і виходами ПЛК, тобто як інформацію про стан об'єкта управління передати в програму через фізичні входи контролера, і як з програми управляти об'єктом через виходи ПЛК.

2.1 Target-файл

Визначимо, як передати системі інформацію про технічні характеристики цільової платформи, тобто контролера. Якщо говорити про ОВЕН ПЛК110, точніше про модифікацію ПЛК110-30.Р-L, то необхідно виділити наступні важливі характеристики:

- 18 дискретних входів;
- 12 дискретних виходів (е / м реле);
- призначена для користувача пам'ять - 3 Мб;
- 2 інтерфейсу RS-232;
- 2 інтерфейсу RS-485;
- інтерфейс Ethernet;
- інтерфейс для програмування USB-device,
- вбудований годинник реального часу.

Реальна кількість характеристик значно більше, адже системі програмування необхідно знати тип процесора, під який необхідно створювати програмний код, обсяг доступної для використання оперативної пам'яті ПЛК і т.д.

Прописувати вручну все це важко, тому із врахуванням CoDeSys прийнятий наступний інструмент: компанія виробник контролерів, в даному випадку – ОВЕН, надає так звані файли цільової платформи, що містять в собі всю необхідну системі інформацію про використаний ПЛК. Ці самі файли ще називають для стислості target-файлами на CD-диску (від англійського target). До виробу ОВЕН ПЛК додається дистрибутив CoDeSys (розділі 1.1, target).

Перед установкою target-файлів корисно закрити всі запущені на комп'ютері екземпляри CoDeSys. Потім, запустивши CD і опинившись в розділі «Зміст компакт-диска», натиснути кнопку «Программы» (рис. 2.1). Далі вибираємо кнопку «Встановити «target-файл», а потім «Установить target-файлы версии 2.10».


Navigator OWEN PLC110-160 v7
Содержание компакт-диска
Документация
Программы
Обучение
Дополнительно
Контакты
Smart Software Solutions
Выход
Navigator OWEN PLC110-160 v7
Программы для ПЛК (часть 1)
Установить CoDeSys 2.3.9.26
Установить Target-файл
Замена встроенного ПО
Библиотеки CoDeSys
Библиотеки ОВЕН
Продолжение
Перед установкой таргет-файла убедитесь что его версия поддерживается прошивкой ПЛК. При установке убедитесь в соответствии таргет-файла модификации ПЛК
Содержание
Выход
Navigator OWEN PLC110-160 v7
Target-файлы и прошивка ПЛК110/160
Установить Target-файлы версии 2.10
Действующая прошивка
Программа обновления ПО версии 2.14.0 для ПЛК110-30
Программа обновления ПО версии 2.14.0 для ПЛК110-32
Программа обновления ПО версии 2.14.0 для ПЛК110-60
Программа обновления ПО версии 2.14.0 для ПЛК160
История изменений прошивок
В данном разделе находится следующее программное обеспечение:
1) Target-файлы, необходимые для написания проекта в среде программирования CoDeSys.
2) Программа обновления встроенного ПО контроллеров ОВЕН ПЛК (прошивка).
Внимание! Установленный на компьютере Target-файл должен соответствовать прошивке контроллера.
Прошивкам 2.05.X - 2.07.X соответствует таргет файл версии 2.07
Прошивкам 2.10.X - 2.14.X соответствует таргет файл версии 2.10
Содержание
Назад
Выход

Рис. 2.1

На комп’ютері буде запущена спеціалізована програма-майстер (рис. 2.2): необхідно двічі натиснути кнопку «Далее», а потім один раз «Установить». Після цього майстер встановить файли цільової платформи для всіх існуючих на поточний момент модифікацій ОВЕН ПЛК110 і ПЛК160. Їх близько десятка (використовуються зазвичай 2-3 модифікації)

.

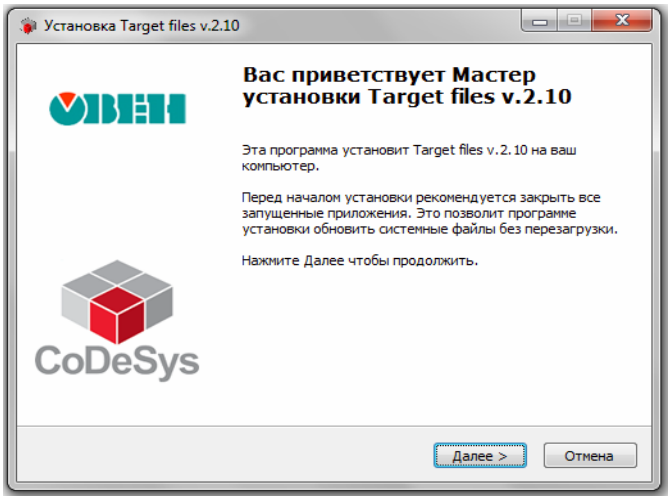


Рис. 2.2

Розглянемо ще один спосіб установки target-файлів при якому можна вибрати тільки ті модифікації, які дійсно потрібні. Для цього необхідно відкрити вміст вашого CD.У папці targets \ Версія 2.10 існує декілька архівів з англомовними назвами ПЛК. Наприклад, розглянута вище модифікація ПЛК110-30.P-L тут матиме назву PLC110.30_l (рис. 2.3).

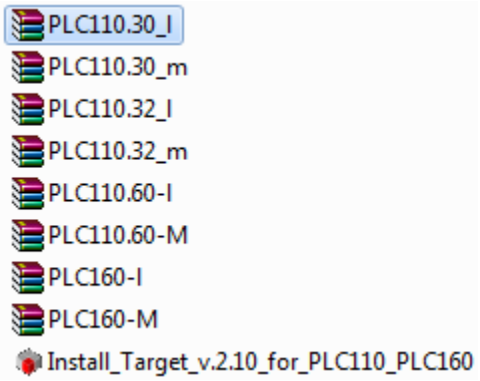


Рис. 2.3

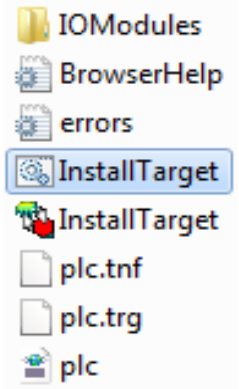


Рис. 2.4

Завдання – розпакувати вміст цього файлу PLC110.30_l. В отриманій при цьому папці запускається файл InstallTarget.bat. (Рис. 2.4). Після цього на екрані на короткий час з’явиться вікно завантаження. Потім процедура установки буде завершена, необхідний файл буде встановлено до відповідних директорії. *Звертати увагу до розширення * .bat.* У папці також міститься файл InstallTarget.exe (Завдання – розпакувати вміст цього файлу PLC110.30_l.) В отриманій при цьому папці ми запускаємо файл InstallTarget.bat. (Рис. 2.4). Після цього на екрані на короткий час з’явиться

вікно завантаження. Потім процедура установки буде завершена, необхідний файл буде встановлено до відповідних директорії. Будьте уважні до розширення * .bat. в папці також міститься файл InstallTarget.exe ( ). Робота з ним детально описана в керівництві з експлуатації ОВЕН ПЛК.

2.2 Додавання target-файлу в проект CoDeSys

Після того, як файл цільової платформи встановлений на комп'ютері, його можна додати в проект. Таким чином, однозначно визначається, для якої модифікації ПЛК буде створюватися в подальшому той чи інший алгоритм. Для початку треба створити новий проект.

Для цього в меню «Файл» необхідно вибрати пункт «Создать». Найперше вікно, яке з'являється при створенні нового проекту, це вікно «Настройка целевой платформы» (рис. 2.5). Робота з ним детально описана в керівництві з експлуатації ОВЕН ПЛК.

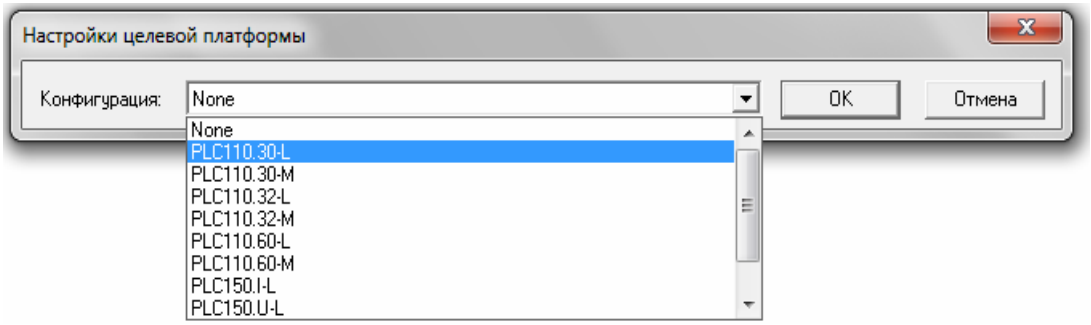


Рис. 2.5

Натискаємо ЛКМ на напис «none» і в списку, що розкривається бачимо все ті target-файли, які встановили в попередньому підрозділі. Серед них вибираємо PLC110.30-L. Саме з цієї модифікацією ПЛК будемо працювати. Вікно прийме трохи інший вид, відкриється доступ до додаткових налаштувань. Однак нам поки всі ці параметри не потрібні, за замовчуванням там все необхідне вже зроблено. Натискаємо «ОК» (рис. 2.6).

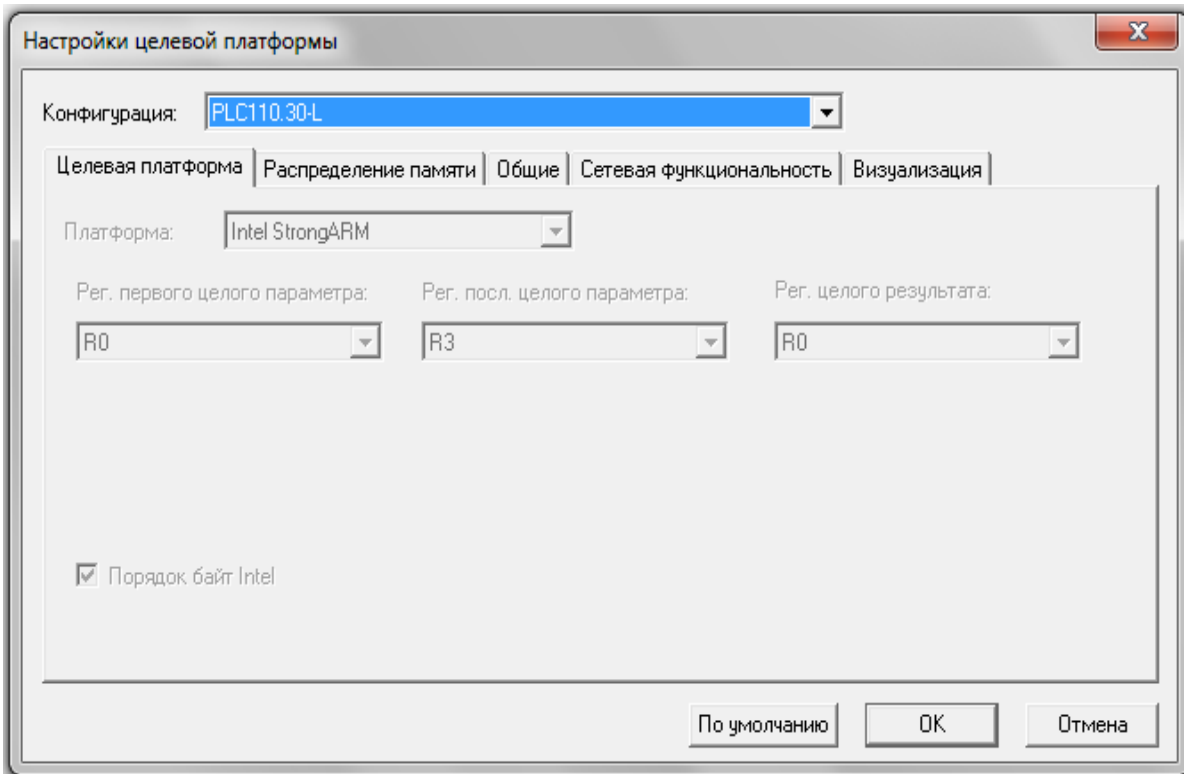


Рис. 2.6

У наступному вікні (рис. 2.7) система програмування пропонує вибрати нам мову реалізації. Можна вибрати мову функціональних блоків CFC згідно з малюнком і натиснути «ОК». Потім проект корисно зберегти ().

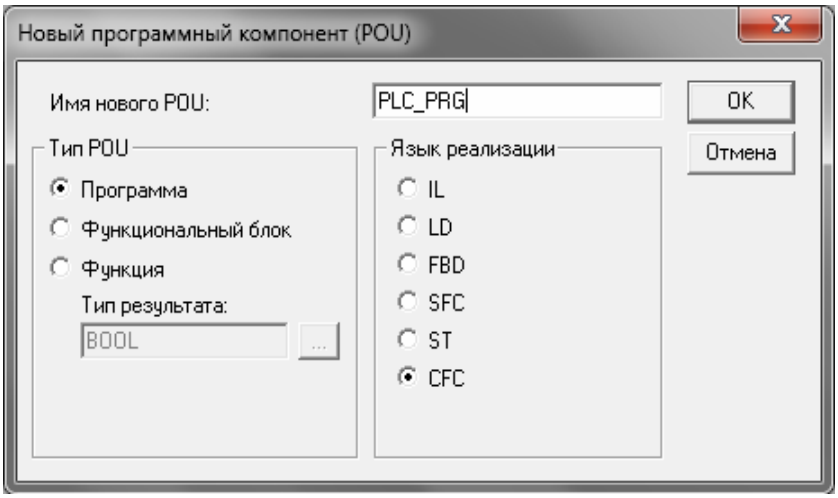


Рис. 2.7

2.3 Оголошення змінних для входів і виходів

Практично всі дані, які використовуються при описі алгоритму, в CoDeSys задаються у вигляді змінних. Змінні мають ім’я і тип даних, тобто тип значення, яке зберігається в цієї змінної. Значення, які ПЛК отримує зі входів і передає на виходи, також використовуються в програмі у вигляді змінних. Завдання – визначити, яка змінна буде зберігати значення того чи іншого виходу або входу. Визначення змінних, пов’язаних зі входами і

виходами (глобальні змінні), проводиться в ресурсі «Конфигурация ПЛК». Для цього в менеджері об’єктів переходимо на крайню праву вкладку «Ресурсы» (рис. 2.8). Потім вибираємо потрібний пункт, двічі клацаючи ЛКМ на написи Конфигурация ПЛК.

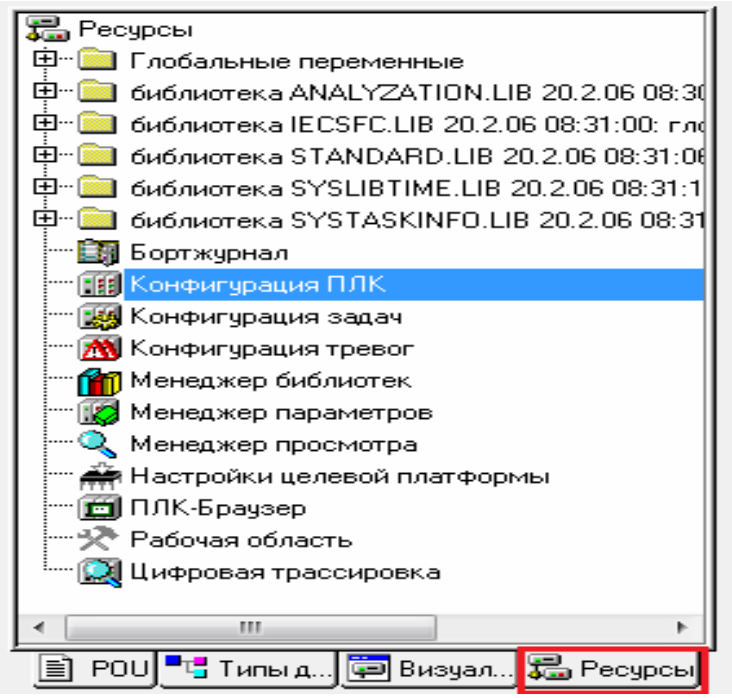


Рис. 2.8

Зліва від менеджера об’єктів відкриється вікно конфігурації. Натискаємо ЛКМ на ⊞ поруч з написом PLC110_30, таким чином розкриваємо конфігурацію контролера (рис. 2.9). Тут представлені власні входи і виходи ПЛК. Послідовно натискаючи ЛКМ на поруч з відповідною групою, можна побачити всю конфігурацію, що включає в себе швидкісні і звичайні дискретні входи (рис.2.10), швидкісні і звичайні дискретні виходи (рис.2.11).

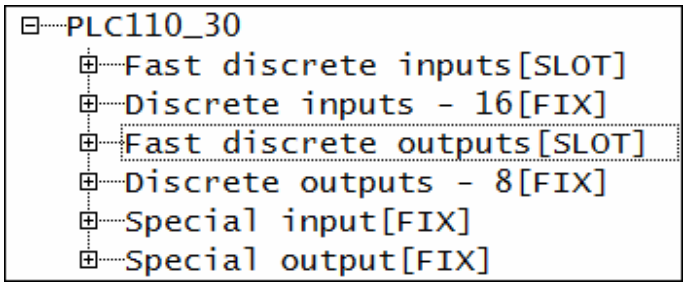


Рис. 2.9

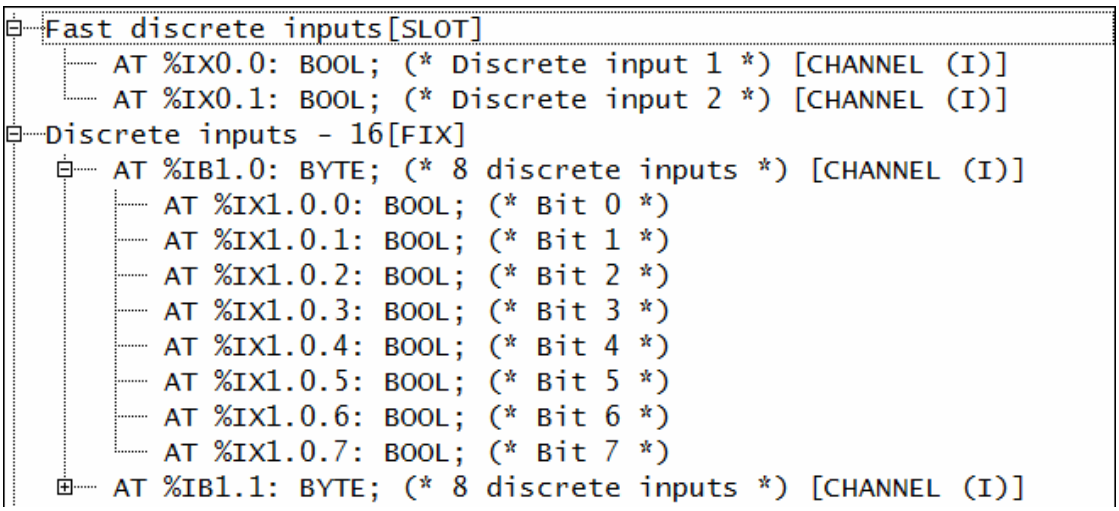


Рис. 2.10

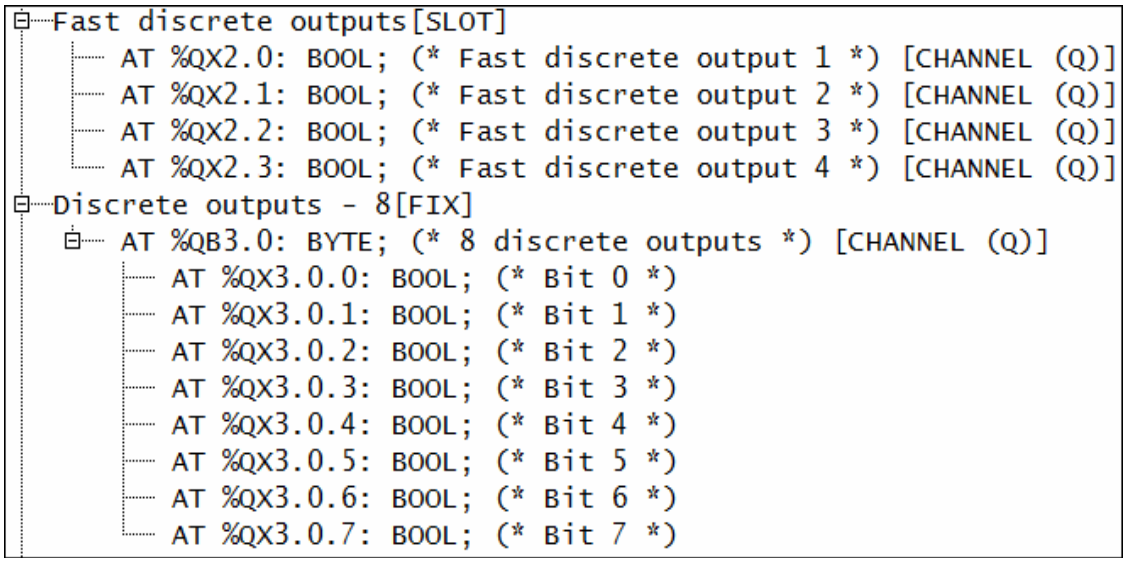


Рис. 2.11

Необхідно підкреслити, що при виборі модифікації ПЛК з релейними виходами (буква Р в позначенні відповідної модифікації, наприклад, ПЛК110-30.220.Р-L) швидкісні виходи не передбачені.

