Міністерство освіти і науки України

Національний університет водного господарства та природокористування

Кафедра автоматизації, електротехнічних та комп‘ютерно-інтегрованих технологій

04-03-389М

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт з дисципліни **«Проектування комп‘ютерно-інтегрованих та робототехнічних систем»** для здобувачів вищої освіти другого (магістерського) рівня за освітньо-професійною програмою «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» спеціальності 174 «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» денної та заочної форм навчання

Рекомендовано науково-методичною радою з якості ННІ ЕАВГ
Протокол № 11 від 02.07.2024 р.

Рівне – 2024

Методичні вказівки до виконання лабораторних робіт з дисципліни «Проектування комп‘ютерно-інтегрованих та робототехнічних систем» для здобувачів вищої освіти другого (магістерського) рівня за освітньо-професійною програмою «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» спеціальності 174 «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» денної та заочної форм навчання. [Електронне видання] / Стеценко А. М. – Рівне : НУВГП, 2024. – 183 с.

Укладач: Стеценко А. М., к.т.н., доцент кафедри автоматизації, електротехнічних та комп‘ютерно-інтегрованих технологій.

Відповідальний за випуск: Древецький В. В., д.т.н., професор, завідувач кафедри автоматизації, електротехнічних та комп‘ютерно-інтегрованих технологій.

Керівник групи забезпечення спеціальності 174 «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» Рудик А. В., д.т.н., професор, професор кафедри автоматизації, електротехнічних та комп‘ютерно-інтегрованих технологій.

Зміст

Робота №1. Розробка і дослідження системи контролю та керування функціонуванням адміністративної будівлі (система типу «Розумній дім») на базі програмованого логічного контролера.

1. Мета роботи

Навчитися розробляти системи контролю та керування функціонуванням адміністративної будівлі (система типу «Розумній дім») на базі програмованого логічного контролера. Вивчити будову, правила підключення, характеристики, меню контролера Zelio Logic2. Навчитися програмувати контролер Zelio Logic2 мовою драбинкових діаграм (LD – Ladder Diagram).

2. Теоретичні відомості

2.1 Системи контролю та керування типу «Розумний дім»

Розумний будинок, або "smart home", являє собою концепцію сучасного житла, де різні системи автоматизовані для підвищення комфорту, безпеки та ефективності. Ця ідея стає дедалі популярнішою у всьому світі, включаючи Україну, завдяки розвитку технологій Інтернету речей (IoT) та штучного інтелекту.

Поняття «інтелектуальний дім» було сформульовано Інститутом інтелектуальної будівлі у Вашингтоні в 70-ті роки XX століття як «будинок, що забезпечує продуктивне і ефективне використання робочого простору». Основною особливістю інтелектуальної будівлі є об'єднання окремих підсистем різних виробників у єдиний керований комплекс.

Однією з головних переваг розумного будинку є його здатність забезпечувати зручність і комфорт. Завдяки автоматизації, мешканці можуть дистанційно керувати освітленням, опаленням, кондиціонуванням повітря та іншими побутовими пристроями за допомогою смартфонів або голосових асистентів. Наприклад, ви можете налаштувати систему так, щоб вона вмикала світло або підігрівала воду до певного часу, або ж автоматично відчиняла штори вранці. Це не лише підвищує рівень комфорту, але й сприяє економії енергії.

Безпека також є важливим аспектом розумного будинку. Системи відеоспостереження, датчики руху, розумні замки та

сигналізація можуть бути інтегровані для забезпечення повного контролю за будинком. Такі системи можуть надсилати сповіщення на телефон власника у разі виявлення незвичних активностей, що дозволяє оперативно реагувати на потенційні загрози.

Енергоефективність розумного будинку є ще однією значною перевагою. Завдяки використанню датчиків та інтелектуальних систем керування, можна значно зменшити споживання енергії. Наприклад, система може автоматично вимикати освітлення та електроприлади в кімнатах, де нікого немає, або регулювати температуру в приміщеннях залежно від наявності мешканців. Це не лише знижує витрати на енергію, але й сприяє збереженню довкілля.

Розумний будинок також відкриває нові можливості для розваг та комунікацій. Інтеграція з мультимедійними системами дозволяє створювати домашні кінотеатри, слухати музику у різних кімнатах або керувати іграми за допомогою голосових команд. Крім того, розумні дзвінки та відеоконференції можуть зробити спілкування з близькими ще зручнішим.

Отже, розумний будинок – це не лише технологічний тренд, а й значний крок у напрямку створення більш комфортного, безпечного та енергоефективного житла. Впровадження таких технологій у побут може кардинально змінити наше життя на краще, зробивши його більш зручним та гармонійним.

Основним компонентом розумного будинку є його здатність автоматизувати різноманітні процеси. Це досягається завдяки інтелектуальним системам, які взаємодіють між собою та з користувачем через Інтернет. Наприклад, системи освітлення можуть автоматично регулювати яскравість в залежності від часу доби та природного освітлення, а клімат-контроль підтримує оптимальну температуру в приміщеннях. Ці функції не тільки забезпечують комфорт, але й значно знижують витрати на електроенергію.

Безпека також є важливою складовою розумного будинку. Системи відеоспостереження, датчики руху та тривоги допомагають вчасно виявити небезпеку і забезпечують спокій господарів. Крім того, деякі системи дозволяють дистанційно керувати замками дверей, забезпечуючи додатковий рівень захисту.

Ще однією важливою перевагою розумного будинку є його інтеграція з мобільними пристроями. Користувач може керувати всіма системами свого дому через смартфон або планшет, що робить життя зручнішим і більш гнучким. Відстеження споживання енергії, автоматичне керування домашніми приладами та навіть управління системою поливу на дачі – все це стає можливим завдяки сучасним технологіям.

Не менш важливим є аспект екологічності. Розумні будинки сприяють зменшенню енергоспоживання та зниженню викидів вуглекислого газу. Автоматизація процесів, оптимізація використання ресурсів і використання відновлювальних джерел енергії – все це робить розумний будинок не лише комфортним, але й екологічно чистим.

Таким чином, розумний будинок – це не лише сучасний комфорт, а й крок до сталого розвитку. Він відкриває нові горизонти для зручності, безпеки та екологічності, роблячи наше життя не лише більш комфортним, а й більш свідомим. У світі, де технології невпинно рухаються вперед, розумний будинок стає не просто бажаним, а необхідним елементом нашого повсякденного життя.

Першим розумним будинком у світі став "Будинок трону", спроектований японським професором Кеном Сакамурою в Токіо наприкінці 1980-х років. Датчики погоди відкривали вікна, коли на вулиці був свіжий вітер, і вмикали кондиціонер у спеку; якщо радіо грало занадто голосно, вікна автоматично зачинялися, щоб не заважати сусідам; якщо надходив дзвінок на телефон, комп'ютер знижував гучність аудіосистеми тощо.

Основою "розумного" керування інженерними системами та системами життєзабезпечення є технологія, яка забезпечує інтелектуальне керування, регулювання і гнучку оптимізацію цих систем:

- керування опаленням: керування системами опалення різних типів, економія витрат на опалення;
- водопостачання: контроль протікання води у всіх приміщеннях, керування системами водо підготовки;
- газопостачання: контроль витікання газу;
- охорона: контроль проникнення у приміщення, периметральний контроль, імітація присутності людей,

імітація присутності тварин, обмеження доступу, відеонагляд;

- пожежна безпека: контроль займання;
- керування електроенергією: контроль електромереж, економія затрат;
- керування мікрокліматом у приміщенні: керування температурою, відносною вологістю повітря, контроль вмісту небезпечних газів;
- оповіщення: голосове оповіщення, дзвінок на мобільні та міські номери, sms, Internet;
- освітлення: зв‘язок із присутністю людей, керування за часом, сценарне керування, сценарії зовнішнього освітлення;
- будильник: голос, телефон, мелодії;
- голосовий пейджер: запис та відтворення повідомлень;
- басейн: керування нагрівом, керування фільтрацією;
- полив: автоматичний полив;
- жалюзі: автоматичне відкриття-закриття жалюзі та воріт, сценарне керування жалюзі;
- керування побутовими пристроями: керування усіма пристроями з одного пульта дистанційного керування, зв‘язок між приміщеннями, вхідний дзвінок з широким вибором мелодій.

2.2 Технічні засоби для реалізації системи «Розумний будинок»

Для реалізації системи «Розумний дім» використовуються різноманітні технічні засоби: давачі, контролери, пульти та панелі керування, домофони, GSM модеми, виконавчі механізми і регулюючі органи. Для реалізації функцій керування використовуються прості за набором функцій програмовані контролери із контролерного ряду тієї чи іншої фірми-виробника: Zelio Logic (Schneider Electric), Logo (Siemens), Mitsubishi (Mitsubishi Electric), Forth Logic (Електросвіт) та інші. Розглянемо детальніше програмований контролер Zelio Logic, на базі якого і будемо реалізовувати систему керування адміністративною будівлею.

2.2.1 Будова контролера Zelio Logic2

Загальний вигляд контролера Zelio Logic2 представлено на рис. 1.1.

Рис. 1.1 Загальний вигляд контролера Zelio Logic2

Передня панель

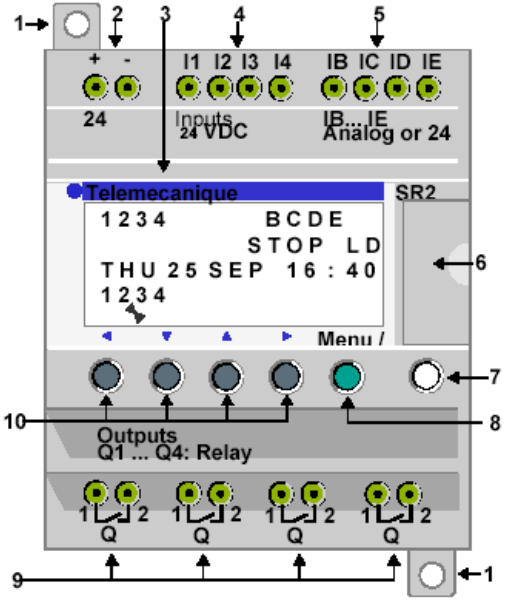


Рис. 1.2 Вигляд передньої панелі контролера Zelio Logic2

Таблиця 1.1

Позиція	Опис
1	Висувна монтажна ніжка
2	Гвинтові клеми для підключення живлення
3	Рідкокристалічний дисплей (4 рядки, 18 символів у кожному)
4	Гвинтові клеми для дискретних входів
5	Гвинтові клеми для аналогових входів (0...10 В), які можуть використовуватися як дискретні у деяких моделях
6	Роз‘єм для резервної пам‘яті або кабеля під‘єднання ПК
7	Клавіша Shift
8	Клавіша вибору і підтвердження
9	Гвинтові клеми дискретних виходів
10	Клавіши-стрілки (або сконфігуровані Z-клавіши)

Дисплей

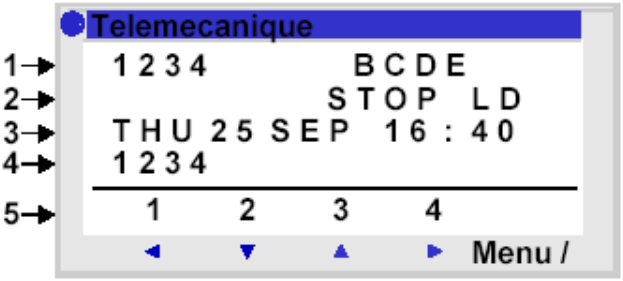


Рис. 1.3 Дисплей контролера Zelio Logic2

Таблиця 1.2

Позиція	Опис
1	Відображення стану входів (B...E відображають аналогові входи)
2	Відображення режиму роботи (RUN / STOP) та режиму програмування (LD / FBD)
3	Відображення дати (число і час для пристроїв, що підтримують таку можливість)
4	Відображення стану виходів

5	Контекстне меню / кнопки швидкого доступу / іконки, які відображають режим роботи

2.2.2 Характеристики та підключення

Підключення контролера постійного струму

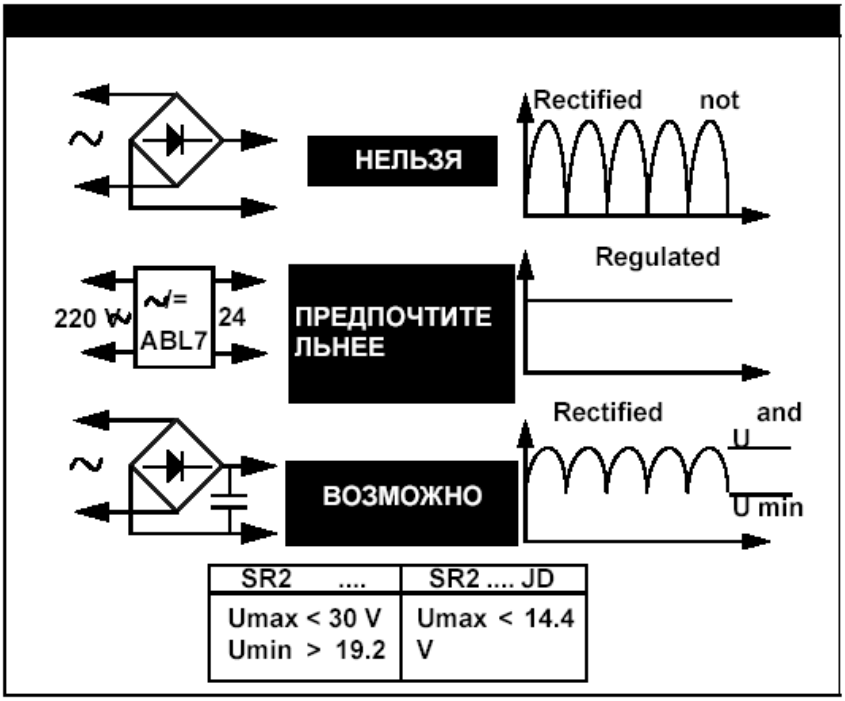


Рис. 1.4 Правила підключення контролера постійного струму

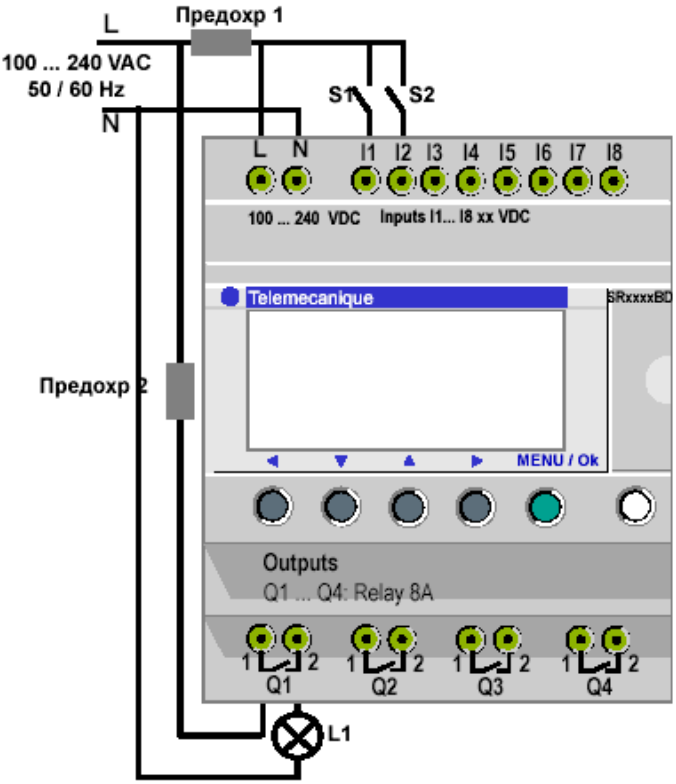


Рис. 1.5 Приклад підключення 2-х вимикачів до входів I1 та I2 та лампочки до виходу Q1 контролера

2.2.3 Меню

Головне меню

Таблиця 1.3

№	Назва	Призначення
1	**Programming**	Введення драбинкових діаграм.
2	**Parameters**	Введення та зміна параметрів елементів програми.
3	**Monitoring**	Доступний тільки в режимі LD / RUN. Даний режим використовується для динамічного відображення стану входів-виходів контролера. Користувач може динамічно змінювати значення параметів функцій, якщо

		вони розблоковані (клавіши Shift + Param).
4	**RUN / STOP**	Запуск чи зупинка програми. Коли здійснюється перемикання від STOP до RUN, програма ініціалізується.
5	**Configuration**	Меню конфігурації.
6	**Clear Program**	Видалення програми.
7	**Transfer**	Передача програми у модуль резервної пам‘яті або навпаки.
8	**Language Menu**	Меню вибору мови.
9	**Version**	Визначення версії системних компонентів: вбудованого програмного забезпечення, FBD-функції, LD-функцій.
10	**Fault**	Меню помилок, яке дозволяє відображати номари помилок або попереджень, виявлених програмним забезпеченням контролера.

Меню конфігурації – Configuration

Таблиця 1.4

№	Назва	Призначення
1	**Password**	Меню введення пароля.
2	**Filter**	Вибір швидкості визначення зміни стану всіх дискретних входів.
3	**Zx-Keys**	Дозволяє користувачу дозволити або заборонити використання клавіш курсору у якості кнопок. Доступно тільки в режимі LD.
4	**Change D/T**	Задання дати та часу для модулів з годинником.
5	**Change Summ/Wint**	Автоматична зміна часового діапазону літо-зима для модуля з годинником.
6	**Watchdog Cycle**	Задання тривалості циклу виконання програми.

2.2.4 Програмування контролера Zelio Logic2 мовами драбинкових діаграм (LD) та FBD-блоків

Контролер Zelio Logic2 можна програмувати за допомогою командних клавіш (мова LD) та за допомогою спеціалізованого програмного забезпечення Zelio Soft2 (мови LD та FBD).

Розглянемо правила програмування контролера Zelio Logic2 за допомогою спеціалізованого програмного забезпечення Zelio Soft2.

2.2.4.1 Створення нової програми

Для того, щоб створити нову програму необхідно виконати наступні дії.

1. Вибрати меню File/New або натиснути піктограму Create a new program під час запуску програми Zelio Soft2. В результаті з‘явиться вікно вибору типу модуля.

2. Виберіть потрібний модуль за допомогою ЛК миші. Модулі згруповані за такими ознаками:

- кількість входів-виходів;
- наявність або відсутність дисплею;
- можливість під‘єднання шасі розширення.

В результаті здійснення вибору типу модуля з‘являється весь список модулів даного типу у вигляді таблиці.

3. Виберіть модуль подвійним натисканням миші або натисніть клавішу Далее (Next). На цьому етапі може виникнути такі ситуації:

- модуль не підтримує можливості розширення та програмується лише мовою LD – перейдіть до кроку 7;
- модуль не підтримує можливості розширення і програмується мовами LD та FBD – перейдіть до кроку 6;
- модуль підтримує розширення, тоді на екрані з‘явиться інформація про тип модуля, вибраний у попередньому пункті та типи можливих розширень. Натиснувши кнопку Add, можна вибрати необхідне розширення. Для переходу до наступного кроку натисніть кнопку Далее. Модуль розширення при необхідності можна видалити кнопкою Delete.

4. Виберіть необхідний модуль розширення.

5. Підтвердіть вибір (кнопка Далее). В результаті з‘явиться вікно вибору мови програмування.

6.Виберіть потрібну мову (LD або FBD). Натисніть кнопку Далее (Next).

7. З‘явиться вікно редагування програми.

2.2.4.2 Мова драбинкових діаграм (Ladder Diagram Language)

Загальні положення

Програма, написана мовою LD, складається з мереж з‘єднаних між собою елементів. Виконання програми відбувається згідно географічного розміщення елементів зверху донизу і зліва направо.

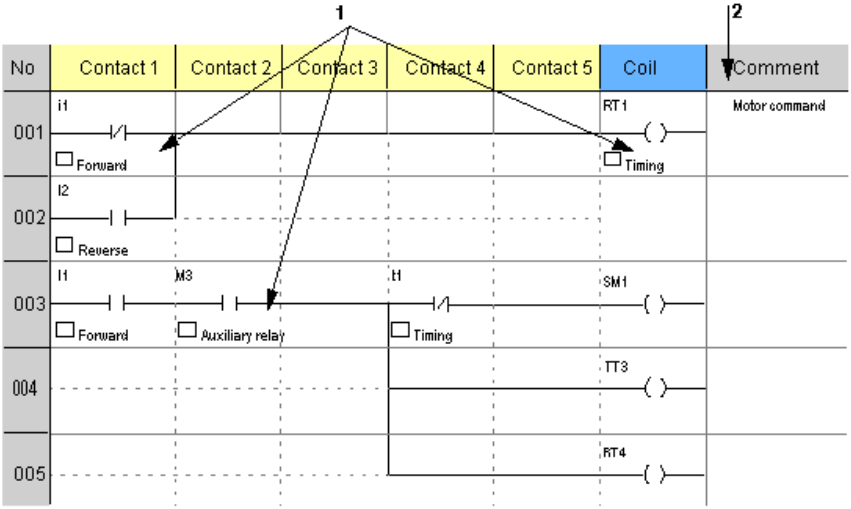


Рис. 1.6 Приклад програми, написаної мовою LD

Таблиця 1.5

№	Елемент	Функція
1	Графічні елементи	- Входи-виходи контролера (кнопки, давачі, реле, світлові індикатори тощо). - Контакти функціональних блоків (таймерів, лічильників тощо). - Логічні операції. - Внутрішні змінні (допоміжні реле) контролера.
2	Коментар	Для кожної лінії драбинкової діаграми або конкретного графічного елемента (є необов‘язковим).

Вікно створення програми поділяється на декілька частин (рис. 1.7).

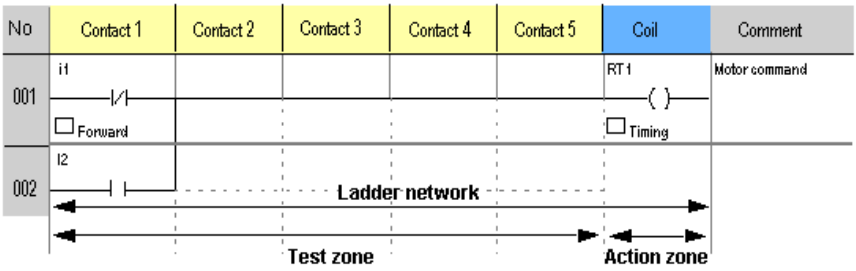


Рис. 1.7 Структура вікна створення програми мовою LD

Кожен рядок LD-програми складається з набору графічних елементів, які розміщені у прямокутниках сітки:

- максимальна кількість рядків – 120;
- кожен рядок може містити до 5 контактів і 1 котушку.

LD-діаграма поділена на 2 зони:

- test zone, в якій розміщуються контакти;
- action zone, в якій розміщуються котушки (відображає результат операцій).

Графічні елементи у мові LD

1. Контакти

Дані елементи розміщуються у test zone і займають 1 чарунку.

Таблиця 1.6

Назва	Графічне представ-лення (LD)	Графічне представ-лення (Electrical)	Функції
Нормально відкритий контакт			Замикається, коли асоційований з ним елемент (вимикач, давач), активний.
Нормально замкнутий контакт			Розмикається, коли асоційований з ним елемент (вимикач, давач), активний.

2. Елементи зв‘язку

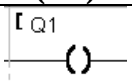
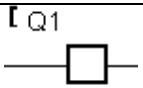
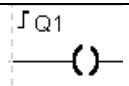
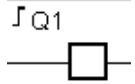
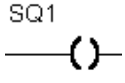
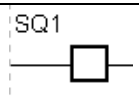
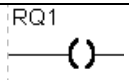
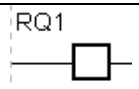
Використовуються для зв‘язку між елементами, що розміщені у test-зоні або action-зоні.

Таблиця 1.7

Назва	Графічне представ-лення	Функції
Горизонтальний зв‘язок		Використовується для зв‘язку між послідовно розміщеними елементами.
Вертикальний зв‘язок		Використовується для зв‘язку між паралельно розміщеними елементами.

3. Котушки

Таблиця 1.8

Назва	Графічне представ-лення (LD)	Графічне представ-лення (Electrical)	Функції
Пряма котушка			На котушку подається живлення, якщо контакти, до яких вона під‘єднана, активні.
Імпульсна котушка			На котушку подається живлення, якщо контакти, до яких вона під‘єднана, змінюють свій стан.
Замкнена котушка (Set)			На котушку подається живлення, якщо контакти, до яких вона під‘єднана, активні. ЇЇ стан не зміниться навіть якщо її контакти розімкнуться.
Розімкнена котушка (Reset)			Котушку стає неактивною, якщо контакти, до яких вона під‘єднана, активні. Вона зберігає свій стан незалежно

			від подальшої поведінки її контактів.

Увага! Котушки Set і Reset повинні використовуватися разом. Функція Reset має пріоритет перед функцією Set. Кожен з виходів Q повинен використовуватися у програмі тільки 1 раз, крім котушок Set і Reset.

Етапи створення LD-програми

1. Складіть список входів-виходів і присвойте їм коментарі.
2. Складіть список необхідних керуючих функцій (підрахунок кількості машин, дискретне керування).
3. Реалізуйте кожну функцію за такою схемою.

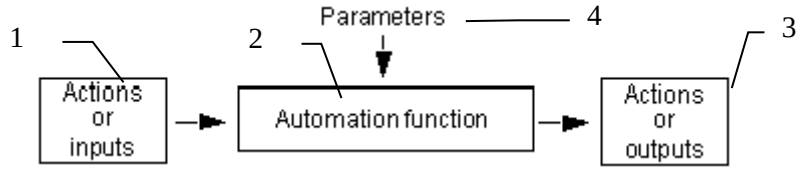


Рис. 1.8 Схема реалізації керуючої функції

1 – керуючі дії або входи;
2 – керуюча функція;
3 – керуючі дії або виходи;
4 – параметри.

4. Присвойте коментар кожній функції.
5. Протестуйте кожну функцію, використовуючи режим емуляції.

Режими програмування

1. Режим Zelio (Zelio Entry Mode)

Даний режим дозволяє створити програму мовою LD за допомогою емуляції використання командних кнопок на передній панелі контролера.
Приклад:

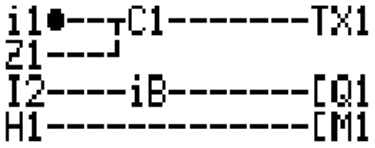


Рис. 1.9 Приклад LD-програми у режимі Zelio

Даний режим зручний тим, що надає програмісту можливість навчитися програмувати контролер через командні кнопки у середовищі Zelio Soft, не маючи самого контролера.

2. Вільний режим (Free Entry Mode)

Даний режим дозволяє засвоїти прийоми створення програм за допомогою самого середовища програмування без імітації передньої панелі контролера, а саме:

- використання панелей інструментів;
- створення програм за допомогою перетягування елементів у потрібне місце драбинкової діаграми;
- використання вікон параметрів елементів;
- можливість переглядати усю програму у цілому.

Приклад:

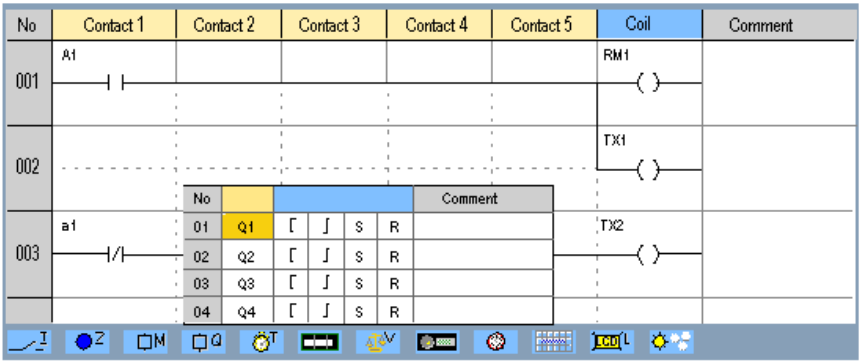


Рис. 1.10 Приклад LD-програми у вільному режимі

Даний режим надає іще ряд додаткових переваг:

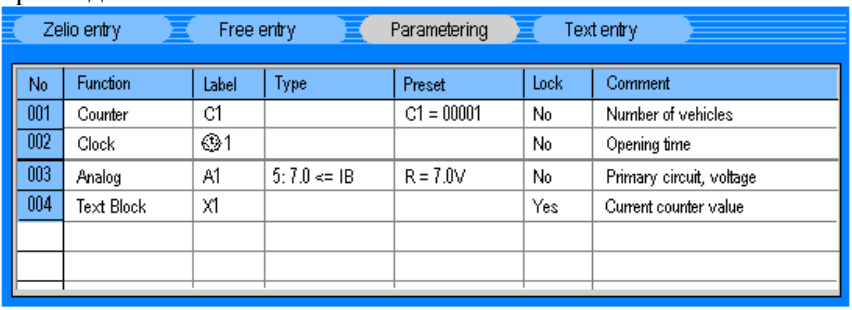
- вибір типу представлення символів (LD, електричні аналоги);
- можливість додати коментар до рядка.

3. Параметричний режим (Parameter Mode)

Даний режим дозволяє представити усі функції керування, які використовуються у програмі, з їх параметрами у вигляді списку, який має такі позиції:

- номер за порядком;
- функція: таймер, лічильник тощо;
- мітка, що визначає функціональний блок;
- тип;
- уставка;
- вказівник замикання;
- коментар.

Приклад:



No	Function	Label	Type	Preset	Lock	Comment
001	Counter	C1		C1 = 00001	No	Number of vehicles
002	Clock	1			No	Opening time
003	Analog	A1	5: 7.0 <= IB	R = 7.0V	No	Primary circuit, voltage
004	Text Block	X1			Yes	Current counter value

Рис. 1.11 Список функціональних блоків, використаних у LD-програмі, при використанні параметричного режиму

Параметри конкретного функціонального блоку можна змінити, якщо двічі натиснути мишкою на рядку, де він описаний.

4. Текстовий режим (Text Entry Mode)

Даний режим дозволяє представити усі входи-виходи, які використані у програмі, у вигляді списку. Важливо присвоїти кожному входу та виходу зрозумілий коментар для того, щоб зробити програму зрозумілішою.

Приклад:

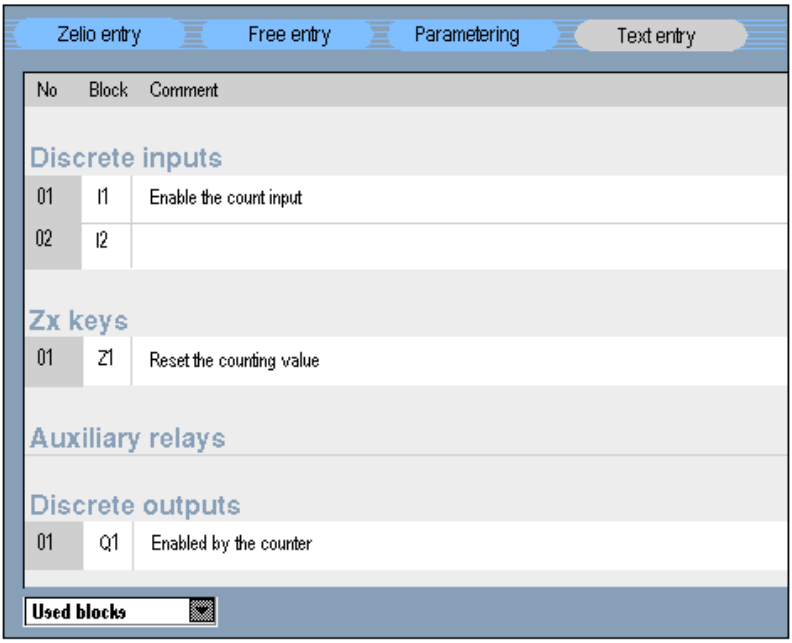


Рис. 1.12 Представлення входів та виходів у текстовому режимі

Введені коментарі будуть відображатися під графічними зображеннями елементів у драбинковій діаграмі (рис. 1.7).

Елементи мови драбинкових діаграм (LD)

Таблиця 1.9

№	Піктограма у Zelio Soft	Назва
1		Дискретні входи (Discrete Inputs)
2		Дискретні виходи (Discrete Outputs)
3		Входи-виходи Modbus (Modbus Inputs-Outputs)
4		Допоміжні реле Auxiliary Relays)
5		Z-клавіши (Zx Keys)
6		Лічильники (Counters)

7		Функціональний блок порівняння лічильника (Counter Comparator)
8		Швидкий лічильник (Fast Counter)
9		Годинники (Clocks)
10		Функціональний блок переходу на зимовий та літній час (Change to Summer/Winter Time)
11		Таймери (Timers)
12		Аналогові компаратори (Analog Comparators)
13		Функціональний блок Текст (Texts)
14		Функціональний блок підсвітки екрану (LCD Screen Backlighting)

Автоматична перевірка LD-програми

Середовище програмування Zelio Soft2 дозволяє здійснювати емуляцію програм, їх покрокову перевірку та налагодження. Якщо знайдена якась помилка, піктограма  , що розташована у верхній частині вікна редагування програми, змінює свій колір з блакитного на червоний.

Приклад:

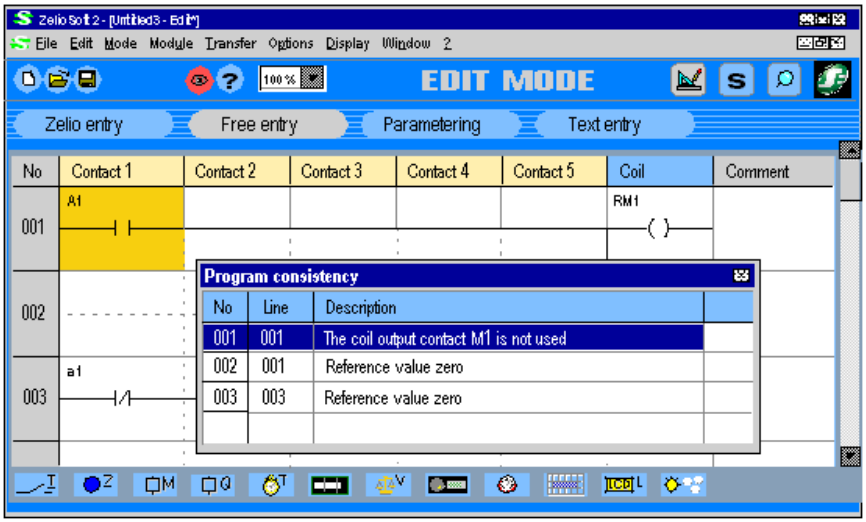


Рис. 1.13 LD-програма у режимі налагодження

Натиснувши на піктограму “Око”, можна відкрити діалогове вікно, яке містить наступну інформацію:

- номер помилки;
- місцезнаходження помилки: рядок або стовпчик;
- опис помилки.

Подвійне натискання мишки на назві помилки призводить до підсвітки конкретного елемента програми, де міститься помилка.

Увага! Навіть якщо символ “Око” став червоним, програма може бути запущена на емуляцію. Елементи, розміщені у програмі з допущенням помилок (відсутнє з‘єднання), будуть неактивні.

Емуляція та моніторинг LD-програми

Функція емуляції використовується для виконання програми безпосередньо в Zelio Soft2 (у реальному часі).
Приклад:

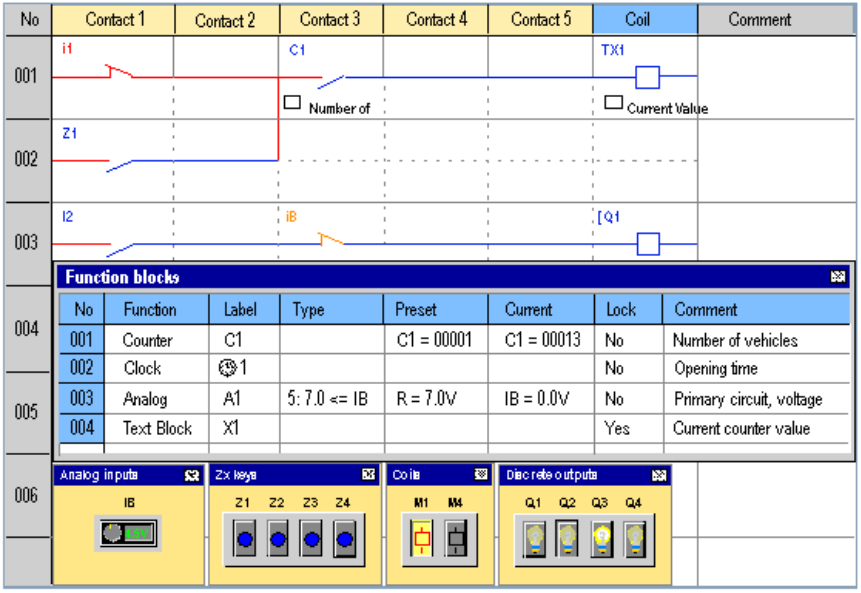


Рис. 1.14 LD-програма у режимі емуляції

Таблиця 1.10

Піктограма	Функція
	Перехід у режим емуляції
	Перехід у режим редагування
	Перехід у режим моніторингу
	Запуск програми на виконання
	Зупинка виконання програми
	Пуза/Запуск: зупиняє або перезапускає програму (активна тільки у режимі Run)
	Емуляція збою енергопостачання (активна тільки у режимі Run)

Функція моніторингу використовується для виконання програми в контролері (у режимі реального часу) і одночасної візуалізації в

Zelio Soft 2. Потрібно, щоб на комп‘ютері і в контролері була запущена та сама програма. У даному режимі користувач може:

- відображати значення різних параметрів у вікні функціональних блоків;
- натиснути на блок, щоб змінити його параметри.

3. Програма роботи

1. Вивчити будову та характеристики контролера Zelio Logic 2.
2. Дослідити правила підключення контролера Zelio Logic 2.
3. Навчитися користуватися меню контролера Zelio Logic 2
4. Навчитися програмувати контролер Zelio Logic 2 мовою драбинкових діаграм (LD) за допомогою командних клавіш передньої панелі.
5. Навчитися програмувати контролер Zelio Logic 2 мовою драбинкових діаграм (LD) у середовищі Zelio Soft 2.
6. Навчитися записувати програми у контролер.
7. Виконати завдання, поставлені викладачем.

4. Опис лабораторного обладнання

1. Персональний комп‘ютер.
2. Операційна система Windows.
3. Лабораторний стенд з логічним контролером Zelio Logic.
4. Середовища програмування Zelio Soft, Zelio Soft 2.

5. Порядок виконання роботи

1. Напишіть програму для програмованого логічного контролера (ПЛК) Zelio Logic мовою драбинкових діаграм (LD) для автоматизації роботи підземної автостоянки. При створенні програми виберіть модель контролера SR1B122BD Version 1.6, середовище для програмування – Zelio Soft. Виконайте дії, описані у п.2.2.4.1 теоретичних відомостей.
2. Проведіть емуляцію та налагодження програми у середовищі програмування Zelio Soft. Для цього скористайтеся меню Simulation та натисніть кнопку Run. Після проведення перевірки роботи програми натисніть кнопку Stop.

3. Запишіть створену програму у ПЛК та дослідіть її роботу. Для цього необхідно виконати наступні дії:

- подати живлення на стенд;
- перевести контролер у режим STOP;
- ввійти у головне меню, натиснувши клавішу Sel./OK;
- вибрати пункт меню Transfer.\ PC -> Module;
- у середовищі програмування Zelio Soft вибрати пункт меню Transfer\ Transfer Program\ PC -> Module.

4. Навчиться керувати запуском та зупинкою програми з клавіатури контролера та з комп‘ютера. Для керування виконанням програми з клавіатури контролера скористайтеся пунктом меню Run\ Stop, з комп‘ютера – пунктами меню середовища Zelio Soft Transfer\ Run Module, Transfer\ Stop Module.
5. Для відслідковування роботи програми на комп‘ютері виберіть пункт меню Mode\ On Line Monitoring Mode.
6. Складіть пояснюючу таблицю використаних входів і виходів ПЛК, в якій зазначте назву входу-виходу, тип, призначення.

<u>Вимоги до програми:</u>

- Обрахунок кількості машин у гаражі, якщо в‘їзд та виїзд з гаража ізольовані. При заповненні автостоянки передбачити світлову сигналізацію. При цьому система повинна закрити в‘їзні ворота.
- Передбачити також ручний режим роботи стоянки (обрахунок кількості машин у гаражі, блокування сигналу на закриття вхідних воріт, коли необхідно пропустити спеціальні служби (пожежних, швидку допомогу тощо)).
- Часова затримки при відкритті та закритті воріт у гаражі, так щоб машина встигла проїхати.
- Заборонити доступ на стоянку у неробочі години і дозволити охороні відключати цю функцію у разі необхідності. Стоянка працює з понеділка по п‘ятницю з 8:00 до 19:00, в суботу з 9:00 до 17:00 і повністю зачинена у неділю.
- З метою безпеки необхідно вилучати токсичні викиди, такі як CO_2, за допомогою вентилятора, коли рівень їх

концентрації перевищує допустимий (рівень концентрації викидів визначається за допомогою спеціального давача, який забезпечує вихідну величину від 0 до 10 В).

- Світло у підземному гаражі повинно автоматично вмикатися по прибуттю автомобіля або за допомогою 2-х кнопочних перемикачів, розташованих біля пішохідного переходу (одна для ескалатора і одна для сходів; через в‘їзні ворота пішохідний доступ закритий). З метою економії електроенергії світло повинно автоматично вимикатися через 10 хвилин, за цей час клієнт, як правило, встигає припаркуватися і піднятися ескалатором або взяти машину і залишити автостоянку.

