ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КЕРУЮЧИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

Кафедра спеціалізованих комп'ютерних систем

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних та самостійних робіт з дисципліни

«КОНТРОЛЕРИ В СУЧАСНИХ СИСТЕМАХ ЗАЛІЗНИЧНОЇ АВТОМАТИКИ І ТЕЛЕМЕХАНІКИ»

Частина 1

Харків – 2019

Методичні вказівки розглянуто і рекомендовано до друку

на засіданні кафедри спеціалізованих комп’ютерних систем 25 лютого 2019 р., протокол № 9.

Розглядається вмикання семисегментного індикатора (ССІ) та ряду пристроїв комп'ютерної інженерії транспорту.

Призначено для студентів факультету ІКСТ зі спеціальності 123 «Комп’ютерна інженерія» другого рівня вищої освіти (магістр) усіх форм навчання.

Укладач

доц. В. М. Бутенко

Рецензент

проф. В. І. Мойсеєнко

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних та самостійних робіт
з дисципліни
«КОНТРОЛЕРИ В СУЧАСНИХ СИСТЕМАХ ЗАЛІЗНИЧНОЇ АВТОМАТИКИ І ТЕЛЕМЕХАНІКИ»

Частина 1

Відповідальний за випуск Бутенко В. М.

Редактор Буранова Н. В.

Підписано до друку 12.03.19 р.

Формат паперу 60х84 1/16. Папір писальний.

Умовн.-друк. арк. 5,5. Тираж 50. Замовлення №

Видавець та виготовлювач Український державний університет
залізничного транспорту,
61050, Харків-50, майдан Фейєрбаха, 7.
Свідоцтво суб’єкта видавничої справи ДК № 6100 від 21.03.2018 р.

ЗМІСТ

ВСТУП

Вивчення практичних аспектів створення сучасних систем залізничної автоматики (ЗАТ) засобами мікропроцесорних контролерів – кропіткий процес високого рівня технологічності. Починається він з моделювання зазначених систем у середовищі програмування Unity Pro 3.0 або вище для створення елементів технологічного спрямування залізничного транспорту на прикладі класичних компонентів залізничної автоматики як складової частини систем забезпечення руху поїздів [1].

Курс розглядається як загальнотехнічна компонента галузевої складової та передбачає ознайомлення з відповідними методиками створення і дослідження моделей і підпрограм відповідно до робочої програми дисципліни, затвердженої кафедрою. Для створення апаратно-програмних модулів комп'ютерної інженерії систем використовується множина мов програмування, яка відповідає EN 61131/IEC 61131-3:2013. Побудова та дослідження моделей і програмного забезпечення при розробленні нових складних технічних систем ЗАТ часто є значною частиною у загальній сукупності робіт, що проводяться на початкових фазах проекту рішення.

Для виконання лабораторних робіт у лабораторіях кафедри СКС університету встановлено програмне забезпечення (ПЗ) Unity Pro XL v 3.0, надане фірмою Шнайдер Електрик Україна, (http://www.schneider-electric.ua), що функціонує у навчально-демонстраційному режимі. Для самостійної роботи студентів можливе застосування будь-якої версії легального ПЗ Unity Pro. Однак у випадку невідповідності версії на студентському та лабораторному ПК саме на студента покладається відповідальність за демонстрацію самостійно розроблених програмних модулів при виконанні лабораторної роботи. Практика показує, що краще або застосовувати версію 3.0, або приносити переносний ПК для демонстрації самостійно розроблених схем-програм практикуму.

ЛАБОРАТОРНА РОБОТА 1
Установлення, інтерфейс і створення нових проектів мовою FBD в Unity Pro, налагодження елементарних функцій алгебри логіки

Мета роботи: набуття практичних навичок установлення, створення проектів і програм мовою FBD (Function Block Diagram) в Unity Pro, налагодження елементарних функцій алгебри логіки (ФАЛ) з кодуванням сигналів.

Обладнання та ПЗ: персональна електронно-обчислювальна машина (ПЕОМ) із системним програмним забезпеченням (СПЗ) Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або вище.

Хід виконання роботи

1 Вивчити весь теоретичний матеріал для досягнення мети лабораторної роботи (ЛР).

2 Створити проект за послідовністю, викладеною нижче, та ввести програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити «заготовку» до лабораторної роботи (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

6 Оформити висновки й отримати бали за ЛР у викладача.

Послідовність створення проекту

Unity Pro – це програмне середовище конфігурації, програмування, налагодження і діагностування виконавчої системи та промислових контролерів, яке є результатом розвитку двох програмних продуктів: PL7 Pro – середовище програмування першої групи контролерів, Concept – середовище програмування другої групи контролерів.

Виконавча система Unity – це програмне забезпечення, яке виконується у контролері. Виконавча система базується на операційній системі (ОС) Unity, яка вже є у завантаженні

програмного логічного контролера (ПЛК) та бере участь в усіх операціях, але нас цікавить моделювання роботи схем булевої алгебри в зазначеному середовищі.

На робочому столі знайти ярлик програми Unity Pro, запустити його (або через меню «Пуск»–програми). Потім створюємо (**File-New**) на диску ПЕОМ (у лабораторії кафедри D:\SE) новий проект (використавши для його назви тільки цифри та латинські літери ASCII таблиці) та обираємо процесорний модуль **TSX P57 2634M**, вибравши тип шасі (X-Bas) і, **за бажанням,** скомпонувавши ПЛК модулями введення-виведення за допомогою панелі, яка розташована у лівому нижньому вікні (рисунок 1), для прикладу навчального класу **Schnider Electric УкрДУЗТ** – модулі дискретні TSX DEY 32D2K TSX, DSY 32T2K та аналоговий TSX AEY 414.

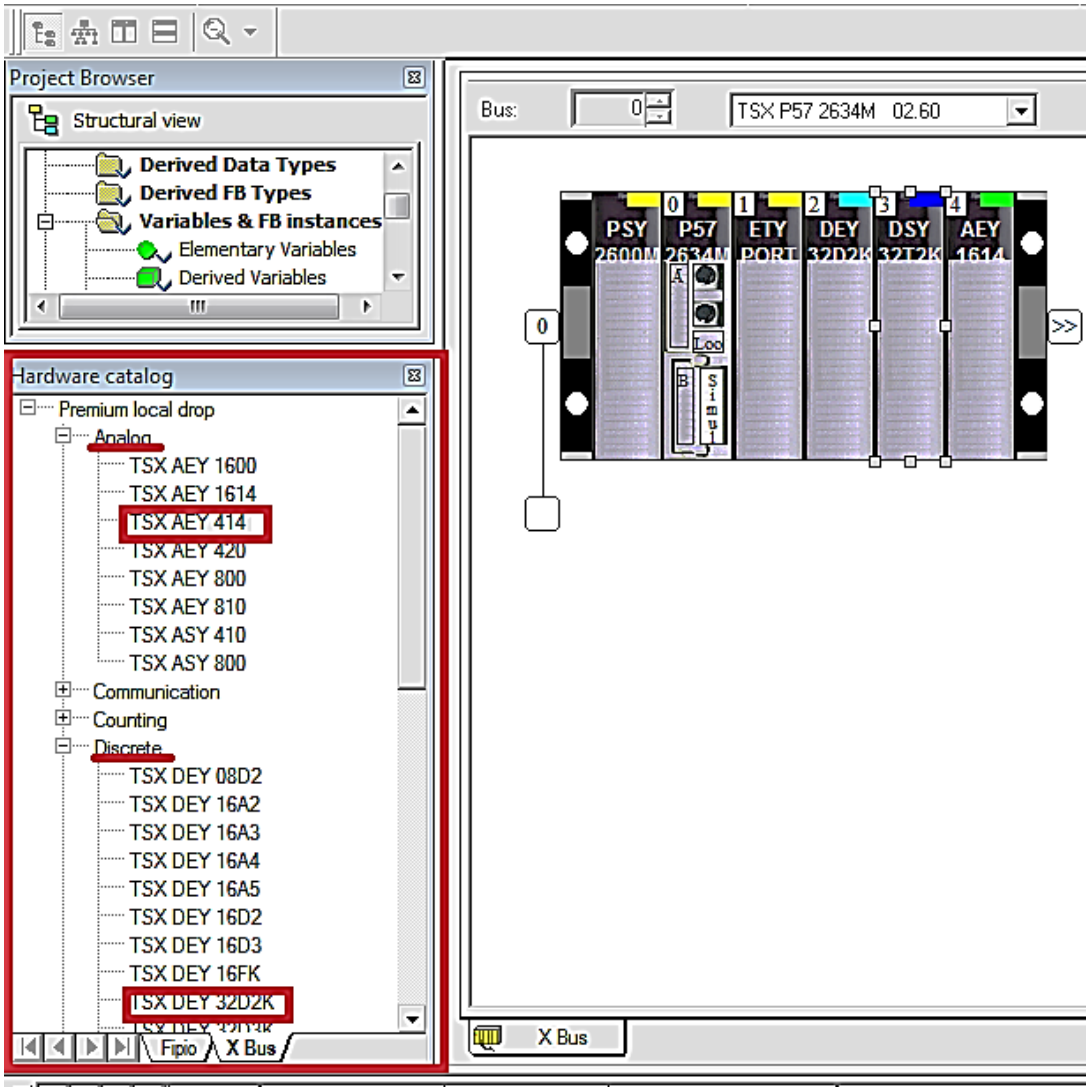


Рисунок 1 – Конфігурування ПЛК у проекті

Після завершення компонування заданих проектом або планом ЛР модулів (як показано для ЛР на рисунку 2) активізуємо модуль ETY PORT і відкриваємо його для налаштування (рисунок 3). За бажанням, аналогічно налаштовуються інші модулі ПЛК.

