МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

М.О. Подустов, А. К. Бабіченко, І. Г. Лисаченко,
О. Г. Шутинський, В. О. Лобойко, О. М. Дзевочко, О.В. Пугановський

ПРОГРАМУВАННЯ ПРОМИСЛОВИХ КОНТРОЛЕРІВ VIPA В СЕРЕДОВИЩІ WINPLC V5

Навчальний посібник
для студентів спеціальності
«Автоматизація та комп'ютерно-інтегровані технології»

Затверджено Вченою радою НТУ «ХПІ»

Харків
НТУ «ХПІ»
2018

УДК 004.415:681.51(075.8)

П00

А в т о р с ь к и й к о л е к т и в:

М.О. Подустов, д-р техн. наук, проф., А.К. Бабіченко, канд. техн. наук, проф., І.Г. Лисаченко, канд. техн. наук, доцент, В.О. Лобойко, канд. техн. наук, доцент, О.Г. Шутинський, канд. техн. наук, доцент, О.М. Дзевочко, канд. техн. наук, доцент, О.В.Пугановський

Р е ц е н з е н т и :

І.О. Фурман, д-р техн. наук, проф. ХНТУСГ;

Л.І. Нефьодов, д-р. техн. наук, проф. ХНАДУ

Затверджено Вченою радою НТУ «ХПІ» як навчальний посібник для студентів спеціальності «Автоматизація та комп'ютерно-інтегровані технології», протокол № 6 від 06.07.2018 р.

П 00 **Програмування** промислових контролерів VIPA в середовищі WinPLC V5: навч. посібник / Подустов М.О., Бабіченко А.К., Лисаченко І.Г. та ін. – Х.: Вид-во «НТУ "ХПІ"», 2018. – 195 с.

ISBN 000-000-000-000-0

У посібнику надано стислу характеристику та порядок застосування промислових контролерів VIPA та їхнього середовища програмування WinPLC V5. Описано порядок розроблення прикладного програмного забезпечення для систем управління технологічними процесами. Наведені приклади та практичні рекомендації щодо застосування апаратних та програмних засобів VIPA.

Призначено для студентів спеціальності «Автоматизація та комп'ютерно-інтегровані технології» усіх форм навчання. Посібник також буде корисний для спеціалістів, котрі працюють у галузі промислової автоматизації.

Іл. 138. Табл. 39. Бібліогр. 17 назв.

УДК 004.415:681.51(075.8)

ISBN 000-000-000-000-0

ЗМІСТ

ВСТУП. 5
РОЗДІЛ 1 ЗАГАЛЬНІ ВІДОМОСТІ ПРО АПАРАТНЕ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ЗАСОБІВ ВИРОБНИЦТВА *VIPA*. 7
1.1 Загальні відомості про апаратно-програмне забезпечення засобів виробництва *VIPA*. 7
1.2 Апаратне забезпечення засобів виробництва *VIPA*. 10
1.2.1 Опис контролерів *VIPA System 100V*. 10
1.2.2 Опис контролерів *VIPA System 200V*. 22
1.2.3 Опис контролерів *VIPA серії System 300*. 30
1.3 Програмне забезпечення засобів виробництва *VIPA*. 40
1.4 Приклад розроблення ППЗ для установки водопостачання. . . . 64
РОЗДІЛ 2 ПРАКТИКУМ З РОЗРОБЛЕННЯ ППЗ ДЛЯ ПЛК *VIPA* В СЕРЕДОВИЩІ *WINPLC7 V5*. 75
2.1 Розроблення ППЗ для системи дискретного управління водонагрівачем. 75
2.1.1 Апаратна складова проекту для *VIPA115*. 79
2.1.2 Програмна складова проекту для *VIPA115*. 84
2.2 Розроблення ППЗ для системи водопостачання з використанням таймерів та лічильників. 85
2.3 Розроблення ППЗ для системи аналогового керування вихідним пристроєм водонагрівача. 92
2.4 Розроблення ППЗ для системи управління водонагрівачем з аналоговим датчиком. 97
2.4.1 Апаратна складова проекту для *VIPA214*. 99
2.4.2 Програмна складова проекту для *VIPA214*. 103
2.5 Розроблення ПІД-регулятора температури для системи гарячого водопостачання 106

2.5.1 Апаратне конфігурування ПІД-регулятора для *VIPA313SC*. .113
2.5.2 Програмна складова ПІД-регулятора для *VIPA313SC*. 118
РОЗДІЛ 3 ІНТЕГРАЦІЯ КОНТРОЛЕРІВ *VIPA* ТА ЗАСОБІВ ЛЮДИНО-МАШИННОГО ІНТЕРФЕЙСА. 130
3.1 Розроблення ЛМІ на основі взаємодії ОП *VIPA OP03* та ПЛК *VIPA115* для системи гарячого водопостачання. 130
3.2 Реалізація протоколу *Modbus RTU/ASCII* в контролерах *VIPA*. . . 137
3.2.1 Реалізація протоколу *Modbus* в ПЛК *VIPA115* як головного пристрою. 139
3.2.2 Реалізація протоколу *Modbus* в ПЛК *VIPA115* як підлеглого пристрою. 149
3.3 Реалізація шини *PROFIBUS-DP* в контролерах *VIPA*. 150
3.3.1 Апаратна структура PROFIBUS-мережі. 153
3.3.2. Програмування головного пристрою в мережі *PROFIBUS*. .159
3.4 Реалізація обміну даними між ПЛК *VIPA* за допомогою глобальних блоків даних 165
РОЗДІЛ 4 РОЗРОБЛЕННЯ ППЗ ДЛЯ СИСТЕМИ УПРАВЛІННЯ УСТАНОВКОЮ ЗМІШУВАННЯ РЕЧОВИН НА ЗАСОБАХ *VIPA*. 169
Список літератури. 190

ВСТУП

Навчальний посібник призначений для вивчення будови та основ програмування контролерів *VIPA* в середовищі *WinPLC V5* та проведення комп'ютерного практикуму зі студентами денної та заочної форм навчання спеціальності 151 – «Автоматизація та комп'ютерно-інтегровані технології».

Посібник складається з чотирьох розділів. В першому розділі викладені основні правила роботи в середовищі *WinPLC V5* для програмування промислових контролерів *VIPA* виробництва Німеччина [1, 2]. Методика вивчення програмного забезпечення передбачає самостійне ознайомлення з відомостями, яки викладені у першому розділі. Другий та інші розділи присвячений власне виконанню завдань комп'ютерного практикуму.

Перший розділ вміщує необхідні теоретичні відомості про середовище програмування контролерів та опис їхньої будови, тобто апаратну структуру та технічні характеристики. Для прикладу у першому розділі описані склад та технічні можливості контролерів *VIPA* серій *System100V, System200V* та *System300S* [3].

Виконання комп'ютерного практикуму з другого розділу дозволяє студентам більш ефективно засвоїти в подальшому теорію та практику застосування середовища *WinPLC V5* для розроблення прикладного програмного забезпечення (ППЗ) для локальних систем керування технологічними установками та обладнанням. Це забезпечується комплексним підходом до виконання окремих завдань практикуму. Тобто спочатку студенти виконують типові завдання з розроблення ППЗ, яке реалізує різноманітні алгоритми керування та функції управління, які притаманні інформаційно-управляючим компонентам.

Отримання навичок з розроблення ППЗ побудовано за принципом поступового ускладнення. В комп'ютерному практикумі розглянуті питання дискретного керування установкою, використання таймерів та лічильників, оброблення сигналів від аналогових датчиків, використан-

ня складних законів регулювання (П, ПІ та ПІД) тощо.

У третьому розділі студенти самостійно вирішують індивідуальні завдання та конфігурують операторські панелі (ОП) для налаштування зв'язку з програмованими логічними контролерами (ПЛК). Наведена методика створення людино-машинного інтерфейсу (ЛМІ), тобто організація взаємодії промислових контролерів з панелями оператора. Буде розглянуто порядок конфігурування панелі та налаштування зв'язку з контролером за допомогою послідовного інтерфейсу за протоколом *MPI*. Також розглянуті питання організації обміну даними за протоколами *Modbus, MPI,* а також налаштування мережі *PROFIBUS DP*.

В останньому розділі наведена методика створення ППЗ системи керування для реактора зі змішувачем для управління безперервним технологічним процесом.

Для розроблення, завантаження і налагодження ППЗ для програмованих логічних контролерів (ПЛК) *VIPA* усіх моделей використовується єдине середовище програмування *WinPLC V5*. Необхідно відмітити, що вказане середовище повністю відповідає вимогам стандарту стосовно технологічного програмування IEC 61131-3 [4, 5]. При цьому для налагодження проектів користувача може використовуватися як реальний контролер, так і емулятор ПЛК, який може бути запущений на ПК користувача (так звана станція розробника ППЗ). Програма конфігурування ОП також має емулятор, тому не потрібна для тестування взаємодії з програмою користувача в ПЛК. Контролери входять до складу стендів з симуляторами вхідних та вихідних сигналів, які дозволяють імітувати роботу системи управління технологічним об'єктом (процесом).

Опис основних принципів роботи з контролерами та операторськими панелями виробництва *VIPA Gmbh на*ведено в технічних керівництвах [6].

Насамкінець зазначимо, що обладнання та програмне забезпечення для комп'ютерного практикуму надано компанією VIPA через свого офіційного представника «СВ-Альтера» (м. Харків) [2] *відповідно до програми підтримки вищих навчальних закладів за кошти, які складають 10 % від реальної ціни.*

РОЗДІЛ 1
ЗАГАЛЬНІ ВІДОМОСТІ ПРО ПРОГРАМНЕ ТА АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ ЗАСОБІВ ВИРОБНИЦТВА *VIPA*

1.1 Загальні відомості про апаратно-програмне забезпечення засобів виробництва *VIPA*

Під час розроблення та впровадження апаратно-програмних засобів (АПЗ) в комп'ютерно-інтегровані системи управління технологічними процесами та виробництвами виникла необхідність створення стандарту, що регламентує усі аспекти створення та використання апаратно-програмних засобів в інформаційно-керуючих системах управління технологічними процесами та виробництвами (ІКСУ ТП та В). Важливим результатом цієї праці було створення стандарту IEC 61131. Мета впровадження цього стандарту полягає в тому, щоб визначити та стандартизувати конструкцію та функціональні можливості програмованих логічних контролерів, а також описати технологічні мови програмування, щоб користувач міг застосовувати різні системи автоматизації процесів та виробництв без будь-яких труднощів.

Для підтримки цього стандарту після початкового ухвалення була сформована велика група зацікавлених осіб та організацій, так звана *PLCopen*. Велика кількість головних виробнгиків ПЛК стали членами цієї асоціації, наприклад, такі компанії, як *VIPA*, *Siemens*, *Schneider Electric*, *ABB*, *GE Fanuc*, *Mitsubishi Electronic*, *Moeller*, *Omron*, *Rosemount*. Більшість компаній – членів асоціації пропонує АПЗ, які підтримують цей стандарт. Вважається, що технологічні мови програмування стануть основними в галузі промислової автоматизації, а контролери – основним засобом керування технологічними процесами.

Стандарт IEC 61131 [4] складається з п'яти частин, які на державному рівні без редагування та перекладу затверджені в Україні:

- Частина 1. Загальна інформація.

- Частина 2. Вимоги до устаткування та тестування.
- Частина 3. Мови програмування.
- Частина 4. Керівництво користувача.
- Частина 5. Засоби комунікації.

Частини 1…3 з цього стандарту були прийняті без редагування як європейський стандарт EN 61131.

Основним апаратним засобом ІКСУ є «вільно програмований контролер», який згідно зі стандартом IEC 61131-1 визначається так:

Цифрова електронна система, яка призначена для застосування в промисловості та використовує програмовану пам'ять для внутрішнього зберігання орієнтованих на користувача програм для того, щоб здійснити певні логічні функції, послідовність управління, вибір інтервалу часу, виконання розрахунку та інших математичних дій, управління через цифрові або аналогові входи і виходи різними типами устаткування або процесами.

Контролери та пов'язані з ними периферійні пристрої розроблені так, щоб вони могли бути легко об'єднані в розподілену мережу управління та легко використовуватися у всіх призначених для них операціях. Таким чином, вільно програмований контролер – це комп'ютер, який спеціально розроблений для вирішення завдань управління процесами та пристроями. При цьому будь-яке програмне забезпечення, яке відповідає стандарту IEC 61131, повинне забезпечувати низку можливостей для користувача. Тому воно складається з обов'язкових елементів:

1. *Засоби введення програм користувача (редактори).* Для створення та редагування програм однією з мов програмування.

2. *Засоби перевірки синтаксису*. Для перевірки правильності програми та вхідних даних для виключення помилок.

3. *Транслятор або компілятор*. Для перетворення вхідної програми в машинний код.

4. *Комунікатор.* Для завантаження програми користувача з ПК до пам'яті ПЛК для її тестування (або зворотно з ПЛК до ПК – вивантаження).

5. *Засіб тестування (налагодження).* Для підтримки користувача під час розроблення програм та при усуненні помилок в процесі налагодження, а також перевірки програми користувача шляхом:

- контролю стану входів та виходів, таймерів, лічильників тощо;
- перевірки послідовності покрокових операцій, команд тощо;
- моделювання роботи програми за допомогою примусового призначення сигналів на входах та виходах, призначення констант;

6. *Засіб відображення стану об'єктів систем управління:*

- надання інформації про стан устаткування, процесу управління та статус ПЛК;
- відображення стану вхідних і вихідних сигналів;
- відображення та запис змін зовнішніх сигналів і станів внутрішніх даних;
- контроль поточного часу виконання програм;
- виконання програми в режимі реального часу.

7. *Засіб документування.* Для складання опису контролера та програми користувача, що складається з наступного:

- опису конфігурації апаратної частини ПЛК;
- роздруківок програми користувача з відповідними даними і ідентифікаторами для сигналів, а також коментарями;
- довідника всіх даних: входів, виходів, таймерів, лічильників;
- опису модифікацій програми.

Далі в цьому посібнику на прикладі середовища *WinPLC*, що є основним середовищем для програмування контролерів *VIPA* різних сімейств та моделей, будуть розглянуті усі елементи, що перелічені вище. Відзначимо, що даний апаратно-програмний комплекс від компанії *VIPA* повністю відповідає вимогам стандарту *IEC 61131-3*.

1.2 Апаратне забезпечення засобів виробництва *VIPA*

Усі контролери виробництва компанії *VIPA* різних серій мають такі загальні властивості та призначені для побудови централізованих та розподілених систем управління, а саме:

- вбудовані оперативну (ОЗП) та постійну пам'ять (ПЗП) для збереження програми користувача та поточних даних;
- підтримують стандартні карти пам'яті (*MMC*) для збереження програм та даних;
- розширені комунікаційні можливості для створення промислових мереж розподілених систем управління різного рівня складності (протоколи *ModBus* та *Profibus*, послідовні інтерфейси *RS-232* та *RS-485*, а також *Ethernet*).

1.2.1 Опис контролерів VIPA System 100V

Контролер *VIPA System 100V* [7] – це серія мікро-ПЛК модульного типу, який зображений на рис.1.1. Завдяки дуже компактній конструкції та відмінному співвідношенню ціна/продуктивність серія *System 100V* є ідеальним рішенням для невеликих систем управління. До цій серії входять три моделі процесорних модулів різних модифікацій з різною кількістю каналів та різними комунікаційними можливостями. Це моделі типу – *Standard*, *PtP* та *DP-slave*. Усі моделі забезпечують підключення до своєї локальної паралельної шини до 4-х модулів введення/виведення (крім *CPU112* – ця модель процесору, яка не розширюється) за допомогою спеціального шинного з'єднувача на задній стороні модулів. У кожному процесорному модулі є MP^2I-інтерфейс для програмування та зв'язку з іншими пристроями. Так само в окремих моделях є інтерфейси *RS232* та/або *RS485*. Особливість серії *System 100V* – це можливість підключення до 4 сигнальних або функціональних модулів серії *System 200V*, тобто ці модулі сумісні з процесорами серії *System 100V*.

Основні технічні та експлуатаційні характеристики ПЛК, що входить до складу стенду: це процесорний модуль типу *CPU115SER* з 16 кБ ОЗП та 24 кБ ПЗП, з інтерфейсом *RS-485*, можливістю підключення за схемою «точка-точка» (*PtP*), має вбудовані дискретні входи DI 16(20)xDC 24V та 4 (або 2) лічильника з тактом рахування до 30 кГц та дискретні виходи DO 16(12)xDC 24V, 0.5A і два виходи ШІМ з тактовою частотою 50 кГц. В табл.1.1 наведені основні технічні характеристики процесорний модуль типу *CPU115SER*.

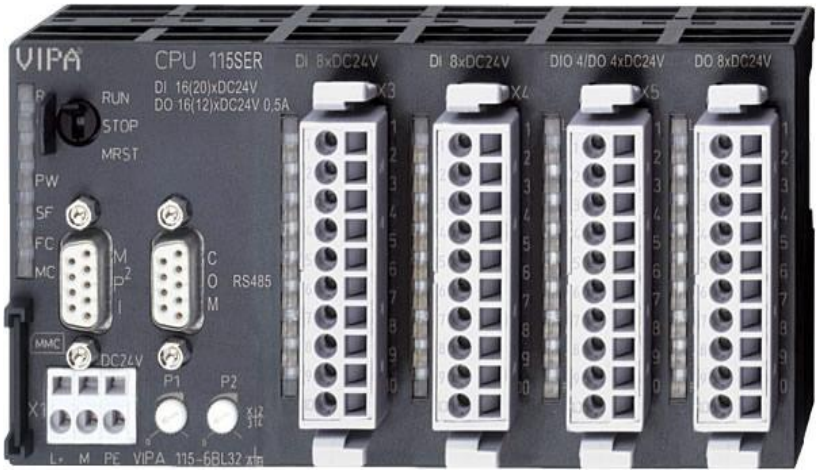


Рис. 1.1. Загальний вигляд ПЛК *VIPA System 115V*

Конструктивно ПЛК виконаний з моноблока, який складається: з субмодуля процесора та чотирьох субмодулей каналів введення/виведення. На лицевій панелі субмодулей ПЛК розміщені від'ємні клемні з'єднувачі X3...X6 з 10-ма контактами, які мають два контакти для живлення (+24 В та 0 В) та 8-м сигнальних контактів. Закріплення проводів в з'єднувачі здійснюється за допомогою механічних пружинних фіксаторів.

Таблиця 1.1 – Технічні характеристики ПЛК *VIPA* модель *CPU115SER*

Загальні властивості	– 16 (20) входів / 16 (12) виходів – 16 кБ ОЗП/24 кБ ПЗП/*MMC*-карта, до 512 кБ
Живлення	*DC* 24 В / струм 110 mA
Входи	
кількість	16 (20)
довжина лінії з'єднання	– 1000 м для екранованого кабелю – 600 м для кабелю без екрану
рівень вхідного сигналу	«0» – 0 … 5 В пост. струму «1» – 15 … 28.8 В пост. струму
час перемикання: «0»→«1» та «1»→«0»	3 мс
об'єм пам'яті вхідних даних	3 байти
Виходи	
кількість	16 (12)
довжина лінії з'єднання	– 1000 м для екранованого кабелю – 600 м для кабелю без екрану
вихідний струм (без навантаження)	50 мА
час перемикання: «0» → «1» «1» → «0»	 до 100 мкс до 350 мкс
об'єм пам'яті вихідних даних	3 байти
Час виконання операцій	
логічна операція з бітами	0,25 мкс
операція зі словами	1,2 мкс
Вбудовані лічильники (апаратні)	
кількість	4
розрядність	32

Продовження табл.1.1

максимальна частота відліку	30 кГц
Можливості щодо програмування	
кількість лічильників	256 – Z0…Z255
кількість таймерів	256 – T0…T255
кількість меркерів	8192 біт (2^{13}) – M0.0…1023.7
кількість блоків	OB – 14 (макс. 16 кБ) FB – до 1024 (макс. 16 кБ) FC – до 1024 (макс. 16 кБ) DB – до 2047 (макс. 16 кБ)
Розмір пам'яті області введення/виведення	
для входів/виходів	1024 байти / 1024 байти
для відображення вх./вих. процесу	128 байтів
Комунікаційні можливості	
тип інтерфейсу	*RS485*
тип з'єднувача	*Sub-D, 9-pin, female*
протоколи обміну	*1)MP²I (MPI/RS232)* *2)PtP (COM)–ASCII, STX/ETX, ModBus (M, S)*
швидкість обміну	від 150 біт/с до 115.2 біт/с
довжина лінії зв'язку	до 500 м
Експлуатаційні характеристики	
розміри	152,4мм x 76мм x 48мм
вага	302 г
робоча температура зовнішнього середовища	0 ÷ 60 ^{0}C

На рис. 1.2 зображена лицева панель процесорного субмодуля типу *CPU115SER* з поясненнями щодо органів керування та індикації ПЛК. На рис. 1.2 підписи мають таки значення:

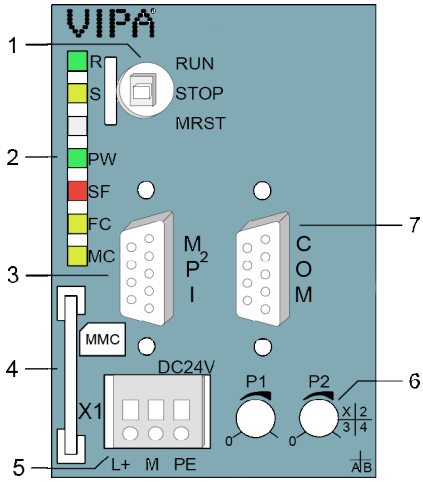


Рис. 1.2. Лицева панель *CPU115SER*

1 – перемикач режимів роботи ПЛК (RUN/STOP/MRST);

2 – світлодіоди стану ПЛК:

R – ПЛК у стані *RUN*, зеленого кольору;

S – ПЛК у стані *STOP*, жовтого кольору;

PW – живлення ввімкнено, зеленого кольору;

SF – групова помилка обладнання, червоного кольору;

FC – примусове присвоєння значень змінним

MC – наявність карти пам'яті у слоті, жовтого кольору;

3 – з'єднувач інтерфейсу *MP²I*.

4 – слот для підключення карти пам'яті *MMC*;

5 – з'єднувач для підключення живлення *DC24V*;

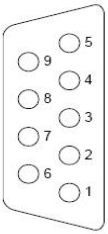
6 – аналогові потенціометри;

7 – з'єднувач інтерфейсу *RS485*.

Інтерфейс *MP²I* забезпечує обмін даними між ПЛК та ПК. З'єднувач *MP²I* (див. поз.3 на рис. 1.2) є комбінованим, тобто вміщує два інтерфейси:

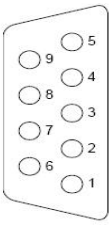
- *MPI (Multi-Point Interface)* для обміну даними між ПЛК та іншими пристроями послідовним інтерфейсом за схемою «мульті-точка»;
- *RS232 interface* для завантаження програм користувача до ПЛК за схемою «точка-точка» за допомогою так званого спеціального «зеленого» кабелю від *VIPA*.

Призначеня контактів з'єднувача в «зеленому» кабелі (типу *DUB-S, 9 pin*) для *MP²I* зображено на рис. 1.3., а для *RS485* – на рис.1.4.



Pin	Assignment
1	reserved (must not be connected)
2	M24V
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	P24V
8	RxD/TxD-N (Line A)
9	n.c.

Рис. 1.3. Призначення контактів для *MP²I*



Pin	RS485
1	n.c.
2	n.c.
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	n.c.
8	RxD/TxD-N (Line A)
9	n.c.

Рис. 1.4. Призначення контактів для *RS485*

Субмодуль вхідних каналів ПЛК серії *System* 100V збирає фізичні дані про технологічний процес та формує їхнє зображення в спеціально виділеній області пам'яті процесора. Субмодуль вихідних каналів формує керуючу дію на вихідні пристрої. Кожен вхідний і вихідний канал займає один біт інформації, а його одиничний стан відображається відповідним світлодіодом наявністю світіння зеленого кольору.

Субмодулі каналів введення/виведення ПЛК розподілені наступним чином: 1 – субмодуль типу DI 8xDC24V (X3), 2 – субмодуль типу DI 8xDC24V (X4), 3 – субмодуль типу DIO/DO 4/4xDC24V (X5), 4 – субмодуль типу DO 8xDC24V 0.5A (X6). Перші чотири входи на з'єднувачі X3 можуть використані як апаратні лічильники або як швид-

кі вході для аварійних сигналів, а останні два виходи з'єднувача X5 – як імпульсні виходи. Порядок роботи зазначених входів та виходів визначається при конфігуруванні спеціальних параметрів процесора. На рис. 1.5 показано розміщення входів для фізичних лічильників та виходів для ШІМ-управління.

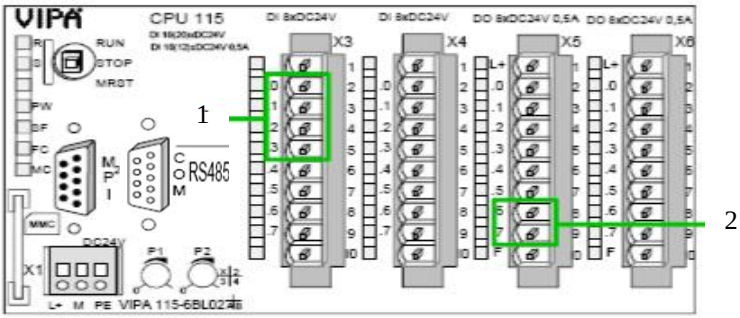


Рис. 1.5. Апаратні входи та виходи субмодулів ПЛК

Якщо не проведено конфігурування ресурсів ПЛК, то відображення фізичної області каналів введення/виведення на пам'ять процесора за умовчанням виглядає так, як це наведено в табл.1.2.

Схеми підключення датчиків до вхідних каналів та вихідних пристроїв до вихідних каналів ПЛК показано на рис. 1.6. Тут треба врахувати, що дискретні вхідні та вихідні сигнали можуть бути сформовані за допомогою зовнішнього джерела живлення ПЛК (підключені до загальної шини «М»). Крім того, в ПЛК на з'єднувачі X5 є чотири комбінованих канали, які можуть бути налаштовані або як вхід, або як вихід. Цім пояснюється змінна кількість входів та виходів в табл. 1.2 (DI 16(20)xDC 24V / DO 16(12)xDC 24V 0.5A).

На рис. 1.7 показані контактна група та світлодіоди субмодуля вхідних каналів з відповідними їм фізичними адресами. Контакти, помічені символом «*» відносяться до з'єднувача X3 та конфігуруються або як лічильники, або як приймачі сигналів «тривога».

Таблиця 1.2 – Адреси в пам’яті процесора для відображення фізичних каналів ПЛК *VIPA CPU115SER*

Область пам’яті процесора для відображення фізичних входів	
0...2 DI	адреси дискретних входів
3...127 DI	вільні адреси дискретних входів
128, 129 Potentiometer P1	адреса потенціометра №1 для формування аналогового сигналу
130, 131 Potentiometer P2	адреса потенціометра №2 для формування аналогового сигналу
132...135	адреси, що зарезервовані
136...139 Counter 0	адреса лічильника №1
140...143 Counter 1	адреса лічильника №2
144...147 Counter 2	адреса лічильника №3
148...151 Counter 3	адреса лічильника №4
152...1021	вільні адреси для аналогових входів
1022	резервна адреса
Область пам’яті процесора для відображення фізичних виходів	
0...2 DO	адреси дискретних виходів
3...127 DO	вільні адреси дискретних виходів
128…1021	вільні адреси для аналогових виходів
1022	резервна адреса

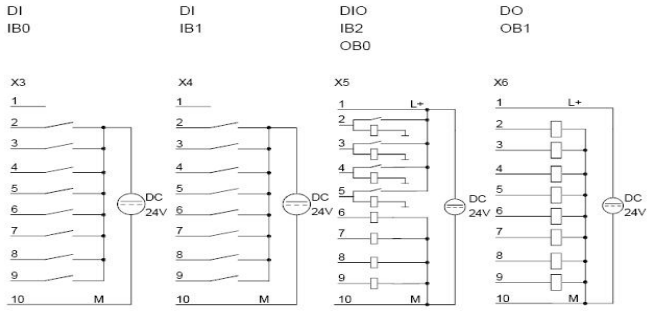
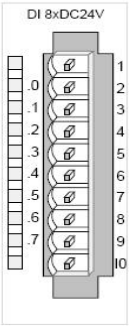


Рис. 1.6. Схеми підключення вхідних та вихідних сигналів до ПЛК



1	not used
2	Input I+0.0*
3	Input I+0.1*
4	Input I+0.2*
5	Input I+0.3*
6	Input I+0.4
7	Input I+0.5
8	Input I+0.6
9	Input I+0.7
10	Ground

Рис. 1.7. Фізичне адресування вхідних каналів

Електрична схема, яка пояснює порядок підключення фізичних вхідних каналів до субмодуля для одного каналу показана на рис. 1.8. Тут показані внутрішні вхідні ланцюги субмодуля, які складаються з оптопари для гальванічного розв'язування, світлодіоди для індикації наявності сигналу «1», опору навантажування для роботи світлодіода та діода обмеження для захисту вхідного каскаду від великої напруги. Граничне значення напруги, при якому ланцюг залишаються працездатними, становить 52 В. Зовнішні вхідні ланцюги складаються з джерела живлення постійного струму напругою 24 В та ключа. Як ключ може виступати будь-який тумблер або дискретний датчик. Усередині ПЛК вхідний каскад з'єднаний з логічними ланцюгами за допомогою внутрішньої, так званої *V*-шини, за допомогою котрої сигнал зчитується з оптопари.

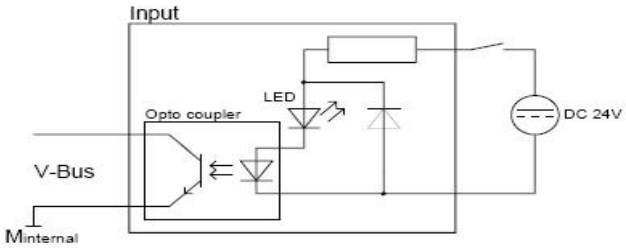


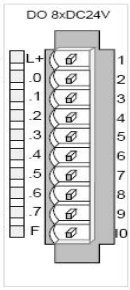
Рис. 1.8. Схема вхідного каскаду субмодуля ПЛК

Субмодуль виходів ПЛК серії *System 100V* також використовує зовнішнє джерело живлення постійного струму напругою 24 В. Жовтий колір світлодіода, позначеного символом «L+», показує наявність жив-

лення для вихідних каналів. Кожен вихідний канал займає 1 біт інформації і його активний стан відображається відповідним світлодіодом наявністю світла зеленого кольору. У разі перевантаження, перегріву або короткого замикання світлодіод, позначений символом «F», буде блимати червоним кольором. Кожен канал може бути навантажений максимальним споживаним струмом до 0,5 А.

На рис. 1.9 показані контактна група та світлодіоди субмодуля вихідних каналів з відповідними їм фізичними адресами. Контакти, які помічені символом « * » відносяться до з'єднувача X5 і конфігуруються як імпульсні виході для формування ШІМ-сигналу.

Електрична схема, яка пояснює порядок підключення фізичних вихідних каналів до субмодуля для одного каналу показана на рис. 1.10. Тут показані внутрішні вхідні ланцюги субмодуля, які складаються з оптопари для гальванічного розв'язання, світлодіоди для індикації наявності сигналу «1», опору навантаження для роботи світлодіода та підсилювача постійного струму для управління зовнішнім реле. Зовнішній вихідний ланцюг каналу складається з джерела живлення постійного струму напругою 24 В та керованого реле. Усередині ПЛК вихідний каскад з'єднаний з логічними ланцюгами за допомогою внутрішньої *V*-шини, за допомогою якої сигнал надходить до оптопари.



1	Supply voltage DC 24V
2	Output Q+0.0
3	Output Q+0.1
4	Output Q+0.2
5	Output Q+0.3
6	Output Q+0.4
7	Output Q+0.5
8	Output Q+0.6*
9	Output Q+0.7*
10	Supply voltage ground

Рис. 1.9. Фізичне адресування вихідних каналів

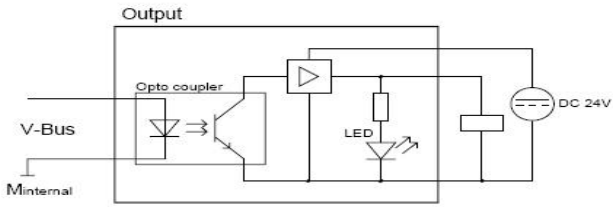


Рис. 1.10. Схема вихідного каскаду субмодуля ПЛК

Субмодуль входів/виходів ПЛК серії *System 100V* має чотири комбінованих каналу, тобто може бути використаний або як вхід, або як вихід. Кожен канал має функцію діагностики стану. Зелене світіння світлодіода, який позначений символом «L+» показує наявність живлення для каналів або наявність одиничного вхідного сигналу. У разі короткого замикання в навантаженні каналу вихід утримується у стані «0» і визначається помилка. Якщо в каналі перевантаження, перегрів або коротке замикання світлодіод, позначений символом «F», буде блимати червоним світлом. Кожен канал може бути навантажений максимальним споживаним струмом до 0,5 А.

На рис. 1.11 показані контактна група та світлодіоди субмодуля каналів входів/виходів з відповідними їм фізичними адресами. Контакти, які помічені символом «*» відносяться до з'єднувача X5, конфігуруються як імпульсні виходи для формування ШІМ-сигналу.

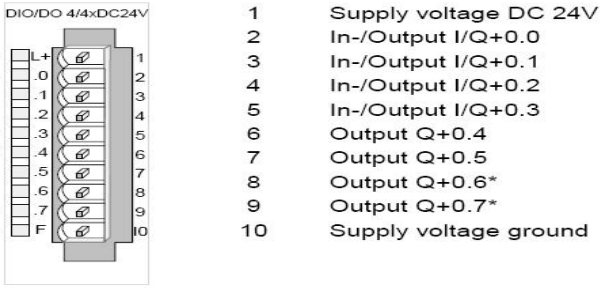


Рис. 1.11. Фізичне адресування вхідних/вихідних каналів

Електрична схема, яка пояснює порядок підключення фізичних вхідних та вихідних каналів є комбінацією двох схем, що зображені на рис. 1.8 і рис. 1.10, представлена на рис. 1.12. Призначення елементів схеми ідентично описаному вище для вхідних та вихідних каналів.

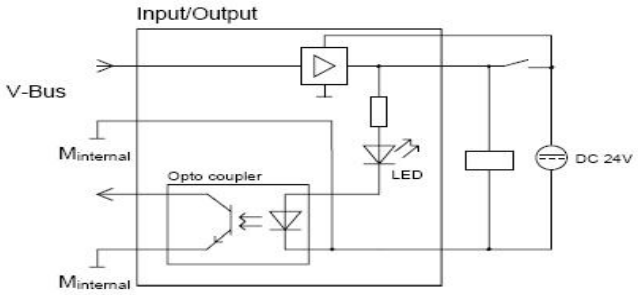
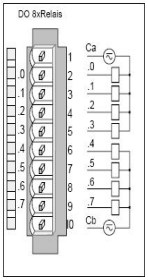


Рис. 1.12. Схема ланцюгу входів/виходів субмодуля ПЛК

Релейний вихідний субмодуль ПЛК розділений на дві групи по 4 канали. Тут немає індикаторів напруги навантаження та помилок. Контактна група та електрична схема релейних каналів ПЛК показані відповідно на рiс.1.13 і рис. 1.14. Однак, хоча в даному ПЛК такий субмодуль відсутній, все ж розглянемо порядок його роботи для повноти уявлення про можливості ПЛК серії *System 100V*.



1	Supply voltage Ca
2	Relay output Q+0.0
3	Relay output Q+0.1
4	Relay output Q+0.2
5	Relay output Q+0.3
6	Relay output Q+0.4
7	Relay output Q+0.5
8	Relay output Q+0.6
9	Relay output Q+0.7
10	Supply voltage Cb

Рис. 1.13. Фізичне адресування релейних вихідних каналів

На рис. 1.14 зображено, що керуючий сигнал посилюється в підсилювачі постійного струму, вмикає світлодіод за допомогою навантажувального опору та керує внутрішнім електромагнітним вихідним реле, яке в свою чергу комутує зовнішнє джерело змінної напруги 230 В або постійної напруги 30 В. Для захисту індуктивного навантаження у зовнішній ланцюг паралельно навантаженню застосовують *RC*-ланцюг, який усуває паразитні коливання.

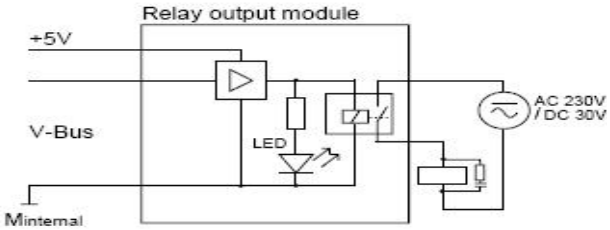


Рис. 1.14. Схема релейного вихідного каскада субмодуля ПЛК

1.2.2 Опис контролерів VIPA System 200V

Контролер *VIPA System 2xxV* [8] – це компактний модульний ПЛК теж модульного виконання, який зображений на рис.1.15. Завдяки широкому переліку модулів ця серія ідеально підходить для реалізації систем автоматизації малого та середнього рівня складності. Особливість серії *System 200V* – це наявність годинника реального часу.

Всі моделі ЦПУ забезпечують підключення до своєї локальної магістралі інших модулів за допомогою спеціальної об'єднавчої плати на кілька сигнальних модулів (на 2, 3, 5 або 9 модулів). Також можливе підключення до 32 модулів розширення в один ряд за допомогою комунікаційних модулів. Якщо використати зовнішні інтерфейси центрального процесора, то можливе підключення до 126 модулів за допомогою методу обміну «*master/slave*». Крім того ПЛК даної серії інтегруються

Рис. 1.15. Загальний вигляд ПЛК *VIPA System 214DP*

до мереж старшої серії більш потужних процесорних модулів *System300V* та *System300S*. Таки властивості серії *System200V* забезпечуються процесорними модулями з різним розміром ОЗП та ПЗП, широким переліком комунікаційних інтерфейсів. Це моделі процесорів *Standard, NET, PtP, DP-master, DP-slave и CAN-master*.

До складу навчального стенда входить центральний процесор моделі *CPU214DP* з інтерфейсом *MP²I* для завантаження програм користувача та комунікації з іншими ПЛК. Також ця модель обладнана вбудованим комунікаційним модулем для інтегрування в розподілену систему управління за допомогою шини *Profibus* зі статусом *DP-slave*. Цей модуль може забезпечувати обмін даними з процесорами 300-ої серії в режимі веденого пристрою.

Зовнішній вигляд та основні технічні характеристики процесорного модуля зображені на рис. 1.15 та відповідно в табл. 1.3.

На лицьовій панелі процесорного модуля розміщені два з'єднувача для інтерфейсів *MP²I* і *DP-slave* та клемний з'єднувач X1 з 2-ма контактами для підключення живлення (+24 В та 0 В). Фіксація проводів в з'єднувачі живлення здійснюється за допомогою механічних пружинних фіксаторів.

Таблиця 1.3 – Технічні характеристики процесора типа *CPU214DP*

Живлення	DC 24 В / струм 1.5 А
Пам'ять	48 кБ ОЗП / 80 кБ ПЗП, слот для *MMC*-карти, до 512 кБ
Час виконання операцій	
лог. операція з бітами	0,18 мкс
операція зі словами	0,72 мкс
Можливості програмування	
кіл. лічильників	256 – Z0…Z255
кіл. таймерів	256 – T0…T255
кіл. меркерів	8192 біт (2^{13}) – M0.0…1023.7
кіл. блоків	OB – 14 (макс. 16 кБ) FB – до 1024 (макс. 16 кБ) FC – до 1024 (макс. 16 кБ) DB – до 2047 (макс. 16 кБ)
Розмір пам'яті області введення/виведення	
для входів/виходів	1024 байти / 1024 байти
для відображення вх./вих. процесу	128 байт
Комунікаційні можливості	
тип інтерфейсу	*RS485*
тип з'єднувача	*Sub-D, 9-pin, female*
протоколи обміну	*1) MP²I (MPI/RS232)* *2) PROFIBUS DP-slave*
швидкість обміну	от 9,6 кбіт/с до 12 Мбіт/с
Експлуатаційні характеристики	
розміри	50.8мм x 76мм x 80мм
вага	150 г
роб. темп. зовн. серед.	0 ÷ 60 ^{0}C

На рис. 1.16 зображена лицева панель процесорного модуля *CPU214DP*. Враховуючи, що зовнішній вигляд процесора частково збі-

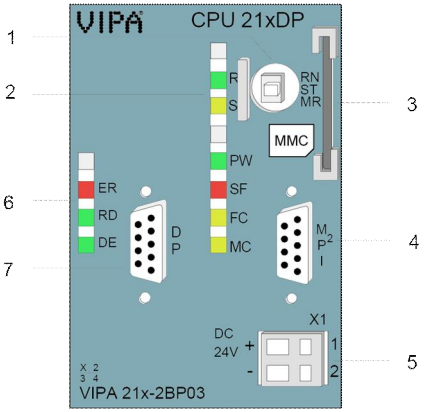


Рис. 1.16. Лицева панель *CPU214DP*

гається з процесором *CPU115SER*, тому надамо пояснення щодо відмінностей. Відмінності пов'язані з наявністю в модулі вбудованого інтерфейсу *PROFIBUS* зі статусом *DP-slave*. Це власне з'єднувач, який позначений символами «DP» та додаткові світлодіоди. Зважаючи на рис. 1.16 надамо пояснення щодо органів керування та індикації. Отже, пояснення щодо панелі індикації стану інтерфейсу *PROFIBUS*:

6 – світлодіоди стану комунікаційного модуля *DP-slave*:

ER – похибка процесору у стані *RUN*, червоного кольору:

а) повільна зміна стану (2 Гц) – помилка завантаження даних;

б) швидка зміна стану (10 Гц) – живлення менше 18 В;

в) зміна стану по черзі з RD – якщо конфігурація *DP-Master* невірна відносно *DP-slave*;

г) зміна стану одночасно з RD – якщо центральний процесор має невірну конфігурацію;

RD – статус обміну даними, зеленого кольору:

а) ввімкнений, якщо в режимі обміну даними *V*-шина швидше *PROFIBUS*;

б) вимкнений, якщо в режимі обміну даними *V*-шина повільніше *PROFIBUS*;

в) змінює стан, якщо тест обміну позитивний та завантаження даних пройшло без помилок;

г) зміна стану по черзі з ER, якщо конфігурація, яка передана від *DP-Master* невірна;

д) зміна стану одночасно з ER, якщо центральний процесор має невірну конфігурацію;

DE – активність шини *PROFIBUS*, зеленого кольору.

Якщо не проведено конфігурування ПЛК, то відображення області фізичних дискретних та аналогових каналів введення/виведення на область пам'яті ПЛК за замовчуванням відповідає табл. 1.4 та зображена на рис. 1.17.

Таблиця 1.4 – Області пам'яті ПЛК для відображення фізичних каналів введення/виведення

0...127	Периферійні адреси входів/виходів дискретних сигнальних модулів
128…1023	Периферійні адреси аналогових входів/виходів сигнальних модулів

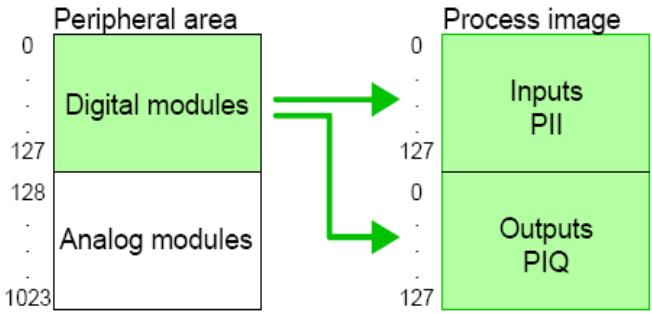


Рис. 1.17. Відображення фізичних адрес в пам'яті процесора

Для забезпечення виконання усіх функцій контролером на стенді процесорний модуль доповнений сигнальними модулями різних типів.

Рис. 1.18. Модуль *SM234*

Це, по-перше, дискретні модулі вхідних – *SM221* (зам. №221-1BF10), вихідних – *SM222* (зам. №222-1BF30) та вхідних/вихідних сигналів – *SM223* (зам. №223-1BF00) і, по-друге, аналоговий модуль вхідних/вихідних сигналів – *SM234* (зам. №234-1BD60). На стенді модулі розміщені в зазначеному порядку. Зазначимо, що дискретні модулі мають схемні рішення щодо оброблення вхідних сигналів та формування вихідних подібні до розглянутих раніше в п. 1.2.1 при описи контролерів 100-ї серії. Крім того, ці модулі сумісні з процесорами 100-ї серії. Відмінність полягає лише у принципах адресування каналів введення/виведення, які є периферійними та залежать від місцеположення модуля відносно центрального процесора (див. рис. 1.17 та табл.1.4). Причому адресування дискретних та аналогових модулів розділено. Тому розглянемо лише модуль аналогового введення/виведення *SM234*.

Модуль аналогового введення/виведення *SM234*, який зображений на рис.1.18, призначений для підключення датчиків з аналоговими сигналами різних типів та формування управляючих сигналів для аналогових виконавчих пристроїв. Як видно з рис.1.18 цей модуль має іншу клемну колодку (18-ть контактів) в порівнянні з модулями 100-ї серії. Це пов’язано з кількістю пристроїв, які підключаються до модуля. До модуля можливе підключення 4 аналогових датчика, причому 4-ий ка-

нал призначений для підключення резистивного сигналу або термометра опору різних типів (контакти 11 та 12 на рис.1.18). В модулі два вихідних аналогових канали, які є універсальними. Вони можуть формувати або сигнал постійного струму або напругу на зовнішньому навантаженні для аналогового регулювання. На модулі встановлено світлодіод для індикації наявності групової помилки червоного кольору (SF) будь-якого каналу або модуля в цілому.

Розглянемо більш докладніше схему підключення та властивості модуля *SM234*. На рис. 1.19 зображені дві схеми підключення датчиків: перша – для підключення датчиків з сигналами напруги, друга – для підключення сигналу постійного струму. Відмітимо, що схема підключення термометрів опору є двопровідною. Діапазон вхідних аналогових сигналів дуже великий. В табл.1.5 зведені відомості про основні властивості аналогових каналів модуля *SM234*.

