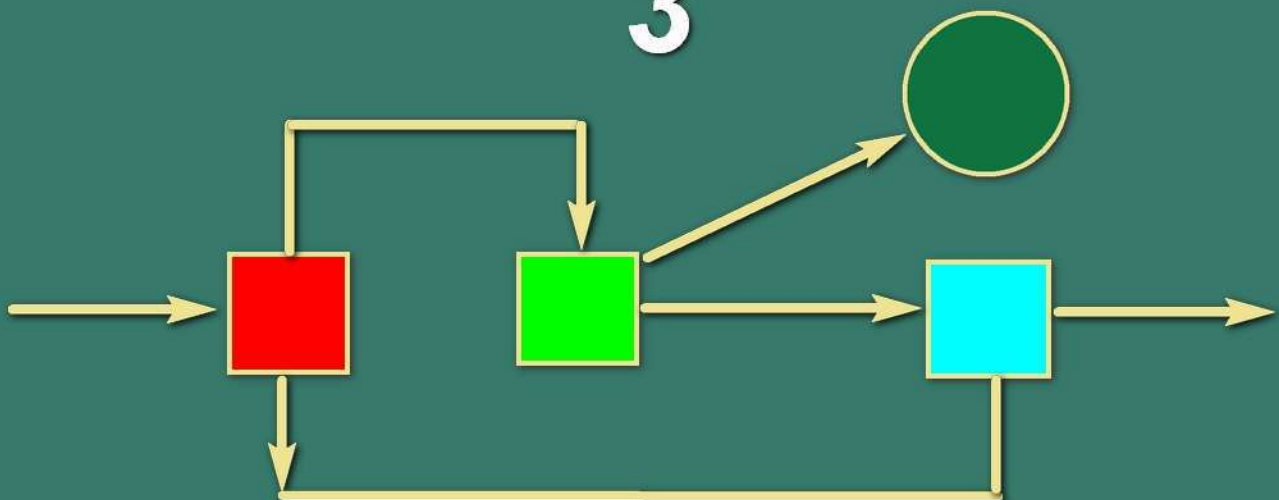




О.М. Пупена
І.В. Ельперін

КОНТРОЛЕРИ ТА ЇХ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

3



О.М. ПУПЕНА, І.В. ЕЛЬПЕРІН

КОНТРОЛЕРИ ТА ЇХ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

КУРС ЛЕКЦІЙ

ЧАСТИНА 3

для студентів напрямку "Автоматизація та комп'ютерно-інтегровані технології" денної та заочної форм навчання

КИЇВ
НУХТ
2011

ПУПЕНА О.М., ЕЛЬПЕРІН І.В. Контролери та їх програмне забезпечення. Курс лекцій для студ. напр. 6.50202 "Автоматизація та комп'ютерно-інтегровані технології" денної та заочної форм навчання. Частина 3. – К.: НУХТ, 2011. – 48 с.

Рецензент **Беляєв Ю.Б.**, д-р техн. наук

Пупена О.М., кандидат техн. наук

Ельперін І.В., кандидат техн. наук

Видання подається в авторській редакції

Навчальне видання

КОНТРОЛЕРИ ТА ЇХ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

КУРС ЛЕКЦІЙ

ЧАСТИНА 3

Укладачі: Пупена Олександр Миколайович
Ельперін Ігор Володимирович

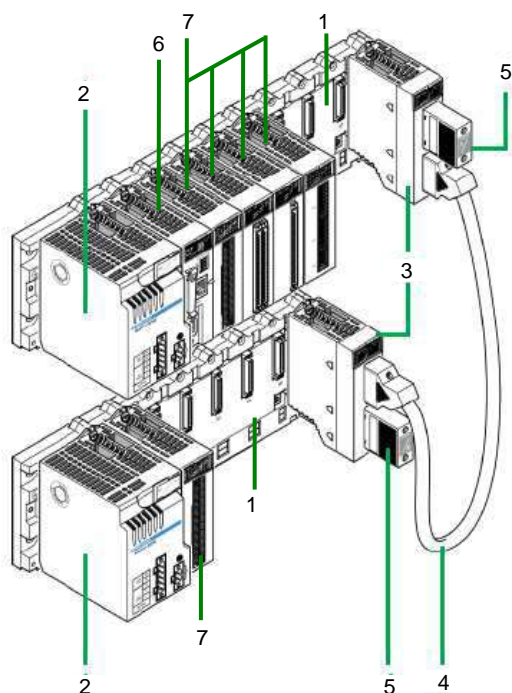
© О.М. Пупена, І.В. Ельперін, 2011
© НУХТ, 2011

1. АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ МІКРОПРОЦЕСОРНИХ КОНТРОЛЕРІВ MODICON M340.

1.1. Архітектура M340.

Modicon M340 – промисловий контролер нового покоління фірми Schneider Electric, для програмування якого використовується програмне забезпечення **UNITY PRO**. Modicon M340 – контролер модульного типу, конфігурація якого вибирається в залежності від кількості входів-виходів і алгоритму управління. Модулі кріпляться на **шасі**, яке виконує механічну та електричну функції. Така конструкція дає можливість гарячої заміни модулів без зупинки контролера. M340 може включати від 1-го до 4-х шасі з різною кількістю місць для установки модулів (від 4-х до 12-ти), об'єднаних між собою **BusX** шиною, загальною довжиною до 30 м.

Конструктивно M340 може складатись з таких основних елементів (рис.1.1):



1. Шасі, на яких встановлюються модулі.
2. Модуль живлення, який обов'язково повинен бути присутнім в кожному шасі, і який встановлюється на спеціально відведеному місці у шасі
3. Модуль розширення для контролерів побудованих на базі декількох шасі.
4. Кабелі розширення BusX, що з'єднує модулі розширення на суміжних шасі.
5. Термінуючі резистори в кінцевих модулях розширення архітектури M340.
6. Процесорний модуль, який обов'язково розміщується в посадочному місці з номером 00 у шасі, яке має номер 0.
7. Модулі вводу/виводу та модулі спеціального призначення, які розміщуються в будь якому посадочному місці.

Рис.1.1.Архітектура Modicon M340

Основним конструктивним елементом контролера є шасі (рис.1.2). З одного боку, шасі використовується як конструктивний елемент, на якому розміщуються й закріплюються окремі модулі контролера, з іншого – шасі має загальну шину BusX, по якій відбувається як живлення модулів, установлених в шасі, так і обмін сигналами та даними між окремими модулями контролера. Шасі може кріпитися як на стандартну DIN-рейку так і з допомогою гвинтів.

Шасі відрізняються за кількістю місць для встановлення модулів, відповідно на 4 (BMX XBP 0400), 6 (BMX XBP 0600), 8 (BMX XBP 0800) та 12 (BMX XBP 1200) позицій. У разі необхідності використовувати велику кількість модулів контролер може складатись з декількох шасі (рис.1.2). У цьому випадку в роз'єм XBE кожного шасі встановлюються модулі розширення XBE 1000, які з'єднуються BusX кабелем (кутовим BMX XBC •••К або прямим TSX CBY •••К, де ••• - довжина в дециметрах). Кожен модуль розширення має перемикач за допомогою якого виставляється адреса шасі в діапазоні від 0 до 3 (рис.1.3). Послідовність адресації шасі може не співпадати з їх фізичним розміщенням, однак процесорний модуль

завжди повинен знаходитись в шасі за номером 0. В кінцевих модулях розширення встановлюють термінатори шини TSX TLY EX типу А та В, відповідно у вхідний роз'єм – для першого модуля розширення та вихідний – для останнього.

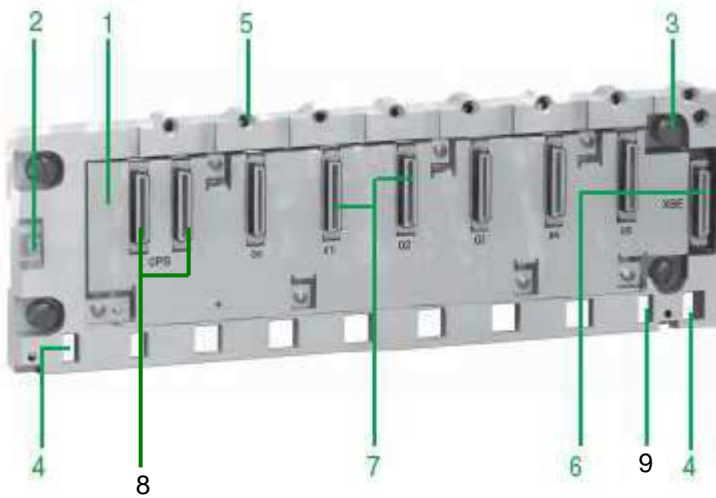


Рис.1.2. Шасі Modicon M340

1. Металева рама.
2. Клема заземлення.
3. Отвори для кріплення шасі.
4. Кріплення для заземлення екранів кабелів.
5. Різьбові отвори під гвинт для закріплення кожного модуля.
6. Роз'єм для модуля розширення (маркований як XBE).
7. Роз'єми для процесорного модуля, модулів вводу/виводу, комунікаційних модулів та модулів спеціального призначення.
8. Роз'єми для модуля живлення (маркований як CPS).
9. Отвори для установочних штирів модулів.

Всі модулі шасі, включно процесорний, живляться по внутрішній шині від модуля живлення BMX CPS... (рис.1.4). Модуль живлення підбирається по типу живлення (постійний або змінний струм) та споживаної потужності і вставляється у кожне шасі в роз'єми з маркуванням CPS (рис.1.2). Розрахунок споживаної потужності залежить від кількості і типу модулів, які встановлюються у шасі, приклад якого розглянутий у розділі 1.5. Цей розрахунок можна також виконати у програмуванні UNITY PRO в процесі конфігурування апаратної частини ПЛК. Будь який модуль живлення M340 має аварійне реле, яке відключається при зупинці контролера, або

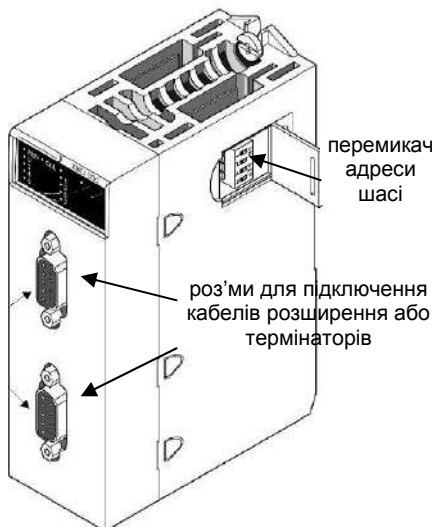


Рис.1.3. Модуль розширення XBE 1000

коли система само діагностики виявить некоректне значення вихідної напруги модуля живлення..

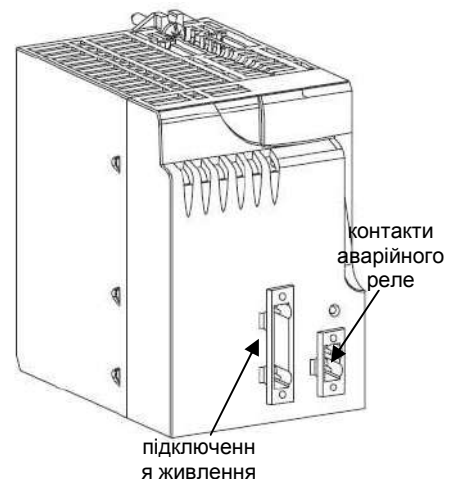


Рис.1.4. Модуль живлення BMX CPS...

1.2. Процесорні модулі

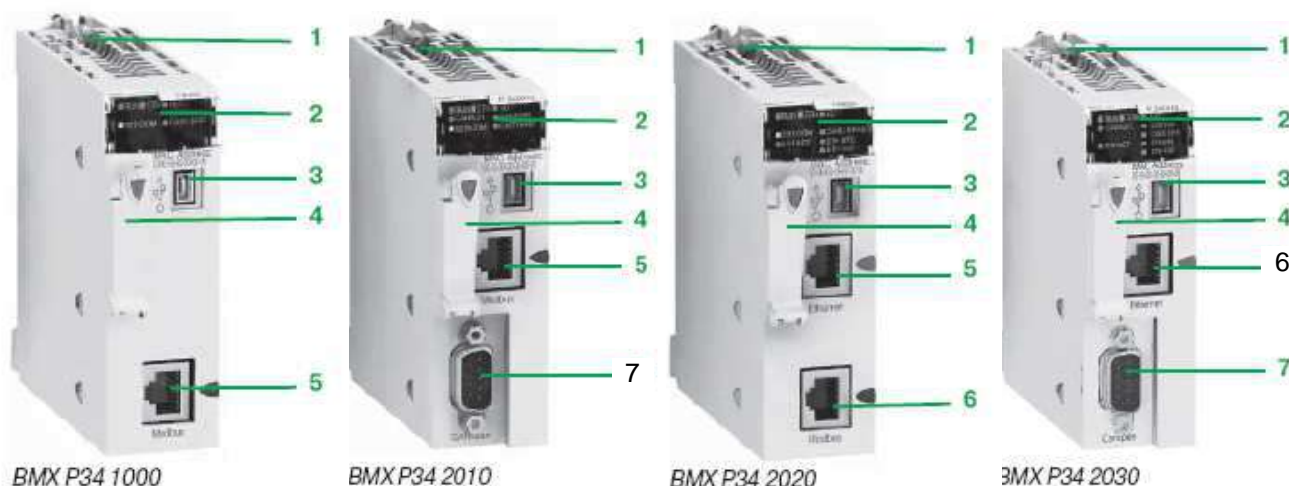
Процесорні модулі M340 відрізняються функціональними можливостями, швидкістю обробки інструкцій, кількістю входів/виходів, які може обробляти контролер, кількістю спеціальних каналів, об'ємом доступної оперативної пам'яті та вбудованими в модуль ЦПУ комунікаційними засобами.

Таблиця 1.1.

Загальні характеристики процесорних модулів

Характеристика		BMX P34 1000	BMX P34 2000	BMX P34 2010	BMX P34 2020	BMX P34 2030
Макс. кількість	шасі	2	4			
	дискретних вх+вих.	512	1024			
	аналогових вх+вих	128	256			
	лічильних каналів	20	36			
Об'єм RAM	загальний розмір	2048 Кб	4096 Кб			
	для програм, констант, символів	1792 Кб	3584			
	для даних	128 Кб	256 Кб			
Макс. кількість об'єктів	локалізовані внутрішні біти %Mi	16250	32464			
	локалізовані внутр. слова %MWi	32464				
	нелокалізовані внутрішні дані	128 Кб	256 Кб			
вбудовані комунікації	послідовний RS-485/RS-232C	+	+	+	+	-
	Ethernet TCP/IP	-	-	-	+	+
	CANOpen	-	-	+	-	+

У кожному процесорному модулі M340 є вбудований USB-інтерфейс(рис.1.5, поз 3), який призначений для підключення терміналу програмування (комп'ютер зі встановленим UNITY PRO), а також для з'єднання зі операторськими станціями з встановленим програмним забезпеченням SCADA/HMI, а також з операторськими панелями. Для цього можна використати спеціальний екранований кабель, який поставляється у комплекті з процесорним модулем M340, або стандартний USB кабель з роз'ємом mini B. У будь якому випадку довжина кабелю не може перевищувати 5 м.



1. Гвинт для закріплення модуля на шасі.
2. Блок індикації.
3. Роз'єм USB mini B для підключення терміналу програмування, або засобів SCADA/HMI;
4. Відсік для карти пам'яті;
5. Роз'єм RJ45 для підключення кабелю послідовного інтерфейсу RS-485 та RS-232C, по Modbus RTU/ASCII або символьного режиму (маркування чорним кольором);
6. Роз'єм для підключення кабелю Ethernet TCP/IP 10BASE-T/100BASE-TX (маркування зеленим кольором).

Рис.1.5. Процесорні модулі Modicon M340

У спеціальному слоті (рис.1.5, поз 4) розміщується SD-карта пам'яті. На карті, що входить у комплект стандартної поставки M340 (об'ємом 8 Мбайт), зберігається загрузочний проект, вбудовані діагностичні Веб-сторінки, а також при необхідності вихідний код проекту, константи та діалогові таблиці. Альтернативний варіант – використання карти обсягом 128 Мб, з підтримкою збереження даних користувача з прикладної програми, а також файлових операцій через FTP Сервер.

Кожний процесорний модуль може вміщувати один або два вбудовані комунікаційні канали з комбінації (рис 1.5): послідовний Modbus Serial RS-232/RS-485, Ethernet TCP/IP та CANOpen. Крім функцій обміну з іншими пристроями системи, Modbus RTU (Serial) та Modbus TCP/IP (Ethernet) забезпечують доступ терміналу програмування UNITY PRO до контролера.

1.3. Дискретні модулі.

1.3.1. Загальна характеристика. Модулі дискретних входів/виходів M340 являють собою стандартні модулі, які займають один слот. Ці модулі відрізняються за типом каналів (вхідні, вихідні, змішані), за кількістю каналів, за типом вхідних та вихідних каналів і за способом підключення. Ці модулі можна встановлювати у будь-яке посадочне місце шасі, окрім місця для живлення (PS) та процесорного модуля. Дозволяється гаряча заміна модулів (при включеному живленні).

1.3.2. Типи модулів. Дискретні модулі можуть мати входи/виходи постійного струму (DC) на 24 VDC та 48 VDC з позитивною (sink) або негативною (source) логікою підключення, або змінного струму (AC) на 100-240 VAC.

Таблиця 1.2.

Основні технічні характеристики дискретних модулів.

Позначення модуля	Кількість каналів	Характеристики каналів	Підключення
Модулі дискретних входів			
BMX DDI1602	16	24 VDC, позитивна логіка	20-конт. з'ємна кол.
BMX DDI1603	16	48 VDC, позитивна логіка	20-конт. з'ємна кол.
BMX DAI1602	16	24 VDC негативна логіка або 24 VAC	20-конт. з'ємна кол.
BMX DAI1603	16	48 VAC	20-конт. з'ємна кол.
BMX DAI1604	16	100..120 VAC	20-конт. з'ємна кол.
BMX DDI3202K	32	24 VDC, позитивна логіка	40-конт. роз'єм
BMX DDI6402K	64	24 VDC, позитивна логіка	два 40-конт. роз'єми
Модулі дискретних входів та виходів (змішані)			
BMX DDM16022	8 Вх	24 VDC, позитивна логіка	20-конт. з'ємна кол.
	8 Вих	24 VDC, захищені, позитивна логіка, 0.5 А	
BMX DDM16025	8 Вх	24 VDC, позитивна логіка	20-конт. з'ємна кол.
	8 Вих	релейні VDC/VAC, незахищені, 2 А	
BMX DDM3202K	16 Вх	24 VDC, позитивна логіка	40-конт. роз'єм
	16 Вих	24 VDC, захищені, позитивна логіка, 0.1 А	
Модулі дискретних виходів			
BMX DDO3202K	32	24 VDC, захищені, позитивна логіка, 0.1 А	40-конт. роз'єм
BMX DDO6402K	64	24 VDC, захищені, позитивна логіка, 0.1 А	два 40-конт. роз'єми
BMX DDO1602	16	24 VDC, захищені, позитивна логіка, 0.5 А	20-конт. з'ємна кол.
BMX DDO1612	16	24 VDC, захищені, негативна логіка, 0.1 А	20-конт. з'ємна кол.
BMX DAO1605	16	тиристорні 100...240VAC, незахищені, 0.6 А	20-конт. з'ємна кол.
BMX DRA0805	8	релейні VDC/VAC, незахищені, 3 А	20-конт. з'ємна кол.
BMX DRA1605	16	релейні VDC/VAC, незахищені, 2 А	20-конт. з'ємна кол.

Доступні модулі з транзисторними або релейними виходами. Виходи можуть бути захищені від короткого замикання. Всі дискретні входи та виходи ізольовані від внутрішньої шини. У таблиці 1.2 наведені основні технічні характеристики дискретних модулів.

1.3.3. Способи підключення. Дискретні модулі за способом підключення зовнішніх сигналів можуть бути з 20-контактною з'ємною клемною колодкою (рис.1.6 варіант А) або з 40-контактними з'єднувальними роз'ємами (рис.1.6 варіант Б).



1- корпус; 2- маркування модуля; 3- панель індикації станів каналів; 4 – роз'єм для підключення з'ємної клемної колодки (варіант А) або виносної клемної колодки (варіант Б)

Рис.1.6. Зовнішній вигляд дискретних модулів з різними варіантами підключення

Для модулів з клемною колодкою (варіант А) додатково замовляється 20-контактна з'ємна клемна колодка BMX FTV 20•0, або готовий кабель, який на одному кінці має клемну колодкою, а на іншому вільні провідники (з розпушеними кінцями) з кольоровим маркуванням (рис.1.8, а). Існують три види 20-контактних клемних колодок:

- гвинтова клемна колодка BMX FTV 2000;
- колодка з гвинтовими зажимами BMX FTV 2010;
- пружинна клемна колодка BMX FTV 2020;

З'ємні клемні колодки поставляються з аксесуарами для кодування, що дає можливість забезпечити унікальний механічний ключ для кожної пари – модуль-клемна колодка (рис.1.7). Іншими словами, кодування виключає можливість підключення клемної колодки, яка була встановлена на модулі до іншого модуля.