Однак структура конфігурації області введення-виведення (раніше назвали конфігурацією ПЛК) залишається аналогічною моделям, в яких швидкі виходи присутні. Тобто, релейні виходи в пункті Fast discrete outputs працюють з тією ж швидкістю, що і виходи в групі Discrete outputs.

Для того, щоб використовувати в алгоритмі значення на тому чи іншому вході або виході, необхідно навпроти відповідного рядка в конфігурації ПЛК задати ім'я змінної.

Надалі це ім'я використовується при написанні програми. Для завдання імені необхідно двічі натиснути ЛКМ на написи AT у відповідній рядку, і в розпочатому невеликому віконці надрукувати ім'я змінної. Ім'я повинно бути записано латинськими буквами і цифрами в одне слово.

На рис. 2.12 подібним чином вказується ім’я змінної ***in1***, пов’язуючи її з першим швидкісним входом ПЛК.

Після завдання імені необхідно натиснути Enter. Тепер у залежності від наявності або відсутності сигналу на першому вході ПЛК в змінної ***in1*** буде з’являтися значення «Істина» (TRUE) або «Ложь» (FALSE).

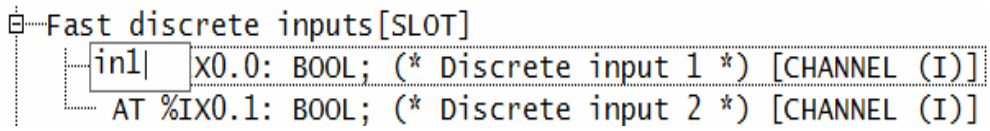


Рис. 2.12

Аналогічним чином необхідно задати ще три входи, один швидкісний і два звичайних, присвоївши їм імена ***in2***, ***in3*** і ***in4*** відповідно (рис. 2.13). Звичайні входи ПЛК110-30 розбиті на дві групи по вісім входів.

Розкриваємо спочатку блок Discrete inputs, а потім одну з груп 8 discrete inputs. Уже всередині групи навпроти потрібних рядків прописуєте імена змінних in3 і ***in4***. Тим же способом задаються імена змінних для виходів ПЛК.

Вам необхідно розкрити блок Fast discrete outputs, натиснувши ЛКМ на символі ⊞. Потім, двічі натиснувши ЛКМ на буквах AT, для трьох перших виходів вказати імена змінних ***out1***, ***out2*** і ***out3***. Зроблені зміни корисно зберегти. Після цього конфігурація ПЛК повинна прийняти вигляд, зображений на рис. 2.13.

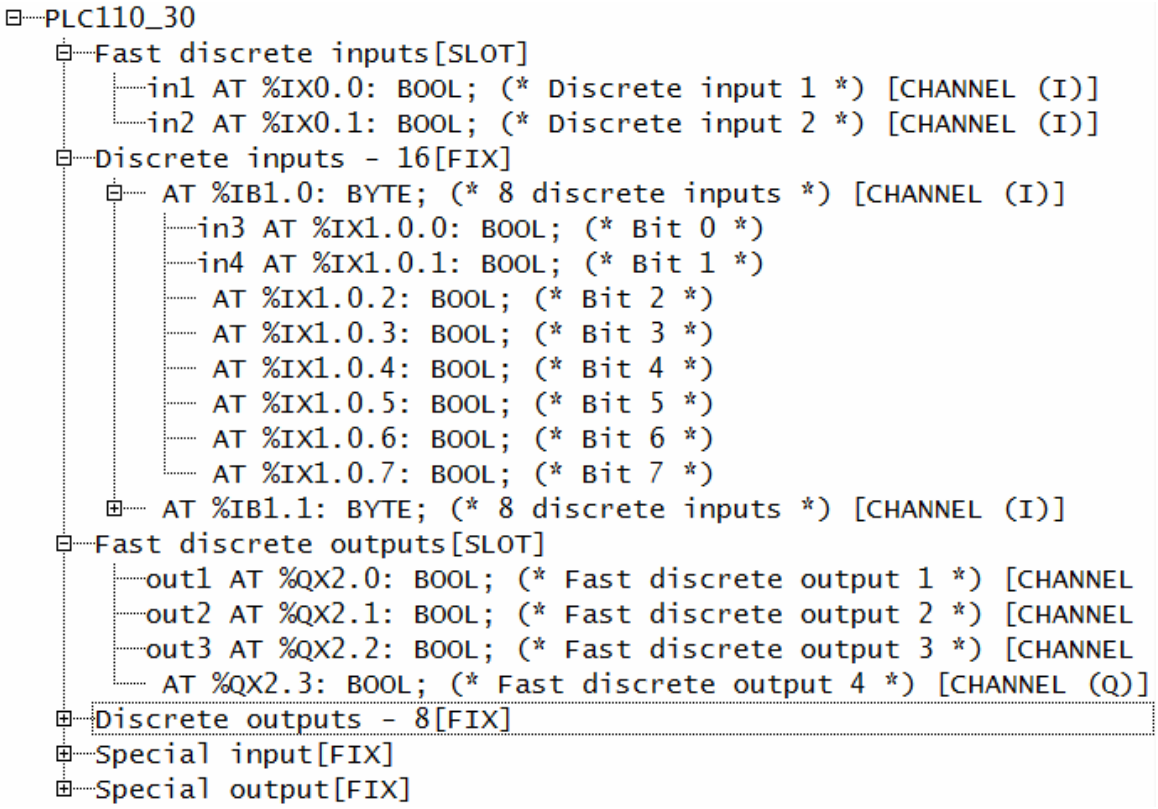


Рис. 2.13

2.4 Операції присвоювання дискретних значень

Задавши імена змінних для входів і виходів, визначивши ці змінні в проекті (в будемо називати такі змінні глобальними), отримуємо можливість використовувати входи і виходи ПЛК в якому-небудь простому завданні. Наприклад, задавати стан виходу в залежності від сигналу на вході.

Переходимо до редагування головної програми PLC_PRG. Для цього в менеджері об'єктів натискаємо ЛКМ на вкладці POU. Потім знаходимо напис PLC_PRG (PRG) і двічі натискаємо на ній ЛКМ (рис. 2.14).

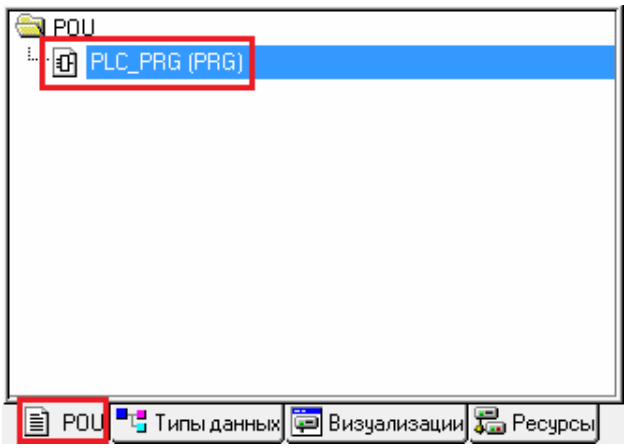


Рис. 2.14

Використовуючи кнопки «Вход» ( ) і «Выход» ( ) або пункти контекстного меню, переносимо на робочу область заготовку алгоритму (рис. 2.15) (детально в розділі 1). У прикладі перший вхід ПЛК буде керувати станом першого виходу.

Другий вхід буде впливати на другий вихід. Що стосується третього виходу, то його стан буде визначатися сигналами на третьому і четвертому входах.

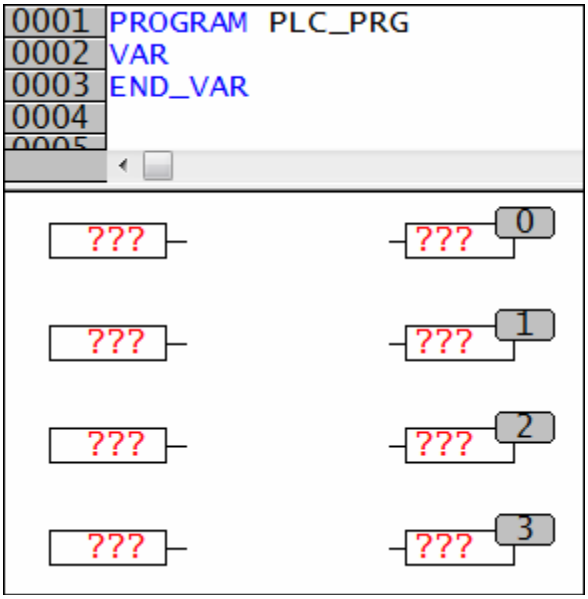


Рис. 2.15

Нагадаємо, що під час написання алгоритму замість червоних знаків питання необхідно підставити імена змінних. Наприклад, в лівому верхньому елементі ми будемо викликати значення на першому вході ПЛК, тобто значення змінної ***in1***. Необхідно виділити вміст елемента, натиснувши всередині нього ЛКМ (???). Потім з клавіатури задається ім'я змінної.

Якщо пам'ятаєте назви своїх змінних, оголошених в проекті, написати відповідне ім'я буде не важко.

Бувають ситуації, коли імена змінних складні та їх багато, є шанс помилитися в написанні, тому можна скористатися наступною зручною функцією CoDeSys.

Виділяємо вміст елемента (знаки питання в нашому випадку), а потім на клавіатурі натискаєте кнопку F2. На екрані з'являється вікно «Ассистент ввода».

Як тільки та чи інша змінна оголошена в проекті, наприклад глобальна змінна в конфігурації ПЛК (рис. 2.13) або локальна змінна в області визначення (Рис. 1.16), її можна знайти в асистенті введення. Наприклад, необхідна змінна ***in1***, пов'язана з першим дискретним входом контролера. У вікні асистента введення, в лівій частині ми вибираємо категорію «Глобальные переменные». У відкритому справа списку виділяємо ЛКМ потрібну змінну і натискаємо «ОК» (рис. 2.16).

Після цього вікно асистента закривається, а в потрібному місці з'являється необхідна змінна. Залишається натиснути Enter на клавіатурі.

Потренуйтеся викликати вікно «Асистента введення» кнопкою F2. Задайте імена змінних для чотирьох верхніх елементів відповідно до рис. 2.17 і проведіть відповідні лінії зв'язку.

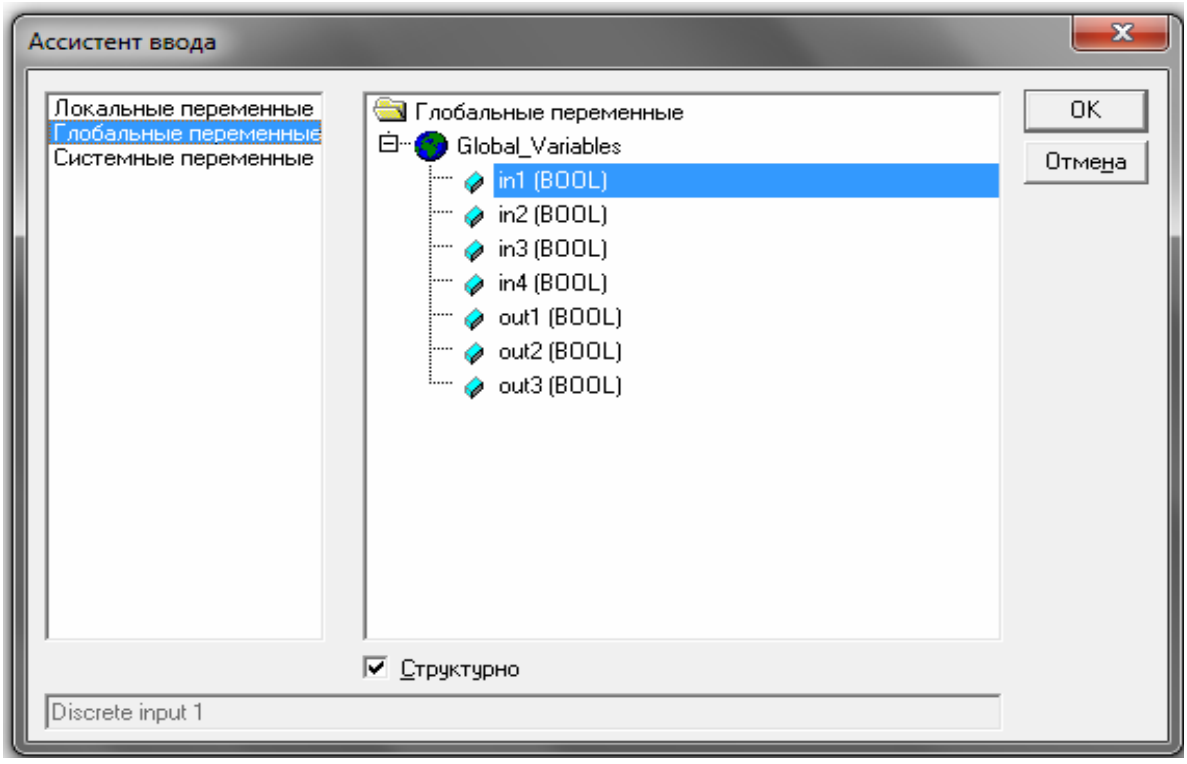


Рис. 2.16

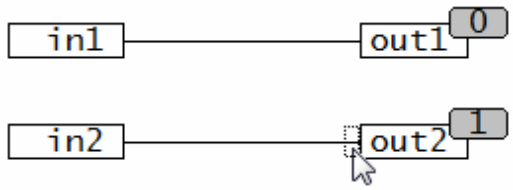


Рис. 2.17

Таким чином, при наявності сигналу на вході, тобто в змінних ***in1*** або ***in2***, відбувається спрацьовування певного виходу ***out1*** або ***out2***. І навпаки – у відсутності сигналу на вході вихід залишається розімкненим. Додамо в другий ланцюжок інверсію. Тобто задамо включення другого виходу тоді, коли немає сигналу на вході. Для цього необхідно виділити ЛКМ край лінії зв'язку поруч зі змінною ***out2*** (Рис. 2.17).

Після цього зверху, на панелі швидкого доступу натискаємо кнопку «Инверсия»  . По-іншому, можна натиснути ПКМ на виділеному кінці лінії зв'язку і в контекстному меню вибрати пункт «Инверсия» (рис. 2.18).

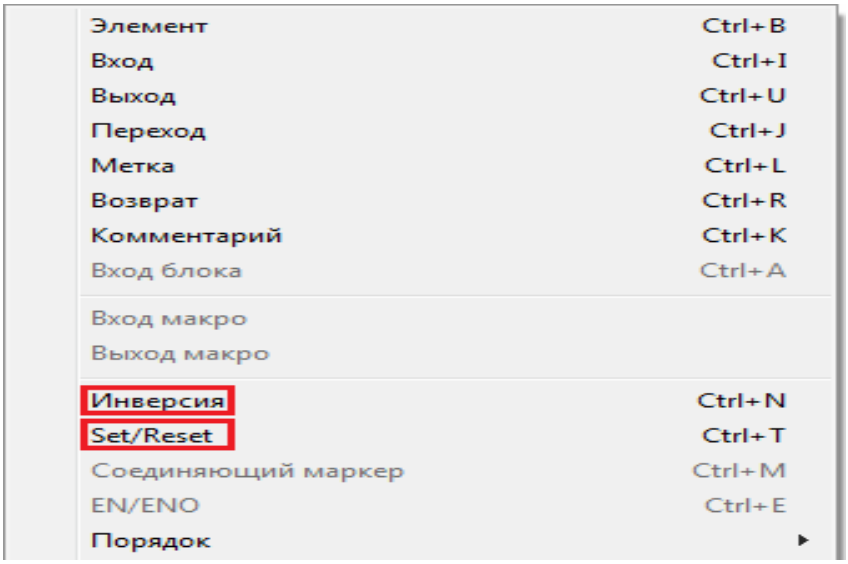


Рис. 2.18

У результаті поряд зі змінною ***out2*** з'явиться невелике коло (рис. 2.19), яке вказує на інвертування (логічне заперечення) значення, що приходить по лінії зв'язку від змінної ***in1***. *Я*кщо приходить значення TRUE, то в вихідну змінну записується FALSE і навпаки. Послідовне натискання на кнопку  дозволяє додавати і прибирати інверсію на лінії зв'язку.

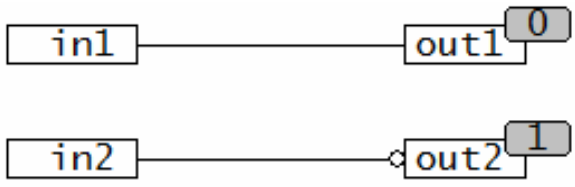


Рис. 2.19

Наприклад, необхідно додати ще два ланцюжки. Залежно від стану змінних ***in3*** і ***in4*** будемо включати або вимикати вихідну змінну ***out3***. Доповнюємо алгоритм викликом цих змінних так, як це зроблено на рис. 2.20.

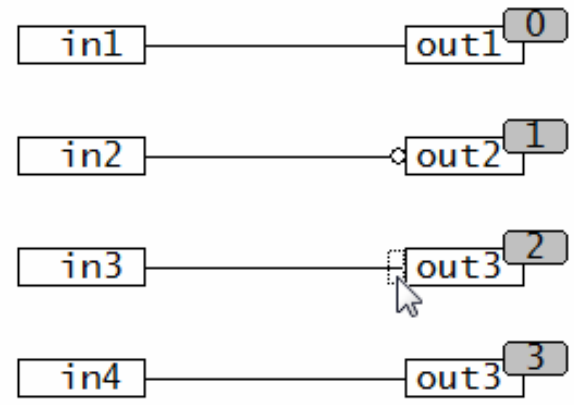


Рис. 2.20

Вельми часто в технологічному програмуванні, тобто при створенні алгоритмів роботи обладнання виникає наступна задача: необхідно розділити умови включення якогось виконавчого механізму і умови його відключення.

Наприклад, при управлінні вентилятором умовою його запуску може бути висока температура в приміщенні. А умовою зупинки того ж вентилятора буде закінчення відліку заданого часу.

Для виконання таких роздільних умов, у CoDeSys реалізовані операції установки і *скидання значення дискретної змінної.*

Входи і виходи ПЛК оголошені, як дискретні.

Таким чином, можна поставити включення виходу ***out3***, якщо є сигнал на вході ***in3***. Вимкнення ***out3*** можливо тоді, коли є сигнал на вході ***in4***.

Для реалізації цього алгоритму ми виділяємо лінію зв'язку поруч зі змінною ***out3*** (рис. 2.20).

Потім знаходимо на панелі швидкого доступу кнопку  і натискаємо її ЛКМ. Поруч з вихідною змінної з'являється символ  (рис. 2.21).

Тепер до змінної ***out3*** в цьому ланцюжку буде застосована операція установки. Якщо ще раз натиснути на кнопку  , то з'явиться символ, тобто буде налаштована операція скидання  . При наступному натисканні  система поверне ланцюжок в стан звичайного привласнення, і додаткові символи  і  будуть автоматично прибрані.

Замість використання кнопки на панелі швидкого доступу можна знайти пункт Set / Reset в контекстному меню (рис. 2.18). Це меню викликається натисканням ПКМ, зробити це необхідно на виділеній лінії зв'язку. У підсумку, повинно вийти чотири ланцюжки, зображені на рис. 2.21.

