

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”



**Конфігурування ПЛК та програматора в SIMATIC Manager
Робота в S7-PLCSIM Simulating Modules.**

**Інструкція до лабораторної роботи №1
з курсу “Промислові Контролери”**

для студентів базового напрямку “Комп’ютеризовані системи
автоматики” та “Системна інженерія – Інтернет речей”

Львів 2025

Конфігурування ПЛК та програматора в SIMATIC Manager.

Робота в S7-PLCSIM Simulating Modules: Інструкція до лабораторної роботи №1 з курсу “Промислові контролери” для студентів базового напрямку “Комп’ютеризовані системи автоматики” та базового напрямку “Системна інженерія – Інтернет речей”/ Укл.: Яцук Ю. В., Павельчак А. Г. –Львів: Львівська політехніка. –2025. – 32 с.

Укладачі: Ю. В. Яцук – к.т.н., доцент
А.Г. Павельчак – к.т.н., доцент

Відповідальний за випуск: А. Й. Наконечний, д.т.н., професор

Кафедра “Комп’ютеризовані системи автоматики” Національного університету “Львівська політехніка” вважає своїм приємним обов’язком висловити сердечну вдячність фірмі “СВ-Альтера” за надану матеріально-технічну, наукову та організаційну підтримку в постановці нового курсу “Контролери систем керування”, зокрема в створенні нових лабораторних робіт та підготовці методичного забезпечення.

Особливо вдячні директору “СВ-Альтера Львів” Гануляку Андрію Михайловичу та начальнику відділу Петришину Мар’яну Дмитровичу за цінні поради та рекомендації, що сприяли суттєвому покращенню інформаційного та методичного забезпечення

Мета роботи: отримати навички роботи в системі автоматизації SIMATIC та освоїти конфігурування програмованих логічних контролерів та модулів розширення для них, навчитися тестувати розроблені програми в середовищі S7-PLCSIM Simulating Modules.

Вступ

Для роботи машин, механізмів і процесів майже завжди на всіх виробничих ділянках поряд із подачею енергії необхідні і елементи керування. В будь-якій машині або устаткуванні необхідно мати можливість розпочинати, керувати, спостерігати і завершувати виробничий процес. Сьогодні для вирішення задач автоматизації виробництва використовуються системи керування на основі програмованих контролерів ПЛК.

Система промислових контролерів Vipa об'єднує в собі серії 100V, 200V, 300S та 500S. Різні за потужністю та функціональністю ці контролери дозволяють вирішувати широкий спектр задач автоматизації, і можуть бути інтегровані у виробничі системи керування завдяки широким комунікаційним можливостям. Крім цього, система 300S з технологією Speed7 є революційним за швидкодією продуктом, який з легкістю може використовуватися в задач з дуже високою швидкістю реакції (<1 мс).


Програмування сучасних ПЛК здійснюється за допомогою спеціалізованих програмних продуктів - інтегрованих середовищ розробки - **IDE (Integrated Development Environment)**, опрацьованих виробниками ПЛК та сумісних з продуктованими моделями ПЛК. Прикладом такого **IDE** є програмне середовище компанії Siemens - **STEP 7 Simatic Manager**.

Для блочного програмування STEP7 Simatic Manager використовує мови, що відповідають DIN EN 6.1131-3. Це LAD (ladder logic або ladder diagram – це мова, яка за своїм виглядом нагадує електричні схеми релейної логіки), FBD (function block diagram – діаграми функціональних блоків) та мова STL (statement list – список операторів або список мнемонік, асемблероподібна мова). Також можливим є використання мови SCL (structured control language – структурована мова керування, паскалеподібна мова програмування високого рівня). Різне представлення програмного коду дозволяє кожному користувачу вибрати найбільш зручний для його розуміння опис функції керування. Саме ці широкі можливості представлення вирішуваної задачі управління значно спрощує програмування ПЛК класу Siemens та Vipa.

1. Запуск SIMATIC Manager та створення нового проекту

SIMATIC Manager – це графічна оболонка користувача для роботи з об'єктами STEP7 (проекти, файли з програмами користувача, блоки, HW-Station і допоміжних програм). Програмний продукт SIMATIC використовується як для програмування програмованих логічних контролерів (ПЛК) фірми Siemens (LOGO!, S7-200, S7- 300/S7-400), так і тих фірм виробників, ПЛК яких є сумісні за системою команд з ПЛК S7-300/S7-400, наприклад, ПЛК фірми Vipa.

Сімейство контролерів Vipa складається з лінійки ПЛК, які призначені для виконання як простих, так і складних задач. Незважаючи на різні геометричні розміри, всі ПЛК даної лінійки схожі за експлуатаційними характеристиками, організацією пам'яті, структурою даних, адресуванням, мовами програмування і списком інструкцій.

Запуск SIMATIC Manager відбувається шляхом подвійного натискання лівої кнопки миші (Double Click) на іконці , що знаходиться на робочому столі, або через стартове меню Пуск→SIMATIC→SIMATIC Manager.

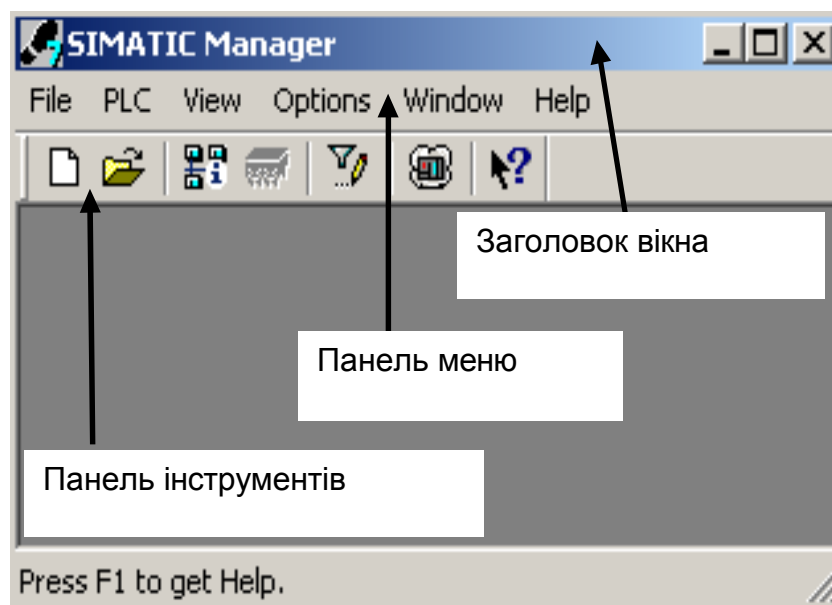



Рис.1. Вигляд вікна SIMATIC Manager

Заголовок вікна містить назву проекту і кнопки управління вікном програми. **Панель меню** містить меню, доступні для даного проекту. **Панель інструментів** відображає кнопки, які найчастіше використовуються.

Програмування розпочинається з відкриття або створення нового проекту (project). Для створення нового проекту використовуються команда з меню

File→New (Файл→Створити новий) або іконка із зображенням чистого аркуша паперу . Далі у вікні створення нового проекту New Project (рис. 2), в графі “Ім’я” (Name) вказується ім’я проекту, наприклад “Lab1”. В полі “Шлях розташування” (Storage location (path):) вказується шлях до папки, в якій зберігатиметься проект. За замовчуванням шлях для зберігання проектів є “C:\Program Files\SIEMENS\STEP7\s7proj”. Після введення назви і шляху збереження проекту натискаємо кнопку OK.

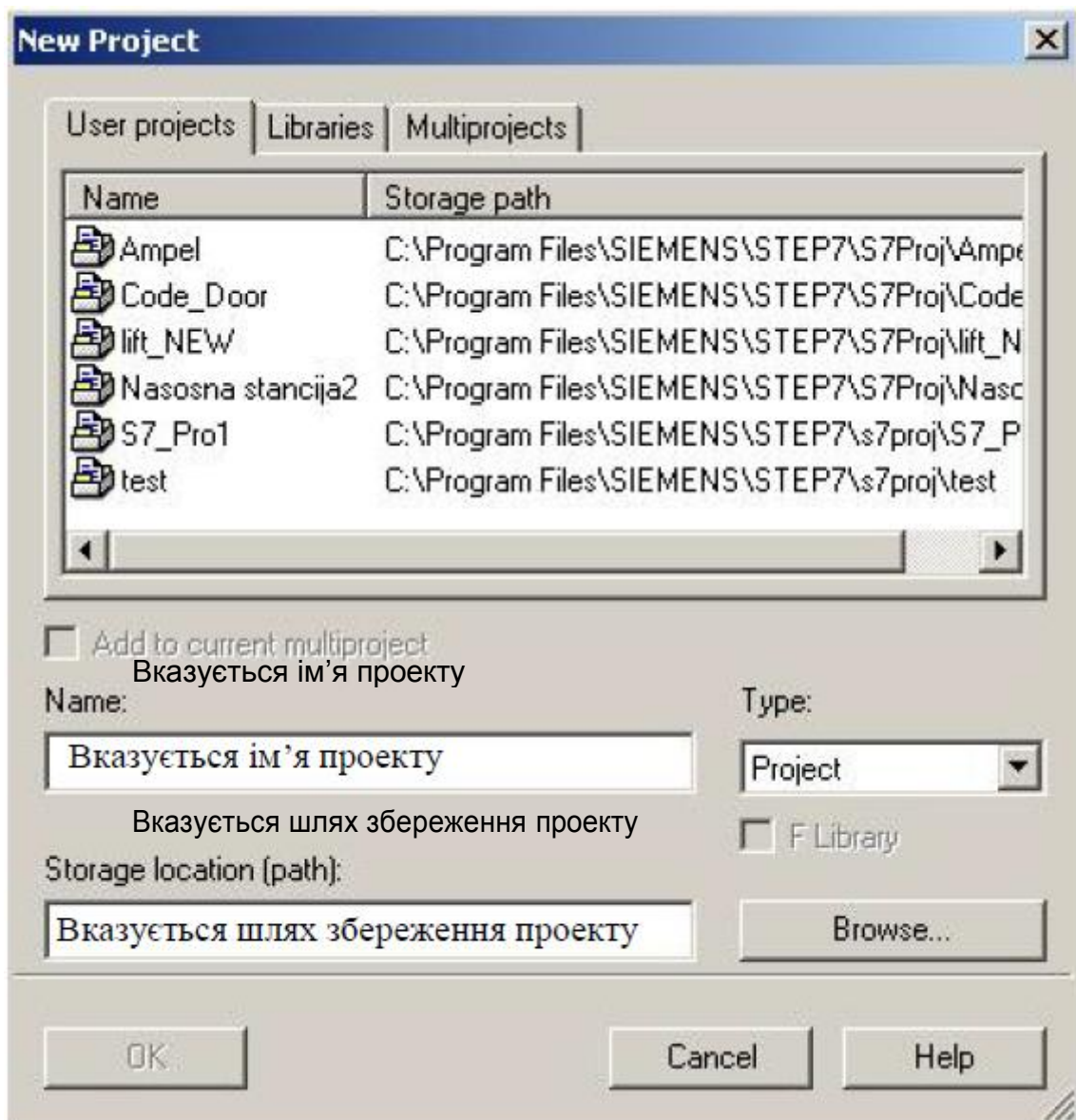


Рис.2. Вікно створення нового проекту

Дані в проекті зберігаються в об’єктах. Об’єкти в проекті розташовуються у вигляді дерева (ієрархічна будова) (рис. 3).

Вікно проекту розділене на дві частини: ліва частина – структура проекту; права частина – відображає вміст виділених об’єктів лівої

частини. На рис.3 у лівій частині вікна проекту виділено об'єкт Blocks (блоки), у правій частині вікна відображено вміст цього об'єкту.