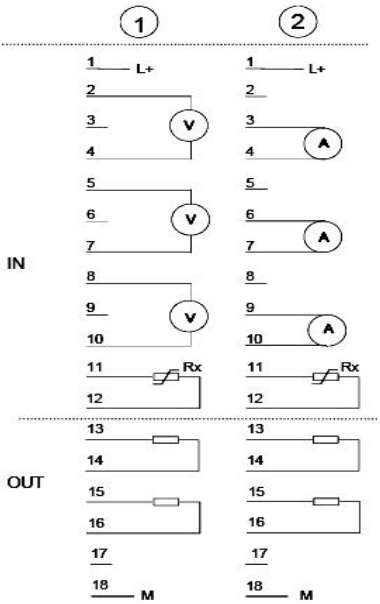


Рис. 1.19. Схема підключення до модуля *SM234*

В процесі вимірювань вхідні аналогові сигнали в модулі обробляються АЦП, а вихідні – ЦАП. Після перетворення в модулі формуються інформаційні байти кожного каналу, розміром в два байта, тобто типу *WORD*, які зберігаються у вбудованій пам'яті модуля. Всього зберіга-

ється шість слів (12 байт). Формат представлення даних наступний: значення виміряного параметра займає 15 молодших біт слова, а старший, 16-ий біт, показує знак параметра («0» – позитивне значення, «1» – негативне значення).

Таблиця 1.5 – Технічні характеристики каналів модуля *SM234*

Аналогові входи	Напруга: +1…+5 V, 0…10 V, -10…+10 V, -400…+400 mV, -4…+4 V Струм: -20…+20 mA, 0…20 mA,4…20 mA Сигнал опору: 0…600 Om, 0…3000 Om Термометр опору: Pt100, Pt1000, Ni100, Ni1000
Аналогові виходи	Напруга: +1…+5 V, 0…10 V, -10…+10 V Струм: -20…+20 mA, 0…20 mA, 4…20 mA
Розрізнення цифрового перетворення	АЦП: 16 біт ЦАП: 12 біт
Час перетворення	АЦП: 7…272 ms ЦАП: 1.5 ms/ch

Нижче показаний принцип перетворення виміряного значення вхідного сигналу в десяткове число і зворотно. Наприклад, для струмового уніфікованого сигналу 4-20 мА формули перерахунку виглядають так:

$$V = 27648 \cdot \frac{I-4}{16} \text{ та } I = V \cdot \frac{16}{27648} + 4,$$

де V – виміряне значення струму, яке представлене десятковим числом;
I – виміряне значення фізичного параметра (струм 4…20 мА).

Структурна схема модуля *SM234* зображена на рис.1.20. Виходячи зі схеми в модулі використане циклічне оброблення вхідних сигналів в одному АЦП, до якого подається аналоговий сигнал за допомогою мультиплексора, який комутує по черзі вхідні канали. Вихідні сигнали в модулі формуються окремо, для кожного каналу свій ЦАП.

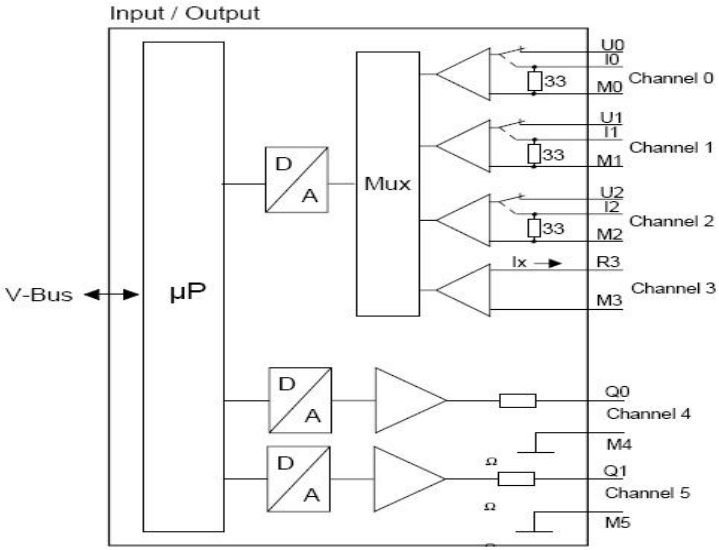


Рис. 1.20. Структурна схема внутрішніх ланцюгів модуля *SM234*

1.2.3 Опис контролерів VIPA серії System 300

Контролери *VIPA* 300-ої серії представлені двома серіями: *System 300V* [9] та *System 300S* [10]. Різниця полягає в типе мікропроцесора, який керує роботою модуля. В серії *System 300V* використаний звичайний тип мікропроцесора, який сумісний та подібний мікропроцесору від *Siemens*. Інша серія *System 3xxS* побудована на найбільш потужному та швидкому мікропроцесорі типу *SPEED7*, що виробляється компанією *VIPA*. Контролер серії *System 300S* має вигляд, який зображений на

рис.1.21, та призначений для вирішення завдань автоматизації середнього та великого рівня складності. Він також має конструктивну, функціональну та програмну сумісність з системою управління *SIMATIC S7-300* виробництва компанії *Siemens*.

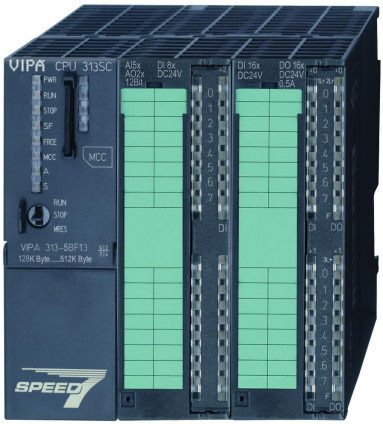


Рис. 1.21. Загальний вигляд ПЛК *VIPA System 3xxS*

Контролери *VIPA System 300S* виконані на базі технології *SPEED7* (власна розробка компанії *VIPA*) та програмуються крім середовища *WinPLC* також за допомогою середовища *STEP®7* компанії *Siemens*. Пам'ять для зберігання програм і даних вбудована в ЦПУ, завдяки чому необхідність в застосуванні додаткової карти пам'яті відсутня, на відміну від процесорів *SIMATIC S7-300*, де вона обов'язково потрібна. При необхідності об'єм робочої пам'яті ПЛК *VIPA System 300S* можна розширити за допомогою карти пам'яті *MMC*. Кожен процесорний модуль *VIPA System 300S* має порт *Ethernet* з підтримкою зв'язку з *PG/OP* (програматором). Крім того, цей модуль має вбудовані інтерфейси *MPI* та *PROFIBUS-DP*. Взагалі усі процесорні модулі розподілені за своїми функціональними можливостями на таки типи: *Standard*, *NET-CPUs*, *PROFINET*, та *class C*. В основному класифікація типів стосується комунікаційних можливостей процесорних модулів.

В комп'ютерному практикумі буде застосовано два стенда з ПЛК серії *VIPA System 300S*. Це ПЛК на базі процесорного модуля *CPU313SC/PtP-Speed7* (зам. № VIPA 313-5BF13), який зображений на рис.1.21 та ПЛК на базі *CPU314ST/DPM* (зам. № VIPA 314-6CF02).

Обидва процесорних модуля можуть виконувати функцію ПЛК в повному обсязі завдяки наявності у їхньому складі субмодулів для каналів дискретного та аналогового введення/виведення. Тому огляд процесорних модулів буде проведений на прикладі процесорного модуля *CPU313SC/PtP-Speed7*, який в подальшому буде мати скорочену назву – *CPU313SC*. Буде лише зазначено відмінності модуля *CPU314ST/DPM* (в подальшому *CPU314ST*) в порівнянні з розглянутим.

Спочатку зазначимо основні властивості ПЛК *CPU313SC*. У цього модуля два сигнальних субмодуля, які спільно з процесорним модулем об'єднані в одному корпусі. На борту субмодулів присутні:

- 24 дискретних вхідних канали;
- 16 дискретних вихідних канали;
- чотири аналогових вхідних канали для напруги та струму;
- один канал вимірювання опору або температури за допомогою термоперетворювача опору;
- два аналогових вихідних канали.

Усі дискретні та аналогові канали побудовані за схемами, що розглянуті раніше, у п.1.2.1 та п.1.2.2 (стосовно аналогових входів та виходів). Відмінність полягає у переліку доступних налаштувань типів сигналів при конфігуруванні в залежності від особливостей виконання вхідних та вихідних ланцюгів.

В процесорі є вбудована робоча пам'ять розміром 128 кБ, яка може бути розширена до 512 кБ за допомогою карти *MMC*. Пам'ять для завантаження теж вбудована та має об'єм 512 кБ. Модуль має вбудовані апаратні лічильники (три з частотою рахування до 30 кГц). На передній панелі модуля під кришкою встановлені з'єднувачі зажимного типу для живлення модуля (X1), типу *DB-9 (F)* для *MPI*-інтерфейсу (X2) та для комунікації з іншими пристроями (X3) та з'єднувач типу *RJ-45* для підключення до мережі Ethernet (X5). Модуль підтримує інтерфейс

«точка-точка» (*PtP*) для зв'язку з іншими пристроями з використанням низці протоколів, серед яких є *Modbus-Master*. При цьому використовується послідовний інтерфейс *RS-485*, який забезпечує зв'язок до 500 м на швидкостях від 150 біт до115,2 кбіт. Інши характеристики модуля зараз не суттєві. Тому зупинимося лише на схемах підключення до сигнальних субмодулів. На рис. 1.22 та 1.23 зображені відповідно схема комбінованого субмодуля з аналоговими та дискретними каналами (X11) і схема субмодуля з дискретними каналами введення/виведення (X12).

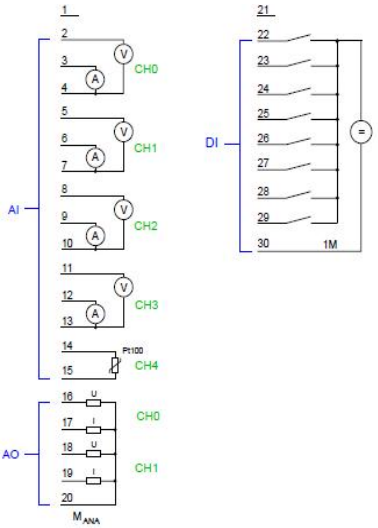


Рис. 1.22. Схема підключення комбінованого субмодуля

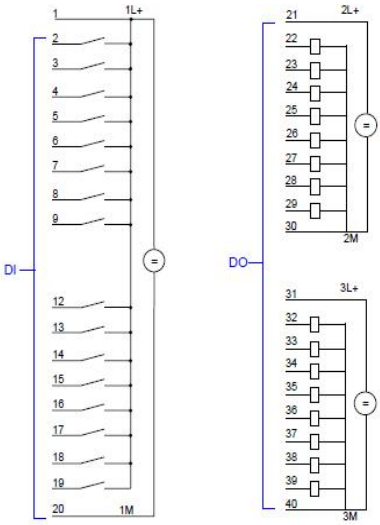


Рис. 1.23. Схема підключення дискретного субмодуля

Виходячи з аналізу схеми підключень каналів можна відмітити її деяку відмінність від раніше розглянутих. Так вхідні аналогові канали мають загальний контакт для напруги та струму окремо кожний. У вихідних аналогових каналів взагалі немає електричного розв'язування,

вони усі мають загальний контакт M_{ana}, який є нульовим для ЦАП. Дискретні вихідні канали мають групове розв'язання по вісім каналів в групі з загальним живленням від контактів M з відповідним номером. Вхідні дискретні канали підключені за схемою з загальним L+. Теж з груповим електричним розв'язанням на три групи. Лише резистивній вхід подібний раніше розглянутому при вивченні модуля *SM234* в п.1.2.2.

Принцип адресування усіх каналів в субмодулях буде розглянутий пізніше при ознайомленні з порядком конфігурування процесорних модулів 300-ої серіїї.

Розглянемо ПЛК на базі модуля *CPU314ST*. У процесора 2 Мб пам'яті для завантаження та 512 кБ робочої пам'яті, яка може бути розширена за допомогою *MMC* карти до 2МБ (50% програма/50% дані). Модуль має вбудовані апаратні лічильники (4 з частотою рахування до 100 кГц). На передній панелі модуля під кришкою встановлені з'єднувачі зажимного типу для живлення модуля (X1), типу *DB-9 (F)* для *MPI*-інтерфейсу (X2) та для комунікації з іншими пристроями (X3) та з'єднувач типу *RJ-45* для підключення до мережі Ethernet (X5). Модуль підтримує інтерфейси *PtP* або *PROFIBUS-DP* на з'єднувачі X3 відповідно до налаштувань при конфігуруванні процесорного модулю. Інтерфейс *PtP* забезпечує зв'язок з іншими пристроями по шині *RS-485* з використанням низці протоколів, серед яких є *Modbus-Master* на відстані до 500 м на швидкостях від 150 біт до115,2 кбіт. Робота в режимі *PROFIBUS-DP Master* надає можливість підключення до 32 ведених пристроїв на швидкостях від 9.6 кбіт/с до 12 Мбіт/с. При цьому розмір пам'яті для обміну складає по 244 байта на вхід та вихід. Модуль також може бути підключеним як ведений пристрій в режимі *PROFIBUS-DP Slave* з такими ж показниками з швидкості та пам'яті.

Відмітимо, що цей модуль має лише один субмодуль. Це визначає

наступний перелік сигнальних каналів:

- 8 дискретних вхідних канали;
- 8 дискретних вихідних канали;
- чотири аналогових вхідних канали для напруги та струму;
- один канал вимірювання опору або температури за допомогою термоперетворювача опору;
- два аналогових вихідних канали.

На закінчення розглянемо схему підключень до сигнального субмодуля ПЛК на базі модуля *CPU314ST*, яка зображена на рис.1.24.

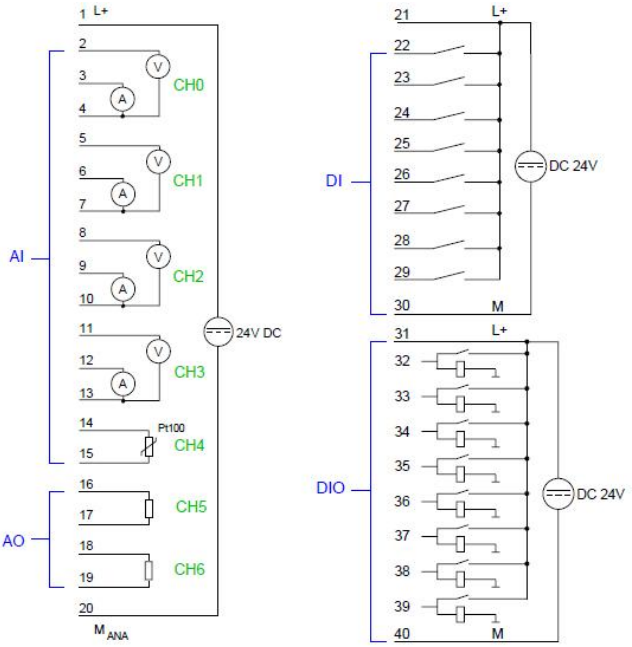


Рис. 1.24. Схема підключення до каналів субмодуля

З огляду на схему підключень, яка зображена на рис. 1.24 відмітимо деякі особливості. Аналогова частина схожа на ту що зображена

на рис.1.23, за виключенням виходів, які мають гальванічне розв'язання. Дискретна частина складається з двох груп, одна з котрих може конфігуруватися або як вхід, або як вихід. Таке схемне рішення вже розглядалося в 1.2.1 стосовно ПЛК VIPA115.

Додамо, що у складі стенду крім процесорного модуля є два окремих модуля аналогового введення та виведення, які з'єднані з основним модулям за допомогою спеціального внутрішньо шинного з'єднувача. Це сигнальні модулі аналогового введення *SM331* (зам. №331-7KB01) на два канали та виведення *SM332* (зам. №331-5HB01) на два канали.

Надамо коротку характеристику зазначених модулів. Отже, модуль аналогового введення *SM331* застосовується для підключення джерел струму та напруги, а також термоперетворювачів опору та термопар. Однак у нього є один недолік: його обидва канали може бути налаштованими одночасно лише на один тип сигналу. Види сигналів, що підключаються до цього модулю наведені в табл. 1.6.

Таблиця 1.6 – Види сигналів, що підключаються до модуля *SM331*

Напруга постійного струму	-80 mV ... +80 mV; -250 mV ... +250 mV; -500 mV ... +500 mV; -1 V ... +1 V; -2.5 V ... +2.5 V; -5 V ... +5 V; +1 V ... +5 V; -10 V ... +10 V
Постійний струм	-3.2 mA ... +3.2 mA; -10 mA ... +10 mA; -20 mA ... +20 mA; 0 mA ... +20 mA; +4 mA ... +20 mA
Опір	0 ... 150 Ohm, 0 ... 300 Ohm, 0 ... 600 Ohm
Термометр опору	Pt100, Ni100
Термопара	type J, type R, type K, type N, type L, type E, type T, type S, type B, type C

На передній панелі модуля *SM331* розташовані від'ємний з'єднувач та індикатори червоного кольору для відображення стану модуля:

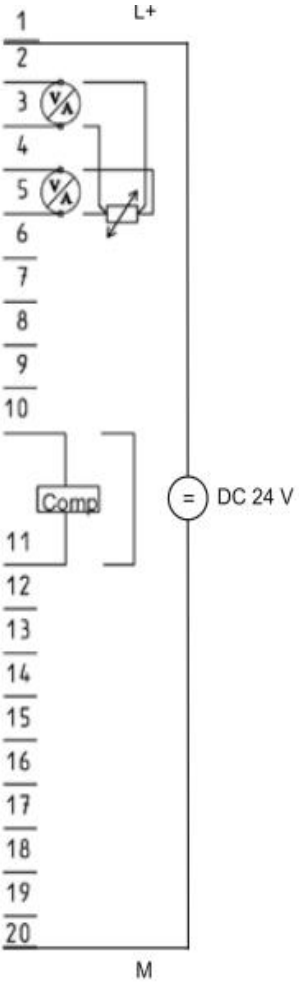


Рис. 1.25. Схема підключення до каналів модуля *SM331*

SF – групова похибка на випадок відмови зовнішнього живлення; якщо похибка, то індикатор постійно активований;

F0, F1 – похибка каналу вимірювання, яка активує індикатор при перебільшенні діапазону виміру; індикатор змінює стан з частотою 1 Гц одночасно з індикатором SF.

В модулі використаний АЦП на 12 біт. Крім того, в модулі реалізовані діагностика стану та програмні переривання. На рис. 1.25 зображена схема підключень модуля. З огляду на схему можна побачити, що можливе 4-х провідне підключення термометра опору, але у цьому випадку будуть задіяні усі контакти, тобто модуль стає одно канальним.

Модуль аналогового *SM332* виведення дозволяє формувати на двох каналах управляючі струмовий сигнал або сигнал напруги. Це стандартні струмові сигнали: -20…+20 мА, 0…20 мА, 4…20 мА; та сигнали напруги: -10…+10 V, 1…5 V, 0…+10 V. Канали в модулі налаштовуються індивідуально, тобто можуть бути різними. Можливе також апаратне відключення каналу під час конфігурування.

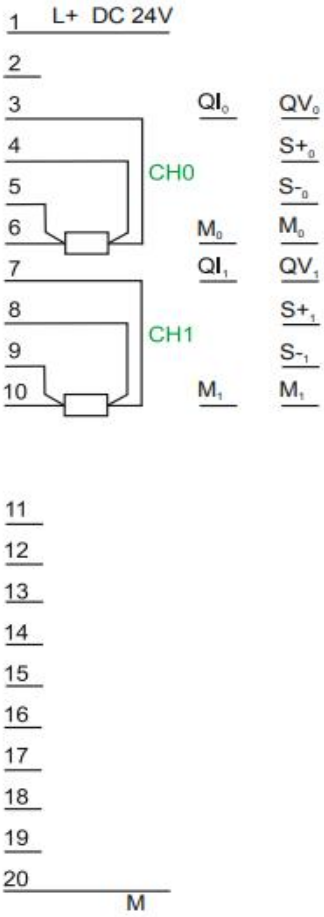


Рис. 1.26. Схема підключення до каналів модуля *SM332*

На передній панелі модуля *SM332* розташовані від'ємний з'єднувач та індикатори для відображення стану модуля:

SF – групова похибка на випадок відмови зовнішнього живлення, індикатор змінює стан з частотою 1Гц;

Q0, Q0 – якщо канал активований, індикатор зеленого кольору активний постійно;

F0, F1 – похибка каналу вимірювання, індикатор активний одночасно з індикатором SF.

В модулі використаний ЦАП на 12 біт. Крім того, в модулі реалізовані діагностика стану та програмне переривання.

На рис. 1.26 зображена схема підключень модуля. З огляду на схему можна побачити, що можливе 2-х або 4-х провідне підключення.

Насамкінець додамо, що до складу стендів також входять пристрої людино-машинного інтерфейсу (*HMI*). Це панелі оператора *VIPA OP03* та *VIPA LQE605-5.7"*, які зображені на рис.1.27 та 1.28. Зазначені пристрої – це універсальні текстовий дисплеї та графічна панелі оператора відповідно, які можуть бути використані спільно з системами автоматизації виробництва VIPA, а також інших виробників.

Основні властивості панелі *OP03*: 2 рядка по 20 символів, інтерфейс *MP_2I*, 256 кбайт пам'яті користувача, можливість підключення двох ПЛК. Графічна панель *VIPA LQE605-5.7"* має монохромний сенсорний екран *(*тип *LCD QVGA)*, до 6 Мбайт вбудованої пам'яті, вбудовані інтерфейси *RS-485*, *MPI*, *PROFIBUS DP*, *USB*.

Відмітимо, що для налаштування панелей *OP03* компанія *VIPA* надає спеціальне програмне забезпечення. Наприклад, для текстової панелі *OP03* – це програма-конфігуратор *OPM* (*Operator Panel Manager*), за допомого якої проводиться конфігурування ресурсів панелі, завантажується проект до панелі або налагоджується за допомогою програмного емулятора. Завантаження проектів до панелі та зв'язок з ПЛК здійснюється за допомогою «зеленого» кабелю з використанням інтерфейсу *MP_2I*.

Рис. 1.27. Загальний вигляд панелі *VIPA OP03*

Рис. 1.28. Загальний вигляд панелі *VIPA LQE605-5.7"*

Проте розгляд питання конфігурування панелі буде розглянуто в другому розділі. Це пов'язано з необхідністю узгодження налаштувань панелі з програмою користувача в контролері.

На завершення підрозділу додамо наступне. Для реалізації додаткових функцій пристроями *VIPA* компанія надає додаткове програмне забезпечення – пакет *VIPA Software Tools*. Це набір пов'язаних між со-

бою інструментальних засобів для програмування, конфігурування та обслуговування засобів автоматизації. В комп'ютерному практикумі в подальшому будуть теж застосовані наступні програмні продукті:

- *WinPcap* – програма для програмування ПЛК по *Ethernet*;
- *WinNCS* – конфігурування мереж *Ethernet* (*TCP/IP*) та *PROFIBUS DP*;
- *OP-Manager* – конфігурування панелей *OP03*;
- *OPC-Server* – обмін даними між SCADA-системами та ПЛК *VIPA* з використанням протоколів *MPI, TCP/IP* та *RFC1006*;

1.3 Програмне забезпечення засобів виробництва *VIPA*

Подальше знайомство з апаратно-програмними засобами виробництва компанії *VIPA* пов'язано з необхідністю встановлення СПЗ, тобто системи програмування контролерів *WinPLC7 V5*. Для цього необхідно завантажить з *Internet* інсталяційний пакет з офіційного сайту виробника за адресою http://www.vipa.com/en/service-support/downloads/software/, вибрати з переліку пакетів *WinPLC7 V5 Demo*. Далі встановіть цей пакет на ПК. Коли пакет буде встановлений запустить його. В результаті буде відкрито стартове вікно середовища. Хоча інтерфейс користувача середовища доступний лише на європейських мовах, він супроводжується інтуїтивно зрозумілими кнопками та піктограмами, які притаманні Windows-орієнтованим додаткам. Відзначимо, що усі проекти в середовищі позначені терміном *solution*. В стартовому вікні є спеціальне поле з можливістю активування *LE-Version*, яка орієнтована на роботу лише з ПЛК серії *System 100V*. Але це буде повнофункціональна версія. Якщо активувати кнопку обмежень версії можна побачити обмеження, які має *LE-Version*. Тому, якщо потрібно розробляти проекти для інших платформ необхідно залишити все так, як є. На стартовому вікні є також основне меню з переліком команд *File* та *Help*.

В основному полі вікна розміщені кнопки, які дозволяють зробити наступне:

1) створити проект;

2) створити проект з мінімально необхідним складом компонентів;

2) відкрити існуючий проект;

3) імпортувати проект, який створений в *STEP7* від *Siemens*.

Отже, скористайтесь другою кнопкою для створення проекту або виберіть команду основного меню з переліку *File*. З'явиться вікно з запитанням про створення проекту з ім'ям за умовчанням, на яке надайте позитивну відповідь. Далі відкриється вікно проекту з мінімальним набором компонентів та ресурсів для подальшого розроблення. Це вікно поділено на різні функціональні зони:

- основне меню зі списком команд;
- рядок з кнопками швидкого доступу до найбільш потрібних команд з основного меню;
- менеджер проекту з вкладеннями (*Edit project:*);
- робочий простір для редагування компонентів проекта зі вкладеннями (*Content*);
- інформаційне поле зі вкладеннями (*Tools…*);
- інформаційний рядок зі статусом ПЛК та каналу зв'язку з ним;
- кнопки виклику інструментів та додаткових компонентів (*Speedbar*).

В зоні *Edit project:* є чотири вкладення: *Solution*, *Catalog*, *CPU-Control Center* та *Call structure*. Вкладення *Solution* потрібно для керування ресурсами проекту, вкладення *Catalog* надає програмні ресурсі, вкладення - *Control Center* слугує для управління станом та режимами роботи ПЛК. Вкладення *Call structure* потрібне для редагування структур.

На даному етапі знайомства з середовищем *WinPLC7 V5* цього достатньо. Але поступово будемо розширювати знання про інтерфейс користувача та можливості середовища. При цьому, будь-які дії користу-

вача в середовищі виконуються ЛКМ або ПКМ та відображаються в інформаційному полі.

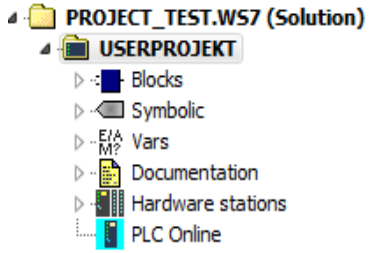


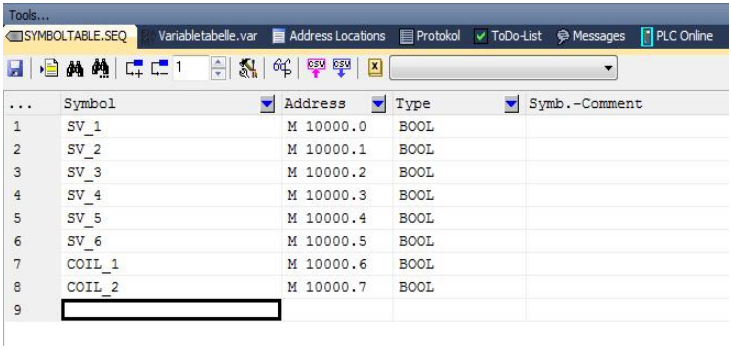
Рис. 1.29. Дерево проекту

Проект, який був створений, необхідно зберегти на носій. Для цього у переліку *File* з основного меню виберіть команду *Save solution as* . З'явиться стандартне вікно для збереження проекту у файл. Введіть в потрібне поле ім'я проекту, при необхідності змініть папку зберігання. В результаті з'явиться вікно з проектом, у якого нове ім'я. Далі можна діяти різними способами. Спочатку розробити програмну складову проекту, потім налаштувати апаратну, або навпаки. Для першого проекту без наявного ПЛК почніть з розроблення програмної складової. В подальшому можливе додавання апаратної складової без суттєвих перероблень проекту. До того ж, для налагодження програми користувача буде використаний програмний емулятор контролера. На рис. 1.29 зображено дерево новоствореного проекту *PROJECT_TEST.WS7 (Solution)*. В дереві проекту присутня папка, яка вважається субпроектом, у якого за умовчанням ім'я *USERPROJECT*. Цей субпроект складається з таких компонентів:

- *Blocks* – блоки, це власне елементи програми користувача;
- *Symbolic* – таблиця символьних імен змінних проекту;
- *Vars* – таблиця призначень значень змінним для налагодження програми в *Online*-режимі на емуляторі або в ПЛК;
- *Documentation* – інструмент документування проекту;
- *Hardware stations* – інструмент конфігурування апаратних ресурсів ПЛК;
- *PLC Online* – інструмент відображення стану емулятора або ПЛК в *Online*-режимі.

Взагалі, за допомогою контекстного меню можна добавляти, видаляти та змінювати імена компонентів дерева проекту. Склад контекстного меню залежить від міста, де його активувати.

Зараз список *Blocks* вміщує лише один блок. Цей блок має ім'я *OB1*. Також є таблиця символьних імен змінних проекту, але вона пуста. Якщо не створена конфігурація станції, тоді невідоме адресний простір змінних. Тому почнемо зі створення символьних імен в таблиці. Вона знаходиться внизу робочого простору субпроекту во вкладенні *SYMBOLTABLE.SEQ*. Отже за допомогою подвійного кліка відкрийте перший рядок та введіть символьне ім'я, наприклад, це буде *SV_1*. Система за умовчанням надасть змінній з даним ім'ям абсолютну адресу M10000.0, а тип змінної буде визначений за умовчанням *BOOL*. Далі за допомогою кнопки копіювання та інкрементування створить ще п'ять змінних. Усього буде створено шість змінних. У сьомому рядку введіть нове символьне ім'я, наприклад, це буде *COIL_1*. Подібно створить ще одну змінну. В результаті таблиця символьних імен буде виглядати так, як це зображено на рис. 1.30.



...	Symbol	Address	Type	Symb.-Comment
1	SV_1	M 10000.0	BOOL	
2	SV_2	M 10000.1	BOOL	
3	SV_3	M 10000.2	BOOL	
4	SV_4	M 10000.3	BOOL	
5	SV_5	M 10000.4	BOOL	
6	SV_6	M 10000.5	BOOL	
7	COIL_1	M 10000.6	BOOL	
8	COIL_2	M 10000.7	BOOL	
9				

Рис. 1.30. Таблиця символьних імен проекту

Далі необхідно заповнити блок *OB1*. Відкрийте його на редагування. Згорніть поле *Tools…* зі вкладенням *SYMBOLTABLE.SEQ* за допо-

могою спеціального символу згортання для збільшення робочого простору. Фрагмент поля з робочим простором зображень на рис. 1.31. Робочий простір поділений на таки зони:

- рядок з кнопками інструментів для роботи з блоком;
- таблиця з інтерфейсом блока;
- зона для введення програми з командами для редагування;
- рядок зі статусом;
- рядок з інформацією про змінну.

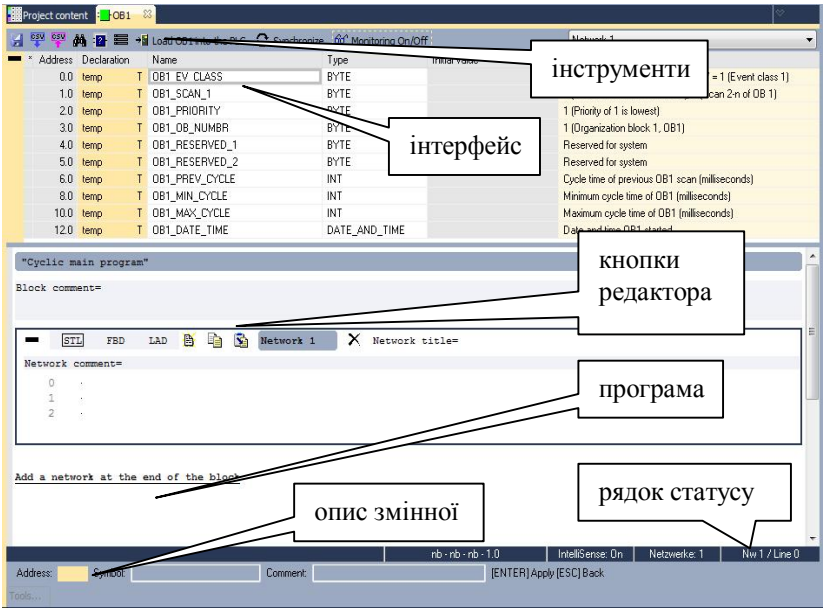
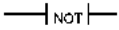


Рис. 1.31. Робочий простір з редактором блока

Зауважимо, що так виглядає робочий простір з редактором будь-якого блока з деякими відмінностями. Відмінності полягають в відсутності компонентів блоків, наприклад, блоків даних. За допомогою кнопки згортання «▬» можливо збільшити робочий простір. Розгортання

компонентів робочого простору здійснюється за допомогою кнопки «┼». За умовчанням редактором блока є мова *STL*, яка є текстовою і на початку ознайомлення досить важкою для засвоєння. Тому змінимо мову текстового редактора на мову *LAD*, яка є графічною та вважається прийнятною для створення програм користувача, які реалізують алгоритми автоматики та сигналізації, тобто оброблення змінних типу *BOOL*. Зауважимо що ця мова має назву «релейно-контактні схеми» (РКС). Зміна мови реалізації всередині блока здійснюється простим кликанням ЛКМ по відповідній кнопці мови. Причому локалізація зміни мови можлива на рівні так званого ланцюга – *Network*.

Перед розробленням програми користувача розглянемо базові принципи мови РКС. Дискретні логічні інструкції працюють з двома числами – «1» та «0». Ці дві цифри утворюють базис системи числення, так званої двійковій системою. Цифри «1» і «0» називаються двійковими цифрами (binary digits) або просто «бітами». При роботі зі схемами, що використовують контакти та обмотки, значення «1» означає активний стан або протікання струму, а «0» – неактивний стан або відсутність протікання струму. Бітові логічні інструкції інтерпретують стан сигналів «1» і «0» та комбінують їх за правилами булевої логіки. Ці комбінації дають результат «1» або «0», так званий «Результат Логічної Операції» (RLO). Нижче дана характеристика типів команд, як елементів ланцюгів на мові *LAD*.

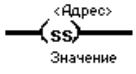
Команди як елементи ланцюга – подаються у вигляді окремих елементів, яким не потрібні ні адреси, ні параметри: 

.

Команди як елементи ланцюга з адресою – ці команди представляються як окремі елементи, для яких потрібно вводити адресу: 

.

Команди як елементи ланцюга з адресою і значенням – ці команди представлені як окремі елементи, для яких потрібно вводити адресу і

значення: 

.

Команди у вигляді блоків з формальними параметрами, які з'єднуються з лініями входів та виходів. Входи знаходяться з лівого боку блоку; виходи – з правого боку блоку. Вхідні параметри доступні для зчитування, а вихідні – для запису та зчитування. Для цих параметрів потрібно враховувати тип даних. Для блоків є можливість управління активністю блоків за допомогою спеціальних входів EN та виходів ENO. Якщо на вході EN присутній сигнал «1», блок виконує задану функцію і транслює цей сигнал на вихід ENO. Параметри блоку EN і ENO мають тип BOOL та можуть перебувати в області пам'яті I, Q, M, D або L. Параметри EN та ENO функціонують відповідно до таких принципів:

- якщо EN не активована (тобто стан сигналу дорівнює «0»), то блок не виконує свою функцію, і ENO НЕ активується (тобто стан сигналу теж дорівнює «0»);
- якщо EN активований (тобто стан сигналу одно «1») та відповідний блок виконує свою функцію без помилок, то ENO теж активується (тобто стан сигналу теж одно «1»);
- якщо EN активований (тобто стан сигналу одно «1») та при обробці функції виникає помилка, то ENO НЕ активується (тобто стан сигналу дорівнює «0»).

Вхідні елементи в мові *LAD*: ┤ ├ – це нормально-відкритий контакт (сканування з очікуванням сигнального стану «1») та ┤/├ – нормально-закритий (сканування з очікуванням сигнального стану «0»).

Вихідний елемент в мові *LAD*: **-( )-** – це обмотка, яка є термінатором (завершальним елементом) ланцюга і привласнює (*assigns*) *RLO* (результат логічної операції) безпосередньо операнду.

На рис. 1.32 показаний приклад з елементами ланцюгів. Контакти та обмотки є операндами булевого типу можуть бути входами, виходами, меркерами.

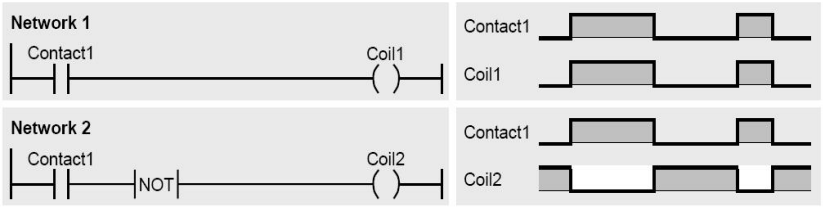
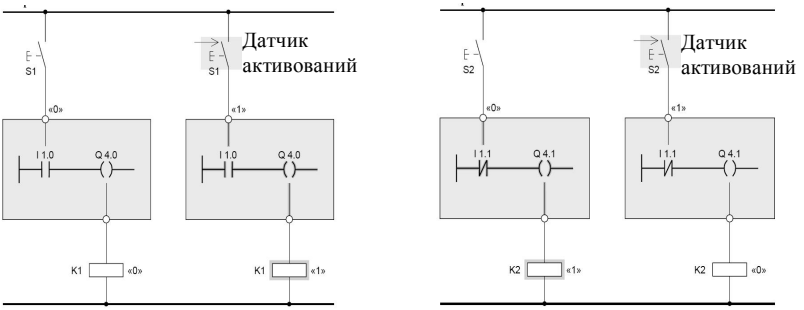


Рис. 1.32. Приклад використання елементів мови *LAD*

Розглянемо, як працюють контакти та обмотки в ланцюгах. На рис. 1.33 показані схеми з нормально-закритим (1) та нормально-відкритим контактами (2). Надамо деякі пояснення щодо рисунка. Якщо контакт типу NC, то програма користувача в ПЛК перевіряє його на наявність активного стану та активує обмотку (стан FALSE) у разі замкнення контактів датчика. Для контакту типу NO обмотка активована (стан TRUE), а програма очікує на активний стан датчика, який вимикає обмотку (стан FALSE).



1) 2)

Рис. 1.33. Принципи оброблення контактів
(1 – NO-контакт, 2 – NC-контакт)

Щодо паралельного з'єднання контактів, то це еквівалентно проведенню логічної операції «OR» над контактами. Це означає, що «струм» в логічному ланцюгу передається на обмотку у випадку за-

мкнення хоча б одного з контактів в паралельній лінії. Для випадку послідовного з'єднання контактів, то це подібно логічної операції «AND». Це означає, що «струм» в логічному ланцюгу передається на обмотку у випадку замкнення усіх контактів в послідовній лінії. На рис. 1.34 та 1.35 зображено приклади послідовного та паралельного з'єднань. В табл. 1.7 наведені «гарячі клавіші» в редакторі мови *LAD*.

Таблиця 1.7– Гарячі клавіші мови *LAD*

F2	Додати NO контакт
F3	Додати NC контакт
F7	Додати обмотку
F8	Відкрити паралельну лінію
F9	Закрити паралельну лінію
DEL	Видалити вибраний елемент
Alt+Back	Відмінити останню дію

Операція інверсії ┤**NOT**├ інвертує *RLO*. Можливо інвертувати контакт в ланцюгу. Але додавання *NOT*-контакту в паралельну лінію, що починається в середині ланцюга, не допускається.

Логічна операція «XOR» («виключаюче АБО») над контактами передбачає одночасну перевірку двох контактів в нормальному та інверсному стані в послідовних лініях, які з'єднані паралельно. Результат логічної операції «XOR» буде дорівнювати стану TRUE лише для випадку протилежного стану контактів.

У середовищі програмування, крім присвоювання *RLO*, доступні таки функції для обмоток:

- обмотки встановлення **-(S)** та скидання **-(R)** як одиничні програмовані операції з пам'яттю;
- блокові елементи **RS**- і **SR**-тригери як функції, що працюють з пам'яттю;

- конектори (midline outputs) **-(#)-** проміжні виходи (з'єднувачі);
- обмотки виявлення позитивного фронту імпульсу струму *RLO* **-(P)** та негативного фронту **-(N)**;
- блокові елементи **POS** та **NEG** як елементи оцінки фронту операндів.

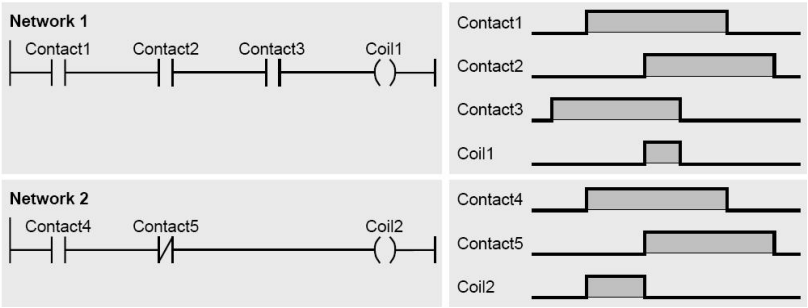


Рис. 1.34. Ланцюг з послідовним з'єднанням контактів
та діаграма стану обмоток

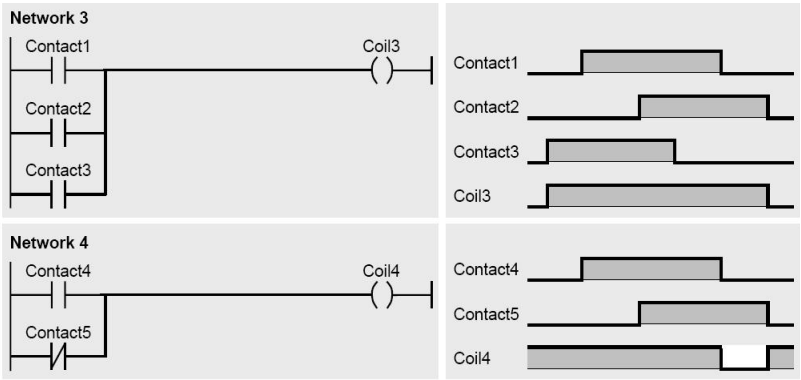


Рис. 1.35. Ланцюг з паралельним з'єднанням контактів
та діаграма стану обмоток

Якщо струм є в обмотці встановлення **-(S)**, то операнд над обмоткою переходить у сигнальний стан «1». Якщо струм є в обмотці ски-

дання **-(R)**, то операнд над обмоткою скидається в сигнальний стан «0». При відсутності струму в обмотках встановлення або скидання стан операнду залишається без змін. На рис. 1.36 показано як працюють обмотки встановлення та скидання.

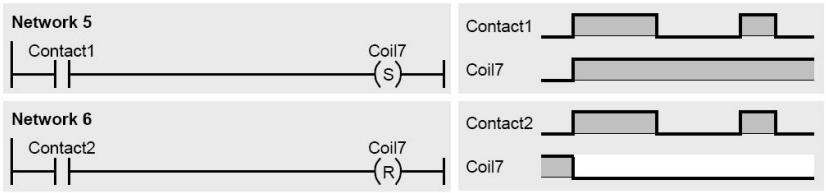


Рис. 1.36. Ланцюги з обмотками встановлення та скидання
та діаграма стану обмоток

Блоковий елемент з пам'яттю реалізує операції зі збереженням статусу (тригер). Загальний бінарний операнд розташовується над блоковим елементом. Вхід **S** (*set input*) блочного елемента в даному випадку відповідає обмотці встановлення, вхід **R** (*reset input*) – обмотці скидання. Сигнальний стан операнда знаходиться на виході **Q** блока. Є два варіанти тригера: у вигляді блочного елемента **SR** (пріоритет скидання) та **RS** (пріоритет встановлення). В блоці з пріоритетом скидання або встановлення операнд буде встановлений, якщо вхід встановлення має сигнальний стан «1», а вхід скидання має сигнальне стан «0». Відповідно, якщо на вході скидання знаходиться «1», а на вході установки – «0», то операнд буде скинутий, тобто отримає стан «0». Сигнальний стан «0» на обох входах не впливає на тригер. Якщо обидва входи тригера одночасно у стані «1», то блоки з пам'яттю реагують по-різному: **SR**-тригер скидається, а **RS**-тригер – встановлюється. В **SR**-тригері пріоритет має вхід скидання (*reset input*). Це означає, що тригер буде або скинутим або залишиться у стані «0», якщо струм тече «одночасно» на вході встановлення та вході скидання. Протилежний результат буде в **RS**-тригері, при одночасному сигналі «1» на його входах. Відмітимо,

тригер з пріоритетом скидання є «нормальною» формою функції для роботи з пам'яттю, так як стан скидання (сигнальний стан «0») зазвичай безпечніше або менш ризикований стан для технологічної установки. На рис. 1.37 та 1.38 відповідно зображені блочні елементи з пріоритетом скидання та встановлення.

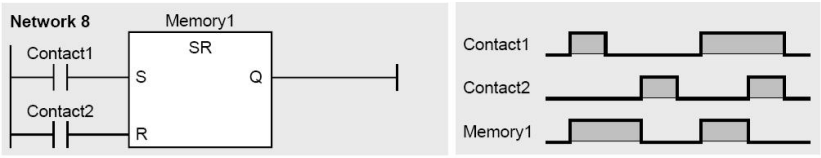


Рис. 1.37. Ланцюг з блоком з пріоритетом скидання
та діаграма стану операнду пам'яті

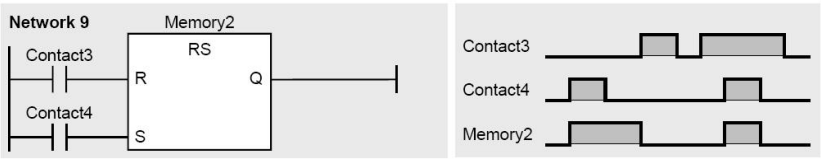


Рис. 1.38. Ланцюг з блоком з пріоритетом встановлення
та діаграма стану операнду пам'яті

Так як оператори в програмі користувача виконуються послідовно, процесор спочатку встановлює операнд пам'яті (*memory operand*), тому що вхід встановлення обробляється першим, але потім знову скидає його, коли обробляє вхід скидання. Поки решта програма обробляється, операнд пам'яті залишається у скинутому стані. Якщо операнд пам'яті є виходом, це встановлення має місце лише у вихідний таблиці образу процесу, а зовнішній вихід у відповідному модулі виходу залишається незмінним. Процесор CPU не передає вихідну таблицю образу процесу в модулі виходів до кінця програмного циклу.

