МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова

ЗАСТОСУВАННЯ ЦИФРОВИХ ДВІЙНИКІВ ТЕХНІЧНИХ ЗАСОБІВ АВТОМАТИЗАЦІЇ ДЛЯ РОЗРОБЛЕННЯ ПРОГРАМНО-ТЕХНІЧНИХ КОМПЛЕКСІВ АСУ ТП

Навчальний посібник

ВИДАВНИЦТВО ХАРКІВ 2023 І. С. ІВАНЧЕНКА

УДК 681.51
Н40

Рецензенти:

– Притчин С. Е., д-р техн. наук, професор кафедри автоматизації та інформаційних систем, Кременчуцький національний університет імені Михайла Остроградського;

– Хорошайло Ю. Є., канд. техн. наук, професор, завідувач кафедри проектування та експлуатації електронних апаратів, Харківський національний університет радіоелектроніки.

Рекомендовано Вченою радою
Харківського національного університету радіоелектроніки
(протокол №5/1 від 04.05.2023 р.).

Невлюдов І. Ш.

Н40 Застосування цифрових двійників технічних засобів автоматизації для розроблення програмно-технічних комплексів АСУ ТП : Навчальний посібник / І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова. – Харків: Видавництво Іванченка І. С., 2023. – 267 с.

ISBN 978-617-8059-95-8
DOI: 10.30837/978-617-8059-95-8

У навчальному посібнику розглядаються питання практичного застосовування віртуальних макетів технічних засобів автоматизації, які використовуються для розроблення програмно-технічних комплексів сучасних АСУ ТП. Наведено приклади найпоширеніших технічних засобів автоматизації технологічних процесів і реалізації керування ними за допомогою релейно-контактної логіки.

Зміст навчального посібника «Застосування цифрових двійників технічних засобів автоматизації для розроблення програмно-технічних комплексів АСУ ТП» відповідає освітнім програмам підготовки бакалаврів і магістрів спеціальності 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка».

УДК 681.51

ISBN 978-617-8059-95-8
DOI: 10.30837/978-617-8059-95-8

ЗМІСТ

3 ОПИС ВІРТУАЛЬНИХ ПРИЛАДІВ

3.1 Віртуальний прилад «Світлова колона»

Даний віртуальний прилад є цифровим двійником світлосигнальної колони для візуалізації стану промислового обладнання. Детально про роботу даного обладнання розказано в п. 2.1 цього посібника.

На рис. 3.1 подано зовнішній вигляд інтерфейсу віртуального приладу.

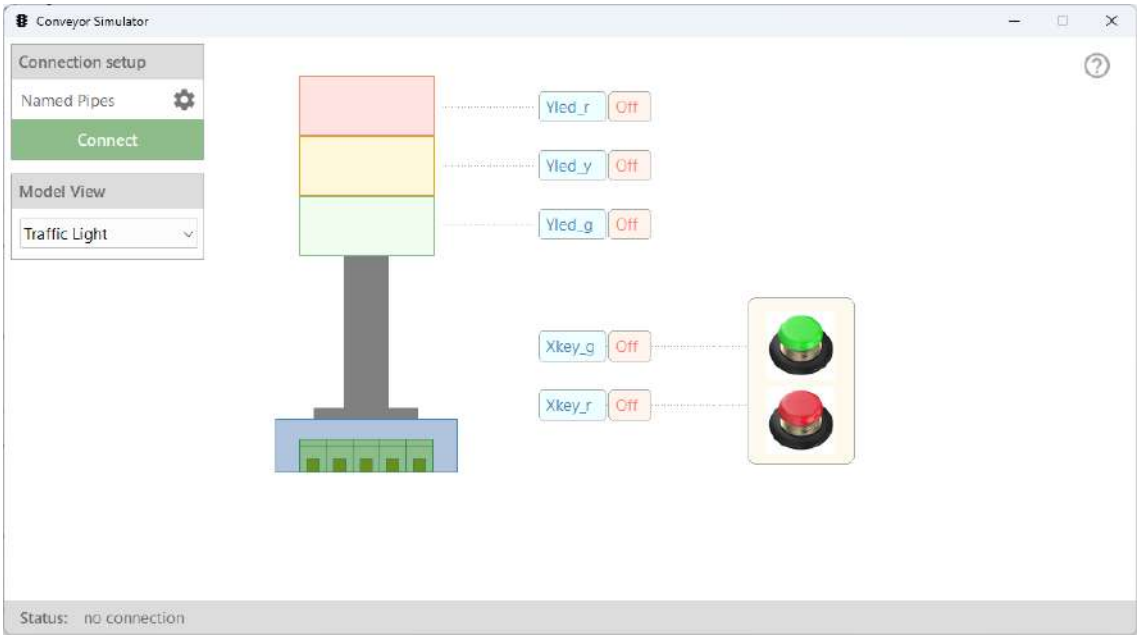


Рисунок 3.1 – Зовнішній вигляд інтерфейсу віртуального приладу «Світлова колона»

Віртуальний прилад повністю реалізує поведінку реального приладу та може працювати із зовнішньою програмою або іншим пристроєм за одним із можливих протоколів:

– Named Pipes для Inter-Process Communication;

– Modbus TCP/IP;

– Serial Protocol.

Іменовані канали (Named Pipes) забезпечують міжпроцесний зв’язок (Inter-Process Communication) між сервером каналів і одним або кількома клієнтами каналів. Вони пропонують більше функціональних можливостей, ніж анонімні канали, які забезпечують міжпроцесний зв’язок на локальному комп’ютері.

Іменовані канали підтримують повний дуплексний зв’язок через мережу та кілька екземплярів сервера, зв’язок на основі повідомлень та уособлення клієнта, що дозволяє процесам підключення використовувати власний набір дозволів на віддалених серверах.

Цей тип взаємодії є основним для роботи з програмою LDmicro, що є інструментом створення технологічних програм мовою LD.

Modbus TCP/IP використовується в промисловості для поєднання засобів автоматизації в єдину промислову мережу. В віртуальному приладі наявність підтримки цього протоколу надає можливість використовувати його спільно з інтегрованим середовищем CODESYS в якому відсутній прямий доступ до об’єкту керування з програми, робота якої симулюється. Використання посередника SoftPLC (іде в наборі утиліт разом з CODESYS) та протоколу Modbus TCP/IP надає можливість застосовувати даний віртуальний прилад в єдиному віртуальному комплексі.

Підтримка Serial Protocol надає можливість використовувати віртуальний прилад для налагодження програм разом з реальними пристроями, наприклад із Arduino. Реалізація даного протоколу в віртуальному макеті дає можливість використовувати його не тільки для поєднання з середовищем розробки програмного засобу і налагодження написаної програми, а й для самостійного використання програми, як незалежного віртуального пристрою.

В інтерактивному інтерфейсі можна виділити:

– органи керування (зелена та червона кнопки);

– органи відображення інформації (червона, жовта та зелена лампа).

Для полегшення сприйняття принципу роботи цифрового двійника, на екранній формі органи відображення інформації поєднані в єдину конструкцію, що зовні виглядає ідентично реальному пристрою (рис. 2.3).

Кнопки керування мають візуальний ефект в процесі взаємодії з ними користувача. Наведення курсора та натискання кнопки миші призводить до того,

що віртуальна кнопка теж «натискається», а відпускання повертає її в нормальне положення. Віртуальна кнопка працює в режимі нормально розімкнених контактів. Її стан можна контролювати за візуальною міткою (рис. 3.2).

В інтерфейсі візуальної мітки є таки елементи:

– назва змінної;

– поточний стан змінної;

– лінія поєднання з елементом керування або відображення.

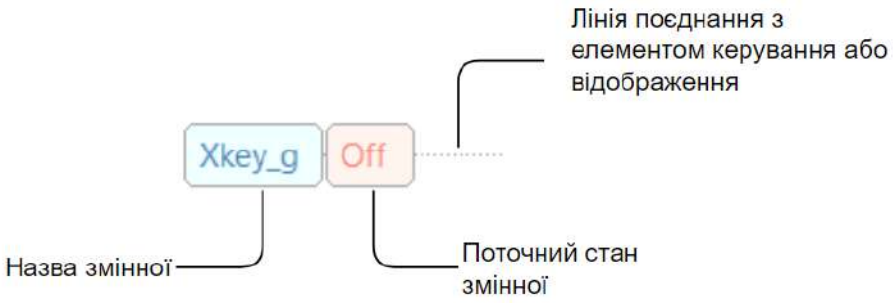


Рисунок 3.2 – Візуальна мітка стану елементу керування

Назва змінної – це ідентифікатор, що використовується в зовнішній програмі для доступу до цифрового двійника, до потрібного органу керування або візуалізації. Літера «X» в префіксі назви змінної означає, що вона пов’язана з вхідною інформацією для ПЛК (кнопка пов’язана з зовнішнім вхідним контактом ПЛК). Літера «Y» – означає, що змінна пов’язана з вихідною інформацією, яка йде від ПЛК (лампа пов’язана з зовнішнім вихідним контактом (реле) ПЛК).

Наприклад, щоб прочитати стан зеленої кнопки віртуального макету, необхідно в програмі LD створити такий рядок (рис. 3.3).

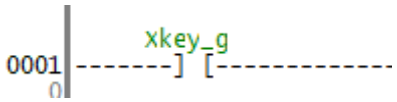


Рисунок 3.3 – Читання стану зеленої кнопки віртуального макету з назвою «Xkey_g»

Назви змінних, що вказані у віртуальних приладах неможна змінювати, тому при написанні програми потрібно чітко стежити за відповідними мітками, що пов’язані з елементами керування та відображення інформації.

Поточний стан змінної залежить від стану органу керування або від результатів роботи технологічної програми. Якщо змінна пов’язана з кнопкою, або іншим органом керування на віртуальному приладі, то її стан змінюється в незалежності від того, чи підключена до віртуального приладу, наприклад, програма LDmicro. З натисканням кнопки можна бачити, що стан змінної змінився (рис. 3.4).

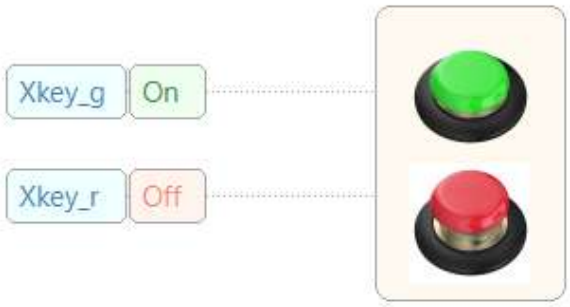


Рисунок 3.4 – Зміна стану змінної з натисканням кнопки

З рис. 3.4 можна бачити, що натиснута зелена кнопка ініціювала зміну стану змінної «Xkey_g» на «On», в той же час, не натиснута червона кнопка представлена значенням «Off» змінної «Xkey_r».

Для управління засобами візуалізації віртуального приладу, наприклад, лампами світлової колони, необхідно поєднати їх з програмними компонентами, наприклад, підключити до вихідного реле.

Підключення до вихідного реле означає, що його назва співпадає з назвою відповідної змінної. Наприклад, щоб мати змогу керувати червоною лампою світлової колони, яка представлена змінною «Yled_r», необхідно в технологічній програмі мовою LD в LDmicro створити відповідне вихідне реле з такою самою назвою (рис. 3.5).

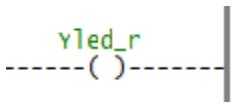


Рисунок 3.5 – Створення вихідного реле для керування
червоною лампою світлової колони

Таким чином, для вирішення задачі увімкнення червоної лампи, якщо натиснути зелену кнопку, програма мовою LD буде мати наступний вигляд (рис. 3.6).

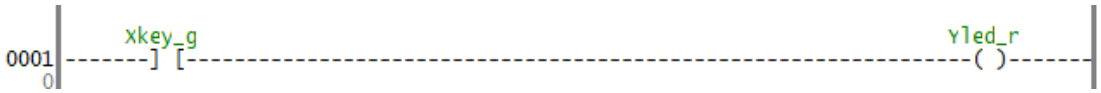


Рисунок 3.6 – Увімкнення червоної лампи
з натисканням зеленої кнопки

Процес взаємодії між віртуальним приладом і програмою LDmicro подано на рис. 3.7.

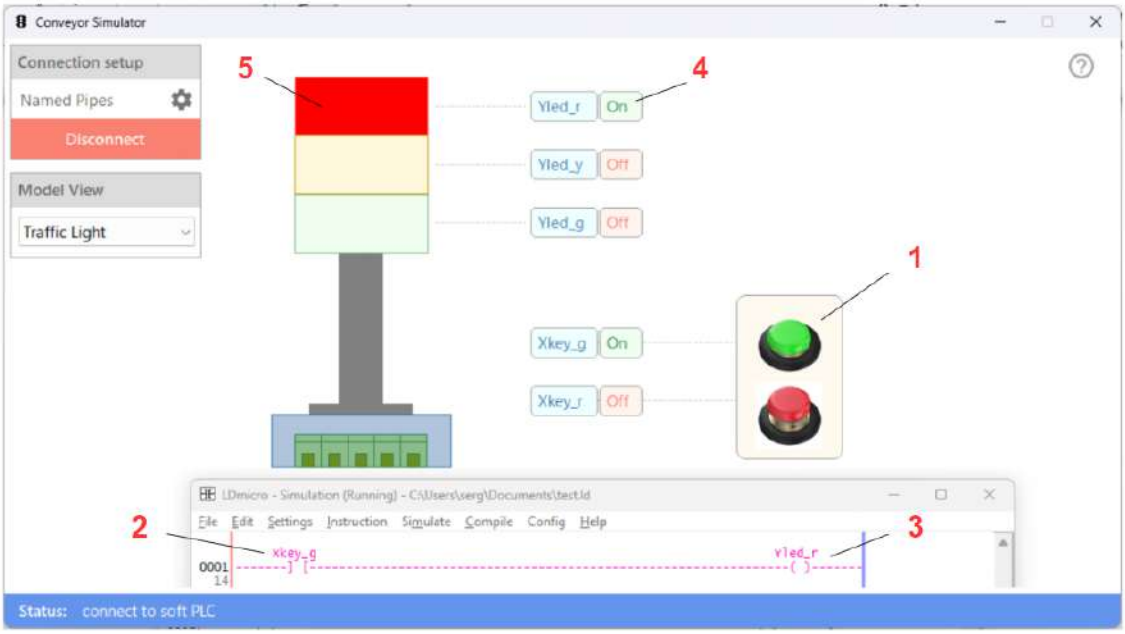


Рисунок 3.7 – Процес взаємодії між віртуальним приладом і програмою LDmicro

Натискання на зелену кнопку призводить до зміни стану змінної «Xkey_g» з «Off» на «On» (рис. 3.7, поз. 1).

Стан змінної зчитується внутрішньою функцією віртуального приладу та передається засобами «Inter-Process Communication» до програми LDmicro, в якій запущено симуляцію роботи програми керування світловою колоною.

На рис. 3.7 (поз. 2) можна бачити, що вхідний контакт з відповідною назвою «Xkey_g» позначено червоним кольором. Це означає, що LDmicro отримав інформацію про стан вхідних контактів, виконав ідентифікацію та змінив стан відповідного вхідного контакту з «false» на «true».

За правилами LD, якщо контакт має стан «true», то через нього проходить напруга і, в даному випадку, вмикає зовнішнє реле «Yled_r» (рис. 3.7, поз. 3).

Інформація про стан вихідного реле «Yled_r» засобами «Inter-Process Communication» передається від програми LDmicro до віртуального приладу, що призводить до зміни стану змінної «Yled_r» (рис. 3.7, поз. 4).

Зміна стану змінної «Yled_r» відображається на екрані ПК у вигляді «запалення» відповідної червоної лампи (рис. 3.7, поз. 5).

Управління іншими лампами та зчитування стану кнопок виконується аналогічно.

Підключення LDmicro до віртуального приладу відбувається наступним чином. На початку роботи у віртуальному приладі необхідно обрати тип з’єднання з програмою, в якій виконується написання технологічного коду. В нашому випадку обирається тип з’єднання «Named Pipes» для підключення до модифікованої версії LDmicro. Дана програма є постачальником інформації через NamedPipes.

Після обирання типу з’єднання необхідно натиснути кнопку «Connect» (рис. 3.8). Після цього віртуальний прилад постійно очікує з’єднання з іншою програмою та надходження від неї інформації про стан вихідних контактів. Також він очікує можливості передавання на зовнішню програму даних про стан вхідних контактів.

Підключення відбувається один раз впродовж сеансу роботи з програмою LDmicro. Для внесення змін в технологічну програму сеанс розривати немає потреби. Після натискання кнопки «Disconnect» потрібно завершити, та знов розпочати роботу із програмою.

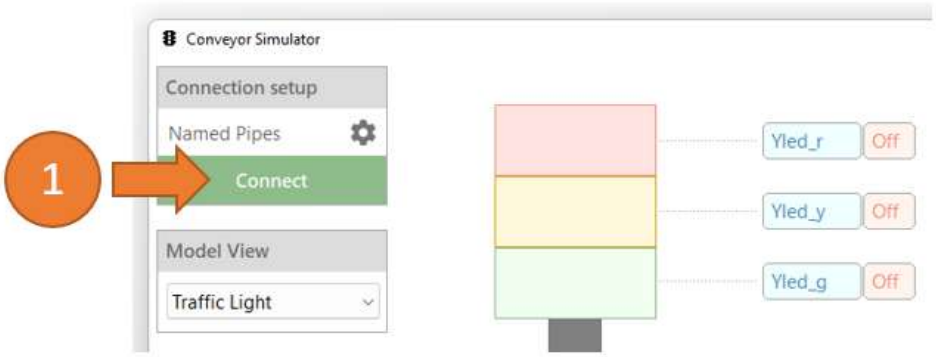


Рисунок 3.8 – Підключення віртуального приладу до постачальника інформації через NamedPipes

В програмі LDmicro для підключення до віртуального приладу необхідно виконати два наступних кроки (рис. 3.9):

– обрати режим симуляції у відповідному меню;

– виконати старт симуляції в реальному часі.

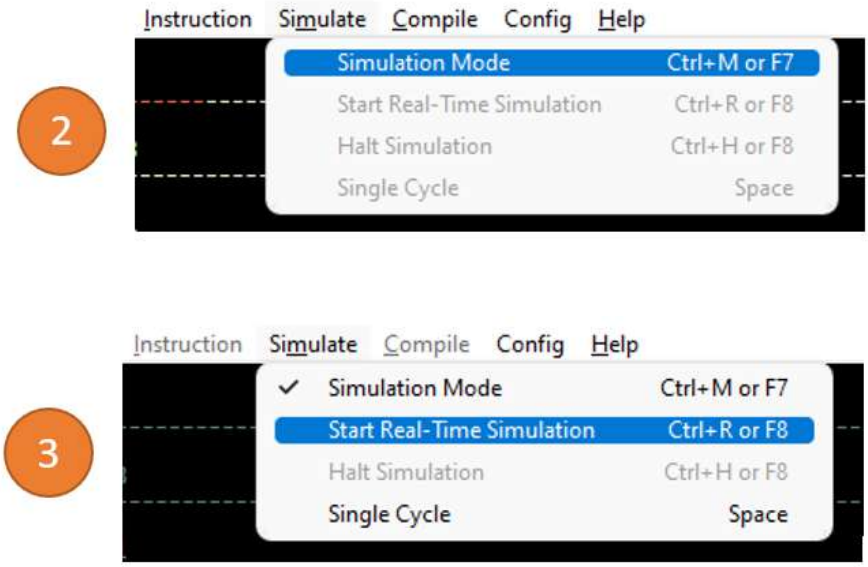


Рисунок 3.9 – Підключення до віртуального приладу з боку LDmicro

Якщо потрібно внести зміни в технологічну програму управління ПЛК, потрібно зупинити симуляцію з боку LDmicro. У віртуальному приладі натискати

кнопку «Disconnect» не потрібно. Дана програма повинна постійно перебувати в режимі очікування та реагування на запити від програм-клієнтів.

3.2 Віртуальний прилад «Штампувальний автомат»

Зовнішній вигляд інтерфейсу користувача віртуального приладу подано на рис. 3.10.

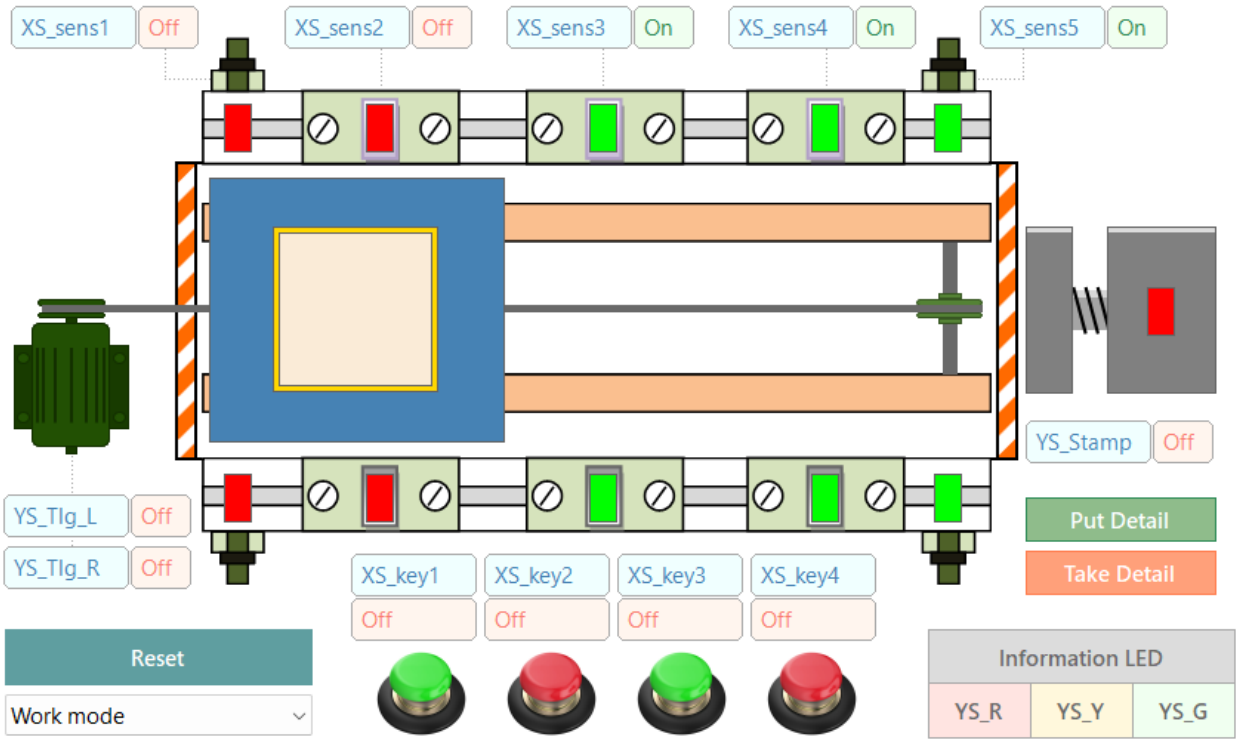


Рисунок 3.10 – Зовнішній вигляд інтерфейсу користувача віртуального приладу «Штампувальний автомат»

Даний віртуальний прилад за функціональними можливостями відповідає реальному лабораторному макету «Штампувальний автомат», зовнішній вигляд якого показано на рис. 2.13.

У відповідності до єдиної концепції побудови віртуальних приладів, кожен елемент керування та відображення інформації має візуальну мітку стану (рис. 3.2).

Позначення основних елементів віртуального приладу подано на рис. 3.11.

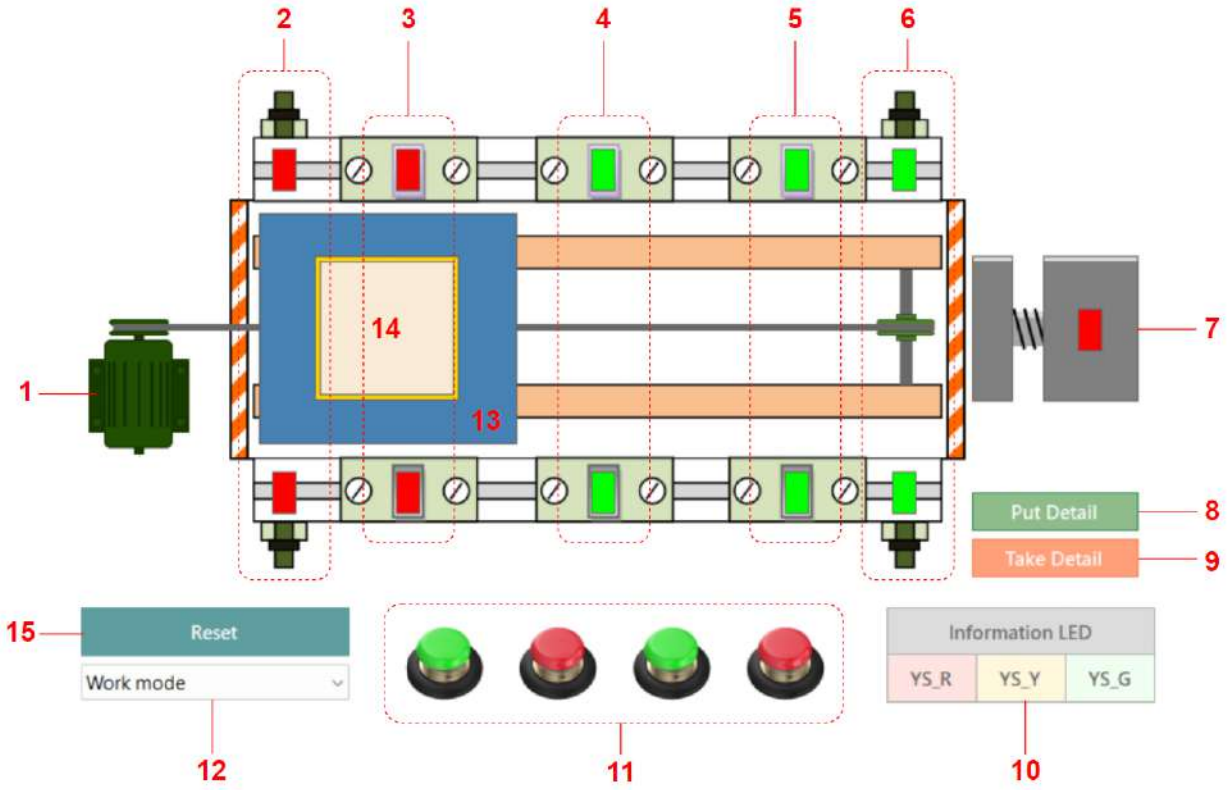


Рисунок 3.11 – Позначення основних елементів віртуального приладу

Теліжку 13 з деталлю 14 рухає двигун 1. Рух може відбуватись зліва на право або в зворотному напрямку. Фізичного обмеження руху не існує, це основна особливість, як фізичного макету, так і його цифрового двійника. Зупинка теліжки виконується програмним шляхом на основі даних, що отримуються з датчиків 2 та 6. Датчик 2 – це лівий кінцевий датчик, а 6 – правий.

Всі датчики в даному пристрої побудовані за принципом закритого світлового каналу та мають розгалужену в просторі структуру. Принцип роботи такого датчика подано в розділі 2.2 на рис. 2.17.

Однією з умов роботи керуючої програми є запобігання виїзду теліжки за обмежувальну лінію (рис. 3.12). Технологічна програма повинна оброблювати інформацію з кінцевих датчиків 2 та 6 і вчасно змінювати стан вихідних реле, до яких підключено двигун 1, що рухає теліжку.

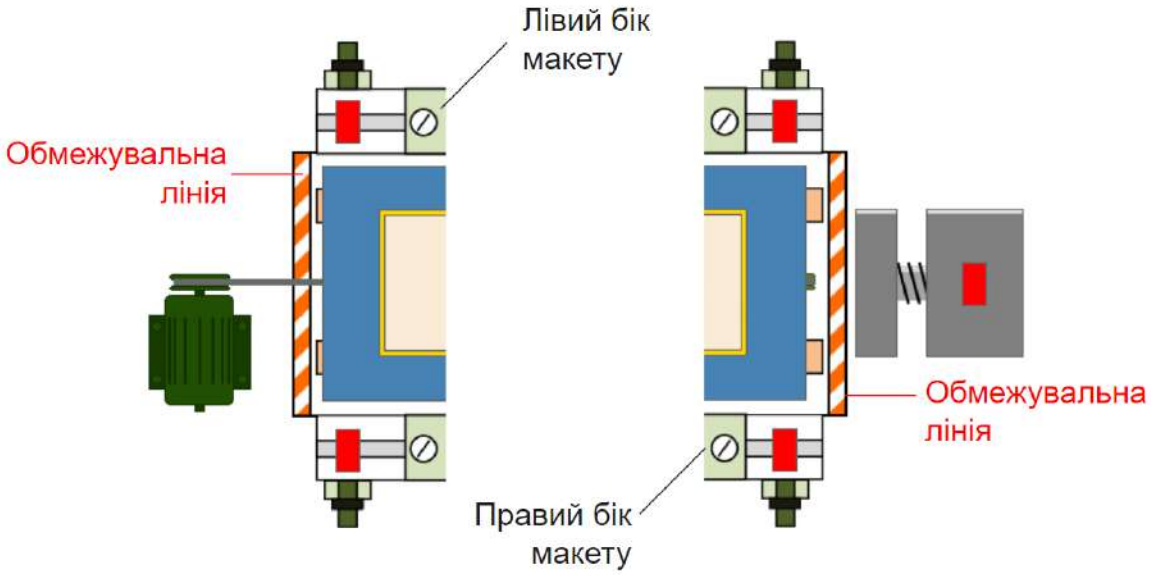


Рисунок 3.12 – Обмежувальна лінія штампувального автомата

У віртуальному приладі передбачена функція оброблення помилки керування та, в разі виїзду теліжки за обмежувальну лінію (ліву або праву), на екрані з’явиться попереджувальна інформація (рис. 3.13), а робота макету буде аварійно зупинена до натискання кнопки Reset (3.11, поз. 15).

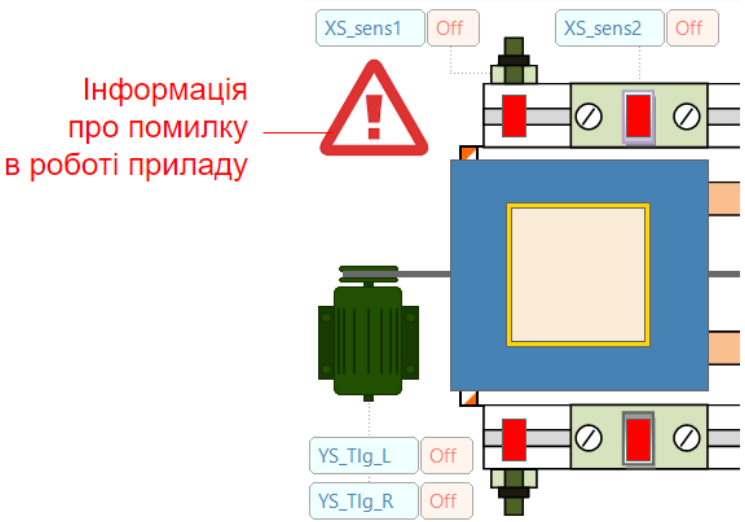


Рисунок 3.13 – Виїзд теліжки за обмежувальну лінію

Дана ситуація в реальній сфері застосування автоматизованого пристрою є критичною, тому в цифровому двійнику вона оброблюється на рівні основної програми управління віртуальним приладом.

Управління роботою двигуна, що приводить до руху теліжку, відбувається двома сигналами «YS_Tlg_L» та «YS_Tlg_R». Реакція двигуна на різні комбінації сигналів подана у табл. 3.1

Таблиця 3.1 – Режим роботи двигуна в залежності від комбінації сигналів

YS_Tlg_L	YS_Tlg_R	Режим роботи двигуна
0 (Off)	0 (Off)	Двигун не працює
1 (On)	0 (Off)	Теліжка рухається вліво
0 (Off)	1 (On)	Теліжка рухається вправо
1 (On)	1 (On)	Невизначена ситуація. Рух теліжки призупинено

Датчики 3, 4 та 5 (рис. 3.11) призначені для контролю наявності вантажу, що міститься на теліжці. Зелений колір датчика означає, що він знаходиться в стані спрацювання (між випромінювачем та приймачем немає перешкоди). Червоний колір означає, що між частинами датчика знаходиться якийсь предмет. Датчик 3 контролює зону завантаження. Датчик 4 визначає середню позицію розташування теліжки з вантажем. Датчик 5 контролює наявність деталі на теліжці в робочій зоні штампувального автомата.

Механізм штампування (рис. 3.11, поз. 7) керується вихідним сигналом «YS_Stamp». Поява логічної одиниці на цьому реле призводить включення механізму приводу штампу. Цей процес займає деякий час та не залежить від логічної одиниці на реле «YS_Stamp» після початку процесу руху робочої частини штампу. В керуючій програмі можна згенерувати короткий імпульс для вмикання даного режиму.

Робоча частина штампу повернеться в початковий стан очікування керуючого сигналу після завершення процесу штампування. Також, після завершення даного процесу на деталі (рис. 3.11, поз. 14) з’явиться візуальна мітка, що підтверджує виконання операції (рис. 3.14).

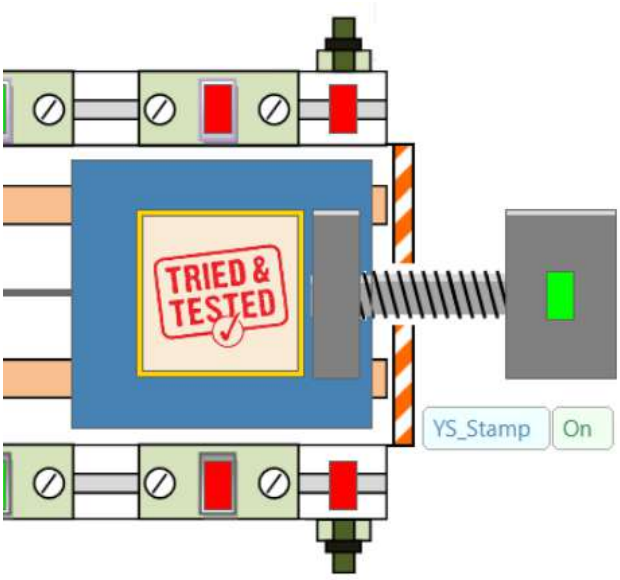


Рисунок 3.14 – Виконання процесу штампування

Мітка буде очищена після ручного зняття та встановлення нової деталі на теліжку. Для керування цим процесом в віртуальному приладі передбачені дві кнопки:

– Put Detail (покласти нову деталь на теліжку);

– Take Detail (прибрати деталь з теліжки).

Прибрати деталь з теліжки можна в будь який час. Програма, що керує автоматом повинна опрацьовувати цю ситуацію та, наприклад, зупиняти рух теліжки і повертати її до зони завантаження.

У віртуальному приладі додатково передбачені декілька органів керування та візуалізації для реалізації розвинутих алгоритмів керування процесом штампування. Наприклад, є інформаційне поле (рис. 3.11, поз. 10) з трьома світлодіодами, що виконує функції подібні до світлової колони (рис. 3.15).

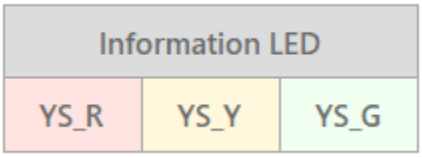


Рисунок 3.15 – Інформаційне поле з трьома світлодіодами

Дане поле має три індикатори, які керуються відповідними вхідними сигналами:

– червоний світлодіод керується сигналом «YS_R»;

– жовтий світлодіод керується сигналом «YS_Y»;

– зелений світлодіод керується сигналом «YS_G».

Також передбачено блок з чотирьох кнопок (рис. 3.11, поз. 11), який користувач може використовувати для виконання функцій, що передбачені в завданні (рис. 3.16).

Рисунок 3.16 – Блок кнопок користувача

Кожна кнопка з блоку прив’язана до відповідної змінної:

– кнопка 1 пов’язана зі змінною «XS_key1»;

– кнопка 2 пов’язана зі змінною «XS_key2»;

– кнопка 3 пов’язана зі змінною «XS_key3»;

– кнопка 4 пов’язана зі змінною «XS_key4»;

Відповідні інформаційні мітки сигналізують про поточний стан кнопок. Кнопки мають нормально відкриті контакти та не мають фіксації, тобто після відпускання повертаються у вимкнений стан.

Віртуальний прилад має два режими роботи (елемент вибору на рис. 3.11, поз. 12):

– Work mode (робочий режим);

– Demo mode (демонстраційний режим).

В робочому режимі прилад реагує тільки на зовнішню програму керування, наприклад, яку завантажено в режимі симуляції в модифікованій програмі LDmicro. Принцип роботи подано на рис. 3.17.

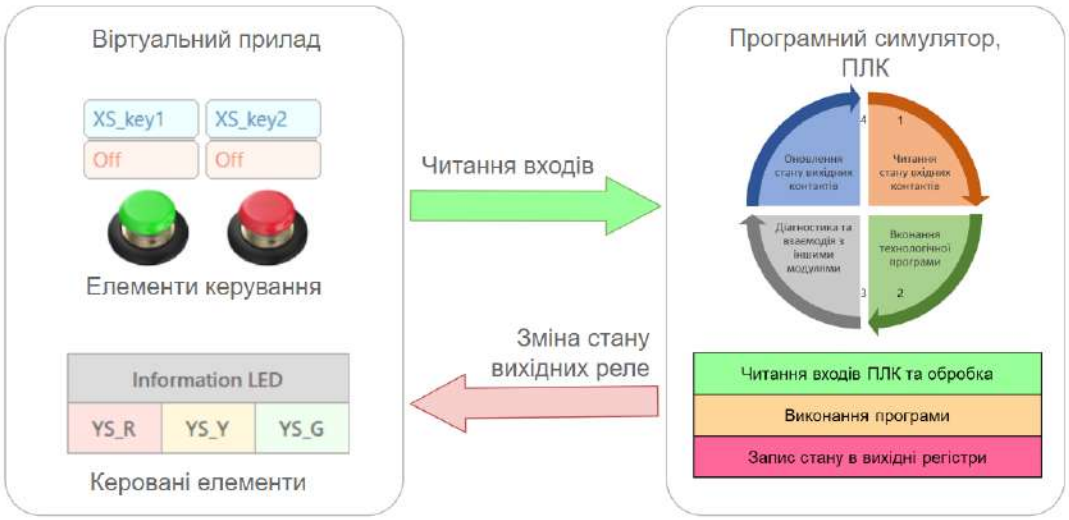


Рисунок 3.17 – Принцип взаємодії віртуального приладу і програмного симулятора ПЛК

Віртуальна програма виробляє сигнали, які пов’язані з вхідними контактами ПЛК. Технологічна програма для ПЛК опитує вхідні контакти, аналізує їх стан, та, за розробленим алгоритмом дій, виробляє вихідні сигнали на реле, що пов’язані з відповідними елементами у віртуальному приладі.

Елементи інтерфейсу, які мають назву з першою літерою «X», підключені до вхідних контактів та виробляють сигнали керування. Наприклад, такими елементами є кнопки.

Елементи інтерфейсу, що мають назву з першою літерою «Y», підключені до вихідних контактів та вони отримують сигнали керування від ПЛК. Такими елементами, наприклад, є двигун, штампувальний механізм, світлодіодні індикатори.

В режимі демонстрації віртуальний прилад налаштовано на виконання команд від кнопок керування без реалізації функцій контролю стану датчиків автоматизованої системи.

Після переходу в демонстраційний режим запускається двигун та теліжка рухається вправо. Для керування її рухом можна використовувати кнопки. Наприклад, натискання на кнопку 1 («XS_key1») призводить до зміни напрямку

руху теліжки. Якщо вона їхала вправо, то після натискання на кнопку 1 рух зміниться на протилежний – вліво.

Кнопка 2 («XS_key2») вмикає рух механізму штампа. Якщо, в той час, коли механізм дійде до крайньої позиції, там буде знаходиться деталь, то на ній з'явиться відмітка про виконання процесу штампування.

Кнопки 3 та 4 в демонстраційному режимі не задіяні. Кнопки Put Detail (покласти нову деталь на теліжку) та Take Detail (прибрати деталь з теліжки) працюють в звичайному для них режимах.

3.3 Віртуальний прилад «Конвеєр та технологічна лінія»

Зовнішній вигляд інтерфейсу користувача віртуального приладу «Конвеєр та технологічна лінія» подано на рис. 3.18. Даний віртуальний прилад є цифровим двійником мехатронного модуля транспортування та розподілення фірми FESTO, принцип дії якого описано в п. 2.3 цього посібника.

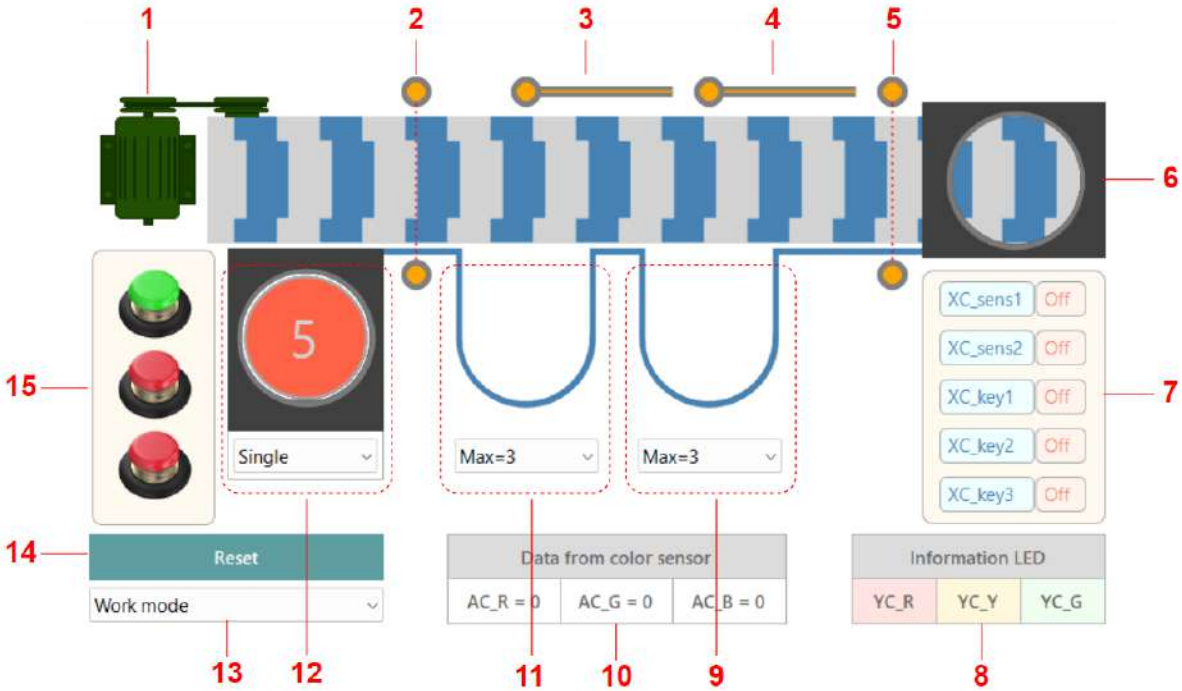


Рисунок 3.18 – Зовнішній вигляд інтерфейсу користувача віртуального приладу «Конвеєр та технологічна лінія»

Особливістю реалізації даного приладу є поєднання конвеєрної лінії з модулем живлення деталями конвеєрної лінії в єдиний виробничій модуль. Принцип дії модуля живлення від фірми FESTO подано в п. 2.4.

Програмна реалізація цих двох макетів дозволяє виконувати функції:

– видачі деталей для живлення конвеєрної лінії;

– переміщення їх конвеєрною лінією;

– за потреби, розподілення потоку деталей в залежності від характеристик деталей, або за іншим принципом;

– управління режимом роботи макету за допомогою додаткового блоку кнопок;

– візуалізація поточного стану всіх елементів макету.

Розглянемо призначення елементів керування та візуалізації стану віртуального макету. Двигун 1 (рис. 3.18) рухає конвеєрну стрічку тільки в одному напрямку – зліва на право, тому для включення двигуна використовується тільки один сигнал та відповідна змінна «YC_Moto».

Віртуальний макет має три датчики:

– датчик рахунку деталей на вході конвеєра (рис. 3.18, поз. 2);

– датчик рахунку деталей на виході з конвеєра (рис. 3.18, поз. 5);

– аналоговий датчик кольору деталі (рис. 3.18, поз. 11), що конструктивно поєднаний з датчиком рахунку деталей на вході (рис. 3.18, поз. 2).

Датчики 2 та 5 дискретні. Сигнал на їх виході може бути одним з двох можливих станів: true або false (on та off).

Датчик рахунку деталей (рис. 3.18, поз. 2) видає сигнал логічної одиниці, коли вона перетинає його промінь, і утримує незмінний вихідний сигнал протягом всього часу, поки деталь не вийде за його межі. З цим датчиком пов’язана змінна «SC_sens1». Датчик 5 (рис. 3.18) працює аналогічним чином, а його стан пов’язаний зі змінною «SC_sens2».

Для визначення кольору деталей, що надходять до конвеєрної лінії використовується аналоговий датчик (рис. 3.18, поз. 11). Його розташовано поряд з датчиком 2, та кожен раз, коли деталь проходить повз нього, на виході отримуємо комбінацію із трьох сигналів R, G та B (рис. 3.19).

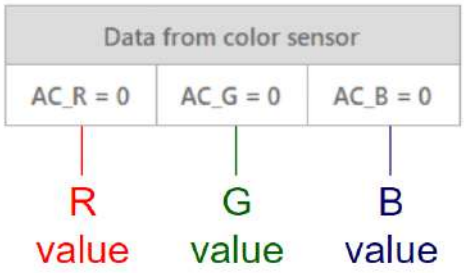


Рисунок 3.19 – Три канали датчика кольору

В датчику 11 застосовано 8-ми бітний АЦП. Кожен з трьох каналів формує на виході число в діапазоні від 0 до 255, а комбінація із трьох чисел описує колір деталі, що пройшла повз датчик.

За кожним із каналів закріплена відповідна змінна. Червоний канал пов'язано зі змінною «AC_R», зелений – «AC_G», синій – «AC_B». Перша літера в назві змінної означає, що це дані з АЦП та за правилами програми LDmicro вони можуть зчитуватись вбудованою інструкцією ReadADC.

Рухомі заслінки 3 та 4 (рис. 3.18) призначені для розділення потоку деталей на конвеєрі. Вони можуть займати одно з двох стійких положень (рис. 3.20).

В нормальному режимі, коли на привід заслінки не надходить керуючий сигнал, вона перебуває в горизонтальному положенні (рис. 3.20, а). Якщо надходить керуючий сигнал, спрацьовує привід заслінки і вона переводиться в положення, коли потік деталей перекривається під кутом 45 градусів (рис. 3.20, б). В такому її положенні деталі, що рухаються конвеєрною стрічкою, змінюють напрям руху та зміщуються в бік одного з накопичувачів 9 або 11.

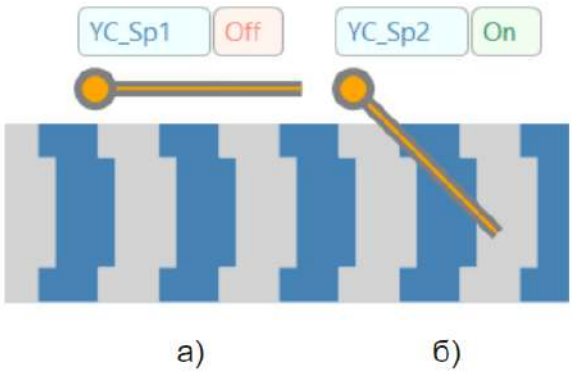


Рисунок 3.20 – Рухомі заслінки для розділення потоку деталей

Принцип дії рухомих важелів подано на рис. 3.21.

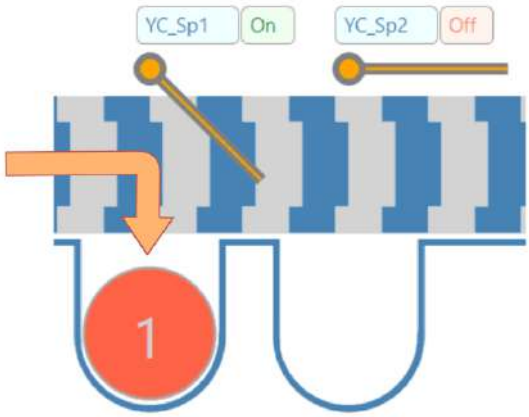


Рисунок 3.21 – Принцип дії рухомих важелів

З надходженням деталі в накопичувач збільшується число, яке відображає поточну кількість виробів в даній зоні.

В віртуальному макеті можна задавати максимально можливу кількість деталей в накопичувачі (рис. 3.22). В цьому разі, в керуючій програмі збоку ПЛК необхідно слідкувати за тим, щоб цей ліміт не було перевищено.

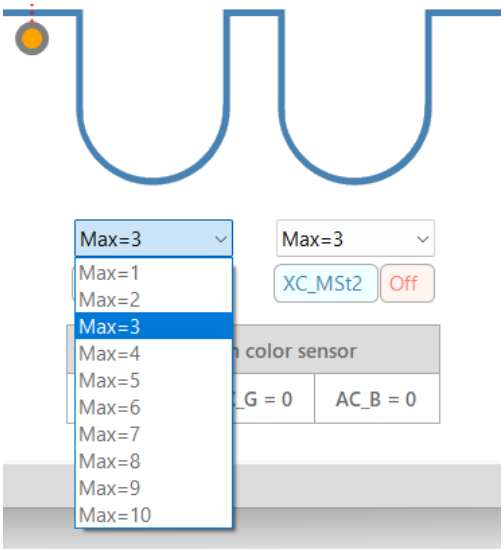


Рисунок 3.22 – Управління максимально можливою кількістю деталей в накопичувачі

У випадку, коли максимальна кількість деталей досягнута, а програма все одно перенаправляє нову деталь в накопичувач, віртуальний прилад згенерує помилку. Помилка буде показана користувачу у вигляді діалогового вікна (рис. 3.23).

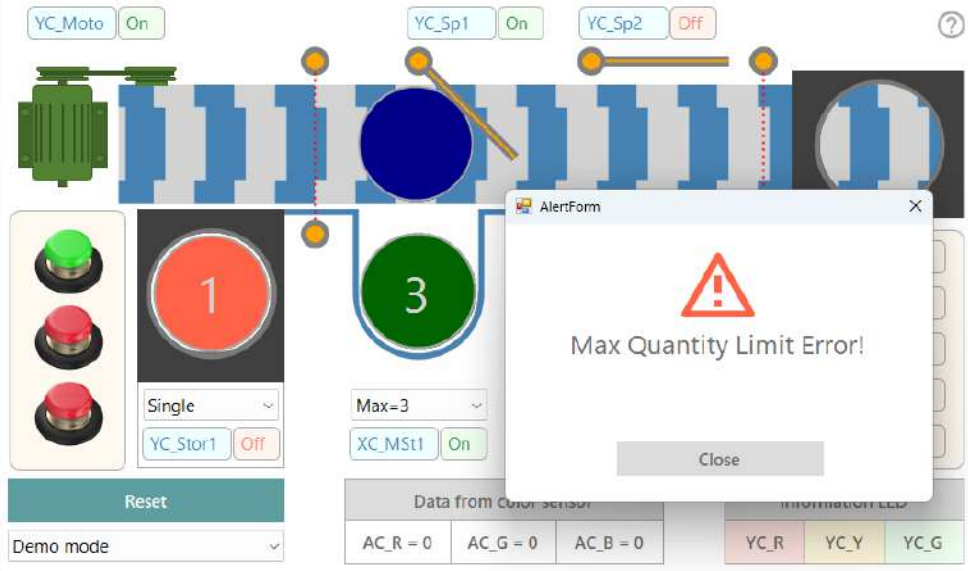


Рисунок 3.23 – Вікно, що сповіщає про помилку

З рис. 3.23 можна бачити, що для накопичувача встановлено ліміт кількості деталей, що дорівнює трьом, але програма керування направила ще одну, синю, деталь в той самий накопичувач. В даному випадку на екрані з'являється повідомлення про помилку режиму виконання «Max Quantity Limit Error!». Після натискання на кнопку «Close» потрібно перезапустити програму натиснувши на кнопку «Reset» (рис. 3.18, поз. 14).