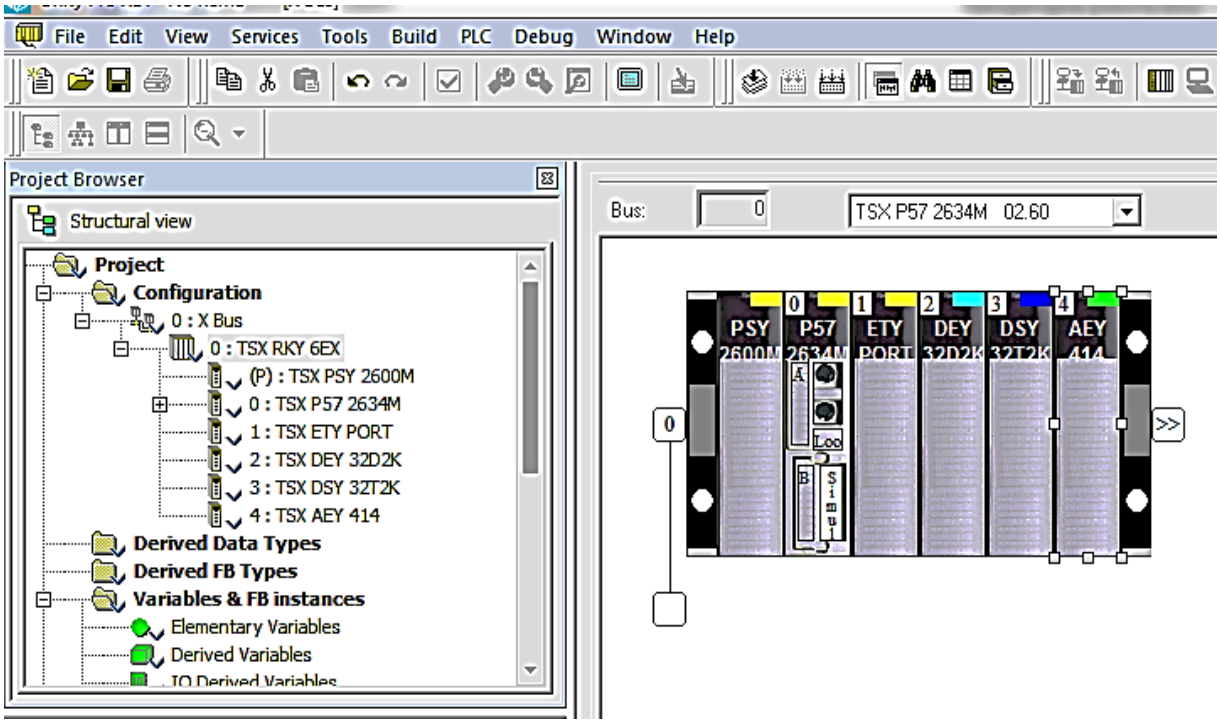


Рисунок 2 – Вибір модуля ПЛК

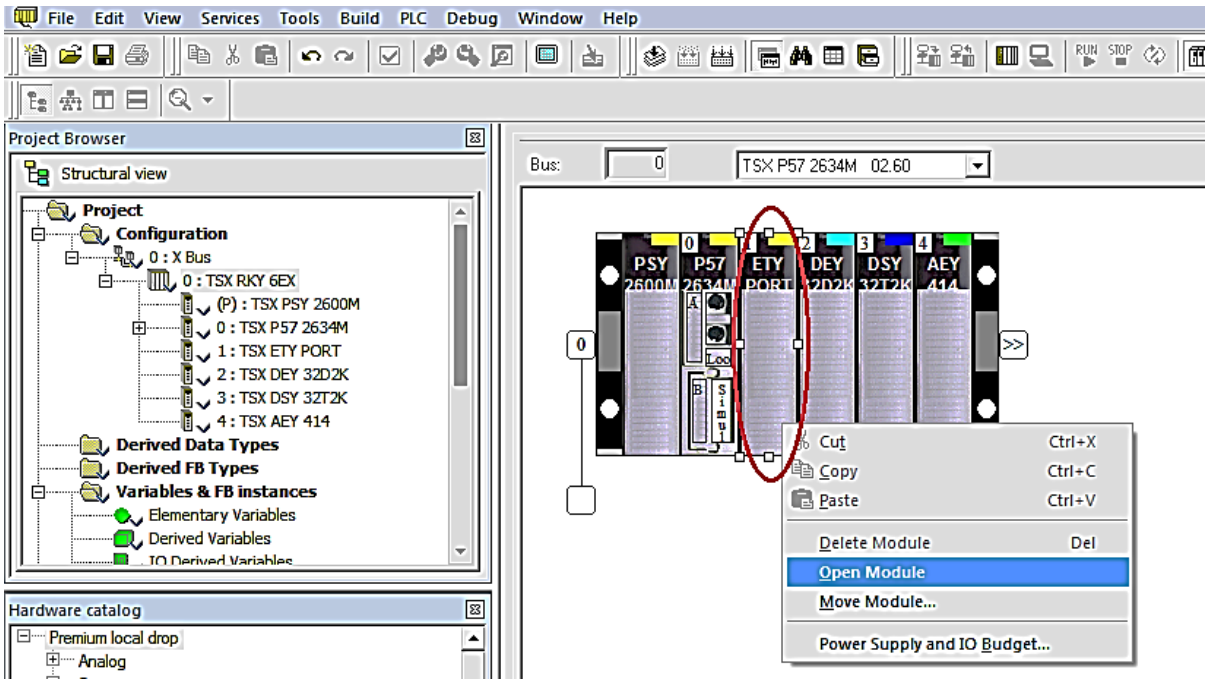
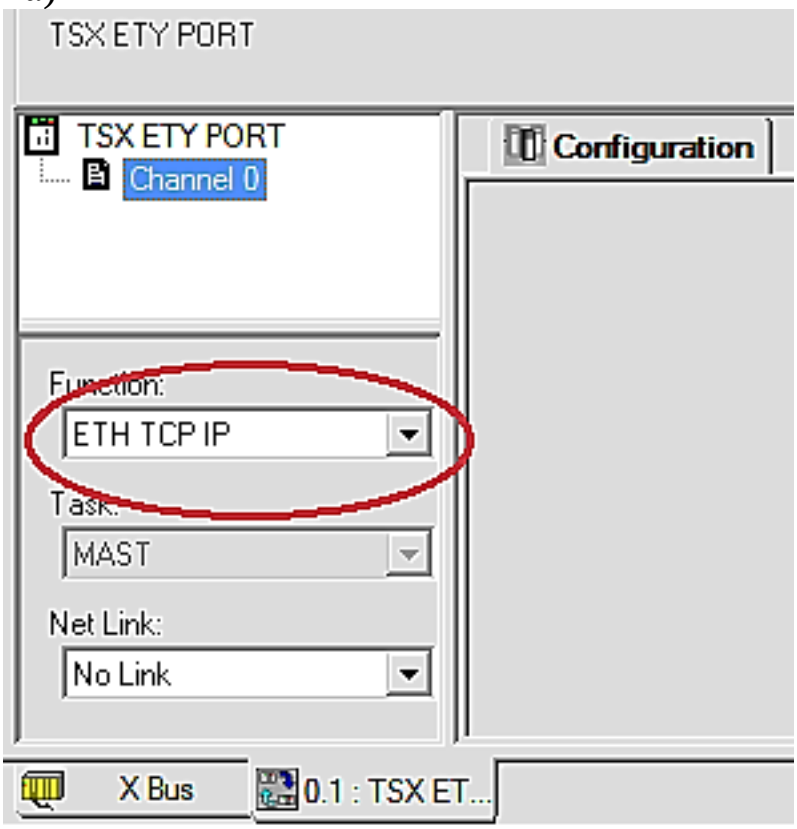


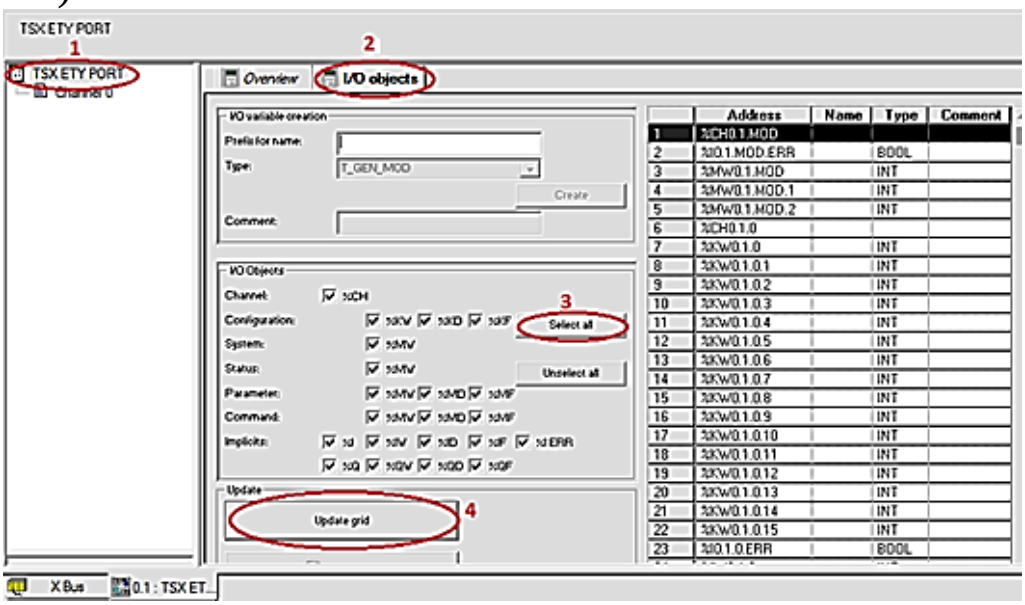
Рисунок 3 – Налаштування модуля TSX ETY PORT

У цьому самому вікні обираємо TSX ETY PORT (рисунок 4, а) (Open Module –> Channel 0 –> знизу Function), перемикаємо на ETH TCP IP і, зробивши налаштування у закладці I/O objects (рисунок 4, б), зберігаємо налаштування.

а)



б)



а – вибір протоколу ETH TCP IP;
б – налаштування у закладці I/O objects
Рисунок 4 – Налаштування модуля TSX ETY PORT

Далі переходимо в меню PLC (англ. *Programmable Logic Controller (PLC)*), як показано на рисунках 5, 6.

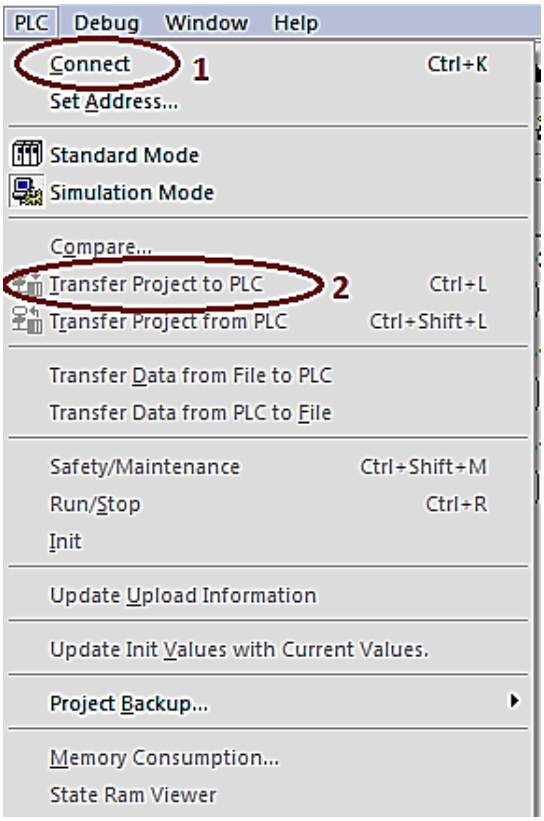


Рисунок 5 – Зв'язок із контролером

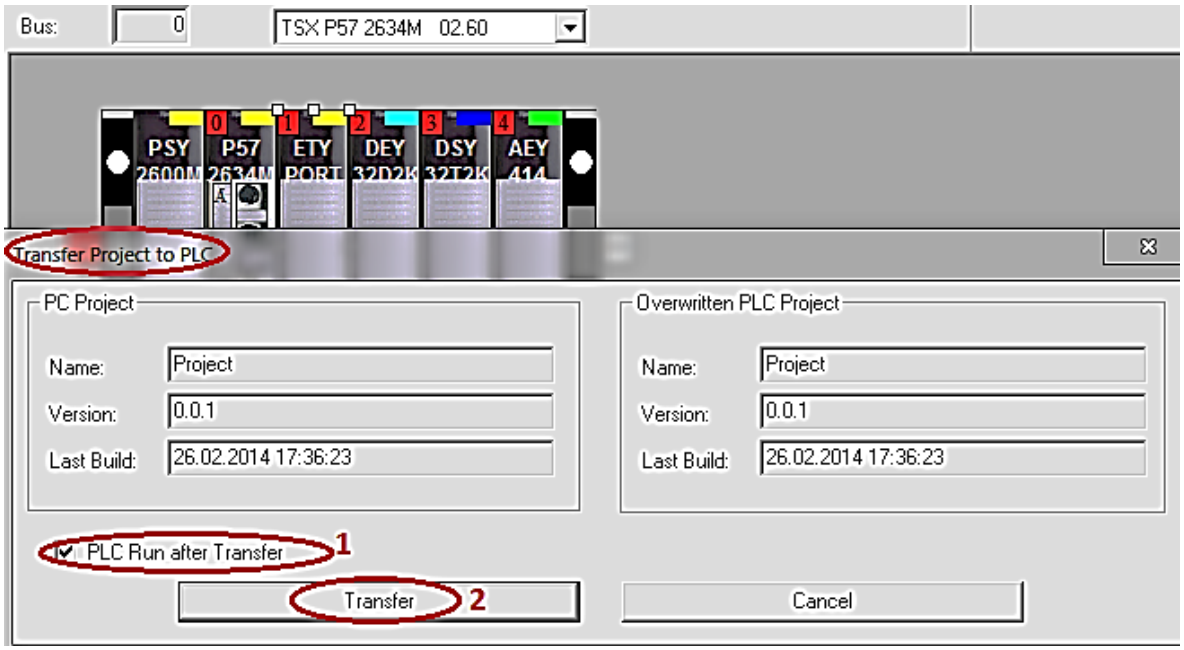


Рисунок 6 – Передача проекту до ПЛК/PLC

У випадку відсутності зв’язку перевірити встановлення опції «Simulation mode», якщо ПЛК не підключений до ПК.

Для того щоб зупинити програму, обираємо меню PLC і натискаємо Stop, потім знову – PLC та обираємо Disconnect.

Послідовність дій введення типового завдання проекту

Типове завдання ЛР 1. Розробити схему-програму мовою FBD кодування стану чотирьох кнопок (булеві змінні a0, a1, a2, a3) у дворозрядний код (булеві змінні x1, x0) та візуалізувати коди x1x0 (00, 01, 10, 11) на стандартному семисегментному індикаторі (рисунок 7, а) із сегментами (a, b, c, d, e, f, g) у вигляді цифр 0, 1, 2, 3 відповідно до значень, показаних у таблиці 1.

Таблиця 1 – Відповідність значень булевих змінних у різних елементах схеми типового завдання ЛР 1

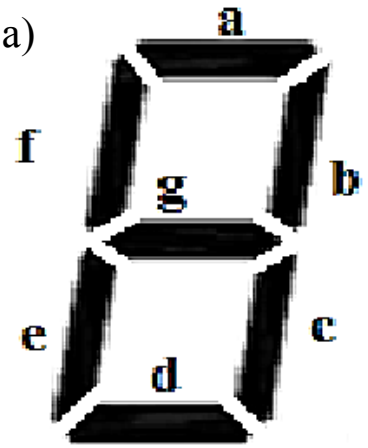
	Кнопки				Код		Сегменти							Цифра
змінні	a0	a1	a2	a3	x1	x0	a	b	c	d	e	f	g	
Зна-чен-ня	1	–	–	–	0	0	1	1	1	1	1	1	0	0
	–	1	–	–	0	1	0	1	1	0	0	0	0	1
	–	–	1	–	1	0	1	1	0	1	1	0	1	2
	–	–	–	1	1	1	1	1	1	1	0	0	1	3

За отриманими раніше аналітичними виразами визначення a0= $\overline{a1 + a2 + a3}$ та кодування x0=a3+a1, x1= $\overline{a0}\,\overline{a1}$ сформовано a=d=$\overline{\overline{x1}\,x0}$, b=1, c= $\overline{x1\overline{x0}}$, e=$\overline{x0}$, f= $\overline{x0}\,\overline{x1}$, g=x1. На рисунку 7, а зображено семисегментний індикатор з позначенням сегментів та комбінаційну схему перекодування a0, a1, a2, a3 в x0 та x1 із увімкненням усіх сегментів (рисунок 7,б).