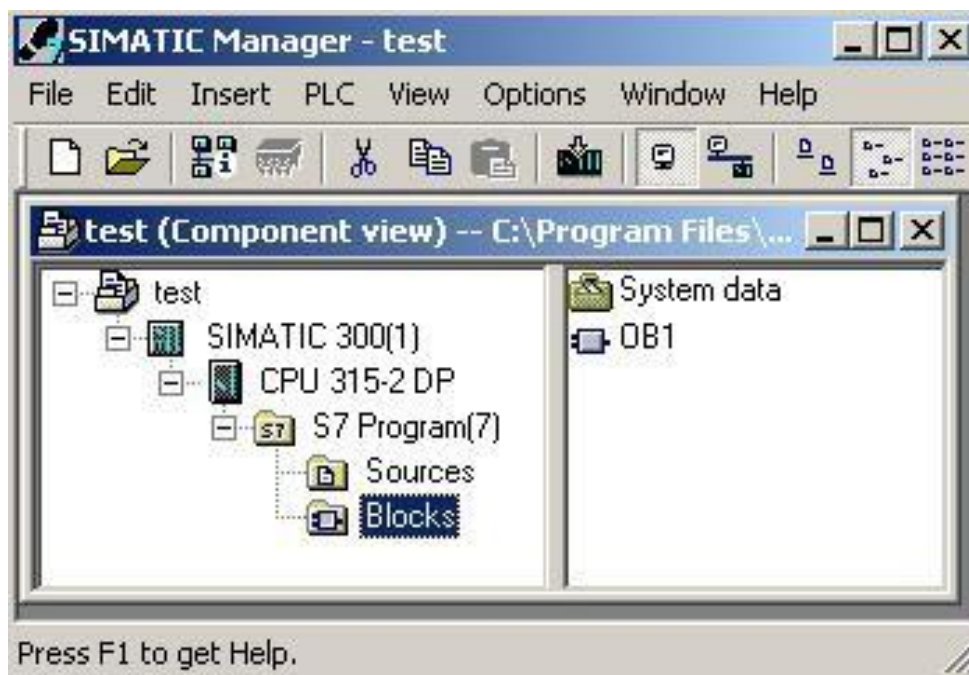


Рис. 3. Вигляд вікна SIMATIC Manager з проектом

Натискання по квадратику зі знаком плюс (мінус) в лівій частині вікна призведе до відображення (приховування) додаткових рівнів структури проекту. Вибір об'єктів в лівій частині вікна відобразить його вміст в правій частині.

Ієрархія проекту.

1-й рівень: Проект. Кожен проект представляє базу даних, в якій зберігаються дані.

2-й рівень: Станція (наприклад, SIMATIC 300). Зберігається інформація про конфігурацію апаратури. Станція відповідає програмованому контролеру, тобто охоплює як ЦПУ, так і дискретні, аналогові та функціональні модулі. Проект може вмішувати декілька станцій, зв'язаних між собою підмережею (наприклад PROFIBUS).

3-й рівень: ЦПУ. Містить програму користувача та таблицю з'єднань.

4-й рівень: Програма S7. Містить все програмне забезпечення поставленої задачі автоматизації.

5-й рівень: Вміщує блоки користувацької програми (OB, FC, DB та інші).

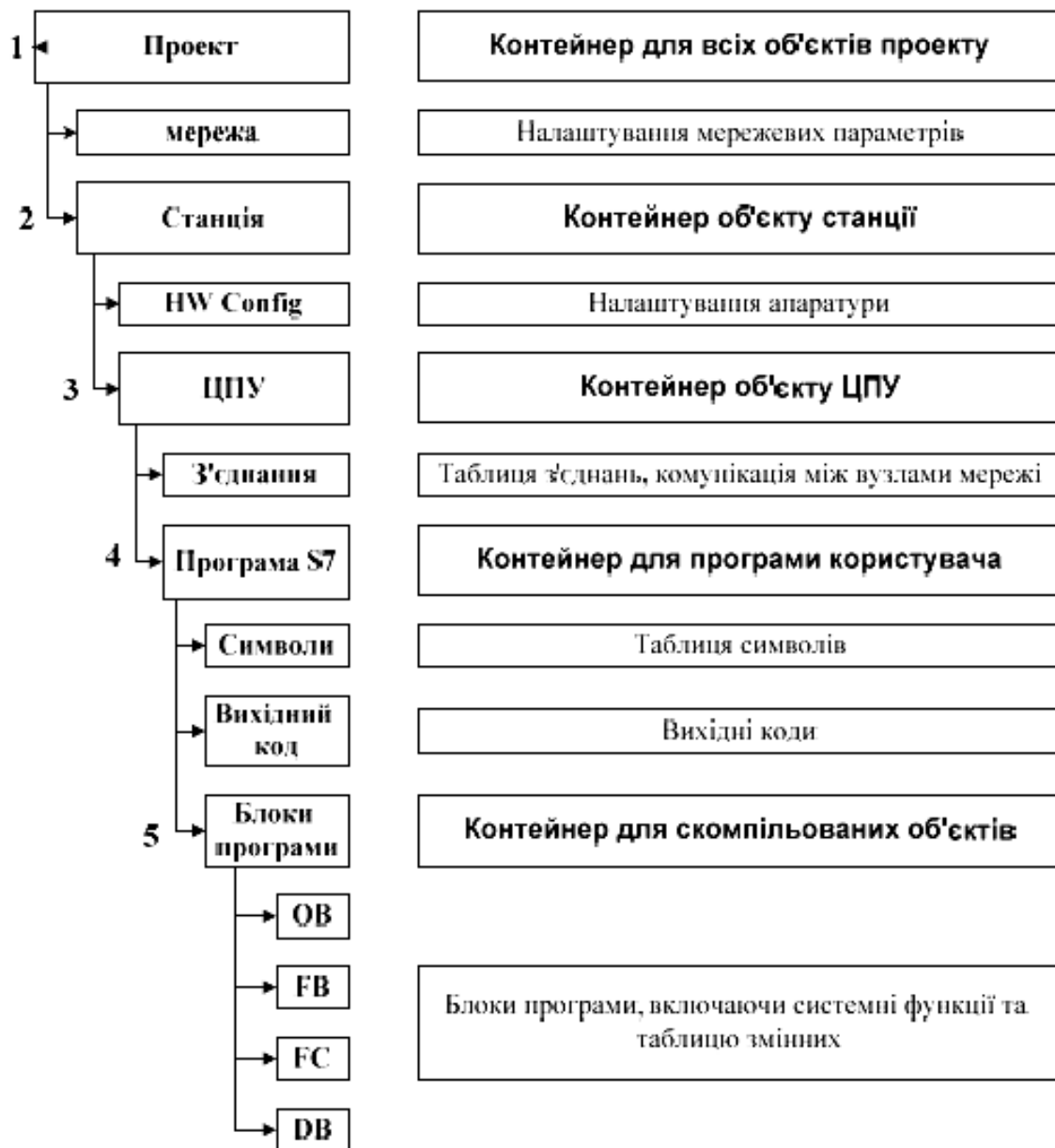


Рис.4. Структура проекту STEP7

1. Модуль HW Config

Для конфігурування ПЛК використовується модуль **HW Config**. Під “конфігуруванням” розуміється розміщення стійок, дискретних та аналогових, інтерфейсних та комунікаційних модулів у вікні станції. Конфігурування виконується автономно без підключення до ПЛК. Аналогічно модуль HW Config використовується для адресації та параметризації модулів розширення.

Послідовність конфігурування ПЛК Віра серії 100V та 200V.

1. у проекті додати станцію SIMATIC300;
2. зайти в модуль конфігурування апаратури HW Config;
3. вставити 300-у стійку;
4. в стійку вставити CPU315-2DP та створити мережу PROFIBUS;
5. приєднати до мережі стійку для ПЛК Віра 100V та в неї вставити необхідний ЦПУ 100-ї серії;
6. в конфігураційній таблиці розташувати необхідні модулі в тій послідовності, в якій вони розташовані на стенді;
7. зберегти та скомпілювати налаштування;
8. при необхідності створити таблицю символів;
9. закрити модуль HW Config.

Конфігурування ПЛК Віра 200V виглядає так само, як і 100-ї серії лише з однією відмінністю: до мережі PROFIBUS необхідно приєднати стійку для ПЛК Віра 200V та в неї встановити ЦПУ 200-ї серії і відповідні їм модулі.

Послідовність конфігурування ПЛК Віра 300 S.

1. у проекті додати станцію SIMATIC 300;
2. зайти в модуль конфігурування апаратури HW Config;
3. вставити 300-у стійку;
4. додати CPU318-2DP (Simatic300/CPU-300/CPU318-2DP/6ES7 318-2AJ00-0AB0)
5. в конфігураційній таблиці розташувати компоненти в необхідній послідовності (починаючи з слоту номер 4);
7. в кінці списку конфігураційної таблиці вставити модуль комунікаційного процесора CP 343-1 (343-1EX11);
8. задати IP-адресу та маску підмережі;
9. зберегти та скомпілювати налаштування;
10. при потребі створити таблицю символів;
11. закрити модуль HW Config.

Примітка: комунікаційний процесор CP 343-1 використовується як віртуальний, і дозволяє використовувати інтегрований в ЦПУ SPEED7 інтерфейс Ethernet. Модуль CP343-1 завжди має бути в кінці списку конфігураційної таблиці (після всіх існуючих модулів).

Розглянемо більш детально конфігурування ПЛК, на прикладі Віра 200V. Запускаємо програму SIMATIC Manager. Створюємо новий проект та називаємо його "Lab1". Шлях для збереження проекту має мати вигляд:

`"C:\ProgramFiles\SIEMENS\STEP7\s7proj\GRUPA\PRIZVUSCHE\"...`

де **GRUPA**—назва групи; **PRIZVUSCHE**—прізвище студента.З панелі меню вибираємо команду Insert→Station→SIMATIC 300 Station. Цим ми додаємо в проект 300-у станцію. Розкриваємо станцію у лівій частині вікна двічі натискаємо на об'єкт Hardware. Це відкриє модуль конфігурування апаратури HW Config.

В інструменті конфігурування апаратури HW Config (рис. 5) необхідно визначити які модулі і в якій послідовності мають бути розміщені на стійці.

Щоб розпочати розташовувати модулі, спочатку необхідно створити стійку (Rack). Для цього у вікні Hardware Catalog (каталог модулів рис. 5) розкриваємо пункт SIMATIC 300, натискаючи лівою кнопкою миші на квадратик з плюсом ліворуч від об'єкту, далі розкриваємо пункт RACK-300 та “перетягуємо” елемент Rail у вільне поле в лівій верхній частині вікна проекту (операція drag&drop).

Цю саму операцію можна виконати, двічі натиснувши на елементі Rail. Після виконання цієї дії з'явиться стійка з представленням слотів на направляючій рейці.

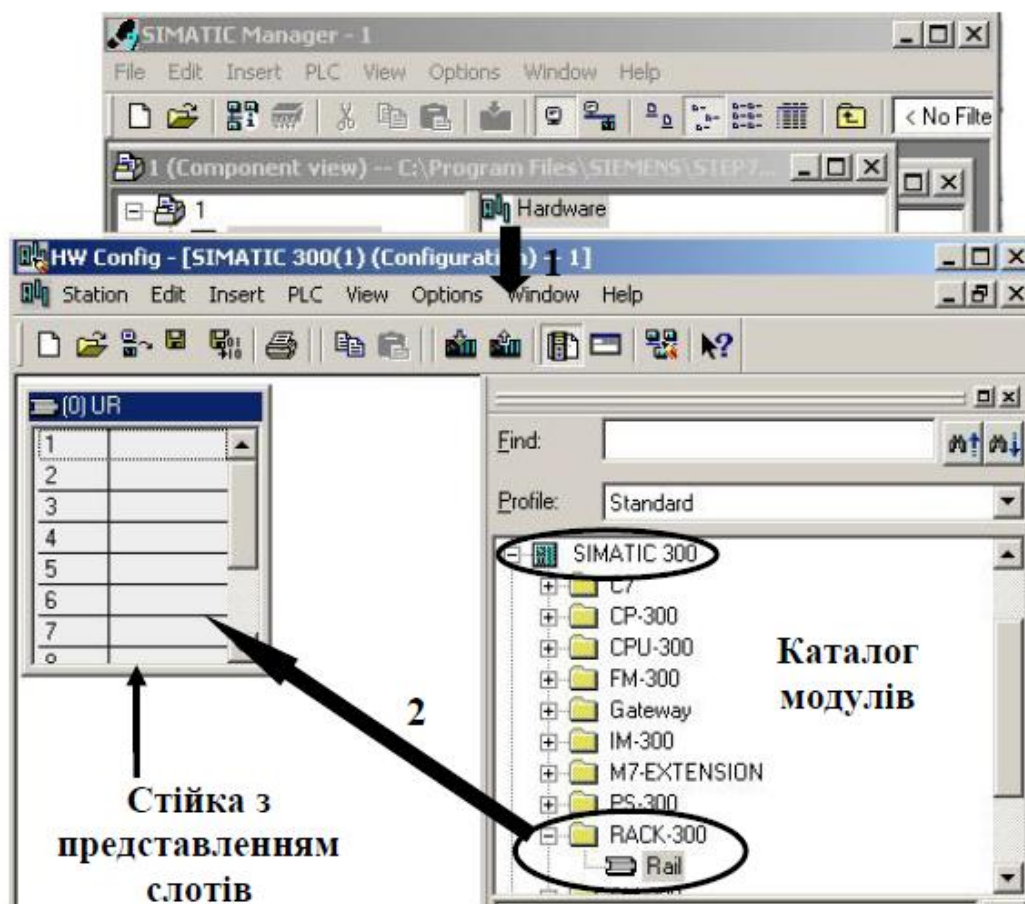


Рис.5. Вікно модуля конфігурування апаратури HW Config

Далі необхідно вибрати та додати CPU 315-2 DP в другий слот стійки. Для цього виділяємо слот номер 2, у вікні каталогу модулів розкриваємо такі пункти: SIMATIC 300, CPU-300, CPU 315-2 DP (рис. 6, крок 1), 6ES7 315-2AF03-0AB0 (рис. 6, крок 2) та двічі натискаємо на пункт V1.2 (рис. 6, крок 3).