Можна розміщувати тригери всередині ланцюга. Контакти можуть бути з'єднані послідовно та паралельно і при входах, і при виходах.

Другий вхід блочного елемента пам'яті допускається залишати вільним, тобто не підключеним. В одному ланцюзі можна з'єднувати кілька тригерів між собою. Тригери можна компонувати послідовно або паралельно. Також допустимо розміщення тригера після Т-гілки або у зве-ні, яке починається від лівої шини. На рис. 1.39 зображено приклад ланцюга з тригером всередині ланцюга та ланцюга з декількома тригерами.

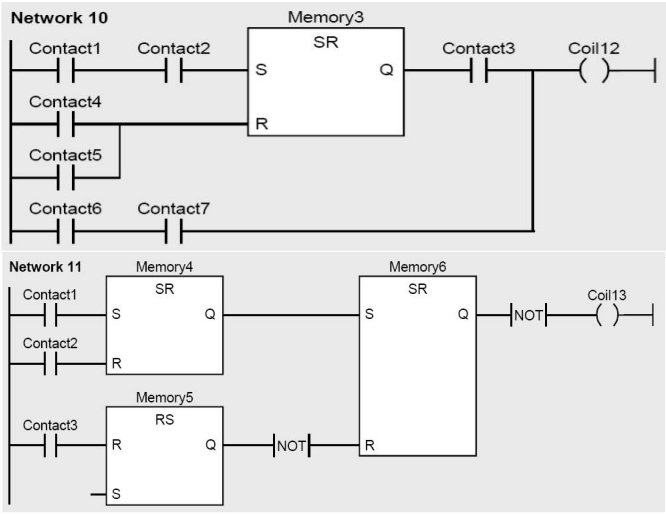


Рис. 1.39 Ланцюги з тригерами встановлення та скидання

Конектори (*midline outputs*) є проміжними елементами. *RLO*, дійсний для конектора, зберігається в операнде над цим конектором. Конектор є одиночною обмоткою в ланцюзі. Цей операнд може бути знову опитано в іншій точці програми, що дозволяє обробляти *RLO*, дійсний для конектора, в іншому місці програми. Сам конектор не впливає на проходження струму в ланцюзі.

Для проміжного зберігання двійкових результатів використовують бінарні операнди:

- біти тимчасових локальних даних, якщо проміжний результат

потрібен лише всередині блоку;

- біти статичних локальних даних доступні лише в межах функціонального блоку; вони зберігають сигнальний стан до їх повторного використання, навіть за межами цього блоку;
- меркери доступні глобально у фіксованій кількості відповідно до моделі процесора; для досягнення прозорості програмування необхідно уникати багаторазового використання одних і тих же меркерів для різних завдань;
- біти даних в глобальних блоках даних доступні з любого місця програми, але вимагають, щоб перед їх використанням відповідні блоки даних були відкриті.

Нижче на рис.1.40 показано, як проміжний результат зберігається в конекторі. *RLO* з ланцюга, що формується входами *Contact1*, *Contact*2 або *Contact*4 і *Contact*5, зберігається в конекторs *Midl_out1*. Якщо умова логічної операції виконується, та якщо сигнальний стан входу *Contact*3 у стані «1», то *Coil16* активується. Збережений стан *RLO* використовує-ся в наступному ланцюзі двома способами: з одного боку, проводиться перевірка виконання *RLO* спільно з *Contact*6, а з іншого боку, проводиться перевірка невиконання *RLO* спільно з *Contact*7.

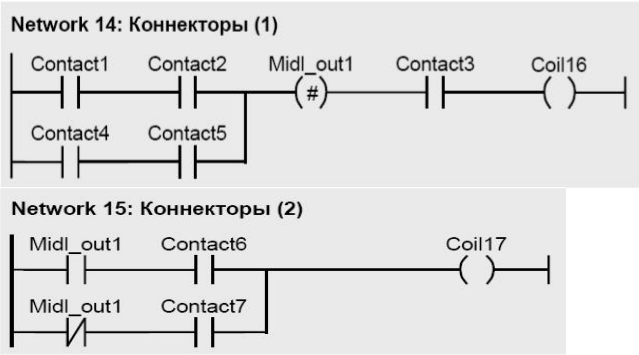


Рис. 1.40 Ланцюги з конекторами

За допомогою функції оцінювання фронту (*edge evaluation*) можна виявити зміну сигнального стану, тобто фронт сигналу. Фронт є позитивним, коли сигнал змінюється з «0» на «1». В протилежному випадку фронт негативний. На мові *LAD* еквівалентом елемента оцінки фронту є імпульсний контактний елемент (*Pulse Contact Element*). Якщо цей елемент генерує імпульс при включеному реле, це відповідає позитивному фронту. Імпульс при виключенні відповідає негативному фронту. Виявлення фронту сигналу (зміна в сигнальному стані) реалізується в програмі. Процесор порівнює поточний *RLO* (наприклад, результат перевірки стану входу) зі збереженим *RLO* на попередньому циклу сканування. Якщо сигнальні стани різні, то фронт сигналу присутній. Збережений *RLO* розташований в «меркері фронту» («*edge memory bit*») (він не обов'язково повинен бути меркером). Це повинен бути операнд, для якого доступний сигнальний стан, коли знову зустрічається оцінка фронту (в наступному програмному циклі), і який не використовується в будь-якому іншому місці програми. Як операнди в програмі можуть використовуватись меркери, біти даних в глобальних блоках даних і біти статичних локальних даних в функціональних блоках.

Меркер фронту збереже «старий» *RLO*, з яким процесор здійснив останню обробку оцінки фронту. Якщо фронт сигналу в даний момент присутній, тобто якщо поточний *RLO* відрізняється від сигнального стану меркера фронту, то процесор коригує сигнальний стан меркера фронту шляхом привласнення йому «нового значення» *RLO*. При наступній обробці оцінки фронту (зазвичай в наступному програмному циклі) сигнальний стан меркера фронту той же, що і у поточного *RLO* (якщо він тим часом не змінився), і процесор більше здійснює виявлення фронту.

Виявлений фронт відображається *RLO* після оцінки фронту. Якщо процесор виявляє фронт сигналу, то він встановлює *RLO* в «1» після оцінки фронту (тобто в ланцюзі є струм). Якщо фронт сигналу НЕ ви-

явлений, то *RLO* дорівнює «0». Отже, сигнальний стан «1» після оцінки фронту означає «фронт виявлений» («*edge detected*»). Присутність сигнального стану «1» на короткий час, як правило, протягом лише одного програмного циклу. Якщо процесор не виявляє фронт в наступному циклі (якщо «вхідний *RLO*» оцінки фронту не змінюється), він встановлює *RLO* назад в «0» після оцінки фронту.

Можна обробляти *RLO* відразу після оцінки фронту, тобто, наприклад, зберігати його за допомогою обмотки встановлення, або комбінувати його з використанням розташованих далі контактів, або зберігати його в бінарному операнді, так званому «імпульсному меркері» («*pulse memory bit*»). Імпульсний меркер застосовується, коли *RLO* з оцінки фронту повинен використовуватися в іншому місці програми; це, так би мовити, проміжний буфер для виявленого фронту (імпульсний контактний елемент в діаграмі схеми). Операндами, що застосовані як імпульсний меркер, є меркери, біти даних в глобальних блоках даних і біти тимчасових і статичних локальних даних.

Оцінка фронту струму (виділення фронту *RLO*) відображається обмоткою, яка всередині містить **P** (для позитивного фронту) або **N** (для негативного фронту). Над обмоткою розташовується меркер фронту, бінарний операнд, в якому зберігається «старий» *RLO* з попередньої оцінки фронту. Оцінка фронту виявляє зміну в ланцюзі: зі «струм в ланцюзі є» на «струм в ланцюзі відсутній» та навпаки.

Приклад на рис. 1.41 відображає оцінку позитивного та негативного фронту. Якщо до паралельного ланцюгу, що складається з *Contact1* і *Contact3*, застосовується оцінка позитивного фронту, то вона генерує короткий імпульс за допомогою *EMemBit1*. Якщо *Contact2* в цей момент замкнутий, то блок тригера *Memory7* встановлюється. Блок тригера *Memory7* скидається знову імпульсом від *EMemBit2*, якщо в послідовному ланцюгу, який складається з *Contact4* та *Contact5*, застосовується оцінка негативного фронту, яка виникає при перериванні електричного струму. Оцінка фронту не

може бути розміщена безпосереднє на лівій шині.

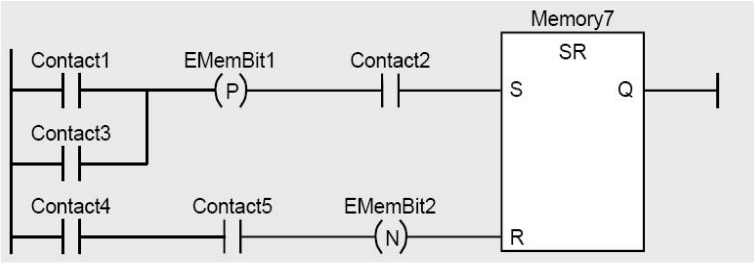


Рис. 1.41 Ланцюги з обмотками оцінювання фронтів

На мові *LAD* можлива оцінку фронту операнда з використанням блочного елемента: **NEG** або **POS**. Над блоком розміщується операнд, для якого оцінюється зміна сигнального. Меркер фронту, який зберігає «старий» сигнальний стан з попереднього програмного циклу, розташований біля входу *M_BIT*. Елемент оцінки фронту має немаркований вхід та вихід *Q* і в такому вигляді «вставляється» у ланцюг замість контакту.

На рис. 1.42 зображено приклад з блоками оцінювання позитивного та негативного фронтів в операнді. Якщо струм тече в немаркованому вході, вихід *Q* при наявності фронту генерує імпульс; якщо в даному вході немає струму, вихід *Q* залишається неактивним. Можна вставити оцінку фронту на місце будь-якого контакту, навіть в паралельну гілку, яка починається на лівій шині. Оцінка фронту в верхній лінії генерує імпульс, якщо операнд *Contact1* змінює свой сигнальний стан з «0» на «1». Цей імпульс встановлює тригер *Memory0*. Оцінка фронту завжди активується прямим з'єднанням немаркованого входу з лівої шиною. Оцінка фронту струму активується контактом *Contact2*. Якщо оцінка фронту активується подачею «1» на немаркований вхід, то вона генерує

імпульс при зміні сигнального стану бінарного операнда *Contact3* з «1» на «0».

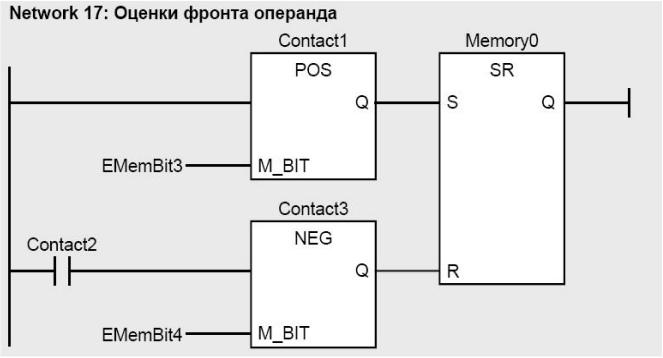


Рис. 1.42 Ланцюги з блоками оцінювання фронтів

На цьому огляд логічних операцій над бітовими операндами закінчено. Розглянемо інші оператори з переліку *Catalog*, а саме оператори порівняння, оператори перетворення типів, оператори для дій над цілими числами та числами з плаваючою крапкою, оператори зсуву та обертання, оператори логічного оброблення слів. Таймери та лічильники розглянемо у другому розділі. Зауважимо, якщо буде вибрана мова функціональних блоків (*FBD*), то усі логічні операції в програмі подаються у вигляді блоків, а змінні ліворуч є вхідними (доступна функція читання), а праворуч – вихідними (доступні функції читання та запису).

У вкладенні *Catalog* у переліку *Word logic* логічного оброблення слів у короткому (16 битів) та довгому (32 бита) форматах. Це побітові операції логічного додавання (*OR*), множення (*AND*) та операція *XOR*. Результат логічної операції буде мати той же тип, що й вхідні операнди. Особливістю оброблення за логічною функцією *XOR* є те, що аналізуються біти слів по-розрядам, а потім виробляється загальне слово. Як-

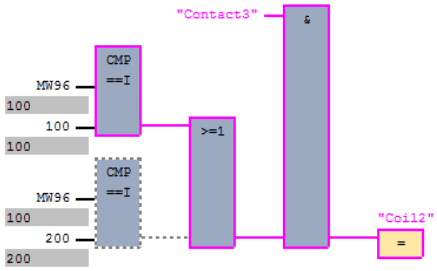


Рис. 1.43 Ланцюг з блоками порівняння операндів типу INT

що біти відрізняються, то відповідний біт результату буде у стані TRUE.

Для явного присвоєння значень операндам застосовується оператор MOVE, який може бути застосованим до операндів будь якого типу. Операнди можуть бути розміром 8, 16 або 32 біти.

Оператори порівняння працюють з операндами трьох типів. Це типи цілочислений розміром 16 битів (INT) і 32 біта (DINT) та число з плаваючою крапкою одинарної точності (REAL) у форматі *IEEE754* [11]. Функція порівняння можлива лише з однотипними операндами. Результат порівняння може бути присвоєний змінним булевого типу, наприклад, дискретному виходу, меркеру, тощо. На рис. 1.43 зображено ланцюг з блоками порівняння цілочислених операндів з формуванням умов вмикання обмотки “Coil2”. Якщо контакт “Contact3” у стані TRUE та операнд MW96 дорівнює або значенню «100», або – значенню «200», то обмотка “Coil2” буде у стані TRUE.

Оператори для арифметичних та інших дій також розділені на групи в залежності від типів операндів, що обробляються. Так для цілочислених типів обох розмірів (16 та 32 біти) є оператори додавання, віднімання, множення та ділення. Додатково для операндів типу *DWORD* є оператор MOD, який обчислює залишок від ділення. Усі зазначені оператори знаходяться у вкладенні *Catalog* у переліку *Integer fct.*. Оператори для оброблення операндів у форматі *IEEE754* [6] знаходяться в переліку *Floating point fct.*. До цього переліку входять операто-

ри, які реалізують додатково функцію отримання модулю числа (ABS), отримання квадратного кореня (SQRT) та квадрату числа (SQR), а також логарифмічні та тригонометричні (зворотні) функції. Звичайно тригонометричні функції обробляють дані у форматі значень кутів у радіанах.

Для перетворення типів даних у вкладенні *Catalog* у переліку *Converter* є оператори перетворення типів: отримання першого та другого доповненнь та перетворення числа у *BCD*-форматі у цілочислений тип та зворотно (для обох розмірів цілих чисел), перетворення подвійного цілого у число у форматі *IEEE754* та зворотно. Перше доповнення (INVI) для чисел розміром 16 битів є результатом побітової операції *XOR* з числом-шаблоном відповідного розміру, а саме з числом $FFFF_h$ (для числа розміром 16 битів). Результат операції – це обернення на протилежні біти вхідного числа. Друге доповнення (NEGI) – це зміна знака числа розміром 16 битів. Такий же результат буде по відношенню до числа у форматі *IEEE754*, якщо застосувати функцію NEGR. У переліку *Converter* є оператор отримання цілого значення (TRUNK) від числа у форматі *IEEE754* (тобто відкидання дробової частини числа). Також є можливість звичайного округлення (RND) чисел у форматі *IEEE754*, або округлення до більшого (CEIL) або меншого (FLOOR) цілого числа.

На завершення розглянемо оператори побітового обертання та зсуву. Ці оператори знаходяться у вкладенні *Catalog* у переліку *Integer fct.*. Оператори також поділені за принципом типів даних, що обробляються. Це оператори оброблення цілих чисел, слів та подвійних слів (тобто без знакових чисел). Обертання можливе лише для подвійних слів.

Інші групи операторів відносяться до мови *STL*. Це оператори оці-

нювання статусу слова (*Status bits*), керування програмою користувача (*Program control*) і оператори виклику блоків даних та переходів і міток (*Jumps*) у програмі користувача.

В подальшому в програмах будуть використані різноманітні операнди. Тому для спрощення роботи з ними наведемо табл. 1.8 з різними простими типами даних, які можуть бути співвіднесені з операндами в програмах користувача. Зауважимо, типи даних насамперед визначають розмір пам'яті для зберігання значення. Відповідно, виходячи з розміру пам'яті визначається діапазон значень. Також тип даних визначає перелік допустимих дій над відповідним операндом.

Таблиця 1.8 – Прості типи даних та їхня характеристика

Тип даних та опис	Розмір	Формат зображення	Діапазон значень	Приклад
BOOL (біт)	1	Булевий текст	TRUE/FALSE	TRUE
BYTE (байт)	8	*HEX*-число	B#16#00…B#16#FF	B#16#10
WORD (слово)	16	Двійкове число	2#0…2#1111_1111_1111_1111	2#1001_0111_1100_1101
		HEX-число	W#16#0… W#16#FFFF	W#16#1A0B
		BCD-число	C#0…C#999	C#975
		Без знакове десяткове число	B#(0,0)… B#(255,255)	B#(10,20)
DWORD (подвійне слово)	32	Двійкове число	2#0…2#1111_1111_1111_1111_1111_1111_1111_1111	2#1001_0111_1100_1101_1100_0001_1100_0011
		HEX-число	W#16#0000_0000… W#16#FFFF_FFFF	DW#16#1A0B_1234
		Без знакове десяткове число	B#(0,0,0,0)… B#(255,255,255,255)	B#(10,14,100,255)
INT (ціле)	16	Десяткове зі	-32768…32767	121

		знаком		
DINT (подвійне ціле)	32	Десяткове зі знаком	-2147483648… 2147483647	L#121
REAL (число у форматі *IEEE754*)	32	Число з плав. крапкою	+/-3.402823e+38 +/-1.175494e-38	1.2145e+2

Продовження таблиці 1.8

S5TIME (час у форматі *Siemens*)	16	Інтервал часу з тактом 10мс	S5T#0H_0M_0S_10MS… S5T#2H_46M_30S_00MS	S5T#6M_30S
TIME (час у *IEC*-форматі)	32	Інтервал часу з тактом 1мс, ціле зі знаком	T#24D_20H_31M_23S _648MS T#24D_20H_31M_23S _647MS	T#1M360MS
DATE	16	Інтервал часу з тактом 1 доба	D#1990-1-1… D#2168-12-31	D#2018-4-30
TIME_OF_DAY	32	Час доби, з тактом 1мс	TOD#0:0:0.0 … TOD#24:59:59.999	TOD#10:2:5.3
CHAR	8	Символи код. таблиці *ASCII*	‘A’, ‘B’, ‘C’, усього 128 символів	‘A’

Складені типи даних визначають групи даних, що мають розмір більш ніж 32 біт, або групи даних, складені з різних типів даних. Вони визначають структури і масиви або в описі змінних кодового блоку, або в блоці даних. У середовищі *WinPLC V5* можливі такі складенні типи даних:

- DATE_AND_TIME (дата та час);
- STRING (рядок);
- ARRAY (масив даних);
- STRUCT (структура);
- FB та SFB (дані екземплярів функціональних блоків).

В табл. 1.9 наведені позначення та опис складених типів даних.

В табл. 1.10 наведені відомості щодо адресування операндів в різних зонах пам'яті ПЛК. Це таки зони пам'яті:

- пам'ять введення/виведення (I/O);
- меркерна пам'ять (M);
- пам'ять периферійного введення/виведення (Peripheral I/O);
- пам'ять лічильників та таймерів (C, T);
- пам'ять кодових блоків та блоків даних (OB, FB, FC, SFB, SFC та DB);
- пам'ять типів даних, які визначені користувачем (UDT).

Таблиця 1.9 – Складені типи даних та їхня характеристика

Тип	Опис
DATE_AND_TIME	Розмір – 64 біти. Від зберігання – двійково-десятковий: байт 0 – рік; байт 1 – місяць; байт 2 – день; байт 3 – година; байт 4 – хвилини; байт 5 – секунди; байт 6 та ½ байта 7 – мілісекунди ½ байта 7 – день тижня
STRING	Визначає заголовок з двох байт та 254 символи (тип CHAR)
ARRAY	Визначає групу даних одного типу. Індекс елемента масиву – тип INT
STRUKT	Визначає групу даних різних типів. Можливо об'явить масив зі структур або структуру зі структур та масивів
FB та SFB	Визначає структуру екземплярів блоків даних для декількох функціональних блоків в одному блоці даних

На початку циклу сканування операційна система опитує входи сигнальних модулів та запам'ятовує їхнє значення у відповідній області пам'яті. Далі програма користувача може використовувати ці значення при циклічної обробці. Наприкінці циклу програма користувача розраховані значення переносить з пам'яті виходів та формує значення виходів сигнальних модулів.

Таблиця 1.10 – Області пам'яті за призначенням операндів та адресний простір

Адреса у форматі IEC	Опис	Тип даних	Діапазон
Область пам'яті входів/виходів та маркерної пам'яті			
I/Q/M	біт	BOOL	0.0…65535.7
IB/QB/MB	байт	BYTE, CHAR	0…65535
IW/QW/MW	слово	WORD, INT, S5TIME, DATE	0…65534
D/QD/MD	подвійне слово	DWORD, DINT, REAL, TOD, TIME	0…65532
Область периферійної пам'яті			
PIB / PQB	байт	BYTE, CHAR	0…65535
PIW / PQW	слово	WORD, INT, S5TIME, DATE	0…65534
PIB / PQB	подвійне слово	DWORD, DINT, REAL, TOD, TIME	0…65532
Лічильники та таймери			
C / T	слово лічильника / слово таймера	COUNTER TIMER	0…65535 0…65535
Кодові блоки та блоки даних			
OB	Організаційний блок	OB	1…65535

FB, FC, SFB, SFC	Функціональні блоки та функції, системні ФБ та Ф	FB, FC, SFB, SFC	0…65535
DB	Блоки даних	DB	1…65535
UDT	Типи даних користувача	UDT	0…65535

На початку циклу сканування операційна система опитує входи сигнальних модулів та запам'ятовує їхнє значення у відповідній області пам'яті. Далі програма користувача може використовувати ці значення при циклічної обробці. Наприкінці циклу програма користувача розраховані значення переносить з пам'яті виходів та формує значення виходів сигнальних модулів.

Область пам'яті для зберігання слів лічильника та таймера дозволяє при їх актуалізації зберігати поточні значення відліку та часу. Для звертання до блоків даних в програмі користувача використовують покажчики для позначення на тип даних, що зберігається. Так, для битів це буде DBX0.0… DBX65535.7, для байта даних – DBB0…DBB65535, для слова – DBW0…DBW65534, для подвійного слова – DBD0…DBD65532. Подібним чином можна звертатись до временних локальних даних в залежності від типу даних. Ці дані належать виключно блоку, в якому вони об'явлені. Звертання до биту локальних даних – L0.0…L65535.7, до байта – LB0…LB65535. Зауважимо, що ці дані є проміжними та після закриття кодового блока будуть втрачені.

1.4 Приклад розроблення ППЗ для установки водопостачання

Для подальшого ознайомлення з середовищем програмування розробимо програму користувача, яка реалізує алгоритм дистанційного керування установкою водопостачання. Схема установки водопостачання зображена на рис. 1.44.

Умови завдання:

1) Опис системи управління водопостачанням.

Об'єктом керування є ємність для накопичення води, яка наповнюється із свердловини за допомогою насоса (nasos), який встановлений на вхідному трубопроводі. Вода витрачається для поливу. В ємності встановлено два контактних датчика, які визначають мінімальний (min_l) та максимальний рівні (max_l). Датчики рівня поплавкого типу підключені по схемі *NC*, тобто, нормально-замкнений контакт. Це означає, якщо ємність порожня, то вони у замкненому стані, але, якщо рівень води в ємності поступово збільшується, вони по черзі розмикаються, спочатку датчик мінімального рівня, потім – датчик максимального рівня. В вихідному трубопроводі встановлений електромагнітний клапан (klapan). Система управління може працювати у двох режимах: ручному або автоматичному.

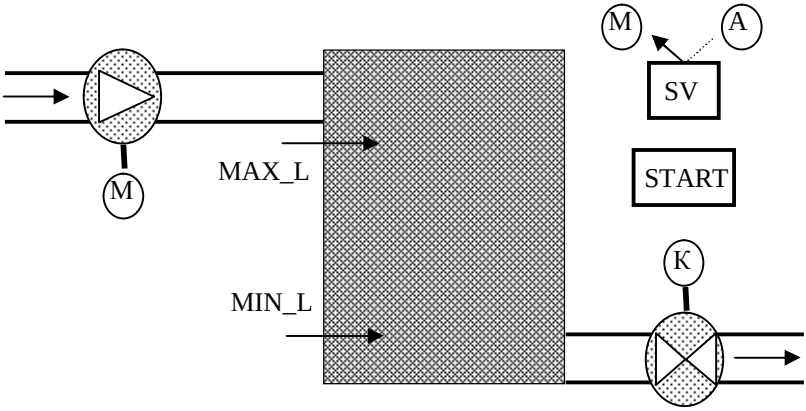


Рис. 1.44. Схема установки водопостачання

2) Алгоритм роботи системи управління водопостачанням.

1. Перемикач SV визначає режим роботи установки. Якщо його контакти розімкнути, то установка працює в ручному режимі. В протилежному випадку – режим автоматичний.

2. В ручному режимі, якщо контакти кнопки START в розімкненому стані, то установка не працює. В протилежному випадку – установка працює. В цьому режимі враховуються сигнали від датчиків рівня для перемикання живлення насоса та клапана.

3. В автоматичному режимі, якщо контакти датчика min_l та max_l замкнені, то ємність порожня. Автоматично вмикається насос та вимикається клапан. Якщо контакти датчика min_l та max_l розімкнуті, то ємність заповнена: автоматично вимикається насос, та вмикається клапан. Якщо контакти датчика max_l розімкнені, контакти датчика min_l замкнені (рівень в ємності середній), то клапан залишається включеним, а насос – не вмикається.

Зауважимо, що приклад є умовним і має лише навчальну мету.

Символьні імена в програмі користувача відповідають фізичному устаткуванню. Також, в процесі розроблення буде застосована змінна, яка буде зберігати комбінацію станів датчиків рівня. Це буде так званий прапорець (flag_l).

Почніть розроблення програми користувача з формування умови залежно від стану датчиків рівня. Для цього виділить ЛКМ ланцюг, який перетвориться з суцільної лінії чорного кольору на пунктирну лінію синього кольору. Додайте послідовно до ланцюга два нормально відкритих контакти. Для цього ЛКМ виберіть піктограму «**-| / |-**». Завершить ланцюг елементом обмотка «**-( )-**», який визначає результат логічної операції над двома попередніми елементами. На рис.1.45 показаний результат будови першого ланцюгу.

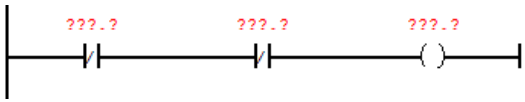


Рис. 1.45. Ланцюг з елементами «контакт» та «обмотка»

Елементам ланцюгу (див. рис. 1.45) необхідно надати абсолютні адреси (або надати символьні імена), тобто об'явити змінні. Для цього ЛКМ виділить символи «???.?» та в підсвічене червоним кольором поле введіть абсолютну адресу, наприклад, I1.0. та натисніть кнопку «Enter» на клавіатурі. Нижче поля редагування знаходиться рядок для редагування таблиці символів, який зображений на рис. 1.46. Заповніть поля символьного імені та коментаря відповідно. Для створення паралельного з'єднання в ланцюзі потрібно ЛКМ виділить місце роз'єднання. Далі ЛКМ виберіть піктограму  , з'являться два символи «↓» на синьому фоні. Додайте нормально-закритий контакт «┤ ├». Закриття паралельного з'єднання в ланцюзі здійснюється за допомогою піктограми  . Зауважимо, що якщо активована кнопка  , то елементи ланцюга будуть зображені з символьними іменами. Як видно з рис. 1.47 символьні імена виділяються подвійними лапками, наприклад, «flag_l».

Рис. 1.46. Рядок для введення даних про змінну

Далі введіть абсолютну адресу наступного контакту (I1.1) . Для обмотки абсолютна адреса буде M0.0. В результаті буде отримано ланцюг, який зображений на рис.1.47.

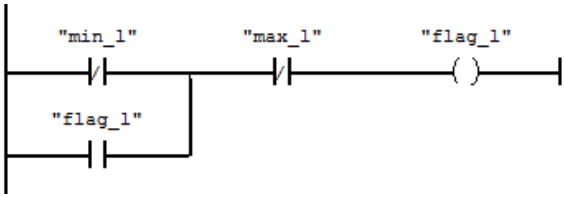


Рис. 1.47. Ланцюг аналізу стану дискретних датчиків рівня

Надамо пояснення щодо рис. 1.48. По-перше, цей ланцюг визначає стан датчиків рівня та формує статус прапорця. По-друге, в ланцюг паралельно змінної мінімального рівня додано значення прапорця для так званого «ефекту підхвату». Щодо логіки роботи цього ланцюгу, то змінна flag_l прийме статус «TRUE», якщо обидва датчики будуть у стані «FALSE». Такий статус свідчить про то, що ємність порожня. В цьому випадку насос ввімкнеться, а клапан буде в цей час закритим. Також змінна flag_l буде в статусі «TRUE», навіть, якщо рівень в ємності буде вище мінімального рівня (завдяки підхвату), що не призведе до перемикання насоса та клапана. Лише після перевищення максимального рівня змінна flag_l отримає статус «FALSE». В цей момент клапан відкриється, а насос вимкнеться. Поступове зменшення рівня в ємності не призведе до змін стану насоса та клапана, поки рівень не досягне мінімального значення.

Наступний ланцюг призначений для керування станом насоса. Він зображений на рис. 1.48. В даному ланцюзі враховано стан перемикача вибору режиму роботи установки та стан кнопки ручного керування установкою. Третій ланцюг, який зображений на рис. 1.49, керує станом клапана. Як бачимо, цей ланцюг відрізняється від попереднього (див. рис. 1.48) інвертуванням прапорця flag_l, тобто змінна в цьому ланцюзі буде оброблена як нормально-відкритий контакт. Подібний результат можна отримати за допомогою елемента «┤NOT├» для інвертування стану обмотки klapan. Для додавання цього елементу потрібно в стовпчику з ім'ям *Edit project:* ЛКМ вибрати вкладення *Catalog*, а в ньому – перелік логічних операторів *Bit logic*.

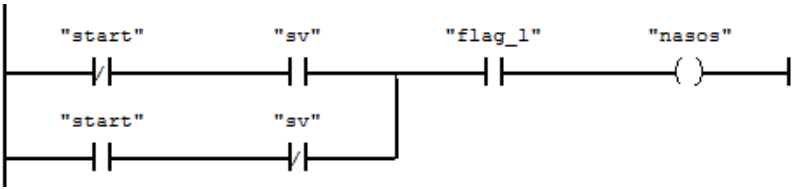


Рис. 1.48. Ланцюг керування насосом

На рис. 1.50 зображено кінцевий результат щодо роботи редактора таблиці символьних імен.

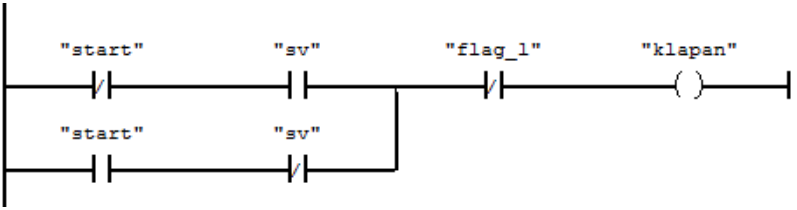
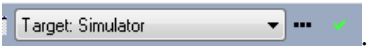


Рис. 1.49. Ланцюг керування клапаном

...	Symbol	Address	Type	Symb.-Comment
1	min_l	I 1.0	BOOL	min level
2	max_l	I 1.1	BOOL	max level
3	flag_l	M 0.0	BOOL	flag level
4	klapan	Q 0.6	BOOL	0 - OFF, 1 - ON
5	start	I 1.6	BOOL	button start with fix 1- ON
6	sv	I 1.7	BOOL	svitch 0 - man, 1 - auto
7	nasos	Q 0.7	BOOL	0 - OFF, 1 - ON
8				

Рис. 1.50. Таблиця символьних імен

Наступний етап розроблення програми користувача – це завантаження її до ПЛК для налагодження. Але зараз буде використаний програмний симулятор контролера. Вибір програмного симулятора здійснюється в основному рядку з інструментами, в змінному переліку



. Натиснення ЛКМ на кнопку «▪▪▪» активує вікно налаштувань програмного симулятора. Зараз налаштування залишимо без змін, тобто «за умовчанням».

Завантаження програми користувача здійснюється натисненням ЛКМ на першу кнопку в групі кнопок  основного рядку з інструментами. Відкриється діалогове вікно із повідомленням, що блок не збережений і запитанням про його збереження. Після позитивної відповіді середовище завантажить програму користувача до програмного симулятора ПЛК. При цьому відкриється вкладення з центром керування ПЛК *CPU-Control Center* в стовпчику *Edit project*. Запустить контролер (симулятор) на виконання програми користувача натисненням ЛКМ на кнопку *RUN*. Для спостерігання за станом змінних активуйте натиском ЛКМ на кнопку *Monitoring* у стовпчику *Speedbar* праворуч вікна інтерфейсу користувача. З'явіться попередження, що в режимі *Monitoring* деякі функції недоступні. Погодьтесь з попередженням натиснувши ЛКМ кнопку *OK*. Далі робочий простір з програмою користувача змініть зовнішній вигляд, причому червоним кольором будуть помічені лінії зв'язку та змінні зі статусом «TRUE». Якщо стан лінії зв'язку «FALSE», вона відображується пунктиром чорного кольору. Фрагмент вікна з програмою користувача в режимі налагодження зображений на рис. 1.51.

На рис. 1.51 система водопостачання знаходиться в режимі ручного керування (змінна SV у стані FALSE), а кнопка «Старт» активована (змінна START у стані TRUE). Це означає, що ємність порожня (змінна merker у стані TRUE), і відповідно насос включений (змінна nasos у стані TRUE). В режимі налагодження можлива онлайн-зміна стану змінних простим натиснення ЛКМ на відповідну змінну. Однак є більш наочний спосіб спостерігання за змінними в програмі користувача. Це

використання макета віртуального ПЛК. Для запуску цього інструмента натисніть ЛКМ на кнопку *PLC-Mask* у стовпчику *Speedbar* праворуч екрану інтерфейсу користувача. На рис. 1.52 зображений макет симулятора контролера, проте він потрібує додаткового налаштування.

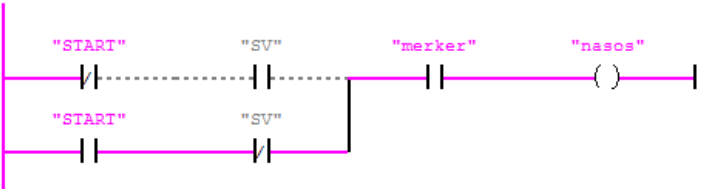


Рис. 1.51. Ланцюг з елементами в режимі *Monitoring*

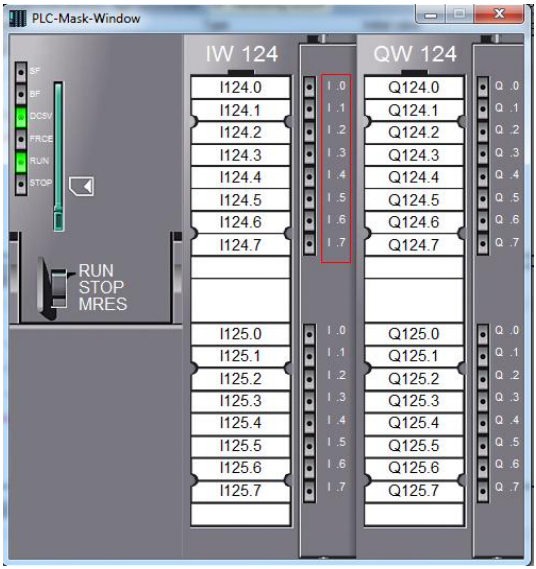


Рис. 1.52. Макет симулятора ПЛК

Як бачимо з рис. 1.52 макет ПЛК має два модулі: модуль вхідних сигналів *IW124* та модуль вихідних сигналів *QW124*. Але адреси моду-

лів не відповідають адресам змінних в програмі користувача. Потрібно встановити для вхідного та вихідного модулів вірну адресу: ця адреса дорівнює нулю. Для цього подвійним кліком ЛКМ по модулю відкрийте вікно і введіть потрібне значення для обох модулів (це буде «0»). Далі ЛКМ натискайте на покажчики праворуч, які окреслені червоною рамкою на модулі введення (вони стануть зеленими) та спостерігайте за зміною стану покажчиків на модулі виведення. Результат таких дій поданий на рис. 1.53. Зараз активований автоматичний режим: ємність заповнена, насос не працює, а клапан відкритий для споживання води. Якщо закрити вікно з макетом ПЛК, то усі налаштування будуть втрачені.

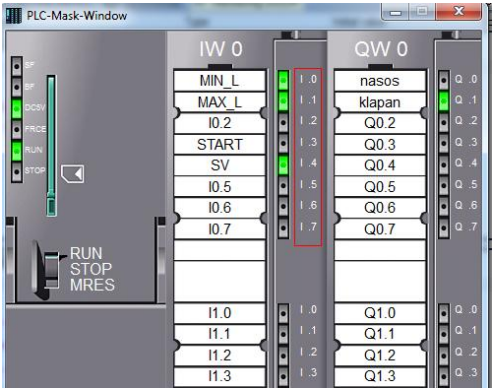


Рис. 1.53 – Макет симулятора ПЛК з налаштованими адресами відповідно до програми користувач

В середовищі *WinPLC V5* є ще один інструмент налагодження програми користувача. Це режим відображення образу процесу (*Display process-image-window*). Він активується натисканням ЛКМ на кнопку  , яка знаходиться в рядку інструментів швидкого доступу. Однак, спочатку активуйте режими символьного адресування та режим відо-

браження символьної інформації (відповідно натиснувши ЛКМ на кнопки  та  в рядку інструментів швидкого доступу). Далі натисніть ЛКМ на кнопку  для запуску режиму відображення образу процесу. Відкриється вікно для відображення стану змінних, яке зображено на рис.1.54. Проте, цей режим теж потрібує налаштування. Для налаштування режиму відображення стану змінних ПКМ в зоні вікна відображення образа процесу відкрийте контекстне меню та виберіть команду *Configure*. Відкриється вікно *Process-Image-Configuration*, яке зображене на рис.1.55. В цьому вікні можна додати або видалити змінні образа процесу.

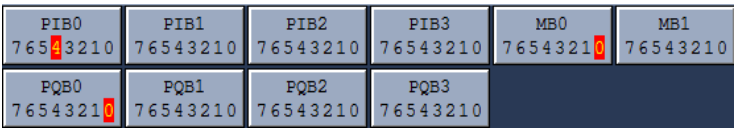


Рис. 1.54. Вікно відображення образа процесу

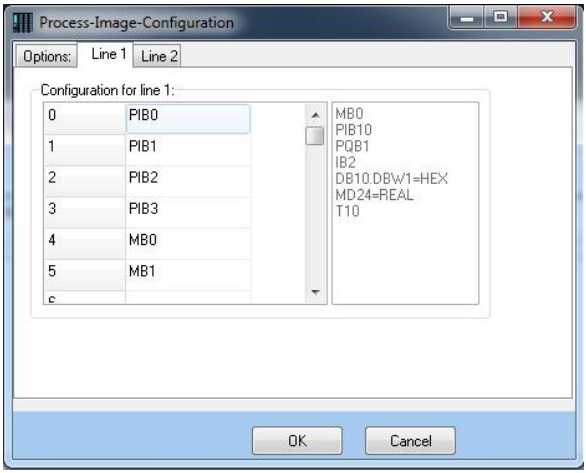


Рис. 1.55. Вікно налаштування образа процесу

Отже, на прикладі розроблення ППЗ для системи управління установкою водопостачання було розглянуто основні етапи програмування ПЛК *VIPA* та проведено знайомство з інтерфейсом користувача середовища програмування *WinPLC V5*. Насамкінець відмітимо, що в цьому проекті використані лише логічні оператори. У другому розділі ми ще повернемося до розглянутого проекту та доповнимо його додатковими функціональними можливостями, наприклад використаємо таймери та лічильники. Також в наступних прикладах будуть розглянуті питання створення функціональних блоків та функцій з блоками даних.

Питання для самостійного контролю.

1. Наведіть коротку характеристику стандарту *IEC 61131*.

2. Яка структура та принцип дії ПЛК скануючого типу?

3. Перелічить основні технічні характеристики ПЛК *VIPA*.

4. Яка структура та склад *WinPLC V5*? Перелічить основні модулі середовища програмування?

5. Який порядок взаємодії ПЛК та ПК? Що означає поняття «програматор»?

7. Надайте коротку характеристику мов програмування ПЛК у середовищі *WinPLC V5*.

8. Надайте пояснення щодо понять «операнд» та «оператор» у середовищі *WinPLC V5*.

9. Який принцип адресування операндів в пам’яті ПЛК *VIPA*?

10. Наведіть характеристику простих типів даних у середовищі *WinPLC V5*.

11. Перелічіть складені типи даних. Надайте їхню характеристику.

12. Як співвідносяться глобальні та локальні змінні.

13. Яке призначення організаційних блоків у програмі користувача у середовищі *WinPLC V5*?

14. Надайте пояснення щодо елемента програми «блок».

15. Перелічить основні типи блоків у програмі користувача.

16. Надайте пояснення щодо визначень «актуальний параметр», «формальний параметр».

17. Поясніть, що означає інтерфейс та структура кодового блоку і блоку даних.

18. Яким чином співвідносяться значення реальних каналів та їхнє зображення в пам'яті контролера?

РОЗДІЛ 2
ПРАКТИКУМ З РОЗРОБЛЕННЯ ППЗ ДЛЯ ПЛК *VIPA* В СЕРЕДОВИЩІ *WINPLC7 V5*

2.1 Розроблення ППЗ для системи дискретного управління водонагрівачем

Розроблення ППЗ для системи дискретного управління водонагрівачем почнемо з розгляду умов завдання. Схема установки гарячого водопостачання зображена на рис. 2.1.

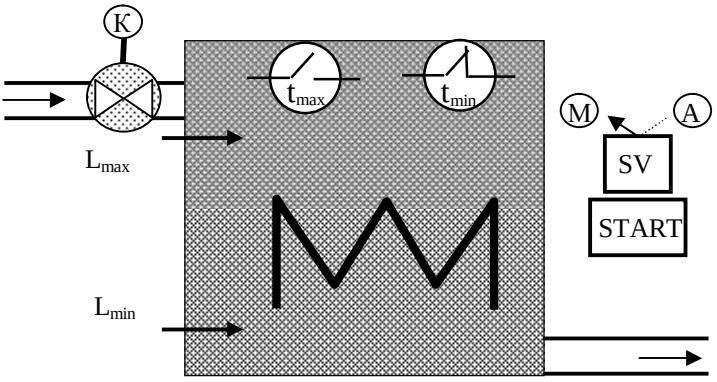


Рис. 2.1. Схема установки гарячого водопостачання

Умови завдання:

1) Опис системи управління установкою гарячого водопостачання

Установка гарячого водопостачання – це ємність з електричним нагрівачем. На вхідному трубопроводі встановлений електромагнітний клапан (klapan), стан якого залежить від стану контактних датчиків рівня. Датчики визначають мінімальний (min_l) та максимальний рівні (max_l). Датчики рівня поплавкого типу підключені по схемі *NC*, тобто, нормально-замкнений контакт. Це означає, якщо ємність порожня, то вони у замкненому стані, але, якщо рівень води в ємності поступово збільшується, вони по черзі розмикаються, спочатку датчик мінімального рівня, потім – датчик максимального рівня. Електричний нагрівач вмикається та вимикається в залежності від стану контактних датчиків температури при наявності води в ємності. Датчики визначають мінімальне (min_t) та максимальне значення температури (max_t). Якщо температура води нижче 40°C, то датчик з контактами типу *NO* замкнеться, а *NC* – розімкнеться. Якщо температура вище 40°C, то датчик з контактами типу *NO* розімкнеться, а *NC* – замкнеться. Аналіз стану датчиків з боку контролера дозволяє визначити в якому стані повинен бути нагрівач. Для керування установкою в системі передбачені перемикач режимів (SV) та кнопка ручного вмикання нагрівача (start). Перемикач на два положення режимів встановлює ручний або автоматичний режим роботи. В ручному режимі вмикання та вимикання нагрівача здійснюється за допомогою кнопки з фіксуванням.

2) Алгоритм роботи системи управління гарячим водопостачанням.

1. Перемикач SV визначає режим роботи установки. Якщо його контакти розімкнути, то установка працює в ручному режимі. В протилежному випадку – режим автоматичний.

2. В ручному режимі, якщо контакти кнопки start в розімкненому стані, то нагрівач не працює. В протилежному випадку – нагрівач працює. В цьому режимі враховуються сигнали від датчиків рівня та від датчиків температури для керування живленням нагрівача та клапана.

3. В автоматичному режимі, якщо контакти датчика min_l та max_l замкнені, то ємність порожня. Автоматично вмикається клапан. Якщо контакти датчика min_l та max_l розімкнуті, то ємність заповнена: автоматично вимикається клапан. Якщо контакти датчика max_l розімкнені, контакти датчика min_l замкнені (рівень в ємності середній), то клапан залишається відкритим, нагрівач ввімкнений.