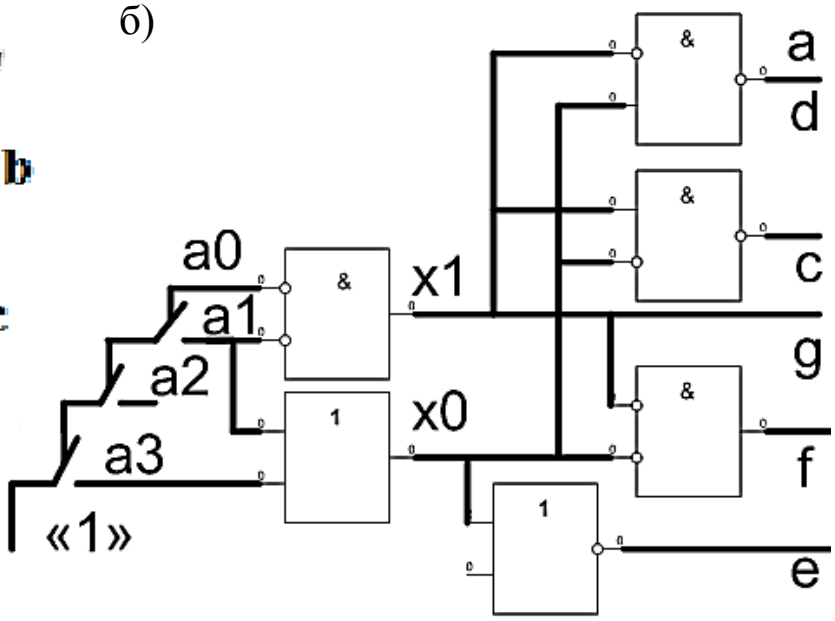
Під час цієї роботи було використано ПЗ Unity Pro 3.0, і саме скріншоти з нього наведено в тексті, але в окремих випадках використовувались зображення з інших версій з подібними повідомленнями та ідентичними функціями.

Наступний крок виконання ЛР – перейти у вікно **Project Browser** (один з варіантів – комбінацією клавіш Alt+1), на вкладці якого знайти рядок **Elementary Variables,** вибрати його та ввести 14 змінних типу **BOOL,** як показано на рисунку 8.

а)



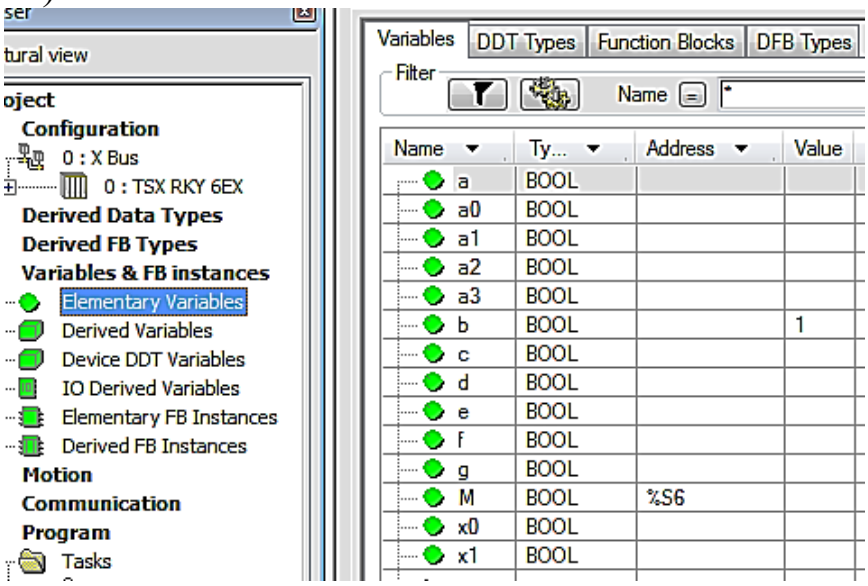
б)



а – зображення семисегментного індикатора з позначенням сегментів; б – схема кодування a0, a1, a2, a3 із увімкненням сегментів

Рисунок 7

а)



б)

а – зображення вікна створення змінних типу BOOL;
б) спрощене зображення семисегментного індикатора та двох кнопок шин сигналів x0 та x1

Рисунок 8

Ще один крок виконання ЛР – у вікні **Project Browser** (один із варіантів – комбінацією клавіш Alt+1), на вкладці знайти рядок Program, обрати Tasks, потім MAST, далі Section і правою кнопкою миші викликати контекстне меню і вибрати New Section (рисунок 9).

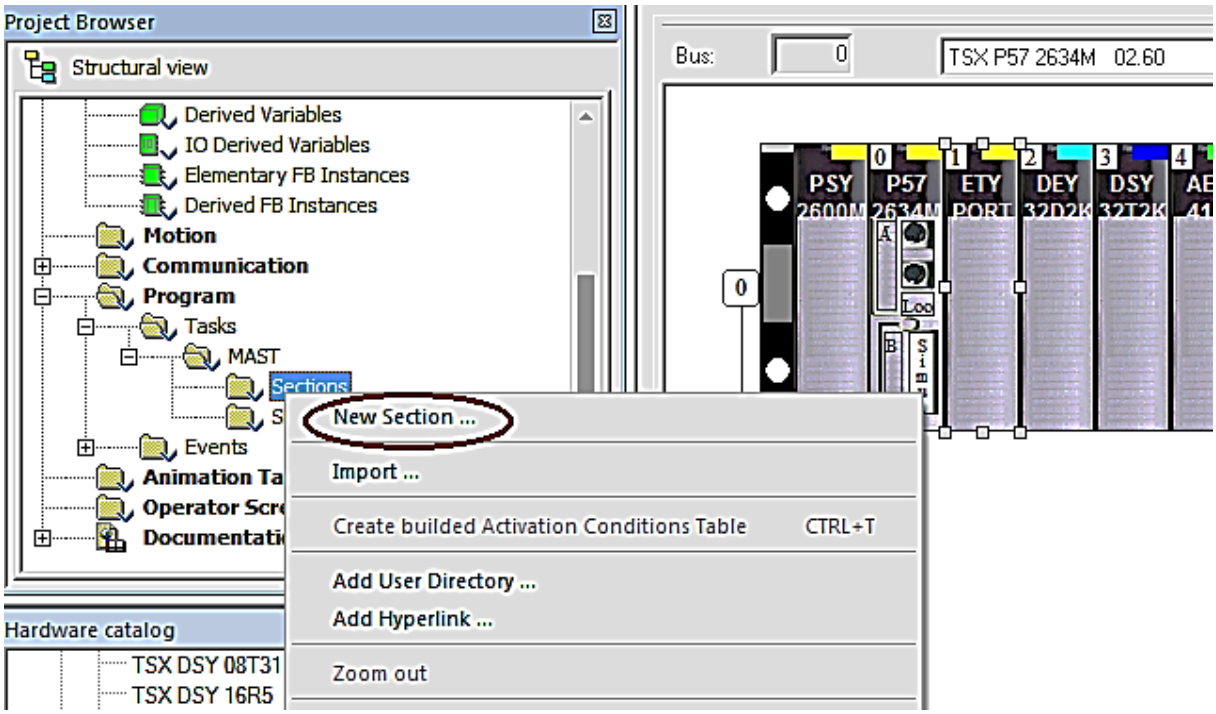


Рисунок 9 – Створення секції програми в групі MAST

Далі вводимо ім’я секції (використовувати тільки латинські літери з можливим додаванням цифр, наприклад, **laba01**) та обираємо параметр мови програмування (FBD) (рисунок 10).

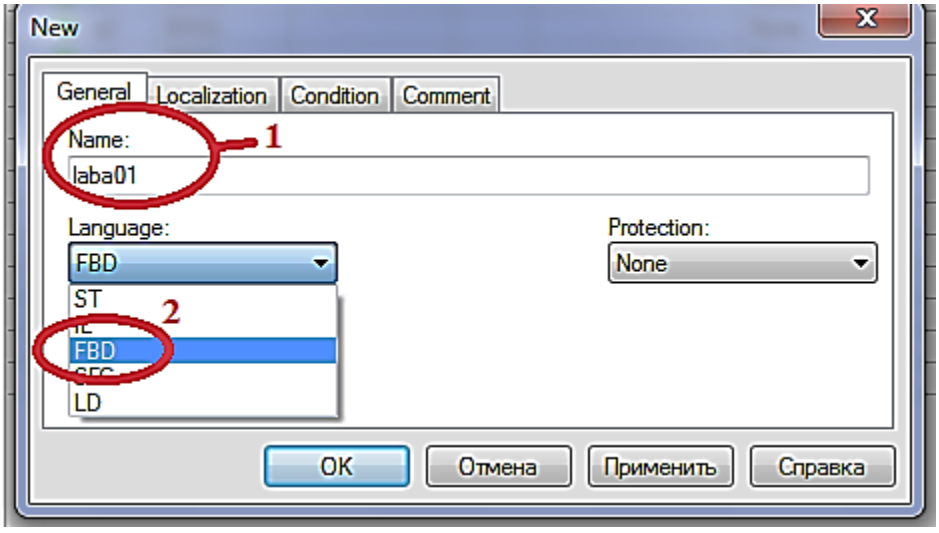


Рисунок 10 – Введення назви секції та вибір мови програмування

За допомогою кнопок (наприклад ) на верхній допоміжній панелі будуємо схему-програму, як показано на рисунку 11. Для вибору блоків **OR**-АБО, **AND**-ТА можливе використання комбінації клавіш Ctrl+I.

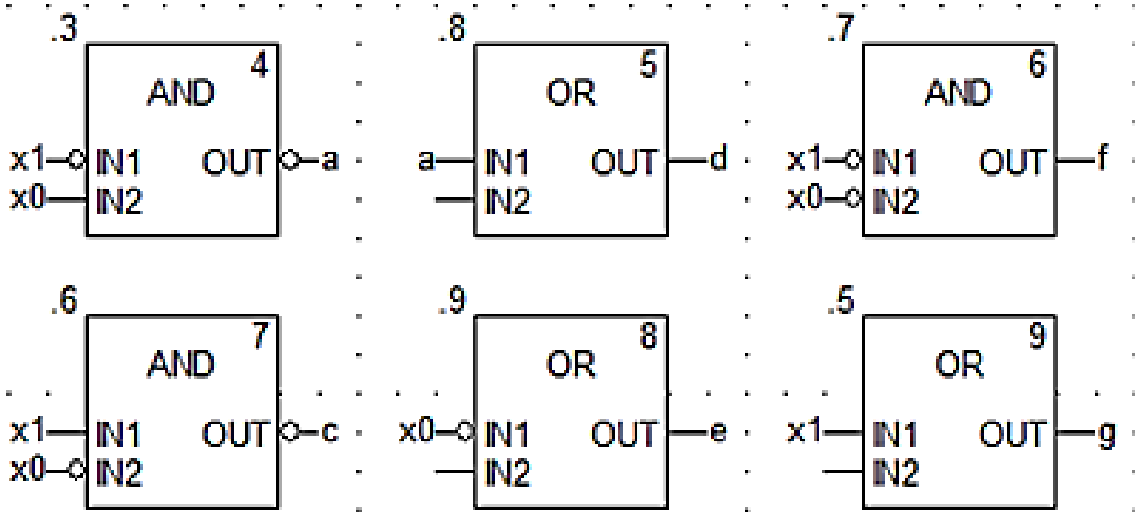


Рисунок 11 – Приклад схеми-програми мовою FBD

Після компіляції (Ctrl+B), з'єднання з ПЛК (Ctrl+K), передачі/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) і запуску (Run) її у віртуальному ПЛК (Ctrl+R) має з’явитись зображення схеми-програми мовою FBD (рисунок 12) з індикацією зеленим кольором булевих змінних, які мають стан «1», та червоним кольором – змінних, які дорівнюють «0».

Після перевірки правильності роботи схеми за таблицею істинності (таблиця 1) рекомендується зупинити ПЛК (повторним натисканням Ctrl+R) та від'єднатися від нього (Ctrl+K). Далі стає можливою візуалізація роботи схеми-програми на екрані користувача «Operator Screen».

Наступний крок виконання ЛР – у вікні **Project Browser** (один з варіантів виклику – комбінацією клавіш Alt+1) на лівому вікні (рисунок 13, а) обираємо «Operator Screen». Правою кнопкою миші викликаємо меню і вибираємо «New Screen».

У вікні, яке з’явилось, записуємо назву Scr_laba01 (рисунок 13, б). На полі цього вікна проектуємо спрощене зображення семисегментного індикатора та двох кнопок шин сигналів x0 та x1 (рисунок 8, б).