Примітка: якщо після кроку 3 з'являється повідомлення “Ви не можете вставити ЦПУ в цей слот” (You cannot put a CPU in this slot) – це означає, що вибрано невірний слот. Закрийте повідомлення та виберіть слот номер 2, і повторіть крок 1-3. Пункти V1.0, V1.1 та V1.2 означають версію прошивки ПЛК. Пункт V1.2 – найновіша (остання) версія прошивки.

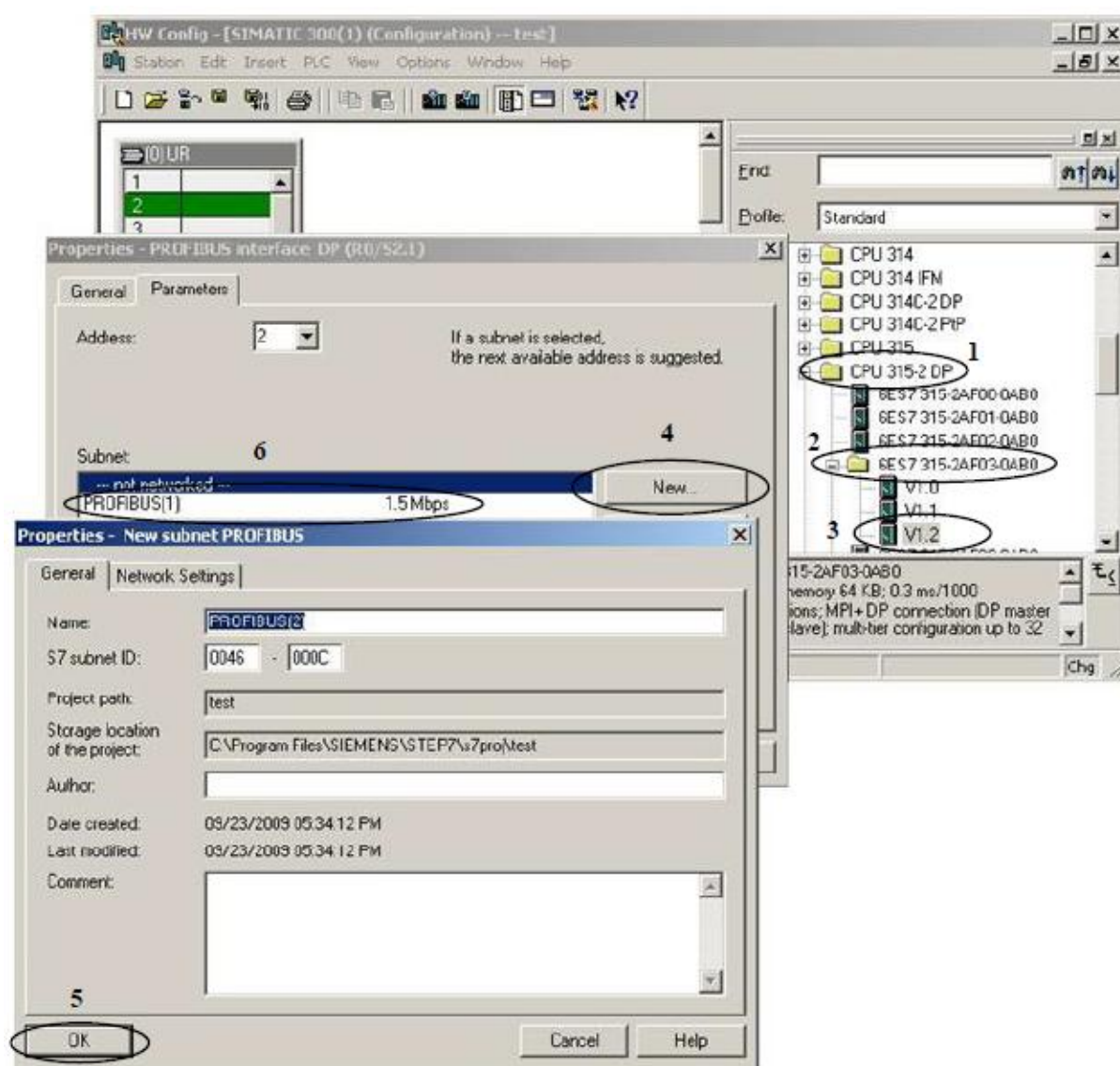


Рис.6. Конфігурування апаратури ПЛК Vira 200V

В результаті виконаної операції з'явиться вікно Properties – PROFIBU Sinterface DP. В ньому натискаємо кнопку **New** (рис. 6, крок 4) та у вікні Properties - New subnet PROFIBUS, нічого не змінюючи, натискаємо кнопку **OK** (рис. 6, крок 5). Таким чином ми створили віртуальну мережу PROFIBUS(1) (рис. 6, крок 6), яка необхідна для підключення ПЛК Vira 200V у Step7. Ще раз натискаємо кнопку **OK**.

Результатом виконаних дій є створена стійка (Rack) з процесором фірми Siemens CPU 315-2DP та мережею PROFIBUS(1)DP master.

Щоб підключити стійку для ПЛК Vira 200V до мережі PROFIBUS, необхідно вибрати модуль Vira_CPU21x з пункту PROFIBUS DP, як це показано на рис. 7, крок 1-4, та перетягнути його до лінії, яка позначає мережу PROFIBUS (рис. 7, крок 5). У вікні, яке з'явиться після цієї операції, пересвідчуємось, що значення поля Address (адреса) є рівним "1". Якщо все вірно – натискаємо кнопку **OK**.

Це створить стійку для ПЛК Vira, в яку вставляються ЦПУ та модулі розширення.

Після цього виділяємо стійку для Vira 200V – в лівій частині вікна HWConfig, знизу відобразиться порожня конфігураційна таблиця, в якій і розміщується апаратура. В слоті №0, конфігураційної таблиці, розташовуємо ЦПУ (тип ЦПУ вказаний в табл. 5). З каталогу модулів по черзі вибираємо модулі та встановлюємо (операція drag&drop або double click) їх у слоти, починаючи з першого.

Примітка: Модулі додаємо, починаючи з ЦПУ (слот 0), модуль дискретних входів (слот 1) і т.д. Кожна стійка має свою конфігураційну таблицю. Наприклад, якщо вибрати стійку **(0)UR**, конфігураційна таблиця буде містити вже встановлений процесор **CPU 315-2DP**. Якщо вибрати стійку **Vira_CPU**, відповідно конфігураційна таблиця міститиме ЦПУ **214-2BP02**. Заголовок активної стійки виділяється синім кольором, подібно до активних вікон в Windows. Заголовок активної конфігураційної таблиці співпадає з заголовком активної стійки.

Після завершення конфігурування ПЛК та модулів розширення конфігураційна таблиця повинна мати вигляд, зображений на рис.7.

При компонуванні апаратури інструмент HW Config автоматично призначає початкові адреси входам-виходам модулів. Кожний слот отримує фіксовану адресу, яка складається з номеру стійки і номеру слота. Таким чином, кожен вхід або вихід модуля розширення отримує свою унікальну адресу. Ця адреса відображається у конфігураційній

таблиці відповідної стійки в колонках I Address (Вхідна адреса) та Q Address (Вихідна адреса).

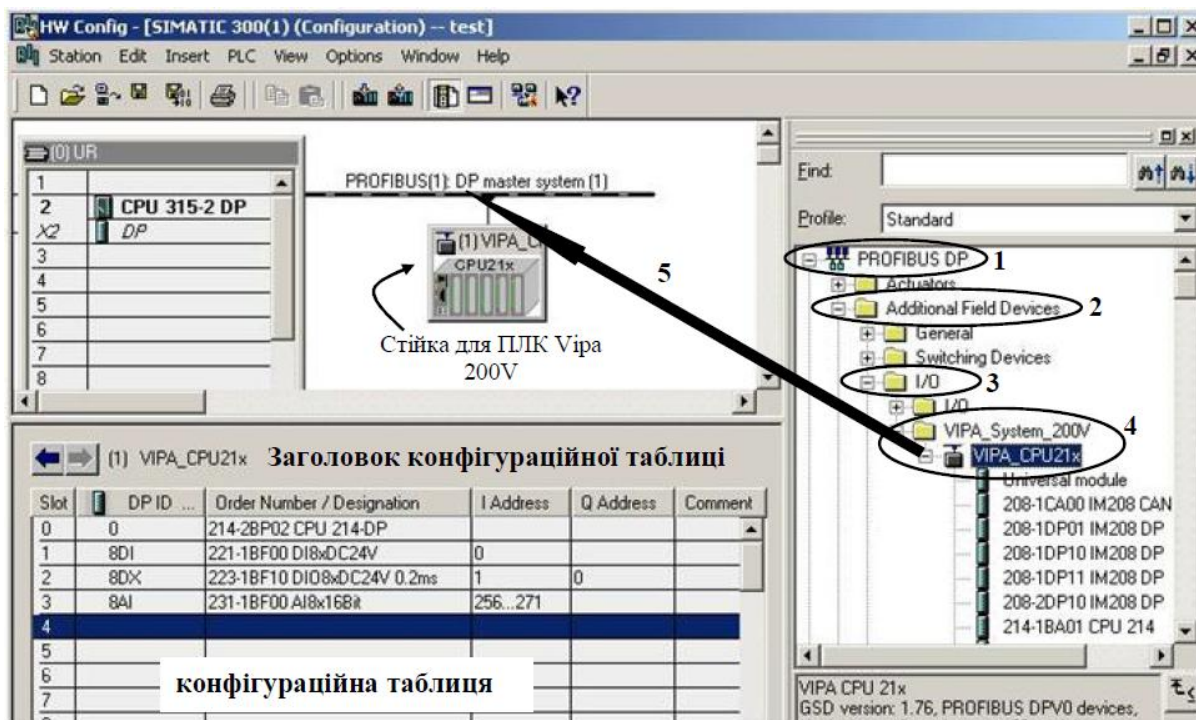


Рис. 7. Компонування модулів ПЛК Віра 200V в HW Config

Примітка: звертання до входів-виходів модулів в програмі S7 відбувається за адресами. Наприклад, звертання до першого входу модуля, розташованого в слоті №1 конфігураційної таблиці ПЛК Віра, має вигляд **I0.0**, а до третього виходу модуля розташованого в слоті №2 - **Q0.2** Така адресація називається абсолютною (absolute addressing). **Q0.2** означає, що програма звертатиметься до виходу (ідентифікатор Q), нульовий байт (0), третій біт (2). Відповідно **I0.0** – вхід (ідентифікатор I), нульовий байт, нульовий біт.

Після компонування і адресації цифрових та аналогових модулів збережіть та скомпілюйте дані станції. Для цього необхідно зайти в меню Station→Save and Compile або натиснути кнопку на панелі

інструментів 

Ця команда не тільки зберігає налаштування, але і компілює конфігураційну таблицю та записує скомпільовані дані в об'єкт System data (Системні дані) контейнеру Blocks (Блоки). Команда Station→Save зберігає дані конфігураційної таблиці з усіма параметрами проекту на жорсткий диск.

Нехай до першого входу дискретного модуля 221-1BF10 підключена “кнопка 1”, а до третього виходу дискретного модуля 223- 1BF00 підключена “червона лампочка”. Абсолютна адреса першого входу дискретного модуля 221-1BF10 у програмі буде I0.0, а третього виходу дискретного модуля 223-1BF00 – Q0.2, згідно конфігураційної таблиці (рис. 7). Якщо входу I0.0 та виходу Q0.2 присвоїти ім’я, наприклад “Button_1” та “Lamp_Red” відповідно, то в програмі можна звертатись до цих входів-виходів за їхнім символьним ім’ям (рис. 8). Звертання у програмі до входу (виходу) модуля за його символьним ім’ям називається символьною адресацією. Щоб присвоїти входам (виходам) модуля символьні імена, необхідно відкрити таблицю символів відповідного модуля.

Присвоїмо символьні імена модулю розташованому у слоті №1 конфігураційної таблиці для ПЛК Vira 200V. Натискаємо правою кнопкою миші на цьому модулі та вибираємо пункт Edit Symbols або з меню Edit→Symbols.

Таблиця символів вибраного модуля показана на рис. 9. У колонці Address (адреса) відображаються абсолютні адреси модуля. В колонці Symbol (символ) можна присвоїти входам символьні імена. У колонці Data type (тип даних) – вказаний тип даних. Колонка Comment (коментарі) використовується для внесення коментарів.

У таблиці символів модуля дискретних входів 221-1BF00 присвойте символи входам так, як це показано на рис. 9.