4. В автоматичному режимі, якщо контакти датчика min_t розімкнені та max_t замкнені, то температура води менше 40 °С. Нагрівач вмикається. Якщо контакти датчика min_t замкнені та max_t розімкнені, то температура води більше 40 °С і нагрівач вмикається. Якщо контакти датчиків max_t та min_t, або одночасно розімкнені, або замкнені (температура води 40 °С), то нагрівач не змінює свого стану.

Вирішення завдання:

Для системи керування в якості керуючого пристрою використаємо ПЛК *VIPA System100V*, який був розглянутий у першому розділі. Проте, необхідно більш докладніше розглянути стенд з цім контролером. Схема розташування компонентів на стенді зображена на рис. 2.2.

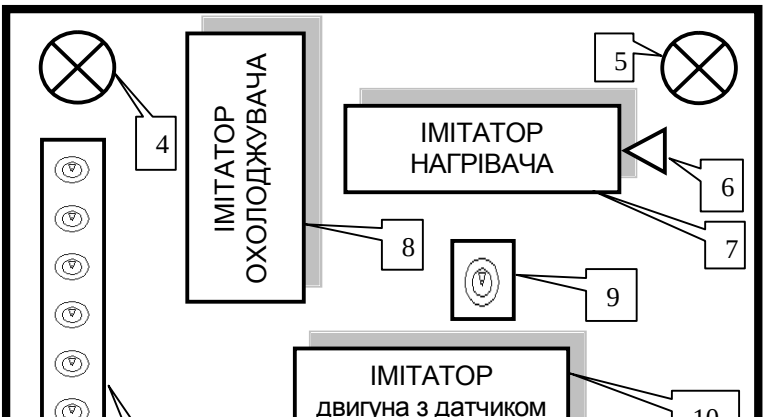


Табл. 2.1 – Електричні з'єднання каналів субмодулів ПЛК з датчиками та вихідними пристроями

Субмодулі ПЛК *VIPA System115V*							
X3 DI		X4 DI		X5 DIO		X6 DO	
Абс. адреса	Датч.	Абс. адреса	Датч.	Абс. адреса	Датч./ Вих. пр.	Абс. адреса	Вих. пр.
I0.0	-	I1.0	SV1	I2/Q0.0	-	Q1.0	-
I0.1	-	I1.1	SV2	I2/Q0.1	-	Q1.1	-
I0.2	-	I1.2	SV3	I2/Q0.2	-	Q1.2	-
I0.3	-	I1.3	SV4	I2/Q0.3	-	Q1.3	-
I0.4	Prox.S	I1.4	SV5	I2.4	-	Q1.4	-
I0.5	-	I1.5	SV6	I2.5	-	Q1.5	-
I0.6	NC T/S	I1.6	SV7	I2.6	Cooler	Q1.6	Ind1
I0.7	NO T/S	I1.7	SV8	I2.7	Heater	Q1.7	Ind2

На рис. 2.2 цифрами позначені компоненти стенда:

1 – перемикач вмикання/вимикання загального живлення стенда;

2 – контролер *VIPA* серії *System 100V* модель *CPU115DIO32SER* (зак. № 115-6BL32) із зовнішнім джерелом живлення *VIPA* типу *PS207* (зак. №207-1BA00), є моноблок, який поєднує в одному корпусі модуль центрального процесора та субмодулі дискретних каналів введення/виведення;

3 – імітатори дискретних вхідних сигналів, 8 перемикачів типу «сухій контакт»;

4, 5 – імітатор електродвигуна з безконтактним індуктивним датчиком для обчислення кількості оборотів або інтервалів часу та кнопка вмикання імітатора;

6 – дискретні датчики температури з NO та NC контактами на 40 °С;

7 – імітатор нагрівача, дротовий опір в керамічному корпусі типу ПЕВ-100 номіналом 750 Ом;

8 – імітатор охолоджувача, вентилятор постійного струму з напругою живлення 12 В для обдування повітрям нагрівача;

9, 10 – індикатори (можуть виконувати роль ламп сигналізації аварій імітаторів або індикації режимів роботи установки).

Схема електричних з'єднань каналів введення/виведення субмодулів ПЛК *VIPA System115* з датчиками та вихідними пристроями подана в табл.2.1.

2.1.1. Апаратна складова проекту для VIPA115

Отже запустить середовище для програмування контролерів *VIPA* та створіть в стартовому вікні новий проект. Далі на підставі завдання та схеми електричних з'єднань проведіть налаштування апаратних ресурсів проекту. Для цього подвійним кліком ЛКМ у вкладенні *Solution* дерева проекту потрібно вибрати ресурс *Hardware stations*. Далі подвійним кліком ЛКМ активуйте команду *Create new*. В діалоговому вікні *Create new station:*, яке відкриється, введіть ім'я станції в поле *Name of station:*, наприклад це ім'я буде таким: *VIPA_CPU_115*. В результаті відкриється вікно *Station-Offline---VIPA115* програми-

конфігуратора апаратних ресурсів зі вкладенням *Select PLC-system* для вибору апаратної платформи. Виходячи із умов завдання, виберіть ЛКМ рядок *VIPA System 100V* та натисніть ЛКМ на кнопку *Create*. В результаті у вікні з ім'ям *Station-Offline---VIPA115* з'явиться вкладення *UR0* з рядками слотів та стовпчиками з назвою, номерами для замовлення та адресами для заповнення модулями. Вкладення *UR0* (скорочення англ. від «*universal rack*») є відображенням фізичного розміщення центрального процесорного та сигнального модулів на спеціальну *DIN*-рейку (розміром 35 мм). Зауважимо, що сигнальні модулі перед встановленням на рейку з'єднуються з процесорним модулем та між собою за допомогою спеціального шинного з'єднувача, який розміщується позаду модулів. Ці з'єднувачі є різних розмірів, тобто розраховані на з'єднання двох, чотирьох або вісьмох модулів. Крім того з'єднувачі є універсальними для модулів серій *System100V* та *System200V*, тобто сумісними. Кількість слотів в рейці *UR0* є фіксованою (програмно) та дорівнює 32-м слотам (від 0-го до 31-го), але відповідає можливостям процесорних модулів (наприклад, до процесорів серії *System100V*, крім *CPU112*, можливо підключити до 4-х модулів).

Модулі повинні встановлюватись на рейку *UR0* починаючи з першого посадочного місця (слот «0»). Приклад заповнення рейки *UR0* процесорним модулем *CPU115DIO32SER* (зам. № 115-6BL32) зображений на рис. 2.3.

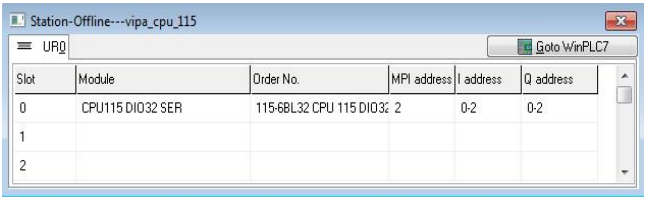
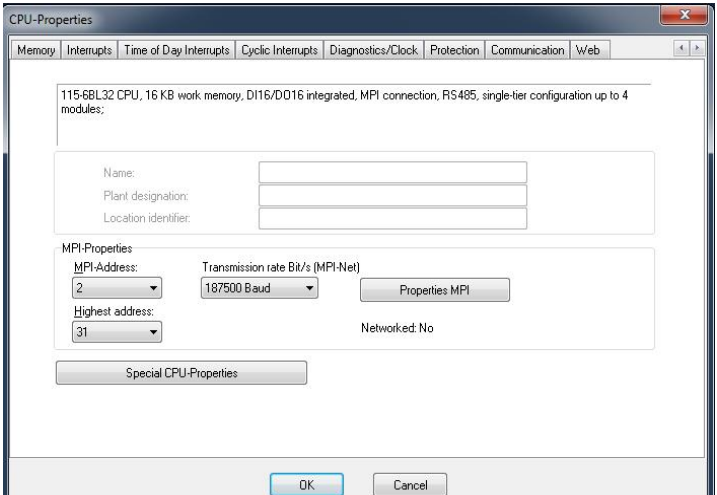


Рис. 2.3. Приклад заповнення рейки *UR0* модулем *VIPA115*

Для отримання результату, який зображений на рис. 2.3 потрібно зробити наступне. Виділіть ЛКМ рядок з нульовим слотом на рейці *UR0*. Він одразу підсвітиться синім кольором. Якщо обрати потрібний процесорний модуль, то рядок змінить колір на темно-зелений. Подвійний клік ЛКМ по обраному модулю заповнить посадочне місце. При цьому підсвічування рядку відключиться. Далі до рейки можна додати інші модулі (сигнальні або функціональні з переліку у каталозі), які сумісні з цією моделлю процесору. Причому, якщо вибрати потрібний модуль, то підсвітиться темно-зеленим кольором рядок слоту, до котрого можливе підключення цього модуля. Також програма позначить світло-зеленим кольором ті слоти, до яких можливе апаратне підключення інших модулів. Програма дозволяє заповнити усі слоти, але отримана програмна конфігурація буде невірною.

Отже, для додавання до рейки *UR0* модуля центрального процесору *CPU115DIO32SER* виділить потрібний слот. Далі праворуч у вікні каталогу з переліком модулів *VIPA* послідовно ЛКМ відкрийте папки «*VIPA 100V/200V*», «*CPU*», «*CPU11X*», «*CPU115*». Виділить та подвійним кліком ЛКМ додайте процесорний модуль *115-6BL32 CPU 115DIO32 SER* до першого слота у рейку *UR0*. Для налаштування властивостей модуля подвійним кліком ЛКМ по слоту з модулем відкрийте вікно *CPU-Properties*. Зображення вікна *CPU-Properties* для налаштування властивостей модуля подано на рис. 2.4.



Вікно налаштувань властивостей модуля (див. рис. 2.4) складається з декількох вкладень. Це вкладення: *General*, *Startup*, *Cycle/Clock Memory*, *Retentive Memory*, *Memory*, *Interrupts*, *Time of Day Interrupts*, *Cyclic Interrupts*, *Diagnostics/Clock*, *Protection*, *Communication* та *WEB*. В основному ці вкладення призначені для налаштування режимів роботи модуля та визначення характеристик його ресурсів: часу циклу сканування, об'єму пам'яті для зберігання даних користувача, програмних та апаратних переривань, функцій діагностування та параметри зв'язку. Зазначимо, що налаштування у вкладеннях «за умовчанням» оптимізовані на вирішення більшості стандартних завдань. В подальшому ми розглянемо деякі налаштування у цих вкладеннях. Проте зараз зосередимося на вкладенні *General* (див. рис.2.4). В цьому вкладенні вказані налаштування основних властивостей процесорного модуля.

Вікно налаштувань основних властивостей модуля поділено на три зони. Верхня зона інформує про склад модуля та його основні апаратні ресурси. Нижче розміщена зона, до якої вносяться дані про розробника та проект, які додаються автоматично, якщо заповнені відповідні поля при створенні проекту та при налаштуванні властивостей системи програмування *WinPLC V5*. Ще нижче розміщена зона з ім'ям *MPI-Properties* для налаштувань параметрів *MPI*-мережі. В цій зоні вказана *MPI*-адреса модуля «2». Якщо модуль буде використовуватись один, адресу можна залишити як є. Зауважимо, що адреса «0» зарезервована за середовищем програмування, тобто за програматором *PG*. Його фун-

кцію виконує ПК зі встановленим СПЗ, тобто середовищем *WinPLC V5*. Для створення *MPI*-мережі потрібно ЛКМ натиснути на кнопку *Properties MPI*. Далі додати *MPI*-мережу за допомогою кнопки *New*, але зараз для вирішення завдання це не потрібно. В нижній частині вікна налаштувань розміщена кнопка *Special CPU-Properties* для налаштувань спеціальних властивостей субмодулів. Це потрібно для налаштування апаратних входів та виходів для оброблення аварійних сигналів, сигналів від енкодерів та формування ШІМ-сигналів. Якщо ЛКМ натиснути на кнопку *Special CPU-Properties*, то відкриється вікно *Module-Parameters* зі вкладеннями *Address/Type* та *Parameters*. Зображення вікна *Module-Parameters* зі вкладенням *Parameters* подано на рис. 2.5.

На рис. 2.5 показане вкладення *Parameters* з параметрами субмодулів (поз. 1, рис. 2.5) та відкрито перелік *Selection* для налаштувань першого каналу (поз. 2, рис. 2.5). Якщо коротко описати перелік «1» з рис. 2.5, то в ньому наведені категорії налаштувань параметрів апаратних входів та виходів. В переліку «2» на рис. 2.5 зазначені можливі варіанти значень параметрів для вибору. Зараз ці налаштування не мають значення, тому закрийте вікно натиснувши на кнопку *OK*. Далі закрийте вікно *CPU-Properties* також натиснувши на кнопку *OK*.

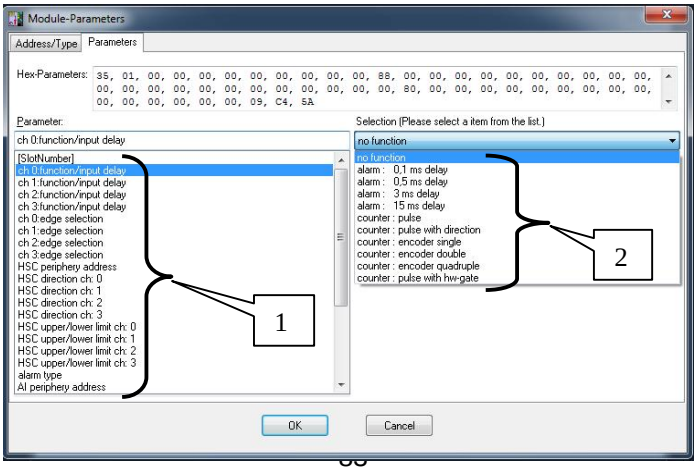


Рис. 2.5. Вікно налаштувань властивостей субмодулів *VIPA115*

Наступним кроком додайте створену апаратну конфігурацію до проекту. Для цього необхідно ЛКМ вибрати перелік *File/Project* з основного меню програми-конфігуратора, а в ньому команду – *Save the active Station in the WinPLC7-Project*. Далі підтвердить пересилання конфігурації натисненням ЛКМ на кнопку *Да*. Наступний запит в діалоговому вікні підтвердить запам'ятовування конфігурації станції перед пересиланням до проекту. На завершення програма-конфігуратор повідомить про успішне копіювання конфігурації до проекту повідомленням «*All Files are successful copied!*». В результаті в проекті з'явиться новий компонент – блок *SDB* (*System Data Block*), який є конфігураційним блоком апаратних ресурсів ПЛК. Зауважимо, що в демо-версії програми-конфігуратора пряма передача конфігурації до ПЛК неможлива. Для випадку повнофункціональної версії програми завантаження конфігурації здійснюється за допомогою команди *Send configuration to the PLC* в переліку *Online* з основного меню програми.

2.1.2. Програмна складова проекту для VIPA115

Після створення апаратної конфігурації можна почати розробляти програму користувача. З огляду на умови завдання (див. стор. 72) використаємо деякі результати, які викладені у першому розділі. Символьна таблиця змінних зображена на рис. 2.6, а програма користувача – на рис. 2.7.

Програма користувача (див. рис. 2.7) складається з аналізу стану датчиків рівня та температури і двох умов для керування клапаном та нагрівачем. В умовах для керування клапаном та нагрівачем враховано стан кнопок керування та дискретних датчиків температури та рівня.

...	Symbol	Address	Type	Symb.-Comment
1	min_l	I 1.0	BOOL	min level
2	max_l	I 1.1	BOOL	max level
3	flag_l	M 0.0	BOOL	flag level
4	min_nc_t	I 0.6	BOOL	min temperatura NC
5	max_no_t	I 0.7	BOOL	max temperatura NO
6	flag_t	M 0.1	BOOL	flag temperatura
7	klapan	Q 0.6	BOOL	klapan 0 - OFF, 1 - ON
8	start	I 1.6	BOOL	button 0 - OFF,1 - ON

Рис. 2.6. Символьна таблиця змінних

На закінчення зауважимо, що розроблена програма користувача демонструє лише принципи логічного керування вихідними пристроями в залежності від результатів логічних умов, які формуються дискретними датчиками. Для реальних систем керування в програмі користувача потрібно враховувати усі можливі аварійні ситуації та відповідні дії на їхнє виникнення, тобто формування сигналів блокування та сигналізації. Крім того потрібно структурувати проект для подальшого тиражування екземплярів ФБ та функцій. Це дозволяє без суттєвих часових витрат масштабувати (розширити) проект.

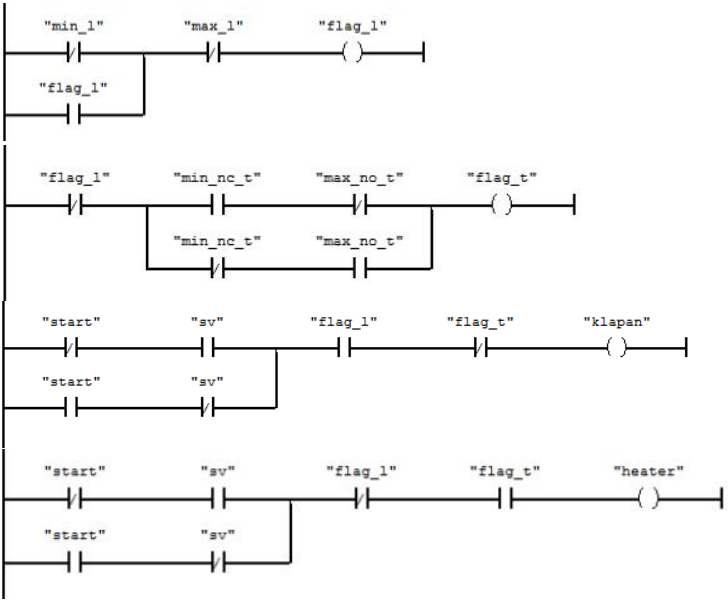


Рис. 2.7. Програма користувача для установки
гарячого водопостачання

2.2 Розроблення ППЗ для системи водопостачання з використанням таймерів та лічильників

В цьому завданні додамо нові умови до попереднього завдання, викладеного у п. 2.1. Так, клапан повинен вмикатись із затримкою на 10 *с* після вмикання нагрівача, а нагрівач вимикатись із затримкою на 5 *с* після вимикання клапана. Додатково потрібно рахувати кількість вмикань клапана та нагрівача, наприклад, для визначення моменту їхньої заміни в установці.

Перед модернізацією програми користувача спочатку розглянемо компоненти вкладення *Catalog* з вікна *Edit project:*, а саме таймери та лічильники. Ці компоненти розміщені у переліках *Timer* та *Counter* зі вкладення *Catalog*. Розглянемо більш докладніше таймери та лічильники з точки зору їхнього застосування в системах керування технологічними установками.

Таймери, як програмний компонент, дозволяють реалізувати часові інтервали для спостереження, очікування або можуть генерувати імпульсі. Таймери мають область пам’яті, яка зарезервована для них в пам’яті центрального процесорного пристрою. Ця область пам’яті резервує одне 16-бітове слово для кожного таймерного адреса. При програмуванні кількість таймерів залежить від типу та моделі процесорного модуля, наприклад для модуля центрального процесору *CPU115DIO32SER* ця кількість дорівнює 256 таймерів (див. табл. 1.1).

В середовищі *WinPLC7 V5* крім стандартних таймерів із стандарту *IEC61131-3* доступні спеціальні *S7*-таймери, а саме:

1) у вигляді блочних елементів:

- S_IMPULS (SI) – таймер «Імпульс»;
- S_VIMP (SV) –таймер «Розширений імпульс»;
- S_EVERZ (SE) – таймер «Затримка вмикання»;
- S_SEVERZ (SS) – таймер «Затримка вмикання з пам’яттю»;
- S_AVERZ (SA) – таймер «Затримка вимикання»;

2) у вигляді обмоток:

- (SI) – обмотка таймера «Імпульс»;
- (SV) – обмотка таймера «Розширений імпульс»;
- (SE) – обмотка таймера «Затримка вмикання»;
- (SS) – обмотка таймера «Затримка вмикання з пам’яттю»;
- (SA) – обмотка таймера «Затримка вимикання».

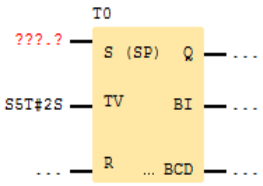


Рис. 2.8. Зображення блоку імпульсного таймеру (*SP*)

На мові *LAD* та *FBD* блочний таймер виглядає так, як це зображено на рис. 2.8. Формальні параметри таймера описані в табл. 2.2. Якщо використаний таймер у вигляді обмотки, то його тип розміщений всередині обмотки. Параметр часу розміщений нижче обмотки у спеціальному форматі *S5TIME*.

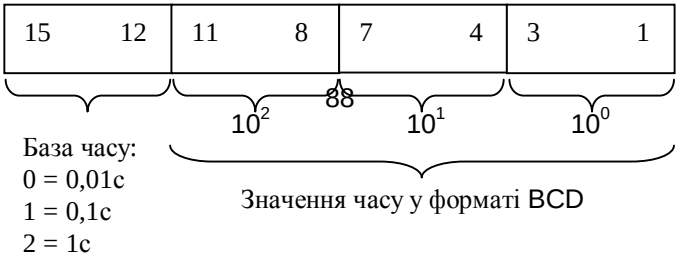
Табл. 2.2 – Інтерфейс блока таймера (на прикладі таймера *SP*)

Ім’я формального параметра	Тип даних	Призначення параметра
S	BOOL	Вхід запуску таймера
TV	S5TIME	Параметр часу
R	BOOL	Вхід скидання таймера
BI	WORD	Поточне значення часу у двійковому коді

BCD	WORD	Поточне значення часу у двійково-десятковому (BCD) коді
Q	BOOL	Стан таймера

Щоб скинути таймер, використовують обмотку скидання. Для перевірки стану таймера можна застосувати NO- та NC-контакти, які відображують стан таймера. Збереження значення параметра часу у двійковій формі в операнді розміром *WORD* можливе за допомогою блочного елемента *MOVE*. Параметр часу TV можна задати як константу, у вигляді операнда розміром *WORD* або як змінну у форматі *S5TIME*. Тривалість часу задається в годинах, хвилинах, секундах та мілісекундах. Область значень параметру часу таймера TV знаходиться у діапазоні від S5TIME#10ms до S5TIME#2h46min30s (що відповідає 9990 *с*). Точність представлення часу дорівнює 10 *мс* (за умовчанням). Для визначення константи можна використовувати префікси S5TIME# або скорочену S5T#. Визначення тривалості як константи буде таким: S5TIME#10s, що визначає інтервал часу тривалістю 10 *с*. Якщо в програмі користувача використано, наприклад, операнд MW20 для визначення тривалості, то цей операнд буде мати тип *WORD*. Якщо використано змінну із символьним ім'ям TIME1, яка містить тривалість часу – це буде змінна типу *S5TIME*.

Формат часу типу *S5TIME* структурно складається зі значення часу (*time value*) та бази часу (*time base*). Розраховується тривалість часу за наступною формулою: *Тривалість = Значення_часу × база_часу*. Отже, тривалість – це інтервал часу, протягом якого таймер активний (тобто веде відлік часу). Значення часу – це число циклів, які відпрацьовує таймер. База часу визначає інтервал часу для розрахунку тривалості. На рис. 2.9 показана структура формати часу типу S5TIME.



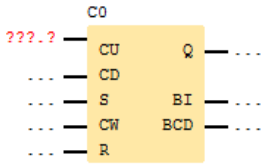


Рис. 2.10. Зображення
блока лічильника

Зауважимо, що точність представлення часу залежить від бази часу. Так, інтервал часу в 1 *с* можна задати по-різному:

- 2001_{hex}, якщо база часу дорівнює 1 *с*;
- 1010_{hex}, якщо база часу дорівнює 0,1 *с*;
- 0100_{hex}, якщо база часу дорівнює 0,01 *с*.

Останній варіант представлення інтервалу є найбільш прийнятним.

Лічильники дозволяють в програмі користувача рахувати імпульси (фронти імпульсів) від фізичних датчиків або різноманітні події. Лічильники можуть вести рахунок на зростання (прямий рахунок), на зменшення (зворотний рахунок) або в обох напрямах. Значення параметра рахунку займає три розряди (від 000 до 999) у форматі *BCD*. Параметр обчислення лічильників розташовується в системній пам'яті процесорного модуля. Кількість лічильників в програмі визначається версією модуля, наприклад для модуля центрального процесору *CPU115DIO32SER* ця кількість дорівнює 256 лічильників (див. табл. 1.1).

У середовищі *WinPLC7 V5* крім стандартних лічильників зі стандарту *IEC61131-3* є спеціальні, а саме:

1) у вигляді блочних елементів:

- ZAEHLER (S_CUD) – лічильник двонаправленого лічіння;
- Z_VORW (S_CU) – лічильник прямого лічіння;

- Z_RUECK(S_CD) – лічильник зворотного лічіння;

2) у вигляді обмоток:

- —(SC) – обмотка для встановлення начального значення параметра лічіння;
- —(CU) – обмотка лічильника прямого лічіння;
- —(CD) – обмотка лічильника зворотного лічіння.

На мові *LAD* та *FBD* блочний лічильник виглядає так, як це зображено на рис. 2.10. Формальні параметри лічильника описані в табл. 2.3. Якщо використаний лічильник у вигляді обмотки, то його тип розміщений всередині обмотки. Параметр рахування розміщений нижче обмотки.

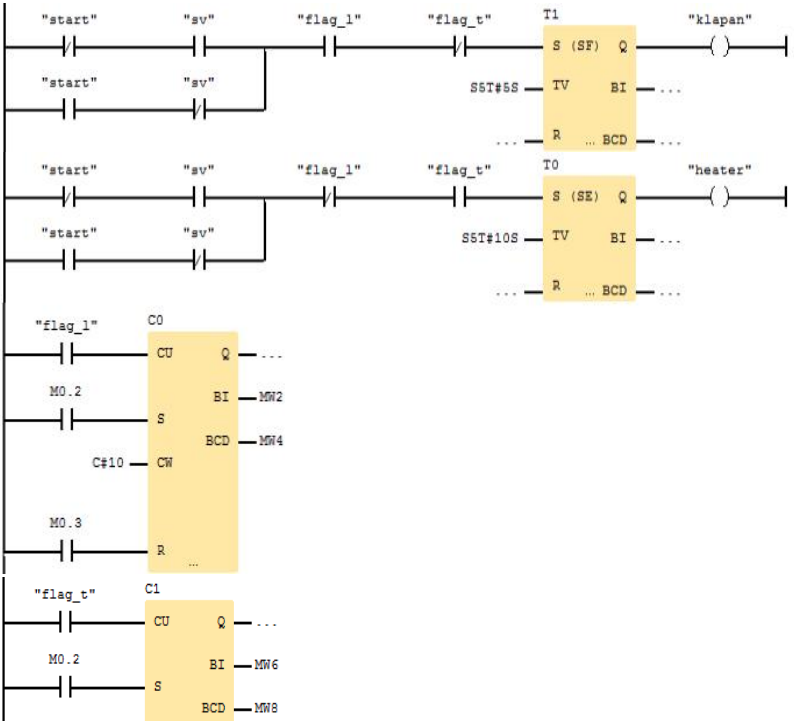
Лічильник встановлюється (занесення до лічильника значення параметра рахування), коли на вході установки S змінюється сигнал з «0» на «1» (тобто з'являється передній фронт). Якщо лічильник встановлений, то параметр рахування приймає значення на вході CW. Значення параметру лічіння визначається за допомогою змінної типу *WORD*. Якщо параметр лічіння константа, то це виглядає так: C#100 або W#16#0100, але лише для чисел в десятковому форматі. У разі використання операнда параметр рахування буде таким: MW56 (мер екерне слово) або "Count value" (символьне ім'я) типу *WORD*. Скидання лічильника (присвоєння параметру рахування значення «0») здійснюється, якщо на вході R буде передній фронт імпульсу.

Табл. 2.3 – Інтерфейс блока двонаправленого лічильника

Ім'я формального параметра	Тип даних	Призначення параметра
CU	BOOL	Вхід прямого відліку
CD	BOOL	Вхід зворотного відліку
S	BOOL	Вхід запуску лічильника

CW	WORD	Вхід попереднього встановлення параметра рахування
R	BOOL	Вхід скидання
BI	WORD	Поточне значення відліку у двійковому коді
BCD	WORD	Поточне значення відліку у двійково-десятковому коді (*BCD*)
Q	BOOL	Стан лічильника

Зараз можна почати розробляти програму користувача відповідно до завдання. Апаратна складова проекту, яка була створена раніше, залишається без змін. Змінюється лише програмна складова, а саме ланцюги керування вихідними пристроями. Крім того до програми буде додано два лічильника, для рахування кількості вмикань вихідних пристроїв. На рис. 2.11 зображена частина програми, яка відрізняється від програми користувача, яка зображена на рис. 2.7. Це ланцюги з 3-го по 6-ий (без ланцюгів аналізу дискретних датчиків рівня та температури).



На закінчення розгляду цього прикладу зауважимо, що розроблена програма користувача демонструє принципи логічного керування вихідними пристроями в залежності від результатів логічних умов, які формуються дискретними датчиками та з можливістю встановлення інтервалів часу затримки для керування вихідними пристроями.

2.3 Розроблення ППЗ для системи аналогового керування вихідним пристроєм водонагрівача

Розроблення ППЗ для системи аналогового управління вихідним пристроєм водонагрівача, який є дискретним елементом, можливо за допомогою використання ШІМ-сигналу. В даному прикладі буде доопрацьована програма користувача з попереднього підрозділу. Використання ШІМ-сигналу для управління вихідним пристроєм потрібує налаштування апаратного виходу субмодуля контролера в програмі конфігурування.

В цьому завданні використайте попередній проект з п. 2.2. Тому відкрийте та збережіть його з новим ім'ям. Далі запустить програму-конфігуратор апаратних ресурсів контролерів *VIPA*. На цей раз потрібно відкрити вікно *Module-Parameters* для налаштувань спеціальних властивостей процесорного модуля, а саме апаратних виходів сигналь-

ного субмодуля для роботи в режимі ШІМ-управління. Зображення вікна *Module-Parameters* зі вкладенням *Parameters* подано на рис. 2.5. Електричний нагрівач згідно табл. 2.1 підключений до 8-го виходу субмодуля вихідних каналів X3. Він може працювати в режимі генерування ШІМ-сигналу. Цей вихід відповідає вихідному каналу з ім'ям *Ch 1* в апаратній конфігурації модуля. Спочатку у вікні *Module-Parameters* ЛКМ виберіть параметр *PWM Mode* і праворуч, у полі *Selection* виберіть з випадаючого переліку рядок *Channel 1 enabled* для активування ШІМ на потрібному виході. Далі встановіть значення параметрів ШІМ-сигналу у вікні *Module-Parameters*:

- для параметра PWM timebase channel 1 – 1 *мс*;
- для параметра PWM period channel 1 – 10000 одиниць;
- для параметра PWM duty channel 1 – 1000 одиниць;
- для параметра PWM min pulse channel 1 – 1000 одиниць.

Після проведених налаштувань збережіть отриману конфігурацію та перенесіть її до проекту, як це було показано раніше.

Далі в програмі користувача відкрийте блок OB1 для редагування та додайте в 4-ий ланцюг новий елемент. Це буде блок SFC223 із бібліотеки стандартних блоків. Для додавання цього блока до ланцюга ЛКМ виділіть потрібне місце (після таймера T0 в 4-му ланцюгу), далі ЛКМ виберіть вкладення *Catalog* і відкрийте папку *Standard library* з переліком *Vipa*. Подвійним кликом ЛКМ додайте блок SFC223 до ланцюга у блоці OB1.

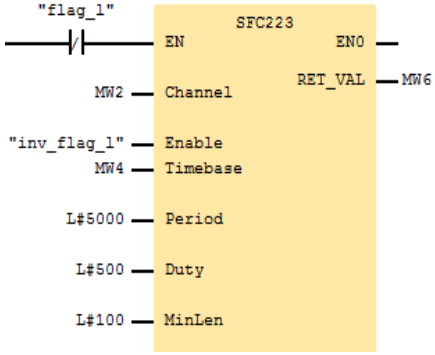


Рис. 2.12. Налаштування параметрів блока SFC223

В режимі ШІМ-регулювання змінна з ім'ям "heater" типу *BOOL* не потрібна, тому видаліть її з ланцюга. Далі блок SFC223 необхідно налаштувати, тобто з'єднати

його входи та виходи з потрібними операндами, наприклад як це показано на рис. 2.12. На рисунку показані параметри ШІМ, які співпадають з параметрами налаштування апаратних виходів в конфігурації субмодуля виходів процесорного модуля. Додамо у блоці OB1 операнди до блока SFC223. Але спочатку визначимо вже задіяні операнди в проекті. Для цього ЛКМ натисніть на кнопку  (команда *Assigment*), яка знаходиться в переліку основного меню середовища. Відкриється вікно з адресами операндів, які вже об'явлені в проекті. На рис. 2.13 зображено вікно співвідношень *Assigment* для поточного проекту з ім'ям *VIPA115.prj*, в якому відображені вже задіяні операнди. Цей інструмент дозволяє запобігти перехрещенню адресів меркерної пам'яті з боку операндів в пам'яті процесора. Відмітимо, що пам'ять у процесорному модулі розподілена на байти, тобто має байтове адресування.

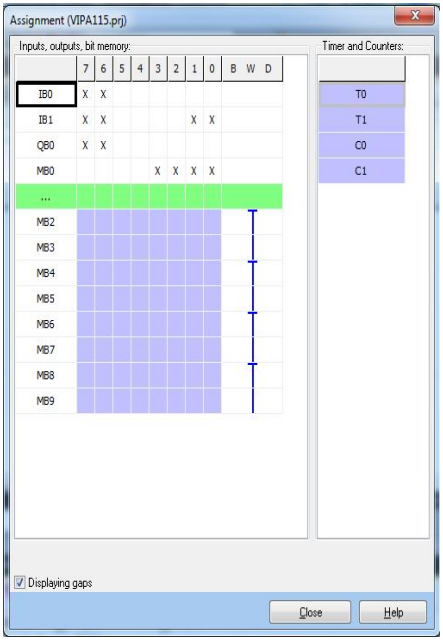


Рис. 2.13. Розподіл пам'яті в *CPU*

Як бачимо з рис. 2.13, останній адрес операнда – це MB9. Тому наступний операнд може мати абсолютну адресу починаючи з MB10. Для типів *INT* – потрібно два байти (розмір *WORD*), тому операнд з абсолютною адресою MW10 буде використовувати пам'ять з адресами MB10 та MB11. Відповідно тип *DINT* потребує чотири байти, а операнд буде таким – MD10. Більш наочно процес призначення адрес в пам'яті про-

цесора демонструє рис. 2.14. На цьому рисунку зображені байти для зберігання значень слова та подвійного слова. Адресування слова MW11 є помилковим, тому що використовує байти із суміжних слів MW10 та MW12.

Проте, ці параметри визначені константами, тому їхня зміна в процесі виконання програми користувача неможлива. Тому скористуємося ще одною можливістю, що надає середовище програмування *WinPLC7 V5*, а саме використання в проекті блоків даних (DB). Це надасть більшої гнучкості програмі користувача.

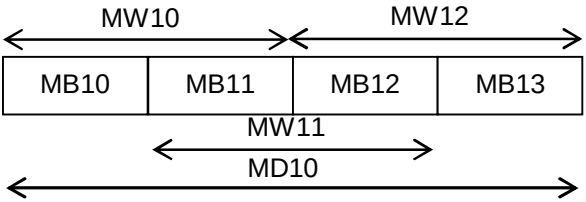


Рис. 2.14. Принцип адресування операндів в пам'яті ПЛК

Отже додайте до переліку *Blocks* у папку *DB* новий компонент – блок даних. Для цього ЛКМ у вкладенні *Solution* послідовно відкрийте переліки *Blocks* та *DB*, а в переліку *DB* подвійним кліком ЛКМ натисніть на команду * *Create new*. Праворуч від менеджера проекту відкриється вікно *Project content* з діалоговим вікном *Create a new block* для об'явлення блока даних та попередньо наданим номером блока. Натисніть ЛКМ на кнопку *Create a new block*. В результаті проведених дій в проекті з'явиться новий компонент – блок DB1, який одразу буде доступний для редагування. Далі необхідно заповнити цей блок. Для заповнення блоку даних необхідно враховувати типи даних формальних параметрів функціонального блока SFC223. Так, параметри ШІМ-регулятора (*Period, Duty, MinLen*) мають тип *DINT*, дозвіл на роботу

блока (*Enable*) – *BOOL*, номер каналу (*Channel*) та база часу (*Timebase*) – *INT*. Далі заповнимо блок DB1 параметрами, як це зображено на рис. 2.15. В даному випадку у блоці оголошень складний тип даних структура, яка складається з параметрів з різними типами даних.

Враховуючи вміст блоку даних DB1 необхідно внести деякі зміни до блоку OB1. По-перше, замініть операнд "flag_t" в 2-му ланцюзі замініть на перший операнд en_shim з блоку DB1. Для цього клікніть ЛКМ по існуючому операнду, який буде підсвічений. Далі введіть символи «DB1.», які активують режим автозаповнення в середовищі про-

*	Address	Declaration	Name	Type	Initial value	Comment
		var S		STRUCT		
.	0.0	var S	en_shim	BOOL	FALSE	enable
.	2.0	var S	channel	INT	1	channel ch0 - 1, ch1 - 2
.	4.0	var S	timebase	INT	0	timebase 0 - 0.1ms, 1 - 1ms
.	6.0	var S	period	DINT	L#5000	period 0-60000
.	10.0	var S	duty	DINT	L#500	duty 0-1000
.	14.0	var S	minlen	DINT	L#100	minlen 0-6000
.	18.0	var S	return_val	WORD	W#16#0000	kod error
		var S		END_STRUCT		

Рис. 2.15. Вміст блоку даних DB1 для блока SFC223

грамування *WinPLC7 V5*. Це виглядає так, як зображено на рис. 2.16. Подвійним кліком виберіть ЛКМ запропонований операнд з блоку даних DB1. В результаті обмотка отримає операнд з новим ім'ям en_shim.

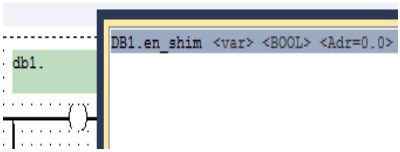


Рис. 2.16. Додавання операнда з блока даних DB1

Подібно до описаного вище заповніть входи та виходи функціонального блока SFC223. На рис. 2.17 зображено результат присвоєння операндів формальним параметрам блока SFC223.

На закінчення заува-

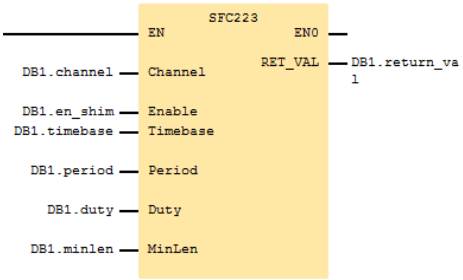


Рис. 2.17. Блок SFC223

жимо, що в даному прикладі було застосовано можливості, які надає використання блоків даних. В середовищі програмування *WinPLC7 V5* такі блоки подібні структурам, які описані в стандарті *IEC61131-3*. Але це не виключає можливості створення складних типів даних, які можуть бути структурами у складі блоків даних.

2.4 Розроблення ППЗ для системи управління водонагрівачем з аналоговим датчиком

Розроблення ППЗ для системи управління водонагрівачем з аналоговим датчиком почнемо з розгляду умов завдання. Схема установки гарячого водопостачання з аналоговим управлінням зображена на рис. 2.18. Відмінність від попередніх схем полягає в використанні аналогового датчика для вимірювання температури – термоперетворювача опору (термометра опору) або термоелектричного перетворювача (термопари). На відмінність від рис. 2.1 на рис. 2.18 замість контактних датчиків температури в установці гарячого водопостачання використаний аналоговий датчик – термометр опору *Pt100*.

Умови роботи системи гарячого водопостачання подібні до умов, які описані в п. 2.1. Але як керуючий пристрій в цьому прикладі буде використаний ПЛК *VIPA System200V*. Тому розглянемо стенд з ПЛК *VIPA System200V* більш докладніше. Схема розташування компонентів на стенді зображена на рис. 2.19.

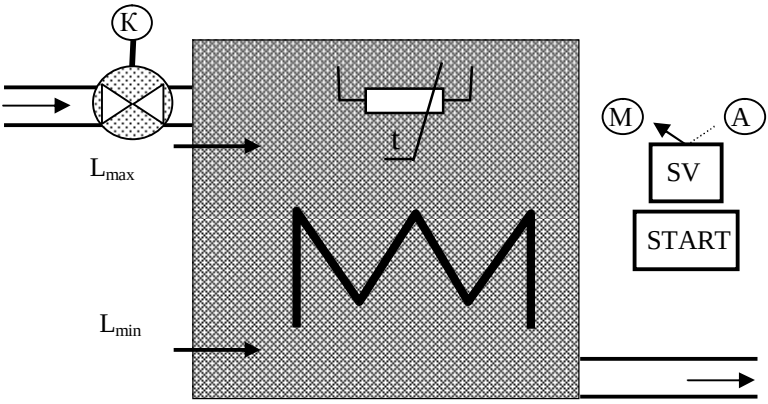


Рис. 2.18. Схема установки гарячого водопостачання
з аналоговим датчиком температури

На рис. 2.19 цифрами позначені компоненти стенда:

1 – перемикач загального живлення стенда;

2 – контролер *VIPA* серії *System 200V* модель CPU214DP (зам. №214-2BP02) і з зовнішнім джерелом живлення *VIPA* типу PS207 (зам. №207-1BA00), який є центральним процесорним модулем. У цьому модулі немає сигнальних субмодулів. Тому, до процесорного модуля підключені таки сигнальні модулі: DI221 (зам. №221-1BF10), DO222 (зам. №222-1BF30), DIO223 (зам. №223-1BF00) та AIO234 (зам. №234-1BD60);

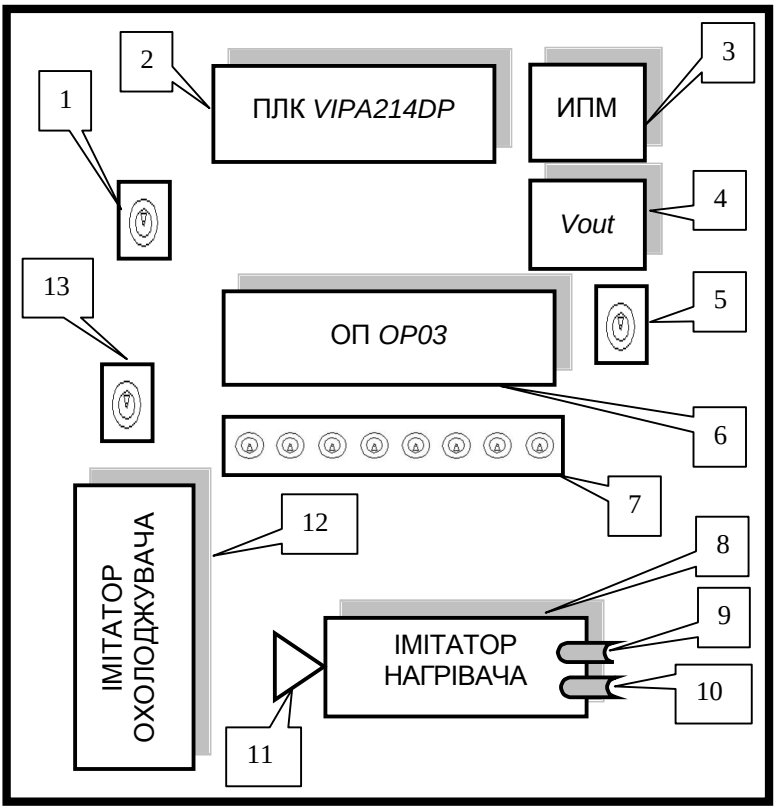


Рис. 2.19. Загальний вигляд стенда *VIPA System 200V*

3 – нормуючий термоперетворювач типу *ИПМ 0196* для перетворення сигналу термоЕДС від термопари типу ТЖК (тип *J*) в уніфікований струмовий сигнал 4-20 *мА*;

4 – індикатор вихідного сигналу напруги постійного струму 0…10 *V*;

5, 6 – перемикач живлення та операторська панель *VIPA OP03*;

7 – імітатори дискретних сигналів, 8 перемикачів типу «сухий контакт»;

8 – імітатор нагрівача, дротовий опір в керамічному корпусі типу ПЕВ-100 номіналом 750 *Ом*;

9 – термопара типу ТЖК (тип *J*);

10 – термоперетворювач опору *Pt100* (з НСХ 1,385);

11 – дискретні датчики температури з NO та NC контактами, які реагують на змінення температури відносно 40 °*C*;

12 – імітатор охолоджувача, вентилятор обдування повітрям постійної напруги 12 *V*.

Схема електричних з'єднань сигнальних модулів введення/виведення ПЛК *VIPA System214V* з датчиками та вихідними пристроями подана в табл. 2.4.

2.4.1. Апаратна складова проекту для VIPA214

Отже запустить середовище програмування *WinPLC7 V5* для контролерів *VIPA* та створіть та збережіть на жорсткому диску новий проект. Далі на підставі завдання та схеми електричних з'єднань проведіть налаштування апаратних ресурсів проекту подібно тому, як це описано в п. 2.1. Але як апаратну платформу виберіть *VIPA214*. Для цього, виходячи із завдання, виберіть ЛКМ рядок *VIPA System 200V* та натисніть ЛКМ на кнопку *Create*. Відкриється вікно *Station-Offline---VIPA214* зі

вкладенням *UR0* з таблицею слотів для заповнення модулями.

Додайте, як це було описано раніше, в перший слот модуль центрального процесору *CPU214DP* (зам. №214-2BP02). Для цього праворуч у вікні з каталогом модулів *VIPA* послідовно ЛКМ відкрийте папки «*VIPA 100V/200V*», «*CPU*», «*CPU 21X*», «*CPU 214*», «*CPU 214 DP Slave*». Подвійним кліком ЛКМ додайте процесорний модуль *CPU 214-2BP02* до першого слота у вкладення *UR0*.