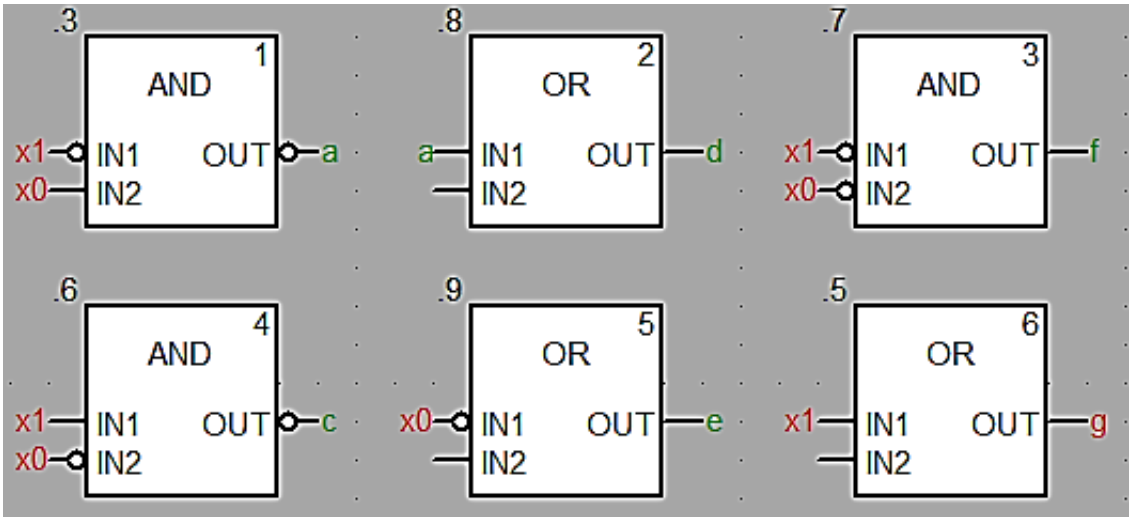
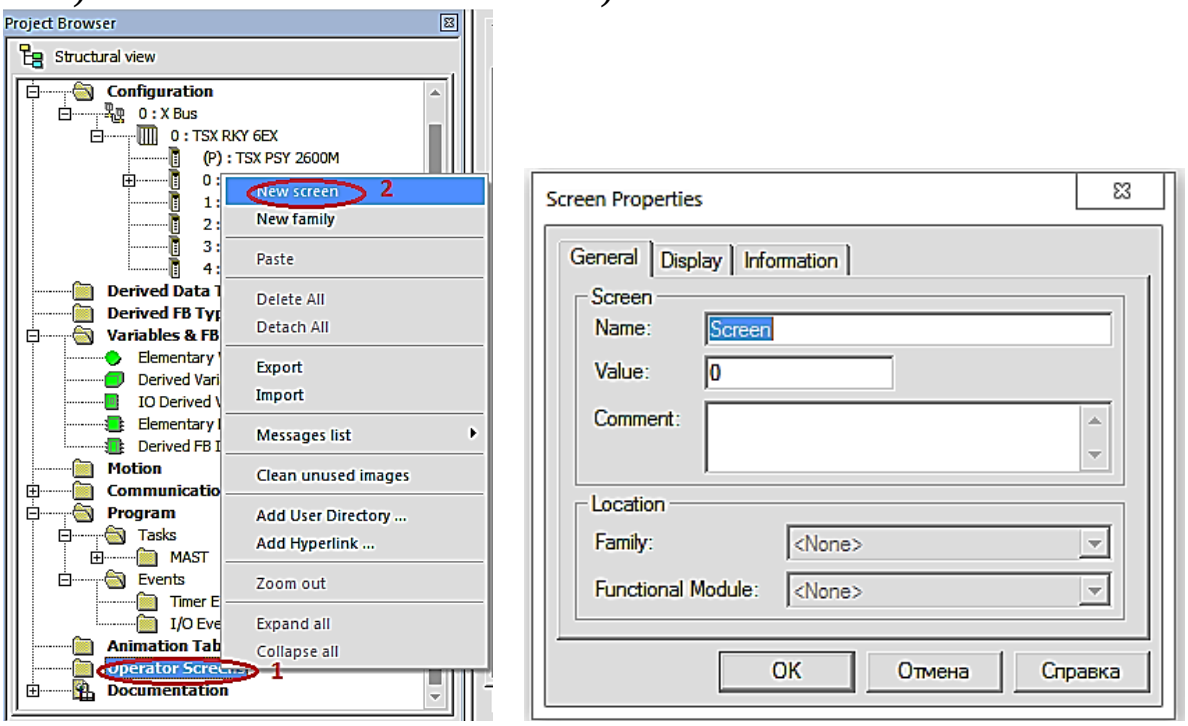


Рисунок 12 – Зображення працюючої програми мовою FBD

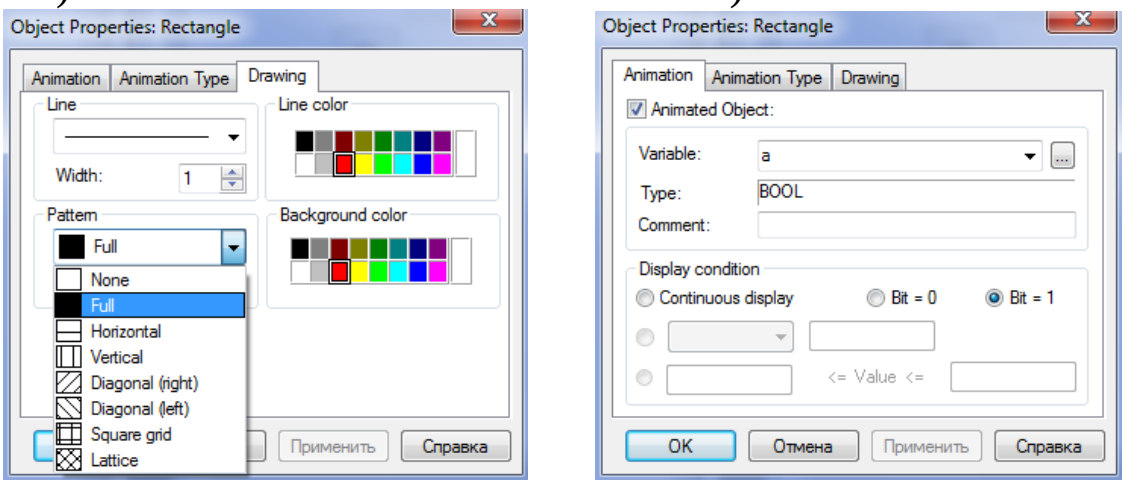
а) б)



а – створення нового елемента екранної візуалізації;
б – створення назви вікна налаштування візуалізації
Рисунок 13 – Заповнення форм

Для цього в утвореному вікні візуалізації за допомогою правої кнопки миші (ПКМ) створюємо новий об’єкт – сегмент «а» типу Rectangle – «прямокутник» і, знову натискаючи ПКМ, за допомогою опції Properties задаємо параметри, як на рисунках 14, 15.

а) б)



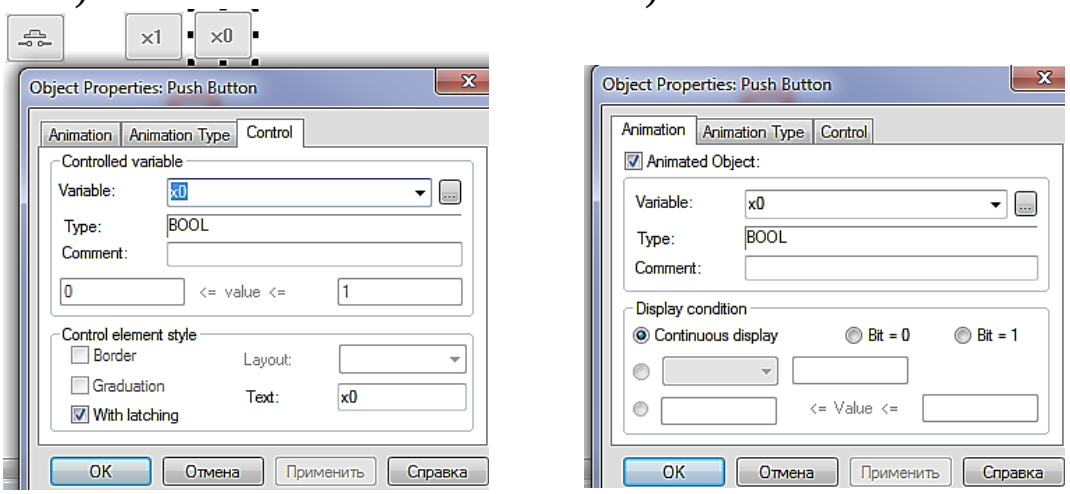
а – візуалізація; б – анімація об'єкта – сегмента «a»

Рисунок 14 – Налаштування форми зображення елемента

За аналогічною процедурою створюється решта сегментів.

Наступним кроком візуалізації є створення кнопок типу Push Button із назвами x0 та x1 для емуляції сигналів x0 та x1 (зразок – рисунок 15, а, б).

а) б)



а – візуалізація; б – анімація об'єкта – кнопка шини «x0»

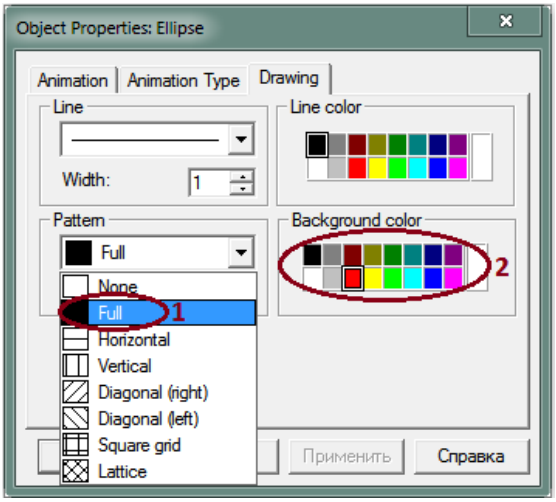
Рисунок 15 – Налаштування форми зображення елемента

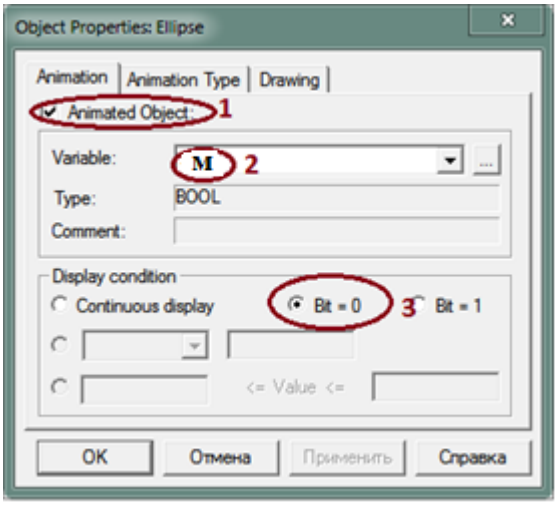
За аналогічною процедурою створюється кнопка шини «x1». Наступним кроком виконання роботи є повторне з'єднання із ПЛК (Ctrl+K), передача/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) і запуск (Run) її у віртуальному ПЛК (Ctrl+R). Вибрати

екран користувача (наприклад Scr_laba01) і після натискання F7, або включивши режим «Enable Variable Modification», іншим способом натиснувши ЛКМ на піктограмі  , моделювати комбінації шин x1x0, 00, 01, 11, 10 та перевіряти правильність зображень цифр за таблицею 1. У випадку невідповідності провести коригування схеми-програми або налаштування об’єктів екранів оператора та повторну перевірку.

На операторському екрані, у вікні візуалізації, за допомогою ПКМ, створюємо новий об’єкт «коло» і, знову натискаючи ПКМ, за допомогою опції Properties задаємо параметри, як на рисунку 16.

а) б)





а – візуалізація; б – анімація об'єкта

Рисунок 16 – Налаштування форми зображення елемента «коло»

Підсилене завдання. Зобразити у зошиті схему-програму мовою FBD шифрування сигналів від натискання кнопок a1, a2, a3 в код змінних x1, x0. Відображення на семисегментному індикаторі має здійснюватися автоматично за таблицею 1.

Відповісти на контрольні питання, виконати індивідуальне завдання.

Контрольні питання

1 Як сконфігурувати ПЛК з різними модулями в проекті?

2 Яких типів бувають модулі?

3 Як активувати модуль ETY PORT?
4 Які мови програмування підтримуються Unity Pro?
5 У якому вікні можна зображувати кола, прямокутники?
6 Як зробити візуалізацію, залежну від змінних проекту?
7 Які розширення мають файли з проектом Unity Pro?

Індивідуальне завдання

Додатково створити необхідні змінні і в окремій секції програм реалізувати функцію, відобразивши результат на тому самому операторському екрані (наприклад Scr_laba01). Приймаючи символ «∩»/∧ за кон'юнкцію («і»), символ «U»/V за диз'юнкцію, реалізувати функції за варіантами:

1) $y1 = x1 \cap x2 \cap x3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x6 \cup \bar{x}7$;
2) $y2 = \bar{x}1 \cap x2 \cap \bar{x}3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = \bar{x}6 \cup \bar{x}7$;
3) $y3 = \bar{x}1 \cap \bar{x}2 \cap \bar{x}3 \cap (x4 \cup \bar{x}5) \cup y19, y19 = \bar{x}6 \cup x7 \cap x9$;
4) $y4 = \bar{x}1 \cap x2 \cap \bar{x}3 \cap \bar{x}4 \cap x5 \cup y100, y100 = \bar{x}6 \cup x7$;
5) $y5 = x1 \cap x2 \cap x3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}7$;
6) $y6 = x1 \cap x2 \cap x9 \cap (\bar{x}4 \cup \bar{x}6) \cap y19, y19 = x8 \cup \bar{x}7$;
7) $y7 = x9 \cap x2 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}7 \cup \bar{x}3$;
8) $y8 = x9 \cap x2 \cap x3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}7 \cup \bar{x}1$;
9) $y9 = x1 \cap x9 \cap x3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}7 \cup \bar{x}2$;
10) $yA = x1 \cap x2 \cap x9 \cap (\bar{x}4 \cup \bar{x}3) \cap y19, y19 = x8 \cup \bar{x}7 \cup \bar{x}5$;
11) $yB = x1 \cap x2 \cap x3 \cap (\bar{x}9 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}9 \cup \bar{x}7$;
12) $yC = x9 \cap \bar{x}2 \cap x3 \cap (\bar{x}4 \cup \bar{x}5) \cap y19, y19 = x8 \cup \bar{x}7 \cup \bar{x}1$.

ЛАБОРАТОРНА РОБОТА 2
Синтез схем вмикання ССІ 3х8 на комбінаційних системах та ССІ 4х10 на комутаторах MUX

Мета роботи: набуття практичних навичок синтезу комбінаційних схем і розроблення схем-програм мовою FBD в Unity Pro, налагодження елементарних функцій алгебри логіки у різних формах ДДНФ або ДКНФ та налагодження елементарних функцій алгебри логіки на комутаторах (**MUX)**.

Обладнання і ПЗ: цифрова ПЕОМ із системним програмним забезпеченням Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або вище.

Хід виконання роботи

1 Вивчити весь теоретичний матеріал для досягнення мети ЛР.