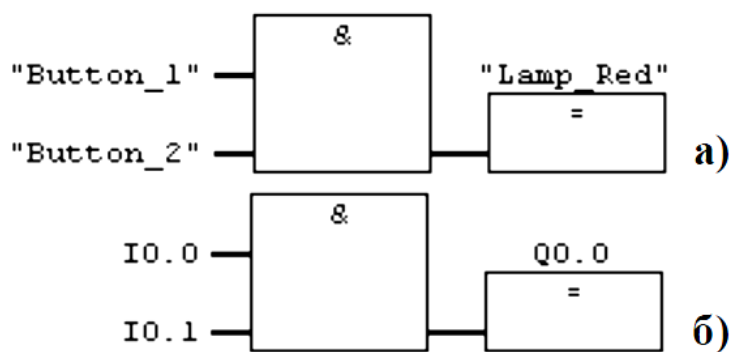



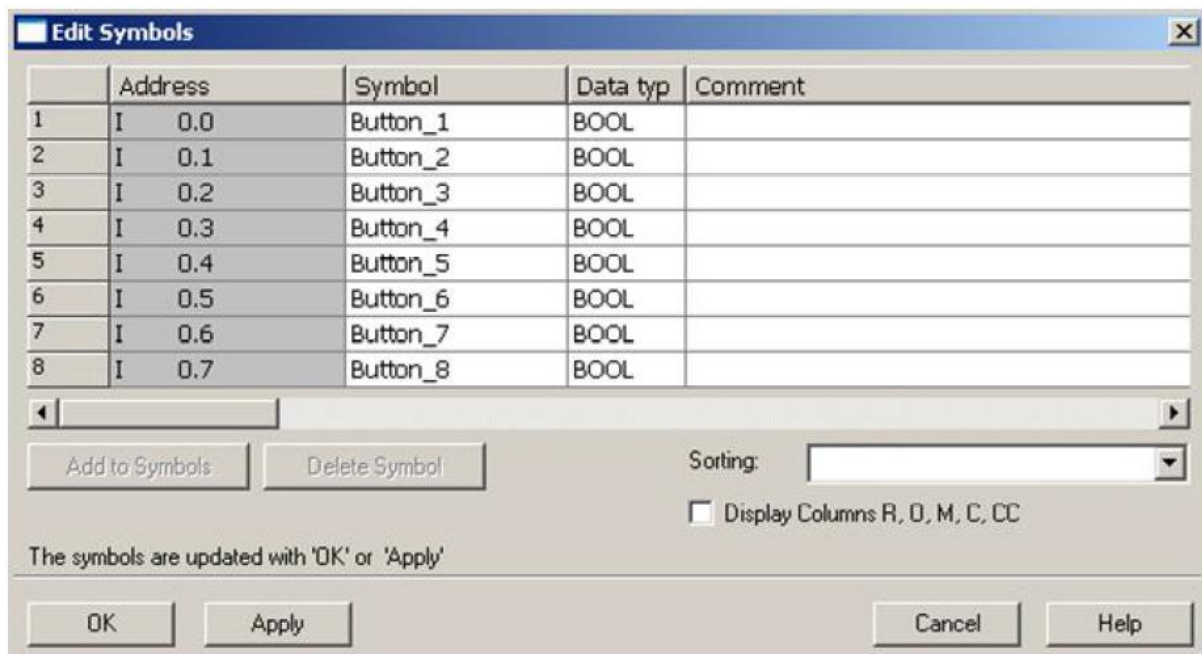


Рис. 8. Приклади адресації входів та виходів: а) символьна адресація в FBD б) абсолютна адресація в FBD

Після завершення редагування таблиці символів натискаємо кнопку ОК. Зберігаємо та компілюємо дані станції, натиснувши кнопку  Закриваємо модуль HW Config

Примітка: для завантаження конфігурації або програми в ПЛК використовують “зелений кабель”, крім випадків, коли ПЛК працює по протоколу *Ethernet*. Для завантаження створеної конфігурації (програми) в ПЛК або симулятор, необхідно натиснути кнопку  або з меню PLC→Download (ПЛК→Завантажити в ПЛК). Для зчитування конфігурації з ПЛК використовується кнопка  або команда з меню PLC→Upload (ПЛК→Завантажити в програматор). Після завершення конфігурування в дереві проекту з’явиться новий об’єкт CPU 315-2 DP (рис. 3), який містить об’єкт S7 Program. В свою чергу S7 Program містить об’єкт Blocks (блоки програми), Sources (вихідні програми) та Symbols (таблиця символів). Блок програм (Blocks) містить відкомпільовані дані (System data) і порожній організаційний блок головної програми OB1.



	Address	Symbol	Data type	Comment
1	I 0.0	Button_1	BOOL	
2	I 0.1	Button_2	BOOL	
3	I 0.2	Button_3	BOOL	
4	I 0.3	Button_4	BOOL	
5	I 0.4	Button_5	BOOL	
6	I 0.5	Button_6	BOOL	
7	I 0.6	Button_7	BOOL	
8	I 0.7	Button_8	BOOL	

Рис. 9. Приклад таблиці символів модуля дискретних входів

3. Організаційний блок головної програми OB1

Програмне забезпечення ПЛК складається з операційної системи і програми користувача. Операційна система – це сукупність всіх інструкцій, які здійснюють керування системними ресурсами. Операційна система є складовою частиною ПЛК до якої у користувача немає доступу в режимі запису, крім випадку поновлення програми. Програма користувача (головна програма) представляє собою сукупність інструкцій для опрацювання сигналів, з допомогою яких здійснюється керування процесом у відповідності до поставленої задачі автоматизації.

Перед початком опрацювання користувацької програми ЦПУ виконує програму запуску. Ця програма запускається, наприклад, при включенні живлення ПЛК. Головна програма розташовується в організаційному блоці OB1, який **циклічно** опрацьовується центральним процесором.

У контейнері Blocks можна створювати блоки користувацької програми (табл. 1). Щоб вставити блок, потрібно вибрати у лівій частині проекту контейнер Blocks та у правій частині вікна навігномуполінатиснутиправукнопкумишітазмению Insert New Object вибирати необхідний блок.

Створіть по одному порожньому блокові FB, FC, DB та VAT.

Таблиця 1. Блоки користувацької програми

Об'єкт	Опис	
OB	Організаційні блоки	Зберігають відкомпільовані код і дані користувацької програми
FB	Функціональні блоки	
FC	Функції	
DB	Блоки даних	
SFC	Системні функції	Зберігають інтерфейси викликів для системних блоків, інтегрованих в ЦПУ
SFB	Системні функціональні блоки	
Системні дані	Блоки системних даних	Зберігають відкомпільовані дані конфігураційної таблиці
UDT	Типи даних	Зберігають оголошення користувацьких типів даних
VAT	Таблиці змінних	Зберігають змінні для спостереження та модифікації

Відкрийте блок OB1. При першому запуску блоку, з'явиться вікно Properties – Organization Block (Властивості організаційного блоку), де в графі Created in Language: (Створений мовою:) можна вибрати мову програмування: STL, LAD або FBD. Вибираємо FBD та натискаємо кнопку ОК.

Вікно блоку OB1 (головна програма) складається з таких частин: каталог елементів – містить елементи (блоки) бітової логіки, компаратори, лічильники, математичні функції, таймери, блоки функцій та інші; таблиця

опису змінних; тіло програми – складається із заголовку, поля коментарів та області програми).

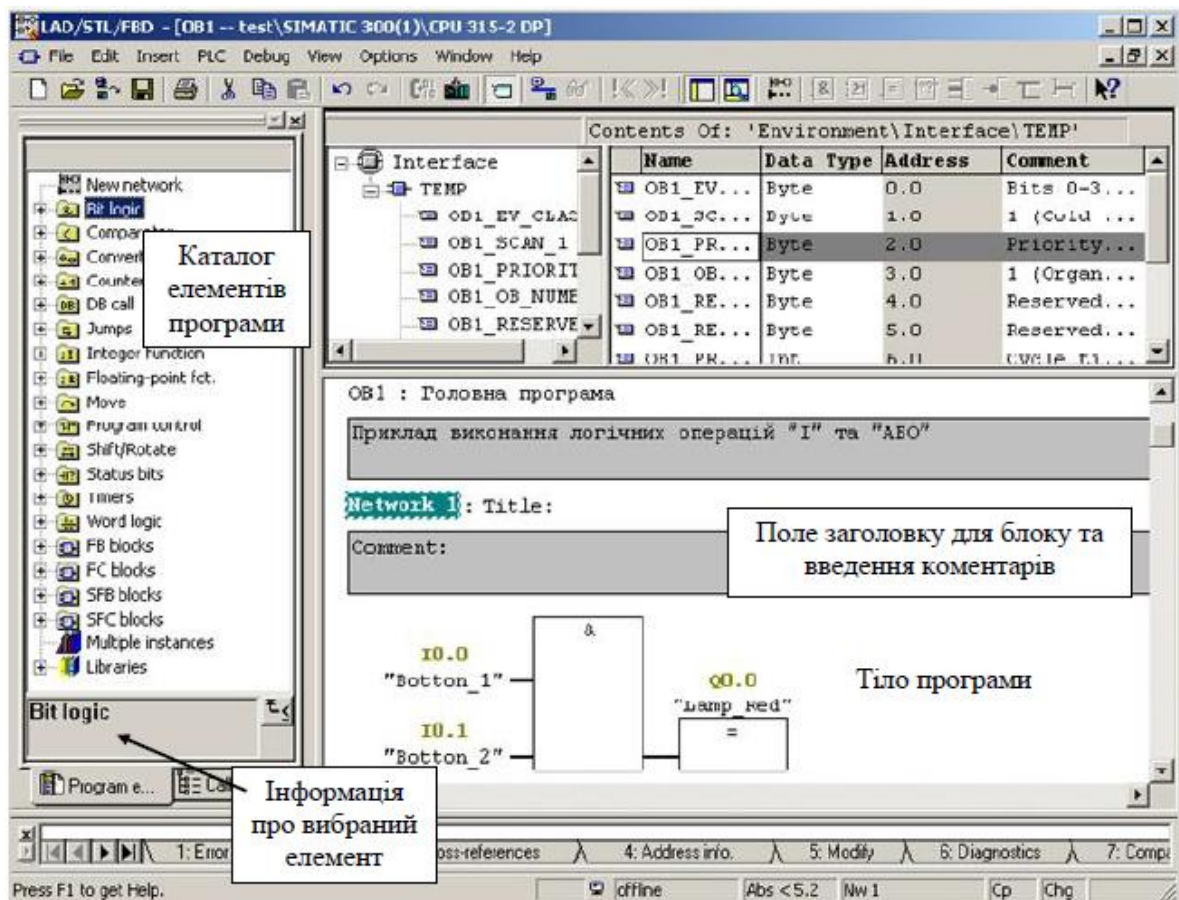


Рис. 10. Вікно користувацької програми OB1

Примітка: в каталозі елементів доступні елементи тільки для мов LAD та FBD. Переключитись між мовами програмування можна з меню **View→LAD**, **View→STL** та **View→FBD**.

Додавання елементів в тіло програми проводиться за допомогою перетягування (операція drag&drop) відповідних елементів з каталогу у сегмент (Network) програми. Програма, створена мовами LAD або FBD складається з логічних елементів, які з'єднані між собою з метою формування логічної операції. Завершується логічна операція, як правило, елементом присвоєння рис. 11.

Виберіть елемент «І» з каталогу елементів та вставте його у тіло програми. На входах елементу розташовані символи (???), це означає, що входам не задано жодних значень. Задайте адресу I0.0 верхньому входу. Другому (нижньому) входу присвойте адресу I0.1. Далі “витягніть” елемент «АБО» та встановіть його на виході елемента «І». Другому входу «АБО» задайте адрес I0.2. Завершіть логічну операцію присвоєнням результату логічної функції операнду Q0.0, “витягнувши” відповідний елемент присвоєння з каталогу елементів.

Примітка: кожна логічна операція має виконуватись у своєму сегменті.

Для того щоб додати новий сегмент, використовуємо команду з меню Insert→Network. Редактор програми автоматично нумерує сегменти, починаючи з першого. Кожен блок може вміщати до 999 сегментів. Для кожного сегменту можна задати назву (заголовок) та вписати коментарі.

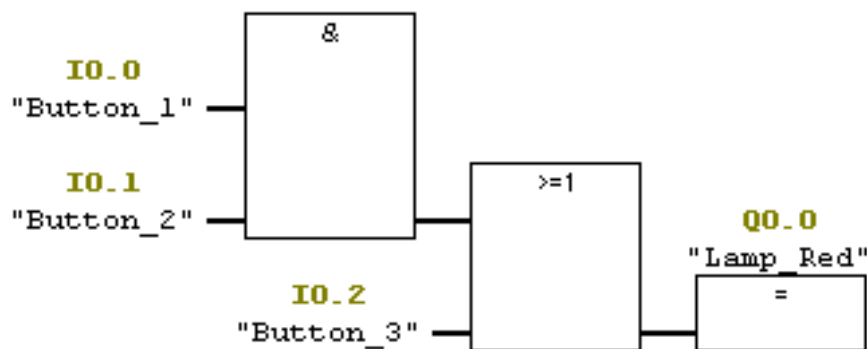


Рис. 11. Приклад представлення операції бінарної логіки в FBD.