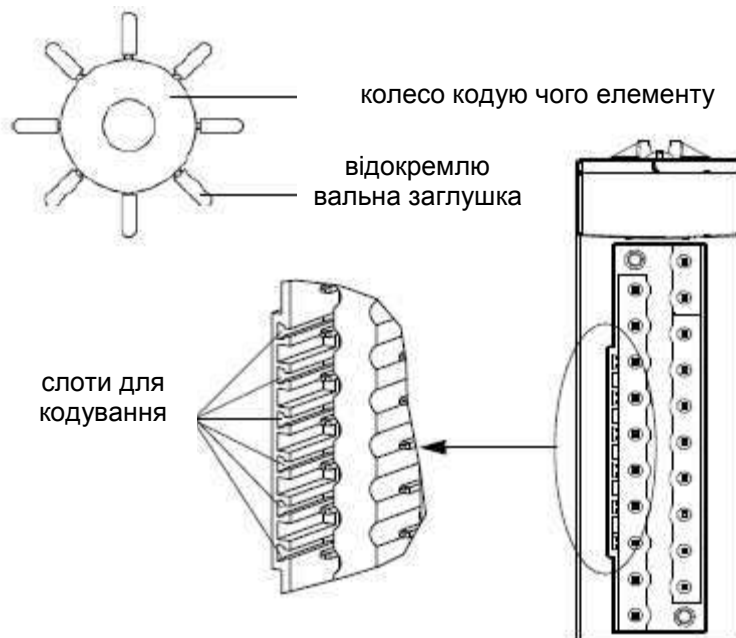


Рис.1.7. Механічне кодування модулів

Модулі з роз'ємами (варіант Б) на 32 канали мають один 40-контактний роз'єм, на 64 канали – два роз'єми. До таких модулів додатково замовляються спеціальні кабелі з 40-контактним з'єднувачем в одному з двох варіантів:

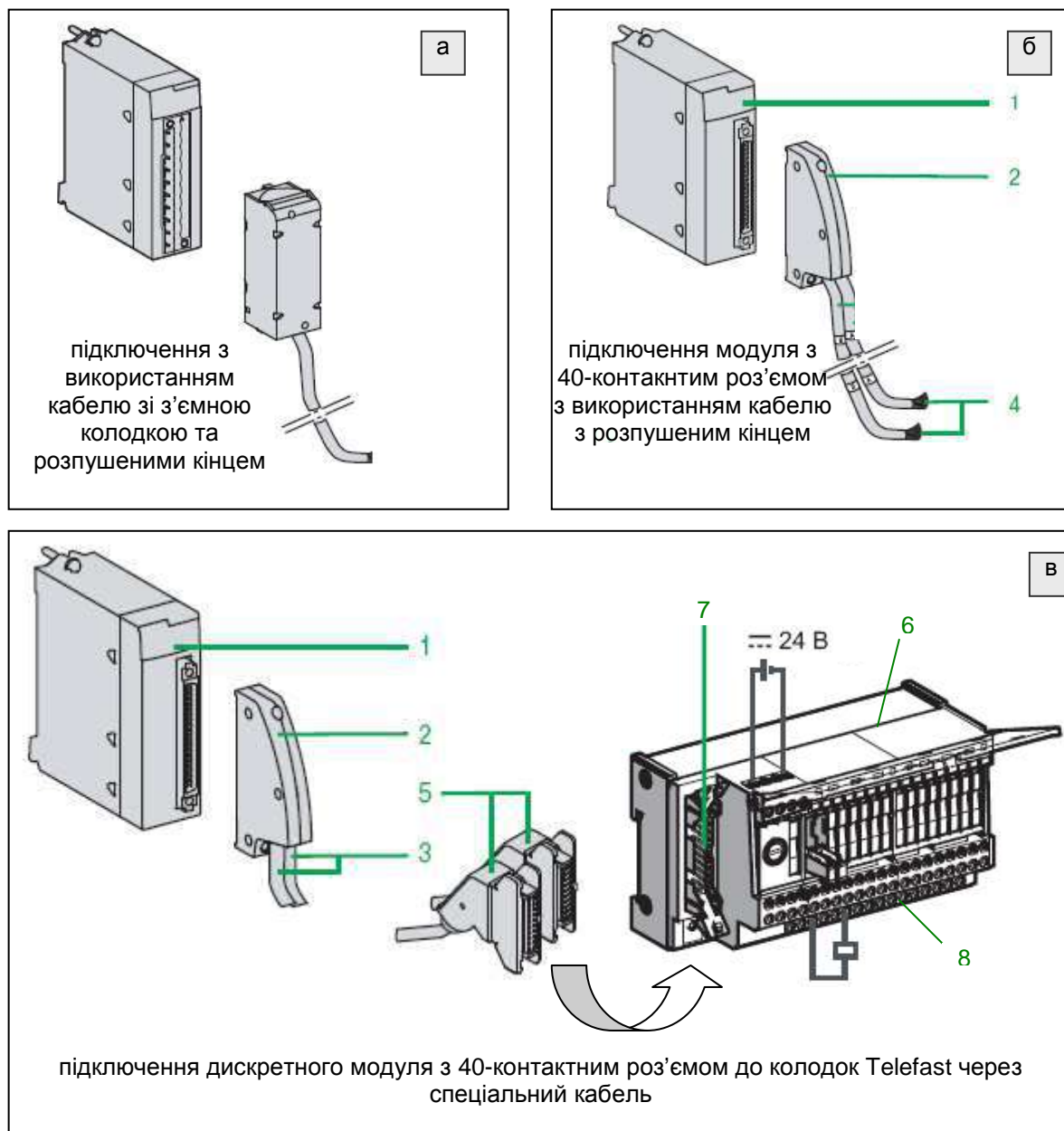
- FCW••3, які з іншого боку мають розпушений кінець з кольоровим маркуванням провідників (рис.1.8. б);
- FCC••3, які з іншого боку мають два з'єднувачі HE10 для підключення до виносних клемних колодок типу Telefast ABE (рис.1.8.в).

Підключення з використанням кабелів з розпушеним кінцем проводиться через додаткову клемну колодку довільного виробника.

Підключення модулів через кабелі з HE10 з'єднувачами проводиться тільки з використанням спеціальних виносних блоків з клемними колодками системи швидкого монтажу **Telefast ABE**. Schneider Electric пропонує дуже велику гаму блоків Telefast для дискретних модулів, які відрізняються:

- кількістю та типом каналів, які обслуговує даний блок;
- типом клем (гвинтові, пружинні);
- наявністю розподілення живлення;
- наявністю гальванічних розв'язок між каналами, між блоком та дискретним модулем;
- вбудованими додатковими функціями перетворення сигналу (вбудовані або з'ємні твердотільні або електромеханічні реле на різні потужності)
- наявністю додаткових функцій захисту;
- наявністю світлових індикаторів;
- наявністю можливості ручного включення/відключення сигналу;
- іншими додатковими опціями.

Усі блоки Telefast мають змінний плавкий запобіжник, який захищає входи/виходи модуля від перевантаження.



1 - дискретний модуль; 2 - 40-контактний роз'єм; 3 – кабель FCC••3; 4 – розпушений кінець кабелю; 5 – з'єднувачі типу HE10 для підключення до виносних клемних колодок типу Telefast; 6 – виносна клемна колодка типу Telefast; 7 – роз'єм типу HE10; 8 – клеми для підключення зовнішніх сигналів;

Рис.1.8. Способи підключення технічних засобів до дискретних модулів

Одним із універсальних блоків Telefast для дискретних входів/виходів є ABE7H16R21, який може підключатися до будь яких модулів з 40-контактним з'єднувачем з використанням кабелю FCC••3 (•• - залежить від довжини кабелю). Він використовується для підключення 16 дискретних входів або 16 дискретних виходів окремими парами гвинтових клем колодки.

Перелік необхідних аксесуарів для дискретних модулів зведений в таблицю 1.3. У таблиці 1.3 не наведений перелік аксесуарів для способів підключення кабелів з розпушеним кінцем та клемних колодок з підключенням до Telefast. У таблиці 1.3 також наведений тільки один варіант блоку Telefast.

Таблиця 1.3.

Монтажні аксесуари для підключення дискретних модулів.

Позначення модуля	Тип підключення	Спосіб підключення
Модулі дискретних входів		
BMX DDI1602	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DDI1603	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DAI1602	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DAI1603	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DAI1604	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DDI3202K	40-контактний роз'єм	кабель FCC••3 (від 0.5 до 10 м) + Telefast ABE 7H16R21 – 2 шт.
BMX DDI6402K	40-контактний роз'єм	(кабель FCC••3 (від 0.5 до 10 м) + Telefast ABE 7H16R21 – 2 шт.) – 2 комплекти
Модулі дискретних входів та виходів (змішані)		
BMX DDM16022	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DDM16025	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DDM3202K	40-контактний роз'єм	кабель FCC••3 (від 0.5 до 10 м) + Telefast ABE 7H16R21 – 2 шт.
Модулі дискретних виходів		
BMX DDO3202K	40-контактний роз'єм	кабель FCC••3 (від 0.5 до 10 м) + Telefast ABE 7H16R21 – 2 шт.
BMX DDO6402K	два 40-контактні роз'єми	(кабель FCC••3 (від 0.5 до 10 м) + Telefast ABE 7H16R21 – 2 шт.) – 2 комплекти
BMX DDO1602	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DDO1612	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DAO1605	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DRA0805	20-контактна з'ємна колодка	з'ємна клемна колодка BMX FTB 20•0
BMX DRA1605	20-контактна з'ємна колодка.	з'ємна клемна колодка BMX FTB 20•0

1.3.4. Схеми підключення. На рисс.1.9 – 1.11 показані схеми підключення дискретних датчиків та виконавчих механізмів до деяких модулів зі з'ємною клемною колодкою. На рис.1.12 показана схема підключення до модулів з 40-контактним роз'ємом, на прикладі модуля змішаного типу BMX DDM3202K та блоку Telefast ABE 7H16R21.

BMX DDI 1602 (DC)

BMX DAI 1602/1603/1604 (AC)

BMX DAI 1602 (DC негат. логіка)

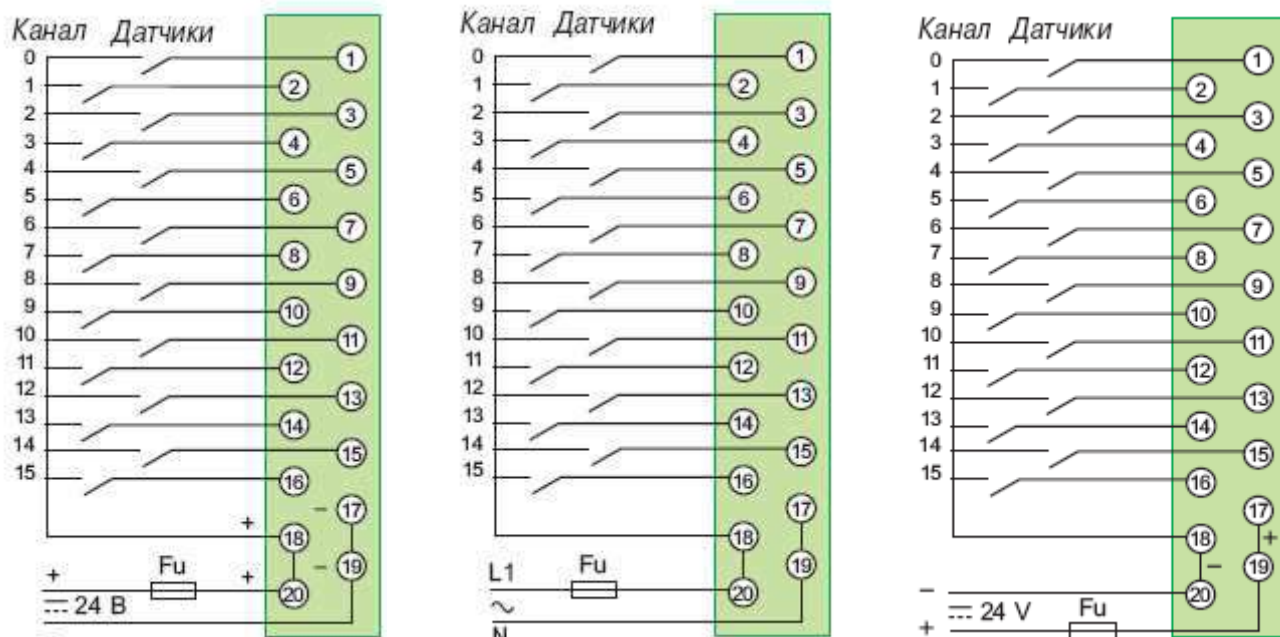


Рис.1.9. Підключення модулів дискретних входів зі з'ємними колодками.

BMX DDO 1602 (DC)

BMX DDO 1612 (DC негат. логіка)

BMX DRA 1605 (реле)

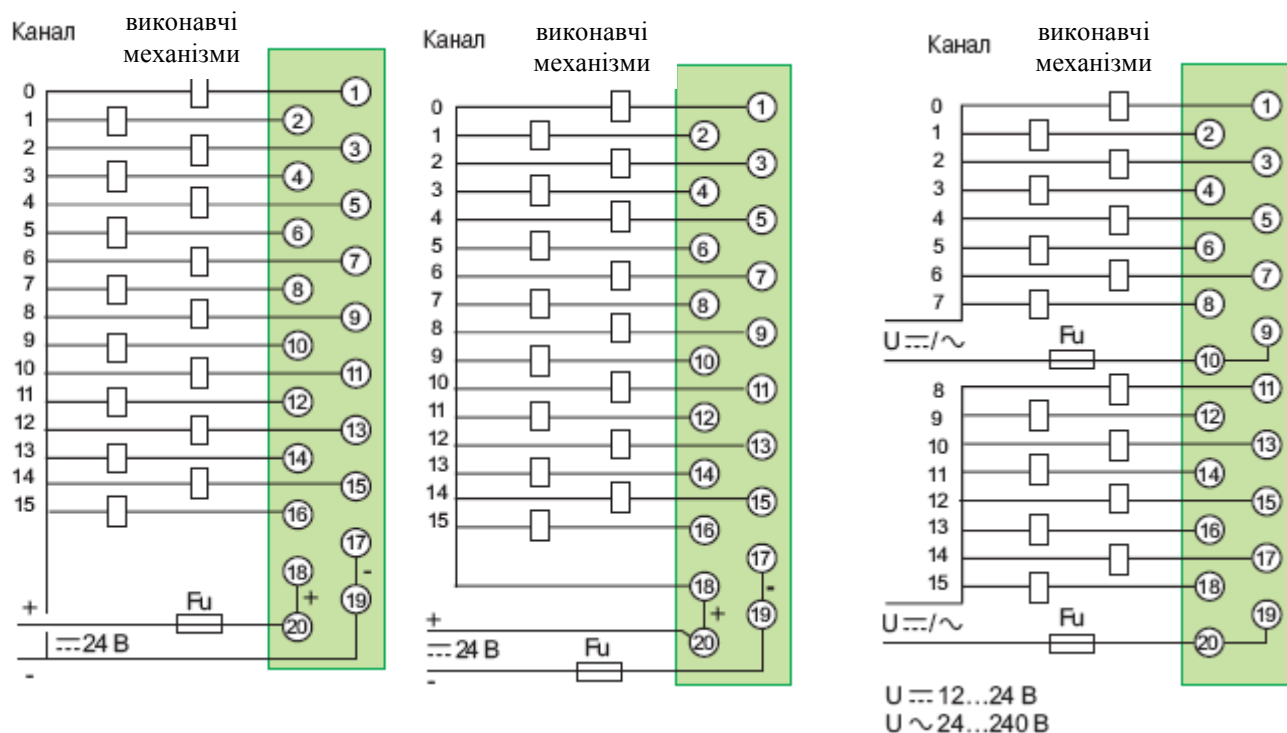


Рис.1.10. Підключення модулів дискретних виходів зі з'ємними колодками.

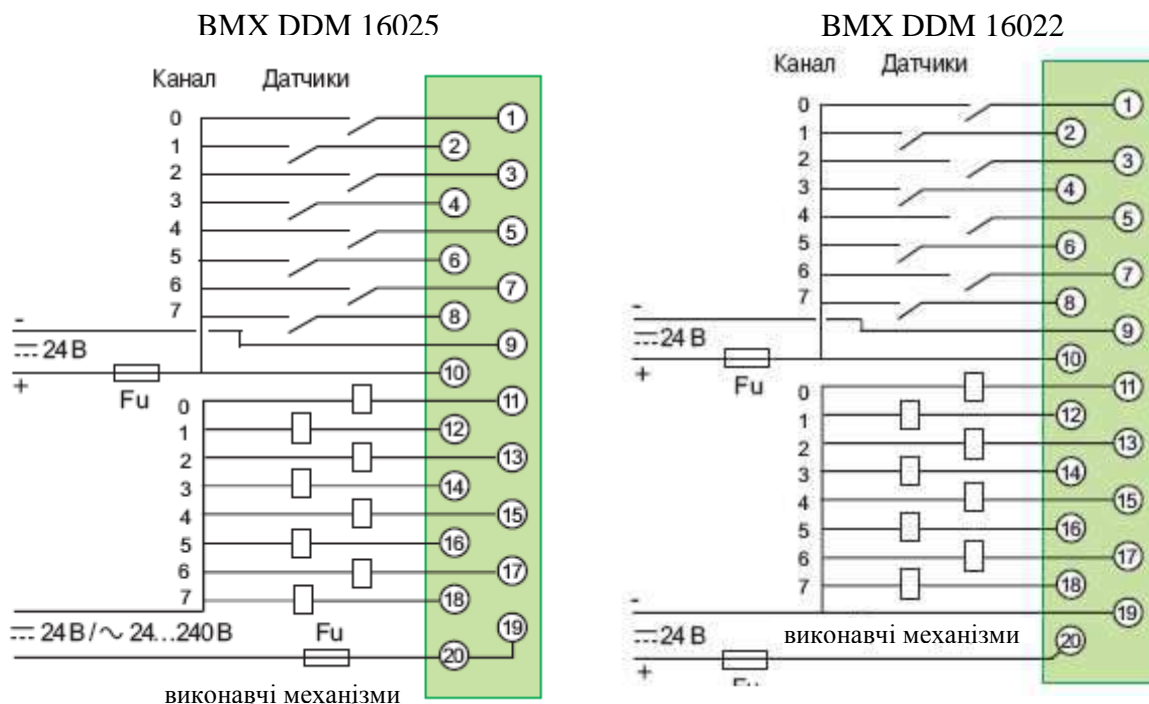


Рис.1.11. Підключення змішаних дискретних модулів зі з'ємними колодками.

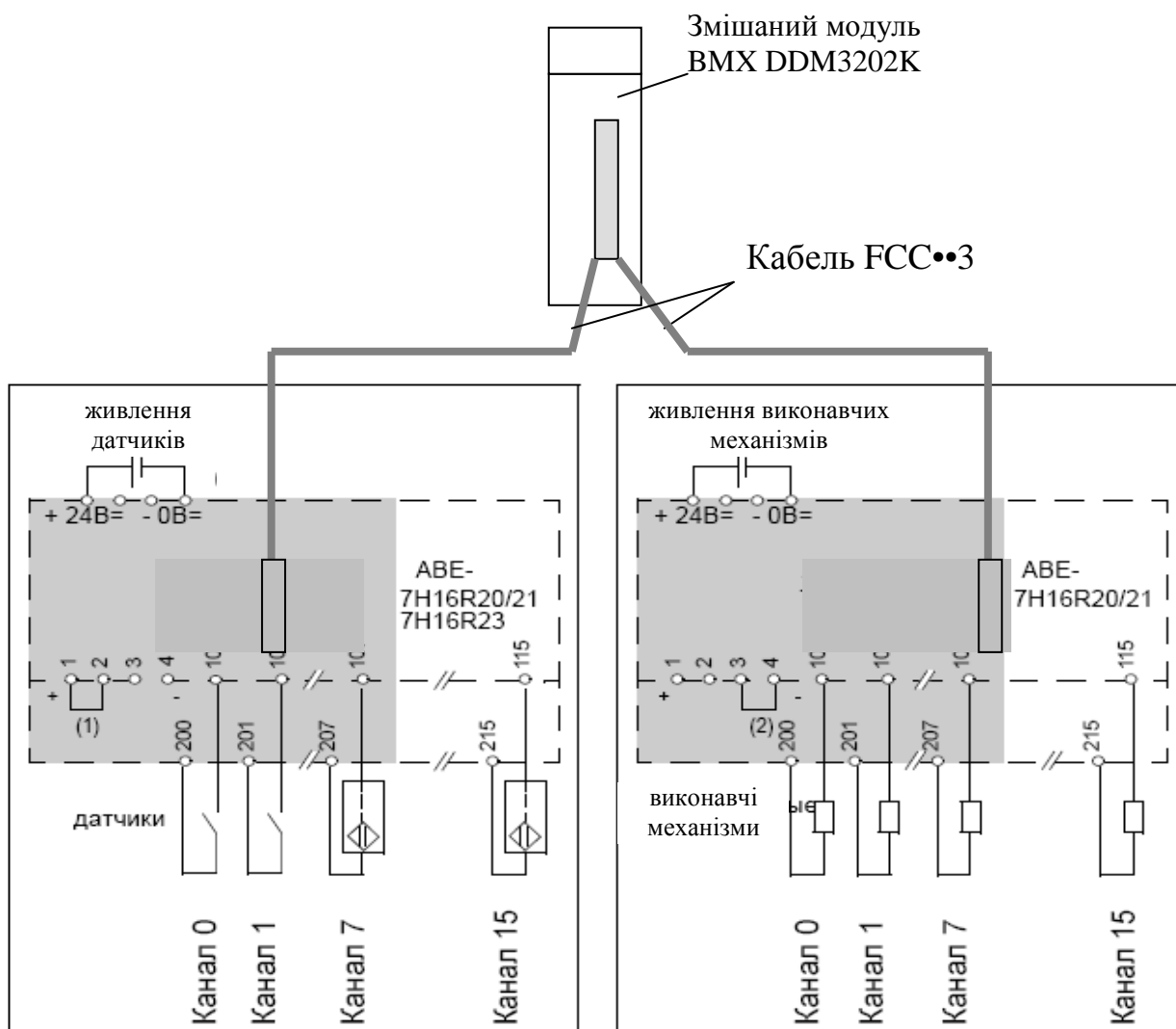


Рис.1.12. Схема підключення датчиків та виконавчих механізмів до Telefast ABE 7H16R21 на прикладі модуля BMX DDM3202K.