2 Створити проект за наведеною після теоретичного матеріалу послідовністю та ввести схему-програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити «заготовку» до лабораторної роботи (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

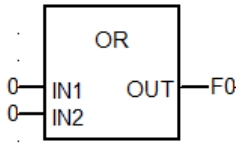
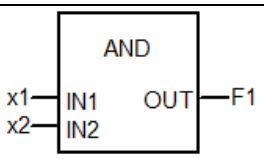
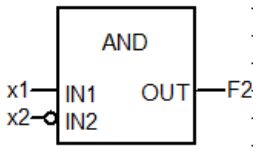
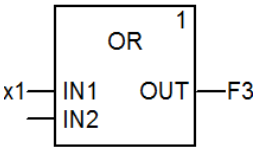
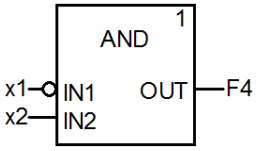
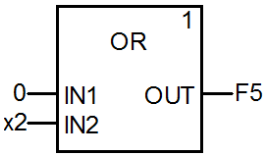
6 Оформити звіт з висновками й отримати бали за ЛР у викладача.

Теоретичний матеріал

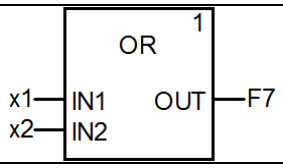
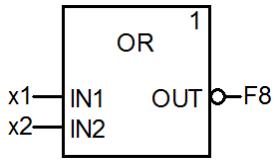
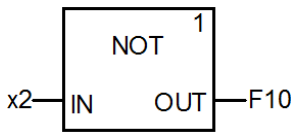
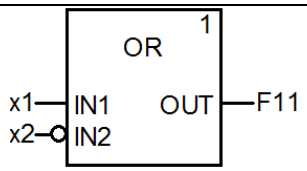
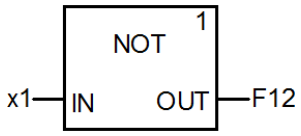
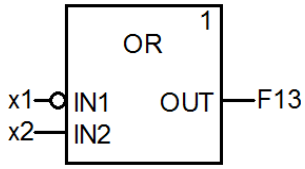
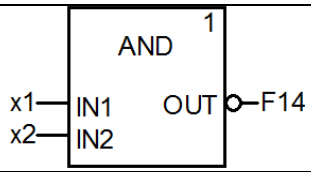
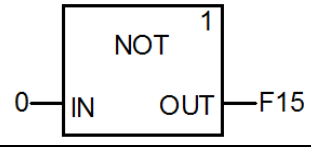
У пристроях залізничної автоматики, телемеханіки, обчислювальної техніки, в тому числі у мікропроцесорах, існує багато комбінаційних схем. Під комбінаційними схемами розуміють логічні схеми, сигнал на виході яких у кожний момент часу визначається комбінацією вхідних сигналів у той самий момент часу.

Синтез комбінаційних схем полягає у визначенні таких способів поєднання деяких найпростіших схем, названих логічними елементами (таблиця 2), при яких побудований пристрій реалізує поставлене завдання із перетворення вхідної двійкової інформації.

Таблиця 2 – Функції та еквівалентні їм FFB-типи мови FBD

Логічні аргументи і функції	Значення аргументів і функцій				Функція (з назвою у дужках) та її запис за допомогою операцій І, АБО, НІ	FFB-тип логічного елемента схеми-програми мовою FBD для ПЗ Unity Pro
Аргум. x_1	0	0	1	1		
Аргум. x_2	0	1	0	1		
1	**2**	**3**	**4**	**5**	**6**	**7**
F_0	0	0	0	0	Константа нуль (генератор нуля) $F_0 = 0$	
F_1	0	0	0	1	Кон'юнкція (кон'юнктор) $F_1 = x_1 \cdot x_2$	
F_2	0	0	1	0	Заборона 2-го аргументу (елемент заборони) $F_2 = x_1 \cdot \overline{x_2}$	
F_3	0	0	1	1	Повторення1-го аргументу (повторювач) $F_3 = x_1$	
F_4		1	0	0	Заборона 1-го аргументу (елемент заборони) $F_4 = \overline{x_1} \cdot x_2$	
F_5	0	1	0	1	Повторення 2-го аргументу (повторювач) $F_5 = x_2$	
F_6	0	1	1	0	Нерівнозначність (суматор за модулем 2) $F_6 = \overline{x_1}\, x_2 + x_1\, \overline{x_2} = x_1 \oplus x_2$	

Продовження таблиці 2

1	2	3	4	5	6	7
F_7	0	1	1	1	Диз'юнкція (диз'юнктор) $F_7 = x_1 + x_2$	
F_8	1	0	0	0	Операція Пірса (елемент Пірса) $F_8 = \overline{x_1 + x_2}$	
F_9	1	0	0	1	Рівнозначність (еквівалентор) $F_9 = x_1 x_2 + \overline{x_1}\,\overline{x_2}$	
F_{10}	1	0	1	0	Заперечення 2-го аргументу (інвертор) $F_{10} = \overline{x_2}$	
F_{11}	1	0	1	1	Імплікація від 2-го аргументу до 1-го $F_{11} = x_1 + \overline{x_2}$	
F_{12}	1	1	0	0	Заперечення 1-го аргументу (інвертор) $F_{12} = \overline{x_1}$	
F_{13}	1	1	0	1	Імплікація від 1-го аргументу до 2-го (імплікатор) $F_{13} = \overline{x_1} + x_2$	
F_{14}	1	1	1	0	Операція Шеффера (елемент Шеффера) $F_{14} = \overline{x_1 \cdot x_2}$	
F_{15}	1	1	1	1	Константа одиниця (генератор одиниці) $F_{15}=1$	

Синтез комбінаційних схем поділяють на чотири етапи:

1) утворення таблиці істинності для ФАЛ, яка описує роботу проектованої логічної схеми (найчастіше на підставі словесного опису принципу роботи);

2) утворення математичної формули для ФАЛ, що описує роботу схеми, яку синтезують, у вигляді ДДНФ або ДКНФ (на підставі таблиці істинності);

3) аналіз отриманої ФАЛ з метою побудови різних варіантів її математичного виразу (на основі законів булевої алгебри) та знаходження найкращого з них відповідно до того чи іншого критерію. На цьому етапі здійснюється мінімізація ФАЛ;

4) утворення функціональної (логічної) схеми пристрою з елементів, які складають вибраний базис. **Але у нашому випадку синтез комбінаційних схем реалізується схемою-програмою мовою FBD для ПЗ Unity Pro 3.0 або вище.** В таблиці 2 наведено функції й еквівалентні їм FFB-типи мови FBD, а також їх запис за допомогою операцій І, АБО, НІ.

Диз'юнктивна нормальна форма (ДНФ) у булевій логіці — нормальна форма, в якій булева формула має вигляд диз'юнкції декількох кон'юнктів (де кон'юнктами називаються кон'юнкції декількох пропозиційних символів або їх заперечень).

Досконалою диз'юнктивною нормальною формою (ДДНФ) булевої функції називається диз'юнкція тих конституент одиниці, які перетворюються в одиницю на тих самих наборах змінних, що й задана функція. ДДНФ має задовольняти такі умови:

- у ній немає однакових доданків;
- жоден із доданків не містить двох однакових співмножників;
- жоден із доданків не містить змінну разом із її запереченням;
- у кожному окремому доданку є як співмножник або змінна x_i, або її заперечення для будь-якого i = 1, 2, …, n.

У практиці перетворювання логічних формул існує чіткий порядок виконання дій. У разі відсутності у виразі дужок першими мають виконуватися операції інверсії (заперечення), потім — операції кон’юнкції (логічного множення), останніми — операції диз’юнкції (логічного складання).

Наявність у виразі дужок змінює порядок дій, і тоді в першу чергу виконуються операції у дужках.

Кількість різних наборів значень аргументів ФАЛ кінцева, виходячи з цього будь-яка ФАЛ може бути задана таблицею з 2^N рядками. Зліва у таблиці (таблиця 3) проставляються номери

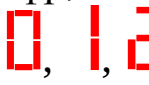
рядків (номери цифр, що відображатимуться), потім – значення функції семи сегментів для кожного з наборів змінних, а справа — набори значень аргументів функції. Такий спосіб задання функції для семи сегментів названо табличним.

Синтез логічних пристроїв у різних базисах розглянемо на прикладі побудови схем увімкнення сегментів семисегментного індикатора з трьома входами x2, x1, x0.

Послідовність створення проекту

Для виконання ЛР 2 необхідно відкрити створений проект ЛР 1 та зберегти його як laba02 або самостійно створити проект за процедурою, наведеною у ЛР 1 із назвою laba02.

Послідовність дій введення типового завдання проекту

Типове завдання ЛР 2. Розробити схему-програму мовою FBD кодування стану восьми кнопок (булеві змінні a0, a1, a2, a3, a4, a5, a6, a7) у дворозрядний код (булеві змінні x2, x1, x0) та візуалізувати коди x2x1x0 (000, 001, 010, 011, 100, 101, 110, 111) на стандартному семисегментному індикаторі із сегментами (a, b, c, d, e, f, g) у вигляді цифр, зображення яких пропонується виводити в такому вигляді:  , відповідно до значень, показаних у таблиці 3.

Для синтезу схем, що реалізовують різні ФАЛ, найбільш зручним є аналітичний спосіб задання ФАЛ, коли ФАЛ має вигляд алгебраїчного виразу, який отримують у результаті використання яких-небудь логічних операцій до змінних функції алгебри логіки. Для знаходження ДДНФ з таблиці істинності 3 вибирають тільки ті рядки, де стоять набори змінних, які перетворюють функцію в 1. Це 0-й, 2-й, 3-й, 5-й, 6-й і 7-й рядки. Виписують кон'юнкції, які відповідають вибраним рядкам. При цьому, якщо аргумент x_i входить до даного набору як 1, він записується у кон'юнкцію без змін; а якщо x_i входить до даного набору як 0, то у відповідну кон'юнкцію записується його заперечення. Наприклад, для сегмента «a» в ДДНФ він буде мати вигляд такого алгебраїчного виразу: $a_{ДДНФ}= \overline{x2}\,\overline{x1}\,\overline{x0} + \overline{x2}\,x1\,\overline{x0} + \overline{x2}\,x1x0 + x2\,\overline{x1}\,x0 + +x2\,x1\,\overline{x0} + x2x1x0.$

Цю саму функцію, для сегмента «a», можна записати у вигляді ДКНФ. Для цього із таблиці істинності вибирають набори аргументів, на яких значення функції дорівнює 0. Виписують диз'юнкції, які відповідають названим наборам аргументів. При

цьому, якщо аргумент x_i входить у даний набір як 0, він вписується у диз'юнкцію, яка відповідає набору, без змін; а якщо x_i входить у набір як 1, то у відповідну диз'юнкцію вписують його заперечення. ДКНФ буде мати вигляд такого алгебраїчного виразу: $a_{ДКНФ}= (\overline{x2} + x1 + x0) \wedge (x2 + x1 + \overline{x0})$.

Таблиця 3 – Задання ФАЛ за допомогою таблиці істинності для сегментів ССІ 3х8

Номер рядка	Значення функцій ССІ 3х8							Значення аргументів			Примітка
	a	b	c	d	e	f	g	x_2	x_1	x_0	
0	1	1	1	1	1	1	0	0	0	0	
1	0	1	1	0	0	0	0	0	0	1	
2	1	1	0	1	1	0	1	0	1	0	
3	1	1	1	1	0	0	1	0	1	1	
4	0	1	1	0	0	1	1	1	0	0	
5	1	0	1	1	0	1	1	1	0	1	
6	1	0	1	1	1	1	1	1	1	0	
7	1	1	1	0	0	0	0	1	1	1	

Очевидно, що для реалізації ***а*** ($a_{ДКНФ}$) раціональніше вибрати схему в програмі, яка буде містити тільки три блоки з трьома входами (аналог F11, F13) та з двома входами F14 з таблиці 2.

За аналогічною процедурою студентам необхідно скласти ДДНФ та ДКНФ для решти сегментів таблиці 3 і вибрати, яка функція більш раціональна для програмування. Збільшення кількості входів у блоків OR, AND та інших реалізується шляхом «розтягування» блока ПКМ за нижню межу блока.