Редактор FBD та LAD виконує логічні операції зліва направо, зверху вниз. Зліва, у логічних елементах, знаходяться входи, а справа – виходи.

Розглянемо логічну операцію, реалізовану на рис. 11. На входи логічного елемента “І” заведено операнд I0.0 (абсолютна адреса) або “Button_1” (символьна адреса) та операнд I0.1 (Button_2). На вхід логічного елемента “АБО” передається результат логічної операції (RLO) елемента “І”, а на інший вхід операнд I0.2. Результат логічної операції “АБО” присвоюється операнду Q0.0 (Lamp_Red). Результатом виконання логічної функції (рис. 11) операнд Q0.0 встановлюється в “1” згідно табл. 2.

Таблиця 2. Таблиця істинності програми

I0.0	I0.1	I0.2	Q0.0
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

4. Модуль S7-PLCSIM

Модуль S7-PLCSIM використовується для тестування та відлагодження програми на симуляторі ПЛК. Цей модуль є складовою частиною програмного забезпечення STEP 7 (версії Professional). Для тестування роботи програми з ПЛК немає необхідності приєднувати реальний контролер. S7-PLCSIM дає можливість моніторингу та модифікації різних параметрів та змінних, які використовуються програмою.

Запустіть модуль S7-PLCSIM.





Для цього у вікні проекту натисніть кнопку або з меню виберіть Options→Simulate Modules. Головне вікно S7-PLCSIM (рис. 12) складається із заголовку вікна, панелі меню, панелі інструментів та робочої області. Саме в ній розміщується імітований ПЛК, модулі входів-виходів, таймера та лічильники. Об'єкт ПЛК містить три пункти меню:

- RUN-P – запускає програму ПЛК на виконання. Цей режим використовується як для запуску ПЛК, так і для завантаження нової програми;

- RUN – запускає програму ПЛК на виконання. Цей режим не дозволяє під час виконання програми, завантажувати у ПЛК будь-які частини програми або налаштування;

- STOP – зупиняє виконання головної програми ПЛК. В цьому режимі можна завантажувати у ПЛК нову програму.

Кнопка MRES використовується для очищення пам'яті ПЛК. У лівій частині ПЛК відображається стан, в якому зараз перебуває контролер.

Модулі входів-виходів використовуються для спостереження або зміни станів відповідних входів-виходів. Щоб вставити додаткові вхідні об'єкти (модулі), використовуємо команду з панелі меню Insert→Input Variable (Вставити→Вхідні змінні) або кнопку  на панелі інструментів. З'явиться об'єкт **IB0** (Input Byte 0 – вхідний байт 0). Вихідні об'єкти додаються командою Insert→Output Variable (Вставити→Вихідні змінні) або кнопкою . Так само можливо додати як блоки відображення стану таймера Insert→Timer (кнопка ), так і блоки лічильника Insert→Counter (кнопка ).

Створіть по одному блоку вхідних, вихідних модулів, блок таймера та блок відображення стану лічильника.

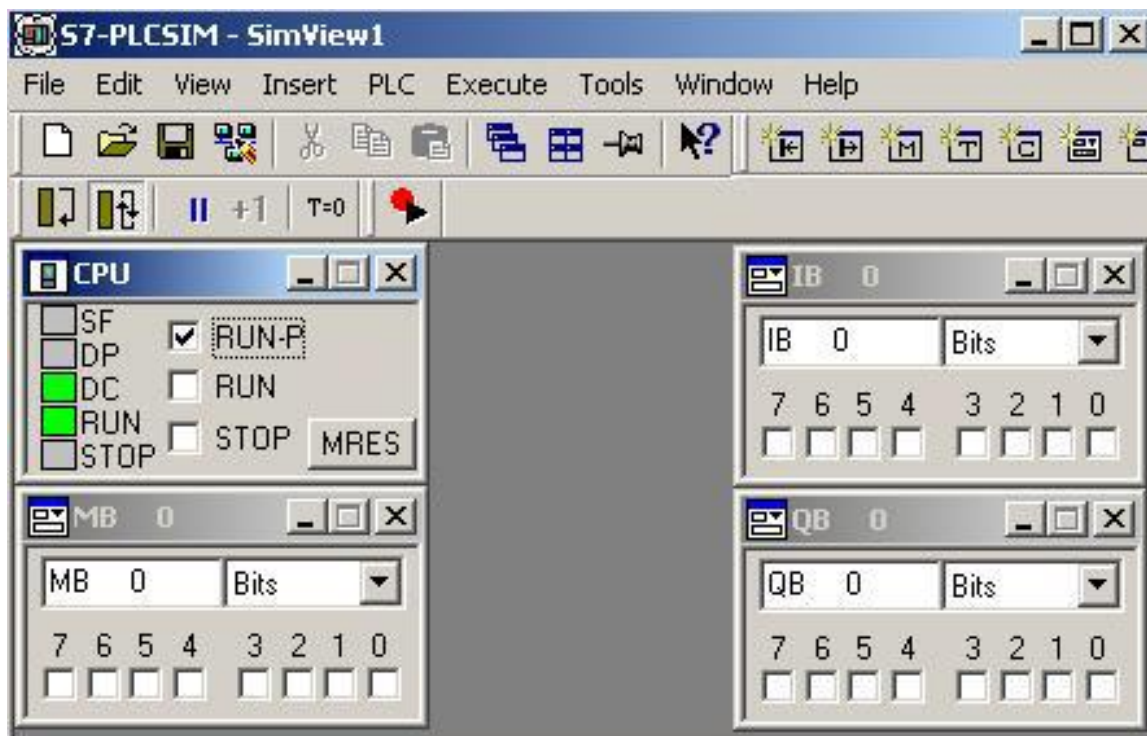



Рис. 12. Вікно модуля S7-PLCSIM.

Для завантаження програми в симулятор у вікні SIMATIC Manager натисніть кнопку  або виберіть команду з меню PLC→Download (ПЛК→Завантажити).

В S7-PLCSIM виберіть меню PLC і переконайтесь, що пункт Power On (включено) та пункт меню Execute→Scan Mode→Continuous Scan (Виконати→Метод сканування→Безперервний) відмічені символом “●”.

Завантажте програму в симулятор та переведіть ПЛК в режим RUN-P.

Для цього натисніть на квадратик біля напису RUN-P в блоці ПЛК (CPU). Це запустить програму в ПЛК на виконання.

Змінюючи відповідні біти блоку входів (об’єкт IB), дослідіть зміну виходів блоку QB.

Наприклад, при встановленні нульового та першого бітів 0 або тільки при встановленні другого біту модуля входів IB, встановлюється нульовий біт у модулі виходів QB 0, згідно табл.

2. 5. Налаштування програматора у Step 7

Кожен процесорний модуль Vira серії 100V, 200V обладнаний інтерфейсом MP2I, який використовується для завантаження та відлагодження проектів в ЦПУ, обміну даними між програматором

(ПК) та ЦПУ (контролером), обміну даними між ЦПУ і операторськими панелями. MP2I об'єднує в собі два інтерфейси MPI та RS232. Підтримка інтерфейсу RS232 реалізована на незадіяних контактах роз'єму MPI, дозволяє встановити з'єднання “точка–точка” з комп'ютером, без використання спеціального MPI-адаптера, лише з допомогою так званого “зеленого кабелю”. Для налаштування MPI-адаптера в Step7, у вікні проекту виберіть команду Options→Set PG/PC Interface (Опції → Налаштування програматора).

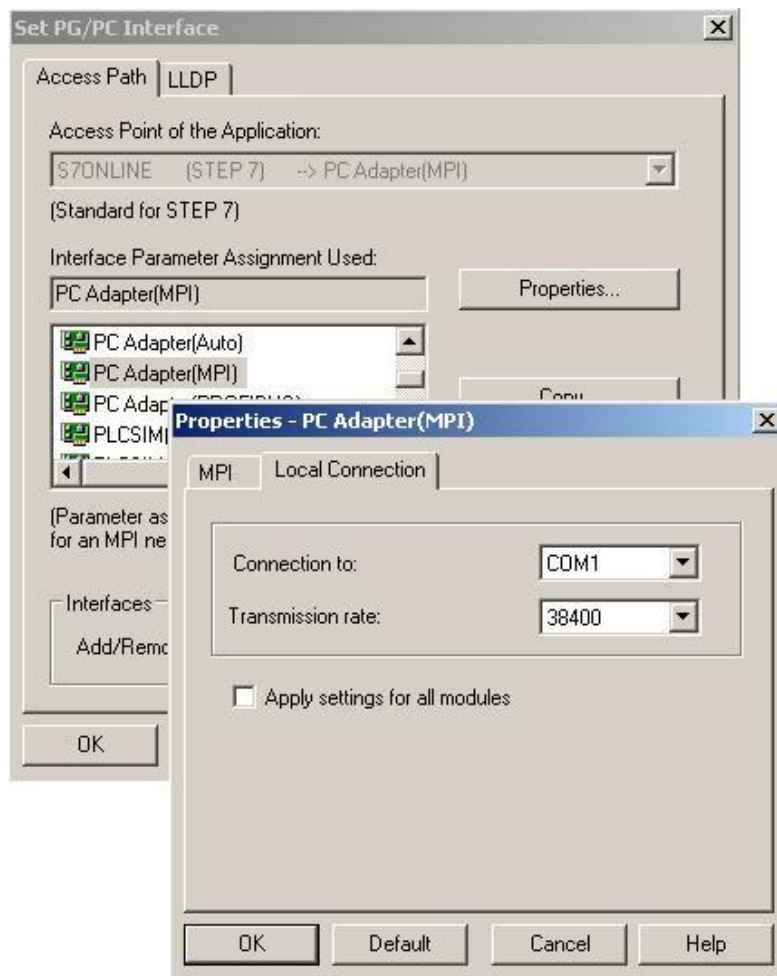


Рис. 13. Вікно налаштування програматора для MPI з'єднання

У вікні SetPG/PCInterface виберіть інтерфейс PC Adapter (MPI) та зайдіть в налаштування інтерфейсу MPI-адаптера (рис. 13), натиснувши кнопку Properties. Виберіть закладку Local Connection (Локальні з'єднання) і встановіть необхідний COM-порт та швидкість передачі даних рівною 38400. Завершивши конфігурування, підтвердіть зміни, натиснувши кнопку OK.

Примітка: для встановлення передачі даних через інтерфейс Ethernet, у вікні налаштування програматора виберіть відповідний драйвер мережі (протокол TCP/IP).

6. Зіставлення програми FBD та синтезованої цифрової схеми

Програма, написана мовою FBD, ззовні є схожою до цифрових схем, синтезованих на логічних елементах. Ця схожість проявляється у тому, що «логіка» у них практично однакова. Однак, основна відмінність полягає у тому, що FBD це, перш за все, програма, яка виконується послідовно, команда за командою, у той час як цифрова схема виконується паралельно у часі зліва (від входів) направо (до виходів) по всій «ширині» схеми.

Навики синтезу цифрових схем, після внесення певних корекцій, можуть бути успішно використані при написанні програм FBD. Для цього розглянемо алгоритм послідовності виконання програми FBD.

Програму FBD, розміщену в OB1, можемо умовно порівняти з функцією алгоритмічної мови C++, виклик якої циклічно повторюється. Вона, як і функція C++, на самому початку приймає значення вхідних змінних (цифрових та аналогових входів, напр. I0.2), а вкінці повертає значення вихідних змінних (цифрових та аналогових виходів, напр. Q1.3). Усередині OB1 можуть викликатися інші функції, виконуватися різні булеві та арифметичні операції над змінними, багато-кратно змінюватися значення вихідних величин. Важливий момент полягає у тому, що передача нових значень у фізичні виходи виконується лише після завершення повного чергового циклу програми OB1, а поточні зміни значень для аналогових та цифрових виходів лише фіксуються у пам'яті. Будь-яка зміна значень входів набирає чинності лише з початку наступного циклу.

Програма FBD виконує послідовно донизу усі понумеровані сегменти (Network1, Network2 і т.д.). В межах одного сегменту, як вже зазначалося, виконання операцій здійснюється так: зліва направо, зверху вниз. Тобто першими виконуються верхні лінійки логічних операцій, а їхні результати можуть бути використані у нижніх лініях коду програми. Важливо звернути вашу увагу на моменти сходження лінійок коду та деревовидного його розгалуження. На рис. 14 ці моменти є детально проілюстровані, а у табл. 3 наведені результати значень вузлів програми для декількох послідовно виконаних циклів. Порядок виконання логічних операцій вказаний у правих нижніх кутах цих блоків.