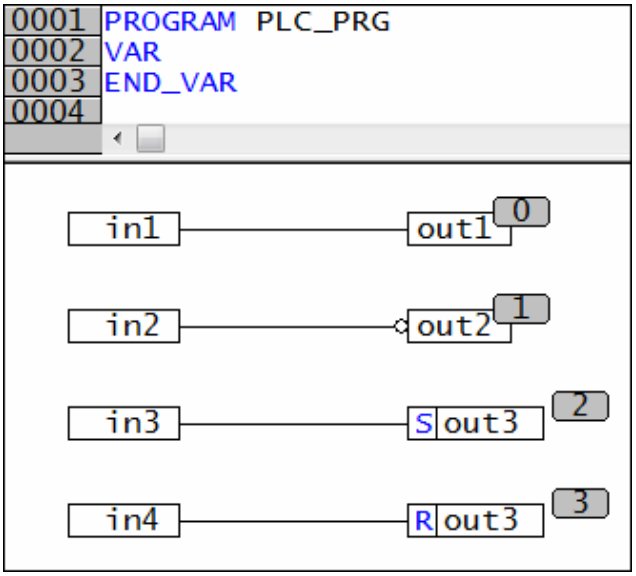


Рис. 2.21

Алгоритм готовий і його корисно зберегти. Далі необхідно розставити порядок обробки (Рис. 1.30) і зробити компіляцію проекту, наприклад, натиснувши кнопку F11, або звірившись з рис. 1.31.

Якщо є помилки, виправити їх і знову скомпілювати проект.

При роботі з даним алгоритмом випадково можна поставити зайві локальні змінні.

В області оголошення змінних, між ключовими словами VAR і END_VAR не повинно бути ніяких записів (Рис. 2.22). Якщо записи є, видаляють їх вручну.

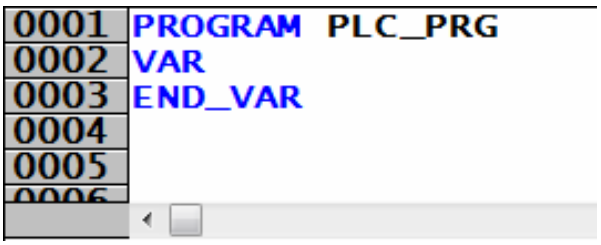


Рис. 2.22

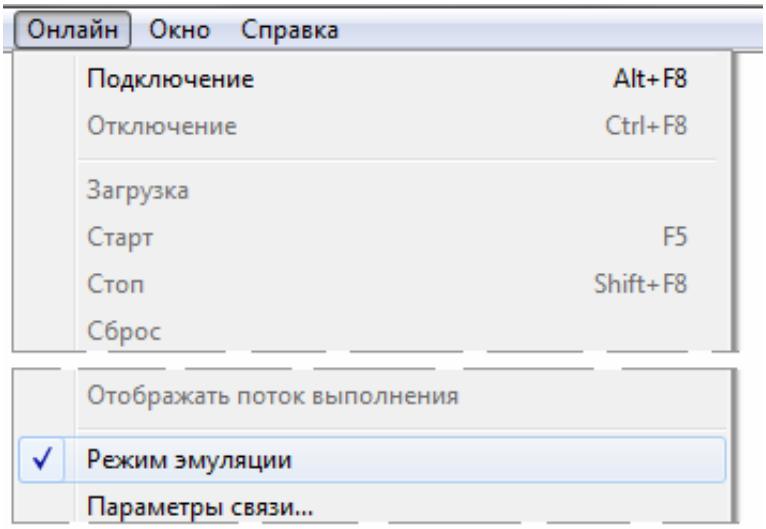


Рис. 2.23

Коли помилки виправлені, включають *режим емуляції*. Для цього необхідно зайти в

меню «Онлайн» і поставити галочку в пункті «Режим эмуляции» (рис. 2.23). Надалі ця галочку прибирається. Запускається проект на виконання («Онлайн»-«Подключение» або Alt + F8).

Для запуску виконання програми натискаємо кнопку F5.

Зовнішній вигляд вашої програми повинен бути схожим на те, що зображено на рис. 2.24.

В CoDeSys прийнято наступне колірне кодування: чорним кольором підсвічуються ті змінні, значення яких FALSE. Синій колір використовується тоді, коли в змінну записано значення TRUE. Тими ж кольорами відображаються лінії зв'язку при відсутності або проходженні через них сигналу. Наприклад, у результаті інверсії на виході ***out2*** з'являється значення «TRUE».

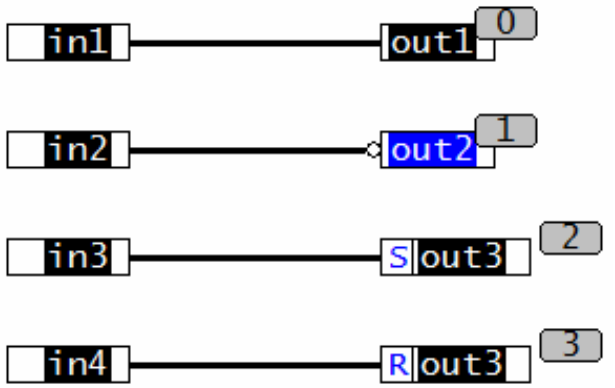


Рис. 2.24

Для того, щоб змінити значення вхідних змінних і вплинути таким чином на стан виходів використовується спосіб, вивчали раніше. Оскільки в емуляції працювати з фізичними кнопками і перемикачами не є можливим, то задаємо значення безпосередньо у вікні CoDeSys. Необхідно двічі

натиснути ЛКМ на потрібну змінну, щоб поміняти її значення (рис. 2.25) і продовжувати натискати до тих пір, поки не з’явиться потрібне значення.

Для запису значень у програму використовується комбінацію клавіш Ctrl + F7 (також рис. 1.44).

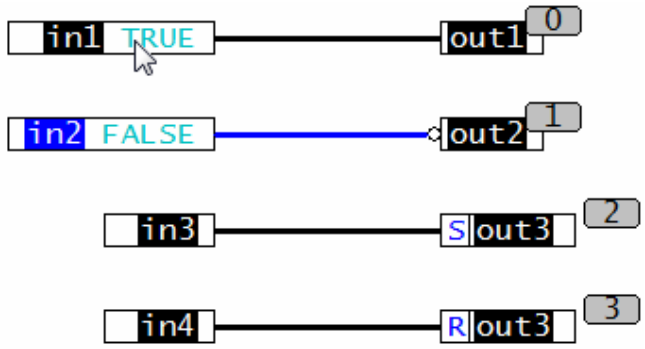


Рис. 2.25

Докладніше розглянемо роботу операцій установки і скидання на двох нижніх ланцюжка. Коли в змінних ***in3*** і ***in4*** значення FALSE, змінна ***out3*** залишатиметься в тому

стані, в якому вона перебувала до виконання цих операцій. Тобто вона зберігає своє значення (рис. 2.26 і рис. 2.27)

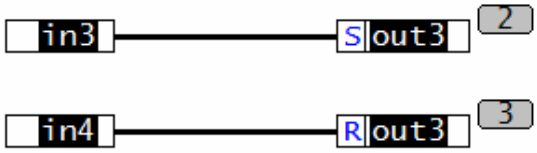


Рис. 2.26

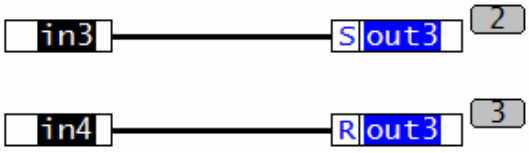


Рис. 2.27

Коли на вході ***in3*** з'являється значення TRUE, відбувається включення виходу ***out3*** (рис. 2.28).

У свою чергу коли на вході ***in4*** з’являється значення TRUE, відбувається відключення виходу ***out3*** (рис. 2.29).

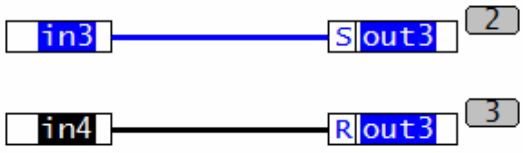


Рис. 2.28

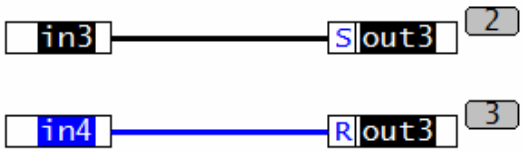


Рис. 2.29

Коли на обох входах є параметр «TRUE», необхідно оцінити черговість виконання операцій за таким принципом: остання операція, вироблена зі змінною, задає її стан в момент закінчення циклу ПЛК. Контролер (і емуляції теж) працюють циклічно. І створений алгоритм обробляється послідовно відповідно до номерів операцій. Порядок виконання задається перед запуском проекту на виконання.

У проекті черговість виконання ланцюжків йде зверху вниз. Таким чином, операція скидання у нас виробляється після операції установки. І значить, що остання операція – це скидання. Тому на рис. 2.30 змінна ***out3*** має значення FALSE і пофарбована в чорний колір. Коли цей алгоритм буде завантажений в ПЛК, контакти вихідного реле контролера, пов'язані зі змінною ***out3***, в подібній ситуації будуть розімкнуті.

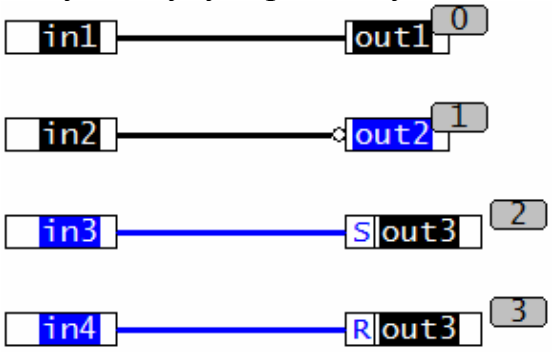


Рис. 2.30

Останні два ланцюжки алгоритму є реалізацією звичайного RS-тригера, тобто тригера з пріоритетом скидання. Якщо поміняти місцями порядок обробки останніх ланцюжків (рис. 2.31) і поставити скидання перед установкою, то отримаємо інше значення на виході. Тут ми реалізували SR-тригер, який має пріоритет по сигналу установки. При наявності значення «TRUE» одночасно на входах ***in3*** і ***in4*** вихідна змінна залишиться включеною.

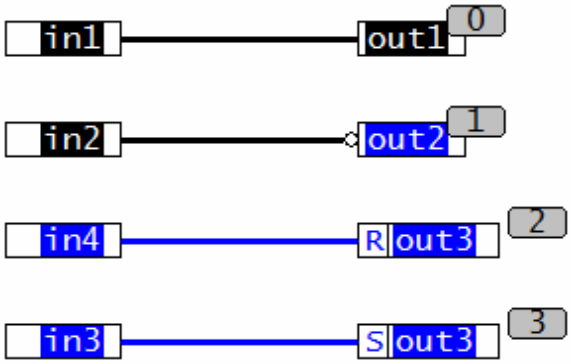


Рис. 2.31

2.5 Настройка зв'язку з ПЛК і завантаження проекту в контролер

Завантажимо алгоритм безпосередньо в ПЛК і подивимося, як він буде працювати безпосередньо на цільовій платформі. Для початку необхідно вийти з режиму виконання в рамках емуляції роботи проекту (Ctrl + F8 або «Онлайн» - «Отключение»). Потім потрібно зняти галочку навпроти пункту «Режим эмуляции» в меню «Онлайн» (Рис. 2.23).

Тепер CoDeSys при наступних запусках проекту на виконання буде прагнути підключитися до ПЛК. Для успішного підключення необхідно задати канал зв'язку системи програмування і контролера. У комплекті поставки ПЛК є кабель для програмування через порт RS-232, тобто через звичайний COM-порт комп'ютера. Про налаштування з'єднання з іншим інтерфейсом зазначається в керівництві по експлуатації, яке додається до комплекту поставки ОВЕН ПЛК.

Для початку необхідно підключити фізично контролер до комп'ютера. Потрібний порт, позначений як Debug RS-232, знаходиться на верхній кришці приладу (рис. 2.32). Підключення кабелю до COM-порту комп'ютера проводиться тільки при відключеному живлення ПЛК!

Рис. 2.32

Для настройки зв'язку в CoDeSys знаходимо в меню «Онлайн» пункт «Параметры связи» і заходимо в нього (рic.2.33). У вікні «Communication Parameters» (рис.2.34), в лівій частині розташовується список настроєних для даного проекту каналів зв'язку.

У центральному вікні видно настройки вибраного каналу. Якщо немає потрібних настройок, то створюється новий канал. Для цього в правій частині вікна «Communication Parameters» натискаємо кнопку «New». Відкривається додаткове вікно (Рис. 2.35).

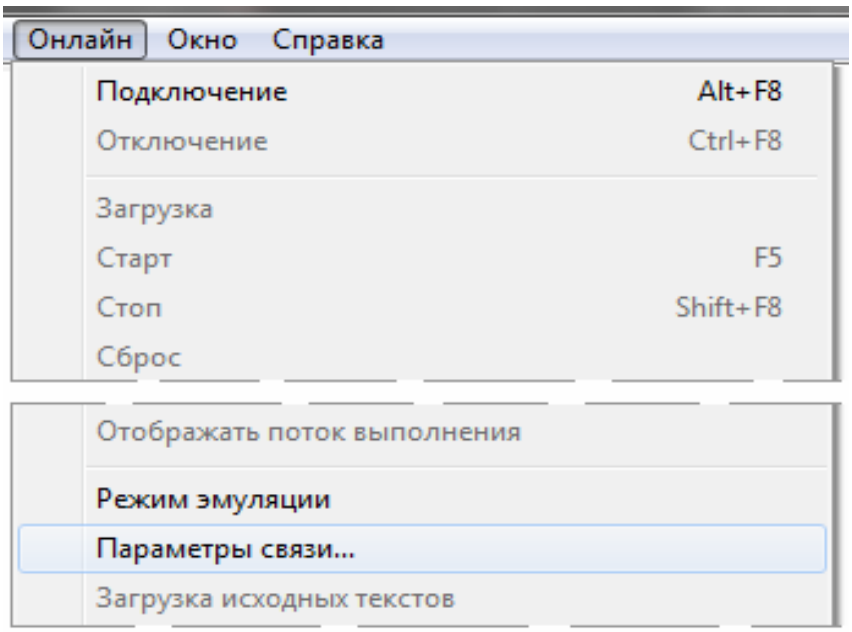


Рис. 2.33

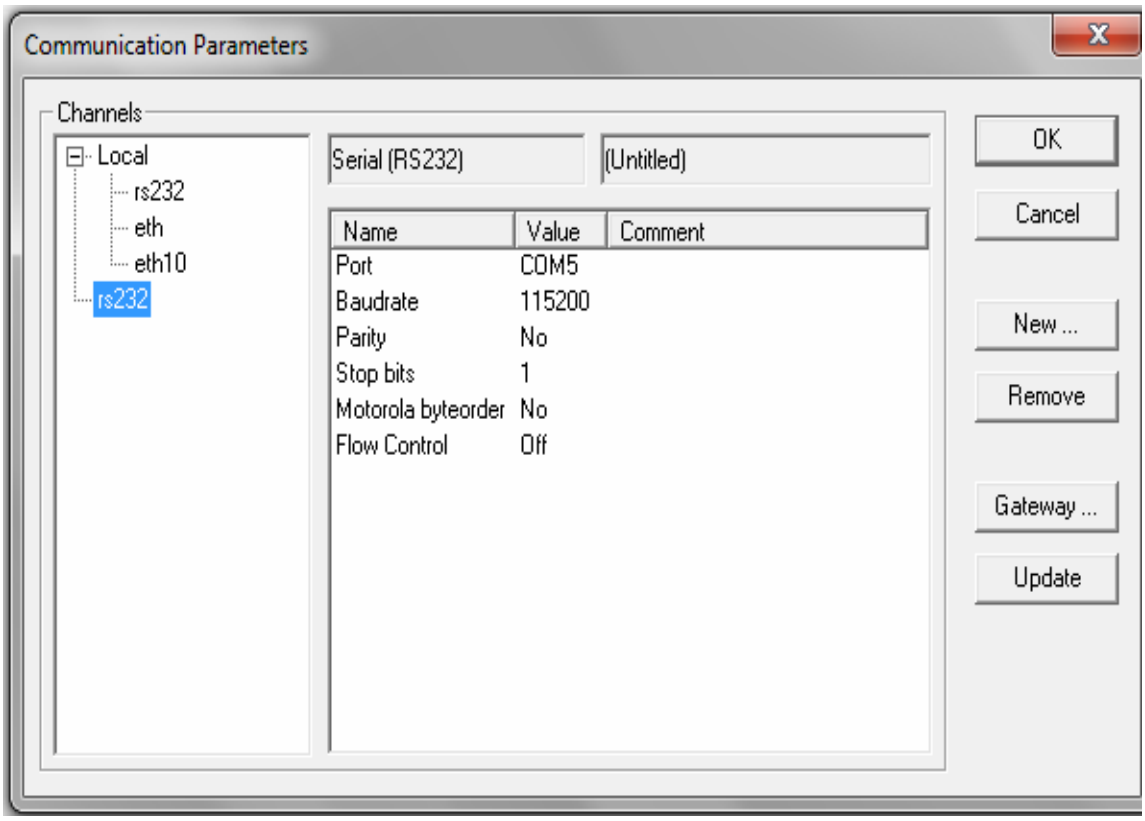


Рис.2.34

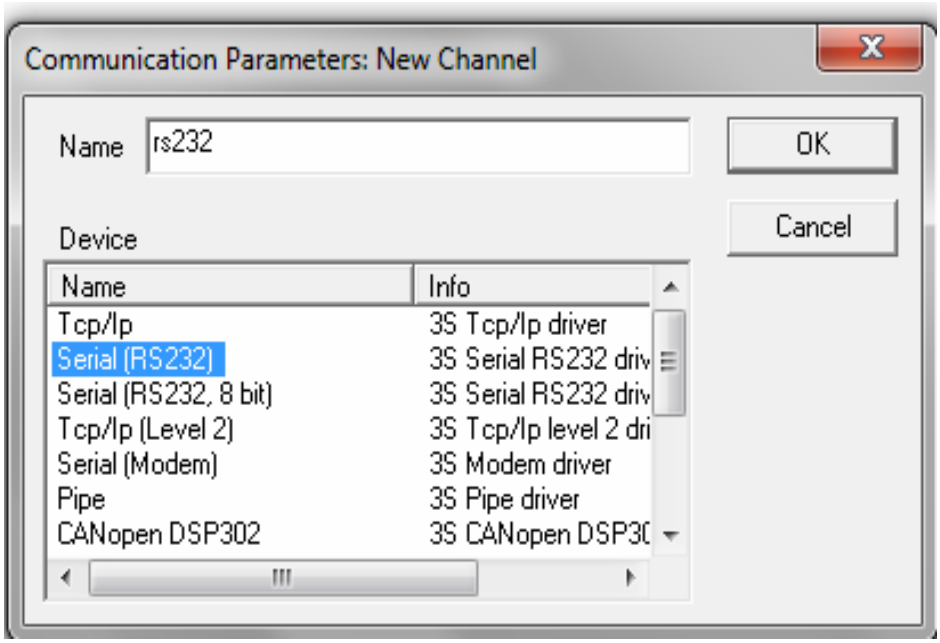


Рис.2.35

У цьому вікні, зверху в поле «Name» прописується довільна назва нового каналу, наприклад, rs232. Тепер *важливо зі списку внизу вибрати правильний драйвер для роботи*.

Потрібний пункт називається Serial (RS232), він другий зверху. Вибираємо його за допомогою ЛКМ, як показано на рис.2.35. Потім натискаємо кнопку «ОК» і повертаємося до роботи з вікном «Communication Parameters».

В лівій частині вікна «Communication Parameters» з'явився новостворений канал rs232.

Настройки його властивостей здійснюються в центральній частині вікна. Необхідно змінити значення для першого рядка, для цього вказується номер того COM-порту, до якого підключили ПЛК, використовуючи кабель з комплекту поставки. Наприклад, використовується COM5. Двічі натискається ЛКМ на поточній настройці COM1, потім за допомогою кнопок «вгору» і «вниз» на клавіатурі комп'ютера вибирається COM5.

Якщо у буде інший номер порту, що використовується, виставляються необхідні настройки. Можна вибрати порт, послідовно натискаючи ЛКМ на відповідному полі.

У другому рядку «Baudrate» необхідно вибрати швидкість з'єднання. *Незалежно від моделі контролера ми завжди вибираємо швидкість 115200 б/с*. Також, як і в верхньому рядку, два рази натискається на поточній швидкості 9600 і за допомогою стрілок на вашій клавіатурі доводите швидкість до потрібного значення і натискаєте Enter. Усі інші настройки залишається без змін і натискаємо «ОК» для підтвердження.