За відомою з ЛР 1 процедурою створюються додаткові змінні та кнопки шини «х2», «х1», «х0». Наступним кроком виконання роботи є повторне з'єднання з ПЛК (Ctrl+K), передача/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) і запуску (Run) її у віртуальному ПЛК (Ctrl+R). Вибрати екран користувача (наприклад Scr_laba02) і після натискання F7, або включивши режим «Enable Variable Modification» іншим способом – натиснувши ЛКМ на піктограмі  , моделювати комбінації шин х2х1х0, 001, 011, 010, 000, 100, 101, 111, 110 та перевіряти

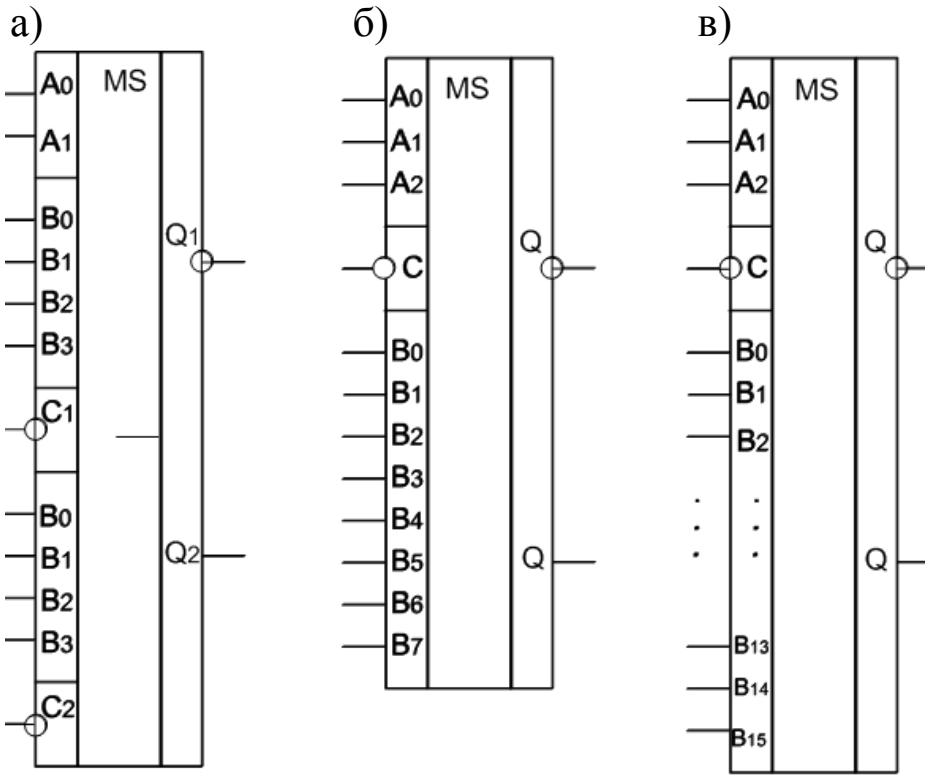
правильність зображень цифр згідно з таблицею 3. У випадку невідповідності провести коригування схеми-програми або налаштування об’єктів екранів оператора та провести повторну перевірку.

У випадку реалізації схем засобами MUX слід оновити склад різних серій мікросхем, що застосовуються у пристроях залізничної автоматики та телемеханіки, як елементів середнього ступеня інтеграції – комутатори (мультиплексори – MUX). Мультиплексор – це схема з одним виходом Q, k+1 керуючими входами $A_0, A_1, \ldots, A_к$, n+1 інформаційними входами $B_0, B_1, \ldots, B_n$ та входом дозволу роботи мультиплексора C для подачі синхронізуючого сигналу. Якщо подавати на керуючі входи відповідні сигнали у вигляді двійкового коду, до виходу комутатора підключається один із його інформаційних входів. Існують комутатори, які здійснюють вибір одного з чотирьох, восьми або шістнадцяти інформаційних сигналів. Умовне позначення таких комутаторів наведено на рисунку 17. При подачі сигналу на вхід C вихідна змінна мультиплексора Q повторює змінну інформаційного входу B_n з номером $n=2^{k+1}$, що задається двійковим кодом на керуючих входах. За відсутності синхронізуючого сигналу (C=0, якщо вхід прямий) зв'язок між інформаційними входами і виходом відсутній.

Функціонування мультиплексора визначається за таблицею 4.

Таблиця 4 – Функціонування мультиплексора

Керуючі входи		Вхід дозволу	Вихід
A_1	A_0	C	Q
~	~	0	0
0	0	1	B_0
0	1	1	B_1
1	0	1	B_2
1	1	1	B_3



а – К155КП2; б – К155КП7; в – К155КП1
Рисунок 17 – Типи комутаторів серії К155

Рівняння комутаторів 1 з 4, 1 з 8, 1 з 16 можна записати таким чином:

$$F_{1\text{-}4} = \overline{A_1}\,\overline{A_0}B_0 + \overline{A_1}\,A_0B_1 + A_1\overline{A_0}B_2 + A_1A_0B_3; \qquad (1)$$

$$\begin{aligned} F_{1\text{-}8} = {} & \overline{A_2}\,\overline{A_1}\,\overline{A_0}B_0 + \overline{A_2}\,\overline{A_1}\,A_0B_1 + \overline{A_2}\,A_1\,\overline{A_0}B_2 + \\ & + \overline{A_2}\,A_1\,A_0B_3 + A_2\,\overline{A_1}\,\overline{A_0}B_4 + A_2\,\overline{A_1}\,A_0B_5 + \\ & + A_2\,A_1\,\overline{A_0}B_6 + A_2\,A_1\,A_0B_7; \end{aligned} \qquad (2)$$

$$\begin{aligned} F_{1\text{-}16} = {} & \overline{A_3}\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}B_0 + \overline{A_3}\,\overline{A_2}\,\overline{A_1}\,A_0B_1 + \overline{A_3}\,\overline{A_2}\,A_1\,\overline{A_0}B_2 + \\ & + \overline{A_3}\,\overline{A_2}\,A_1\,A_0B_3 + \overline{A_3}\,A_2\,\overline{A_1}\,\overline{A_0}B_4 + \overline{A_3}\,A_2\,\overline{A_1}\,A_0B_5 + \\ & + \overline{A_3}\,A_2\,A_1\,\overline{A_0}B_6 + \overline{A_3}\,A_2\,A_1\,A_0B_7 + A_3\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}B_8 + \\ & + A_3\,\overline{A_2}\,\overline{A_1}\,A_0B_9 + A_3\,\overline{A_2}\,A_1\,\overline{A_0}B_{10} + A_3\,\overline{A_2}\,A_1\,A_0B_{11} + \\ & + A_3\,A_2\,\overline{A_1}\,\overline{A_0}B_{12} + A_3\,A_2\,\overline{A_1}\,A_0B_{13} + A_3\,A_2\,A_1\,\overline{A_0}B_{14} + \\ & + A_3\,A_2\,A_1\,A_0B_{15}. \end{aligned} \qquad (3)$$

У виразах (1) – (3) через B_i й A_j позначені відповідно сигнали, які подаються на інформаційні та керуючі входи мультиплексорів.

Нехай ми маємо довільні ФАЛ трьох, чотирьох і п'яти змінних, тоді

$$\begin{aligned} &F_3(x_3, x_2, x_1);\\ &F_4(x_4, x_3, x_2, x_1);\\ &F_5(x_5, x_4, x_3, x_2, x_1). \end{aligned} \qquad (4)$$

Позначимо x_5, x_4, x_3, x_2 відповідно через A_3, A_2, A_1, A_0, а x_1 – через B.

$$\begin{aligned} &F_3(A_1, A_0, B);\\ &F_4(A_2, A_1, A_0, B);\\ &F_5(A_3, A_2, A_1, A_0, B). \end{aligned} \qquad (5)$$

Використовуючи метод функціональної декомпозиції, подамо функції (5) у вигляді таких розкладів:

$$\begin{aligned} F_3(A_1, A_0, B) = {} & F_3(0,0,B)\overline{A_1}\,\overline{A_0} + F_3(0,1,B)\overline{A_1}\,A_0 + \\ & + F_3(1,0,B)A_1\overline{A_0} + F_3(1,1,B)A_1A_0; \end{aligned} \qquad (6)$$

$$\begin{aligned} F_4(A_2, A_1, A_0, \hat{A}) = {} & F_4(0,0,0,\hat{A})\overline{A_2}\,\overline{A_1}\,\overline{A_0} + F_4(0,0,1,\hat{A}) \wedge \\ & \wedge\overline{A_2}\,\overline{A_1}\,A_0 + F_4(0,1,0,\hat{A})\overline{A_2}\,A_1\overline{A_0} + F_4(0,1,1,\hat{A})\overline{A_2}\,A_1A_0 + \\ & + F_4(1,0,0,\hat{A})A_2\,\overline{A_1}\,\overline{A_0} + F_4(1,0,1,\hat{A})A_2\,\overline{A_1}\,A_0 + F_4(1,1,0,\hat{A}) \wedge \\ & \wedge A_2\,A_1\overline{A_0} + F_4(1,1,1,\hat{A})A_2\,A_1A_0; \end{aligned} \qquad (7)$$

$$\begin{aligned} F_5(A_3, A_2, A_1, A_0, B) = {} & F_5(0,0,0,0,B)\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,\overline{A_0} + \\ & + F_5(0,0,0,1,B)\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,A_0 + F_5(0,0,1,0,B)\overline{A_3}\,\overline{A_2}\,A_1\overline{A_0} + \\ & + F_5(0,0,1,1,B)\overline{A_3}\,\overline{A_2}\,A_1A_0 + F_5(0,1,0,0,B)\overline{A_3}\,A_2\,\overline{A_1}\,\overline{A_0} + \end{aligned}$$

$$
\begin{aligned}
&+F_5(0,1,0,1,B)\overline{A_3}\,A_2\,\overline{A_1}\,A_0+F_5(0,1,1,0,B)\overline{A_3}\,A_2\,A_1\overline{A_0}+\\
&+F_5(0,1,1,1,B)\overline{A_3}\,A_2\,A_1A_0+F_5(1,0,0,0,B)A_3\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}+\\
&+F_5(1,0,0,1,B)A_3\,\overline{A_2}\,\overline{A_1}\,A_0+F_5(1,0,1,0,B)A_3\,\overline{A_2}\,A_1\overline{A_0}+\\
&+F_5(1,0,1,1,B)A_3\,\overline{A_2}\,A_1A_0+F_5(1,1,0,0,B)A_3\,A_2\,\overline{A_1}\,\overline{A_0}+\\
&+F_5(1,1,0,1,B)A_3\,A_2\,\overline{A_1}\,A_0+F_5(1,1,1,0,B)A_3\,A_2\,A_1\overline{A_0}+\\
&+F_5(1,1,1,1,B)A_3\,A_2\,A_1A_0.
\end{aligned}
\qquad (8)
$$

Розглядаючи вирази (6) – (8), легко переконатися у тому, що всі значення функцій F_3, F_4, F_5, які містяться у правій частині цих виразів, можуть набувати значень тільки з множини {0, 1, B }.

Вирази (6) – (8) описують будь-яку ФАЛ відповідно трьох, чотирьох і п’яти змінних.

Попарне зіставлення виразів (1) і (6), (2) і (7), (3) і (8) показує, що коли на керуючі входи комутаторів подавати змінні A_1 A_0; A_2 A_1 A_0; A_3 A_2 A_1 A_0, а на відповідні інформаційні входи — величини з множини {0, 1, B або $\overline{B}$} за таблицею істинності заданої функції, то можна записати такі рівності:

$$F_{1\text{-}4} = F_3; \quad F_{1\text{-}8} = F_4; \quad F_{1\text{-}16} = F_5. \qquad (9)$$

Ці вирази свідчать про те, що комутатори 1 з 4, 1 з 8, 1 з 16 є універсальними логічними елементами, що реалізують будь-яку ФАЛ відповідно трьох, чотирьох і п’яти змінних [23].

Але у нашому випадку синтез комбінаційних схем реалізується схемою-програмою мовою FBD для ПЗ Unity Pro 3.0 або вище з широким різноманіттям функцій, процедур і функціональних блоків (FFB). У **ПЗ Unity Pro** є бібліотечні програмні блоки (EF-модуль/блок) FFB типу, які називаються MUX і моделюють роботу мультиплексора 1 з N (де N – від 2 до 16) з вибором (комутацією) каналу через змінну типу INT. Таким чином, для перетворення керуючих булевих (BOOL) сигналів A_3 A_2 A_1 A_0 (x3, x2, x1, x0) у тип INT для входу K (керуючого) EF-блока MUX (комутатора) необхідно перекодувати змінні. Оскільки в бібліотеці **Unity Pro** немає EF-блока BIT_TO_INT, то його потрібно або створити самостійно, або замінити двома

послідовно з'єднаними BIT_TO_WORD і WORD_TO_INT. На рисунку 18 зображено вигляд схеми/програми функції комутатора.

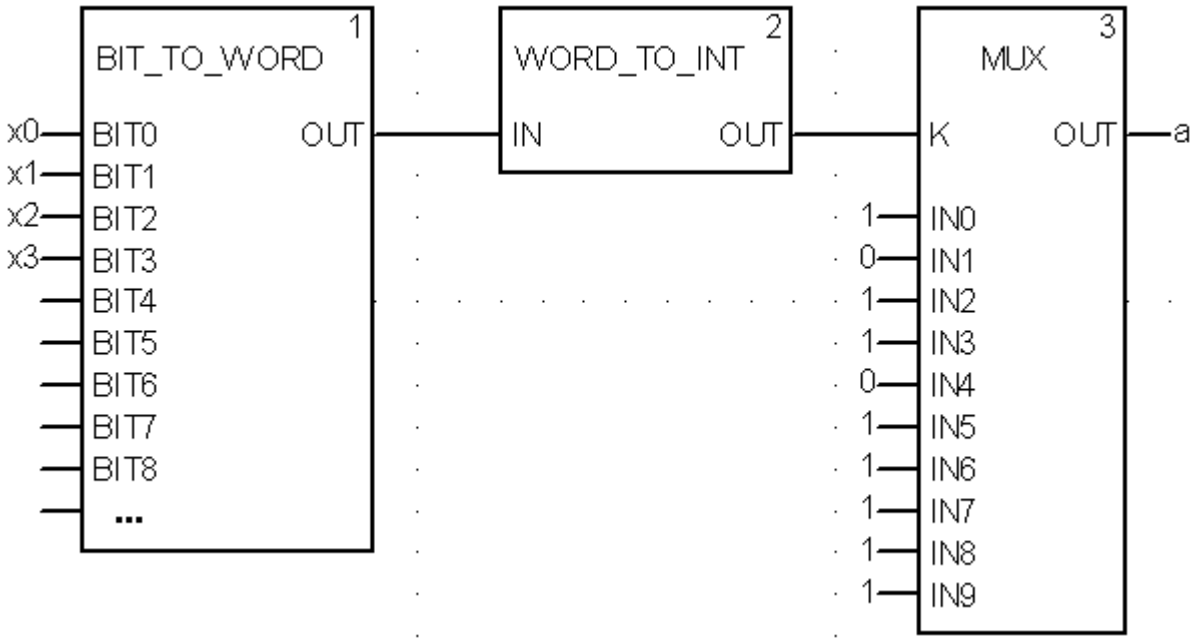


Рисунок 18 – Схема комутатора 1 із 10, реалізована для увімкнення сегмента «a» індикатора

Синтез логічних пристроїв на базі мультиплексора (MUX) розглянемо на прикладі побудови схем увімкнення десяти цифр сегментами семисегментного індикатора з чотирма входами x3, x2, x1, x0.