OB1 : "Main Program Sweep (Cycle)"

Network 1: Ілюстрація ходу виконання програми

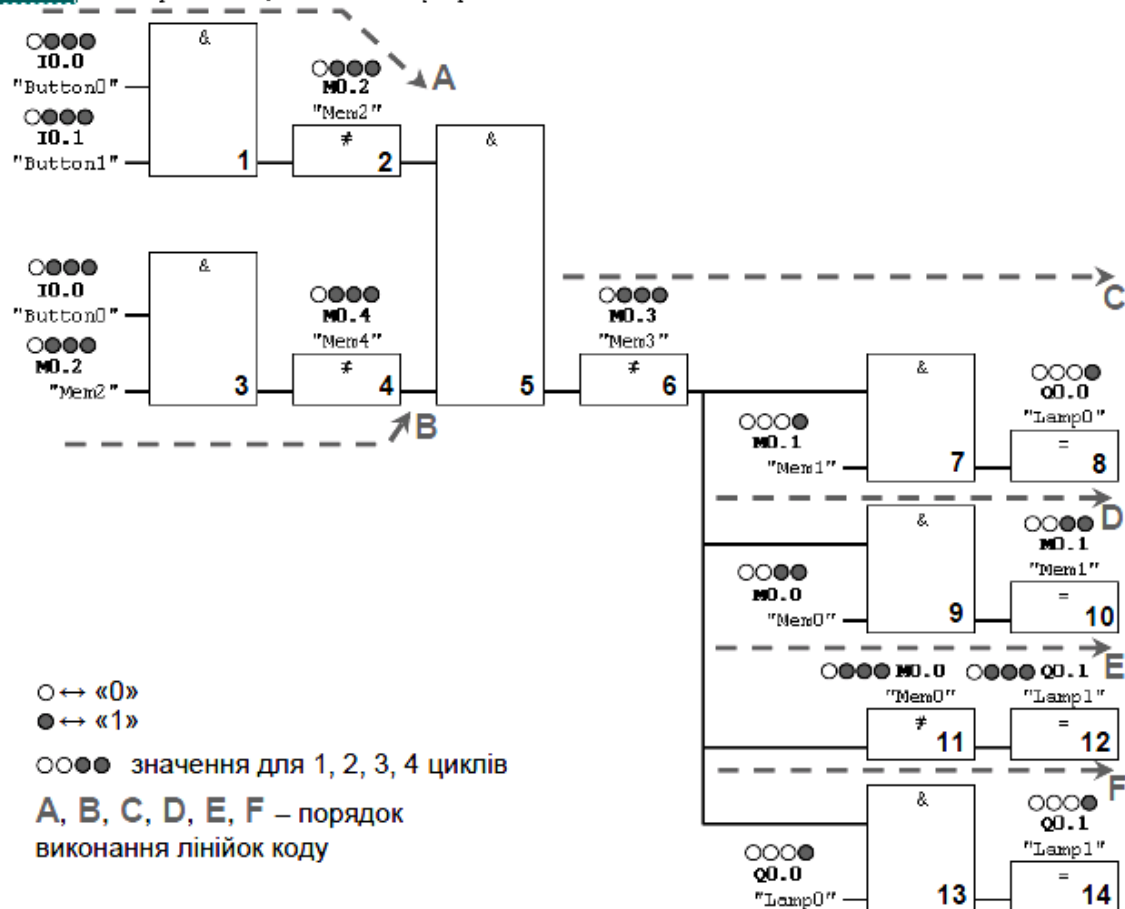


Рис. 14. Ілюстрація послідовності виконання програми FBD

Таблиця 3. Значення результатів вищенаведеної програми

Символ. адреса	Абсол. адреса	Значення			
		1 цикл	2 цикл	3 цикл	4 цикл
Button0	I0.0	○ «0»	● «1»	● «1»	● «1»
Button1	I0.1	○ «0»	● «1»	● «1»	● «1»
Mem0	M0.0	○ «0»	● «1»	● «1»	● «1»
Mem1	M0.1	○ «0»	○ «0»	● «1»	● «1»
Mem2	M0.2	○ «0»	● «1»	● «1»	● «1»
Mem3	M0.3	○ «0»	● «1»	● «1»	● «1»
Lamp0	Q0.0	○ «0»	○ «0»	○ «0»	● «1»
Lamp1	Q0.1	○ «0»	○ «0»	○ «0»	● «1»

На початку виконання першого циклу програми (рис. 14) значення входів Button0 (I0.0) та Button1 (I0.1), а також значення виходів Lamp0 (Q0.0), Lamp1 (Q0.1) та комірок пам'яті Mem0 (M0.0), Mem1 (M0.1), Mem2

(M0.2), Mem3 (M0.3) і Mem4 (M0.4) є рівними «0». Після завершення 1-го циклу значення не змінюються.

Перед початком другого циклу встановлюємо значення входів “Button0” (I0.0) та “Button1” (I0.1) рівними «1». У програмі 5-й елемент “I” є однорівневим елементом ланки програмного коду, і для виконання наступних після нього операцій необхідно, щоб усі програмні рівні, що заходять у нього, були виконані (це лінійки **A** та **B**). На першому рівні (**A**) значення RLO (Result of logic operation, результат логічної операції) елемента 1 заноситься у комірку пам’яті Mem2. При обробленні другого входу однорівневого елемента №5, використовується вже збережене значення Mem2, як вхідна величина операції 3-го елемента “I”. Значення RLO 5-го елемента заноситься у пам’ять Mem3. Далі виконується лінійка коду **C**. Значення виходу Lamp0 дорівнюватиме «0». Після виконання другого рівня коду **D**, значення Mem1 теж «0», оскільки значення Mem0 = «0», а його нове значення «1» присвоюється лише в наступній лінійці коду **E**. Важливим моментом є те, що вихід Lamp1 в лінійці **E** встановлюється в «1», а в наступній лінійці **F** знову приймає «0». При активізації виходів в кінці циклу Lamp1 навіть не “смикнеться”.

На третьому циклі виконання лінійок коду **A**, **B**, **C** буде аналогічне попередньому циклу. У лінійці **D** у комірку пам’яті Mem1 встановиться вже «1», оскільки вхідне значення Mem0 для 9-го елемента “I” на попередньому циклі прийняло «1». Виконання лінійок коду **E** та **F** теж незмінне.

На четвертому циклі вихід Lamp0 (лінійка **C**) прийме значення «1», оскільки вхідне значення для 7-го елемента “I” Mem1 змінилося в попередньому циклі з «0» на «1». Змінене значення Lamp0 вплине на зміну значення з «0» на «1» Lamp1. В наступних циклах значення елементів вже не будуть змінюватися.

Вищеподаний аналіз виконання програми FBD дає нам можливість зробити висновок, що цифрова схема, реалізована на простій логічній функції (тобто без елементів пам’яті та зворотних зв’язків), буде ідентична до її реалізації мовою FBD.

Таблиця істинності. Для формалізації завдання, поставленого перед цифровою схемою, необхідно проаналізувати значення функції (виходу) для кожної комбінації значень аргументів (входів). Отримані значення представляються у вигляді таблиці істинності.

Таблиця 4. Таблиця істинності для натиснення двох або трьох кнопок

№	Входи					Мінтерм	Вихід Q 0.0
	I0.0 A	I0.1 B	I0.2 C	I0.3 D	I0.4 E		F
00	0	0	0	0	0	\overline{ABCDE}	0
01	0	0	0	0	1	$\overline{ABCD}E$	0
02	0	0	0	1	0	$\overline{ABCD}\overline{E}$	0
03	0	0	0	1	1	$\overline{AB}CDE$	1
04	0	0	1	0	0	$\overline{ABC}D\overline{E}$	0
05	0	0	1	0	1	$\overline{ABC}DE$	1
06	0	0	1	1	0	$\overline{AB}CD\overline{E}$	1
07	0	0	1	1	1	$\overline{AB}CDE$	1
08	0	1	0	0	0	$\overline{AB}C\overline{D}\overline{E}$	0
09	0	1	0	0	1	$\overline{AB}C\overline{D}E$	1
10	0	1	0	1	0	$\overline{AB}CD\overline{E}$	1
11	0	1	0	1	1	$\overline{AB}CDE$	1
12	0	1	1	0	0	$\overline{AB}CD\overline{E}$	1
13	0	1	1	0	1	$\overline{AB}CDE$	1
14	0	1	1	1	0	$\overline{A}BCDE$	1
15	0	1	1	1	1	$\overline{A}BCDE$	0
16	1	0	0	0	0	$A\overline{B}\overline{C}\overline{D}\overline{E}$	0
17	1	0	0	0	1	$A\overline{B}\overline{C}\overline{D}E$	1
18	1	0	0	1	0	$A\overline{B}\overline{C}D\overline{E}$	1
19	1	0	0	1	1	$A\overline{B}\overline{C}DE$	1
20	1	0	1	0	0	$A\overline{B}CD\overline{E}$	1
21	1	0	1	0	1	$A\overline{B}CD\overline{E}$	1
22	1	0	1	1	0	$A\overline{B}CDE$	1
23	1	0	1	1	1	$A\overline{B}CDE$	0
24	1	1	0	0	0	$A\overline{B}C\overline{D}\overline{E}$	1
25	1	1	0	0	1	$A\overline{B}C\overline{D}E$	1
26	1	1	0	1	0	$A\overline{B}CD\overline{E}$	1
27	1	1	0	1	1	$A\overline{B}CDE$	0
28	1	1	1	0	0	$A\overline{B}C\overline{D}\overline{E}$	1
29	1	1	1	0	1	$A\overline{B}C\overline{D}E$	0
30	1	1	1	1	0	$A\overline{B}CDE$	0
31	1	1	1	1	1	$ABCDE$	0

$$F(A, B, C, D, E) = \overline{A} \overline{B} C D E + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C \overline{D} E + \overline{A} \overline{B} C \overline{D} \overline{E} + \overline{A} \overline{B} C D E + \\ + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C D E + \overline{A} \overline{B} C \overline{D} E + \overline{A} \overline{B} C \overline{D} \overline{E} + \overline{A} \overline{B} C D E + \\ + \overline{A} \overline{B} C D E + \overline{A} \overline{B} C D E + \overline{A} \overline{B} C D E + \overline{A} \overline{B} C D E + \overline{A} \overline{B} C D E + \\ + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C D \overline{E} + \overline{A} \overline{B} C D \overline{E}.$$

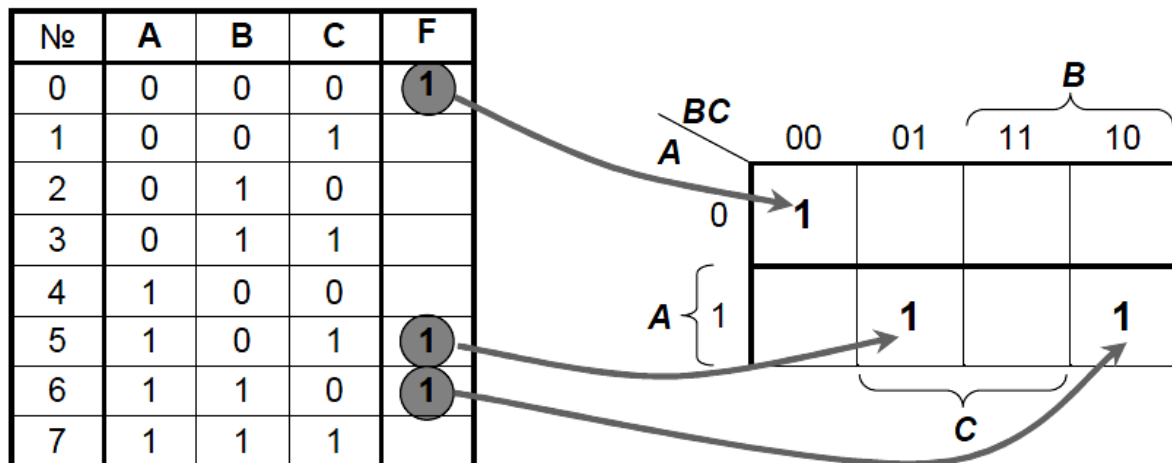
Спрощення булевих виразів. Карти Карно. Для процедури спрощення користуються аналітичними (алгебраїчне спрощення), графічними (карти Карно) та програмними (алгоритм на основі методу Куайна-МакКласкі) методами. Дуже популярними для задач мінімізації, через свою наочність, є карти Карно. Однак на практиці вони мають застосування лише для виразів до 5 вхідних величин.