Всі деталі, що пройшли конвеєром та не були направлені в інший потік, проходять вздовж датчика 5 та потрапляють до накопичувача 6 (рис. 3.18). Кількість деталей, що порахована датчиком 6 відображається на фоні деталі в кінцевому накопичувачі.

Магазин деталей (рис. 3.18, поз. 12) живить конвеєр виробами. Він може працювати в двох режимах (рис. 3.24):

– Single: видача деталей по одній;

– Auto: автоматична видача деталей.

Рисунок 3.24 – Вибір режиму роботи магазину деталей

Кількість деталей, що залишилась в магазині, відображається у вигляді числа, що знаходиться на фоні чергової деталі.

Видача деталей в режимі Single виконується по одній в разі надходження сигналу керування на вхід магазину. Даний сигнал пов'язано зі змінною «YC_Stor1». Сигнал логічної одиниці запускає поршень, який виштовхує деталь на конвеєр (рис. 3.25). Доти, доки механізм не завершить процес виштовхування та не повернеться у вихідне положення, він не приймає та не обробляє вхідні керуючі сигнали.

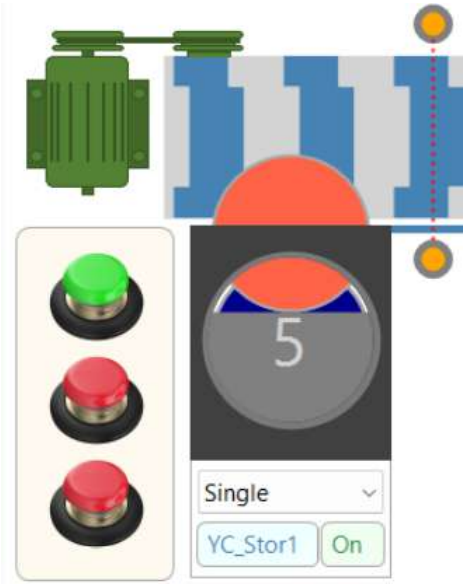


Рисунок 3.25 – Виштовхування деталі з магазину на конвеєр

В автоматичному режимі, механізм видачі деталей працює циклічно. Після завершення видачі чергової деталі, виконується новий цикл виштовхування, доки на закінчяться всі деталі в магазині.

Автоматичний режим працює обох режимах роботи програми – в демонстраційному режимі та в режимі симуляції.

У віртуальному макеті передбачені декілька додаткових органів керування та візуалізації для можливості впровадження різних алгоритмів керування конвеєрною лінією.

Для відображення інформації про поточний стан роботи макету можливо використовувати блок з трьох різнокольорових світлодіодів (рис. 3.18, поз. 8). Інформаційний блок має три індикатори, що керуються відповідними вхідними сигналами (рис. 3.26):

– червоний світлодіод керується сигналом «YC_R»;

– жовтий світлодіод керується сигналом «YC_Y»;

– зелений світлодіод керується сигналом «YC_G».

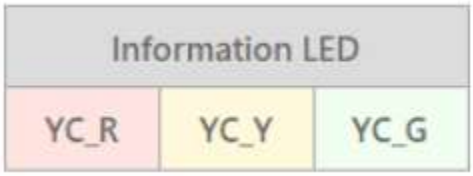


Рисунок 3.26 – Інформаційний блок з трьома світлодіодами

Для керування віртуальним макетом можна застосовувати три кнопки (блок кнопок на рис. 3.18, поз. 15). Поточний стан кожної кнопки можна контролювати за відповідною інформаційною міткою. Всі мітки для трьох кнопок та двох датчиків зібрані в єдиний блок (рис. 3.18, поз. 7).

Кожна кнопка з блоку прив’язана до відповідної змінної (рис. 3.27):

– кнопка 1 (зелена) пов’язана зі змінною «XC_key1»;

– кнопка 2 (червона) пов’язана зі змінною «XS_key2»;

– кнопка 3 (червона) пов’язана зі змінною «XS_key3».

Рисунок 3.27 – Інформаційний блок для індикації стану датчиків та кнопок віртуального макету

При написанні програми керування необхідно мати на увазі, що кнопки мають нормально відкриті контакти та не мають фіксації – після відпускання повертаються у вимкнений стан.

В режимі демонстрації перша кнопка (зелена) запукає процес видачі нової деталі з магазину. Друга кнопка керує заслінкою (рис. 3.18, поз. 3). З натисканням цієї кнопки, заслінка переводиться в положення, що перекриває напрям руху деталі та змінює його для скидання виробу в накопичувач 11. Повторне натискання цієї кнопки повертає важіль в початкове положення і потік деталей відновлюється.

Третя кнопка керує заслінкою 4 (рис. 3.18). З натисканням цієї кнопки, заслінка переводиться в положення, що перенаправляє деталі в накопичувач 9. Повторне натискання на кнопку повертає важіль в початкове положення.

Зміна режимів роботи відбувається з допомогою списку (рис. 3.18, поз. 13).

Щоб перевести макет в початковий стан треба натиснути кнопку «Reset» (рис. 3.18, поз. 14).

3.4 Віртуальний прилад «Модуль формування та відображення стану дискретних сигналів»

Даний віртуальний прилад дозволяє виконувати формування дискретних сигналів для дослідження реакції програми, що розробляється, для управління ПЛК. Також, завдяки наявності великої кількості елементів візуалізації, його можна

використовувати для наочної демонстрації роботи технологічної програми в режимі керування дискретними виходами ПЛК.

Зовнішній вигляд віртуального макету подано на рис. 3.28. Робоче поле являє собою набір перемикачів і кнопок для формування дискретних сигналів та світлодіодів для відображення стану дискретних виходів ПЛК.

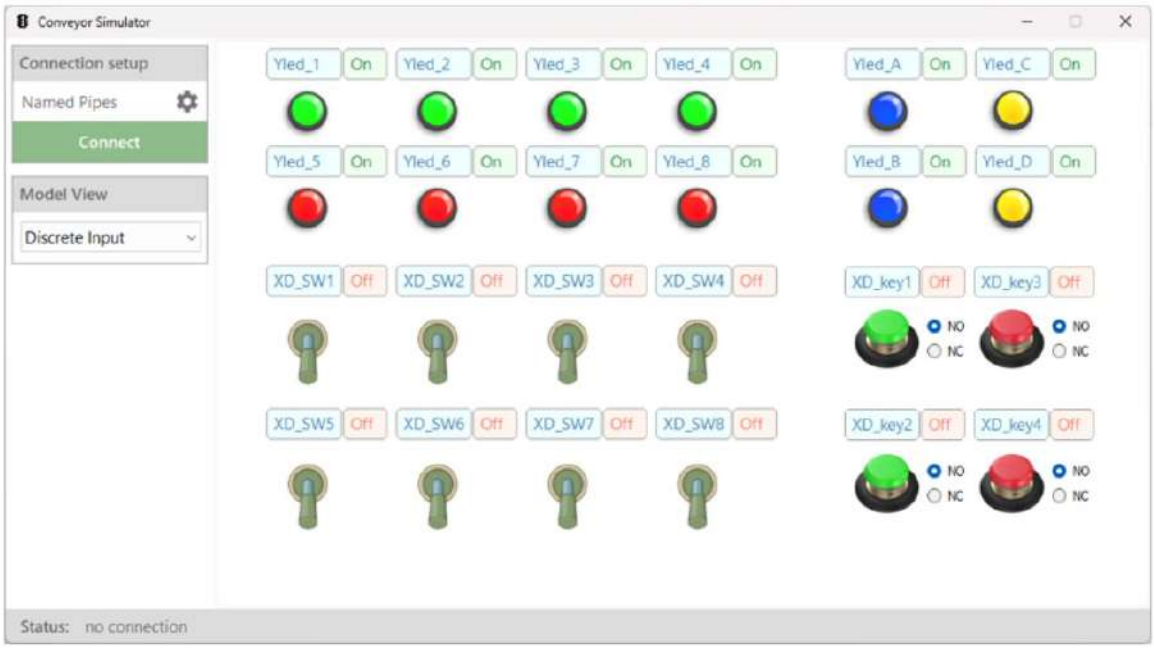


Рисунок 3.28 – Зовнішній вигляд віртуального макету «Модуль формування та відображення стану дискретних сигналів»

Робоче поле можна розбити на сім зон (рис. 3.29).

Перша робоча зона містить чотири зелених світлодіоди. За кожним з них закріплена відповідна змінна. В даному макеті всі назви світлодіодів починаються з єдиного префікса «Yled_», після якого йде порядковий номер приладу. Наприклад, «Yled_1», «Yled_2» і так далі, до четвертого.

Друга зона містить чотири червоних світлодіоди. Їм відповідають змінні від «Yled_5» до «Yled_8».

Відповідні інформаційні мітки на формі програми показують, в якому стані знаходиться на поточний момент кожен світлодіод. Самі світлодіоди теж змінюють колір. У вимкненому стані всі вони, не залежно від робочого кольору, мають сірий

вигляд. В режимі, коли на світлодіод приходить логічна одиниця, він запалюється робочим кольором.

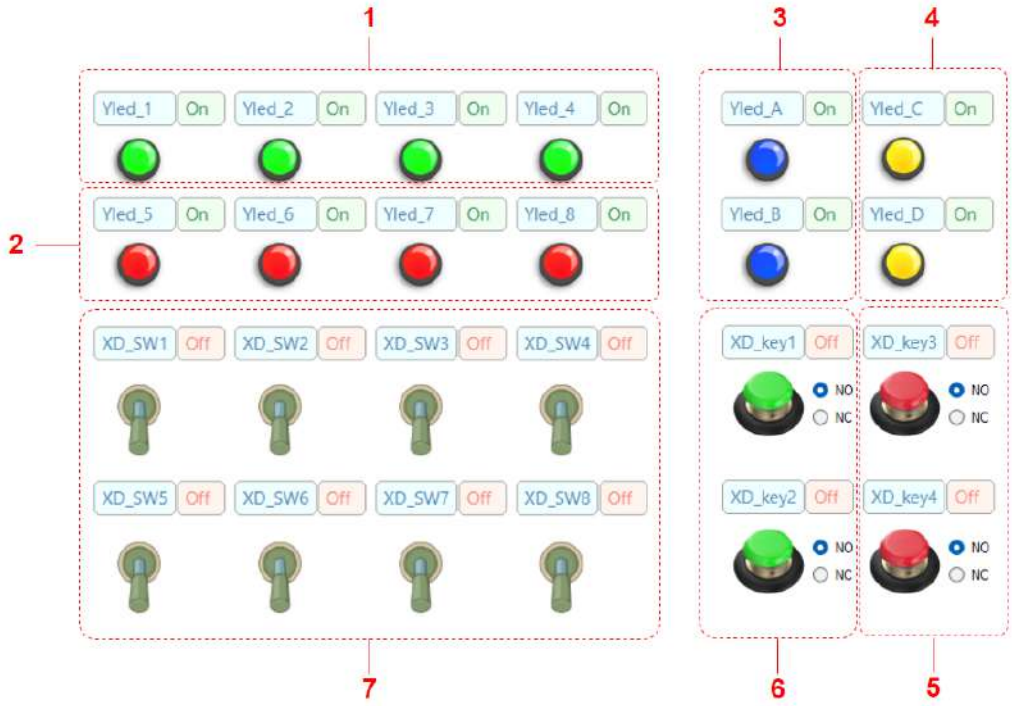


Рисунок 3.29 – Схема розбиття робочої зони віртуального макету

Приклад різних станів роботи світлодіодів подано на рис. 3.30.

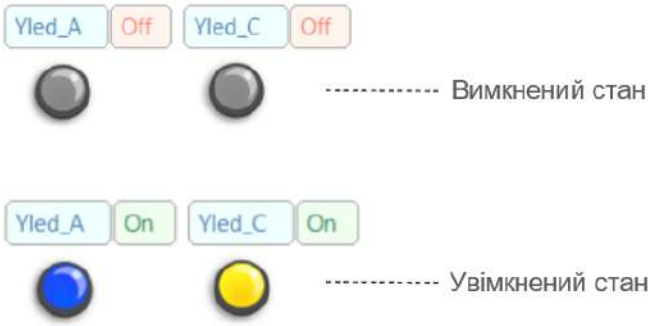


Рисунок 3.30 – Приклад різних станів роботи світлодіодів

Робоча зона 3 представлена двома блакитними світлодіодами з прив’язкою до змінних «Yled_A» та «Yled_B», четверта зона – жовтими світлодіодами з прив’язкою до змінних «Yled_C» та «Yled_D».

У віртуальному макеті передбачено наявність двох видів джерел дискретних сигналів. Перший вид – це звичайні кнопки з двома можливими типами контактів: нормально розімкненими (NO) та нормально замкненими (NC) контактами (робочі зони 5 та 6). В робочій зоні 5 знаходяться дві кнопки червоного кольору «XD_key3» та «XD_key4», в зоні 6 – зеленого («XD_key1» та «XD_key1»). Робота кнопок в зонах 5 та 6 аналогічна. Стан кожної кнопки показано в відповідній інформаційній мітці. «Off» відповідає стану, коли кнопка відпущена, а стан «On» – натиснута.

Біля кожної кнопки розташовані додаткові прапорці обирання типу контакту. Кожна кнопка в процесі використання макету може бути налаштована на роботу в режимі нормально відкритого (NO) або нормально замкненого (NC) контакту (рис. 3.31).

Рисунок 3.31 – Режими роботи кнопок

В залежності від обраного режиму роботи змінюється принцип використання даних елементів. На рис. 3.31 можна бачити, що у зелених кнопок в нормальному (відпущеному) стані на виході немає робочої напруги, а в червоних є. Відповідні інформаційні мітки відображають робочий стан кнопок. Натискання на кнопку приводить до зміни робочої напруги на виході. Після відпускання кнопки миші, кнопка повертається в нормальне положення.

Інший вид джерела дискретних сигналів – це тумблери (робоча зона 7). Такий вид перемикачів має два стійкі стани. Натискання на зображення тумблера

призводить до зміни його стану. В такому стані він залишається також після відпускання кнопки миші. На рис. 3.32 подано два різних стани тумблера (вимкнений та увімкнений).

а) б)

Рисунок 3.22 – Два стани тумблера:
а) тумблер вимкнений; б) тумблер увімкнений

Використання тумблерів є доцільним, коли необхідно одночасне виставлення та утримання потрібних сигналів на декількох дискретних входах ПЛК, в той час, коли кнопки використовуються для оперативної зміни стану на конкретному вході контролера.

3.5 Віртуальний прилад «Програмований логічний контролер. Модулі вводу-виводу дискретних сигналів»

Даний віртуальний прилад використовується для симуляції роботи навчального ПЛК NTech PLC206-D. Принцип роботи реального пристрою подано в розділі 2.2.3.

До входів ПЛК підключено модуль формування дискретних сигналів EDS-8, принцип дії якого подано у розділі 2.6.1. Даний модуль використовується для імітації подачі на вхід ПЛК потрібної комбінації дискретних сигналів.

Зовнішній вигляд віртуального лабораторного макету подано на рис. 3.33. Студенту пропонується схема підключення модуля формування дискретних сигналів до відповідних дискретних входів ПЛК.

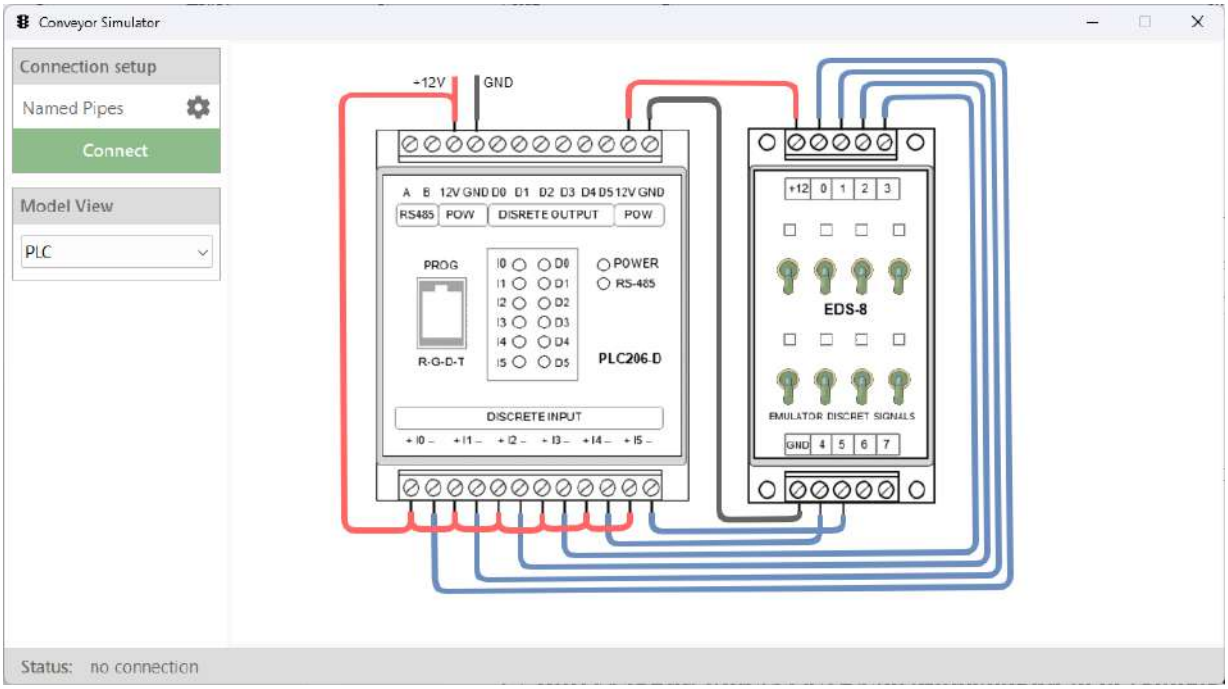


Рисунок 3.33 – Зовнішній вигляд віртуального лабораторного макету «Програмований логічний контролер»

В макеті реалізована схема включення вхідних контактів із загальним плюсом. Керуючий мінус надходить від модуля формування дискретних сигналів з натисканням на відповідний тумблер (рис. 3.34).

Вмикання будь-якого тумблера призводить до подачі на вхід ПЛК керуючого сигналу. Це підтверджується запалюванням відповідного світлодіоду, що відображає стан дискретного входу. Одночасно з цим вмикається світлодіод, який пов’язаний з тумблером на модулі формування дискретних сигналів, а сам тумблер переводиться в положення «Увімкнено».

На рис. 3.34 показано випадок, коли увімкнені два тумблери «0» та «1», які за допомогою з’єднувальних дротів підключені до дискретних входів «I0» та «I1» ПЛК.

В залежності від керуючої програми, яка працює в ПЛК, на вихідних клемах можуть з’являтись різні комбінації дискретних сигналів. Індикатори «D0» … «D5» відображають поточний стан вихідних контактів ПЛК.

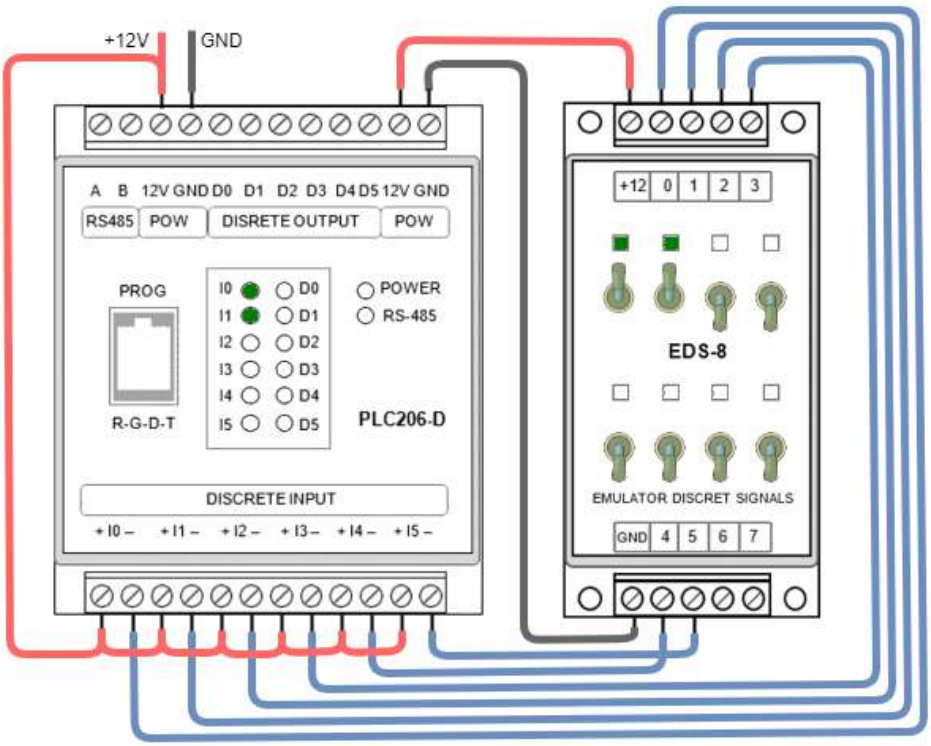


Рисунок 3.34 – Формування дискретних сигналів

3.6 Віртуальний прилад «Дозатор»

Зовнішній вигляд інтерфейсу користувача наведено на рис. 3.35.

Віртуальний прилад «Дозатор» розроблено для проведення моделювання процесу змішування речовин в певній пропорції та візуалізації поведінки виконавчих пристроїв в залежності від команд керуючої програми.

Виконавчими пристроями даного макету є:

– клапан подачі синьої рідини «YSPK_1»;

– клапан подачі червоної рідини «YSPK_2»;

– клапан зливу набраної дози рідини «YSPK_3».

Керування клапанами відбувається за допомого технологічної програми, що виконується в симуляторі ПЛК. Якщо подана логічна одиниця на відповідне реле, назва якого співпадає з однією з вказаних міток, то вмикається режим симуляції наповнення дозуючого бункеру.

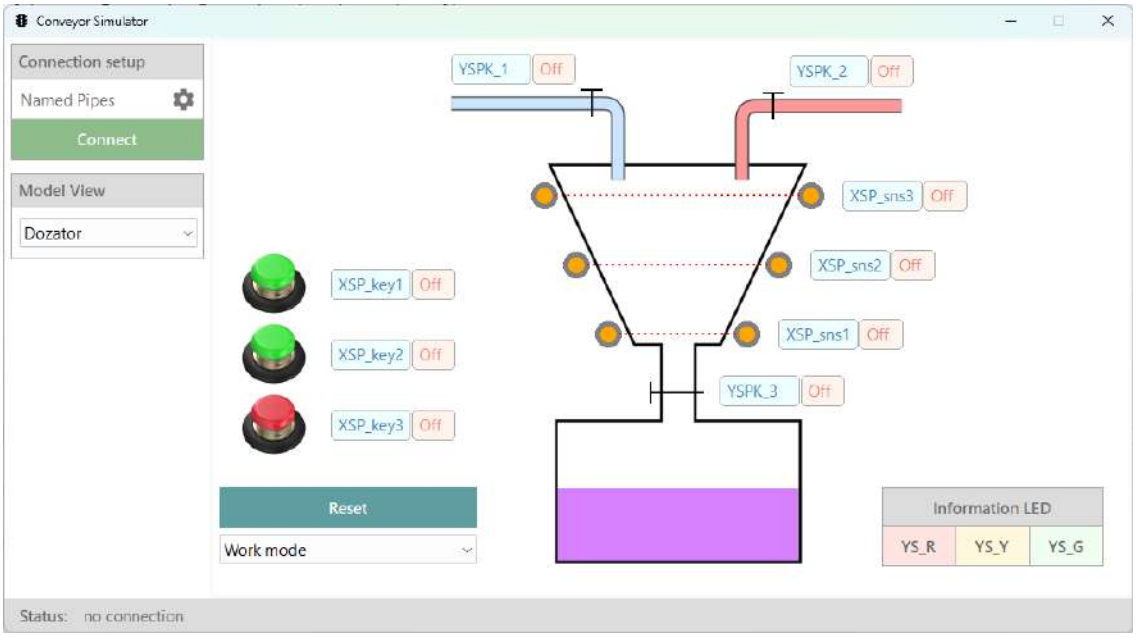


Рисунок 3.35 – Інтерфейс користувача віртуального приладу «Дозатор»

На рис. 3.36 подано приклад вмикання клапану «YSPK_2». Весь час, поки клапан відкрито, відбувається поступове наповнення дозуючого бункеру. Рівень рідини поступово збільшується.

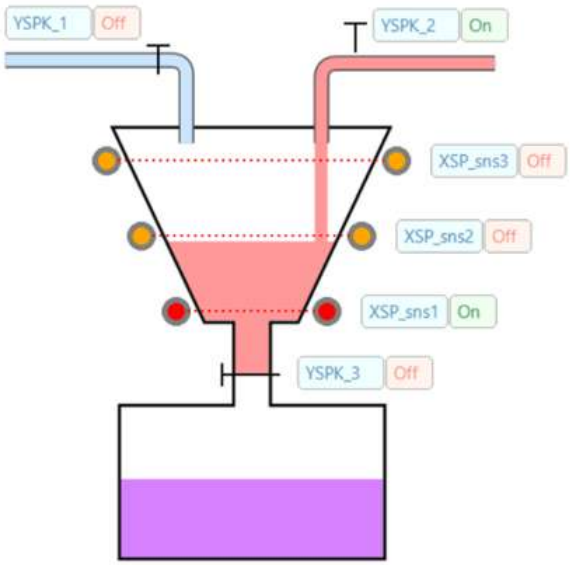


Рисунок 3.36 – Приклад вмикання клапана «YSPK_2»

В приладі передбачено наявність трьох датчиків рівня:

– нижній датчик «XSP_sns1»;

– середній датчик «XSP_sns2»;

– верхній датчик «XSP_sns3».

Коли рівень рідини досягає відповідного датчика, його стан змінюється з «Off» на «On» (з «Вимкнено» на «Увімкнено»). Інформація про поточний стан передається в керуючу програму через обраний інтерфейс взаємодії.

Якщо рівень рідини в бункері перевищить максимально допустимий, то це буде визначено, як аварійна ситуація та на екрані з’явиться вікно із зазначенням помилки (рис. 3.37).

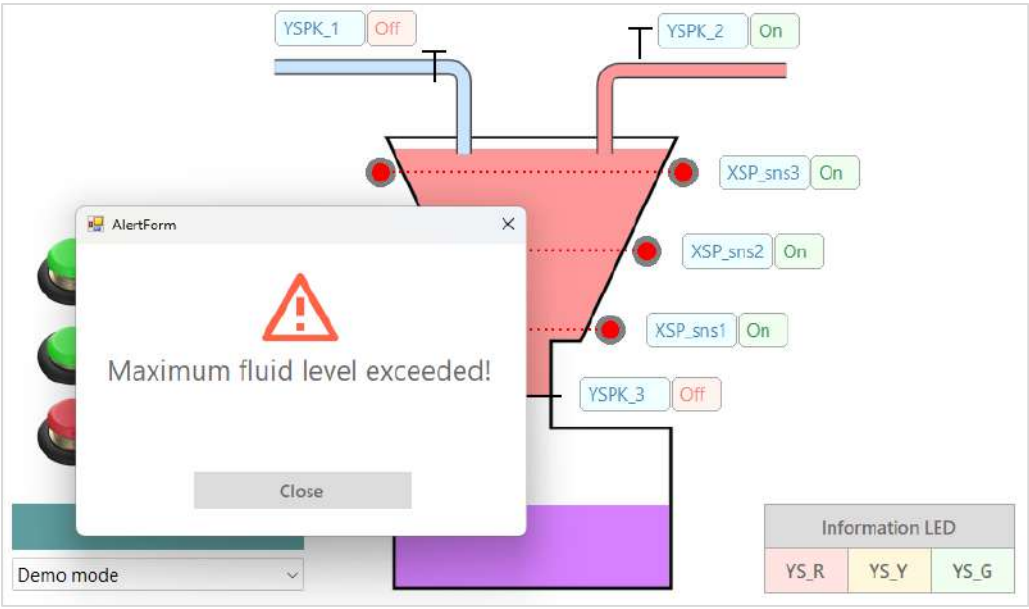


Рисунок 3.37 – Обробка аварійної ситуації

Натискання на кнопку «Close» призведе до скидання стану всіх елементів віртуального пристрою в початковий стан.

Для керування роботою пристрою передбачено три кнопки «XSP_key1», «XSP_key2» та «XSP_key3». В технологічній програмі кнопки можна використовувати за власним бажанням, певний алгоритм залежить від завдання, що вирішується.

Також передбачено режим демонстрації, в якому можна ознайомитись с принципом роботи приладу. В даному режимі кнопки виконують такі функції:

– кнопка 1 «XSP_key1» вмикає та вимикає клапан подачі синьої рідини «YSPK_1»;

– кнопка 2 «XSP_key2» керує клапаном подачі червоної рідини «YSPK_2»;

– кнопка 3 «XSP_key3» вмикає або вимикає клапан зливу рідини в резервуар «YSPK_3».

Інформаційний блок (рис. 3.38) може використовуватись для індикації стану віртуального приладу в різних режимах роботи.

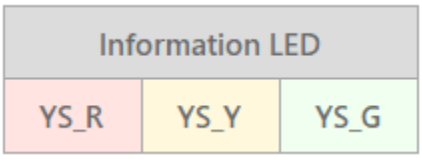


Рисунок 3.38 – Інформаційний блок з трьома світлодіодами

Інформаційний блок має три індикатори, які керуються відповідними вхідними сигналами:

– червоний світлодіод керується сигналом «YS_R»;

– жовтий світлодіод керується сигналом «YS_Y»;

– зелений світлодіод керується сигналом «YS_G».

3.7 Віртуальний макет «Аналого-цифровий перетворювач»

Даний віртуальний макет є комбінацією двох приладів:

– цифровий індикатор;

– аналого-цифровий перетворювач.

3.7.1 Віртуальний прилад «Семисегментний чотирьохрозрядний цифровий індикатор»

Інтерфейс користувача віртуального макету «Аналого-цифровий перетворювач» подано на рис. 3.39.

Верхня частина робочого вікна програми представляє собою окремий віртуальний прилад «Семисегментний чотирьохрозрядний цифровий індикатор» (рис. 3.39, поз. 1). Він може використовуватись як самостійно, так і в комплексі із аналого-цифровим перетворювачем.

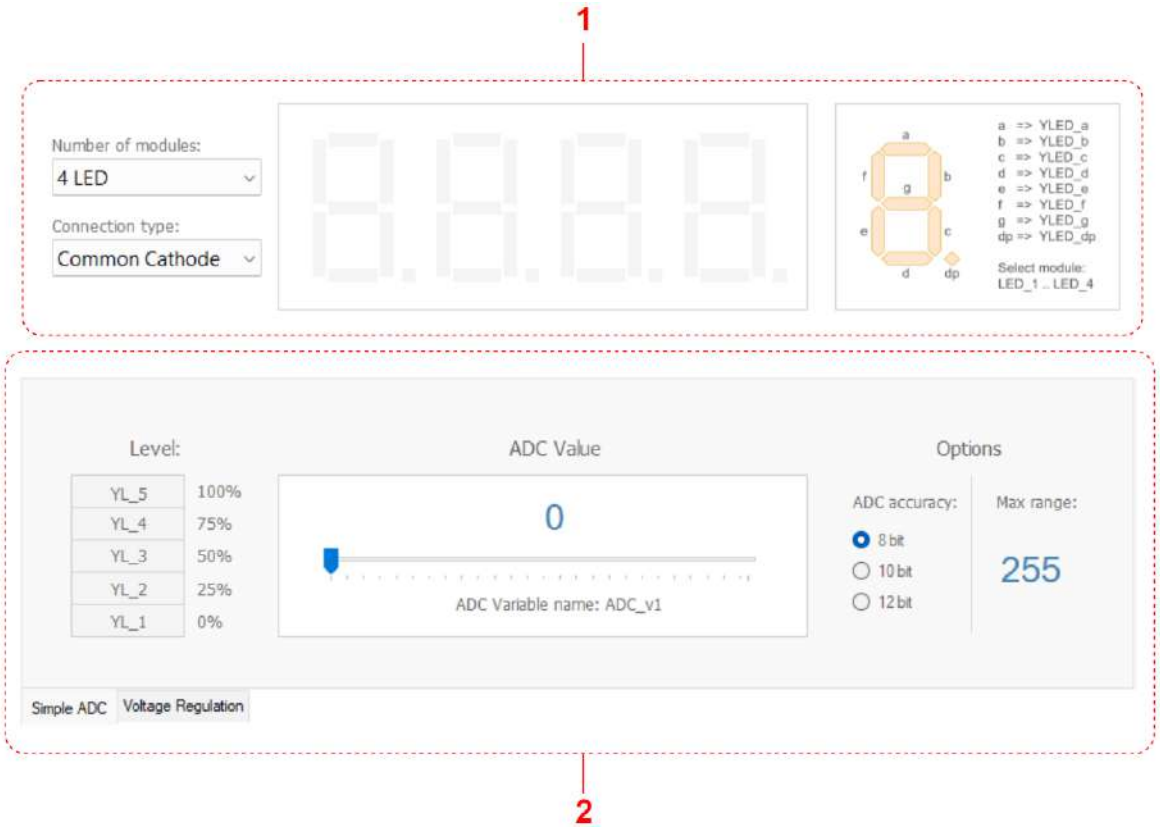


Рисунок 3.39 – Інтерфейс користувача віртуального макету «Аналого-цифровий перетворювач»

Нижня частина (рис. 3.39, поз. 2) має необхідні органи керування для виконання дослідження методів введення аналогових сигналів за допомогою ПЛК.

Віртуальний прилад «Семисегментний чотирьохрозрядний цифровий індикатор» призначений для відображення поточної інформації засобами ПЛК та дослідження методів організації динамічної індикації в процесі роботи з багаторозрядними цифровими індикаторами.

На рис. 3.40 подано інтерфейс віртуального приладу «Семисегментний чотирьохрозрядний цифровий індикатор». Основна візуальна частина приладу – це

модуль цифрового семисегментного індикатора (рис. 3.40, поз. 1). Якщо підключитися до макету за допомогою одного з доступних інтерфейсів, наприклад, за допомогою «Named Pipes», модуль цифрового індикатора відображатиме знаки що є комбінацією увімкнених сегментів.

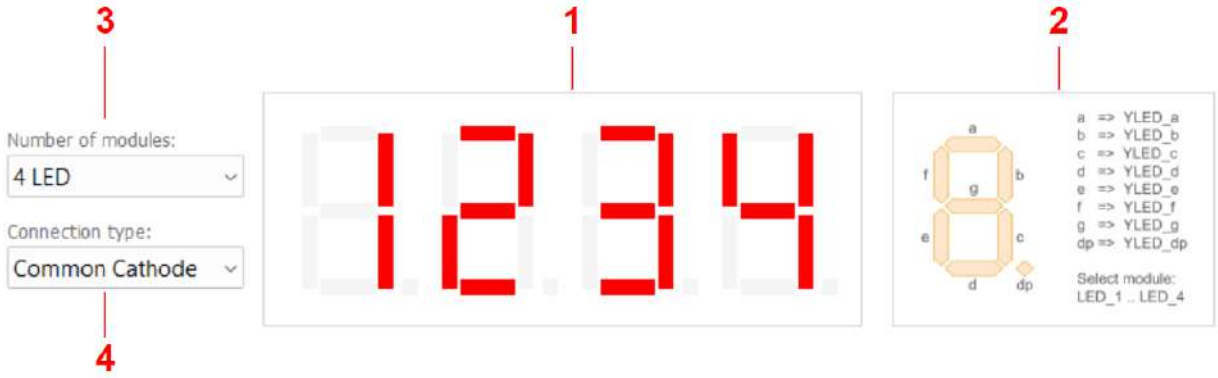


Рисунок 3.40 – Інтерфейс віртуального приладу «Семисегментний чотирьохрозрядний цифровий індикатор»

Керувати можна кожним сегментом незалежно. Для зручності створення керуючої програми в лівій частині інтерфейсу (рис. 3.40, поз. 2) подана стандартна схема розташування та найменування сегментів індикатора. Також там наведені назви віртуальних змінних, що закріплені за цими сегментами.

Наприклад, для відображення цифри «1» на індикаторі, необхідно включити два сегменти «b» та «c». Даним сегментам відповідають такі змінні:

– сегмент «b» керується змінною «YLED_b»;

– сегмент «c» керується змінною «YLED_c».

Окремі змінні «LED_1»…«LED_4» пов’язані з роздільними точками між розрядами індикатору.

Кількість розрядів індикатора можна змінювати в процесі роботи з віртуальним приладом в залежності від потреб, що вказані в завданні. Для цього передбачено елемент «Number of modules» (рис. 3.40, поз. 3).

На рис. 3.41 подано приклад обрання двох розрядів зі списку можливих варіантів.

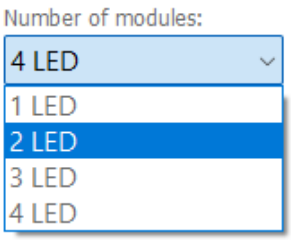


Рисунок 3.41 – Приклад обрання двох розрядів
зі списку можливих варіантів

На рис. 3.42 можна бачити результат зміни в інтерфейсі програми для відображення двох цифр.

Рисунок 3.42 – Відображення двох цифр на цифровому індикаторі

Ще однією можливостю є обирання типу підключення індикатора до ПЛК. Можливі варіанти:

– за схемою із загальним катодом;

– за схемою із загальним анодом.

На рис. 3.43 подано приклад обрання різних варіантів підключення.

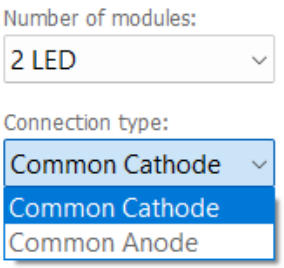


Рисунок 3.43 – Приклад обрання різних варіантів підключення
індикаторів до ПЛК

В залежності від обраного способу змінюються умови вмикання сегментів. В першому варіанті, сегменти запалюються, якщо надходить логічна одиниця, а в другому – логічний нуль.

3.7.2 Віртуальний прилад «Аналого-цифровий перетворювач»

Програмний інтерфейс віртуального приладу подано на рис. 3.44.

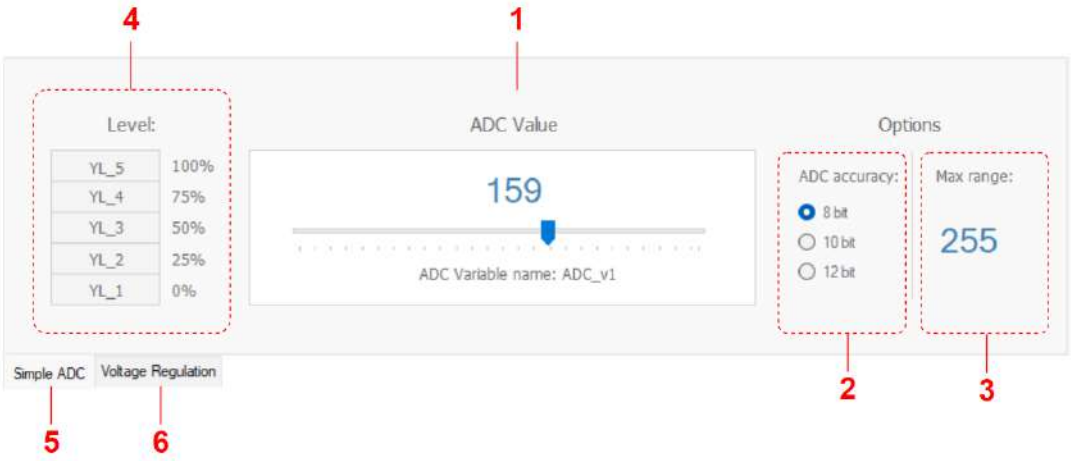


Рисунок 3.44 – Програмний інтерфейс віртуального приладу «Аналого-цифровий перетворювач»

В основному режимі даний прилад реалізує спрощений інтерфейс керування АЦП. За допомогою регулятору (рис. 3.44, поз. 1) можна змінювати значення на виході віртуального АЦП. Поточне значення відображається у вигляді числа над повзунком регулятора.

Діапазон регулятора змінюється в межах від 0 до максимального припустимого значення для обраної розрядності АЦП:

– АЦП 8 біт має максимальний діапазон значень 255;

– АЦП 10 біт має максимальний діапазон значень 1023;

– АЦП 12 біт має максимальний діапазон значень 4095.

Розрядність АЦП обирається за допомогою трьох прапорців (рис. 3.44, поз. 2), при цьому, в залежності від обраної розрядності змінюється число «Max Range» (рис. 3.44, поз. 3).

Для доступу до значення АЦП з боку ПЛК передбачена змінна «ADC_v1». Саме таку назву повинна мати змінна в програмі LD_micro для отримання поточного числа.

В інтерфейсі програми є візуальний інтерактивний елемент (рис. 3.44, поз. 4) для відображення рівня будь-якої величини, яку необхідно надати користувачу в процесі роботи програми.

Наприклад, за допомогою даного елементу можна вивести на екран графічне представлення рівня рідини в резервуарі, або рівня напруги на виході акумулятора.

Приклад використання елементу «Level» показано на рис. 3.45.

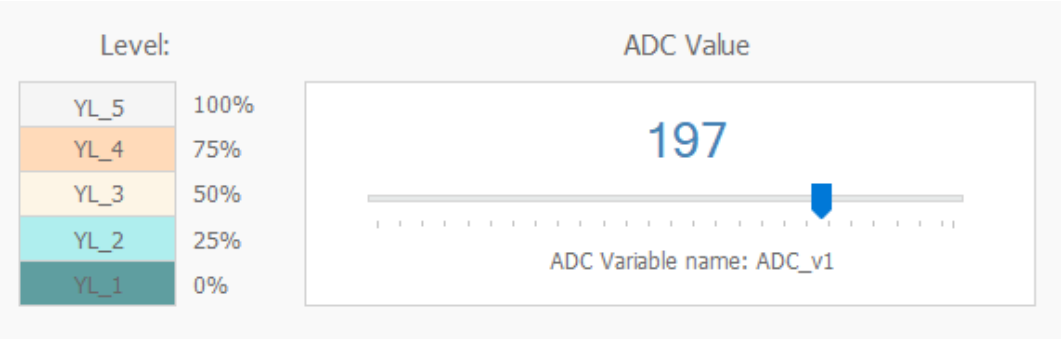


Рисунок 3.45 – Приклад використання елементу «Level»

Сегменти елемента керуються окремо та кожен з них має власну змінну в діапазоні від 0 до 100 відсотків. Відповідні змінні мають назви: «YL_1»…«YL_5».

Для увімкнення сегменту необхідно подати на нього логічну одиницю. В даному випадку він змінить свій колір з нейтрального на той, який закріплений за ним (рис. 3.45).

За допомогою кнопок 5 та 6 (рис. 3.44) можна перемикати режим роботи віртуального приладу:

– простий АЦП (рис. 3.44, поз. 5);

– регулятор напруги (рис. 3.44, поз. 6).

Зовнішній вигляд приладу в режимі регулятора напруги подано на рис. 3.46.

Даний режим призначений для дослідження методів перетворення вхідної інформації від АЦП в реальне значення вимірюваної величини.

В програмі передбачена можливість зміни величини опорної напруги за допомогою відповідних прапорців (3 або 5 В) (рис. 3.46, поз. 1).

У випадку, коли вхідна напруга перевищує опорну, найпростіша схема вимірювання за допомогою АЦП будується на основі резистивного дільника.

В інтерфейсі віртуального приладу є відповідний блок елементів (рис. 3.46, поз. 2) для введення значень резисторів дільника.

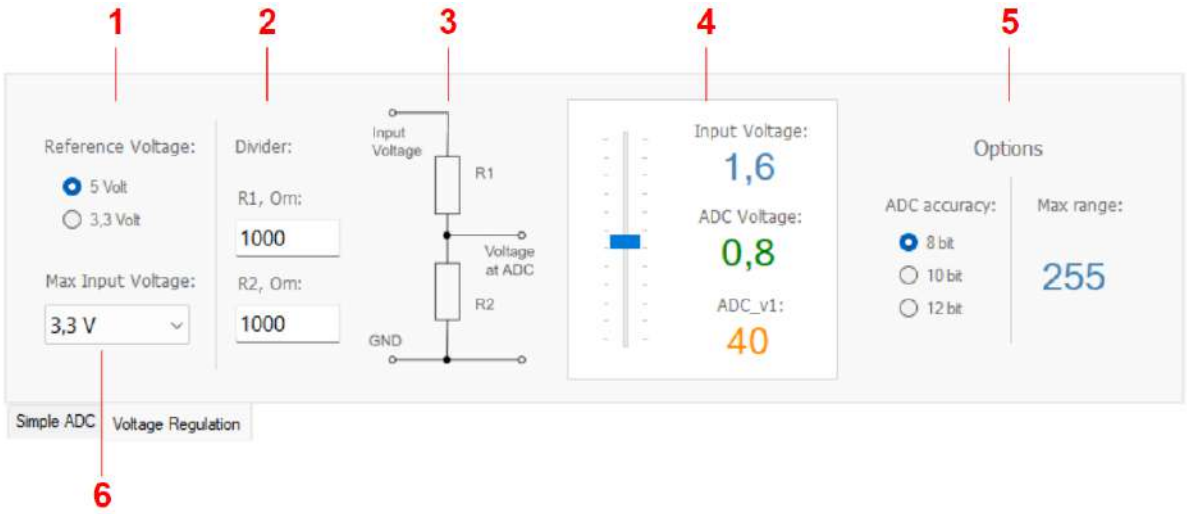


Рисунок 3.46 – Зовнішній вигляд приладу в режимі регулятора напруги

Схема дільника з позначенням резисторів наведена в блоці 3 інтерфейсу програми (рис. 3.46). Інтерфейс регулятора напруги подано на рис. 3.47.

Зміна вхідної напруги в обраному діапазоні відбувається за допомогою слайдеру (рис. 3.47, поз. 1).

В залежності від положення регулятора змінюється:

– значення вхідної напруги (рис. 3.47, поз. 2);

– орієнтовне значення напруги, що обчислюється на основі даних від АЦП (рис. 3.47, поз. 3);

– значення змінної «ADC_v1», що відповідає поточному вимірюваною АЦП (рис. 3.47, поз. 4).

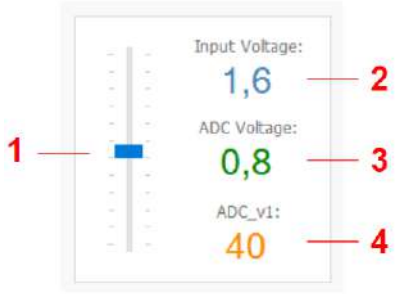


Рисунок 3.47 – Інтерфейс регулятора напруги

Обирати розрядність АЦП можна за допомогою блоку 5 інтерфейсу програми (рис. 3.46). Для обрання максимального значення вхідної напруги використовується список 6, що випадає. Можливі значення лежать в діапазоні: «3.3 В, 5.0 В, 9.0 В, 10.0 В, 12.0 В, 15.0 В, 24.0 В, 30.0 В, 36.0 В». Принцип розрахунку напруги за визначеним значенням АЦП подано в розділі 6.2.5.

3.8 Контрольні питання

1. Який вигляд має інтерфейс користувача віртуального приладу «Світлова колона»?
2. Яким чином реалізується зв’язок між віртуальним приладом і зовнішньою програмою або іншим пристроєм?
3. Що таке візуальна мітка?
4. Як виконати підключення віртуального приладу до постачальника інформації?
5. Як реалізується підключення LDmicro до віртуального приладу?
6. Які основні елементи має віртуальний прилад «Штампувальний автомат»? Поясніть принцип його функціонування.
7. Поясніть принцип взаємодії віртуального приладу і програмного симулятора ПЛК.
8. Які функції може виконувати віртуальний прилад «Конвеєр та технологічна лінія»? Які датчики передбачено в даному приладі?
9. Для чого використовується віртуальний прилад «Модуль формування та відображення стану дискретних сигналів»?
10. Яке призначення віртуального приладу «Програмований логічний контролер. Модулі вводу-виводу дискретних сигналів»?
11. Поясніть принцип роботи віртуального приладу «Дозатор».
12. Особливості роботи з віртуальним макетом «Аналого-цифровий перетворювач».

4 ПОБУДОВА ЛОГІЧНИХ ЕЛЕМЕНТІВ ЗАСОБАМИ РЕЛЕЙНО-КОНТАКТНОЇ ЛОГІКИ

Мова релейно-контактної логіки (Ladder diagram, LD) – мова релейної (східчастої) логіки, призначена для програмування промислових контролерів. Синтаксис мови є зручним і забезпечує наочний інтерфейс логіки роботи контролера, який полегшує не лише задачі програмування і введення в експлуатацію, але й швидкий пошук неполадок у підключеному до контролера обладнанні. Програма керування, написана мовою релейної логіки має наочний та інтуїтивно зрозумілий графічний інтерфейс, що подає логічні операції, як електричні кола із замкненими та розімкненими контактами. Проходження або відсутність струму у цьому колі відповідає результату логічної операції («істина», якщо струм проходить і «хибність», якщо струм не проходить).

4.1 Вхідні та вихідні контакти LD

4.1.1 Вхідні контакти

Сходинки розташовані між двома електричними шинами (рис. 4.1).

Рисунок 4.1 – Східчаста діаграма та мова LD

Зазвичай сходинка починається з контакту. Він є входом для логіки управління, а реле (котушка) – виходом (рис. 4.2).

Умовою вмикання реле є проходження електричного струму всіма контактами сходинки зліва направо до самого реле. Певна комбінація розташування контактів і котушок в схемі формує певні логічні комбінації (ключі).

Рисунок 4.2 – Приклад простої LD-діаграми

Контакти можуть отримувати вхідні сигнали як від внутрішніх, так і від зовнішніх джерел. У випадку підключення контактів до зовнішніх джерел його назва починається з літери «X» (e**X**ternal input). У випадку отримання сигналів від внутрішніх джерел перша літера буде «R» (inte**R**nal input) (рис. 4.3).

Рисунок 4.3 – Зовнішні та внутрішні джерела сигналів

Налаштування типу вхідних сигналів виконується за допомогою відповідного діалогу користувача (рис. 4.4).

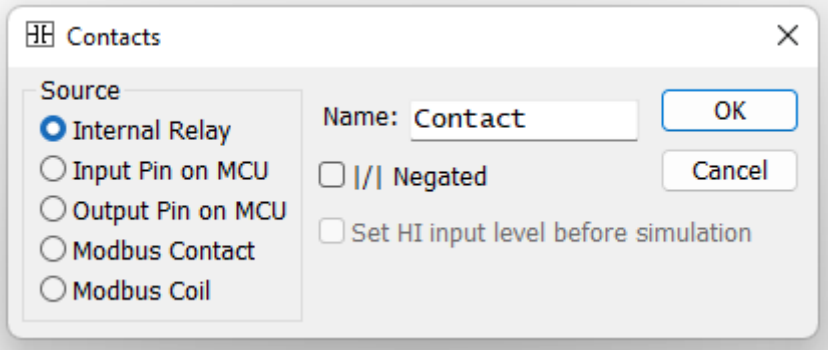


Рисунок 4.4 – Налаштування типу вхідних сигналів

Крім внутрішнього реле і вхідного (зовнішнього) контакту мікроконтролеру, в якості джерела вхідного сигналу можуть бути:

– вихідній контакт мікроконтролеру (для перевірки стану вихідних контактів ПЛК);

– контакт Modbus;

– котушка Modbus.