Типове завдання другої частини ЛР 2. Розробити схему-програму мовою FBD кодування стану десяти кнопок (булеві змінні k0, k1, k2, k3, k4, k5, k6, k7, k8, k9) у дворозрядний код (булеві змінні x3, x2, x1, x0 або A3, A2, A1, A0) та візуалізувати коди x3x2x1x0 (0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001) на стандартному семисегментному індикаторі із сегментами (a, b, c, d, e, f, g) у вигляді цифр, зображення яких пропонується виводити в такому вигляді: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, відповідно до значень, показаних у таблиці істинності за аналогією до таблиці 3.

Синтез схем-програм ФАЛ на базі комутаторів, при якому ФАЛ має вигляд стандартної схеми-програми, як на рисунку 18, є найбільш зручним. Однак є невелика відмінність. Входи IN0...IN9, при прямому кодуванні (до 16 входів), фактично є табличними

значеннями булевої функції виходу (вихід OUT). Тому для побудови схеми ФАЛ необхідно скопіювати необхідну кількість (сім – від кількості сегментів) EF-блоків типу MUX для кожного із сегментів (a, b, c, d, e, f, g). Виходи (OUT) семи EF-блоків типу MUX приєднати до булевих змінних (a, b, c, d, e, f, g), а на входи IN0...IN9 подати 0 або 1 у суворій відповідності із таблицею істинності. Набір змінних x3x2x1x0 засобами двох послідовно з'єднаних блоків BIT_TO_WORD і WORD_TO_INT перетворюють у змінну Adr формату INT. Саме змінну Adr формату INT необхідно подати на вхід "K" семи EF-блоків типу MUX.

За відомою з ЛР 1 процедурою створюються додаткові змінні та кнопки шини «x3», «x2», «x1», «x0». Наступним кроком виконання роботи є повторне з'єднання з ПЛК (Ctrl+K), передачі/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) і запуску (Run) її у віртуальному ПЛК (Ctrl+R). Вибрати екран користувача (наприклад Scr_laba02б) і після натискання F7, або включивши режим «Enable Variable Modification» іншим способом – натиснувши ЛКМ на піктограмі  , моделювати комбінації шин x3x2x1x0, 0001, 0011, 0010, 0000, 0100, 0101, 0111, 0110, 1000, 1001 та перевіряти правильність зображень цифр за таблицею істинності (створеною самостійно). У випадку невідповідності провести коригування схеми-програми або налаштування об’єктів екранів оператора та провести повторну перевірку.

Контрольні питання

1 Що становить собою мова функціональних блоків FBD?

2 Для чого служать зв'язки при розробленні програм на FBD?

3 Яке призначення входів і виходів функціональних блоків?

4 Як розташовуються функціональні блоки в редакторі програми FBD?

5 Визначення булевої функції.

6 Поясніть термін ДНФ.

7 Поясніть термін КНФ.

8 Поясніть термін ДДНФ.

9 Поясніть термін ДКНФ.

10 Як реалізувати з'єднання з ПЛК?

11 Як реалізувати передачі/завантаження (Load) схеми-програми до ПЛК?

12 Як реалізувати запуск (Run) програми у віртуальному ПЛК?

13 Як скласти таблицю істинності для CCI 4x10, 4x16(F)?

14 Як скласти таблицю істинності для CCI 5x32, 6x46?

15 Яка потенційна межа відображення символів на CCI?

Індивідуальне завдання

Скласти таблицю істинності, зробити для CCI 4x16(F) аналітичний запис і схему-програму за варіантом:

1) сегмента «a» у досконалій ДНФ;
2) сегмента «a» у досконалій КНФ;
3) сегмента «b» у досконалій ДНФ;
4) сегмента «b» у досконалій КНФ;
5) сегмента «c» у досконалій ДНФ;
6) сегмента «c» у досконалій КНФ;
7) сегмента «d» у досконалій ДНФ;
8) сегмента «d» у досконалій КНФ;
9) сегмента «e» у досконалій ДНФ;
10) сегмента «e» у досконалій КНФ;
11) сегмента «f» у досконалій ДНФ;
12) сегмента «f» у досконалій КНФ.

ЛАБОРАТОРНА РОБОТА 3
Створення функціональних блоків користувача (DFB) для синтезу модуля вмикання CCI 4x16(F)min та індикації багаторозрядності

Мета роботи: набуття практичних навичок створення функціональних блоків користувача (DFB) та синтезу комбінаційних схем мінімізованих засобами карт Карно й розроблення схем-програм мовою FBD з використанням DFB блоків в Unity Pro та налагодження мінімізованих ФАЛ.

Обладнання та ПЗ: цифрова ПЕОМ із системним програмним забезпеченням Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або більш пізнішої.

Хід виконання роботи

1 Вивчити весь теоретичний матеріал, для досягнення мети ЛР.

2 Створити проект за планом, викладеним нижче, та ввести схему-програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити підготовку до ЛР (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

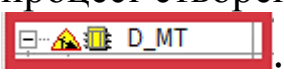
5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

6 Оформити звіт з висновками й отримати бали за ЛР у викладача.

Послідовність створення проекту

Для виконання ЛР 3 необхідно відкрити створений проект ЛР 1 і зберегти його як laba03 або самостійно створити проект за процедурою, наведеною в ЛР 1, з назвою laba03.

Послідовність дій введення типового завдання проекту

Як і в попередній роботі, компонуємо всі необхідні апаратні модулі контролера та активуємо й налагоджуємо їх згідно зі схемою підключення. Наступний крок – відкриваємо вікно Derived FB Types, щоб створити новий DFB-тип (рисунок 19), на основі якого потім будуть створюватися екземпляри. Крім унікальної назви типу (за правилами створення ідентифікаторів, латинські літери без пробілів), необхідно створити локальні вхідні та вихідні змінні (див. рисунок 19) у DFB-типі та логіку «перетворення» вхідних змінних у вихідні, у випадку імітації бітового меандру з періодом $T_{імп}$ = 1 с (D_MT-меандр часу, у якого $t_{імп}$= 0,5 с, $t_{пауз}$= 0,5 с) рисунок 20. Мовою FBD створюємо схему (рисунок 20), натискаючи на об’єкт Sec_MT 2 рази ПКМ. У процесі створення цих елементів буде відображатися такий знак: 

. Це означає, що елемент у процесі обробки. Після завершення створення треба скомпілювати програму, щоб не було помилок, за допомогою символа  .

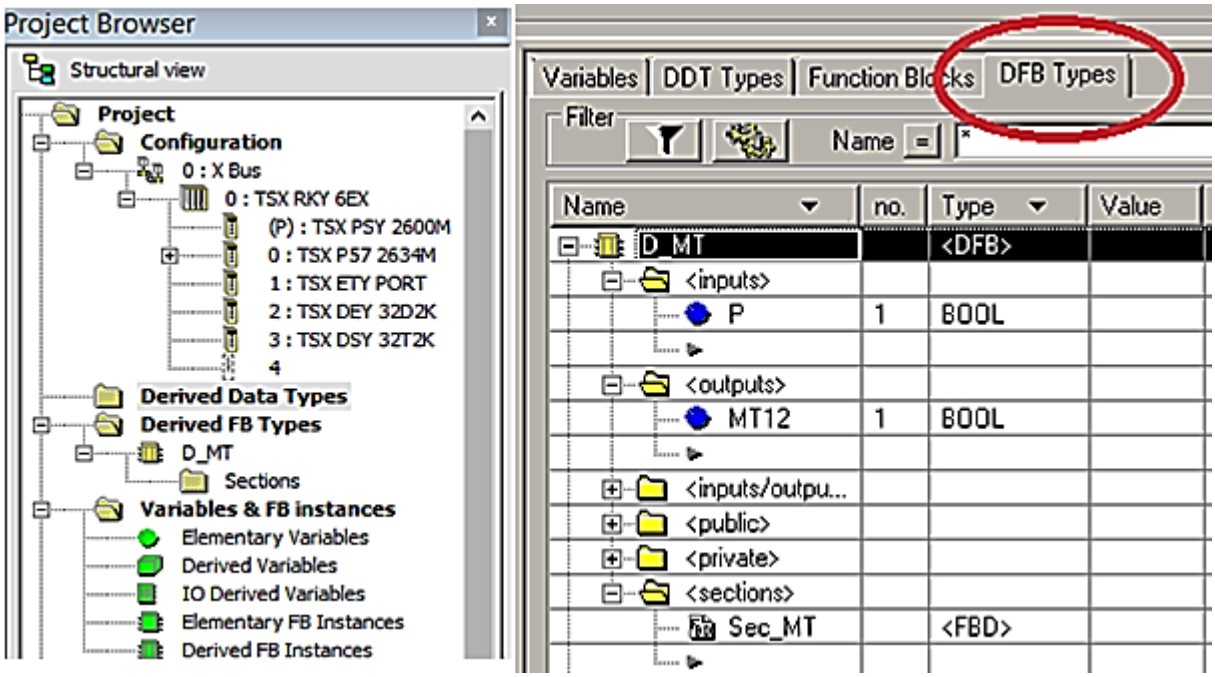


Рисунок 19 – Створення в розділі проекту Derived FB Types DFB-типу з назвою «D_MT»

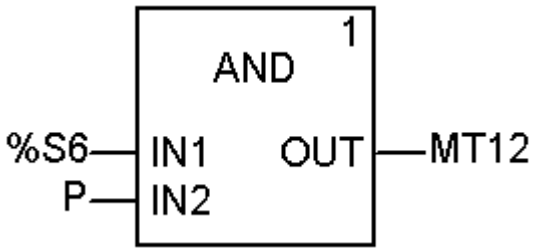


Рисунок 20 – Внутрішня схема (логіка) секції Sec_MT DFB-блока створюваного типу D_MT

У MAST секції (laba03) виключаємо меню вибору (створення) блока Ctrl+I (рисунок 21) та, ввівши в поле «FFB» значення «D_MT», а в поле «Instance/екземпляр» назву екземпляра блока типу «D_MT» (можливо MT, або MT1 за замовчуванням D_MT_0, D_MT_1...), створюємо екземпляр функціонального блока користувача. Якщо на вхід «P» такого блока подати «сигнал 1», то на виході MT12 буде відтворюватися імпульсний сигнал – меандр.

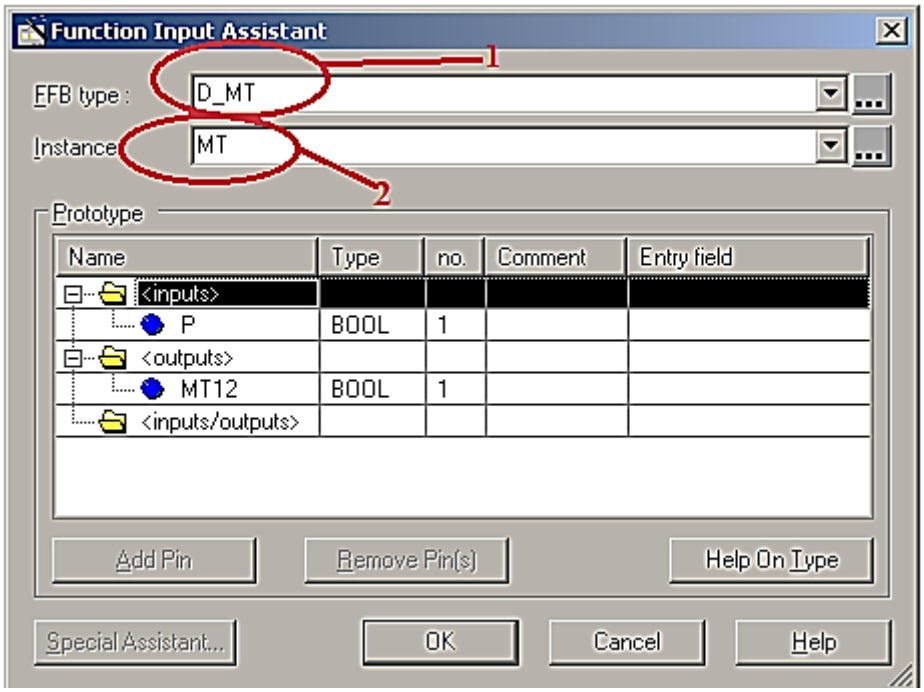


Рисунок 21 – Створення екземпляра з назвою «MT» функціонального блока користувача DFB-типу з ім'ям «D_MT» зі сформованими раніше вхідними і вихідними змінними та програмою перетворення змінних

Відкриваємо/створюємо операторський екран Screen_laba03 і зображуємо об'єкт типу «еліпс», як на рисунку 22, і використаємо «кружечок» для відображення восьмого «dp» сегмента модифікованого CCI, який буде залежати від вихідної змінної типу BOOL з назвою «MT12» блока з ім'ям «MT» DFB типу. Така змінна матиме параметр запису MT.MT12 з можливістю задавати її в полі Variable (рисунок 22).

Можемо використовувати вихідну змінну MT12 блока MT як елемент складної змінної MT.MT12 в параметрах анімації «еліпс» – восьмого сегмента CCI або інших секціях програм.