1.4. Аналогові модулі

1.4.1. Загальна характеристика. Модулі аналогових входів/виходів M340 являють собою стандартні модулі, які займають один слот. Як і дискретні модулі, аналогові відрізняються за типом каналів (вхідні, вихідні, змішані), за кількістю каналів, за характеристикою і діапазоном сигналів (напруга, струм, термометри опору, тощо), наявністю гальванічного розподілення і за способом підключення. Ці модулі можна встановлювати у будь-яке посадочне місце шасі, окрім місця для живлення (PS) та процесорного модуля. Дозволяється гаряча заміна модулів (при включеному живленні).

1.4.2. Типи модулів. Перелік всіх типів аналогових модулів M340 наведений в таб.1.4.

Таблиця 1.4.

Основні технічні характеристики аналогових модулів.

Позначення модуля	Кількість каналів	Діапазон сигналу	Характеристики каналів	Підключення
Модулі аналогових входів				
BMX ART 0414	4	мВ, термометри опору, термопари	16-бітні, ізоляція між каналами, час опитування модуля - 400 мс	40-контактний роз'єм
BMX ART 0814	8	мВ, термометри опору, термопари	16-бітні, ізоляція між каналами, час опитування модуля - 400 мс	40-контактний роз'єм
BMX AMI 0410	4	±10В,0...10В,0...5В, 0...20мА,4...20 мА	16-бітні, ізоляція між каналами, час опитування модуля - 5 мс	20-контактна з'ємна колодка
BMX AMI 800	8	±10В,0...10В,0...5В, 0...20мА,4...20 мА	16-бітні, з загальною точкою підключення, час опитування модуля – 9 мс	28-контактна з'ємна колодка
BMX AMI 810	8	±10В,0...10В,0...5В, 0...20мА,4...20 мА	16-бітні, ізоляція між каналами, час опитування модуля - 9 мс	28-контактна з'ємна колодка
Модулі аналогових входів та виходів (змішані)				
BMX AMM 0600	4 Вх	±10В,0...10В,0...5В, 0...20мА,4...20 мА	14-бітні для U, 12-бітні для I, загальна точка, час опитування модуля - 5 мс	20-конт. з'ємна кол.
	2 Вих	±10В,0...20мА,4...20 мА	12-бітні для U, 11-бітні для I, загальна точка	
Модулі аналогових виходів				
BMX AMO 0210	2	±10В,0...20мА,4...20 мА	16-бітні, ізоляція між каналами	20-конт. з'ємна кол.
BMX AMO 410	4	±10В,0...20мА,4...20 мА	16-бітні, ізоляція між каналами	20-конт. з'ємна кол.
BMX AMO 802	8	0...20мА,4...20 мА	16-бітні, загальна точка	20-конт. з'ємна кол.

Аналогічно аналоговим модулям Modicon Premium, аналогові вхідні модулі M340 виконують функції:

- сканування вхідних каналів різного діапазону за допомогою безконтактного мультиплексування;
- аналогово-цифрове перетворення;

- фільтрація сигналів;
- моніторинг модуля: тестування ланок перетворення, вхідний контроль перевищування рівня сигналу, тест наявності клемної колодки.

Модулі аналогових виходів виконують функції:

- цифро-аналогове перетворення;
- захист каналів модулів від перевантаження;
- моніторинг модуля: тест перетворення, тест виходу за межі, тест наявності клемної колодки.

1.4.3. Способи підключення. Подібно дискретним модулям за способом підключення зовнішніх сигналів, аналогові модулі можуть бути: з 20-контактною з'ємною клемною колодкою, з 28-контактною клемною колодкою або з 40-контактними з'єднувальними роз'ємами. З'ємні клемні колодки поставляються з аксесуарами для кодування, що дає можливість забезпечити унікальний механічний ключ для кожної пари – модуль-клемна колодка.

Для модулів, які підключаються через з'ємну клемну колодку, остання замовляється окремо як у звичайному варіанті, так і з підключеним кабелем розпушеним на кінці. 20-контактні клемні колодки для аналогових модулів такі самі як і для дискретних (BMX FTV 2000/2010/2020). Для модулів BMX AMI 800/810 замовляється 28-контактна клемна колодка BMX FTV 2820 з пружинними зажимами.

Підключення модулів BMX ART 0414/0814 з 40-контактними роз'ємами можна виконувати з використання блоків Telefast ABE 7CPA412 та кабеля BMX FCA ••2, (де •• залежить від довжини). Такий спосіб підключення схематично зображений на рис.1.13.

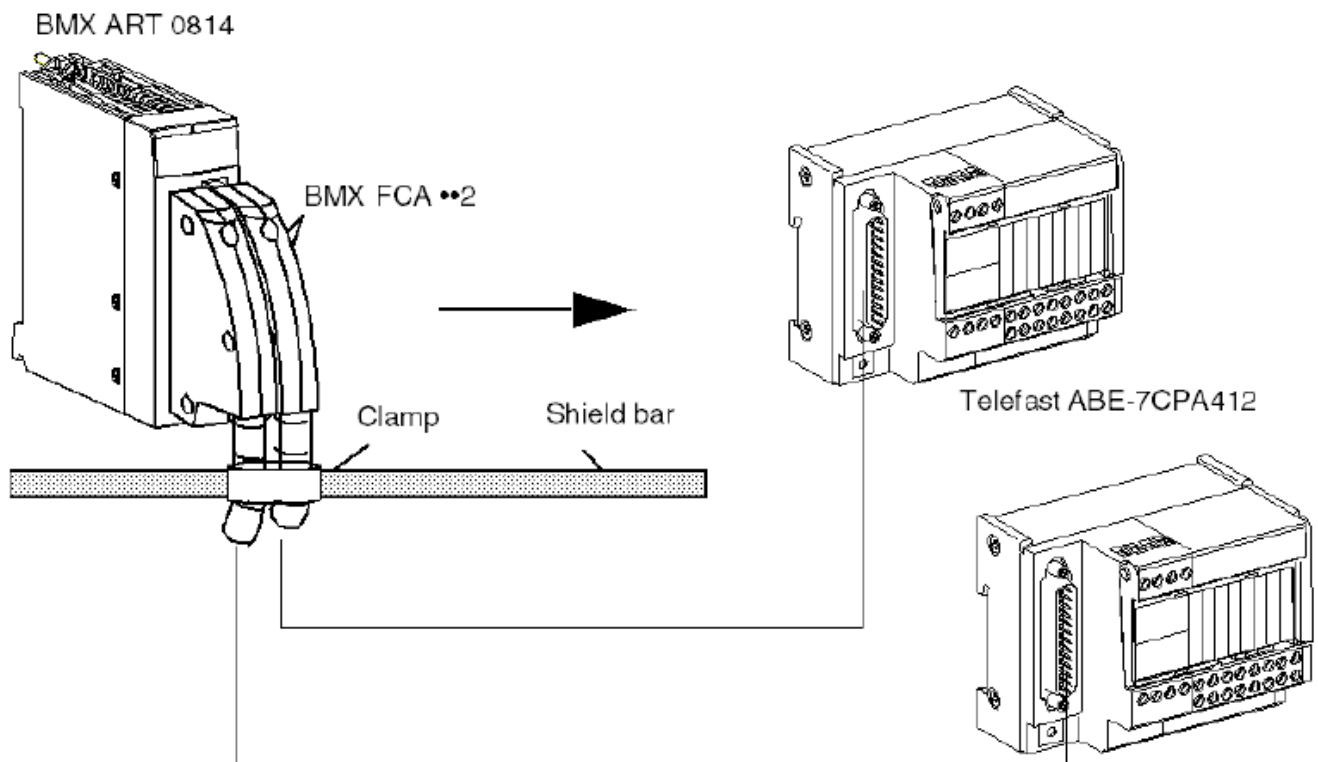


Рис.1.13. Підключення Telefast ABE 7CPA412 до модуля BMX ART 0814.

Перелік необхідних аксесуарів для аналогових модулів зведений в таблицю 1.5. У таблиці 1.5 не наведений перелік аксесуарів для способів підключення кабелів з розпушеним кінцем та клемних колодок з підключенням до Telefast.

Таблиця 1.5.

Монтажні аксесуари для підключення аналогових модулів.

Позначення модуля	Тип підключення	спосіб підключення
Модулі аналогових входів		
BMX ART 0414	40-конт. роз'єм	кабель BMX FCA ••2 (1.5, 2 або 5 м) + Telefast ABE 7CPA412
BMX ART 0814	два 40-конт. роз'єми	(BMX FCA ••2(1.5, 2 або 5 м) + Telefast ABE 7CPA412) – 2 комплекти
BMX AMI 0410	20-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 20•0
BMX AMI 800	28-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 2820
BMX AMI 810	28-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 2820
Модулі аналогових входів та виходів (змішані)		
BMX AMM 0600	20-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 20•0
Модулі аналогових виходів		
BMX AMO 0210	20-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 20•0
BMX AMO 410	20-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 20•0
BMX AMO 802	20-конт. з'ємна кол.	з'ємна клемна колодка BMX FTB 20•0

1.4.4. Схеми підключення. На рис.1.14 показані схеми підключення аналогових датчиків з уніфікованим сигналом до модулів зі з'ємною клемною колодкою. Всі канали можуть підтримувати як підключення датчиків з виходами як по струму так і по напрузі. На рис.1.14 показаний приклад для обох випадків. Схема підключення аналогових датчиків до змішаного модуля BMX AMM 0600 показана на рис.1.16.

На рис.1.15 показані схеми підключення датчиків температури (термометри опору, термопар) до модулів BMX ART 0414/0814. Всі канали підтримують можливість підключення термопар, термометрів опору по 2-х, 3-х та 4-х провідних схемах. Для реалізації підключення як правило використовується Telefast ABE 7CPA412 (див. таб.1.5), призначення клем якого відповідно до контактів роз'єму теж вказані на рис.1.15.

Схеми підключення виконавчих механізмів до аналогових модулів BMX АМО 0210/0410/0802, а також до аналогового змішаного модуля BMX АММ 0600 показані на рис.1.16. Всі вхідні канали BMX АМО 0210/0410 та BMX АММ 0600 рівнозначні і можуть підключатися як по напрузі так і по струму. На рис.1.16 показані приклади для обох способів.

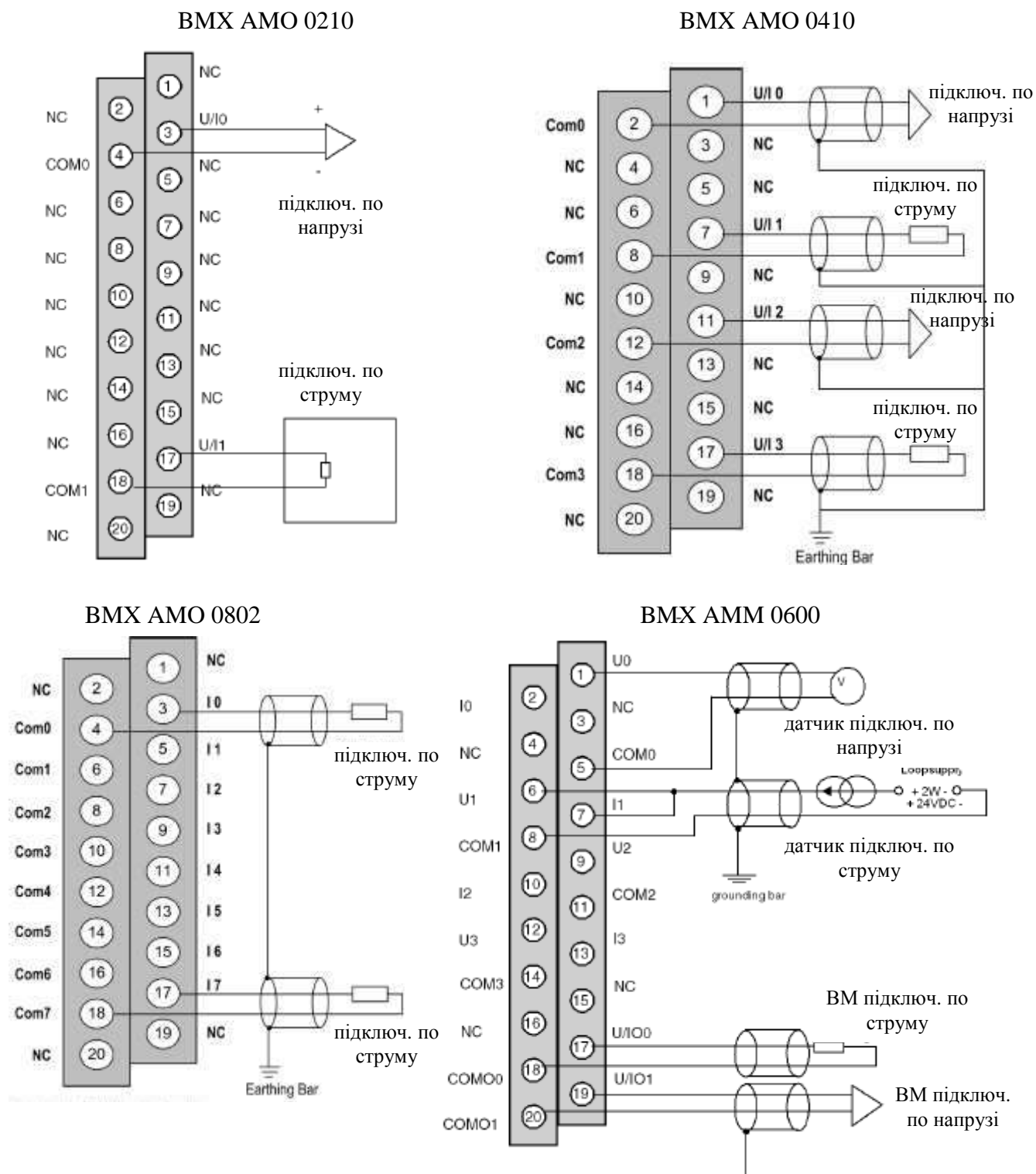


Рис.1.16. Схема підключення виконавчих механізмів до аналогових вихідних модулів BMX АМО BMX та схема підключення датчиків та ВМ до змішаного аналогового модуля АММ.

1.5. Вибір модуля живлення.

Споживана потужність, яка необхідна для модулів, встановлених на монтажному шасі, залежить від типу даних модулів. Тому для того щоб вірно вибрати модуль живлення, який забезпечить необхідне споживання необхідно проводити розрахунок енергоспоживання,

Живлення модулів по шасі проводиться через два виходи модуля живлення 24 В (24V_BAC) та 3,3 В (3V3_BAC). Вихід 24V_BAC використовується для живлення встановлених на монтажному шасі модулів входів-виходів та процесорного модуля, а вихід 3V3_BAC використовується тільки для живлення модулів входів-виходів.

Модулі живлення, що живляться напругою 100...240VAC (BMX CPS 2000 та BMX CPS 3500), додатково мають зовнішній вихід 24 В (24V_SENSORS), який можна використати для живлення датчиків або виконавчих механізмів.

Під час провдення розрахунку енергоспоживання модулів живлення BMX CPS 2000/3500 необхідно враховувати наступні вимоги:

- сумарна потужність, яка споживається по виходах модуля живлення 3V3_BAC, 24V_BAC та 24V_SENSORS не повинна перевищувати загальну корисну потужність модуля живлення:

$$I_{3V3_BAC} * 3,3 \text{ В} + I_{24V_BAC} * 24 \text{ В} + I_{24V_SENSORS} * 24 \text{ В} \leq P_{PS} \quad (1.1)$$

- сумарна потужність, яка споживається по двом виходах модуля живлення 3V3_BAC та 24V_BAC не повинна перевищувати загальну корисну потужність модуля живлення по цим виходам:

$$I_{3V3_BAC} * 3,3 \text{ В} + I_{24V_BAC} * 24 \text{ В} \leq P_{3V3_24V} \quad (1.2)$$

В формулах (1.1) та (1.2) I_{3V3_BAC} – сумарний споживаний струм модулями по 3V3_BAC, I_{24V_BAC} – сумарний споживаний струм модулями по 24V_BAC, $I_{24V_SENSORS}$ – сумарний споживаний струм по зовнішньому виходу 24V_SENSORS, P_{PS} – загальна корисна потужність модуля живлення, P_{3V3_24V} – максимальна сумарна потужність модуля на виходах 3V3_BAC та 24V_BAC.

При розрахунку енергоспоживання модулів живлення BMX CPS 2010/3020 необхідно враховувати наступну вимогу:

- сумарна потужність, яка споживається по виходах модуля живлення 3V3_BAC та 24V_BAC не повинна перевищувати загальну корисну потужність модуля живлення:

$$I_{3V3_BAC} * 3,3 \text{ В} + I_{24V_BAC} * 24 \text{ В} \leq P_{PS} \quad (1.3)$$

У таблиці 1.6 наведені характеристики потужності модулів живлення по різним виходам. Зокрема звідти можна вибрати величини P_{PS} та P_{3V3_24V} при виборі конкретного модуля живлення.

Величини I_{3V3_BAC} та I_{24V_BAC} розраховуються по середньому споживанню модулів, які наведені в таблиці 1.7. Ці значення представляють собою величину між максимальним споживаним струмом та нормальним споживаним струмом. Використовуючи дані з цієї таблиці, можна вирахувати сумарний споживаний струм по кожному монтажному шасі для кожного виходу модуля живлення, у відповідності з набором встановлених на монтажному шасі модулів.

Таблиця 1.6.

Характеристик потужності модулів живлення.

Потужність	BMX CPS 2000	BMX CPS 2010	BMX CPS 3020	BMX CPS 3500
Напруга живлення	100...240 VAC	24 VDC	24...48 VDC	100...240 VAC
Загальна корисна потужність (P_{PS})	20 Вт	17 Вт	32 Вт	36 Вт
Потужність на виході 3V3_BAC монтажного шасі	8,3 Вт (2,5 А)	8,3 Вт (2,5 А)	15 Вт (4,5 А)	15 Вт (4,5 А)
Потужність на виході 24V_BAC монтажного шасі	16,5 Вт (0,7 А)	16,5 Вт (0,7 А)	31,2 Вт (1,3 А)	31,2 Вт (1,3 А)
Максимальна сумарна потужність на виходах 3V3_BAC та 24V_BAC (P_{3V3_24V})	16,5 Вт	16,5 Вт	31,2 Вт	31,2 Вт
Сумарна корисна потужність на споживання зовнішніми датчиками 24V_SENOSRS	10,8 Вт (0,45 А)	-	-	21,6 Вт (0,9 А)

Таблиця 1.7.

Споживані струми різних модулів.

Тип модуля	Каталожний номер	Середній споживаний струм, в мА		
		На виході 3V3_BAC	На виході 24V_BAC	На виході 24V_SENOSRS
Процесорні модулі	BMX P34 1000	-	72	-
	BMX P34 2000	-	72	-
	BMX P34 2010/20102	-	90	-
	BMX P34 2020	-	95	-
	BMX P34 2030/20302	-	135	-
Аналогові модулі	BMX AMI 0410	150	45	-
	BMX AMI 0800	150	41	-
	BMX AMI 0810	150	54	-
	BMX AMM 0600	240	-	120
	BMX AMO 0210	150	110	-
	BMX AMO 0410	150	140	-
	BMX AMO 0802	150	135	-
	BMX ART 0414	150	40	-
	BMX ART 0814	220	50	-
Комунаційні модулі	BMX NOE 0100	-	90	-
	BMX NOE 0110	-	90	-
Лічильні модулі	BMX EHC 0200	200	40	80
	BMX EHC 0800	200	-	80
Дискретні вхідні модулі	BMX DAI 0805	103	13	-
	BMX DAI 1602	90	-	60
	BMX DAI 1603	90	-	60
	BMX DAI 1604	90	-	-
	BMX DDI 1602	90	-	60
	BMX DDI 1603	75	-	135
	BMX DDI 1604T	75	-	135
	BMX DDI 3202 K	140	-	110
	BMX DDI 6402 K	200	-	110
Дискретні вихідні модулі	BMX DAO 1605	100	95	-
	BMX DDO 1602	100	-	-
	BMX DDO 1612	100	-	-

	BMX DDO 3202 K	150	-	-
	BMX DDO 6402 K	240	-	-
	BMX DRA 0804T	100	110	-
	BMX DRA 0805	100	55	-
	BMX DRA 1605	100	95	-
Дискретні змішані модулі	BMX DDM 16022	100	-	30
	BMX DDM 16025	100	50	30
	BMX DDM 3202 K	150	-	55
Модуль розширення				

1.6. Приклади.