Тепер запустимо проект на виконання (Alt + F8). Якщо все було зроблено правильно, то зв'язок буде встановлено. В іншому випадку через кілька секунд CoDeSys видасть повідомлення про помилку (рис. 2.36).

Необхідно перевірити правильність настройок відповідно до рекомендацій попереднього параграфа або керівництва по експлуатації. Не забудьте включити живлення ПЛК і почекати 10-15 секунд, поки він перезавантажиться.

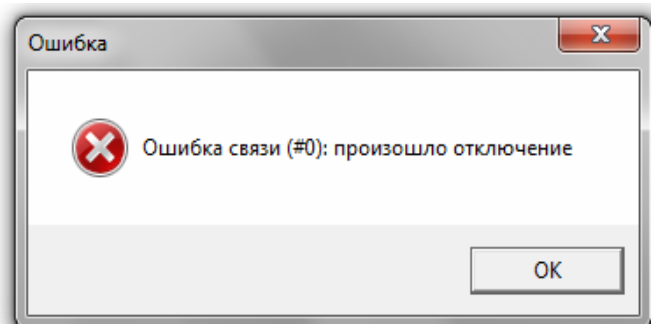


Рис.2.36

Якщо зроблено все правильно, система видасть запит на завантаження нової програми. Якщо в контролері на поточний момент програм немає, то з'явився повідомлення з рис. 2.37 або з рис. 2.38. У будь-якому випадку, на будь-яку з цих пропозицій відповідають згодою, тобто натисканням кнопки «Да».

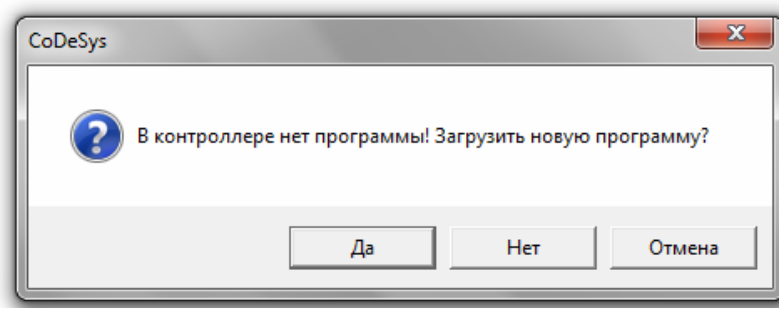


Рис.2.37

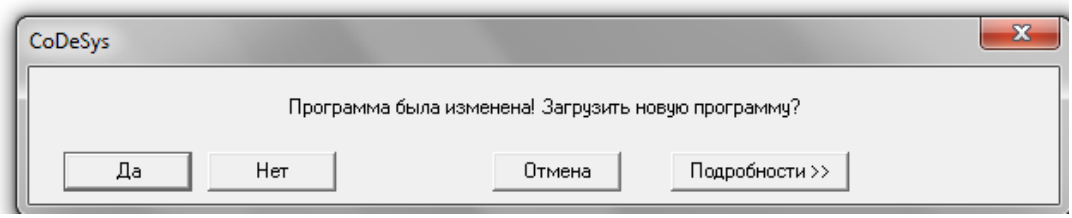


Рис.2.38

Проект завантажується в контролер. Після закінчення завантаження може з'явитися системне повідомлення. Якщо це станеться, в вікні, ви просто натискаєте «ОК».

Кнопка F5, яка запускає програму в роботу.

Зовнішній вид програми не відрізняється від того, що й в режимі емуляції.

А завдання значень для змінних тепер повинні вироблятися безпосередньо з фізичних входів ПЛК. Для цього заздалегідь підключити до нього кнопки або перемикачі, що імітують сигнали. Логіка роботи алгоритму залишилася та ж, що докладно розглядалася раніше.

Відстежувати стан фізичних входів і виходів ПЛК в процесі виконання алгоритму можна як з вікна редагування програми (рис 2.24), так і з вікна конфігурації ПЛК.

Зовнішній вигляд конфігурації в режимі виконання представлений на рис. 2.39.

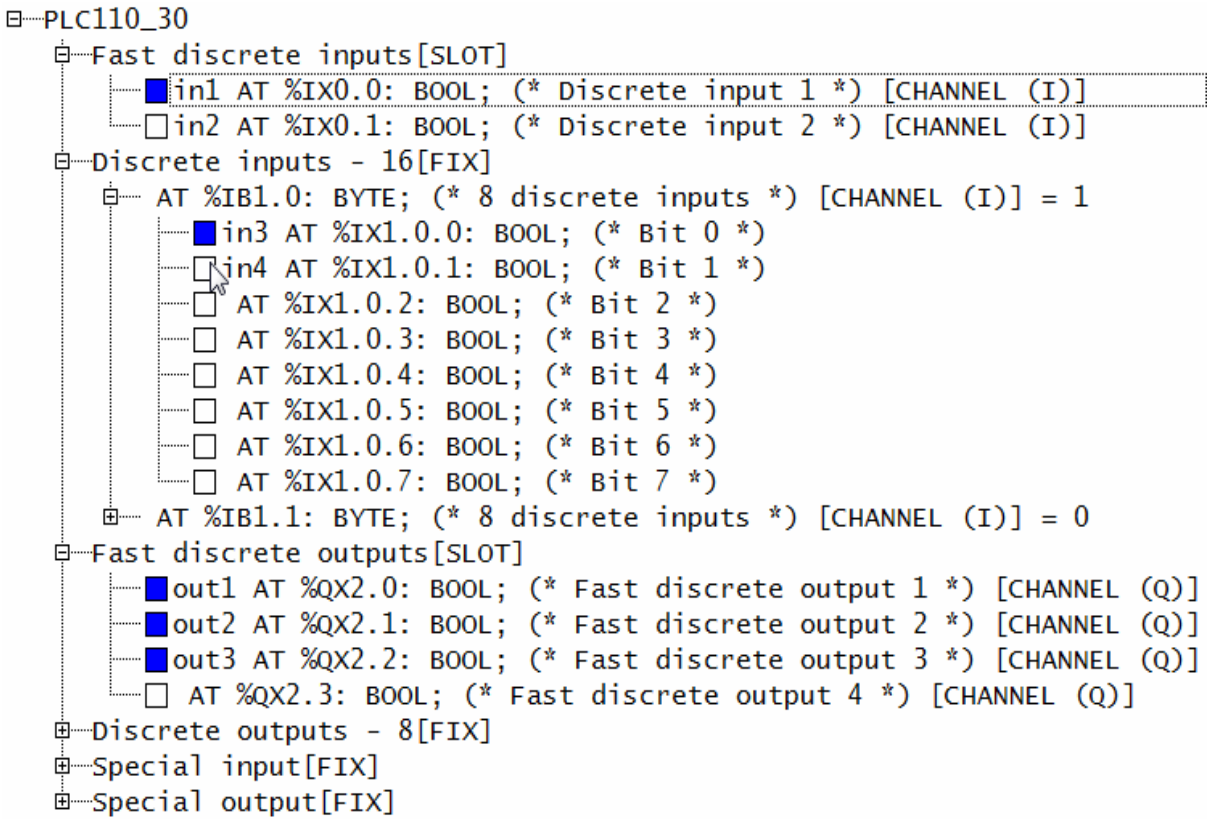


Рис. 2.39

3. Логіка

У третій частині мова йтиме про логічних операціях, тобто про операції «И», «ИЛИ», «НЕТ»

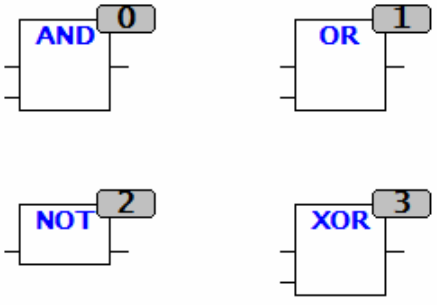


Рис. 3.1

Коли завантажується система програмування, то створюється новий проект через меню «Файл». Там необхідно вибирати пункт «Создать». У вікні «Настройка целевой программы» вибираємо той тип ПЛК, з яким

будемо працювати (Рис. 2.5 і рис. 2.6). Вибираю контролер ОВЕН ПЛК**1xx**, наприклад ПЛК110 або 160. (див. рис. 2.5 і рис.2.6). Працюємо з мовою функціональних блоків CFC. Саме цей мова вибираємо при створенні головної програми PLC_ PRG (рис. 2.7).

Переходимо на вкладку ресурси, вибираємо пункт «ККонфигурация ПЛК». Раніше зазначили, що робота ПЛК відбувається циклічно. У програмі описуються дії для одного циклу, одного повторення. Далі, після завантаження проекту в ПЛК, контролер постійно повторює підготовлений алгоритм. Саме в цьому різниця між програмуванням додатків для персонального комп'ютера, і роботою з завданнями АСУ ТП.

Задаємо контролеру час циклу його роботи (час циклу ПЛК), тобто визначаємо, з якою частотою необхідно повторювати обробку програми. Для цього у вікні «Конфигурация ПЛК» виділяємо верхній пункт, наприклад «PLC 150 I». Потім переходимо на вкладку «Параметры модуля», розташовану в правій верхній частині екрану (рис. 3.2).

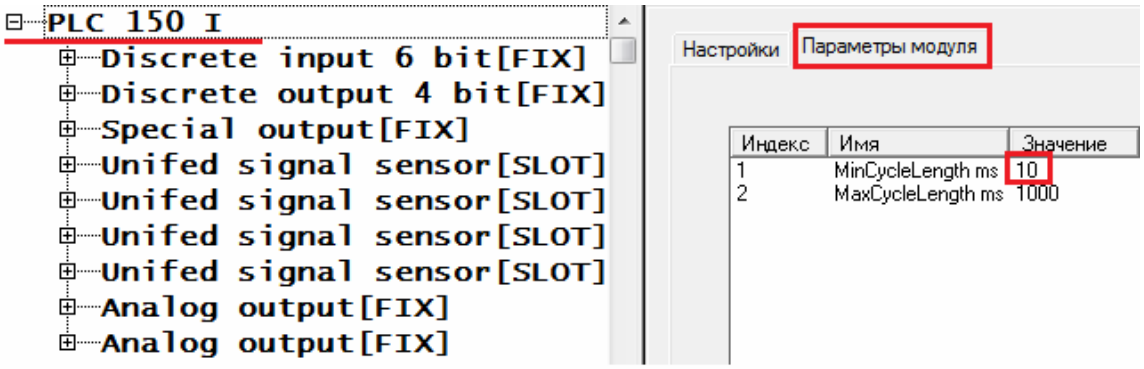


Рис. 3.2

За замовчуванням, час циклу ПЛК1xx становить 1 мс. Тобто кожну мілісекунду ПЛК заново запускає обробку нашого алгоритму, яку повинен встигнути завершити. Для простих програм така швидкість обробки цілком допустима. У більш серйозних завданнях корисно задати певний запас продуктивності ПЛК. Тобто вказати час циклу, скажімо, 10 мс. Для 80% завдань, що вирішуються ПЛК1xx, цього буде достатньо. Як правило відразу після створення проекту задається час циклу 10 мс. Для цього в першому рядку, в поле «Значение» замість 1 мс поставте 10 мс (рис. 3.2) – це час циклу, за який контролер буде обробляти нашу програму.

Щоб розібратися з роботою дискретної логіки, нам знадобиться кілька входів і виходів ПЛК: розкриваються елементи конфігурації ПЛК і задаються шість дискретних входів. Нехай це будуть змінні ***x1***, ***x2***, ***x3*** і т.д.

Щоб визначити ім'я змінної для того чи іншого входу або виходу два рази клацаєте ЛКМ на написи «AT» в потрібному рядку, вводиться ім'я змінної (англійськими літерами та цифрами без пробілів), натискається «Enter» на клавіатурі. Таким чином, вийшло 6 дискретних входів. За аналогією задаємо чотири дискретних виходи – це будуть змінні ***y1***, ***y2***, ***y3*** і *y4* (рис. 3.3).

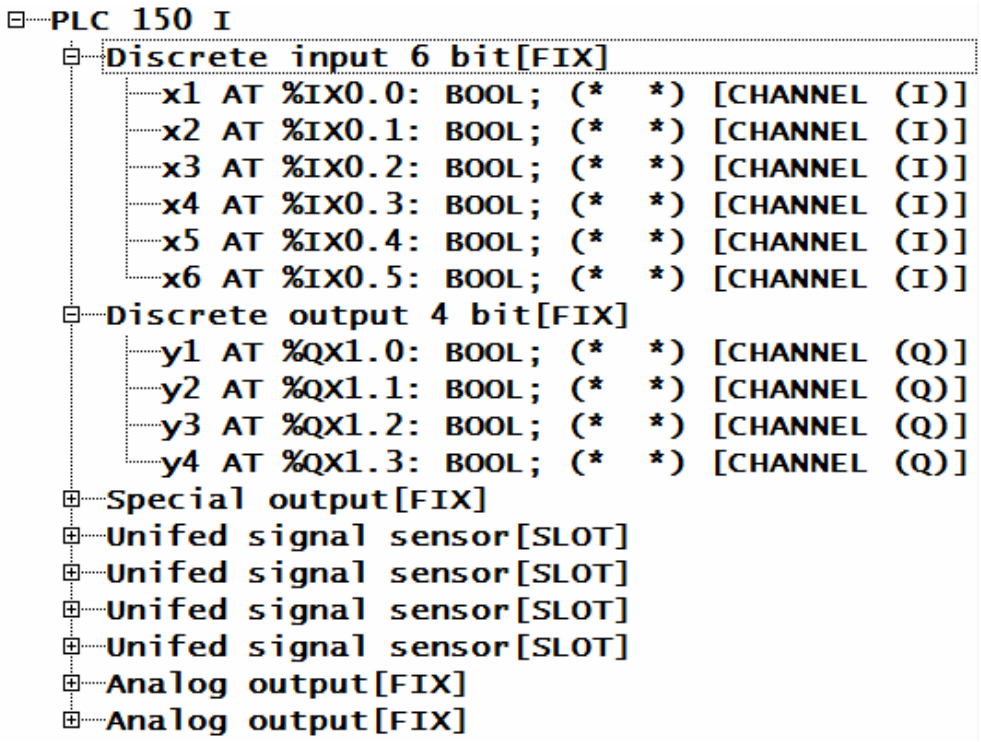


Рис. 3.3

Отже необхідно:

1. *визначитися з target-файлом;*
2. *створити головну програму і вибрати мову програмування;*
3. *задати час роботи циклу ПЛК;*
4. *визначитися з необхідними входами і виходами;*
5. ***зберегти** проект на комп'ютері.*

Повернемося до головної програми, яка знаходиться на вкладці «POU». Переходимо на неї і два рази клацаємо ЛКМ на програму «PLC_PRG» (рис. 3.4).

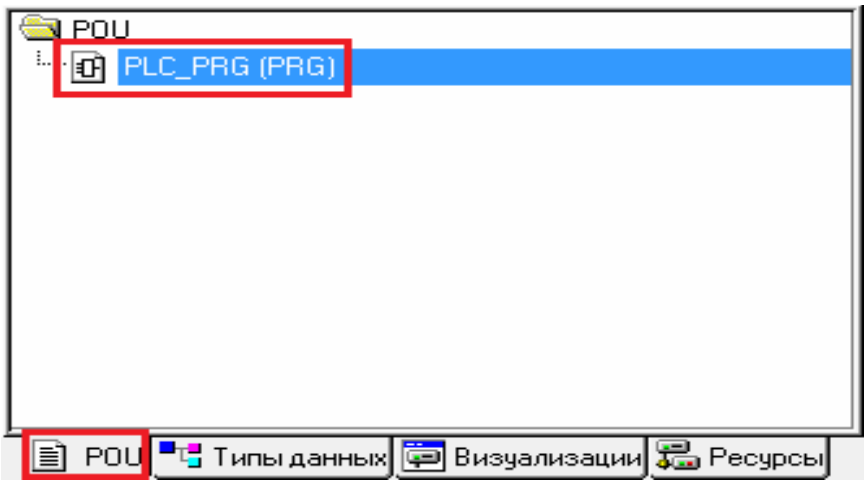


Рис. 3.4

Для того щоб задати в алгоритмі ту чи іншу логічну операцію, чинять наступним чином. У верхній частині екрану, на панелі швидкого доступу є кнопка «Элемент» () (див. рис. 1.22 і рис. 3.5). Натискають на неї, переносять курсор на робочу область і натискають лівою кнопкою миші ще раз, встановлюючи елемент в потрібному місці робочої області. За замовчуванням блок, який додали, буде виконувати операцію логічного «І». Вона задається ключовим словом «AND» (рис. 3.6). Аналогічний блок можна викликати через контекстне меню, натиснувши в робочій області ПКМ і вибравши пункт «Элемент».

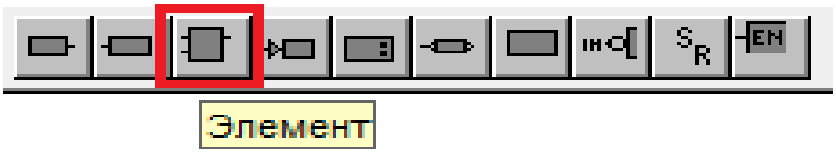


Рис. 3.5

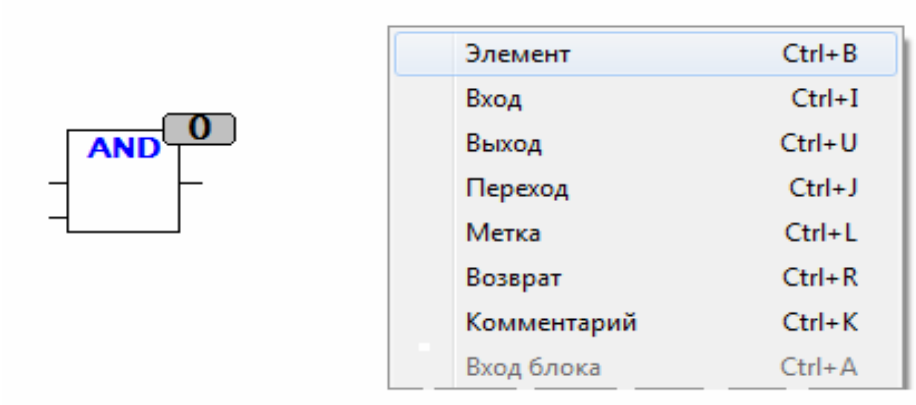


Рис. 3.6

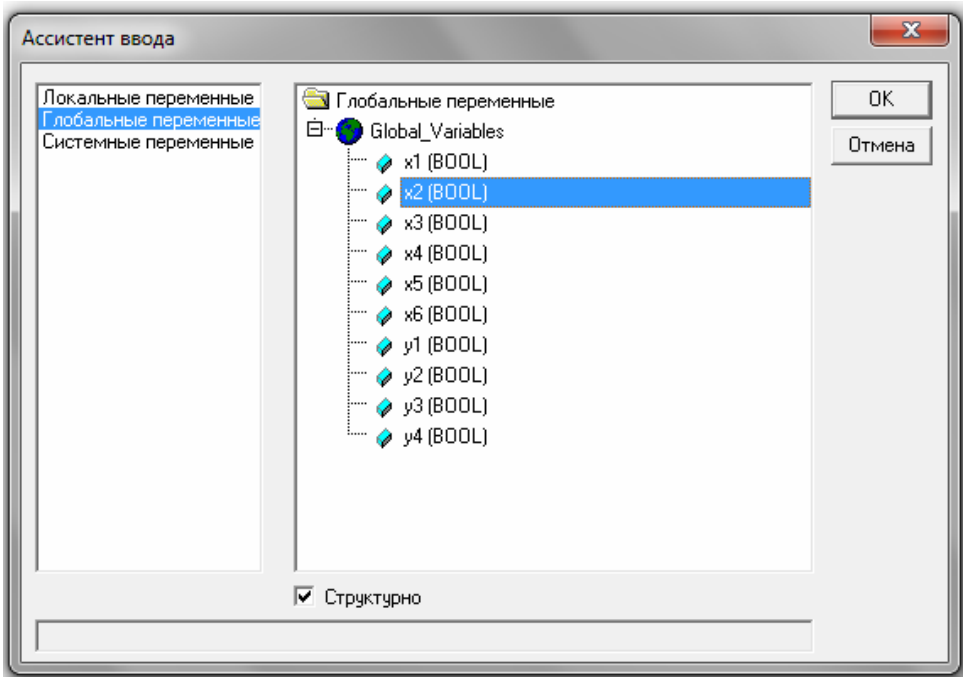


Рис. 3.7

На вхід операції «AND» подаються змінні типу «BOOL»: такими змінними є дискретні входи. У нашій програмі викликають значення змінної, використовуючи кнопку «Вход» .