Табл. 2.4 – Електричні з'єднання ПЛК *VIPA214* з датчиками та виконавчими пристроями

Сигнальні модулі ПЛК *VIPA System214V*							
SM221 DI		SM223 DIO		SM222 DO		SM234 AIO	
Абс. адреса	Датч.	Абс. адреса	Датчик/ Вих. пр.	Абс. адреса	Вих. пр.	Абс. адреса	Датчик/ Вих. пр.
I0.0	SV1	I1/Q0.0	вхід NO T/S	Q0.0	-	Ch0 PIW256	вхід 4…20mA
I0.1	SV2	I1/Q0.1	вхід NC T/S	Q1.1	-	Ch1 PIW258	вхід 4…20mA
I0.2	SV3	I1/Q0.2	-	Q1.2	-	Ch2 PIW260	-
I0.3	SV4	I1/Q0.3	-	Q1.3	-		
I0.4	SV5	I1/Q0.0	-	Q1.4	-	Ch3 PIW262	вхід Pt100
I0.5	SV6	I1/Q0.1	-	Q1.5	-		
I0.6	SV7	I1/Q0.2	вихід cooler	Q1.6	-	Ch4 PQW256	вихід 0…10V
I0.7	SV8	I1/Q0.3	вихід heater	Q1.7	-	Ch5 PQW258	вихід 4…20mA

Для налаштування властивостей модуля подвійним кліком ЛКМ по рядку *CPU-Properties* відкрийте відповідне вікно. В результаті з'явиться вікно налаштування модуля, подібне до поданого на рис. 2.4. У нижній частині вікна налаштувань розміщена кнопка для *Special*

CPU-Properties, яка відкриває діалогове вікно для налаштувань спеціальних властивостей процесорного модуля. Це налаштування властивостей ПЛК для роботи в мережі *PROFIBUS* в якості веденого пристрою (*Slave*). Для вирішення поточного завдання зараз це не потрібно. Проте процесорний модуль потрібно доповнити сигнальними модулями *DI221* (зам. №221-1BF10), *DIO223* (зам. №223-1BF00), *DO222* (зам. №222-1BF30) та *AIO234* (зам. №234-1BD60). Для цього потрібно їх додати до конфігурації ПЛК із переліку сигнальних модулів *VIPA-SM-200V*. В результаті буде отримана апаратна конфігурація ПЛК, зображення якої подано на рис. 2.20. Сигнальні модулі дискретних входів та виходів не потребують додаткового налаштування. Проте, для вирішення завдання потрібно налаштувати параметри модуля аналогового введення/виведення для оброблення сигналу від термоперетворювача опору (*Pt100*) або термоелектричного перетворювача (термопара типу ТЖК).

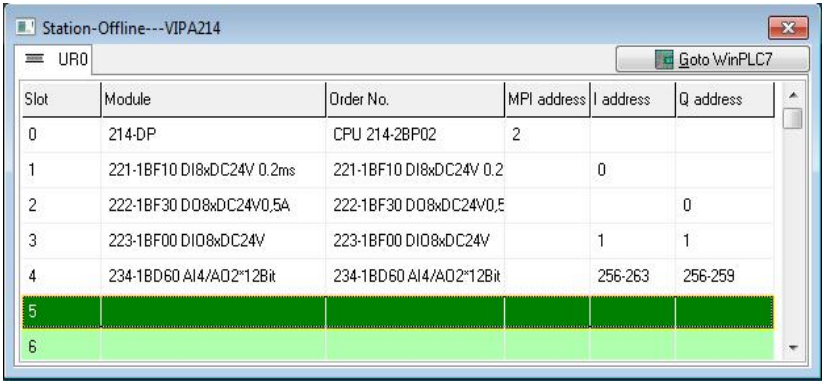


Рис. 2.20. Конфігурація ПЛК на стенді *VIPA System 200V*

Натисніть ЛКМ на рядок з модулем аналогового введення/виведення *AIO234*. Далі у вікні *Module-Parameters* оберіть параметр *channel 0:function*, а в переліку *Selection – S7 Current 4…20 mA*, як це показано на рис. 2.21.

На рис. 2.22 зображено вікно з налаштуваннями четвертого каналу *channel 3:function* з переліком доступних варіантів. Цей канал призначений для підключення резистивних датчиків, зокрема, термоперетворювачів опору (*Pt100*, *Pt1000*, *Ni100*, *Ni1000*) та змінних резисторів (до 600 та 3000 Ом). Згідно табл. 2.4 виберіть тип датчика *Pt100 2-wire -200°C…850°C*.

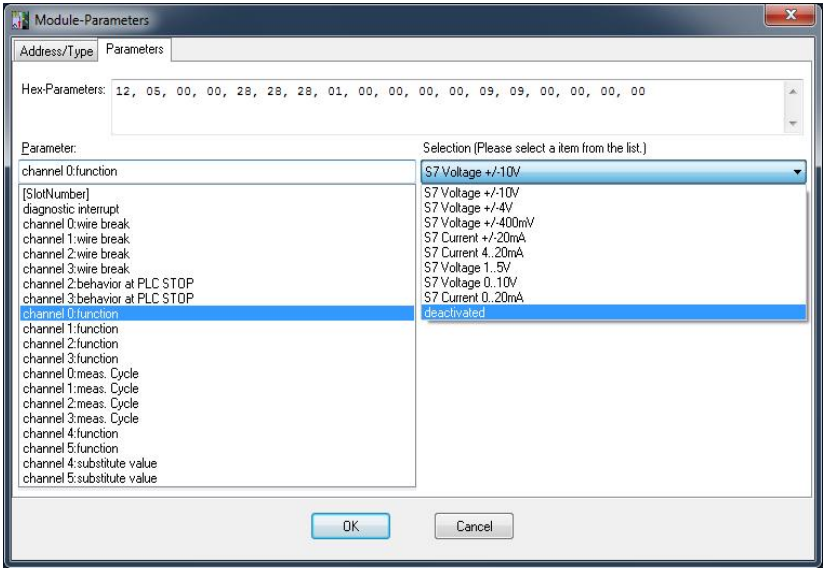


Рис. 2.21. Налаштування вхідного каналу *Ch0* модуля *AIO234*

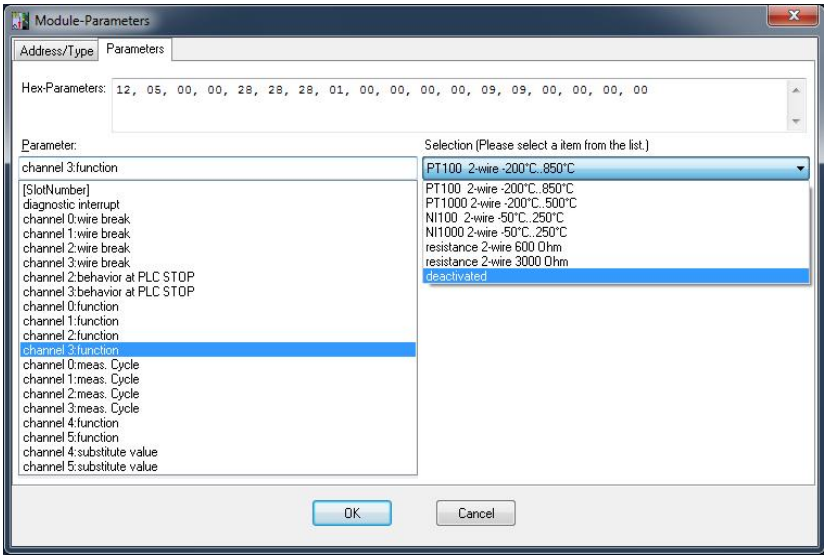


Рис. 2.22. Налаштування вихідного каналу *Ch3* модуля *AIO234*

Наступним кроком додамо конфігурацію апаратних ресурсів ПЛК до проекту. Для цього зробимо подібно тому, як це описано раніше. Тобто необхідно ЛКМ вибрати перелік *File/Project* з основного меню програми-конфігуратора, а в ньому команду – *Save the active Station in the WinPLC7-Project*. Далі підтвердить пересилання конфігурації ПЛК до проекту натисненням ЛКМ на кнопку *Да*.

На цьому можна вважати апаратну конфігурацію ПЛК закінченою.

2.4.2. Програмна складова проекту для VIPA214

Отже, можна почати розробляти програму користувача. З огляду на умови завдання (див. рис. 2.18) використаємо деякі результати, які викладені у п. 2.1. Програма користувача зображена на рис. 2.23.

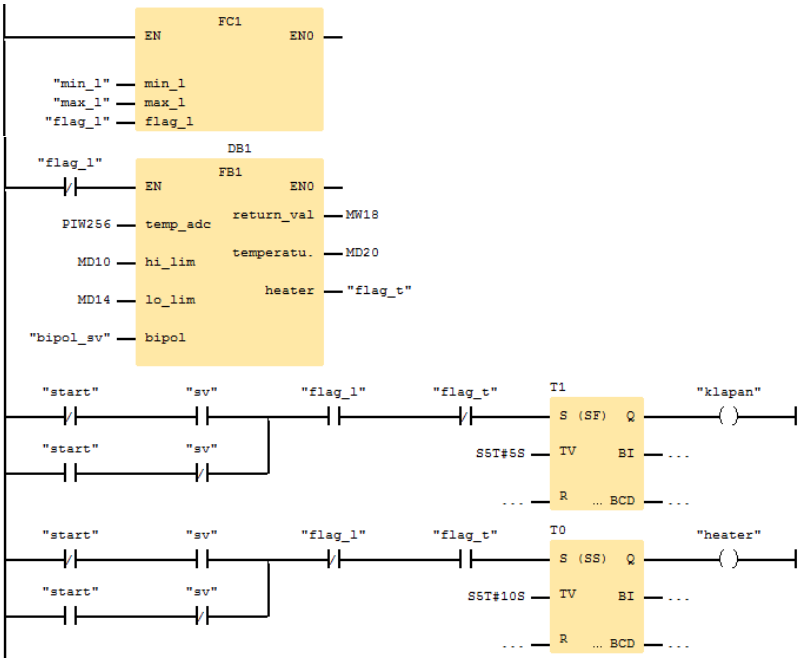


Рис. 2.23. Програма користувача для установки гарячого водопостачання з аналоговим датчиком

Надамо деякі пояснення щодо програми користувача. В програмі використані нові структурні компоненти програми користувача. Це – функція (*FC1*) та функціональний блок (*FB1*) з блоком даних (*DB1*).

Функція призначена для аналізу стану датчиків рівня. Для створення функції виберіть у вікні *Edit project:* вкладення *Solution*, в ньому послідовно перелікі *Blocks* та *FC* та ЛКМ натисніть у переліку *FC* на команду * *Creat new.* Відкриється вікно редактора функції. Заповніть поле об'явлення змінних функції та власне функцію ланцюгом для аналізу стану датчиків рівня, як це зображено на рис. 2.24 та рис. 2.25.

*	Address	Declaration	Name	Type	Initial value
	0.0	in -->	min_l	BOOL	
	0.1	in -->	max_l	BOOL	
		out <--			
	2.0	in_out <-->	flag_l	BOOL	
		temp T			

Рис. 2.24. Інтерфейс функції FC1

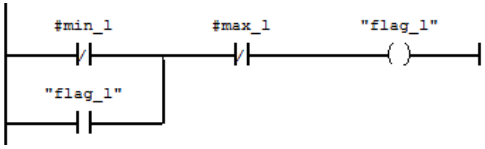


Рис. 2.25. Вміст функції FC1

Функціональний блок додається в програму користувача подібно до додавання функції, але до переліку *FB* у менеджері проекту *Solution*. Цей блок призначений для реалізації позиційного регулятора. В блоці перший ланцюг перетворює кодове число після АЦП в модулі аналого-

вого введення у фізичний параметр – температуру. Далі поточне значення температури порівнюється із завданням. Інтерфейс блоку *FB1* зображено на рис. 2.26, а вміст блоку – на рис. 2.27. Зауважимо, що інтерфейс блока даних *DB1* співпадає з інтерфейсом функціонального блока *FB1*.

* Address	Declaration	Name	Type	Initial value
0.0	in -->	temp_adc	INT	0
2.0	in -->	hi_lim	REAL	8.500000e+02
6.0	in -->	lo_lim	REAL	-1.500000e+02
10.0	in -->	bipol	BOOL	TRUE
12.0	out <--	return_val	WORD	W#16#0000
14.0	out <--	temperatura	REAL	0.000000e+00
18.0	out <--	heater	BOOL	FALSE
	in_out <-->			
20.0	var S	max_temp	REAL	5.000000e+01
24.0	var S	min_temp	REAL	4.500000e+01
	temp T			

Рис. 2.26. Інтерфейс блока FB1

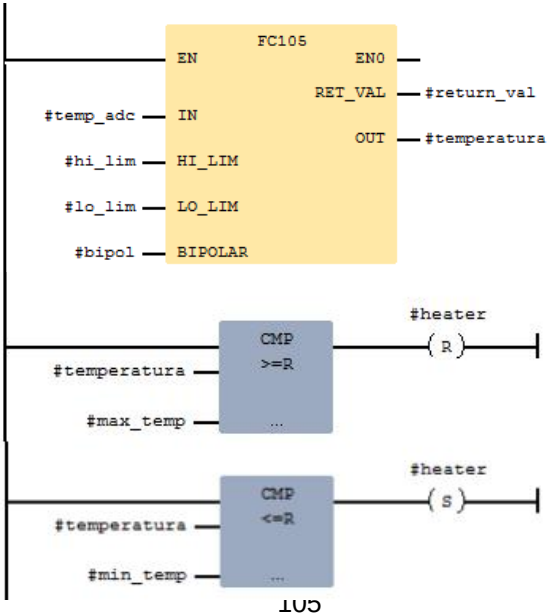


Рис. 2.27. Вміст блока FB1

На завершення відмітимо, що в програмі користувача формальні параметри функціонального блока та функції позначені символом «#».

2.5 Розроблення ПІД-регулятора температури для системи гарячого водопостачання

Розроблення ППЗ для системи гарячого водопостачання почнемо з розгляду умов завдання. Схема системи гарячого водопостачання з аналоговим управлінням зображена на рис. 2.28. Умови роботи системи подібні до умов, які описані в п. 2.1. Проте регулювання температурою рідини в ємності здійснюється за ПІД-законом з використанням ШІМ-сигналу для керування дискретним вихідним пристроєм – нагрівачем.

Керуючим пристроєм в цьому прикладі буде використаний ПЛК *VIPA System300S*, властивості якого були розглянуті у першому розділі. Тому розглянемо більш докладніше стенд з ПЛК *VIPA System300S*. Схема розташування компонентів на стенді зображена на рис. 2.29.

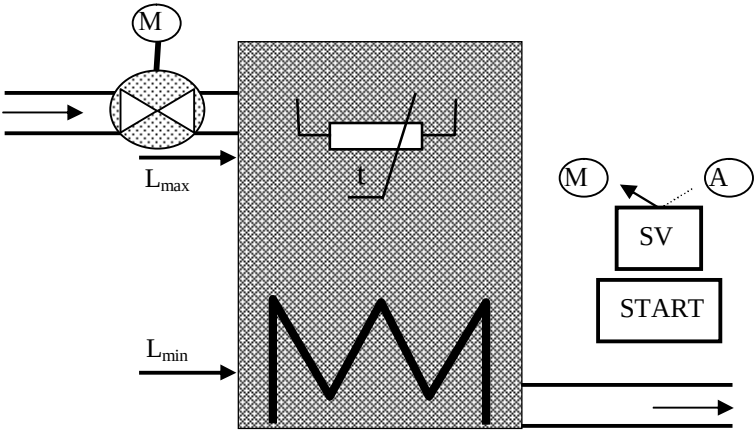


Рис. 2.28. Схема установки гарячого водопостачання з аналоговим датчиком температури

На рис. 2.29 цифрами позначені такі компоненти стенда:

1 – Контролер *VIPA* серії *System 300S* модель *CPU313SC* (зам. №313-5BF03-0AB0), який є центральним процесорним модулем та зовнішній блок живлення типу *VIPA PS307* (зам. №307-1BA00). До процесорного модуля входять сигнальні субмодулі: *VIPA DI24/DO16* та *VIPA AI5/AO2*;

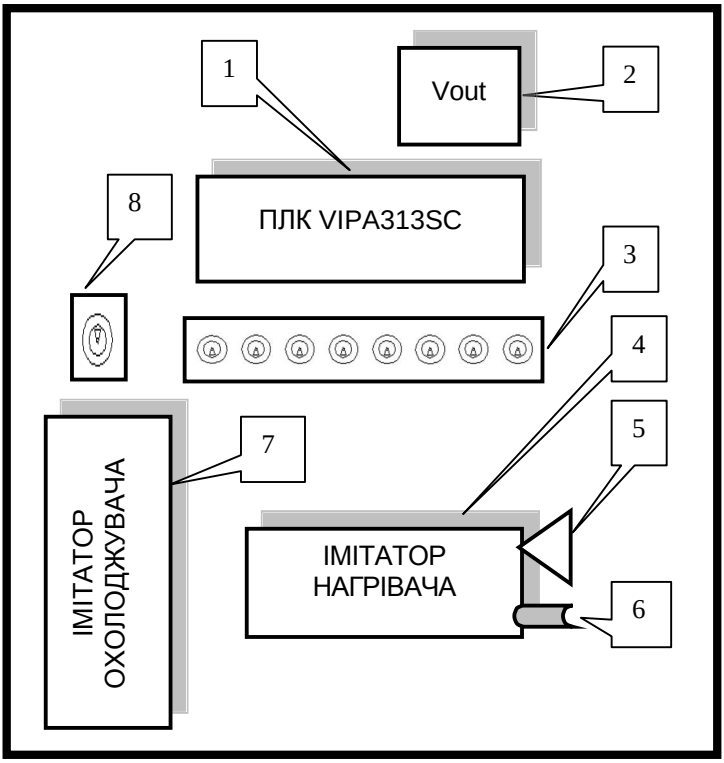


Рис. 2.29. Загальний вигляд стенда *VIPA System 313SC*

2 – індикатор вихідного сигналу напруги постійного струму 0…10 V;

3 – імітатори дискретних сигналів, 8 перемикачів типу «сухій контакт»;

4 – імітатор нагрівача, дротовий опір в керамічному корпусі типу ПЕВ-100 номіналом 750 Ом;

5 – дискретні датчики температури з NO та NC контактами на 40 °С;

6 – термоперетворювач опору Pt100 (з НСХ 1.385);

7 – імітатор охолоджувача, вентилятор обдування повітрям з живленням постійної напруги 12 В.

8 – перемикач живлення імітатора нагрівача.

Схема електричних з'єднань сигнальних субмодулів ПЛК *VIPA System313SC* з датчиками та вихідними пристроями наведена в табл. 2.5. В таблиці вказані абсолютні адреси аналогових та дискретних каналів та елементи стенда, які до цих каналів підключені.

Табл. 2.5 – Електричні з'єднання ПЛК *VIPA313SC* з датчиками та вихідними пристроями

<table>
<tr><td colspan="8">Сигнальні субмодулі ПЛК VIPA313SC</td></tr>
<tr><td colspan="4">X11</td><td colspan="4">X12</td></tr>
<tr><td>Абс. адреса</td><td>Датчик</td><td>Абс. адреса</td><td>Дат-чик</td><td>Абс. адреса</td><td>Дат-чик</td><td>Абс. адреса</td><td>Вих. прист.</td></tr>
<tr><td>Ch0
PIW752</td><td>вхід
4…20mA</td><td rowspan="8">PIW126
входи</td><td>-</td><td rowspan="8">PIW124
входи</td><td>SV1</td><td rowspan="8">PQW124
виходи</td><td>-</td></tr>
<tr><td></td><td></td><td>-</td><td></td><td>SV2</td><td></td><td>-</td></tr>
<tr><td>Ch1
PIW754</td><td>вхід
-</td><td>-</td><td></td><td>SV3</td><td></td><td>-</td></tr>
<tr><td></td><td></td><td>-</td><td></td><td>SV4</td><td></td><td>-</td></tr>
<tr><td>Ch2
PIW756</td><td>вхід
-</td><td>-</td><td></td><td>SV5</td><td></td><td>-</td></tr>
<tr><td></td><td></td><td>-</td><td></td><td>SV6</td><td></td><td>-</td></tr>
<tr><td>Ch3
PIW758</td><td>вхід
-</td><td>-</td><td></td><td>SV7</td><td></td><td>-</td></tr>
<tr><td></td><td></td><td>-</td><td></td><td>SV8</td><td></td><td>-</td></tr>
<tr><td>Ch4
PIW760</td><td>вхід
Pt100</td><td rowspan="3"></td><td rowspan="3"></td><td rowspan="3">PIW125</td><td>NO</td><td rowspan="3">PQW125</td><td>heater</td></tr>
<tr><td></td><td></td><td>NC</td><td></td><td>cooler</td></tr>
<tr><td>Ch5</td><td>вихід</td><td>-</td><td>-</td></tr>
</table>

PQW752	4…20mA				-		-
Ch6	вихід				-		-
PQW754	0…10V				-		-
					-		-
					-		-

Згідно умов завдання розробимо структурну схему регулятора, яка поясніть наступні рішення щодо розроблення апаратної та програмної складової проекту. На рис. 2.30 зображена структура аналогового регулятора з дискретним виконавчим пристроєм (нагрівачем). Це так звана автоматична система регулювання (АСР) температурою.

Програмний ПІД-регулятор з аналоговим та дискретним управлінням в середовищі *WinPLC7 V5* реалізується за допомогою функціональних блоків, які входять до бібліотеки стандартних компонентів. У переліку *Sfb_Sfc*, яке входить до *Standard library* є декілька типів ПІД-регуляторів. Це безперервний (SFB41 CONT_C) та ступінчатий (SFB42 CONT_S) регулятори. Додатково в бібліотеці є функціональний блоки для керування виконавчими механізмами різних типів. Наприклад, блок SFB43 PULSEGEN призначений для формування ШІМ-сигналу для керування реверсивним електродвигуном за принципом «Більше-Меньше». Це може бути виконавчий механізм з електричним двигуном, який має дві обмотки для обертання в обох напрямах, наприклад типу *МЭО* або *МЭП*. Також в бібліотеці є блоки керування аналоговим та дискретним виконавчими механізмами, які працюють сумісно з регуляторами. Для регулювання специфічних параметрів у бібліотеці є вимірювачі кількості імпульсів та частоти.

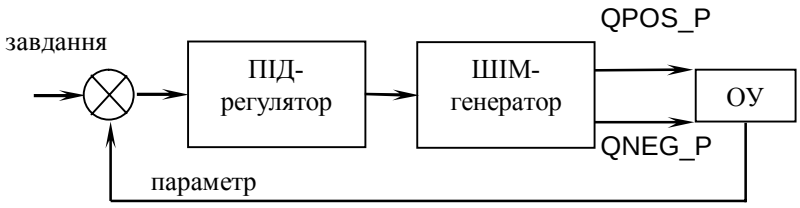


Рис. 2.30. Структурна схема АСР температурою

При роботі з функціональним блоком ПІД-регулятора є можливість залежно від вимог технологічного процесу активувати або блокувати його окремі функції. Взагалі ПІД-регулятор працює як пропорційний регулятор, де інтегральна і диференціальна складові підключаються паралельно і можуть бути ввімкнутими або вимкнутими кожна окремо. Таким чином можна реалізовувати П, ПІ, ПД та ПІД закони регулювання. Функціональні блоки можуть використовуватись для створення каскадного регулятор зі встановленим заданим значенням. В блоках регуляторів передбачено ручний режим керування для змінювання значення вихідної величини регулятора (управляюча дія). Для регулювання двох- та трьохпозиційним приводом вихідний аналоговий сигнал регулятора може оброблятись з використанням широтно-імпульсною модуляцією. Поточне значення технологічного параметру подається на регулятор в одному з двох форматів: або, як цілочисельне слово, яке безпосередньо йде від сигнального модуля аналогових входів (АЦП); або, як слово з плаваючою крапкою, після перетворення у цей формат (див. рис.2.27 з функцією *FC105* для перетворення коду від АЦП у реальне значення аналогового параметру). Управляюча дія від регулятора може бути отримана також в одному з двох форматів: або у вигляді цілого числа типу *WORD*, яка безпосередньо подається на аналоговий вихід сигнального модуля; або у вигляді числа з плаваючою крапкою. Але перед відправленням на модуль аналогових виходів дане значення необхідно перетворити у стандартний формат (за допомогою функції *FC106* для перетворення аналогового значення параметру у код ЦАП). В блоці регулятора є можливість вибору формату даних. При використанні поточного значення і управляючої дії у форматі числа з плаваючою крапкою точність регулювання підвищується. В регуляторі реалі-

зовані додаткові функції:

- обмеження вхідного та вихідного параметру за мінімальним та максимальним значеннями;
- нормалізація вхідного і вихідного параметрів (кут нахилу та зсув характеристик).

Розглянемо алгоритм реалізації ПІД-регулятора у функціональних блоках на прикладі використання безперервного регулятора (SFB41 CONT_C), який зображений на рис. 2.31 [10].

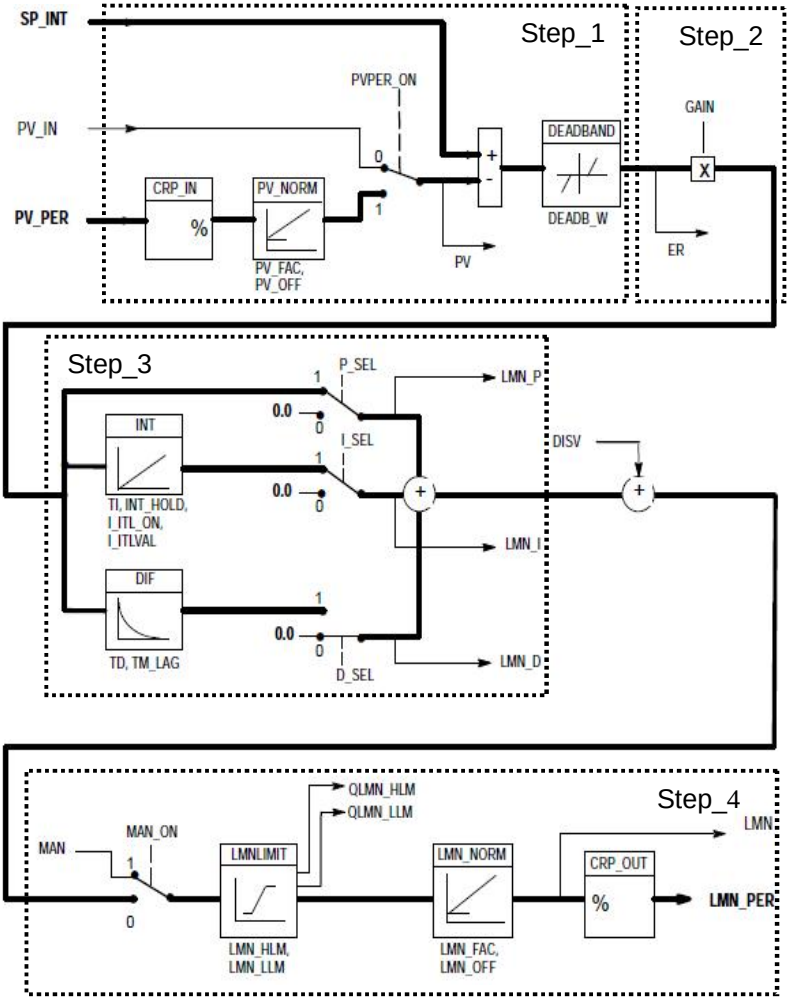


Рис. 2.31. Структурна схема ПІД-регулятора

Алгоритм регулятора складається з декількох основних кроків, які подані нижче:

- попереднє оброблення значення технологічного параметра (*Step_1*) для обчислення сигналу помилки (відхилення);
- обчислення пропорційної складової (*Step_2*) регулюючої дії;
- паралельне обчислення інтегральної та диференціальної складової (*Step_3*) регулюючої дії;
- керування у ручному режимі (*Step_4*).

Надамо пояснення щодо позначень на рис. 2.31. Вхідний параметр подається або як числове значення у форматі числа з плаваючою крапкою PV_IN після перетворення у блоці *FC105* або як цілочисельне значення PV_PER від АЦП периферійного пристрою. Тип вхідного параметра перемикається за допомогою параметру PVPER_ON.

Перерахунок цілочисельного значення у число з плаваючою крапкою здійснюється за формулою:

$$OUT\_CRP\_IN=PV\_PER*100/27648. \quad (2.1)$$

Значення OUT_CRP_IN нормується за формулою:

$$OUT\_NORM=OUT\_CRP\_IN*PV\_FAC+PV\_OFF. \quad (2.2)$$

За умовчанням коефіцієнт PV_FAC дорівнює одиниці, а зсув PV_OFF – нулю.

Далі вхідне значення PV_IN порівнюється із завданням SP_INT для обчислення похибки ER. Якщо помилка становить значення, яке менш ніж пороги чутливості у блоці DEADBAND, то подальший розрахунок управляючої дії припиняється до наступного циклу сканування ПЛК. В протилежному випадку значення відхилення ER множиться на коефіцієнт пропорційності GAIN для отримання пропорційної складової LMN_P. Якщо значення параметру DEADB_W дорівнює нулю, то функція налаштування чутливості регулятора вимкнена. Значення пропорційної складової враховується під час обчислення інтегральної LMN_I та диференціальної LMN_D складових. При цьому вибір типу закону регулювання (П, ПІ, ПД або ПІД) визначається відповідними перемикачами P_SEL, I_SEL та D_SEL. До розрахованого значення сигналу керування додається значення параметра збурення DISV.

На кроці *Step_4* реалізовано перемикання на ручний режим за допомогою перемикача MAN_ON. Якщо регулятор знаходиться у ручному режимі керування, то вихідне значення регулюючої дії (LMN та LMN_PER) отримує від вхідного значення параметра MAN. Для обох режимів роботи регулятора управляюча дія обмежується мінімальним QLMN_LLM і максимальним значенням QLMN_HLM та нормалізується за допомогою коефіцієнта множення LMN_FAC та параметра зсування LMN_OFF. Крім того, якщо керування здійснюється за допомогою сигналу у цілочисельному значенні, то його визначають у відповідному блоці. Насамкінець зауважимо, що формули обмеження та нормування сигналу управляючої дії подібні до формул 2.1 та 2.2 з відповідними параметрами.

Розглянемо алгоритм реалізації ступінчатого ПІД-регулятора, який зображений на рис. 2.32 [10]. Алгоритм регулятора складається з декількох основних кроків, які подані нижче:

- попереднє оброблення значення технологічного параметра (*Step_1*) для обчислення сигналу помилки (відхилення);

- обчислення пропорційної складової (*Step_2*) регулюючої дії;
- обчислення інтегральної та диференціальної складової регулюючої дії та вибір режиму ручного керування (*Step_3*).

Надамо пояснення щодо позначень на рис. 2.32. Перший та другий кроки алгоритму співпадають з відповідними кроками алгоритму безперервного ПІД-регулятора (див. рис.2.31). Відрізняється лише третій крок. Отже, після обчислення пропорційної складової на другому кроці та врахування збурення DISV на 3-му кроці сигнал зменшується на значення інтегральної складової. Ця складова обчислюється з врахуванням часу роботи двигуна між крайніми положеннями MTR_TM.

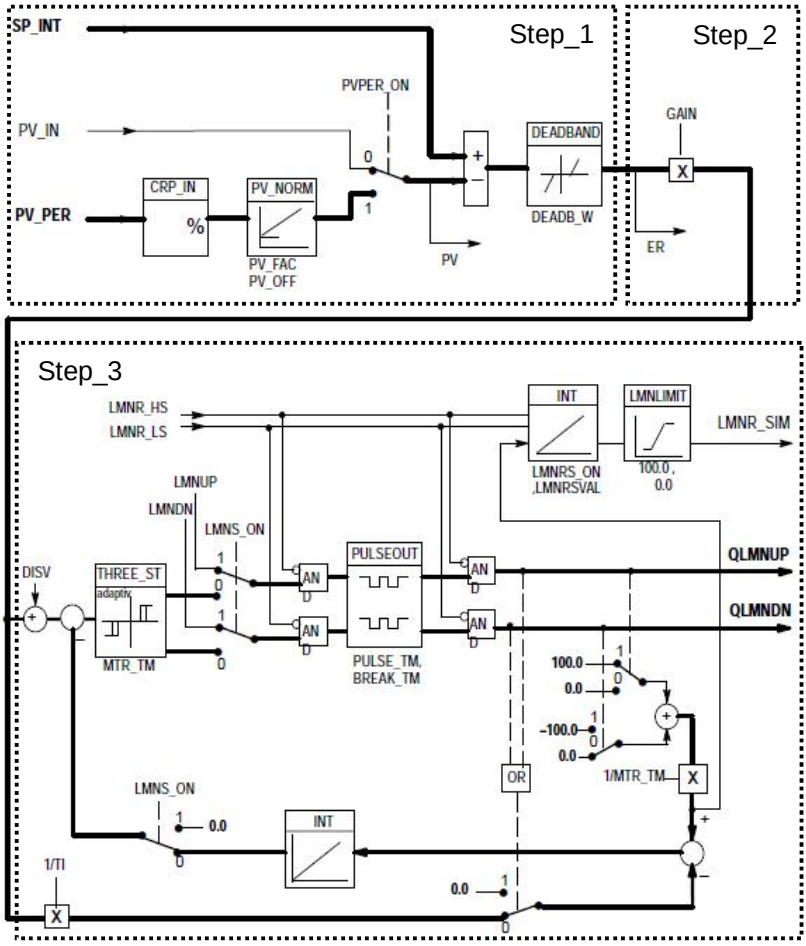


Рис. 2.32. Структурна схема ступінчатого ПІД-регулятора

Сигнал керування обробляється в 3-ступеневому адаптивному блоці THREE_ST (*adaptive*), де формуються керуючи імпульси «більше» або «менше». Далі, в залежності від режиму роботи регулятора, ручний або автоматичний (в залежності від стану LMNS_ON), вихідний імпульсний сигнал від блока THREE_ST обробляється сумісно з сигналами від датчиків досягнення крайніх положень за допомогою логічних операторів *AND*. Це потрібно для блокування роботи електродвигуна, якщо досягнуто крайні положення регулюючого органу. В блоці PULSEOUT формуються імпульси сигналів керування електродвигуном, які теж обробляються сумісно з сигналами від датчиків досягнення крайніх положень за допомогою логічних операторів *AND*. В алгоритмі передбачено також зворотній зв'язок для врахування інтегральної складової, яка обчислюється за допомогою параметрів MTR_TM та TI.

2.5.1. Апаратне конфігурування ПІД-регулятора для VIPA313SC

Отже запустить середовище програмування контролерів *VIPA* та створіть новий проект. Далі на підставі завдання та схеми електричних з'єднань проведіть налаштування апаратних ресурсів контролера. Для цього подвійним кліком ЛКМ у вкладенні *Solution* дерева проекту потрібно вибрати ресурс *Hardware stations*. Далі подвійним кліком ЛКМ активувати команду *Create new*. В діалоговому вікні *Create new station:*, яке відкриється, введіть ім'я станції в поле *Name of station:*, наприклад *VIPA313*. В результаті відкриється вікно програми-конфігуратора апа-

ратних ресурсів з вікном *Select PLC-system* для вибору апаратної платформи *Station-Offline---VIPA313*.

Усього кожна рейка має 11-ть посадочних місць, але це не означає, що вони усі можуть бути заповненими. Існують деякі обмеження для апаратного конфігурування процесорів серії *System 300S*. По-перше, перше посадочне місце зарезервоване для встановлення модуля живлення. Крім того, в залежності від моделі процесорного модуля деякі слоти будуть автоматично заповнені сигнальними та функціональними субмодулями. Наявність субмодулів визначається властивостями процесорного модуля та можливістю реалізації інтерфейсних та комунікаційних функцій. Друге місце зарезервоване виключно для встановлення процесорного модуля. На третьому посадочному місці знаходиться інтерфейсний модуль *IM360*, який необхідний для з'єднання рейок *UR0* та *UR1*, а модуль *IM361* – для з'єднання рейок від *UR1* до *UR3*. На рейці *UR3* одинадцятий слот зарезервований для так званої *SPEED*-шини. Зрозуміло, що дані обмеження зменшують кількість можливих для встановлення сигнальних та функціональних модулів до 8-і модулів на рейкі *UR0*, *UR1* та *UR2* (з 4-го по 11 місце). На рейку *UR3* можливе встановлення лише до 7-і модулів (з 4-го по 10 місце). Це пов'язано з наявністю *SPEED*-шини. Таким чином повна конфігурація станції серії *VIPA System 300S* вміщує 32 модуля з процесорним модулем включно. Це відповідає властивостям процесорних модулів щодо кількості периферійних модулів. На стандартну шину можуть бути встановлені модулі з папок *DI-300*, *DO-300*, *DI/DO-300*, *AI-300*, *AI/AO-300* та *AO-300*.

Виходячи із завдання, виберіть ЛКМ рядок *VIPA System 300S* та натисніть ЛКМ на кнопку *Create*. Відкриється вікно *Station-Offline---VIPA313* з активованим вкладенням *UR0* з таблицею слотів для заповнення модулями. Додайте процесорний модуль з переліку «*S7-300*», «*CPU Speed7*», «*CPU 313SC*». Подвійним кліком ЛКМ додайте процесорний модуль 6ES7 313-5BF03-0AB0 до другого слота у рейці *UR0*.

Це місце зарезервоване виключно для встановлення процесорного модуля. В результаті до вікна буде додано центральний процесорний модуль та віртуальні сигнальні та функціональні субмодулі, як це зображено на рис. 2.33.

Якщо подвійним кліком ЛКМ відкрити процесорний модуль *CPU 313SC SPEED7* для редагування його параметрів, то відкриється вікно зі вкладеннями, яке подібне до вікна, зображеного на рис. 2.4. Проте замість кнопки «*Special CPU-Properties*» буде кнопка «*Ethernet CP-Properties (PG/OP-channel)*», за допомогою якої налаштовуються властивості *Ethernet*-канала для завантаження програм (з'єднання *PG*) та обміну даними (з'єднання *OP*) по мережі *Ethernet*. Проведемо налаштування *Ethernet*-з'єднання. Для цього натисніть ЛКМ на вказану кнопку. У відкритому вікні *Properties CP343* виберіть вкладення *Common Options*. Далі натисніть ЛКМ на кнопку *Properties Ethernet*. Заповніть поля *IP-Address* та *Subnet mask* вікна *Properties Ethernet-Interfaces (*вкладення *Parameters)*, наприклад так, як це зображено на рис. 2.34.

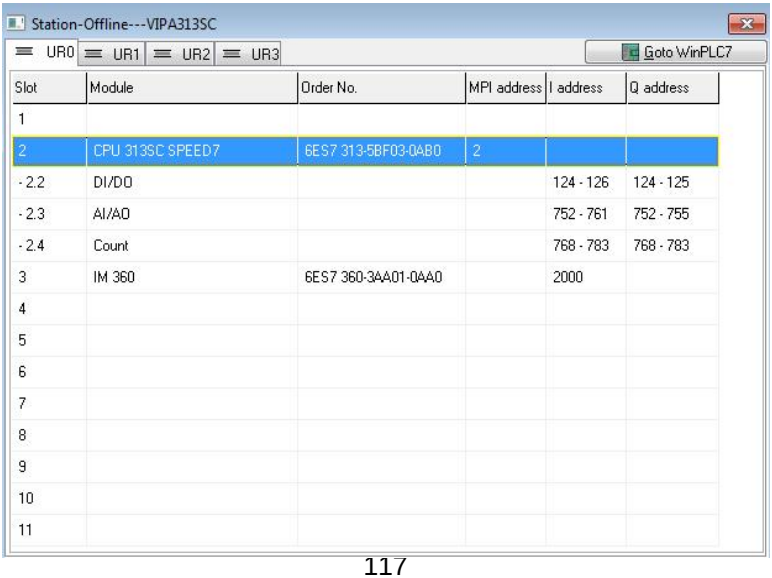


Рис. 2.33. Конфігурація модуля *VIPA System 313SC*

Якщо в локальній мережі *Ethernet* використаний маршрутизатор, то поставте ЛКМ прапорець напроти рядка «*Use router*» та додайте його *IP-Address* у відповідне поле. Зауважимо, що основні властивості віртуального модуля *CP343* є такими:

- підтримка протоколів *TCP/IP* з використанням програмних інтерфейсів обміну «відправлення-приймання;
- підтримка протоколів *UDP* та *S7-communication*;
- можливість маршрутизації з фіксованою *MAC*-адресою без використання програматора.

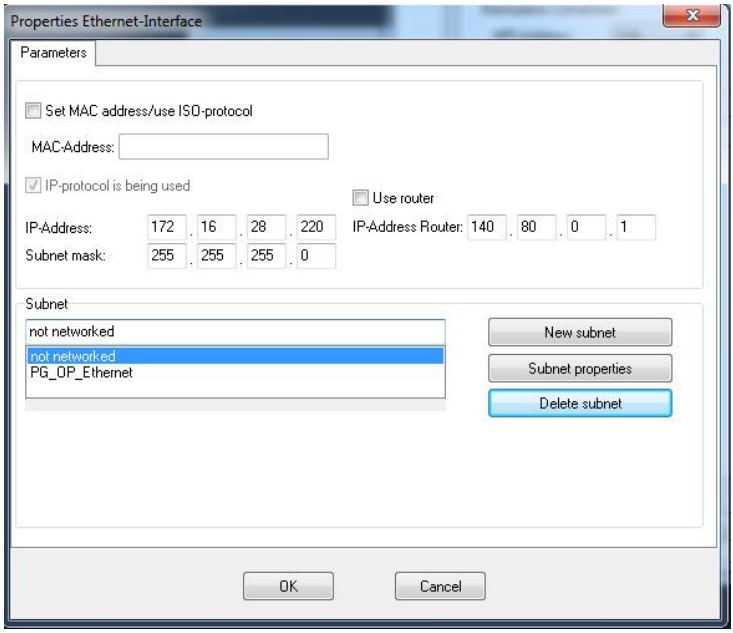


Рис. 2.34. Налаштування *Ethernet*-з'єднання *CPU VIPA313SC*

Вікно з налаштування властивостей модуля також доступно у вкладенні *UR3*, де включений віртуальний субмодуль *Speed7 Ethernet (CP343)*.

Як зображено на рис. 2.32 до складу процесорного модуля входять три субмодуля:

- у рядку слота 2.2 – субмодуль дискретних каналів введення/виведення;
- у рядку слота 2.3 – субмодуль аналогових каналів введення/виведення;
- у рядку слота 2.4 – субмодуль для налаштування апаратних лічильників та ШІМ-каналів.

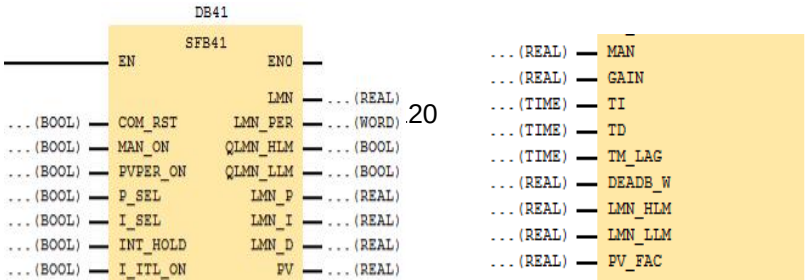
Будова дискретних субмодулів введення/виведення подібна будові молодших версій сигнальних субмодулів *VIPA* (див. рис. 1.8, 1.10 та 1.12), які розглянуті у першому розділі. Тому розглянемо лише субмодуль аналогового введення/виведення, який має деякі відмінності від модуля *SM234* (див. рис.1.19 та табл. 1.5). У рядку субмодуля вказані адреси аналогових каналів (для входів – 752…761, для виходів – 752…755) по два байти на кожний канал, усього 10 байтів пам'яті для аналогових входів та 4 байта пам'яті для аналогових виходів. Відкрийте ЛКМ цей субмодуль для налаштувань аналогових каналів та перейдіть у вікні *Properties AI/AO 300* до вкладення *Inputs*. В цьому вкладенні здійснюється налаштування аналогових вхідних каналів: чотири – для активних сигналів струму або напруги та один – для пасивного сигналу опору. Налаштуйте канали згідно з табл. 2.5. В каналі для пасивного сигналу опору виберіть тип підключення вихідного сигналу датчика – *RTD-2L*, а тип датчику – *Pt100Std*. Нагадаємо, що дискретним виходом, до якого підключений нагрівач буде керуватись з боку регулятора ШІМ-сигналом. Тому потрібно налаштувати відповідний вихід на роботу в режимі ШІМ. Для цього відкрийте на редагування субмодуль з 3-го рядку, тобто субмодуль налаштування апаратних дискретних входів та виходів.

Інші канали залишить без змін. Закрийте вікно натиснувши ЛКМ на кнопку *OK* та передайте конфігурацію до проекту, як це було описано раніше.

2.5.2. Програмна складова ПІД-регулятора для VIPA313SC

Перед розробленням програми користувача з ПІД-регулятором розглянемо можливості середовища *WinPLC V5* щодо реалізації регулювання. Так в бібліотеці стандартних ФБ середовища є блоки безперервного *CONT_C* (SFB41) та ступінчатого *CONT_S* (SFB42) ПІД-регуляторів. Різниця між регуляторами полягає у типі управляючого сигналу: у безперервного – це лінійний аналоговий сигнал, у ступінчатого – дискретний. Для роботи обох блоків потрібно до проекту з організаційним блоком OB1, в якому реалізований основний цикл сканування контролера треба додати ще два організаційні блока. Це блоки OB100 та OB35, які відповідають за «холодний» перезапуск програми користувача (одноразово) та за перезапуск регулятора (постійно з циклом сканування). Час системного переривання блока OB35 за умовчанням становіть 100 *мс*. Виконавчим пристроєм у прикладі буде електропідігрівач, який буде живиться змінною напругою (див. поз. 4 на рис. 2.29 та табл. 2.5). Для керування підігрівачем буде використаний блок *PULSEGEN* (SFB43). Цей приклад надає можливість зрозуміти принцип дії та порядок конфігурування регуляторів з дискретним вихідним пристроєм типу «нагрівач».