Процес мінімізації можна описати за допомогою такого алгоритму:
(для отримання мінімальної ДНФ – диз'юнктивної нормальної форми)

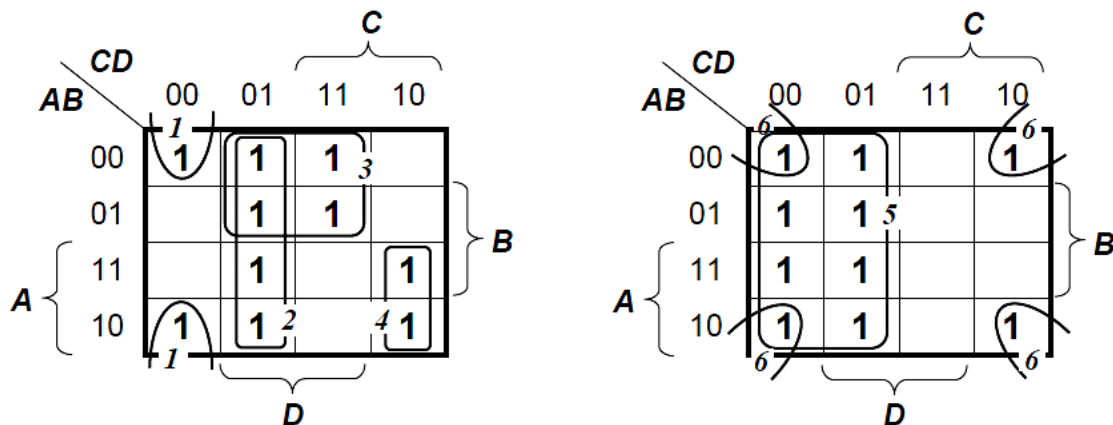
- 23

4. не суміжні групи, розташовані симетрично до осі(ей), можуть об'єднуватися в одну (для карт з 5 та більше змінними);
5. групи можуть пересікатися;
6. у відповідності до виділених груп на карті, визначаються нові члени логічної функції.

Покажемо на прикладі карту Карно з 3-ма змінними та процес її заповнення. Перелік вхідних змінних розбиваємо на дві частини, та розмішуємо на карті – першу частину по вертикалі, другу по горизонталі – у нашому випадку A та BC відповідно. Значення, відповідно до змінних, теж розбиваються на дві частини та наносяться на краї карти. Перша “1” з таблиці істинності відповідає $ABC=000$. Розбиваємо це значення на дві частини: $A=0$ та $BC=00$, та вносимо нашу “1” у клітинку карти за цими адресами. Друга “1” відповідає $ABC=101$. Знову розбиваємо це значення на 2 частини: $A=1$ та $BC=01$, та вносимо у клітинку карти згідно цих адрес і т.д.



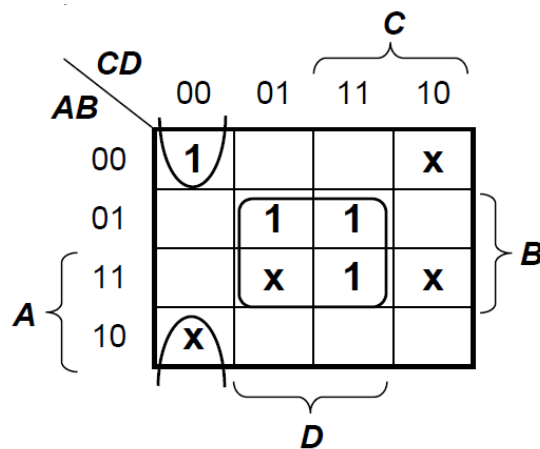
Наступний етап полягає в оптимальному об'єднанні “1” в максимально великі групи. На нижче розташованому рисунку показані основні можливості щодо об'єднань: групування клітинок по краях карти, перехресні групування.



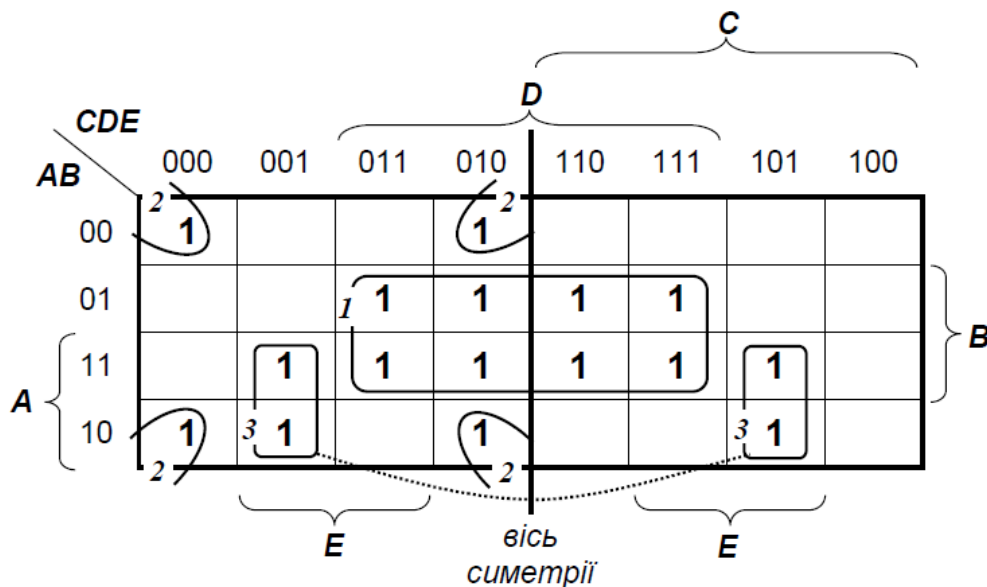
гр1: $\overline{B}\overline{C}\overline{D}$; гр2: $\overline{C}\overline{D}$; гр3: $\overline{A}\overline{D}$; гр4: $AC\overline{D}$ гр5: \overline{C} ; гр6: $\overline{B}\overline{D}$

Після об'єднання усіх "1" в групи, визначають їхні мінтерми. Якщо об'єднується 2 одинички, то з мінтерму випадає одна зі змінних, якщо 4 одинички – випадає 2 змінних, 8 одиничок – 3 змінних і т.д. Вилученню підлягають ті змінні, у які група входить частково, тобто розміщується і в прямій, і в інверсній її частині. Решта змінні записуються в залежності від того, куди група входить повністю – в прямі їхні області чи інверсні.

Варто звернути увагу на так званий випадок не повністю визначеної таблиці істинності, для якої деякі комбінації вхідних значень ніколи не трапляються. У цьому випадку в таблицю істинності для значень функцій вписують Х. Вважається, що такі записи можуть приймати довільні значення. При мінімізації функції від таких значень можна отримати певні вигоди: в залежності, що вигідніше, знак Х можна розглядати як "1", або як "0".



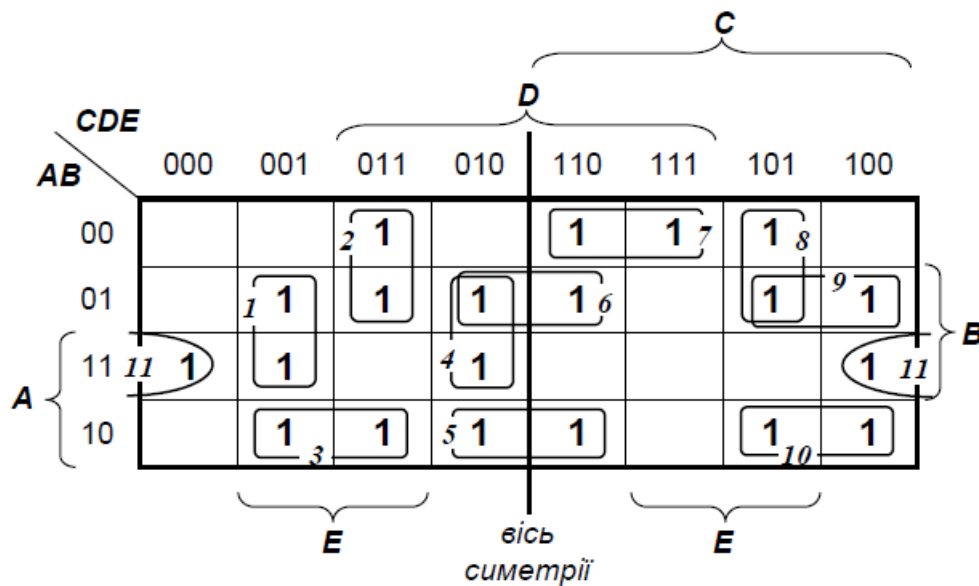
Розглянемо ще випадок карти Карно для 5-ти змінних.



$$\text{гр1: } BD; \text{ гр2: } \overline{BCE}; \text{ гр3: } \overline{ADE}$$

Як зазначалося в алгоритмі мінімізації, оптимальною картою для мінімізації є область 4x4 клітинки (для 4-х змінних). Для 5 та більше змінних проводять осі симетрії між такими областями. На прикладі ми бачимо як групуються одинички, що розташовані в різних областях. Вони мають бути розташовані дзеркально відносно осі симетрії, і не обов'язково мають бути в сусідніх клітинках.

Тепер перейдемо до мінімізації нашого прикладу. З таблиці істинності (табл. 4) перенесемо одинички на карту Карно та утворимо оптимальні групи.



гр1: $\overline{B}\overline{C}\overline{D}E$; гр2: $\overline{A}\overline{C}\overline{D}E$; гр3: $\overline{A}\overline{B}\overline{C}E$; гр4: $\overline{B}\overline{C}\overline{D}\overline{E}$;
 гр5: $\overline{A}\overline{B}\overline{D}\overline{E}$; гр6: $\overline{A}\overline{B}D\overline{E}$; гр7: $\overline{A}\overline{B}CD$; гр8: $\overline{A}\overline{C}\overline{D}\overline{E}$;
 гр9: $\overline{A}BC\overline{D}$; гр10: $\overline{A}BCD$; гр11: $AB\overline{D}\overline{E}$

Рис. 15. Мінімізація поставленої задачі методом карти Карно

У відповідності до виділених груп на карті Карно, визначимо нові члени нашої логічної функції:

$$F(A, B, C, D, E) = \overline{B}\overline{C}\overline{D}E + \overline{A}\overline{C}\overline{D}E + \overline{A}\overline{B}\overline{C}E + \overline{B}\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}\overline{D}\overline{E} + \overline{A}\overline{B}D\overline{E} + \overline{A}\overline{B}CD + \overline{A}\overline{C}\overline{D}\overline{E} + \overline{A}BC\overline{D} + \overline{A}BCD + AB\overline{D}\overline{E}.$$

Тепер, маючи мінімізований вираз функції для програмованого виходу, можемо переходити до перенесення його на програму FBD.

Obj : "Main Program Sweep (Cycle)"