Приклад 1.1. Конфігурування контролера М340.

Завдання. Підберіть складові ПЛК М340 для забезпечення його роботи за умов, наведених в таблиці 1.8. Вибрані складові сформуєте у вигляді відомості.

Таблиця 1.8.

Вимоги	Кількість або наявність
Живлення ПЛК (24 VDC або 220 VAC)	220 VAC
Кількість дискретних входів 24 VDC, позитивна логіка	59
Кількість дискретних входів 115 VAC	67
Кількість аналогових входів 4-20 мА	26
Кількість аналогових входів 0-10 В	15
Кількість дискретних виходів струм комутації до 0,1 А 24 VDC	74
Кількість дискретних виходів струм комутації до 0,5 А 24 VDC	23
Кількість дискретних виходів струм комутації до 2 А 24 VDC	22
Кількість дискретних виходів струм комутації до 3 А 24 VDC	21
Кількість аналогових виходів 0-10 В	20
Кількість аналогових виходів 4-20 мА	19
Підключення по Modbus RTU на RS485	+
Підключення по Modbus/TCP на Ethernet	+
Підключення по CANOpen	-
Програма користувача займає (Кб)	2500

Рішення.

Конфігурування ПЛК М340 будемо проводити у такій послідовності:

1. Вибір процесорного модуля.
2. Вибір модулів вводу/виводу.
3. Вибір аксесуарів для модулів вводу/виводу.
4. Компонування шасі, вибір додаткових модулів та аксесуарів для шасі.
5. Вибір модулів живлення.

1. Вибір процесорного модуля. Процесорний модуль підбираємо по наступним критеріям: кількість каналів кожного типу, тип та кількість комунікаційних каналів, об'єм пам'яті користувача.

- кількість аналогових входів + виходів: $26+15+20+19=80$

- кількість дискретних входів + виходів: $59+67+74+23+22+21=266$

По таблиці 1.1 підбираємо процесорний модуль. По кількості каналів будь який процесорний модуль задовольняє умовам задачі, тому орієнтуємось тільки на кількість пам'яті та наявним комунікаціям. Серед процесорних модулів M340 підходить тільки BMX P34 2020.

2. Вибір модулів вводу/виводу проводиться по таблицям 1.2 та 1.4.

59 ВД 24 VDC - BMX DDI6402K - 1 шт. (64, 5 вільних)

67 ВД 115 VAC - BMX DAI1604 - 4 шт. ($5 \cdot 16 = 80$, 13 вільних)

21 ДВ 3 А - BMX DRA 0805 - 3 шт. ($8 \cdot 3 = 24$, 3 вільних)

22 ДВ 2 А - BMX DRA 1605 - 2 шт. ($16 \cdot 2 = 32$, 10 вільних)

23 ДВ 0,5 А - BMX DDO 1602 - 2 шт. ($16 \cdot 2 = 32$, 9 вільних)

74 ДВ 0,1 А - BMX DDO 6402K - 1 шт. (64 + 9 з BMX DDO 1602 + 1 з BMX DRA 1605)

26+15 ВА - BMX AMI 810 - 5 шт. ($8 \cdot 5 = 40$) + BMX AMI 0410 - 1 шт. (4)

20 АВ U - BMX AMO 410 - 5 шт. ($4 \cdot 5 = 20$)

19 АВ I - BMX AMO 802 - 3 шт. ($8 \cdot 3 = 24$)

З метою зменшення вартості системи, для дискретних вихідних каналів малих струмів комутації ми задіяли вільні канали модулів з більшими струмами комутації. Слід зазначити, що наведений варіант компонування не єдине рішення для даної задачі, тобто може бути декілька варіантів компонування ПЛК модулями вводу/виводу.

3. Вибір аксесуарів для модулів вводу/виводу проводиться по таблицям 1.3 та 1.5. Зведемо аксесуари та модулі в одну таблицю.

Таблиця 1.9.

Модуль вводу/виводу		Аксесуари		
Найменування	Кількість	Найменування	Кількість	примітка
BMX DDI6402K	1	BMX FCC053	2	кабель 0.5 м, з двома HE10 з'єднувачами
		Telefast ABE 7H16R21	4	16-канальний клемний блок
BMX DAI1604	4	BMX FTB 2010	4	20 контактна з'ємна колодка з гвинтовими зажимами
BMX DRA 0805	3	BMX FTB 2010	3	20 контактна з'ємна колодка з гвинтовими зажимами
BMX DRA 1605	2	BMX FTB 2010	2	20 контактна з'ємна колодка з гвинтовими зажимами
BMX DDO 1602	2	BMX FTB 2010	2	20 контактна з'ємна колодка з гвинтовими зажимами
BMX DDO 6402K	1	BMX FCC053	1	кабель 0.5 м, з двома HE10 з'єднувачами
		Telefast ABE 7H16R21	2	16-канальний клемний блок
BMX AMI 810	5	BMX FTB 2820	5	28 контактна з'ємна клемна колодка
BMX AMI 0410	1	BMX FTB 2010	1	20 контактна з'ємна колодка з гвинтовими зажимами
BMX AMO 410	5	BMX FTB 2010	5	20 контактна з'ємна колодка з гвинтовими зажимами
BMX AMO 802	3	BMX FTB 2010	3	20 контактна з'ємна колодка з гвинтовими зажимами

4. Вибір шасі, додаткових модулів та аксесуарів для шасі. Загальна кількість модулів разом з процесорним: 1 CPU + 10 DI + 3 DO + 6 AI + 8 AO = 28. Таким чином можна вибрати 2 монтажних шасі на 12 місць (BMX XBP 1200) та 1 шасі на 4 місця (BMX XBP 0400). До кожного шасі необхідно по 1-му модулю розширення, які зв'язуються BusX кабелями. Тобто додатково треба замовити модулі XBE 1000 - 3 шт., кабелі TSX CBY 010K – 2 шт. та один набір з двох термінаторів TSX TLY EX.

Розміщення модулів в шасі вибираємо довільно. Однак на практиці, модулі з найбільшим споживанням струму розподіляють рівномірно між шасі.

Для зручності підрахунку споживаної потужності кожним модулем, розміщення покажемо у вигляді таблиці 1.10. В цій таблиці заповнюємо тільки поля шасі, місце та найменування модулів (за виключенням модулів живлення - місце PS).

5. Вибір модулів живлення. За умови задачі, живлення ПЛК проводиться напругою 220 VAC. Таким чином, в залежності від споживаної потужності модулів на шасі, необхідно вибрати один з двох модулів BMX CPS 2000 або BMX CPS 3500. Вважаємо, що вихід модуля для живлення датчиків та виконавчих механізмів (24V_SENOSRS) не використовується. Для кожного модуля шасі за таблицею 1.7 визначаємо середні споживані струми і вписуємо їх в таблицю 1.10.

В рядках з місцем PS в колонках 3V3_BAC та 24V_BAC вказуються сумарні споживані струми по кожному шасі, та споживані потужності ($I \cdot U$). По таблиці 1.6 та з використанням формули (1.2) визначається який з модулів живлення підходить для живлення даного шасі.

Як видно з таблиці, для шасі 00 та 01 задовольняє модуль живлення BMX CPS 2000, оскільки сумарна потужність по виходам 3V3_BAC та 24V_BAC не перевищує показники P_{3V3_24V} та P_{PS} для даного модуля. Для шасі 02 сумарна потужність по цим виходам 21.96 Вт, що більше ніж максимально дозволена в **BMX CPS 2000** (16,4 Вт), отже необхідно вибрати модуль живлення BMX CPS 3500.

Таблиця 1.10.

Шасі	Модуль		Середній споживаний струм, в мА	
	місце	Найменування модуля	На виході 3V3_BAC	На виході 24V_BAC
00	PS	BMX CPS 2000 $P_{3V3_24V} = 16,5$ Вт	$I_{3V3_BAC}=300$ мА, P=0.99Вт	$I_{24V_BAC}=260$ мА, P=6.24Вт
			$\Sigma P=7,23$ Вт	
	00	BMX P34 2020	0	95
	01	BMX DRA 0805	100	55
	02	BMX DRA 0805	100	55
	03	BMX DRA 0805	100	55
	XBE			
01	PS	BMX CPS 2000 $P_{3V3_24V} = 16,5$ Вт	$I_{3V3_BAC}=1480$ мА, P=4,88 Вт	$I_{24V_BAC}=460$ мА, P=11,04 Вт
			$\Sigma P=15,92$ Вт	
	00	BMX DDI 6402K	200	0
	01	BMX DAI 1604	90	0
	02	BMX DAI 1604	90	0
	03	BMX DAI 1604	90	0
	04	BMX DAI 1604	90	0
	05	BMX DRA 1605	100	95

02	06	BMX DRA 1605	100	95
	07	BMX DDO 1602	90	0
	08	BMX DDO 1602	90	0
	09	BMX DDO 6402K	240	0
	10	BMX AMO 802	150	135
	11	BMX AMO 802	150	135
	XBE			
	PS	BMX CPS 3500 $P_{3V3_24V} = 31,2 \text{ Вт}$	$I_{3V3_BAC}=1800 \text{ мА}, P=5,94 \text{ Вт}$	$I_{24V_BAC}=675\text{мА}, P=16,2 \text{ Вт}$
			$\Sigma P=21,96 \text{ Вт}$	
	00	BMX AMI 810	150	54
	01	BMX AMI 810	150	54
	02	BMX AMI 810	150	54
	03	BMX AMI 810	150	54
	04	BMX AMI 810	150	54
	05	BMX AMI 0410	150	45
	06	BMX AMI 0410	150	45
	07	BMX AMI 0410	150	45
	08	BMX AMI 0410	150	45
	09	BMX AMI 0410	150	45
	10	BMX AMI 0410	150	45
	11	BMX AMO 802	150	135
	XBE			

Результат. Перелік всіх необхідних складових М340 для даної задачі наведемо у вигляді відомості в таблиці 1.11.

Таблиця 1.11.

Відомість обладнання для компонування ПЛК М340.

По- зи- ція	Найменування	Кількість	Примітка
	BMX XBP 1200	2	шасі на 12 місць
	BMX XBP 0400	1	шасі на 4 місця
	XBE 1000	3	модуль розширення
	TSX TLY EX	1	комплект з двох термінаторів BusX
	TSX CBY 010K	2	BusX кабель 1 м
	BMX CPS 2000	2	модуль живлення 100...240 VAC, 20 Вт
	BMX CPS 3500	1	модуль живлення 100...240 VAC, 36 Вт
	BMX P34 2020	1	процесорний модуль, Serial + Ethernet
	BMX DDI6402K	1	дискретний вхідний модуль на 64 канали
	BMX DAI1604	4	дискретний вхідний модуль на 16 каналів
	BMX DRA 0805	3	дискретний вихідний модуль на 8 каналів
	BMX DRA 1605	2	дискретний вихідний модуль на 16 каналів
	BMX DDO 1602	2	дискретний вихідний модуль на 16 каналів
	BMX DDO 6402K	1	дискретний вихідний модуль на 64 канали
	BMX AMI 810	5	аналоговий вхідний модуль на 8 каналів
	BMX AMI 0410	1	аналоговий вхідний модуль на 4 канали
	BMX AMO 410	5	аналоговий вихідний модуль на 4 канали
	BMX AMO 802	3	аналоговий вихідний модуль на 8 каналів
	BMX FTB 2010	20	20 контактна з'ємна колодка з гвинтовими зажимами
	BMX FTB 2820	5	28 контактна з'ємна клемна колодка
	Telefast ABE 7H16R21	6	16-канальний клемний блок

BMX FCC053	3	кабель 0.5 м, з двома HE10 з'єднувачами
------------	---	---

2. ПРОГРАМУВАННЯ КОНТРОЛЕРІВ MODICON M340.

2.1. Загальні положення.

На сьогоднішній день для програмування контролерів Modicon Quantum, Premium, Atrium та M340 використовується єдине програмне забезпечення **UNITY PRO**. Цей інструментарій дає користувачу такі можливості:

1. Створення апаратної конфігурації та програми користувача для контролерів Modicon, а саме:
 - використання мультизадачного режиму: одна MAST, одна FAST, декілька EVT, одна AUX (тільки для QUANTUM та PREMIUM);
 - використання 5-ти мов програмування згідно стандарту MEK 61131-3: LD, ST, IL, FBD, SFC;
 - поділу програми користувача на секції (Section), кожна з яких може бути написана на різних мовах програмування MEK;
 - використання підпрограм (SR);
 - функціональне структурування проекту користувача;
 - доступу до великої бібліотеки функцій та функціональних блоків (FFB), які доступні на будь якій мові програмування;
 - створення функціональних блоків користувача (DFB);
 - використання поряд з локалізованими (located, прив'язаними до конкретної комірки пам'яті) та нелокалізованих (unlocated, не прив'язаних до конкретної комірки) даних;
 - використання масивів, структурних типів користувача;
2. Відладгодження програми користувача, а саме:
 - використання програмного емулятору (simulator) контролера з підтримкою більшості функцій UNITY та можливості доступу з інших програмно-технічних засобів до нього по Modbus/TCP;
 - анімації змінних безпосередньо в редакторах за допомогою кольору, відображення числових та текстових значень;
 - управління та контролю змінних за допомогою таблиць анімацій (Animation Table);
 - перегляду стану кроків мови SFC;
 - використання точок переривання (Break Point), покрокового виконання програми, точки спостереження (Watch point);
 - зміни програми користувача в режимі виконання контролером програми управління.
3. Експлуатації та обслуговування, а саме:
 - створення та використання графічних сторінок (Operator Screens) з анімацією технологічного процесу (подібно засобам HMI);
 - використання вбудованих діагностичних засобів для контролю стану будь якої частини контролера;

- використання вбудованого вікна тривоги Alarm Viewer для перегляду стану діагностичного буферу контролера;
- 4. Автоматичного створення документації по проекту.
- 5. Імпорту та експорту частин проекту в форматі *.XML для можливості їх використання в інших програмних засобах.

2.2. Структура програми користувача.

2.2.1. Задачі. Програма користувача M340 може виконуватись в контексті однієї або декількох *задач* (Tasks) та може включати обов'язкову основну задачу MAST та додаткові FAST та EVENTS. При конфігуруванні апаратного забезпечення до кожної задачі, "прив'язуються" входи, які будуть обпитуватися на початку виконання задачі, та виходи – в кінці виконання задачі. Кожна задача запускається на виконання тільки при умові знаходження ПЛК в *режимі виконання (RUN)*. У *режимі зупинки (STOP)* відбувається тільки циклічне опитування входів.

2.2.2. Задача MAST. У ПЛК завжди функціонує як мінімум одна задача – *MAST*, яка може виконуватись у циклічному чи періодичному режимах (рис.2.1). Обидва режими передбачають циклічне виконання програми задачі, яка починається зі зчитування входів та закінчується записом виходів. У *циклічному режимі*, початок наступного виклику задачі MAST починається відразу по закінченню обробки попередньої. Таким чином, час між викликами задачі MAST залежить від тривалості її виконання. У *періодичному режимі*, інтервал між викликами задачі задається в проекті UNITY PRO. Цей інтервал вибирається зазвичай більшим за максимальний час обробки задачі. Тобто, в більшості випадків, між викликами задачі MAST контролер буде виконувати внутрішню обробку (діагностичні операції, комунікаційний обмін, тощо).

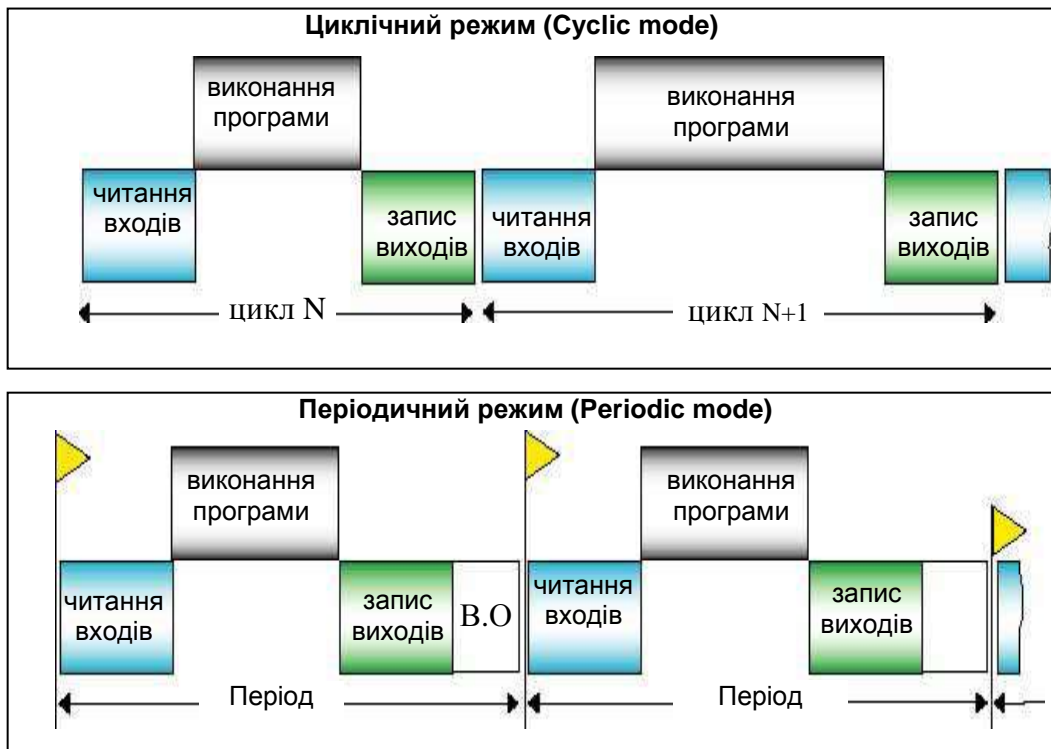


Рис.2.1.Циклічний та періодичний режим роботи задачі MAST

Програма задачі MAST представляє собою набір **секцій** (рис.2.2), кожна з яких може бути написана на будь-якій з мов MEK 61131-3: IL, LD, ST, FBD чи SFC. Секції виконуються одна за одною, в порядку розміщення їх в гілці Sections (див. рис.2.2). У свою чергу, програми в секціях можуть викликати підпрограми, які записуються в SR Sections цієї ж задачі.

MAST задача контролюється **сторожовим таймером** (WATCH DOG), який конфігурується в UNITY PRO. У випадку перевищення тривалості виконання задачі MAST за встановлене максимально допустиме значення, контролер перейде в режим STOP, що не допустить його "зависання".

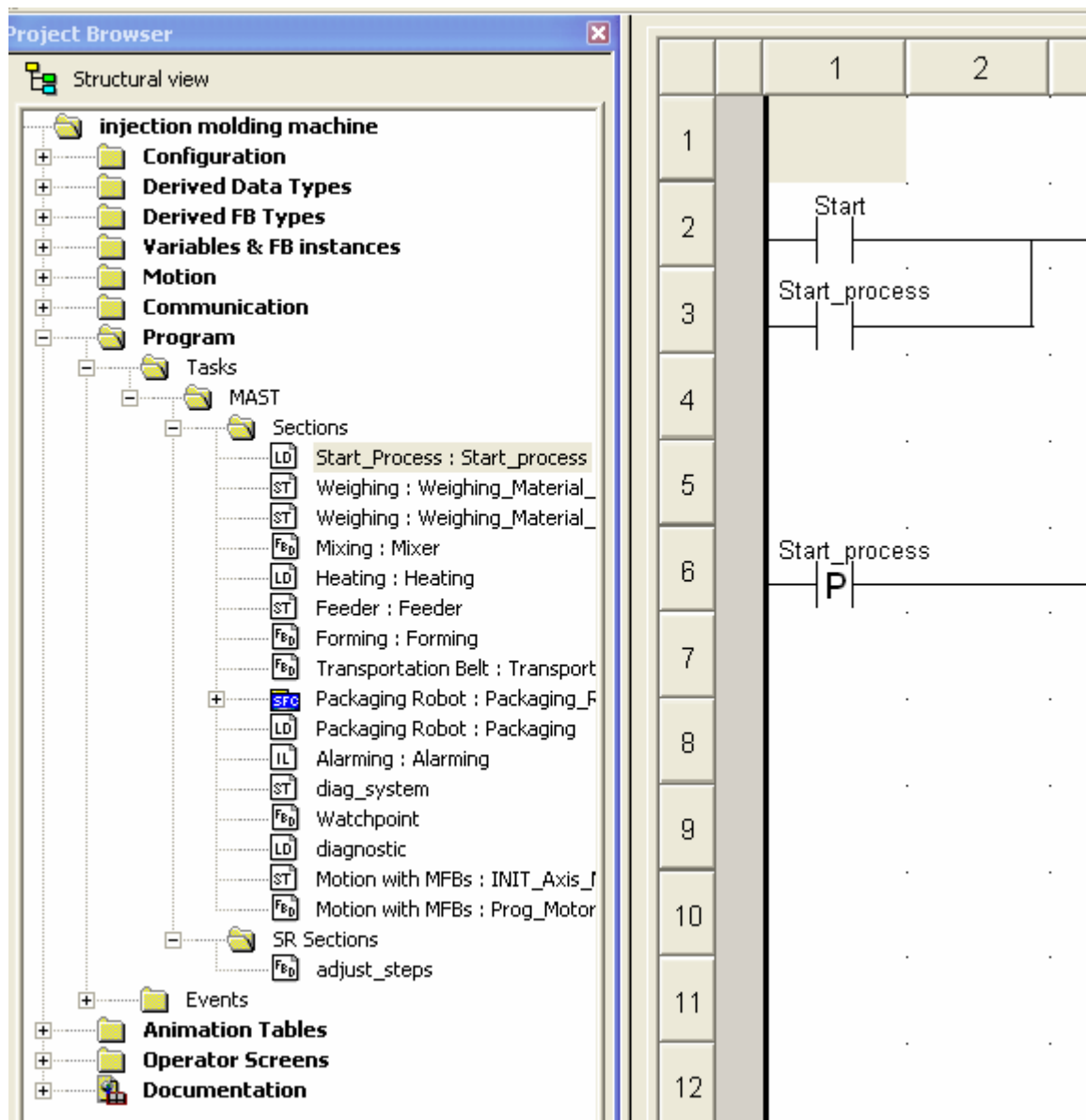


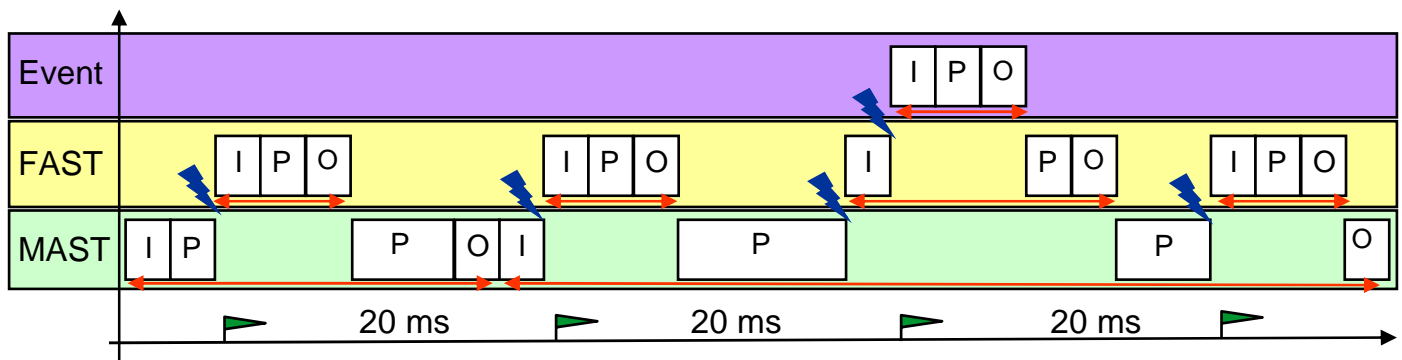
Рис.2.2.Приклад вигляду структури задачі MAST.