Можна це зробити через кнопку «F2» на клавіатурі. У вікні вибирають список глобальних змінних, і в ньому знаходять потрібні нам змінні ***x1*** і ***x2*** (рис.3.7).

Далі проводять лінії зв'язку.

На виході операції логічного «И» (блок AND) буде з'являтися сигнал TRUE або FALSE, тобто значення типу BOOL. Це значення можна присвоїти для змінної того ж типу, наприклад, виходу ***y1***.

Для цього додають вихід на робочу область. Замість знаків питання прописують ім'я для змінної ***y1***. Потім проводять лінію зв'язку між виходом операції AND і змінної ***y1***. Результат на рис. 3.8.

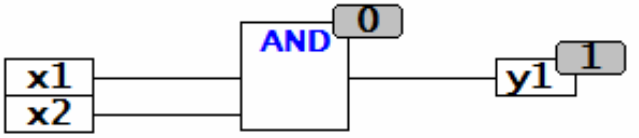


Рис. 3.8

Під час першої роботи з CoDeSys виникають деякі складності із зміною положення тих чи інших елементів програми в робочій області. Якщо необхідно пересувати один або кілька елементів програми в інше місце, виділяють потрібну частину алгоритму, утримуючи ЛКМ (Рис.3.9).

Усі елементи повинні цілком потрапити у виділену область. У результаті, після відпускання ЛКМ один або кілька елементів виявляться виділеними пунктиром (рис.3.10)

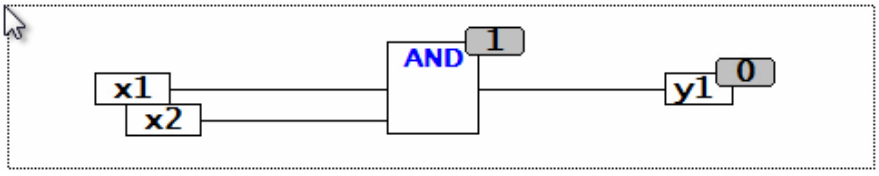


Рис. 3.9

Натискають ЛКМ на виділеній області так, як це зроблено на рис.3.10, наприклад, всередині блоку. І, утримуючи ЛКМ, перетаскують відразу всі виділені об'єкти в потрібну частину робочої області.

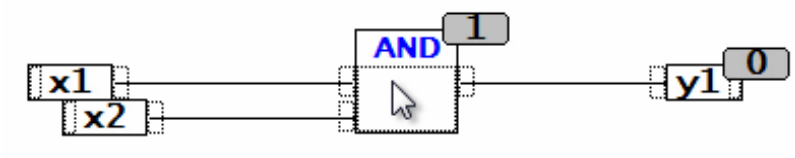


Рис. 3.10

Якщо необхідно подібним чином поміняти положення тільки однієї змінної, тоді треба врахувати наступне (рис.3.11): входи перетягуються за ліву частину елемента, як це показано на прикладі ***x3***. Виходи переносять, «чіпляючись» ЛКМ за праву частину елемента, наприклад, як показано на рис.3.11 для змінної ***y3***.

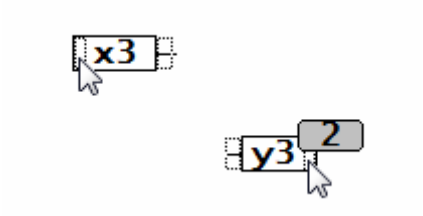


Рис. 3.11

Якщо необхідно попрацювати з операцією «ИЛИ» замість «И», то зробити це можна в такий спосіб. Додається новий елемент, виділяється заголовок ЛКМ і друкується нова назва «OR», потім натискається «Enter» (рис. 3.12).

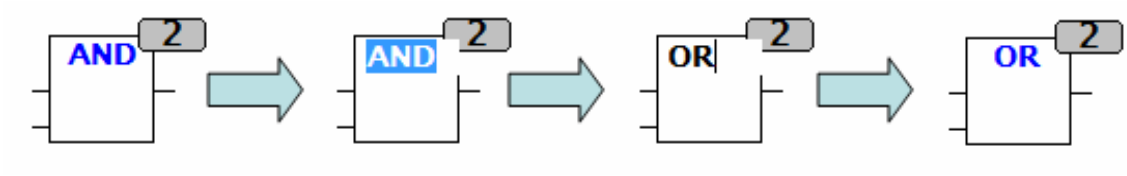


Рис. 3.12

Отримали блок «ИЛИ». Викликають в алгоритмі значення дискретних входів ПЛК, на цей раз ***x3*** і ***x4***. Потім підводять лінії зв'язку до блоку OR. Результат роботи цього блоку передають на другий дискретний вихід контролера, тобто в змінну ***y2***. Результат зображено на рис.3.13.

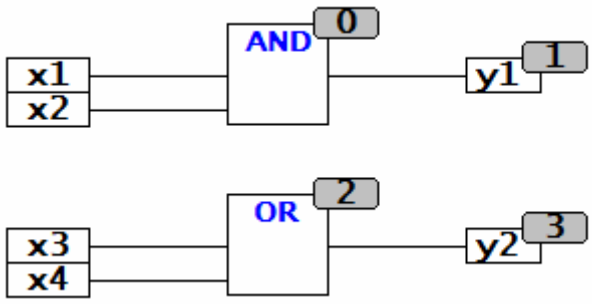


Рис. 3.13

Тепер з операцією «логическое НЕ» – блок NOT. Розберемо інший спосіб, як можна задати операцію «ллогическое НЕ». Натискають кнопку «Элемент» або через ПКМ викликають контекстне меню і вибираєте пункт «Элемент».

Таким чином створюється новий блок і замість напису AND в ньому задають ключове слово NOT. На вихід операції NOT підключають змінну ***y3*** (рис. 3.14).

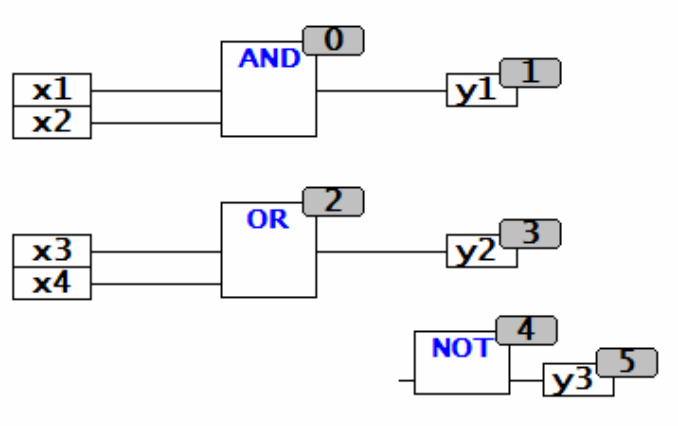


Рис. 3.14

З виходу одного і того ж блоку або з однієї і тієї ж змінної лінію зв'язку можна вести на один або кілька можливих приймачів цього самого сигналу.

Пояснення: з операції OR можна протягнути другу лінію зв'язку на вхід операції NOT. Для цього натискають ЛКМ на вихід блоку OR, тобто на самий кінець виходить з нього лінії зв'язку (рис. 3.15).

Утримуючи ЛКМ, перетягуємо другу лінію зв'язку на вхід блоку NOT (рис. 3.15), потім відпускаємо ЛКМ. Отриманий результат зображений на рис. 3.17. Таким же чином можна додати і третю, і четверту, і ще більшу кількість зв'язків.

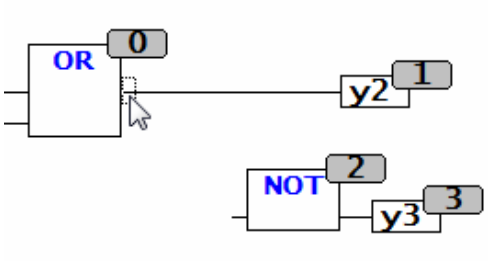


Рис. 3.15

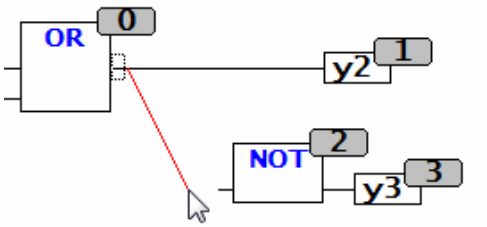


Рис. 3.16

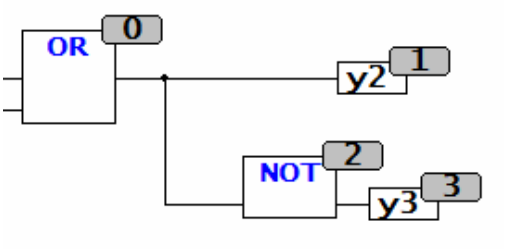


Рис. 3.17

Тим же способом можна значення однієї і тієї ж змінної передавати на кілька блоків. Приклади наведено на рис.3.18, рис.3.19 і рис.3.20. Щоб видалити непотрібну лінію зв'язку, виділяють один з її кінців (рис.3.18). Потім натискають на клавіатурі кнопку Del (Delete).

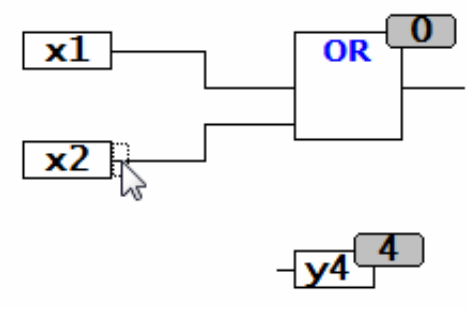


Рис. 3.18

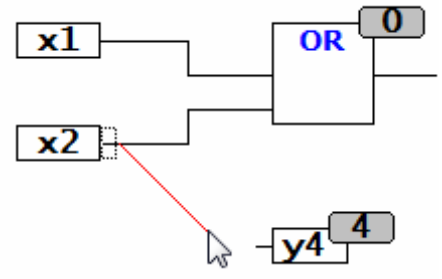


Рис. 3.19

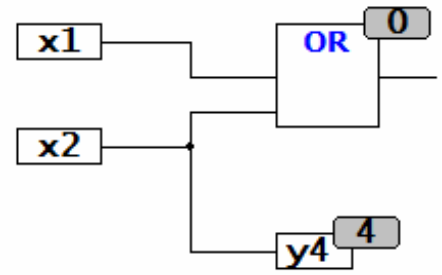


Рис. 3.20

Повернемося до алгоритму (рис.3.21). Коли на виході операції OR буде значення FALSE, на виході ***y3*** буде з'являтися сигнал TRUE, в той час як у змінної ***y2*** він буде пропадати.

І навпаки, коли на виході OR з'являється сигнал TRUE, він записується в ***y2***. У блоці NOT цей сигнал буде перетворюватися в FALSE і надходити в змінну ***y3***. А в результаті релейні виходи ПЛК весь час будуть в різних станах.

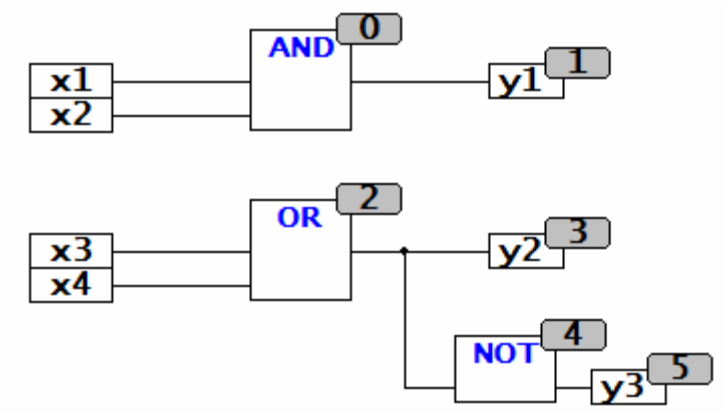


Рис. 3.21

Логічний оператор «Исключающие ИЛИ» часто використовуваний, наприклад, для реалізації прохідних вимикачів. «Исключающие ИЛИ» задаються за допомогою ключового слова XOR.

На вхід цієї операції подають значення змінних типу BOOL, наприклад, змінні ***x5*** і ***x6***. Для того щоб використовувати повторно якісь готові частини вже написаного алгоритму, можна скористатися стандартними операціями «Копировать» та «Вставить».

При додаванні блоку XOR використовують ці операції. Для початку, виділять ланцюг з оператором OR, як показано на рис.3.22

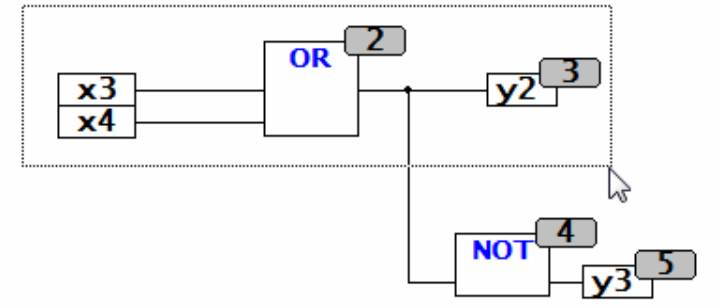


Рис. 3.22

Потім зверху, на панелі швидкого доступу натискають кнопку «Копировать»  (рис.3.23) або поєднання клавіш Ctrl + C.

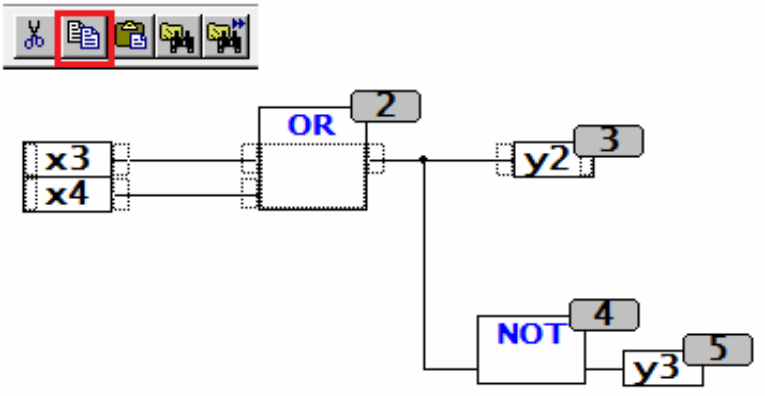


Рис. 3.23

Потім вставляють зроблену копію, для цього натискають кнопку «Вставить» (рис.3.24). На екрані з’явиться копія обраної частини алгоритму.

Натиснувши і утримуючи ЛКМ на виділених частинах копії, що з’явилася, перетягують її на нове місце в робочій області (рис.3.25)

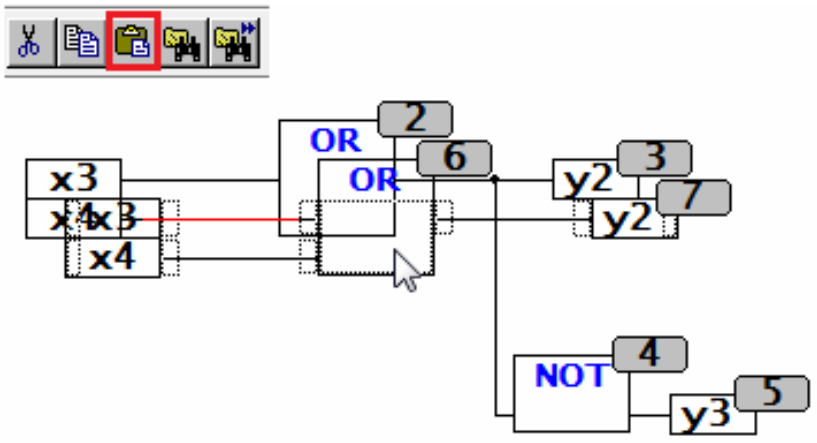


Рис. 3.24

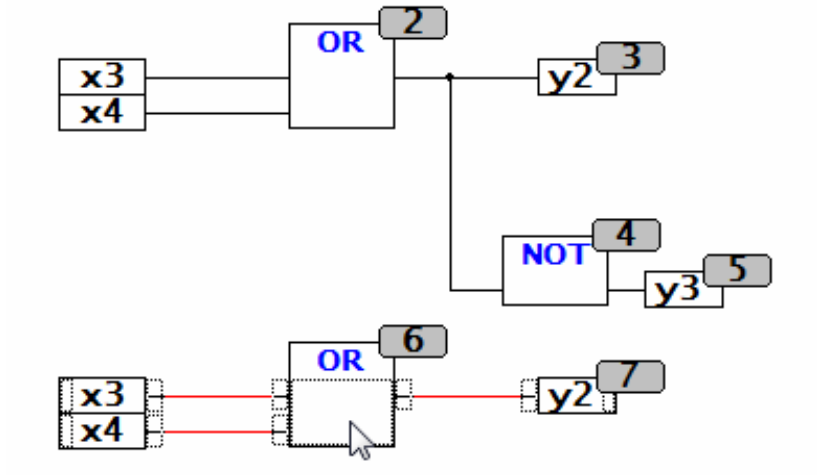


Рис. 3.25

Для заміни в заголовку елемента оператора OR на оператор XOR у якості вхідних даних використовують змінні ***x5*** і ***x6***.

Для заміни імені змінної натискають на відповідний блок входу ЛКМ. Потім з клавіатури вводите нове ім’я.

Результат роботи блоку XOR передають в змінну ***y4***. Кінцевий вид алгоритму представлений на рис.3.2

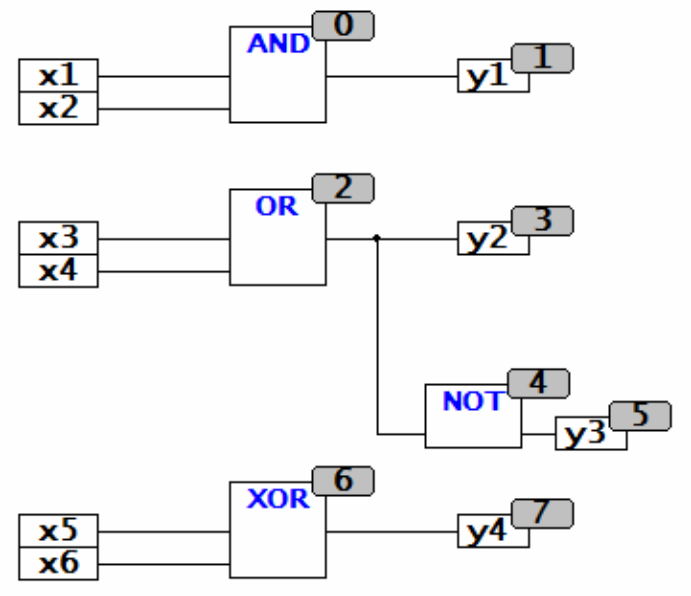


Рис.3.26

Суть операції XOR в наступному: коли значення змінних ***x5*** і ***x6*** будуть неоднаковими, наприклад, один – «FALSE», а інший – «TRUE», в змінної ***y4*** буде сигнал «TRUE».

Тобто вихід ***y4*** буде включений. Коли ***x5*** одно ***x6*** (або обидві змінні в стані «FALSE», або обидві в стані «TRUE»), тоді на виході «Исключающего ИЛИ» буде значення «FALSE» перевіримо, запустивши програму безпосередньо на ПЛК.

Перед завантаженням проекту на виконання необхідно:

1. Розставити порядок обробки: натиснути ПКМ на вільну область, вибирати «Порядок», а потім «В соответствии с потоком данных». Нумерація елементів, тобто порядок їх обробки, стане більш правильною. (Див. в першій частині).

2. Зберігти проект (Ctrl + S).

3. Запустити проект на перевірку, тобто компіляцію. Для цього в меню «Проект» вибирають пункт «Компилировать все», або натискаємо кнопку F11. Якщо внизу, у вікні повідомлень помилок «0», то цей простий алгоритм запускають на виконання. Якщо помилки є, їх виправляють і знову запускають компіляцію.

Находять в меню «Онлайн», вибирають пункт «Подключение», зв'язуються з ПЛК, завантажують нову програму. Якщо внизу загоряється напис «Онлайн», це означає, що є зв'язок з контролером.

На самому ПЛК в цей момент повинен спалахнути на світлодіод «СВЯЗЬ». Щоб програма почала працювати, необхідно зняти її з паузи. Для цього в меню «Онлайн» вибирають пункт «Старт», або натискають кнопку F5. Внизу з'явиться активний напис «Запущено» (рис.3.27).