Network 1: Викліл=1, якщо натиснені будь-які 2 чи 3 кнопки

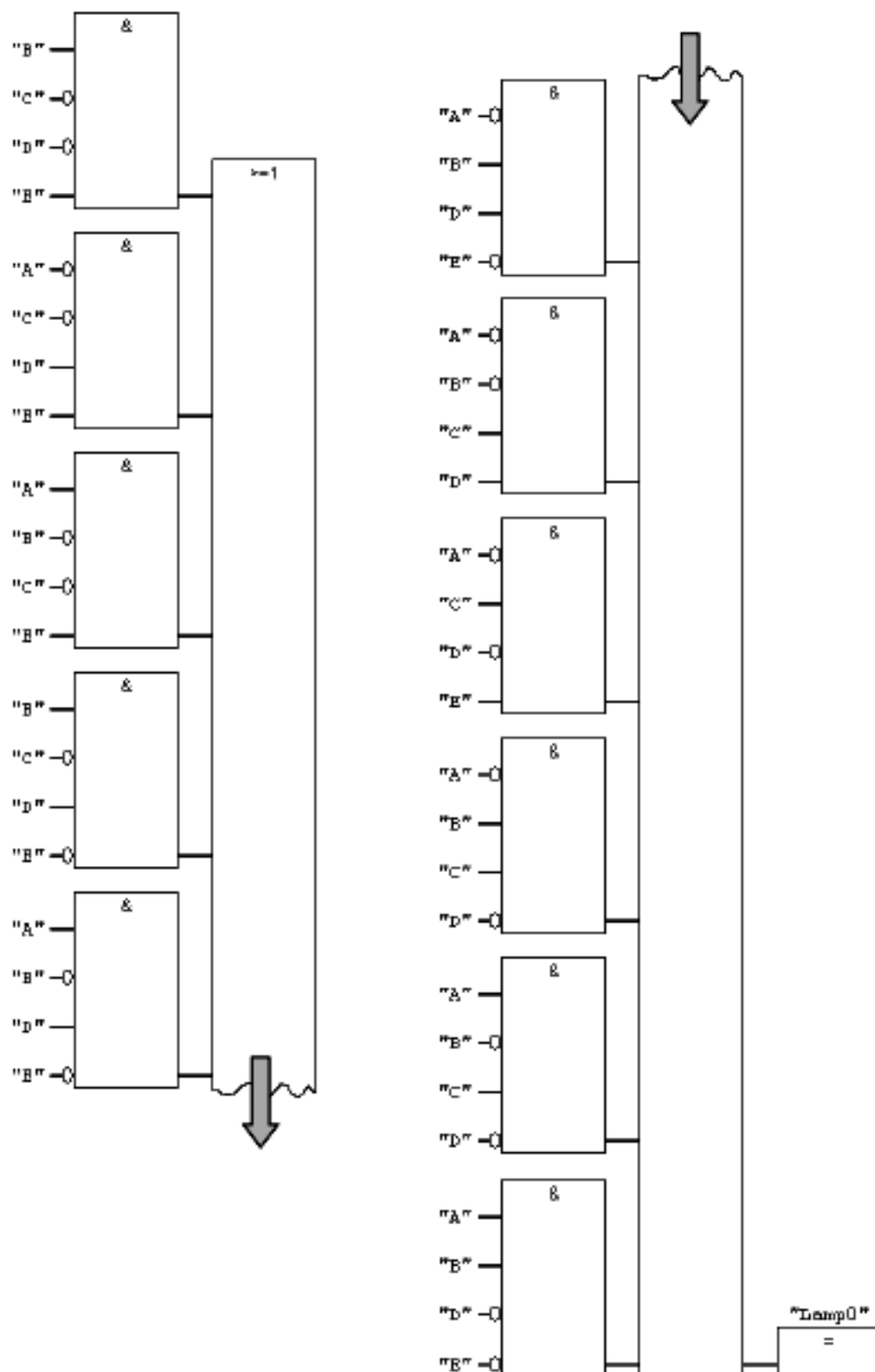


Рис. 16. Програмна реалізація прикладу мовою FBD

7. Конфігурація навчального стенду

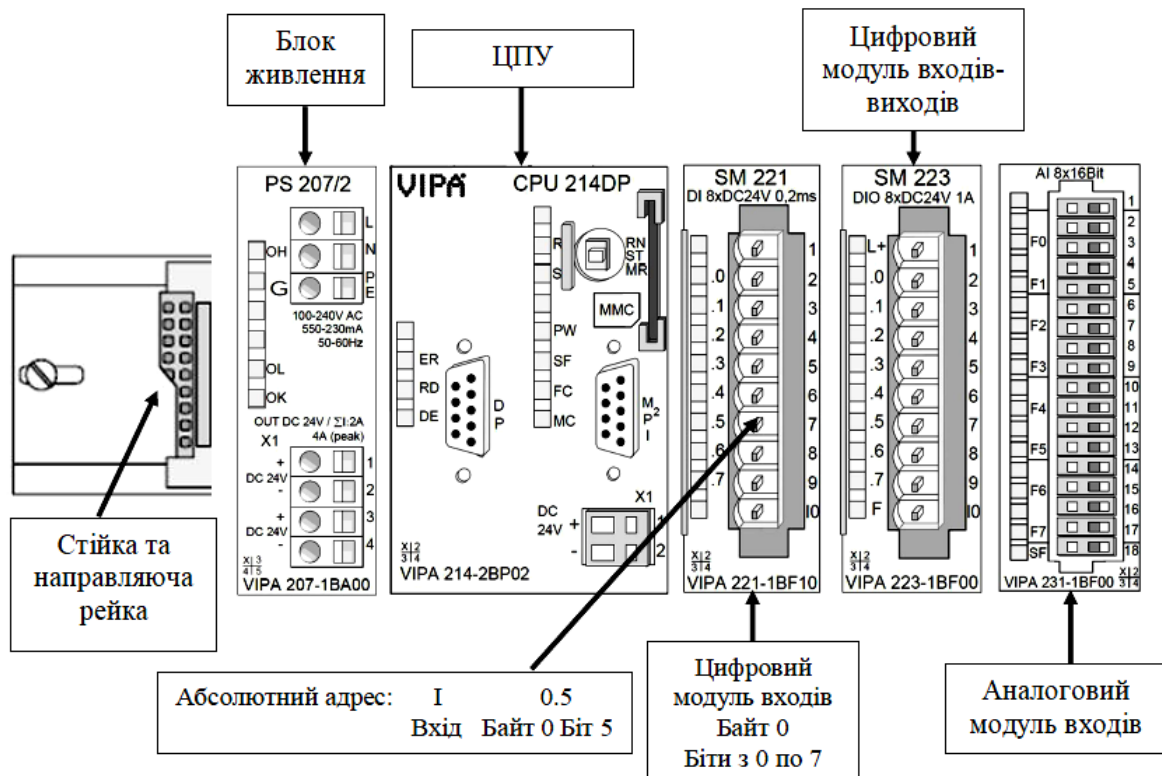


Рис. 17. Конфігурація навчального стенду на базі ПЛК Vipa 200V

Таблиця 5. Конфігурація навчального стенду на базі ПЛК Vipa 200V

№	Назва модуля	Vipa Модель
1	ЦПУ	214-2BP02
2	Модуль дискретних входів	221-1BF10
3	Модуль дискретних входів/виходів	223-1BF00
4	Модуль аналогових входів	231-1BF00

Таблиця 6. Конфігурація навчального стенду на базі ПЛК Vipa 300S

№	Назва модуля	Vipa Модель
1	ЦПУ	313-6FC03
2	Модуль аналогових входів	332-5HB01
3	Модуль аналогових виходів	331-7KB01

8. Порядок виконання роботи

1. Вдома детально вивчити поданий у інструкції довідковий матеріал до лабораторної роботи.
2. У навчальній лабораторії, в присутності викладача, у програмі Step7 сконфігурувати ПЛК Vira серії 200V згідно табл. 5 та написати програму мовою FBD, як це показано на рис. 11. Дослідити в симуляторі роботу програми.
3. Згідно варіанту (порядкового номера в журналі викладача) завдання (табл. 7), вдома скласти таблицю істинності поставленої задачі та мінімізувати її з допомогою карт Карно, а в лабораторії спроектувати та налагодити програму. Входам модуля дискретних входів присвоїти наступні символи: I0.0 – A, I0.1 – B, I0.2 – C, I0.3 – D, I0.4 – E.
4. Роботу програми у симуляторі представити викладачу.
5. За результатами виконаної роботи оформити звіт та захистити його.

Таблиця 7. Завдання до лабораторної роботи

№ п/п	Завдання
1	Вихід Q0.0 встановлюється в «1» при натисканні будь-яких двох (з п'яти) кнопок, при умові, що жодна з двох натиснутих кнопок не є першою (I0.0), а також при натисканні комбінації $\overline{A}\overline{B}\overline{C}DE$.
2	Якщо натиснута лише перша (I0.0) кнопка, вихід Q0.0 встановлюється в «1». При натисканні будь-яких інших трьох кнопок (крім першої I0.0) або комбінації $\overline{A}BCDE$ вихід Q0.4 встановлюється в «1».
3	Вихід Q0.1 встановлюється в «1» при натисканні будь-яких двох (з п'яти) кнопок, при умові, що жодна з двох натиснутих кнопок не є другою (I0.1), а також при натисканні комбінації $\overline{A}\overline{B}\overline{C}DE$.
4	Якщо натиснута лише друга (I0.1) кнопка, вихід Q0.1 встановлюється в «1». При натисканні будь-яких інших трьох кнопок (крім другої I0.1) або комбінації $\overline{A}BCDE$ вихід Q0.3 встановлюється в «1».
5	Вихід Q0.2 встановлюється в «1» при натисканні будь-яких двох (з п'яти) кнопок, при умові, що жодна з двох натиснутих кнопок не є третьою (I0.2)), а також при натисканні комбінації $AB\overline{C}DE$.
6	Якщо натиснута лише третя (I0.2) кнопка, вихід Q0.2 встановлюється в «1». При натисканні будь-яких інших трьох кнопок (крім третьої I0.2) або комбінації $AB\overline{C}DE$ вихід Q0.1 встановлюється в «1».

№ п/п	Завдання
7	Вихід Q0.3 встановлюється в «1» при натисканні будь-яких двох (з п'яти) кнопок, при умові, що жодна з двох натиснутих кнопок не є четвертою (I0.3), а також при натисканні комбінації $\overline{A}BCDE$.
8	Якщо натиснута лише четверта (I0.3) кнопка, вихід Q0.3 встановлюється в «1». При натисканні будь-яких інших трьох кнопок (крім четвертої I0.3) або комбінації $ABC\overline{D}E$ вихід Q0.2 встановлюється в «1».
9	Вихід Q0.4 встановлюється в «1» при натисканні будь-яких двох (з п'яти) кнопок, при умові, що жодна з двох натиснутих кнопок не є п'ятою (I0.4), а також при натисканні комбінації $ABC\overline{D}E$.
10	Якщо натиснута лише п'ята (I0.4) кнопка, вихід Q0.4 встановлюється в «1». При натисканні будь-яких інших трьох кнопок (крім п'ятої I0.4) або комбінації $ABCDE$ вихід Q0.0 встановлюється в «1».
11	Вихід Q0.0 встановлюється в «1» при натисканні будь-яких трьох (з п'яти) кнопок, при умові, що натиснута п'ята кнопка (I0.4)), а також при натисканні комбінації $ABC\overline{D}E$.
12	Вихід Q0.3 встановлюється в «1» при натисканні будь-яких трьох (з п'яти) кнопок, при умові, що натиснута друга кнопка (I0.1), а також при натисканні комбінації $ABC\overline{D}E$.
13	Вихід Q0.2 встановлюється в «1» при натисканні будь-яких трьох (з п'яти) кнопок, при умові, що натиснута третя кнопка (I0.2), а також при натисканні комбінації $\overline{A}BCDE$.
14	Вихід Q0.1 встановлюється в «1» при натисканні будь-яких трьох (з п'яти) кнопок, при умові, що натиснута четверта кнопка (I0.3), а також при натисканні комбінації $ABC\overline{D}E$.
15	Вихід Q0.4 встановлюється в «1» при натисканні будь-яких трьох (з п'яти) кнопок, при умові, що натиснута перша кнопка (I0.0), а також при натисканні комбінації $ABC\overline{D}E$.

9. Структура звіту

1. Номер і назва лабораторної роботи, із зазначенням її виконавця.
2. Мета роботи.
3. Завдання до лабораторної роботи.
4. Короткі теоретичні відомості, що необхідні для виконання лабораторної роботи.
5. Таблиця символів модулів використаних у програмі.

6. Таблиця істинності згідно варіанту, заповнені карти Карно та мінімізований вираз функції.
7. Остаточна версія програми.
8. Висновки.

10. Контрольні запитання

1. Що таке ПЛК?
2. Які мови програмування використовує SIMATIC Step7?
3. Для чого використовується модуль HW Config?
4. В чому полягає різниця між символною та абсолютною адресацією?
5. Для чого використовується організаційний блок OB1?
6. Яка послідовність виконання логічних операцій мовами FBD та LAD?
7. Для чого використовується модуль S7-PLSIM?
8. Для чого використовуються карти Карно?
9. Розкажіть суть методу мінімізації булевих виразів методом карт Карно.
10. Як мінімізувати вираз з допомогою карт Карно для 4-х змінних?
11. В чому полягає складність у мінімізації виразу для 5-и та більше змінних?

Список літератури

1. Hans Berger. Automating with STEP7 in LAD and FBD. Programmable Controllers SIMATIC S7-300/400. 5th revised and enlarged edition, 2012. – 451 p.
2. Hans Berger. Automating with STEP7 in STL and SCL. Programmable Controllers SIMATIC S7-300/400. 6th revised and enlarged edition, 2012.- 553 p.
3. С.Т. Jones. STEP 7 in 7 Steps. First Edition, 2006.- 464 с.
4. Конфігурація модулів VIPA з допомогою ПЗ STEP 7 Simatic Manager. <https://www.cta.ru/articles/cta/spravochnik/v-zapisnuyu-knizhku-inzhenera/124982/>
5. Barry Wilkinson. Digital System Design. 2d. Ed. Prentice Hall. 1992. – 538 p.
6. Схемотехніка електронних систем: У 3 кн. Кн. 2. Цифрова схемотехніка: Підручник / В.І. Бойко, А.М. Гуржій, В.Я. Жуйков та ін. – 2-ге вид., допов. і переробл. – К.:Вища шк. – 2004. – 423 с.

ЗМІСТ

Вступ	1
1. Запуск SIMATIC Manager та створення нового проекту	2
2. Модуль HW Config	5
3. Організаційний блок головної програми OB1	12
4. Модуль S7-PLCSIM	16
5. Налаштування програматора у Step 7	17
6. Співставлення програми FBD та синтезованої цифрової схеми	19
7. Конфігурація навчального стенду	28
8. Порядок виконання роботи	29
9. Структура звіту.....	30
10. Контрольні запитання	31
Список літератури	31