Контакт і котушка Modbus використовуються в разі застосування розподіленої архітектури поєднання технічних засобів автоматизації з використанням модулів розширення входів-виходів за допомогою промислової мережі.

Контакт (Contact) – це програмний тип кнопкового вимикача. Контакти можуть бути нормально розімкнені (NO) та нормально замкнені (NC). Кнопка з контактами «NO» має контакти, що замикаються при натисканні. З відпусканням кнопки контакти знов розмикаються. Кнопка «NC» має контакти, що розмикаються при натисканні. Через такі кнопки струм протікає доки кнопка не буде натиснута (розімкнена). На рис. 4.5 подано приклад таких кнопок.

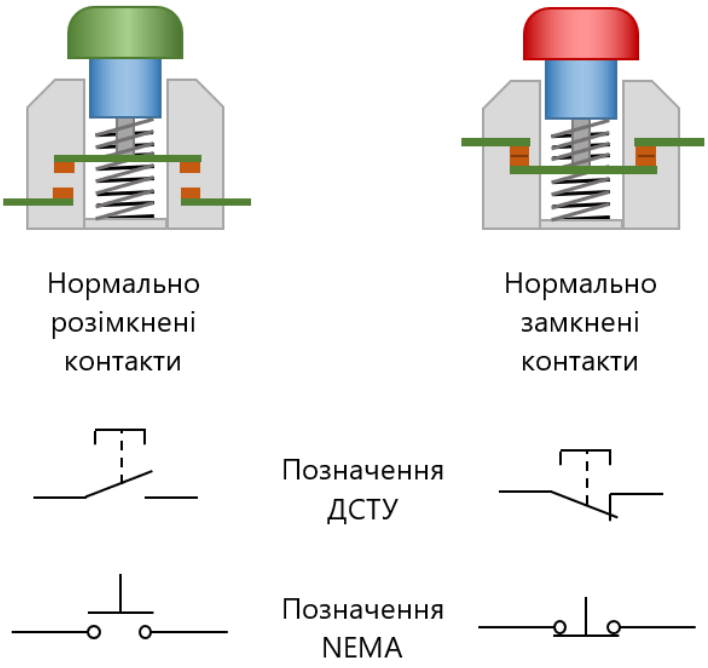
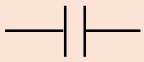
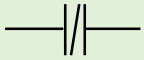
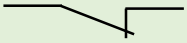
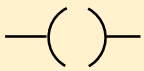
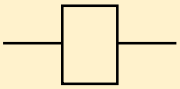


Рисунок 4.5 – Приклади варіантів виконання кнопок

На LD-діаграмі контакти та реле зображені за стандартами промислової автоматики. В таблиці 4.1 наведено приклад зображення релейних елементів на LD-діаграмі та ЄСКД.

Таблиця 4.1 – Приклад зображення релейних елементів на LD-діаграмі та ЄСКД

Назва компонента	**Зображення**	
	LD	**ЄСКД**
Контакт, що замикається		
Контакт, що розмикається		
Котушка реле		

Для того, щоб застосовувати кнопку із нормально замкненими контактами (NC) на діаграмі LD в програмі LDmicro необхідно виставити позначку «|\| Negated» у вікні налаштування вхідних сигналів (рис. 4.4). Приклад використання нормально відкритих та нормально замкнених контактів показано на рис. 4.6.

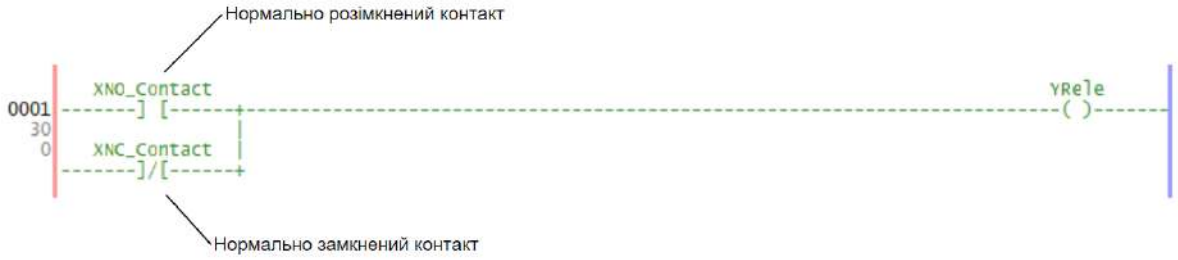
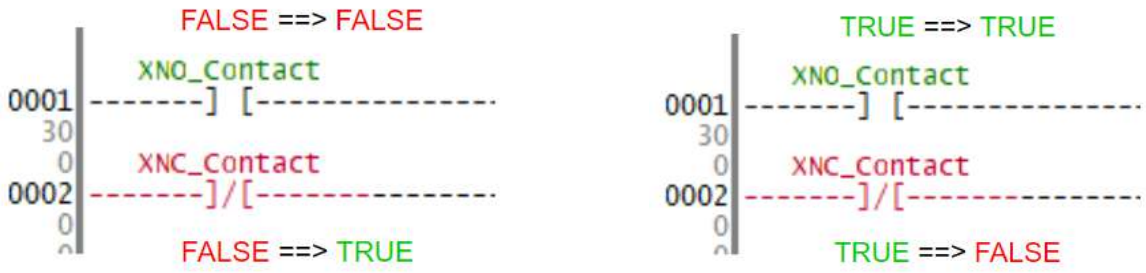


Рисунок 4.6 – Приклад використання нормально розімкнених та нормально замкнених контактів

Інструкції з різним типом контактів ведуть себе по різному в програмі LDmicro (рис. 4.7).



а) б)

Рисунок 4.7 – Поведінка інструкцій з різними типами контактів

Коли від джерела живлення до нормально розімкненого контакту надходить сигнал, то на його виході буде логічний нуль, якщо кнопка не натиснута (рис. 4.7, а). Коли на вхід надходить напруга і кнопка при цьому замкнена, то на виході з’явиться сигнал високого рівня (рис. 4.7, б).

Нормально замкнені кнопки поводяться по іншому. Коли на вході сигнал логічної одиниці, а кнопка не натиснута (знаходиться в стані false), то на виході буде сигнал логічної одиниці (рис. 4.7, а). З натисканням кнопки (кнопка спрацьовує та знаходиться в режимі true) на виході буде сигнал логічного нуля (false) (рис. 4.7, б).

На рис. 4.8 подано два режими роботи вхідних контактів. На рис. 4.8, а показано випадок, коли дві кнопки знаходяться в ненатиснутому стані. На рис. 4.8, б показано випадок, коли кнопки XNO_Contact та XNC_Contact натиснуті.

4.1.2 Вихідний елемент (котушка)

Котушка є завершальним елементом в електричному ланцюзі та найчастіше напряму поєднується з вихідними контактами ПЛК. До вихідних контактів підключаються зовнішні виконуючі пристрої, наприклад, світлодіоди, двигуни постійного струму тощо. Також до вихідних контактів можуть підключатися внутрішні реле, які можуть використовуватись в програмі як вхідні контакти.

Так само, як і для вхідних, для вихідних контактів (Rele) можна виконати налаштування варіантів вмикання.

а)

б)

Рисунок 4.8 – Два режими роботи вхідних контактів

Реле можуть бути:

– з нормально розімкненими контактами (NO або Normal);

– з нормально замкненими контактами (NC або Negated);

– котушка із пам’яттю для встановлення вихідного сигналу в стан логічної одиниці (S);

– котушка із пам’яттю для скидання вихідного сигналу і переведення його в стан логічного нуля (R);

– котушка, що працює в режимі Т-тригера, який перемикає вихідний сигнал з кожним надходженням логічної одиниці на її вхід (T).

Т-тригер часто використовується для виявлення переднього фронту вхідного логічного сигналу. Вихід такого тригеру буде встановлено в true, коли на вхід прийде зростаючий фронт сигналу. У випадку, коли після скидання вхідного сигналу він знов почне зростати, на виході з’явиться сигнал логічного нуля, або false.

Приклад вікна налаштування варіантів вмикання котушки подано на рис. 4.9.

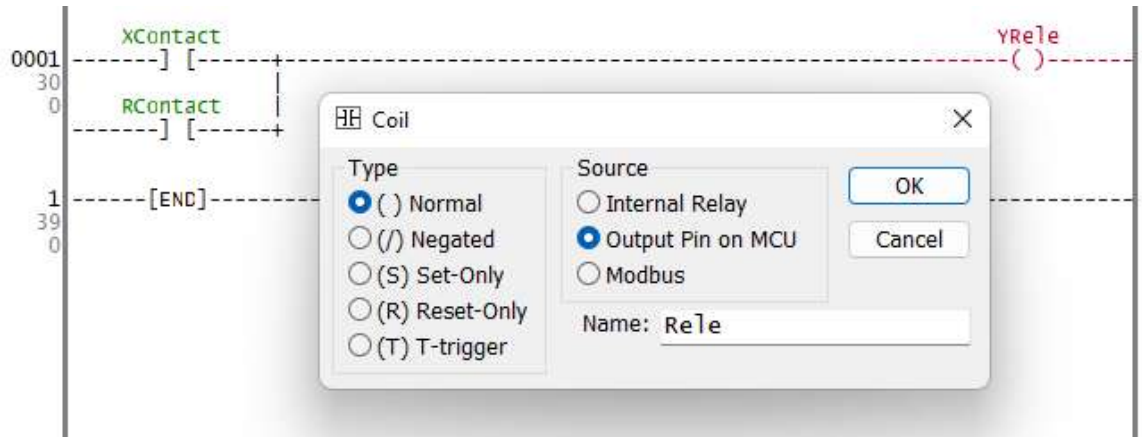


Рисунок 4.9 – Приклад вікна налаштування варіантів вмикання котушки

4.2 Основи бінарної логіки

4.2.1 Основні поняття алгебри логіки

Алгебра логіки, розроблена в середині IX ст. ірландським математиком Д. Булем і є науковою основою роботи всіх цифрових пристроїв. У ній діють принципи, схожі зі звичайною алгеброю, але буквами (символами) позначають не числа, а висловлювання. В алгебрі Буля змінні приймають тільки два дискретних значення: логічна «1» приписується істинному вислову і логічний «0» – хибним (неістинним). Символи не можна розглядати як арифметичні числа, тобто алгебра логіки є алгеброю станів, а не чисел. Апарат алгебри логіки використовують як для аналізу, так і для проектування (синтезу) логічних пристроїв будь-якої складності в системах цифрової обробки інформації. В цьому випадку можна проводити всі дослідження виключно математично.

Аналіз логічних пристроїв проводять, розглядаючи вхідні сигнали $x_1, x_2, \dots x_n$ в якості аргументів і представляючи відповідні вихідні сигнали логічного пристрою (ЛП) у вигляді функції y_i , як показано на рис. 4.10.

В цьому випадку аналітичне співвідношення

$$y_i = f_i(x_1, x_2, \dots x_n) \tag{4.1}$$

встановлює в явному вигляді відповідність між значенням функції і різними значеннями комбінацій аргументів. Для n бінарних (які можуть приймати тільки два значення) аргументів можливе число комбінацій «0» і «1» буде таким:

$$i_{[n]} = 2^n. \tag{4.2}$$

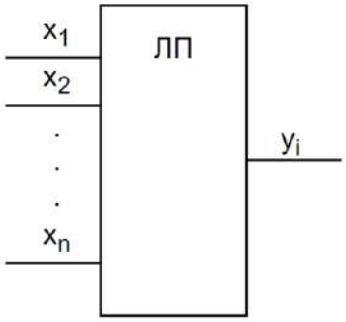


Рисунок 4.10 – Графічне позначення логічного пристрою

Наприклад, якщо $n = 2$ маємо чотири наступні комбінації з двох елементів: 00; 01; 10; 11. Якщо $n = 3$ маємо вісім комбінацій з трьох елементів: 000; 001; 010; 100; 011; 101; 110; 111.

На практиці для спрощення процедури аналізу складних ЛП їх подають у вигляді комбінації найпростіших логічних елементів (ЛЕ) (рис. 4.11) за аналогією з елементарними ланками в структурних схемах автоматики.

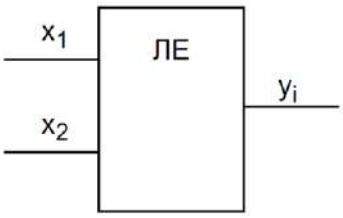


Рисунок 4.11 – Логічний елемент з двома вхідними сигналами x_1 і x_2 і одним виходом y_i

4.2.2 Основні логічні функції

Як зазначалося вище, логічні вирази можуть бути складними, тобто, як і раніше приймаючи лише два стани, являти собою комбінацію більш простих

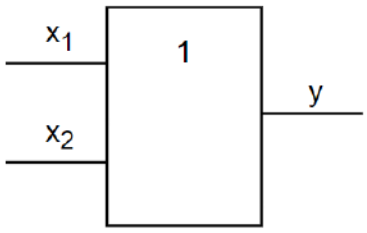
логічних виразів. Складні логічні вирази формуються з простіших за допомогою логічних зв'язків, які також називаються логічними операціями. Розглянемо приклади найбільш поширених логічних функцій.

1. Логічне додавання або диз'юнкція позначається символом ∨, називається також операцією «АБО». Ця операція описується для найпростішої функції двох змінних x_1 і x_2 у вигляді логічної формули:

$$y = x_1 \vee x_2 \ . \tag{4.3}$$

Співвідношення (4.3) означає, що функція y дорівнює «1», якщо хоча б один з аргументів (x_1 або x_2) дорівнює «1».

На рис. 4.12 подано умовне позначення логічного елементу АБО та його таблиця істинності.



Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Рисунок 4.12 – Логічний елемент АБО

Зауважимо, що таблицею істинності називають функціональний взаємозв'язок значень вихідної величини y_i – логічного пристрою кожної з можливих i-х комбінацій вхідних змінних $x_1, x_2 \ldots x_n$, представлених в табличній формі. Як зазначалося, для функції n змінних таких комбінацій буде $Б_n = 2^n$. У найпростішому випадку двох змінних x_1 і x_2 таблиця істинності буде налічувати $Б_n = 2^2 = 4$ комбінації цих змінних.

Аналізуючи рис. 4.12 можна відзначити, що стовпець y відповідає операції логічного додавання. Найбільш просто цю операцію можна реалізувати за

допомогою контактного ланцюга з двома паралельно увімкненими контактами. Сигнал y на виході такого ланцюга з'явиться тільки в тому випадку, якщо хоча б один з контактів буде замкнений. На рис. 4.13 показана його еквівалентна контактна схема.

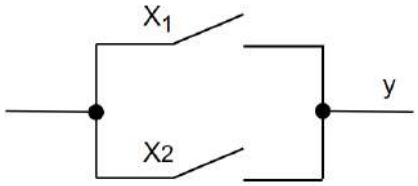


Рисунок 4.13 – Контактна схема логічного елемента АБО

У цифровій електроніці операцію логічного додавання легко реалізувати за допомогою двох діодів (з незалежними входами), працюючих на один навантажувальний пристрій з опором R. Принципова схема такого електронного ланцюга представлена на рис. 4.14.

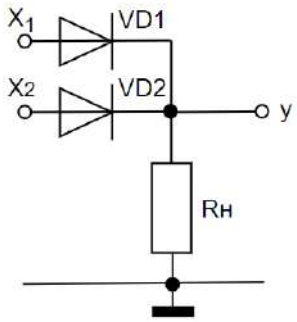
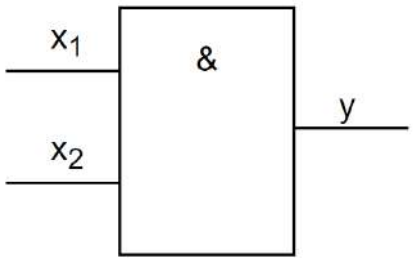


Рисунок 4.14 – Принципова схема логічного елементу АБО

Зі схеми видно, що сигнал на виході ланцюга, який відповідає логічній «1», має місце тільки в тому випадку, якщо на вході хоча б одного з діодів також існує сигнал, відповідний логічній «1». Цей сигнал відкриває діод, через що в навантажувальному пристрої з'являється струм, який забезпечує вихідну напругу ланцюга, що відповідає логічній одиниці.

Логічне множення або кон'юнкція, що позначається символом ∧ називається операцією «І». Умовне позначення кон'юнкції на логічних схемах називають

амперсандом («&»). На рис. 4.15 наведено зображення логічного елементу та таблиця істинності.



Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Рисунок 4.15 – Логічний елемент І

Для зручності запису складних логічних функцій символ кон'юнкції можна умовно ототожнювати зі знаком звичайного множення. Для функції двох змінних в цьому випадку маємо

$$y = x_1 \wedge x_2 = x_1 \cdot x_2 \tag{4.4}$$

Співвідношення (4.4) показує, що y = 1 тільки в тому випадку, коли обидва аргументи (x_1 і x_2) дорівнюватимуть логічній одиниці.

Операція логічного множення може бути реалізована контактним ланцюгом, принципова схема якої подана на рис. 4.16.

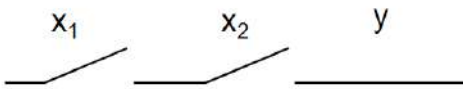


Рисунок 4.16 – Контактна схема логічного елемента І

Принципова схема електричного з'єднання, дія якої аналогічна контактній, показана на рис. 4.17.

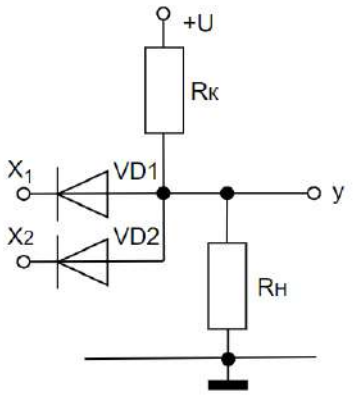


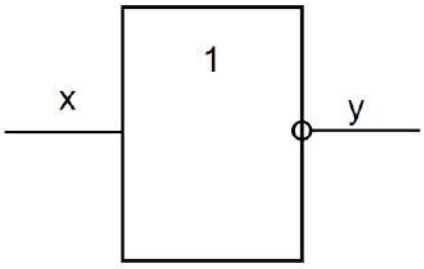
Рисунок 4.17 – Принципова схема електричного з'єднання
логічного елемента І

Сигнал на виході схеми, що дорівнює приблизно $+U$ та відповідає $y = 1$, можна отримати тільки в тому випадку, якщо обидва діоди ввімкнені, тобто до їх катодів поданий високий потенціал. Такий стан відповідає вхідним сигналам $x_1 = 1$ і $x_2 = 1$.

Логічне заперечення або інверсія позначається рискою над зміною і називається операцією НІ. Ця операція записується так:

$$y = \overline{x}. \tag{4.5}$$

Операція НІ виконується над однією змінною x та значення y завжди протилежно значенню цієї змінної. Умовне позначення логічного елементу НІ показано на рис. 4.18.



Таблиця істинності	
Вхід	Вихід
x_1	y
0	1
1	0

Рисунок 4.18 – Умовне позначення логічного елементу НІ

Реалізація логічної операції НІ може бути також здійснена контактною схемою, але, на відміну від схем, розглянутих раніше, за допомогою нормально замкнених контактів електромагнітного реле (рис. 4.19).

Відсутність напруги на обмотці реле ($x = 0$) передбачає замикання ланцюга і появи сигналу на його виході, що відповідає логічній одиниці ($y = 1$). При наявності напруги (логічної одиниці) на обмотці реле ($x = 1$) ланцюг розмикається та сигнал на виході схеми відсутній ($y = 0$).

Рисунок 4.19 – Контактна схема логічного елементу НІ

Логічна операція інверсії порівняно легко реалізується в електроніці ланцюгом найпростішого підсилювача з підключенням транзистора в схему із загальним емітером, яка характеризується інверсною властивістю. Принципова схема електричного з'єднання, дія якої відповідає логічному елементу НІ, подана на рис. 4.20.

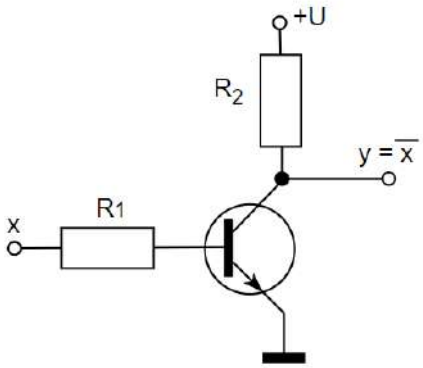


Рисунок 4.20 – Принципова схема електричного з'єднання,
дія якої відповідає логічному елементу НІ

Якщо транзистор працює в режимі насичення (з вхідною напругою, яка відповідає логічній одиниці), вихідний сигнал $y = U_{KE} = 0$. U_{KE} – напруга між колектором і емітером транзистора. Якщо вхідний сигнал відсутній ($x = 0$),

транзистор вимкнений, сигнал на виході $y = U_{KE} = +U$, що відповідає логічній одиниці.

Зіставляючи таблиці істинності для операцій диз'юнкції і кон'юнкції можна обґрунтувати наступні співвідношення алгебри Буля, які мають велике практичне значення.

Принцип дуальності, який зручно виразити у вигляді двох положень:

$$\text{якщо } x_1 \vee x_2 = y, \text{ то } \overline{x}_1 \vee \overline{x}_2 = \overline{y}, \tag{4.6}$$

$$\text{якщо } x_1 \wedge x_2 = y, \text{ то } \overline{x}_1 \wedge \overline{x}_2 = \overline{y}. \tag{4.7}$$

Правило де Моргана випливає як наслідок принципу дуальності і формулюється у вигляді двох логічних співвідношень:

$$\begin{gathered} \overline{x_1 \vee x_2} = x_1 \wedge x_2 = \overline{x}_1 \cdot \overline{x}_2; \\ \overline{x_1 \cdot x_2} = \overline{x_1 \wedge x_2} = \overline{x}_1 \vee \overline{x}_2. \end{gathered} \tag{4.8}$$

Наведені співвідношення (4.6) – (4.8) можна легко узагальнити для n вхідних сигналів $x_1, x_2 \ldots x_n$. Їх широко використовують для перетворення складних логічних функцій до простішого вигляду (мінімізації функцій) в процесі проєктування (синтезу) логічних пристроїв цифрової електроніки.

Розглянуті вище основні логічні операції І, АБО і НІ утворюють функціонально повний набір, тобто дозволяють реалізувати будь-які логічні функції (перетворення) комбінаційної логіки. Для цього можна навіть обмежитися двома операціями, наприклад АБО і НІ. Однак з використанням тільки цих трьох елементів не завжди вдається отримати логічні пристрої найпростішого виду. Тому в логічних системах знаходять застосування інші типові елементи, які реалізують інші логічні операції.

Особливе значення в цифровий мікроелектроніці приділяється двом універсальним логічним операціям, кожна з яких здатна самостійно утворити функціонально повний набір. Як відомо, в разі застосування тільки одного базового

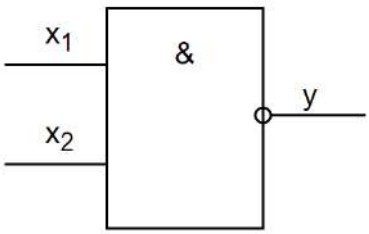
елемента спостерігається помітне ускладнення проектованих логічних пристроїв. Однак в інтегральній технології зручність виготовлення одного базового елементу має вирішальне значення. Тому універсальні логічні елементи складають основу більшості інтегральних цифрових мікросхем. Універсальні логічні операції, які реалізуються базовими елементами, містять два наступні різновиди: функція Шеффера та функція Пірса.

Функція Шеффера, яка позначається символічно вертикальною рискою «|» (штрих Шеффера), відображає операцію І-НІ. Для найпростішої функції двох змінних x_1 і x_2 в цьому випадку отримують:

$$x_1|x_2 = \overline{x_1 \cdot x_2} = \overline{x_1 \wedge x_2} = y_{\text{ш}}. \qquad (4.9)$$

Формула (4.9) вказує на те, що функція $y_{\text{ш}} = 0$ тоді і тільки тоді, коли $x_1 = x_2 = 1$.

На рис. 4.21 наведено зображення логічного елементу І-НІ та таблиця істинності.



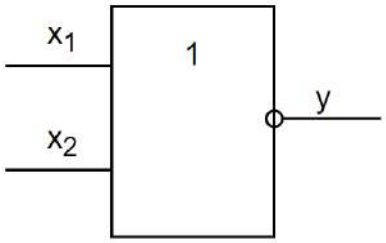
Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

Рисунок 4.21 – Логічний елемент І-НІ

Функція Пірса, що позначається символічно вертикальною стрілкою (стрілка Пірса), позначає операцію АБО-НІ. Для функції двох змінних x_1 і x_2 вона записується у вигляді

$$x_1 \downarrow x_2 = \overline{x_1 \vee x_2} = y_{\text{п}}. \qquad (4.10)$$

Співвідношення (4.10) означає, що $y_{\text{п}} = 1$ тоді і тільки тоді, коли $x_1 = x_2 = 0$. На рис. 4.22 подано зображення логічного елементу АБО-НІ та таблиця істинності.



Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

Рисунок 4.22 – Логічний елемент АБО-НІ

Реалізацію операцій І-НІ і АБО-НІ не складно виконати контактними колами, застосовуючи електромагнітні реле з нормально замкненими контактами (сигнал на вході управління реле відсутній і, відповідно, відсутня напруга на його обмотці). Для реалізації операції І-НІ електромагнітні реле включають в ланцюг паралельно, а в разі операції АБО-НІ – послідовно.

Також до елементарних логічних елементів можна віднести заперечне АБО та заперечне АБО-НІ.

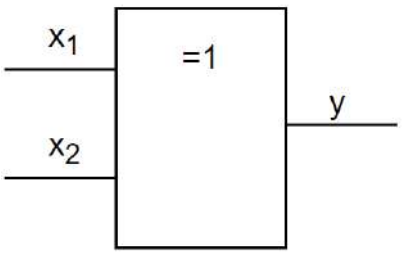
Заперечне АБО можна записати формулою

$$x_1 \oplus x_2 = y. \tag{4.11}$$

Співвідношення (4.11) означає, що $y = 0$ тоді, коли $x_1 = x_2 = 0$, або $x_1 = x_2 = 1$. В інших випадках $y = 1$. На рис. 4.23 подано зображення логічного елементу заперечне АБО та таблиця істинності.

Заперечне НІ можна записати формулою:

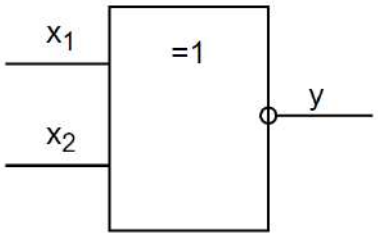
$$\overline{x_1 \oplus x_2} = y. \tag{4.12}$$



Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Рисунок 4.23 – Логічний елемент заперечне АБО

Співвідношення (4.12) означає, що $y = 1$ тоді, коли $x_1 = x_2 = 0$, або $x_1 = x_2 = 1$. В інших випадках $y = 0$. На рис. 4.24 наведено зображення логічного елементу заперечне НІ та таблиця істинності.



Таблиця істинності		
Входи		Вихід
x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	1

Рисунок 4.24 – Логічний елемент заперечне НІ

4.3 Створення логічних елементів засобами LD

4.3.1 Приклад створення логічного елементу НІ

Розглянемо приклади створення логічних елементів засобами контактних схем. Першим і найпростішим елементом є логічний елемент НІ. Умовне позначення логічного елементу НІ та таблиця істинності показані на рис. 4.18. Для його створення потрібні дві інструкції LD – контакт та котушка реле. Для котушки треба обрати тип «Negated» (рис. 4.25).

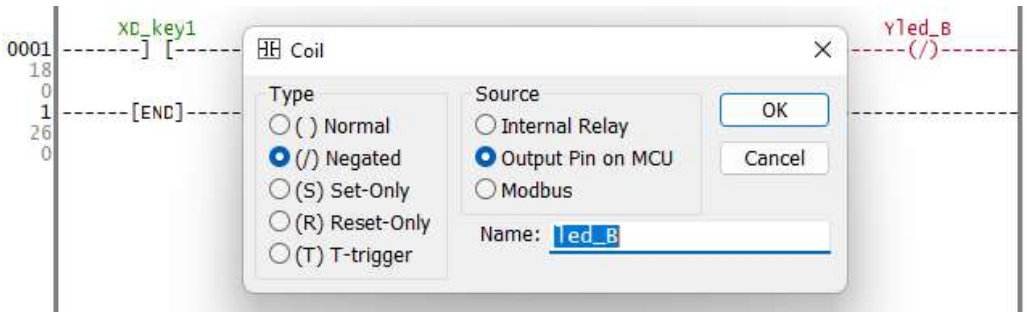


Рисунок 4.25 – Створення елементу НІ

В результаті отримуємо схему, що зображена на рис. 4.26.

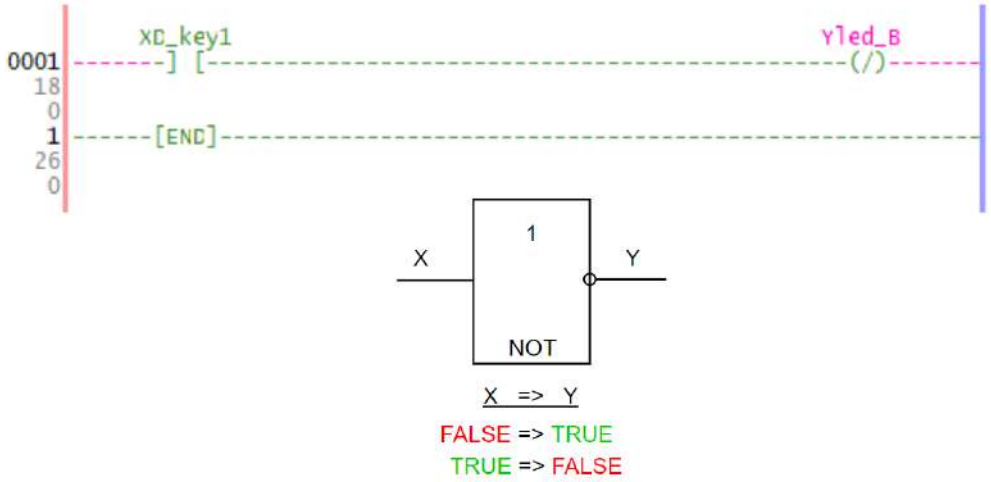
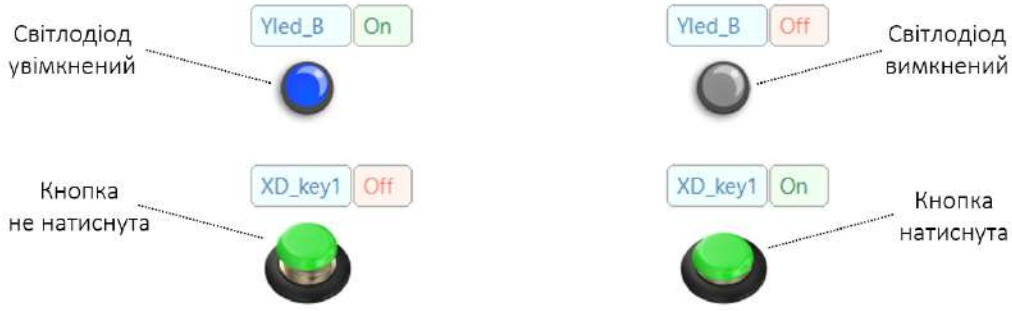


Рисунок 4.26 – Схема елементу НІ

На рис. 4.27 подано приклад роботи створеної схеми.



а) б)

Рисунок 4.27 – Приклад роботи схеми логічного елементу НІ

На початку роботи з подачею живлення кнопка XD_key1 не натиснута і реле, яке відповідає за світлодіод Yled_B, вимкнене, але його контакти замкнені і світлодіод горить (рис. 4.27, а). З натисканням кнопки XD_key1 спрацьовує реле, його контакти розмикаються і світлодіод вимикається (рис. 4.27, б).

4.3.2 Приклад створення логічного елементу I

Умовне позначення логічного елементу I та таблиця істинності показані на рис. 4.15. Для його створення потрібні три інструкції LD: два контакти та котушка реле (рис. 4.28).

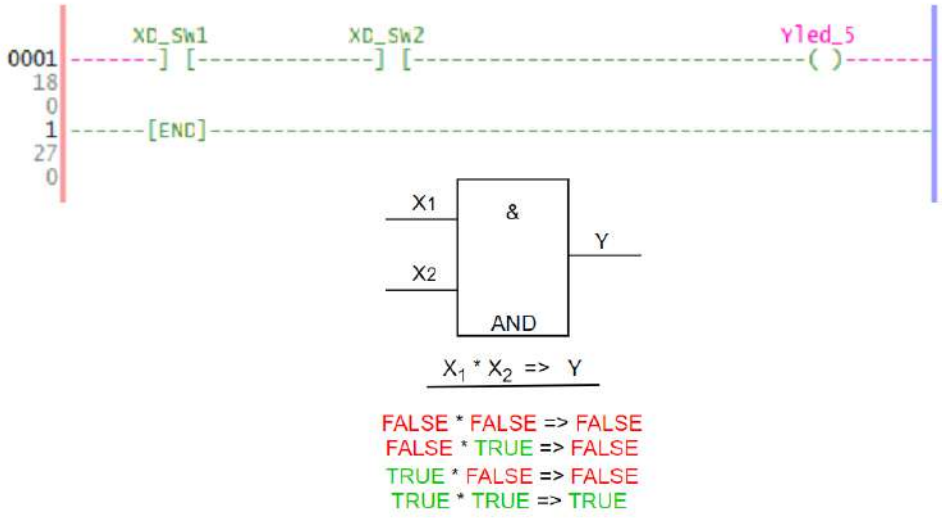


Рисунок 4.28 – Схема елементу I

На рис. 4.29 подано приклад роботи створеної схеми.

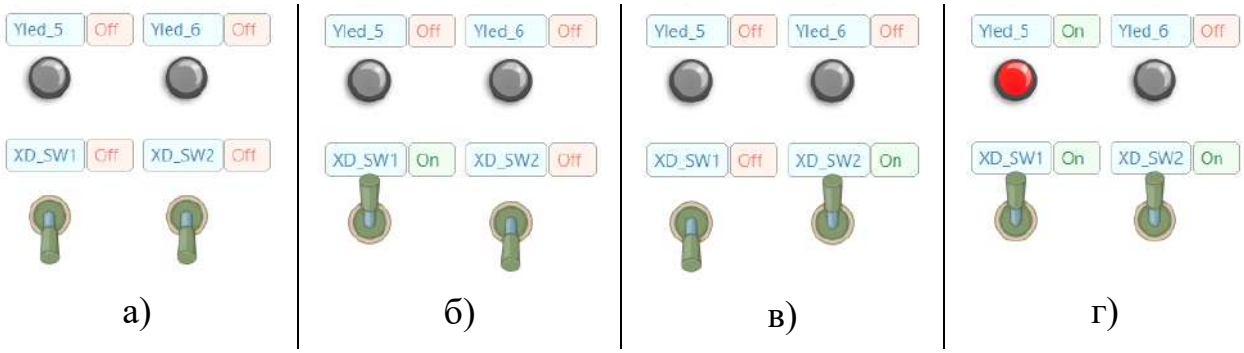


Рисунок 4.29 – Приклад роботи схеми логічного елементу I

З вмиканням живлення перемикачі XD_SW1 та XD_SW2 вимкнені, струм через них не тече, тому реле, яке відповідає за світлодіод Yled_5 теж вимкнене, і світлодіод не горить (рис. 4.29, а).

З вмиканням перемикача XD_SW1 через нього починає текти струм, але завдяки тому, що перемикач XD_SW2 вимкнений, далі він не проходить. Реле та світлодіод Yled_5 залишаються теж вимкненими (рис. 4.29, б). Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.29, в).

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2, тому струм через них проходить і потрапляє на реле, до контактів якого підключений світлодіод Yled_5. Таким чином, реле спрацьовує та світлодіод Yled_5 загоряється (рис. 4.29, г).

4.3.3 Приклад створення логічного елементу І-НІ

Умовне позначення логічного елементу І-НІ та таблиця істинності подані на рис. 4.21. Для його створення за основу беремо схему, що зображена на рис. 4.28, але робимо незначні зміни. Потрібно змінити тип котушки реле – в даному варіанті вона повинна працювати в інверсному режимі (рис. 4.30).

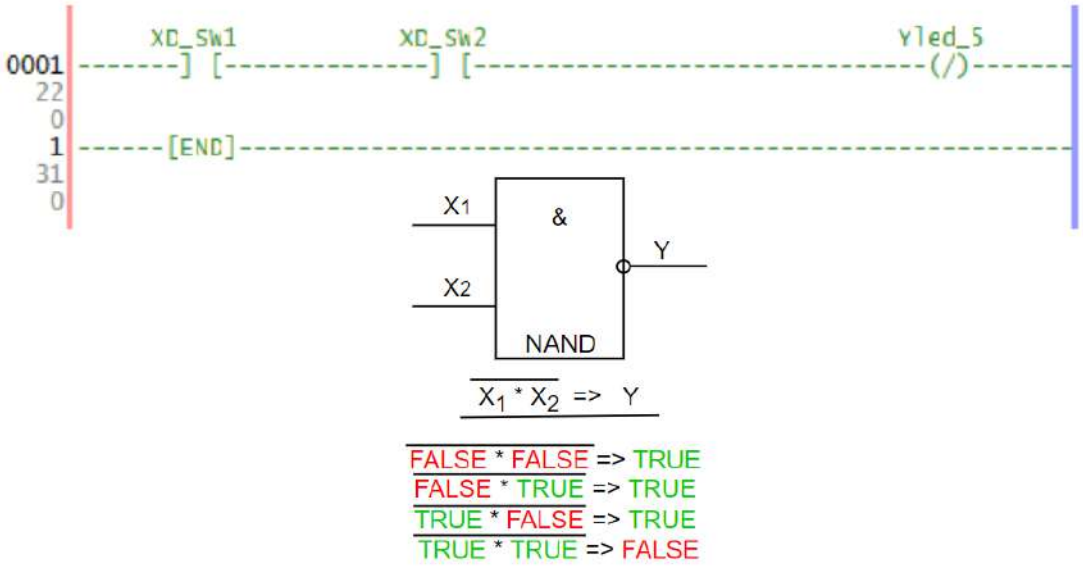
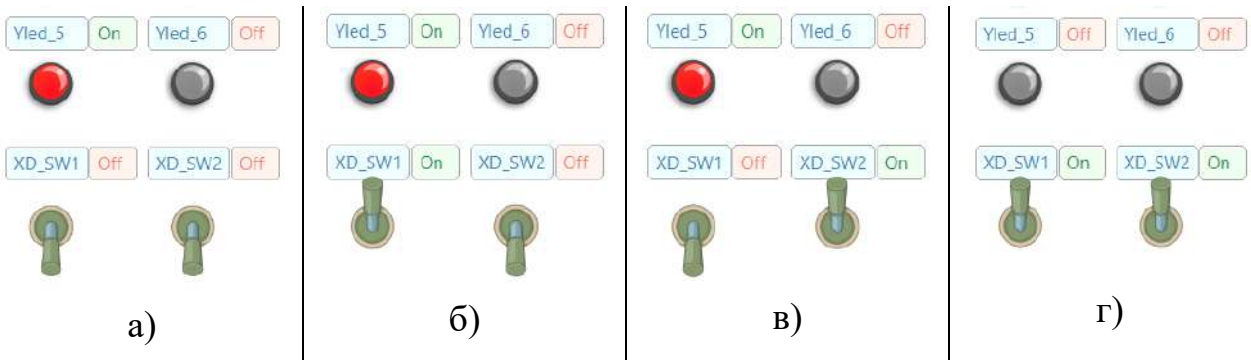


Рисунок 4.30 – Схема елементу І-НІ

На рис. 4.31 подано приклад роботи схеми логічного елементу І-НІ.



а) б) в) г)

Рисунок 4.31 – Приклад роботи схеми логічного елементу І-НІ

Якщо увімкнене живлення, перемикачі XD_SW1 та XD_SW2 вимкнені, струм через них не тече, реле вимкнене, але через те, що використано інверсний тип контактів, світлодіод Yled_5 увімкнений (рис. 4.31, а).

З вмиканням перемикача XD_SW1 через нього починає текти струм, але через те, що перемикач XD_SW2 вимкнений, далі він не проходить. Реле залишається вимкненим, а світлодіод Yled_5 залишається увімкненим (рис. 4.31, б). Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.31, в).

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2, тому струм через них проходить і потрапляє на реле, до контактів якого під'єднаний світлодіод Yled_5. Таким чином, реле спрацьовує, але завдяки його інверсним контактам світлодіод Yled_5 вимикається (рис. 4.31, г).

4.3.4 Приклад створення логічного елементу АБО

Умовне позначення логічного елементу АБО та таблиця істинності показані на рис. 4.12. Для створення даного елементу поєднаємо паралельно два перемикача XD_Sw1 та XD_Sw1, а до виходу під'єднаємо світлодіод через реле Yled_5. Схема поєднання елементів показана на рис. 4.32.

На рис. 4.33 подано приклад роботи схеми логічного елементу АБО. Порядок роботи схеми наступний. З вмиканням живлення перемикачі XD_SW1 та XD_SW2 вимкнені, струм через них не тече, тому реле, що відповідає за світлодіод Yled_5 теж вимкнене, і світлодіод не горить (рис. 4.33, а).

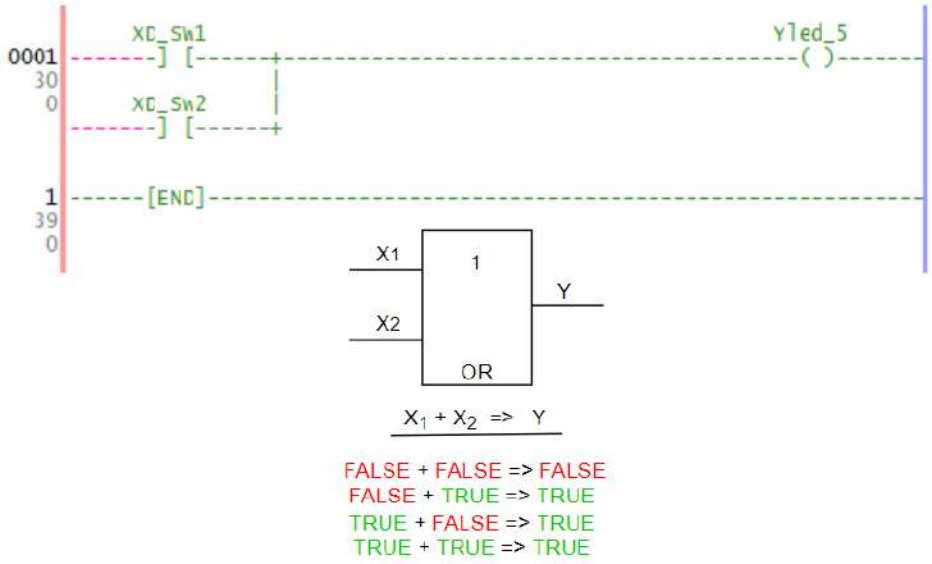
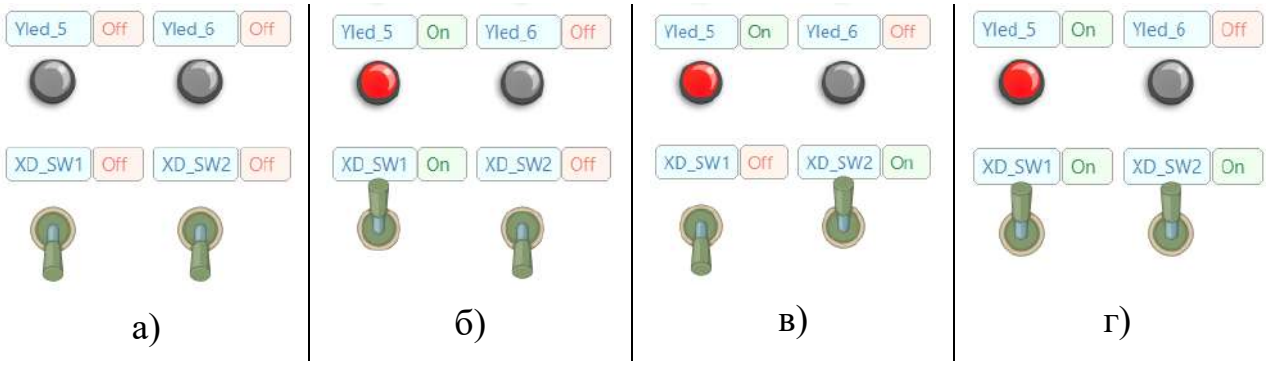


Рисунок 4.32 – Схема елементу АБО

З вмиканням перемикача XD_SW1 через нього починає текти струм, і хоча перемикач XD_SW2 вимкнений, він проходить далі до реле. Реле вмикається і світлодіод Yled_5 запалюється (рис. 4.33, б). Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.33, в).



а) б) в) г)

Рисунок 4.33 – Приклад роботи схеми логічного елементу АБО

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2, тому струм проходить одночасно через два перемикача і потрапляє на реле, до контактів якого під’єднаний світлодіод Yled_5. Таким чином, реле спрацьовує і світлодіод Yled_5 запалюється (рис. 4.33, г).

4.3.5 Приклад створення логічного елементу АБО-НІ

Умовне позначення логічного елементу АБО-НІ та таблиця істинності показані на рис. 4.22. Для створення даного елементу виконаємо модифікацію попередньої схеми, змінивши полярність контактів вихідного реле на протилежну. Все інше залишимо без змін. Схема поєднання елементів показана на рис. 4.34.

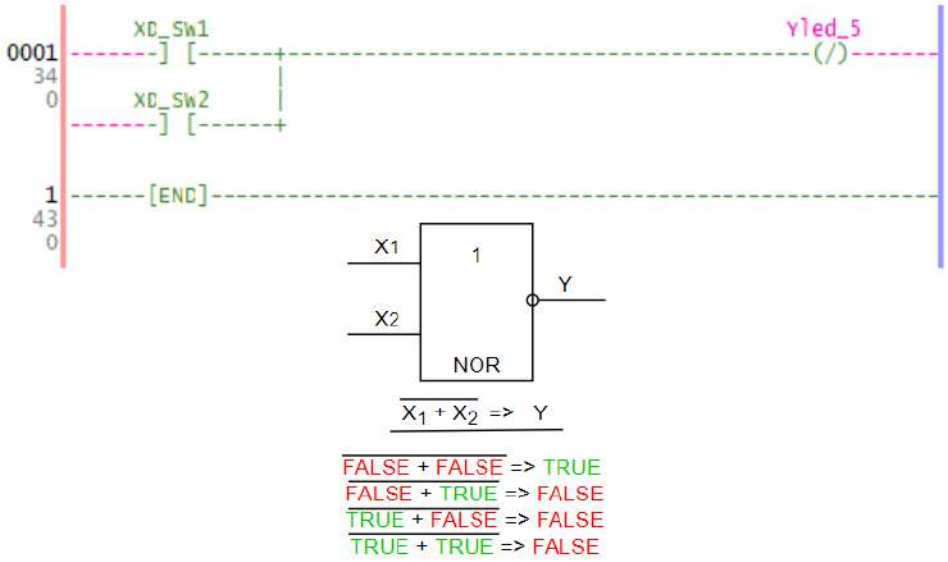


Рисунок 4.34 – Схема елементу АБО-НІ

На рис. 4.35 показано приклад роботи схеми логічного елементу АБО-НІ.

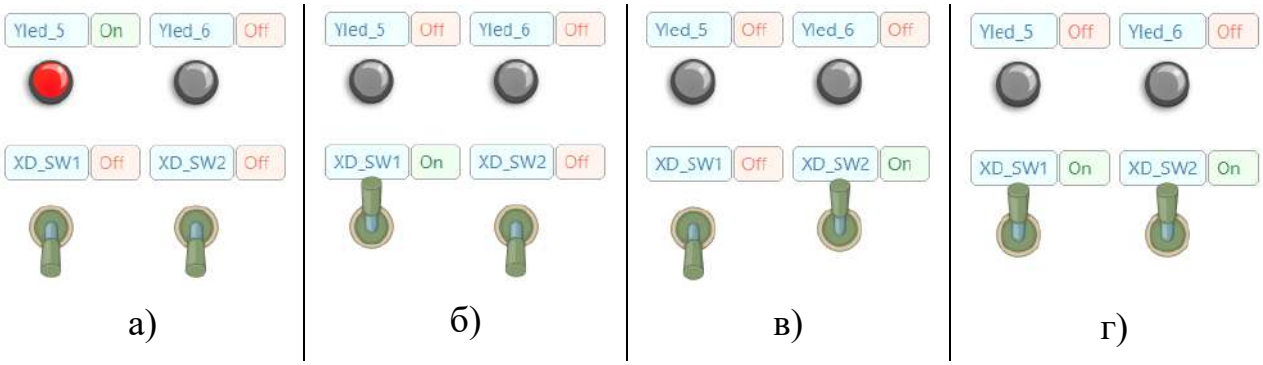


Рисунок 4.35 – Приклад роботи схеми логічного елементу АБО-НІ

Порядок роботи схеми наступний. З вмиканням живлення перемикачі XD_SW1 та XD_SW2 вимкнені, струм через них не тече, тому реле, що відповідає

за світлодіод Yled_5, теж вимкнене, але завдяки інверсному вмиканню контактів реле світлодіод горить (рис. 4.35, а).

З вмиканням перемикача XD_SW1 через нього починає текти струм, і хоча перемикач XD_SW2 вимкнений, він проходить далі до реле. Реле вмикається, але завдяки інверсному вмиканню його контактів, світлодіод Yled_5 вимикається (рис. 4.35, б). Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.35, в).

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2, тому струм проходить одночасно через два перемикача і потрапляє на реле, до контактів якого під'єднаний світлодіод Yled_5. Таким чином, реле спрацьовує та світлодіод Yled_5 вимикається, але завдяки інверсному вмиканню вихідних контактів (рис. 4.35, г).

4.3.6 Приклад створення логічного елементу заперечне АБО

Умовне позначення логічного елементу заперечне АБО та таблиця істинності показані на рис. 4.23. Для створення даного елементу необхідно залучити два перемикачі з двома парами контактів, що мають два стани – увімкнено і вимкнено. Контакти повинні бути під'єднані попарно (нормально розімкнений NO та нормально замкнений NC) та по діагоналі (NO навпроти NC та NC навпроти NO). До виходу такого з'єднання під'єднаємо світлодіод через реле Yled_5. Схема поєднання елементів подана на рис. 4.36.