Аналіз задачі:

Таблиця 1.11

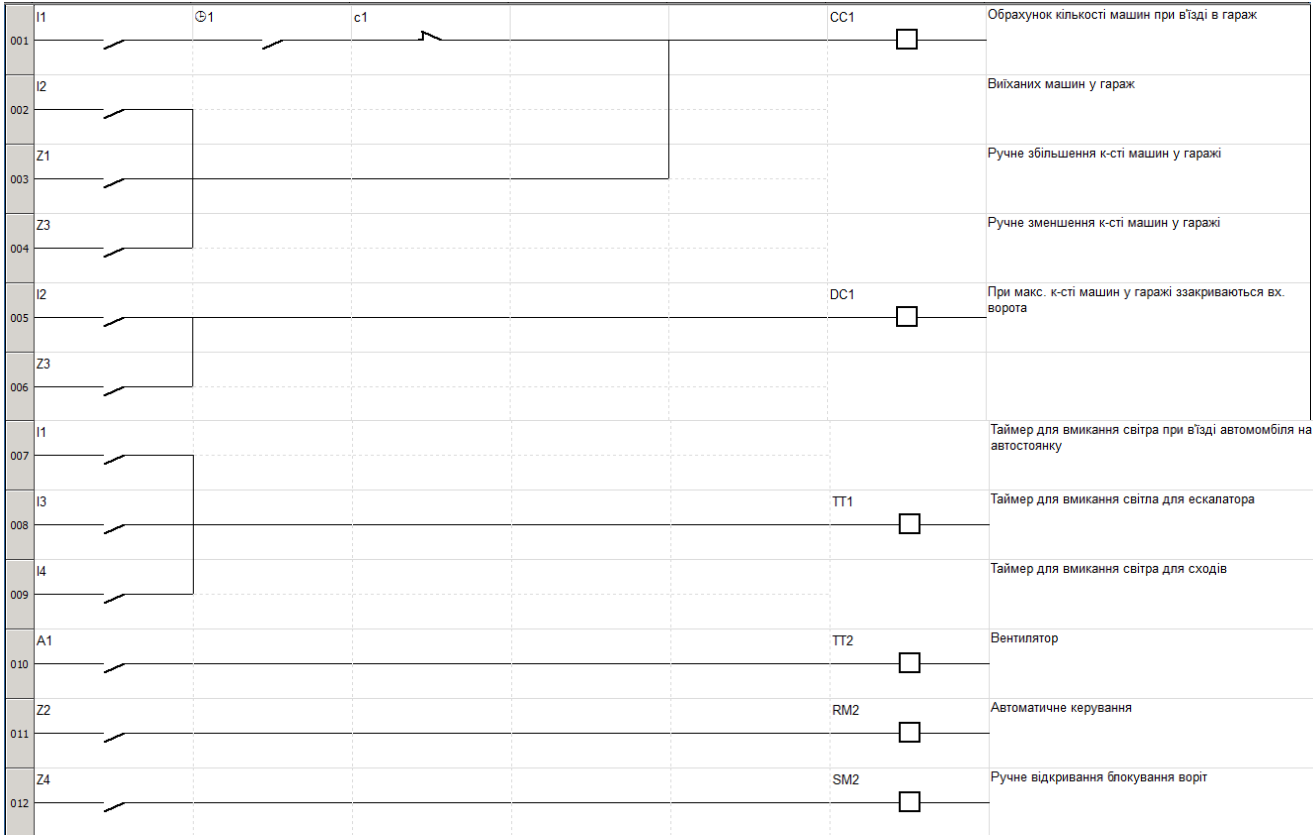
Позначення реле	*Опис*
Вхід I1	Виявлення в‘їзду машини
Вхід I2	Виявлення виїзду машини
Лічильник C1	Підрахунок кількості машин на автостоянці
Вихід Q1	Індикація заповненості стоянки
Вихід Q2	Блокування бор‘єру (заборона відкриття бор‘єру)
Функціональна клавіша Z4	Ручне блокування бар‘єру
Функціональна клавіша Z2	Відновлює автоматичне керування в‘їздом
Функціональна клавіша Z1	Ручне збільшення кількості машин на стоянці
Функціональна клавіша Z3	Ручне зменшення кількості машин на стоянці
Функціональний блок Годинник 1	Обробляє години доступу
Входи I3 та I4	Кнопки в точках доступу для пішоходів для включення освітлення автостоянки. Один для підйомника і один для сходів (для пішоходів доступу через в‘їзд для машин немає)

Вихід Q3	Керування світлом
Функціональний блок Таймер 1	Таймер для світла (10 хв)
Аналоговий вхід IB	Давач рівня CO_2
Аналоговий функціональний блок A1, порогове значення відповідає 8.5 В	Порівнює значення, видане давачем, зі значенням уставки
Вихід Q4	Керує вентилятором продувки забрудненого повітря
Функціональний блок Таймер 2	Таймер вентилятора (15 хв)

Зверніть увагу, що для реалізації даної задачі необхідний блок Zelio з аналоговими входами, годинником, мінімум 4-ма дискретними входами, мінімум 4-ма дискретними виходами.

Створення програми мовою LD:

Зверніть увагу, що котушки CC і DC повинні з‘являтися у програмі лише один раз для одного і того ж лічильника.


001
I1
Ⓣ1
c1
CC1
Обрахунок кількості машин при в'їзді в гараж
002
I2
Виїханих машин у гараж
003
Z1
Ручне збільшення к-сті машин у гаражі
004
Z3
Ручне зменшення к-сті машин у гаражі
005
I2
DC1
При макс. к-сті машин у гаражі зсакриваються вх. ворота
006
Z3
007
I1
Таймер для вмикання світра при в'їзді автомомбіля на автостоянку
008
I3
TT1
Таймер для вмикання світла для ескалатора
009
I4
Таймер для вмикання світра для сходів
010
A1
TT2
Вентилятор
011
Z2
RM2
Автоматичне керування
012
Z4
SM2
Ручне відкривання блокування воріт

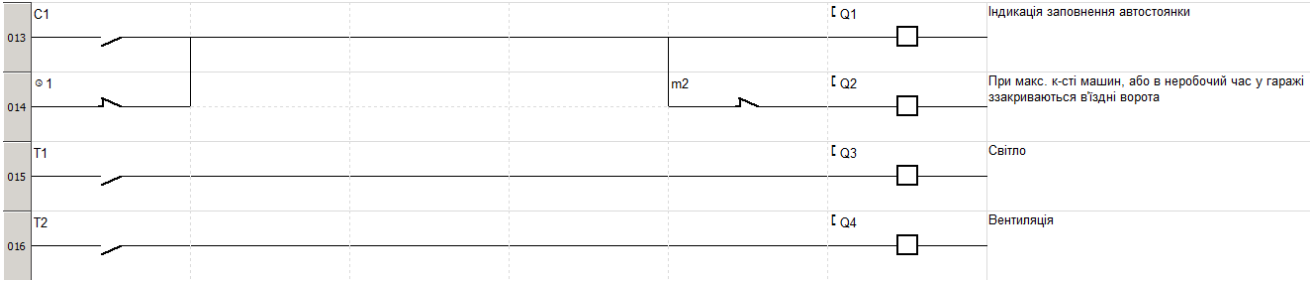
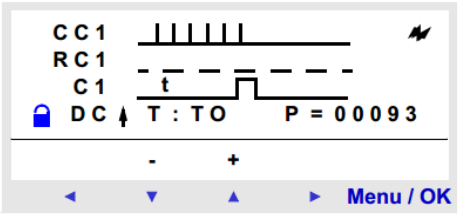
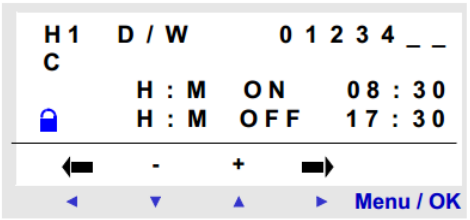
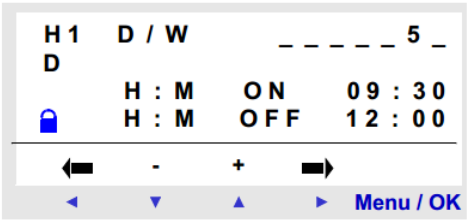
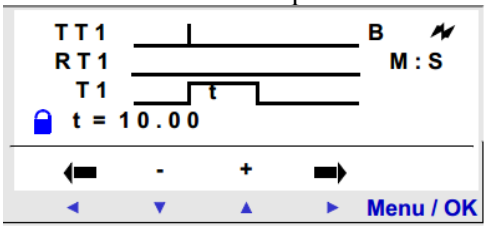
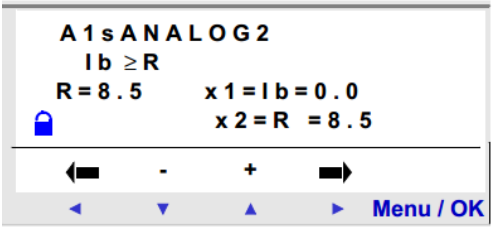
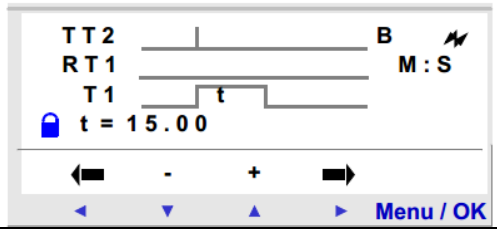


Рис. 1.15 Програма мовою LD

Конфігурування функціональних блоків:

Таблиця 1.12

Функціональний блок	*Опис*
ФБ Лічильник 1 	Значення уставки = 93 (Максимальне число транспортних засобів на автостоянці. Для проведення емуляції програми це значення доцільно зменшити).
ФБ Годинник 1  	Години роботи: З понеділка по п‘ятницю з 8:30 до 17:30. В суботу з 9:30 до 12:00. Неділя зачинено. Використовуються 2 інтервали.
ФБ Таймер 1 	Таймер освітлення: 10 хв.

Аналоговий ФБ А1 	Порівнює виміряний рівень CO_2 з уставкою: 8.5 В.
ФБ Таймер 2 	Тривалість роботи вентилятора при підвищенні порогу по CO_2: 15 хв.

6. Контрольні запитання

1. Яким чином можна охарактеризувати системи контролю та керування типу «Розумний дім»?
2. Яка будова контролера Zelio Logic2?
3. Які основні характеристики контролера Zelio Logic2?
4. Які правила підключення Zelio Logic2?
5. Назвіть пункти та функції головного меню контролера Zelio Logic2.
6. Назвіть пункти та функції меню конфігурації контролера Zelio Logic2.
7. Яким чином запрограмувати контролер мовою драбинкових діаграм (Ladder Diagram Language)?
8. Назвіть основні графічні елементи мови драбинкових діаграм.
9. Яким чином записати програму у ПЛК з комп‘ютера? Яким чином зчитати записану програму з ПЛК на комп‘ютер?
10. Як можна відслідкувати роботу програми з комп‘ютера у реальному часі?

Робота №2, №3. Розробка і дослідження системи контролю та керування температурою повітря і вологості ґрунту в теплиці на базі програмованого логічного контролера та її випробування мовами LD та FBD. Розробка екрану супервізора.

1. Мета роботи

Навчитися розробляти системи контролю та керування температурою повітря і вологості ґрунту в теплиці на базі програмованого логічного контролера. Навчитися програмувати контролер Zelio Logic2 мовами драбинкових діаграм (LD - Ladder Diagram) та функціональних блоків (FBD – Function Block Diagram). Навчитися розробляти дворівневі системи контролю та керування на базі Zelio Logic2.

2. Теоретичні відомості

2.1.1 Фактори, що впливають на розвиток та врожайність сільськогосподарських культур

Вода – один з елементів родючості ґрунту. У період росту рослини споживають велику кількість води, з якої 0,01-0,03 % іде на утворення рослинних тканин, решта витрачається на транспірацію (випаровування) листям і стеблами рослин. Сільськогосподарські культури пред‘являють певні вимоги до водного режиму ґрунту. Максимальна врожайність досягається лише при оптимальній кількості вологи, живлення, тепла, повітря і світла. При нестачі або надлишку вологи в ґрунті врожайність культур знижується. У першому випадку рослини страждають від нестачі вологи і поживних речовин, у другому – від нестачі повітря у ґрунті. Однією із основних умов, які визначають потребу рослин у воді, являється тривалість критичних фаз розвитку рослин. Розрізняють 4 фази, в яких рослини найбільш чутливі до нестачі вологи в ґрунті: проростання, кущіння, цвітіння і дозрівання. Необхідний для сільськогосподарських культур водний режим ґрунту створюється відповідним режимом зрошення, який встановлює норми, терміни і кількості поливів в залежності від біологічних особливостей культур, природних і господарських умов.

При визначенні витрат води на зрошення сільськогосподарської культури враховують транспірацію води через листя та стебла

рослин, випаровування з поверхні ґрунту. Це сумарне випаровування називають водоспоживанням, або евапотранспірацією. Водоспоживання залежить від кліматичних умов, кількості теплової енергії, яка надходить на поверхню, вологості ґрунту, виду та врожайності культури.

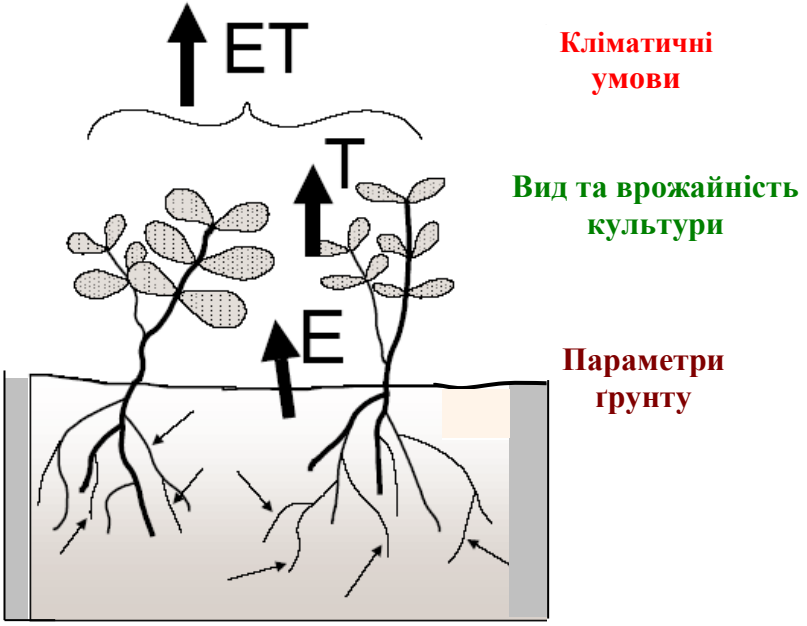


Рис. 2.1. Фактори, що впливають на евапотранспірацію

2.1.2 Керування мікрокліматом та вологістю грунту у теплицях: сучасні підходи та технології

У сучасному світі сільське господарство стрімко розвивається, і однією з важливих його складових є теплиці. Вони дозволяють забезпечувати стабільні урожаї незалежно від погодних умов, створюючи ідеальні умови для росту рослин. Ключовим фактором успішного вирощування культур у теплицях є ефективне керування мікрокліматом та вологістю ґрунту.

Мікроклімат у теплиці включає в себе кілька параметрів: температуру, вологість повітря, освітлення та концентрацію вуглекислого газу. Від цих показників залежить, наскільки

комфортно буде рослинам рости та розвиватися. Для досягнення оптимальних умов використовується низка технологій.

Температурний режим у теплицях регулюється за допомогою систем опалення та вентиляції. Сучасні системи опалення дозволяють підтримувати необхідну температуру в холодну пору року, а системи вентиляції забезпечують видалення надлишкового тепла та вологи влітку. Використання автоматичних термостатів та датчиків температури дає змогу підтримувати постійний температурний режим без участі людини.

Вологість повітря є не менш важливим показником. Надмірна вологість може призвести до розвитку грибкових захворювань, тоді як недостатня вологість негативно впливає на ріст рослин. Для контролю цього параметра використовуються зволожувачі повітря та осушувачі. Крім того, автоматичні системи зрошення можуть регулювати рівень вологості залежно від потреб рослин.

Освітлення у теплицях також грає важливу роль, особливо в зимовий період, коли природного світла недостатньо. Використання штучного освітлення, зокрема LED-ламп, дозволяє забезпечити рослинам необхідний рівень освітлення, стимулюючи їх ріст та фотосинтез.

Керування концентрацією вуглекислого газу у теплиці сприяє підвищенню врожайності. Вуглекислий газ є необхідним для процесу фотосинтезу, тому підтримання його оптимального рівня сприяє активнішому росту рослин. Це досягається за допомогою спеціальних систем, які подають CO2 у теплицю.

Що стосується вологості ґрунту, то вона є ключовим фактором для здорового росту рослин. Недостатнє або надмірне зволоження може негативно вплинути на врожайність. Для контролю вологості ґрунту використовуються різні системи зрошення: крапельне, дощувальне та підземне зрошення. Крапельне зрошення є найбільш ефективним, оскільки воно дозволяє доставляти воду безпосередньо до кореневої системи рослин, мінімізуючи її втрати та забезпечуючи рівномірне зволоження.

Сучасні технології дозволяють автоматизувати процеси керування мікрокліматом та вологістю ґрунту. Використання датчиків та автоматичних систем керування дозволяє створювати оптимальні умови для вирощування різних культур, забезпечуючи їх стабільний ріст та високу врожайність. Такі системи можуть

працювати в автономному режимі, аналізуючи дані з датчиків та автоматично вносячи необхідні корективи.

Важливу роль у цьому процесі відіграють комп'ютерні програми та мобільні додатки, які дозволяють фермерам віддалено контролювати всі параметри мікроклімату та вологості ґрунту. Завдяки цьому, керування теплицею стає більш зручним та ефективним.

Використання сучасних технологій для керування мікрокліматом та вологістю ґрунту у теплицях є важливим кроком у напрямку підвищення ефективності сільського господарства. Це дозволяє не тільки забезпечити високі врожаї, але й зменшити витрати на ресурси, такі як вода та енергія. У кінцевому результаті це сприяє підвищенню рентабельності тепличного господарства та збереженню навколишнього середовища.

Отже, керування мікрокліматом та вологістю ґрунту у теплицях є складним, але надзвичайно важливим завданням. Сучасні технології надають широкі можливості для автоматизації цього процесу, що дозволяє створювати оптимальні умови для вирощування різноманітних культур. Це не тільки підвищує врожайність, але й робить сільське господарство більш ефективним та екологічно чистим.

2.2 Програмований логічний контролер Zelio Logic для керування теплицею

Розробимо систему керування температурою та вологістю ґрунту у теплиці на базі програмованого контролера Zelio Logic фірми Schneider Electric. Будову, характеристики, правила підключення, меню програмованого контролера Zelio Logic та правила написання програм мовою драбинкових діаграм розглянуто у роботі 1. Тому розглянемо правила програмування контролера мовою функціональних блоків.

2.2.1 Програмування контролера Zelio Logic2 мовою FBD-блоків

Програми для контролера Zelio Logic2 мовою функціональних блоків (FBD – Function Block Diagram) можна писати, використовуючи спеціалізоване програмне забезпечення Zelio Soft2.

2.2.1.1 Створення нової програми

Для того, щоб створити нову програму необхідно виконати наступні дії.

1. Вибрати меню File/New або натиснути піктограму Create a new program під час запуску програми Zelio Soft2. В результаті з‘явиться вікно вибору типу модуля.

2. Виберіть потрібний модуль за допомогою ЛК миші. Модулі згруповані за такими ознаками:

- кількість входів-виходів;
- наявність або відсутність дисплею;
- можливість під‘єднання шасі розширення.

В результаті здійснення вибору типу модуля з‘являється весь список модулів даного типу у вигляді таблиці.

3. Виберіть модуль подвійним натисканням миші або натисніть клавішу Далее (Next). На цьому етапі може виникнути такі ситуації:

- модуль не підтримує можливості розширення та програмується лише мовою LD – перейдіть до кроку 7;
- модуль не підтримує можливості розширення і програмується мовами LD та FBD – перейдіть до кроку 6;
- модуль підтримує розширення, тоді на екрані з‘явиться інформація про тип модуля, вибраний у попередньому пункті та типи можливих розширень. Натиснувши кнопку Add, можна вибрати необхідне розширення. Для переходу до наступного кроку натисніть кнопку Далее. Модуль розширення при необхідності можна видалити кнопкою Delete.

4. Виберіть необхідний модуль розширення.

5. Підтвердіть вибір (кнопка Далее). В результаті з‘явиться вікно вибору мови програмування.

6.Виберіть потрібну мову (LD або FBD). Натисніть кнопку Далее (Next).

7. З‘явиться вікно редагування програми.

2.2.3 Мова FBD-блоків (Function Blocks Diagram Language)

При програмуванні мовою FBD-блоків можливе використання 2-х типів вікон:

- вікно редагування;
- вікно супервізора.

Вікно редагування

FBD-програми створюються у вікні редагування. Вікно редагування можна запустити за допомогою меню Mode/Edit або за допомогою піктограми  .

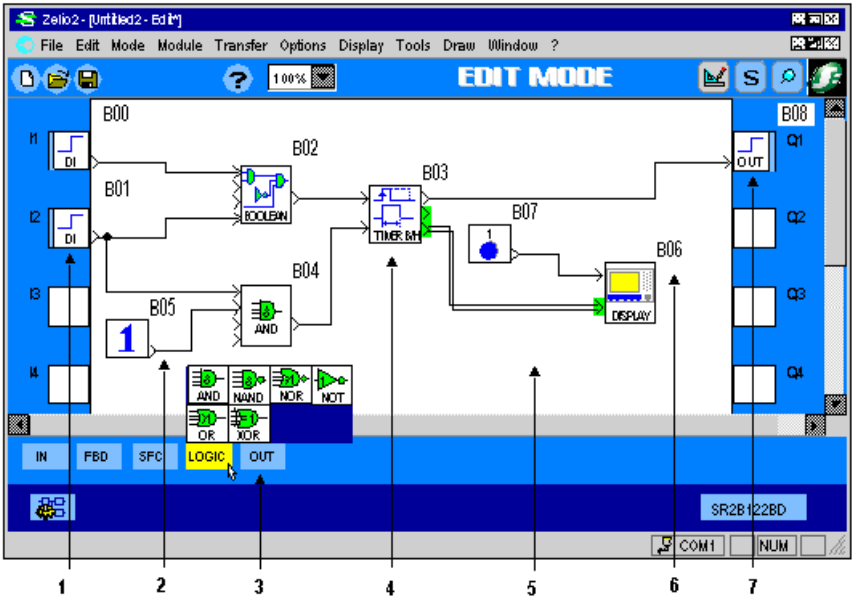


Рис. 2.2 Вікно редагування FBD-програми

Таблиця 2.1

Номер	Опис
1	Входи Zelio
2	З‘єднання між 2-ма функціональними блоками
3	Панель функціональних блоків
4	Функціональний блок
5	Вікно програми
6	Номер функціонального блоку
7	Виходи Zelio

Вікно супервізора

Вікно супервізора можна запустити наступним чином:

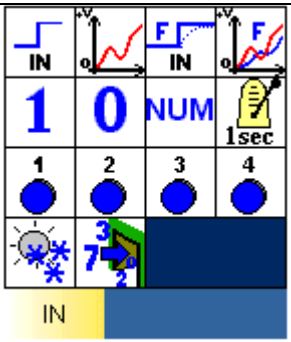
- **симуляція (simulation):** пункт меню mode/simulation або піктограма  ;
- **моніторинг (monitoring):** пункт меню mode/monitoring або піктограма  .

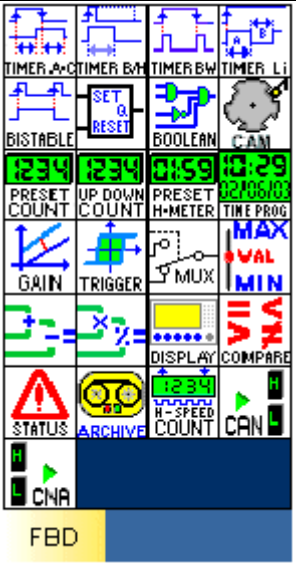
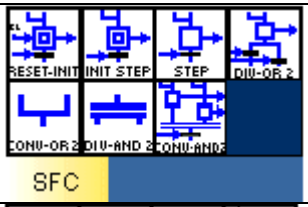
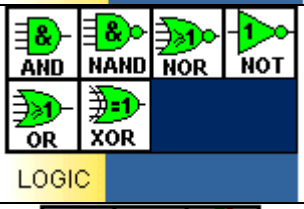
У вікно супервізора можна вставити частини FBD-програми, скопійовані у вікні редагування. У вікні супервізора можна малювати, вставляти текст і малюнки (*.bmp). Доступний також режим емуляції.

Панель елементів

Усі елементи, які необхідні для створення FBD-програми, розміщені на панелі елементів у нижній частині вікна редагування. Елементи розділені на функціональні групи.

Таблиця 2.2

1. Входи	

2. Стандартні функції	
3. SFC-функції	
4. Логічні функції	
5. Виходи	
6. Входи-виходи Modbus	

Елементи мови FBD-блоків

1. Входи

Дискретні входи (Discrete-type inputs) – 

Тип дискретного входу можна вибрати із вікна Параметри (Parameters).

Таблиця 2.3

Тип	Неактивний стан	Активний стан
Дискретний вхід (discrete input)		
Контакт (contact)		
Вимикач (limit switch)		
Давач наближення (proximity sensor)		
Давач присутності (presence sensor)		
Кнопка з під світкою (illuminated pushbutton)		
Перемикач (selector switch)		
Кнопка (pushbutton)		
Нормально відкрите реле (normally open relay)		

Дискретний вхід з фільтром (Filtered discrete-type input) – 

З метою зменшення або навіть ліквідації завад після дискретного входу встановлюють фільтр. Якщо сигнал стабільний на протязі усього визначеного періоду часу, вихід дискретного входу з фільтром приймає виміряне значення, в іншому випадку сигнал лишається незмінним.

Значення параметру (від 1 до 255), введене у вікно Параметри (Parameters), використовується для визначення мінімального часу, на протязі якого сигнал давача повинен бути стабільним. Даний параметр множиться на циклічний час модуля (Zelio).

У режимі симуляції або моніторингу при активації дискретного входу з фільтром він набуває такого вигляду – 

.

Аналоговий вхід (Analog-type input) – 

Аналоговий вхід доступний для всіх модулів постійного струму. Вхідна аналогова напруга конвертується у числове значення від 0 до 255 8-бітним АЦП. Аналоговими входами можуть бути входи від IB до IG модуля.

За замовчуванням значення аналогового входу лежить в межах 0 – 10 В постійного струму. Тип електричного з'єднання входу можна конфігурувати у вікні Параметри (Parameters). У цьому ж вікні можна вибрати тип аналогового входу.

Таблиця 2.4

Тип	Зображення
Вхід за замовчуванням (Input by default)	
Вхід (Input)	
Температура (Temperature)	

Потенціометр (Potentiometer)	

Аналоговий вхід з фільтром (Filtered analog input) – 
Робота такого входу аналогічна аналоговому з тією відмінністю, що після аналогового входу встановлений низькочастотний фільтр.

- Фільтр повністю відновлює вхідний сигнал (частоту, амплітуду та зсув фаз), якщо частота сигналу значно нижча за частоту зрізу фільтра.
- Коли частота вхідного сигналу наближається до частоти зрізу фільтра, вихідний сигнал тієї ж частоти стає меншим за амплітудою і зсунутим за фазою.
- Коли частота вхідного сигналу рівна частоті зрізу фільтра, вихідний сигнал знижується на 30% і зсувається за фазою на 45^0.
- Якщо частота вхідного сигналу перевищує частоту зрізу фільтра, вихідний сигнал значно знижується, а зсув за фазою досягає 90^0.

У вікні Параметри (Parameters) зазначаються такі параметри аналогового входу:

- вхідна напруга; за замовчуванням – від 0 до 10 В постійного струму;
- частота зрізу фільтра (від 0,06 до 88,25 Гц).

Ціле числове значення (Integer type input) – 
Дана функція використовується для введення цілого 16-бітного цілого числа в діапазоні від –32768 до +32767. Цілий числовий вхід може бути присвоєний входам від J9 до JB модулів розширення.

Спеціальні входи у мові FBD (Special inputs in FBD language)
У мові FBD доступні наступні спеціальні входи:

- кнопка;
- дискретні константи;

- числові константи;
- літній час;
- миготіння на протязі 1 секунди.

Дані входи можуть розміщуватися на сторінці FBD-програми.

Кнопки (Button type inputs) – 

Входи типу “кнопка” відповідають клавішам на передній панелі Zelio. Дані входи можуть бути вставлені у FBD-діаграму і у режимі симуляції або моніторингу можуть симулювати контакти.

Дискретні константи (Discrete constant-type inputs)

Існує 2 типи дискретних констант:

- **1** - 

;
- **0** - 

.

У режимі емуляції або моніторингу можна використовувати дані входи в інверсному стані. Тоді символи набувають червоного кольору.

Числові константи (Numerical constant-type inputs) – 

Числова константа **NUM** представляє собою ціле число у діапазоні від –32768 до +32767. Дана константа може використовуватися для встановлення значення функціональних не з’єднуваних входів:

- GAIN;
- COMP IN ZONE;
- TRIGGER.

Значення константи можна встановити у вікні Параметри (Parameters). У режимі емуляції або моніторингу є можливість змінювати константу.

Вхід відображення літнього та зимового часу (Summer Time Input)

Вхід активний на протязі літнього часу –  і неактивний на протязі зимового часу –  .

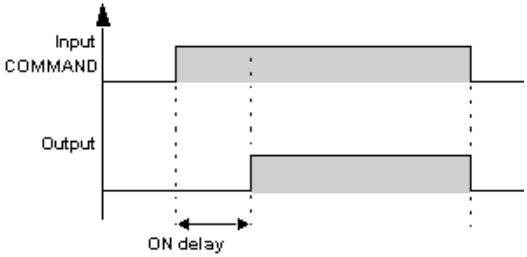
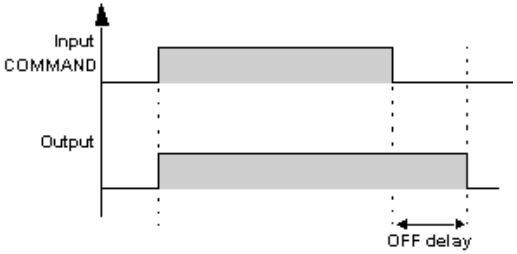
Імпульсний вхід (Blinking input)

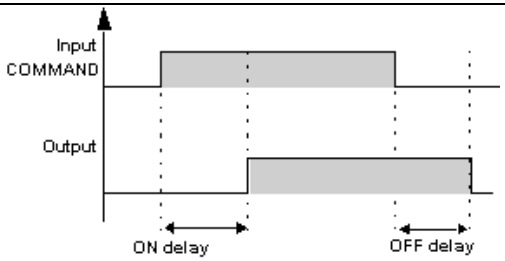
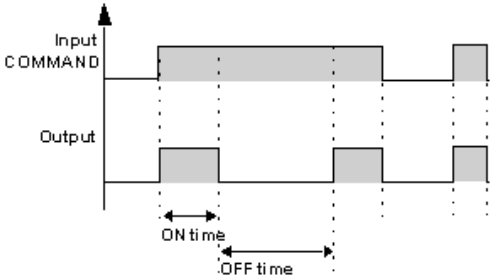
Даний вхід активний кожну секунду. Його активний символ –  , його неактивний символ –  .

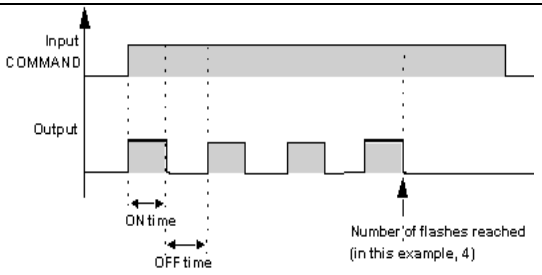
2. Стандартні функції

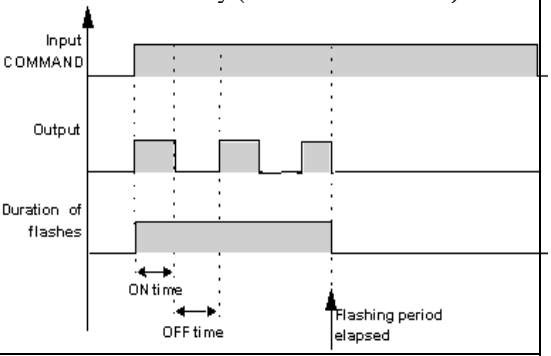
Таблиця 2.5

<table>
<tr><th>Функція</th><th>Сим-вол</th><th>Пояснення</th></tr>
<tr><td>BOOLEAN Equation</td><td>  </td><td>Вихід приймає значення 1 або 0 в залежності від значень входів згідно таблиці істинності.</td></tr>
<tr><td>SET-RESET</td><td>  </td><td>Вихід Q набуває значення 1 або 0 згідно таблиці істинності:
<table>
<tr><th>SET</th><th>RESET</th><th>Q</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>імпульс 0 → 1</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>імпульс 0 → 1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>стан виходу залежить від встановленого пріоритету входів</td></tr>
</table></td></tr>
<tr><td>PRESET COUNT Up-Down Counter</td><td>  </td><td>Лічильник, який використовується для підрахунку імпульсів від нуля до значення уставки (preset) та від значення уставки до нуля.</td></tr>
<tr><td>FAST COUNTER</td><td>  </td><td>Лічильник, який дозволяє вести підрахунок імпульсів з частотою до 1 КГц. Входи даного лічильника неявно зв‘язані з входами I1 (сумуючий вхід) та I2 (декрементуючий вхід) контролера. Тому входи I1 та I2 не</td></tr>
</table>

		рекомендується використовувати для інших цілей при використанні швидкого лічильника. Даний функціональний блок активний лише тоді, коли активний вхід ENABLE. Роботу лічильника не можна симулювати у середовищі програмування.
UP-DOWN COUNTER		Лічильник , який використовується для підрахунку імпульсів, починаючи від значення уставки, яка вираховується поза межами функціонального блоку. Щоб занести у лічильник значення уставки, необхідно підключити до входу PRESET константу NUM, аналоговий вхід тощо та подати 1 на вхід PRESET FORCING. Вихід лічильника активний, якщо число підрахованих імпульсів дорівнює або більше значення уставки PRESET.
TIMER A/C		Таймер, який може працювати у трьох режимах: *A* - затримка включення (on-delay):  *C* - затримка виключення (off-delay):  *A/C* - комбінований режим:

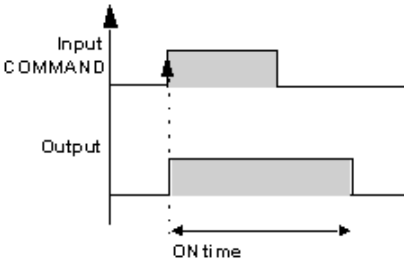
		
TIMER BW		Таймер використовується для генерації імпульсу певної тривалості на виході при подачі імпульсу на вхід таймера. Можливі типи вхідних імпульсів: • імпульс по передньому фронту; • імпульс по задньому фронту; • обидва типи імпульсів. При налаштуванні таймера потрібно вибрати один із зазначених типів вхідних імпульсів.
TIMER Li		Таймер працює у режимі Li, який призначений для генерації імпульсів. Є три модифікації даного режиму: 1. Тривала генерація імпульсів (Continuous Flashing)  2. Генерація певного числа імпульсів(Number of Flashes)

<table>
<tr>
<td></td>
<td></td>
<td>


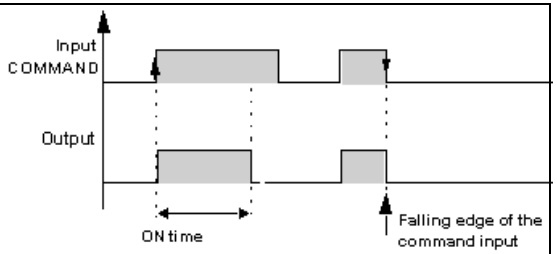
3. Генерація імпульсів на протязі певного часу (Duration of flashes)


</td>
</tr>
<tr>
<td>TIMER B/H</td>
<td> 
</td>
<td>Даний таймер генерує імпульс виході під час приходу переднього фронту імпульсу на вхід. Даний таймер може працювати у двох режимах:

B

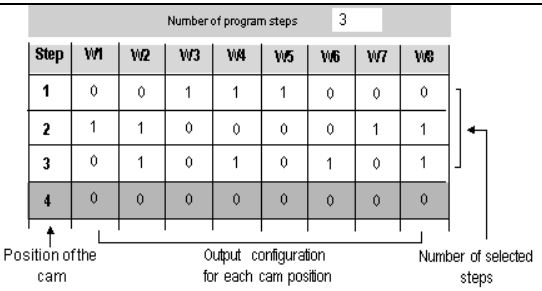


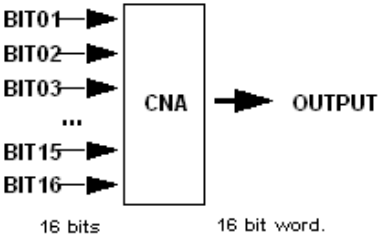
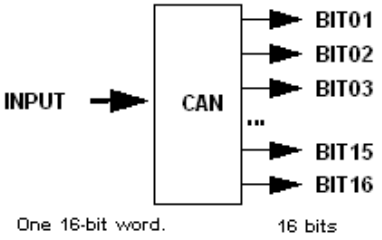
H
</td>
</tr>
</table>

		
COMP IN ZONE Comparison		Функція порівняння використовується для порівняння значення змінної з мінімальним та максимальним значеннями.
PRESET H-METER hour counter		Даний блок призначений для вимірювання проміжку часу, на протязі якого активний командний вхід блоку (Command). Коли період часу активності входу досягає значення уставки, стає активним вихід блоку. Уставка за часом може задаватися в годинах (32757 максимум) та у хвилинах (від 0 до 59). Скидання виходу блоку в нуль відбувається шляхом активації входу Reset.
SCHMITT TRIGGER		Блок тригер Шмідта дозволяє контролювати значення аналогової змінної по відношенню до двох сталих значень – нижньої межі та верхньої межі. Дискретний вихід блоку змінює свій стан на протилежний, якщо: • вхідне значення менше значення нижньої межі (min); • вхідне значення більше значення верхньої межі (max). Вихід не змінює свого значення, якщо вхідне значення знаходиться між min і max. Блок працює, тільки якщо активний дискретний вхід Enable.
COMPARE		Блок порівняння призначений для порівняння значень двох аналогових змінних. Вихід блоку активний, якщо результат порівняння змінних Value 1 та Value 2 позитивний, а дозволяючий вхід Enable активний або ні з чим не з'єднаний. Конкретну операцію порівняння потрібно вибрати у вікні параметрів даного блоку.

GAIN		Функція для масштабування аналогових сигналів за формулою: **Calculation Output = A/B * Calculation Input +C** Для роботи даного функціонального блоку його вхід Enable Function Input повинен бути активний або ні з чим не з'єднаний.
DISPLAY		Дана функція дозволяє виводити на екран контролера такі дані, як текст (до 72 символів), дату , час, числове значення. У програмі одночасно можуть бути активними до 32 блоків Display. Вхід Enable Function Input має бути активним для запуску блоку. Вхід Value Input визначає дані, які будуть виводитися на екран: • вхід ні з чим не з'єднаний – на екран виводитимуться дані, задані користувачем у User options zone при налаштуванні блоку; • вхід з'єднаний з джерелом даних (виходом попереднього функціонального блоку) – на екран виводитимуться дані, які надходять на вхід Value Input.
TIME PROG		Функціональний блок Годинник, який дозволяє задавати інтервали часу у днях тижня, годинах, хвилинах, коли блок активний чи неактивний.
BISTABLE Impulse Relay		Бістабільне імпульсне реле, яке змінює стан свого виходу на протилежний під час приходу запускаючого імпульсу по передньому фронту на його вхід Command. Вхід Reset блоку призначений для скидання його виходу в нуль.
Multiplexing		Функціональний блок мультиплексора, призначений для комутування двох інформаційних входів (Channel A і Channel B) на один вихід. Адресу входу для опитування задає адресний вхід Command. Якщо вхід Command=0 або ні з чим не з'єднаний, то опитується вхід A, якщо Command=1, то опитується вхід B.

ADD-SUB		Функціональний блок для здійснення арифметичних операцій додавання та віднімання трьох цілочисельних даних. Значення виходу блоку розраховується за формулою: **Calculation Output = Input 1 + Input 2 – Input 3** Якщо вхід блоку Error Propagation=1, розрахунок не проводиться. Вихід Error/Overrun=1, якщо результат операції вийшов за межі відрізка [-32768; +32767] або вхід Error Propagation=1.
MUL-DIV		Функціональний блок для здійснення арифметичних операцій множення та ділення трьох цілочисельних даних. Значення виходу блоку розраховується за формулою: **Calculation Output = Input 1 * Input 2 / Input 3** Якщо вхід блоку Error Propagation=1, розрахунок не проводиться. Вихід Error/Overrun=1, якщо результат операції вийшов за межі відрізка [-32768; +32767], вхід Error Propagation=1 або вхід Input 3=0.
CAM BLOCK		Функціональний блок кулачкового механізму контролює роботу 8-ми дискретних виходів. На кожному кроці виходи приймають наперед задані значення. Обертання кулачкового механізму контролюють 2 дискретні входа: • MOVE FORWARD – перехід на крок вперед (має більший пріоритет); • MOVE BACKWARD – перехід на крок назад. Вхід RESET використовується для ініціалізації функціонального блоку (переведення його на перший крок). Вихід POSITION вказує на номер поточного кроку. При конфігуруванні кулачкого механізму необхідно задати наступні параметри: • кількість кроків (number of steps) від 1 до 50; • значення виходів на кожному кроці.