2.2.3. Задача FAST. У багатозадачній структурі програми M340, крім MAST можуть функціонувати також одна задача FAST та декілька задач Event. Задача **FAST** запускається завжди в періодичному режимі і має вищий пріоритет за задачу MAST. Вона призначена для процесів які потребують частішої обробки ніж ті, які обробляються в задачі MAST. Аналогічно до структури програми MAST, програма FAST також записується у секціях, однак використання мови SFC для задачі FAST не допускається.

2.2.4. Задачі EVENT. Задачі **Event** викликаються коли відбувається певна подія: **EVTi** – по апаратним подіям, **TIMERi** – по таймерним. Задачі EVTi (де i – номер задачі даного типу) прив'язуються до певної події, які фіксуються модулями спеціального призначення. Задачі TIMERi (де i – номер задачі даного типу) прив'язуються до таймера спеціального призначення, який запускається в програмі користувача спеціальною функцією ITCNTRL.

2.2.5. Пріоритетність задач. Задачі TIMERi мають вищий пріоритет ніж FAST, а EVTi – вище ніж TIMERi. Задачі з вищим пріоритетом будуть переривати виконання менш пріоритетних задач, які будуть продовжувати виконання задачі після закінчення виконання більш пріоритетної задачі. На рис.2.3 показаний

приклад роботи контролера в багатозадачному режимі, з тривалістю періода задачі FAST - 20 мс.



I – опитування входів "прив'язаних" до задачі; O – запис виходів;
P – виконання програми задачі

Рис.2.3. Приклад роботи контролера у багатозадачному режимі

2.3. Структура пам'яті M340

2.3.1. Розподіл пам'яті прикладної програми. Пам'ять, яку займає прикладна програма UNITY (Application), складається з таких розділів:

- локалізовані та нелокалізовані дані – дані користувача;
- системні дані;
- програма користувача, включно символи та коментарі;
- константи;

При увімкненому контролері ці дані розміщуються в оперативній пам'яті ROM процесорного модуля (рис.2.4). Додатково у ній також виділена область пам'яті, для можливості внесення зміни у програму користувача в режимі онлайн, тобто не зупиняючи роботу контролера.

Для збереження програми користувача та констант при вимкненому живленні контролера використовується SD карта. При завантаженні або зміні прикладної програми в контролері, вона автоматично зберігається на карті пам'яті SD, а при увімкненні контролера – зчитується з неї. Крім прикладної програми на карті зберігаються дані WEB-сервера для доступу до контролера через порт Ethernet, а також файли користувача (тільки для карт типу MPF). Для збереження значень локалізованих та нелокалізованих змінних при вимкненому живленні, використовується внутрішня флеш пам'ять процесорного модуля.

2.3.2. Локалізовані дані користувача. Програма користувача може оперувати локалізованими та нелокалізованими даними. Розміщення *локалізованих даних (located data)* у пам'яті наперед визначене, що дає можливість звернутися до них за адресою.

При створенні змінних, можна вказати комірку розміщення даних для неї в конкретній області, що дає можливість оперувати з локалізованими даними не за адресою а за символьним ім'ям змінної. Процес прив'язки змінних до конкретної комірки пам'яті будемо називати *локалізацією*. Змінні, які прив'язані до локалізованої області пам'яті будемо називати *локалізованими змінними*.

У залежності від призначення, локалізовані дані розміщені в декількох областях:

- **%M** – область даних для внутрішніх булевих (Boolean) змінних;
- **%MW** – область даних для внутрішніх числових змінних;
- **%S** – область даних для системних булевих змінних;
- **%SW** – область даних для системних числових змінних;
- **%I, %IW, %Q, %QW** – область даних асоційованих з каналами модулів ПЛК;
- **%KW** – область констант.

Області внутрішніх змінних призначені для довільного використання, наприклад, для збереження проміжних результатів розрахунку, мережного обміну, тощо. Кожна комірка області адресується унікальним номером, починаючи від 0 та закінчуючи номером останньої сконфігурованої змінної даної області. Область %M адресується побітно, а %MW - 16-розрядними словами. Так, наприклад, адреса наступної комірки після %MW16 є комірка з адресою %MW17.

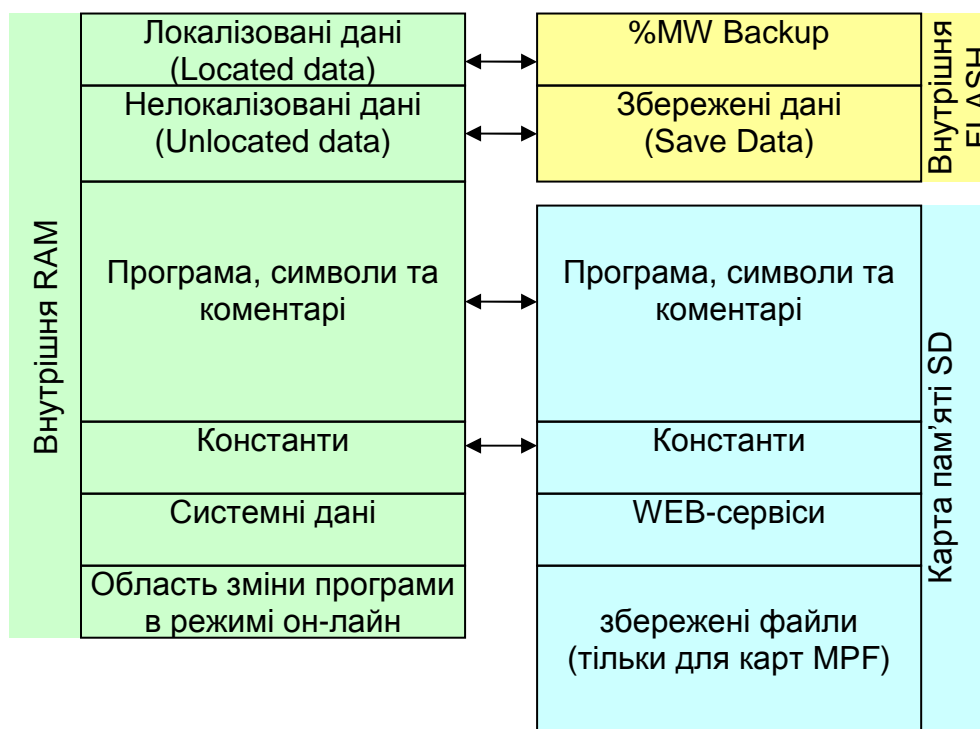


Рис.2.3. Структура пам'яті M340

У областях системних змінних %S та %SW знаходиться інформація про функціонування контролера. Кожна комірка має своє призначення і адресується номером. Так, наприклад, системний біт %S0 переводиться у стан логічної "1" при холодному старті (при увімкненні живлення), а на наступний цикл задачі MAST переводиться в "0". А у змінній %SW53 знаходиться інформація про поточний рік. Область %S адресується побітно, а %SW - 16-розрядними словами.

В області даних, що асоційована з каналами модулів ПЛК, знаходиться інформація про стан кожного каналу: числове значення, інформація про помилку, конфігураційна інформація, тощо. Адреса та розмір комірок цієї області пов'язані з адресами та типом каналів, за які вони відповідають. Більш детально адресацію каналів розглянемо в наступному підрозділі.

В області констант розміщені ті дані, які не можуть змінюватися в режимі виконання контролером програми користувача. Область %KW адресується 16-розрядними словами. Кожна комірка має унікальний номер, починаючи з 0.

2.3.3. Нелокалізовані дані користувача. Розміщення *нелокалізованих даних* (*unlocated data*) невідоме користувачу і вибирається середовищем UNITY PRO при кожній побудові (**BUILD**) проекту. Це значить, що звернення до нелокалізованих даних за адресою неможливе.

Якщо при описі змінної в UNITY PRO не вказується конкретно її розміщення в пам'яті, вона буде знаходитись в області нелокалізованих даних. Такі змінні будемо називати *нелокалізованими змінними*. Використання нелокалізованих змінних звільняє користувача від необхідності слідкування за виділенням області даних. Одним з недоліків використання нелокалізованих змінних є ускладнення при забезпеченні обміну ними по промисловим мережам.

Крім локалізованих змінних, в області нелокалізованих даних зберігаються екземпляри функціональних блоків EFB та DFB, які розглядаються у розділах 2.5 та 2.6 .

2.4. Робота з даними в UNITY PRO.

2.4.1. Типи даних. Прикладна програма користувача може оперувати даними наступним чином:

- звертаючись до них за адресою (тільки для локалізованих даних);
- через змінні, звертаючись до них по імені змінної;
- через **екземпляр** функціонального блоку;

У будь якому випадку у прикладній програмі UNITY операції з даними проводяться чітко згідно їх типу. **Тип даних** визначає структуру, формат, набір атрибутів та визначених операцій над цими даними. Всі типи даних UNITY можна поділити на чотири різні категорії:

EDT (Elementary Data Type) – елементарні типи даних ;

DDT (Derived Data Type) – похідні типи даних;

EFB (Elementary Function Block) – елементарні функціональні блоки;

DFB (Derived Function Block) – похідні функціональні блоки.

EFB та DFB функціональні блоки розглянуті в розділах 2.5 та 2.6.

Розглянемо елементарні та похідні типи даних.

В UNITY представлені наступні *елементарні типи даних (EDT)*: BOOL, EBOOL, INT, DINT, UINT, UDINT, BYTE, WORD, DWORD, REAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME, STRING, STRING[n]. Їх характеристика наведені у табл. 2.1.

Таблиця.2.1

Характеристика деяких елементарних типів даних (EDT).

Тип	Призначення	кількість біт, формат	Діапазон значень	форми відображення константи
BOOL	булевий	8	TRUE/FALSE	TRUE або 1 , FALSE або 0
EBOOL	булевий з можливістю	8	TRUE/FALSE	TRUE або 1 , FALSE або 0

	форсування та визначення фронтів			
INT	ціле, числові операції	16, 16-й біт - знак	-32768...32767	десятькова, двійкова (2#), вісімкова(8#), шіснадцяткова(16#), наприклад: 3 – десятикова; 2#0000000000000011 - :двійкова; 8#000003 - вісімкова; 16#0003 - шіснадцяткова.
DINT	подвійне ціле, числові операції	32, 31-й біт - знак	0...65535	десятькова, двійкова (2#), вісімкова(8#), шіснадцяткова(16#)
WORD	слово, побітові операції	16	16#0...16#FFFF	двійкова (2#), вісімкова(8#), шіснадцяткова(16#)
REAL	реальне, число з плаваючою комою	32, 31-й біт – знак, 8 біт експоненційна, 23 біт - мантіса	Нормальні значення: 3.4028235e+38... 1.1754944e-38, -0...0, 1.1754944e-38... 3.4028235e+38	0.456 аналогічно -1.32e12 1.0E+6 аналогічно 1000000 Не гарантований результат DEN : -1.1754944e-38 ... 1.1754944e-38 Нескінченність: < 3.4028234e+38 – INF >+3.4028234e+38 + INF QNaN та SNAN – невірний формат числа
TIME	беззнакове подвійне ціле	32, зберігає значення в мілісекундах	0...4294967295 мс, або T#0MS...T#49D_17H_2M_47S_295MS	Часова константа T# , де D – дні, H – години, M – хвилини, S – секунди, MS – мілісекунди, Наприклад: T#16S_500MS – 16,5 с

Похідні типи даних (DDT) – це складені типи даних, на базі елементарних типів. Це можуть бути масиви та структури, як вже визначені в UNITY та доступні через бібліотеку, так і визначені користувачем в процесі створення проекту. Змінні на основі похідних типів даних також будемо називати **структурними**.

2.4.2. Робота зі змінними в UNITY. UNITY дає можливість оперувати з даними через змінні. Змінні створюються у середовищі UNITY PRO у розділі "Variables & FB instances". При створенні змінної визначається її ім'я та тип. Ім'я повинно бути унікальним, містити символи латинського алфавіту, не містити службові символи та пробіли. Тип вибирається з доступних в UNITY елементарних та похідних типів.

Серед властивостей змінної доступна адреса розміщення даних (Address). Якщо при визначенні змінної, у властивості Address вказати комірку пам'яті (дивись локалізовані змінні користувача), то змінна буде локалізованою. В іншому випадку, змінна буде нелокалізована, отже адреса розміщення її буде невідомою, тобто до даних можна буде звертатися тільки через ім'я змінної. Для кожної змінної можна вказати початкове значення, тобто значення при ініціалізації.

Приклади вибору змінних та визначення їх властивостей наведені на рис.2.4. Для нелокалізованих змінних в полі Address (3 колонка) нічого не пишеться.

Масиви вибираються з визначенням типу елементів, а також початкового та кінцевого індексу. При локалізації масиву, вказується тільки адреса початкової комірки, всі інші комірки прив'язуються автоматично починаючи з вказаної. Кількість зайнятих комірок залежить від типу та кількості елементів масиву.

Типи даних DWORD, DINT, REAL, TIME займають два слова, тому при локалізації розміщуються в двох суміжних комірках.

Name	Type	Ad...	Value	Comment
Array1	ARRAY[0..1] OF INT			нелокалізована змінна-масив з 2-х елементів типу INT
Array1[0]	INT		16#23F2	0-й елемент масиву зі значенням ініціалізації в 16-ковому форматі
Array1[1]	INT		2568	1-й елемент масиву зі значенням ініціалізації в 10-ковому форматі
Array2	ARRAY[5..6] OF INT	%Mw100		змінна-масив з 2-х елементів типу INT, прив'язаний до комірок %Mw100 та %Mw101
Array2[5]	INT	%Mw100	2#000111100001...	елемент масиву з номером 5 зі значенням ініціалізації в 2-ковому форматі
Array2[6]	INT	%Mw101	100	елемент масиву з номером 6 зі значенням ініціалізації в 10-ковому форматі
Bool1	BOOL		TRUE	нелокалізована змінна типу BOOL
Ebool2	EBOOL		FALSE	нелокалізована змінна типу EBOOL
Bool3	EBOOL	%M200	TRUE	змінна типу EBOOL, прив'язана до %M200
Bool4	BOOL	%Mw200.7		змінна типу BOOL, прив'язана до 7-го біту слова %Mw200
Real1	REAL		16.5	нелокалізована змінна типу REAL
Real2	REAL	%Mw150	1.25e+4	змінна типу REAL, прив'язана до комірок %Mw150 та %Mw151
Int1	INT		2345	нелокалізована змінна типу INT
Int2	INT	%Mw160	16#ABCD	змінна типу INT, прив'язана до %Mw160
Int3	INT	%IW0.1.1		змінна типу INT, прив'язана до %IW0.1.1
Time1	TIME		T#25s350ms	нелокалізована змінна типу TIME зі значенням ініціалізації 25 секунд 350 мілісекунд
Time2	TIME	%Mw170	T#2h16m34s	змінна типу TIME зі значенням ініціалізації 2 год 16 хв 34 сек, прив'язана до комірок %Mw170 та %Mw171

Рис.2.4. Приклади змінних в UNITY PRO

При створенні структурних даних спочатку визначають структурний тип (DDT), а потім створюють на основі цього типу змінну. Для звернення до елементу структурної змінної (поля), вказують спочатку назву змінної, а потім через крапку - поле змінної.

Звернення до елементів масиву проводиться через квадратні дужки. Так наприклад для звернення до 0-го елементу масиву Array1 (див. рис.2.4) необхідно написати: Array1[0] .

Можливе побітове звернення до змінних. Для цього після назви змінної через крапку вказується номер біта. Наприклад для звернення до 7-го біта змінної Int1 (див. рис.2.4) необхідно записати: Int1.7 . Аналогічно можна звернутися також до біта у локалізованій області пам'яті. Наприклад запис %MW100.4, означає що йде звернення до 4-го біта комірки %MW100.

2.4.3. Адресація каналів вводу/виводу. Задачі MAST та FAST починаються зі зчитування даних з вхідних каналів ПЛК а закінчуються записом даних у вихідні канали. Процес зчитування та запису проходить автоматично, неявно для користувача. Прив'язка каналів до задач проводиться при конфігурації апаратної частини M340.

Інформація про стан та значення вхідних каналів M340 знаходиться у комірках %I та %IW, а значення вихідних – у комірках %Q та %QW. Кожна комірка з даних областей пам'яті відповідає за конкретний канал, в залежності від розміщення модуля у шасі та номеру каналу. Тобто топологічна адреса розміщення даних, які відповідають за значення каналу, визначається: номером шасі, на якому розташований модуль; розміщенням модуля у шасі та каналом на модулі. Таким чином:

%Ir.m.c – адреса відповідає за дискретний вхід на шасі з номером *r*, модулі на посадочному місці *m*, каналу *c*.

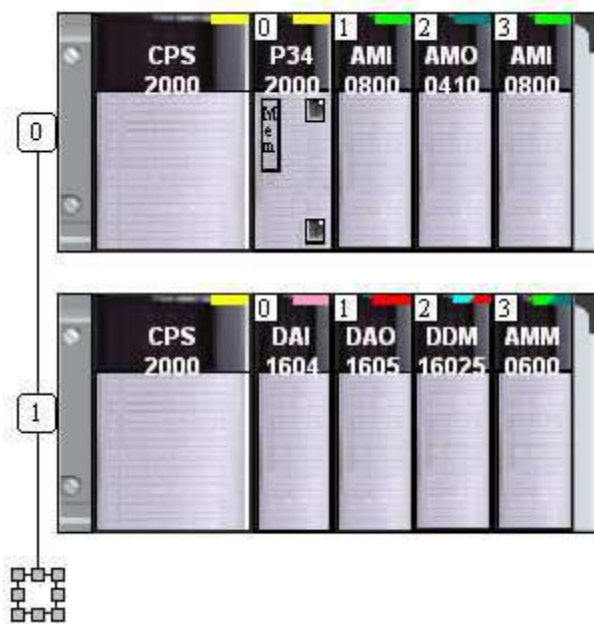
%IW $r.m.c$ - адреса відповідає за аналоговий вхід на шасі з номером r , модулі на посадочному місці m , каналу c .

%Q $r.m.c$ – адреса відповідає за дискретний вихід на шасі з номером r , модулі на посадочному місці m , каналу c .

%QW $r.m.c$ – адреса відповідає за аналоговий вихід на шасі з номером r , модулі на посадочному місці m , каналу c .

На рис.2.5 показані приклади топологічних адрес які відповідають за значення каналів. Для модулів з одностипними каналами (тільки входи або тільки виходи) номер каналу співпадає з порядковим номером входу або виходу (якщо рахувати з 0).

Для дискретних змішаних модулів, перші 16 каналів (0-15) виділяються під входи, а виходи починають адресуватися з 16-го. Навіть якщо у змішаному модулі тільки 8 входів (як на рис.2.5), адресація дискретних виходів все одно починається з 16-го. Для аналогових змішаних модулів АММ 0600, перші 4-ри канали (0-3) виділяються під входи, а виходи починають нумеруватися з 4-го.