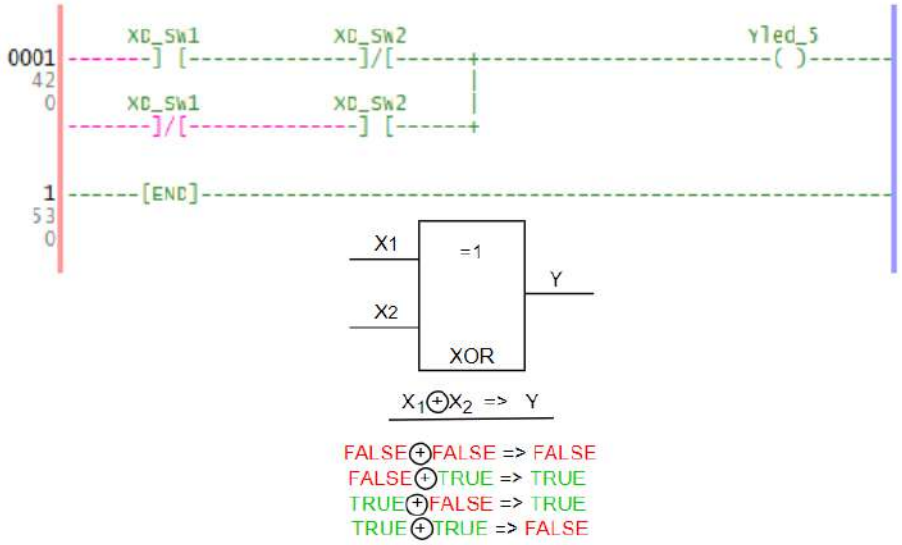
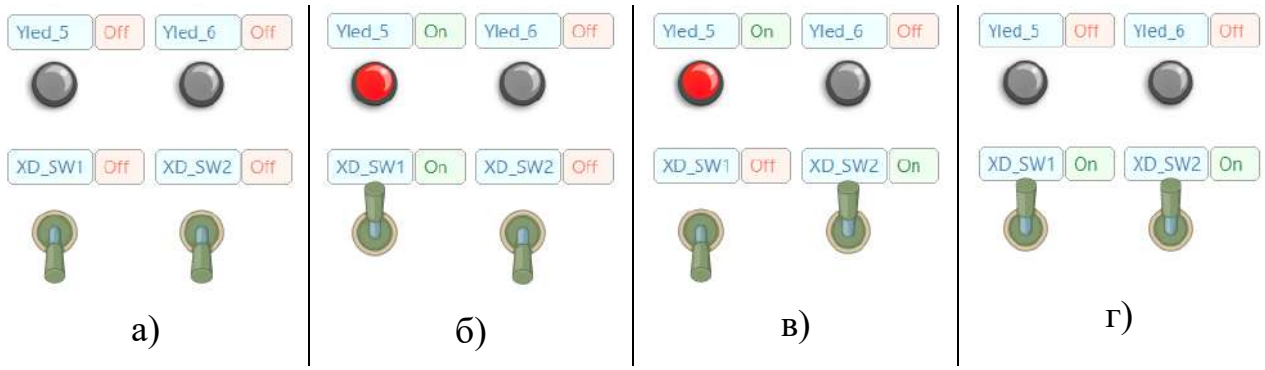


Рисунок 4.36 – Схема елементу заперечне АБО

На рис. 4.37 подано приклад роботи схеми логічного елементу заперечне АБО.



а) б) в) г)

Рисунок 4.37 – Приклад роботи схеми логічного елементу заперечне АБО

Порядок роботи схеми наступний. При вмиканні живлення перемикачі XD_SW1 та XD_SW2 вимкнені, завдяки тому, що використовуються дві пари контактів кожного з них, струм протікає через нормально замкнені контакти XD_SW1, але не протікає через нормально відкриті XD_SW1 (рис. 4.38). Таким чином, струм до реле не доходить і воно вимкнено, світлодіод Yled_5 не горить (рис. 4.37, а).

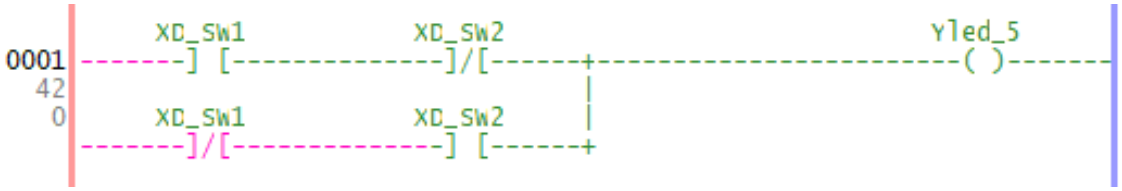


Рисунок 4.38 – Режим роботи логічного елементу заперечне АБО, в якому $x_1 = 0$ і $x_2 = 0$

З вмиканням перемикача XD_SW1 через його NO контакти починає текти струм, а через NC контакти – ні (рис. 4.39). В такому випадку струм проходить верхнім плечем схеми і потрапляє до реле. Реле вмикається і світлодіод Yled_5 запалюється (рис. 4.37, б).

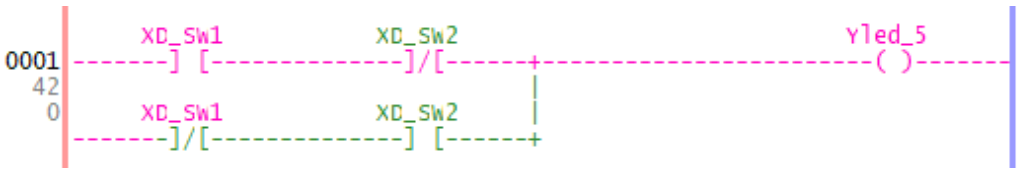


Рисунок 4.39 – Режим роботи логічного елементу заперечне АБО, в якому $x_1 = 1$ і $x_2 = 0$

Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.37, в). В даному випадку струм тече через нижнє плече схеми (рис. 4.40).

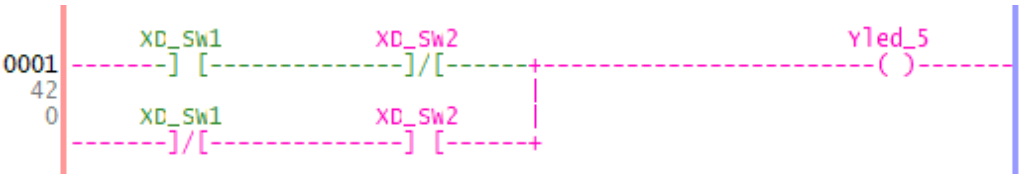


Рисунок 4.40 – Режим роботи логічного елементу заперечне АБО, в якому $x_1 = 0$ і $x_2 = 1$

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2. Дана ситуація подібна до першої, коли два перемикача вимкнені (рис. 4.41). Завдяки тому, що в увімкненому стані перемикача XD_SW1 його NC контакти розімкнені, струм далі них не тече, реле, до контактів якого під'єднаний світлодіод Yled_5, вимкнено і світлодіод не горить (рис. 4.37, г).

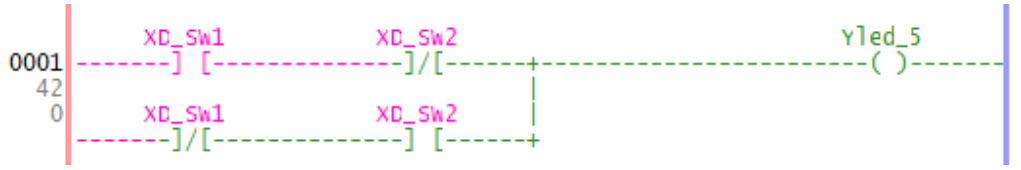


Рисунок 4.41 – Режим роботи логічного елементу XOR, в якому $x_1 = 1$ і $x_2 = 1$

4.3.7 Приклад створення логічного елементу заперечне НІ

Умовне позначення логічного елементу заперечне НІ та таблиця істинності показані на рис. 4.24. Для створення даного елементу необхідно змінити схему, що зображена на рис. 4.36. Зміни стосуються перемикача XD_SW2. Потрібно змінити порядок вмикання його контактів на протилежний (рис. 4.42).

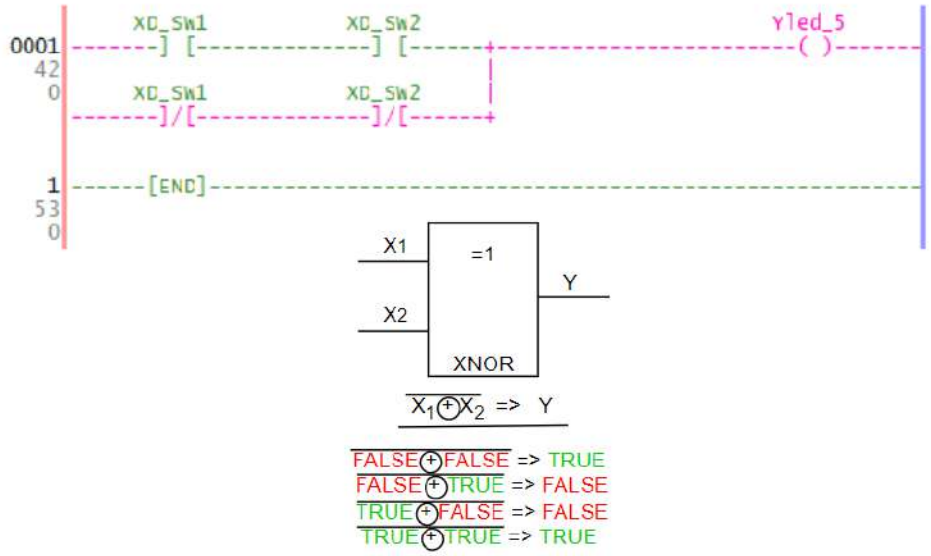
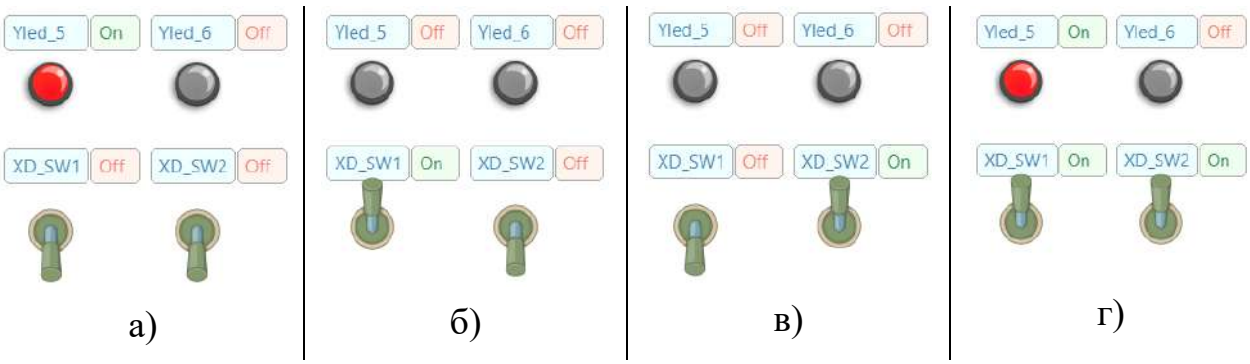


Рисунок 4.42 – Схема елементу заперечне НІ

Тобто контакти повинні бути увімкнені наступним чином: нормально розімкнений NO XD_SW1 та нормально розімкнений NO XD_SW2 на першій сходинці, і нормально замкнений NC XD_SW1 та нормально замкнений NC XD_SW2 на другій сходинці. До виходу такого з'єднання під'єднаний світлодіод через реле Yled_5. На рис. 4.43 показано приклад роботи схеми логічного елементу заперечне НІ.

Порядок роботи схеми в даній конфігурації наступний. З вмиканням живлення перемикачі XD_SW1 та XD_SW2 вимкнені, але завдяки тому, що використовуються дві пари контактів NO та NC кожного з них, з'єднаних послідовно, струм протікає через нормально замкнені контакти XD_SW1 та XD_SW2 (рис. 4.44). Таким чином, струм доходить до реле і воно вмикається – світлодіод Yled_5 горить (рис. 4.43, а).



а) б) в) г)

Рисунок 4.43 – Приклад роботи схеми логічного елементу заперечне НІ

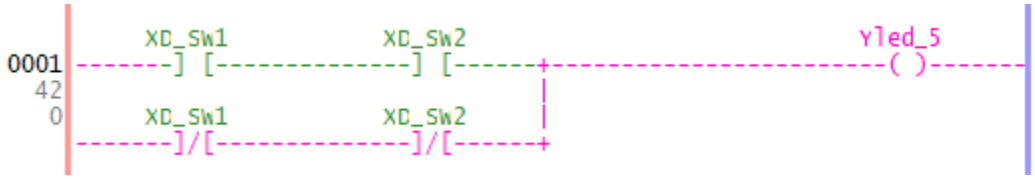


Рисунок 4.44 – Режим роботи логічного елементу заперечне НІ,
в якому $x_1 = 0$ і $x_2 = 0$

З вмиканням перемикача XD_SW1 через його NO контакти починає текти струм, а через NC контакти – ні. При цьому перемикач XD_SW2 вимкнено (рис. 4.45). В такому випадку струм не може пройти ні через верхнє, ні через нижнє плече. Реле вимкнено і світлодіод Yled_5 не горить (рис. 4.43, б).

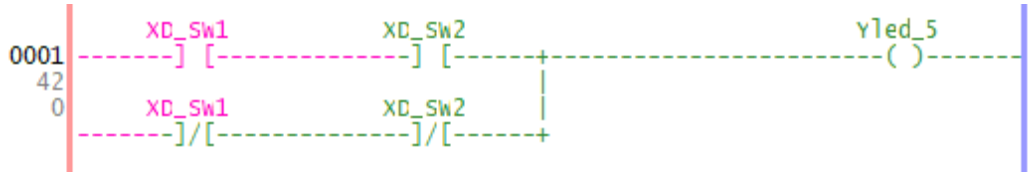


Рисунок 4.45 – Режим роботи логічного елементу заперечне НІ,
в якому $x_1 = 1$ і $x_2 = 0$

Така ж сама ситуація виникає, якщо увімкнути перемикач XD_SW2 та вимкнути XD_SW1 (рис. 4.43, в). В даному випадку струм тече через нижній контакт XD_SW1, але контакт NC XD_SW2 вимкнено (рис. 4.46).

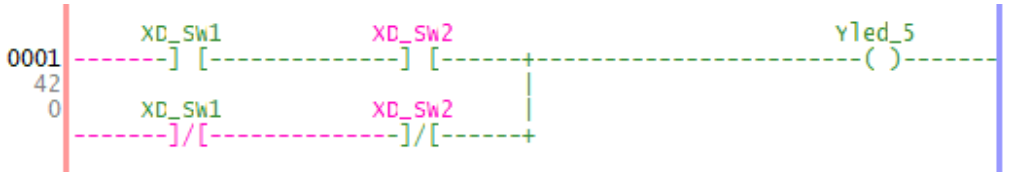


Рисунок 4.46 – Режим роботи логічного елементу заперечне НІ, в якому $x_1 = 0$ і $x_2 = 1$

В останній комбінації увімкнені обидва перемикачі XD_SW1 та XD_SW2. Дана ситуація подібна до першої, коли два перемикача вимкнені (рис. 4.47).

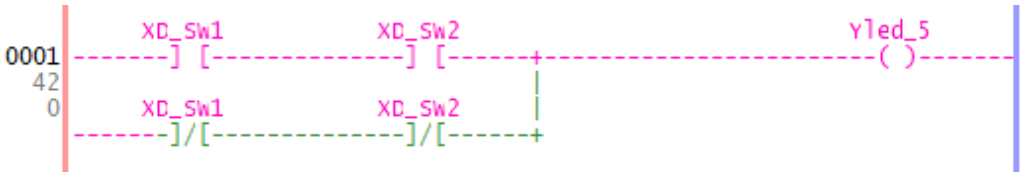


Рисунок 4.47 – Режим роботи логічного елементу заперечне НІ, в якому $x_1 = 1$ і $x_2 = 1$

Завдяки тому, що в увімкненому стані перемикачів XD_SW1 і XD_SW2 їх NO контакти замкнені, струм проходить через них далі до реле Yled_5, яке вмикає світлодіод (рис. 4.43, г).

4.3.8 Моделювання роботи асинхронного тригеру

Тригером називається електронний пристрій, призначений для запису, зберігання та зчитування двійкової інформації. На рис. 4.48 зображено умовне позначення асинхронного RS-тригера, в якому відбувається перемикання з одного стійкого стану до іншого під дією певної сукупності роздільних імпульсів напруги на входах, що управляють.

Тригер має інформаційні входи R і S, а також інформаційні виходи: прямий Q та інверсний $\overline{Q}$. Вхід R (від англійського слова Reset) є входом встановлення тригера в стан логічного 0, вхід S (від англійського слова Set) – входом встановлення тригера в стан логічної 1.

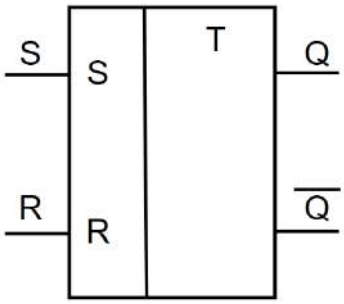
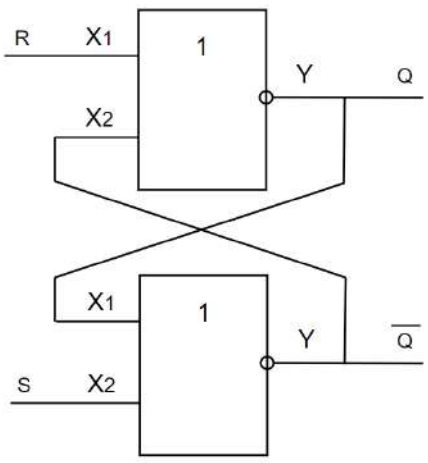


Рисунок 4.48 – Умовне позначення асинхронного RS-тригера

На рис. 4.49 подана структурна логічна схема та таблиця істинності RS-тригера, побудованого на двох логічних елементах АБО-НІ. Роботу RS-тригера ілюструє таблиця істинності.

З таблиці істинності видно, що RS-тригер працює тільки у випадках, коли на інформаційні входи R і S надходять окремі сигнали логічний нуль та логічна одиниця (або навпаки). З одночасною подачею на інформаційні входи R і S логічного нуля тригер знаходиться в стані збереження попередньої інформації на його інформаційних виходах. Якщо ж на інформаційні входи R і S одночасно надходять сигнали логічної одиниці, то тригер перебуває у стані заборони.



Таблиця істинності		
Входи		Вихід
S	R	Q
0	0	Q_0
0	1	0
1	0	1
1	1	-

Рисунок 4.49 – Структурна логічна схема та таблиця істинності RS-тригера

Для створення RS-тригеру у вигляді контактної схеми використовуватиме два елементи АБО-НІ, які розглянуто в пункті 4.3.5. Приклад такої схеми подано на рис. 4.50.

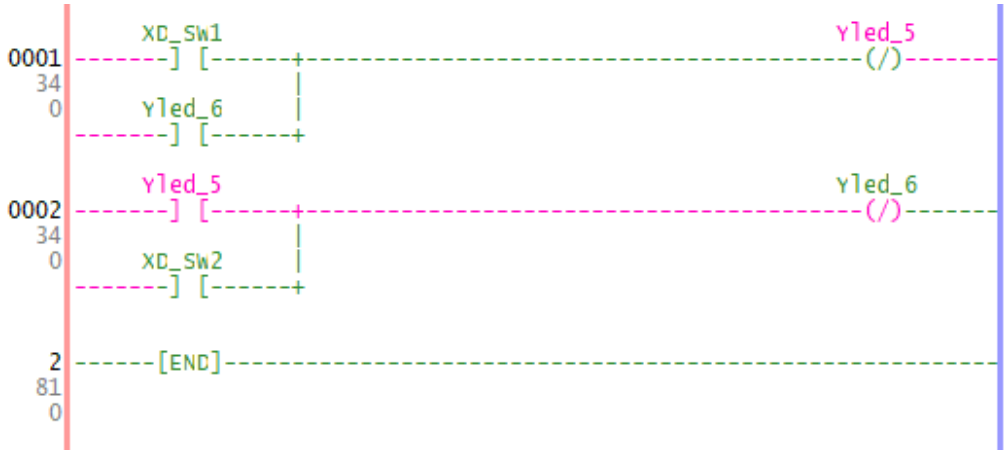


Рисунок 4.50 – RS-тригер у вигляді контактної схеми

Відповідно до рис. 4.49 перший елемент АБО-НІ реалізовано з використанням перемикача XD_SW1 та реле з інверсним виходом Yled_5. Другий елемент АБО-НІ реалізовано з використанням перемикача XD_SW2 та реле з інверсним виходом Yled_6. Вихід Yled_6 ($\overline{Q}$) другого елементу під'єднано на вхід X2 першого елементу АБО-НІ, а вихід Yled_5 (Q) другого елементу підключено на вхід X1 другого елементу АБО-НІ.

Виконаємо перевірку працездатності RS-тригеру на віртуальному макеті. У випадку, що зображено на рис. 2.51 перемикач XD_SW1 знаходиться у ввімкненому положенні, а перемикач XD_SW2 – у вимкненому.

Рисунок 4.51 – Режим роботи RS-тригеру, коли на вході R знаходиться логічна одиниця, а на вході S – логічний нуль

На рисунку 4.52 можна бачити внутрішній стан ланок контактної схеми.

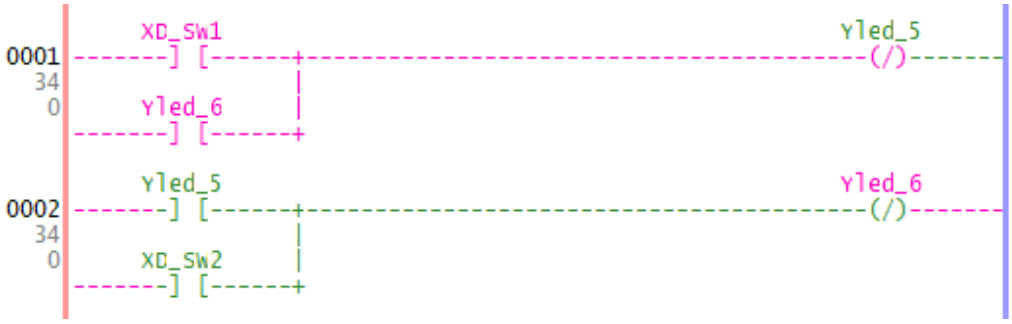


Рисунок 4.52 – Внутрішній стан ланок контактної схеми, якщо R = 1, S = 0

Завдяки тому, що і перемикач XD_SW1 і вихід реле Yled_6 знаходяться в стані логічної одиниці, вихід Yled_5 буде в стані логічного нуля. Можна бачити, що поточний стан відповідає таблиці істинності, що подана на рис. 4.49.

Розглянемо інший стійкий стан роботи RS-тригеру, коли на вході R знаходиться логічний нуль, а на вході S – логічна одиниця (рис. 4.53).

Внутрішній стан ланок контактної схеми показано на рис. 4.54.

Рисунок 4.53 – Режим роботи RS-тригеру, коли на вході R знаходиться логічний нуль, а на вході S – логічна одиниця

З рис. 4.54 можна бачити, що і перемикач XD_SW2 знаходиться у ввімкненому стані, а XD_SW1 – у вимкненому. Враховуючи інверсні виходи реле Yled_5 та Yled_6 на вході першого елементу АБО-НІ знаходиться логічний нуль, а на вході другого – логічна одиниця. В такому стані вихід реле Yled_6 знаходяться

в стані логічної одиниці, вихід Yled_6 – в стані логічного нуля. Таким чином, поточний стан тригеру відповідає таблиці істинності, що подана на рис. 4.49.

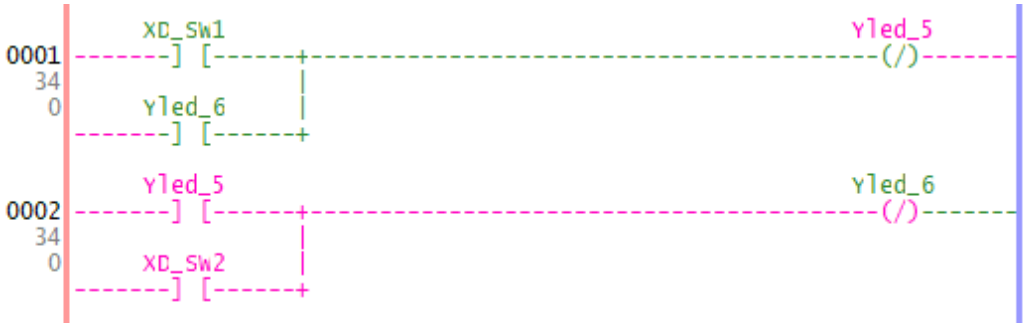
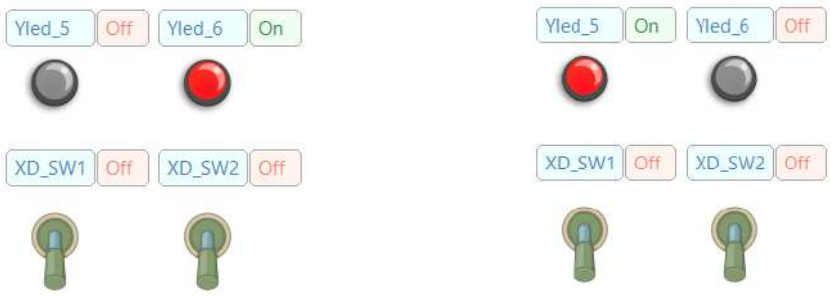


Рисунок 4.54 – Внутрішній стан ланок контактної схеми, якщо $R = 0$, $S = 1$

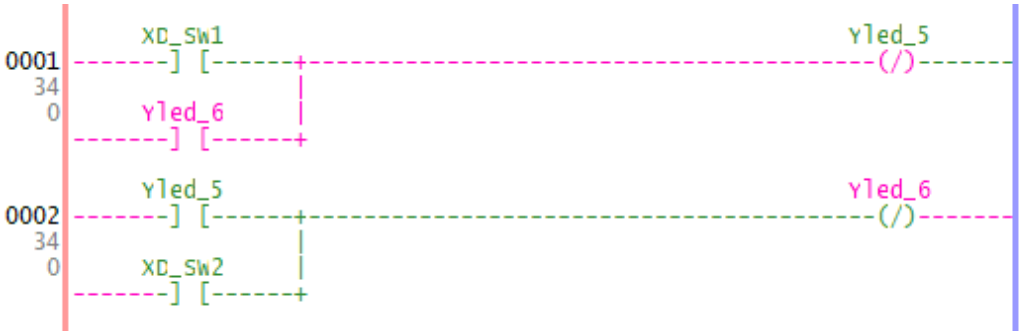
В таблиці істинності (рис. 4.49) показано, що у випадку, коли на обох входах тригеру знаходиться логічний нуль, на виходах фіксується попередній стійкий стан. Перевіримо це на практиці для двох попередніх варіантів роботи тригера. Спочатку встановимо на вході стан $R = 1$, $S = 0$ і далі переводимо обидва перемикачі у вимкнений стан (рис. 4.55, а), а потім встановимо $R = 0$, $S = 1$ і знову переводимо обидва перемикачі у вимкнений стан (рис. 4.55, б).



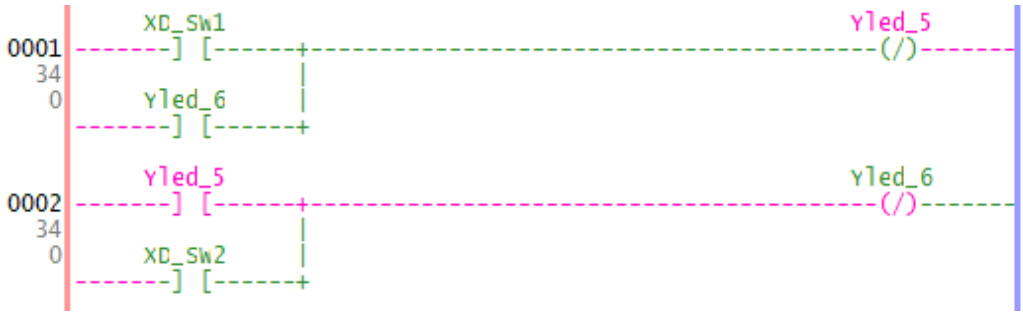
а) б)

Рисунок 4.55 – Фіксація попереднього стійкого стану роботи RS-тригеру

Внутрішній стан ланок контактної схеми обох режимів роботи тригеру подано на рис. 4.56. На рис. 4.56, а подано випадок переведення тригеру в режим фіксації після стану $R = 1$, $S = 0$, а на рисунку 4.56, б – переведення в режим фіксації після стану $R = 0$, $S = 1$.



а)



б)

Рисунок 4.56 – Випадки переведення тригеру в режим фіксації після стану $R = 1, S = 0$ (а), та $R = 0, S = 1$ (б)

На рис. 4.57 подано випадок невизначеного стану, коли на інформаційні входи R і S одночасно надходять сигнали логічної одиниці та тригер переходить у стан заборони.

Рисунок 4.57 – Стан заборони на виході RS-тригеру

Відповідна контактна схема, з якої можна дізнатися про внутрішній стан ланок тригеру, подана на рис. 4.58.

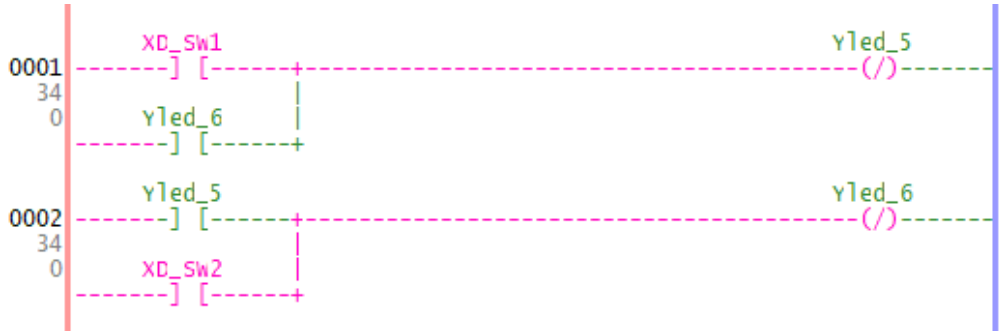


Рисунок 4.58 – Стан контактної схеми для одночасного надходження логічних одиниць на вхід RS-тригеру

4.3.9 Приклад створення програми для автоматизованої охоронної сигналізації на логічних елементах

Охоронна система може знаходитись у двох станах:

– охорона;

– тривога.

Перехід між станами виконується за умови спрацювання відповідних датчиків. Перехід зі стану «Охорона» до стану «Тривога» здійснюється, коли спрацьовує один з датчиків: відкриття дверей або руху. Повернення до стану «Охорона» відбувається після натискання кнопки скидання.

Схема алгоритму роботи охоронної сигналізації подана на рис. 4.59.

Алгоритм роботи модуля управління охоронною сигналізацією наступний:

– на початку роботи система знаходиться в режимі «Охорона»;

– вмикається зелений світлодіод, що сигналізує про знаходження в режимі «Охорона»;

– запускається цикл перевірки стану контактів датчику відкриття дверей XD_SW3 або датчику руху XD_SW4;

– якщо хоча б один із датчиків спрацював, вимикається світлодіод «Охорона» та система переходить в режим «Тривога»;

– вмикається червоний світлодіод – індикатор режиму «Тривога»;

– вмикається звукова сигналізація Ybuzzer;

– запускається цикл перевірки стану контактів кнопки скидання сигналу «Тривога» XDkey_1;

– якщо виявиться, що кнопка скидання натиснута, система послідовно повертається в режим «Охорона»;

– вимикається червоний світлодіод – індикатор режиму «Тривога»;

– вимикається звукова сигналізація Ybuzzer;

– виконується перехід в початковий режим роботи.

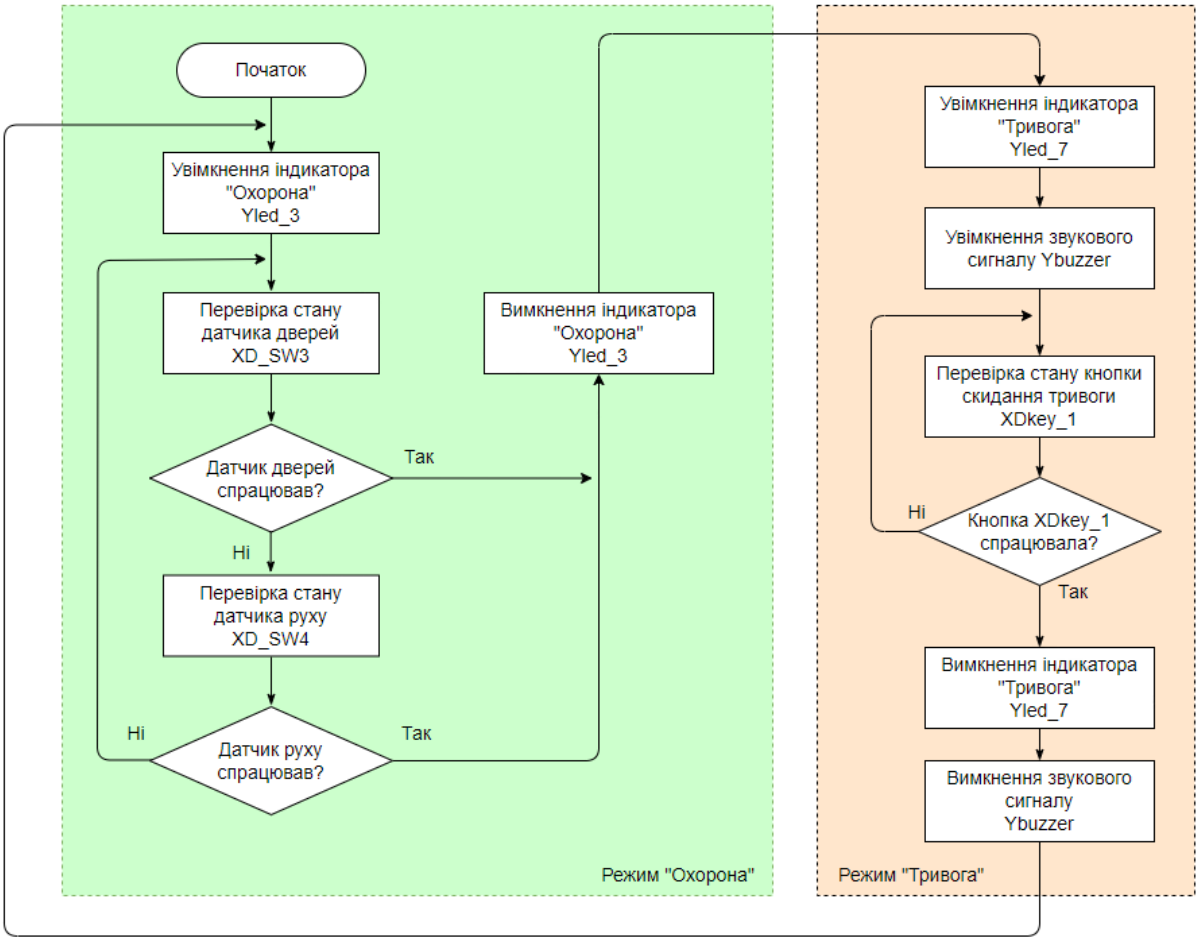


Рисунок 4.59 – Алгоритм роботи модуля управління охоронною сигналізацією

З наведеного алгоритму можна бачити, що охоронна сигналізація вмикається, якщо спрацював будь-який з двох датчиків. Таким чином, для реалізації алгоритму засобами контактної схеми, використовуємо поєднання цих датчиків за схемою АБО. На рис. 4.60 подана контактна схема системи охоронної сигналізації. З поданого рисунку можна бачити, що окрім логічного елементу АБО в схемі також застосовані RS-тригери для управління світлодіодами Yled_3 та Yled_7.

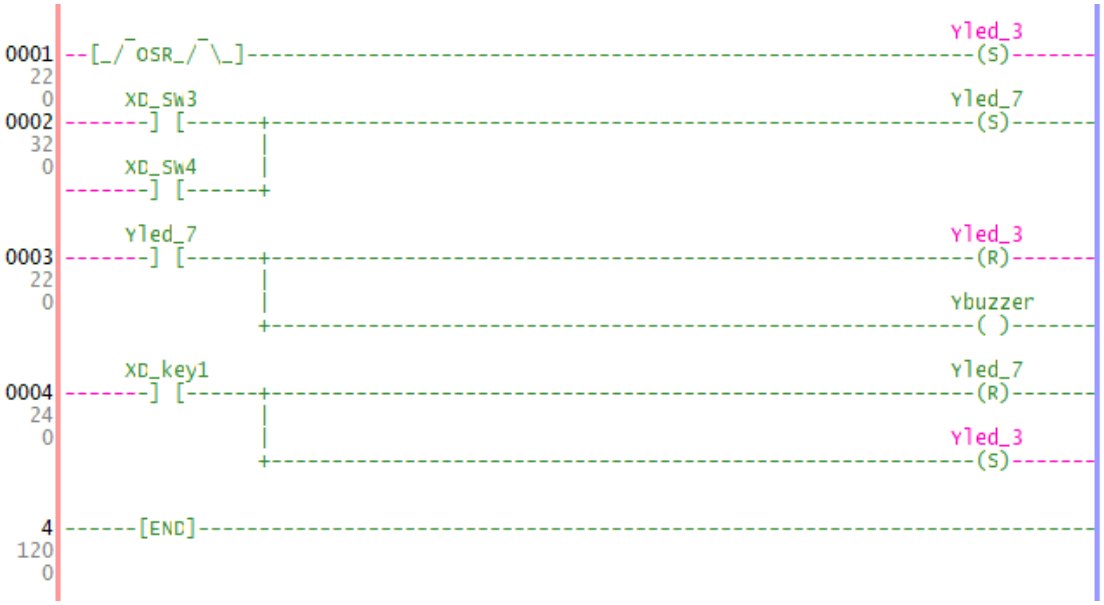


Рисунок 4.60 – Контактна схема системи охоронної сигналізації

Для перевірки схеми застосовуємо програму емулятор «Discrete Input». Роль датчика відкриття дверей виконує перемикач XD_SW3, а роль датчика руху – XD_SW4. Світлодіод Yled_3 сигналізує про режим «Охорона», а світлодіод Yled_7 – про режим «Тривога».

Після вмикання охоронної сигналізації всі датчики вимкнені і знаходяться в режимі охорони. Система охоронної сигналізації також знаходиться в режимі охорони – зелений світлодіод горить (рис. 4.61).

Рисунок 4.61 – Режим «Охорона»

Спрацювання будь-якого з датчиків (XD_SW3 або XD_SW4) приводить до спрацювання охоронної сигналізації (рис. 4.62)

а) б) в)

Рисунок 4.62 – Варіанти спрацювання датчиків охоронної сигналізації:
а) спрацювання датчика відкриття дверей; б) спрацювання датчика руху;
в) спрацювання обох датчиків

В такому положенні система буде знаходитись до натискання на кнопку скидання XD_key1 навіть після повертання датчиків охоронної сигналізації в нормальний режим (рис. 4.63).

Режим «Тривога» Режим «Охорона»

Рисунок 4.63 – Переведення системи в режим «Охорона» з режиму «Тривога» натисканням на кнопку скидання SD_key1

4.4 Синтез багатотактних автоматів

Схема є однотактною, якщо стан її виходів визначається тільки комбінацією значень вхідних сигналів і не залежить від послідовності їх надходження. У багатотактних схемах стан виходів залежить не тільки від набору вхідних сигналів у певний момент часу, але й від послідовності їх надходження або від внутрішнього стану схеми.

В процесі синтезу схем на тригерах, умови роботи багатотактних схем подаються у вигляді графів переходів.

Граф переходів – це графічне зображення послідовності роботи багатотактної схеми. Елементами графа є вершини і ребра. Вершини відповідають станам схеми і позначаються кружками. Ребра – це лінії із стрілками, що з'єднують вершини і показують напрям переходу з одного стану схеми в інший.

Кількість вершин графа при синтезі асинхронних схем на RS-тригерах визначається з умови:

$$2^n \geq S, \tag{4.1}$$

де S – кількість станів схеми;

2^n – кількість вершин графа;

n – кількість тригерів.

З огляду на метод спрощення виразів булевої алгебри – карту Карно, вершини графа рекомендується розміщувати таким чином, щоб вони створювали конфігурацію 2×2, якщо $n = 2$, конфігурацію 2×4, якщо $n = 3$, конфігурацію 4×4, якщо $n = 4$.

Карта Карно – один з графічних способів подання логічних функцій. Для функцій n змінних вона складається з 2^n клітинок, причому кожна клітинка відповідає певному набору змінних. Вигляд карт Карно для функцій двох, трьох і чотирьох змінних зображено на рис. 4.64. Вхідні змінні розміщуються з зовнішніх сторін карти проти її рядків або стовпців.

Значення вхідної змінної стосується усіх клітинок у рядку або стовпці і дорівнює 1, якщо проти рядка або стовпця є дужка з позначенням цієї змінної. Для

решти рядків і стовпців значення змінної дорівнює 0. У клітинках карти записується те значення функції, яке вона має у наборах вхідних змінних, що відповідають цим клітинкам.

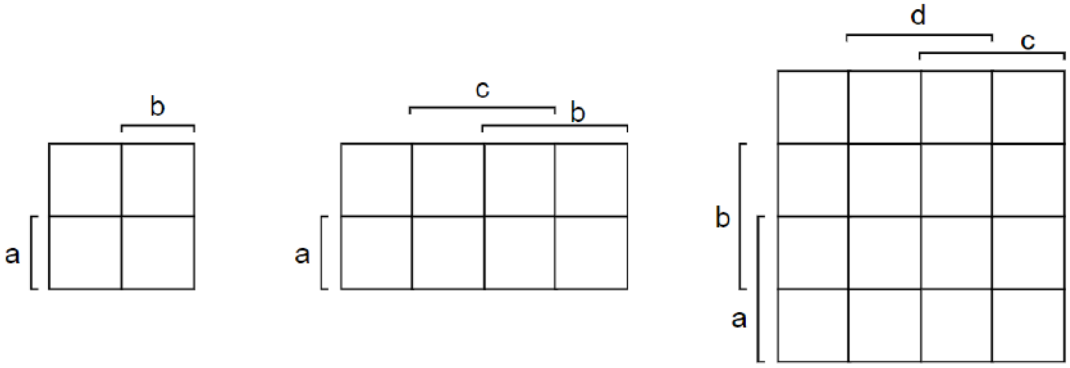


Рисунок 4.64 – Форма запису карт Карно

Наприклад, якщо кількість тригерів $n = 3$, то граф переходів матиме вигляд, як подано на рис. 4.65.

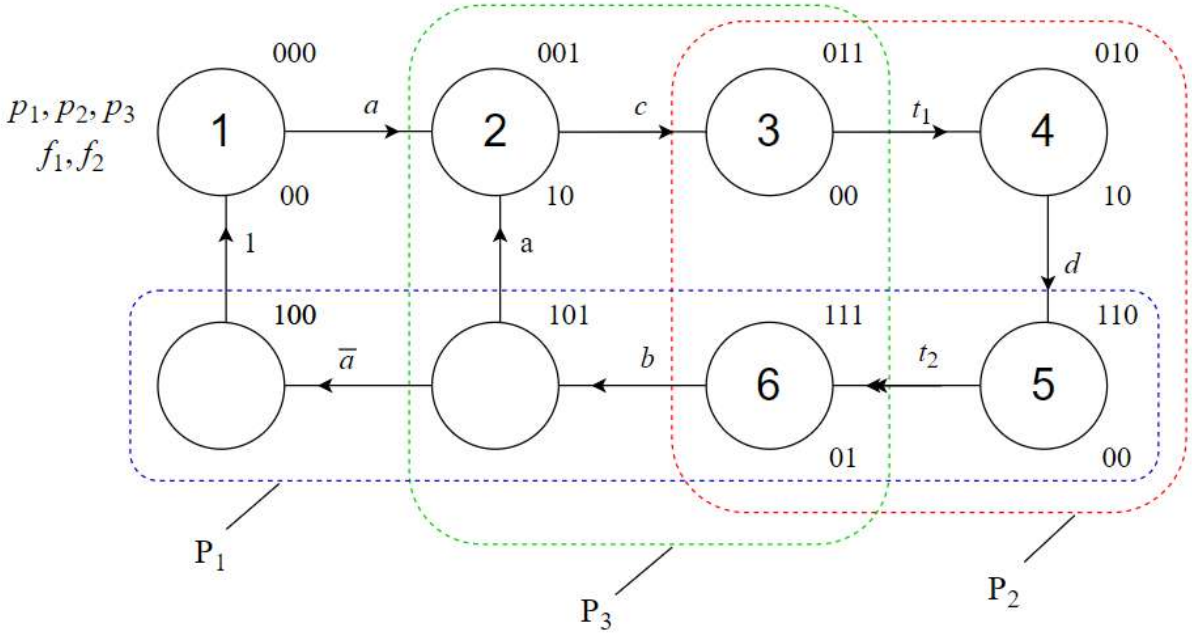


Рисунок 4.65 – Вигляд графа переходів, якщо кількість тригерів $n = 3$

Вихідні сигнали тригерів P_1, P_2, P_3 виконують роль проміжних змінних, тому вихідні сигнали тригерів позначаються буквами p_1, p_2, p_3. Ці позначення записують зверху ліворуч від графа.

Кожна вершина графа кодується набором значень вихідних сигналів тригерів. Коди вибираються так, щоб для сусідніх вершин вони відрізнялися значенням тільки однієї змінної. В початковому стані схеми (стан очікування) зазвичай приймають, що усі проміжні змінні дорівнюють нулю. Значення проміжних змінних для кожної вершини записуються над кружками у тій черговості, в якій записані позначення тригерів.

Вершини, між якими повинні відбуватися переходи, з'єднують ребрами із стрілками. Над стрілками або праворуч від них, якщо ребра спрямовано вертикально, записують позначення вхідних сигналів, що спричинюють ці переходи.

Під час побудови схем на асинхронних RS-тригерах переходи можна робити тільки між сусідніми вершинами. Якщо ця умова не виконується, то необхідно передбачити так звані природні переходи (за рахунок подавання вхідного сигналу одиниці) через проміжні нестійкі стани.

У деяких системах промислової автоматики потрібно забезпечувати певну затримку часу між операціями, що виконуються механізмами. У схемах таких систем застосовуються елементи часу, які затримують сигнали елементів, що реалізують проміжні функції.

На першому етапі синтезу схем із спеціальними технологічними затримками необхідно зазначити, які операції супроводжуються затримками і вказати величини затримок.

Основна особливість процесу синтезу з урахуванням технологічних затримок виявляється під час вибору кількості проміжних змінних і розміщення їх станів. Кожному переходу схеми з одного стану в інший з незалежною затримкою має відповідати змінювання значення однієї проміжної змінної з 1 на 0 або з 0 на 1. Більшість реальних елементів часу, що застосовуються в схемах промислової автоматики, забезпечують регульовану затримку часу тільки в разі змінювання сигналу на їх вході з 0 на 1. Сигнал на виході елемента часу зникає одночасно зі зникненням сигналу на його вході. Такі ж самі функції виконують програмно реалізовані таймери програмованих логічних контролерів. Друга особливість синтезу схем із затримками – схема перебуває в нестійких станах протягом технологічної затримки.

Побудуємо граф переходів, що описує роботу схеми керування механізмом, який працює в режимі циклів, що повторюються, та має технологічні затримки. Наприклад, це може бути вантажна рухома платформа, яка переміщується між одним пунктом завантаження і двома пунктами розвантаження A, B, C (рис. 4.66). Під час прибуття в ці пункти вантажна платформа замикає кінцеві вимикачі SQ_1, SQ_2, SQ_3.

В початковому положенні платформи натиснутий кінцевий вимикач SQ_1. Цикл роботи починається після надходження команди «Пуск». Платформа переміщується в положення, що фіксується кінцевим вимикачем SQ_2, зупиняється на час Δt_1. Далі переміщується в положення, що фіксується кінцевим вимикачем SQ_3, зупиняється на час Δt_2 , а потім повертається в початкове положення. Для повторення циклу необхідно знов подати команду «Пуск».

Якщо ця команда надходить безперервно, то після відпрацьовування одного циклу автоматично починається наступний.

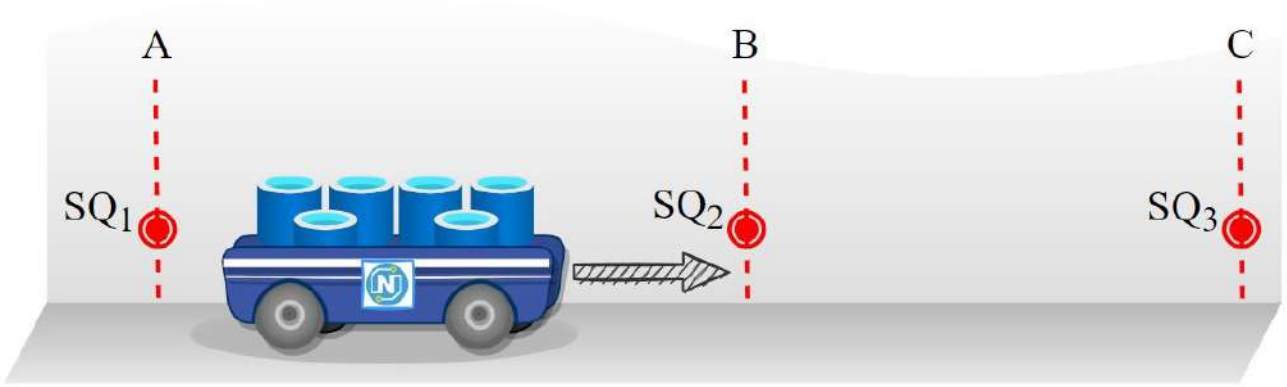


Рисунок 4.66 – Переміщення вантажної платформи

Приймемо такі позначення вхідних і вихідних сигналів, а також сигналів таймерів, які необхідно розглядати як вхідні сигнали для графа переходів.

Вхідні сигнали:

– a – команда «Пуск»;

– b, c, d – сигнали кінцевих вимикачів SQ_1, SQ_2, SQ_3 відповідно;

– t_1, t_2 – сигнали таймерів, що надають затримки Δt_1 і Δt_2.

Вихідні сигнали:

– f_1 – команда на переміщення платформи з початкового положення;

– f_2 – команда на повернення платформи в початкове положення.

Побудову графа переходів починаємо з визначення кількості станів, в яких може перебувати схема автоматичного керування. Таких станів шість:

– 1 – початкове положення;

– 2 – переміщення з вихідного положення;

– 3 – зупинка протягом часу Δt_1;

– 4 – подальше переміщення;

– 5 – зупинка протягом часу Δt_2;

– 6 – повернення в початкове положення.

Виходячи з кількості станів схеми $6 < 2^3$, визначаємо кількість тригерів $n = 3$ і кількість вершин графа переходів $2^3 = 8$. Позначаємо проміжні змінні p_1, p_2, p_3, будуємо 8 вершин графа і кодуємо їх комбінаціями значень проміжних змінних (рис. 4.65).

Кожному стану схеми ставимо у відповідність одну з вершин графа, причому стани, між якими повинен відбуватися перехід згідно з умовами роботи схеми, розміщуємо у сусідніх вершинах. Вільні вершини використовуємо для переходу зі стану 6 в стан 2 під час роботи схеми в режимі циклів, що повторюються, або в стан 1 при відпрацьовуванні одиночних циклів.

Позначення вихідних сигналів f_1 і f_2 записано знизу позначень проміжних змінних. Значення f_1 і f_2 для кожного стану схеми записуємо під відповідними вершинами графа.

Синтез схеми полягає у записі умов вмикання і скидання кожного тригера. Для цього окреслюють замкненою лінією всі стани на графі переходів, в яких значення вихідного сигналу даного тригера дорівнює одиниці. Вхідні сигнали схеми, позначення яких вказані на ребрах, що заходять в одержану замкнуту область, встановлюють тригер в стан 1, а вхідні сигнали на ребрах, що виходять з цієї області, скидають тригер в стан 0.

Умови вмикання тригера записуються у вигляді добутку сигналу на ребрі, що заходить в область, і сигналів решти тригерів, стан яких не змінюється при переході, позначеному ребром. Наприклад, якщо сигнал на ребрі, що заходить в область з одиничним значенням вихідного сигналу тригера p_1, дорівнює a, а тригери P_2 і P_3

не перемикаються, а зберігають стан $p_2 = 1$, $p_3 = 0$, то умова вмикання тригера P_1 записується у вигляді

$$S_{P_1} = ap_2\overline{p_3}. \tag{4.2}$$

Якщо в замкнуту область входить кілька ребер, то умова вмикання тригера записується у вигляді суми добутків відповідних сигналів, складених для кожного ребра.

Умова скидання тригера записується аналогічно для кожного ребра, що виходить з даної області, і подається у вигляді формули R_{P_i}. Описану процедуру виконують для кожного тригера і визначають для них умови вмикання і скидання.