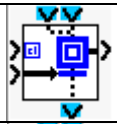
		
ARCHIVE		Функціональний блок для запам‘ятовування значень двох змінних. Входи блоку ARCHIVE: • LATCHING – дискретний вхід, при подачі імпульсу на який відбувається запам‘ятовування значень змінних; • RESET – дискретний вхід скидання блоку, при цьому запам‘ятовані раніше дані втрачаються; • ARCHIVE VALUE 1 – перший вхід для запам‘ятовування; • ARCHIVE VALUE 2 – другий вхід для запам‘ятовування. Значення змінних запа’товуються разом із відомості про час і дату їх надходження. Виходи блоку ARCHIVE: • VALID ARCHIVE – дискретний вихід, який вказує на наявність запам‘ятованих даних; • MINUTE – хвилина (від 0 до 59); • HOUR – година (від 0 до 23); • DAY – день (від 1 до 31); • MONTH – місяць (від 1 до 12); • YEAR – рік (від 0 до 99); • ARCHIVE 1 – значення входу VALUE 1; • ARCHIVE 2 - значення входу VALUE 2.
Module STATUS		Дана функція дозволяє користувачу керувати статусом Zelio 2 і змінювати тим самим поведінку його FBD або SFC програми.

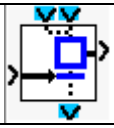
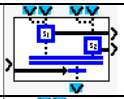
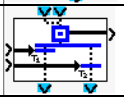
CAN bits to word conversion		Функціональний блок, який перетворює 16-бітний двійковий код у 16-бітне ціле слово. 
CAN word to bits conversion		Функціональний блок, який перетворює 16-бітне ціле слово у 16-бітний двійковий код. 

3. SFC-функції

SFC- функції (Sequential Function Chart) ідентичні функціям мови Grafcet стандарту IEC 1131-3. Мова програмування Grafcet призначена для представлення програми у графічній структурованій формі. Діаграма, складена із SFC- функцій, виконується з верху до низу і складається з кроків (Steps) та переходів (Transitions) між ними.

Таблиця 2.6

Функція	Символ	Опис
Initial Step		Початковий крок
Resettable Initial Step		Початковий крок з ініціалізацією за командою.

Step		Крок
AND Divergence		Перехід до двох кроків одночасно
AND Convergence		Перехід від двох паралельно виконуваних кроків до одного
OR Divergence		Перехід від одного кроку до одного або двох кроків
OR Convergence		Перехід від одного або чотирьох кроків до одиничного кроку

4. Логічні функції

У мові програмування FBD є можливість використовувати логічні функції.

Таблиця 2.7

Функція	**Символ**	**Опис**	**К-сть входів**	**Тип входів**
NO		Вихід активний, якщо вхід неактивний або не підключений і навпаки.	1	Digital
AND		Вихід активний, якщо усі входи активні або не підключені.	4	Digital
OR		Вихід активний, якщо активний хоча б 1 з входів.	4	Digital
NO AND		Вихід активний, якщо хоча б 1 вхід неактивний. Якщо усі входи активні або не підключені, то вихід неактивний.	4	Digital
NO OR		Вихід активний, якщо усі входи неактивні або не підключені. Якщо хоча б 1	4	Digital

		вхід активний, то вихід неактивний.		
EXCLUSIVE OR		Вихід активний, якщо один вхід неактивний, а інший вхід активний або не підключений. Якщо обидва входи активні, неактивні або не підключені, то вихід неактивний.	2	Digital

5. Виходи

Дискретні виходи (Discrete-type outputs) – 
Тип дискретного виходу можна вибрати із вікна Параметри (Parameters).

Таблиця 2.8

Тип	Неактивний стан	Активний стан
Дискретний вихід (descrete output)		
Нормально відкрите реле (normally open relay)		
Лампа (lamp)		
Твердотільне реле (solid state relay)		
Вентиль (valve)		
Поршень (actuator)		
Двигун (motor)		
Резистор (resistance)		

Звуковий сигнал (audible signal)		
Зелена індикаторна лампа (green indicator lamp)		
Червона індикаторна лампа (red indicator lamp)		
Оранжева індикаторна лампа (orange indicator lamp)		
Світловий індикатор (indicator light)		
Нагрівач (heating)		
Вентилятор (fan)		

Цілочисельний вихід (Integer-type output) – 

Дана функція використовується для генерування 16-бітного цілого числа у діапазоні від -32768 до +32767. Цілий числовий вихід може бути присвоєний виходам від O9 до JB модулів розширення.

Вихід підсвітки екрану (LCD Screen Backlighting output)

Дана функція дозволяє програмі контролювати режим підсвічування екрану, вона не може бути присвоєна релейним виходам.

- неактивний стан, 

- активний стан.

6. Входи-виходи Modbus

Модуль розширення Modbus SR3 MBU01BD може бути підключений до базового модуля Zelio SR3 BxxxBD. У FBD режимі

4 (16 бітів) вхідних слова (з J1XT1 по J4XT1) і 4 вихідних слова (з O1XT1 по O4XT1) доступні програмно. Модуль Zelio працює лише у режимі Modbus slave.

Параметри

Параметри встановлюються у робочому середовищі за допомогою пункту меню Edition\Configuration у режимі program\Extension MODBUS Tab або при подвійному натисканні на піктограму Program Configuration  .

При запуску програми Zelio ініціалізує модуль розширення Modbus.

Модуль має 4 параметри:

- число UART з'єднань і frame формат у мережі Modbus;
- швидкість передачі даних у біт/с;
- UART- парність;
- мережева адреса модуля розширення Modbus.

Входи Modbus

Модуль розширення Modbus SR3 MBU01BD має 4 (16 бітів) входа:

- J1XT1 – шістнадцяткова адреса (0010);
- J2XT1 – 0x0011;
- J3XT1 – %0x0012;
- J4XT1 – %0x0013.

Дані завантажуються з пристрою, що працює у режимі master.

Виходи Modbus

Модуль розширення Modbus SR3 MBU01BD має 4 (16 бітів) виходи:

- O1XT1 – шістнадцяткова адреса (0000);
- O2XT1 – 0x0001;
- O3XT1 – 0x0002;
- O4XT1 – %0x0003.

Дані завантажуються у пристрій, що працює у режимі master.

3. Програма роботи

1. Вивчити будову та характеристики контролера Zelio Logic 2.

2. Дослідити правила підключення контролера Zelio Logic 2.
3. Навчитися користуватися меню контролера Zelio Logic 2
4. Навчитися програмувати контролер Zelio Logic 2 мовою драбинкових діаграм (LD) за допомогою командних клавіш передньої панелі.
5. Навчитися програмувати контролер Zelio Logic 2 мовою драбинкових діаграм (LD) у середовищі Zelio Soft 2.
6. Навчитися програмувати контролер Zelio Logic 2 мовою FBD-блоків у середовищі Zelio Soft 2.
7. Навчитися записувати програми у контролер.
8. Навчитися створювати вікно супервізора у середовищі Zelio Soft 2.
9. Виконати завдання, поставлені викладачем.

4. Опис лабораторного обладнання

1. Персональний комп‘ютер.
2. Операційна система Windows.
3. Лабораторний стенд з логічним контролером Zelio Logic.
4. Середовища програмування Zelio Soft, Zelio Soft 2.

5. Порядок виконання роботи

1. Напишіть програму для програмованого логічного контролера (ПЛК) Zelio Logic мовою драбинкових діаграм (LD) для системи контролю та керування температурою повітря і вологості ґрунту в теплиці. При створенні програми виберіть модель контролера SR1B122BD Version 1.6, середовище для програмування – Zelio Soft. Виконайте дії, описані у п.2.2.1.1 теоретичних відомостей.
2. Проведіть емуляцію та налагодження програми у середовищі програмування Zelio Soft. Для цього скористайтеся меню Simulation та натисніть кнопку Run. Після проведення перевірки роботи програми натисніть кнопку Stop.
3. Запишіть створену програму у ПЛК та дослідіть її роботу. Для цього необхідно виконати наступні дії:

- подати живлення на стенд;
- перевести контролер у режим STOP;
- ввійти у головне меню, натиснувши клавішу Sel./OK;
- вибрати пункт меню Transfer.\ PC -> Module;

- у середовищі програмування Zelio Soft вибрати пункт меню Transfer\ Transfer Program\ PC -> Module.

4. Навчиться керувати запуском та зупинкою програми з клавіатури контролера та з комп‘ютера. Для керування виконанням програми з клавіатури контролера скористайтеся пунктом меню Run\ Stop, з комп‘ютера – пунктами меню середовища Zelio Soft Transfer\ Run Module, Transfer\ Stop Module.
5. Для відслідковування роботи програми на комп‘ютері виберіть пункт меню Mode\ On Line Monitoring Mode.
6. Складіть пояснюючу таблицю використаних входів і виходів ПЛК, в якій зазначте назву входу-виходу, тип, призначення.

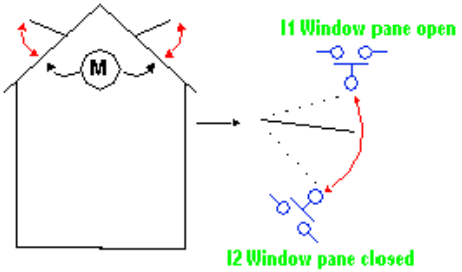
7. Напишіть програму для програмованого логічного контролера (ПЛК) Zelio Logic2 мовою FBD-блоків для системи контролю та керування температурою повітря і вологості ґрунту в теплиці. При цьому використайте середовище програмування Zelio Soft2. Виконайте дії, описані у п.2.2.1.1 теоретичних відомостей.
8. Створіть вікно супервізора, в якому намалюйте ФСА технологічного процесу. Для того, щоб переключитися у вікно супервізора скористайтеся меню Window\Supervision.
9. Проведіть емуляцію та налагодження програми у середовищі програмування Zelio Soft2. Дослідіть роботу програми у середовищі програмування Zelio Soft2. При цьому скористайтеся керуючими кнопками, наведеними у таблиці 9.
10. Складіть пояснюючу таблицю використаних входів і виходів ПЛК, в якій зазначте назву входу-виходу, тип, призначення.

11. Зробіть висновки щодо можливостей двох мов програмування – LD та FBD – для розробки програмного забезпечення для керування технологічним процесом.

<u>Опис процесу та вимоги до програм:</u>

Власник теплиці замовив установку для автоматичного керування вентиляційними вікнами теплиці, встановленими на даху теплиці, та вологістю ґрунту. Теплиця має 2 вентиляційні вікна. Робота даних вікон керується двигуном і двома давачами

положення вікон (відкрито, закрито). На протязі дня вікна відкриваються для вентиляції приміщення з 12:00 до 15:00, коли, як правило, температура найвища. Але, якщо температура є нижчою 10^0C, вікна не відкриваються, а якщо вони вже відкриті, то закриваються.



Додатково вентиляційні вікна відкриваються на протязі дня, коли температура досягає 25^0C. Коли температура падає нижче 25^0C, вентиляційні вікна повинні закриватися.

Вночі вентиляційні вікна залишаються закритими незалежно від температури

Отже, у програмі потрібно використати 3 часові діапазони:

- Ніч: з 21:00 до 7:00
- День: з 7:00 до 12:00 та з 15:00 до 21:00
- Полудень: з 12:00 до 15:00

Роботу програми можна представити у вигляді діаграми:

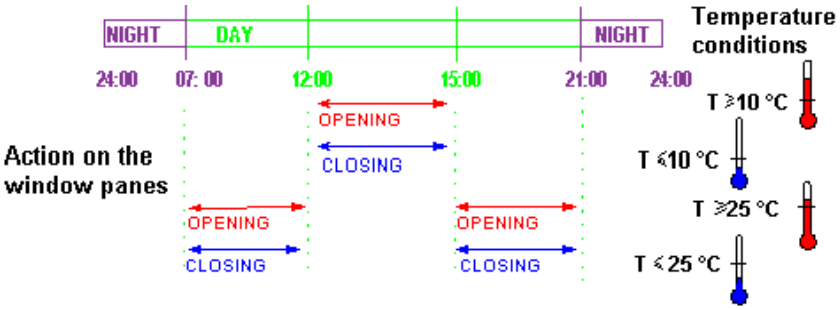


Рис. 2.3 Графік відкриття вентиляційних вікон

Для підтримки заданої вологості ґрунту, яка вимірюється аналоговим давачем вологості, у теплиці передбачений полив дощуванням. Забір води здійснюється насосом із ємності, де вода

відстоюється та нагрівається до кімнатної температури. Ємність наповнюється із водопровідної мережі за допомогою електромагнітного клапану і обладнана двома дискретними давачами рівня (максимального та мінімального). Ввімкнення насосу можна проводити, якщо рівень води в ємності не нижчий за мінімальний. Після проведення поливу ємність необхідно заповнити до максимально допустимого рівня. Для уникнення можливості перезволоження ґрунту тривалість поливу не повинна перевищувати 5 хвилин, а наступний полив можливий тільки через 10 хвилин. Необхідно передбачити також 2 режими поливу: ручний та автоматичний.

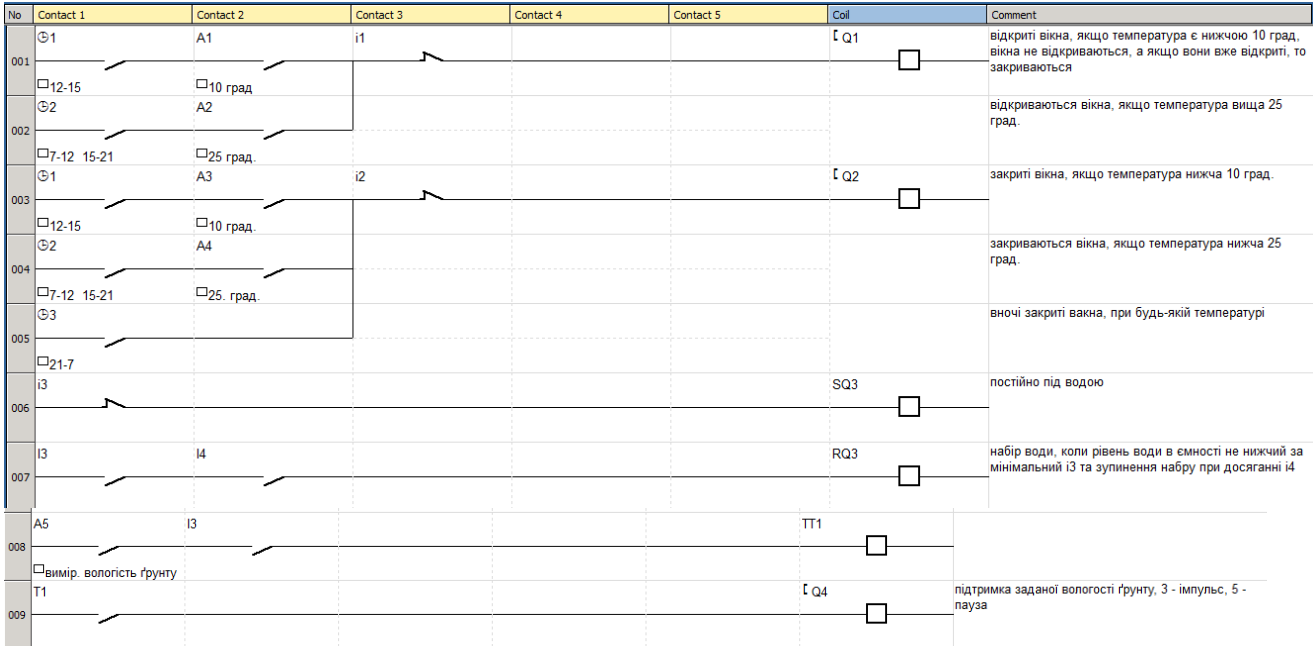
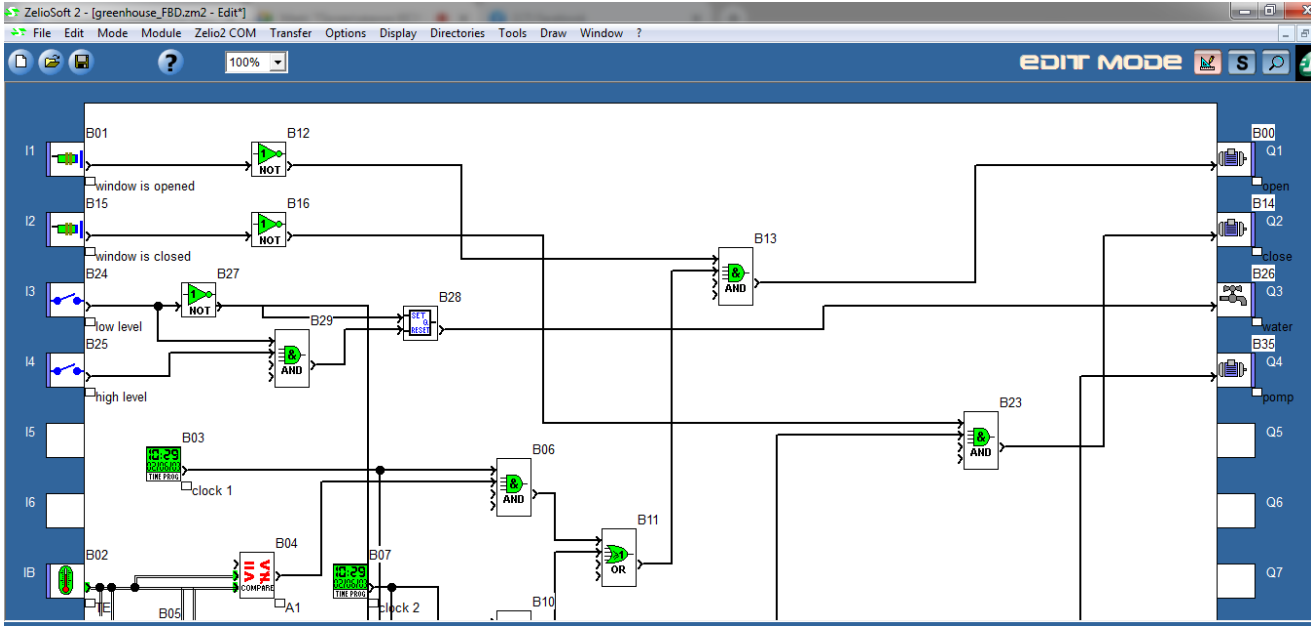
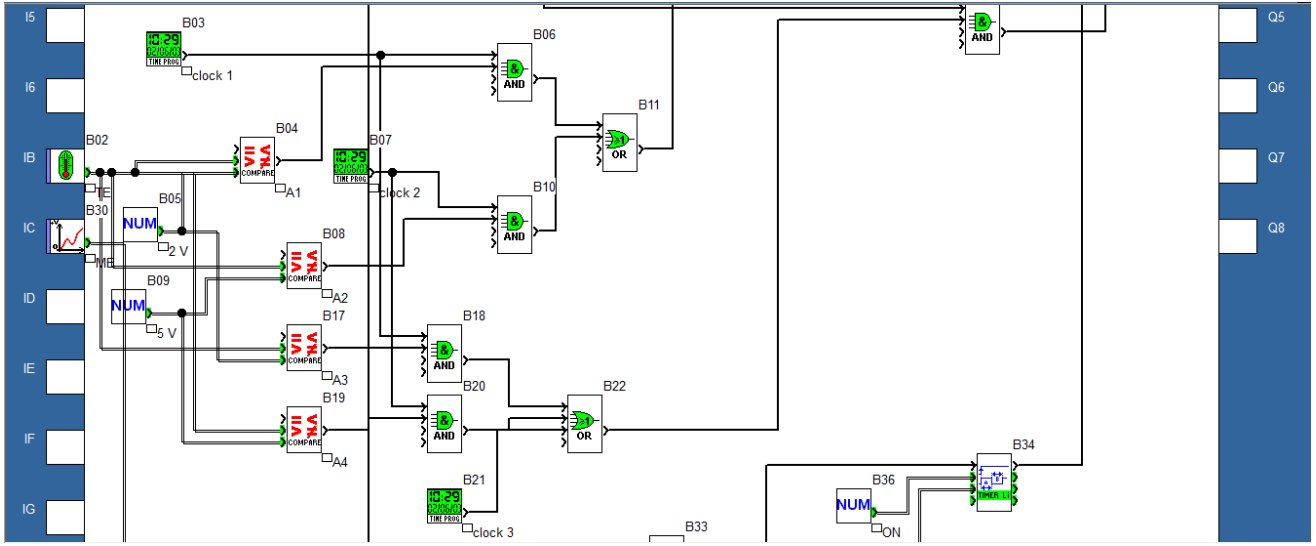


Рис. 2.4 Програма керування теплицею мовою LD


ZelioSoft 2 - [greenhouse_FBD.zm2 - Edit*]
File Edit Mode Module Zelio2 COM Transfer Options Display Directories Tools Draw Window ?
100%
EDIT MODE
I1
I2
I3
I4
I5
I6
IB
B01
window is opened
B12
NOT
B15
window is closed
B16
NOT
B24
low level
B27
NOT
B25
high level
B29
AND
B28
SET & RESET
B13
AND
B03
clock 1
B06
AND
B23
AND
B11
OR
B02
B04
COMPARE
A1
B05
B07
clock 2
B10
B00
Q1
open
B14
Q2
close
B26
Q3
water
B35
Q4
pomp
Q5
Q6
Q7


I5
I6
IB
IC
ID
IE
IF
IG
Q5
Q6
Q7
Q8
B02
B03
clock 1
B04
A1
B05
2 V
B06
B07
clock 2
B08
A2
B09
5 V
B10
B11
OR
B17
A3
B18
B19
A4
B20
B21
clock 3
B22
OR
B30
B33
B34
B36
ON
AND
COMPARE
NUM

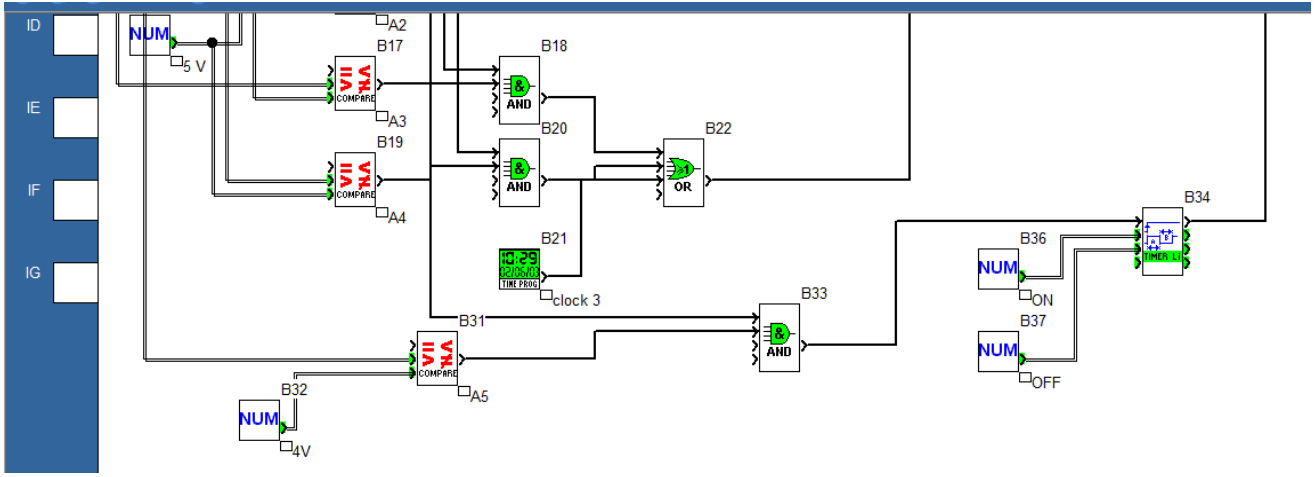


Рис. 2.5 Програма керування теплицею мовою FBD

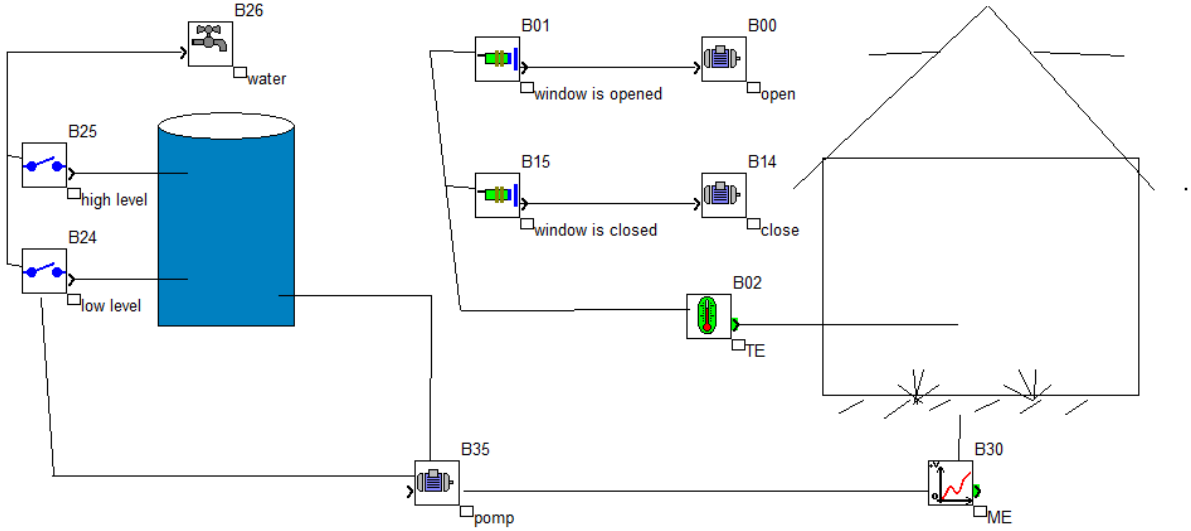


Рис. 2.6 Екран супервізора для моніторингу процесу у режимі реального часу

6. Контрольні запитання

1. Які фактори впливають на розвиток та врожайність сільськогосподарських культур
2. Які функціональні блоки були вами використані у програмі для керування температурою повітря та вологістю ґрунту у теплиці? Яке їх призначення?
3. Для чого призначений функціональний блок ГОДИННИК? Яким чином він програмується?
4. Як визначити параметри ПЛК Zelio Logic за його маркуванням?
5. Яким чином запрограмувати логічний контролер Zelio Logic через командні кнопки?
6. Яким чином запрограмувати логічний контролер Zelio Logic через середовище Zelio Soft?
7. Яким чином запрограмувати контролер мовою драбинкових діаграм (Ladder Diagram Language)?
8. Яким чином запрограмувати контролер мовою FBD-блоків (Function Blocks Diagram Language)?
9. Яким чином записати програму у ПЛК з комп‘ютера? Яким чином зчитати записану програму з ПЛК на комп‘ютер?
10. Як можна відслідкувати роботу програми з комп‘ютера у реальному часі?
11. Для автоматизації яких процесів слід використовувати ПЛК Zelio Logic? Які можливості має ПЛК Zelio Logic для роботи у дворівневих системах контролю та керування?
12. Порівняйте можливості двох мов програмування – LD та FBD – при написанні програм для керування температурою повітря та вологістю ґрунту у теплиці.

Робота №4. Розробка і дослідження системи контролю та керування витратою води у трубопроводі на базі ПЛК Schneider Micro TSX 37-22.

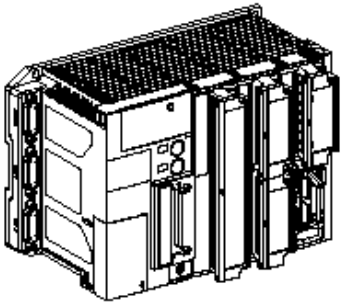
1. Мета роботи

Вивчити будову та характеристики ПЛК Schneider Micro TSX 37-22, навчитися його програмувати із середовища PL7 PRO. Навчитися розробляти дворівневі системи контролю та керування на базі ПЛК Schneider Micro TSX 37-22.

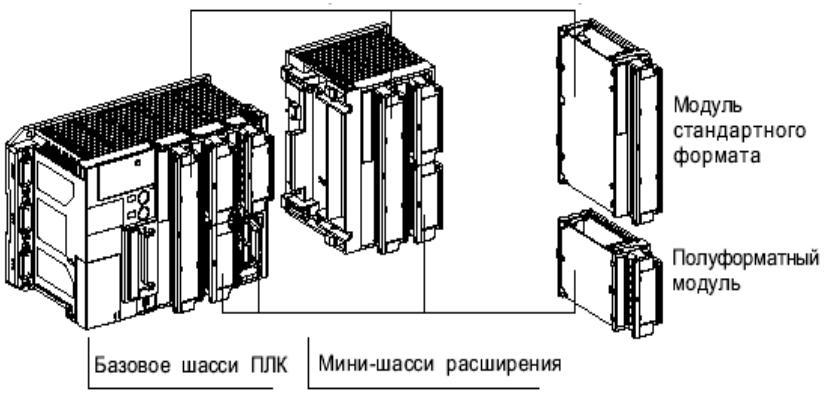
2. Теоретичні відомості

2.1 Представлення ПЛК Schneider Micro TSX 37-22

ПЛК (програмований логічний контролер) TSX 37-22 модульний, з вбудованим астрономічним годинником, з можливістю розширення розміру пам‘яті і встановлення комунікаційного модуля. Контролер має вбудовані аналогові входи-виходи та лічильні канали.



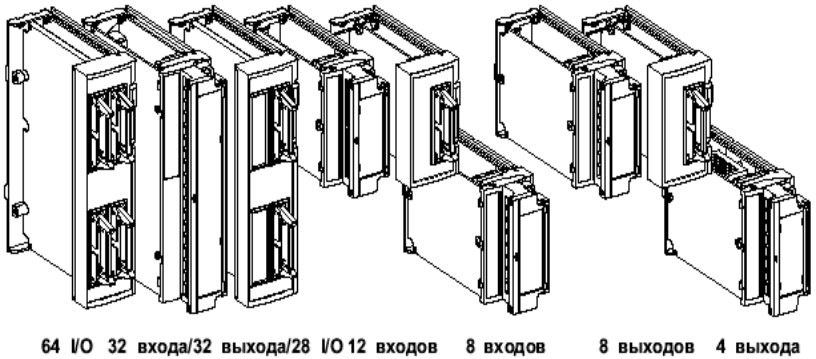
Дискретні модулі бувають двох форматів: стандартний, який займає 1 слот (2 посадочних місця) та полуформатний, який займає тільки одне посадочне місце. Усі інші модулі (аналогові, лічильні та інші) – полуформатні. За допомогою міні-шасі розширення, яке може бути безпосередньо підключено до базового шасі, можна збільшити кількість доступних слотів і, відповідно, кількість модулів, які можна встановити.



2.1.1 Дискретні входи-виходи

Дискретні входи-виходи, які встановлюються в шасі

Дискретні модулі входів-виходів (I/O) відрізняються не тільки форматом (стандартні і напівформатні), але і кількістю каналів (від 4 виходів до 64 I/O), типом входів (постійного - DC або змінного струму - AC), типом виходів (транзисторні або релейні) і підключенням (гвинтова клемна колодка або HE 10 з'єднувач).



У цілому :

- модулі постійного струму (24 V DC) поставляються як із з'єднанням "під гвинт", так і із з'єднувачами HE 10 ; за винятком вихідного модуля 24 V DC/2A і модулів на 32 входа та 32 вихода, які мають тільки підключення "під

гвинт", і модуля високої щільності на 64 входа-виходa, що має тільки НЕ 10 з'єднувачі.

- інші модулі, включаючи модулі змінного струму AC або релейні виходи, завжди обладнані гвинтовою клемною колодкою.

Віддалені дискретні I/O

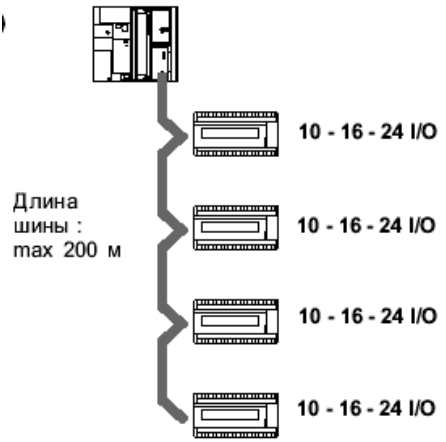
- використання віддалених модулів I/O, у якості яких застосовуються ПЛК типу Nano.

Використання у якості віддалених входів-виходів до 4 TSX 07 дозволяє збільшити відстань до входів-виходів (до 200 метрів) і збільшити їхню кількість у конфігурації.

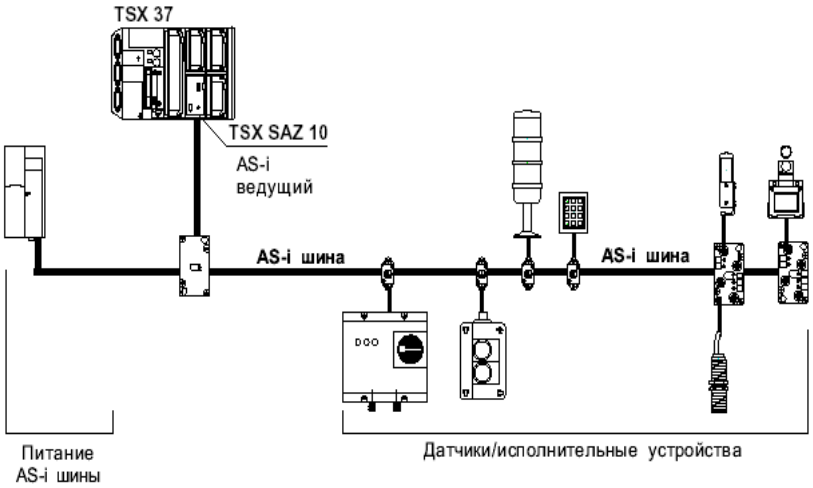
Примітка: Використання віддалених входів-виходів на ПЛК Nano перешкоджає використанню модуля зв‘язку для AS-шини.

- використання модуля зв'язку шини AS-i TSX SAZ 10.

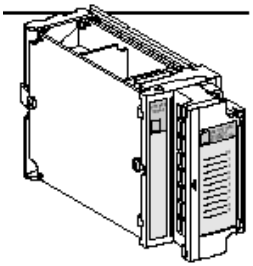
Використання модуля зв'язку AS-i шини дає можливість управляти 124 вхідними і 124 вихідними розподіленими бітами для понад 31 ведених пристроїв з урахуванням обмеження 4 вхідних та 4 вихідних біт на один пристрій. Максимальна довжина шини обмежена 100 метрами.



Примітка: Використання модуля зв'язку AS-i шини перешкоджає можливості використання віддалених модулів на ПКЛ Nano.

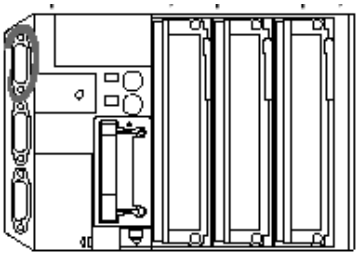


2.1.2 Дискретний модуль безпеки



Напівформатний модуль безпеки TSX DPZ 10D2A виконує повну діагностику схем безпеки. Він здійснює контроль за аварійною зупинкою у випадку виникнення аварійної ситуації відповідно до вимог стандартів безпеки EN 954-1, EN 418 та EN 60204-1.

2.1.3 Аналогові входи-виходи



8 входов 0-10 V и
1 выход 0-10 V, 8 битные

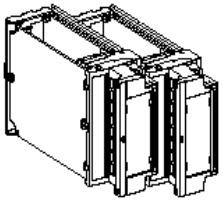
Аналогові модулі TSX 37 відрізняються кількістю каналів, їхніми характеристиками і діапазоном вимірювання (рівень напруги, термопари,термометри опору та ін.).

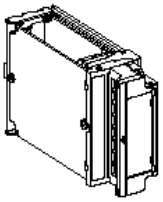
• ПЛК TSX 37-22 має 8 аналогових 0-10 В 8-бітових входів та 1 аналоговий 0-10 В 8-бітовий вихід,

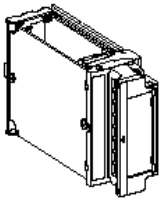
еталонний 10 В вихід, забезпечуючи економічне рішення для цілого ряду прикладних завдань.

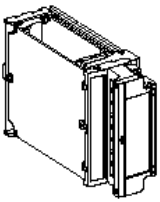
Ці входи можуть бути використані разом з модулем коректування і адаптації, що дозволяє:
- ручне регулювання змінних програми за допомогою 4-х потенціометрів;
- перетворення сигналу 4-20 мА на 0-10 В;
- підключення дискретних сигналів 24 В (тип 1) до аналогових входів.

• Модулі аналогових входів і виходів можуть бути установлені в усі типи ПЛК, забезпечуючи високу точність. Вони відрізняються за кількістю каналів (від 2 до 8) і типами входів-виходів (рівень напруги або струму, термомодулі та ін.). Підключення за допомогою гвинтової клемної колодки.







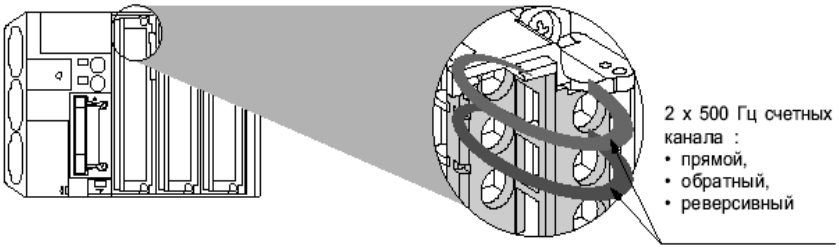


8 входов	**4 входа**	**4 выхода**	**2 выхода**
0-10 В ± 10 В или 0-20 mA 4-20 mA	дифференциальный, многодиапазонный (± 10 В, 4-20 mA, термомодули, Pt 100 и др.	± 10 В	0-20 mA 4-20 mA
12 бит	16 бит	11 бит + знак	11 бит + знак

2.1.4 Лічильні канали

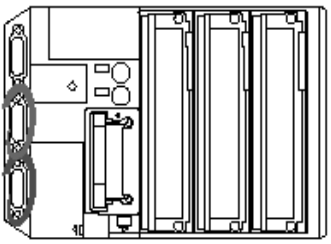
ПЛК TSX 37 дозволяє використовувати три методи рахунку:
• з використанням дискретних входів першого модуля;
• з використанням вбудованих у ПЛК TSX 37-22 лічильних каналів;
• з використанням спеціалізованих лічильних модулів (TSX CTZ1A/2A, TSX CTZ 2AA), які можуть бути встановлені в доступний слот.

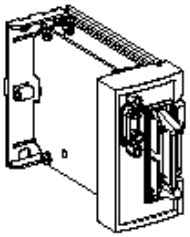
Перші чотири входа дискретного модуля, встановленого у першому слоті ПЛК, забезпечують два 500 Гц реверсивних лічильних канали.

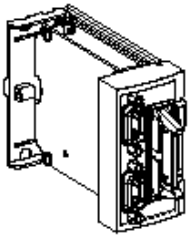


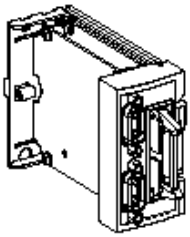
У ПЛК TSX 37-22 вбудовані два нагромаджуючих 10 КГц лічильних канали, які також обладнані функціями скидання, попередньої установки та обнуління лічильників.

Лічильні модулі (прямі, зворотні, реверсивні лічильники) відрізняються за кількістю каналів, граничною частотою (40 КГц або 500 КГц), типом та кількістю відповідних логічних сигналів.




1 x 40 кГц
счетный канал :
• прямой,
• обратный,
• реверсивный
счетчик


2 x 40 кГц
счетных канала:
• прямой,
• обратный,
• реверсивный
счетчик


2 x 500 кГц
счетный канал :
• прямой,
• обратный,
• реверсивный
счетчики

2.1.5 Комунікаційні можливості

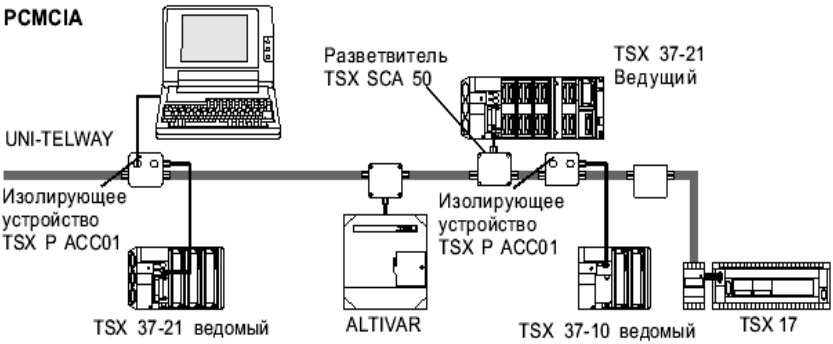
ПЛК TSX 37 легко підключається до багатоточковим послідовним каналам передачі даних за допомогою термінального порту (для всіх моделей ПЛК) і додаткового порту для підключення засобів людино-машинного інтерфейсу (для ПЛК TSX 37-21/22). Ці порти дозволяють підключити (тільки по одному протоколі одночасно):

- програмний термінал або людино-машинний інтерфейс (режим UNI-TELWAY ведучий);
- багатоточковий зв'язок ПЛК по мережі UNI-TELWAY (UNI-TELWAY ведучий-ведений);
- принтер або термінал у символьному режимі;
- модем.