номер шасі	Місце модуля	Тип модуля	Номер каналу	змінна
0	1	AMI 0800	0	%IW0.1.0
		
			7	%IW0.1.7
	2	AMO 0800	0	%QW0.2.0
		
			7	%QW0.2.7
	3	AMI 0800	0	%IW0.3.0
		
			7	%IW0.3.7
1	0	DAI 1604	0	%I1.0.0
		
			15	%I1.0.15
	1	DAO 1605	0	%Q1.1.0
		
			15	%Q1.1.15
	2	DDM 16025 (входи)	0	%I1.2.0
		
			7	%I1.2.7
		DDM 16025(виходи)	15	%Q1.2.15
		
			31	%Q1.2.31
	3	АММ 0600 (входи)	0	%IW1.3.0
		
			3	%IW1.3.3
		АММ 0600 (виходи)	4	%QW1.3.4
			7	%QW1.3.5

Рис.2.5. Приклади топологічних адрес, які відповідають за значення каналів вводу/виводу в UNITY PRO

2.5. Програмування на мові FBD

2.5.1. Загальні принципи побудови. Мова **FBD** (Function Block Diagram) - графічна мова програмування високого рівня, яка дозволяє створювати програми, як набір функцій, процедур та функціональних блоків, входи та виходи яких зв'язані між собою інформаційними зв'язками.

Загальний вигляд програми на мові FBD наведений на рис.2.6. Основними елементами програми є функції (Elementary Function), процедури (Procedure), елементарні функціональні блоки (Elementary Function Block) та похідні функціональні блоки (Derived Function Block). Для зручності опису FBD, прийнята загальна назва цих елементів – **FFB** (Function & Function Block). Графічно в редакторі FBD ці елементи зображені у вигляді прямокутників з входами і виходами.

Входи FFB є вхідними формальними параметрами, а виходи – вихідними. У якості фактичних параметрів на входи можна подавати змінні, топологічну адресу, літеральні константи, результат записаного ST виразу та зв'язок з виходом іншого FFB.

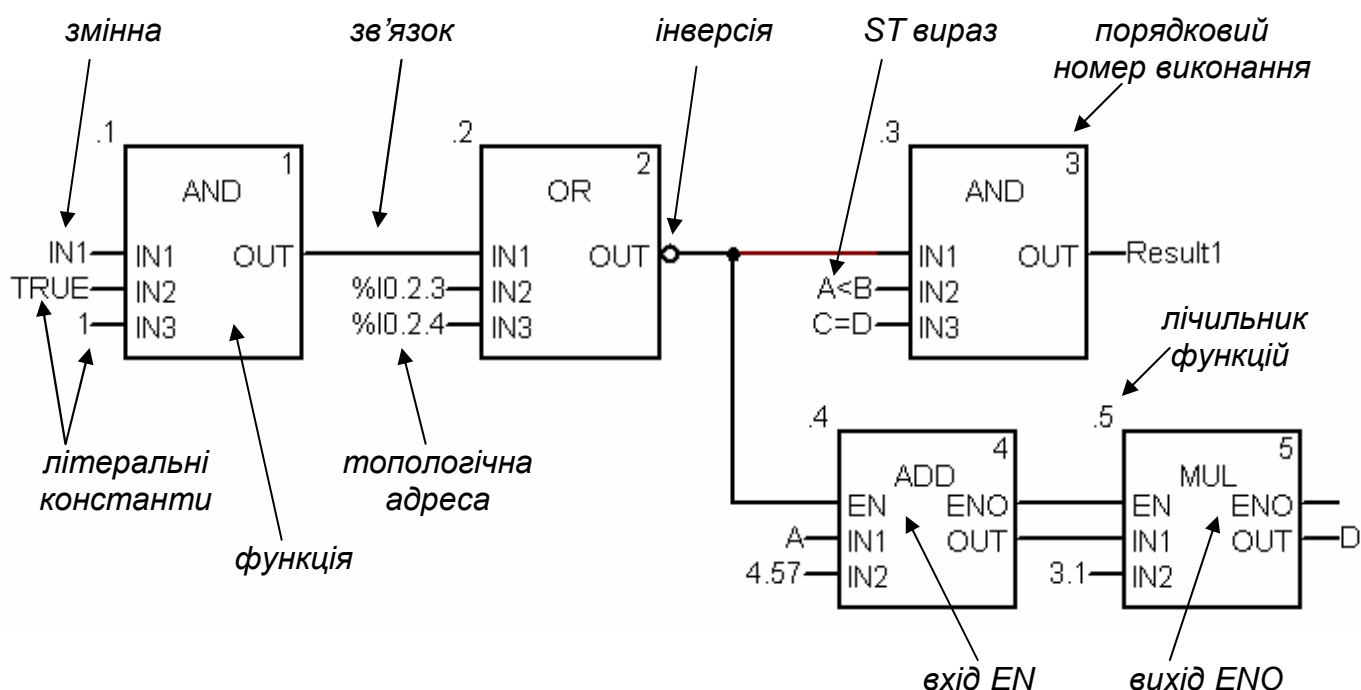


Рис.2.6. Приклад фрагменту програми на мові FBD

2.5.2. FBD зв'язки (link). Булеві та числові входи FFB можуть бути зв'язані з однотипними виходами інших FFB, організовуючи таким чином **FBD ланцюги** (Networks). У одній секції може бути декілька FBD ланцюгів. На рис.2.6 зображений тільки один ланцюг. Зв'язки забезпечують передачу розрахованих вихідних значень FFB, на прив'язані до них входи інших FFB. Один вихід FFB може бути зв'язаний з декількома входами FFB, формуючи таким чином розгалуження. Так на рис.2.6 вихід OUT функції OR, з'єднаний одночасно з входом IN1 функції AND та входом EN функції ADD. Один вхід безпосередньо можна зв'язати тільки з одним виходом іншого FFB. Типи входів та виходів повинні співпадати. Булеві входи та виходи можуть бути інвертовані, що позначається колом.

В UNITY не дозволяється використовувати зв'язки для параметрів FFB **строкового** типу. Також не дозволяється замкнення ланцюгів (зворотній зв'язок). Для вирішення задач з замкнутими ланцюгами використовуються проміжні змінні (рис.2.7).

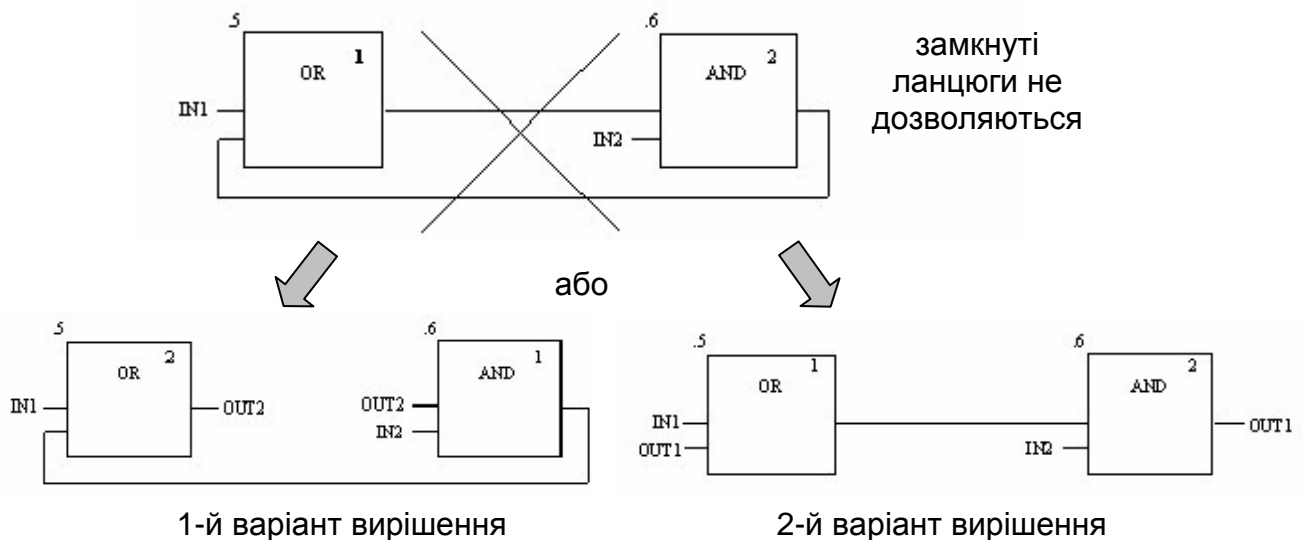


Рис.2.7. Приклад вирішення проблеми замкнутих ланцюгів

2.5.3. Ідентифікація FFB. Кожен FFB в межах секції повинен мати унікальну назву. При встановленні в FBD секції функції або процедури, вони отримують унікальний в межах секції ідентифікатор-лічильник (див. рис.2.6). Елементарні та похідні функціональні блоки ідентифікуються іменем екземпляру (див. розділ 2.6).

2.5.4. Послідовність виконання FFB. Кожний FFB має порядковий номер виконання, який встановлюється автоматично FBD редактором у залежності від: розміщення його в ланцюгу, взаємного розміщення ланцюгів в секції, взаємного розміщення прив'язаних входів та ін. Дозволяється явно змінювати послідовність виконання, якщо воно не перечить послідовності у ланцюгу.

2.5.5. Входи EN та виходи ENO. Вхід EN для FFB дає можливість управляти виконанням даного блока. Тобто FFB буде виконуватися тільки у тому випадку, якщо на вході EN буде логічна одиниця. На виході ENO формується логічна одиниця у тому випадку, якщо FFB буде оброблений без помилок. Таким чином вхід EN можна використати для логічного управління виконанням FFB. Так у фрагменті програми, зображеному на рис.2.6, функції ADD (додавання) та MUL (множення) будуть виконуватися тільки в тому випадку, коли на виході функції OR буде логічний нуль (зверніть увагу на інверсію).

Параметри EN/ENO є опціональними, і можуть бути сховані у редакторі FBD.

2.6. Бібліотека EF/EFB/DFB.

2.6.1. Функції, процедури та функціональні блоки. В програмі користувача можна використовувати різні типи програмних блоків, в які можна передавати фактичні параметри та отримувати на виході результати розрахунку. Можна користуватися як бібліотечними готовими блоками (функції, процедури, елементарні функціональні блоки) так і розробити власні (похідні функціональні блоки).

Функція (Elementary Functions - EF) - бібліотечний програмний блок, який може мати декілька входів і тільки один вихід. Функція не має внутрішньої пам'яті, тобто вона не може між викликами "всередині себе" зберігати розраховані значення.

Це значить, що вихід функції завжди однозначно залежить тільки від станів її входів.

Процедура (Procedure) – бібліотечний програмний блок, який може мати декілька входів і декілька виходів. Крім наявності декількох виходів, відмінність процедур від функцій є в можливості використання параметрів типу VAR_IN_OUT (вхід/вихід).

Елементарний функціональний блок (Elementary Function Block, **EFB**) – готовий бібліотечний програмний блок, який має внутрішню пам'ять, тобто може зберігати проміжні розрахункові значення між викликами. Перед використанням функціонального блоку у програмі користувача, спочатку створюють **екземпляр функціонального блоку** (FB instance), якому дають унікальне ім'я. Перегляд, створення та видалення функціональних блоків в UNITY PRO виконується у розділі "Variables & FB Instances" у підрозділі "Elementary FB Instances". Екземпляр функціонального блоку буде вміщувати всі внутрішні дані, тобто його пам'ять. Дані для екземплярів функціональних блоків розміщуються в нелокалізованій області пам'яті, тому за адресою до них звернутися неможливо.

Похідний функціональний блок (Derived Function Block, **DFB**) – функціональний блок, який розроблений користувачем засобами UNITY PRO. Спочатку в UNITY PRO у розділі проекту Derived FB Types розробляється **DFB тип** (DFB Type), на основі якого потім створюються екземпляри. При створенні DFB типу визначають: ім'я типу; інтерфейс блоку (вказують ім'я та тип входних та вихідних параметрів функціонального блоку); внутрішні змінні та внутрішні екземпляри функціональних блоків, в яких будуть зберігатися проміжні результати між викликами; створюють програму для функціонального блоку на мовах ST, IL, LD або FBD. Використання екземплярів DFB аналогічно як і EFB.

Виходи EFB та DFB є змінними екземпляру, тобто до них можна звертатись як до інших змінних. Для цього вказується ім'я екземпляру а потім через крапку назва виходу. Так наприклад запис "Timer1.Q" означає, що йде звернення до виходу Q екземпляру з іменем "Timer1".

2.6.2. Використання бібліотечних блоків. В **бібліотеці типів** UNITY (Types Library) доступна велика кількість функцій, процедур та елементарних функціональних блоків. Вони можуть бути використані в будь-якій із мов програмування. Так наприклад функція ADD призначена для додавання в мові FBD, однак її можна викликати і в ST, LD та IL, хоча для додавання там зручніше використовувати відповідний оператор "+".

Найбільш загальні програмні блоки зведені у розділі стандартної бібліотеки. Частина з них наведено в таб.2.2. Ряд блоків з бібліотеки управління, які використовуються при реалізації алгоритмів регулювання наведені у таб.2.3

Таблиця 2.2.

Стандартна бібліотека (Standard Library, часткова вибірка)

Математичні (сімейство Mathematics)		
ADD – додавання	SUB – віднімання	SQRT – квадратний корінь
MUL – множення	MOVE – присвоєння	

DIV – ділення	NEG – зміна знаку	
Порівняння (сімейство Comparison)		
EQ – дорівнює (=)	GE – більше або дорівнює (\geq)	LE – менше або дорівнює (\leq)
NE – не дорівнює (\neq)	GT – більше ($>$)	LT – менше ($<$)
Логічні (сімейство LOGIC)		
AND – логічне ТА	SR – SR тригер	FE – визначення переднього фронту
OR – логічне АБО	RS – RS тригер	RE – визначення заднього фронту
XOR – виключне АБО	SET – встановлення в лог. "1"	
NOT – логічне НІ	RESET – встановлення в лог. "0"	
Статистичні (сімейство Statistical)		
MUX – мультиплексор	LIMIT – обмеження, LIMIT_IND – обмеження з індикацією	MAX – вибір максимального
SEL – бінарний вибір	AVE – середнє значення	MIN – вибір мінімального
Таймери та лічильники (сімейство Timers & Counters)		
CTD – лічильник вниз	CTUD – лічильник вгору/вниз	TOF – таймер з затримкою на виключення
CTU – лічильник вгору	TP – таймер формування імпульсу	TON – таймер з затримкою на включення
Перетворення типів (сімейство Type to Type)		
INT_TO_REAL	WORD_TO_BIT	WORD_TO_INT
REAL_TO_INT	BIT_TO_WORD	INT_TO_WORD
Цілочисельне регулювання (сімейство CLC_INT)		
PID_INT – ПІД регулятор	PWM_INT – широтно-імпульсне перетворення	SERVO_INT – управління серводвигунами

Таблиця 2.3

Бібліотека управління (Control Library, часткова вибірка).

PI_B – базовий ПІ регулятор	PWM1 – широтно-імпульсна модуляція
PIDFF – повний ПІД регулятор	SERVO – управління серводвигунами
LAG_FILTER – фільтрація (аперіодична ланка)	SAMPLETM – шаблон часу для періодичного виклику блоків регулювання
SCALING – масштабування	DTIME – транспортне запізнення

2.6.3. Математичні функції. Математичні функції призначені для виконання таких базових операцій як додавання, віднімання, множення, ділення, присвоєння, а також тригонометричних, експоненційних та інших математичних функцій. Базові математичні операції в основному призначені для FBD, оскільки в інших мовах для цього використовуються відповідні оператори.

Більшість функцій у бібліотеці представлені для різних типів даних таких як INT, DINT, UINT, UDINT та REAL. Так для додавання цілих чисел можна використати функцію ADD_INT, або універсальну функцію ADD. Тим не менше, використання універсальної (за типом даних) функції не дозволяє використовувати в якості її параметрів різні за типом дані. Надалі ми будемо розглядати тільки універсальні за типом даних функції.

Всі розглянуті в таб.2.2 функції мають один вихід – результат виконання математичної функції. Функції SUB (віднімання) та DIV(ділення) мають по два входи, функції ADD (додавання) та MUL (множення) можуть мати від 2-х до 32-х входів (вибирається після встановлення), всі інші – один вхід. На рис.2.8 показаний приклад використання математичних функцій для реалізації залежностей:

$$X = \frac{A \cdot B \cdot C + D - E + \sqrt{F} + G}{H};$$

$$Y = -(A \cdot B \cdot C + D - E + \sqrt{F} + G); \quad (2.1)$$

$$Z = A \cdot B \cdot C + D - E + \sqrt{F} + G;$$

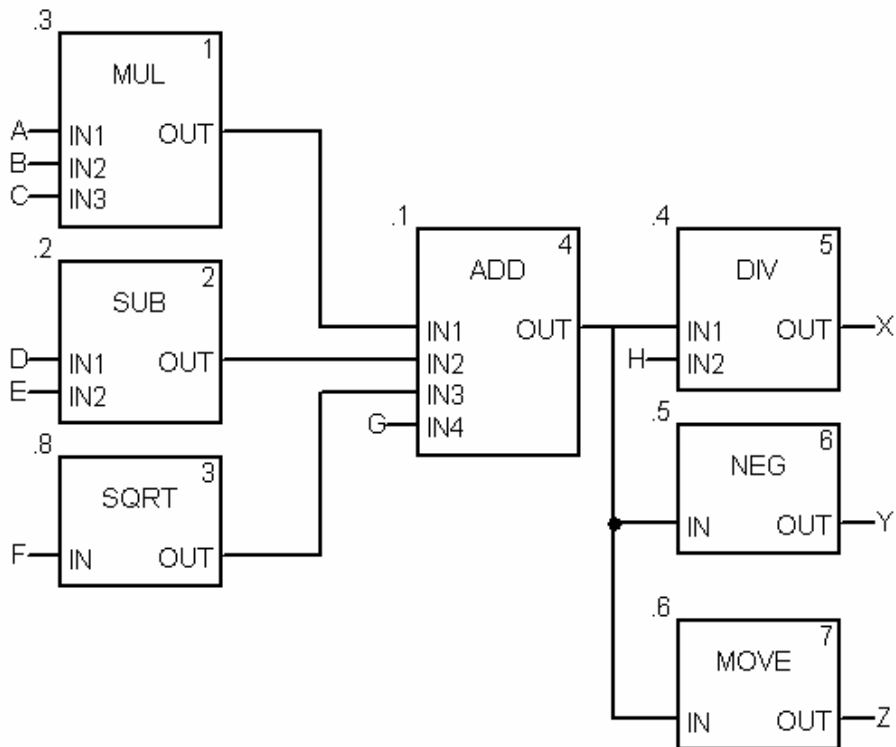


Рис.2.8. Приклад використання математичних функцій в FBD

2.6.4. Функції порівняння. Дані функції використовуються для порівняння двох або більше вхідних величин в мові FBD та LAD (однак дозволяється їх використання і в інших мовах). Результат порівняння є булевою величиною ("0" або "1"). За допомогою функцій порівняння та входів EN зручно проводити управління виконання логікою програми.

На рис.2.9 показаний приклад використання функцій порівняння для виконання логічного управління. При умові що $A=B=C$, змінній Y буде присвоєне значення 0. При умові, що $D \geq F$, виконається дія $Y := D + F$.

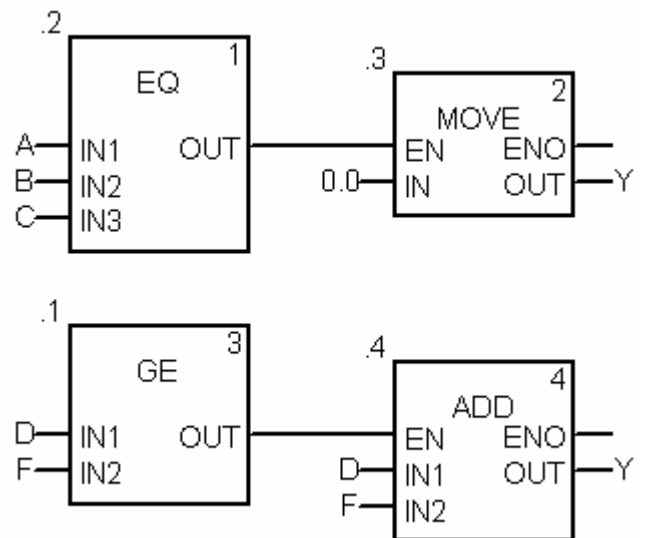


Рис.2.9. Приклад використання функцій порівняння в FBD

Альтернативою функціям порівняння в FBD може бути ST вираз, який повертає булевий результат. Так на рис.2.6 на вході блоку ".3" IN2 використовується результат порівняння $A > B$, а на вході IN3 - $C = D$.

2.6.5. Логічні блоки. Логічні функції AND, OR, XOR, NOT використовуються для логічних операцій переважно в мові FBD, так як у мові ST для цього використовуються оператори, а в LD – спеціальні графічні оператори. Приклад використання даних функцій наведений на рис.2.6. У FBD функцію NOT можна замінити знаком інверсії сигналу, як на виході блоку ".2".

Функції SET та RESET, не мають входів (за винятком EN). При виклику функції SET, вихідний параметр виставляється у логічну одиницю. Аналогічно, при виклику функції RESET вихідний параметр виставляється в логічний нуль.