Розглянемо інтерфейс безперервного регулятора *CONT_C* (SFB41), зображення якого подано на рис. 2.35. Інтерфейс блока відповідає опису алгоритму регулятора, який зображений на рис. 2.31 та описаний раніше. Блок регулятора SFB41 отримує значення формальних параметрів від блока даних DB41.



В табл. 2.6 наведено перелік формальних параметрів стандартного блока SFB41, за допомогою якого реалізований безперервний ПІД-регулятор.

Таблиця 2.6 – Параметри блоку ПІД-регулятора *CONT_C* (SFB41)

Напрям сигналу	Ім’я параметра	Тип параметра	Опис параметра
in	COM_RST	BOOL	повний перезапуск
in	MAN_ON	BOOL	вмикання ручного керування
in	PVPER_ON	BOOL	вмикання периферійних змінних процесу
in	P_SEL	BOOL	вмикання пропорційної складової
in	I_SEL	BOOL	вмикання інтегральної складової
in	INT_HOLD	BOOL	запам’ятати інтегральну складову
in	I_ITL_ON	BOOL	ініціювати інтегральну складову
in	D_SEL	BOOL	вмикання диференційної складової
in	CYCLE	TIME	час циклу обчислення

in	SP_INT	REAL	внутрішнє завдання
in	PV_IN	REAL	вхід змінної процесу
in	PV_PER	WORD	периферійна змінна процесу
in	MAN	REAL	значення параметру керування у ручному режимі
in	GAIN	REAL	коєф. підсилення регулятора
in	TI	TIME	інтервал інтегрування
in	TD	TIME	інтервал диференціювання

Продовження табл. 2.6

in	TM_LAG	TIME	час затримки диф. складової
in	DEADB_W	REAL	зона нечутливості регулятора
in	LMN_HLM	REAL	сигнал досягнення верхньої межі керуючого сигналу
in	LMN_LLM	REAL	сигнал досягнення нижньої межі керуючого сигналу
in	PV_FAC	REAL	коеф. пропор. змінної процесу
in	PV_OFF	REAL	зсув змінної процесу
in	LMN_FAC	REAL	коеф. пропорційності сигналу керування
in	LMN_OFF	REAL	зсув сигналу керування
in	I_ITLVAL	REAL	начальне значення інтегральної складової
in	DISV	REAL	збурення змінної
out	LMN	REAL	сигнал керування
out	LMN_PER	WORD	периферійний сигнал керування
out	QLMN_HLM	BOOL	досягнення верхньої межи сигналом керування

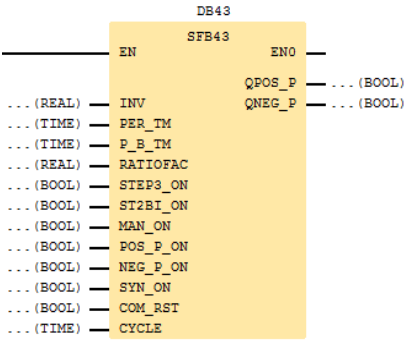


Рис.2.36. Блок *PULSEGEN* (SFB43)

out	QLMN_LLM	BOOL	досягнення нижньої межи сигналом керування
out	LMN_P	REAL	пропорційна складова
out	LMN_I	REAL	інтегральна складова
out	LMN_D	REAL	диференційна складова
out	PV	REAL	змінна процесу
out	ER	REAL	сигнал похибки (відхилення)

В програмі користувача буде використано блок формування ШІМ-сигналу *PULSEGEN* (SFB43) зі своїм блоком даних DB43. Зображення блоку подано на рис. 2.36, а його інтерфейс в табл. 2.7. Зауважимо, що вихід LMN з блоку ПІД-регулятора *CONT_C* є входом INV блока *PULSEGEN*, який в свою чергу формує сигнали керування виконавчим пристроєм.

Таблиця 2.7 – Параметри блоку *PULSEGEN* (SFB43)

Напрям сигналу	Ім’я параметра	Тип параметра	Опис параметра
in	INV	REAL	сигнал керування
in	PER_TM	TIME	період ШІМ
in	P_B_TM	TIME	мін. довжина імпульсу

in	RATIOFAC	BOOL	коефіцієнт пропорційності
in	STEP3_ON	BOOL	вмикання режиму 3-х позиційного регулювання
in	ST2BI_ON	BOOL	вмикання режиму 2-х позиційного регулювання
in	MAN_ON	BOOL	вмикання ручного режиму
in	POS_P_ON	BOOL	вмикання позит. імпульсу
in	NEG_P_ON	BOOL	вмикання негат. імпульсу
in	SYN_ON	BOOL	вмикання синхронізації
in	COM_RST	BOOL	повний рестарт блока
in	CYCLE	TIME	інтервал виклику блока
out	QPOS_P	BOOL	позитивний імпульс
out	QNEG_P	BOOL	негативний імпульс

В табл. 2.7 параметри STEP3_ON та ST2BI_ON визначають режим роботи регулятора за умови, що ввімкнений автоматичний режим роботи. Якщо параметр STEP3_ON знаходиться у стані «*TRUE*», то активується режим керування трьохпозиційним вихідним пристроєм. В протилежному випадку буде активований режим керування двохпозиційним вихідним пристроєм. Причому, якщо параметр ST2BI_ON знаходиться у стані «*TRUE*», то активується режим біполярного керування (від -100% до +100%). Якщо параметр ST2BI_ON знаходиться у стані «*FALSE*», то активується режим уніполярного керування (від 0% до +100%).

В режимі керування трьохпозиційним вихідним пристроєм активний позитивний вихід вмикає нагрівач та вимикає охолоджувач. Якщо активний негативній вихід, то нагрівач вимикається й охолоджувач вмикається. В двохпозиційному режимі керування, якщо виконавчий пристрій нагрівач, то він керується виходом QPOS_P. В протилежному випадку, охолоджувачем керує вихід QNEG_P.

Розрахунок тривалості імпульсу здійснюється за формулою:

$$PULSE=INV/100*PER\_TM. \quad (2.3)$$

В ручному режимі роботи регулятора імпульси генеруються за допомогою управління станом вхідних параметрів POS_P_ON та NEG_P_ON. Для трьохпозиційного вихідного пристрою у блоці передбачено блокування виходів, якщо активовано або деактивовано обидва вхідні параметри. Для двохпозиційного вихідного пристрою стан параметру NEG_P_ON не має значення.

Проект з прикладом ПІД-регулятора складається з програмних компонентів, які подані в табл. 2.8.

Запропонований варіант реалізації безперервного ПІД-регулятора є навчальним. Тому в ньому відсутні функції блокування та контролю помилок. На рис. 2.37 зображено вміст організаційного блоку OB35.

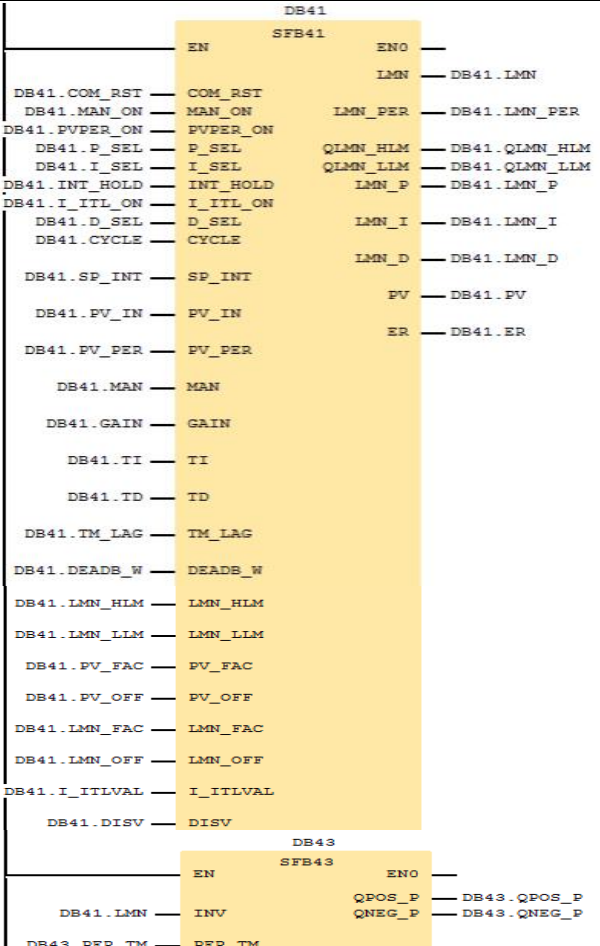
Таблиця 2.8– Вміст проекту з ПІД-регулятором та ШІМ

Блок	Символьне ім'я	Короткий опис
OB1	-	Організаційний блок для основного циклу сканування
OB100	-	Організаційний блок початкового запуску
OB35	-	Організаційний блок з системним перериванням у часі, 100 *мс*
SFB41	CONT_C	Безперервний ПІД-регулятор
SFB43	PWM_H	Генератор ШІМ-сигналу
DB35	-	Екземпляр блока даних для *OB35*
DB100	-	Екземпляр блока даних для *OB100*
DB41	DB_CONT_C	Екземпляр блока даних для *OB41*
DB43	DB_PWM_H	Екземпляр блока даних для *OB43*

На рис. 2.38 зображено вміст блоку OB100. В табл. 2.9 наведені значення деяких параметрів регулятора, які попередньо встановлюються після запуску програми користувача.

Таблиця 2.9 – Параметри налаштування блока з ПІД-регулятором

Параметр	Тип	Значення	Короткий опис
CYCLE	TIME	100 мс	Час квантування
GAIN	REAL	1.5	Пропорційна складова
TI	TIME	25 с	Час інтегрування
TD	TIME	6 с	Час диференціювання
TM_LAG	TIME	2 с	Час запізнення диференційної складової
LMN_HLM	REAL	100.0	Обмеження керуючого сигналу зверху
LMN_LLM	REAL	-100.0	Обмеження керуючого сигналу знизу



```
0          SET
1    //Startup routine for the controller and process
2          =     DB41.COM_RST                          --DB41
3    .
4    //Switch the controller to manual mode
5          =     DB41.MAN_ON                           --DB41
6          L     5.000000e+01
7          T     DB41.MAN                              --DB41
```

Рис.2.38. Блок OB100

При реалізації ПІД-регулятора пропонується для підвищення точності регулювання, використовувати вхідні і вихідні параметри в форматі число з плаваючою крапкою. Тому необхідно перетворити сигнал від модуля аналогових входів («ціле число») за допомогою FC105 попередньо потрібно перевести у формат числа з плаваючою крапкою. Керуючий сигнал від регулятора подається на вхід блока формування ШІМ-сигнала (див. табл. 2.7 та рис. 2.36).

Якщо потрібно реалізувати аналогове керування, то керуючий сигнал від регулятора перетворюється у ціле число за допомогою FC106 і далі передається в модуль аналогових виходів цифро-аналогового пере-

творення. Перетворення сигналів виконується в основному організаційному блоці OB1.

Код блока OB1 буде подібний до блока, який розглянутий у попередньому прикладі. Наприклад, перший ланцюг аналізує стан контактних датчиків рівня та формує меркер для відкриття клапана подавання води в ємність, як в попередніх прикладах. Параметр процесу (температура) вимірюється за допомогою термоперетворювача опору *Pt100 (Std)* та після перетворення в АЦП модуля *AIO234* у вигляді цифрового коду подається на вхід функції аналогового перетворювача FC105. У другому ланцюгу здійснюється перетворення цифрового коду параметру у фізичне значення параметру, тобто значення температури. Значення температури через блок даних DB41 подається на вхід регулятора в блоці OB35. Після розрахунку сигналу керування у блоці ПІД-регулятора сигнал надходить до ФБ ШІМ-регулятора, який керує живленням нагрівача. Для імітації збурення в програмі передбачено ручне вмикання охолоджувача. Останній ланцюг в блоці OB1 потрібен для внесення збурення шляхом ручного вмикання охолоджувача.

Крім сигналу про значення параметра процесу регулятор отримує сигнал від контактних термодатчиків (див. поз. 5 на рис. 2.29, табл. 2.5). Аналіз стану контактних термодатчиків потрібен для блокування роботи нагрівача.

В блоці OB35 ланцюг викликає по перериванню блок регулятора SFB41 кожні 100 мс. Це потрібне для оновлення даних про поточний технологічний процес. Крім того, формується сигнал ручного керування вихідним пристроєм (див. рис. 2.37). Для більшої наочності роботи пропонується встановити в настройках регулятора коефіцієнти пропорційності (GAIN) малого значення та час інтегрування (TI) – велике значення.

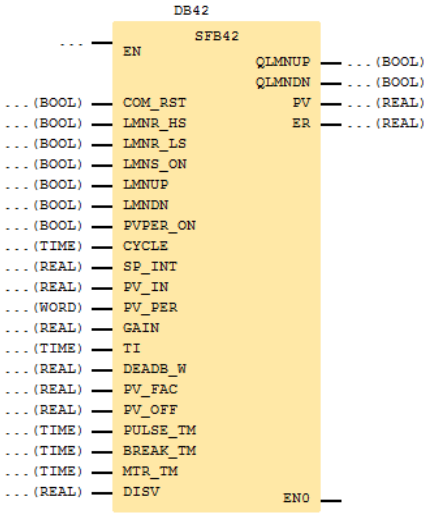


Рис. 2.39. Блок *CONT_S* (SFB42)

Насамкінець розглянемо інтерфейс безперервного регулятора *CONT_S* (SFB42), зображення якого подано на рис. 2.39.

Інтерфейс блока відповідає опису алгоритму регулятора, який зображений на рис. 2.32. Блок регулятора SFB42 отримує значення формальних параметрів від блока даних DB42. В табл. 2.10 наведено перелік формальних параметрів стандартного блока SFB42, за допомогою якого реалізований ступінчатий ПІД-регулятор.

Таблиця 2.10 – Параметри блоку ПІД-регулятора *CONT_S*

Напрям сигналу	Ім’я параметра	Тип параметра	Опис параметра
in	COM_RST	BOOL	повний перезапуск
in	LMNR_HS	BOOL	досягнення верхньої межі виконавчим механізмом
in	LMNR_LS	BOOL	досягнення нижньої межі виконавчим механізмом
in	LMNS_ON	BOOL	вмикання ручного керування
in	LMNUP	BOOL	активування сигналу керування «більше»
in	LMNDN	BOOL	активування сигналу керування «менше»
in	PVPER_ON	BOOL	вмикання читання периферійних змінних процесу
in	CYCLE	TIME	час циклу обчислення

in	SP_INT	REAL	внутрішнє завдання
in	PV_IN	REAL	вхід змінної процесу
in	PV_PER	WORD	периферійна змінна процесу
in	GAIN	REAL	коєф. підсилення регулятора
in	TI	TIME	інтервал інтегрування
in	DEADB_W	REAL	зона нечутливості регулятора
in	PV_FAC	REAL	коєф. пропор. змінної процесу
in	PV_OFF	REAL	зсув змінної процесу
in	PULSE_TM	TIME	мінімальне значення довжини імпульсу
in	BREAK_TM	TIME	мінімальне значення довжини паузи
in	MTR_TM	TIME	час роботи мотора між верхнім та нижнім положеннями
in	DISV	REAL	збурення змінної процесу
out	QLMNUP	BOOL	сигнал керування (більше)
out	QLMNDN	BOOL	сигнал керування (менше)
out	PV	REAL	змінна процесу
out	ER	REAL	сигнал похибки (відхилення)

РОЗДІЛ 3
ІНТЕГРАЦІЯ КОНТРОЛЕРІВ *VIPA* ТА ЗАСОБІВ ЛЮДИНО-МАШИННОГО ІНТЕРФЕЙСА

3.1 Розроблення ЛМІ на основі взаємодії ОП *VIPA OP03* та ПЛК *VIPA115* для системи гарячого водопостачання

В даному підрозділі буде розглянутий порядок створення ЛМІ на основі використання операторських панелей *VIPA*, зокрема панелі *OP03*. Панель має інтерфейс *MPI*, тому пристосована для сумісної роботи з контролерами *VIPA* усіх моделей с цим інтерфейсом. Для налаштування панелі використовують програму-конфігуратор *OPM*.

Отже запустить програму *OPM* для конфігурування панелі *OP03*. Відкриється стартове вікно програми, яке зображене на рис. 3.1.

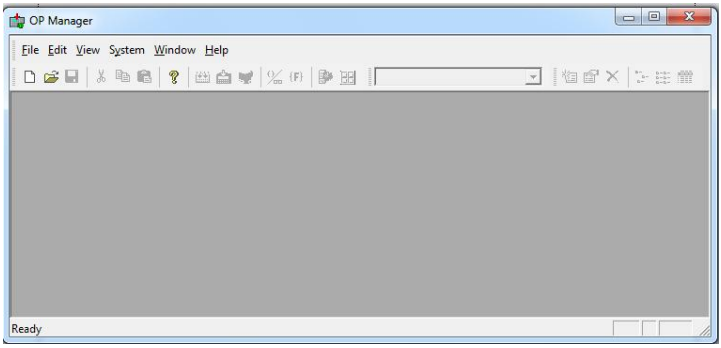


Рис. 3.1. Стартове вікно програми-конфігуратора панелей *VIPA*

Программа *OPM* є *Windows*-орієнтованою, тому вона має відповідний інтерфейс користувача та для неї діють основні способи роботи з додатками *Windows*. Так вікно з інтерфейсом користувача складається з основного меню, робочого простору та інформаційного рядка зі статусом програми.

Розробимо проект, який буде орієнтований на сполучення панелі та контролера з проектом, який розглянутий на рис. 1.44. Нагадаємо, що це проект дискретного керування системою водопостачання. Але враховуючі наявність панелі з можливістю вводу даних перенесемо функції управління в панель. Це перемикач режимів роботи установки та кнопка ручного пуску насоса. Також в панелі будемо відображати статус датчиків рівня, клапана та насоса. Виходячи з наведеного необхідно частково модифікувати програму, тобто змінні перемикача та кнопки відключити від фізичних каналів. Замість них можна використати меркерну пам'ять, наприклад меркери M0.1 та M0.2.

Отже в меню *File* натисніть ЛКМ на кнопку *New*, або натисніть одночасно на комбінацію клавіш Ctrl+N. Відкриється вікно *Step1* майстра налаштування проекту. Заповніть замість імені *Untitled* у полі *Project name:* ім'я конфігурації, наприклад, *Water_Station* та натисніть ЛКМ на кнопку *Далее*. В наступному вікні *Step2* майстер запропонує стандартний шаблон конфігурації, тому погодьтесь на пропозицію та натисніть ЛКМ на кнопку *Готово*. Збережіть проект на жорсткий диск ПК стандартними операціями, використовуючи, наприклад, кнопку для збереження файлу в рядку піктограм швидкого доступу до команд. В результаті буде отримано проект з шаблоном, зображення якого наведено на рис. 3.2.

Почніть конфігурування з налаштування параметрів зв'язку з контролером. Для цього ЛКМ виберіть з дерева проекту *OP03-Water_Station* категорію *Controllers*. Праворуч відкриється вікно з вже доданим контролером *PLC_1* типу *S7 - 300/400*. Подвійним кліком ЛКМ відкрийте рядок з контролером для редагування. У вікні, яке відкриється, змініть у відповідному полі ім'я контролера на *VIPA_115*. Далі ЛКМ натисніть на кнопку *Parameters…* та налаштуйте параметри зв'язку змінивши за необхідністю адреси панелі та контролера. Для випадку з'єднання лише однієї панелі та одного ПЛК залиште адреси як є. Закрийте послідовно вікна *S7-300/400* та *PLC* натиснувши ЛКМ на кнопку *OK*.

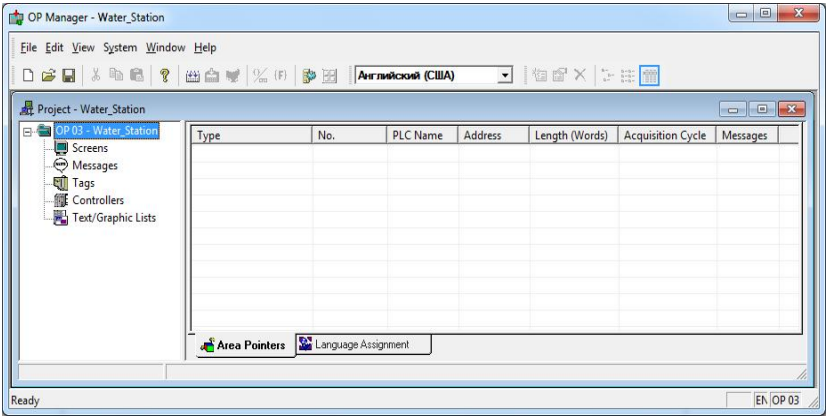


Рис. 3.2. Вікно з шаблоном конфігурації для панелі *VIPA OP03*

Наступним кроком додамо тегі. Для цього ЛКМ з дерева проекту *OP03-Water_Station* виберіть категорію *Tags*. Праворуч відкриється перелік тегів, які є в шаблоні. Це тегі таймерів та лічильників, усього 40 тегів, що показано внизу вікна у інформаційному рядку. Додамо тег, який буде керувати режимом роботи установки водопостачання. Для цього відкрийте контекстне меню ПКМ у робочому просторі вікна *Project Water_Station* і виберіть ЛКМ у переліку меню команду *Tag Insert…* . Відкриється вікно налаштування тега, яке це зображено на рис.3.3.

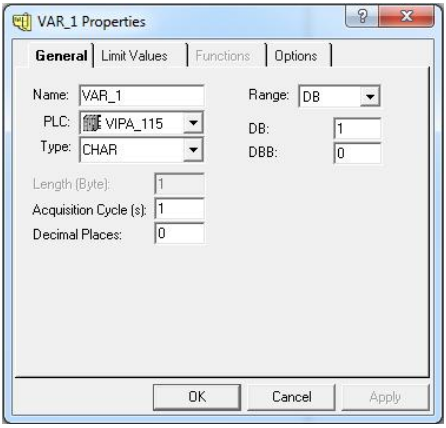


Рис. 3.3. Вікно налаштування тега

В головному вкладенні *General* вікна *VAR_1 Properties* знаходяться поля для введення імені тега, його адреси, типа

змінної та її місця зберігання. Також є поля для налаштування параметрів щодо кількості байт, інтервалу опитування та кількість позицій при відображенні нецілих чисел. Введіть у поле *Name:* ім'я тега, наприклад, *MODE*, який потрібен для передавання сигналу на вмикання або вимикання автоматичного режиму. Поле *PLC:* залиште без змін. У полі *Type:* замість типу *Char*, яке вказане за умовчанням, за допомогою переліку, що випадає, виберіть тип *BOOL*. У вікні *Range* з випадаючого переліку виберіть область пам'яті в ПЛК. В даному випадку це меркерна пам'ять, тому оберіть символ M. Далі вкажіть номер маркерного слова та відповідний біт виходячи з проекту в ПЛК: це буде M0.0. Далі натисніть на кнопку *Apply*, а далі – на кнопку *OK*.

Далі потрібно налаштувати екран для призначення статусу тегу за допомогою функціональних кнопок на панелі. Для цього ЛКМ виберіть категорію *Screens* з дерева проекту та за допомогою контекстного меню додайте нове вікно до вже існуючих. До переліку вікон буде додано вікно з ім'ям *PIC_1*. Поставте курсор ЛКМ у поле вікна зліва у перший рядок та подвійним кліком відкрийте діалог налаштування місця відображення *Input/Output Field*, який зображений на рис. 3.4. Заповніть поля вкладення *General*, а саме: поле *Usage:* – залиште без змін, у полі *Field Type:* – виберіть *Input/Output*, у полі *Display As:* – виберіть *Binary*, а у полі *Tag:* – виберіть раніше створений тег *MODE*. Довжину поля відображення стану змінних процесу *Field Length* залишить за умовчанням (4-е символи). Після додавання тега активується кнопка налаштування його властивостей, що надає можливість його редагувати. На-

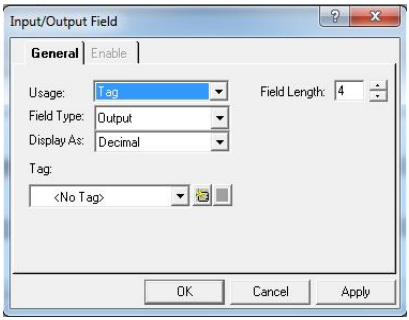


Рис. 3.4. Вікно налаштування поля відображення змінних

тисніть ЛКМ на кнопку *Apply* для підтвердження налаштувань та кнопку *OK* – для закриття вікна. Далі додамо до цього елементу вікна функцію введення значення за допомогою функціональної кнопки на клавіатурі панелі. Для цього ЛКМ клікніть на елемент, □значений нижче рядка знакомісця – « 1». Відкриється діалогове вікно *Soft Key - 1*, яке призначено для програмування кнопок панелі. В цьому вікні у вкладенні *General* знов додайте потрібний тег *MODE* та вкажіть у вкладенні *Functions* тип функції, яку повинна виконувати кнопка панелі. Для цього натисніть ЛКМ на кнопку *Add* та у вікні *Select Object* у переліку *Edit Bit* виберіть команду *Set Bit*. На завершення в наступному вікні виберіть тег *MODE* та підтвердить вибір натисканням ЛКМ на кнопку *OK*. Повторіть усі дії щодо налаштування іншої кнопки для скидання тегу *MODE*, тобто надайте для кнопки «□2» виберіть функцію *Reset Bit*.

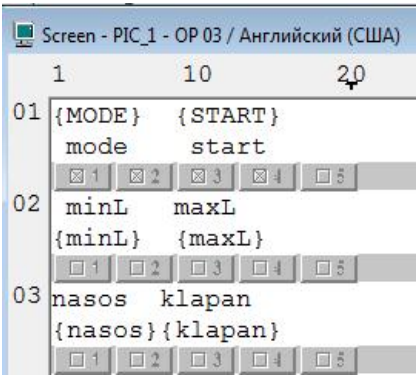


Рис. 3.5. Вигляд екрана користувача

Далі в екрані користувача подібно до описаного створіть елемент відображення та програмні кнопки для ручного вмикання та вимикання насоса за допомогою тега *START*. Також додайте до цього ж екрану елементи, які відображають статус датчиків рівня, клапана та насоса. Результат проведених дій зображений на рис. 3.5.

Перед завантаженням проекту до панелі оператора його потрібно запам’ятати на ПК. Потім за допомогою команди *Compile* з переліку *File* основного меню програми або кнопки  скомпілювати конфігурацію. У разі вдалого компілювання у інформаційному вікні

Output Window з'явиться відповідне повідомлення та розмір проекту для завантаження. Якщо розмір проекту менше 256 кБ, то він може бути завантажений до панелі. Одразу після компілювання будуть активовані ще дві кнопки: це кнопка для завантаження до панелі  та кнопка запуску симулятора панелі  . Якщо панель фізично відсутня, тому використаємо симулятор. Вікно з активованим симулятором панелі оператора показано на рис. 3.6.

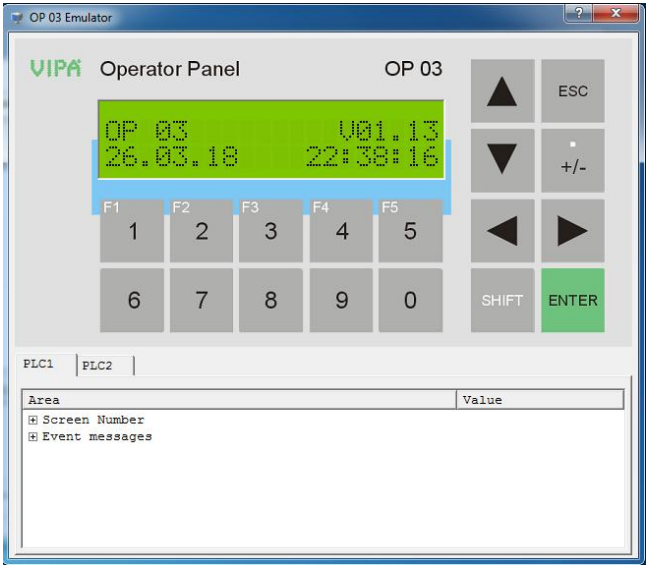


Рис. 3.6. Вікно з екраном симулятора

Спостереження за поточними параметрами програми користувача з боку панелі в режимі симулятора неможливе без наявності контролера, але можливо відкрити вікно користувача та перевірити роботу програмних кнопок для введення значень до тегів. Тому ЛКМ натисніть на кнопку *ENTER* в симуляторі панелі для переходу до її системних вікон. Далі необхідно перейти до вікна користувача з ім'ям PIC_1. Для цього

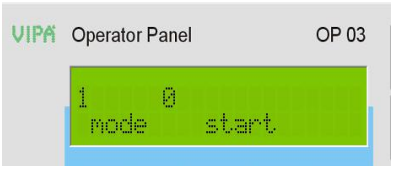


Рис. 3.7. Екран користувача (1 та 2 рядок)

потрібно активувати ЛКМ кнопку *SHIFT* на симуляторі панелі та натиснути ЛКМ на функціональну кнопку *F2*. У вікні панелі з'явиться перелік екранів. Для відкриття екрана користувача потрібно спочатку ЛКМ деактивувати кнопку *SHIFT*. Далі кнопками ▼, ▲, ►, ◄ перемістить курсор всередині вікна на символі з номером екрана до появлення рядка з ім'ям PIC_1. Натисніть ЛКМ на кнопку *ENTER* для відкриття екрану PIC_1 з даними про процес. Відкриється екран з першими двома рядками, який зображений на рис. 3.7. На цьому екрані можна перевірити роботу софт-кнопок. Для цього активуйте кнопку *SHIFT* і далі по черзі натисніть на кнопку *F1* для запуску автоматичного режиму та *F2* – для переходу установки на ручний режим керування. У ручному режимі перевірте можливість вмикання насоса шляхом застосування кнопки *F3* та вимикання насоса – шляхом застосування кнопки *F4*. Подальше натискання ЛКМ на кнопку ▼ призведе до переходу на наступні рядки активного екрану, які показані на рис. 3.8 та 3.9.

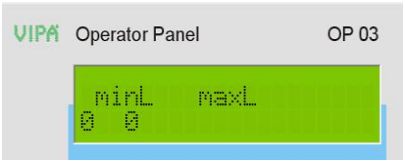


Рис. 3.8. Екран користувача (3 та 4 рядок)

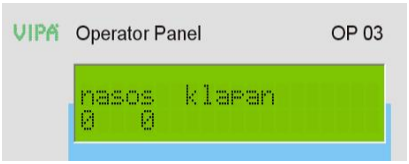


Рис. 3.9. Екран користувача (5 та 6 рядок)

У випадку, коли є можливість фізичного з'єднання панелі та контролера потрібно ЛКМ натиснути на кнопку  . Відкриється вікно *Interface Settings* для налаштування *MPI*-інтерфейсу між панеллю та

ПЛК. В цьому вікні ЛКМ активуйте кнопку *Settings…* для налаштування параметрів інтерфейсу у вікні *MPI Settings*, а саме: параметрів *COM*-порту комп'ютера та мережних налаштувань – *MPI*-адреси панелі. Після введення потрібних налаштувань ЛКМ послідовно два рази на кнопку *OK* для закриття вікон. Якщо налаштування параметрів зроблені вірно почнеться процес завантаження проекту до панелі. Зауважимо, що для завантаження проекту до панелі використовують так званий «зелений кабель» від *VIPA*. На завершення зазначимо, що можна налаштувати загальні властивості панелі. Це робиться шляхом натискання ЛКМ на кнопку у рядку швидкого доступу  або команду *Settings…* в переліку *System* у головному меню програми. Відкриється вікно, яке зображено на рис. 3.10. У вікні *Settings* можна налаштувати формат часу та дати, паролі доступу та софт-кнопки *F1…F5*. У вікні є кнопка *Defaults…* для встановлення параметрів «за умовчанням».

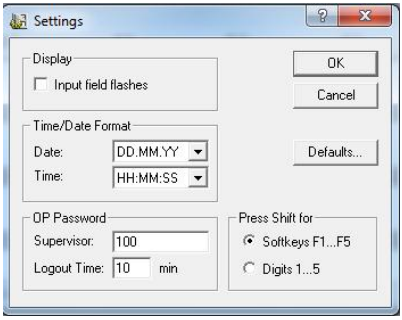


Рис. 3.10. Вікно налаштувань властивостей панелі

3.2 Реалізація протоколу *Modbus RTU/ASCII* в контролерах *VIPA*

Створення розподіленої системи управління та організація зв'язку ПЛК *VIPA* з іншими апаратними та програмними засобами з використанням протоколу *Modbus RTU/ASCII* [12] можливе для процесорних модулів *VIPA*, якщо вони мають порт *RS232/RS485* з підтримкою протоколу. Налаштуємо контролер *VIPA115* для зв'язку за протоколом *Modbus RTU*. Цей контролер має у своєму складі порт *RS485* з підтримкою протоколу *Modbus RTU/ASCII* як головного так і підлеглого при-

строю. У даному прикладі буде розглянуто лише принцип організації обміну даними за протоколом. Але у подальшому можливе доповнення будь-якого проекту з програмою користувача функцією обміну даними за протоколом *Modbus RTU/ASCII*. Не зупиняючись на принципах реалізації саме протоколу *Modbus* у різних пристроях, лише покажемо особливості застосування цього протоколу у контролерах *VIPA* на прикладі *VIPA115*. Щодо відомостей про протокол, то їх можна отримати в джерелах [12, 13].

Отже, в контролері *VIPA115* є можливість програмного налаштування його послідовного інтерфейсу *RS485* для роботи за протоколом *Modbus*. Для цього потрібно в проекті, який був розроблений раніше, додатково налаштувати цей інтерфейс. Також потрібно додати до програмної частини проекту функціональні блоки для реалізації протоколу *Modbus*. Виходячи із моделі обміну даними «головний-підлеглий» необхідно призначити головний пристрій. Якщо контролер буде працювати сумісно з панеллю оператора в РСУ, то його призначають головним. Але у цьому випадку потрібно організувати додаткову перевірку на відсутність зв'язку з панеллю та діагностування каналу зв'язку за допомогою коду помилки, який формує блок передавання та приймання даних. Крім того потрібно урахувати умови роботи лінії зв'язку: її довжину, швидкість обміну та параметри кадру (кількість символів, паритет та кількість стопових бітів). Усе перераховане впливає на якість та достовірність обміну даними. Якщо головним пристроєм буде обрана панель, то в неї теж потрібно організувати контроль наявності зв'язку з контролером. Для перевірки працездатності проекту обміну даними за протоколом *Modbus* використаємо дві програми, які виконують функцію головного та підлеглого пристрою. Це відповідно програми *ModSim32* [14] та *CAS Modbus Scanner* [15], які працюють під управлінням операційних систем *Windows* усіх версій включно до 7-ої. Процедури встановлення, використання вказаних програм є досить простими і не потре-

бують додаткових пояснень. Достатньо звернуться до керівництва до цих програм щоб налаштувати їх на прийом запитів від ведучого пристрою (від контролера) та формування відповідей з боку веденого пристрою (від програми *ModSim32*), а також відправлення запитів до контролеру з боку симулятора ведучого пристрою (від програми *CAS Modbus Scanner).* Фізично контролер та ПК з програмами з'єднуються за допомогою перетворювача інтерфейсів *USB/RS485* або *RS232/RS485*, який працює в автоматичному режимі щодо управління потоком даних для напівдуплексного режиму обміну.

3.2.1. Реалізація протоколу Modbus в ПЛК VIPA115 як головного пристрою

В цьому прикладі апаратне налаштування ПЛК *CPU115DIO32SER* (зак. № 115-6BL32) не потрібне. Це пов'язано з тим, що інтерфейс зв'язку *RS-485* не конфігурується, а налаштовується безпосередньо в програмі користувача за допомогою бібліотечних функцій. Також не потрібне конфігурування інтерфейсу для старших моделей процесорних модулів, наприклад, не потребує конфігурування модуль *CPU313SC* (№VIPA 313-5BF03-0AB0). Проте, в модулі процесора *CPU314ST/DPM* (№VIPA 314-6CF02) інтерфейс зв'язку є комбінованим, тобто може перемикатись програмно у програмі конфігурування апаратних ресурсів. Цей інтерфейс може працювати в режимі шині *PROFIBUS-DP* (ведучий або ведений пристрій) або в режимі *PtP* (з'єднання типу «точка-точка»), але лише як головний пристрій. Таким чином, після створення проекту в середовищі *WinPLC V5* необхідно запустить модуль апаратного конфігурування ресурсів контролера. Далі в мережних параметрах центрального процесорного модуля *CPU314ST/DPM* потрібно вибрати необхідний режим зв'язку (*PROFIBUS-DP* або *Modbus*).

Як було раніше зазначено, розробимо проект обміну даними за протоколом *Modbus-RTU* між ПЛК *VIPA CPU115* (головний пристрій)

та програмним додатком, який запущений на ПК під управлінням операційної системи *Windows*. В нашому випадку це буде програмний симулятор підлеглого пристрою *ModSim32*, який буде з'єднаний з ПЛК за допомогою перетворювача інтерфейсів (ПІ) *RS232/RS485* та двох провідного кабелю. Схема електричних з'єднань наведена в технічній документації на модуль [7]. Лише відмітимо, що для фізичного з'єднання ПЛК та ПК крім ПІ, потрібен кабель, у якого з одного боку є з'єднувач типу *D-Sub-9 (M)*, а з іншого боку проводи приєднуються безпосередньо до перетворювача інтерфейсів за допомогою гвинтових затискачів. Причому лінія «А» (*Rx/Tx-N*) з'єднана з 8-им піном з'єднувача типу *D-Sub (M)*, а лінія «В» (*Rx/Tx-P*) з'єднана з 3-ім піном з'єднувача типу *D-Sub (M)*.

В програмі користувача після подавання живлення до ПЛК спершу буде здійснюватись однократне налаштування інтерфейсу *RS485* для роботи за протоколом *Modbus*, а потім періодичне зчитування значення параметру, наприклад, типу INT із підлеглого пристрою (програми-симулятора *ModSim32*).

Насамперед розглянемо деякі відомості щодо налаштування обміну даними в контролерах *VIPA* за протоколом *Modbus*. Обмін даними між прикладною програмою та *Modbus* відбувається через реалізований в ПЛК протокол та вхідні (*IN*) та вихідні (*OUT*) буфери розміром 2x256 байт. Але протокол *Modbus* для контролерів *VIPA* не є «рідним», тому він реалізований програмно за допомогою системних функцій. В ПЛК *VIPA* реалізовані функції протоколу, які наведені в табл. 3.1.

Для реалізації обміну даними у складі бібліотеки стандартних функцій є функція для налаштування інтерфейсу *RS232/RS485* (*SFC216*) та функції для передавання (*SFC217*) та приймання даних (*SFC218*). Усі функції мають входи для параметрування та обміну і повертають код

помилки для визначення причини порушення зв'язку та кількість байтів, які були оброблені у буфері обміну. Взагалі обмін здійснюється за допомогою використання блоків даних, які виконують функцію буферної пам'яті. Надамо коротку характеристику вказаних функцій.

Табл. 3.1 – Функції протоколу *Modbus*, які реалізовані в ПЛК *VIPA*

Код функції	Назва функції	Порядок обміну даними
01h	Читання *n* вихідних біт	*IN* – вхідний буфер веденого ПЛК
02h	Читання *n* вхідних біт	*OUT* – вихідний буфер веденого ПЛК
03h	Читання *n* вихідних регістрів	*IN* – вхідний буфер веденого ПЛК
04h	Читання *n* вхідних регістрів	*OUT* – вихідний буфер веденого ПЛК
05h	Запис одного вихідного біту	*IN* – вхідний буфер веденого ПЛК
06h	Запис одного вихідного регістру	*IN* – вхідний буфер веденого ПЛК
10h	Запис *n* вихідних регістрів	*IN* – вхідний буфер веденого ПЛК

Функція налаштування інтерфейсу *SFC216* викликається лише один раз перед початком роботи програми користувача або у випадку її перезавантаження. Тому цю функцію доцільно викликати з організаційного блоку який спрацьовує лише в таких випадках. Це блок *OB100*. В табл. 3.2 наведено програмний інтерфейс функції *SFC216* з поясненнями щодо вибору параметрів для налаштування фізичного інтерфейсу.

Окремо зауважимо, що порт контролера може працювати за іншими протоколами. Це таки протоколи: *ASCII*, *STX/ETX*, *3964R*, *USS* та *Modbus*, але розглянемо лише *Modbus*-протокол, який є найбільш поширеним. Інші протоколи рідко застосовують в промислової автоматизації. Теж стосується інших параметрів налаштування. Тому наведені відомості про найбільш часто застосовані значення параметрів.

Табл. 3.2 – Інтерфейс функції *SFC216* (*SER_CFG*)

Ім'я параметру	Клас параметру	Тип даних	Призначення параметра
Protocol	IN	BYTE	Вибір режиму обміну 5 – *Modbus RTU / Master* 6 – *Modbus ASCII / Master* 7 – *Modbus RTU / Slave* 8 – *Modbus ASCII / Slave*
Parameter	IN	ANY	Покажчик на *DB* з параметрами протоколу (*timeout* та *address*)
Baudrate	IN	BYTE	Швидкість обміну у бодах: 09_h – 9600; $0A_h$ – 14400; $0B_h$ – 19200; $0C_h$ – 38400; $0D_h$ – 57600; $0E_h$ – 115200
CharLen	IN	BYTE	Довжина кадру: 2 – 7 біт лише для *Modbus ASCII*; 3 – 8 біт
Parity	IN	BYTE	Парність: 0 – *No*, 1 – *Odd*, 2 – *Even*
StopBits	IN	BYTE	Кількість стопових бітів: 1 – 1 біт, 3 – 2 бітів
FlowControl	IN	BYTE	Керування потоком даних: 0 – *RTS Off*; 1 – *RTS Auto* (*RS-485*); 2 – *HW flow* (лише для *ASCII*)
RetVal	OUT	WORD	Код помилки: 0 – помилок немає

Для звертання з боку функції *SFC216* до блоку даних *DB* з параметрами протоколу (Parameter) для випадку ведучого пристрою формат блоку виглядає так:

DBW0: Timeout WORD (затримка відповіді *10 мс)

для випадку веденого пристрою формат блоку виглядає так:

DBW0: Address BYTE (адреса веденого пристрою в мережі)

DBW1: Timeout WORD (затримка відповіді *10 мс).

Вихідний параметр функції *SFC216* RetVal повідомляє про наявність або відсутність помилок виконання функції. В табл. 3.3 наведено коди та опис помилок конфігурування функцій.

Табл. 3.3 – Коди помилок виконання *SFC216* (*SER_CFG*)

Код помилки	Коментар з роз'ясненням
0000_h	Помилок немає
$809A_h$	Помилка ініціювання інтерфейсу (порт зачинений)
$8x24_h$	Помилки налаштування параметрів інтерфейсу
$828x_h$	Помилкове значення параметрів інтерфейсу
8092_h	Помилковий доступ до *DB* (блок даних занадто короткий)

Функції передавання даних *SFC217* та приймання даних *SFC218* викликаються періодично в головному циклі сканування контролеру з організаційного блоку *OB1*. В табл. 3.4 та 3.5 наведено інтерфейси функцій з поясненнями щодо вибору параметрів.

Звертання з боку функції передавання або приймання даних до блоку даних (буферу обміну *IN*) за допомогою покажчика виглядає так: DataPtr:=P#DB5.DBX0.0BYTE124. Це звертання до змінних у блоці *DB5* в діапазоні від 0-го до 124-го байту, усього 125 байт.

Вихідний параметр функцій *SFC217* та *SFC218* RetVal повідомляє про наявність або відсутність помилок виконання функцій передавання

та приймання даних. В табл. 3.6 та 3.7 наведено коди та опис помилок конфігурування функцій. Але в таблицях наведено не усі коди помилок. Якщо повернуте значення коду помилки немає у таблицях, то потрібно звернутись до документації до процесорного модуля.

Табл. 3.4 – Інтерфейс функції для передавання даних *SFC217* (*SER_SND*)

Ім'я параметру	Клас параметру	Тип даних	Призначення параметра
DataPtr	IN	ANY	Покажчик на буфер з даними, що передаються
DataLen	OUT	WORD	Кількість даних, які відправлені
RetVal	OUT	WORD	Код помилки: 0 – помилок немає

Табл. 3.5 – Інтерфейс функції для приймання даних *SFC218* (*SER_RCV*)

Ім'я параметру	Клас параметру	Тип даних	Призначення параметра
DataPtr	IN	ANY	Покажчик на буфер з даними, що приймаються
DataLen	OUT	WORD	Кількість даних, які прийняті
Error	OUT	WORD	Не використовується
RetVal	OUT	WORD	Код помилки: 0 – помилок немає

Розглянемо процес обміну даними, який зображений на рис. 3.11. На рис. 3.11 зображено як контролер виконує функцію головного пристрою. Дані, які потрібно процесору відправити, зберігаються у буфері для відправлення (*OUT*) розміром 2x256 байтів, а потім проходять через фізичний інтерфейс. Якщо інтерфейс приймає дані, він зберігає їх у буфері для приймання (*IN*) розміром 2x256 байтів і далі зчитується

процесором. Перетворення даних здійснюється згідно протоколу автоматично. Крім того, головний пристрій очікує підтвердження (квитанції) від підлеглого на випадок здійснення функцій запису. У разі відсутності відповіді здійснюється повторне відправлення даних. Якщо відповідь надійшла, то вона обробляється за допомогою функції *SFC 218 SER_RCV*. Якщо транзакція була виконана без помилок, контролер починає новий цикл обміну.