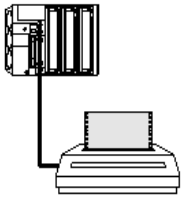
Ізолюючий пристрій TSXPACC 01 дозволяє підключати ПЛК до шини UNI-TELWAY, коли відстань перевищує 10 метрів. Додатково він дублює термінальний порт, що дозволяє підключити до ПЛК TSX 37-10 одночасно програмний термінал і засоби людино-машинного інтерфейсу.

ПЛК TSX 37-21 і TSX 37-22 обладнані слотом для комунікаційної карти формату PCMCIA (дуплексний або напівдуплексний послідовний асинхронний зв'язок, UNI-TELWAY, JBUS/MODBUS, FIPWAY, Agent FIPIO and Modbus+).

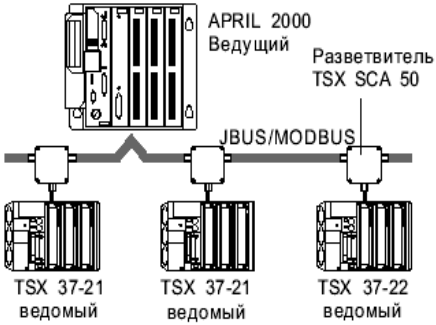
Связь UNI-TELWAY ведомого через терминальный порт и ведещего через карту PCMCIA



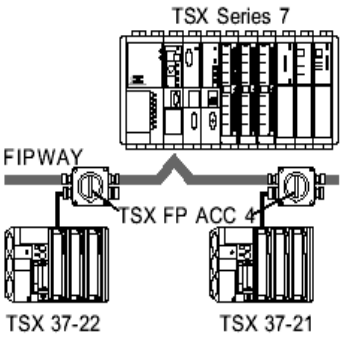
Символьный режим через терминальный порт



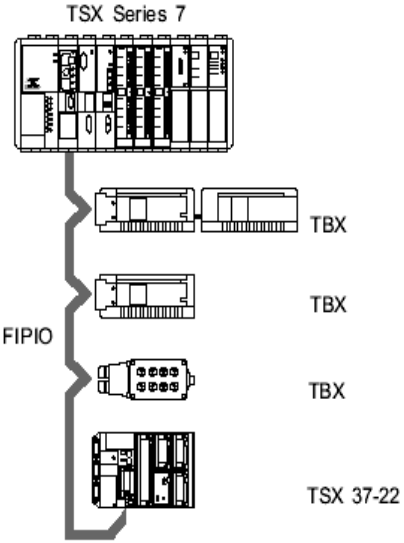
Связь по JBUS/MODBUS через коммуникационную карту



Подключение к сети FIPWAY через коммуникационную карту

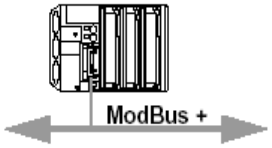


Связь по FIPIO через коммуникационную карту



Подключение сети Modbus + через коммуникационную карту

TSX 37 21/22



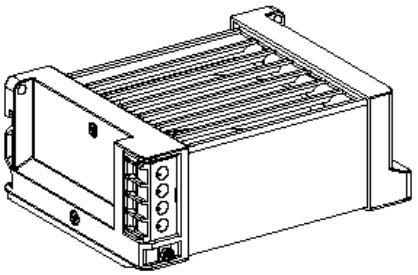
2.1.6 Вимушена вентиляція

Залежно від типу ПЛК (TSX 37-10 або TSX 37-21/22 з або без шасі розширення), один або два вентиляційних модуля можуть бути встановлені над кожним ПЛК, для того щоб примусовим способом охолоджувати різні модулі. Ці вентилятори повинні використовуватися у наступних випадках:

• **Навколишня температура у діапазоні 25°C...60°C**: примусова вентиляція збільшує термін служби різних компонентів ПЛК TSX Micro (збільшення 25% у MTBF).

• **Навколишня температура у діапазоні 60°C...70°C:** оскільки температура без вентиляції обмежена 60°C, примусова вентиляція використовується для зменшення температури всередині модулів більше, ніж на 10°C и приведення внутрішньої температури модулів до еквівалента навколишнього середовища 60°C.
У цьому випадку термін служби виробу збільшується більше, ніж на 50%.

Є три типи модулів вентилятора :
• вентиляторі напругою живлення 110 В змінного струму;
• вентилятор з напругою живлення 220 В змінного струму;
• вентилятор з напругою живлення 24 В постійного струму.



! Використання примусової вентиляції не дозволяється у випадках, коли у ПЛК встановлені аналогові модулі TSXAEZ414 .

2.2 Базове виконання ПЛК TSX 37-22

2.2.1 Загальний огляд

ПЛК TSX 37-21 або TSX 37-22 містять шасі з вбудованим джерелом живлення 24 В постійного струму (TSX 37-21 101 і TSX 37-22 101) або 100-240 В змінного струму(TSX 37-21 001 і TSX 37-22 001), процесор, пам'ять, систему резервного копіювання і 3 слота для модулів.

Використання міні-шасі розширення TSX RKZ 02 дає можливість додати до ПЛК 2 додаткових слота і, таким чином, одержати 5 слотів для установки модулів. У кожен з 5 слотів може бути встановлений або один модуль стандартного формату або два напівформатних модулі (за винятком першого слота, у який можуть встановлюватися тільки модулі стандартного формату). У нижченаведеній таблиці наведені **максимальні** конфігурації ПЛК

TSX 37-21 і TSX 37-22 (максимальна кількість модулів входів-виходів):

ПЛК		TSX v	37-21	37-22
Дискретные	Максимальное количество входов-выходов	базовые + расширение + удаленные (TSX 07)	332	332
		базовые + расширение + удаленные (AS-i шина)	472	472
		в базовом шасси ПЛК	192	192
		в базовом + мини-шасси расширения	256	256
		удаленные (4 TSX 07)	96	96
		удаленные на AS-i шине (124 вх. + 124 вых.)	248	248
	Максимальное количество модулей	28 или 32 дискретных входов-выходов	5	5
		64 дискретных вх.-вых. (высокая плотность)	3	3
		для удаленных вх-вых(TSX 07 или AS-i шина)	1	1
Аналоговые	Максим. кол-во модулей аналоговых вх-вых		4	4
	Максим. кол-во аналоговых входов в базовом шасси		32	32
	Максим. кол-во аналоговых выходов в базовом шасси		16	16
	Максим. кол-во интегрированных аналоговых входов		_	8
	Максим. кол-во интегрированных аналоговых выходов		_	1
Счетные	Максим. кол-во 500 Гц счетных каналов на дискр. входах		2	2
	Максим. кол-во счетных модулей (в ПЛК) (1)		4	4
	Максим. кол-во 40 кГц и/или 500 кГц счетных каналов		7	7
	Максим. кол-во интегрированных счетных каналов (10 кГц)		_	2
Коммуникации (2)	Кол-во коммуникационных карт (выделенный слот)		1	1

(1) Лічильні модулі можуть бути встановлені тільки у базове шасі ПЛК.

У TSX 37-21/22 можуть бути встановлені 4 аналогових і 4 лічильні модулі.

(2) Комунікаційні карти формату PCMCIA (FIPWAY, Agent FIPIO , Modbus+).

Два термінальних порта RS 485 з 8-штирьковими роз‘ємами стандарту DIN дозволяють підключати до ПЛК:

• TER: програмний термінал FTX або PC-сумісну ПЕОМ, а також підключати ПЛК до шини UNI-TELWAY через ізолюючий пристрій TSXPACC 01;
• AUX: людино-машинний інтерфейс або принтер.

За замовчуванням, для роботи в UNI-TELWAY обидва порти сконфігуровані у режимі ведучого на швидкості 9600 бод. Вони також можуть бути сконфігуровані для роботи в UNI-TELWAY у режимі веденого або у символьному режимі ASCII.

Дисплейний блок відображає всю необхідну інформацію для діагностики та підтримки ПЛК (базового шасі та шасі розширення) і його модулів.

Два слота формату PCMCIA призначені для карт розширення пам'яті і комунікаційної карти.

ПЛК TSX 37-22 також має 3 з'єднувачі для підключення інтегрованих аналогових входів-виходів і лічильників.

2.2.2 Дисплейний блок

Дисплейний блок призначений для відображення інформації, необхідної для діагностики і підтримки ПЛК і його модулів. Для цього він має:

- 8 індикаторних ламп стану, які відображають інформацію про режими роботи та функціонування ПЛК (індикаторні лампи RUN, TER, I/O, ERR, BAT), а також режими роботи дисплея (R I/O, WRD й DIAG);
- блок із 96 індикаторних ламп, які можуть відображати:

- у режимі відображення локальних I/O (горять індикаторні лампи BASE або EXT): стан всіх входів-виходів базового шасі ПЛК і шасі розширення;

- у режимі відображення віддалених I/O (горять індикаторні лампи R I/O): стан дискретних I/O кожного веденого на AS-і шині,

- у режимі діагностики (горить індикаторна лампа DIAG):

Для локальних I/O: помилка модуля (всі індикаторні лампи, пов'язані із цим модулем, повільно мигають) або помилка каналу (індикаторна лампа, пов'язана із цим каналом, мигає швидко),

Для віддалених I/O на AS-i шині: стан кожного веденого (несправний ведений мигає).

- у режимі відображення об'єктів (горить індикатор WRD): значення максимум 16 слів %MWi, %SWi або %KWi (ці слова відображаються у двійковому або шістнадцятковому форматі); стан групи із 64 бітів %Mi, %Si or %Ki; стан бітів входів і виходів модулів TSX 07, використовуваних у якості дискретних віддалених входів-виходів.

- кнопки, які дозволяють змінювати режими роботи дисплейного блоку і дозволяти відображення інформації.

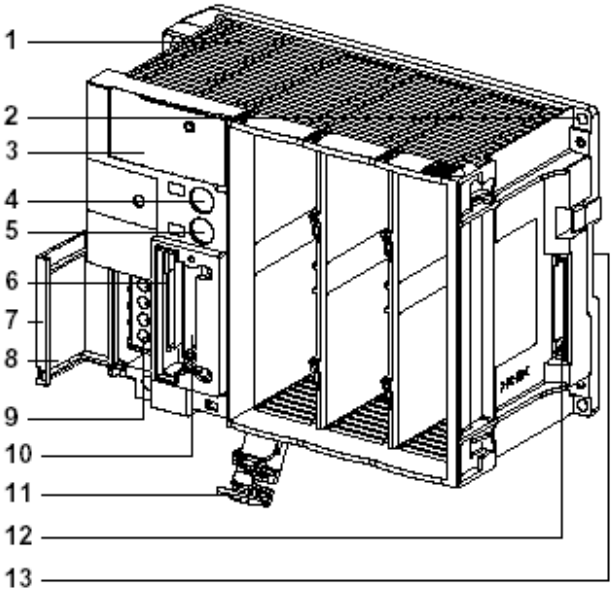
2.2.3 Загальний вигляд

1. Шасі з трьома слотами для модулів, вбудоване джерело живлення, процесор з його пам'яттю.
2. Монтажні точки ПЛК.
3. Дисплейний блок.
4. Термінальний порт TER.
5. Порт для підключення людино-машинного інтерфейсу.
6. Слот для карти розширення пам'яті. Якщо карта не встановлена, цей слот закривається кришкою, яка не повинна зніматися. Її видалення викликає зупинку ПЛК.
7. Кришка для доступу до контактів джерела живлення.
8. Ярлик, що заповнюється при зміні батарейки.
9. Контакти джерела живлення.
10. Слот для комунікаційної карти.
11. Кришка для доступу до додаткової батареї і перемикача захисту операційної системи.
12. З'єднувач для підключення шасі, звичайно закритий з‘ємною кришкою.
13. Пристрій для монтажу на DIN рейці.
14. З'єднувачі для підключення вбудованих аналогових і лічильних функцій.

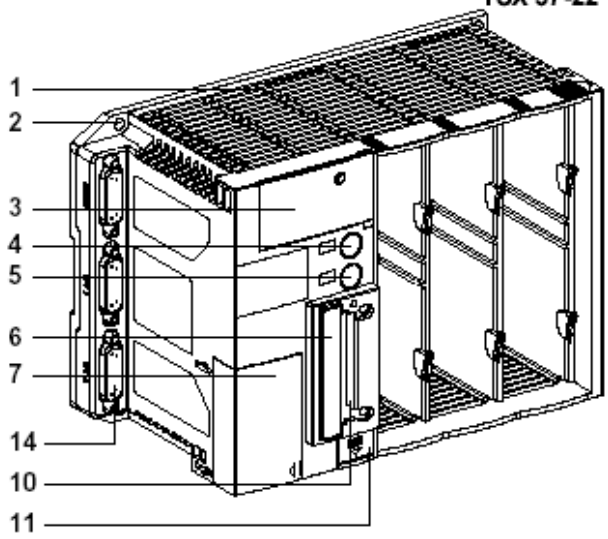
Зауваження :

Для забезпечення ступеня захисту IP20 ви повинні закрити спеціальними кришками вільні слоти. Ці кришки не входять у комплект поставки і повинні бути замовлені окремо у кількості по 10 штук, тип TSX RKA 01

TSX 37-21



TSX 37-22



2.3 Характеристики
2.3.1 Процесори

Тип ПЛК		TSX 37-10	TSX 37-21	TSX 37-22
Функции	К-во дискр. вх-вых ПЛК + удаленные TSX 07 ПЛК+ удаленные AS-i шина	 268 408	 332 472	 332 472
	К-во встроенных UNI-TELWAY соединений	1	1	1
	Коммуникационные модули	0	1	1
	Астрономические часы	нет	да	да
	Встроенные аналог. вх-вых	нет	нет	да
	Встроенные счетчики 500 Гц 10 кГц	да нет	да нет	да да
Память	Внутр.энергонезавис.RAM • программа (100% Булевск) • данные • константы	14K слов 4.7/2.7 Кинст. 1 Кслов (2) 128 слов (2)	20 Кслов 7.9/4.5 Кинстр. 2 Кслов (2) 128 слов (2)	20 Кслов 7.9/4.5 Кинст. 2 Кслов (2) 128слов (2)
	Встроенная Flash Eprom	16 Кслов (3)	16 Кслов (3)	16 Кслов (3)
	Карта PCMCIA 32 K16 • программа (100% Бул.) (1) • данные (во внутр. RAM) • константы (4)	_	32 Кслов 18.5/10.5 Кин. 17.5 Кслов 128 слов (2)	32 Кслов 18.5/10.5Кин 17.5 Кслов 128слов (2)
	Карта PCMCIA 64 K16 • программа (100% Бул.) (1) • данные (во внутр. RAM) • константы (5)	_	64 Кслов 40/22 Кинстр. 17.5 Кслов 128 слов (2)	64 Кслов 40/22 Кинстр 17.5 Кслов 128 слов (2)
Время вып. .Кинстр (7)	RAM (100% Булевские) PCMCIA (100% Булевские)	0.3 мс _	0.15 мс 0.225 мс	0.15 мс 0.225 мс
Системные затраты времени		1.9 мс	1.6 мс	2.3 мс
Структура приложения	Основная задача	1	1	1
	Быстрая задача	1	1	1
	Обработка событий	1 to 8	1 to 16	1 to 16
Функциональ-ные блоки	Таймеры	64 (6)	64 (6)	64 (6)
	Счетчики	32	32	32

(1) Значення 1 відповідає програмі мовою Список Інструкцій. 2-мовою Сходинкових діаграм.

(2) Значення за змовчуванням, при потребі може бути збільшено.

(3) 15 Кслів використовується для копії програми + 1 Кслов для копії %MW .
(4) Може бути збільшене до 24.5 Кслів.
(5) Може бути збільшене до 32 Кслів.
(6) Максимум 16 таймерів з дискретністю 10 мс.
(7) Крім системних витрат часу та керування входами-виходами.

2.3.2 Джерела живлення

Джерела живлення змінного струму

Тип ПЛК			**TSX 37-10/21/22**
Первичные	Номинальное напряжение		100...240 В
цепи	Предельные напряжения		85-264 В
	Номинальная частота		50-60 Гц
	Ограничения частоты		47-63 Гц
	Потребляемый ток		0.7 А при 100 В 0.3 А при 240 В
	Пусковой ток (2)		< 60 А
Вторичные	+ 5 В пост.тока	Номинальный ток(1)	2.8 А
цепи		Пиковый ток	3.2 А
	+ 24 В реле	Номинальный ток (1)	0.5 А
		Пиковый ток	0.6 А
	+ 24 В датчики	Номинальный ток (1)	0.4 А
	(3)	Пиковый ток	0.6 А
	Общая	Номинальная	24 Вт
	мощность (4)	Пиковая	32 Вт
Изоляция	Диэлектрическая прочность	первичные цепи/ вторичные цепи	2500 Vrms 50/60 Гц

1. Номінальні струми відповідають споживанню 2/3 одночасно активних входів-виходів. Джерело живлення може нормально функціонувати без провалів на піковій потужності, що відповідає 100% активності входів-виходів.

2. Це значення показує, що кола живлення повинні витримувати пусковий струм 60 А. Це необхідно враховувати, коли кілька пристроїв будуть включатися одночасно, або для вибору пристроїв захисту.

3. Для джерел живлення змінного струму кількість давачів, що живляться від них, на напругу 24 В, не повинне перевищувати 100 для базового шасі. У випадку перевищення цієї кількості, необхідно підключити зовнішнє джерело живлення.

4. Загальна потужність не є сумою потужностей максимальної кількості виходів, що допускаються у конфігурації. Вона розраховується залежно від конфігурації, що відповідає оптимальному використанню ПЛК.

Джерела живлення постійного струму

Тип ПЛК			**TSX 37-10/21/22**
Первичные	Номинальное напряжение		24 В
цепи	Ограничения напряжения (включая пульсации)		19-30 В 19-34 В (3)
	Пиковое/пиковые пульсации		5 % от Un F = 90 Гц до 1 кГц
	Потребление тока		2 А
	Пусковой ток (2)		< 60 А
Вторичные	+ 5 В пост.тока	Номинальный ток (1)	2.8 А
цепи		Пиковый ток	3.2 А
	Общая	Номинальная	16 Вт
	мощность (4)	Пиковая	18 Вт
Изоляция	Диэлектрическая прочность	первичные цепи/ вторичные цепи	не изолирован

1.Номінальні струми відповідають споживанню 2/3 одночасно активних входів-виходів. Джерело живлення може нормально функціонувати без провалів на піковій потужності, що відповідає 100% активності входів-виходів.

2. Це значення показує, що кола живлення повинні витримувати пусковий струм 60 А. Це необхідно враховувати, коли кілька пристроїв будуть включатися одночасно, або для вибору пристроїв захисту.

3. 34 В на одну годину, для пристроїв батарейного живлення з зарядкою.

4. Загальна потужність не є сумою потужностей максимальної кількості виходів, що допускаються у конфігурації. Вона

розраховуэться выдповыдно до конфігурації, що відповідає оптимальному використанню ПЛК.

Захист джерел живлення

Джерела живлення вбудовані в TSX 37-10, TSX 37-21 та TSX 37-22, захищені від перевантажень і коротких замикань.

Короткі замикання і перевантаження в ланцюгах живлення давачів 24 В не впливають на інші кола. Напруга у колах живлення давачів 24 В відновлюється, коли причина збою зникає.

Службові сигнали

Якщо під час роботи напруга живлення ПЛК поза допустимих меж, генерується спеціальний сигнал (Power Fail - Збій живлення).

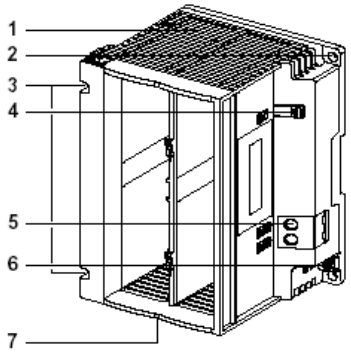
Джерело живлення програмного термінала

Напруга живлення + 5 В постійного струму, що подається ПЛК на термінальний порт, не забезпечує достатньої потужності для живлення кишенькового програмного термінала з дуже низьким споживанням (< 200 ма).

2.4 Локальне розширення входів-виходів

2.4.1 Міні-шасі розширення

Шасі розширення TSX RKZ 02 дозволяє додати до ПЛК два слота, у кожний з яких можна встановити або один стандартний, або два напівформатних модулі.



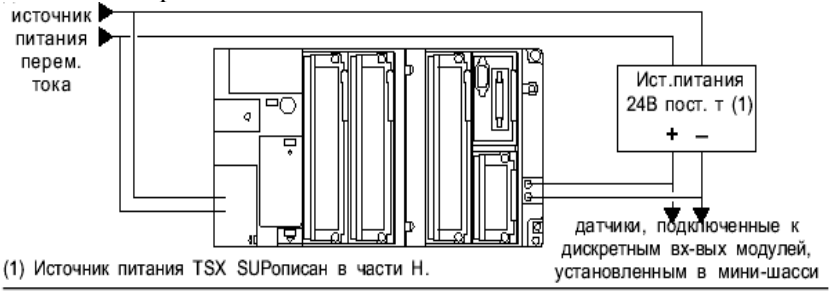
1. Шасі розширення з двома слотами.
2. Монтажні точки для міні-шасі.
3. Гвинти для кріплення міні-шасі до базового.
4. Індикаторна лампа, що показує наявність додаткової напруги 24 В (для релейних або аналогових модулів).
5. Контакти джерела живлення, закриті зйомною кришкою.
6. Клема заземлення.
7. Роз‘єм для підключення до базового ПЛК (об'єднуюча шина і продовження заземлення).

Примітка :

Для забезпечення ступеня захисту IP20, ви повинні закрити спеціальними кришками вільні слоти. Ці кришки не входять у комплект поставки і повинні бути замовлені окремо у кількості по 10 штук, тип TSX RKA 01.

2.4.2 Джерело живлення

Коли ПЛК TSX 37-10/21/22 живиться від джерела живлення змінного струму, останній не забезпечує напругу живлення 24 В постійного струму для мыні-шасі розширення. У цьому випадку, якщо в шасі розширення є релейні або аналогові модулі, додаткове джерело живлення 24 В постійного струму повинне бути підключене до відповідних контактів шасі розширення. Джерело живлення 24 В базового шасі забезпечує живлення входів-виходів шасі, які його вимагають, забезпечуючи споживання <=400 ма. Якщо планується більше споживання, то необхідно використати додаткове джерело живлення.



2.5 Представлення програмного забезпечення PL7

2.5.1 Представлення

Система **PL7 Junior** є програмним забезпеченням для роботи ПЛК TSX 37 і TSX/PMX/PCX 57 під керуванням Windows. Програмне забезпечення **PL7 Micro** може використовуватися тільки для програмування ПЛК TSX 37.

Програмне забезпечення **PL7 Pro** доповнює функції, які забезпечуються програмним забезпеченням PL7 Junior, можливістю створення функціональних блоків користувача – DFB (Derived Function Block), екранів роботи програми і функціональних модулів.

Програмне забезпечення **PL7 Prodyn** є інструментом розробки (контролю, діагностики, підтримки) для ПЛК TSX 37 і

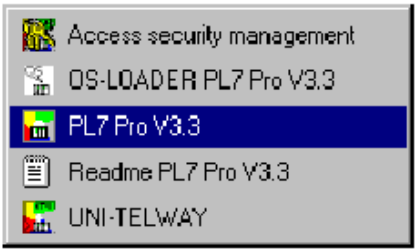
TSX/PMX/PCX 57. Дане програмне забезпечення не може використовуватися для створення або модифікації прикладної програми.

2.5.1.1 Запуск PL7

Меню **ПУСК**:

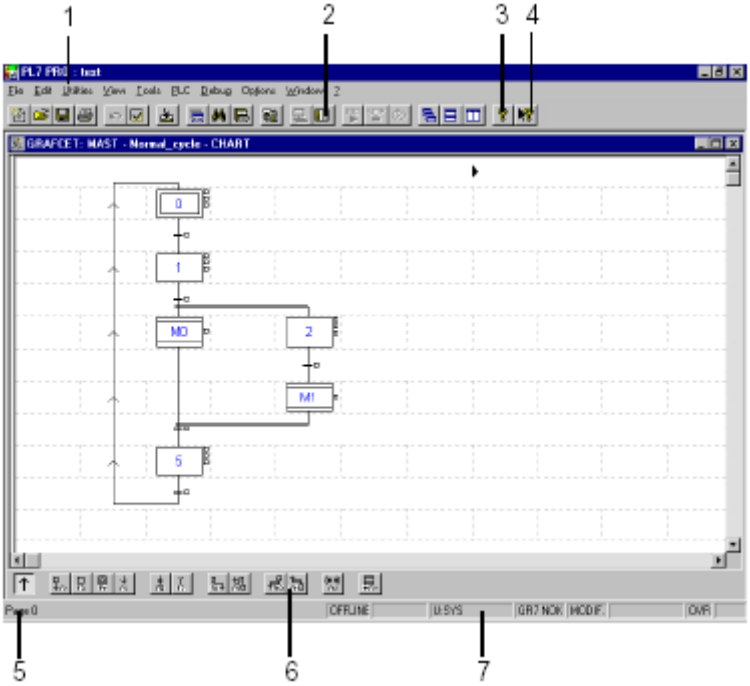
1. Виберіть групу **Програми.**
2. Виберіть групу **Modicon Telemecanique.**

Виберіть іконку **PL7.**



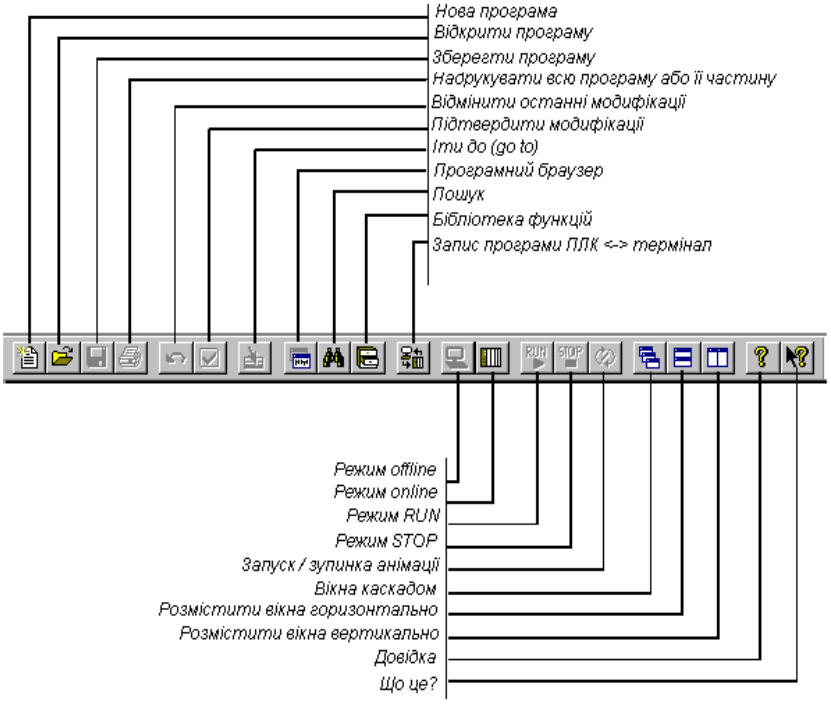
2.5.1.2 Стандартні елементи

Програмне забезпечення PL7 використовує віконний інтерфейс, який виглядає наступним чином:



1. Меню, яке надає доступ до всіх функцій програмного середовища PL7.
2. Панель інструментів, що забезпечує швидкий доступ до всіх стандартних функцій за допомогою мишки.
3. Довідка.
4. Що це?
5. Рядок стану.
6. Палітра графічних елементів.
7. Стан роботи.

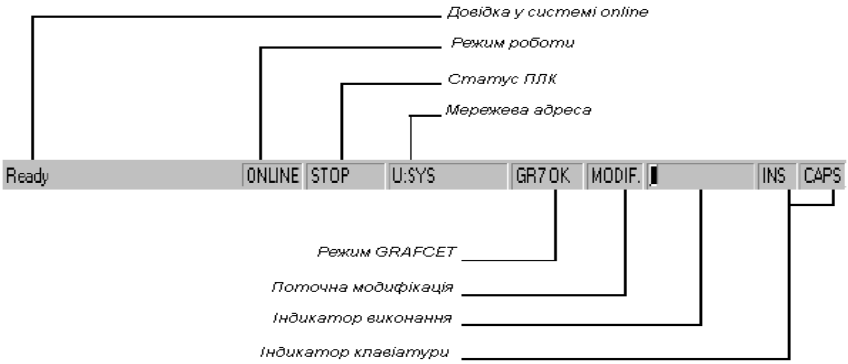
2.5.1.3 Панель інструментів



Дані функції також доступні через меню.

2.5.1.4 Рядок стану

Рядок стану, розміщений у нижній частині екрану, надає наступну інформацію:



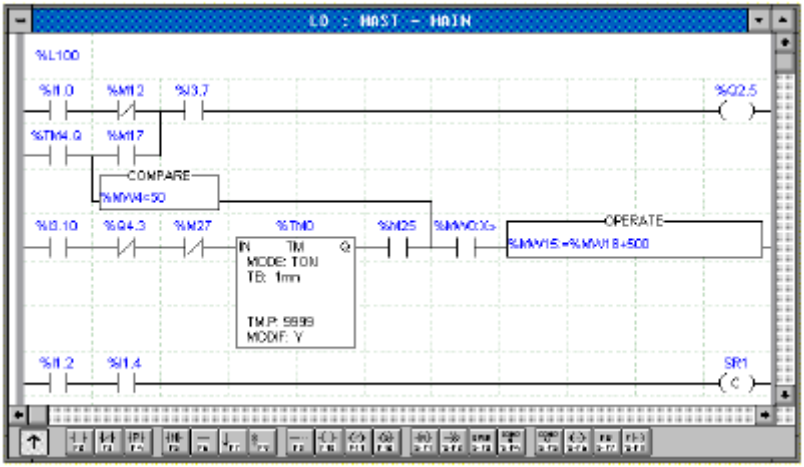
2.5.1.5 Функції програмного забезпечення PL7

	PL7 Micro	PL7 Junior	PL7 Pro	PL7 Pro Dyn
Programming	TSX Micro	TSX Micro and TSX Premium	TSX Micro and TSX Premium	no
Language				display
G7 CHART	yes	yes	yes	
MACROS	no	TSX Premium	TSX Premium	
LD	yes	yes	yes	
IL	yes	yes	yes	
ST	no	yes	yes	
Sections	yes	yes	yes	
Function modules	no	no	yes	
Debugging	yes	yes	yes	no
Adjustment	yes	yes	yes	yes
Diagnostics	no	yes	yes	yes
Runtime screens	no	no	creation/use	creation/use
DFB types	no	use	creation/use	no
Diagnostic DFB	no	no	TSX/PCX/PMX 57	display
Storage of symbols in the PLC		TSX Premium	TSX Premium	TSX Premium
Application doc. file	yes	yes	yes	yes

Програмне забезпечення **PL7** містить:

• графічну мову, мову Драбинкових діаграм Ladder (LD) для розшифровки діаграм Ladder, які є особливо підходящими для комбінаторної обробки. Вона містить основні графічні елементи: контакти, котушки та блоки. Числові обчислення можуть записуватися в операційні блоки. Палітра графічних елементів надає доступ до різних графічних елементів за допомогою мишки або клавіатури:

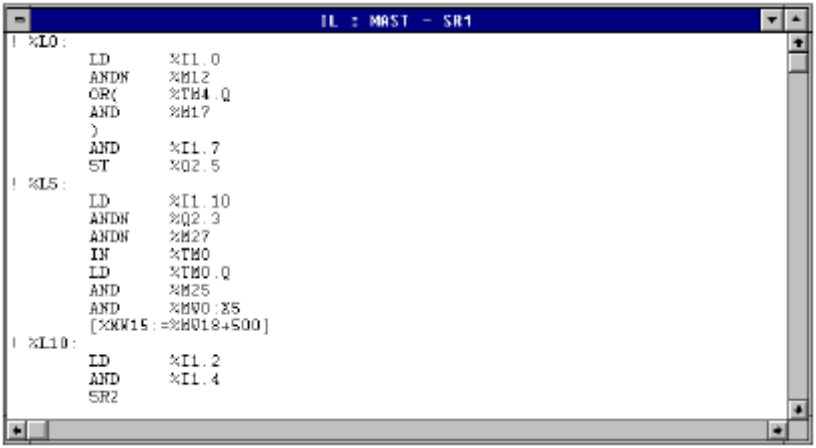
Ранг конструюється за допомогою вибору елемента з графічної панелі і розміщення його у певному місці сітки екрана. декілька рангів можуть бути розміщені в одному вікні.



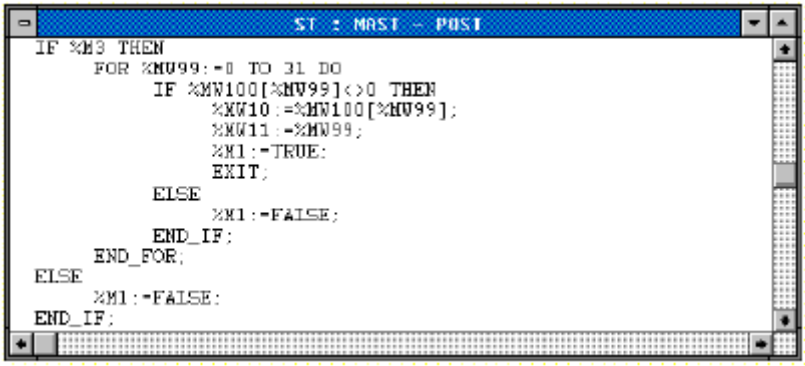
• Логічна мова (Boolean), мова Список команд (Instruction List (IL)), яка є «машинною» мовою для запису логічних і числових операцій обробки.

Редактор мови Список команд призначений для вводу команд та операндів за допомогою клавіатури. Форматування (вирівнювання інструкцій) здійснюється редактором автоматично. Операнди можуть вводитися та відображатися або у символьному форматі, або у форматі адреси. Ключові слова мови та коментарі

відображаються іншим кольором для полегшення читання програми.

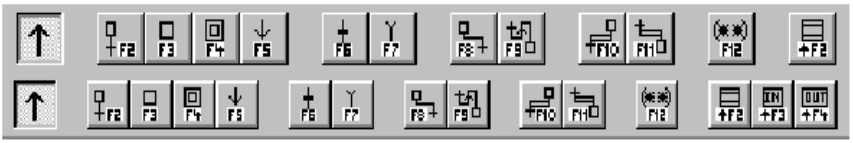


• Мова Структурований текст (Structured Text (ST)), що є однією з мов «обробки даних» і дозволяє робити структурований запис логічних і числових обчислень.

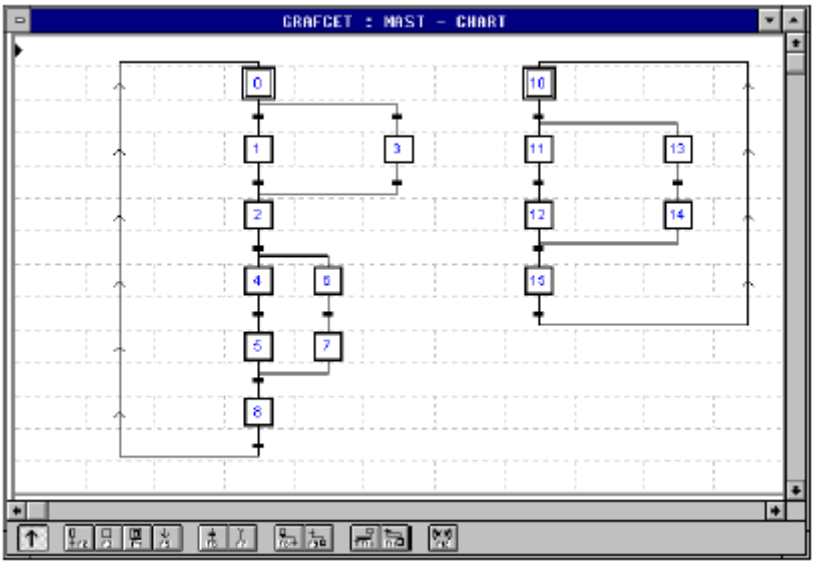


• Мова Графсет (Grafcet), що використовується для представлення операцій систем керування користувача у графічному та структурованому вигляді.

Редактор мови Grafcet має широкий набір інструментів для спрощення побудови програми. палітра графічних об‘єктів, які можна обирати за допомогою мишки або клавіатури (кроки, переходи, зв‘язки, конектори, мокрокроки тощо):



Нумерування кроків здійснюється автоматично. Програмний модуль мовою Grafcet може складатися із 8 Grafcet-сторінок максимум.



Дані мови використовують функціональні блоки з попередніми уставками (таймери, лічильники, і т.д.), які доповнюються спеціалізованими функціями, які визначаються програмою (аналоговий пристрій, комунікаційний, лічильний тощо), і просто спеціалізованими функціями (керування часом, символьний обробка і т.п.).

Об'єкти мови можуть замінятися символами шляхом використання редактора змінних або ж редактора програми в режимі реального часу.

Програмне забезпечення PL7 відповідає стандарту IEC 1131-3.

2.5.2 Однозадачна структура

У програмному забезпеченні ця структура приймається за замовчуванням. Вона містить у собі одне завдання - майстер-завдання.

Майстер-задача

Ця задача може бути або циклічною, як було сказано вище (установка за замовчуванням), або періодичною. Для циклічної операції задачі утворюють зв'язки одна з одною без будь-яких пауз. Для періодичної операції задачі з'єднуються одна з одною через обумовлений користувачем період.

2.5.3 Багатозадачна структура

Багатозадачна структура для ПЛК TSX 37 і TSX 57 дозволяє більш ефективне використання високопродуктивних прикладних програм реального часу шляхом прив'язки певної задачі до кожної прикладної програми. Кожна така програма управляється задачею.
Ці задачі є незалежними і виконуються «паралельно» головним процесором, який керує їхніми пріоритетами так само, як і виконанням.

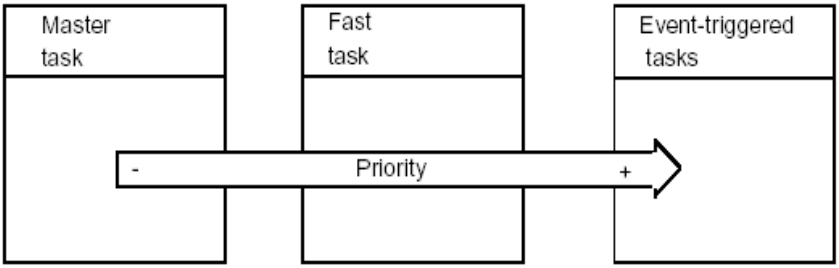
Головна мета цього типу структури полягає в:

• Оптимізації використання енергії, необхідної на обробку.
• Спрощенні розробки та налагодження. Кожне завдання пишеться і налагоджується незалежно від інших.
• Структуризації програми. Кожне завдання має унікальну функцію.
• Оптимізації працездатності.

Багатозадачна система пропонує майстер-задачу, швидку задачу, а також від 8 до 64 задач, які запускаються відповідно до подій (залежно від процесора).

Швидка задача

Швидка задача (необов'язкова), що є періодичною, використовується для виконання коротких операцій обробки з більш високим пріоритетом, ніж майстер-задача. Коли вона програмується, вона автоматично запускається системою під час старту. Швидка задача може також бути зупинена і потім знову запущена активізацією системного біта.

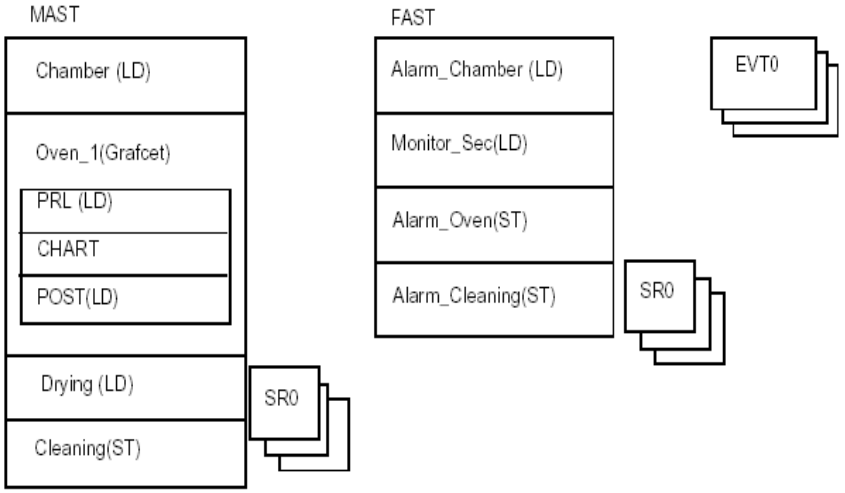


Задачі, що перемикають згідно подій

На відміну від задач, описаних вище, задачі цього типу не поєднуються з періодом. Вони запускаються по виклику, що видається певними модулями. Ці задачі мають найвищий пріоритет. Виконувані ними операції свідомо зроблені короткими, так щоб вони не перетиналися з виконанням інших завдань.