Рис. 3.27

Починає діяти блок NOT. На вхід у нього приходить значення «FALSE», на виході ми бачимо сигнал «TRUE», який передається на відповідний дискретний вихід ***y3*** (рис. 3.28).

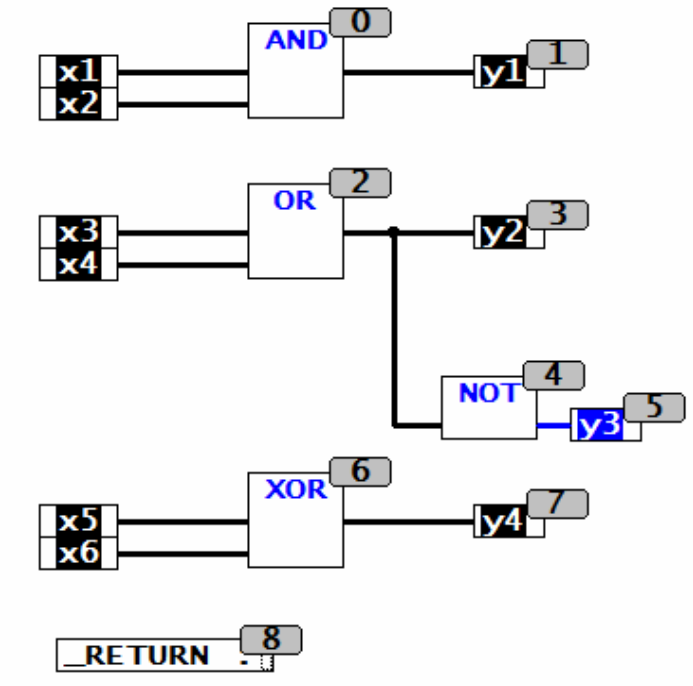


Рис. 3.28

Щоб перевірити програму подають сигнал на змінну ***x1***, на змінну ***x2***, з'являється «TRUE» в змінної в ***y1*** (риіс.3.31).

Тобто спрацьовує перший вихід. Якщо один з вхідних сигналів відключити, то в ***y1*** буде «FALSE» (FALSE) (рис.3.29).

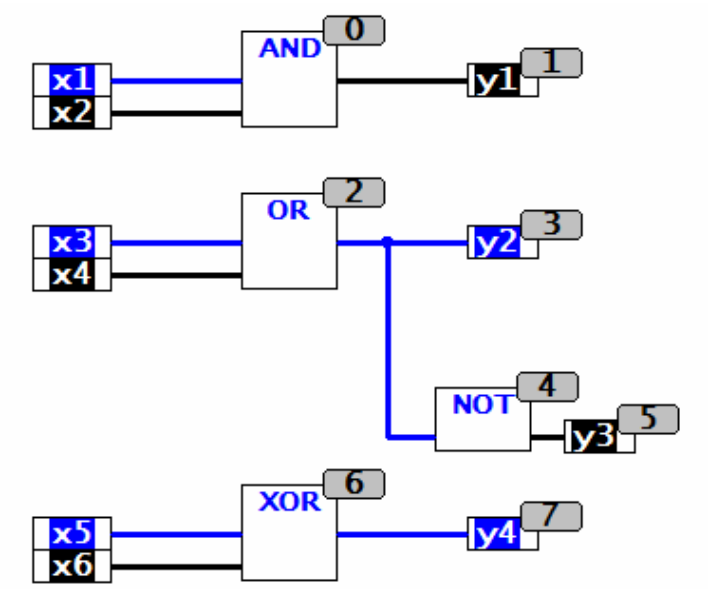


Рис. 3.29

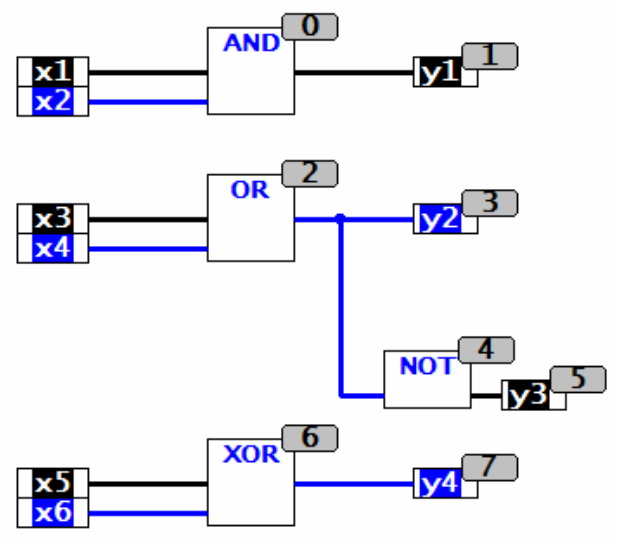


Рис. 3.30

Коли з'являється сигнал на третьому або четвертому вході, загоряється вихід ***y2***, а вихід ***y3*** гасне, оскільки там стоїть інверсія (рис.3.30).

Якщо з'являється сигнал на четвертому вході, ситуація не змінюється. Якщо пропадає сигнал на третьому вході,ситуація знову не змінюється.

Коли обидва входи, ***x3*** і ***x4*** у вимкненому стані, тоді сигнал на виході операції «ИЛИ» пропадає (рис.3.28).

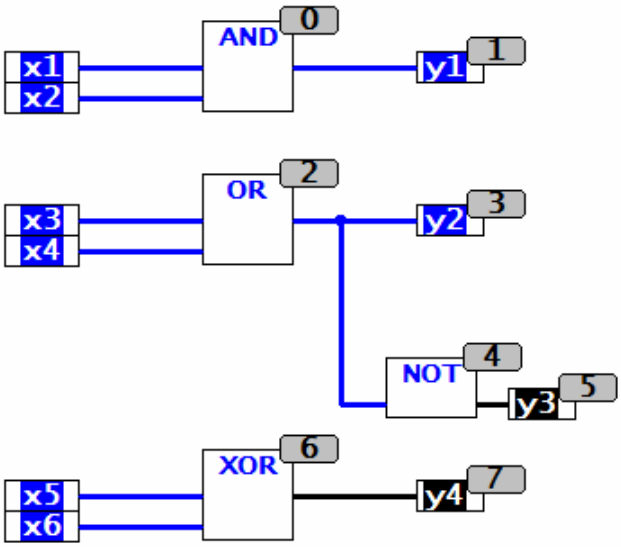


Рис. 3.31

Перевірте роботу блоку XOR відповідно до сигналів, представленими на рис.3.28-3.31.

Через меню «Онлайн» вибирають «Отключение», тобто розривають зв'язок CoDeSys і контролера. Сама програма в ньому залишається.

Якщо на ПЛК працюють зі входами, то виходи програми будуть перемикатися.

Часто бувають ситуації, коли для операції AND або для операції OR буде необхідно більше, ніж два вхідних значення. Задати додаткові входи можна наступним чином: виділяють ЛКМ блок, в якому необхідно додати вхід. Потім натискають на ньому ПКМ, вибирають пункт «Вход блока». З'являється ще один вхід (рис.3.32). Такого ефекту можна домогтися, виділивши блок і натиснувши поєднання клавіш Ctrl + A. Якщо якийсь вхід не потрібен, то виділяють його ЛКМ і натискають Delete на клавіатурі (рис.3.33).

Час від часу корисно зберігати свій проект!

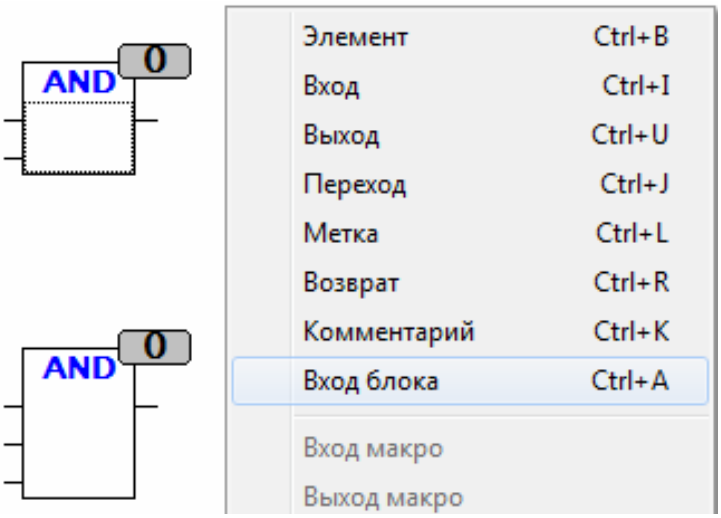


Рис. 3.32

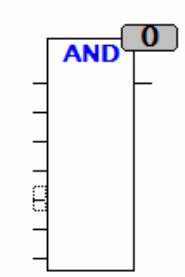


Рис. 3.33

Розділ 2. Обчислення для скомпонованого управляючого обчислювального комплексу (УОК) імовірності безвідмовної роботи або середній час напрацювання до відмови.

2.1 Переліки вхідних та вихідних сигналів та даних складаються відповідно до вимог стандарту і мають чітко визначати назву сигналу, форму його представлення, діапазон та періодичність змінення, джерело формування або приймача тощо. Форми відповідних таблиць наведені у табл. 3.1, …, 3.6.

До переліків вхідних необхідно вносити усі сигнали, що надходять до обчислювального комплексу:

– від технологічних датчиків;

– з операторських пультів (уставки, команди, сигнали, що визначають режими роботи системи тощо);

– від суміжних систем автоматизації.

До переліків вихідних необхідно вносити усі сигнали, що надходять з виходів обчислювального комплексу:

– на виконавчі механізми;
– на операторські пульти, табло (індикація стану об’єкта та системи,
– попереджувальна та аварійна сигналізація;
– до суміжних систем автоматизації.

2.2 Компонування УОК зазвичай здійснюють у наступному порядку:

1) прийняття рішення щодо загальної структури УОК (зосереджена чи розподілена) та в разі вибору розподіленої структури – щодо кількості віддалених контролерів;

2) вибір плати центрального процесорного пристрою (плати ЦПП);

3) вибір плат розширення пам’яті на твердотілих дисках (flash-пам’яті);

4) вибір периферійного обладнання (дисплея, клавіатури, принтера тощо) та (за необхідності) модулів, що підтримують його роботу;

5) вибір модулів введення / виведення сигналів;

6) вибір клемних плат, що забезпечують підключення модулів введення/ виведення до об’єкта автоматизації;

7) вибір комунікаційних модулів для підключення віддалених контролерів або інших УОК (за необхідності);

8) вибір пасивної об’єднуючої плати (кросової плати);

9) вибір конструктивних елементів (корпусів, шасі, каркасів тощо)

10) компонування віддалених контролерів (для розподілених УОК).

Компонування УОК здійснюється з використанням промислових каталогів фірм-виробників обчислювальної техніки та засобів автоматизації, які містять інформацію щодо номенклатури та технічних характеристик продукції. Для виконання контрольної роботи здобувачі вищої освіти можуть користуватись каталогами чи інформацією, яка є доступною в мережі Internet.

Під час виконання завдання необхідно керуватись основними вимогами, що висуваються до УОК, а саме:

– *достатня продуктивність* – ОК повинен виконувати необхідні обчислення досить швидко, щоб за час, витрачений на визначення керуючого впливу, ситуація на об’єкті не встигла істотно змінитися. У протилежному випадку управління буде неефективним, а можливо навіть шкідливим;

– *захищеність від впливу шкідливих факторів* навколишнього середовища – кліматичні умови на промисловому об'єкті можуть виявитися досить важкими. Техніка автоматизації повинна функціонувати при температурах навколишнього повітря від – 5°С до +70°С, вологості до 90% з конденсацією вологи, сильному пилоутворенні, вібраціях і прямих механічних ударах, в умовах дії потужних джерел електричних та магнітних полів, радіоперешкодах тощо;

– *висока надійність*;

– *ремонтопридатність і зручність обслуговування*;

– *задовільна вартість;*

– у*ніфікаці*я, що дає можливість застосовувати техніку різних виробників, не піклуючись про її електричну, програмну й конструктивну сумісність;

– *відповідність ергономічним нормам.*

При виконанні завдання слід звертати увагу на наступні особливості компонування УОК:

1) Розподілену структуру УОК доцільно використовувати за умов значної віддаленості датчиків, виконавчих механізмів та операторських пультів від місця розташування центрального комп'ютера. Для зв'язку віддалених контролерів з центральним комп'ютером необхідно використовувати промислові зовнішні інтерфейси, які дозволяють передавати дані на відповідні відстані.

2) У процесі вибору плати ЦПП слід звернути увагу на:

– типи шин внутрішнього інтерфейсу, що підтримуються платою, оскільки від цього залежить можливість використання тих чи інших модулів розширення;

– наявність у складі плати ЦПП відеоконтролера, оскільки від цього залежить необхідність використання окремого модуля для видачі даних на монітор;

– наявність на платі ЦПП послідовних портів промислових зовнішніх інтерфейсів RS-485, RS-422, Ethernet тощо, оскільки від цього залежить необхідність використання додаткових комунікаційних плат;

– наявність на платі ЦПП гнізд для мікросхем flash-пам'яті, оскільки від цього залежить необхідність використання окремих модулів твердотілих дисків.

3) При виборі модулів введення / виведення сигналів слід керуватись укладеними переліками сигналів (див. п.2.1). При цьому необхідно виходити з достатньої кількості каналів введення та виведення даних, можливості перетворення вхідних сигналів відповідного типу (аналогових, дискретних, частотних) та діапазону змінення, наявності на модулях гальванічної розв'язки (усі зовнішні сигнали не повинні мати спільних потенціалів з джерелами живлення УОК).

4) Обираючи клемні плати, необхідно забезпечити достатню кількість клемних з'єднувачів, забезпечити гальванічну розв'язку сигналів, якщо вона відсутня на модулях введення / виведення.

5) Комунікаційні модулі потрібно вибирати, якщо існує необхідність організації зв'язку УОК з іншими системами автоматизації або віддаленими контролерами збирання та передачі даних.

6) Пасивну об'єднуючу плату обирають, виходячи з кількості модулів розширення, що увійшли до складу УОК та типу внутрішнього інтерфейсу, за допомогою якого ці модулі мають об'єднуватись в комплексі.

7) Конструктивні елементи обирають, орієнтуючись на тип раніше обраної пасивної об'єднуючої плати, тобто з огляду на можливість розміщення усередині конструктивного елемента усіх модулів УОК.

8) Послідовність компонування віддалених контролерів визначається особливостями їхнього конструктивного виконання, але, в цілому, є подібною до послідовності компонування центрального комп’ютера.

За результатами компонування створюється структурна схема УОК згідно з вимогами стандарту. Приклад структурної схеми УОК наведено на рис. 2.1.

Структурна схема має відобразити:

– склад УОК (усі плати, модулі, пристрої та інші елементи);

– зв'язок між елементами УОК (наявність фізичних ліній передачі сигналів та даних);

– взаємне розташування елементів УОК (угрупування певних елементів в межах спільних конструктивів, панелей, приміщень тощо).

Елементи схеми зображуються, зазвичай, прямокутниками 20×30 мм, шини внутрішнього інтерфейсу – вузькими прямокутниками довільної довжини висотою 10 мм .

Зв’язок між елементами УОК на структурних схемах зображають товстими смугами або смугастими чи об’ємними стрілками. Зазвичай, *об’ємні стрілки вказують на паралельний*, а *смугасті – на послідовний спосіб передачі інформації*. Направленість об’ємних стрілок співпадає з напрямом передачі сигналів по шині даних внутрішнього інтерфейсу.

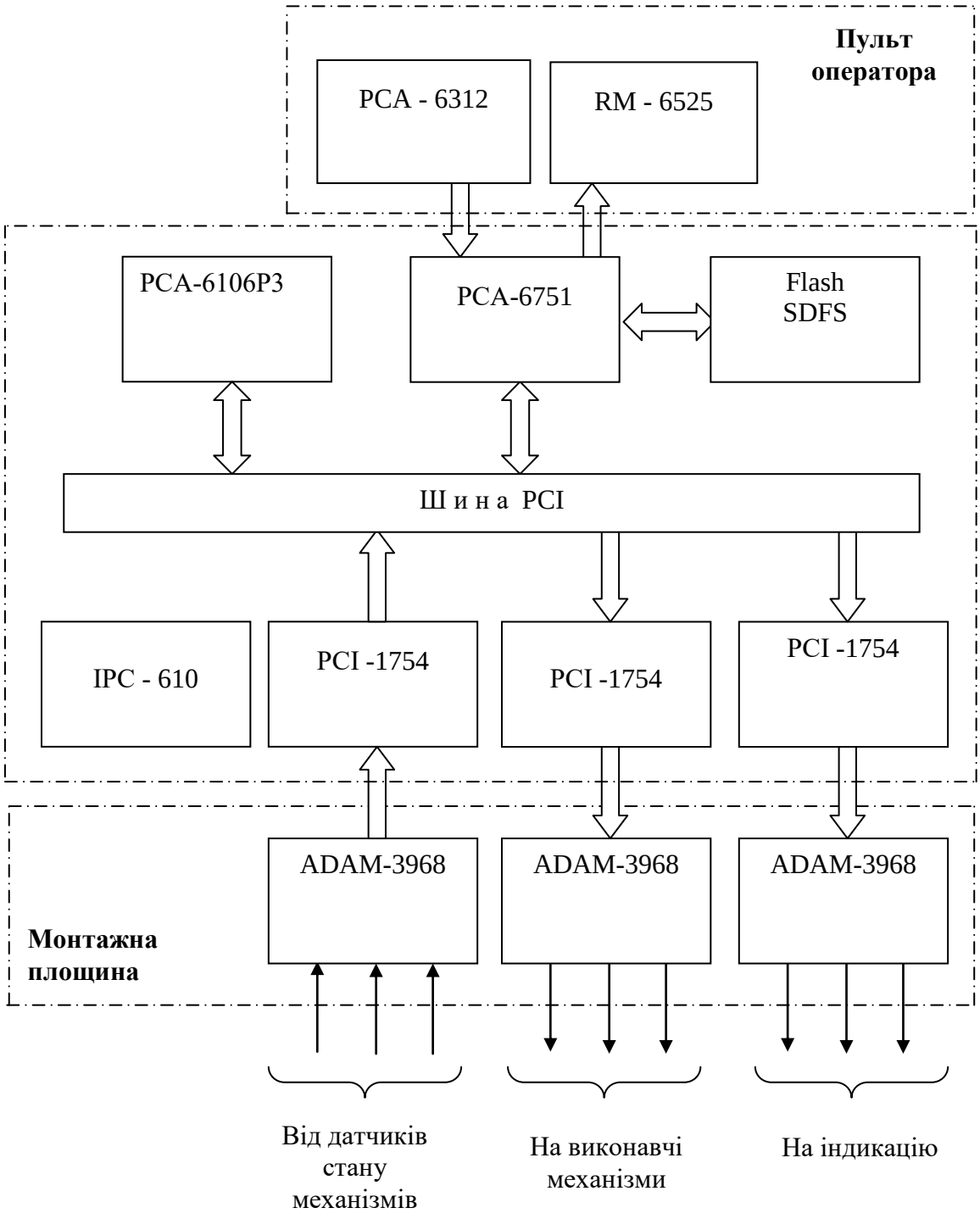


Рис. 2.1. Структурна схема УОК

Для відображення взаємного розташування елементів УОК на структурній схемі використовують штрих-пунктирні контури, що охоплюють елементи, які розміщені у спільних конструктивах (каркасах, корпусах, щитах) або приміщеннях (машинних залах, операторських постах тощо).

Структурна схема має обов’язково містити таблицю переліку елементів, форма якої наведена на рис. 2.2.

Таблиця 2.1 Перелік елементів

Поз. позначення	Найменування	Кіль-кість	Примітка
A1	Одноплатний промисловий комп'ютер половинного розміру PCA – 6751	1	Містить відеоконтролер
A2, A3	Плата цифрового введення / виведення 64-канальна PCI – 1754	2	з гальванічною ізоляцією
A4…A7	Клемна плата ADAM – 3968	4	

Рис. 2.2. Фрагмент переліку елементів структурної схеми

2.3. Обчислення показників надійності УОК ведеться у три етапи за наступною послідовністю:

1) складання логічної схеми надійності;

2) вибір й уточнення показників надійності окремих елементів УОК;

3) безпосереднє обчислення показників надійності.

На першому етапі УОК представляється у вигляді логічної схеми, що характеризує стан (працездатний чи непрацездатний) комплексу в залежності від стану окремих елементів. При цьому виходять з того, що елементи на логічній схемі надійності можуть сполучатись трьома способами: послідовним, паралельним навантаженим та паралельним ненавантаженим.