Табл. 3.6 – Коди помилок виконання функції *SFC217* (*SER_SND*)

Код помилки	Коментар з роз'ясненням
0000_h	Готов до передавання даних
1000_h	Немає передавання (довжина буфера даних дорівнює 0)
$8x24_h$	Помилкове значення параметрів функції (x – означає будь-яку цифру)
2000_h	Відправлено без помилок
2001_h	Відправлено з помилками
9000_h	Переповнення буферу даних
9001_h	Об'єм даних занадто великий (>256 байтів)
9002_h	Об'єм даних занадто малий (<2 байтів)

Табл. 3.7 – Коди помилок виконання функції *SFC218* (*SER_RCV*)

Код помилки	Коментар з роз'ясненням
0000_h	Помилок немає
1000_h	Прийнятий буфер занадто малий (дані втрачені)
$8x24_h$	Помилкове значення параметрів функції (x – означає будь-яку цифру)
$809A_h$	Порт не створений (немає доступу)
$809B_h$	Порт не налаштований

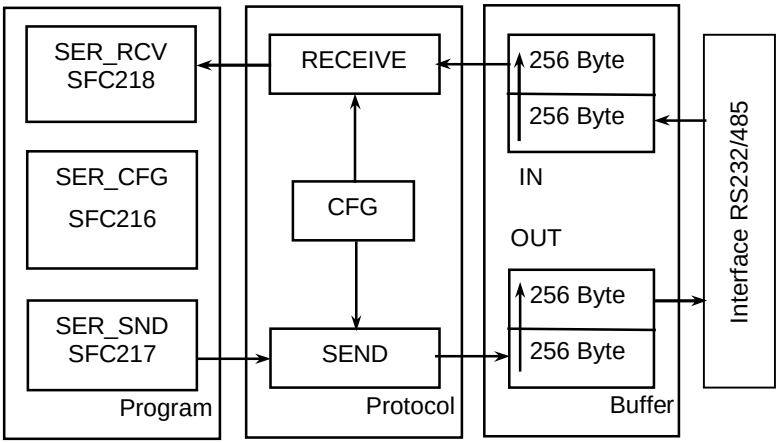


Рис. 3.11. Схема реалізації обміну за протоколом *ModBus* (*Master*)

Програму користувача почнемо розробляти з формування структури програми. Як вже зазначалось, в проекті необхідно створити організаційні блоки *OB1* та *OB100*. В блоці *OB100* потрібно викликати функцію *SFC216* для налаштування інтерфейсу. Для передавання параметрів до функції потрібно створити блок даних для налаштування інтерфейсу. Нехай нам потрібно налаштувати інтерфейс для обміну даними за протоколом *Modbus-Master* з такими значеннями параметрів: швидкість – 9600 біт/с, формат кадру – «8-n-1» та блок даних для параметрування з часом очікування відповіді (*timeout*) – *DB100*.

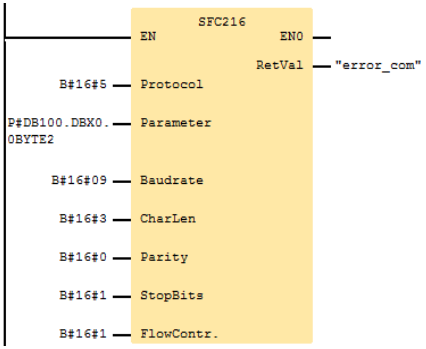


Рис. 3.12. Вміст блока *OB100*

Результат параметрування інтерфейсу *RS-485* для обміну даними за протоколом *Modbus-Master* показаний на рис. 3.12.

Особливість налаштувань функції *SFC216* у блоці *OB100* – це покажчик на блок даних *DB100*, з якого процесор зчитує значення параметру часу очікування відповіді та передає його до формального параметра *Parameter* функції *SFC216*. У рядку покажчика вказано на зчитування двох байтів. Вміст блока даних *DB100*, на який посилається покажчик, зображено на рис. 3.13. Як бачимо на рис. 3.13 змінна *timeout* блоку даних *DB100* має значення *W#16#000A*, що дорівнює 10*10мс=100мс.

Declaration	Name	Type	Initial value	Comment
var S		STRUCT		
var S	timeout	WORD	W#16#000A	
var S		END_STRUCT		

Рис. 3.13. Вміст блоку *DB100*

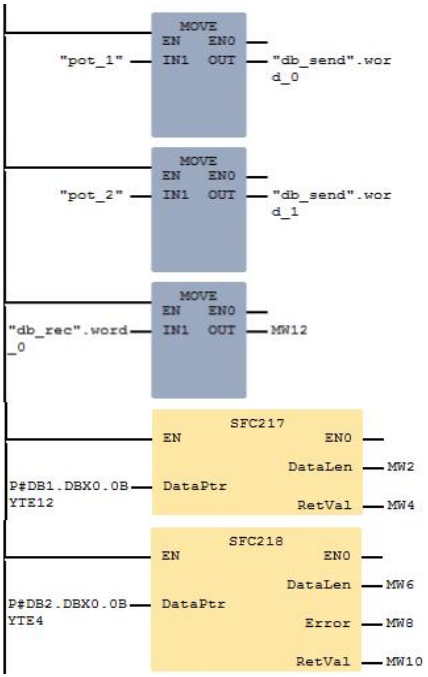


Рис. 3.14. Вміст блоку OB1

В організаційному блоці *OB1* перед викликом функцій для передавання даних здійснюється пересилання значення аналогових потенціометрів до блока даних *DB1*. Функція приймання даних зчитує значення параметра із блока даних *DB2* та передає до меркерного слова *MW12*. На рис. 3.14 зображено ланцюги організаційного блока *OB1*. У блоці *OB1* викликаються функції передавання та приймання даних з посиланням на відповідні блоки даних – *DB1* та *DB2*. У рядку покажчиків на параметр *DataPtr* вказано на

зчитування 12-и байтів з блоку *DB1* та 4-х байтів з блоку *DB2*. Вміст блоків даних зображений на рис. 3.15 та 3.16. В блоці даних *DB1* вказано адресу веденого пристрою, номер функції, адресу початкового регістра у веденому пристрої, кількість регістрів, кількість байтів даних та власне змінні для обміну (*word_0*, *word_1*) зі значенням аналогових потенціометрів контролера. Взагалі блок даних *DB1* – це структура кадру транзакції з функцією передавання значення аналогових потенціометрів як чисел типу *INT*.

	Address	Declaration	Name	Type	Initial value	Comment
		var S		STRUCT		
.	0.0	var S	dummy	BYTE	B#16#00	для вирівнювання даних
.	1.0	var S	slave	BYTE	B#16#10	адреса веденого пристрою
.	2.0	var S	fun_kod	BYTE	B#16#10	номер функції
.	3.0	var S	add_hi	BYTE	B#16#00	початкова адреса регістра (ст. байт)
.	4.0	var S	add_low	BYTE	B#16#00	початкова адреса регістра (мол. байт)
.	5.0	var S	kol_hi	BYTE	B#16#00	кількість регістрів (ст.байт)
.	6.0	var S	kol_low	BYTE	B#16#02	кількість регістрів (мол.байт)
.	7.0	var S	kol_byte	BYTE	B#16#04	кількість байтів даних
.	8.0	var S	word_0	WORD	W#16#0000	значення потенціометра №1
.	10.0	var S	word_1	WORD	W#16#0000	значення потенціометра №2
.		var S		END_STRUCT		

Рис. 3.15. Вміст блоку *DB1*

	Address	Declaration	Name	Type	Initial value
		var S		STRUCT	
.	0.0	var S	word_0	WORD	W#16#00AB
.	2.0	var S	word_1	WORD	W#16#0010
.		var S		END_STRUCT	

Рис. 3.16. Вміст блоку *DB2*

Для перевірки роботи програми користувача ПЛК необхідно використати програму, яка імітує роботу веденого *Modbus*-пристрою. Як було раніше зазначено, це можливо за допомогою програми *ModSim32*. Лише додамо, що в програмі потрібно визначити адресу веденого «16» та створити необхідну кількість регістрів для обміну даними (по два регістри – для запису та зчитування з боку ведучого). Насамкінець необхідно визначити ін-

терфейс (номер *COM*-порту ПК) та параметри обміну (*9600, 8-n-1*), які відповідають налаштуванням інтерфейсу контролера.

3.2.2. Реалізація протоколу Modbus в ПЛК VIPA115 як підлеглого пристрою

Якщо контролер виконує функцію веденого *Modbus*-пристрою, то програма користувача подібна розглянутий у п.3.2.1 на рис. 3.14. Відмінність полягає у складі блока *OB100* для параметрування інтерфейсу обміну, де до значення таймауту додається адреса веденого пристрою. Тобто, відмінність полягає у параметрах, до яких звертається функція *SFC216*. Якщо контролер виконує функцію веденого пристрою, то в блоці даних *DB100* вказується адреса веденого пристрою. При цьому покажчик, який вказує на параметри протоколу буде таким: P#DB100.DBX0.0BYTE3. Отже, розглянемо процес обміну даними, який зображений на рис. 3.17.

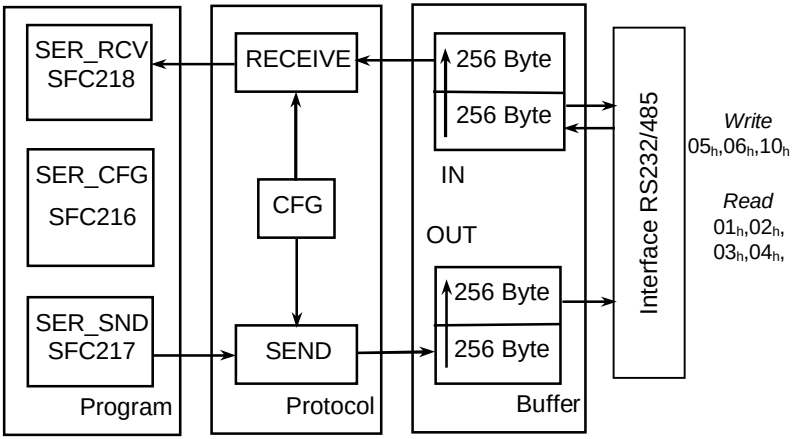


Рис. 3.17. Схема реалізації обміну за протоколом *ModBus* (*Slave*)

Дані, які контролер повинен відправити за запитом ведучого пристрою, зберігаються у буфері для відправлення розміром 2x256 байт.

Ведучий пристрій може запитати значення за допомогою стандартних функцій читання (функції читання 02_h, 04_h). Якщо ведений пристрій отримує дані від ведучого (функції запису 05_h, 06_h, 10_h), вони зберігаються у буфері прийому розміром 2x256 байт і можуть бути автоматично прочитані процесором. За допомогою функцій читання 01_h та 03_h ведучий пристрій отримує доступ до даних у буфері *IN* веденого.

Для перевірки роботи програми користувача необхідно використати програму, яка імітує роботу ведучого *Modbus*-пристрою шляхом формування запитів. Це програма *CAS Modbus Scanner*. В налаштуваннях програми необхідно вказати адресу веденого «16» та створити потрібні запити на зчитування та запис даних відповідно до доступних функцій протоколу. Також потрібно вказати параметри обміну: формат кадру та швидкість обміну даними. Програма відправляє запити на зчитування або запис даних однократно або циклічно. Отримані відповіді можна спостерігати у інформаційному вікні у вигляді кадрів або розкодоване значення в робочому просторі програми.

3.3 Реалізація шини *PROFIBUS-DP* в контролерах *VIPA*

Шина *PROFIBUS* [16] – це міжнародний стандарт, який застосовується для здійснення мережної взаємодії між пристроями розподіленої системи управління. Це в основному пристрої польового та технологічного рівня виробництва *Siemens* та *VIPA*. Шина *PROFIBUS* є власною розробкою компанії *Siemens*, яка підтримана багатьма виробниками і на цей час є стандартом [16, 17]. Вона реалізована на послідовній польовій шині з використанням інтерфейсу *RS-485*. Взагалі існує декілька специфікацій стандарту на шину *PROFIBUS*. Але зосередимось лише на специфікації *PROFIBUS-DP*, яка використовується на нижньому (датчики/виконавчі пристрої) або середньому (технологічний процес) рівні управління. Стандарт *PROFIBUS-DP* – це спеціальний прото-

кол, призначений, головним чином, для вирішення завдань автоматизації в умовах виробництва. Він являє собою високоефективну альтернативу паралельного підключення між ПЛК та віддаленої периферією. Протокол *PROFIBUS-DP* був розроблений для високошвидкісного обміну даними між ПЛК та датчиками і виконавчими пристроями. Обмін даними проводиться циклічно з синхронним або асинхронним режимом. Протягом одного циклу ведучий пристрій зчитує вхідні величини від ведених пристроїв і записує в них вихідну інформацію.

Властивості ведучого пристрою шини *PROFIBUS-DP* налаштовуються в конфігураторі апаратних ресурсів середовища *WinPLC V5* від *VIPA*. Спочатку до конфігурації ПЛК додається процесорний модуль. У даному випадку це буде модуль процесора *CPU314ST/DPM* (№VIPA 314-6CF02). Далі потрібно перемкнути в основних властивостях процесорного модуля інтерфейс *RS-485* на режим роботи *PROFIBUS-DP*. Інші налаштування апаратних ресурсів для даного прикладу не мають значення. При перенесенні конфігурації до ПЛК одночасно в ньому створюється програмний модуль ведучого пристрою *PROFIBUS-DP*. Під час перезапуску контролера ведучий *DP*-пристрій створює власну область даних в адресному простору пам'яті процесорного модуля. В контролері можливе використання спільно з процесором для роботи ведучого пристрою *PROFIBUS-DP* карти пам'яті формату *MMC* в якості зовнішнього накопичувача. Зауважимо, що робота модуля *CPU314ST/DPM* в режимі *PROFIBUS-DP Master* надає можливість підключення до 32 ведених пристроїв на швидкостях від 9.6 кбіт/с до 12 Мбіт/с. При цьому розмір пам'яті для обміну складає по 244 байта на вхід та вихід. Якщо модуль *CPU314ST/DPM* налаштований на роботу в режимі *PROFIBUS-DP Slave*, то параметри зв'язку і розмір об'єму пам'яті буде подібним параметрам *PROFIBUS-DP Master*. Функцію веденого пристрою буде виконувати процесорний модуль *CPU214DP* (№VIPA 214-2BP02).

Контролери з'єднуються за допомогою за допомогою спеціального кабелю. Це пов'язано з тим, що шина *PROFIBUS* може забезпечувати

живлення польових пристроїв (датчиків або виконавчих механізмів). Але, можливе використання звичайної скрученої пари проводів. Якщо звернутись до опису інтерфейсу *RS-485* в процесорних модулях, то у обох модулів інтерфейси мають з'єднувачі типу *D-Sub (F)*. Таким чином з'єднувальний кабель буде з обох боків зі з'єднувачами типу *D-Sub (M)*. На рис. 3.18 [10] зображено зовнішній вигляд з'єднання з позначенням ліній зв'язку та живлення. Як бачимо на рис.3.18, в схемі підключення на кінцях шини використані термінальні резистори між лініями «А» та «В». Крім того, в лінях здійснено «підтягування» до ліній живлення, тобто використано метод поляризаційного зсуву для підвищення надійності зв'язку на великих (до 1000 м) відстанях у широкому діапазоні швидкостей. Але на невеликих відстанях можливо лише з'єднання пристроїв за допомогою лише сигнальних ліній без резисторів зсуву.

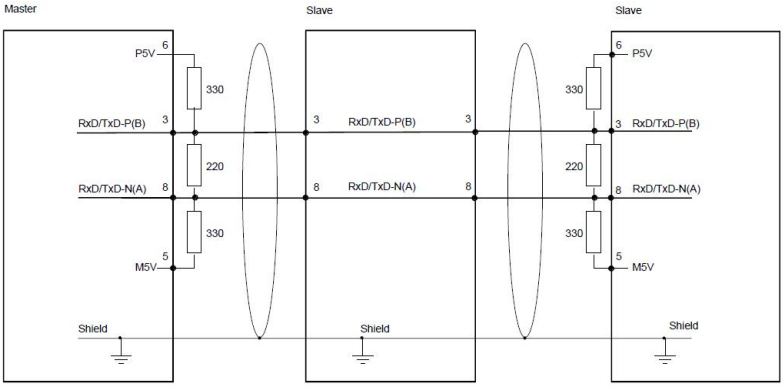


Рис. 3.18. Схема з'єднання пристроїв у шині *PROFIBUS*

Зазначимо, що існує залежність максимальної довжини лінії зв'язку від швидкості обміну для шини *PROFIBUS*. Параметри цієї залежності наведені в табл. 3.8.

Табл. 3.8 – Залежність максимальної довжини лінії зв'язку від швидкості обміну в шині *PROFIBUS*

Швидкість обміну, кбіт/с	Максимальна лінія зв'язку, м
9.6	1200
19.2	1200
187.5	1000
500.0	400
1500.0	200
3000.0	100
6000.0	100
12000.0	100

3.3.1. Апаратна структура PROFIBUS-мережі

Отже, створимо новий проект та відразу налаштуємо конфігурацію ресурсів для двох процесорних модулів. Процесорний модуль *CPU314ST/DPM* – буде виконувати функцію ведучого пристрою *PROFIBUS-DP Master*. Процесорний модуль *CPU214DP* – буде виконувати функцію веденого пристрою *PROFIBUS-DP Slave*. Одразу зазначимо, що контролери в мережі будуть обмінюватись такими параметрами: значенням аналогових каналів та байтами входів та виходів в обох ПЛК.

В програмі для конфігурування апаратних ресурсів створить станцію з ім'ям *VIPA314*. У вікні вибору серії ПЛК виберіть серію процесорів *VIPA SPEED7* і додайте модель з № VIPA 314-6CF02. До процесорного модуля додайте два сигнальні модуля – модуль аналогового введення *SM331* (№331-7KB01) та модуль аналогового виведення *SM332* (зам. №331-5HB01).

Одразу налаштуйте вхідні та вихідні аналогові канали сигнальних модулів. Модуль введення налаштуйте на оброблення уніфікованого сигналу постійної напруги *0…10V* на першому каналі. Модуль виведен-

ня налаштуйте на формування уніфікованого сигналу постійного струму *4…20mA* на першому каналі та уніфікованого сигналу постійної напруги *0…10V* на другому каналі. Також налаштуйте аналоговий вхід в субмодулі процесорного модуля для оброблення пасивного сигналу опору від термоперетворювача опору з НСХ *Pt100*. Для цього у вікні *Station-Offline---VIPA314* потрібно перейти до вкладення *SpeedBus* та відкрити на редагування рядок «*Slot 100*». Далі потрібно діяти подібно до того як це було описано у першому розділі для налаштувань ресурсів процесорного модуля *VIPA115*. Тобто, потрібно відкрити вікно *Module-Parameters* для налаштувань параметрів субмодулів. Спочатку у вкладенні *Parameters* треба активувати субмодулі процессорного модуля. Далі необхідно знайти рядок *channel 4:function* та в меню *Selection*, що випадає, вибрати рядок *Pt100 2-wire -200°C…850°C*. Інші налаштування залиште такими як вони є за умовчанням. Закрийте вікно *Module-Parameters* шляхом натиснення ЛКМ на команду *OK*.

Далі потрібно налаштувати інтерфейс *Ethernet* для з'єднання з ПЛК. Для цього потрібно у вікні *Station-Offline---VIPA314* перейти до вкладення *UR3* та відкрити на редагування рядок «*Slot 10*» з ім'ям *Speed7 Ethernet (CP343)*. Відкриється вікно *Properties CP343*, зображення якого подібно рис. 3.19. Відмітимо, що вікно *Properties CP343* для *VIPA314* має відмінність від вікна *Properties CP343* для *VIPA313*, яке зображено на рис 2.32. Ця відмінність полягає у наявності додаткових вкладень, які пов'язані з наявністю внутрішньої шини *SPEED7*.

Для налаштувань інтерфейсу *Ethernet* натисніть на кнопку *Properties Ethernet*. Відкриється вікно *Properties Ethernet-Interface* зі вкладенням *Parameters*, зображення якого подібно рис. 3.20. У відповідні поля введіть *IP*-адресу та маску локальної мережі.

Якщо в локальній мережі *Ethernet* використаний маршрутизатор, то поставте ЛКМ прапорець напроти рядка *Use router* та додайте його *IP*-адресу у відповідне поле. Закрийте послідовно вікна *Properties CP343* та

Properties Ethernet-Interface послідовним натисненням ЛКМ на кнопки ОК. В результаті повернетесь до вікна *Station-Offline---VIPA314*.

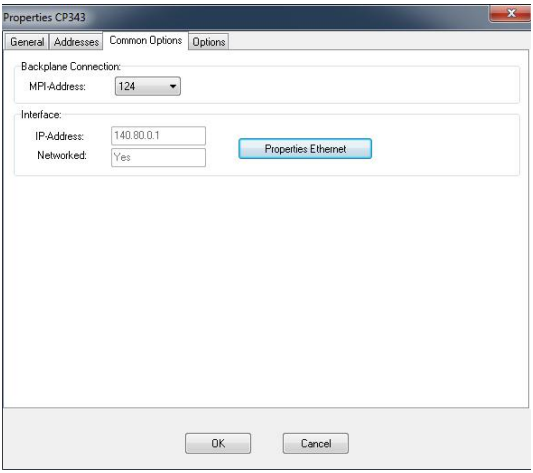


Рис. 3.19. Вікно налаштувань властивостей *CP343*

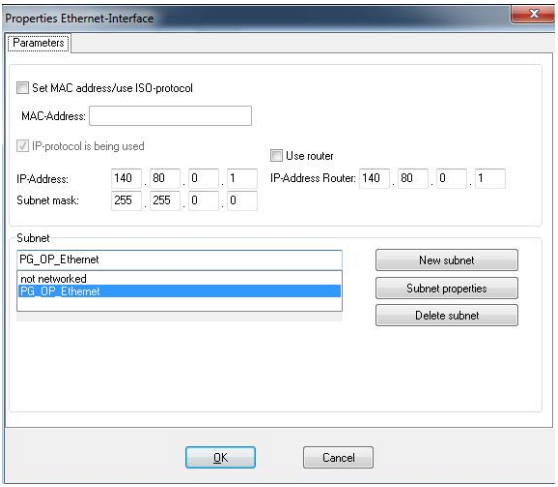


Рис. 3.20. Вікно налаштувань властивостей *Ethernet*

Відкрийте вікно налаштувань вбудованого модуля *PROFIBUS-DP*, яке викликається натисканням ЛКМ на рядок *X2* другого слоту з ім'ям *DP*. Відкриється вікно *DP-Child*, яке зображено на рис.3.21.

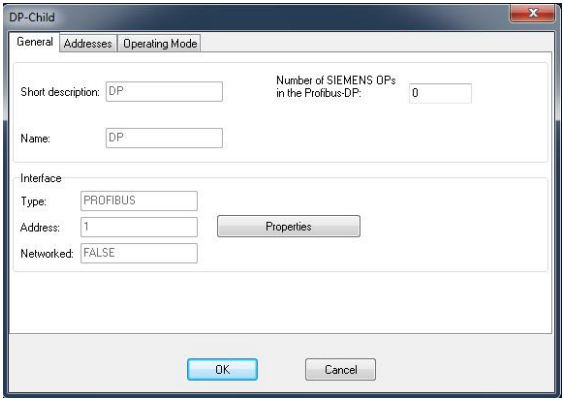


Рис. 3.21. Вікно налаштувань властивостей *PROFIBUS-DP*

Вікно *DP-Child* вміщує три вкладення: *General*, *Addresses* та *Operating Mode*. За умовчанням субмодуль *PROFIBUS-DP* налаштований на роботу в мережі як головний пристрій. За необхідністю можливе перемикання субмодулю у режим роботи в якості підлеглого пристрою у вкладенні *Operating Mode* шляхом встановлення меркера у відповідноме поле (*DP-Slave*). У вкладенні *Addresses* вказуються адреси пристроїв, але поля для налаштування неактивні для режиму ведучого пристрою. Крім того, за умовчанням активований принцип адресування *Selection from System* наявністю меркера у відповідному полі. Основні налаштування субмодуля *PROFIBUS-DP* здійснюються у вкладенні *General*. У вікні *DP-Child* (див. рис.3.21) ЛКМ натисніть на кнопку *Properties*. Відкриється вікно *Properties DP Master*. Адреса ведучого пристрою за умовчанням дорівнює «1». Але це можливе лише після створення мережі *PROFIBUS*. Для цього у вікні на рис.3.21 ЛКМ натисніть на кнопку *New*. Відкриється вікно *Properties Profibus* з активним

вкладенням *General*. Змінимо адресу ведучого пристрою у полі *No. Mastersystem* на інше значення, наприклад, на «2». Перейдіть у вкладення *Network Settings* для налаштування параметрів обміну. Залиште налаштування без змін. Закрийте послідовно усі вікна натисканням ЛКМ на кнопку *OK*. Після проведених дій з'явиться нове вікно, яке зображене на рис.3.22. У цьому вікні показана структура мережі *PROFIBUS* зі станцією *VIPA314*, яка виконує функцію головного пристрою та розміщена у першому рядку. Далі потрібно додати до мережі підлеглі пристрої. Це буде станція *VIPA214*. Для цього потрібно праворуч у каталозі знайти відповідний процесорний модуль. В дереві каталогу послідовно відкрийте вузли *Additional Field Devices DP, I/O, VIPA Gmbh, VIPA_CPU21x(VIPA_21x.gse)* та додайте процесорний модуль *VIPA_CPU21x*. Це означає, що до мережі *PROFIBUS* додається процесорний модуль *VIPA* моделі *System200V* з вбудованим субмодулем *DP Slave*. Відповідно, до апаратної конфігурації буде додано таргет-файл *VIPA_21x.gse*.

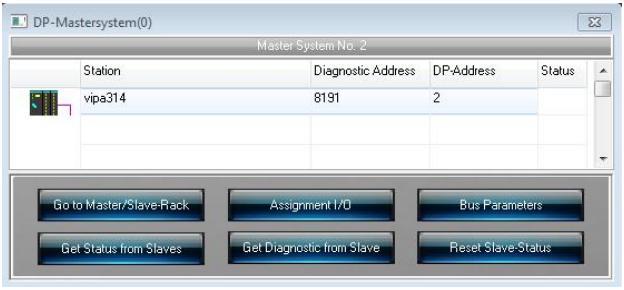


Рис. 3.22. Вікно зі структурою мережі *DP-Mastersystem*

Результат додавання веденого пристрою до мережі *PROFIBUS* зображений у на рис.3.23. Як видно з рис.3.23 адреса веденого пристрою «1». Нагадаємо, що при конфігуруванні *VIPA214* адреса веденого пристрою при налаштуванні дорівнювала «5». Тому подвійним кліком ЛКМ по рядку *VIPA_CPU21x* відкрийте вікно *Properties DP-Slave* з ак-

тивованим вкладенням *General* для налаштувань веденого пристрою. Замініть адресу веденого в *DP-Mastersystem* на потрібне значення у полі *DP-Address*. Інші налаштування залиште без змін.

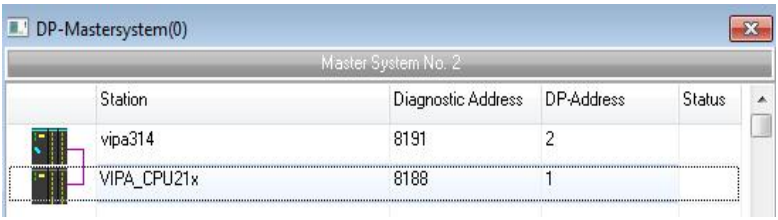


Рис. 3.23. Вікно мережі *DP-Mastersystem* з веденим пристроєм

Далі необхідно заповнити слоти сигнальними модулями, які входять до ПЛК *VIPA214DP* (див. рис. 2.19). Для цього потрібно ЛКМ виділити слот зі станцією *VIPA_CPU21x* (див. рис. 3.23) і за допомогою контекстного меню вибрати команду *Go to Master/Slave-Rack* для відкриття конфігурації веденого пристрою. Модулі потрібно брати послідовно з переліку в каталозі *Additional Field Devices DP→I/O→ VIPA Gmbh→VIPA_CPU21x(VIPA_21x.gse)→Module*. Таким чином отримаємо конфігурацію, яка зображена на рис. 3.24, яка подібна той, що зображена на рис. 2.19.

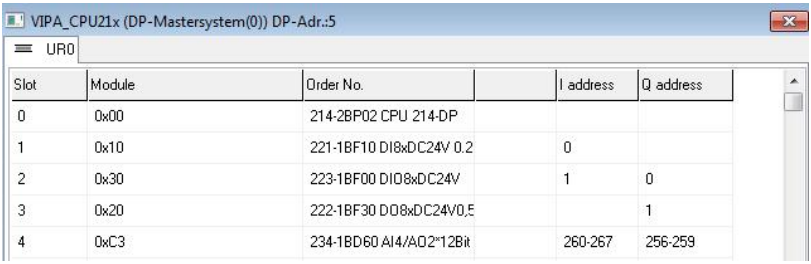


Рис. 3.24. Конфігурація веденого пристрою *VIPA_CPU21x*

Якщо вибрати команду *Show net-view* з переліку основного меню *Net-view*, то з'явиться графічне зображення мережі *PROFIBUS*, яке зображене на рис. 3.25. Інше з'єднання на рис. 3.25 означає наявність інтерфейсу *Ethernet* для програмування ПЛК та здійснення зв'язку з панеллю оператора для обміну даними.

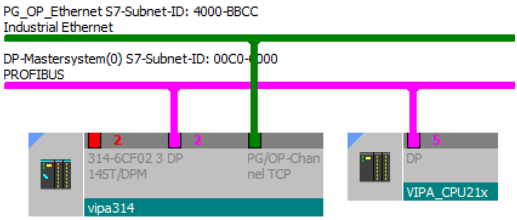


Рис. 3.25. Структура мережі *DP-Mastersystem*

Насамкінець, необхідно передати конфігурацію апаратних ресурсів ведучого та веденого пристроїв до ПЛК *VIPA314* та *VIPA214* відповідно.

3.3.2. Програмування головного пристрою в мережі PROFIBUS

Перед розробленням програмної частини ведучого пристрою для роботи у мережі *PROFIBUS* розглянемо порядок обміну між ведучим та веденим пристроєм. Спочатку ведучий пристрій робить перевірку відповідності параметрів мережі її дійсному стану, а саме тип пристрою, формат та довжина даних, а також кількість входів та виходів дозволяє забезпечити ефективний захист від помилок конфігурації. Крім того, ведучий пристрій може самостійно змінювати конфігурацію мережі. Якщо в статусі світлодіоду на лицевій панелі DE (обмін даними активний), то ведучий пристрій надсилає нові основні дані до веденого, то телеграма веденого передає останні вхідні дані ведучому.

Обмін даними між ведучим та веденим пристроями виконується циклічно з використанням буферів відправлення та приймання, подібно до обміну за протоколом *Modbus*. Але процес обміну поділяється на два етапи: цикл обміну безпосередньо по мережі *PROFIBUS* та цикл обміну

всередині веденого по внутрішньої *V*-шині контролера з сигнальними модулями. При цьому використовується спеціальна область периферійної пам'яті в процесорі (пам'ять для відображення входів та виходів *PII* та *PIQ*). Зауважимо, що цикли обміну незалежні один від одного, але розмір периферійної пам'яті повинен співпадати з пам'яттю буферів обміну. Для кожної моделі процесора існують обмеження на розмір буферів обміну. Так для *VIPA214* він становіть 64 байт для обох буферів, а для *VIPA314* – 1 кбайт (1024 байти) для обох буферів у статусі ведучого пристрою. Якщо *VIPA314* виконує функцію веденого, то розмір його буферної пам'яті складає 244 байти.

На рис. 3.26 зображено схему обміну даними між *VIPA314* (головний пристрій з адресою «2») та *VIPA214* (підлеглий пристрій з адресою «5»). На схемі зображено, що дані (значення маркерних слів MB0 та MB1) передаються через вбудовані в ПЛК субмодулі шини *PROFIBUS-DP* один одному. Для обміну по шині *PROFIBUS-DP* у кожному субмодулі також буде організовано область пам'яті для вихідних та вихідних даних розміром по два байти кожна. Ці області пам'яті будемо адресувати так: для *VIPA314* пам'ять вхідна – IB10, вихідна – QB20, а для *VIPA214* пам'ять вхідна – IB30, вихідна – QB40.

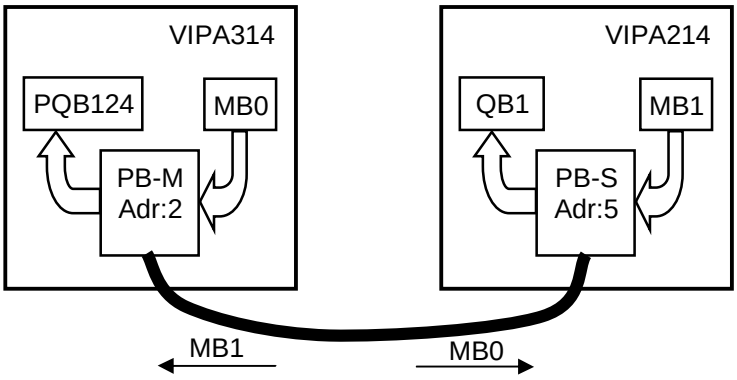


Рис. 3.26. Структурна схема з'єднання двох ПЛК в мережі *PROFIBUS*

Крім того, для роботи веденого пристрою потрібно вказати адреси для параметричних («800», довжиною 24 байти) та діагностичних даних («900», довжиною 6 байтів) й адресу для зчитування статусу даних *DP*-пристрою («1000», довжиною 2 байти). Усі ці зони пам'яті мають фіксований розмір, який визначається характеристиками конкретної моделі процесорного модуля із вбудованим субмодулем *PROFIBUS-DP*. Адреси призначаються автоматично в процесі апаратного конфігурування відповідного субмодуля.

Отже, необхідно провести додаткове налаштування субмодулів *PROFIBUS-DP*. Спочатку відкрийте вікно для налаштувань веденого пристрою, яке зображене на рис. 3.24. Викличте для налаштування процесорний модуль *VIPA CPU214DP*, який знаходиться у першому рядку таблиці з модулями. Відкриється вікно, яке зображене на рис. 3.27.

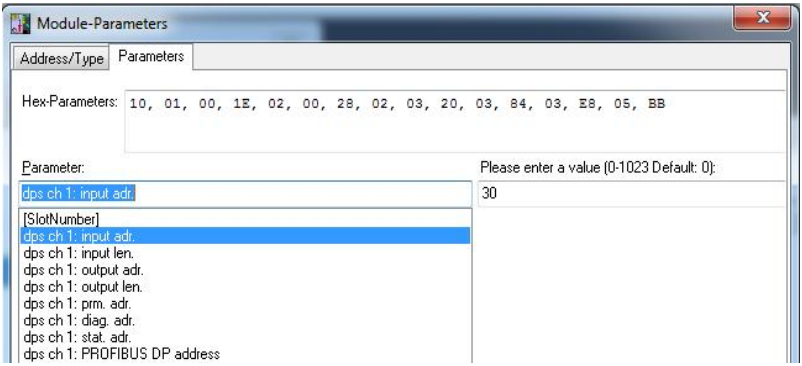


Рис. 3.27. Налаштування субмодуля *PROFIBUS* в ПЛК *CPU214DP*

Введіть потрібні значення до полів адрес вхідних та вихідних даних, кількість байтів для обміну та адреси і розмір пам'яті для параметричних, діагностичних даних та для інформації про статус обміну даними. Результат введення значень параметрів налаштувань субмодуля

PROFIBUS в ПЛК *VIPA CPU214DP* відображається у полі *Hex-Parameters* у вигляді *HEX*-коду (див. рис. 3.27). Цей код потім передається до процесорного модуля у вигляді апаратної конфігурації. Подібні налаштування потрібно зробити для головного *PROFIBUS*-пристрою у ПЛК *VIPA CPU314DP*.

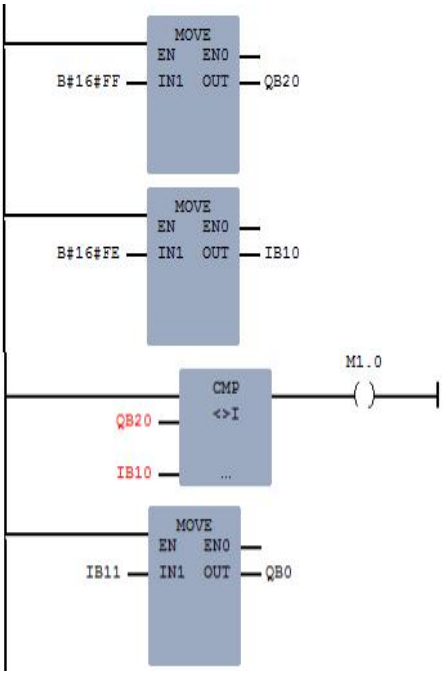


Рис. 3.28. Вміст блоку *OB1* у головному пристрої *PROFIBUS*

Програма користувача для ведучого пристрою ПЛК *VIPA314* буде складатись з двох задач, яки виконуються під керуванням організаційних блоків: блока *OB1* (основний цикл сканування) та блока *OB35* (виконання алгоритму з системним часовим переривання).

Вміст блоку *OB1* зображений на рис. 3.28. В цьому блоці реалізований алгоритм тестування шини *PROFIBUS* за допомогою контрольного байта (*QB20*) з пам'яті виходів ведучого пристрою, який порівнюється з вхідним байтом веденого пристрою (*IB20*). Останній ланцюг зчитує значення вхідного байта (*IB11*) із веденого та завантажує у вихідний байт (*QB0*) субмодуля дискретних виходів ПЛК *VIPA314*.

Вміст блоку *OB35* зображений на рис. 3.29. потрібен для системного переривання з циклом 100 *мс*. В блоці *OB35* буде здійснюватись

віднімання одиниці від значення у меркерному байті *MB0* кожні 100 *мс* та заноситись до вихідного байта з адресою *QB21* для відправлення до ведучого пристрою. На завершення завантажте програму користувача та оновить конфігурацію в ПЛК *VIPA314* з налаштованим головним пристроєм *PROFIBUS DP*.

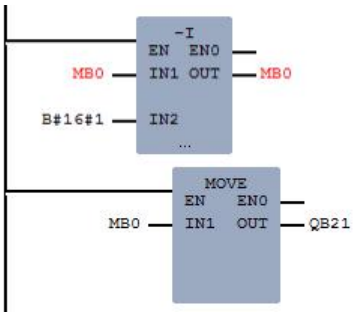


Рис. 3.29. Вміст блоку *OB35* у головному пристрої *PROFIBUS*

Програма користувача для веденого пристрою ПЛК *VIPA214* буде складатись теж з двох задач, яки виконуються під керуванням організаційних блоків: блока *OB1* та блока *OB35*. Програма в блоці *OB1* буде подібною до програми, яка зображена на рис. 3.28, але з деякими відмінностями. Так в блоці *OB1* зчитується вхідний байт з веденого пристрою до процесорного модулю для подальшого керування виходами сигнального модулю. На рис. 3.30 зображено вміст блоку *OB1*.

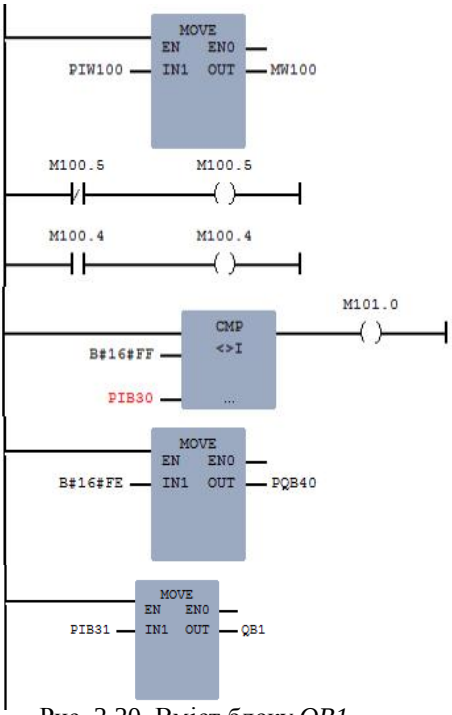


Рис. 3.30. Вміст блоку *OB1* у підлеглому пристрої *PROFIBUS*

В блоці *OB35* також реалізований лічильник з дода-

ванням одиниці та збереження значення для відправлення до ведучого пристрою. Програма в блоці *OB35* буде подібною до програми, яка зображена на рис. 3.29, але з відлік буде починатись від значення у двійковому значенні 0x00 до 0xff. Так, лічильник буде запам'ятовувати значення до меркерного байта *MB1*. Це значення буде пересилатись до *PQB41* (це вихідний байт веденого пристрою в ПЛК *VIPA214*). На рис. 3.31 зображено вміст блоку *OB35*.

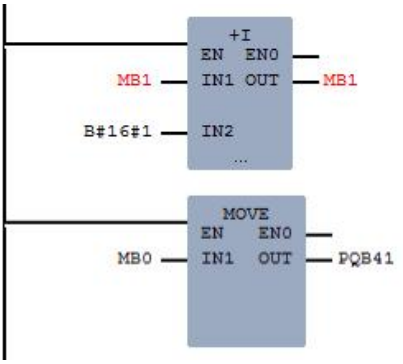


Рис. 3.31. Вміст блоку *OB35* у підлеглому пристрої *PROFIBUS*

Після заповнення вмісту блоків та апаратного конфігурування ПЛК *VIPA214* завантажить їх до контролера. Перевірте роботу мережі *PROFIBUS* шляхом одночасного підключення до контролерів. Причому зв'язок з ПЛК *VIPA314* з боку середовища *WinPLC V5* організуйте за допомогою інтерфейсу *Ethernet*, а до ПЛК *VIPA214* підключитесь за допомогою «зеленого» кабелю.

Таким чином ведучий та ведений пристрій будуть обмінюватись значенням лічильників та передавати їхнє значення один одному для керування дискретними виходами з циклом 100 *мс*.

Насамкінець додамо, що стан *PROFIBUS*-пристроїв можна перевірити із середовища апаратного конфігурування з вікна конфігурування *DP-Mastersystem(0)* шляхом натиснення ЛКМ на кнопки *Get Status from Slaves* та *Get Diagnostic from Slave* (див. рис. 3.22). Також можливо визначення процесу обміну даними в мережі *PROFIBUS* за допомогою спостереження за станом світло діодів на лицевій панелі процесорних модулів контролерів (див. рис. 1.16).

3.4 Реалізація обміну даними між ПЛК *VIPA* за допомогою глобальних блоків даних

Якщо потрібно організувати обмін даними між ПЛК *VIPA*, які не мають вбудованого модулю *PROFIBUS DP*, то ця задача вирішується за допомогою інтерфейсу *RS232* з протоколом *MPI* з використанням глобальних блоків даних. Увесь обмін налаштовується в конфігурації ресурсів ПЛК.

Отже налаштуємо обмін даними між ПЛК *VIPA115* та *VIPA313*. Нагадаємо, що дані процесорні модулі не мають вбудованого модулю *PROFIBUS DP*. Крім того, якщо в процесорному модулі *VIPA115* задіяний інтерфейс *RS485* для обміну даними, наприклад, з панеллю оператора за протоколом *Modbus*, то для обміну з іншим ПЛК можливе лише використання інтерфейсу *RS232* з протоколом *MPI*. З боку ПЛК *VIPA313* налаштуємо обмін для отримання стану входів *VIPA115*. Тобто, програма користувача в обох ПЛК буде виконуватись самостійно, але контролер *VIPA313* буде додатково отримувати інформацію про стан входів *VIPA115* за допомогою глобальних даних. Але з початку потрібно налаштувати мережу з протоколом *MPI*. Нагадаємо, що налаштування *MPI*-мережі здійснюється в основних властивостях процесорного модуля, як це зображено на рис. 2.3. Після натиснення ЛКМ на кнопку *Properties MPI* у зоні налаштування властивостей мережі. Далі, у вікні *Properties MPI-interface* необхідно ЛКМ натиснути на кнопку *New*. В результаті до вкладення *Parameters* вікна *Properties MPI-interface* буде додана *MPI*-мережа з ідентифікатором MPI(0) S7-Subnet-ID: 00CE-0000. Закрийте вікно *Properties MPI-interface* натисненням ЛКМ на кнопку *OK*. Залишилось змінити адресу ПЛК в *MPI*-мережі. Нехай *MPI*-адреса ПЛК *VIPA115* буде такою – «15». Одразу, у вікні мережної конфігурації з'явиться вузол *VIPA115* з підключенням до

MPI-мережі. Далі необхідно додати нову станцію до мережі. Це буде контролер *VIPA313* з *MPI*-адресою, яка дорівнює, наприклад, значенню «13». Результат додавання нового вузла до мережі зображений на рис. 3.32.

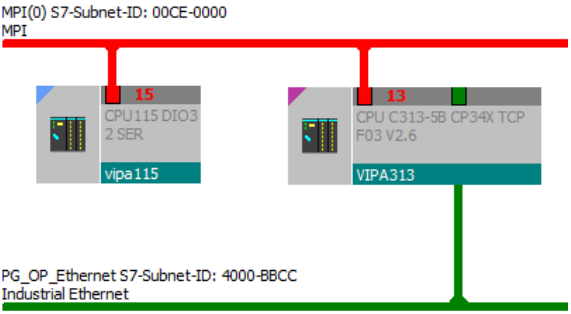


Рис. 3.32. Структура *MPI*-мережі

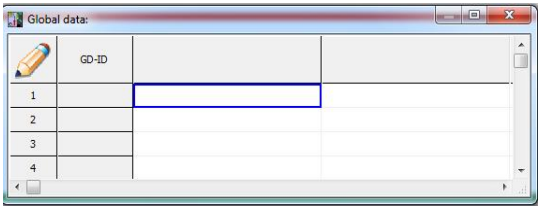


Рис. 3.33. Таблиця глобальних даних

Якщо ЛКМ на вузлі *VIPA115* або *VIPA313* викликати контекстне меню та вибрати команду *Define global data*, то буде створена таблиця глобальних даних, фрагмент якої зображений на рис. 3.33. У глобальній таблиці даних ви можете налаштувати обмін даними між процесорами *S7*, які підключені за допомогою мережі *MPI*. Додайте до стовпчиків таблиці вузли, які братимуть участь у обміні даними. Для цього використайте контекстне меню з командою *Insert PLC column*. Таблиця містить стовпець для кожного вузла PLC – *VIPA115* та *VIPA313*, де потрібно ввести адреси, які будуть ідентифікаторами в процесі обміну даними. Нехай необхідно

обмінюватись станом дискретних входів між процесорними модулями. Так стан дискретних входів *VIPA115 IB0* буде передаватись до меркерного байта *MB0* в пам'яті *VIPA313*, а стан дискретних входів *VIPA313 IB124* буде передаватись до меркерного байта MB124 в пам'яті *VIPA115*. Результат налаштування зображений на рис. 3.34.