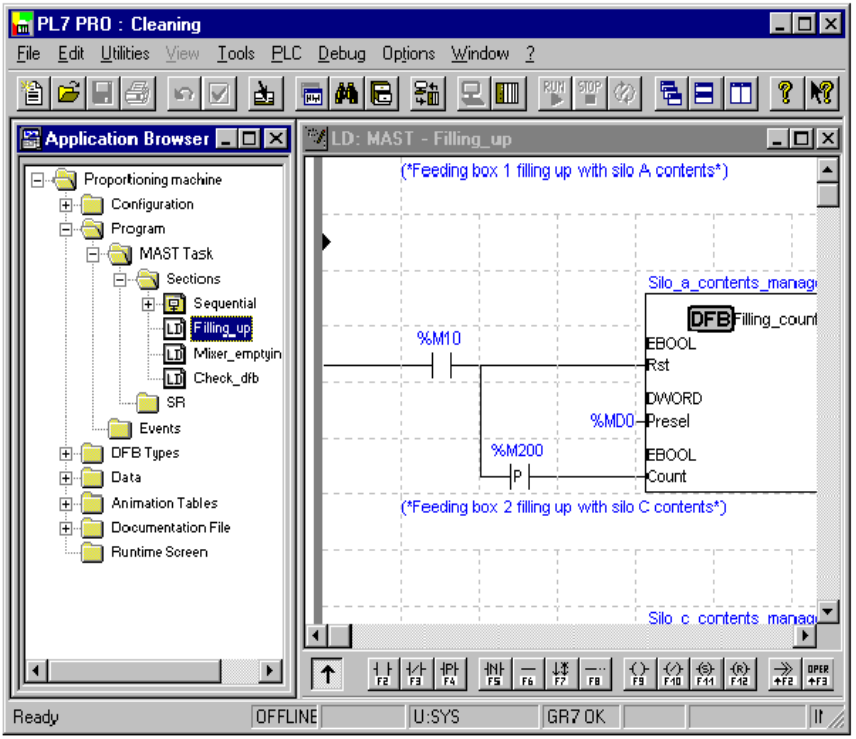
2.5.4 Структурне і модульне програмування

Програмні завдання PL7 складаються з декількох частин («секцій») і підпрограм. Кожна із цих секцій може програмуватися мовою, що найбільше підходить для виконання процесу.



Це розподіл на секції означає, що може використовуватися структурне програмування і програмний модуль може генеруватися або об'єднуватися без будь-яких труднощів.

Підпрограми також можуть викликатися з будь-яких секцій завдання, які їй належать, або з інших підпрограм цього ж завдання.

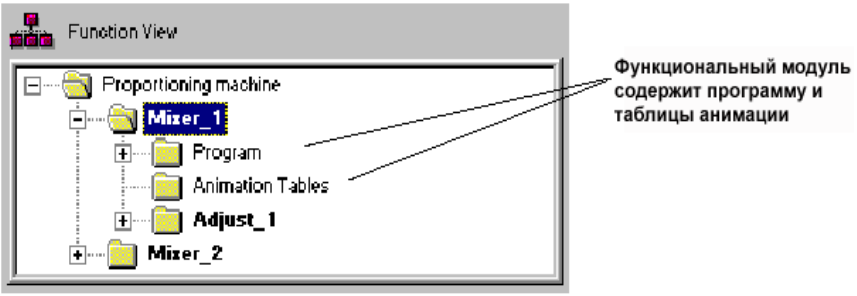


2.5.5 Структурування у якості функціональних модулів

Функціональний модуль - це група програмних елементів (секцій, подій, макро-кроків, таблиць анімації тощо), використовуваних для виконання функції контрольної системи.

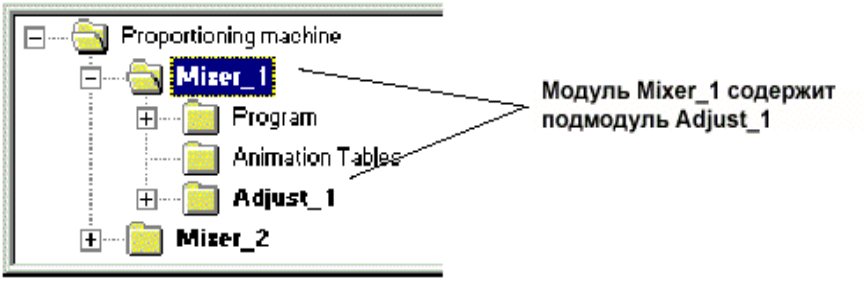
Функціональний модуль визначається встановленим числом атрибутів (ім'я, коментар, програмування, відповідні таблиці анімації, і т.п.).

Функціональний модуль містить програмну директорію, що складається з одного або більше модулів з кодом, і директорію для таблиць анімації.



Функціональний модуль може і сам бути поділений на функціональні модулі нижчого рівня, які виконують одну або більше підфункцій головної функції контрольної системи.

Тільки PL7 PRO є продуктом, що може використовуватися для установки функціональних модулів на ПЛК TSX/PMX/PCX57.



2.5.6 Символьне програмування

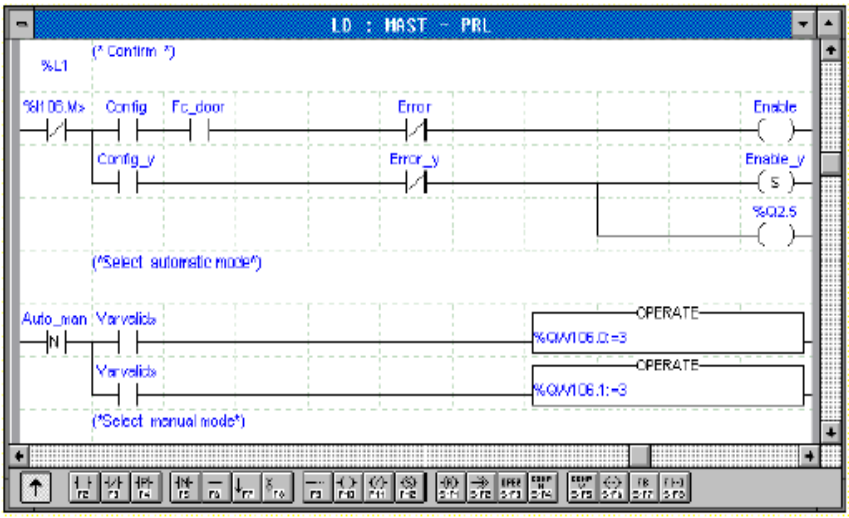
Користувач може ввести або відобразити об'єкти:

• або за їхньою адресою (наприклад: %Q2.5),

• або за буквеним рядком (максимум 32 літери), відомим як «символ» (наприклад, Fc_door).

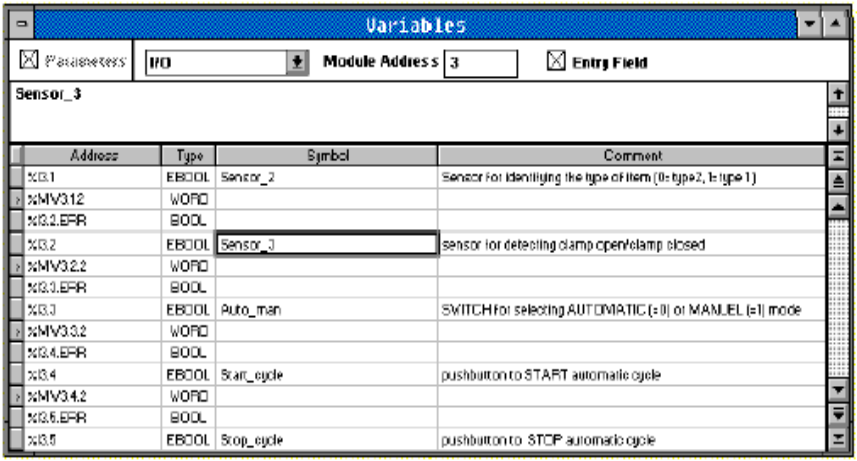
Примітка: Об'єкти, асоційовані з функціональними блоками користувача DFB, є винятково символьними об'єктами.

Приклад: символьне відображення об'єктів мови Ladder.



Адреси та символи в мові Ladder можуть відображатися одночасно.

Об'єкти мови можуть замінятися символами шляхом використання редактора змінних або редактора програми у режимі підключення.



Address	Type	Symbol	Comment
%I3.1	EBOOL	Sensor_2	Sensor for identifying the type of item (0= type2, 1= type 1)
%MW3.12	WORD		
%I3.2.ERR	BOOL		
%I3.2	EBOOL	Sensor_3	sensor for detecting clamp open/clamp closed
%MW3.2.2	WORD		
%I3.3.ERR	BOOL		
%I3.3	EBOOL	Auto_man	SWITCH for selecting AUTOMATIC (=0) or MANUEL (=1) mode
%MW3.3.2	WORD		
%I3.4.ERR	BOOL		
%I3.4	EBOOL	Start_cycle	pushbutton to START automatic cycle
%MW3.4.2	WORD		
%I3.5.ERR	BOOL		
%I3.5	EBOOL	Stop_cycle	pushbutton to STOP automatic cycle

Ця база даних символів, керована програмним редактором змінних, є глобальною в межах станції ПЛК.

Примітка:

Деякі модулі допускають символи для автоматичного розміщення на тому ж місці, що й асоційовані з ними об'єкти.

Символи та коментарі, які зберігаються в ПЛК TSX Premium
• Функція

Символи і коментарі можуть зберігатися в ПЛК TSX Premium (TSX/PMX/PCX 57202,57302, 57402, 57452 V3.3) у відповідності з наступними типами платами пам'яті:

• Плата пам'яті на 128 Кслів: TSXMRP 2128P
• Плата пам'яті на 256 Кслів: TSXMRP 3256P

Коли програма зберігається в ПК, створюється файл із розширенням *.STX. Зображення програми у формі файлу *.STX включає також символи і коментарі.

Функція «Зберігання символів і коментарів у ПЛК TSX Premium PLC" забезпечує користувачеві PL7 ту ж невимушеність програмування та роботи з відповідними символами і коментарями, як чи доступно зображення програми у формі STX на ПК, чи ні.

2.5.7 Команди PL7

Всі мови PL7 використовують той самий набір команд.

З метою спрощення вони розділені на 2 набори: основні (базові) команди та додаткові команди.

Основні команди

Вони включають основні логічні команди, функціональні блоки, арифметичні та логічні команди над цілими (integer).

Додаткові команди

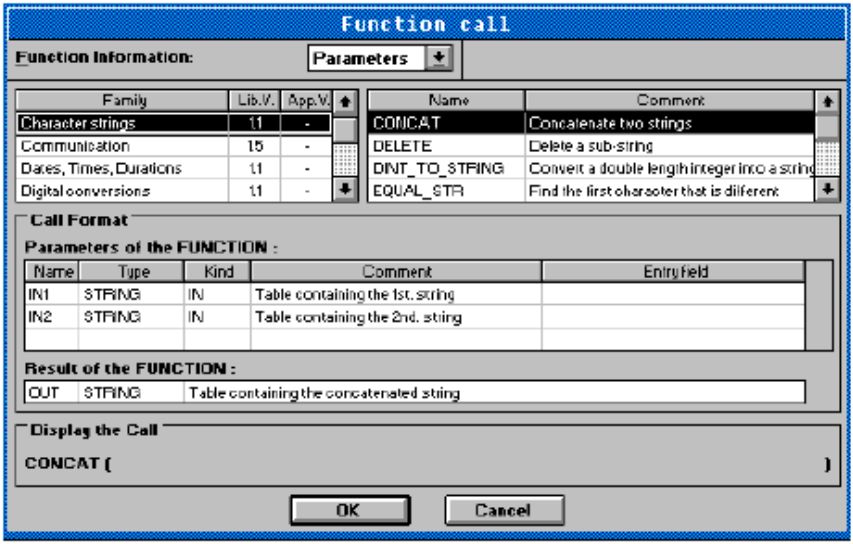
Вони включають команди, які необхідні при більш складному програмуванні.

Вони поділені на 2 типи:

• Мова PL7: це збільшує продуктивність обробки мови через певні функції (маніпуляція буквеними рядками, керування часом і т.п.),
• Додатки: вони надають функції, які при обробці залежать від програми. Наприклад, функції для комунікаційної програми:
- PRINT для відправлення повідомлення зі стандартних буквених рядків на термінал або принтер.
- SEND для відправлення повідомлення програмі.
- PID : функція ПІД-контролю.

Довідка при вході у функцію

При вході вікно довідки надає доступ до всіх функцій мови. Це вікно доступно весь час, у тому числі під час програмування.

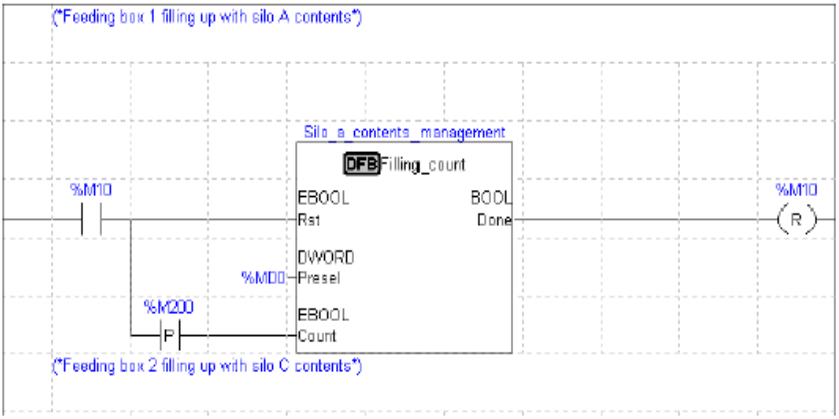


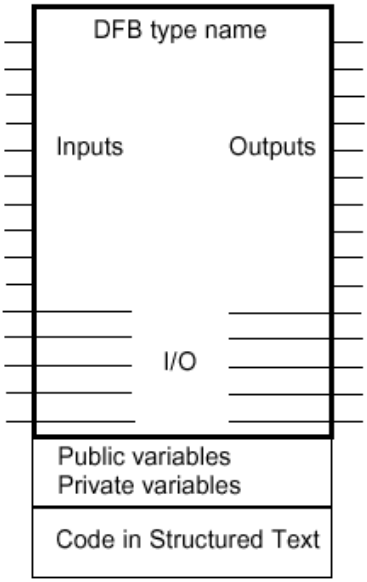
2.5.8 Функціональні блоки користувача

Програмне забезпечення PL7 Pro може використовуватися для створення функціональних блоків користувача DFB для ПЛК Premium.

Блоки DFB створюються мовою Структурованого тексту (Structured Text) і потім можуть використовуватися в секції або підпрограмі кожної з використовуваних мов (вони також можуть використовуватися із програмним забезпеченням PL7 Junior).

Приклад DFB, використовуваного в мові Ladder:





Головними компонентами DFB є:

- ім'я;
- вхідні і вихідні параметри;
- змінні типу public і private;
- код мовою Структурованого Тексту (Structured Text);

DFB може мати максимум 15 входів та/або входів/виходів і 15 виходів та/або входів/виходів. Створений одного разу кожен DFB може використовуватися в програмі кілька разів.

Розробник програмує модель DFB (названу «типом DFB»), і при кожному використанні користувач визначає ім'я зразка, використовуючи редактор змінних або вхідне вікно довідки для обраної мови.

2.6 Конфігурація програмного термінала (комп‘ютера)

2.6.1 Мінімальна конфігурація

Processor	486DX5133MHz
System	Windows 95
RAM	32Mb
Drives	Hard disk 50 Mb for the software 25 Mb for the temporary directories Floppy disk
Ports	COM serial port available for connection to the PLC (COM1 to COM4) Parallel printer port (LPT1 to LPT4)
Monitor	VGA

2.6.2 Типова конфігурація

Processor	Pentium 166	
Systems	Windows 95 / 98	Windows NT 4.0
RAM	32Mb	48Mb
Drives	Hard disk 50 Mb for the software 25 Mb for the temporary directories CD-ROM Floppy disk	
Ports	COM serial port available for connection to the PLC (COM1 to COM4) Parallel printer port (LPT1 to LPT4)	
Monitor	At least VGA (SVGA with color management on 24 bits is recommended)	

2.7 Методи керування витратою води у трубопроводах

Керування витратою води у трубопроводах є важливим аспектом сучасної водопостачальної системи. Оптимальне використання водних ресурсів не лише забезпечує надійне постачання, але й сприяє економії води та зменшенню витрат. Існує кілька методів керування витратою води, кожен з яких має свої переваги та застосування.

Першим методом є використання механічних регуляторів витрати, таких як клапани та заслінки. Вони дозволяють регулювати потік води вручну або автоматично, залежно від

налаштувань. Ці пристрої ефективно працюють у великих системах водопостачання, де потрібне точне керування потоком. Автоматичні клапани можуть реагувати на зміни тиску та витрати, забезпечуючи стабільність системи.

Другим методом є застосування електронних лічильників води з можливістю дистанційного моніторингу та керування. Такі системи використовують датчики для вимірювання витрати води в реальному часі та передають дані до центрального контролера. Це дозволяє оперативно реагувати на зміни витрати, виявляти витоки та оптимізувати споживання води. Дистанційне керування дає змогу регулювати витрату води в різних частинах системи з мінімальними затратами часу.

Третій метод включає впровадження інтелектуальних систем керування водопостачанням, які використовують алгоритми машинного навчання та штучного інтелекту. Такі системи здатні аналізувати великі обсяги даних, прогнозувати витрати води та оптимізувати роботу всіх елементів водопостачальної мережі. Вони можуть автоматично налаштовувати режими роботи насосів та клапанів, враховуючи поточні потреби та умови.

Важливою складовою керування витратою води є також регулярне обслуговування та модернізація трубопроводів. Своєчасна заміна зношених ділянок, виявлення та усунення витоків допомагають підтримувати ефективність системи. Використання сучасних матеріалів та технологій у будівництві трубопроводів сприяє зниженню втрат води та підвищенню надійності.

Таким чином, ефективне керування витратою води у трубопроводах потребує комплексного підходу, що включає використання сучасних технологій, регулярне обслуговування систем та підвищення обізнаності населення. Лише завдяки цьому можна забезпечити стабільне водопостачання та збереження цінних водних ресурсів для майбутніх поколінь.

3. Програма роботи

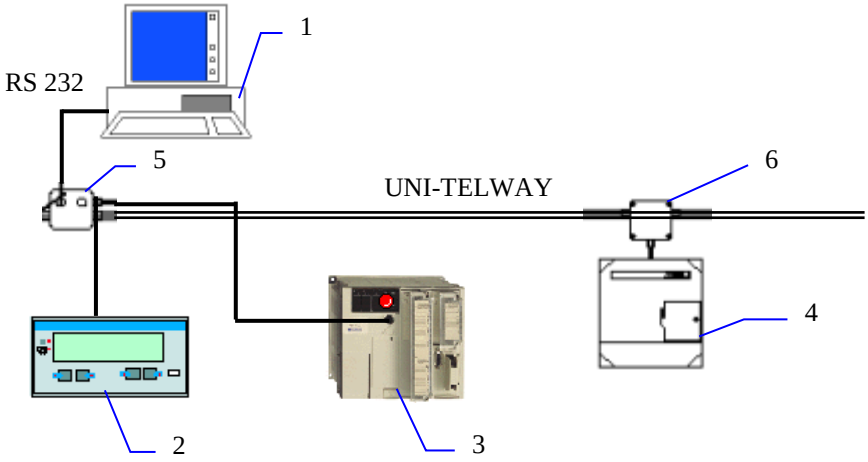
1. Вивчити будову та характеристики промислового логічного контролера (ПЛК) Schneider Micro TSX 37-22.
2. Дослідити правила підключення ПЛК Schneider Micro TSX 37-22.

3. Вивчити види модулів введення-виведення для ПЛК Schneider Micro TSX 37-22.
4. Дослідити правила адресації вхідних і вихідних каналів.
5. Вивчити правила написання програм для промислового контролера Schneider Micro TSX 37-22 у середовищі PL7 PRO.
6. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу регулювання витрати води у трубопроводі.
7. Створити другий рівень системи контролю і керування технологічним процесом.

4. Опис лабораторного обладнання

1. Персональний комп‘ютер.
2. Операційна система Windows.
3. Лабораторний стенд з ПЛК Schneider Micro TSX 37-22.
4. Середовище програмування ПЛК Schneider Micro TSX 37-22 PL7 PRO, середовище програмування операторської панелі Magelis XBT-L-1000.

Лабораторний стенд:



1- АРМ оператора; 2 - операторська панель Magelis XBT P 012010;

3 - ПЛК Schneider Micro TSX 37-22; 4 - частотний перетворювач Altivar 58; 5 - адаптер відгалуженняTSX P ACC01; 6 - розгалуджувач TSX SCA 62.

5. Порядок виконання роботи

1. Вивчити будову та характеристики промислового логічного контролера (ПЛК) Schneider Micro TSX 37-22.
2. Дослідити правила підключення ПЛК Schneider Micro TSX 37-22.
3. Визначити тип модуля дискретних входів, вставленого у третій слот контролера.
4. Визначити правила під‘єднання до модуля дискретних входів.
5. Визначити тип модуля дискретних виходів, вставленого у четвертий слот контролера.
6. Визначити правила під‘єднання до модуля дискретних виходів.
7. Вивчити правила з‘єднання порту вбудованих аналогових входів-виходів контролера кабелем ABE7 CPA01 з клемником Telefast ABE7 CPA01, вивчити призначення його клем. Вивчити правила під‘єднання давачів до клемника Telefast.
8. Після перевірки викладачем правильності усіх з‘єднань ввімкнути живлення стенду. Дослідити роботу дисплею контролера. Замикаючи вхідні контакти модуля дискретних входів за допомогою під‘єднаних раніше кнопок спостерігати за змінами на дисплеї.
9. Під‘єднати контролер Schneider Micro TSX 37-22 до COM-порту комп‘ютера за допомогою комунікаційного кабелю TSX PCX 1031.
10. Запустити середовище PL7 PRO.
11. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу регулювання витрати води у трубопроводі, використовуючи мови програмування драбинкових діаграм, послідовних інструкцій та структурованого тексту.

12. Створити анімаційну таблицю для відслідковування значень використаних у програмі змінних. Наряду з анімаційною таблицею використати операторську панель Magelis.
13. Створити вікно супервізора (Run-Time screen) засобами програмного забезпечення PL7 PRO, в якому намалювати ФСА технологічного процесу.
14. Записати програму у контролер. Для цього виконати наступні дії:
    - виконати команду Conect;
    - вибрати напрям запису програми PC -> PLC;
    - перевести контролер у режим RUN.
15. Провести моніторинг роботи програми за допомогою анімаційної таблиці, вікна супервізора.

<u>Опис технологічного процесу та вимоги до програми:</u>

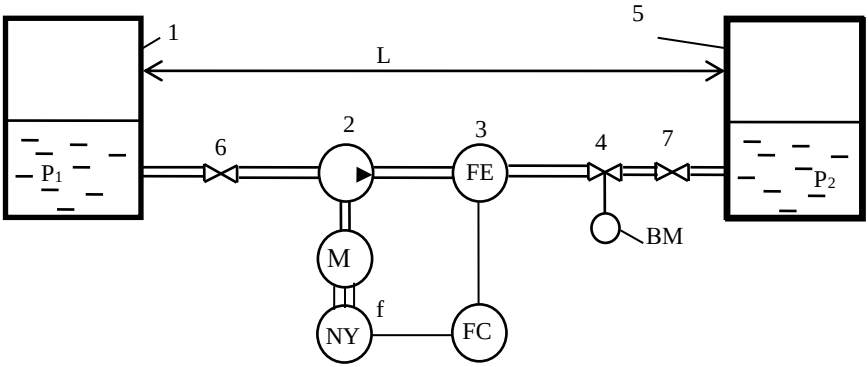


Рис. 4.1 ФСА регулювання витрати:

1, 5- технологічні апарати; 2- насос; 3- діафрагма; 4- регулюючий клапан; 6, 7- запірні вентилі; FE – давач витрати, FC – регулятор витрати, NY^f – частотний перетворювач, M – двигун.

Витрата води у трубопроводі вимірюється давачем витрати і регулюється шляхом зміни продуктивності насоса за допомогою перетворювача частоти. Цей спосіб є економічно вигіднішим, ніж керування витратою шляхом байпасування або зміни гідравлічного опору регулюючого клапану 4. При частотному керуванні витратою

можливі 2 режими: плавне керування аналоговим сигналом (керуючий сигнал подається на аналоговий вхід частотного перетворювача) та ступінчате керування дискретними сигналами (керуючі сигнали подаються на дискретні входи частотного перетворювача, кожен з яких відповідатиме за певну частоту напруги живлення насоса).

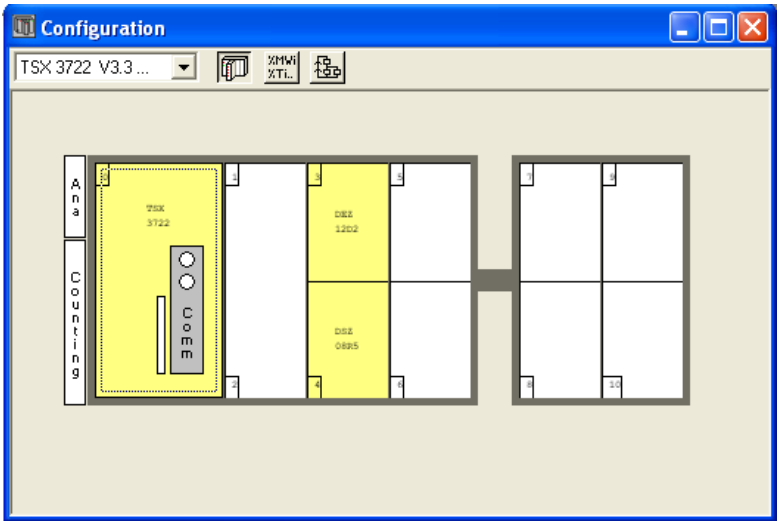


Рис. 4.2 Підключення та конфігурування модулів вводу-виводу

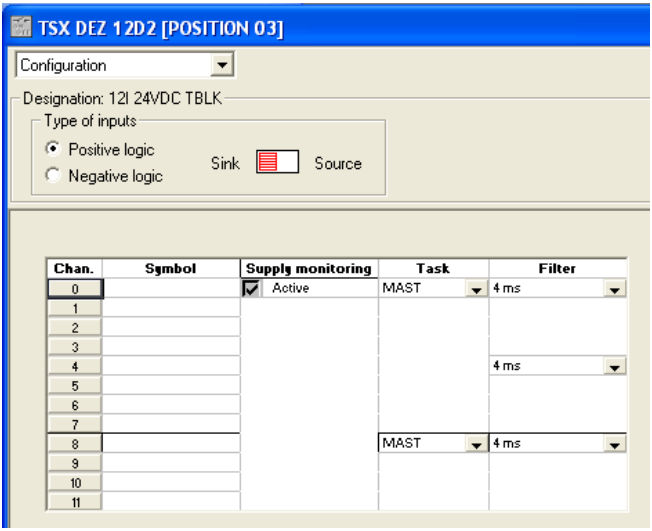


Рис. 4.3 Модуль дискретного вводу у слоті №3

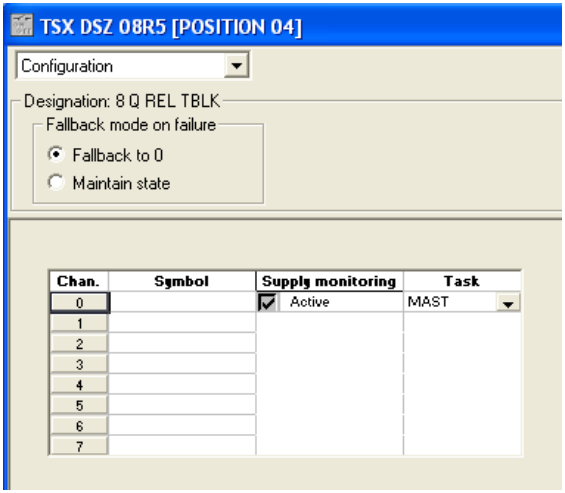


Рис. 4.4 Модуль дискретного виводу у слоті №4

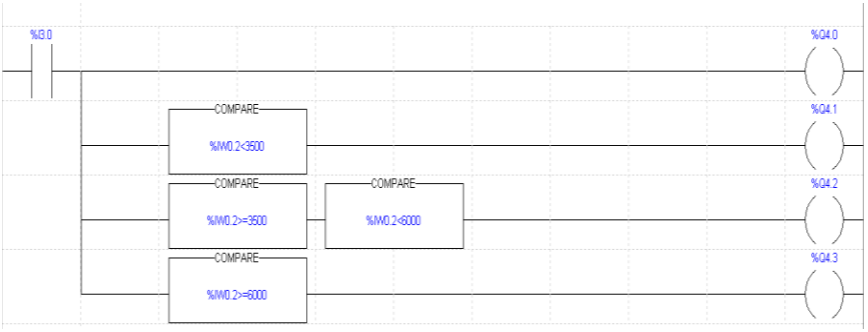


Рис. 4.5 Програма мовою LD

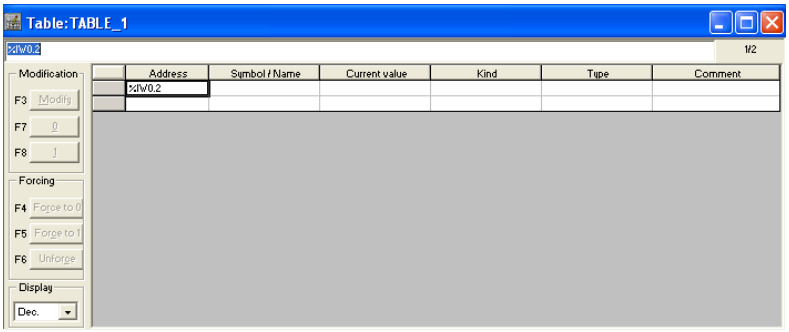


Рис. 4.6 Анімаційна таблиця

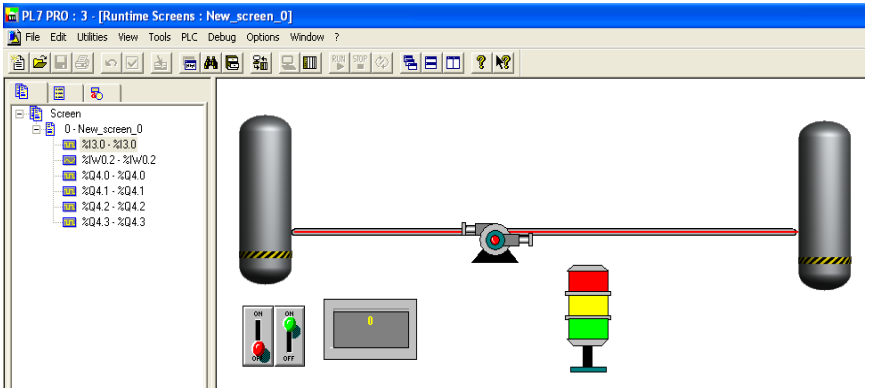


Рис. 4.7 Екран візуалізації

6. Контрольні запитання

1. Яка будова та основні характеристики промислового контролера Schneider Micro TSX 37-22?
2. Які є типи дискретних входів-виходів у контролера Schneider Micro TSX 37-22?
3. Назвіть типи аналогових входів-виходів контролера Schneider Micro TSX 37-22.
4. Які є методи підрахунку імпульсів за допомогою контролера Schneider Micro TSX 37-2?
5. Які комунікаційні можливості має промисловий контролер Schneider Micro TSX 37-22?
6. Яким чином створити нову програму у середовищі програмування PL7 PRO?
7. Якими мовами можна запрограмувати ПЛК Schneider Micro TSX 37-22? Які їх особливості та можливості?
8. Для чого призначені анімаційні таблиці?
9. Яким чином створити другий рівень автоматизованої системи керування технологічним процесом (АСК ТП) у середовищі PL7 PRO?
10. Яким чином записати створену програму у пам‘ять промислового контролера Schneider Micro TSX 37-22 та запустити її на виконання?

Робота №5. Розробка і дослідження системи контролю та керування процесом сушіння сипучих матеріалів з використанням ПІД регулятора на базі ПЛК Schneider Micro TSX 37-22.

Робота №6. Розробка і дослідження системи контролю та керування технологічним процесом з використанням функціональних блоків на базі ПЛК Schneider Micro TSX 37-22.

1. Мета роботи

Навчитися використовувати функціональні блоки при програмуванні промислового контролера Schneider Micro TSX 37-22 мовами драбинкових діаграм LD, послідовних інструкцій IL, структурованого тексту ST у середовищі програмування PL7 PRO. Навчитися розробляти дворівневі системи контролю та керування на базі Schneider Micro TSX 37-22.

2. Теоретичні відомості

2.1 Створення програмного забезпечення мовами LD, IL та ST з використанням стандартних функціональних блоків для керування технологічним процесом на базі ПЛК Schneider Micro

2.1.1 Загальні відомості про функціональні блоки

У середовищі PL7 PRO доступні 6 типів функціональних блоків:

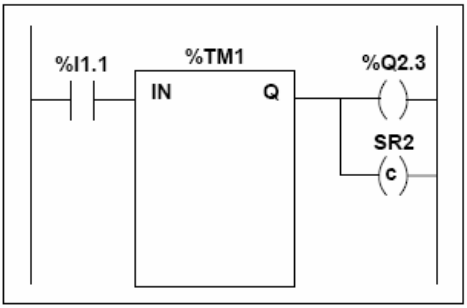
Таблиця 5.1

Тип блока	Позначення	Max TSX 37	Max TSX 57
Таймер	Timer %TMi	64 (*)	255 (*)
Лічильник	Up/Down counter %Ci	32	255
Моностабільний блок	Monostable %MNi	8	255
Регістр	Register %Ri	4	255
Барабанний контролер	Drum controller %DRi	8	255
Таймер (Серія 7)	Timer (Series 7) %Ti	64 (*)	255 (*)

(*) Загальна сума %TMi + %Ti таймерів у програмі не повинна перевищувати 64 для контролера TSX 37 та 255 для контролера TSX 57.

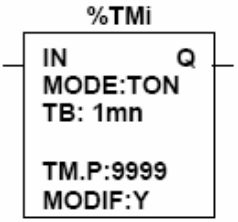
Кожен блок містить:

- Входи (наприклад, IN), які використовуються для керування блоком;
- Виходи (наприклад, Q), які вказують на стан блоку. Кожному виходу ставиться у відповідність вихідний біт (наприклад, %TM1.Q), стан якого можна протестувати у програмі. Крім того, кожен вихід може контролювати одну або декілька котушок (наприклад, %Q2.3 і SR2).



- Внутрішні параметри, які використовуються для налаштування функціональних блоків під вимоги програми (наприклад, уставка – preset, базовий час – time base).

2.1.2 Таймер (Timer %TMi)



Таймер може працювати у трьох режимах:

- **TON –** режим затримки включення (on-delay).
- **TOF** – режим затримки виключення (off-delay).
- **TP** – режим для генерації імпульсів певної тривалості (monostable).

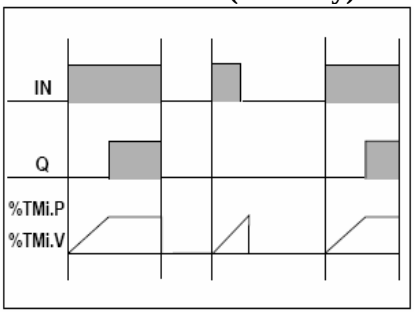
Таблиця 5.2

Характеристики таймера %TMi

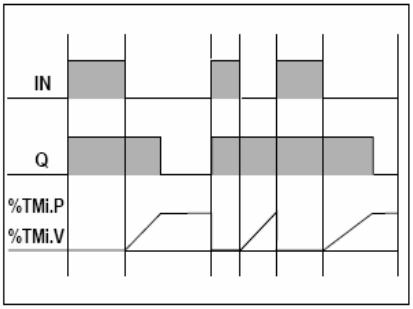
Номер таймера (Timer number)	%TMi	[0, 63] для TSX 37; [0, 254] для TSX 57
Режим роботи (Mode)	TON TOF TP	on-delay (за замовчуванням) off-delay monostable
Базовий час (Time base)	TB	1 min, 1 s, 100 ms, 10 ms (max 16 таймерів).
Поточне значення (Current value)	%TMi.V	Слово, яке нарощується від 0 до %TMi.P, коли таймер запущений. Його можна прочитати і протестувати, але не можна записати програмно.
Значення уставки (Preset value)	%TMi.P	0<=%TMi.P<=9999. Слово, яке можна прочитати, протестувати, записати програмно. За замовчуванням %TMi.P=9999. Результуючий інтервал часу t=%TMi.P*TB.
Доступ з термінала (Adjust via the terminal – MODIF)	Y/N	Y – значення уставки %TMi.P можна змінювати. N - значення уставки %TMi.P змінювати не можна.
Вхід (Input)	IN	Таймер запускається при подачі імпульсу на вхід IN.
Вихід (Otput)	Q	Біт %TMi.Q встановлюється в 1 в залежності від режиму роботи таймера.

Роботу таймера у різних режимах відображають наступні часові діаграми.

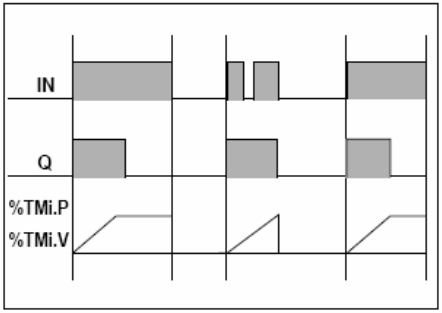
Режим TON (on-delay)



Режим TOF (off-delay)



Режим TP (monostable)

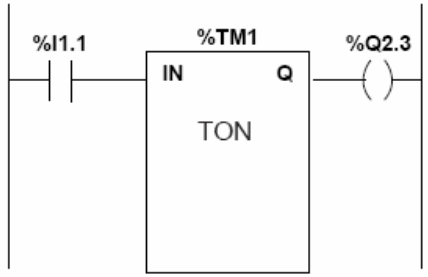


Конфігурація таймера %TMi

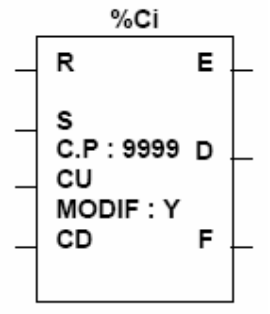
Для кожного таймера, використаного у програмі, необхідно вказати наступні внутрішні параметри:

- Режим роботи (Mode): TON, TOF, TP.
- Базовий час (Time base - TB): 1 min, 1 s, 100 ms, 10 ms.
- Значення уставки (Preset value - %TMi.P): від 0 до 9999.
- Дозвіл на доступ з термінала (Adjust via the terminal – MODIF): Y або N.

Приклади програмування таймера %TMi

<table>
<tr><td>Мова драбинкових діаграм (LD)
 
</td><td>Мова списку інструкцій(IL)
LD %I1.1
IN %TM1
LD %TM1.Q
ST %Q2.3</td></tr>
<tr><td>Мова структурованого тексту (ST)
IF RE %I1.1 THEN
START %TM1 ;
ELSIF FE %I1.1 THEN
DOWN %TM1 ;
END_IF ;
%Q2.3 := %TM1.Q ;</td><td>Команда START %TMi генерує імпульс по передньому фронту на вході таймера IN. Команда DOWN %TMi генерує імпульс по спадаючому фронту на вході таймера IN.</td></tr>
</table>

2.1.3 Лічильник (Up/Down counter %Ci)



Даний функціональний блок використовується для підрахунку імпульсів.

Таблиця 5.3

Характеристики лічильника %Ci

Номер лічильника (Counter number)	%Ci	[0, 31] для TSX 37; [0, 254] для TSX 57
Поточне значення (Current value)	%Ci.V	Слово, яке інкрементується або декрементується залежно від входів CU та CD. Його можна прочитати і протестувати, але не можна записати програмно.

Значення уставки (Preset value)	%Ci.P	0<=%Ci.P<=9999. Слово, яке можна прочитати, протестувати, записати програмно. За замовчуванням %Ci.P=9999.
Доступ з термінала (Adjust via the terminal – MODIF)	Y/N	Y – значення уставки %Ci.P можна змінювати. N - значення уставки %Ci.P змінювати не можна.
Вхід Reset (Reset Input)	R	При R=1 %C.V=0.
Вхід Preset (Preset Input)	S	При S=1 %C.V = %C.P.
Лічильний вхід сумуючий (Upcount input)	CU	Інкременує %C.V під час надходження імпульсу по наростаючому фронту.
Лічильний вхід віднімаючий (Downcount input)	CD	Декременує %C.V під час надходження імпульсу по наростаючому фронту.
Underflow output	E (Empty)	Біт %C.E=1, коли під час процесу віднімання значення %C.V змінює своє значення з 0 на 9999.
Preset reached output	D (Done)	Біт %C.D=1, якщо %C.V = %C.P.
Overflow output	F (Full)	Біт %C.F=1, коли під час процесу додавання значення %C.V змінює своє значення з 9999 на 0.

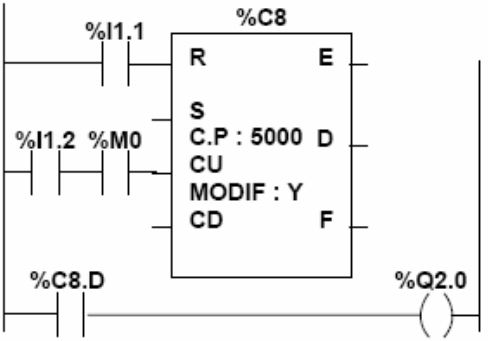
Конфігурація лічильника %Ci

Для кожного лічильника, використаного у програмі, необхідно вказати наступні внутрішні параметри:

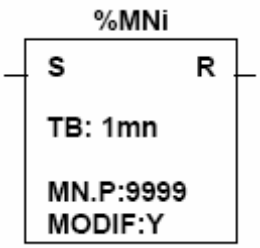
- Значення уставки (Preset value - %Ci.P): від 0 до 9999.
- Дозвіл на доступ з термінала (Adjust via the terminal – MODIF): Y або N.