Застосувавши описану процедуру визначення умов вмикання і скидання тригерів, для графа переходів на рис. 4.65 отримаємо:

$$S_{P_1} = dp_2\overline{p_3}; \tag{4.3}$$

$$R_{P_1} = a\overline{p_2}p_3 + \overline{p_2}\,\overline{p_3}; \tag{4.4}$$

$$S_{P_2} = c\overline{p_1}p_3; \tag{4.5}$$

$$R_{P_2} = bp_1p_3; \tag{4.6}$$

$$S_{P_3} = a\overline{p_1}\,\overline{p_2} + t_2p_1p_2; \tag{4.7}$$

$$R_{P_3} = t_1\overline{p}_1p_2 + \overline{a}p_1\overline{p}_2. \tag{4.8}$$

Формули для вихідних сигналів f_1 і f_2 записуються як комбінаційні функції вихідних сигналів тригерів P_1, P_2, P_3. Дійсно, функція f_1 =1 в станах 2 і 4, тобто в станах, яким відповідають такі комбінації значень вихідних сигналів тригерів: $p_1 = 0$, $p_2 = 0, p_3 = 1$ і $p_1 = 0, p_2 = 1, p_3 = 0$. Тому

$$f_1 = \overline{p}_1\overline{p}_2p_3 + \overline{p}_1p_2\overline{p}_3. \quad (4.9)$$

Аналогічно, функція f_2=1 в стані 6 ($p_1p_2p_3 = 111$), тобто

$$f_2 = p_1p_2p_3. \quad (4.10)$$

Таймер T_1 вмикається в стані 3 ($p_1p_2p_3 = 011$), а таймер T_2 – в стані 5 ($p_1p_2p_3 = 110$), тому

$$T_1 = \overline{p}_1p_2p_3. \quad (4.11)$$

$$T_2 = p_1p_2\overline{p}_3. \quad (4.12)$$

Розглянемо інший приклад. Керування двома двигунами M_1 і M_2 здійснюється за допомогою кнопок «Пуск», «Перший двигун», «Другий двигун» і «Стоп». При натисканні кнопки «Пуск» вмикається двигун M_1, потім за допомогою кнопок «Перший двигун» та «Другий двигун» можна перемикати, який двигун буде працювати. Кнопка «Стоп» вимикає роботу першого або другого двигуна і зупиняє роботу поки не буде натиснута кнопка «Пуск».

Вводимо позначення:

– M_1 – перший двигун;

– M_2 – другий двигун;

– a – кнопка «Пуск»;

– b – кнопка «Стоп»;

– S_1 – кнопка «Перший двигун»;

– S_2 – кнопка «Другий двигун».

Графоперехід за умовами роботи схеми представлений на рис. 4.67.

Запишемо рівняння, які визначають умови вмикання тригерів:

$$S_{P_1} = a\overline{p_2}; \quad (4.12)$$

$$S_{P_2} = S_2p_1. \quad (4.13)$$

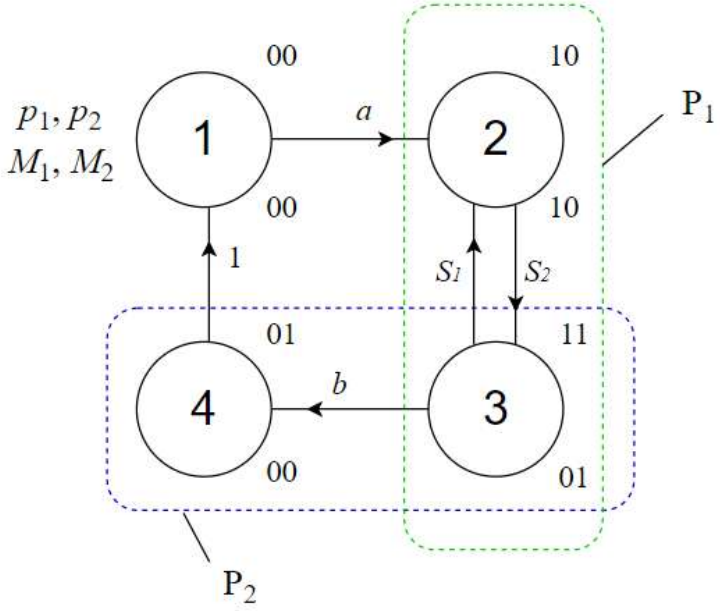


Рисунок 4.67 – Графоперехід за умовами роботи схеми

Умови скидання тригерів:

$$R_{P_1} = bp_2 + b\overline{p}_2 = b; \tag{4.14}$$

$$R_{P_2} = \overline{p}_1 + S_1 p_1. \tag{4.15}$$

Формули для вихідних змінних:

$$M_1 = p_1\overline{p_2}; \tag{4.16}$$

$$M_2 = p_1 p_2. \tag{4.17}$$

За результатами синтезу схеми отримаємо наступне поєднання контактів, що зображено на рис. 4.68.

Для проведення експерименту використаємо віртуальний макет штампувального автомату (рис. 4.69). В якості кнопки «Пуск» застосуємо кнопку *XS_key1* макету, а для кнопки «Стоп» візьмемо *XS_key4*. Кнопками запуску двигунів M_1 та M_2 призначимо *XS_key2* та *XS_key3* відповідно.

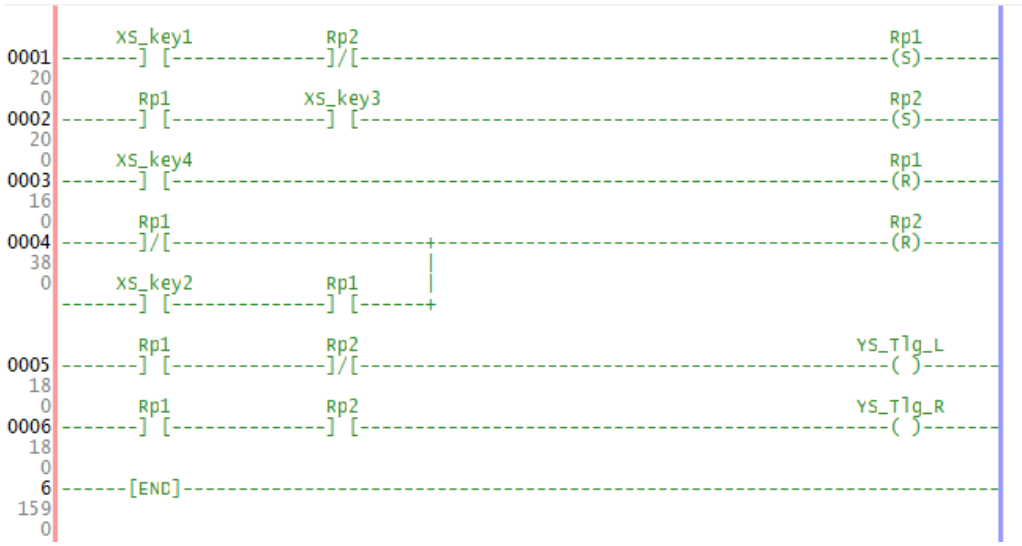


Рисунок 4.68 – Контактна схема рішення другої задачі

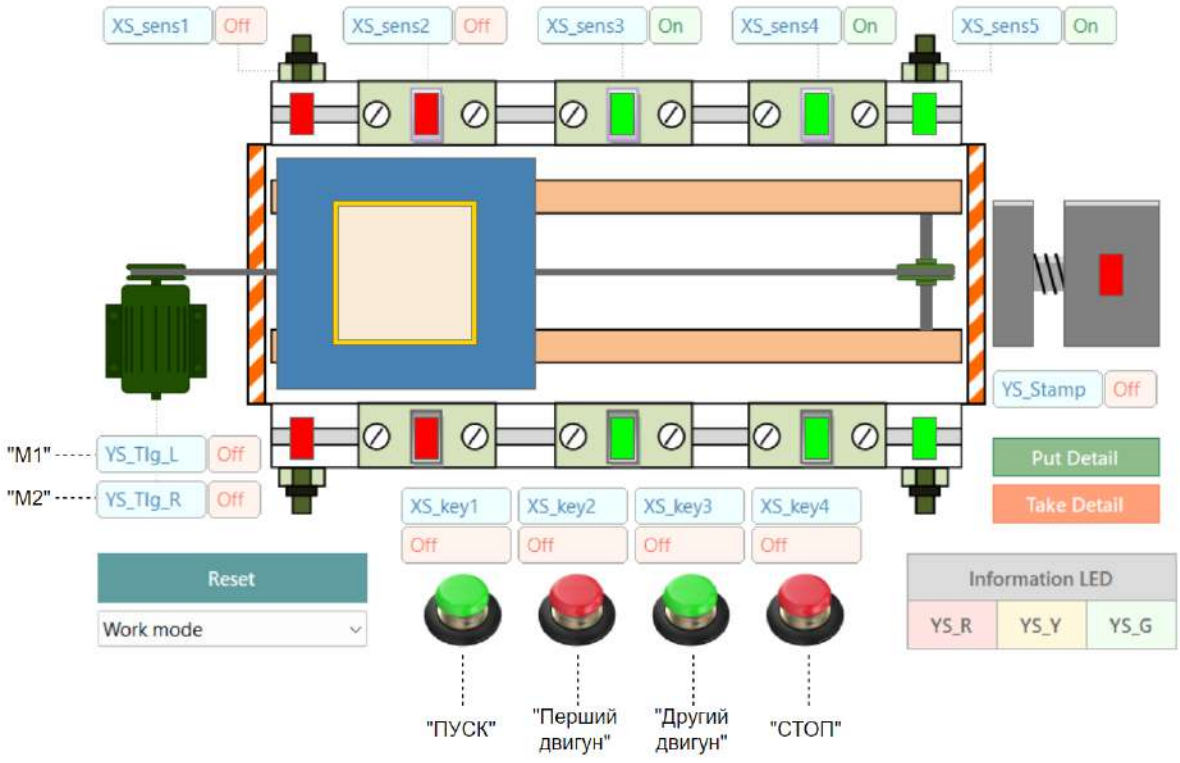


Рисунок 4.69 – Принцип призначення контактів на схемі віртуального макету штампувального автомата

За напрям руху теліжки відповідатимуть контакти реле *YS_Tlg_L* та *YS_Tlg_R*, що відповідають двигунам M_1 та M_2.

Проміжні стани P_1 та P_2 реалізовані внутрішніми контактами реле *RP1* та *RP2* (рис. 4.68). Принцип призначення контактів на схемі віртуального макету подано на рис. 4.69.

Як ще один приклад розглянемо керування дозуючим пристроєм, поданого на рис. 4.70. У вихідному положенні ємність порожня і всі крани перекриті. Кнопка «Пуск» запускає роботу пристрою на один цикл. З надходженням сигналу *a* кнопки «Пуск», спрацьовує датчик першого крану K_1 і у дозуючий пристрій наливається рідина 1 поки не спрацює датчик d_1 – датчик потрібного об'єму рідини 1. Після спрацьовування датчика, K_1 закривається і відкривається K_3, через який рідина 1 зливається у ємність. Після спрацьовування датчика d_0, який показує, що дозуючий пристрій порожній, відкривається кран K_2, що наповнює пристрій рідиною 2 до рівня d_2. Після цього рідина 2 також зливається у ємність. Якщо кнопка «Пуск» натиснута, то цикл повторюється спочатку, якщо ні, то повертаємося у початкове положення.

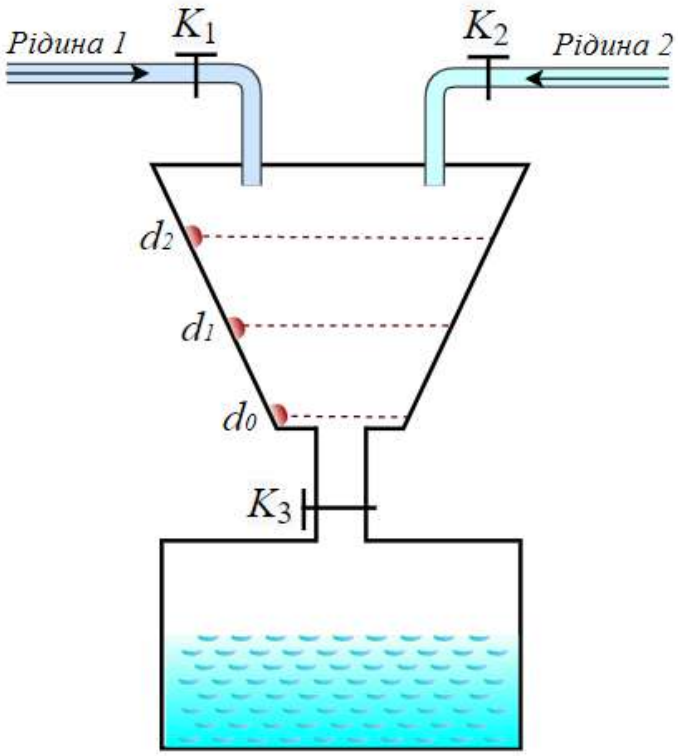


Рисунок 4.70 – Дозуючий пристрій

Таким чином, станів, в яких перебуває система, буде шість:

– 1 – початковий ($K_1 = 0$, $K_2 = 0$, $K_3 = 0$);

– 2 – наливається рідина 1, $K_1 = 1$;

– 3 – виливається рідина 1, $K_3 = 1$;

– 4 – наливається рідина 2, $K_2 = 1$;

– 5 – виливається рідина 2, $K_3 = 1$;

– 6 – повернення у початковий стан ($K_1 = 0$, $K_2 = 0$, $K_3 = 0$).

Графоперехід за умовою задачі подано на рис. 4.71.

Запишемо рівняння які визначають умови вмикання тригерів:

$$S_{P_1} = d_2\overline{p_2}p_3; \tag{4.18}$$

$$S_{P_2} = a\overline{p_1}\,\overline{p_3} + \overline{d_0}p_1p_3; \tag{4.19}$$

$$S_{P_3} = d_1\overline{p_1}p_2; \tag{4.20}$$

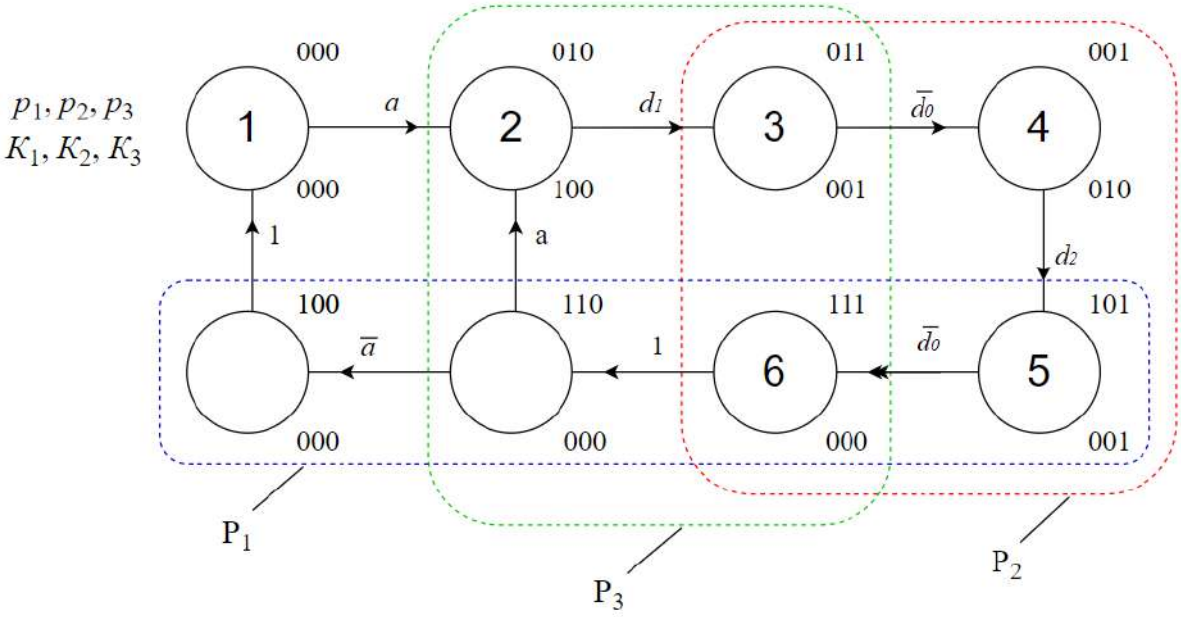


Рисунок 4.71 – Графоперехід за умовою задачі про дозуючий пристрій

Умови скидання тригерів:

$$R_{P_1} = ap_2\,\overline{p_3} + \overline{p_2}\,\overline{p_3};\tag{4.21}$$

$$R_{P_2} = \overline{a}\,p_1\,\overline{p_3} + \overline{d_0}\,\overline{p_1}\,p_3;\tag{4.22}$$

$$R_{P_3} = p_1 p_2;\tag{4.23}$$

Формули для вихідних змінних:

$$K_1 = \overline{p_1}\,p_2\,\overline{p_3};\tag{4.24}$$

$$K_2 = \overline{p_1}\,\overline{p_2}\,p_3;\tag{4.25}$$

$$K_3 = \overline{p_1}\,p_2\,p_3 + p_1\,\overline{p_2}\,p_3;\tag{4.26}$$

Принцип призначення контактів на схемі віртуального макету дозуючого пристрою подано на рис. 4.72.

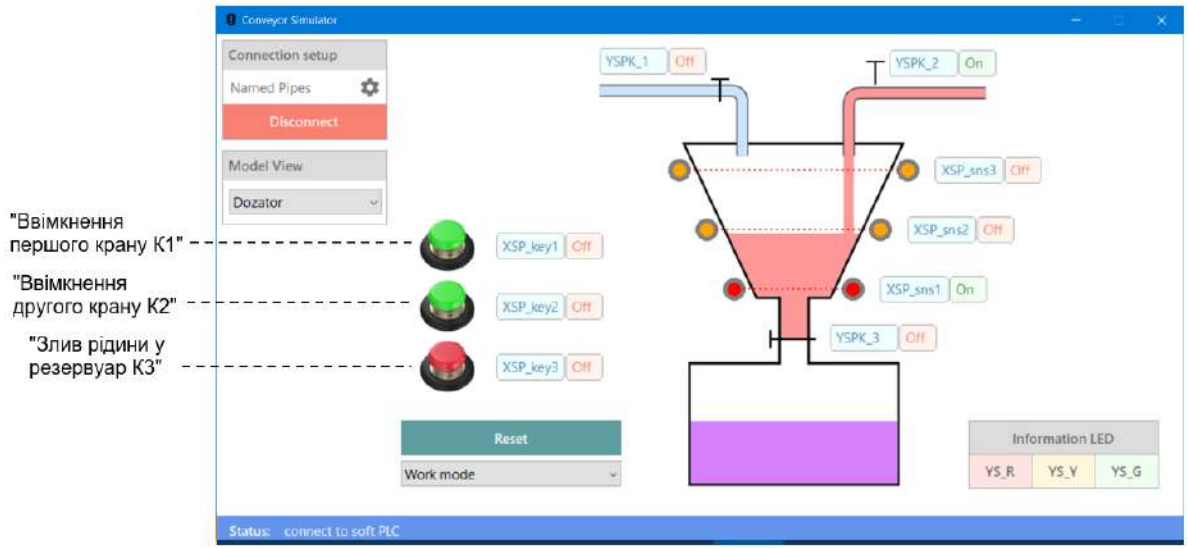


Рисунок 4.72 – Принцип призначення контактів на схемі віртуального макету дозуючого пристрою

Після отримання формул, що описують стани тригерів та вихідних реле можемо виконати синтез керуючої програми мовою LD (рис. 4.73).

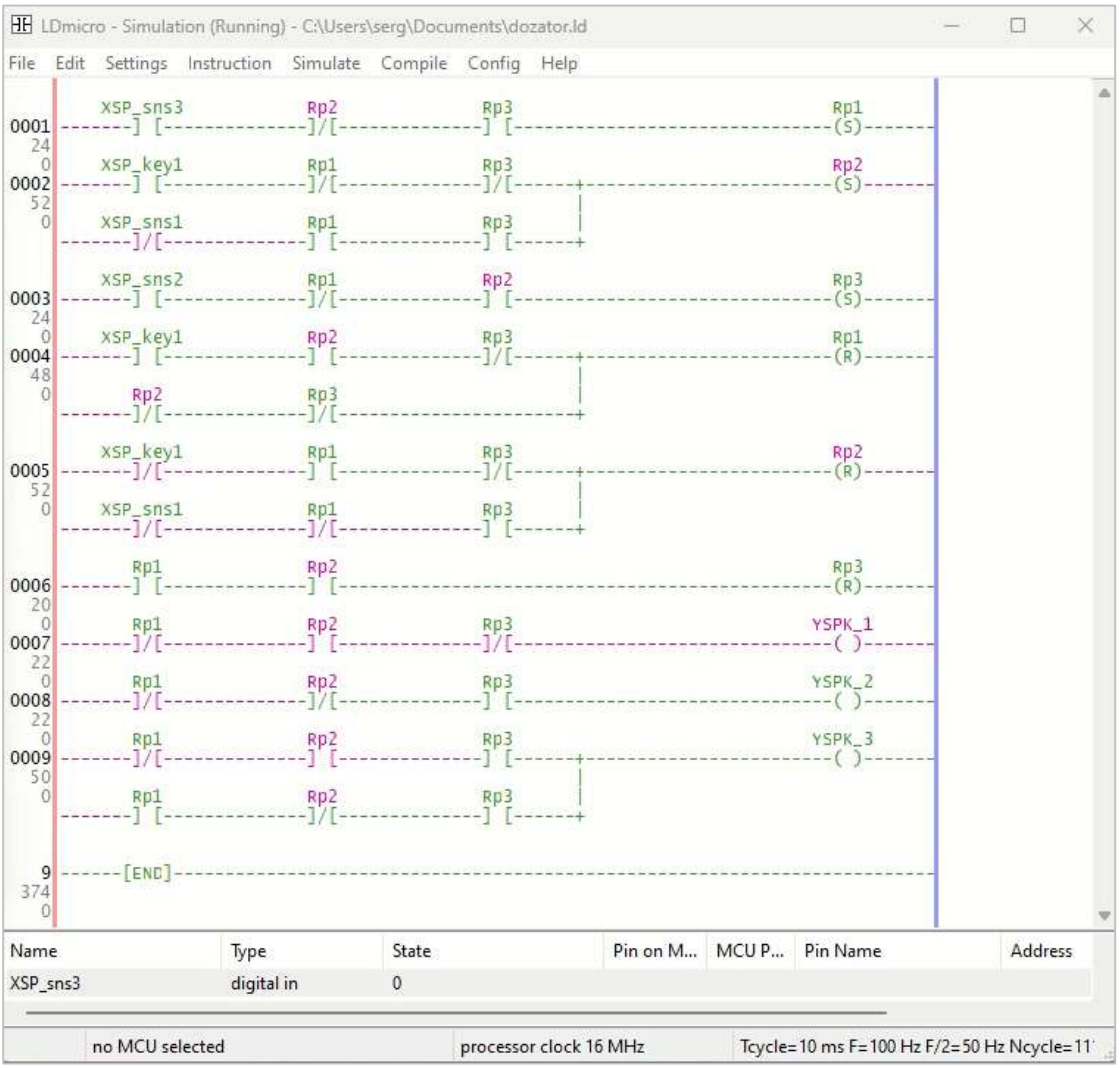


Рисунок 4.73 – Синтезована програма управління дозатором

Створення діаграми відбувається наступним чином:

– результату обчислення формули відповідає вихідне реле;

– стан вихідного реле обирається в залежності від умови вмикання тригеру (S_{P_x}– відповідає стану «Set» (--(S)--), R_{P_x}– «Reset» (--(R)--));

– права частина формули відповідає комбінації станів вхідних контактів;

– змінна без верхньої риски (наприклад, d_2) означає, що використовуються нормально відкриті контакти (--] [--);

– змінна з верхньою рискою (наприклад, $\overline{p_2}$) означає, що використовуються нормально закриті контакти (--] / [--);

– якщо в формулі зустрічається комбінація змінних, що перемножуються, то використовується послідовне поєднання відповідних контактів (рис. 4.73, рядок 1);

– якщо в формулі між комбінаціями змінних, використовується додавання, то використовується паралельне поєднання відповідних контактів (рис. 4.73, рядок 2).

Всі реле є внутрішніми, тому в позначені вони мають першу літеру «R». Всі змінні, що відображають стан відповідного реле представлені внутрішніми контактами. Наприклад, в формулі 4.18 змінна $\overline{p_2}$ представлена контактом Rp2 (рис. 4.73, рядок 1).

Назва зовнішніх кнопок або датчиків обирається в залежності від того, який елемент з інтерфейсу віртуального приладу застосовується для реалізації певної функції. Наприклад, кнопка «Пуск», що представлена в формулі 4.19 змінною «a» в синтезованій програмі реалізована у вигляді зовнішнього контакту «XSP_key1» (рис. 4.73, рядок 2). Вихідні змінні $К_1$…$К_3$, що описуються формулами (4.24)-(4.26) представлені в програмі у вигляді вихідних реле «YSPK_1»…«YSPK_3», відповідно. На рис. 4.74-4.77 подано різні стадії роботи віртуального макету в процесі виконання технологічної програми.

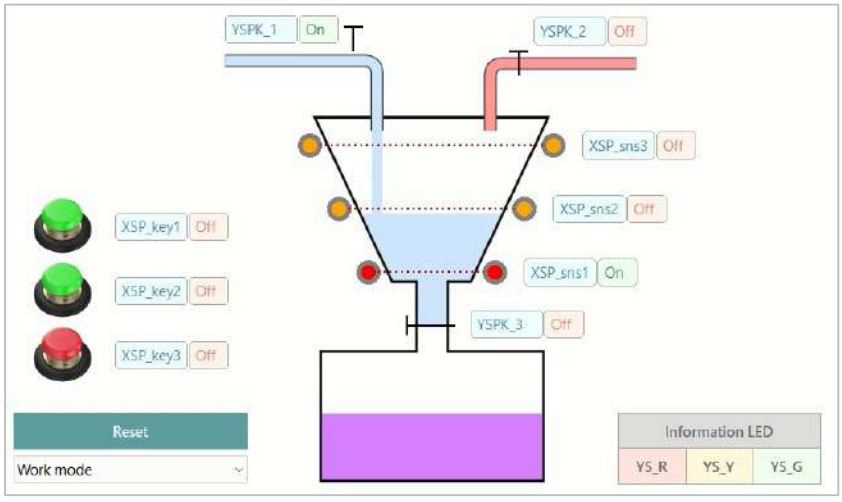


Рисунок 4.74 – Стадія наливу в дозатор синьої рідини до рівня датчика «XP_sns2»

Рис. 4.74 демонструє стадію наливу в дозатор синьої рідини до рівня датчика «XP_sns2». На рис. 4.75 подано стадія зливання відміряної синьої рідини в ємність. В даному стані відкритим є клапан «YSPK_3».

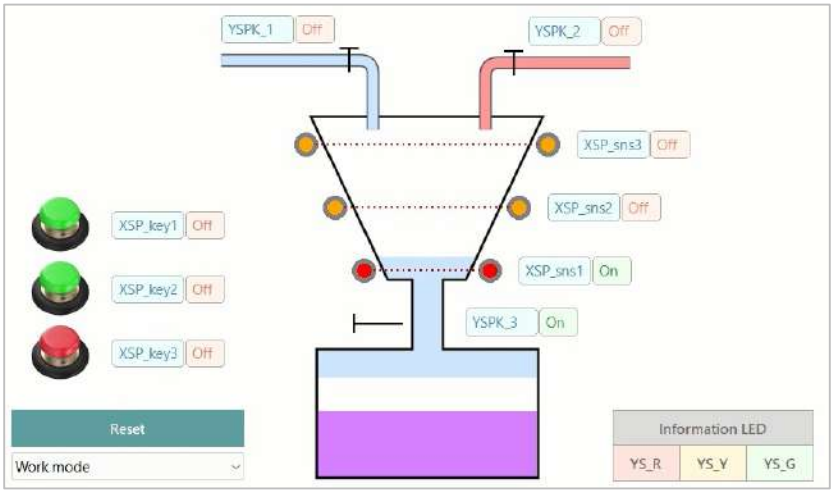


Рисунок 4.75 – Стадія зливання відміряної синьої рідини в ємність

На рис. 4.76 подано стадія наливу в дозатор червоної рідини до рівня датчика «XP_sns3».

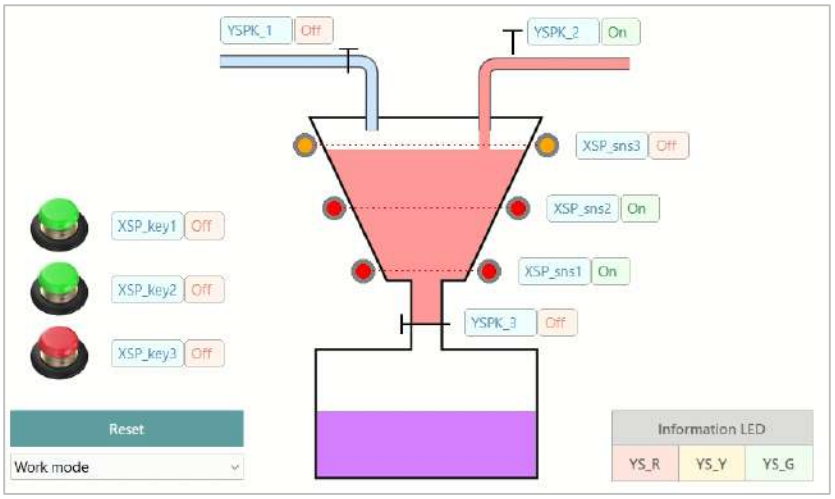


Рисунок 4.76 – Стадія наливу в дозатор червоної рідини
до рівня датчика «XP_sns3»

Рис. 4.77 демонструє стадію зливання відміряної червоної рідини в ємність. В даному стані відкритим є клапан «YSPK_3».

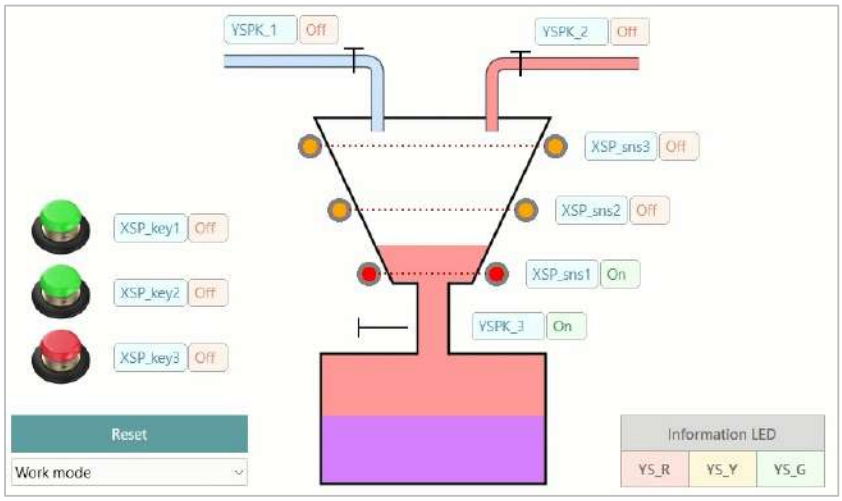


Рисунок 4.77 – Стадія зливання відміряної червоної рідини в ємність

4.5 Контрольні питання

1. Для чого призначена мова релейно-контактної логіки?
2. Який вигляд має східчаста діаграма? Як налаштувати тип вхідних і вихідних контактів?
3. Як позначаються релейні елементи на LD-діаграмі?
4. Назвіть основні логічні функції і запишіть співвідношення для них.
5. Наведіть приклад створення логічного елементу НІ засобами LD.
6. Наведіть приклад створення логічного елементу І засобами LD.
7. Наведіть приклад створення логічного елементу АБО засобами LD.
8. Що таке асинхронний тригер? Поясніть принцип його роботи.
9. Що таке карта Карно? Який принцип їх використання?
10. Як побудувати граф переходів для певної кількості тригерів?

5 ТАЙМЕРИ ТА ЕЛЕМЕНТИ ЗАТРИМКИ В LDMICRO ТА LD

5.1 Основні відомості

Таймери застосовуються для реалізації послідовності синхронізації серед яких можна виділити такі:

– інтервали очікування та спостереження;

– вимірювання параметрів імпульсів;

– генерація імпульсів.

В програмі LDmicro для роботи з імпульсами застосовуються такі інструкції:

– TON (TURN-ON DELAY);

– TOF (TURN-OFF DELAY);

– THI (TIMER HIGH);

– TLO (TIMER LOW);

– RTO (RETENTIVE TIMER);

– RTL (RETENTIVE TIMER LOW);

– TCY (CYCLIC TIMER).

5.1.1 Таймер із затримкою вмикання TON (TURN-ON DELAY)

Коли сигнал, що надходить до інструкції TON, змінюється з **false** на **true**, вихідний сигнал залишається **false** протягом заданого терміну, перш ніж стати істинним. Коли сигнал, що надходить на вхід інструкції TON, змінюється з істинного на хибний, вихідний сигнал негайно стає **false**. Таймер скидається кожного разу, коли вхід стає хибним, тому вхідний сигнал має залишатися істинним протягом заданого терміну часу, перш ніж вихід стане істинним (рис. 5.1).

Приклад роботи таймеру TON показаний на рис. 5.2. Наприклад, в якості параметру часу затримки вмикання зазначимо TON = 100 мс. Тоді, коли на вхід інструкції TON надходить сигнал високого рівня, то на виході відповідний сигнал з’явиться тільки тоді, коли час затримки T_{wait} = TON спливе. Сигнал високого рівня буде утримуватись на виході протягом решти часу T_{input}. У випадку, що зображено на рис. 5.1, $T_{output} = T_{input} – T_{wait}$ = 100 мс.

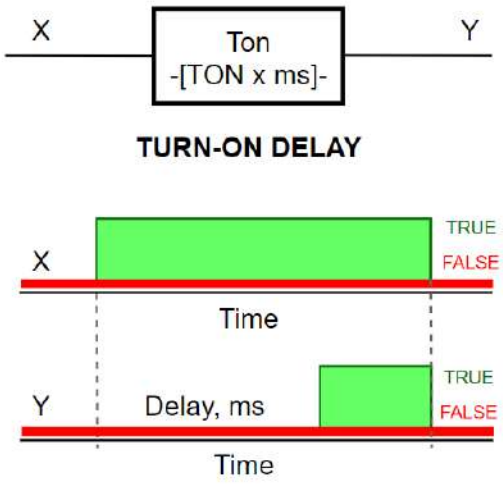


Рисунок 5.1 – Таймер із затримкою вмикання TON

Змінна «Tname», яка відповідає інструкції TON, відраховує час від початку детектування зміни рівня сигналу на вході. Інструкція TON виводить **true**, коли змінна лічильника більша або дорівнює заданій затримці. Змінною лічильника можна керувати в будь-якому місці програми, наприклад, за допомогою інструкції MOV.

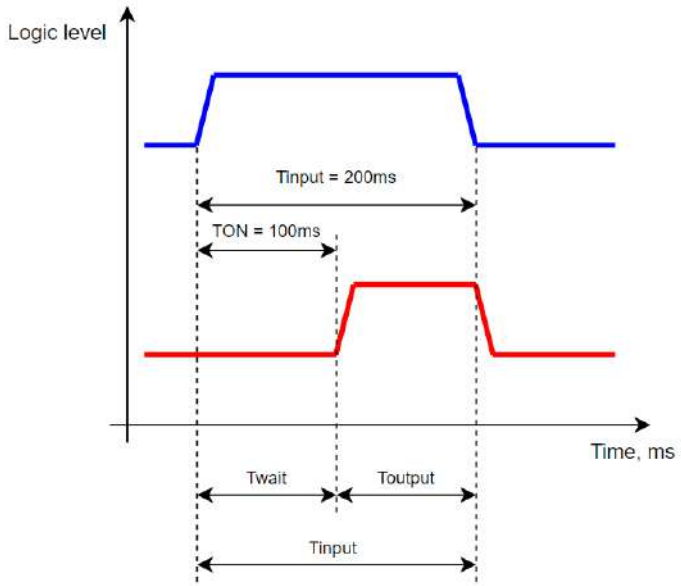


Рисунок 5.2 – Приклад роботи таймеру TON

Час затримки налаштовується за допомогою відповідного діалогового вікна (рис. 5.3).

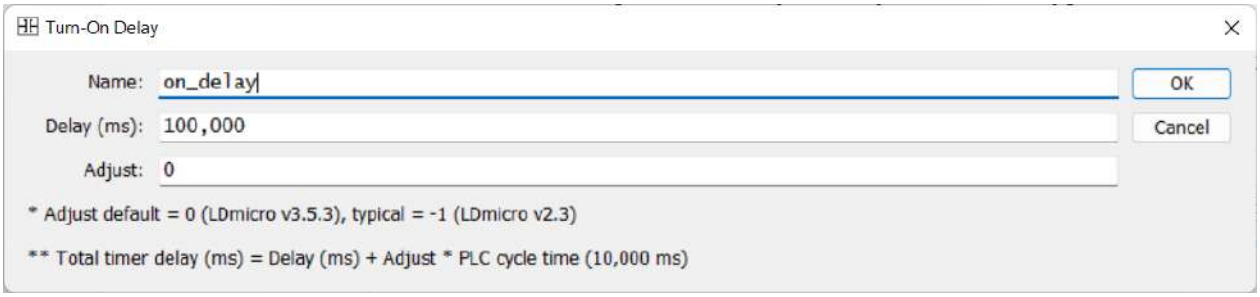


Рисунок 5.3 – Налаштування параметрів таймеру TON

Параметр «Name» задає найменування таймеру та відповідної до нього змінної. За цією назвою, із додаванням префіксу «T», можна звернутись до даної змінної із програми, наприклад, «Ton_delay».

Параметр «Delay (ms)» задає час очікування в мілісекундах. Наприклад, за замовчуванням встановлено значення 100 мс.

Параметр «Ajust» дозволяє більше точно налаштувати таймер за рахунок додаткового зміщення часу на вказану величину. За замовчуванням цей параметр дорівнює нулю та не враховується. Якщо параметр «Ajust» не дорівнює нулю, то загальний час затримки розраховується за формулою:

$$\mathrm{TON} = \mathrm{Tdelay} + \mathrm{Tadjust} \times \mathrm{PLCcycle} \qquad (5.1)$$

де TON – час загальної затримки, мс;

Tdelay – час, що вказано в параметрі «Delay», мс;

Tadjust – значення в полі «Adjust»;

PLCcycle – час машинного циклу, що вказано у відповідному параметрі налаштування мікроконтролеру, мс.

Приклад використання інструкції TON в програмі подано на рис. 5.4. В даному прикладі світлодіод Yled запалиться через одну секунду після натискання та утримання в такому положенні кнопки «Xbutton», тобто із затримкою 1000 мс.

Рисунок 5.4 – Приклад використання інструкції TON в програмі

5.1.2 Таймер із затримкою вимкнення TOF (TURN-OFF DELAY)

Коли сигнал, що надходить до інструкції TOF, змінюється з істинного на хибний, вихідний сигнал залишається істинним протягом заданого часу, перш ніж стати хибним. Коли сигнал, що надходить в інструкцію, змінюється з **false** на **true**, вихідний сигнал негайно стає істинним. Таймер скидається кожного разу, коли вхід стає **false**, тому вхідний сигнал має залишатися хибним протягом заданого терміну часу, перш ніж вихід стане хибним (рис. 5.5).

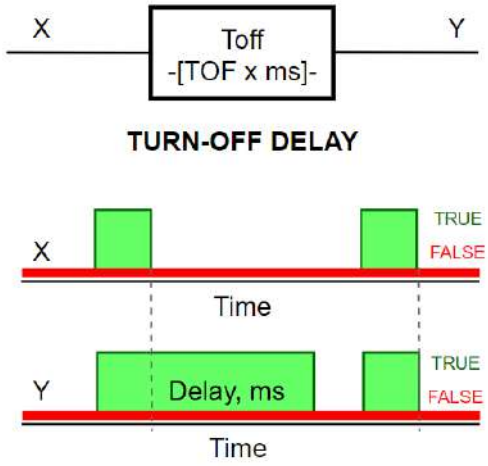


Рисунок 5.5 – Таймер із затримкою вимкнення TOF

Змінна «Tname», що відповідає інструкції TOF, відраховує від нуля час затримки. Інструкція TON виводить **true**, коли змінна лічильника більша або дорівнює заданій затримці. Змінною лічильника можна керувати в будь-якому іншому місці програми, наприклад, за допомогою інструкції MOV.

Приклад роботи таймеру TOF подано на рис. 5.6.

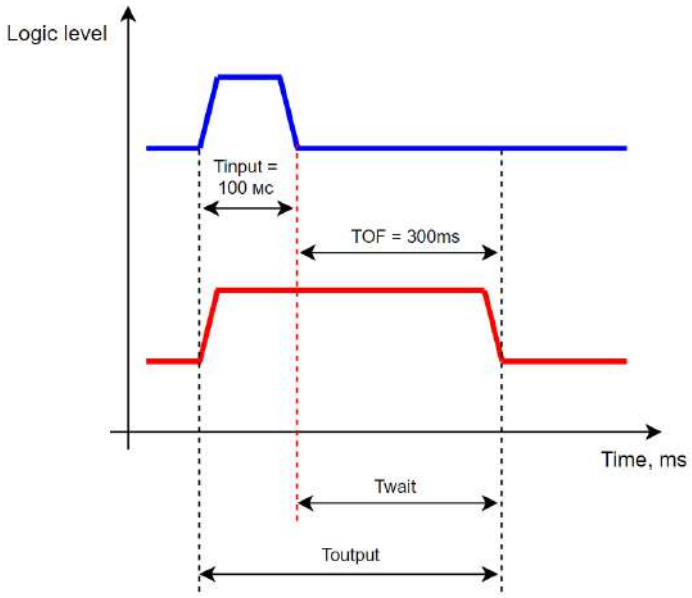


Рисунок 5.6 – Приклад роботи таймеру TOF

Наприклад, в якості параметру часу затримки вмикання зазначимо TOF = 300 мс. Тоді, коли на вході інструкції TOF вхідний сигнал із високого рівня переходить в низький, то на виході даної інструкції сигнал залишається в стані **true** ще певний термін часу (в нашому прикладі 300 мс). Вихідний сигнал зміниться на **false** після спливу часу затримки T_{wait} = TOF. Сигнал високого рівня буде утримуватись на виході протягом часу T_{input}. У випадку, що зображено на рис. 5.6, $\mathrm{Toutput} = \mathrm{Tinput} + \mathrm{Twait} = 400$ мс.

Час затримки налаштовується за допомогою відповідного діалогового вікна (рис. 5.7).

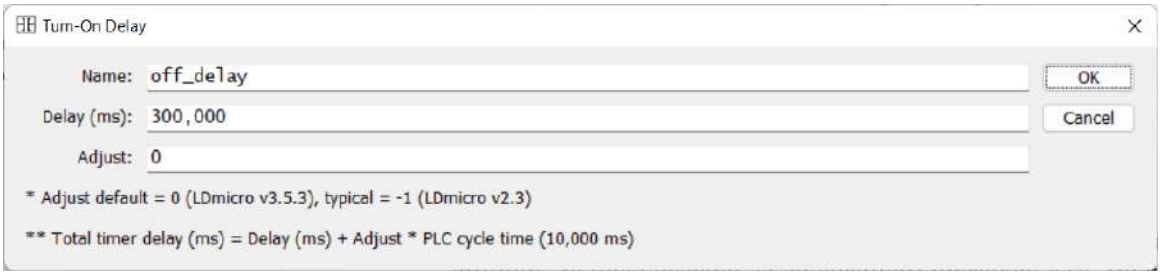


Рисунок 5.7 – Налаштування параметрів таймеру TOF

Параметр «Name» задає найменування таймеру та відповідної до нього змінної. За цією назвою, із додаванням префіксу «T», можна звернутись до даної змінної із програми, наприклад, «Toff_delay».

Параметр «Delay (ms)» задає час очікування в мілісекундах. Наприклад, за замовчуванням встановлено значення 100 мс і в нашому прикладі його змінено на 300 мс.

Параметр «Ajust» дозволяє більш точно налаштувати таймер за рахунок додаткового зміщення часу на вказану величину. За замовчуванням цей параметр дорівнює нулю та не враховується. Якщо параметр «Ajust» не дорівнює нулю, то загальний час затримки розраховується за формулою (5.1).

5.1.3 Таймер фіксованого часу вмикання THI (TIMER HIGH)

Даний вид таймеру застосовується у випадку, коли необхідно формувати вихідні сигнали стандартної заданої тривалості (рис. 5.8).

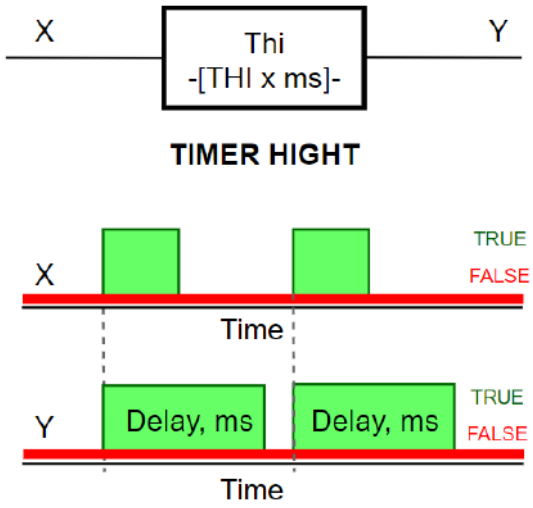


Рисунок 5.8 – Таймер фіксованого часу вмикання THI

Коли сигнал, що надходить до інструкції THI, змінюється з **false** на **true**, вихідний сигнал є істинним протягом заданого часу. Вихідний сигнал скидається через встановлений в параметрах налаштування час, навіть, якщо на вході й далі

залишається сигнал високого рівня. Для перезапуску внутрішнього таймеру необхідно, щоб вхідний сигнал змінився з **false** на **true**.

Приклад роботи таймеру THI подано на рис. 5.9.

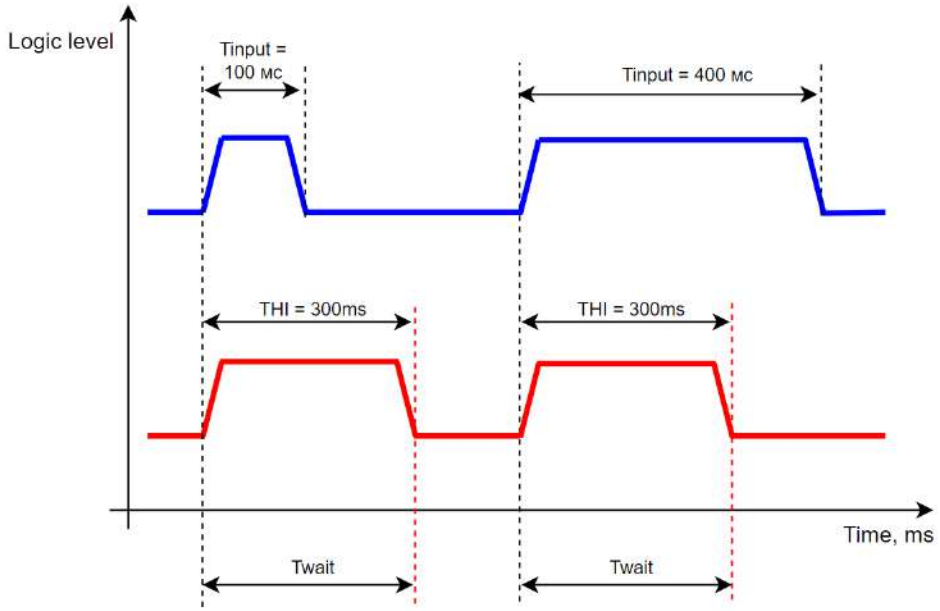


Рисунок 5.9 – Приклад роботи таймеру THI

Із поданого прикладу можна бачити, що вихідний сигнал завжди має фіксовану тривалість T_{wait} в незалежності від тривалості вхідного сигналу. Наприклад, якщо тривалість вхідного сигналу менша за T_{wait} ($Tinput = 100$ мс), або більша ($Tinput = 400$ мс) вихідний сигнал завжди відповідає заданій тривалості $THI = 300$ мс.

Значення часу затримки налаштовується за допомогою відповідного діалогового вікна. Задається ім’я змінної «Name», час очікування в мілісекундах «Delay (ms)» та параметр «Ajust» для більш точного налаштування таймеру.

5.1.4 Таймер фіксованого часу вимкнення TLO (TIMER LOW)

Таймер TLO має зворотний по відношенню до THI ефект. Таймер застосовується, якщо необхідно формувати сигнали заданої тривалості зворотної полярності (рис. 5.10).

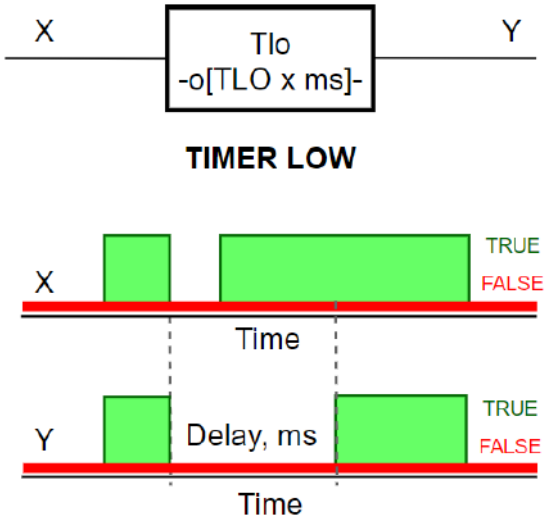


Рисунок 5.10 – Таймер фіксованого часу вмикання зворотної полярності TLO

Приклад роботи таймеру TLO подано на рис. 5.11.

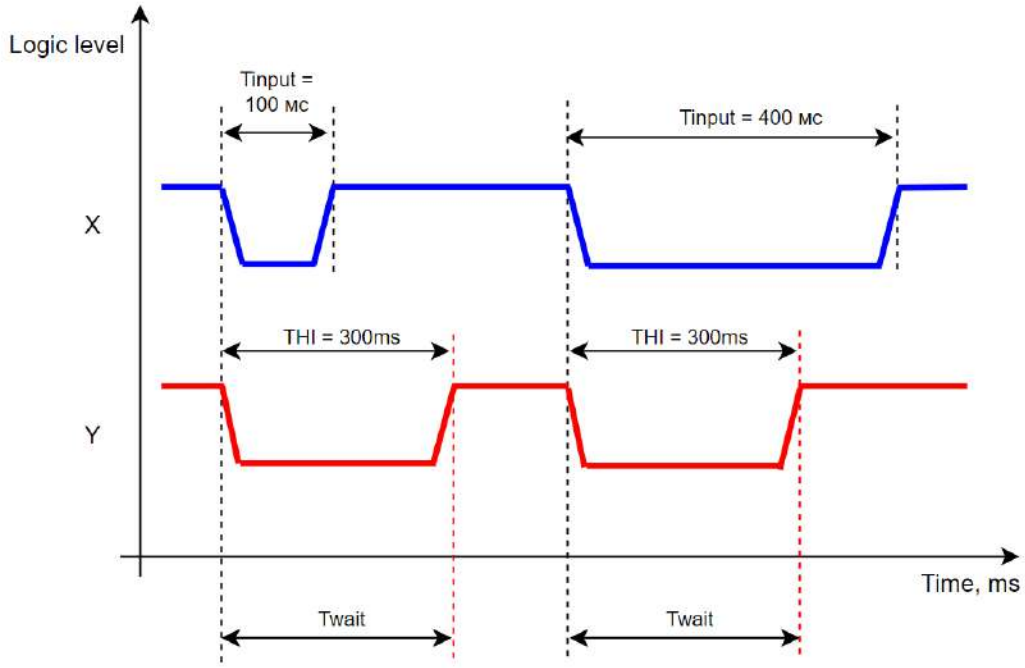


Рисунок 5.11 – Приклад роботи таймеру TLO

Коли сигнал, що надходить до інструкції TLO, змінюється з істинного на хибний, вихідний сигнал є хибним протягом заданого часу. Вихідний сигнал встановлюється істинним через заданий термін T_{wait}. Внутрішній лічильник таймера скидається через встановлений термін, або за умовою, якщо вхідний сигнал є істинним.