Послідовне (основне) з'єднання (рис. 2.3) відповідає випадку, коли при відмові будь-якого елемента відмовляє вся система. За таким з'єднанням напрацювання до відмови усієї системи дорівнює напрацюванню того елемента, у якого вона є найменшою

$$T_c = min(T_j), j = 1,2, \dots, n$$

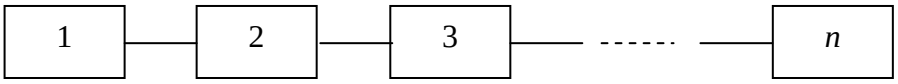


Рис. 2.3. Послідовне з'єднання елементів на логічній схемі надійності

Паралельне навантажене з'єднання (рис. 2.4) відповідає випадку, коли система зберігає працездатність доки залишається працездатним хоча б один задіяний в роботі елемент. При такому з'єднанні напрацювання до відмови усієї системи дорівнює напрацюванню того елемента, у якого вона є максимальною

$$T_e = max(T_j), j = 1,2, \dots, n$$

Паралельне ненавантажене з'єднання (рис. 2.5) відповідає випадку, коли при відмові елемента підключається до роботи черговий резервний елемент й, таким чином, система зберігає працездатність.

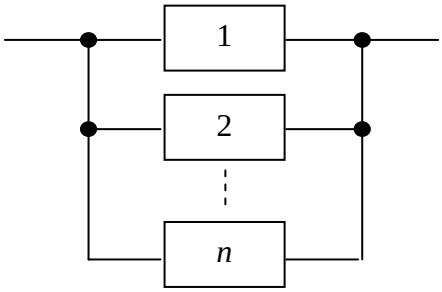


Рис. 2.4. Паралельне навантажене з'єднання елементів на логічній схемі надійності

Відтак, напрацювання до відмови усієї системи дорівнює сумі напрацювань до відмови усіх елементів.

$$T_c = \sum_{0=1}^{n} T_j$$

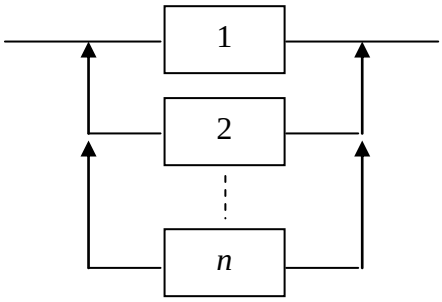


Рис. 2.5. Паралельне ненавантажене з'єднання елементів на логічній схемі надійності

На *другому етапі* уточнюють показники надійності окремих елементів (модулів, плат, пристроїв), що входять до складу УОК (Додаток 1). Якщо в якості показника надійності елемента фігурує середнє напрацювання до відмови m_t, слід перейти до інтенсивності відмов λ_j, скориставшись формулою

$$\lambda_j = \frac{1}{m_{tj}}$$

На *третьому етапі* здійснюють безпосередній розрахунок імовірності безвідмовної роботи системи $p_c(t)$ або середнього часу напрацювання системи до відмови m_{tc}, виходячи з припущення про показове розподілення часу безвідмовної роботи та використовуючи наступні формули.

При послідовному з'єднанні:

– інтенсивність відмов з'єднання:

$$\lambda_c = \sum_{j=1}^{n} \lambda_j \qquad (3.1)$$

– імовірність безвідмовної роботи з'єднання:

$$p_c(t) = \prod_{j=1}^{n} p_j(t) \qquad (3.2)$$

або

$$p_c = e^{-\lambda_c}$$

де $p_j = e^{-\lambda_j t}$ – функція надійності j -го елемента;

– напрацювання до відмови з'єднання

$$m_{t_c} = 1 \Big/ \sum_{j=1}^{n} \frac{1}{m_{t_j}} = {}^{1}\!/_{\lambda_c} \qquad (3.4)$$

При паралельному навантаженому з'єднанні:

– імовірність безвідмовної роботи з'єднання

$$p_c(t) = 1 - \prod_{j=1}^{n} \left[1 - p_j(t)\right] \qquad (3.5)$$

При паралельному ненавантаженому з'єднанні:

– імовірність безвідмовної роботи з'єднання

$$p_n(t) = p_0 + p_n(t) + p_n(t) + \cdots p_{n-1}(t) \qquad (3.6)$$

або

$$p_n(t) = e^{-\lambda_t} \sum_{j=0}^{n-1} (\lambda t)^j \Big/ j! \qquad (3.7)$$

У додатку 2 наведено приклад розрахунку показників надійності для достатньо складної логічної схеми, розібравши який можна з’ясувати усі особливості використання зазначених формул.

Додаток 1

Орієнтовні показники надійності модулів УОК

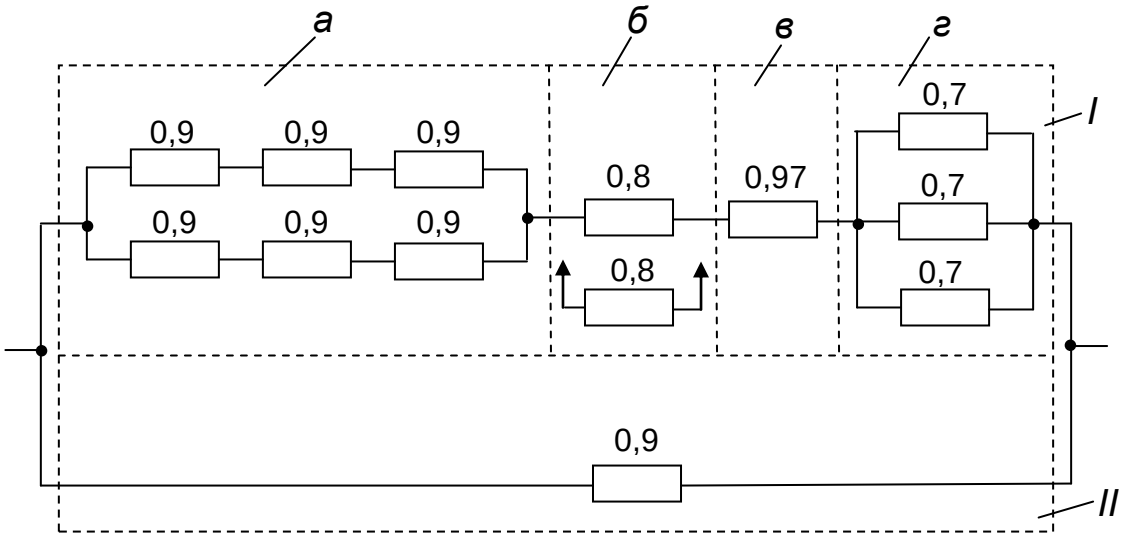
№ №	Тип модуля (плати)	m_t, годин
	Плата ЦПП повного розміру з підтримкою шин PCI та ISA, що містить відео контролер	
	Плата ЦПП повного розміру з підтримкою однієї з шин (PCI чи ISA), що містить відеоконтролер	
	Плата ЦПП повного розміру з підтримкою шин PCI та ISA, що не містить відео контролера	
№	Універсальний модуль АЦП (до 8 каналів)	
5	Універсальний модуль АЦП (8 -16 каналів)	
6	Модуль ЦАП (до 8 каналів)	
7	Модуль введення / виведення дискретних сигналів (до 24 каналів)	
8	Модуль введення / виведення дискретних сигналів (24 - 48 каналів)	
9	Модуль введення / виведення дискретних сигналів (більше 48 каналів)	
	Пасивна об’єднануюча плата (до 8 слотів)	
	Пасивна об’єднануюча плата (більше 8 слотів)	
	Шасі під пасивну плату до 8 слотів	
	Шасі під пасивну плату більше 8 слотів	

Додаток 2

Приклад розрахунку показників надійності

Надана система, схема надійності якої зображена на рисунку. Необхідно визначити ймовірність безвідмовної роботи системи при відомих ймовірностях безвідмовної роботи її елементів (значення ймовірностей зазначені на рисунку).

Рисунок. Логічна схема надійності



Рішення.

З логічної схеми випливає, що система складається із двох (I та II) пристроїв різної надійності.

Пристрій I містить чотири вузли:

З рис. 1 видно, що система складається із двох (I та II) пристроїв різної надійності.

Пристрій I містить чотири вузли:

а – дубльований вузол із постійно включеним резервом, причому кожна частина вузла складається з трьох послідовно з'єднаних (у сенсі надійності) елементів;

б – дубльований вузол за способом заміщення;

в – вузол з одним нерезервованим елементом;

г – резервований вузол із подвійним постійно включеним резервом.

Пристрій *II* являє собою нерезервований пристрій, надійність якого відома.

Оскільки обидва пристрої мають різну надійність, то на підставі формули (3.5) маємо

$$P_c(t) = 1 - \prod_{i=0}^{n}[1 - p_i(t)] = 1 - [1 - p_I(t)] \cdot [1 - p_{II}(t)]$$

Знайдемо ймовірність $p_c(t)$. Імовірність безвідмовної роботи пристрою *I* дорівнює добутку ймовірностей безвідмовної роботи всіх вузлів, тобто

$$p_I(t) = p_a \cdot p_б \cdot p_в \cdot p_г$$

У вузлі *а* кількість елементів основного й резервного ланцюга $n = 3$. Тоді на підставі формул (3.2) та (3.5)

$$p_c(t) = 1 - \prod_{j=1}^{n} \left[1 - p_j(t)\right]^2 = 1 - [1 - 0{,}9^3]^2 \approx 0{,}93$$

У вузлі *б* кратність загального резервування заміщенням $m = 1$, тоді на підставі формули (3.7) маємо

$$P_б(t) = e^{\lambda_0 t} \sum_{i=0}^{I} \frac{(\lambda_0 t)^2}{i!} = e^{-\lambda_0 t}(1 + \lambda_0 t) \approx 0{,}8(1 + 0{,}22) = 0{,}976$$

У вузлі *г* застосоване «гаряче» резервування. На підставі формули (3.5)

$$P_г(t) = 1 - \prod_{i=I}^{3} [1 - p_i(t)] = 1 - [1 - 0{,}7]^3 = 0{,}973$$

Імовірність безвідмовної роботи пристрою *I* буде

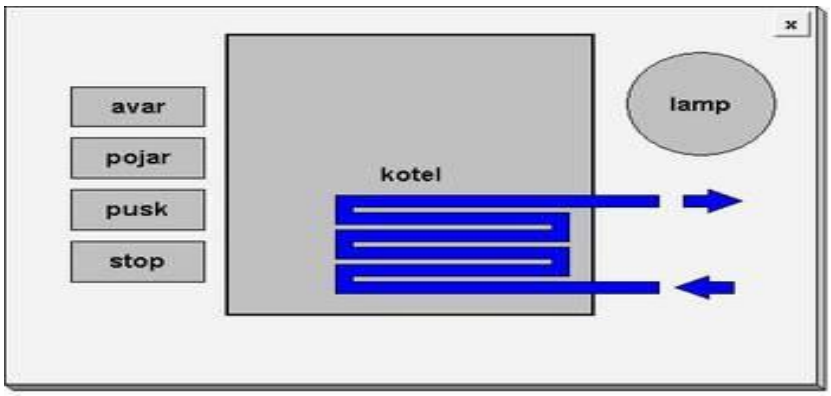
$$p_I(t) = p_a \cdot p_б \cdot p_в \cdot p_г = 0{,}93 \cdot 0{,}976 \cdot 0{,}97 \cdot 0{,}973 = 0{,}857$$

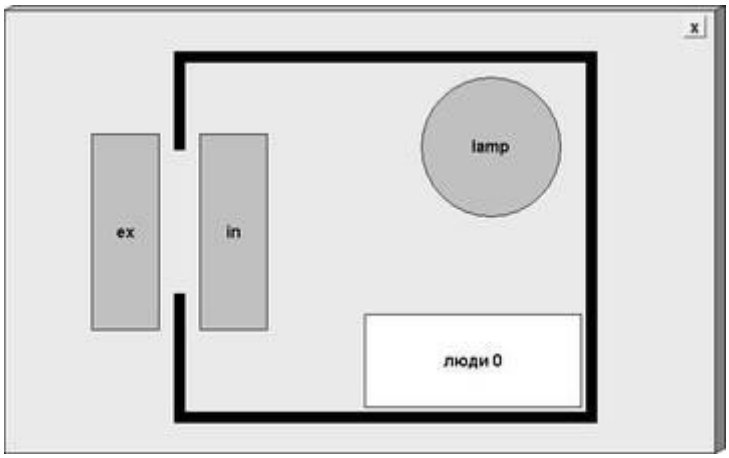
Тоді ймовірність безвідмовної роботи всієї системи буде

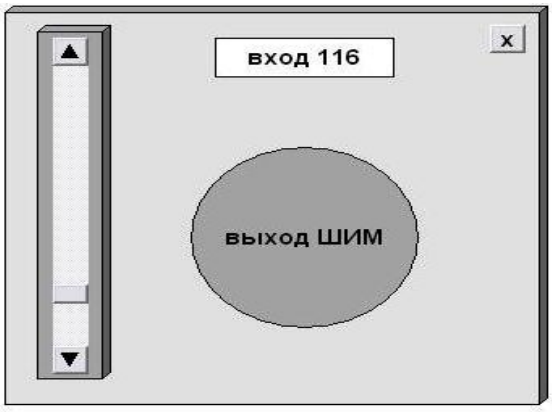
$$P_с(t) = 1 - [1 - p_I(t)] \cdot [1 - p_{II}(t)] = 1 - (1 - 0{,}857) \cdot (1 - 0{,}9) = 0{,}986$$

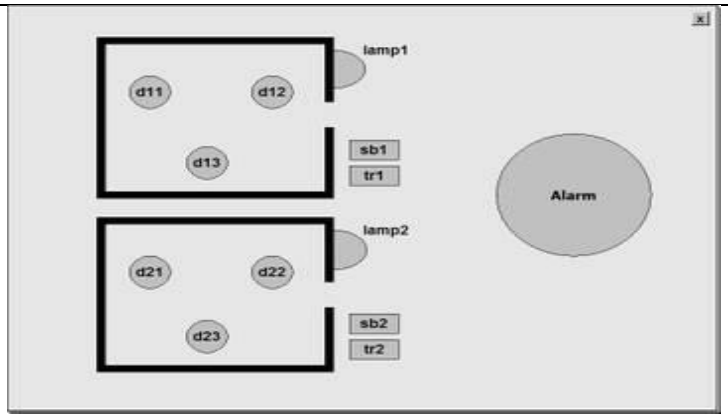
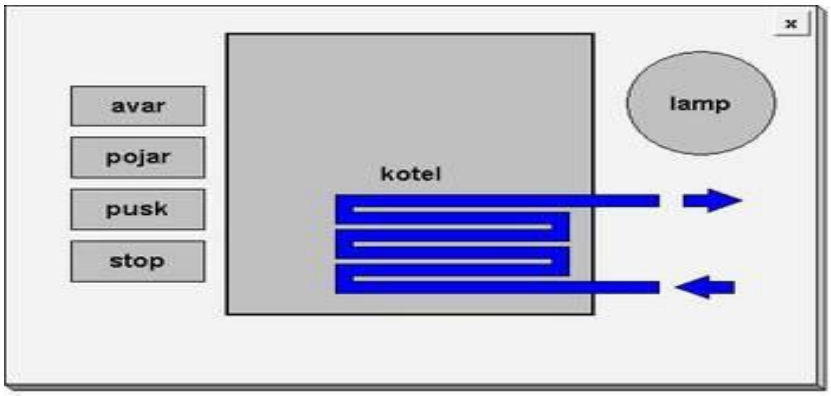
Додаток 3

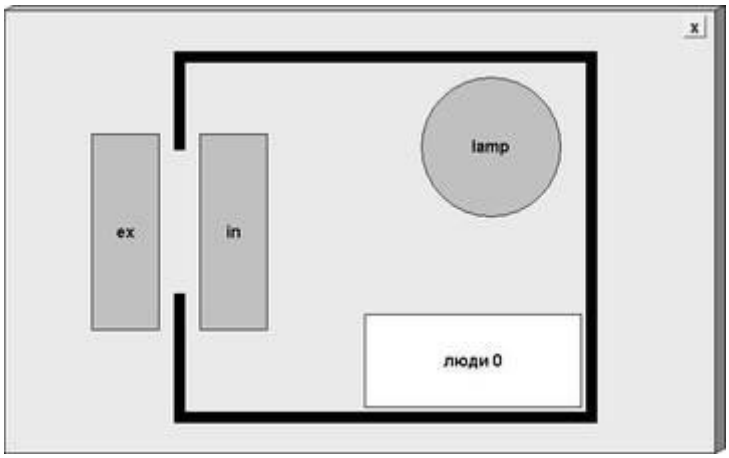
Номер варіанту для здобувача вищої освіти заочної форми навчання визначається згідно номеру в журналі навчальної групи.

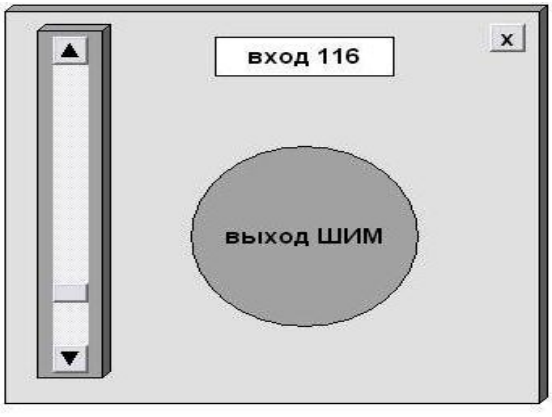
№ вар.	Завдання
1	1. Типи програмованих логічних контролерів. 2. Мови програмування промислових логічних контролерів за стандартом МЕК: IEC 61131-3. 3. Управління котлом (мова написання програми: CFC).  Необхідно реалізувати: Включення сигналізації (lamp) при виникненні будь-якої з аварій (avar або pojar). Відключення котла (kotel) при виникненні будь-якої з аварій. Включення котла з кнопки (pusk), за умови відсутності аварій. Відключення котла з кнопки (stop). 4. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
2	1. Конструктивні виконання і способи кріплення контролерів. 2. Мови програмування промислових логічних контролерів за стандартом МЕК: LD, SFC. 3. Ручне управління клапаном (мова написання програми: CFC).  Необхідно реалізувати:

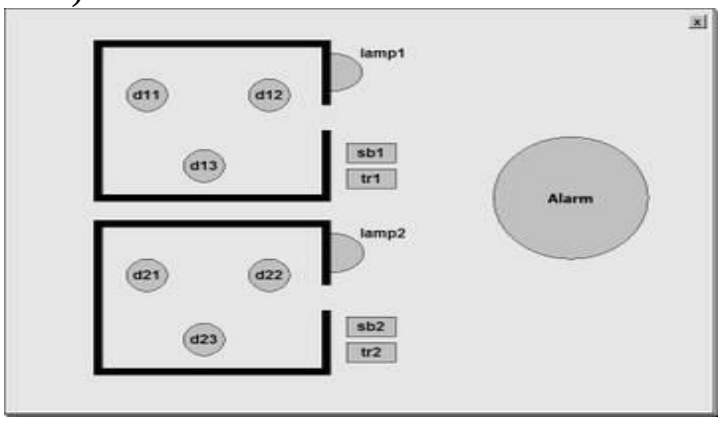
	Плавне збільшення і зменшення ступеня відкриття клапана (pol) з зовнішніх кнопок (plus або minus). Видачу керуючого сигналу 4-20 мА (out) з виходу ПЛК на клапан. Відображення ступеня відкриття клапана (pol) в процентах. Сигналізацію про досягнення кінцевих положень (zakr і otkr). 4. Обчислення показників надійності УОК: а – 0,92; б – 0,8; в – 0,97; г – 0,78.
3	1. Архітектура програмованих логічних контролерів. 2. Мови програмування промислових логічних контролерів за стандартом МЕК: ST, SFC. 3. Управління освітленням в кімнаті (мова написання програми: CFC).  На вході встановлено два дискретних датчика: один зовні (ex), інший усередині кімнати (in). Коли спрацьовує спочатку зовнішній датчик, потім внутрішній, це означає, що людина зайшла в кімнату.Коли спрацьовує спочатку внутрішній датчик, потім зовнішній, це означає, що людина вийшла з кімнати. Необхідно розрахувати кількість людей (ludi) в кімнаті. Якщо людина увійшла – включити світло (lamp), якщо людина вийшла – вимкнути світло (lamp). Поки залишається хоча б одна людина, світло повинне бути включеним. 5. Обчислення показників надійності УОК: а – 0,94; б – 0,8; в – 0,92; г – 0,76.
4	1. Призначення програмного продукту Install Target з програмного пакету CoDeSys. 2. Коаксіальний кабель. 3. Включення насоса ((мова написання програми: CFC). а – 0,88; б – 0,8; в – 0,98; г – 0,8.