Приклади програмування лічильника %Ci

Кожен імпульс на вході %I1.2 (коли внутрішній біт %M0=1) інкрементує лічильник %C8. Вхід %I1.1 призначений для скидання лічильника в нуль.

<table>
<tr>
<td>Мова драбинкових діаграм (LD)
 
</td>
<td>Мова списку інструкцій(IL)
<pre>
LD %I1.1
R %C8
LD %I1.2
AND %M0
CU %C8
LD %C8.D
ST %Q2.0
</pre></td>
</tr>
<tr>
<td>Мова структурованого тексту (ST)
<pre>
IF %I1.1 THEN
    RESET %C8 ;
END_IF ;
%M1 := %I1.2 AND %M0 ;
IF RE %M1 THEN
    UP %C8 ;
END_IF ;
%Q2.0 := %C8.D;
</pre></td>
<td>Команда RESET %Ci призначена для скидання лічильника в нуль. Команда PRESET %Ci встановлює лічильник у значення уставки. Команди UP %Ci та DOWN %Ci інкрементують та декрементують відповідно поточне значення лічильника. Входи CU та CD лічильника скидаються при використанні команд UP і DOWN. Тому потрібно контролювати передній фронт імпульсу на цих входах.</td>
</tr>
</table>

2.1.4 Моностабільний блок (Monostable %MNi)



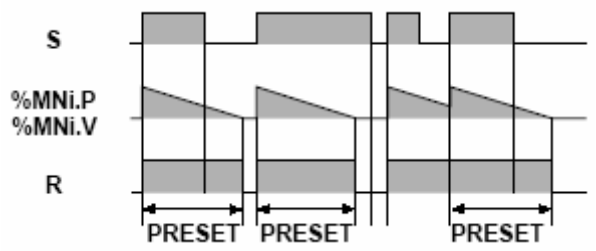
Моностабільний функціональний блок використовується для генерації імпульсів певної тривалості.

Таблиця 5.4

Характеристики моностабільного блоку %MNi

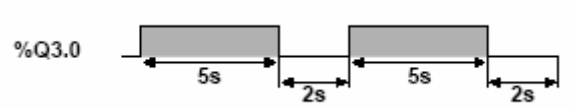
Номер (Number)	%MNi	[0, 7] для TSX 37; [0, 254] для TSX 57
Базовий час (Time base)	TB	1 min, 1 s, 100 ms, 10 ms (1 min за замовчуванням).
Поточне значення (Current value)	%MNi.V	Слово, яке зменшується від %MNi.P до нуля. Його можна прочитати і протестувати, але не можна записати програмно.
Значення уставки (Preset value)	%MNi.P	0<=%MNi.P<=9999. Слово, яке можна прочитати, протестувати, записати програмно. Тривалість імпульсу (PRESET) дорівнює %MNi.P*TB.
Доступ з термінала (Adjust via the terminal – MODIF)	Y/N	Y – значення уставки %MNi.P можна змінювати. N - значення уставки %MNi.P змінювати не можна.
Вхід “Старт” (Start Input)	S (Start)	При надходженні переднього фронту імпульсу на вхід S %MN.V = %MN.P. Далі поточне значення %MNi.V зменшується до нуля.
Моностабільний вихід	R (Running)	Біт %MNi.R=1, коли %MNi.V>0. %MNi.R=0, якщо %MNi.V=0.

Принцип роботи моностабільного функціонального блоку відображає наступна часова діаграма.



Програмування і конфігурація моностабільного блоку %MNi

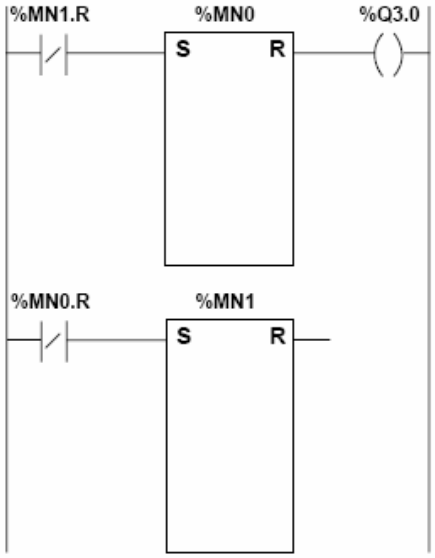
Нехай необхідно розробити програму для генерації імпульсів певної тривалості, причому тривалість імпульсу відмінна від тривалості паузи між двома сусідніми імпульсами.



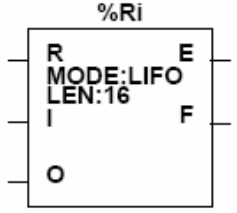
Для вирішення такої задачі необхідно використати 2 моностабільних блока. Блок %MN0 буде відповідати за імпульс, блок %MN1 – за паузу. Для даних функціональних блоків задамо такі значення внутрішніх параметрів:

- TB=100 ms;
- %MN0.P=50, %MN1.P=20;
- MODIF: Y.

Мова драбинкових діаграм (LD)	Мова списку інструкцій(IL)
	LDN %MN1.R ANDN %Q3.0 **S %MN0** LD %MN0.R ST %Q3.0 LDN %MN0.R **S %MN1**

<table>
<tr>
<td> 
</td>
<td></td>
</tr>
<tr>
<td>Мова структурованого тексту (ST)
%M0:=NOT %MN1.R ;
IF RE %M0 THEN
START %MN0 ;
END_IF ;
%Q3.0 := %MN0.R ;
%M1 := NOT %MN0.R ;
IF RE %M1 THEN
START %MN1 ;
END_IF ;</td>
<td>Команда START %MNi використовується для запуску моностабільного блоку. Дана команда генерує передній фронт імпульсу на вході S блоку. Тому необхідно контролювати наявність імпульсу оператором IF при запуску моностабільного блоку.</td>
</tr>
</table>

2.1.5 Регістр (Register %Ri)



Регістр виконує функцію пам‘яті, яка може містити до 255 слів по 16 бітів кожне. Регістр може працювати у двох режимах:

- черга (FIFO – first in, first out) – перший ввійшов, перший вийшов;
- стек (LIFO – last in, first out) – останній прийшов, перший вийшов.

Таблиця 5.5

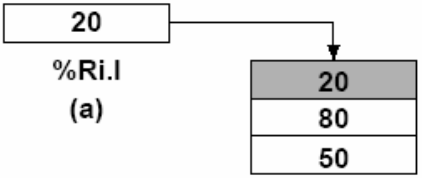
Характеристики регістра %Ri

Номер регістра (Register number)	%Ri	[0, 3] для TSX 37; [0, 254] для TSX 57
Режим (Mode)	FIFO LIFO	Черга Стек (за замовчуванням)
Довжина (Length)	LEN	Число 16-бітних слів (від 1 до 255), яке може бути збережене у регістрі.
Вхідне слово (Input word)	%Ri.I	Вхідне слово регістра. Його можна прочитати, протестувати, записати.
Вихідне слово (Output word)	%Ri.O	Вихідне слово регістра. Його можна прочитати, протестувати, записати.
Архівне слово (або інструкція) (Archiving input or instruction)	I (In)	При подачі імпульсу по передньому фронту вміст слова **%Ri.I** запам‘ятовується у регістрі.
Вхід зчитування (або інструкція) (Retrieval input or instruction)	O (Out)	При подачі імпульсу по передньому фронту зчитується слово, збережене у регістрі, і поміщається у слово **%Ri.O.**
Вхід Reset	R (Reset)	При R=1 відбувається ініціалізація регістра.
Вихід Empty (пустий)	E (Empty)	Біт **%Ri.E** показує, що регістр пустий.
Вихід Full (повний)	F (Full)	Біт **%Ri.F** показує, що регістр максимально заповнений.

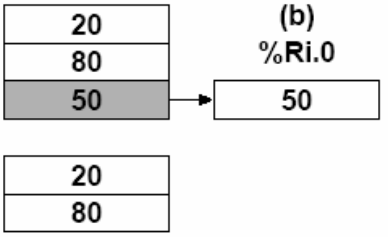
Коли обидва входи **I** та **O** активні одночасно, то спочатку відбувається запам‘ятовування слова у регістрі, а після цього виконується зчитування.

Принцип роботи регістра у режимі FIFO (first in, first out)

Запис у верхню частину стеку вмісту слова **%Ri.I:**

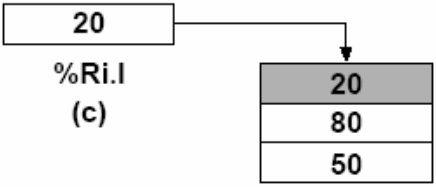


Зчитування даних, які перші надійшли у регістр, і запис їх у слово **%Ri.O:**

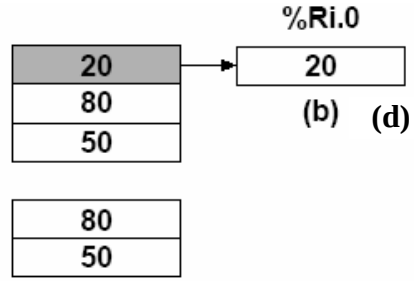


Принцип роботи регістра у режимі LIFO (last in, first out)

Запис у верхню частину стеку вмісту слова **%Ri.I:**



Зчитування даних, які останні надійшли у регістр, і запис їх у слово **%Ri.O:**



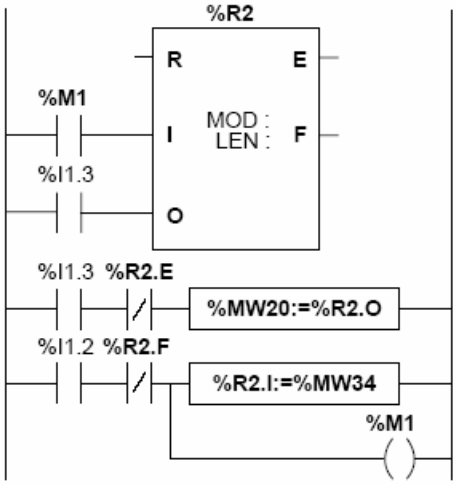
Програмування і конфігурація регістра %Ri

Для кожного регістра, використаного у програмі необхідно задати наступні внутрішні параметри:

- номер (number): [0, 3] для TSX 37; [0, 254] для TSX 57;
- довжина (length): від 1 до 255.

У прикладі, наведеному нижче, вміст слова %MW34 завантажується у слово %R2.I під час замикання контакту %I1.2, якщо регістр не повний (%R2.F=0). Одночасно замикається контакт %M1, який подає командний імпульс на вхід регістра I. Для зчитування даних з регістру, необхідно замкнути контакт %I1.3. І якщо регістр не пустий (%R2.E=0), дані запишуться у змінну %R2.O, а з неї – у слово %MW20.

Мова драбинкових діаграм (LD)



Мова списку інструкцій(IL)

```
LD      %M1
I       %R2
LD      %I1.3
O       %R2
LD      %I1.3
ANDN    %R2.E
[%MW20:=%R2.O]
LD      %I1.2
ANDN    %R2.F
[%R2.I:=%MW34]
ST      %M1
```

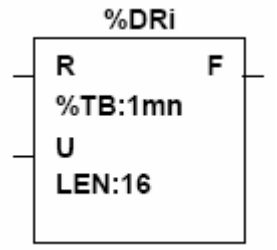
Мова структурованого тексту (ST)

```
IF RE %M1 THEN
    PUT %R2 ;
END_IF ;
IF RE %I1.3 THEN
    GET %R2 ;
END_IF ;
IF (%I1.3 AND NOT %R2.E) THEN
    %MW20 := %R2.O ;
END_IF ;
%M1 := %I1.2 AND NOT %R2.F ;
IF %M1 THEN
    %R2.I := %MW34 ;
END_IF ;
```

У мові ST використовуються такі додаткові команди:

- **RESET %Ri** – ініціалізація регістру;
- **PUT %Ri** – запам‘ятовування вмісту слова %Ri.I у регістрі;
- **GET %Ri** – зчитування даних з регістру у слово %Ri.O.

2.1.6 Барабанний контролер (Drum controller %DRi)



Програмований барабанний контролер функціонує за тим самим принципом, що й електромеханічний командоапарат, який змінює стан певних змінних на кожному кроці згідно із зовнішніми подіями. На кожному кроці верхня точка кожного кулачка замикає контакт. Наприклад, такі командоапарати використовувалися у пральних машинах.

У програмованому барабанному контролері замиканню контакта відповідає присвоєння значення 1 дискретним виходам %Qi.j або внутрішнім бітам пам‘яті %Mi, які називаються контрольними бітами.

Таблиця 5.6

Характеристики барабанного контролера %DRi

Номер барабанного контролера (Drum controllers number)	%DRi	[0, 7] для TSX 37; [0, 254] для TSX 57
Кількість кроків (Length)	LEN	Від 1 до 16 (16 за замовчуванням).
Базовий час (Time base)	TB	1min, 1s, 100ms, 10ms (1min за замовчуванням)
Тривалість виконання кроку (Time envelope or time period of current step)	%DRi.V	0<=%DRi.V<=9999. Тривалість виконання дорівнює %DRi.V*TB. Дане слово доступне тільки для зчитування.
Номер поточного кроку (Current step number)	%DRi.S	0<=%DRi.S<=15. Слово доступне тільки для зчитування.
Вхід Reset	R (Reset)	При R=1 відбувається ініціалізація барабанного контролера, тобто перехід на крок №0.
Вхід Up (Advanced	U (Up)	По передньому фронту

input)		імпульса, який подається на даний вхід, відбувається перехід на наступний крок та поновлення стану контрольних бітів.
Вихід Full (повний)	F (Full)	Біт %DRi.F=1, якщо відбувається виконання останнього кроку барабанного контролера.
Стан кроку (State of a step)	%DRi.Wj	16-бітне слово, яке визначає стани кроку j барабанного контролера i. Доступне тільки для зчитування.
Контрольні біти (Control bits)		Виходи або внутрішні біти пам‘яті, асоційовані з кроками (16 бітів max).

Примітка: Біт %S18=1, якщо відбулася спроба запису несконфігурованого кроку.

Програмування і конфігурація барабанного контролера %DRi

Барабанний контролер містить:

- Матрицю констант (кулачки), яка відображає стан контрольних бітів на кожному кроці роботи.
- Список контрольних бітів.

Step

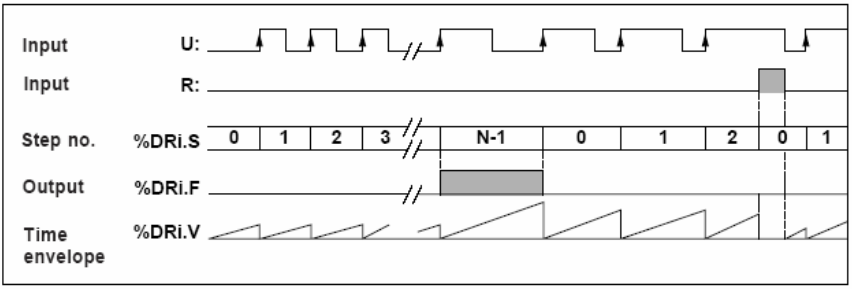
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Address
0	0	0	0	1	0	1	0	1	0	1	1	0	1	1	0	0	%Q2.0
1	0	1	0	0	1	1	1	0	0	0	1	0	0	0	1	0	%Q2.1
2	0	0	1	1	0	1	0	1	0	1	0	0	0	0	0	0	%M23
3	0	0	0	1	0	0	0	0	0	0	1	0	0	1	1	0	%M32
4	1	0	0	0	1	1	1	0	0	0	1	1	0	1	0	1	%Q2.9
5	1	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	%Q2.10
6	1	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	%Q2.11
7	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	%Q2.12
8	1	0	0	0	1	0	0	0	0	1	0	1	0	1	0	0	%M8
9	0	1	0	0	0	1	1	0	0	1	0	0	0	0	1	0	%M9
A	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	%M1
B	1	0	0	1	1	0	0	0	1	0	0	0	0	1	0	1	%Q2.6
C	1	0	1	0	1	0	1	0	0	1	1	1	0	0	0	0	%Q2.7
D	0	1	0	0	1	0	1	1	1	0	0	1	0	0	1	1	%Q2.8
E	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	%M100
F	0	0	1	0	0	0	0	1	0	1	0	0	1	1	1	0	%M13

Bit

Control bits

За таблицею, наведеною вище, можемо побачити, що на кроці №1 біти %Q2.1, %Q2.12, %M9, %M1, %Q2.8, %M100 встановляться в 1. Інші контрольні біти встановляться в 0. Номер поточного кроку нарощується на 1 по передньому фронту імпульса, який подається на вхід U, (або при виконанні команди U).

Діаграма роботи барабанного контролера



Розглянемо приклад програмування барабанного контролера. Припустимо, необхідно, щоб 5 виходів від %Q2.0 до %Q2.4 активізувалися по-черзі на кожному кроці. Перехід на наступний крок здійснюється при подачі імпульса на вхід %I1.1. Вхід %I1.0 ініціалізує барабанний контролер і здійснює перехід на крок 0.

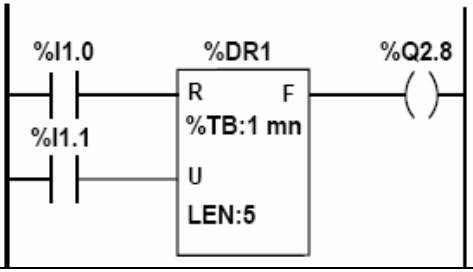
У редакторі змінних налаштуємо барабанний контролер наступним чином:

- кількість кроків LEN=5;

- стан виходів (контрольних бітів) на кожному кроці визначатиметься наступною таблицею:

	Step	Assignment of control bits
	0 1 2 3 4	
	0 : 1 0 0 0 0	%Q2.0
	1 : 0 1 0 0 0	%Q2.1
Bit	2 : 0 0 1 0 0	%Q2.2
	3 : 0 0 0 1 0	%Q2.3
	4 : 0 0 0 0 1	%Q2.4

- базовий час TB=1min.

<table>
<tr>
<td>Мова драбинкових діаграм (LD)
 
</td>
<td>Мова списку інструкцій(IL)
<pre>
LD %I1.0
R   %DR1
LD %I1.1
U   %DR1
LD %DR1.F
ST %Q2.8
</pre></td>
</tr>
<tr>
<td>Мова структурованого тексту (ST)
<pre>
IF %I1.0 THEN
    RESET %DR1 ;
END_IF ;
IF RE %I1.1 THEN
    UP %DR1 ;
END_IF ;
%Q2.8 := %DR1.F ;
</pre></td>
<td>Команда RESET %DRi використовується для ініціалізації барабанного контролера (перехід на крок 0).
Команда UP %DRi здійснює перехід на 1 крок вперед та поновлює стан контрольних бітів. Дана команда генерує передній фронт імпульсу на вході U блоку. Тому необхідно контролювати наявність відповідного імпульсу оператором IF.</td>
</tr>
</table>

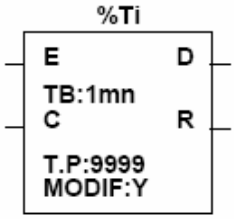
Примітка: При переведенні барабанного контролера на крок 0:

- мовою LD – значення входу U оновлюється поточним значенням;
- мовою IL – вхід U зберігає свій попередній стан;
- мовою ST – вхід U скидається в 0.

Особливі випадки:

- **Холодний старт:** (%S0=1) переведення барабанного контролера на крок 0 з оновленням стану контрольних бітів.
- **Теплий старт:** (%S1=1) оновлення стану контрольних бітів згідно з поточним кроком.
- **Програмний перехід, деактивація або зупинка виконання задачі**: контрольні біти не скидаються в 0.
- **Оновлення стану контрольних бітів:** виникає при зміні переході на новий крок, холодному або теплому старті.

2.1.7 Таймер (Серія 7) (Timer (Series 7) %Ti)



Функціональний блок таймера, який сумісний з блоками Series 7 PL7-2/3.

Таблиця 5.7

Характеристики таймера %Ti

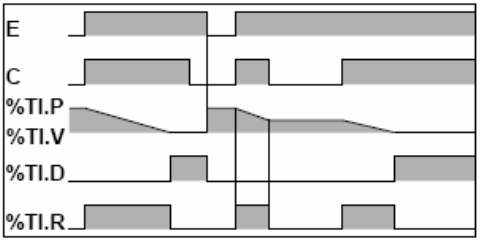
Номер таймера (Timer number)	%Ti	[0, 63] для TSX 37; [0, 254] для TSX 57
Базовий час (Time base)	TB	1 min, 1 s, 100 ms, 10 ms (1 min за замовчуванням).
Поточне значення (Current value)	%Ti.V	Слово, яке зменшується від %Ti.P до 0, коли таймер запущений. Його можна прочитати і протестувати, але не можна записати програмно.
Значення уставки (Preset value)	%Ti.P	0<=%Ti.P<=9999. Слово, яке можна прочитати, протестувати, записати програмно. За замовчуванням %Ti.P=9999.

		Результуючий інтервал часу t=%Ti.P*TB.
Доступ з термінала (Adjust via the terminal – MODIF)	Y/N	Y – значення уставки %Ti.P можна змінювати. N - значення уставки %Ti.P змінювати не можна.
Вхід ініціалізації (Setting input)	E (Enabled)	При E=0 відбувається ініціалізація таймера, %Ti.V = %Ti.P.
Керуючий вхід (Control input)	C (Control)	При C=0 заморожується поточне значення %Ti.V.
Вихід-індикатор виконання (Timer output done)	D (Done)	Біт %Ti.D=1, якщо завершилася робота таймера, тобто якщо %Ti.V=0.
Вихід-індикатор роботи (Timer running output)	R (Running)	Біт %Ti.R=1, якщо %Ti.P>%Ti.V>0 і C=1.

Примітка: Функціональний блок %Ti не можна запрограмувати мовою послідовних інструкцій IL, але можна аналізувати стан об‘єктів %Ti.V, %Ti.P, %Ti.D, %Ti.R даного блоку.

Загальна кількість використаних у програмі таймерів %TMi + %Ti не повинна перевищувати 64 для ПЛК серії TSX 37 і 255 для ПЛК серії TSX 57.

Часова діаграма роботи таймера %Ti



E	0	0	1	1
C	0	1	0	1
%Ti.P %Ti.V	%Ti.V = %Ti.P	%Ti.V = %Ti.P	%Ti.V frozen	%Ti.V decr. from %Ti.P -> 0
%Ti.D	0	0	0	1 if Timer done
%Ti.R	0	0	0	1 if Timer running

Таймер працює, якщо його входи E=1 і C=1. Він функціонує подібно до лічильника, який здійснює віднімання імпульсів.

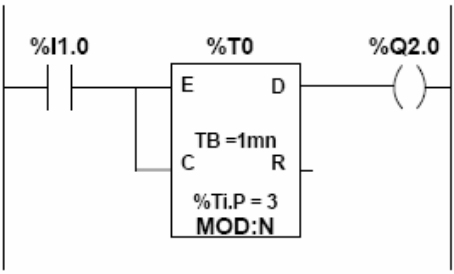
1. Поточне значення %Ti.V зменшується від уставки %Ti.P до 0.
2. Вихідні біти %Ti.R=1, %Ti.D=0.

Коли поточне значення %Ti.V=0, то %Ti.D=1 і %Ti.R=0.

Даний таймер може бути запрограмований для роботи у наступних режимах:

On-time delay (затримка на включення)

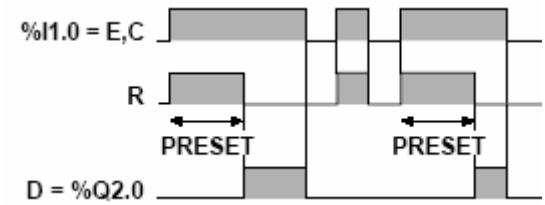
Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

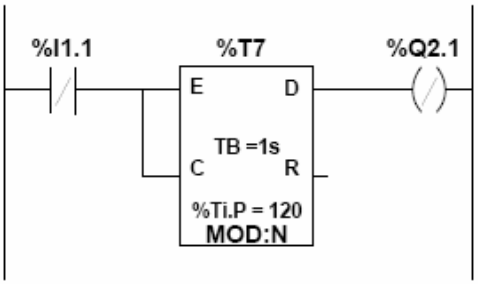
```
IF %I1.0 THEN
    START %T0 ;
ELSE
    PRESET %T0 ;
END_IF ;
%Q2.0 := %T0.D ;
```

Часова діаграма



Off-time delay (затримка на виключення)

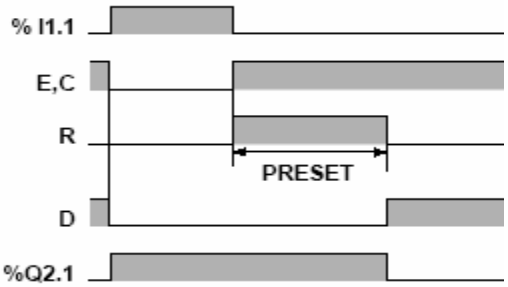
Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

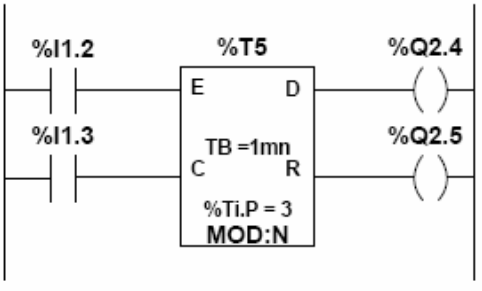
```
IF %1.1 THEN
    PRESET %T7 ;
ELSE
    START %T7 ;
END_IF ;
%Q2.1 := NOT %T7.D ;
```

Часова діаграма



Cumulated (змішаний) on-time delay

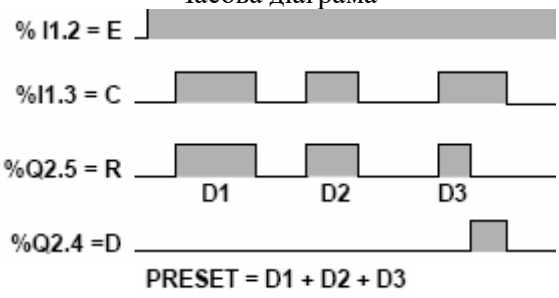
Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

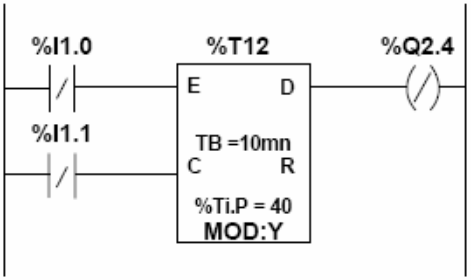
```
IF %I1.2 THEN
    IF %I1.3 THEN
        START %T5 ;
    ELSE
        STOP %T5 ;
    END_IF ;
ELSE
        PRESET %T5 ;
END_IF ;
%Q2.4 := %T5.D ;
%Q2.5 := %T5.R ;
```

Часова діаграма



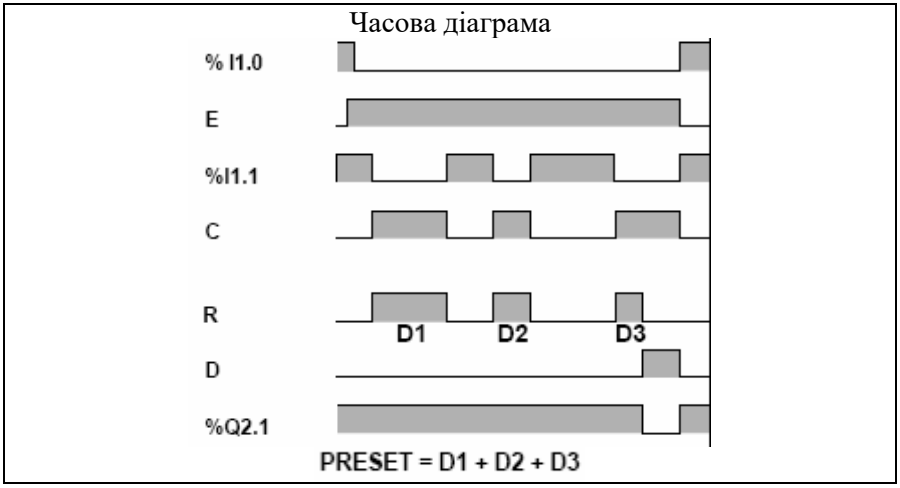
Cumulated (змішаний) off-time delay

Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

```
IF %I1.0 THEN
        PRESET %T12 ;
ELSE
    IF %I1.1 THEN
        STOP %T12 ;
    ELSE
        START %T12 ;
    END_IF ;
END_IF ;
%Q2.4 := NOT %T12.D ;
```



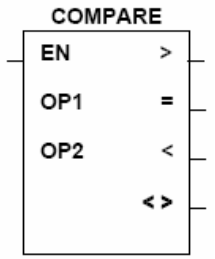
У мові структурованого тексту ST використано 3 команди для програмування таймера:

- PRESET %Ti – скидання таймера;
- START %Ti – запуск таймера;
- STOP %Ti – заморожування поточного значення таймера.

Спеціальні випадки:

- **Холодний старт:** (%S0=1) значення уставки завантажується у поточне значення, а вихід %Ti.D=0.
- **Теплий старт:** (%S1=1) ніяким чином не відображається на поточному значенні таймера.
- **Зупинка ПЛК:** зупинка ПЛК, деактивація поточної задачі або її зупинка не заморожує поточне значення таймера.
- **Програмний перехід**: Поточне значення %Ti.V таймера не заморожується, а продовжує зменшуватися до 0. Біти %Ti.D та %Ti.R можуть бути перевірені іншою частиною програми (до якої здійснений перехід). Однак котушки, напряму під‘єднані до виходів таймера, не активуватимуться, поки вони не будуть просканові контролером.
- **Оновлення стану бітів %Ti.D та %Ti.R:** стан даних бітів може змінюватися під час сканування контролером тієї частини програми, де міститься таймер.

2.1.8 Вертикальний блок порівняння (Vertical comparison block Compare)



Таблиця 5.8

Характеристики вертикального блоку порівняння

Керуючий вхід (Command input)	**EN**	При подачі 1 на даний вхід відбувається порівняння двох операндів.
Більше (Greater than output)	>	Дорівнює 1, якщо OP1>OP2.
Дорівнює (Equal to output)	=	Дорівнює 1, якщо OP1=OP2.
Менше (Less than output)	<	Дорівнює 1, якщо OP1<OP2.
Не дорівнює (Different from output)	<>	Дорівнює 1, якщо OP1<>OP2.
Операнд №1 (Operand #1)	**OP1**	16-бітне слово (може бути індексоване).
Операнд №2 (Operand #2)	**OP2**	16-бітне слово (може бути індексоване).

Конфігурація і програмування вертикального блоку порівняння

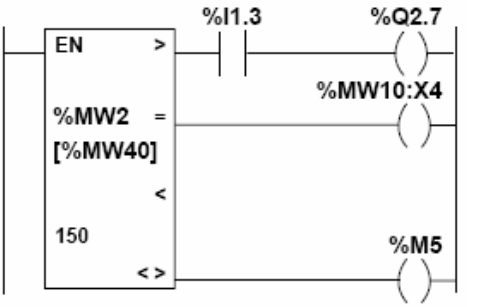
При замиканні керуючого входу відбувається порівняння двох операндів, 4 виходи активізуються згідно результату порівняння. При подачі на керуючий вхід 0 виходи блоку також скидаються в 0.

Приклад:

Першим операндом для порівняння являється слово %MW2, індексоване словом %MW40. Другим операндом є число 150.

Якщо вміст слова %MW2[%MW40] більше за 150 та %I1.3=1, котушка %Q2.7 активізується. Якщо вміст слова %MW2[%MW40]

дорівнює 150, котушка %MW10:X4 активізується. Котушка %M5 замикається, якщо вміст слова %MW2[%MW40] не дорівнює 150.

	Даний функціональний блок не існує у мовах послідовних інструкцій IL та структурованого тексту ST.

Спеціальні випадки:

- **Холодний старт:** (%S0=1) операнд OP1 і оператор OP2 (якщо OP2 є внутрішнім словом) скидаються, а значення виходів поновлюються згідно з результатом їх порівняння з новими значеннями.
- **Теплий старт:** (%S1=1) не має ефекту на роботу блоку порівняння.

2.1.9 Функції, які заміняють роботу таймера

На відміну від стандартних функціональних блоків, дані функції не лімітовані в кількості і можуть використовуватися у коді функціональних блоків користувача (DFB блоків).

Є 4 функції, що заміняють роботу таймера:

- FTON: затримка на включення (On-delay);
- FTOF: затримка на виключення (Off-delay);
- FTP: імпульсний режим (Pulse time delay);
- FPULSOR: прямокутний імпульсний сигнал (square wave signal).

2.1.9.1 FTON on-delay function (функція затримки на включення)

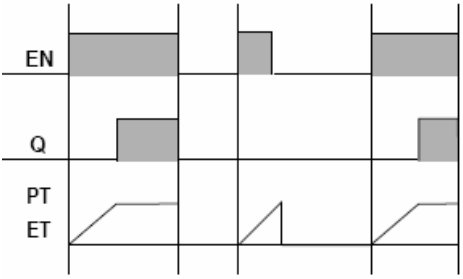
Синтаксис функції FTON

FTON (EN, PT, Q, ET, PRIV)

Таблиця 5.9

Характеристики функції FTON

Запускаючий вхід (“Enabled” input)	EN	По передньому фронту імпульса здійснюється старт на початок відліку часу.
Значення уставки (Preset value)	PT	Вхідне слово, яке визначає тривалість (у сотих долях секунди) паузи. Може бути використане для задання часового інтервалу 5 min 27 s max з точністю 10 ms. Зміна даного слова враховується під час паузи.
Часовий вихід (“Timer” output)	Q	Вихід встановлюється в 1, коли поточне значення досягає уставки, тобто ET=PT.
Поточне значення (Current value)	ET	Вихідне слово, яке наростає від 0 до PT, коли минув час паузи.
Лічильна змінна (Calculation variable)	PRIV	Подвійне слово для зберігання поточних станів.

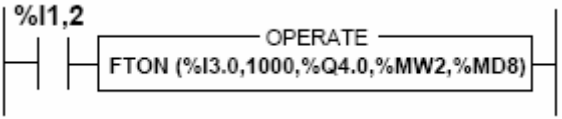


По передньому фронту імпульса, поданого на вхід EN, таймер запускається: його поточне значення ET наростає від 0 до PT (у сотих долях секунди). Вихідний біт Q змінюється на 1, коли поточне значення досягає уставки PT і залишається в 1, поки EN=1.

Коли вхід EN=0, таймер зупиняється, навіть якщо він до того працював: ET при цьому скидається в 0.

Програмування

Мова драбинкових діаграм (LD)



Мова списку інструкцій(IL)

```
LD %I1.2
[FTON (%I3.0,1000,%Q4.0,%MW2,%MD8)]
```

Мова структурованого тексту (ST)

```
IF %I1.2 THEN
    FTON (%I3.0,1000,%Q4.0,%MW2,%MD8) ;
END_IF ;
```

Операнди функції FTON

FTON (EN, PT, Q, ET, PRIV)

Type	EN	PT	Q	ET	PRIV
Indexable words		%MW,%KW,%Xi.T		%MW	
Non-indexable words		%IW,%QW,%SW Immed. val., Num. expr.,%NW		%IW,%QW	
Indexable double words					%MD
Bits	%I,%Q, %M, %S %BLK,%•:Xk,%X		%I,%Q, %M, %S,%•:Xk,%X		

2.1.9.2 FTOF off-delay function (функція затримки на виключення)

Синтаксис функції FTOF

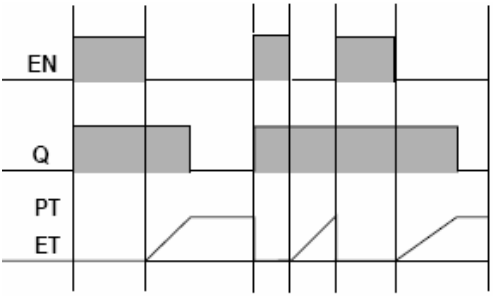
FTOF (EN, PT, Q, ET, PRIV)

Таблиця 5.10

Характеристики функції FTOF

Запускаючий вхід (“Enabled” input)	EN	По передньому фронту імпульса здійснюється старт на початок відліку часу.
Значення уставки	PT	Вхідне слово, яке визначає

(Preset value)		тривалість (у сотих долях секунди) паузи. Може бути використане для задання часового інтервалу 5 min 27 s max з точністю 10 ms. Зміна даного слова враховується під час паузи.
Часовий вихід (“Timer” output)	Q	Вихід встановлюється в 1 по передньому фронту імпульса на вході EN і встановлюється в 0, коли ET=PT.
Поточне значення (Current value)	ET	Вихідне слово, яке наростає від 0 до PT, коли минув час паузи.
Лічильна змінна (Calculation variable)	PRIV	Подвійне слово для зберігання поточних станів.

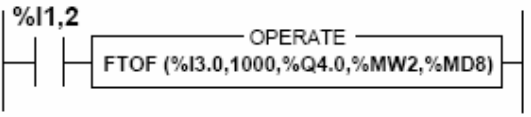


По передньому фронту імпульса, поданого на вхід EN, поточне значення таймера ET=0 (навіть якщо таймер працював). По задньому фронту імпульса на вході EN таймер запускається. Поточне значення тоді змінюється від 0 до PT (у сотих долях секунди).

Вихідний біт Q змінюється на 1 при появі на вході EN переднього фронту імпульса, і стає рівним 0, коли поточне значення досягає уставки PT.

Програмування

Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

```
IF %I1.2 THEN
 FTOF (%I3.0,1000,%Q4.0,%MW2,%MD8) ;
END_IF ;
```

2.1.9.3 FTP pulse delay function (функція генерації імпульсів) дана функція використовується для генерації імпульсу заданої тривалості. Тривалість імпульсу програмується.

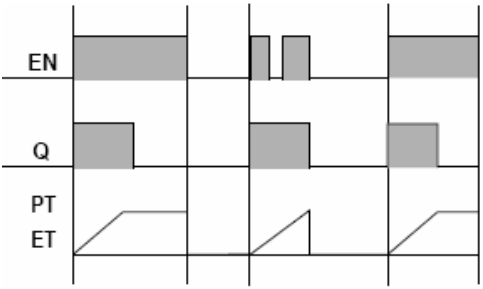
Синтаксис функції FTP

FTP (EN, PT, Q, ET, PRIV)

Таблиця 5.11

Характеристики функції FTP

Запускаючий вхід (“Enabled” input)	EN	По передньому фронту імпульса здійснюється старт на початок відліку часу.
Значення уставки (Preset value)	PT	Вхідне слово, яке визначає тривалість (у сотих долях секунди) паузи. Може бути використане для задання часового інтервалу 5 min 27 s max з точністю 10 ms. Зміна даного слова враховується під час паузи.
Часовий вихід (“Timer” output)	Q	Вихід встановлюється в 1 наприкінці відліку часу.
Поточне значення (Current value)	ET	Вихідне слово, яке наростає від 0 до PT, коли минув час паузи.
Лічильна змінна (Calculation variable)	PRIV	Подвійне слово для зберігання поточних станів.



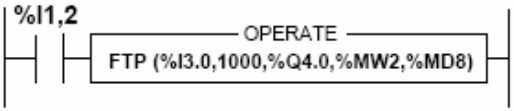
По передньому фронту імпульса, поданого на вхід EN, таймер запускається (якщо він не був запущений раніше). Поточне значення ET тоді наростає від 0 до PT (у сотих долях секунди).

Вихідний біт Q встановлюється в 1 під час запуску таймера і змінюється на 0, коли поточне значення досягає значення уставки PT.

Коли вхід EN і вихід Q знаходяться у значенні 0, то ET=0. Імпульс не може бути перерваний.

Програмування

Мова драбинкових діаграм (LD)



Мова структурованого тексту (ST)

```
IF %I1.2 THEN
 FTP (%I3.0,1000,%Q4.0,%MW2,%MD8) ;
END_IF ;
```

2.1.9.4 FPULSOR function for generation square wave signals (функція для генерації прямокутних імпульсів)

Дана функція використовується для генерації періодичних прямокутних імпульсів. Ширину імпульсу можна змінювати за допомогою двох таймерів:

- TON: on-delay (по імпульсу 1);
- TOFF: off-delay (по імпульсу 0).

Синтаксис функції FPULSOR

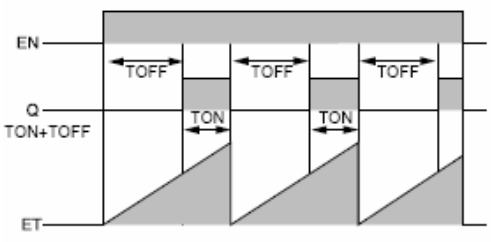
FPULSOR (EN, TON, TOFF, Q, ET, PRIV)

Таблиця 5.12

Характеристики функції FPULSOR

Запускаючий вхід (“Enabled” input)	EN	По передньому фронту імпульса здійснюється старт на початок відліку часу.
Значення уставки	TON	Вхідне слово, яке визначає

імпульс 1 (Preset value pulse at 1)		тривалість (у сотих долях секунди (по імпульсу 1)) паузи. Може бути використане для задання часового інтервалу 5 min 27 s max з точністю 10 ms. Тривалість TON+TOFF=5 min 27 s max.
Значення уставки імпульс 0 (Preset value pulse at 0)	TOFF	Вхідне слово, яке визначає тривалість (у сотих долях секунди (по імпульсу 0)) паузи. Може бути використане для задання часового інтервалу 5 min 27 s max з точністю 10 ms. Тривалість TON+TOFF=5 min 27 s max.
Вихід прямокутного сигналу (Square signal output)	Q	Вихід має значення 0 на протязі періоду TOFF, і 1 на протязі періоду TON.
Поточне значення (Current value)	ET	Вихідне слово, яке наростає від 0 до TON+TOFF, коли минув час паузи.
Лічильна змінна (Calculation variable)	PRIV	Подвійне слово для зберігання поточних станів.