Затримка налаштовується за допомогою відповідного діалогового вікна. Значення часу затримки налаштовується за допомогою відповідного діалогового вікна. Задається ім'я змінної «Name», час очікування в мілісекундах «Delay (ms)» та параметр «Ajust» для більш точного налаштування таймеру.

5.1.5 Таймер з накопиченням часу вмикання RTO (RETENTIVE TIMER)

Ця інструкція відстежує, як довго вхідний сигнал був істинним (true). Таймер рахує загальний час, поки на вході знаходився сигнал логічної одиниці з моменту останнього скидання, або першого увімкнення, додаючи до загального результату кожен новий випадок встановлення значення **true** на вході. Умовне графічне зображення інструкції подано на рис. 5.12.

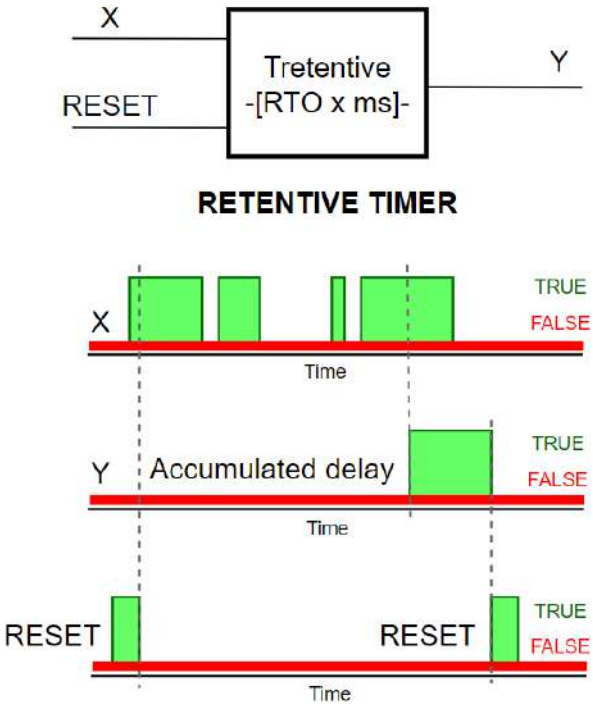


Рисунок 5.12 – Таймер з накопиченням часу вмикання RTO

З рис. 5.12 видно, що для роботи даного таймеру необхідно задіяти додатково ще одну лінію «RESET» для скидання накопичених значень часу вмикання входу «X». З моменту надходження сигналу скидання починається новий час затримки.

Часова діаграма роботи даного виду таймера подана на рис. 5.13. На даній діаграмі показано випадок, коли встановлено загальний час затримки 400 мс (Total delay time = 400 ms). Після надходження сигналу скидання (REST), починається стеження за станом входу «X».

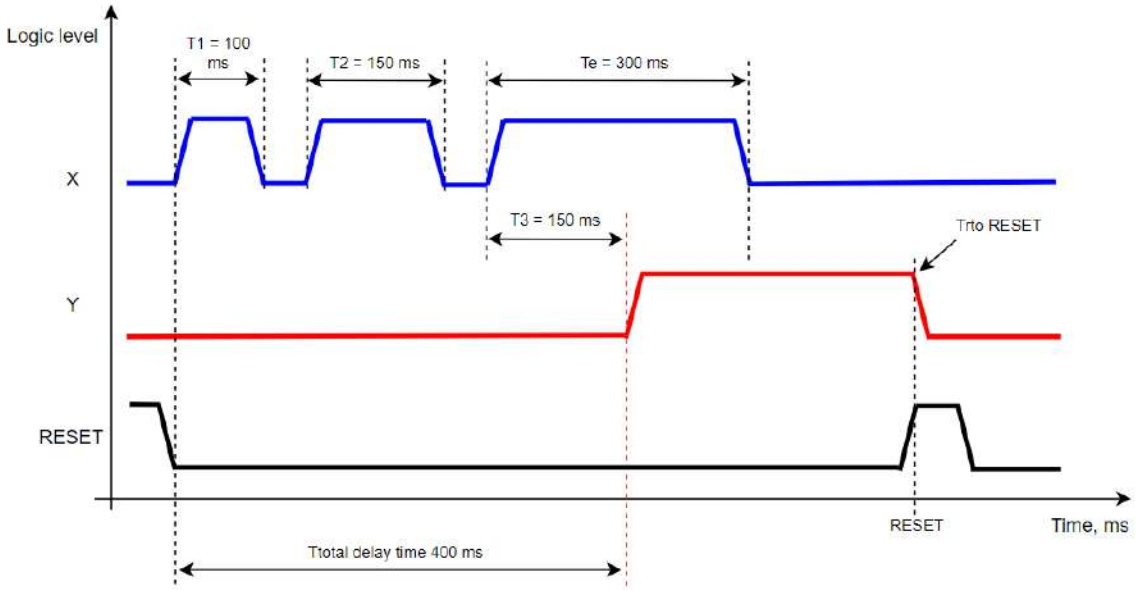


Рисунок 5.13 – Приклад роботи таймеру RTO

На діаграмі показано випадок, коли спочатку на вході «X» з'являється високий рівень, та утримується протягом 100 мс (T1 = 100 ms.). Внутрішній лічильник починає нарощувати значення загального часу прибування входу в стані логічної одиниці. Далі сигнал перемикається у стан логічного нуля. Такий стан таймером не сприймається і внутрішній лічильник не працює.

Далі знову на вхід «X» надходить сигнал високого рівня і тримається в такому стані протягом 150 мс (T2 = 150 ms.). За цей час внутрішній лічильник додає до загального стану ці 150 мс, та коли вхідний сигнал знов перемикається в логічний нуль, сумарне значення становить 250 мс.

Останній випадок, що зображено на рис. 5.13, демонструє ситуацію, коли вхідний сигнал встановлюється в стан логічної одиниці, та не вимикається протягом 300 мс. (Te = 300 ms.). В такому випадку внутрішній лічильник додає до загальної кількості часу тільки решту 150 мс (T3 = 150 ms), та перемикає вихід «Y» у стан логічної одиниці, сигналізуючи про досягнення ліміту в задані 400 мс.

Таким чином, в ситуації, що зображена на рис. 5.13 загальний час очікування складається з трьох часових інтервалів T_1, T_2 та T_3:

$$Ttotal = T1 + T2 + T3 = 100 + 150 + 150 = 400 \text{ мс.}$$

Після того, як вихідний сигнал стане істинним, він залишиться таким навіть після того, як вхідний сигнал стане хибним, якщо сумарне значення вхідного сигналу перевищило задане значення. Тому цей таймер необхідно скинути вручну, використовуючи інструкцію скидання (RESET).

Для налаштування параметрів таймеру використовується діалогове вікно, в якому необхідно задати назву змінної «Name», час затримки «Delay (ms)», та зміщення «Adjust».

На рис. 5.14 подано приклад реалізації даного типу таймеру в програмі мовою LD.

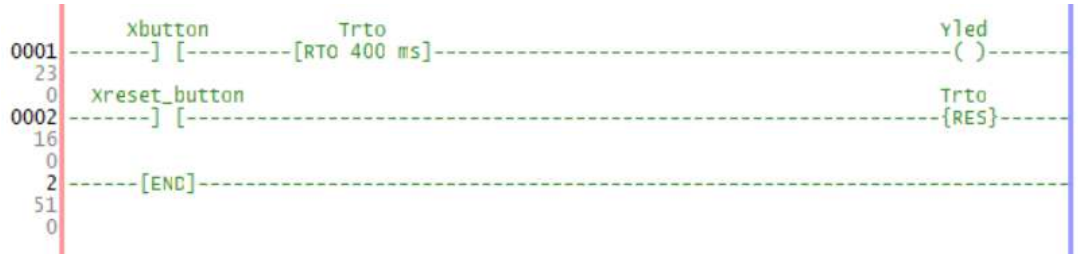


Рисунок 5.14 – приклад реалізації таймеру RTO в програмі

В першому рядку використовуються інструкції «Xbutton» для генерації вхідних сигналів, «Trto» для підрахунку загального часу спрацювання кнопки та «Yled» для індикації завершення процесу підрахунку часу спрацювання кнопки.

Другий рядок програми реалізує функцію скидання по натисканню кнопки «Xreset_button». Для цього застосовується спеціальна інструкція «RES». Ця інструкція обирається в розділі «Counter» головного меню (рис. 5.15).

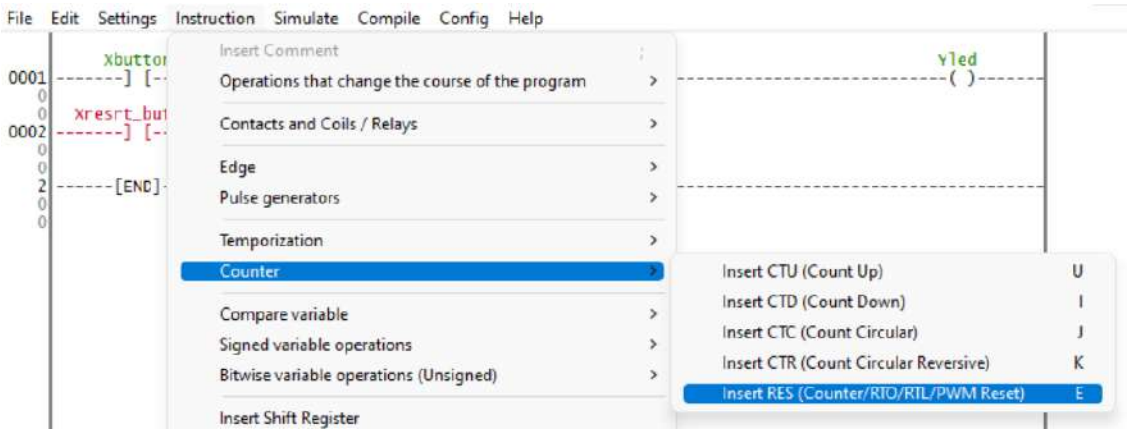


Рисунок 5.15 – Порядок обирання інструкції RES

Інструкції RES може застосовуватись для таймерів RTO і RTL, лічильників та генераторів імпульсів PWM. Після додавання інструкції в програму необхідно її налаштувати. Для цього використовується відповідне меню (рис. 5.16).

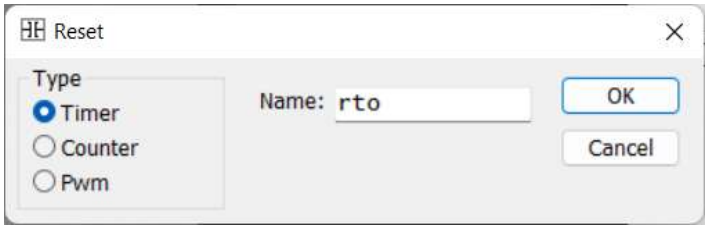


Рисунок 5.16 – Налаштування параметрів інструкції скидання RES

До введеного імені в полі «Name» буде додана відповідна літера в залежності від обраного типу об’єкту «Timer», «Counter» та «PWM».

5.1.6 Таймер з накопиченням часу вимкнення RTL (RETENTIVE TIMER LOW)

Цей таймер є аналогічним до таймеру RTO, але відстежує, як довго вхідний сигнал був хибним (false). Таймер рахує загальний час поки на вході знаходився

сигнал логічного нуля з моменту останнього скидання, або першого вмикання, додаючи до загального результату кожен новий випадок встановлення значення **false** на вході. Умовне графічне зображення інструкції подано на рис. 5.17.

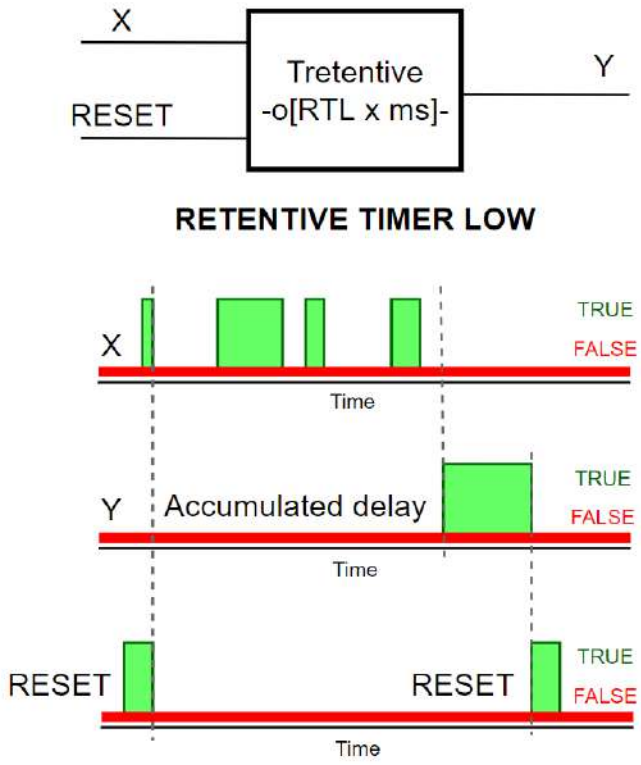


Рисунок 5.17 – Таймер з накопиченням часу вимкнення RTL

На відміну від таймеру RTO, в даному випадку враховується загальний час знаходження вхідної лінії в стані логічного нуля (**false**). Для переведення таймеру в початковий стан після закінчення рахування використовується додаткова лінія та інструкція «RESET».

Приклад часової діаграми роботи таймеру з накопиченням часу вимкнення RTL подано на рис. 5.18. Так само, як і в випадку з таймером RTO, рахується час знаходження вхідної лінії «X» в певному стані, в даному випадку в стані **false**. Загальний час накопичується та, коли його значення перевищить задане, на виході «Y» з’явиться логічна одиниця. В даному прикладі загальний час затримки до появи логічної одиниці на виході таймеру становить 400 мс.

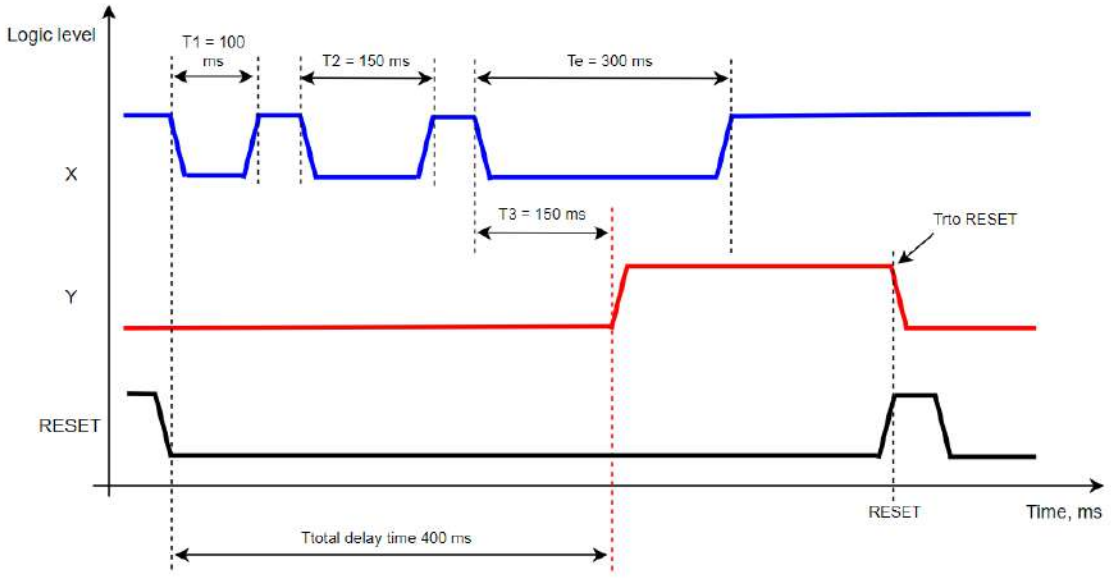


Рисунок 5.18 – Приклад часової діаграми роботи таймеру
з накопиченням часу вимкнення RTL

5.1.7 Циклічний таймер TCY (CYCLIC TIMER)

Ця інструкція використовується у випадку необхідності генерації послідовності прямокутних імпульсів з однаковою шириною позитивного та негативного рівнів (меандр) (рис. 5.19).

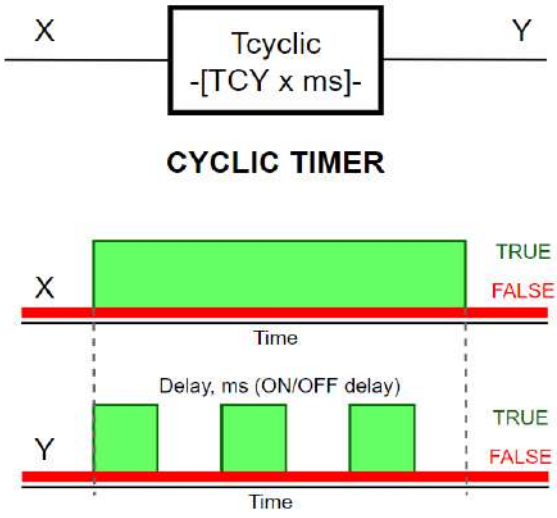


Рисунок 5.19 – Циклічний таймер TCY

Якщо вхідний сигнал істинний, ця інструкція створює меандр з періодом Delay (ms) + Delay (ms). Якщо сигнал, що надходить в інструкцію, є хибним, то вихідний сигнал є хибним. Приклад часової діаграми роботи циклічного таймеру TCY подано на рис. 5.20.

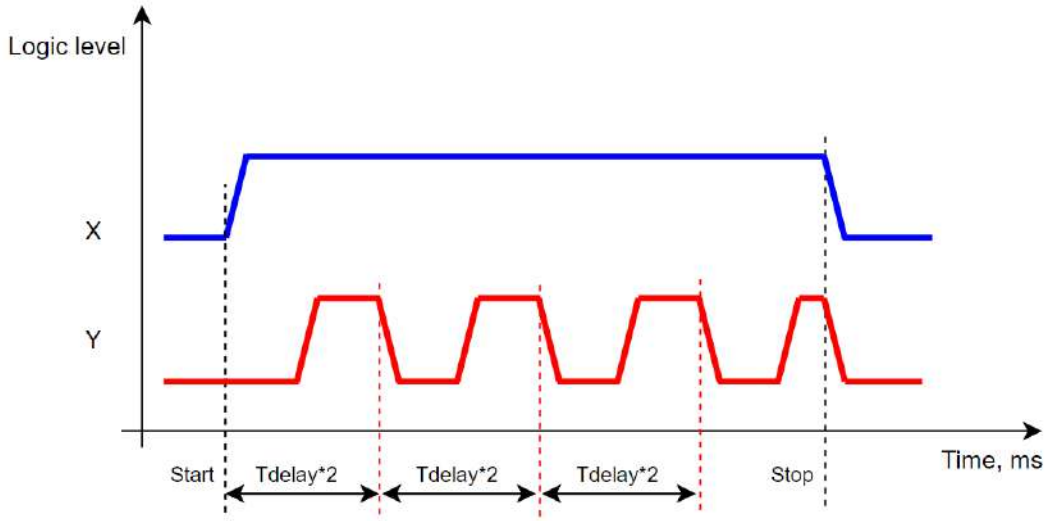


Рисунок 5.20 – Приклад часової діаграми роботи циклічного таймеру TCY

5.2 Приклади використання таймерів в технологічних програмах

5.2.1 Приклад застосування таймерів TON та TOF

Для створення автогенераторів в програмах LD найчастіше застосовується метод з можливостю налаштування часу знаходження в стані логічної одиниці та логічного нуля. Для цього використовуються дві інструкції TON і TOF розташовані послідовно. Приклад автогенератора, який виконано на елементах TON і TOF подано на рис. 5.21. В даному прикладі генератор побудовано в першому рядку, а в другому використовуються послідовності імпульсів для управління увімкненням світлодіоду.

Принцип дії генератора схожий з принципом роботи електромеханічного дзвоника. На рис. 5.22 подана схема електромеханічного дзвоника.

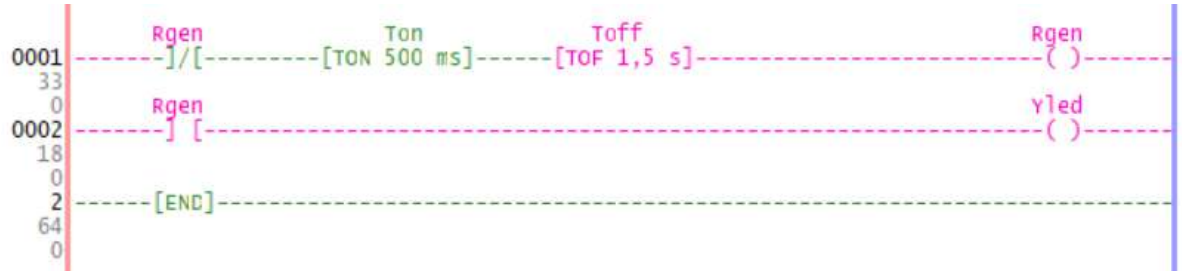


Рисунок 5.21 – Приклад автогенератора, який виконано на елементах TON і TOF

Після натискання на кнопку включення струм через замкнені рухомі контакти подається на котушку. Завдяки магнітній силі, що виникає у котушці, рухома пластина, що зроблена з феромагнітного матеріалу, притягується до її сердечника. Молоточок, розташований на кінці рухомої пластини, б'є по сталевій чашці, в результаті чого роздається звуковий сигнал.

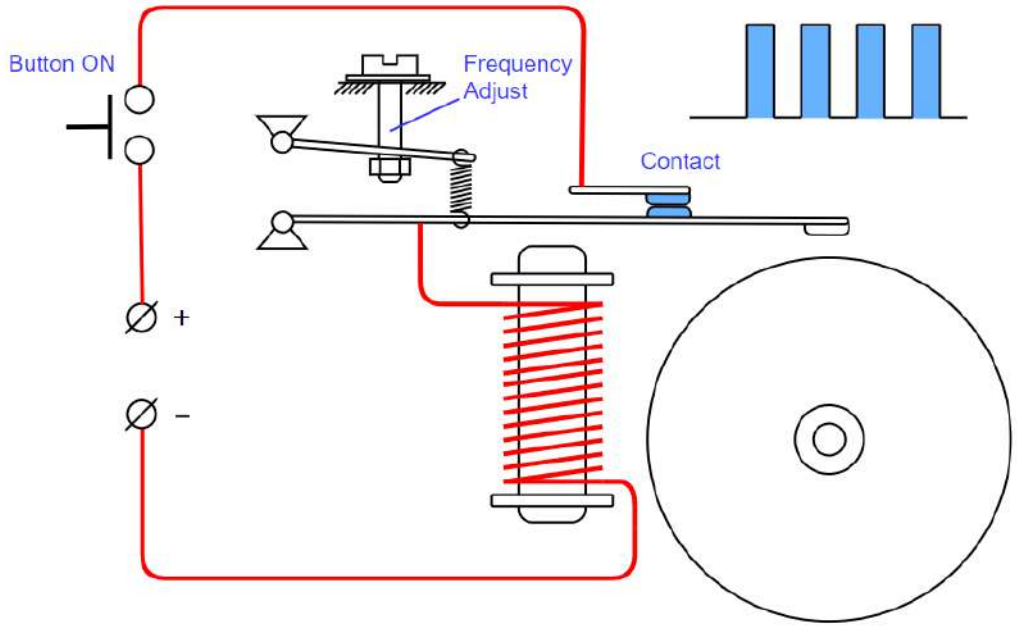


Рисунок 5.22 – Схема електромеханічного дзвоника

Після того, як рухома пластина притягується до сердечника котушки, зникає контакт з електричним з'єднанням і струм в ланцюзі переривається. Під впливом пружини рухома пластина повертається до початкового положення і електричний

контакт знову з’являється. Таким чином, процес зазначений вище повторюється. В результаті роботи такої електромеханічної системи ми одержуємо послідовність імпульсів, частоту якої в обмеженому діапазоні можна змінювати за допомогою регулюючого гвинта, який впливає на жорсткість пружини.

В програмі (рис. 5.21) роль пружини виконують елементи TON і TOF, а котушкою є елемент Coil, розташований справа в крайній позиції. Роль рухомого контакту виконує елемент Contact, розташований зліва в крайній позиції.

Для правильної роботи генератора необхідно, щоб контакт був включений в протилежному напрямку по відношенню до котушки реле, тому в налаштуванні даної інструкції обираємо тип увімкнення «Negated».

Для програми, що зображена на рис. 5.21 відповідною буде часова діаграма, подана на рис. 5.23. Тривалість логічної одиниці задається інструкцією TOF і складає 1,5 с. Тривалість логічного нуля визначається інструкцією TON і складає 0,5 с.

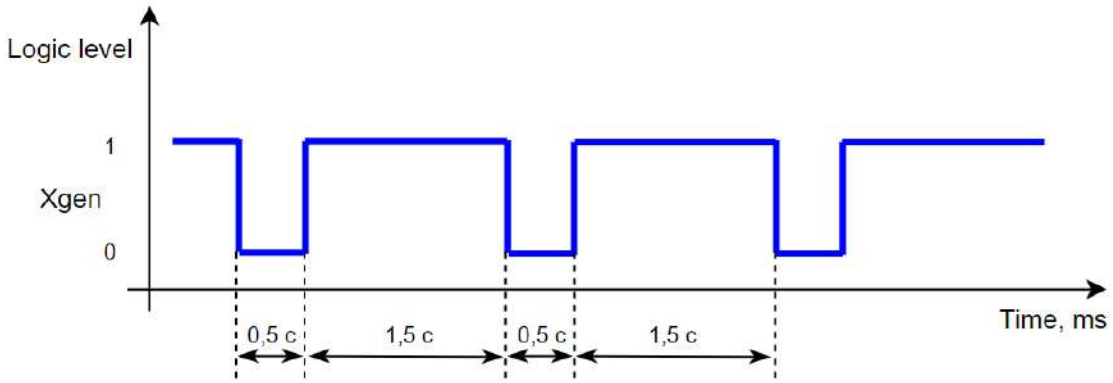


Рисунок 5.23 – Часова діаграма роботи автогенератора
з параметрами: TON = 0,5 с; TOF = 1,5 с

На рис. 5.24 зображено випадок, коли параметри інструкцій TON і TOF поміняли місцями, тобто значення Delay для інструкції TOF визначили, як 0,5 с, а значення Delay для інструкції TON – 1,5 с. В результаті роботи програми отримаємо часову діаграму, що подана на рис. 5.24.

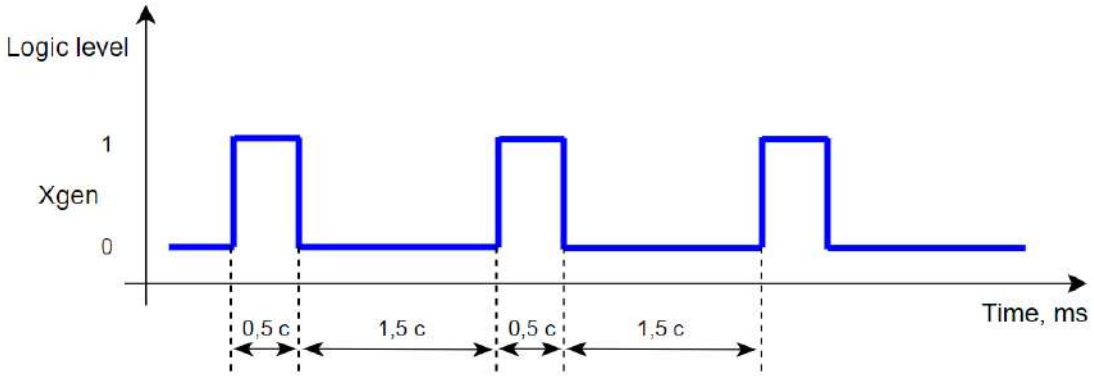


Рисунок 5.24 – Часова діаграма роботи автогенератора
з параметрами: TON = 1,5 с.; TOF = 0,5 с.

Приклад програми, що відповідає даній часовій діаграмі зображено на рис. 5.25.

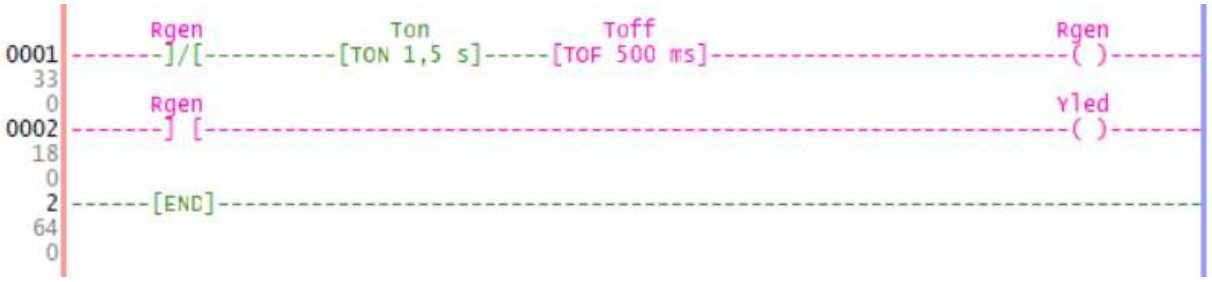


Рисунок 5.25 – Приклад програми, що відповідає часовій діаграмі з рис. 5.24

5.2.2 Приклад застосування циклічного таймеру TCY

У випадку, коли для роботи програми важливим є період слідування імпульсів, а інші параметри значення не мають, можна використовувати циклічний таймер TCY, що генерує меандр заданої частоти.

Приклад програми з використанням таймеру TCY подано на рис. 5.26.

Рисунок 5.26 – Приклад програми з використанням таймеру TCY для генерації імпульсів заданої частоти

На рис. 5.27 показана часова діаграма для даного виду генератору.

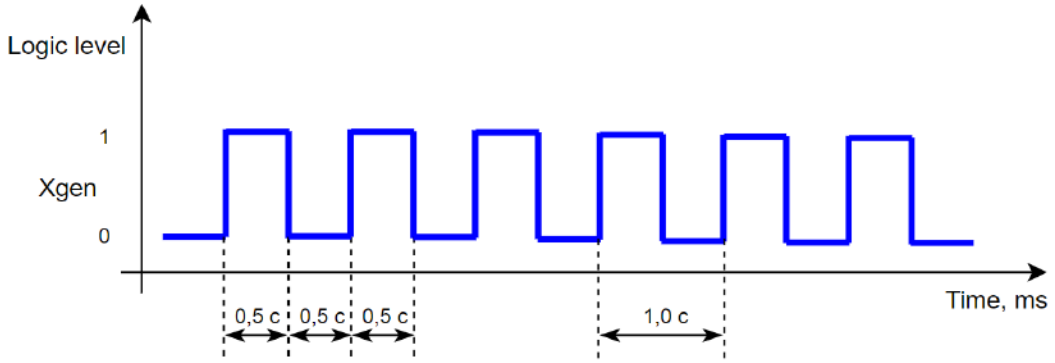


Рисунок 5.27 – Часова діаграма для параметру Delay таймеру TCY, який дорівнює 1000 мс.

Якщо імпульси використовуються безпосередньо в рядку з таймером, програма може бути скорочена (рис. 5.28).

Рисунок 5.28 – Приклад застосування циклічного таймеру TCY для організації блимання світлодіоду

5.2.3 Приклад застосування таймеру фіксованого часу вмикання THI

Розглянемо задачу автоматичного управління освітленням в автомобілі. Суть задачі полягає в наступному. При відчиненні дверей спрацьовує відповідний датчик та вмикається освітлення в салоні. Після зачинення дверей світло повинно горіти ще певний час для зручності водія, а потім автоматично вимикатись. На рис. 5.29 подано основні стадії роботи контролеру: контроль відкриття дверей, затримка вимкнення після зачинення та власне, вимкнення світла в салоні автомобіля.

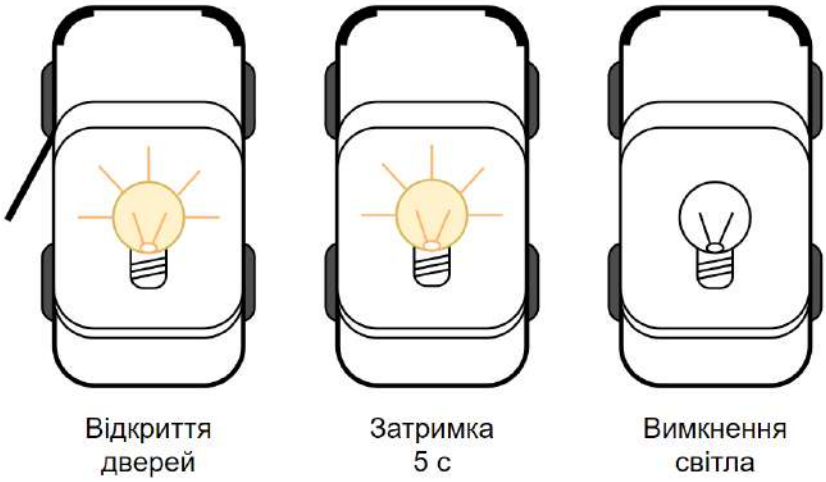


Рисунок 5.29 – Основні стадії роботи контролеру

На рис. 5.30 показано алгоритм роботи контролеру управління світлом в салоні автомобіля. Враховуючи, що в якості датчика відчинення дверей в автомобілях використовується кнопка з нормально замкненими контактами, основним робочим режимом є її знаходження в натиснутому (замкненому) положенні, коли двері зачинені. Таким чином, початковим станом системи є вимкнене світло та очікування, доки кнопка дверей (в нашому випадку, нехай це буде Xkey_g з програми емулятору) буде розімкнена (двері відчиняться) (рис. 5.30, стан 1).

Відразу після розімкнення кнопки повинно увімкнутися світло в салоні автомобіля. Поточним станом системи тепер буде горіння світла та очікування замикання кнопки Xkey_g (рис. 5.30, стан 2).

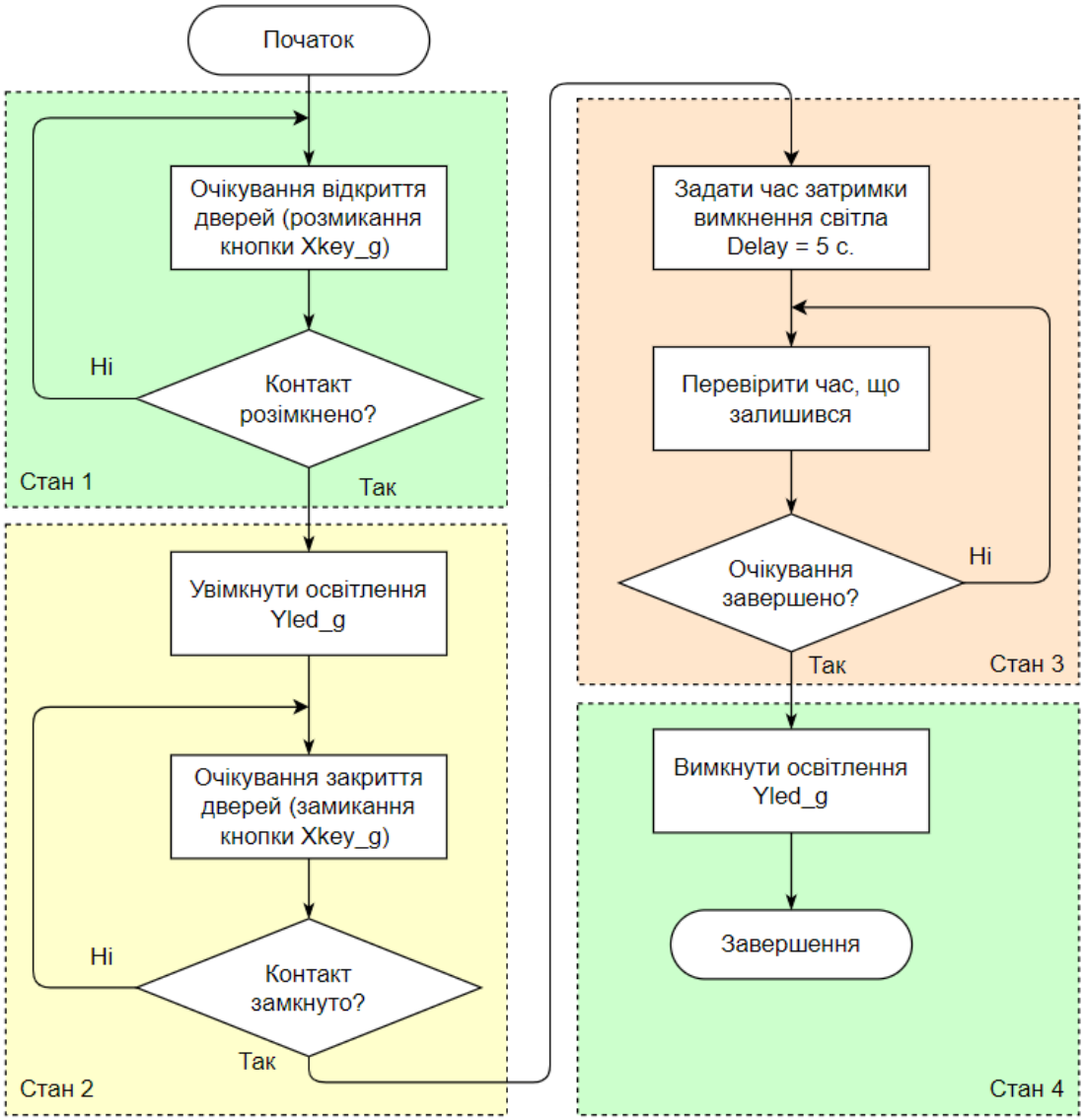


Рисунок 5.30 – Алгоритм роботи контролеру управління освітленням в салоні автомобіля

Після замикання кнопки Xkey_g система переходить в стан 3 (рис. 5.30), в якому на початку роботи задається термін очікування для таймеру фіксованого часу вмикання THI. В нашому прикладі час очікування дорівнює 5 с.

Після завершення стану очікування система переходить в стан 4, коли світло вимикається, а потім повертається до початкового стану 1, та основний цикл роботи програми поновлюється.

Для реалізації описаного алгоритму роботи розробимо програму засобами LDmicro. Приклад такої програми подано на рис. 5.31.

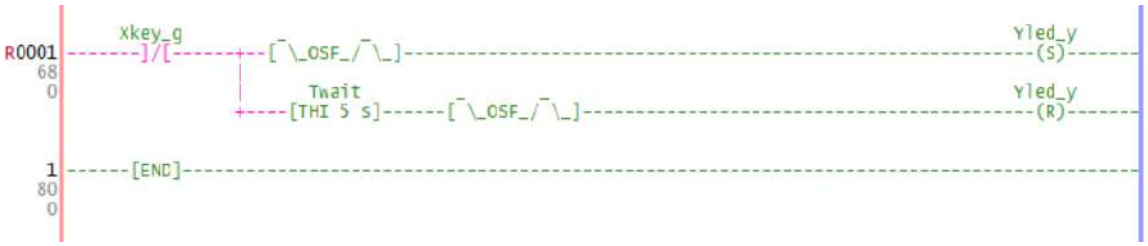


Рисунок 5.31 – Програма управління освітленням в салоні автомобіля

Враховуючи, що основний режим роботи пов’язано з подією розімкнення кнопки дверей, в якості інструкції для стеження за станом кнопки Xkey_g обираємо нормально замкнений контакт.

Для того, щоб обробити стан розімкнення кнопки використовується формувач сигналу за спаданням фронту вхідного сигналу «OSF». Властивість даного формувача така, що при переході сигналу на вході із стану логічної одиниці в логічний нуль, на виході з’являється короткий імпульс логічної одиниці.

На вихід елементу OSF підключено реле Yled_y в режимі «SET», тобто, при надходженні на вхід логічної одиниці, реле спрацьовує та залишається в такому стані доки не вимкнеться напруга живлення, або не прийде сигнал скидання «RESET».

Також, на вихід інструкції «Xkey_g» підключено ще одну гілку релейної схеми, що починається з таймеру фіксованого часу вмикання THI. Ця гілка вступає в роботу після замикання кнопки Xkey_g, коли сигнал логічної одиниці запускає таймер THI. Виходячи з особливості його роботи, на виході з’являється сигнал логічної одиниці на термін часу, що задається параметром «Delay, ms».

Після спливу заданого часу очікування на виході таймеру THI з’являється рівень логічного нуля. Для того, щоб вимкнути реле Yled_y на його вхід необхідно подати сигнал «RESET». Для цього перед цим реле знову ставимо формувач сигналу за спадом фронту вхідного сигналу «OSF». Таким чином, спад сигналу на виході таймера через формувач сигналу, подає сигнал «RESET» на реле Yled_y, яке вимикає світло в салоні автомобіля. Далі система переходить в початковий стан.

5.2.4 Приклад програми керування пультом охоронної сигналізації

Розглянемо приклад використання таймерів для рішення задачі керування пультом охоронної сигналізації. Необхідно реалізувати таку поведінку системи: мешканець відкриває двері та заходить в приміщення, при цьому йому надається 5 секунд для натискання потаємної кнопки та вимкнення сигналізації. Якщо потаємну кнопку не натисне вчасно, спрацює звукова сирена. Вимикати сирену може тільки власник приміщення, який знає місцезнаходження кнопки. Алгоритм роботи програми керування пультом охоронної сигналізації подано на рис. 5.32.

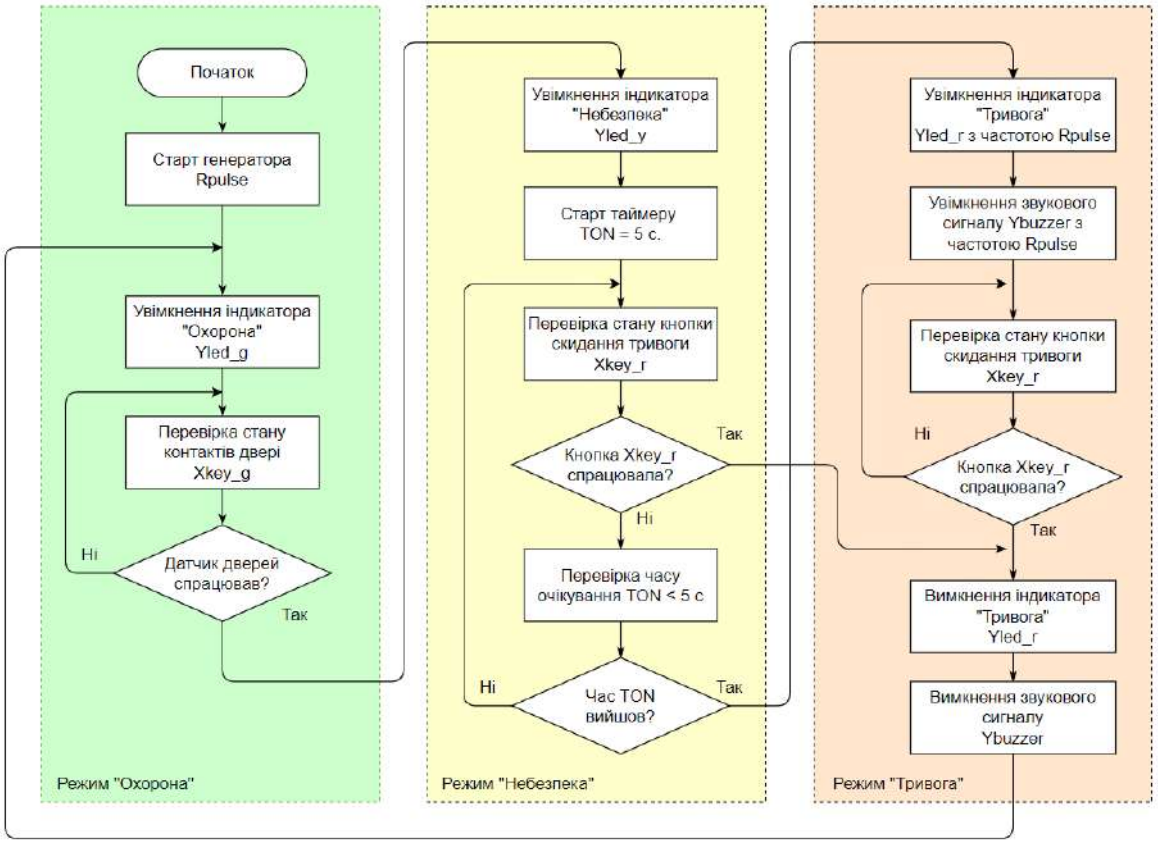


Рисунок 5.32 – Алгоритм роботи програми керування пультом охоронної сигналізації

В роботі алгоритму можна виділити три режими:

– режим «Охорона»;

– режим «Небезпека»;

– режим «Тривога».

На початку роботи програми запускається системний генератор Rpulse, який буде використаний для подачі звукових сигналів у випадку спрацювання сигналізації. В режимі «Охорона» увімкнений зелений сигнал індикатору і програма постійно слідкує за станом контактів датчику вхідних дверей.

Зовнішній вигляд датчику відчинення дверей подано на рис. 5.33. Датчик складається з двох частин:

– нерухома частина кріпиться на дверну раму та до неї приєднуються сигнальні дроти;

– рухома частина кріпиться до дверей, та в ній знаходиться магніт, під впливом якого замикаються контакти геркону в нерухомі частині.

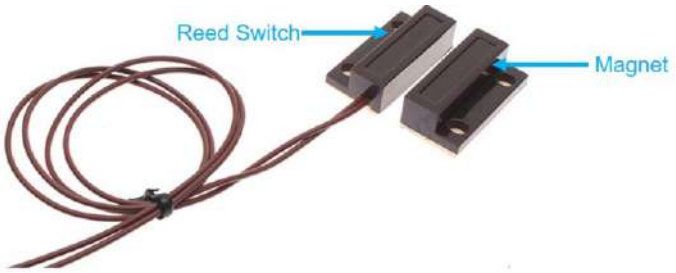


Рисунок 5.33 – Приклад конструкції датчика відчинення дверей

В зачиненому стані магніт знаходиться у безпосередній близькості від геркону та його контакти замкнені (рис. 5.34). У разі відчинення дверей, контакти датчика розмикаються і це є ознакою для охоронної системи, що потрібно виконувати дії, передбачені протоколом захисту приміщення.

За алгоритмом, що поданий на рис. 5.32, після спрацювання датчика відчинення дверей, програма переходить в режим «Небезпека». В даному режимі вмикається жовтий світлодіод та запускається таймер TON, який затримує сигнал від датчика на 5 с. За цей час мешканець приміщення повинен натиснути на потайну кнопку для деактивації звукового та світлового сигналів тривоги. В нашому алгоритмі за це відповідає кнопка Xkey_r. Протягом часу очікування постійно перевіряється стан даної кнопки, і, у випадку її спрацювання, програма вимикає жовтий світлодіод Yled_y і переходить в режим «Охорона», в якому вмикається зелений світлодіод Yled_g.

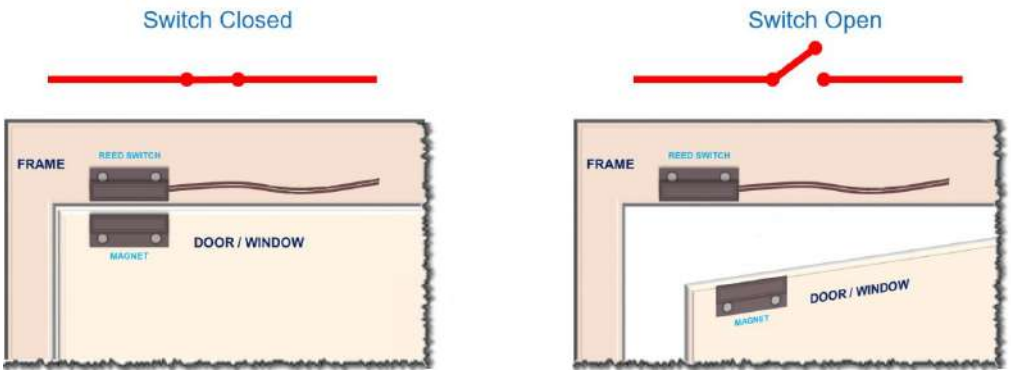


Рисунок 5.34 – Принцип дії датчика відчинення дверей

Якщо за 5 секунд кнопка скидання сигналу тривоги не була натиснута, програма переходить в режим «Тривога», в якому вимикається зелений світлодіод, періодично з частотою Rpulse вмикається червоний світлодіод Yled_r та подається переривчастий звуковий сигнал Ybuzzer. Вимкнути сигнал тривоги можна натиснувши на кнопку Xkey_r. В даному випадку будуть вимкнені червоний та жовтий сигнали світлодіодів, а також звуковий сигнал, і програма перейде до режиму «Охорона». Приклад програми керування пультом охоронної сигналізації, що відповідає алгоритму, представленому на рис. 5.32, наведено на рис. 5.35.

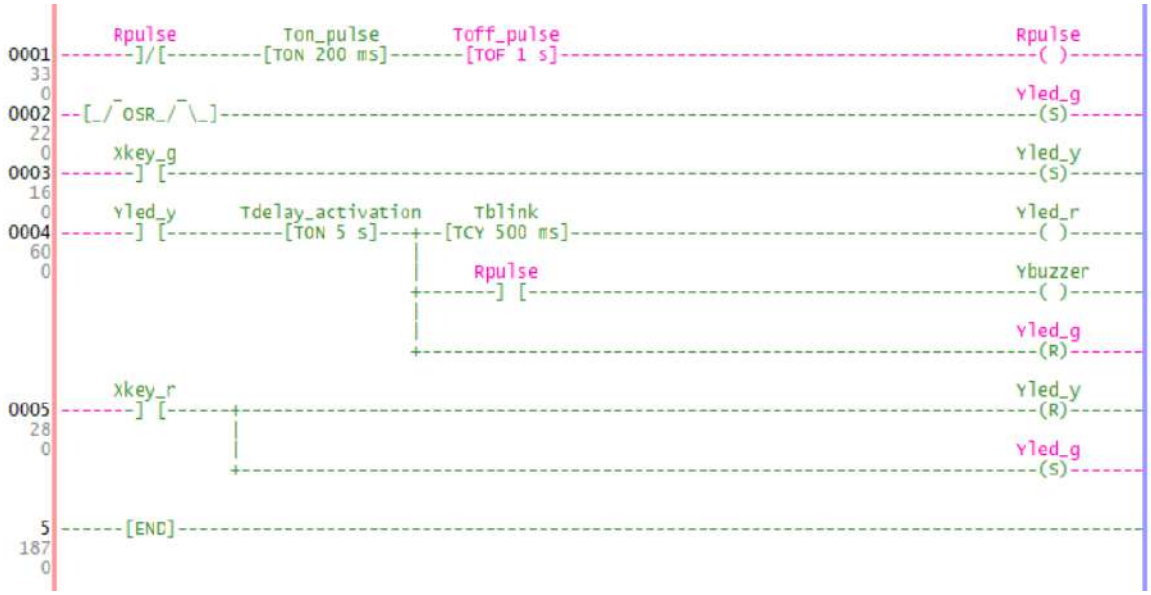


Рисунок 5.35 – Приклад програми керування пультом охоронної сигналізації

В першому рядку реалізовано автоматичний генератор Rpulse. Таймери TON і TOF налаштовані таким чином, щоб на виході генератора сигнал логічної одиниці знаходився 1 с, а сигнал логічного нуля – 200 мс.

На початку роботи програми спрацьовує генератор одиночного імпульсу, розташований в другому рядку. Цей генератор вмикає зелений світлодіод Yled_g, що сигналізує про знаходження програми в режимі «Охорона».

Перевірка кнопки відчинення дверей відбувається в третьому рядку. Якщо кнопка спрацювала, загоряється жовтий світлодіод Yled_y та програма переходить в режим «Небезпека».