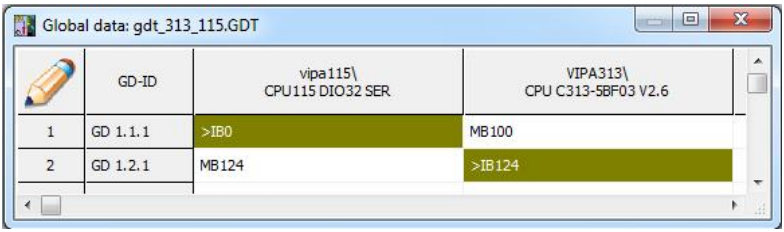


Рис. 3.34. Структура таблиці глобальних даних

Причому, для визначення джерела та отримувача даних в контекстному меню є спеціальні команди: *Change the cell to a sender* та *Change the cell to a receiver*. Також в якості операндів може бути використані елементи блоків даних, які належать до програм користувача в ПЛК. Система самостійно призначає внутрішній ідентифікатор для пар операндів в обох ПЛК, яке знаходиться у стовпчику *GD-ID*. Причому стовпчик *GD-ID* буде заповнений лише після виконання команди компілювання глобального блока даних. Це команда *Compile, create configuration data and save the GDT-File* з контекстного меню. Далі потрібно передати таблицю глобальних даних до відповідних ПЛК за допомогою команди *Send Global-data to active PLC* або *Send Global-data to all PLCs.*

Для додавання вузла до мережі обміну потрібно за допомогою контекстного меню вибрати команду *Insert PLC column*. Зауважимо, що обмін можливий між 15 вузлами в *MPI*-мережі. Тобто, глобальна таблиця може містити не більше 15 вузлів ПЛК.

До рядків з операндами можливе додавання спеціальних рядків:

- рядок «*GST*» (*Global status row*) – операнд зі глобальним ста-

ном. Він визначається як загальна сума , яка відповідає операції *OR* всіх стовпчиків стану таблиці. Ця адреса повинна бути адресою з двома словами;

- рядок «GDS» (*Status row data packet*) – у цьому рядку можливо ввести двослівну адресу для кожного вузла, який бере участь у передачі даних;
- рядок «SR» (*Scan rate*) – визначає частоту сканування для кожного пакета даних в процесі обміну. Може приймати значення від 1 до 255 одиниць.

Швидкість сканування для джерела повинна задовольняти наступним умовам, щоб переконатись, що навантаження на комунікаційну ПЛК зменшується наскільки це можливо:

Scan rate * cycle time > = 60 мс.

Коли створюється глобальна таблиця даних, компілятор автоматично створює різні схеми *GD*. Нижче наведений список основних правил, які застосовуються до цього процесу:

- у глобальному блоці передачі даних може бути задіяно максимум 15 процесорних модулів;
- допустимі такі операнди: входи, виходи, слова таймерів, таймери (лише як відправники), лічильники (лише як відправники), дані з блоків даних;
- довжина байтів області передавання та приймання повинна бути однаковою. Це означає, що якщо центральний процесор передає 10 байт, то довжина області прийому приймаючого центрального процесора повинна також становити 10 байт;
- максимальна довжина даних пакету *GD* становить 32 байти. У цьому випадку доступно 22 байта для фактичної інформації. Для заголовка блоку потрібно 8 байт, і для кожного об'єкта *GD* необхідні два байти. Коли для передачі даних з різних адрес-

них областей використовується пакет *GD*, то кількість байтів, доступних для фактичної інформації, зменшується на два байти на кожну область адреси. Зверніть увагу, що для типу даних *BOOL* також потрібно 1 байт за адресою;

- у рамках схеми *GD* процесор може отримувати максимум один пакет даних *GD* і надсилати один пакет даних *GD*;
- процесор може передавати та отримувати *GD*-пакети в максимум чотирьох схемах *GD*;
- процесори дозволяють сканувати значення в діапазоні від 1 до 255 одиниць. Значення за замовчуванням для швидкості сканування залежить від швидкості обробки центрального процесору;
- процесори серії *S7-300* не підтримують передачу даних, керованих за подіями, за допомогою *SFC*;
- максимальна кількість схем *GD* становить 16.

Насамкінець зауважимо, що фізично процесорні модулі з'єднують за допомогою «зеленого» кабелю від *VIPA*, але можливе й з'єднання за допомогою стандартного кабелю *PROFIBUS* (див. рис. 3.18).

РОЗДІЛ 4
РОЗРОБЛЕННЯ ППЗ ДЛЯ СИСТЕМИ УПРАВЛІННЯ УСТАНОВКОЮ ЗМІШУВАННЯ РЕЧОВИН НА ЗАСОБАХ *VIPA*

Для закріплення навичок з розроблення ППЗ в середовищі *WinPLC V5* в розділі буде наведено приклад розроблення проекту управління реактором або ємністю для змішування.

Наведений в цьому підрозділі приклад демонструє, як реалізувати керування установкою, в який здійснюється дискретно-безперервний технологічний процес змішування двох речовин. Схема установки зображена на рис. 4.1. На пульті керування установкою розміщені кнопки з фіксуванням та індикатори стану виконавчих пристроїв.

Алгоритм роботи установки буде таким. Після включення живлення установки та системи керування програма користувача в ПЛК перевіряє стан датчиків, виконавчих пристроїв (насоса, клапана та мотора мішалки). Якщо немає аварії, ємність порожня, клапан на вихідному трубопроводі закритий та мотори насосів вимкнені, то деблокується мотор мішалки. За допомогою кнопок «Старт» та «Стоп» здійснюється керування мотором мішалки. Також передбачена сигналізація стану мішалки (аварія, робота або очікування вмикання).

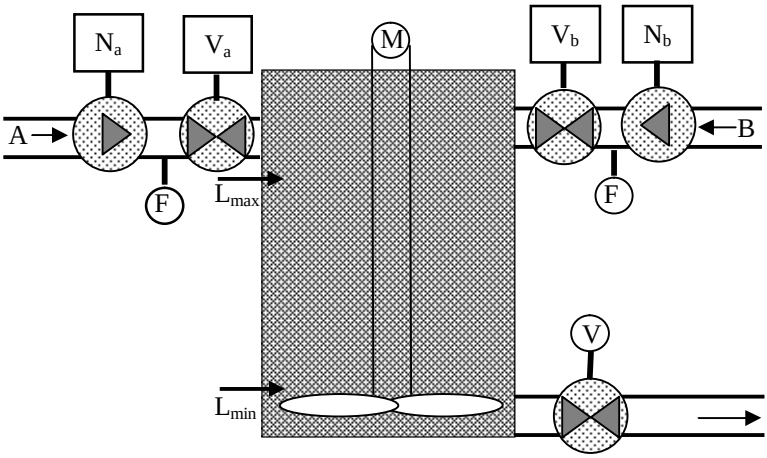


Рис. 4.1. Схема реактора з мішалкою

Розроблення ППЗ почнемо з формування структури програми. Процес виготовлення розчину можна розділити на таки етапи:

- заповнення ємності речовиною «А»;
- заповнення ємності речовиною «В»;
- перемішування розчину;
- злів готового розчину.

Кожен з трубопроводів, по яких подаються речовини «А» та «В»,

повинен виконувати таки умови:

- у вхідних трубопроводах знаходяться контактні датчики наявності витрати;
- включення живлення насоса та відкриття клапана повинно бути заблоковано, якщо спрацьовує датчик максимального рівня, тобто це означає, що ємність заповнена;
- включення живлення насоса повинно бути заблоковано, якщо зливний вентиль відкритий, тобто йде процес зливання готового продукту із ємності;
- впускний клапан може бути відкритим не раніше, ніж через 1 секунду після запуску насоса;
- не пізніше, ніж через одну секунду після зупинки насоса (сигнал датчика витрати) вентилі повинні бути закриті, щоб перешкодити витіканню речовини з трубопроводу в ємність;
- запуск насоса контролюється за часом, тобто протягом 5 с після запуску датчик витрати повинен дати сигнал про її наявність;
- насоси повинні бути в найкоротший час відключені, якщо при роботі насосів датчики витрати не повідомляють про її наявність;
- кількість запусків насосів повинна рахуватись (інтервал технічного обслуговування).

Обладнання ємності для змішування складається з двигуна мішалки та датчиків рівня і виконує таки умови:

- включення двигуна мішалки повинно бути заблоковано, якщо вимірювач рівня показує, що рівень опустився нижче мінімально допустимого рівня;
- двигун змішувача подає сигнал після досягнення номінальної

швидкості. Якщо цей сигнал не отриманий протягом 10 с після запуску двигуна, двигун повинен бути відключений;

- кількість запусків двигуна змішувача має рахуватись (інтервал обслуговування);
- у ємності для змішування повинні бути встановлені два контактних датчика рівня типу NC.

На випускному трубопроводі встановлений клапан, який керується оператором та відкритий до отримання сигналу про те, що ємність порожня.

В установці передбачено керування з боку оператора за допомогою пульта, на якому встановлені кнопки старту та зупинки, індикатори стану установки та аварійний вимикач живлення всій установки.

В якості керуючого пристрою оберемо, наприклад, ПЛК *VIPA313SC*. Тому подібно до попереднього розділу створимо спочатку апаратну конфігурацію контролера. При цьому необхідно врахувати кількість зовнішніх елементів установки, тобто вхідних сигналів від датчиків та кнопок керування та вихідних сигналів для керування виконавчими пристроями. Далі, виходячи з умов завдання розробимо структуру програми користувача, яка зображена на рис. 4.2.

В табл. 4.1, 4.2 та 4.3 наведені символьні імена змінних для керування насосами (для речовин «А» і «В») та мішалкою, в табл. 4.4 – символьні імена датчиків, в табл. 4.5 – символьні імена для зливного трубопроводу, в табл. 4.6 – символьні імена для інших потреб (основні структурні компоненти проекту). Відмітимо, що абсолютне адресування фізичних входів та виходів здійснено без врахування апаратної структури контролера.

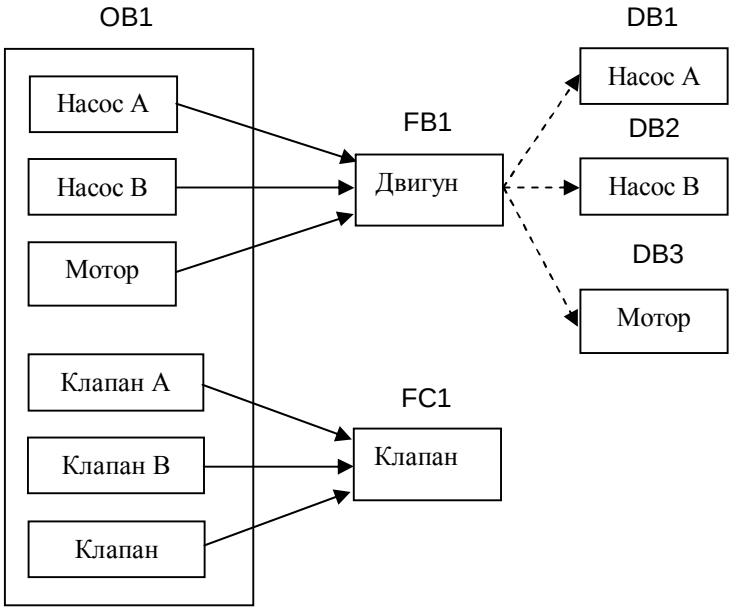


Рис. 4.2. Структура програми користувача

Таблиця 4.1 – Символьні імена для керування насосом А

Символьне ім’я	Адреса	Тип даних	Коментар
Pump_A_Start	I 0.0	BOOL	кнопка запуску насоса А
Pump_A_Stop	I 0.1	BOOL	кнопка зупинки насоса А
Flow_A	I 0.2	BOOL	датчик наявності потоку А
Valve_A	Q 1.0	BOOL	керування клапаном А
Pump_A_ONOFF	Q 1.1	BOOL	стан насосу А
Pump_A	Q 1.2	BOOL	керування насосом А
Pump_A_Fault	Q 1.3	BOOL	несправність насоса А
Pump_A_Servis	Q 1.4	BOOL	необхідне ТО для насоса А

Таблиця 4.2 – Символьні імена для керування насосом В

Символьне ім'я	Адреса	Тип даних	Коментар
Pump_B_Start	I 0.3	BOOL	кнопка запуску насоса В
Pump_B_Stop	I 0.4	BOOL	кнопка зупинки насоса В
Flow_B	I 0.5	BOOL	датчик наявності потоку В
Valve_B	Q 2.0	BOOL	керування клапаном В
Pump_B_ONOFF	Q 2.1	BOOL	стан насоса В
Pump_B	Q 2.2	BOOL	керування насосом В
Pump_B_Fault	Q 2.3	BOOL	несправність насоса В
Pump_B_Servis	Q 2.4	BOOL	необхідне ТО для насоса В

Таблиця 4.3 – Символьні імена для керування мотором мішалки

Символьне ім'я	Адреса	Тип даних	Коментар
Motor_Work	I 0.6	BOOL	датчик роботи мотора мішалки
Motor_Start	I 0.7	BOOL	кнопка запуска мотора мішалки
Motor_Stop	I 1.0	BOOL	кнопка зупинки мотора мішалки
Motor	Q 1.5	BOOL	керування мотором мішалки
Motor_ONOFF	Q 1.6	BOOL	стан мішалки
Motor_Fault	Q 2.5	BOOL	несправність мотора мішалки
Motor_Servis	Q 2.6	BOOL	необхідне ТО для мотора мішалки

Таблиця 4.4 – Символьні імена для датчиків

Символьне ім'я	Адреса	Тип даних	Коментар
Sensor_MAX	I 1.1	BOOL	датчик максимального рівня
Sensor_MIN	I 1.2	BOOL	датчик мінімального рівня
L_MAX	Q 1.7	BOOL	індикатор верхнього рівня
L_MIN	Q 2.7	BOOL	індикатор нижнього рівня

Таблиця 4.5 – Символьні імена для зливного трубопроводу

Символьне ім'я	Адреса	Тип даних	Коментар
Valve_Start	I 1.3	BOOL	кнопка відкриття клапану зливу
Valve_Stop	I 1.4	BOOL	кнопка закриття клапану зливу
VALVE	Q 3.0	BOOL	керування клапаном зливу
VALVE_ONOFF	Q 3.1	BOOL	стан клапана зливу

Таблиця 4.6 – Символьні імена для загального використання

Символьне ім'я	Адреса	Тип даних	Коментар
DB_A	DB 1	DB1	блок даних лінії А
DB_B	DB 2	DB2	блок даних лінії В
DB_M	DB 3	DB3	блок даних мішалки
Alarm_OFF	I 1.5	BOOL	загальне вимикання установки
m_volume	M 0.0	BOOL	аналіз стану датчиків рівня

Наступним кроком створимо основні структурні компоненти проекту, які доповнять організаційний блок OB1. Це будуть функціональні блок FB та функція FC, які будуть викликатись з організаційного блоку OB1 сумісно з їхніми блоками даних DB. Зауважимо, що ці елементи потрібно створити раніше ніж вони будуть викликатись з організаційного блоку OB1.

Отже, після створення функціонального блоку для керування вихідними пристроями вхідного трубопроводу (лінії «А» або «В») заповнимо його. Для обох вхідних трубопроводів цей блок буде один, але викликатись він буде зі своїм блоком даних. Для ФБ існують умови деблокування електродвигуна насоса, які пов'язані зі станом кнопок керування та своєчасністю активування сигналу квітування від датчика ная-

вності потоку. Причому сигнал деблокування зберігається в пам'яті для тимчасових даних (*L–Stack*) блока OB1, який постійно оновлюється відповідно до циклу сканування ПЛК. Це будуть сигнали з символьними іменами Pump_Unlock (деблокування насоса) та Valve_Unlock (деблокування клапана), які логічно пов'язані із сигналами керування (кнопками на пульті). Якщо квитанція від датчика наявності потоку не надійшла своєчасно, то електродвигуни насосів вимикаються. Дана умова реалізується за допомогою таймерів, які запускаються після старту насоса. Якщо кнопка старту активована і немає заборони на вмикання насос вмикається до моменту активування кнопки зупинення. На рис. 4.3 зображено ФБ керування насосом лінії подавання сировини з позначеними вхідними та вихідними параметрами. В блоці аналізується стан елементів керування від пульта, датчиків та виробляється керуючий сигнал для управління електроприводом насоса. Крім того, блок формує сигнали для індикації стану насоса та наявність несправності (для індикаторів на пульті керування), рахує кількість вмикань для сповіщення про необхідність проведення технічного обслуговування. В табл. 4.7 наведено перелік параметрів блока керування насосом. Нагадаємо, що інтерфейс функціонального блока складається з вхідних (IN), вихідних (OUT) та прохідних параметрів (IN_OUT). Крім того, в блоці є локальні змінні так званого *L*-стека. Це статичні змінні (позначені символом S), які зберігаються у відповідному блоці даних та тимчасові змінні

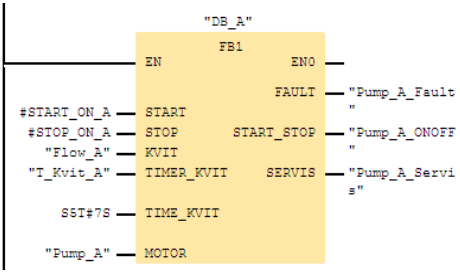


Рис. 4.3. ФБ керування насосом

(позначені символом T), які не зберігаються до наступного циклу сканування. У прикладі тимчасові змінні не використовуються

Кодова частина блоку складається з ланцюгів аналізу станів вхідних змінних та формування вихідних змінних. Це таки ланцюги:

- запуск з підхватом або зупинка;
- контроль запуску (таймер) приводу насосу;
- формування сигналів для індикатора стану насоса;
- контроль кількості стартів з формуванням сигналу ТО.

Таблиця 4.7 – Параметри блока керування насосом

Клас параметра	Символьне ім'я	Тип даних	Коментар
in	START	BOOL	кнопка СТАРТ
in	STOP	BOOL	кнопка СТОП
in	KVIT	BOOL	сигнал квітування
in	TIMER_KVIT	TIMER	таймер квітування
in	TIME_KVIT	S5TIME	час квітування
out	FAULT	BOOL	несправність мотора
out	START_STOP	BOOL	індикатор роботи мотора
out	SERVIS	BOOL	індикатор сервісу
in_out	MOTOR	BOOL	стан мотора
var S	TIME_BIN	WORD	час квітування
var S	TIME_BCD	WORD	час квітування
var S	STARTS	INT	кількість стартів мотора
var S	TRIG_START	BOOL	фронт запуску мотора
temp	-	-	-

Розглянемо докладніше склад ланцюгів ФБ, які зображені на рис. 4.4. Отже перший ланцюг перевіряє стан кнопок керування електроприводом насоса та формує сигнал для старту таймера. Якщо сигнал квітування не надійде вчасно, то виробляється сигнал несправності та електропривод насоса вимикається. В протилежному випадку насос вмикається, сигнал несправності скидається і лічильник додає одиницю до кількості стартів. Коли кількість запусків досягне значення 500, вмикається індикатор, який інформує про необхідність проведення технічного обслуговування.

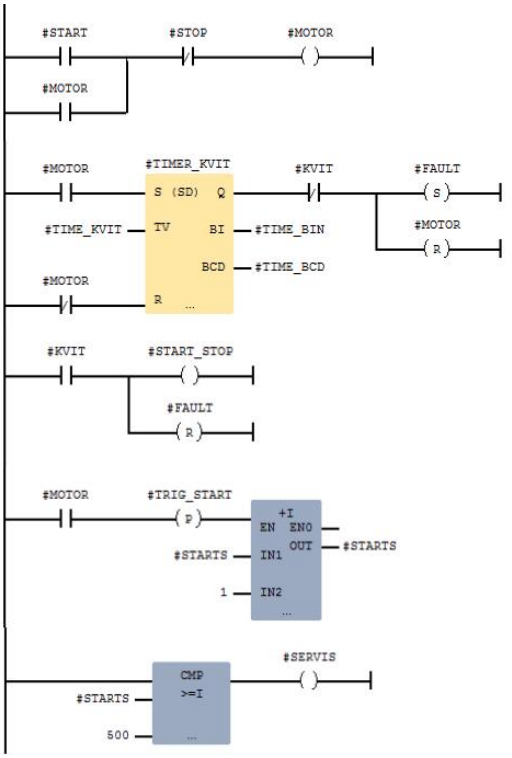


Рис. 4.4. Кодова складова ФБ для керування насосом

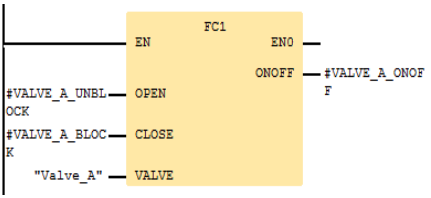


Рис. 4.5. Функція керування клапаном (лінія «А»)

Функція керування клапаном буде теж одна для усіх клапанів, але в кожному випадку її виклику вона отримає сигнали відкриття та закриття клапану ліній подавання сировині. Як і раніше створить у менеджері проекту функцію. Далі заповніть її параметри та складіть її кодову складову. На рис. 4.5 зображена функція зі входами та виходом. В табл. 4.8 подано перелік параметрів функції керування клапаном.

Таблиця 4.8 – Параметри функції керування клапаном

Клас параметра	Символьне ім’я	Тип даних	Коментар
in	OPEN	BOOL	сигнал для відкриття клапана
in	CLOSE	BOOL	сигнал для закриття клапану
out	ONOFF	BOOL	стан клапану
in_out	VALVE	BOOL	керування клапаном
temp	-	-	-

Кодова частина блоку складається з ланцюгів аналізу станів вхідних змінних та формування вихідної змінної. Відмітимо, що відкриття клапану можливе за умовою відсутності сигналу блокування від блоку OB1. Стан змінної блокування зберігається у *L*-стеку цього блока. Зображення кодової частини функції подано на рис. 4.6.

Після створення ФБ та Ф розробимо організаційний блок OB1. Це основний блок програми користувача, в якому закладений алгоритм роботи установки в цілому. В цьому блоці аналізується стан вхідних сигналів від датчиків і кнопок та блоці викликаються ФБ та Ф для

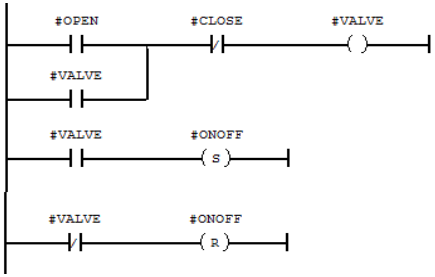


Рис. 4.6. Кодова складова функції керування клапаном

керування вихідними пристроями. Якщо викликається ФБ, то в блоці вказується які вхідні параметри з блоків даних він обробляє. Склад змінних, які обробляються у блоці OB1 поданий у табл. 4.9. Перелік змінних починається з системних змінних блока, які визначають режими оброблення та слідкують за циклом сканування. Усі системні змінні займають усього 20 байтів пам'яті. Локальні змінні блока оновлюються кожен цикл сканування. Символьні імена змінних програми користувача розміщені в табл. 4.10.

Таблиця 4.9 – Параметри організаційного блоку OB1

Клас параметра	Символьне ім'я	Тип даних	Коментар
temp	OB1_EV_CLASS	BYTE	Bits 0-3 = 1 (Coming event), its 4-7 = 1 (Event class 1)
temp	OB1_SCAN_1	BYTE	1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
temp	OB1_PRIORITY	BYTE	1 (Priority of 1 is lowest)
temp	OB1_OB_NUMBR	BYTE	1 (Organization block 1, OB1)
temp	OB1_RESERVED_1	BYTE	Reserved for system
temp	OB1_RESERVED_2	BYTE	Reserved for system
temp	OB1_PREV_CYCLE	INT	Cycle time of previous OB1 scan (ms)

Продовження табл. 4.9

temp	OB1_MIN_CYCLE	INT	Min. cycle time of OB1 (ms)
temp	OB1_MAX_CYCLE	INT	Max. cycle time of OB1 (ms)
temp	OB1_DATE_TIME	DT	Date and time OB1 started
temp	PUMP_A_UNBLOCK	BOOL	деблокування насоса A
temp	VALVE_A_UNBLOCK	BOOL	деблокування клапана A
temp	VALVE_A_BLOCK	BOOL	блокування клапана A
temp	START_ON_A	BOOL	запуск лінії A
temp	STOP_ON_A	BOOL	зупинка лінії A
temp	VALVE_A_ONOFF	BOOL	стан клапана A
temp	PUMP_B_UNBLOCK	BOOL	деблокування насоса B
temp	VALVE_B_UNBLOCK	BOOL	деблокування клапана B
temp	VALVE_B_BLOCK	BOOL	блокування клапана B
temp	START_ON_B	BOOL	запуск лінії B
temp	STOP_ON_B	BOOL	зупинка B
temp	VALVE_B_ONOFF	BOOL	стан клапана B
temp	MOTOR_UNBLOCK	BOOL	деблокування мішалки
temp	START_ON_MOTOR	BOOL	запуск мішалки
temp	STOP_ON_MOTOR	BOOL	зупинка мішалки
temp	VALVE_OPEN	BOOL	клапан зливу відкрити
temp	VALVE_CLOSE	BOOL	клапан зливу закрити

Таблиця 4.10 – Символьні імена програми користувача

Символьне ім’я	Адреса	Тип даних	Коментар
DB_A	DB 1	DB1	блок даних лінії А
DB_B	DB 2	DB2	блок даних лінії В
DB_M	DB 3	DB3	блок даних мішалки
Pump_A_Start	I 0.0	BOOL	кнопка пуску насоса А
Pump_A_Stop	I 0.1	BOOL	кнопка зупинки насоса А
Flow_A	I 0.2	BOOL	датчик наявності потоку А
Pump_B_Start	I 0.3	BOOL	кнопка пуску насоса В
Pump_B_Stop	I 0.4	BOOL	кнопка зупинки насоса В
Flow_B	I 0.5	BOOL	датчик наявності потоку В
Motor_Work	I 0.6	BOOL	датчик роботи мотора мішалки
Motor_Start	I 0.7	BOOL	кнопка пуску мотора мішалки
Motor_Stop	I 1.0	BOOL	кнопка зупинки мотора мішалки
Sensor_MAX	I 1.1	BOOL	датчик максимального рівня
Sensor_MIN	I 1.2	BOOL	датчик мінімального рівня
Valve_Start	I 1.3	BOOL	кнопка відкриття клапану зливу
Valve_Stop	I 1.4	BOOL	кнопка закриття клапану зливу
Alarm_OFF	I 1.5	BOOL	загальне вимикання установки
m_volume	M0.0	BOOL	аналіз стану датчиків рівня
Valve_A	Q 1.0	BOOL	керування клапаном А
Pump_A_ONOFF	Q 1.1	BOOL	стан насосу А
Pump_A	Q 1.2	BOOL	керування насосом А
Pump_A_Fault	Q 1.3	BOOL	несправність насоса А
Pump_A_Servis	Q 1.4	BOOL	необхідне ТО для насоса А
Motor	Q 1.5	BOOL	керування мотором мішалки
Motor_ONOFF	Q 1.6	BOOL	стан мішалки

Продовження табл. 4.10

L_MAX	Q 1.7	BOOL	індикатор верхнього рівня
Valve_B	Q 2.0	BOOL	керування клапаном В
Pump_B_ONOFF	Q 2.1	BOOL	стан насоса В
Pump_B	Q 2.2	BOOL	керування насосом В
Pump_B_Fault	Q 2.3	BOOL	несправність насоса В
Pump_B_Servis	Q 2.4	BOOL	необхідне ТО для насоса В
Motor_Fault	Q 2.5	BOOL	несправність мотора мішалки
Motor_Servis	Q 2.6	BOOL	необхідне ТО для мотора мішалки
L_MIN	Q 2.7	BOOL	індикатор нижнього рівня
VALVE	Q 3.0	BOOL	керування клапаном зливу
VALVE_ONOFF	Q 3.1	BOOL	стан клапана зливу
T_Kvit_A	T 13	TIMER	таймер квітування про наявність потоку А
T_Kvit_B	T 14	TIMER	таймер квітування про наявність потоку В
T_Kvit_M	T 15	TIMER	таймер квітування про роботу мішалки

Фрагмент кодової складової організаційного блока OB1, яка керує подаванням до ємності речовини з лінії «А» зображена на рис. 4.7. Керування процесом подавання речовини з лінії «В» подібно до фрагменту на рис.4.7, але з власними параметрами. В першому ланцюгу реалізовано формування сигналу деблокування роботи насоса. Далі аналізується стан кнопок керування насосом на пульті. Отримані результати логічних операцій є умовами для виконання функціонального блоку, який викликається зі своїм блоком даних. Наступні ланцюги формують сигнали для керування клапаном за допомогою функції. Аналогічно працює частина коду щодо лінії «В» та щодо керування мішалкою.

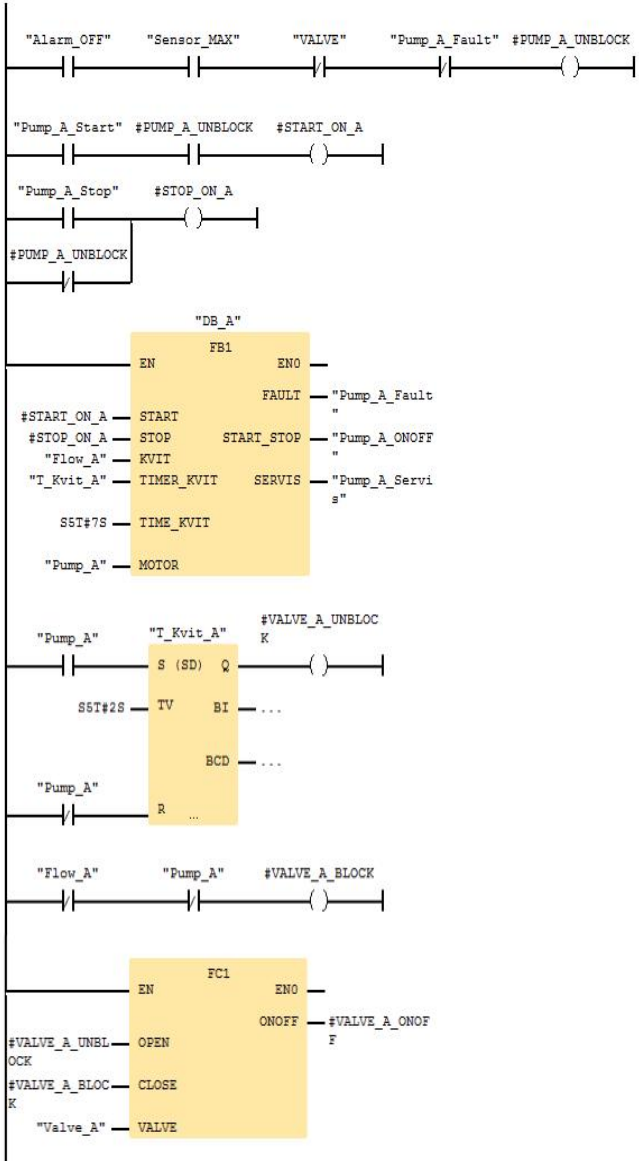


Рис. 4.7. Фрагмент коду керування обладнанням лінії «А» у блоці OB1

Але для вмикання мішалки додається ще одна умова, яка оцінює наявність речовини у ємності. Це ланцюг, подібний до раніше створеного, зображеного на рис. 2.6. Далі РЛО оцінювання стану дискретних датчиків рівня враховується в умові деблокування роботи мішалки. Фрагмент блока з ланцюгами зображений на рис. 4.8. Останній етап роботи установки – це зливання готового продукту. Фрагмент керування клапаном зливу зображений на рис. 4.9. В даній частині блоку теж враховується наявність рідини в ємності та використаний SR-тригер, який блокує відкриття клапану зливу, якщо наявна одночасна присутність сигналів відкриття та закриття.

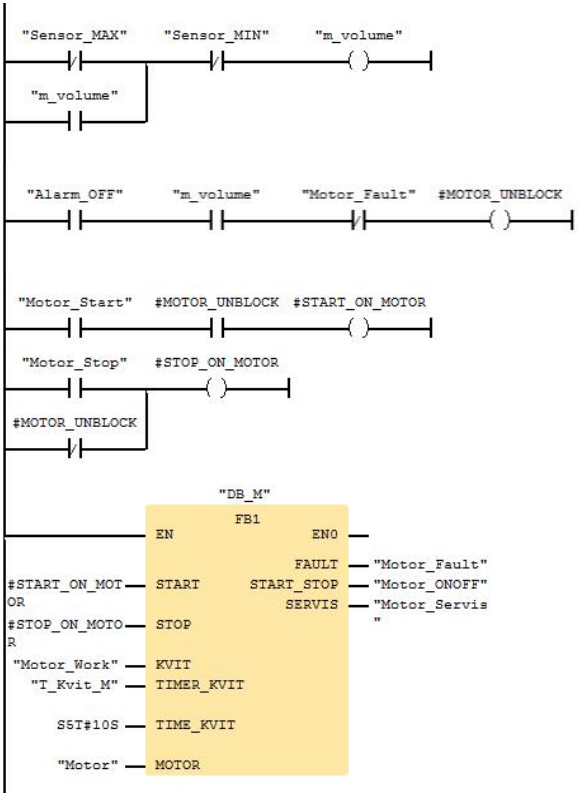


Рис. 4.8. Фрагмент коду керування мішалкою у блоці OB1

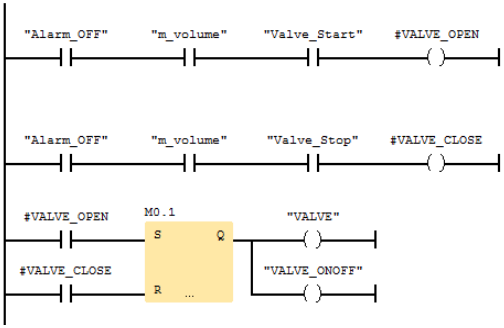


Рис. 4.9. Фрагмент коду керування клапаном зливу у блоці OB1

На завершення розділу нагадаємо, що в установці використано дискретні датчики рівня. Але, якщо в установці крім дискретних датчиків використаний аналоговий датчик рівня з струмовим сигналом 4…20 мА, то в програму користувача необхідно внести деякі зміни. Насамперед, це перетворення сигналу від датчика в значення фізичного параметру. Для вирішення цього завдання можливо використання спеціальної функції *FC105*, яка знаходиться в бібліотечному каталозі стандартних функцій у переліку з ім'ям *Analog*. За допомогою цієї функції можливе перетворення вхідного аналогового сигналу у значення фізичного параметру. Подібне перетворення було розглянуто у п. 2.4. під час створення ППЗ для установки гарячого водопостачання. Фрагмент програми блока, в якому реалізовано перетворення вхідного аналогового параметра, зображений на рис. 2.25. Але в блоках порівняння необхідно вказати реальні значення рівня для формування сигналів для деблокування виконавчих пристроїв. Приклад перетворення аналогового сигналу у фізичний параметр зображений на рис. 4.10. Адреса периферійного входу сигнального модуля визначається виходячи з апаратної конфігурації контролера. Наприклад, перший вхід аналогового сигнального модуля *AIO234* (зак. №234-1BD60) позначається адресою *PIW256*. Якщо використати *ПЛК VIPA313SC* (зак. №313-5BF03-0AB0), то адреса першого аналогового каналу субмодуля для підключення термо-

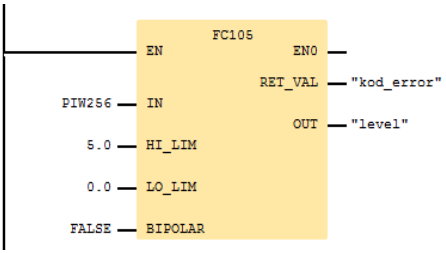
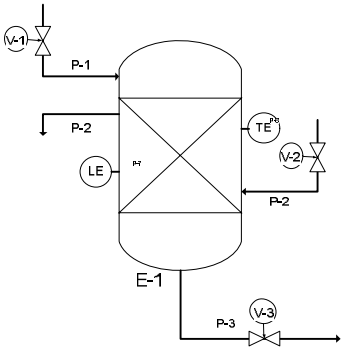


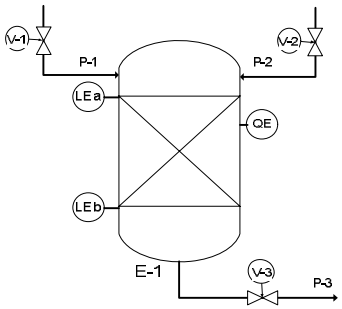
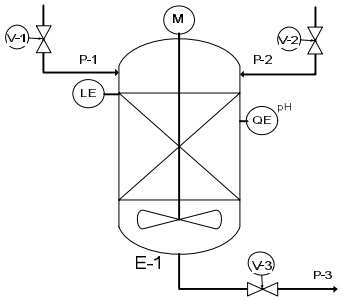
Рис. 4.10. Налаштування перетворювача аналогового сигналу

перетворювача опору буде такою – периферійне вхідне слово *PIW752*. Подібно працює інша бібліотечна функція – *FC106*, яка розраховує вихідне значення для аналогового сигнального модуля, тобто здійснює зворотне перетворення. Ця функція потрібна для керування аналоговим виконавчим механізмом.

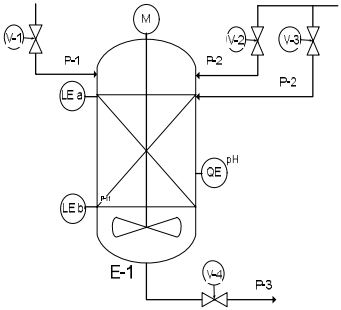
В табл. 4.11 пропонуються завдання для самостійного виконання. В завданнях вказані основні параметри технологічного процесу та алгоритм роботи установки, а функціональні схеми автоматизації визначають типи датчиків та виконавчих механізмів.

Таблиця 4.11 – Варіанти завдань для розроблення ППЗ.

Варіант 1	
	Після натискання кнопки «Пуск» відкривається клапан *V-1* і рідина заповнює апарат на 40 %. Після цього *V-1* закривається і відкривається клапан *V-2* на 90 % на трубопроводі подачі пари. Після досягнення заданої температури 70 °С клапан *V-2* закривається повністю і рідина витримується в апараті 900 с. Коли термін часу вичерпався відкривається клапан *V-3* і рідина зливається з апарата. Коли апарат порожній – цикл повторюється

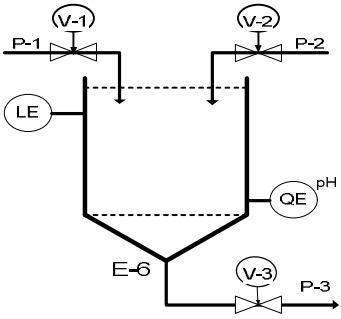
ВАРІАНТ 2	
	Після натискання кнопки «Пуск» відкривається клапан V-1 і апарат заповнюється продуктом до рівня «b». Далі клапан V-1 закривається. Після витримки часу 140 с відкривається клапан V-2 на 10 %. Якщо через 80 с концентрація (pH) в апараті не досягне заданого значення, то клапан V-2 відкрити на 25 %. Коли концентрація досягнута або досягнуто рівень «a» – повністю закрити клапан V-2 і відкрити клапан V-3 (для зливу суміші з апарата). Після досягнення рівня «b» – цикл повторюється
ВАРІАНТ 3	
	Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан V-1, апарат заповнюється водою до рівня 50 %. Далі клапан V-1 закривається і на 60 % відкривається клапан V-2. Продукт з 2-го трубопроводу заповнює апарат до рівня 90 %. Далі клапан V-2 закривається і на 540 с вмикається мішалка. Після вичерпання часу – відкривається клапан V-3 і суміш зливається з апарата. Коли апарат порожній – цикл повторюється

ВАРІАНТ 4

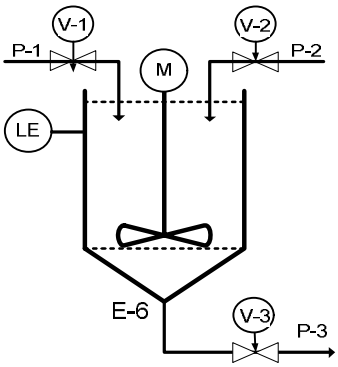


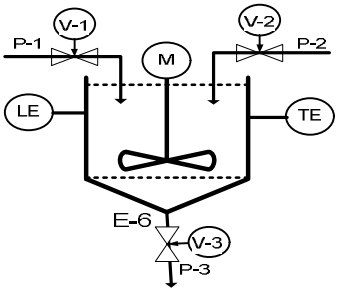
Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан *V-1* і рідина заповнює апарат до рівня «b». Далі клапан *V-1* закривається, вмикається мішалка і на 40 % відкривається клапан *V-2*. Якщо через 175 с концентрація (*pH*) в апараті не досягне потрібного значення, то додатково відкривається клапан *V-3* на 15 %. Якщо концентрація досягнута або досягнуто рівень «a» – закрити клапани *V-2* та *V-3*, відкрити клапан *V-4* (злив суміші за апарата). Після досягнення рівня «b» – цикл повторюється

ВАРІАНТ 5

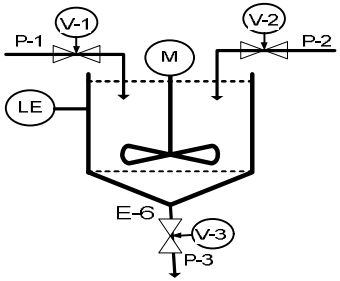


Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан V-1 і вода заповнює апарат на 60 %. Далі клапан V-1 закривається, на 30 % відкривається клапан V-2 і продукт надходить в апарат до досягнення потрібної концентрації. Якщо концентрація не досягнута, а рівень більше 80 % клапан V-2 закривається, клапан V-3 відкривається (рідина виливається з апарата). Після досягнення нульового рівня – цикл повторюється

ВАРІАНТ 6	
	Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан V-1 на 90 % і вода заповнює апарат на 30 %. Далі клапан V-1 закривається, відкривається клапан V-2 і продукт надходить в апарат до рівня 50 %. Клапан V-2 закривається і на 200 с вмикається мішалка. Коли термін часу вичерпався повністю відкривається клапан V-1 і рідина заповнює апарат на 80 %. Закривається клапан V-1 і знову вмикається мішалка на 10 хв. Далі після закінчення часу відкривається клапан V-3 і рідина виливається з апарата. Після досягнення нульового рівня – цикл повторюється

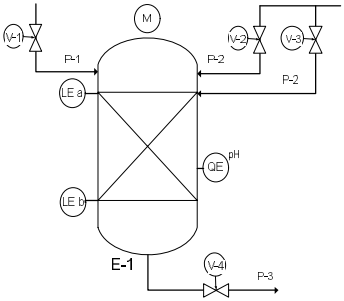
ВАРІАНТ 7	
	Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан *V-1* і вода заповнює апарат на 50 %. Далі клапан *V-1* закривається, відкривається клапан *V-2* і вмикається мішалка, гарячий продукт надходить в апарат до досягнення температури заданої величини. Якщо температура досягнута – вимикається мішалка та клапан *V-2*, відкривається клапан *V-3* і рідина виливається з апарата. Після досягнення нульового рівня – цикл повторюється

ВАРІАНТ 8

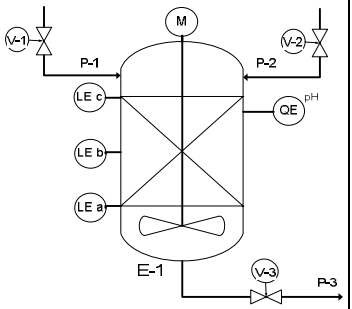


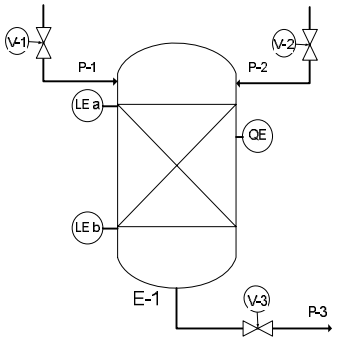
Якщо апарат порожній та натискається кнопка «Пуск», відкривається клапан V-1 і вода заповнює апарат на 40 %. Далі клапан V-1 закривається до 15 %, відкривається клапан V-2 і продукт надходить в апарат до досягнення рівня 80 %. Клапани V-1 та V-2 закриваються і на 220 с вмикається мішалка. Коли час вичерпався відкривається клапан V-3 і рідина виливається з апарата. Після досягнення нульового рівня клапан V-3 закривається. Далі цикл повторюється

ВАРІАНТ 9



Якщо апарат порожній та натискається кнопка «Пуск» відкривається клапан *V-1* і вода заповнює апарат до рівня «b». Далі клапан *V-1* закривається і на 150 с на 15 % відкривається клапан *V-2*. Після цього клапан *V-2* залишають відкритим на 10 % і відкривають клапан *V-3* на 75 %. Коли концентрація (*pH*) в апараті досягне потрібного значення або рівень досягне значення «a», клапани *V-2* і *V-3* повністю закриваються, клапан *V-4* відкривається і рідина виливається з апарата до рівня «b». Далі клапан *V-4* закривається, а цикл повторюється

ВАРІАНТ 10	
	Після натискання кнопки «Пуск» відкривається клапан V-1 на 60 % і вода заповнює апарат до рівня «a». Далі клапан V-1 закривається і на 130 с відкривається клапан V-2 та вмикається мішалка. Після цього клапан V-1 знову відкривається. Коли рівень в апараті досягне «c» або значення pH досягне критичного значення, клапан V-1 закривається. Клапан V-3 відкривається і суміш зливається з апарата до рівня «a». Далі клапан V-3 закривається, а цикл повторюється

ВАРІАНТ 11	
	Після натискання кнопки «Пуск» відкривається клапан V-1 і апарат наповнюється сировиною до рівня «b». Після цього клапан V-1 закривається. Після витримки часу 150 с відкривається клапан V-2 на 10 %. Якщо через 90 с концентрація (pH) в апараті не досягне заданого значення, то клапан V-2 відкрити на 20 %. Якщо досягнуто рівень «a» або потрібна концентрація, повністю закрити клапан V-2 і відкрити клапан V-3 для зливу суміші з апарата. Після досягнення рівня «b» клапан V-3 повністю закривається. Далі цикл повторюється