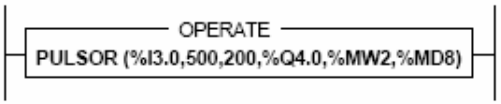


По передньому фронту імпульса на вході EN генерується прямокутний імпульс (якщо до того сигнал на виході був відсутній). Поточне значення ET починає наростати від 0 до TON+TOFF (у сотих долях секунди).

Вихідний біт Q залишається в 0 на протязі часу TOFF і змінюється на 1 на протязі часу TON, потім повертається в 0 і т. д., поки на вході EN є 1.

Програмування

Мова драбинкових діаграм (LD)



Мова списку інструкцій(IL)

```
LD True
[FPULSOR (%I3.0,500,200,%Q4.0,%MW2,
%MD8)]
```

Мова структурованого тексту (ST)

```
IF %I1.2 THEN
    FPULSOR (%I3.0,500,200,%Q4.0,%MW2,%MD8) ;
END_IF ;
```

Операнди функції FPULSOR

FPULSOR (EN, TON, TOFF, Q, ET, PRIV)

Type	EN	TON,TOFF	Q	ET	PRIV
Indexable words		%MW,%KW,%Xi.T		%MW	
Non-indexable words		%IW,%QW,%SW Imm. val. Num. expr.,%NW		%IW,%QW	
Indexable double words					%MD
Bits	%I,%Q, %M, %S %BLK,%•:Xk,%X		%I,%Q, %M, %S,%•:Xk,%X		

2.2 Методи керування процесом сушінні сипучих матеріалів

Сушіння сипучих матеріалів є важливим етапом у багатьох галузях промисловості, включаючи харчову, хімічну та будівельну. Процес сушіння впливає на якість кінцевого продукту, ефективність виробництва та витрати енергії. Тому вибір ефективних методів керування сушінням є надзвичайно важливим. Існує кілька основних методів, які дозволяють оптимізувати цей процес.

Першим методом є конвективне сушіння, яке передбачає передачу тепла від гарячого повітря до матеріалу. Цей метод широко використовується завдяки своїй простоті та

універсальності. Для керування процесом конвективного сушіння застосовують регулювання температури та швидкості потоку повітря. Автоматичні системи керування дозволяють підтримувати оптимальні умови сушіння, зменшуючи час процесу та енергозатрати.

Другий метод – це контактне сушіння, яке здійснюється шляхом передачі тепла через контакт матеріалу з гарячою поверхнею. Цей метод ефективний для матеріалів з високою вологістю, оскільки забезпечує швидке видалення вологи. Керування процесом сушіння досягається за допомогою регулювання температури контактної поверхні та часу контакту. Автоматизація дозволяє точно контролювати ці параметри, що забезпечує стабільну якість продукту.

Третім методом є сушіння в сушарках з киплячим шаром, де матеріал знаходиться в стані псевдокипіння завдяки потоку гарячого повітря. Цей метод забезпечує рівномірне сушіння, що особливо важливо для матеріалів з неоднорідною структурою. Керування процесом досягається через регулювання параметрів повітряного потоку, таких як температура, швидкість і вологість. Сучасні системи автоматизації дозволяють моніторити та коригувати процес в режимі реального часу.

Ще одним ефективним методом є мікрохвильове сушіння, яке базується на використанні електромагнітних хвиль для нагрівання матеріалу. Цей метод забезпечує швидке та рівномірне сушіння, зменшуючи енергозатрати. Процес керується шляхом регулювання потужності мікрохвильового випромінювання та тривалості обробки. Автоматичні системи дозволяють точно контролювати ці параметри, запобігаючи перегріву та зниженню якості продукту.

Крім технічних методів, важливу роль у керуванні процесом сушіння відіграють програмні засоби, які дозволяють моделювати та оптимізувати процеси. Використання спеціалізованого програмного забезпечення допомагає розраховувати оптимальні параметри сушіння для різних матеріалів, знижуючи витрати та підвищуючи ефективність.

Таким чином, ефективне керування процесом сушіння сипучих матеріалів потребує комплексного підходу, що включає вибір відповідного методу сушіння, автоматизацію процесів та використання сучасного програмного забезпечення. Це дозволяє

забезпечити високу якість кінцевого продукту, зменшити енергозатрати та підвищити продуктивність виробництва.

3. Програма роботи

1. Навчитися використовувати функціональні блоки при програмуванні промислового контролера Schneider Micro TSX 37-22 мовами LD, IL, ST у середовищі програмування PL7 PRO.
2. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу сушіння сипучих матеріалів.
3. Створити другий рівень системи контролю і керування технологічним процесом засобами програмного забезпечення PL7 PRO.
4. Перевірити роботу програми.

4. Опис лабораторного обладнання

1. Персональний комп‘ютер.
2. Операційна система Windows.
3. Лабораторний стенд стенд з ПЛК Schneider Micro TSX 37-22.
4. Середовище програмування ПЛК Schneider Micro TSX 37-22 PL7 PRO, середовище програмування операторської панелі Magelis XBT-L-1000.

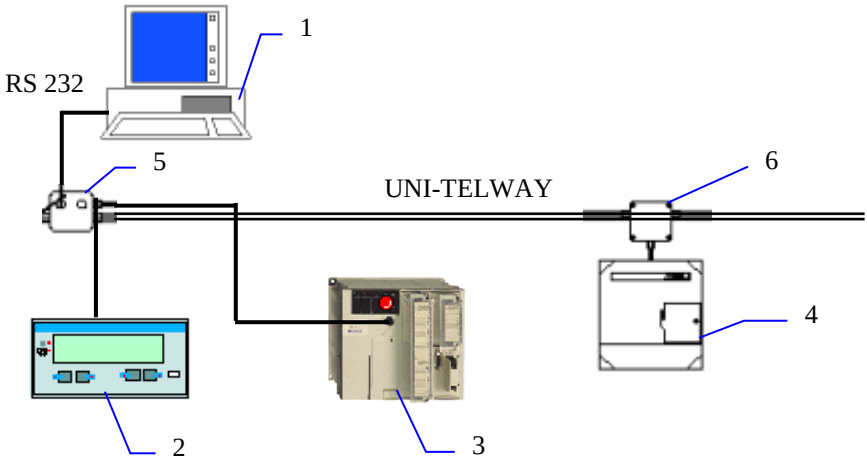


Рис. 5.1 Лабораторний стенд:
1- АРМ оператора; 2 - операторська панель Magelis XBT P 012010; 3 - ПЛК Schneider Micro TSX 37-22; 4 - частотний перетворювач Altivar 58; 5 - адаптер відгалуженняTSX P ACC01; 6 - розгалуджувач TSX SCA 62.

5. Порядок виконання роботи

1. Під‘єднати контролер Schneider Micro TSX 37-22 до COM-порту комп‘ютера за допомогою комунікаційного кабелю TSX PCX 1031. Ввімкнути живлення стенду.
2. Запустити середовище PL7 PRO.
3. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу сушіння сипучих матеріалів, використовуючи мови програмування драбинкових діаграм, послідовних інструкцій та структурованого тексту, а також стандартні функціональні блоки.

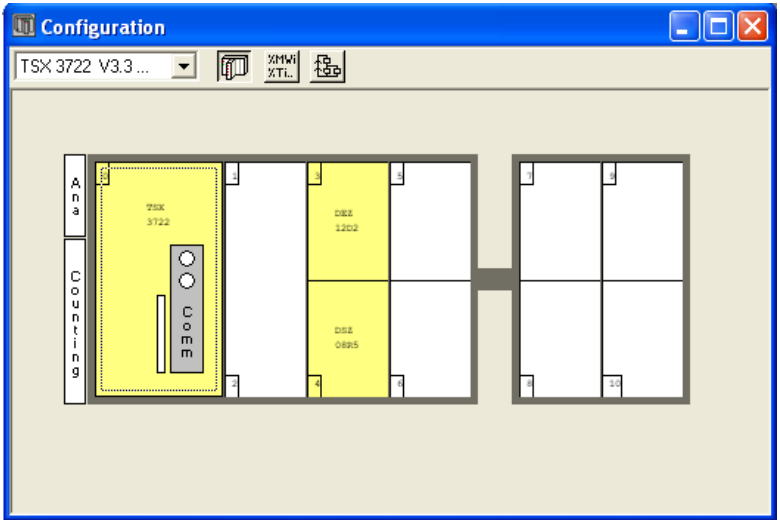


Рис. 5.2 Підключення та конфігурування модулів вводу-виводу

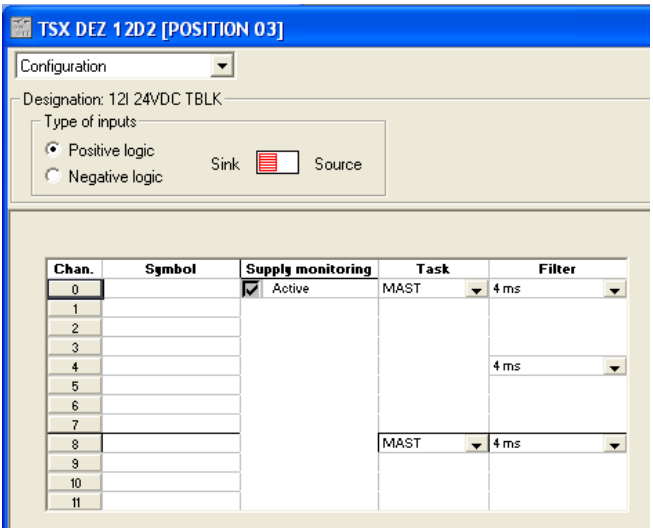


Рис. 5.3 Модуль дискретного вводу у слоті №3

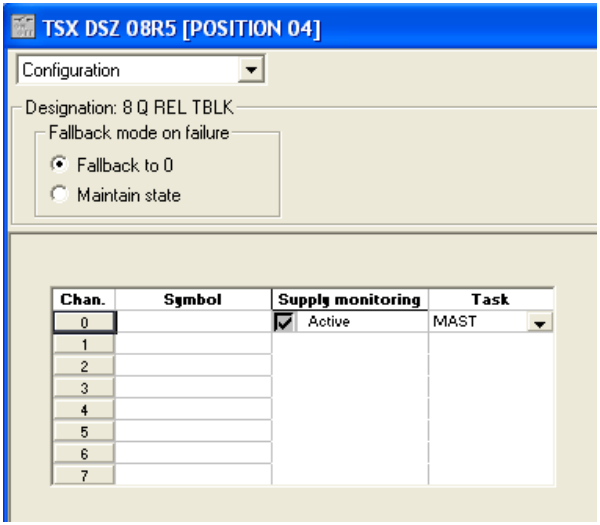


Рис. 5.4 Модуль дискретного виводу у слоті №4

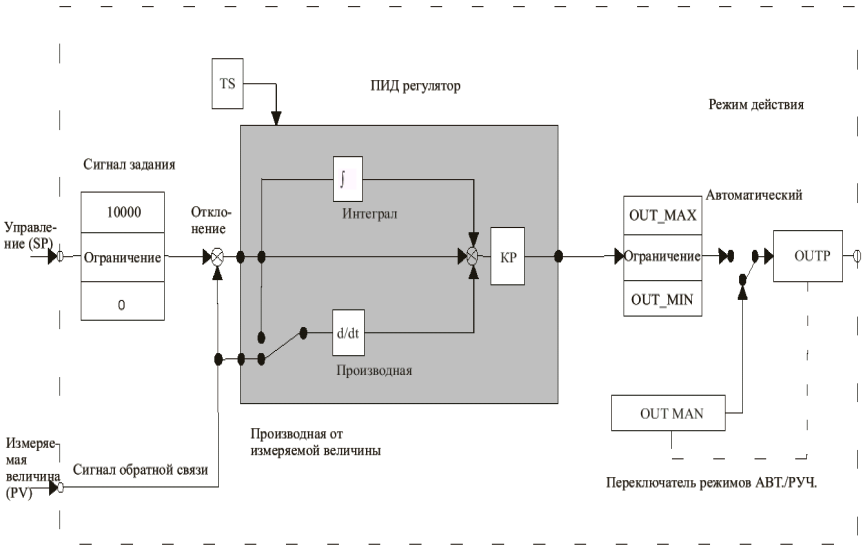


Рис. 5.5 Будова ПІД регулятора

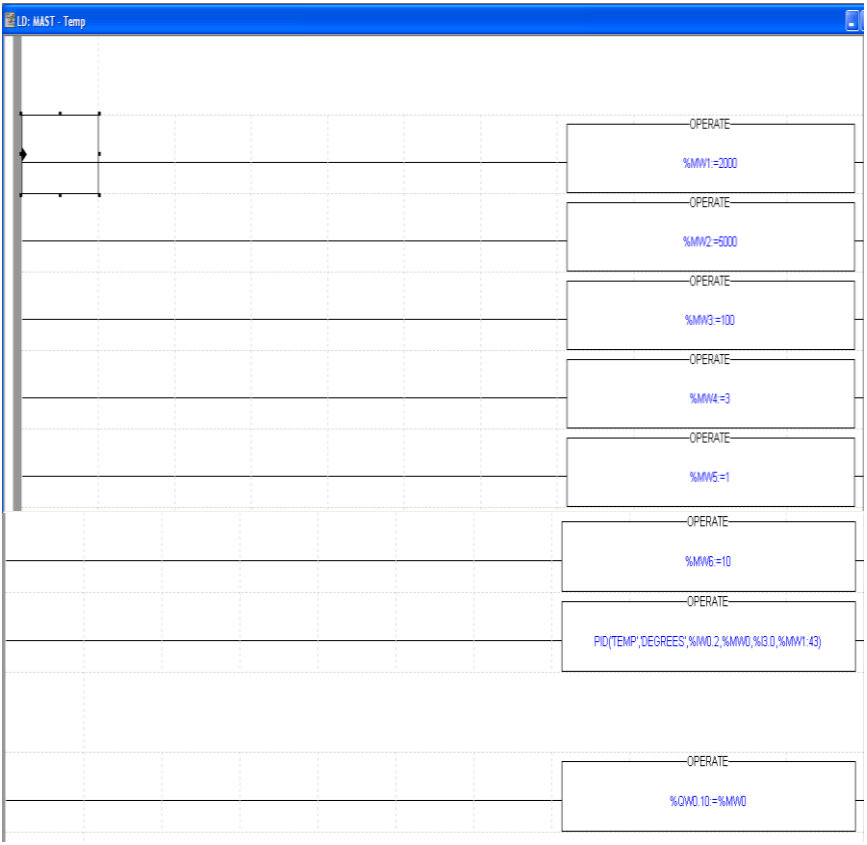


Рис. 5.6 Програма мовою LD

4. Створити анімаційну таблицю для відслідковування значень використаних у програмі змінних. Наряду з анімаційною таблицею використати операторську панель Magelis.

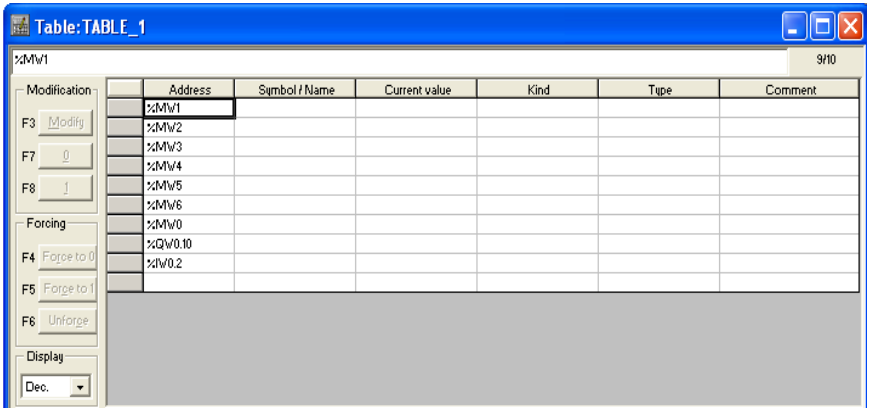


Рис. 5.7 Анімаційна таблиця

5. Створити вікно супервізора (Run-Time screen) засобами програмного забезпечення PL7 PRO, в якому намалювати ФСА технологічного процесу.

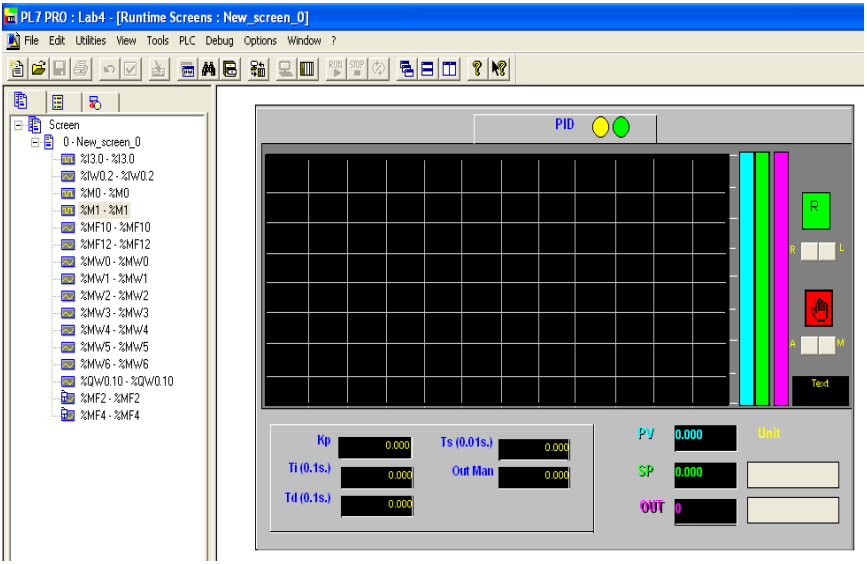


Рис. 5.8 Вікно супервізора

6. Записати програму у контролер. Для цього виконати наступні дії:

- виконати команду Conect;
- вибрати напрям запису програми PC -> PLC;
- перевести контролер у режим RUN.

7. Провести моніторинг роботи програми за допомогою анімаційної таблиці, вікна супервізора.
8. Написати програму для дослідження роботи функціональних блоків. При цьому використати різні функціональні блоки. Використати приклади, наведені у теоретичних відомостях.
9. Провести моніторинг роботи програми.

Опис технологічного процесу та вимоги до програми:

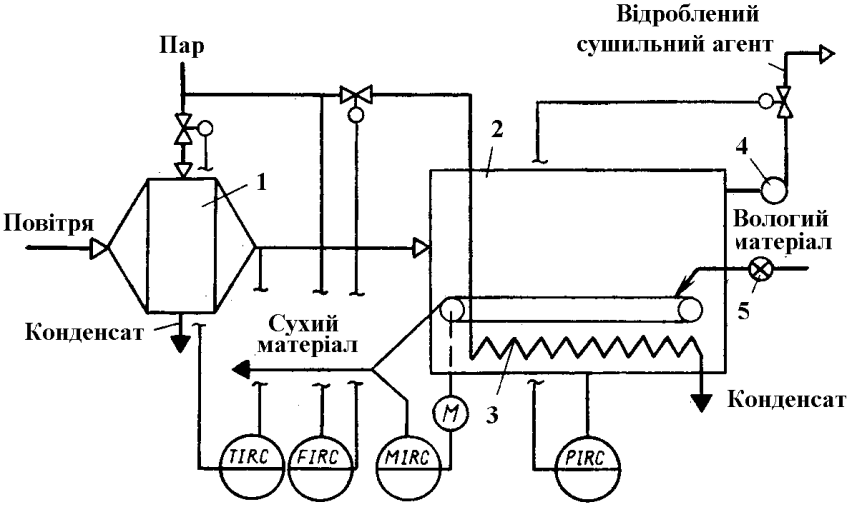


Рис. 5.9 ФСА процесу стрічкової (конвеєрної) сушки: 1 – калорифер, 2 – сушка, 3 – додатковий підігрівач, 4 – вентилятор, 5 - насичувач

У процесі сушіння сипучих матеріалів необхідно контролювати наступні параметри: температуру повітря на виході з калорифера, витрату пари на лінії подачі в сушильну камеру, вологість матеріалу на виході із сушильної камери, тиск повітря у сушильній камері. Температура повітря на виході з калорифера регулюється за допомогою зміни витрати пари на вході калорифера. Витрата пари на лінії подачі в сушильну камеру регулюється за допомогою клапана на цій же лінії. Вологість матеріалу на виході із сушильної

камери регулюється за допомогою зміни швидкості руху конвеєра у сушильній камері. Тиск повітря у сушильній камері регулюється за допомогою клапана на лінії відробленого сушильного агенту із сушильної камери.

6. Контрольні запитання

1. Яким чином використати функціональний блок у програмі для ПЛК Schneider Micro у середовищі програмування PL7 PRO?
2. Поясніть принцип роботи таймера.
3. Поясніть принцип роботи лічильника.
4. Поясніть принцип роботи моностабільного функціонального блоку.
5. Поясніть принцип роботи регістра.
6. Поясніть принцип роботи барабанного контролера.
7. Поясніть принцип роботи таймера серії 7.
8. Поясніть принцип роботи функцій FTON, FTOF, FTP, FPULSOR.
9. Яким чином запрограмувати ПІД регулятор у середовищі PL7 PRO?

Робота №7. Розробка і дослідження системи контролю та керування процесом приготування суміші з трьох компонентів на базі ПЛК Schneider Micro TSX 37-22.

1. Мета роботи

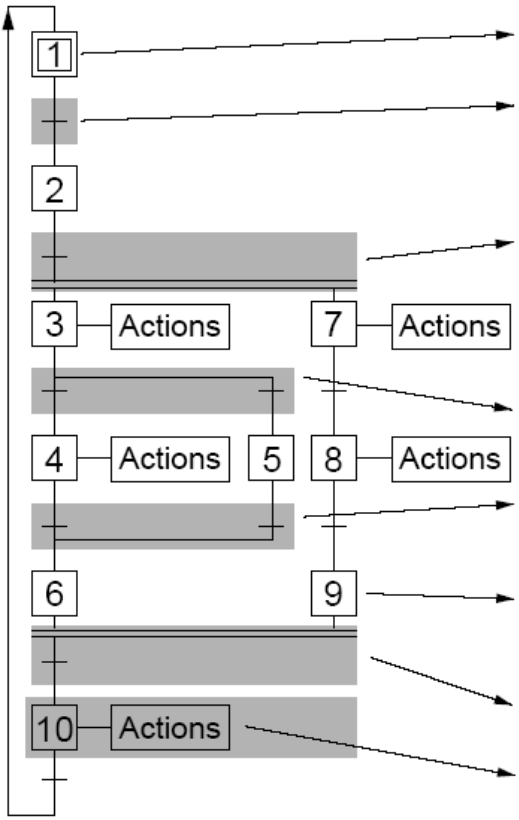
Навчитися програмувати промисловий логічний контролер (ПЛК) Schneider Micro TSX 37-22 мовою Grafcet у середовищі програмування PL7 PRO. Навчитися розробляти дворівневі системи контролю та керування на базі Schneider Micro TSX 37-22.

2. Теоретичні відомості

2.1 Створення програмного забезпечення мовою Grafcet для керування технологічним процесом на базі ПЛК Schneider Micro

2.1.1 Основні принципи написання програм мовою Grafcet

Мова програмування Grafcet ще називається Sequential Function Chart language (SFC) (мова послідовного функціонального рисунку) і належить до стандарту IEC 1131-3. Мова Grafcet використовується для представлення програми у структурованому графічному вигляді. При написанні програми використовуються спеціальні графічні елементи:



Початковий крок: визначає початкову ситуацію ПЛК.
Перехід: визначає умову переходу на наступний крок.

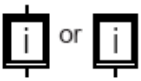
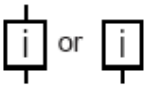
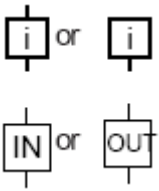
Одночасна активація кроків 3 і 7 (операція AND – початок). Послідовності кроків 3, 4, 5, 6 та 7, 8, 9 виконуються паралельно.
Операція вибіркового переходу OR (початок) від кроку 3 до кроку 4 або до кроку 5.
Кінець операції OR і перехід від кроків 4 і 5 до кроку 6.

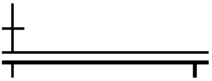
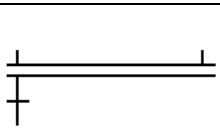
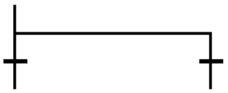
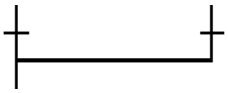
Кінець операції AND, синхронне завершення виконання паралельних послідовностей і перехід до кроку 10.

Крок 10. Виконуються дії, асоційовані з цим кроком.

2.1.2 Графічні елементи мови Grafcet

Таблиця 7.1

Назва	Зображення	Функція
Початкові кроки (Initial steps)		Початкові кроки ініціалізуються при запуску програми або після холодного старту.
Одиничний крок (Single steps)		Максимальна кількість кроків: - 1-96 для TSX 37-10, - 1-128 для TSX 37-20, - 1-250 для TSX 57. Максимальна кількість одночасно активних кроків конфігурується.
Макрокроки (Macro-step)		Макрокроки – цілісна структура (підпрограма), яка складається з кроків і переходів. Максимальна кількість макрокроків – від 0 до 63 для ПЛК TSX 57.
Кроки макрокроків (Macro-step steps)		Максимальна кількість кроків для кожного макрокрока – від 0 до 250 для TSX 57. Кроки IN та OUT – початок та кінець макрокрока. З кроком OUT не можуть бути асоційовані дії.
Переходи (Transitions)		Перехід від одного кроку до іншого. Перехід відбувається, коли умова переходу, асоційована з ним, істинна. Максимальна кількість переходів – 1024. Максимальна кількість переходів, активних одночасно, конфігурується.

Операція AND початок (AND divergences)		Перехід від одного кроку до декількох кроків (максимум 11) одночасно.
Операція AND кінець (AND convergences)		Перехід від декількох кроків до одного. Відбувається одночасна деактивація декількох (максимум 11) кроків.
Операція OR початок (OR divergences)		Перехід від одного кроку до декількох (максимум 11) згідно умов переходу.
Операція OR кінець (OR convergences)		Перехід від декількох кроків до одного. Завершення операції вибору OR.
Конектор-приймач (Source connector)		n – номер кроку, від якого здійснений перехід (крок-джерело).
Направляючий конектор (Destination connector)		n – номер кроку, до якого здійснюється перехід.
З‘єднувальні лінії (Directed links)		Використовуються для переходу до певного кроку (пропуск декількох кроків, повторне виконання кроків).

Максимальна кількість кроків (основні кроки + кроки макрокроків) у секції Grafcet не повинна перевищувати 1024 для ПЛК TSX 57.

2.1.3 Об‘єкти мови Grafcet

Таблиця 7.2

Назва	Адреса	Опис
Біти кроків (1=активний крок) (Step bits)	%Xi	Стан кроку i (i від 0 до n, залежно від процесора).
	%XMj	Стан макрокроку j (j від 0 до 63 для TSX/PMX/PCX 57).
	%Xj.i	Стан кроку i макрокроку j.

	%Xj.IN	Стан вхідного кроку макрокроку j.
	%Xj.OUT	Стан вихідного кроку макрокроку j.
Системні біти (Grafcet system bits)	%S21	Ініціалізує Grafset секцію.
	%S22	Скидає всі секції Grafset в нуль.
	%S23	Заморожує Grafset секцію.
	%S24	Скидає макрокроки в нуль згідно з системними словами %SW22-%SW25.
	%S26	Встановлюється в 1 при: - переповненні таблиці (кроки/переходи); - виконання некоректної секції (направляючий конектор вказує на крок, який не належить до даної секції).
Слова кроків (Step words)	%Xi.T	Активний час кроку і Grafset секції.
	%Xj.i.T	Активний час кроку і макрокроку j.
	%Xj.IN.T	Активний час вхідного кроку макрокроку j.
	%Xj.OUT.T	Активний час вихідного кроку макрокроку j.
Системні слова (Grafcet system words)	%SW20	Слово, яке вказує кількість активних кроків, кроків для активації та кроків для деактивації для даного циклу.
	%SW21	Слово, яке вказує число доступних переходів, недоступних переходів та переходів, які будуть доступні для даного циклу.
	%SW22 – %SW25	Визначають макрокроки для скидання бітом %S24.

Біти кроків %Xi, макрокроків %XMj, кроків макрокроків %Xj.i, %Xj.IN, %Xj.OUT:

- Дорівнюють 1, коли кроки активні.
- Доступні для перевірки у всіх задачах, але записані можуть бути на етапі попереднього програмування (preprocessing) основної задачі мовами LD, IL, ST.
- Дані біти не можуть бути індексовані.

Слова %Xi.T, %Xj.i.T, %Xj.IN.T, %Xj.OUT.T:

- Змінюються від 0 до 9999, нарощуються кожні 100 ms.
- Зміна значень відбувається, коли відповідний крок активний.
- При деактивації відповідного кроку значення часу заморожується.
- При повторній активації кроку значення часу обнулюється, а потім починає наростати.
- Кожне слово для зберігання часу, на протязі якого активний крок, закріплюється за цим кроком.
- Дані слова не можуть бути індексовані.

2.1.4 Можливості мови Grafcet

Можливості мови Grafcet залежать від процесора ПЛК.

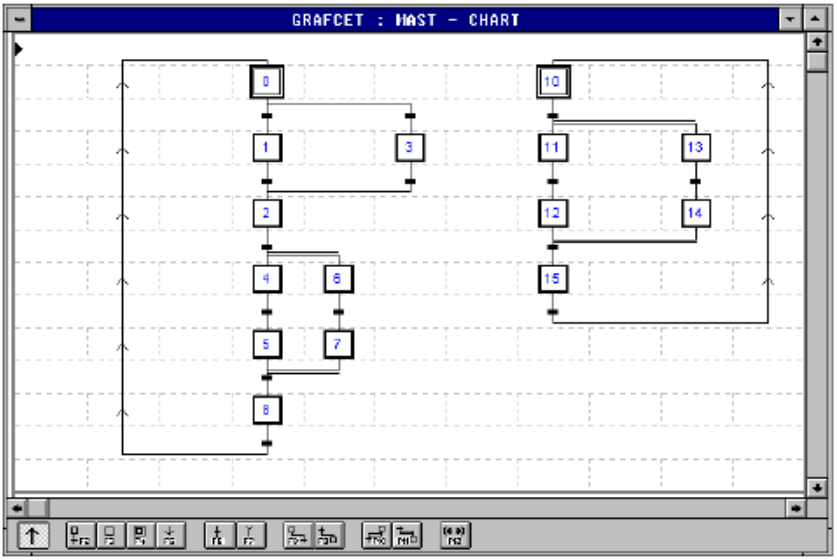
Таблиця 7.3

Number	TSX 37-10		TSX 37-20		TSX 57	
	Default	Max	Default	Max	Default	Max
Main chart steps	96	96	128	128	128	250
Macro-steps	0	0	0	0	8	64
Macro-step steps	0	0	0	0	64	250
Total steps	96	96	128	128	640	1024
Steps active simultaneously	16	96	20	128	40	250
Transitions enabled simultaneously	20	192	24	256	48	400

Кількість одночасно активних переходів не повинна перевищувати 64. Загальна кількість переходів не повинна перевищувати 1024.

2.1.5 Представлення секції Grafcet

Секція Grafcet програмується на 8 сторінках (від 0 до 7). Кожна сторінка має 14 рядків та 11 колонок, які розділяють її на 154 чарунок. Один графічний елемент займає одну чарунку.



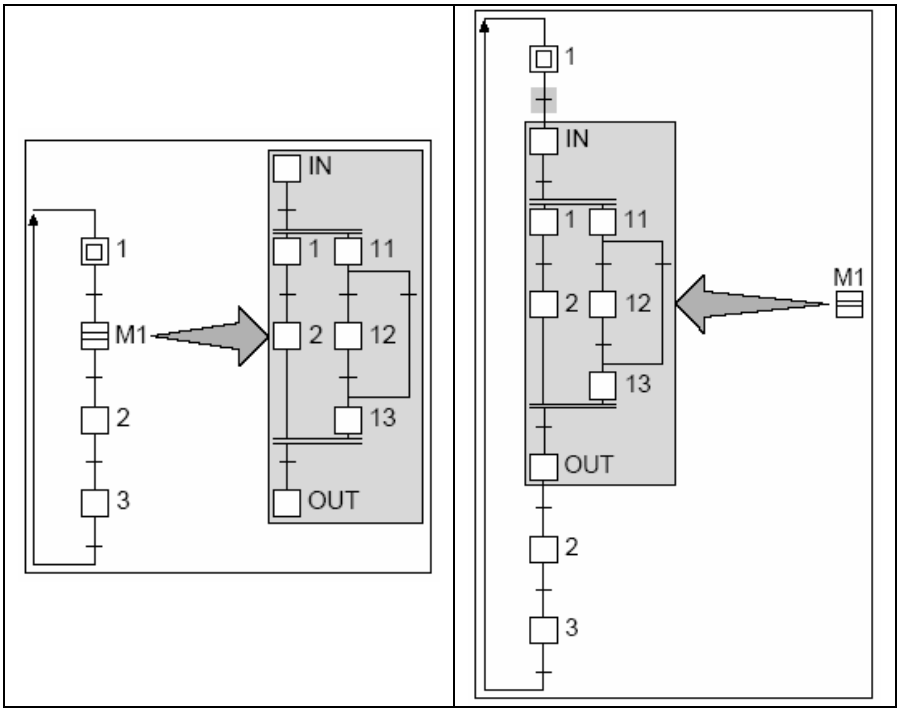
Правила написання програми:

- Перша лінія використовується для розміщення конектора-приймача.
- Остання лінія використовується для розміщення направляючого конектора.
- Парні лінії (від 2 до 12) служать для розміщення кроків і направляючих конекторів.
- Непарні лінії (від 3 до 13) служать для розміщення переходів і конекторів-приймачів.
- Кожен крок нумерується (від 0 до 127) у довільному порядку.
- Декілька програм може бути розміщено на одній сторінці.

2.1.6 Макрокроки (macro-steps)

Макрокроки – цілісна структура, що складається з кроків та переходів між ними. Макрокроки завжди починається з кроку IN і завершується кроком OUT. Крок IN підкоряється тим же правилам, що і одиничні кроки, а крок OUT не може мати асоційованих з ним дій.

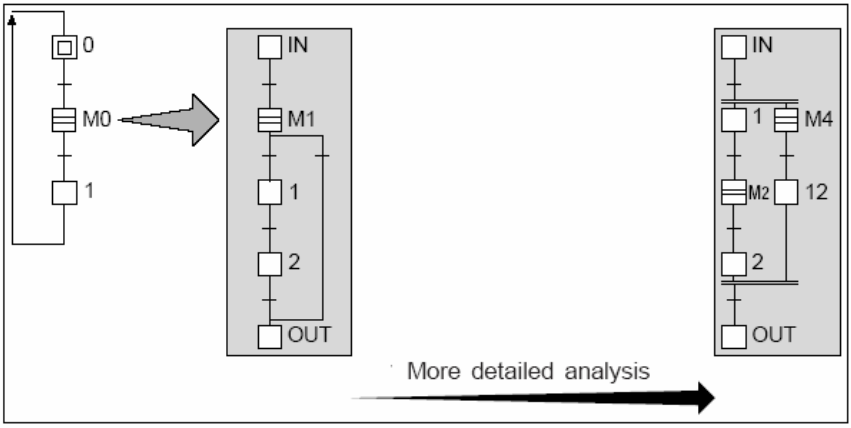
У прикладі, наведеному нижче, макрокрок M1 активізується тоді, коли крок 1 активний і істинним є перехід від кроку 1. Макрокроки M1 деактивується, коли його крок OUT активний і істинним є перехід M1 -> 2. Тоді активним стає крок 2.



У середовищі PL7 можна використати до 64 макрокроків (M0 – M63). Один макрокрок може бути запрограмований на 8 сторінках і містити до 250 кроків плюс крок IN та крок OUT. Один макрокрок може містити в собі інші макрокроки. Максимальне число ієрархічних рівнів – 64.

Макрокроки може містити 1 або кілька початкових кроків (initial steps). Вони активізуються під час запуску програми на виконання або ініціалізуються програмно. Тоді макрокрок стає активним.

Таким чином, використання макрокроків дозволяє спростити структуру основної секції Grafcet, зробити її більш придатною для читання і розуміння.



2.1.7 Дії, асоційовані з кроками

Кожен крок має асоційовані з ним дії, запрограмовані мовами LD, IL або ST. Ці дії виконуються тоді, коли крок, до якого вони належать, активний. Середовище PL7 надає можливість створювати наступні типи дій:

- дії по активації (actions on activation): дії, які виконуються 1 раз під час активації кроку, до якого вони належать;
- дії по деактивації (actions on deactivation): дії, які виконуються 1 раз під час деактивації кроку, до якого вони належать;
- тривалі дії (continuous actions): дії, які виконуються постійно на протязі всього активності кроку, до якого вони належать.

Дані 3 типи дій можуть бути присвоєні для кожного кроку.

Кожна дія являє собою програму, назва якої має наступний вигляд:

MAST - <Grafcet section name> - CHART (or MACROk)- PAGE n %Xi x

x = P1 – дія по активації;

N1 – тривала дія;
P0 – дія по деактивації;
n = номер сторінки;
i = номер кроку, до якого належить дія.

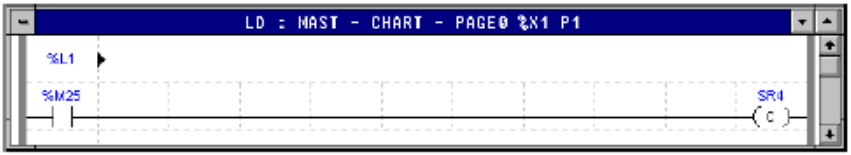
Приклад: MAST - Paint - CHART - PAGE 0 %X1 P1
Дія по активації кроку 1, який розміщений на сторінці 0 секції Paint.

Дії по активації та деактивації кроку

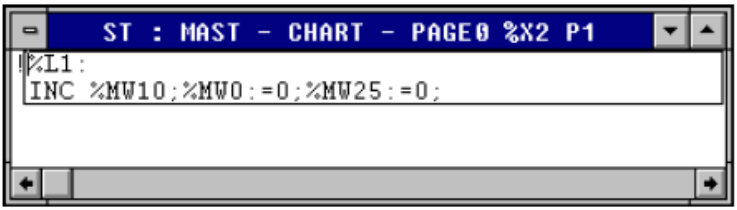
Виконуються тільки 1 раз за кожний цикл програми. Використовуються для виклику підпрограм, нарощування значення лічильника тощо.

Приклади:

- Виклик підпрограми:



- Інкремент слова %MW10, скидання в нуль слів %MW0 та %MW25.



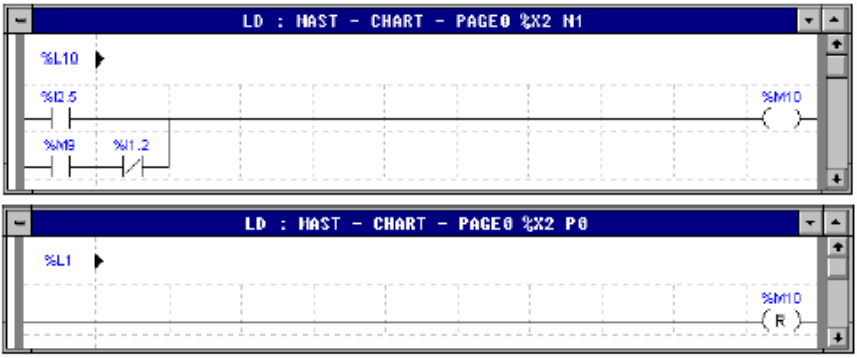
Тривалі дії

- **Дії з умовою (Conditional action)**

Приклад:

Біт %M10 керується дискретним входом %I2.5, внутрішнім бітом %M9 та дискретним входом %I1.2. Поки активний крок 2 буде виконуватися дана дія. Після деактивації кроку 2 значення біта

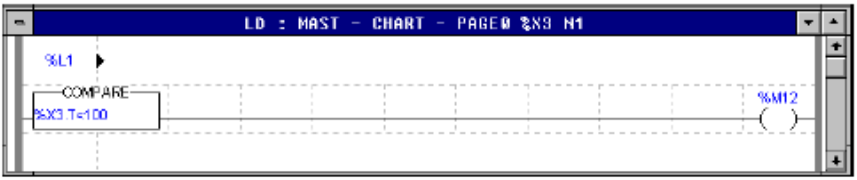
%M10 запам‘ятовується і лишається без змін. Тому необхідно скинути біт %M10 в нуль у дії по деактивації кроку 2.



- **Дії з умовою за часом (Timed conditional action)**

Приклад:

Біт %M12 буде рівним 1, коли час виконання кроку 3 буде меншим за 10 секунд (базовий час становить 100 мс).



- **Безумовні дії (Unconditional actions)**

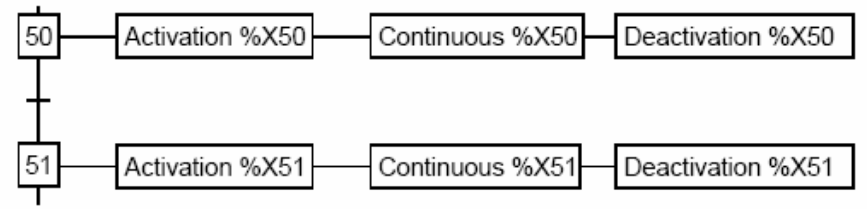
Порядок виконання дій

Приклад:

Коли крок 51 стає активним, дії виконуються у такій послідовності:

- дії по деактивації кроку 50;
- дії по активації кроку 51;
- тривалі дії кроку 51.