В четвертому рядку перевіряється стан жовтого світлодіода і, якщо він спрацював, запускається таймер затримки увімкнення TON «Tdelay_activation». Якщо кнопка скидання «Xkey_r» за час очікування не буде натиснута, увімкнеться світлова та звукова сигналізації.

Скидання сигналу тривоги відбувається в п’ятому рядку. Натискання на кнопку Xkey_r вимкне жовтий сигнал світлодіоду Xled_y і увімкне зелений сигнал Xkey_g. Вимикання жовтого світлодіоду припинить роботу таймеру TON «Tdelay_activation» та відповідного четвертого рядку технологічної програми, в якому також відбувається керування світловою та звукової сигналізацією. Таким чином, програма повертається до початкового стану роботи.

5.3 Контрольні питання

1. Для чого призначені таймери?
2. Поясніть принцип роботи таймеру із затримкою вмикання TON.
3. Поясніть принцип роботи таймеру із затримкою вимкнення TOF.
4. Поясніть принцип роботи таймеру фіксованого часу вмикання THI.
5. Поясніть принцип роботи таймеру фіксованого часу вимкнення TLO.
6. Поясніть принцип роботи таймеру з накопиченням часу вмикання RTO.
7. Поясніть принцип роботи таймеру з накопиченням часу вимкнення RTL.
8. Поясніть принцип роботи циклічного таймеру TCY.
9. Наведіть приклади використання таймерів в технологічних програмах.

6 ВИКОРИСТАННЯ СПЕЦІАЛІЗОВАНИХ ІНСТРУКЦІЙ LD-MICRO

6.1 Застосування цифрових індикаторів в технічних засобах автоматизації

Цифрові семисегментні індикатори знайшли своє застосування в різноманітних пристроях систем автоматики, як прості засоби відображення службової інформації.

Завдяки застосуванню окремих сегментів, що розташовані у вигляді вісімки можна сформувати будь-яку цифру на світлодіодному табло (рис. 6.1).

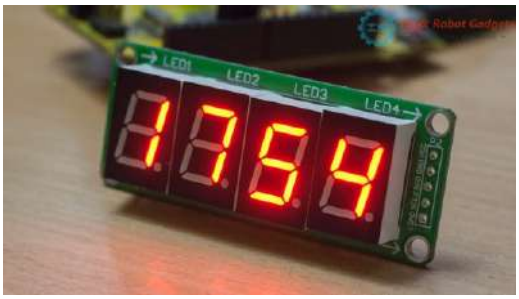


Рисунок 6.1 – Приклад виведення цифр на табло із чотирьох семисегментних індикаторів

Таке розташування сегментів дозволяє відображати будь-яку цифру в діапазоні від 0 до 9, як подано на рис. 6.2.

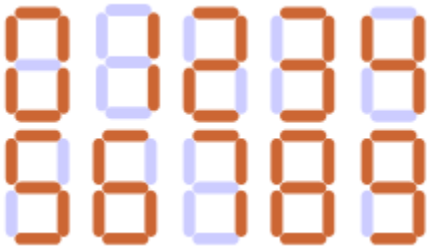


Рисунок 6.2 – Відображення цифр за допомогою семисегментного індикатору

Нажаль, таке розташування сегментів не дозволяє відображати всі можливі букви алфавіту. Наприклад, для букви «X» необхідні сегменти, що розташовані під кутом, а таких в семисегментному індикаторі немає. Тому, для відображення літер застосовуються індикатори із більшим числом сегментів.

Існують декілька різновидів таких пристроїв (рис. 6.3):

– 7-ми сегментні (рис. 6.3, а);

– 9-ти сегментні (рис. 6.3, б);

– 14-ти сегментні (рис. 6.3, в);

– 16-ти сегментні (рис. 6.3, г).

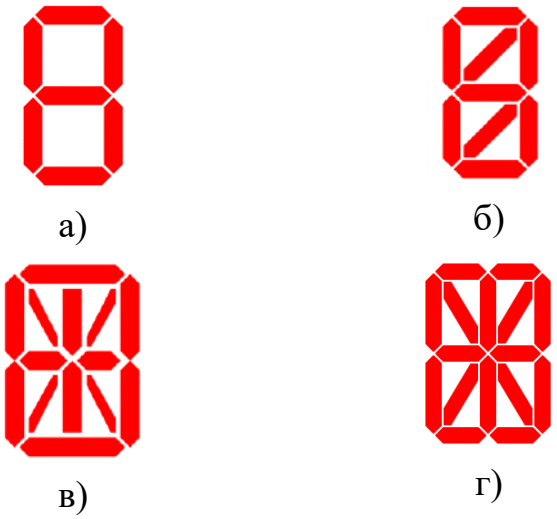


Рисунок 6.3 – Приклади різних типів індикаторів

На рис. 6.4 подано приклад відображення символів на різних типах індикаторів. В даному прикладі застосовуються індикатори із 7, 9, 14 та 16 сегментами. В якості тестового набору застосовуються наступна послідовність символів:

– перший набір: ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?

– другий набір: @ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\ ]

– третій набір: ` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

Для увімкнення кожного сегменту застосовується один, або більше світлодіодів в залежності від розмірів світлодіодного індикатору.

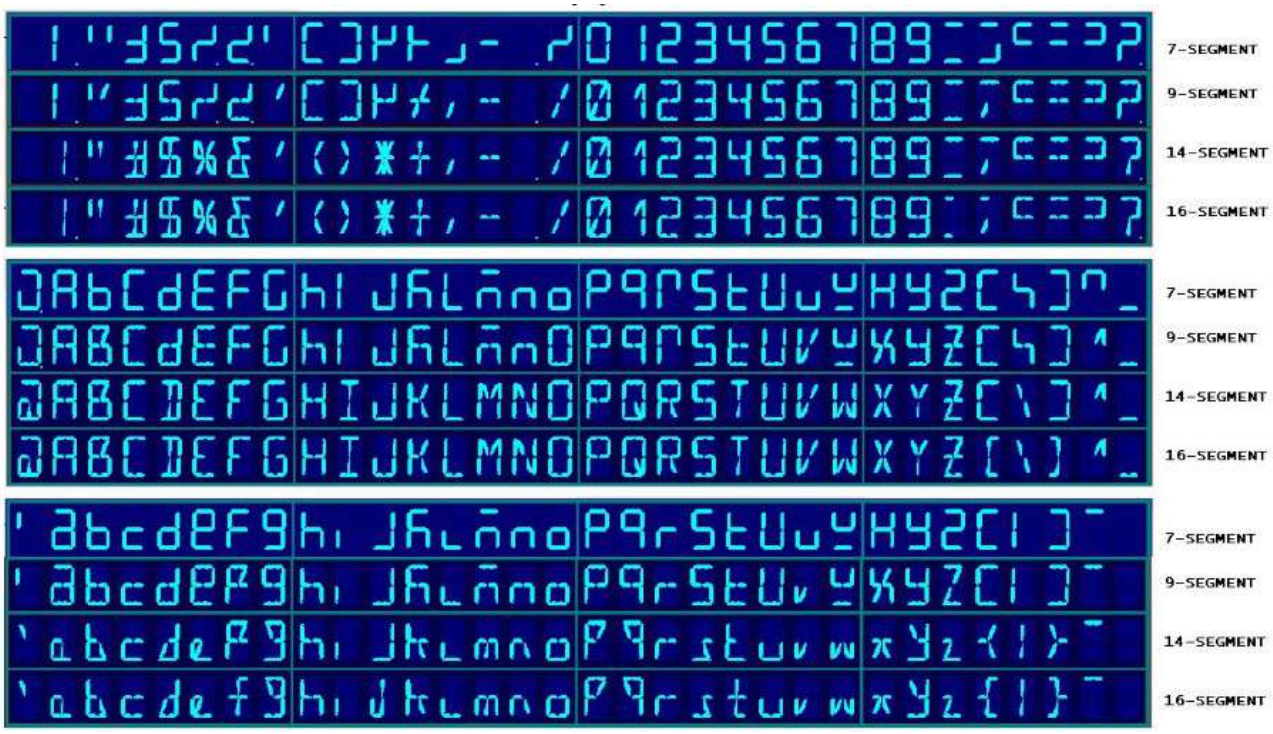


Рисунок 6.4 – Приклад відображення символів
на різних типах індикаторах

В якості прикладу розглянемо схему підключення семисегментного індикатору, що зображена на рис. 6.5.

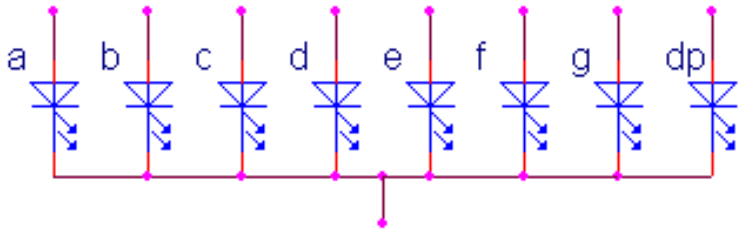


Рисунок 6.5 – Варіант схеми підключення
семисегментного індикатору

Для запалювання кожного світлодіоду, його необхідно під’єднати до джерела живлення через резистор (рис. 6.6).

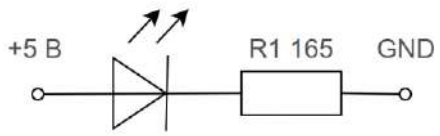


Рисунок 6.6 – Використання резистора
в ланцюзі підключення світлодіоду

Номінал резистора розраховується в залежності від характеристик світлодіоду за формулою:

$$R = \frac{U_0 - U_{vd}}{I_{vd}}, \tag{6.1}$$

де U_0 – напруга живлення, В;

U_{vd} – напруга, що падає на світлодіоді, В (береться із документації на цей світлодіод);

I_{vd} – номинальный струм світлодіода, А.

Виходячи з (6.1), необхідно знати номінальний струм світлодіоду та напругу, що падає на ньому. Ці значення обираються з технічної документації на світлодіод. Наприклад, якщо номінальний струм дорівнює 20 мА, а напруга, що падає на світлодіоді становить 1,7 В, то за (6.1), якщо напруга живлення становить 5 В, необхідний номінал резистору буде становити:

$$R = \frac{U_0 - U_{vd}}{I_{vd}} = \frac{5 - 1{,}7}{0{,}02} = 165 \text{ (Ом)}.$$

Схема підключення семисегментного індикатора до контролеру подана на рис. 6.7. З даної схеми можна бачити, що кожен сегмент підключається но відповідного виходу контролеру через резистор. Також необхідно підключити загальний провід до однієї з клем живлення. На рис. 6.6 показано, що загальний провід підключається до мінусової клеми джерела живлення. Таке включення називається схемою із загальним катодом. Для запалювання кожного сегменту необхідно подавати на нього позитивний сигнал, або, логічну одиницю.

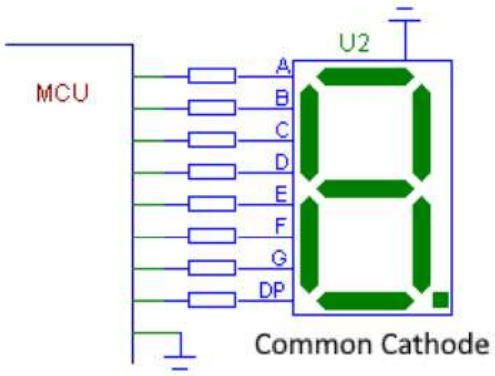


Рисунок 6.7 – Підключення індикатору за схемою
із загальним катодом

Також використовується схема із загальним анодом (рис. 6.8).

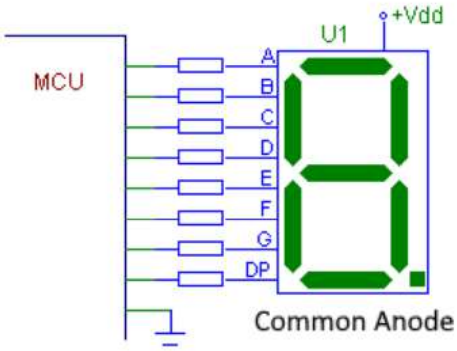


Рисунок 6.8 – Підключення індикатора за схемою
із загальним катодом

В даному випадку загальний провід підключається до позитивної шини живлення. Всередині такого модуля світлодіоди підключені у зворотному напрямку і, для запалювання кожного сегменту, необхідно подавати на нього негативний сигнал або логічний нуль. Існує стандартне позначення кожного сегменту індикатора для посилання на нього в програмі або в описанні принципу роботи. На рис. 6.9 подано типове найменування сегментів у семисегментному індикаторі.

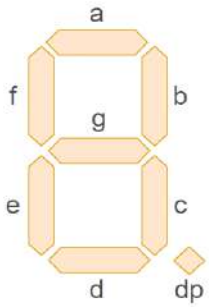


Рисунок 6.9 – Типове найменування сегментів
у семисегментному індикаторі

Необхідно враховувати, що крапка, яка розділяє сегменти, позначається, як «dp» (рис. 6.9).

Дуже часто для відображення інформації використовуються модулі, що містять відразу два, три, чотири, та більшу кількість розрядів індикаторів (рис. 6.10).

Рисунок 6.10 – Світлодіодні модулі із різною кількістю розрядів

В такому випадку для економії провідників, що підключаються до контролеру, застосовують, так званий, динамічний принцип індикації. Такий спосіб підключення передбачає паралельне поєднання всіх сегментів в загальну шину по рядках та по стовпцях, як подано на рис. 6.11.

Для запалювання окремого сегменту в конкретному модулі, необхідно подати живлення на відповідний стовпець, та обрати необхідний сегмент, або декілька сегментів, подавши на нього логічну одиницю, або нуль в залежності від схеми підключення світлодіодів.

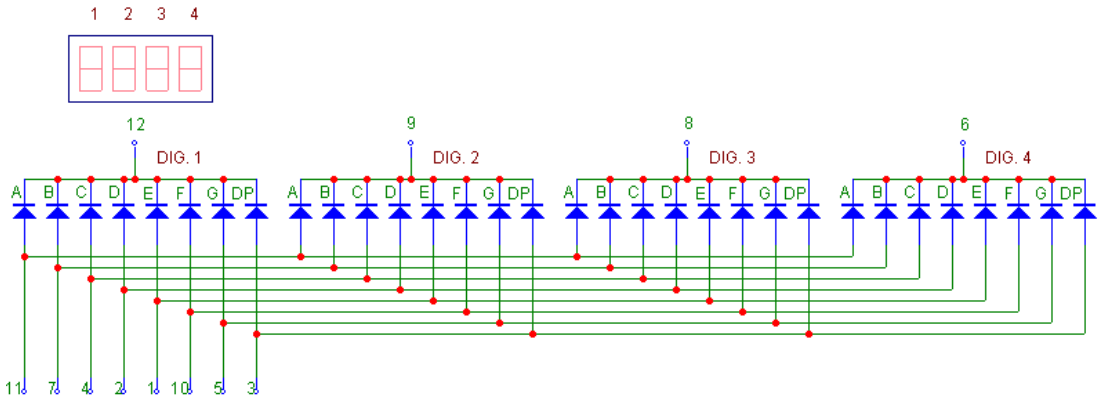


Рисунок 6.11 – Підключення індикаторів в режимі динамічної індикації

В кожний момент вмикається тільки один сегмент, який горить виділений проміжок часу, а далі вмикається наступний. При цьому змінюється послідовність нулів та одиниць на сегментах для відображення необхідного числа. Приклад генерації числа «123» на трьохрозрядному індикаторі подано на рис. 6.12.

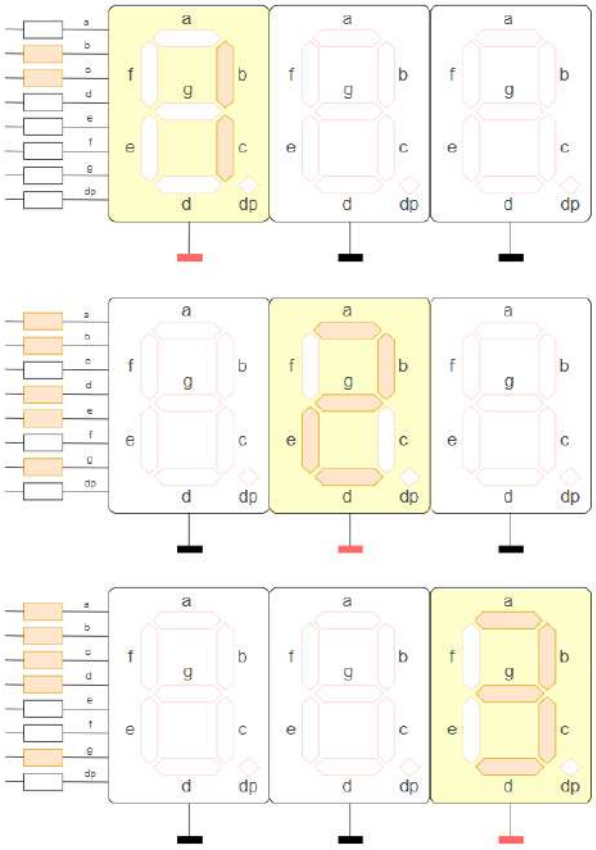


Рисунок 6.12 – Приклад генерації числа «123» на трьохрозрядному індикаторі

Чим більша частота перемикання розрядів, тим непомітніше для ока людини процес вибору відповідного модуля і здається, що всі цифри горять одночасно.

Для управління сегментними індикаторами в LDmicro використовується відповідна інструкція «Segments font converter», що подана на рис. 6.13.

Рисунок 6.13 – Інструкція «Segments font converter»

Для обирання даної інструкції необхідно викликати відповідне меню із підменю «Displays» (рис. 6.14).

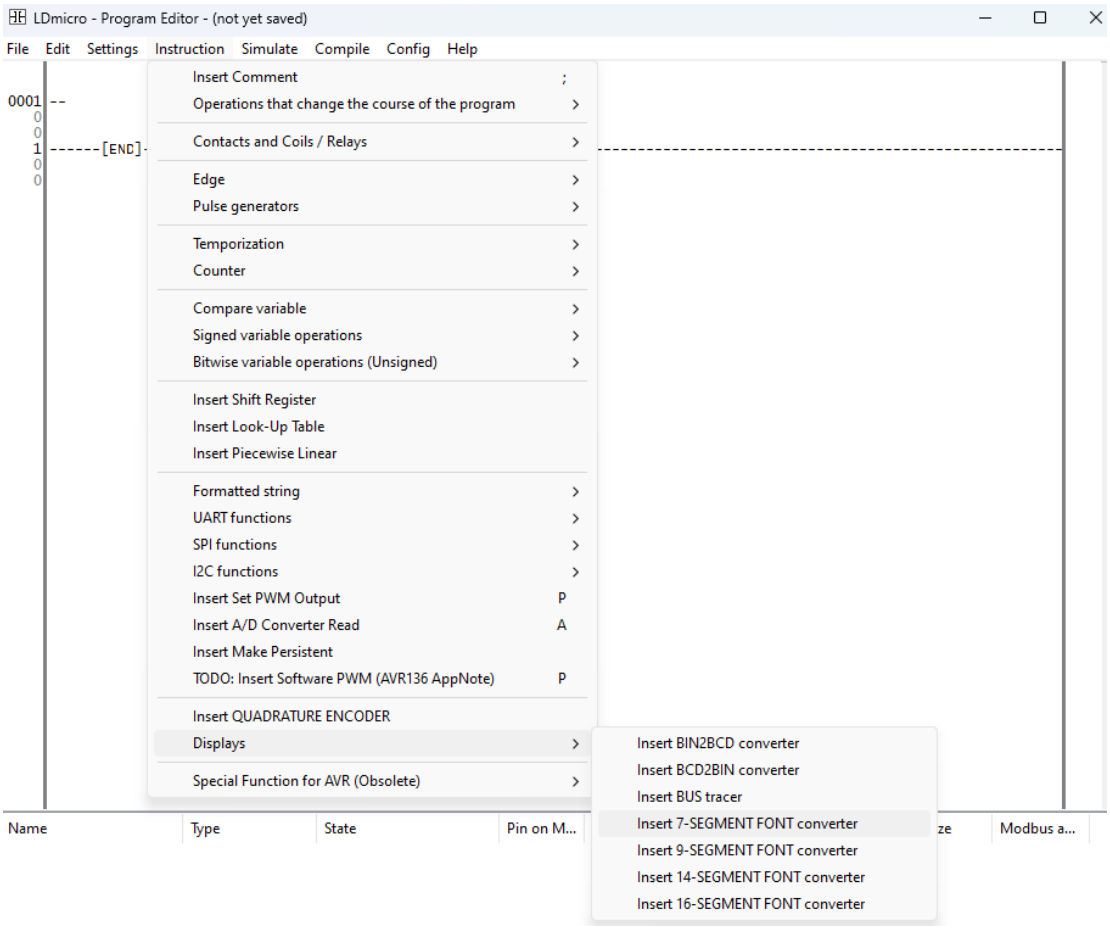


Рисунок 6.14 – Обирання інструкції «Segments font converter»

Інструкція конвертування символу в кодування, що відповідає сегментному індикатору передбачає вибір типу підключення:

– Common Cathode (загальний катод);

– Common Anode (загальний анод).

В інструкції вибір позначається відповідною літерою: «C», або «A» (рис. 6.15).

а) б)

Рисунок 6.15 – Варіанти вибору типу підключення:
а) Common Cathode (загальний катод); б) Common Anode (загальний анод)

Приклад вікна для налаштування параметрів інструкції конвертування символів подано на рис. 6.16.

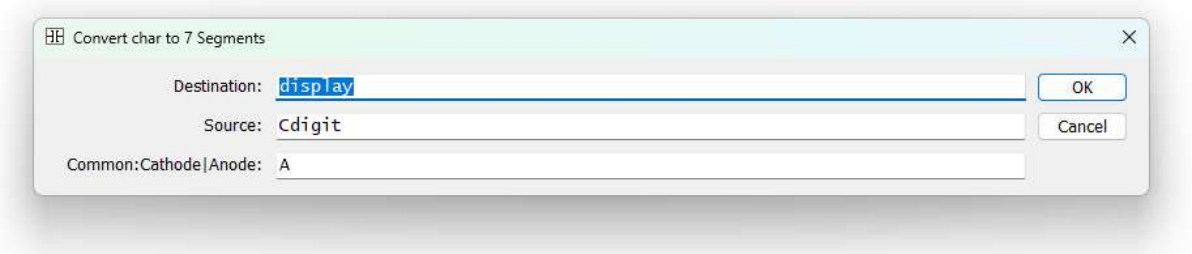


Рисунок 6.16 – Приклад вікна для налаштування параметрів інструкції конвертування символів

В поданому прикладі спочатку задається назва змінної, до якої буде занесено результати конвертування. В нашому випадку – це змінна з назвою «display».

Наступним параметром є назва змінної з якої береться символ для конвертування. В даному прикладі – це змінна «Cdigit», що є циклічним лічильником від 0 до 9. Зміна імпульсів на вході цього лічильника призводить до появи в «Cdigit» нової цифри в межах вказаного діапазону.

Третім параметром є ознака типу підключення індикатору. В даному прикладі – це підключення із загальним анодом (літера «A»).

В LDmicro передбачено чотири різних варіантів кількості сегментів в індикаторі: 7, 9, 14 або 16. Для кожного з них є відповідна інструкція, що можна бачити на рис. 6.14. Як було зазначено раніше, основне призначення інструкції – це перетворення вхідної змінної в послідовність керуючих сигналів для відповідного типу індикатора. Приклад такої послідовності для відображення цифри «4» подано на рис. 6.17.

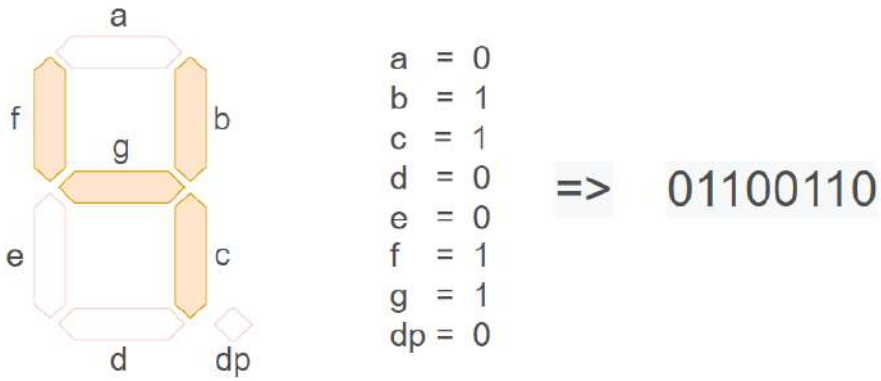


Рисунок 6.17 – Приклад послідовності біт для відображення цифри «4»

В табл. 6.1 наведено принцип запису бітової послідовності і відповідність до сегментів індикатору.

Таблиця 6.1 – Принцип запису бітової послідовності і відповідність до сегментів індикатору

Сегмент	dp	g	f	e	d	c	b	a
Біт в змінній «dest»	7	6	5	4	3	2	1	0

Значення змінної «Source» типу «char» (рис. 6.16) повинні лежати в діапазоні від 0 до 128, що відповідає модифікованій таблиці кодування ASCII. Приклад символів та їх коди подані в табл. 6.2.

Відмінність від звичайної таблиці ASCII полягає в наступному:

– наявність 129 символу градусу із кодом 128(0x80) для відображення температури;

– перший рядок таблиці ASCII замінений на послідовність символів в форматі HEX від 0 до F;

– другий рядок таблиці ASCII замінений на послідовність символів в форматі HEX від 0 до F із включеним сегментом «dp» (крапка, що розділяє сегменти).

Таблиця 6.2 – Приклад символів та їх коди надані в діапазоні від 0 до 128

Колонка / Адреса	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0(0x00)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
16(0x10)	0.	1.	2.	3.	4.	5.	6.	7.	8.	9.	A.	B.	C.	D.	E.	F.
32(0x20)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
48(0x30)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
64(0x40)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
80(0x50)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
96(0x60)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
112(0x70)	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
128(0x80)	°	← degree char 0xB0														

В подальшому, отримана послідовність використовується для увімкнення кожного окремого сегменту, що підключається до відповідного вихідного порту ПЛК. Кожен сегмент необхідно перевірити окремо та визначити, чи встановлено відповідний біт в стан логічної одиниці, чи ні. Для перевірки використовується відповідна інструкція LDmicro, що має назву побітове порівняння I (рис. 6.18).

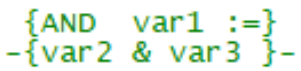


Рисунок 6.18 – Побітова інструкція I

Бітову маску зазвичай використовують для отримання значення біта. Для цього потрібно відключити решту бітів за допомогою побітового логічного «І» (кон'юнкції).

Наприклад, для отримання значення п'ятого біта (рахуючи ліворуч) числа 10111011 потрібно використовувати маску 00001000. В результаті вийде така операція, яка подана на рис. 6.19.

1011**1**011	1011**0**011
&	&
0000**1**000	0000**1**000
=	=
0000**1**000	0000**0**000

Рисунок 6.19 – Приклад накладання бітової маски

Дана інструкція накладає маску на вхідну змінну, та одержує результат «так» або «ні» на виході. На рис. 6.20 подано приклад управління сегментами індикатора.

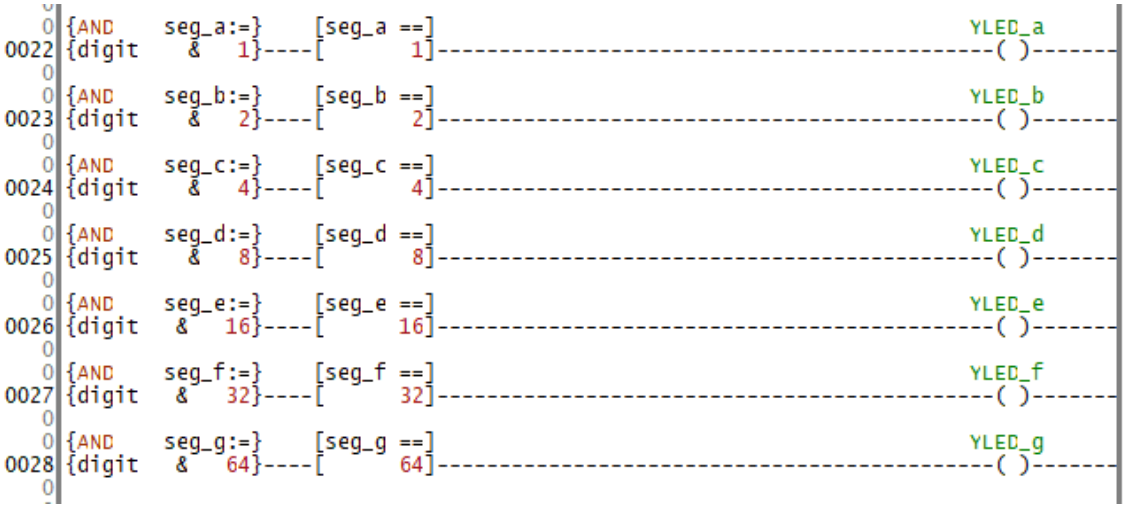


Рисунок 6.20 – Приклад управління сегментами індикатора

Кожна сходинка управляє відповідним сегментом індикатора. Спочатку накладається бітова маска для перевірки наявності логічної одиниці в нульовому розряді змінної «digit».

Записується це так:

[seg_a := digit & 1]

Далі перевіряється, чи містить змінна seg_a одиницю в нульовому розряді після накладання бітової маски:

[seg_a == 1]

Якщо в результаті перевірки результат дорівнює true, то на виході інструкції перевірки буде виставлена логічна одиниця. Ця одиниця вмикає вихідне реле YLED_a, до контактів якого підключено сегмент «a» індикатора і індикатор запалюється.

Таким самим способом відбувається перевірка кожного із розрядів змінної «digit» для кожного сегменту індикатора (рис. 6.20).

Розглянемо фрагмент коду мовою LD для управління чотирьохрозрядним індикатором. На рис. 6.21 подано приклад створення генератора імпульсів для перебирання розрядів індикатора. За допомогою циклічного лічильника змінна CLED набуває значення від 0 до 3.

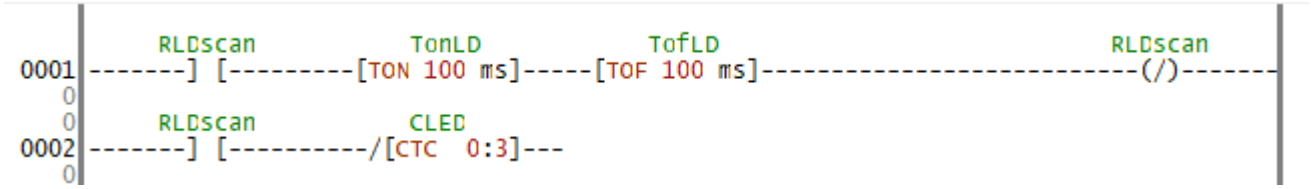


Рисунок 6.21 – Генератор імпульсів для перебирання розрядів індикатору

В наступній частині коду відбувається вмикання відповідного розряду індикатора після перевірки відповідності числа, що міститься в змінній CLED номеру розряду, як подано на рис. 6.22.

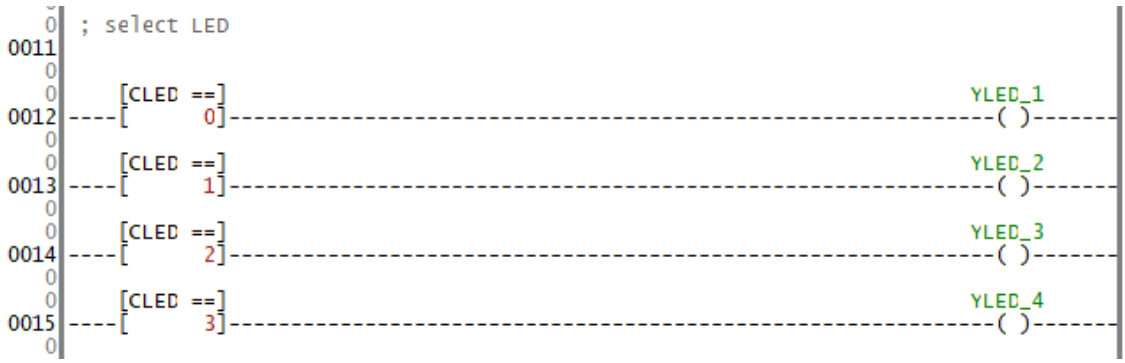


Рисунок 6.22 – Вмикання відповідного розряду індикатору

Далі відбувається перетворення чисел «1», «2», «3» та «4» в послідовність біт для запалювання відповідних сегментів кожного індикатора (рис. 6.23).

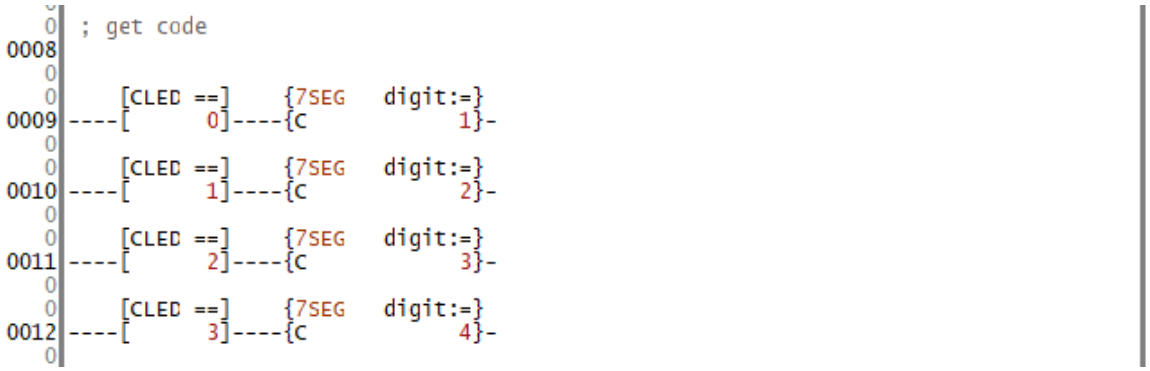


Рисунок 6.23 – Перетворення чисел в послідовність біт
для запалювання сегментів індикатора

З наведеного коду можна бачити, що для відображення інформації використовуються семисегментні індикатори з загальним катодом. При обиранні певного розряду індикатора, в змінну «digit» заноситься код для вмикання відповідних сегментів.

Для завершення даного коду необхідно використати фрагмент, що зображено на рис. 6.20 для накладання бітової маски та увімкнення потрібних сегментів індикатора.

Повний код для управління чотирьохрозрядним індикатором подано на рис. 6.24.

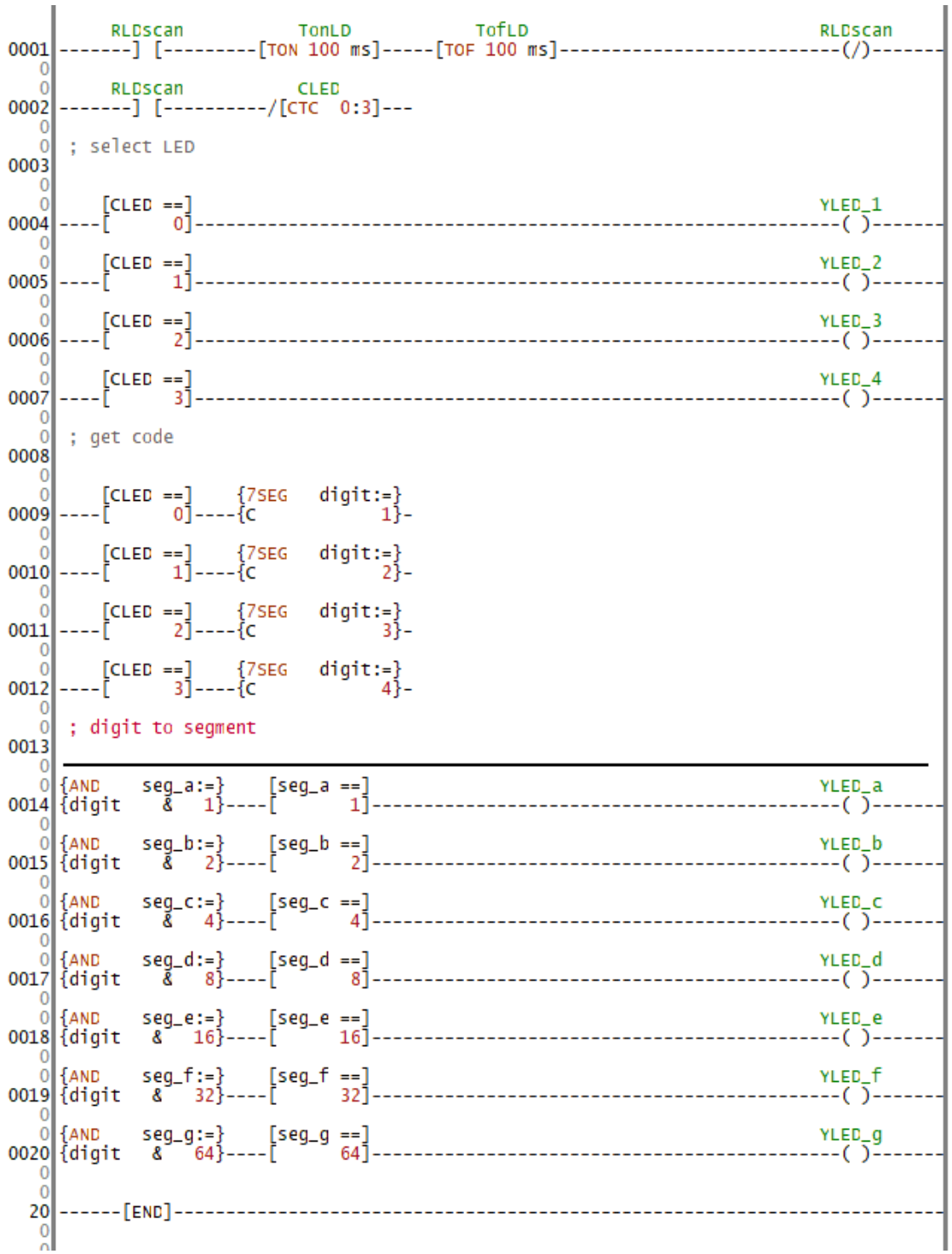


Рисунок 6.24 – Код для управління чотирьохрозрядним індикатором

Приклад виведення числа «1234» на чотирьохрозрядний індикатор у віртуальному макеті подано на рис. 6.25.

Рисунок 6.25 – Приклад виведення числа «1234» на чотирьохрозрядний індикатор в віртуальному макеті

6.2 Організація введення аналогових сигналів засобами LD

6.2.1 Загальні відомості

LDmicro може генерувати код для використання аналого-цифрових перетворювачів, вбудованих у певні мікроконтролери. Для цього використовується відповідна інструкція «Read A/D Convertor» (рис. 6.26).

Рисунок 6.26 – Використання інструкції «Read A/D Convertor»

Після додавання інструкції в схему необхідно ввести унікальне ім’я змінної «Destination». До введеної назви автоматично додається службовий префікс «A», що означає АЦП.

Якщо вхідна умова для інструкції «Read A/D Convertor» є істинною, тоді виконується зчитування даних з аналого-цифрового перетворювача та зберігається

в змінній «ADCnew». Надалі цією змінною можна керувати за допомогою загальних операцій (менше, більше, арифметичні операції тощо).

Засобами налаштування програми можна призначити відповідний порт мікроконтролера для змінної «ADCnew» так само, як обирається порт для цифрового входу або виходу, двічі клацнувши на назві змінної у списку внизу екрана програми.

Якщо вхідна умова для інструкції «Read A/D Convertor» хибна, тоді змінна «ADCnew» залишається без змін.

Для всіх підтримуваних на даний момент пристроїв вхідна напруга 0 вольт відповідає показанню АЦП 0, а напруга на вході, що дорівнює Vdd (напруга живлення мікроконтролеру), відповідає показанню АЦП 1023. Якщо як контролер використовується пристрій AVR, то необхідно підключити його порт AREF до Vdd.

В програмі, що розроблюється, можна використовувати арифметичні операції, щоб масштабувати показання змінної «ADCnew» до більш зручних одиниць, але необхідно пам’ятати, що в LDmicro для цього використовується математичні операції над цілими числами (тип int).

Загалом не всі контакти МК доступні для використання з аналого-цифровим перетворювачем. Програмне забезпечення не дозволить призначати порти, які не прилаштовані для аналогового входу.

Інструкція «Read A/D Convertor» має бути крайньою правою інструкцією у своєму рядку.

Опорна напруга для АЦП (Vref) вказує діапазон перетворення для вбудованого модуля АЦП. Канали, на яких вхідна напруга перевищує Vref, будуть генерувати значення коду перетворення наближене до 0x3FF(1023). Vref вказує на обирання типу опорної напруги:

– напруга живлення МК AVCC;

– внутрішня опорна напруга 2,56 В;

– внутрішня опорна напруга 1,1 В;

– зовнішній контакт Aref та зовнішнє опорне джерело живлення.

Параметр REFS = 0 зворотно сумісний із попередньою версією LDmicro. Для контролерів Atmel AVR поле REFS відповідає регістру REFS1:0 – біти вибору налаштування ADC. Для контролерів Microchip PIC – відповідає регістру PCFG3:0

– біти керування конфігурацією АЦП. Точну інформацію для обирання правильного значення поля REFS можна знайти в документації на обраний для роботи мікроконтролер.

6.2.2 Реалізація індикатору рівня аналогового сигналу у вигляді стовпчикової діаграми

Розглянемо роботу з АЦП на прикладі застосування віртуального макету. Опис даного макету подано в розділі 3.4. Необхідно вирішити задачу зчитування показання датчика, що виробляє аналоговий сигнал та перетворює його в цифровий код за допомогою восьмибітного АЦП і відображення значення у вигляді стовпчикової діаграми.

Робоча область макету, що використовується для вирішення поставленої задачі подано на рис. 6.27.

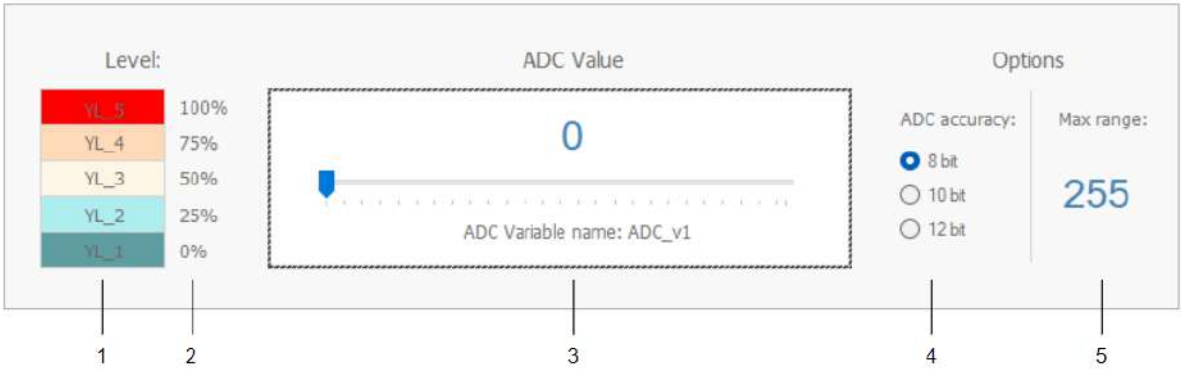


Рисунок 6.27 – Робоча область макету для дослідження АЦП

Стовпчикова діаграма 1 має п’ять рівнів, кожний з яких має свій колір. Кожному рівню відповідає текстова мітка 2, що розміщується праворуч та має такі значення: «0%, 25%, 50%, 75% та 100%». В програмі необхідно обчислити вхідне значення змінної із даними з АЦП та підсвітити відповідний сегмент стовпчикової діаграми в залежності від поточного рівня сигналу.

Кожному сегменту відповідає вихідний контакт ПЛК. Назви цих контактів написані на фоні кожного сегменту. Наприклад, для того, щоб увімкнути перший, нижній сегмент «0%», необхідно подати сигнал логічної одиниці на реле, що пов’язано з вихідним контактом «YL_1».

В області 3 розташовано регулятор значення змінної «ADC_v1», що пов’язана безпосередньо із АЦП ПЛК та передає в нього дані з кожною змінною положення повзунка регулятора.

Регулювання значення АЦП відбувається в межах його максимального значення, яке залежить від розрядності. В налаштуваннях віртуального пристрою можна обрати одне із наступних значень розрядності АЦП (рис. 6.27, поз. 4):

– 8 біт (максимальне значення 255);

– 10 біт (максимальне значення 1023);

– 12 біт (максимальне значення 4095).

Максимальне можливе значення на виході АЦП показано в області 5 інтерфейсу програми.

Для рішення поставленої задачі, на початку діаграми створимо сходинку із зчитуванням поточного значення АЦП в змінну «ADC_v1» (рис. 6.28).

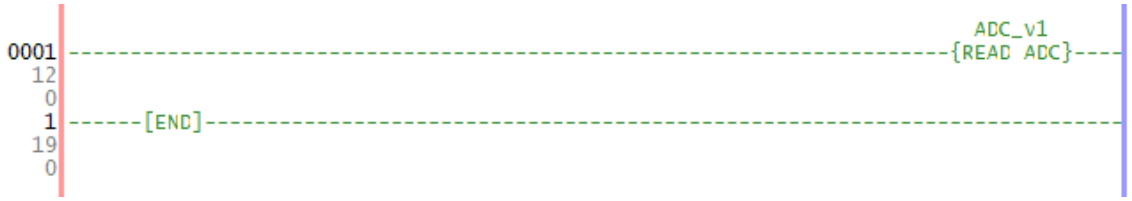


Рисунок 6.28 – Зчитування поточного значення АЦП в змінну «ADC_v1»

Для перевірки отриманого значення використовуємо інструкцію для порівняння його із нульовим рівнем:

$$[\text{ADC\_v1} >= 0]$$

В разі виконання умови, на виході інструкції з’являється логічний нуль, що призводить до увімкнення вихідного реле YL_1 та запалювання першого сегменту на діаграмі.

На рис. 6.29 подано приклад застосування інструкції перевірки і вмикання першого сегменту діаграми.

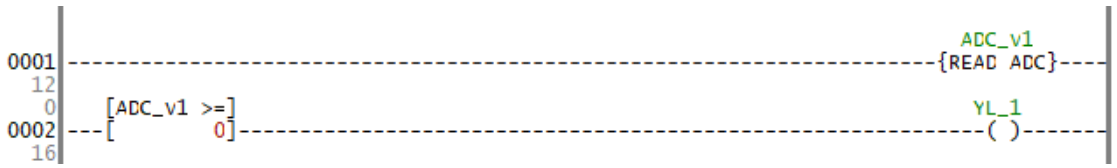


Рисунок 6.29 – Приклад застосування інструкції перевірки
і вмикання першого сегменту діаграми

Аналогічним чином виконуємо перевірку інших значень змінної ADC_v1 та запалювання сегментів від 2 до 5. Повний код діаграми LD подано на рис. 6.30.

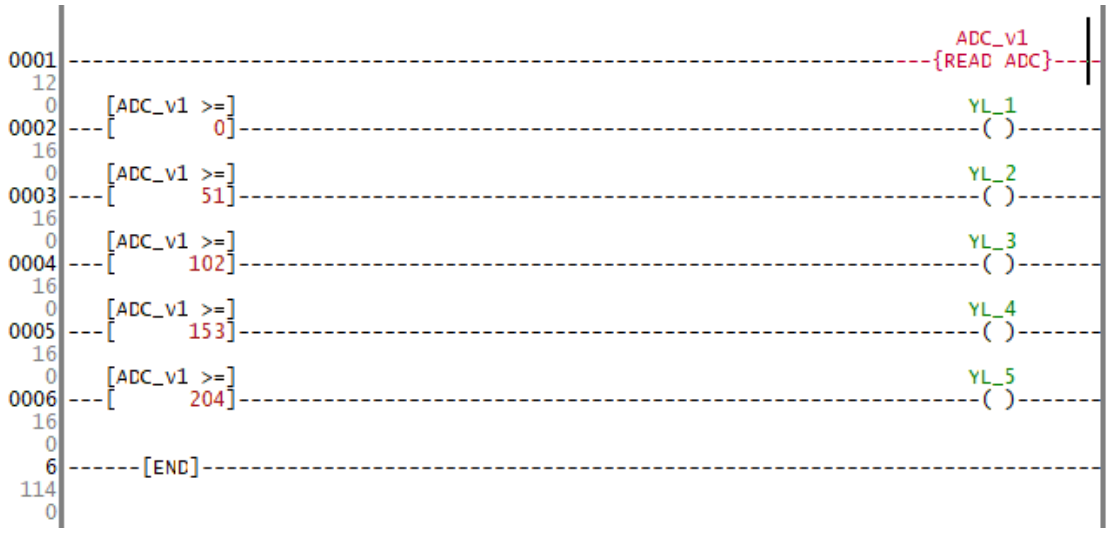


Рисунок 6.30 – Діаграма LD для індикації рівня сигналу
за отриманим значенням від АЦП

На рис. 6.31 показані чотири різних варіанти вмикання сегментів стовпчикової діаграми та відповідні значення змінної ADC_v1.

6.2.3 Відображення отриманих даних від АЦП на семисегментному індикаторі

Виконаємо модифікацію попередньої програми і додамо можливість відображення отриманих від АЦП даних на цифровому семисегментному

індикаторі. Для відображення цифр скористуємось методикою, що розглянута в розділі 6.1.

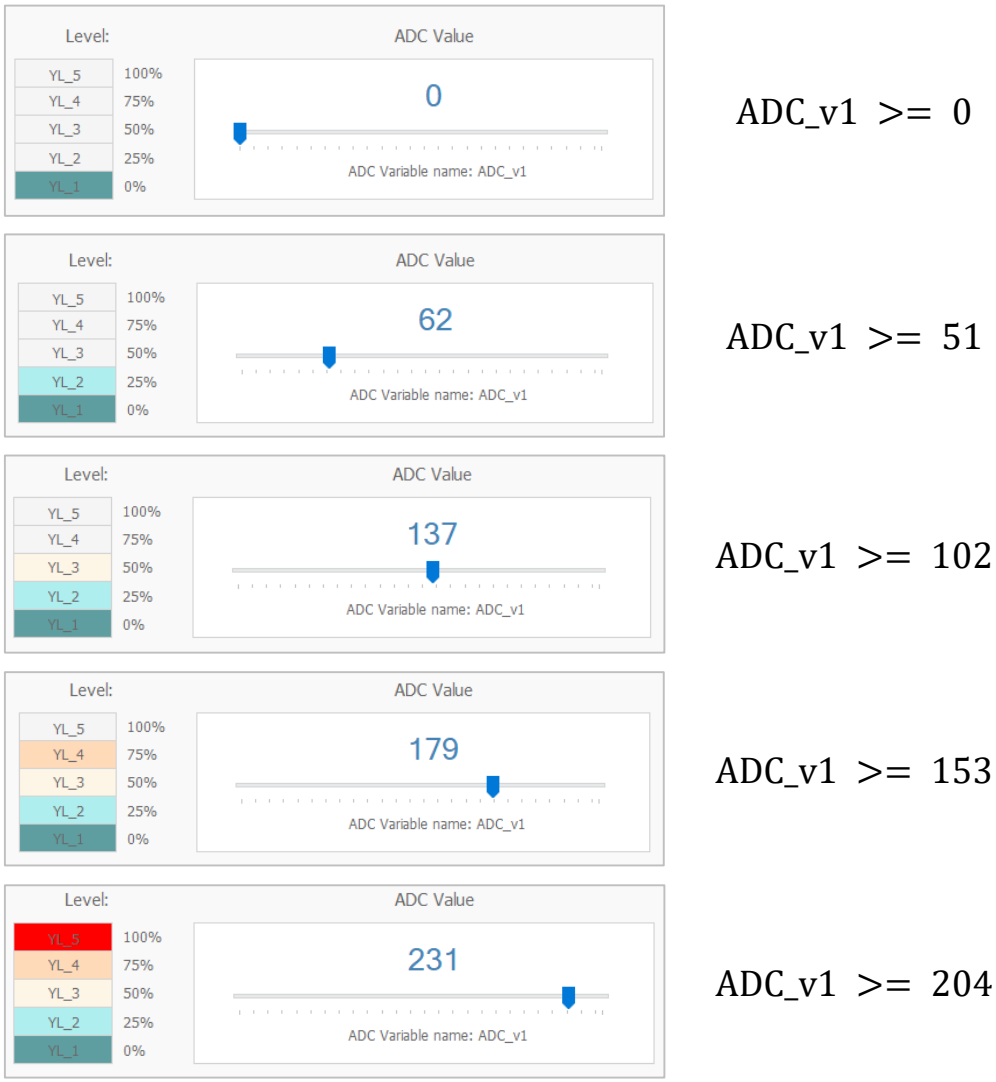


Рисунок 6.31 – Варіанти вмикання сегментів стовпчикової діаграми

Для виконання завдання обираємо АЦП розрядністю 10 біт. Такий АЦП має максимальний діапазон значень від 0 до 1023 (рис. 6.32).