Функції FE та RE служать для відлову відповідно заднього та переднього фронтів у мовах FBD, ST та IL. У якості вхідного параметру використовується змінні типу EBOOL.

Функціональні блоки SR та RS є тригерами. Тобто при поданні логічної "1" на вхід S на виході по передньому фронту встановлюється логічна "1", а на R – логічний "0". SR та RS тригери відрізняються пріоритетністю входів S та R при одночасному поданні на них логічних "1". У RS тригері пріоритетний вхід R, а у SR, – вхід S. Нагадаємо, що функціональні блоки потребують створення **екземплярів**.

На рис.2.10 показаний приклад використання логічних блоків в FBD. Змінна Bool1 встановиться в "1" в тому випадку, коли $A \geq B$ або $C > D$. В іншому випадку, ніяких дій зі змінною Bool1 проводитись не буде. Змінна Bool3 виставиться в "1", коли $C > D$ і встановиться у "0" по передньому фронту змінної ebool2, навіть якщо C буде більше D, оскільки вхід R1 має вищий пріоритет. Однак, якщо на наступний цикл після виникнення логічної "1" в ebool2, C буде більше ніж D, умова фронту сигналу вже не буде працювати, тому Bool3 знову переведеться в логічну "1". У разі виникнення переднього фронту у змінній ebool2 змінна B буде збільшуватися на 1.0.

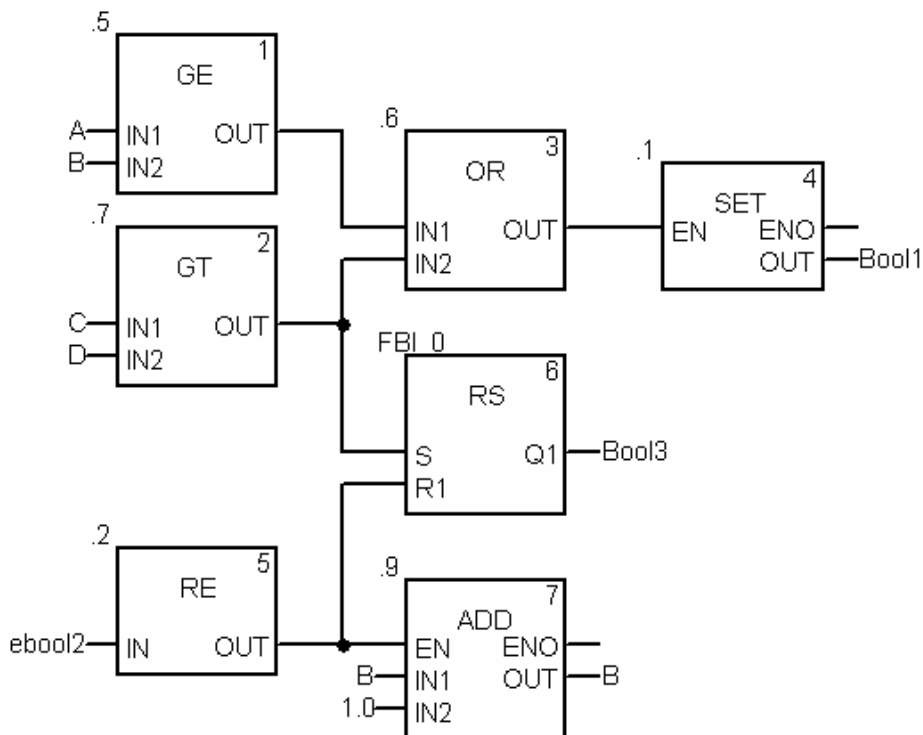


Рис.2.10. Приклад використання логічних блоків в FBD

2.6.6. Статистичні функції. Функції MIN та MAX призначені для запису відповідно мінімального та максимального значення серед вхідних сигналів у вихідний сигнал. Кількість входів може бути змінена до 32.

Функція MUX є мультиплексором, вихід OUT якої перемикається на один із входів IN0-IN30, по номеру що подається на вхід K. Тобто якщо K=2 OUT:=IN2.

Функція SEL перемикає вихід на один із двох входів IN, в залежності від значення вхідного параметру G. Тобто OUT:=IN0 при G=FALSE, OUT:=IN1 при G=TRUE.

Функція AVE реалізує розрахунок зваженого середнього за формулою:

$$Y = \frac{\sum (K_i \cdot X_i)}{\sum (K_i)} \quad (2.2)$$

де K_i – коефіцієнт i -го вхідного значення; X_i – i -те вхідне значення.

Коефіцієнт і значення вхідного сигналу являються парою вхідних параметрів. Тобто три пари сигналів потребують шість входів ($K_X1 \dots K_X6$). Непарні входи є коефіцієнтами, парні – значеннями сигналів.

Функція LIMIT забезпечує обмеження вхідної величини IN по мінімуму (віх MN) та по максимуму (вхід MX). Процедура LIMIT_IND працює аналогічно LIMIT однак має додаткові виходи Y_MAX та Y_MIN, який виставляється в логічну "1" при досягненні відповідно максимального та мінімального значень на вході.

Приклад використання статистичних функцій в FBD наведені на рис.2.11. Для кращого розуміння приклад показаний як в режимі редагування та і в режимі виконання програми з анімацією, де для всіх вхідних та вихідних параметрів показані числові значення. На виході блоку ".4" формується зважене середнє 3-х сигналів ($K_X2:=D$, $K_X4:=E$, $K_X8:=F$) з коефіцієнтами ($K_X1:=0.1$, $K_X3:=0.25$, $K_X5:=0.5$). Блок ".1" вибирає максимальне значення серед 4-рьох входів ($OUT:=98.0$), блок ".5" переключає вихід на перший вхід ($OUT:=IN1$). Серед двох вхідних сигналів блоку ".2" вибирається 0-вий ($OUT:=IN0$), оскільки $G=FALSE$ (для блока ".6" не справджується умова $IN1 < IN2$). Блок ".7" обмежує по максимуму вхідну величину ($MX:=95$), тому на виході $OUT:=95$, а на виходах $MN_IND:=FALSE$ та $MX_IND:=TRUE$.

2.6.7. Таймери та лічильники. Таймери та лічильники в UNITY реалізовані у вигляді функціональних блоків, тому потребують попереднього створення екземплярів.

Лічильники доступні у вигляді 3-х функціональних блоків: STU – підрахунок імпульсів зі збільшенням, CTD – підрахунок імпульсів зі зменшенням, CTUD – реверсивний лічильник.

Лічильник STU збільшує плинне значення CV, по передньому фронту сигналу на вході CU. На вході PV задається уставка. При досягненні плинного значення $CV \geq PV$, вихід Q:=TRUE. При подачі на вхід R:=TRUE, скидає плинне значення в нуль.

Лічильник CTD зменшує плинне значення CV, по передньому фронту сигналу на вході CD. На вході PV задається уставка. При досягненні плинного значення $CV \leq 0$, вихід Q:=TRUE. При подачі на вхід LD:=TRUE, у плинне значення записується значення уставки, тобто $CV:=PV$.

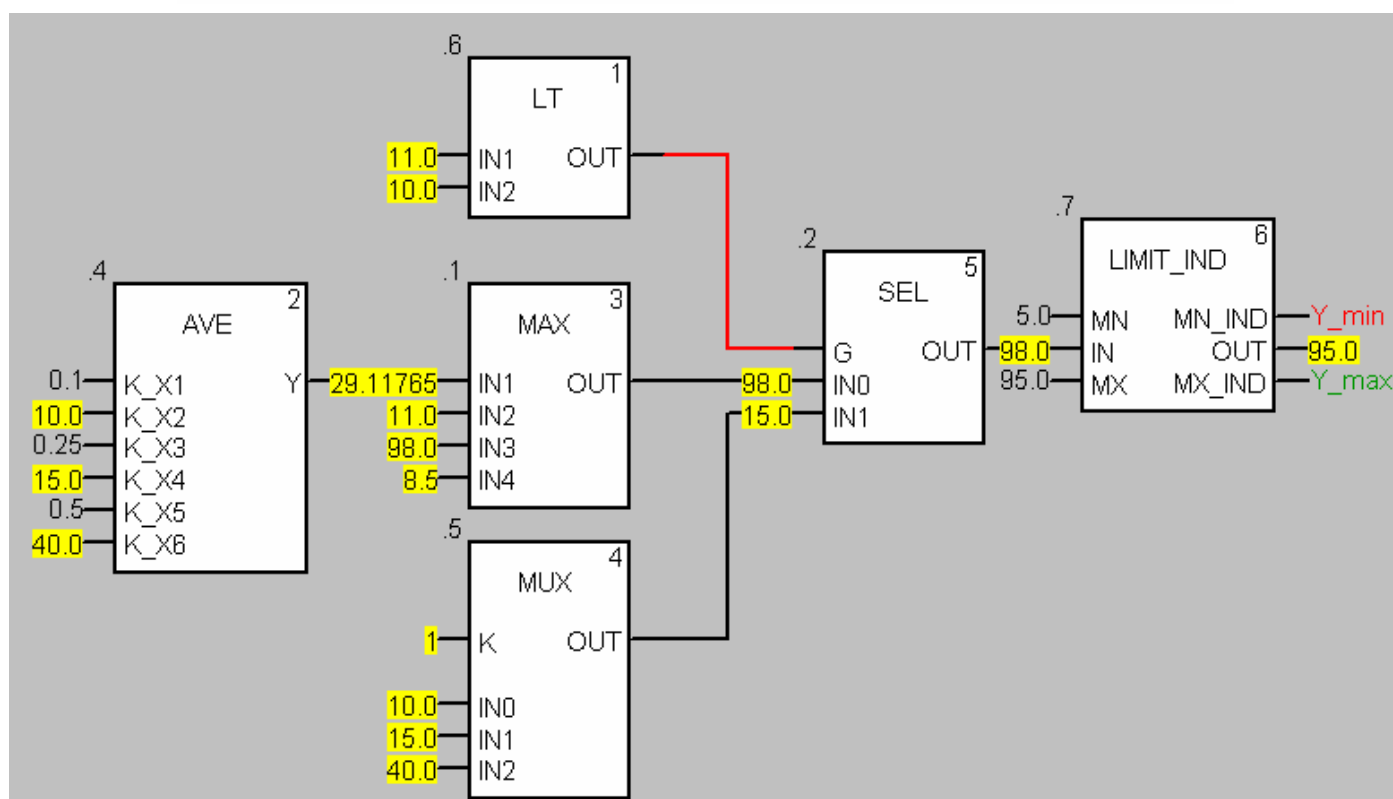
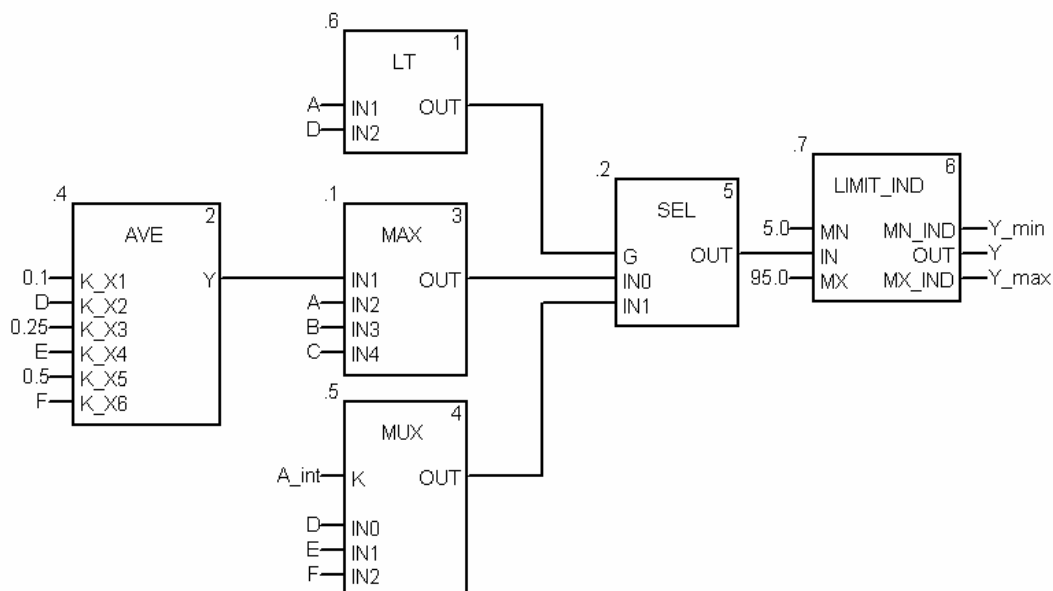


Рис.2.11. Приклад використання статистичних функцій в FBD: зверху – в режимі редактора, внизу – в режимі анімації

Лічильник CTUD, об'єднує в собі функції двох лічильників CTU та CTD. Входи CU та CD призначені відповідно для збільшення та зменшення плинного значення лічильника CV. При $CV \geq PV$, вихід $QU := TRUE$. При $CV \leq 0$, вихід $QD := TRUE$. Якщо $LD = TRUE$, $CV := PV$. Якщо $R = TRUE$, $CV := 0$.

Таймери TON, TOF та TP мають вхід IN – сигнал запуску таймеру, PT – часова уставка таймеру, вихід таймера Q, та плинне значення таймеру ET. Діаграми роботи даних таймерів показані на рис.2.12.

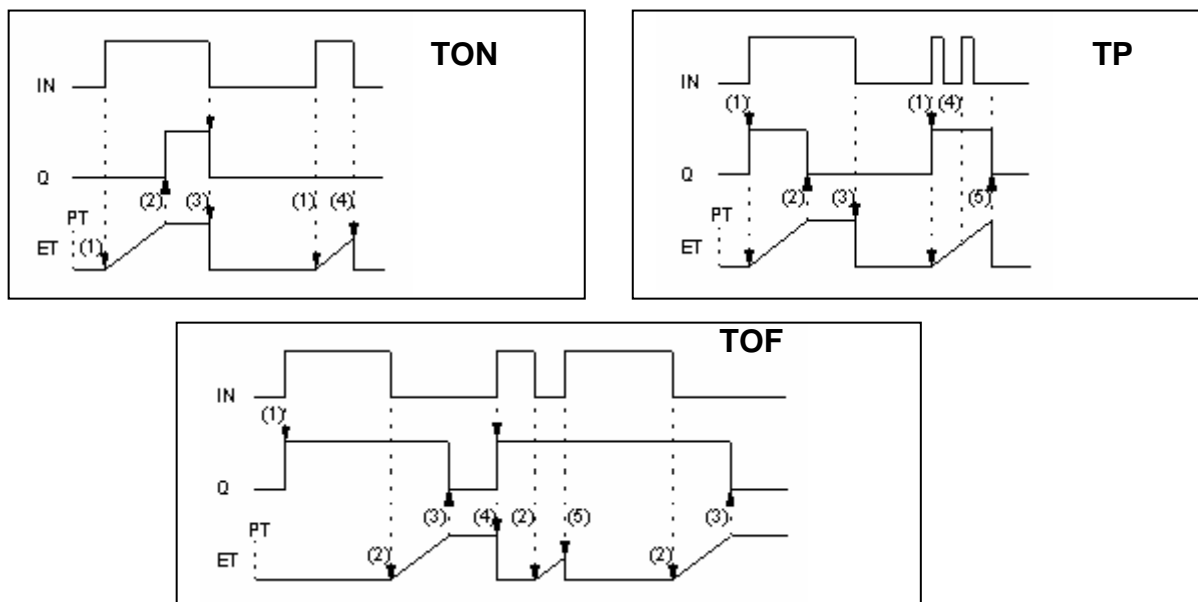


Рис.2.12. Графіки режимів роботи таймерів TON, TOF та TP.

Приклад використання лічильника CTUD та таймера TON в FBD показаний на рис.2.13. Вихід QU лічильника "CounterUD" спрацює при досягненні CV уставки (CV=20). Після досягнення уставки лічильника змінна Y_bool:=TRUE і запуситься таймер Timer1. Через 2с 100 мс після запуску таймера скинеться плинне значення лічильника "CounterUD", тобто CounterUD.CV:=0. Слід звернути увагу на використання виходу EFB "Timer1.Q" в якості фактичного параметру на вході R блоку "CounterUD".

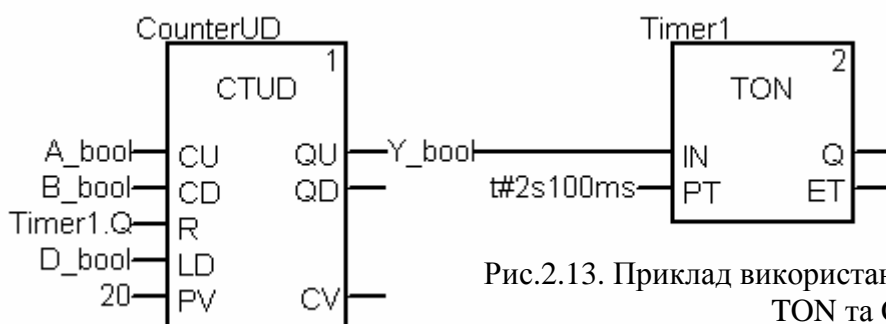


Рис.2.13. Приклад використання функціональних блоків TON та CTUD

2.6.8. Функції перетворення типів. Функція INT_TO_REAL перетворює ціле значення в дійсне, а REAL_TO_INT – навпаки. Дані функції необхідні у випадках, коли над одними даними треба проводити як цілочисельні операції так і операції з плаваючою комою. На рис.2.14 показаний приклад, де аналогову вхідну змінну необхідно помножити на коефіцієнт А (дійсне число) після чого результат подати на аналоговий вихід. Враховуючи, що значення аналогових входів записується у цілочисельні змінні %IW, необхідно спочатку його перетворити в тип REAL (функція INT_TO_REAL), після чого помножити на А. Враховуючи, що значення аналогових виходів зберігається в цілочисельних змінних %QW, розрахований добуток попередньо перетворюється в цілочисельний формат (функція REAL_TO_INT).

Функції INT_TO_WORD та WORD_TO_INT забезпечують перетворення відповідно цілочисельних даних INT в формат слова WORD та навпаки. Формат WORD використовується для побітових операцій над змінними.

Функція WORD_TO_BIT призначена для побітового розкладення входу IN типу WORD. Виходи BIT0...BIT15 виставляються в TRUE або в FALSE відповідно до значення бітів у вхідному слові. Біти у слові рахуються від молодшого (0-й) до старшого (15-й). Функція BIT_TO_WORD – навпаки, по значенням входів BIT0...BIT15 формує вихідне значення OUT типу WORD. На рис.2.14 показаний приклад, де значення змінної %IW0.1.1 інтерпретується як набір бітів. Спочатку значення перетворюється в тип WORD (функція INT_TO_WORD), після чого значення 0-го біту записується в a_bool, 1-го в b_bool, 7-го в c_bool.

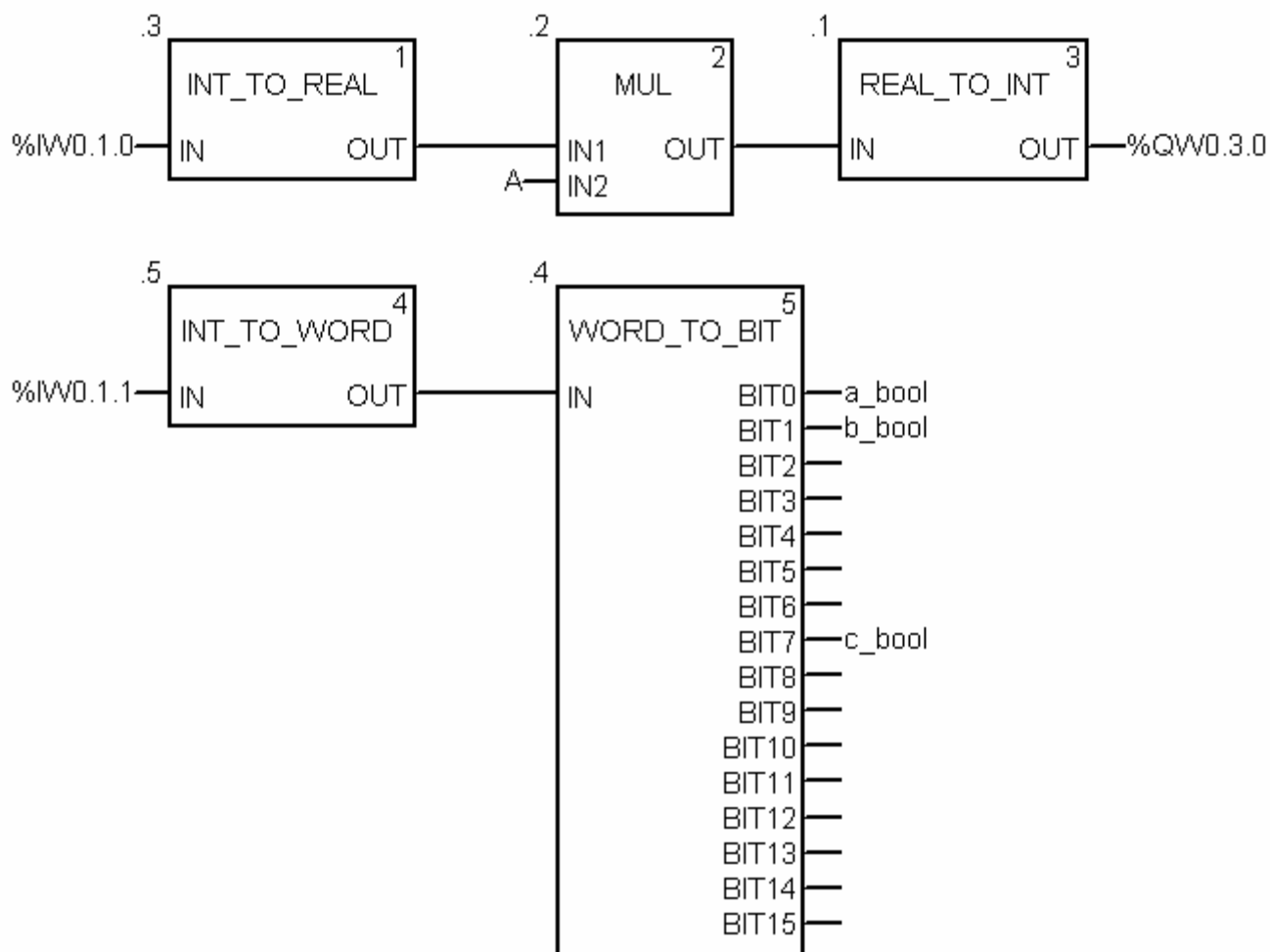


Рис.2.14. Приклад використання функцій перетворення типів в FBD

2.6.9. Цілочисельне регулювання. Процедури UNITY для цілочисельного регулювання PID_INT, PWM_INT та SERVO_INT аналогічні функціям PID, PWM та SERVO, які використовуються в PL7 (див. конспект лекцій "Промислові контролери", частина 2). Приклад використання функції PID_INT наведений на рис.2.15.