Після деактивації кроку 51 тривалі дії, асоційовані з ним, перестають виконуватися.



2.1.8 Дії, асоційовані з переходами

- Кожен перехід має асоційовану з ним умову, яка програмується мовами LD, IL або ST.
- Умова переходу перевіряється ПЛК тоді, коли перехід стає активним.
- Перехід, який не запрограмований, завжди хибний, тобто його результат дорівнює false.

Кожен перехід має назву, яка має наступний вигляд:

MAST - <Grafcet section name> - CHART (or MACROk) - PAGE n %X(i) --> % X(j)

n = номер сторінки;
i = номер кроку, від якого здійснюється перехід;
j = номер кроку, до якого здійснюється перехід.

Приклад: MAST - Paint - CHART - PAGE 0 %X(0) --> %X(1)

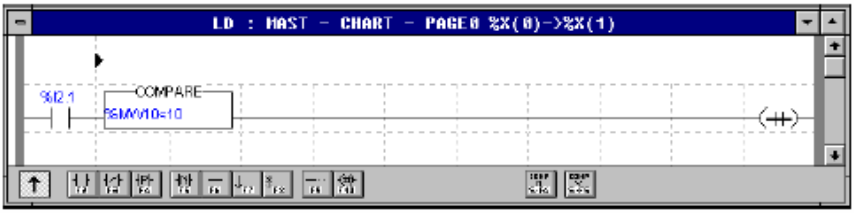
Умова переходу від кроку 0 до кроку 1 на сторінці 0 секції Paint.

Якщо активізуються одночасно декілька кроків, друга адреса є адресою першого кроку зліва.

Правила програмування переходів мовою драбинкових діаграм LD

Програма має вигляд стандартного рангу, як і у звичайній LD програмі. Але можуть використовуватися тільки такі елементи:

- контакти (%Mi, %I, %Q, %TMi.D), блоки порівняння;
- котушки переходу (інші види котушок не важливі у даному випадку).



Правила програмування переходів мовою послідовних інструкцій IL

Програма у даному випадку являє собою список інструкцій тільки тестового типу. Програма для переходу відрізняється від стандартної IL програми наступним:

- відсутня мітка %L;
- відсутні інструкції-дії (дискретні виходи, слова функціональних блоків);
- відсутні переходи на мітки, виклики підпрограм.

Правила програмування переходів мовою структурованого тексту ST

Умова переходу являє собою логічний, арифметичний вираз або їх комбінацію. Така програма відрізняється від стандартної ST програми наступним:

- відсутня мітка %L;
- відсутні арифметичні, умовні операції, циклічні оператори;
- відсутні дії над бітами;
- відсутні переходи на мітки, виклик підпрограм;
- відсутні операції з функціональними блоками.

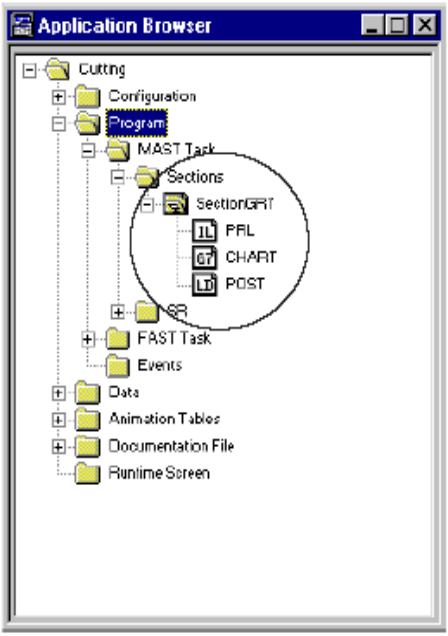
Умова переходу за часом активності кроку

Бувають випадки, коли необхідно, щоб окремий крок виконувався протягом певного інтервалу часу (робота мішалки, випікання хліба, вмикання світла, випалювання цегли тощо).

Приклад:

Необхідно, щоб крок 3 був активним на протязі 15 секунд. Тоді умова переходу від кроку 3 до кроку 4 мовою структурованого тексту матиме вигляд:

2.1.9 Структура Grafcet секції



Програма мовою Grafcet складається з таких послідовно виконуваних секцій:

- початкова частина програми (preprocessing – Prl) – мовою LD, IL або ST;
- основна послідовність (sequential processing – Chart) – мовою Grafcet;
- заключна частина програми (post-processing – Post) – мовою LD, IL або ST.

Grafcet секція завжди належить до MAST задачі.

2.2 Методи керування процесом приготування суміші із сипучих матеріалів

Приготування суміші із сипучих матеріалів є важливим процесом у багатьох галузях промисловості, включаючи будівництво, хімію, фармацевтику та харчову промисловість. Якість кінцевого продукту значною мірою залежить від правильного змішування компонентів, тому ефективне керування цим процесом є надзвичайно важливим. Існує кілька основних методів, які забезпечують оптимальне приготування сумішей.

Перший метод - це механічне змішування, яке виконується за допомогою різноманітних змішувачів, таких як барабанні, лопатеві та стрічкові змішувачі. Механічні змішувачі забезпечують рівномірне перемішування компонентів, що є ключовим для отримання однорідної суміші. Керування процесом механічного змішування досягається через регулювання швидкості обертання, тривалості змішування та порядку додавання компонентів.

Автоматизовані системи дозволяють програмувати та контролювати ці параметри, забезпечуючи стабільну якість суміші.

Другий метод - це пневматичне змішування, яке базується на використанні стисненого повітря для перемішування матеріалів. Цей метод особливо ефективний для змішування легких та дрібнозернистих матеріалів, які можуть утворювати агломерати. Керування процесом пневматичного змішування включає регулювання тиску та потоку повітря, що забезпечує рівномірне перемішування та запобігає утворенню грудок. Автоматизація цього процесу дозволяє точно контролювати умови змішування, підвищуючи якість кінцевого продукту.

Третій метод - це використання ультразвукових змішувачів, які створюють високочастотні вібрації для перемішування матеріалів. Ультразвукове змішування забезпечує дуже високу ступінь однорідності, що важливо для фармацевтичних та хімічних сумішей. Керування процесом ультразвукового змішування здійснюється через регулювання частоти та амплітуди вібрацій. Автоматизовані системи дозволяють налаштовувати ці параметри відповідно до вимог процесу, забезпечуючи оптимальні умови для змішування.

Четвертий метод - це змішування у рідкому середовищі, яке застосовується для приготування сумішей з матеріалів, що можуть бути розчинені або зважені в рідині. Цей метод забезпечує рівномірне розподілення компонентів у рідкому середовищі, що особливо важливо для хімічних та харчових продуктів. Керування процесом включає регулювання швидкості перемішування, температури та концентрації компонентів. Автоматизовані системи дозволяють точно контролювати ці параметри, забезпечуючи високу якість кінцевого продукту.

Нарешті, важливу роль у керуванні процесом приготування сумішей відіграють програмні засоби, які дозволяють моделювати та оптимізувати процеси. Використання спеціалізованого програмного забезпечення допомагає розраховувати оптимальні параметри змішування для різних матеріалів, знижуючи витрати та підвищуючи ефективність.

Таким чином, ефективне керування процесом приготування суміші із сипучих матеріалів потребує комплексного підходу, що включає вибір відповідного методу змішування, автоматизацію

процесів та використання сучасного програмного забезпечення. Це дозволяє забезпечити високу якість кінцевого продукту, зменшити витрати та підвищити продуктивність виробництва.

3. Програма роботи

1. Навчитися програмувати ПЛК Schneider Micro TSX 37-22 мовою Grafcet у середовищі програмування PL7 PRO.
2. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу приготування суміші з трьох компонентів.
3. Створити другий рівень системи контролю і керування технологічним процесом засобами програмного забезпечення PL7 PRO.
4. Перевірити роботу програми.

4. Опис лабораторного обладнання

1. Персональний комп‘ютер.
2. Операційна система Windows.
3. Лабораторний стенд стенд з ПЛК Schneider Micro TSX 37-22.
4. Середовище програмування ПЛК Schneider Micro TSX 37-22 PL7 PRO, середовище програмування операторської панелі Magelis XBT-L-1000.

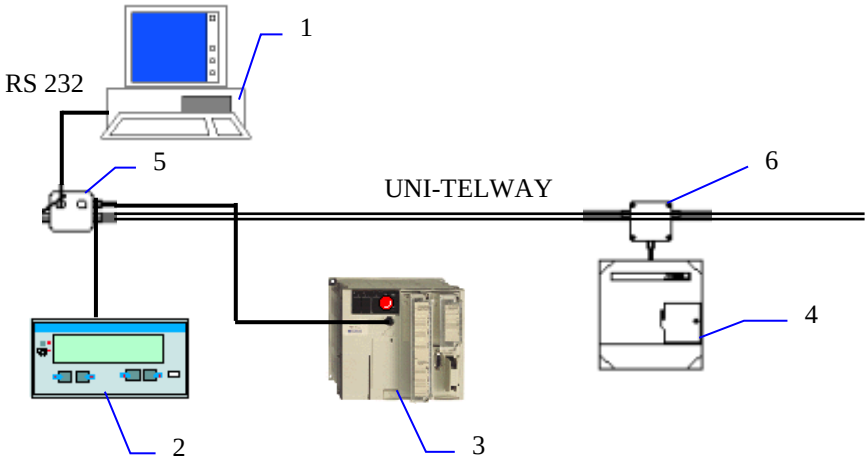


Рис. 7.1 Лабораторний стенд: 1- АРМ оператора; 2 - операторська панель Magelis XBT P 012010;
3 - ПЛК Schneider Micro TSX 37-22; 4 - частотний перетворювач Altivar 58; 5 - адаптер відгалуженняTSX P ACC01; 6 - розгалуджувач TSX SCA 62.

5. Порядок виконання роботи

1. Під‘єднати контролер Schneider Micro TSX 37-22 до COM-порту комп‘ютера за допомогою комунікаційного кабелю TSX PCX 1031. Ввімкнути живлення стенду.
2. Запустити середовище PL7 PRO.
3. Написати програму для контролера Schneider Micro TSX 37-22 для автоматизації технологічного процесу приготування суміші з трьох компонентів мовою програмування Grafcet.

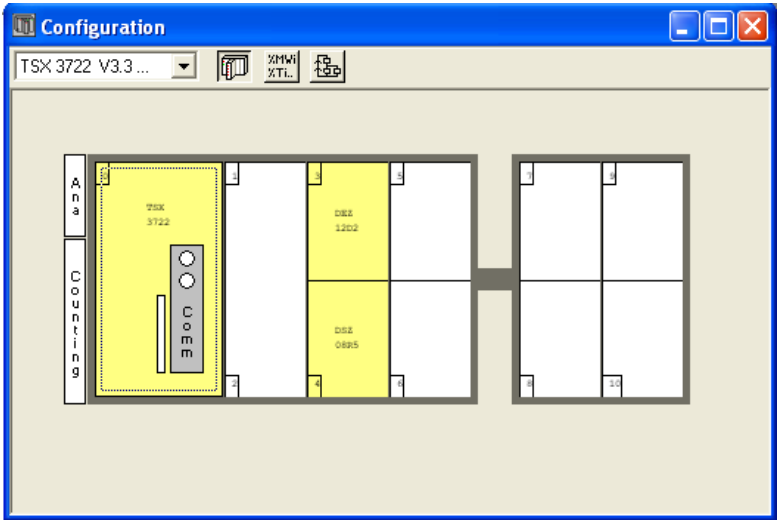


Рис. 7.2 Підключення та конфігурування модулів вводу-виводу

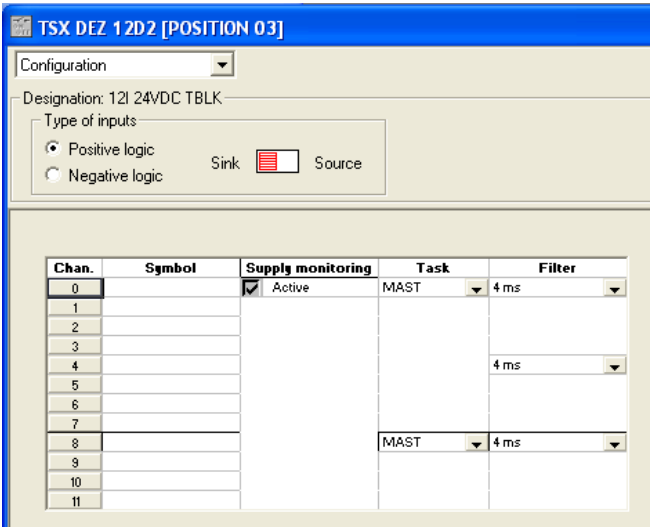


Рис. 7.3 Модуль дискретного вводу у слоті №3

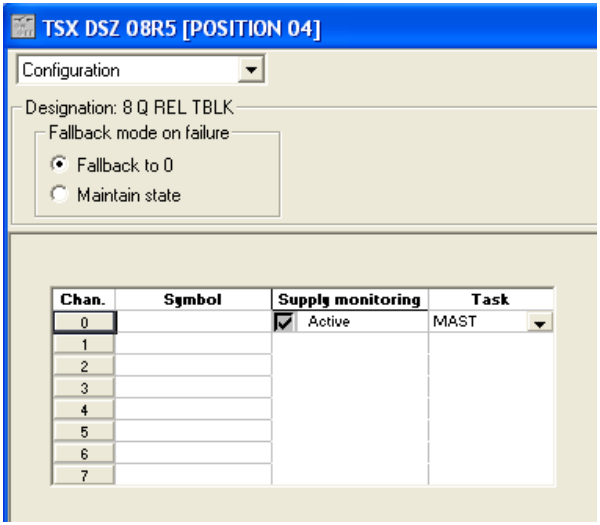


Рис. 7.4 Модуль дискретного виводу у слоті №4

4. Створити анімаційну таблицю для відслідковування значень використаних у програмі змінних. Наряду з анімаційною таблицею використати операторську панель Magelis.

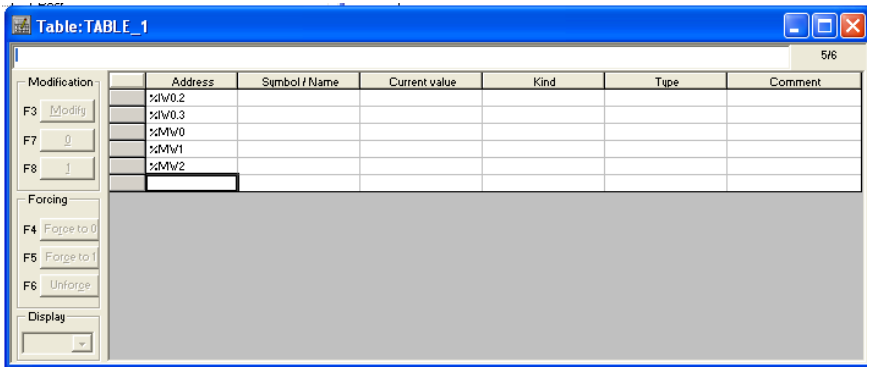


Рис. 7.5 Анімаційна таблиця

5. Створити вікно супервізора (Run-Time screen) засобами програмного забезпечення PL7 PRO, в якому намалювати ФСА технологічного процесу.

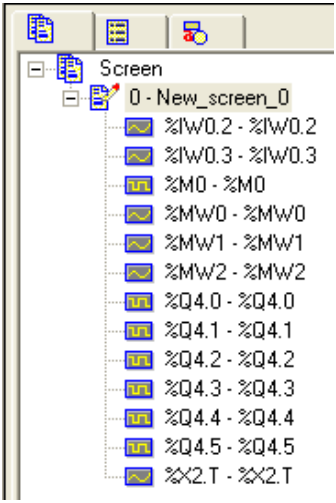
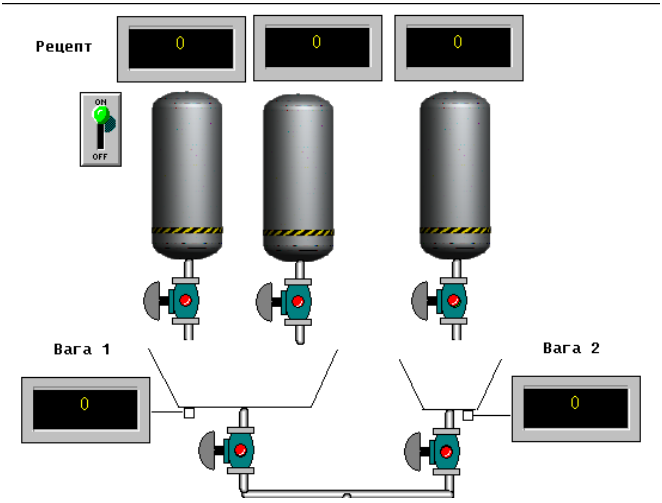


Рис. 7.6 Прив‘язка змінних у вікні супервізора



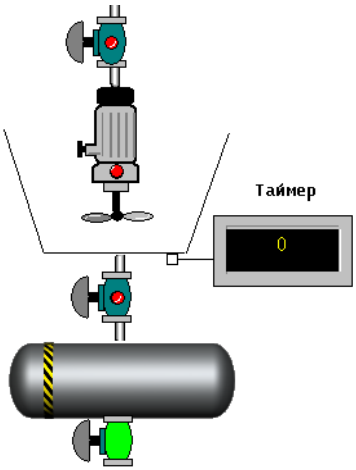


Рис. 7.7 Вікно супервізора

6. Записати програму у контролер. Для цього виконати наступні дії:
- виконати команду Conect;
- вибрати напрям запису програми PC -> PLC;
- перевести контролер у режим RUN.
7. Провести моніторинг роботи програми за допомогою анімаційної таблиці, вікна супервізора.

Опис технологічного процесу та вимоги до програми:

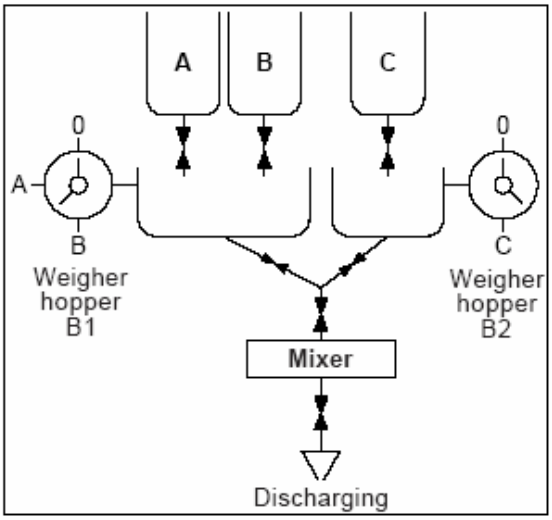


Рис. 7.8 Технологічна схема змішування трьох компонентів

Продукти А і В надходять у бункер В1 і зважуються. Продукт С зважується у бункері В2. Із бункерів В1 і В2 речовини надходять у змішувач, де змішуються на протязі часу, встановленого користувачем. Як тільки час змішування спливає, змішувач зупиняється. Утворена суміш вивантажується, якщо надходить дозволяючий сигнал від оператора.

Функціональний аналіз процесу

Даний процес можна розбити на такі частини:

- зважування трьох компонентів;
- наповнення мішалки;
- змішування трьох компонентів у мішалці;
- відвантажування утвореної суміші.

На рис. 7.9 представлені основні етапи процесу.

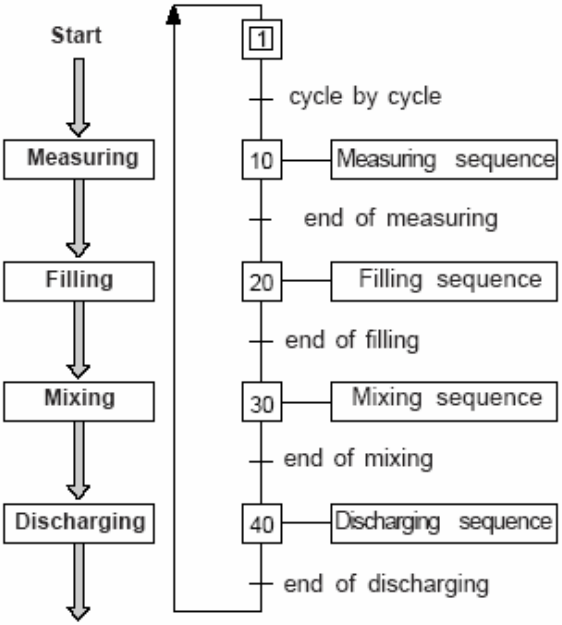


Рис. 7.9 Основні етапи процесу приготування суміші із трьох компонентів

При проведенні більш детального аналізу кожен основний етап можна теж розбити на стадії. Після цього можна приступати до розробки програмного забезпечення.

Складемо таблицю входів-виходів:

Вхід-вихід	*Призначення*
%I3.0	Кнопка запуску процесу
%M0	Команда запуску процесу
%MW0	Уставка ваги компонента A
%MW1	Уставка ваги компонента B
%MW2	Уставка ваги компонента C
%IW0.2	Давач ваги бункера 1
%IW0.3	Давач ваги бункера 2
%Q4.0	Клапан A
%Q4.1	Клапан B
%Q4.2	Клапан C

%M1	Біт переходу з кроку №0 на крок №1
%Q4.3	Клапани з вагових бункерів у мішалку
%M2	Біт переходу з кроку №1 на крок №2
%Q4.4	Запуск мішалки
%M3	Біт переходу з кроку №2 на крок №3
%Q4.5	Клапан на виході мішалки
%M4	Біт переходу з кроку №3 на крок №0

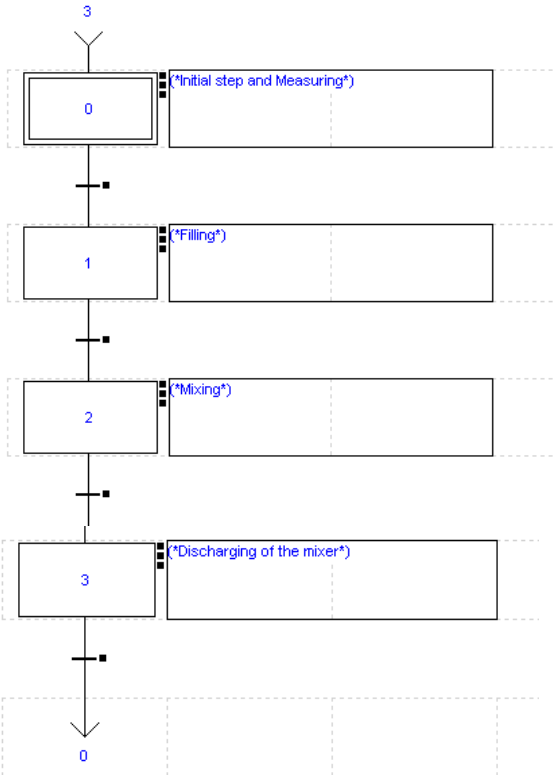
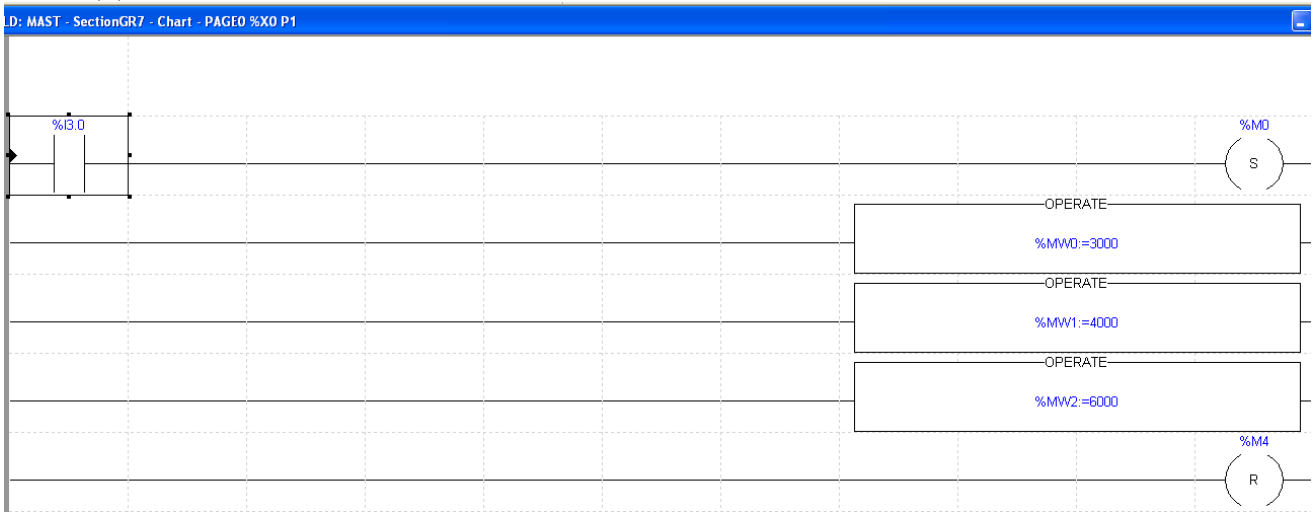


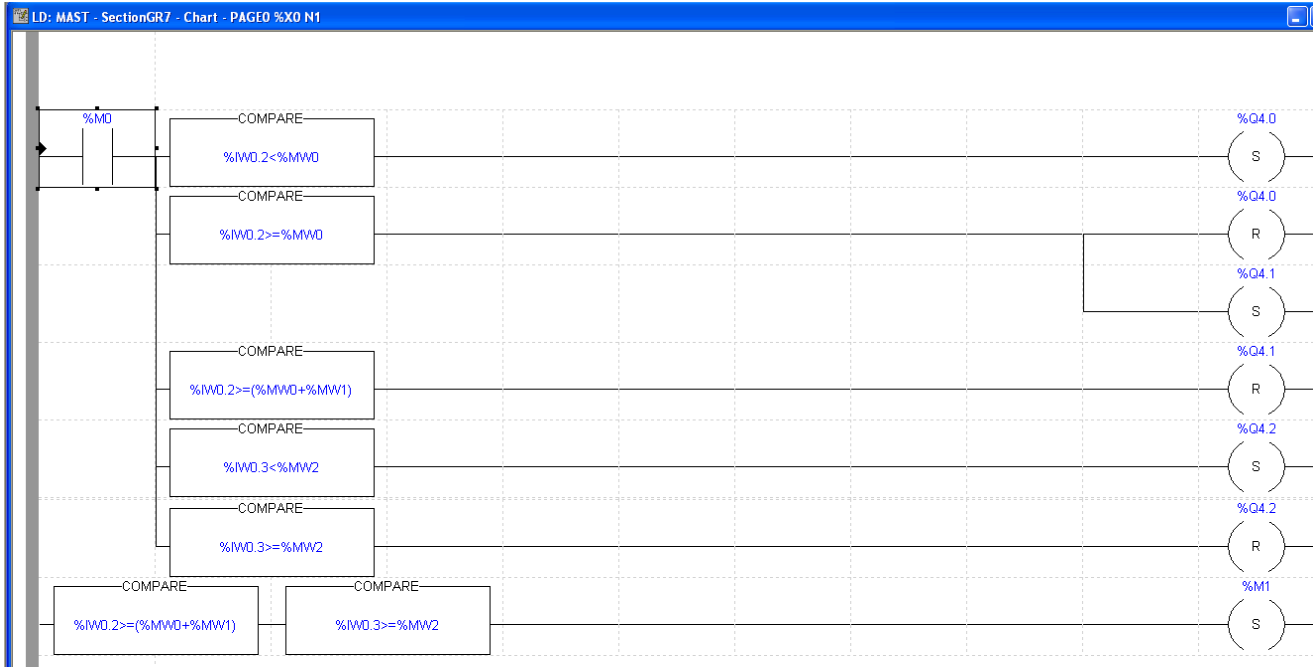
Рис. 7.10 Блок-схема програми мовою Grafcet

Крок №0

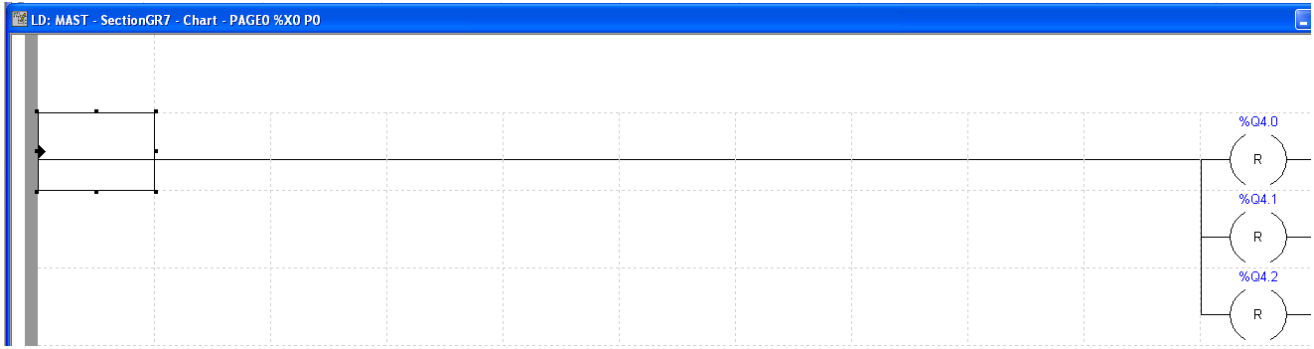
Дія по активації:



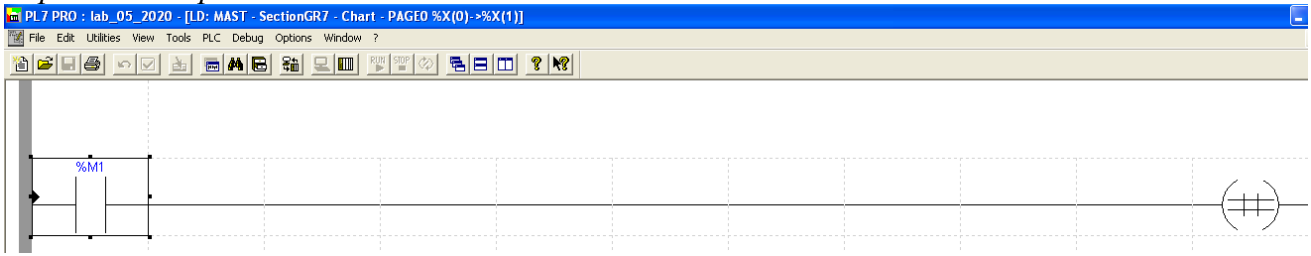
Тривала дія:



Дія по деактивації:

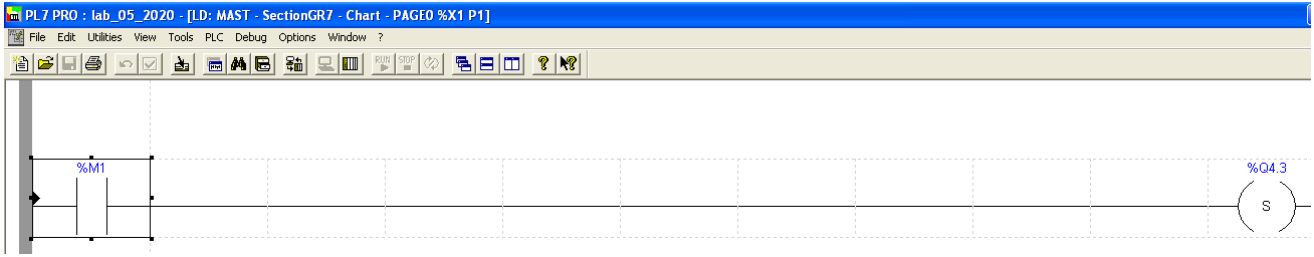


Перехід між кроками №0 та №1:

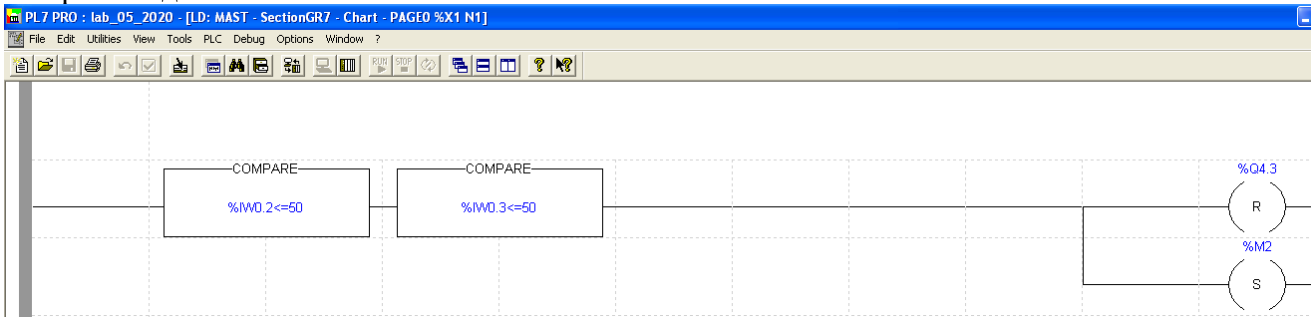


Крок №1

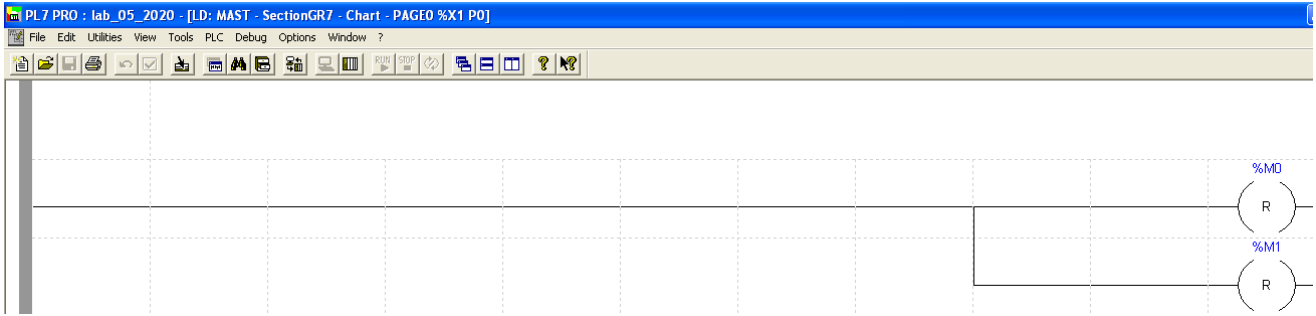
Дія по активації:



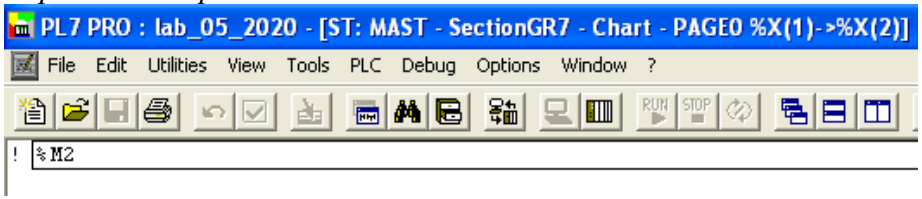
Тривала дія:



Дія по деактивації:

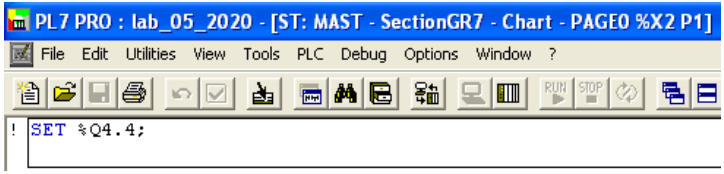


Перехід між кроками №1 та №2:

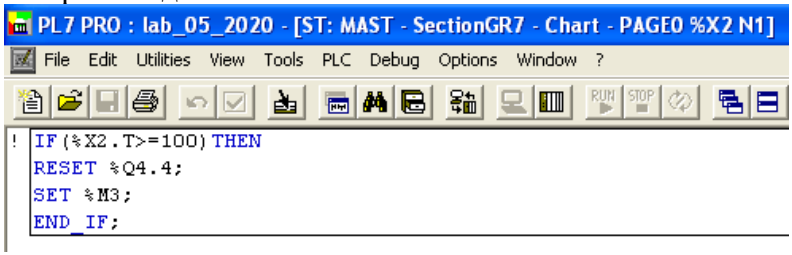


Крок №2

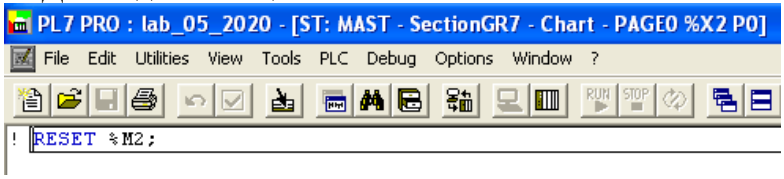
Дія по активації:



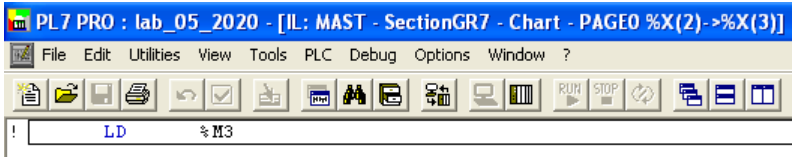
Тривала дія:



Дія по деактивації:

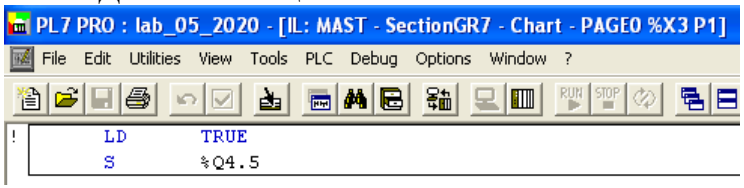


Перехід між кроками №2 та №3:

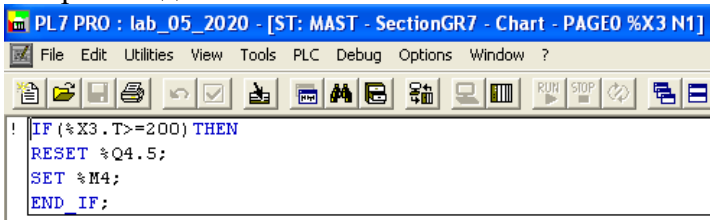


Крок №3

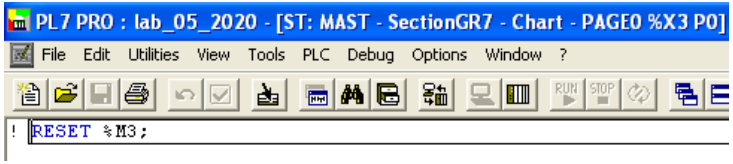
Дія по активації:



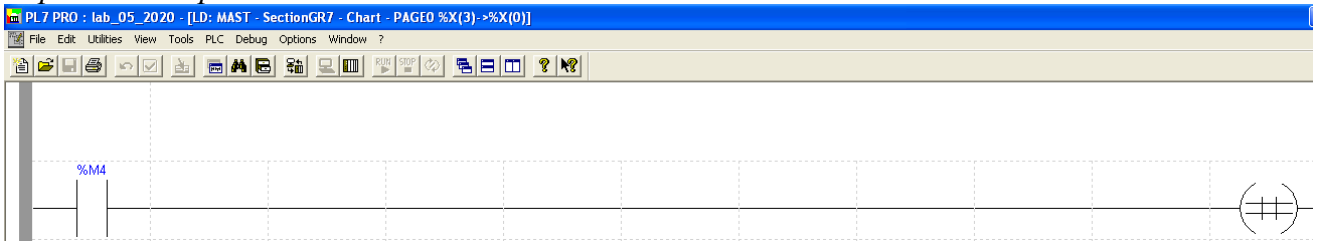
Тривала дія:



Дія по деактивації:



Перехід між кроками №3 та №0:



6. Контрольні запитання

1. Назвіть основні принципи написання програм мовою Grafcet.
2. Які є графічні елементи для написання програм мовою Grafcet?
3. Які є види і якими мовами можна запрограмувати дії для кроків мови Grafcet? Поясніть їх призначення.
4. Які правила програмування переходів між кроками у мові Grafcet?
5. З яких частин складається програмна секція мовою Grafcet? Поясніть призначення кожної з них.
6. Для автоматизації яких процесів найкраще використовувати мову Grafcet при розробці програмного забезпечення для ПЛК?
7. Назвіть основні характеристики SCADA- системи Citect.
8. Перерахуйте та дайте характеристику основним стадіям розробки нового проекту в SCADA- системі Citect.
9. Яким чином налаштувати зв‘язок між пристроями введення-виведення і SCADA- системою Citect?
10. Якими способами можна відредагувати базу даних змінних у SCADA- системі Citect? Дайте характеристику кожному способу.
11. Які засоби візуалізації ходу технологічного процесу є у SCADA- системі Citect?
12. Яким чином можна створити мнемосхему технологічного процесу у SCADA- системі Citect?
13. Яким чином налаштувати об‘єкт джин у SCADA- системі Citect?
14. Яким чином налаштувати об‘єкт суперджин у SCADA- системі Citect?
15. Яким чином відобразити значення технологічного параметру у SCADA- системі Citect?
16. Яким чином створити кнопку та присвоїти їй виконання певної дії у SCADA- системі Citect?
17. Яким чином створити тренд та вивести на нього значення технологічного параметру у SCADA- системі Citect?