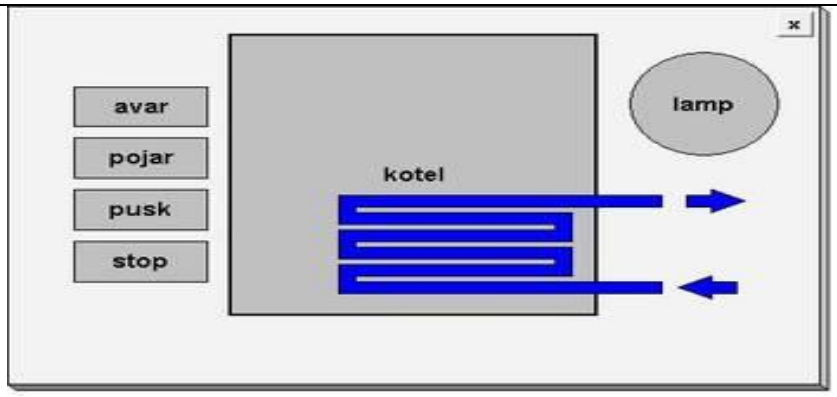
	 При натисканні на кнопку (pusk), насос (nasos) повинен включитися і пропрацювати 10 секунд, потім автоматично відключитися. Необхідно підраховувати кількість включень (c1) двигуна. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
5	1. Призначення програмного забезпечення *CoDeSys-V2.3* з програмного пакету CoDeSys. 2. Поширені протоколи мережевих систем автоматизованого управління. 3. Формування ШІМ-сигналу (програмно, функ. блок BLINK, бібліотека UTIL.lib) (мова написання програми: CFC).  При зміні сигналу на аналоговому вході (inp) необхідно змінювати шпаруватість вихідних імпульсів (out) в діапазоні від 20 до 50%. Період ШІМ дорівнює 1 секунді. 1. Обчислення показників надійності УОК: а – 0,92; б – 0,86; в – 0,8; г – 0,7.
6	1. Призначення програмного забезпечення Gateway-server з програмного пакету CoDeSys. 2. Вита пара UTP. 3. Система пожежної сигналізації будівлі (мова написання програми: CFC).

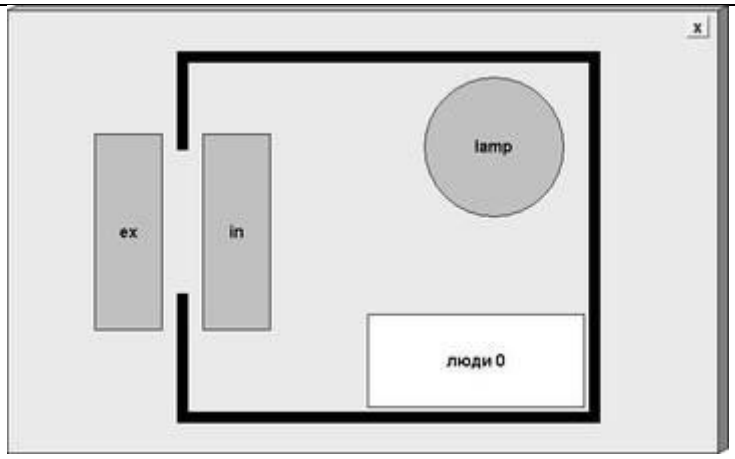
	 У будівлі дві однакові кімнати. У кожній кімнаті встановлено три пожежних датчика (d11, d12, d13 і d21, d22, d23), кнопка ручного включення сигналізації (tr1 і tr2) і кнопка ручного відключення сигналізації (sb1 і sb2). Для кожної кімнати передбачена сигнальна лампа (lamp1, lamp2). Сигналізація пожежі (alarm) є спільною для обох кімнат. Якщо в кімнаті спрацьовує хоча б один з датчиків, то загоряється сигнальна лампа для відповідної кімнати. Лампа гасне, якщо все датчики в кімнаті відключені. Якщо в кімнаті спрацьовує будь-які два з трьох датчиків, то включається пожежна сигналізація. Сигналізація працює до тих пір, поки її не відключать відповідної кнопкою. Сигналізація може бути включена кнопкою перевірки незалежно від стану датчиків. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
7	1. Призначення програмного забезпечення *CoDeSys-HMI* з програмного пакету CoDeSys. 2. Мови програмування промислових логічних контролерів за стандартом МЕК: SFC, LD. 3. Управління котлом (мова написання програми: CFC).  Необхідно реалізувати: Включення сигналізації (lamp) при виникненні будь-якої з аварій (avar або pojar). Відключення котла (kotel) при виникненні будь-якої з аварій. Включення котла з кнопки (pusk), за умови відсутності аварій.

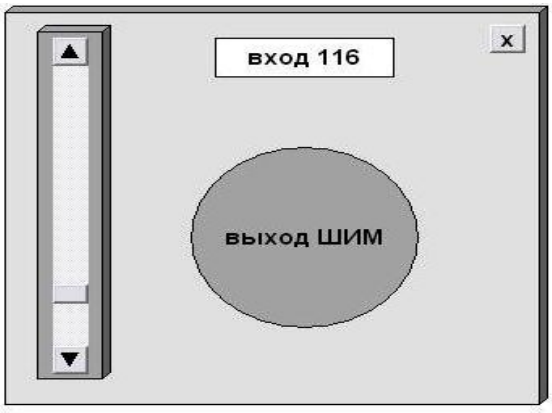
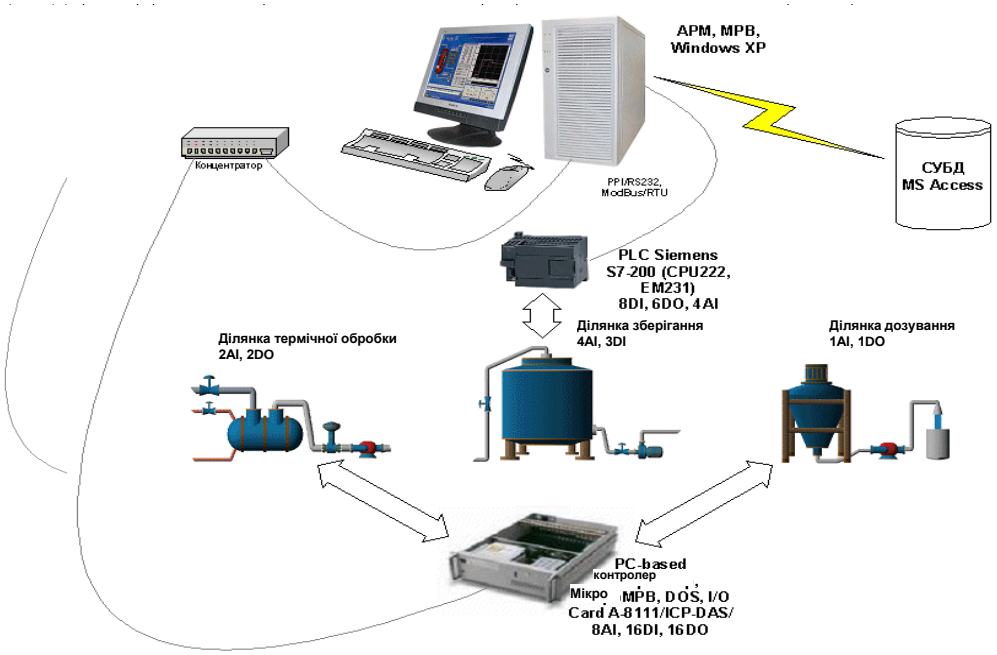
	Відключення котла з кнопки (stop). 5.Обчислення показників надійності УОК: а – 0,9; б – 0,76; в – 0,8; г – 0,76.
8	1. Призначення програмного забезпечення Конфігуратор СП-270 (розробник НП ОВЕН). 2. Мови програмування промислових логічних контролерів за стандартом МЕК: FBD 3. Ручне управління клапаном (мова написання програми: CFC).  Необхідно реалізувати: Плавне збільшення і зменшення ступеня відкриття клапана (pol) з зовнішніх кнопок (plus або minus). Видачу керуючого сигналу 4-20 мА (out) з виходу ПЛК на клапан. Відображення ступеня відкриття клапана (pol) в процентах. Сигналізацію про досягнення кінцевих положень (zakr і otkr). 4. Обчислення показників надійності УОК: а – 0,9; б – 0,88; в – 0,8; г – 0,72.
9	1. Структура АСУ ТП (загальні відомості). 2. Мови програмування промислових логічних контролерів за стандартом МЕК: FBD. 3. Управління освітленням в кімнаті (мова написання програми: CFC).  На вході встановлено два дискретних датчика: один зовні (ex), інший усередині кімнати (in).

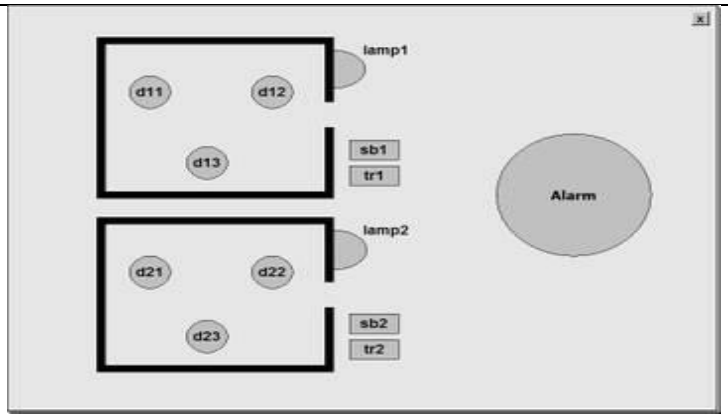
	Коли спрацьовує спочатку зовнішній датчик, потім внутрішній, це означає, що людина зайшла в кімнату. Коли спрацьовує спочатку внутрішній датчик, потім зовнішній, це означає, що людина вийшла з кімнати. Необхідно розрахувати кількість людей (ludi) в кімнаті. Якщо людина увійшла – включити світло (lamp), якщо людина вийшла – вимкнути світло (lamp). Поки залишається хоча б одна людина, світло повинне бути включеним. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,88; г – 0,8.
10	1. Принципи реалізації візуалізації безпосередньо у середовищі програмування. 2. Коаксіальний кабель. 3. Включення насоса ((мова написання програми: CFC).  При натисканні на кнопку (pusk), насос (nasos) повинен включитися і пропрацювати 10 секунд, потім автоматично відключитися. Необхідно підраховувати кількість включень (c1) двигуна. 4. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
11	1. Принципи реалізації Web-візуалізації. 2. Конструктивні відмінності стандартних інтерфейсів: RS-232. 3. Формування ШІМ-сигналу (програмно, функ. блок BLINK, бібліотека UTIL.lib) (мова написання програми: CFC). 

	При зміні сигналу на аналоговому вході (inp) необхідно змінювати шпаруватість вихідних імпульсів (out) в діапазоні від 20 до 50%. Період ШІМ дорівнює 1 секунді. Обчислення показників надійності УОК: а – 0,92; б – 0,8; в – 0,97; г – 0,78.
12	1. Призначення програмного забезпечення CoDeSys-HMI з програмного пакету CoDeSys. 2. Мови програмування промислових логічних контролерів за стандартом МЕК: SFC, IL. 3. Система пожежної сигналізації будівлі (мова написання програми: CFC).  У будівлі дві однакові кімнати. У кожній кімнаті встановлено три пожежних датчика (d11, d12, d13 і d21, d22, d23), кнопка ручного включення сигналізації (tr1 і tr2) і кнопка ручного відключення сигналізації (sb1 і sb2). Для кожної кімнати передбачена сигнальна лампа (lamp1, lamp2). Сигналізація пожежі (alarm) є спільною для обох кімнат. Якщо в кімнаті спрацьовує хоча б один з датчиків, то загоряється сигнальна лампа для відповідної кімнати. Лампа гасне, якщо все датчики в кімнаті відключені. Якщо в кімнаті спрацьовує будь-які два з трьох датчиків, то включається пожежна сигналізація. Сигналізація працює до тих пір, поки її не відключать відповідної кнопкою. Сигналізація може бути включена кнопкою перевірки незалежно від стану датчиків. 5. Обчислення показників надійності УОК: – 0,8; в – 0,92; г – 0,76.
13	1. Принципи реалізації Web-візуалізації. 2. Конструктивні відмінності стандартних інтерфейсів: Ethernet connection. 3. Управління котлом (мова написання програми: CFC).

	 Необхідно реалізувати: Включення сигналізації (lamp) при виникненні будь-якої з аварій (avar або pojar). Відключення котла (kotel) при виникненні будь-якої з аварій. Включення котла з кнопки (pusk), за умови відсутності аварій. Відключення котла з кнопки (stop). 4. Обчислення показників надійності УОК: а – 0,88; б – 0,8; в – 0,98; г – 0,8.
14	1. Вибір засобів відображення інформації. 2. Розробка програмного забезпечення для мікропроцесорних систем. 3. Ручне управління клапаном (мова написання програми: CFC).  Необхідно реалізувати: Плавне збільшення і зменшення ступеня відкриття клапана (pol) з зовнішніх кнопок (plus або minus). Видачу керуючого сигналу 4-20 мА (out) з виходу ПЛК на клапан. Відображення ступеня відкриття клапана (pol) в процентах. Сигналізацію про досягнення кінцевих положень (zakr і otkr). 4. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
15	1. Конфігурація каналу зв’язку по протоколу Modbus. 2. Архітектура програмованих логічних контролерів. 3. Управління освітленням в кімнаті (мова написання програми: CFC).

	 На вході встановлено два дискретних датчика: один зовні (ex), інший усередині кімнати (in). Коли спрацьовує спочатку зовнішній датчик, потім внутрішній, це означає, що людина зайшла в кімнату.Коли спрацьовує спочатку внутрішній датчик, потім зовнішній, це означає, що людина вийшла з кімнати. Необхідно розрахувати кількість людей (ludi) в кімнаті. Якщо людина увійшла – включити світло (lamp), якщо людина вийшла – вимкнути світло (lamp). Поки залишається хоча б одна людина, світло повинне бути включеним. 4. Обчислення показників надійності УОК: а – 0,94; б – 0,8; в – 0,92; г – 0,76.
16	1. Призначення програмного забезпечення CoDeSys-Web-server з програмного пакету CoDeSys. 2. Вибір параметрів управління. 3. Включення насоса ((мова написання програми: CFC).  При натисканні на кнопку (pusk), насос (nasos) повинен включитися і пропрацювати 10 секунд, потім автоматично відключитися. Необхідно підраховувати кількість включень (c1) двигуна. 4. Обчислення показників надійності УОК: а – 0,9; б – 0,82; в – 0,95; г – 0,74.
17	1. Призначення програмного забезпечення *DDE-server* з програмного пакету CoDeSys. 2. Поширені протоколи мережевих систем автоматизованого

	управління. 3. Формування ШІМ-сигналу (програмно, функ. блок BLINK, бібліотека UTIL.lib) (мова написання програми: CFC).  При зміні сигналу на аналоговому вході (inp) необхідно змінювати шпаруватість вихідних імпульсів (out) в діапазоні від 20 до 50%. Період ШІМ дорівнює 1 секунді. Обчислення показників надійності УОК: а – 0,92; б – 0,86; в – 0,8; г – 0,7.
18	1. Основні типи функцій запитів (команд) по протоколу Modbus. 2. Об’єкт та система управління.  Рисунок – Об’єкт та система управління 3. Система пожежної сигналізації будівлі (мова написання програми: CFC).



У будівлі дві однакові кімнати. У кожній кімнаті встановлено три пожежних датчика (d11, d12, d13 і d21, d22, d23), кнопка ручного включення сигналізації (tr1 і tr2) і кнопка ручного відключення сигналізації (sb1 і sb2).

Для кожної кімнати передбачена сигнальна лампа (lamp1, lamp2). Сигналізація пожежі (alarm) є спільною для обох кімнат.

Якщо в кімнаті спрацьовує хоча б один з датчиків, то загоряється сигнальна лампа для відповідної кімнати. Лампа гасне, якщо все датчики в кімнаті відключені. Якщо в кімнаті спрацьовує будь-які два з трьох датчиків, то включається пожежна сигналізація. Сигналізація працює до тих пір, поки її не відключать відповідної кнопкою.

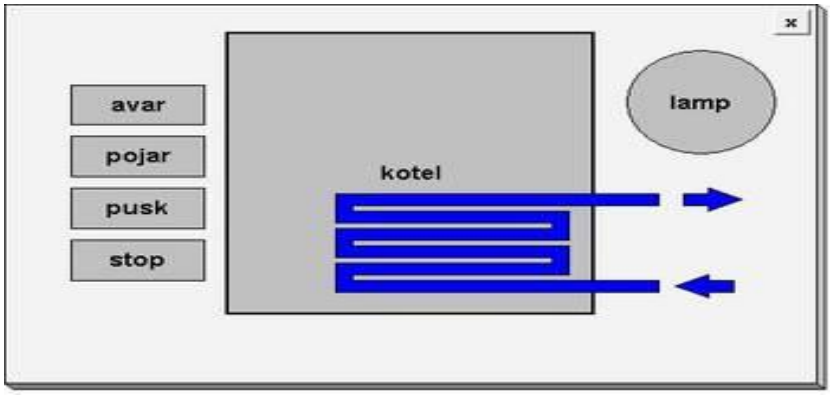
Сигналізація може бути включена кнопкою перевірки незалежно від стану датчиків.

4. Обчислення показників надійності УОК:

а – 0,9; б – 0,82; в – 0,88; г – 0,8.

19

1. Налагодження програм
2. Проектування систем керування безперервними об’єктами.
3. Управління котлом (мова написання програми: CFC).



Необхідно реалізувати:

Включення сигналізації (lamp) при виникненні будь-якої з аварій (avar або pojar).

Відключення котла (kotel) при виникненні будь-якої з аварій.

Включення котла з кнопки (pusk), за умови відсутності аварій.

Відключення котла з кнопки (stop).

4. Обчислення показників надійності УОК:

	а – 0,9; б – 0,82; в – 0,95; г – 0,74.
20	1. Вибір автоматичних регуляторів і виконавчих пристроїв 2. Розробка програмного забезпечення для мікропроцесорних систем. 3. Ручне управління клапаном (мова написання програми: CFC).  Необхідно реалізувати: Плавне збільшення і зменшення ступеня відкриття клапана (pol) з зовнішніх кнопок (plus або minus). Видачу керуючого сигналу 4-20 мА (out) з виходу ПЛК на клапан. Відображення ступеня відкриття клапана (pol) в процентах. Сигналізацію про досягнення кінцевих положень (zakr і otkr). 4. Обчислення показників надійності УОК: а – 0,92; б – 0,86; в – 0,8; г – 0,7.

ЛІТЕРАТУРА

1. Петров И.В. Программируемые контроллеры. Стандартные языки и приемы прикладного программирования / Петров В.И. – М.: 000 «СОЛОН-Пресс», 2004.
2. Веб-страница фирмы Smart Software Soluions Gmb производителя среды CoDeSys. // htth: www.3s- software.com/
3. Среда программирования CODESYS v.2.3. URL: https://owen.ua/ru/programmnoe-obespechenie/sreda-programmirovaniya-codesys-2-3-i-drugoe-programmnoe-obespechenie-dlya-oven-plk
4. Петров И.В. Отладка прикладных ПЛК программ в CoDeSys // Промышленные АСУ и контроллеры. 2006. № 2.
5. Зюбин В.Е. Программирование ПЛК: языки МЭК 61131-3 и возможные альтернативы // Промышленные АСУ и контроллеры. 2005. № 11.
6. Лисаченко І. Г. Програмне забезпечення комп'ютерно-інтегрованих систем управління хіміко-технологічними процесами: навч.-метод. посіб./ І.Г. Лисаченко. – Х.: НТУ «ХПІ», 2012. – 112 с.
7. Иванов А.Н., Золотарев С.В. Построение АСУТП на базе концепции открытых систем. http;/www.osp.ru/pcworld/1998/01/40/htm
8. Основи надійності цифрових систем. / В.С. Харченко та ін. – Харків: Національний аерокосмічний університет «Харківський авіаційний інститут», 2004.
9. Азаров В.Н., Стрельников В.П. Надежность систем управления и автоматики. – К.: НАУ, 2004. – 164 с.