Вхід PARA що передбачає таблицю із 43-х параметрів, задається як масив:

PARA_AR[1..43] of INT

Призначення елементів масиву

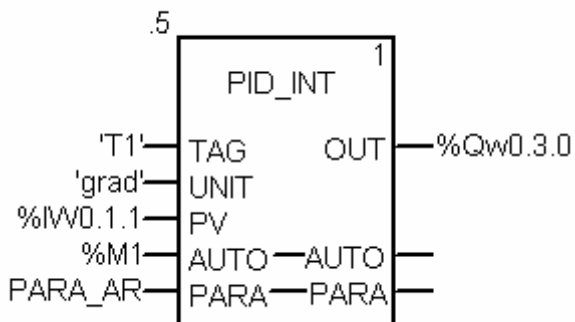


Рис.2.15. Приклад виклику функції PID_INT в FBD

співпадає з PL7.

2.6.10. SCALING (бібліотека Control Lib). Даний функціональний блок призначений для масштабування числових величин. Він реалізовує лінійну залежність вихідної величини (OUT) від вхідної (IN) за формулою:

$$OUT = (IN - in_min) \cdot \frac{(out_max - out_min)}{(in_max - in_min)} + out_min \quad (2.3)$$

Графічно залежність виходу OUT від входу IN показана на рис.2.16. Мінімальні та максимальні вхідні (in_min, in_max) та вихідні (out_min, out_max) величини, відносно яких проводиться масштабування, задаються у вхідному параметрі PARA DDT типу Para_SCALING. Тобто тип даних Para_SCALING включає 4-ри поля типу REAL для завання вхідних та вихідних меж, а також одне поле "clip" типу BOOL для визначення необхідності обмеження вихідної величини (див. рис.2.16).

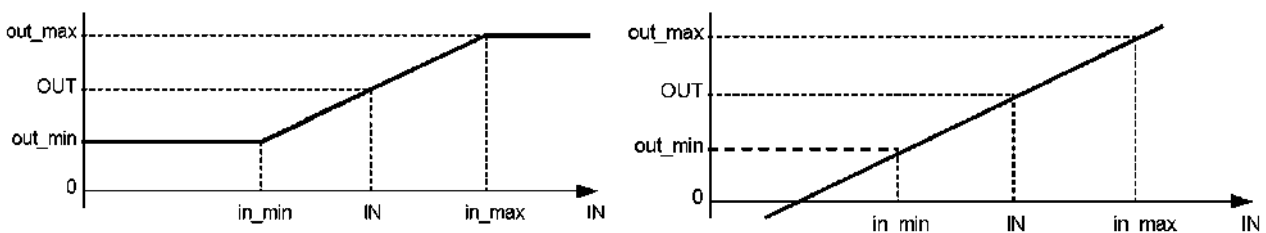


Рис.2.16. Залежність масштабованого вихідного значення (OUT) від вхідного (IN) при обмеженні на вихідний сигнал (зліва, Clip=1) та без обмеження (зправа, Clip=0) для блока SCALING.

Функціонування блоку продемонструємо на прикладі масштабування вхідного аналогового сигналу від датчика температури з діапазоном 0-150°C, який підключений до 1-го каналу 1-го модуля 0-го шасі (%IW0.1.1). Результат масштабування необхідно записати в змінну T1_R.

По замовченню, при опитуванні, сигнали від універсальних аналогових вхідних модулів перетворюються в діапазон 0-10000. Тобто вхідні межі будуть 0-10000, а вихідні 0-150. Для параметрів масштабування створюємо змінну T PARA типу Para_SCALING, властивості VALUE для полів заповнюємо відповідно до рис.2.17 (зверху). Присвоїмо поле clip:=TRUE для обмеження по мінімуму та максимуму вихідної (масштабованої величини).

Вигляд програми користувача на FBD показаний на рис.2.17 (внизу). Створюється екземпляр функціонального блоку SCALE_T1 типу SCALING. Попередньо %IW0.1.1 перетворюється в тип REAL, відповідно до типу параметру IN функціонального блоку SCALING. Вихідний параметр STATUS потрібен для контролю за помилками, в прикладі не використовується.

Name	Type	Value	Comment
T_PARA	Para_SCALING		
in_min	REAL	0.0	мінімальне вхідне значення
in_max	REAL	10000.0	максимальне вхідне значення
out_min	REAL	0.0	мінімальне масштабоване значення
out_max	REAL	150.0	максимальне масштабоване значення
clip	BOOL	true	активувати обмеження по виходу

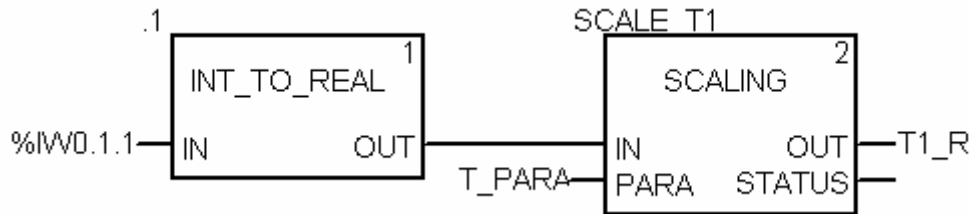


Рис.2.17. Приклад використання функціонального блока SCALING для масштабування аналогового вхідного сигналу: зверху – опис структурної змінної T_PARA типу Para_SCALING, знизу – приклад програми на FBD.

Аналогічним чином можна проводити масштабування вихідної величини.

2.6.11. Загальне представлення сімейства Controller (бібліотека Control Lib). Бібліотека UNITY нараховує велику кількість блоків для реалізації стандартних законів регулювання. Частина з них присутні у бібліотеці для сумісності з проектами, які конвертуються з середовищ CONCEPT та PL7. Це такі сімейства блоків:

- CLC_INT бібліотеки Base Lib (PID_INT, PWM_INT, SERVO_INT);
- CLC бібліотеки Obsolete Lib (PI1, PID1, PIDP1 та інші);
- CLC PRO бібліотеки Obsolete Lib (PI, PID, PID_P, PIP, PPI, PWM та інші).

Наведені сімейства блоків не рекомендується використовувати у новостворюваних проектах UNITY PRO. Для реалізації алгоритмів регулювання пропонується використовувати бібліотеку Control Lib зокрема блоки сімейств Controller, Output Processing, Setpoint Management. Сімейство Controller включає такі блоки:

- PI_V – реалізовує ПІ закон регулювання;
- PIDFF – реалізовує ПІД закон регулювання;
- AUTOTUNE – для автонастройки PI_V та PIDFF;
- SAMPLETM – для періодичного виклику алгоритмів регулювання;
- STEP2 – двохпозиційний регулятор;
- STEP3 – трьохпозиційний регулятор;

Серед блоків сімейства Output Processing можна виділити такі основні блоки:

- PWM1 – реалізовує ШІМ-алгоритм (широтно-імпульсна модуляція) для управління дискретними виконавчими механізмами по аналоговому сигналу та може бути використаний разом з PI_V або PIDFF;
- SERVO – реалізовує ШІМ алгоритм з двома дискретними виходами для управління виконавчими механізмами з прямим і зворотнім ходом (МЕК, МЕО) та може бути використаний разом з PI_V або PIDFF;

Блоки сімейства Setpoint Management призначені для забезпечення алгоритмів регулювання можливістю управління завданням. Зокрема функціональний блок

SP_SEL призначений для можливості переключення завдання з віддаленого на локальний при зв'язаному регулюванні.

Всі блоки бібліотеки Control Lib, алгоритм яких передбачає використання часових інтервалів (наприклад для інтегрування або диференціювання), розраховують ці інтервали як різницю між плинним та попереднім часом виклику блоку. Враховуючи значне використання ресурсів контролера даними блоками, для оптимізації роботи програми контролера рекомендується викликати їх періодично, зсунутими у часі відносно контурів, в яких вони використовуються. Так, наприклад, при наявності 10-ти контурів регулювання, можна викликати зв'язані в контурі блоки з періодичністю 100 мс, але зсунуті один відносно одного на один цикл. Тобто через кожні 50 мс, протягом 10 циклів будуть оброблені всі контури. Періодичний виклик зі зсувом по часу можна забезпечити функціональним блоком SAMPLETM.

2.6.12. SAMPLETM (бібліотека Control Lib). Функціональний блок SAMPLETM з періодичністю, яка визнається вхідним параметром INTERVAL, на один цикл задачі виставляє в значення TRUE вихід Q. Вхідний параметр DELSCANS визначає зміщення в циклах запуску внутрішнього таймеру блоку відносно першого циклу контролеру після холодного старту.

На рис.2.18 показаний приклад використання 2-х екземплярів SAMPLETM, виходи Q яких з періодичністю однієї секунди будуть виставлятися на один цикл в TRUE. Однак включення цих виходів буде зміщене на один цикл один відносно одного.

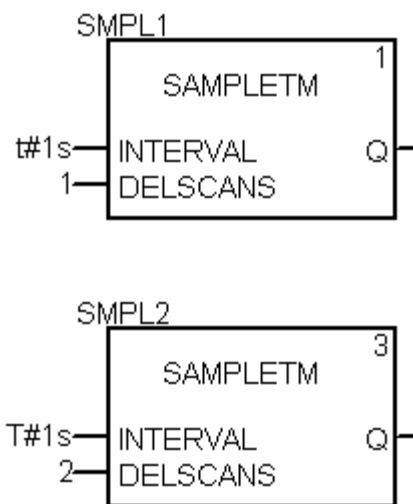


Рис.2.18. Використання 2-х екземплярів SAMPLETM зі зсувом на один цикл відносно один одного

2.6.13. PI_B (бібліотека Control Lib). Функціональний блок PI_B реалізовує ПІ алгоритм регулювання (2.4).

$$OUT = kp \cdot \left(1 + \frac{1}{ti \cdot p} \right) \cdot DEV \quad (2.4)$$

Функціональна структура ПІ-регулятора зображена на рис.2.19. Параметри PI_B наведені в таблиці табл.2.4. Як видно з рис.2.19, в алгоритмі спочатку розраховується розузгодження ($DEV = PV - SP$), яке пройшовши через блок нечутливості (з зоною dband), використовується для розрахунку інтегральної (налаштовується ti) та диференційної (налаштовується kp) складової. Задане значення SP обмежується по мінімуму (pv_inf) та по максимуму (pv_sup). Якщо необхідно управляти ВМ в зворотному напрямку (команда $rev_dir = TRUE$), розраховане значення інвертується (мінусує знак).

В автоматичному режимі ($MAN_AUTO = TRUE$) розраховане значення, пройшовши через блоки Tracking та Limiter подається на вихід OUT. В ручному режимі ($MAN_AUTO = FALSE$) вихід PI_B регулюється ззовні функціонального блока.

Режим Tracking використовується для прямого регулювання OUT зовнішнім алгоритмом. Він може бути використаний для ініціалізації виходу. У нормальному режимі TR_S=FALSE, тобто Tracking вимкнений.

Параметрами out_inf та out_sup відповідно налаштовується обмеження по мінімуму та максимуму вихідної величини.

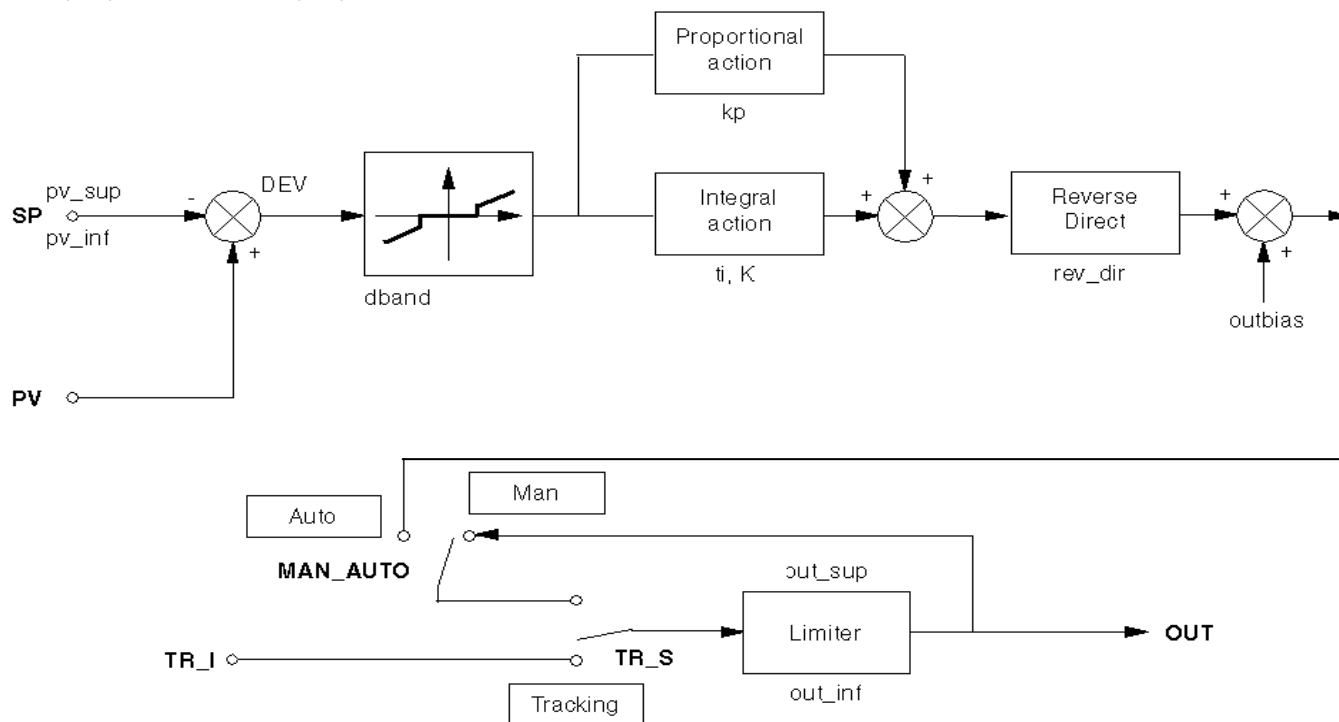


Рис.2.19. Функціональна структура ПІ-регулятора PI_V.

Функціональний блок PI_V може працювати в режим ПІ та П регулятора. Виставивши параметр ti=0s, PI_V переключається в режим П-регулювання, де вихідне значення буде розраховуватись за формулою:

$$OUT = kp \cdot DEV + outbias \quad (2.5)$$

Всі команди, плинні, задані та вихідні значення доступні як параметри функціонального блоку. Налаштування алгоритму проводиться через структурний параметр PARA типу Para_PI_V, поля якого описані в таб.2.7.

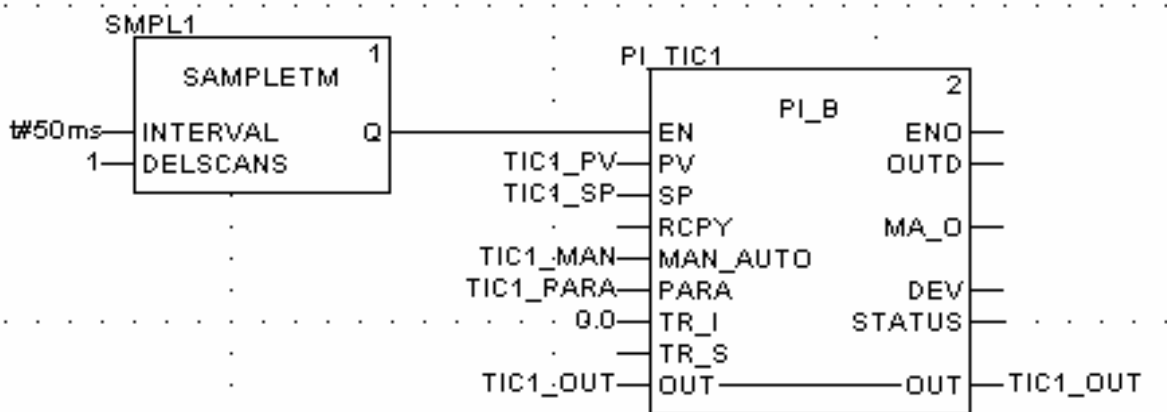
На рис.2.20 показаний приклад використання функціонального блоку PI_V для регулювання температури в контурі з умовною назвою TIC1. Для періодичного виклику ПІ-регулятора використовується функціональний блок типу SAMPLETM (див.2.6.12). Налаштування регулятора проводиться через структурну змінну TIC1_PARA, яка попередньо створюється на базі типу Para_PI_V.

Параметри функціонального блока PI_B.

Вхідні параметри		
PV	REAL	значення вимірювальної величини (плинне значення)
SP	REAL	задане значення (уставка)
RCPY	REAL	дійсне положення виконавчого механізму (використовується при управлінні серво-ВМ разом з EFB SERVO)
MAN_AUTO	BOOL	Режим роботи ПІ-регулятора: 1 : Автоматичний режим 0 : Ручний режим
PARA	Para_PI_B	Параметри регулятора (див. таб.2.7)
TR_I	REAL	Значення ініціалізації
TR_S	BOOL	Команда на включення ініціалізації (1: Включити ініціалізацію)
Вхідні/вихідні параметри		
OUT	REAL	Вихід ПІ-регулятора (в ручному режимі може змінюватися з зовні PI_B)
Вихідні параметри		
OUTD	REAL	різниця між вихідною величиною в плинному і попередньому циклах перерахунку PI_B
MA_O	BOOL	Плинний режим виконання ПІ-регулятора 1: Автоматичний режим 0: інший режим (ручний або режим ініціалізації)
DEV	REAL	Значення розузгодження (PV - SP)
STATUS	WORD	Слово статусу (використовується для контролю за помилками виконання PI_B)

Таб.2.7.Опис структурного типу Para_PI_B .

id	UINT	Використовується для алгоритму автопідстройки (AUTOTUNING)
pv_inf	REAL	обмеження по мінімуму вхідної величини завдання
pv_sup	REAL	обмеження по максимуму вхідної величини завдання
out_inf	REAL	обмеження по мінімуму вихідної величини
out_sup	REAL	обмеження по максимуму вихідної величини
rev_dir	BOOL	0: пряма робота ПІ-регулятора (PV-SP) 1: зворотна робота ПІ-регулятора (SP-PV)
en_rcpy	BOOL	1: використати вхід RCPY (тільки для управління серво-ВМ)
kp	REAL	Коефіцієнт пропорційності
ti	TIME	Час інтегрування
dband	REAL	Зона нечутливості
outbias	REAL	зміщення виходу регулятора в ПІ-режимі функціонування (при ti=0s)



Name	Type	/	Value	Comment
TIC1_MAN	BOOL			режим руч/авт для контуру TIC1
TIC1_PARA	Para_PI_B			Парметри настройки контуру TIC1 (стабілізація температури)
id	UINT			ідентифікатор (службовий, не заповнюється)
pv_inf	REAL		0.0	обмеження по мінімуму вхідної величини
pv_sup	REAL		100.0	обмеження по максимуму вхідної величини
out_inf	REAL		0.0	обмеження по мінімуму вихідної величини
out_sup	REAL		100.0	обмеження по максимуму вихідної величини
rev_dir	BOOL		false	робота в прямому/зворотному режимі
en_rcpy	BOOL		false	вкл/відкл RCPY
kp	REAL		1.0	коефіцієнт пропорційності
ti	TIME		t#5s	час інтегрування
dband	REAL		0.1	зона нечутливості
outbias	REAL		50.0	зміщення для П-режиму (при ti=0s)
TIC1_OUT	REAL			вихід на ВМ подачі пари
TIC1_PV	REAL			значення вимірювальної величини температури
TIC1_SP	REAL			задане значення температури для контуру TIC1

Рис.2.20. Приклад використання функціонального блоку PI_B: зверху – фрагмент програми, знизу – опис змінних в редакторі Data Editor .

1. <https://sites.google.com/site/fieldbusbook/realizacia-platformy/modicon>

Питання та пропозиції відправляйте на pupena_san@ukr.net