Таким чином, щоб мати можливість відображати всі отримані цифри, нам необхідно обрати чотирьохрозрядний семисегментний індикатор.

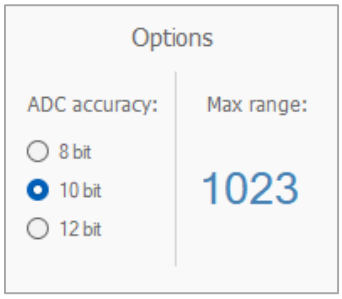


Рисунок 6.32 – Вибір розрядності АЦП

В якості схеми підключення обираємо варіант із загальним катодом, тому що цей варіант передбачає керування кожним сегментом індикатора подачею на нього сигналу логічної одиниці (рис. 6.33).

Рисунок 6.33 – Вибір типу світлодіодного модуля

Значення змінної ADC_v1 задається за допомогою лінійного повзунка і, як було сказано вище, повний діапазон значень складає 0…1023 одиниці. Тому, щоб можна було правильно відобразити кожну цифру із зазначеного діапазону, вміст змінної ADC_v1 потрібно попередньо підготовити: розбити на окремі чотири змінні для тисяч, сотень, десятків та одиниць.

Щоб отримати тисячі із числа, що зберігається в ADC_v1 потрібно поділити це число на 1000 та занести в змінну «th» ціле значення від ділення. В LDmicro існує два типи ділення:

– ціле ділення «DIV» (у вказану змінну заносить цілий результат від ділення);

– ділення із залишком «MOD» (у вказану змінну заносить ціле значення залишку від ділення).

На першому кроці виконаємо операцію ділення «DIV»:

$$\{\text{DIV th} := \text{ADC\_v1}/1000\}$$

Для отримання значення сотень необхідно визначити залишок від ділення вхідної змінної на 1000:

$$\{\text{MOD th\_rem} := \text{ADC}_{\text{v1}}\%1000\}$$

На третьому кроці можемо визначити кількість сотень, що входить до числа, яке зберігається в змінній th_rem:

$$\{\text{DIV hnd} := \text{th\_rem}/100\}$$

Для визначення кількості десятків необхідно визначити залишок від ділення змінної th_rem (залишок після визначення кількості тисяч) на 100:

$$\{\text{MOD hnd\_rem} := \text{th\_rem}\%100\}$$

П’ятим кроком є визначення кількості десятків:

$$\{\text{DIV ten} := \text{hnd\_rem}/10\}$$

Останнім кроком визначаємо кількість одиниць в числі, що зберігається в змінній hnd_rem:

$$\{\text{MOD unit} := \text{hnd\_rem}\%10\}$$

Таким чином, отримуємо LD-діаграму для конвертування вхідного числа в чотири різних змінних для тисяч, сотень, десятків та одиниць (рис. 6.34). Програма для зчитування та відображення значення АЦП на цифровому семисегментному індикаторі складається із шести блоків.

Рисунок 6.34 – Операція конвертації вхідного числа в чотири змінні для кожного розряду індикатора

Блок програми для організації автогенератора для вибору номера розряду індикатора подано на рис. 6.35.

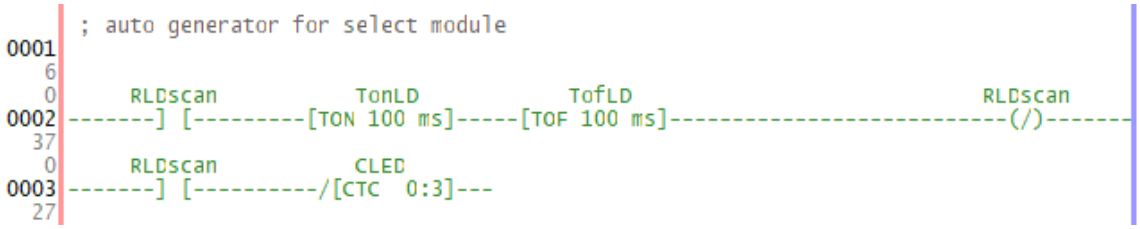


Рисунок 6.35 – Блок програми для організації автогенератора для вибору номера розряду індикатора

Блок програми для зчитування значення змінної ADC_v1 подано на рис. 6.36.

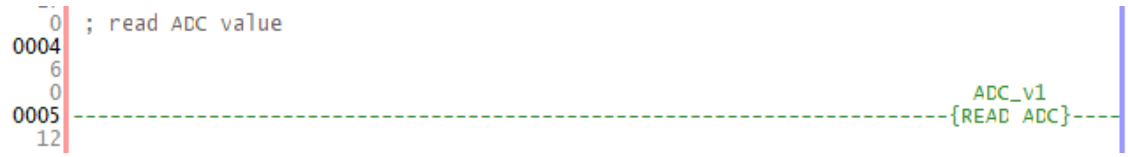


Рисунок 6.36 – Блок програми для зчитування значення змінної ADC_v1

Частина, що відповідає за конвертацію отриманого значення в окремі змінні, що відображатимуться на кожному окремому індикаторі подано на рис. 6.37.

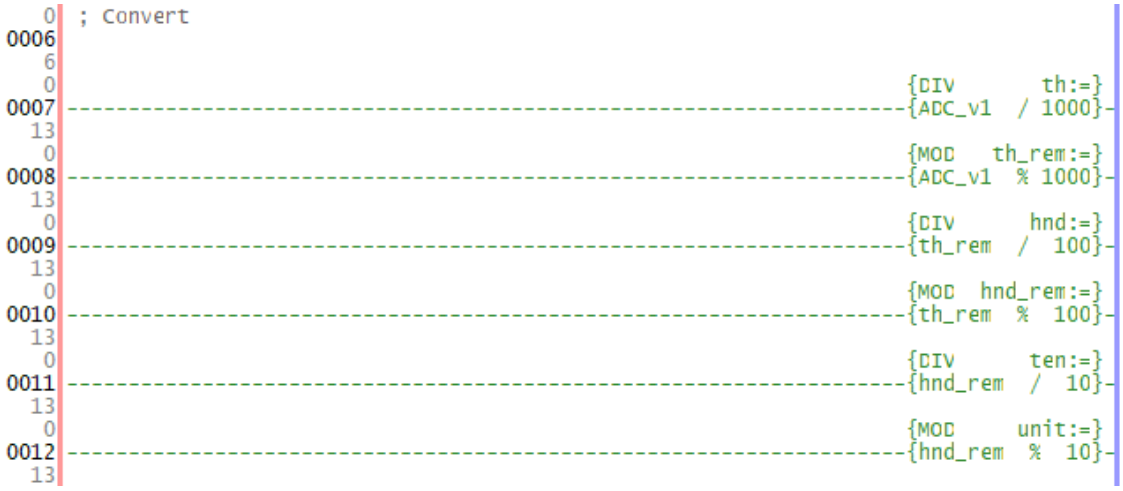


Рисунок 6.37 – Блок програми для зчитування значення змінної ADC_v1

На рис. 6.38 подано блок програми, що послідовно обирає та вмикає відповідний розряд індикатора, реалізуючи динамічний принцип виведення інформації. Номер поточного розряду зберігається в змінній CLED.

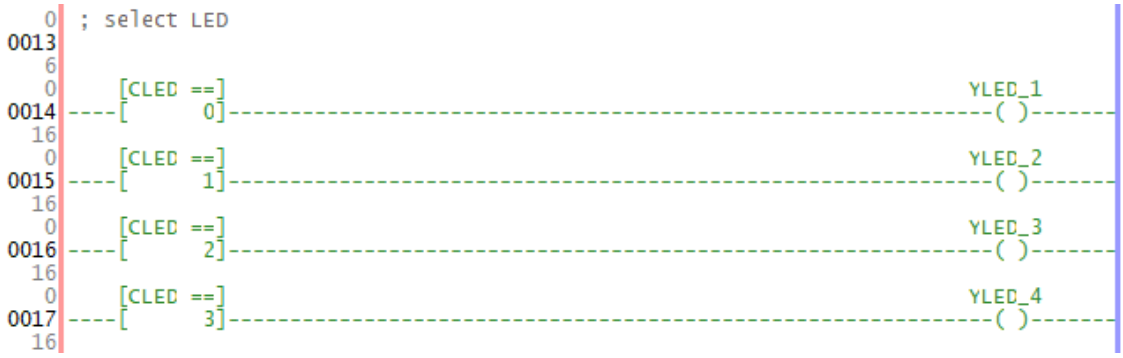


Рисунок 6.38 – Вибір та вмикання номеру розряду

Далі відбувається конвертування значень змінних для зберігання тисяч, сотень, десятків та одиниць в коди семисегментного індикатора (рис.6.39).

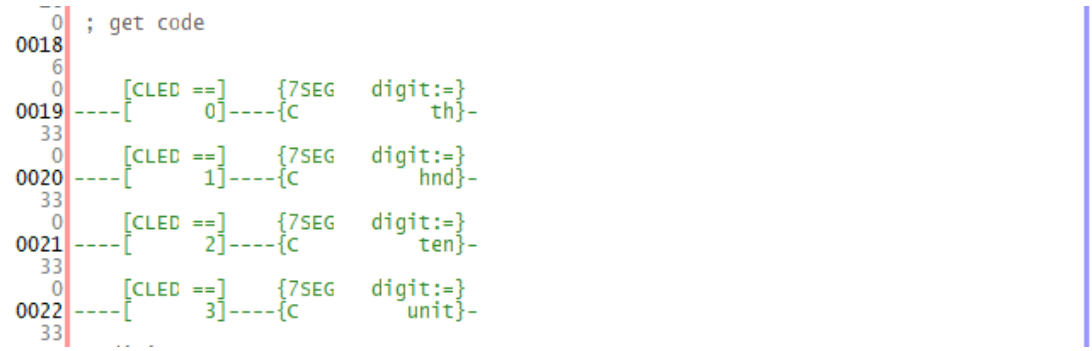


Рисунок 6.39 – Конвертування значень змінних в коди семисегментного індикатора

З поданого фрагменту коду можна бачити, що в змінну digit код для виведення на дисплей потрапляє у відповідності до обраного номеру розряду.

На останньому кроці відбувається вмикання сегментів індикатора у відповідності до конвертованого коду (рис. 6.40).

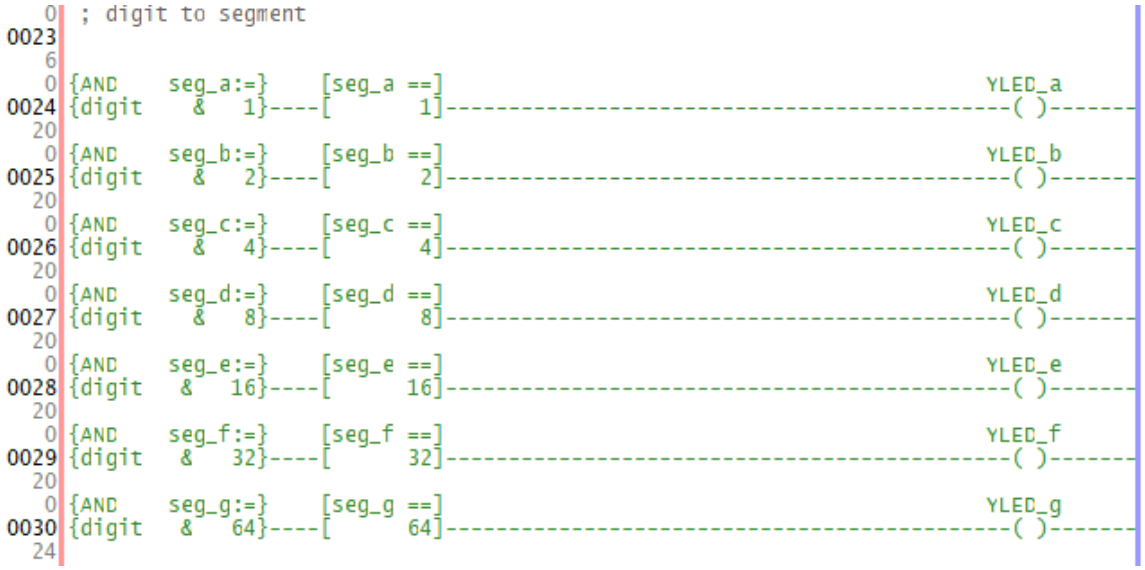


Рисунок 6.40 – Вмикання сегментів індикатора

Приклад роботи програми подано на рис. 6.41.

Рисунок 6.41 – Приклад роботи програми відображення значення АЦП

6.2.4 Конвертація даних від АЦП в значення вимірюваної величини

При вимірюванні будь-якого параметру, наприклад, напруги або температури, доцільно виводити реальне його значення, а не показання з АЦП. В такому випадку необхідно виконати перетворення показань з АЦП в реальні значення вимірюваного параметру.

Якщо максимальна напруга на вході АЦП не перевищує значення опорної напруги (Reference Voltage), то в даному випадку застосовується наступна формула:

$$V = \frac{Vr \cdot ADC}{ADCmax}, \tag{6.1}$$

де Vr – значення опорної напруги (Reference Voltage), В;

ADC – виміряне значення з АЦП;

$ADCmax$ – максимальне значення на виході АЦП, що залежить від його розрядності.

Наприклад, розглянемо наступні вихідні умови: значення опорної наруги дорівнює 5 В, розрядність АЦП становить 10 біт, а на виході АЦП присутнє значення 736.

Виконаємо перетворення значення АЦП в реальну напругу. Для цього означимо, що максимальне значення на виході АЦП $ADCmax = 1023$ ($2^{10} = 1024$). Таким чином, підставивши значення в (6.1) отримаємо:

$$V = \frac{Vr \cdot ADC}{ADCmax} = \frac{5 \cdot 736}{1023} = 3{,}59 \approx 3{,}6 \text{ В}.$$

В LDmicro всі операції цілочисленні, тому для роботи з числами із плаваючою комою необхідно спочатку знайти цілу частину, а потім дробову. Робиться це наступним чином. Для визначення цілої частини необхідно застосувати формулу:

$$V_{int} = int\left\{\frac{Vr \times ADC}{ADCmax}\right\}, \quad (6.2)$$

де int – оператор визначення залишку від ділення.

Для визначення дробової частини застосовується наступна формула:

$$V_{mod} = mod\left\{\frac{Vr \times ADC}{ADCmax}\right\}, \quad (6.3)$$

де mod – оператор визначення залишку від ділення.

Розглянемо приклад створення програми для вимірювання напруги та відображення отриманого значення на індикаторі. Вхідними умовами є:

– максимальна вимірювана напруга – 5 В;

– опорна напруга АЦП – 5 В;

– розрядність АЦП – 10 біт;

– кількість розрядів індикатора – 4 шт.

Для вирішення даної задачі застосовуємо формули (6.2) та (6.3) для отримання цілої та дробової частини від значення напруги, що була виміряна.

Відповідно до (6.2) виконаємо першу операцію множення:

$$V_{int} = Vr \times ADC. \quad (6.4)$$

За умовами задачі $Vr = 5$ В, тому в програмі LDmicro у вікно введення параметрів інструкції MUL в якості першого операнда вказуємо ADC_v1 (отримане значення на виході з АЦП), в якості другого – 5 В. На рис. 6.42 подано приклад заповнення параметрів.

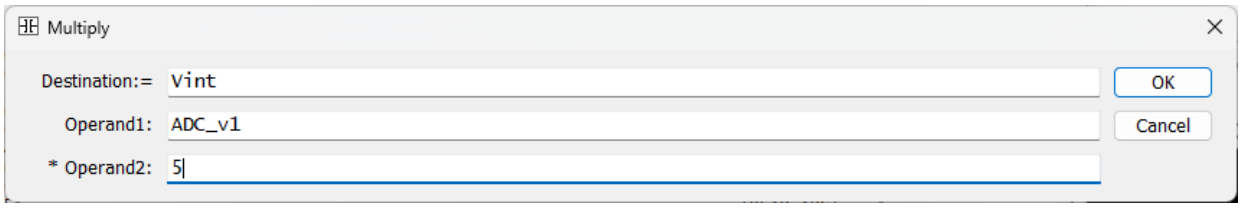


Рисунок 6.42 – Приклад заповнення параметрів інструкції MUL

В результаті отримаємо наступний вигляд діаграми (рис. 6.43).

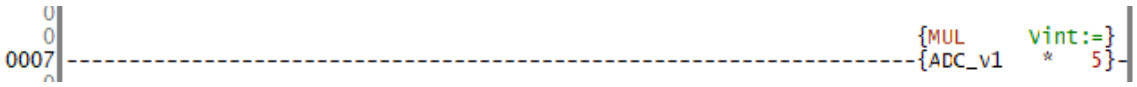


Рисунок 6.43 – Вигляд сходинкової діаграми, якщо використано інструкцію MUL

У відповідності до (6.2), наступною операцією є операція ділення:

$$Volt = \frac{Vint}{ADCmax}, \tag{6.5}$$

де $Vint$ – значення, яке отримане на першому кроці перетворення;

$ADCmax$ – максимальне значення на виході АЦП, що залежить від його розрядності.

За умовами задачі $ADCmax = 1023$, що є максимально можливим виміряним значенням величини за допомогою АЦП розрядністю 10 біт.

В результаті отримаємо наступну комбінацію значень операторів інструкції ділення DIV, що подано на рис. 6.44.

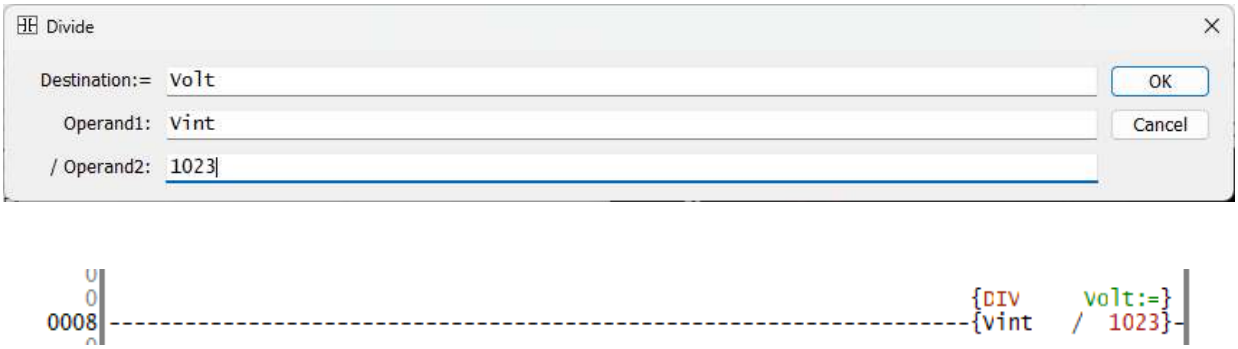


Рисунок 6.44 – Заповнення параметрів інструкції DIV

В результаті виконання інструкції DIV в змінній volt буде записано ціле значення результату ділення. Для отримання дробової частини скористуємось інструкцією MOD (рис. 6.45). Отримане в результаті ділення значення буде записано в змінну rest.

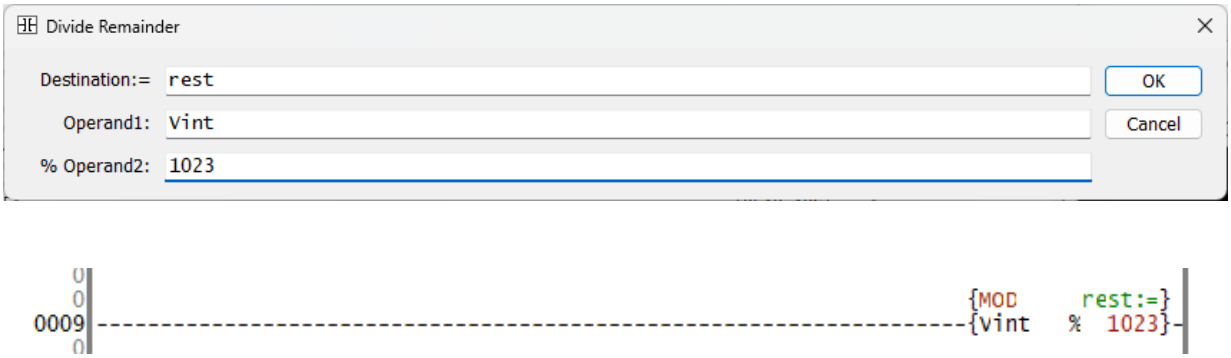
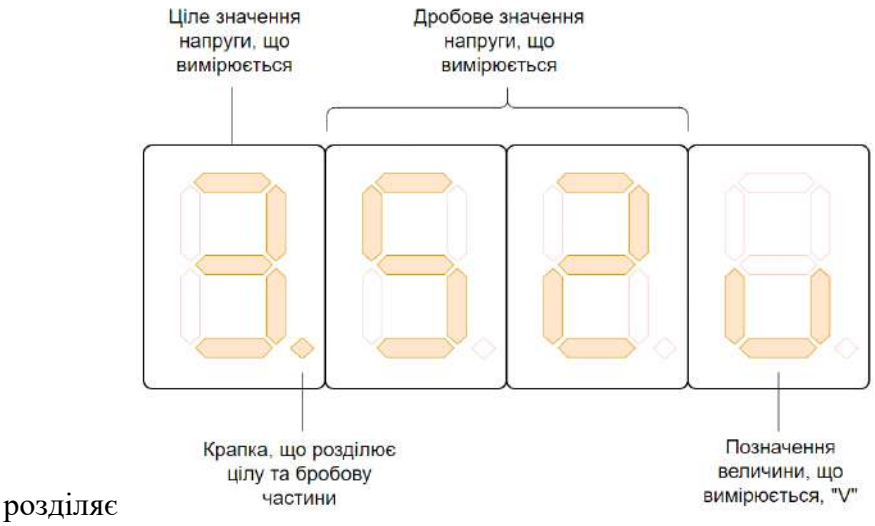


Рисунок 6.45 – Отримання дробової частини за допомогою інструкції MOD

Для відображення виміряного значення на чотирьохрозрядному дисплеї приймемо таку форму подання інформації, що подано на рис. 6.46.

З поданого рисунку можна бачити, що перший розряд чотирьохрозрядного індикатора виділено для відображення цілого значення виміряної напруги. За умовами задачі максимальна вимірювана напруга становить 5 В, тому одного розряду для цього вистачить. Наступні два розряди виділяються для відображення дробової частини виміряної напруги.

Крапка, що розділяє цілу та дробову частини повинна вмикатися відразу після першого розряду. Таким чином на індикаторі відображатиметься значення виміряної напруги з точністю до двох знаків після крапки. Для позначення типу величини, що вимірюється, відведено останній, четвертий розряд індикатора. В нашому прикладі на ньому відображатиметься літера «V».



розділяє

Рисунок 6.46 – Форма подання інформації на дисплеї

Для відображення дробового значення напруги застосовуємо принцип, описаний в попередньому підрозділі посібника, а саме, розкладення числа на розряди методом цілого ділення та визначення залишку. Для визначення сотень виконаємо ціле ділення значення змінної rest на 100, використовуючи інструкцію DIV (рис. 6.47).

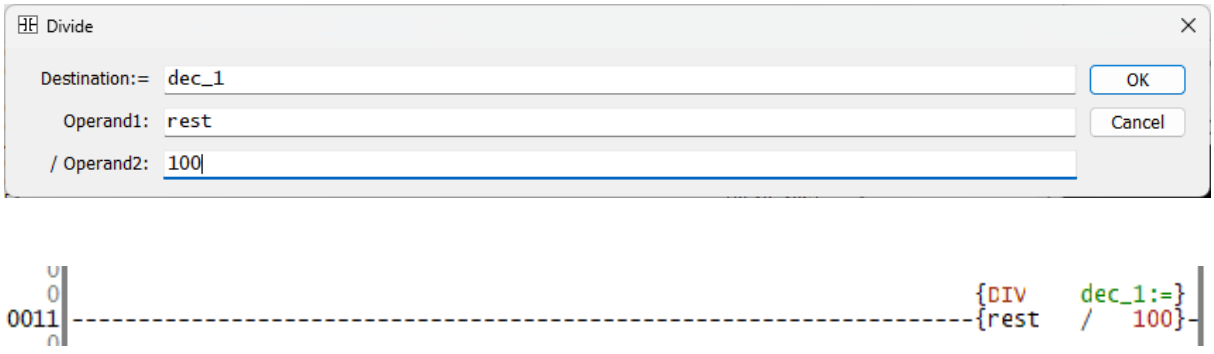


Рисунок 6.47 – Виконання цілого ділення значення змінної rest на 100

Визначимо залишок від ділення на 100 змінної rest за допомогою інструкції MOD (рис. 6.48).

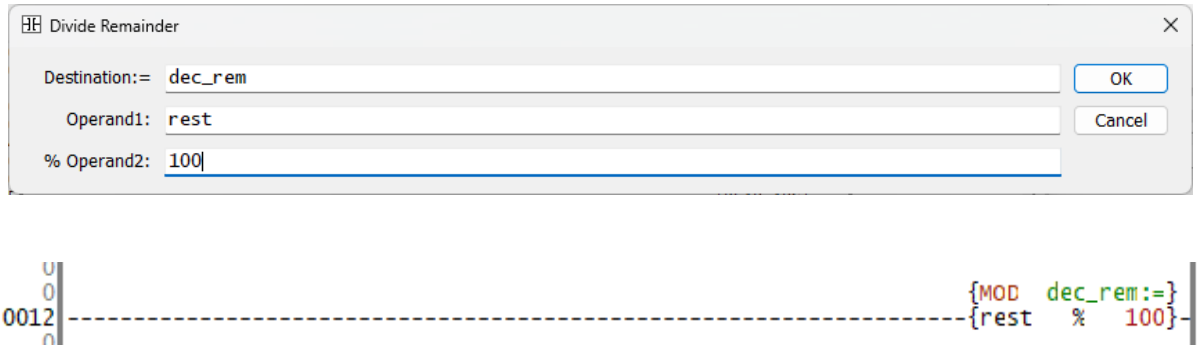


Рисунок 6.48 – Визначення залишку від ділення на 100 змінної rest за допомогою інструкції MOD

Отримане число поміщується в змінну dec_rest та використовується на наступному кроці для пошуку значення десятків в дробовій частині. Для цього виконаємо ціле ділення змінної dec_rest на 10 і занесемо отримане значення в змінну dec_2. На рис. 6.49 подано приклад створення відповідної сходинки в діаграмі LD.

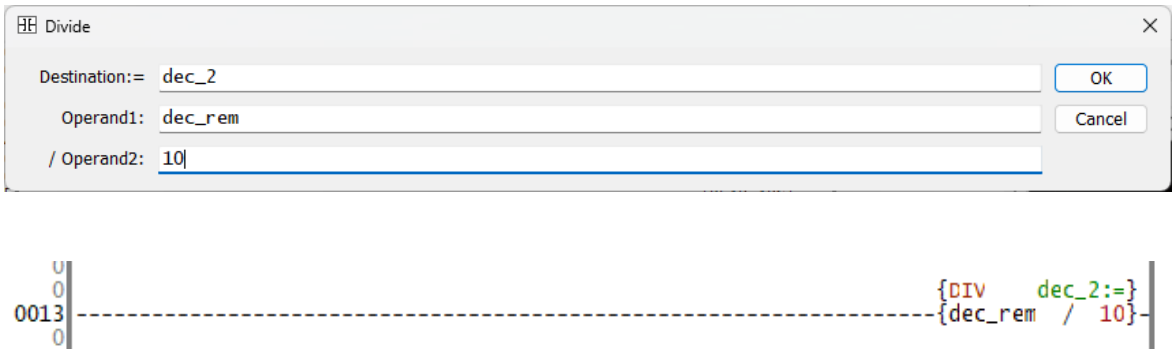


Рисунок 6.49 – Пошук значення десятків в дробовій частині

Поєднавши разом зазначені інструкції остаточно отримуємо блок діаграми LD для конвертації отриманого значення з АЦП та розкладання його на розряди для відображення на дисплеї (рис. 6.50).

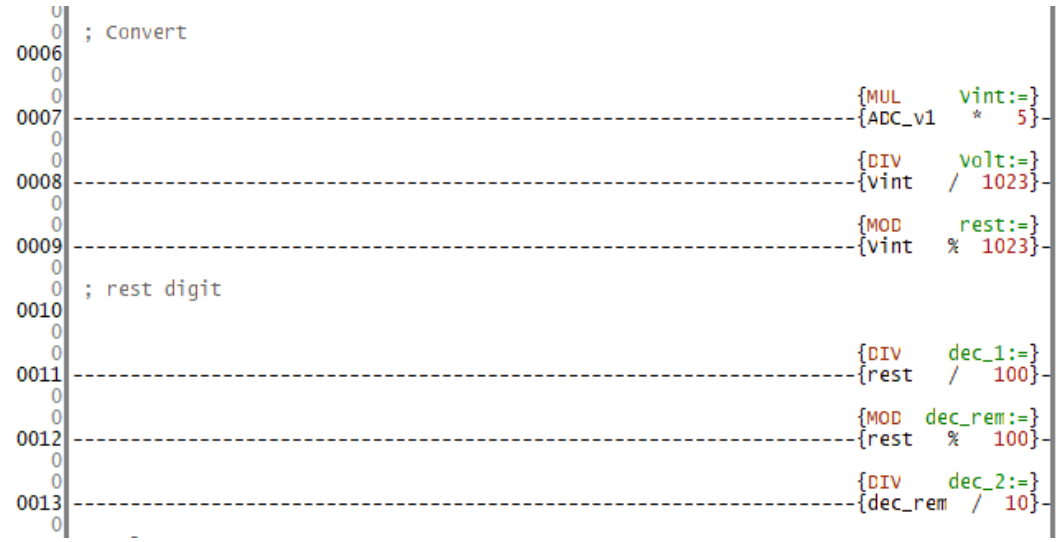


Рисунок 6.50 – Конвертації отриманого значення з АЦП в реальну величину

За прийнятою формою подання інформації на дисплеї після першого розряду необхідно увімкнути роздільну крапку. Для цього в програму необхідно додати рядок перевірки умови роботи з першим розрядом дисплею та, за умовою перевірки, увімкнути реле YLED_dp, що відповідає за відповідний сегмент індикатора. Даний приклад подано на рис. 6.51.

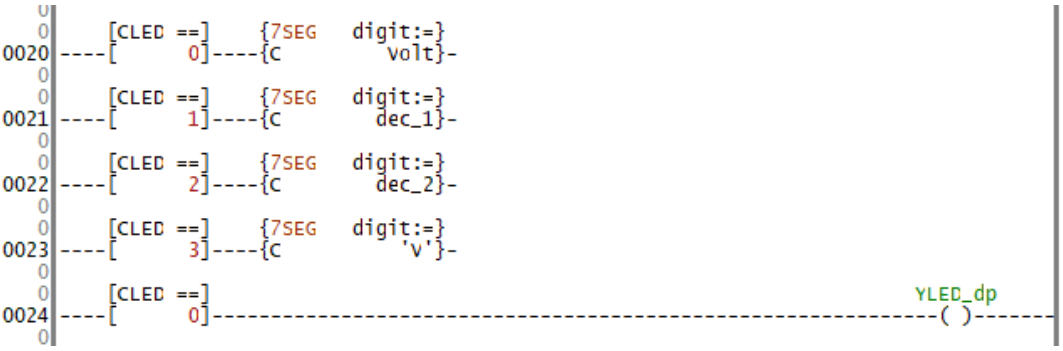


Рисунок 6.51 – Приклад увімкнення роздільної крапки після першого розряду індикатора

В результаті роботи програми отримуємо наступний вигляд інтерфейсу віртуального макету, що подано на рис. 6.52. На даному рисунку можна бачити обрані параметри АЦП у відповідності до поставленої задачі, та значення вхідної напруги на вимірюваному пристрої. Воно задається за допомогою ползункового регулятора, поряд з яким зазначено реальне значення напруги.

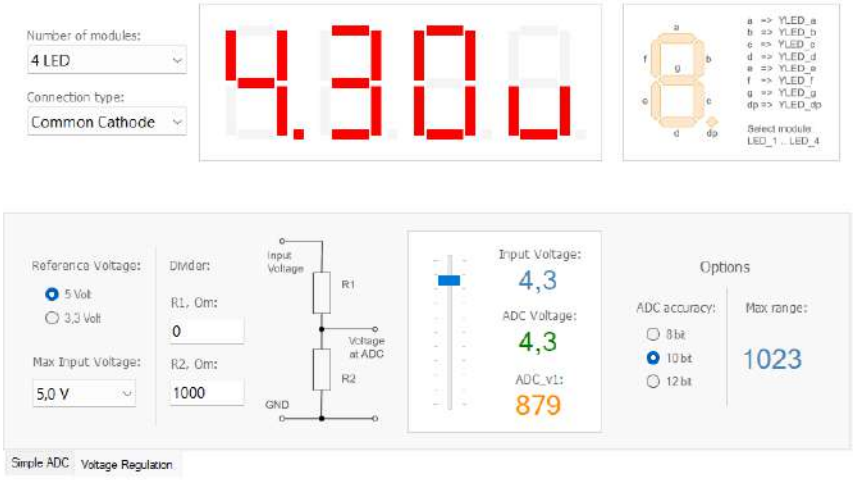


Рисунок 6.52 – Результат відображення виміряної напруги
за допомогою АЦП

Для перевірки роботи програми виконаємо ще одне вимірювання, результати якого подано на рис. 6.53.

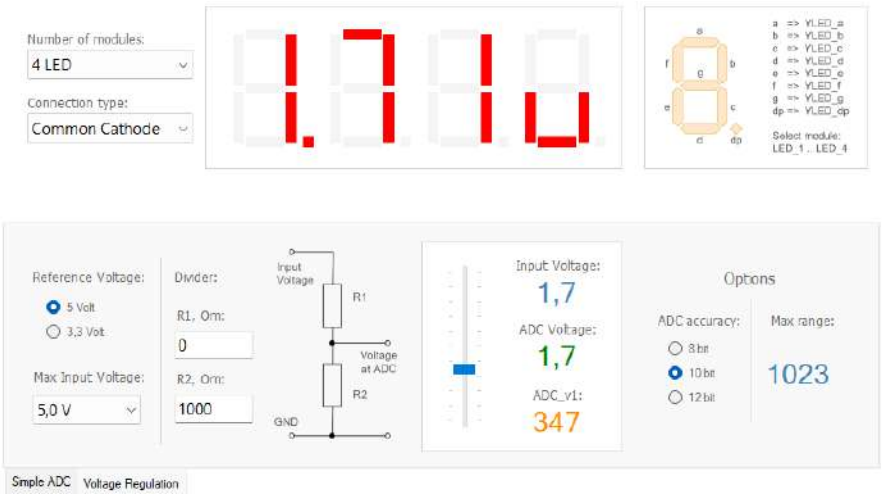


Рисунок 6.53 – Результати вимірювання напруги 1,7 В

Поданий приклад показав, що запропонований метод дозволяє вимірювати вхідне значення із заданою точністю, що досягається відповідною розрядністю АЦП. В результаті проведеного тестування можна бачити, що задане та виміряне значення напруги співпадають. Таким чином, розроблена нами програма працює правильно.

6.2.5 Використання резистивних дільників для вимірювання напруги, що перевищує максимально дозволене значення на вході АЦП

В процесі вимірювання напруги, величина якої перевищує робоче значення АЦП, в ПЛК та інших електронних пристроях можуть використовуватись блоки узгодження напруги, що вимірюється, та напруги на вході АЦП. Основою таких блоків дуже часто становлять резистивні дільники.

Принцип роботи дільника напруги полягає в пропорційному розподіленні напруги в середній точці включення резисторів, один з яких поєднано з мінусовою жилою живлення (загальною масою), а через другий відбувається підключення до точки вимірювання напруги. Схема будови резистивного дільника подана на рис. 6.54.

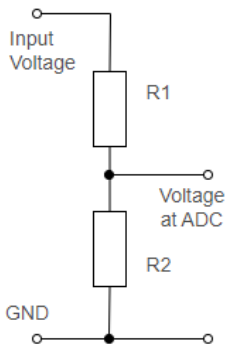


Рисунок 6.54 – Схема будови резистивного дільника

Напруга на виході дільника розраховується за формулою:

$$V_{ADC} = \frac{V_{in} \times R_2}{(R_1 + R_2)}, \tag{6.6}$$

де V_{ADC} – напруга на виході дільника (вхід АЦП), В;

V_{in} – вхідна напруга, В;

R_1, R_2 – опір відповідних резисторів (рис. 6.54), Ом.

Виконаємо розрахунок параметрів дільника, якщо необхідно виміряти напругу на вході, максимальне значення якої може становити 24 В. Для розрахунку

приймемо значення опору резистору R_1 рівним 4,7 кОм. Тоді, для визначення R_2 виконаємо наступні перетворення:

$$V_{ADC}R_1 + V_{ADC}R_2 = V_{in}R_2.$$

$$R_2(V_{in} - V_{ADC}) = V_{ADC}R_1.$$

$$R_2 = \frac{V_{ADC} \times R_{12}}{(V_{in} - V_{ADC})}. \quad (6.7)$$

Підставивши в (6.7) максимальну напругу на вході АЦП, максимальну вхідну напругу та значення опору резистора R_1, отримаємо:

$$R_2 = \frac{V_{ADC} \times R_{12}}{(V_{in} - V_{ADC})} = \frac{5 \times 4700}{(24 - 5)} = 1237 \text{ Ом}.$$

В стандартному ряді опорів знаходимо найближче менше значення, що становить 1,21 кОм. Розрахуємо максимальну вихідну напругу для обраного номіналу резистора:

$$V_{ADC} = \frac{24 \times 1210}{(4700 + 1210)} = 4{,}91 \text{ В}.$$

Отримане розрахункове значення напруги менше припустимого значення на вході АЦП. Таким чином, прийняті номінали резисторів підходять для вимірювання напруги до 24 В. Коефіцієнт резистивного дільника становить:

$$K_d = \frac{V_{ADC}}{V_{in}}. \quad (6.8)$$

Підставивши значення в (6.8) отримаємо:

$$K_d = \frac{4{,}91}{24} \approx 0{,}21.$$

Тепер, отримане значення виміряної напруги за допомогою АЦП необхідно поділити на цей коефіцієнт для отримання реального значення на вході пристрою.

Оскільки в LDmicro немає операцій з плаваючою точкою представимо число 0,21, як дріб:

$$0{,}21 = \frac{21}{100}.$$

Тоді, для визначення реальної напруги в LDmicro, необхідно скористатись виразом:

$$V_m = \frac{V \times 100}{21}, \tag{6.9}$$

де V – напруга за даними з АЦП без використання дільника, що обчислюється за (6.1).

Для того, щоб за допомогою LDmicro обчислити значення вхідної напруги за інформацією з АЦП, скористаємось (6.1):

$$V_m = \frac{\frac{Vr \cdot ADC}{ADCmax} \times 100}{21} = \frac{Vr \cdot ADC \cdot 100}{ADCmax \cdot 21}. \tag{6.10}$$

Якщо відомі значення Vr та $ADCmax$ (6.10) можна спростити:

$$V_m = \frac{5 \cdot ADC \cdot 100}{1023 \cdot 21} = \frac{ADC \cdot 500}{21483}. \tag{6.11}$$

В LDmicro за замовчуванням використовуються цілі 16-розрядні числа. Тому максимальне число, що може бути записано до змінної становить від -32768 до 32767. В новій версії програми додалась можливість збільшення розрядності змінної до максимального значення Int32, але все одно такі інструкції, як MUL, DIV, MOD не можуть працювати зі змінними Int32.

Враховуючи вищесказане, в формулі для визначення виміряної напруги необхідно чисельник та знаменник скоротити на 100, щоб отримати вихідне число, яке буде вміщуватись в діапазон чисел Int16. Таким чином, остаточна формула для вимірювання напруги в діапазоні від 0 до 30 В за допомогою 10-бітного АЦП, при опорній напрузі 5 В матиме вигляд:

$$V_m = \frac{ADC \cdot 5}{214}. \quad (6.12)$$

На рис. 6.55 подано приклад реалізації формули (6.12) засобами LDmicro.

Рисунок 6.55 – Приклад реалізації формули (6.11) засобами LDmicro

На рис. 6.56 подано фрагмент коду для формування цифр на екрані чотирьохрозрядного семисегментного індикатора.

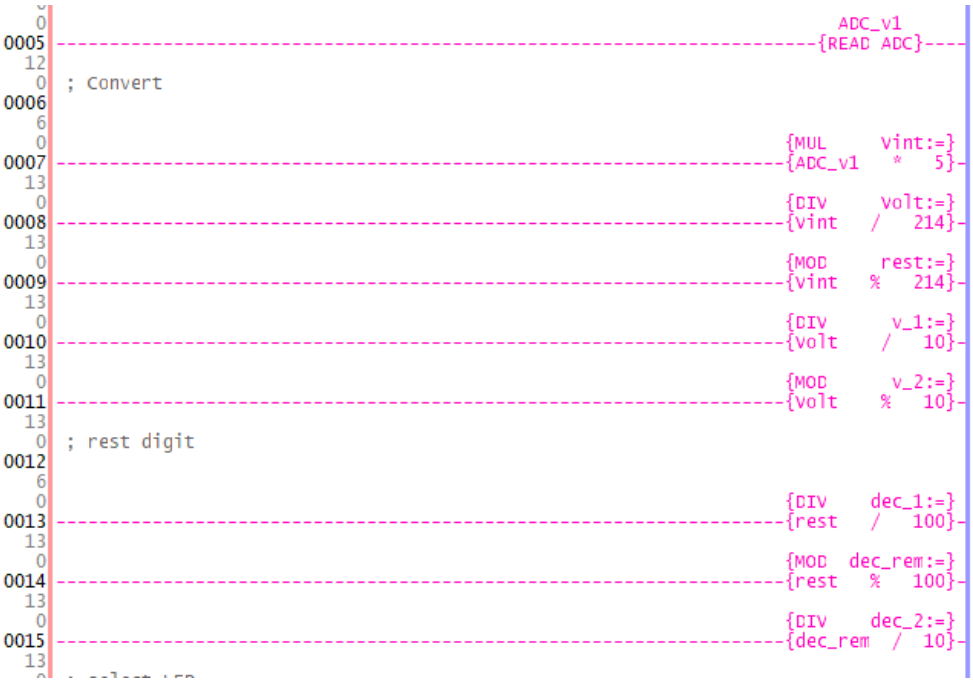


Рисунок 6.56 – Фрагмент коду для формування цифр на екрані чотирьохрозрядного семисегментного індикатора

Максимальна вимірювана напруга може досягати 30 В, тому для відображення цілого значення обираємо два розряди. На рис. 6.57 подано приклад

роботи програми та відображення виміряного значення напруги за допомогою віртуального макету.

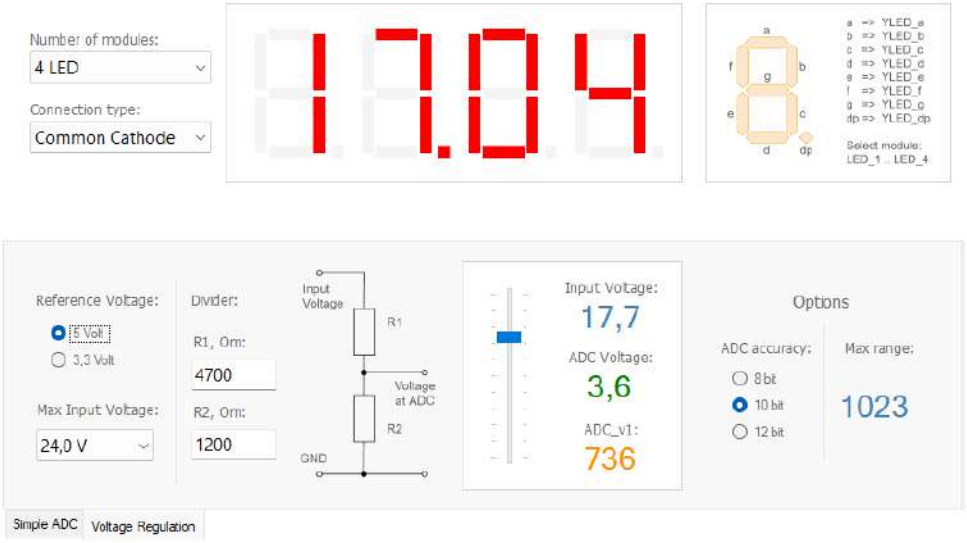


Рисунок 6.57 – Приклад роботи програми та відображення виміряного значення напруги за допомогою віртуального макету

З поданого рисунку можна бачити похибку в вимірюванні, що пов’язана з операціями перетворення та низькою розрядністю змінних в LDmicro.

6.3 Контрольні питання

1. Для чого використовуються цифрові індикатори?
2. Які типи індикаторів існують?
3. Яке призначення інструкції «Segments font converter»?
4. Як налаштувати параметри інструкції конвертування символів?
5. Поясніть принципи використання бітової маски.
6. Як увімкнути відповідні сегменти індикатора, використовуючи мову LD?
7. Для чого призначена інструкція «Read A/D Convertor»?
8. Як реалізується зчитування поточного значення АЦП на мові LD?
9. Побудуйте LD-діаграму, яка реалізує конвертацію вхідного числа в чотири змінні для кожного розряду індикатора.
10. Як відобразити виміряну напругу за допомогою АЦП на віртуальному макеті?

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кафедра комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки [Електронний ресурс] : офіційний веб-портал. – Режим доступу: https://tapr.nure.ua/. – Станом на 01.05.2023. – Назва з екрану.

2. Харківський національний університет радіоелектроніки [Електронний ресурс] : офіційний веб-портал. – Режим доступу: https://nure.ua/. – Станом на 01.05.2023. – Назва з екрану.

3. Digital Twins [Електронний ресурс] : IT-Enterprise. – Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/cifrovoj-dvojnik-digital-twin. – Станом на 01.05.2023. – Назва з екрану.

4. Digital Twins for Industrial Applications [Електронний ресурс] : Industrial Internet Consortium, a program of Object Management Group, Inc. ("OMG"). – Режим доступу: https://www.iiconsortium.org/pdf/IIC_Digital_Twins_Industrial_Apps_White_Paper_2020-02-18.pdf. – Станом на 01.05.2023. – Назва з екрану.

5. ISO/IEC Organization. 2019. ISO/IEC 21823-1 Internet of things (IoT) – Interoperability for iot systems – Part 1: Framework.

6. Festo [Електронний ресурс] : Festo. – Режим доступу: https://www.festo.com/ua/uk/. – Станом на 01.05.2023. – Назва з екрану.

7. Standardization for emerging technologies and innovations [Електронний ресурс] : JETI. – Режим доступу: https://jtc1info.org/technology/advisory-groups/jeti/. – Станом на 01.05.2023. – Назва з екрану.

8. ISO/TC 184. Ad Hoc Group: Data Architecture of the Digital Twin. [Електронний ресурс] : International Organization for Standardization . – Режим доступу: https://www.ththry.org/activities/2020/AdHocGroup_DigitalTwin_V1R8.pdf. – Станом на 01.05.2023. – Назва з екрану.

9. Details of the Administration Shell. Federal Ministry for Economic Affairs and Energy (BMWi) [Електронний ресурс] : ZVEI & Plattform Indutrie 4.0. . – Режим доступу: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details-of-theAsset-Administration-Shell-Part1.html. – Станом на 01.05.2023. – Назва з екрану.

10. Structure of the Administration Shell. Trilateral Perspectives from France, Italy and Germany. Ministry of Economy and Finances & Federal Ministry for Economic Affairs and Energy (BMWi) [Електронний ресурс] : Alliance Industrie du Futur, Piano Industria 4.0 & Plattform Industrie 4.0. – Режим доступу : https://www.plattformi40.de/I40/Redaktion/EN/Downloads/ Publikation/hm-2018-trilaterale-coop.html. – Станом на 01.05.2023. – Назва з екрану.

11. Details of the Administration Shell. Federal Ministry for Economic Affairs and Energy (BMWi) [Електронний ресурс] : ZVEI & Plattform Indutrie 4.0. – Режим доступу : https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/2018-details-ofthe-asset-administration-shell.html. – Станом на 01.05.2023. – Назва з екрану.

12. Eclipse IoT [Електронний ресурс] : Eclipse Foundation. – Режим доступу : https://projects.eclipse.org/projects/iot. – Станом на 01.05.2023. – Назва з екрану.

13. Невлюдов І. Ш. Технологія програмування промислових контролерів в інтегрованому середовищі CODESYS : навч. посіб. / І. Ш. Невлюдов, С. П. Новоселов, О. В. Сичова ; М-во освіти і науки України, Харків. нац. ун-т радіоелектроніки. – Харків : ХНУРЕ, 2019. – 264 с. : іл. – ISBN 978-966-659-265-4.

14. Невлюдов І.Ш. Пневматичні пристрої та засоби автоматизації мехатронних систем: Навчальний посібник / І.Ш. Невлюдов, Л.О. Кривопляс-Володіна, С.П. Новоселов, О.В. Сичова. – Харків: ФОП Панов А.М., 2020 . – 256 с. DOI: 10.30837/978-617-7859-58-0. ISBN 978-617-7859-58-0.

15. Novoselov S., Sychova O. Automated system of technological preparation of production. Intelligent computer-integrated information technology in project and program management : Collective monograph edited by I. Linde, I. Chumachenko. Riga : ISMA. pp.207-224, 2020. DOI: https://doi.org/10.30837/ MMP.2020.207. ISBN 978-9984-891-15-6.

16. Невлюдов І.Ш. Електропневмоавтоматичні приводи в автоматизованих системах керування: Навчальний посібник / І.Ш. Невлюдов, Л.О. Кривопляс-Володіна, С.П. Новоселов, О.В. Сичова. – Харків: ХНУРЕ, 2021 . – 292 с. DOI: 10.30837/978-966-659-332-3. ISBN 978-966-659-332-3.

Навчальне видання

НЕВЛЮДОВ Ігор Шакирович
НОВОСЕЛОВ Сергій Павлович
СИЧОВА Оксана Володимирівна

ЗАСТОСУВАННЯ ЦИФРОВИХ ДВІЙНИКІВ ТЕХНІЧНИХ ЗАСОБІВ АВТОМАТИЗАЦІЇ ДЛЯ РОЗРОБЛЕННЯ ПРОГРАМНО-ТЕХНІЧНИХ КОМПЛЕКСІВ АСУ ТП

Навчальний посібник

Формат 60х84/16.
Ум. друк. арк. – 15,5. Наклад 200 пр. Зам. № 01-06.

Видавництво та друк
ФОП Іванченко І. С.
пр. Тракторобудівників, 89-а/62, м.Харків, 61135.
Тел.: +38-050-40-243-50.
Свідоцтво про внесення суб'єкта видавничої справи
до державного реєстру видавців, виготівників та розповсюджувачів
видавничої продукції серія ДК №4388 від 15.08.2012 р.
www.monograf.com.ua