Послідовність дій розроблення залікового завдання лабораторної роботи

Після успішного виконання вищенаведеного типового завдання, для розуміння створення функціональних блоків користувача (DFB-тип), на прикладі створення генератора меандрових імпульсів (з ім'ям MT), з періодом одна секунда необхідно: синтезувати модуль управління семисегментним індикатором (DFB-тип з ім'ям I4xF), який буде перетворювати

двійковий чотирирозрядний код (вхідні булеві змінні x3, x2, x1, x0) на стандартному семисегментному індикаторі (рисунок 8, б) з виходами a, b, c, d, e, f, g.

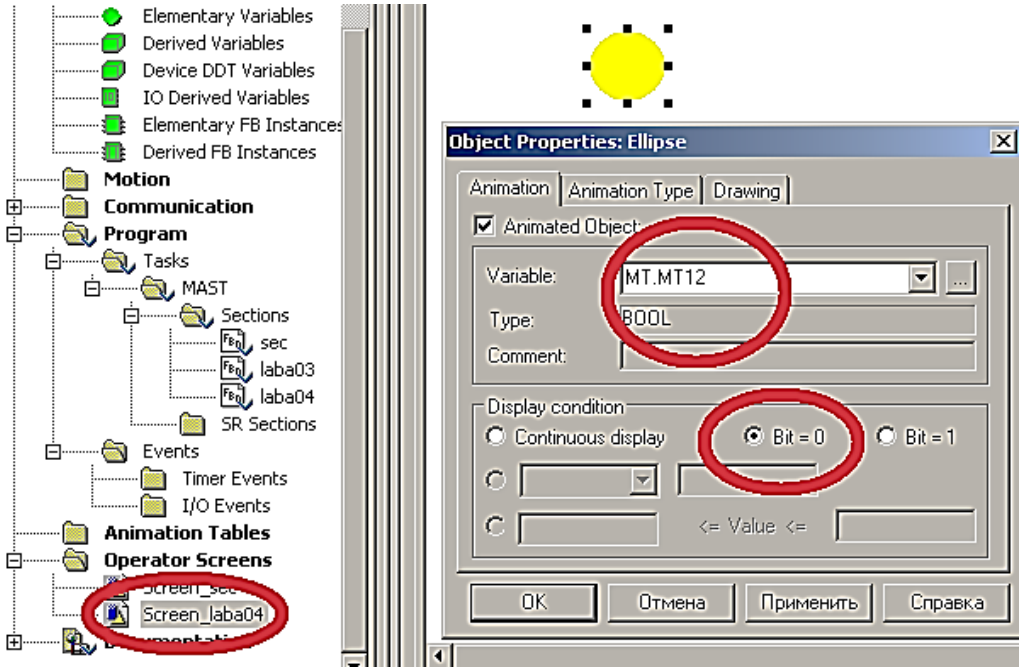


Рисунок 22 – Зображення та налаштування восьмого сегмента «dp» модернізованого ССІ

Використавши створений функціональний блок користувача (DFB-тип з ім'ям I4xF), створити його екземпляр з ім'ям D0 (Digital0) (рисунок 23). Підключити до вхідних булевих змінних x3, x2, x1, x0 блока D0 типу I4xF кодові булеві сигнали, що генерують таку послідовність: 0001, 0011, 0010, 0000, 0100, 0101, 0111, 0110, 1000, 1001, 1011, 1010, 1110, 1111, 1101, 1100.

За відомою з ЛР 1 процедурою створюються додаткові змінні та кнопки шини «x3», «x2», «x1», «x0». Наступним кроком виконання роботи є повторне з'єднання з ПЛК (Ctrl+K) та передача/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) й запуск (Run) її у віртуальному ПЛК (Ctrl+R). Вибрати екран користувача (наприклад Scr_laba03) і після натискання F7, або включивши режим «Enable Variable Modification» іншим способом, натиснувши ЛКМ на піктограмі  , моделювати комбінації шин x3x2x1x0, 0001, 0011, 0010, 0000, 0100, 0101, 0111, 0110, 1000, 1001, 1011, 1010, 1110, 1111, 1101, 1100 та перевіряти

правильність зображень цифр згідно з таблицею істинності (створеною самостійно). У випадку невідповідності провести коригування схеми-програми або налаштування об’єктів екранів оператора та повторну перевірку.

У результаті тестування мають з'явитися всі 16 цифр, зображення яких має виводитися в такому вигляді: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, b, C, d, E, F, відповідно до значень таблиці істинності, раніше самостійно сформованої (таблиця 5). Для отримання доброї оцінки за виконання лабораторної роботи ФАЛ усіх сегментів мають бути оптимізовані.

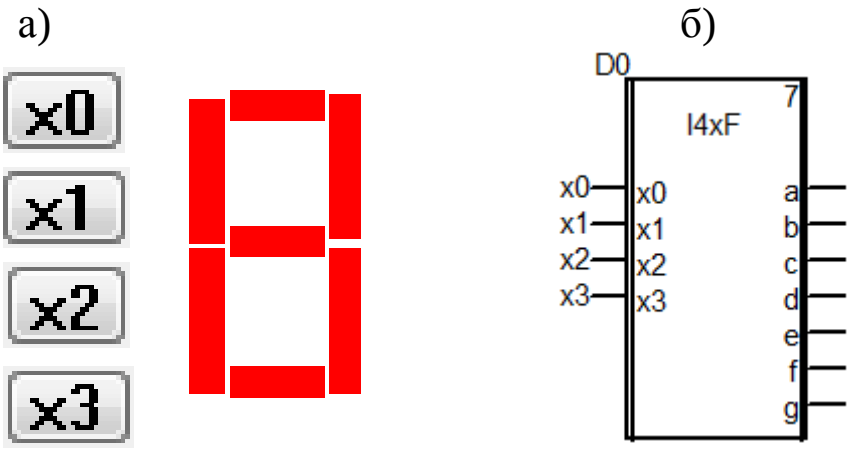


Рисунок 23 – Зображення фрагмента операторського екрана (а) та фрагмент секції схеми-програми завдання (б)

Для перевірки роботи розробленого модуля можливе застосування системної змінної типу слово (WORD), %SW50, яка містить інформацію про секунди реального часу ПЛК у форматі #16SS00 та блок WORD_TO_BIT. Її використання дає змогу самостійно побудувати багаторозрядну систему індикації.

Схему перевірки зображено на рисунку 24.

Контрольні питання

1 Послідовність створення функціонального блока користувача (DFB).

2 Як створити новий DFB-тип?

3 Як створити новий екземпляр блока.

4 Як мінімізувати ФАЛ?

5 Як раціонально виділити частину ФАЛ у блок користувача?

6 Поясніть термін WORD.

7 Поясніть термін INT.

8 Як реалізувати з'єднання з ПЛК?

9 Як реалізувати передачу/завантаження (Load) схеми-програми до ПЛК?

10 Як скласти таблицю істинності для CCI 4x10, 4x16(F)?

11 Як скласти таблицю істинності для CCI 5x32, 6x46?

12 Якою є потенційна межа відображення символів на CCI?

Таблиця 5 – Форми подання чисел для перевірки

Форми подання чисел		
десяткова	двійкова	шістнадцяткова
1	**2**	**3**
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

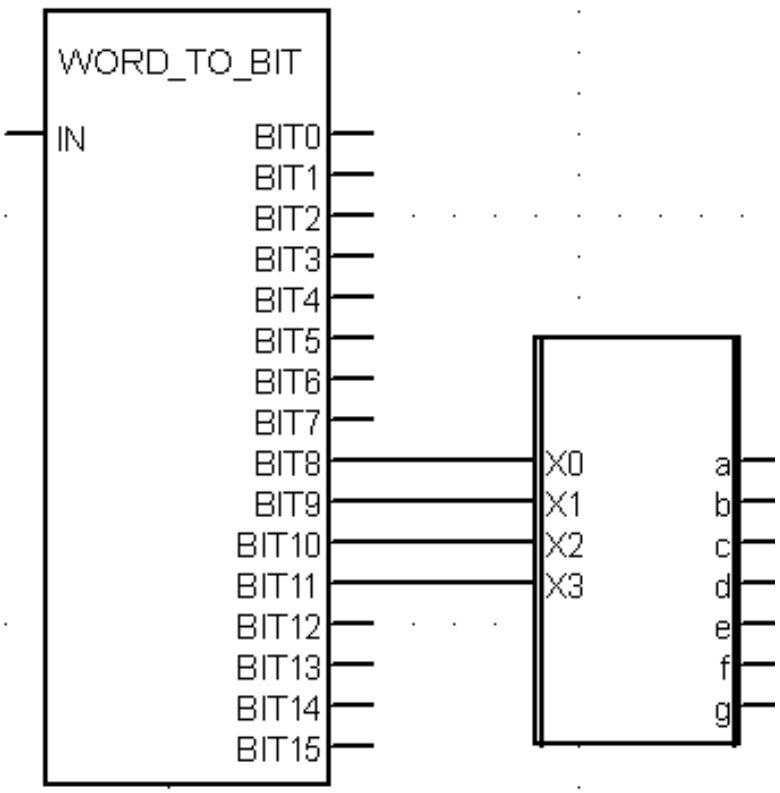


Рисунок 24 – Зображення фрагмента схеми-програми перетворення системних секунд ПЛК у сигнали ССІ

Індивідуальне завдання

Скласти таблицю істинності та зробити для ССІ 5х32 аналітичний запис та схему-програму для реалізації ФАЛ:

1) сегмента «a» у досконалій ДНФ;
2) сегмента «a» у досконалій КНФ;
3) сегмента «b» у досконалій ДНФ;
4) сегмента «b» у досконалій КНФ;
5) сегмента «c» у досконалій ДНФ;
6) сегмента «c» у досконалій КНФ;
7) сегмента «d» у досконалій ДНФ;
8) сегмента «d» у досконалій КНФ;
9) сегмента «e» у досконалій ДНФ;
10) сегмента «e» у досконалій КНФ;
11) сегмента «f» у досконалій ДНФ;
12) сегмента «f» у досконалій КНФ.

ЛАБОРАТОРНА РОБОТА 4
Синтез комбінаційних схем вмикання динамічних процесів типу «біжучі вогні»

Мета роботи: набуття практичних навичок створення та синтезу комбінаційних схем моделювання динамічних процесів типу «біжучі вогні» з розробленням схем-програм мовою FBD з використанням блоків в Unity Pro та налагодження мінімізованих ФАЛ.

Обладнання та ПЗ: цифрова ПЕОМ із системним програмним забезпеченням Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або більш пізнішої.

Хід виконання роботи

1 Вивчити весь теоретичний матеріал для досягнення мети ЛР.

2 Створити проект за планом, викладеним нижче, та ввести схему-програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити проект звіту до ЛР (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

6 Оформити звіт з висновками й отримати бали за ЛР у викладача.

Послідовність створення проекту

Для виконання ЛР 4 необхідно відкрити створений проект ЛР 1 і зберегти його як laba04 або самостійно створити проект за процедурою, наведеною в ЛР 1, з назвою laba04.

Послідовність виконання роботи

Як зазначалось у лекціях, візуальний ефект «біжучі вогні» реалізується шляхом послідовного вмикання не менш ніж трьох світлодіодів на час 0,5 с кожний. Така динамічна модель

реалізується послідовним перебором булевих змінних, які моделюють електронні аналоги реле y_i шляхом вимикання попереднього y_{i-1} та вмикання наступного y_{i+1} із затримкою на спрацювання 0,5 с (FFB-об'єкт TON з параметром PT=t#500ms).

Для виконання роботи необхідно створити новий проект з такими самими апаратними модулями, як і в ЛР 1, і налаштувати модуль ETY PORT. Далі відкриваємо вікно Variables & FB instances -> Elementary Variables і створюємо 5 змінних (рисунок 25).

Name	Type	Address
y1	EBOOL	%Q0.3.1
y2	EBOOL	%Q0.3.2
y3	EBOOL	%Q0.3.3
y4	EBOOL	%Q0.3.4
start	EBOOL	%I0.2.1

Рисунок 25 – Змінні проекту *laba04*

Наступним кроком на вкладці меню Structural view знаходимо параметр Program, вибираємо Tasks, потім MAST, далі Section і ПКМ вибираємо New Section. Нову секцію краще назвати *laba4_1*, а мову програмування вибрати ***LD***. Елемент схеми TON можна знайти на панелі закладок, він буде зображений у такому вигляді:  , або натиснути ПКМ у місці створення екземпляра FBB-об'єкта класу TON з іменем екземпляра FBI_1 та вибрати FFB Input Assistant (рисунок 26), де в FFB-тип ви записуєте TON, і натиснути «ОК».

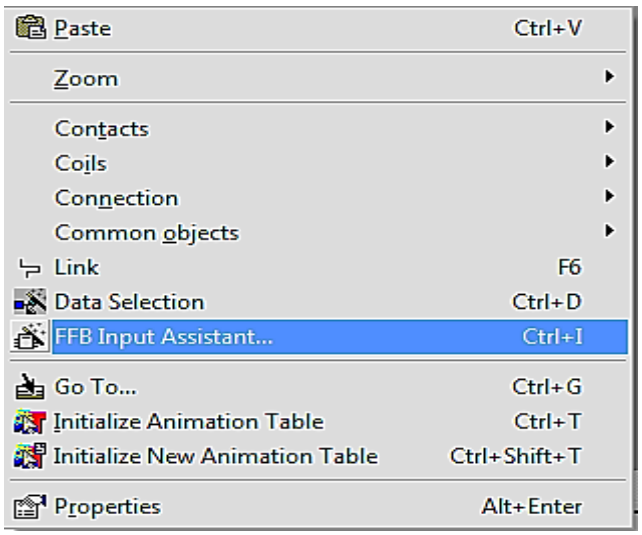
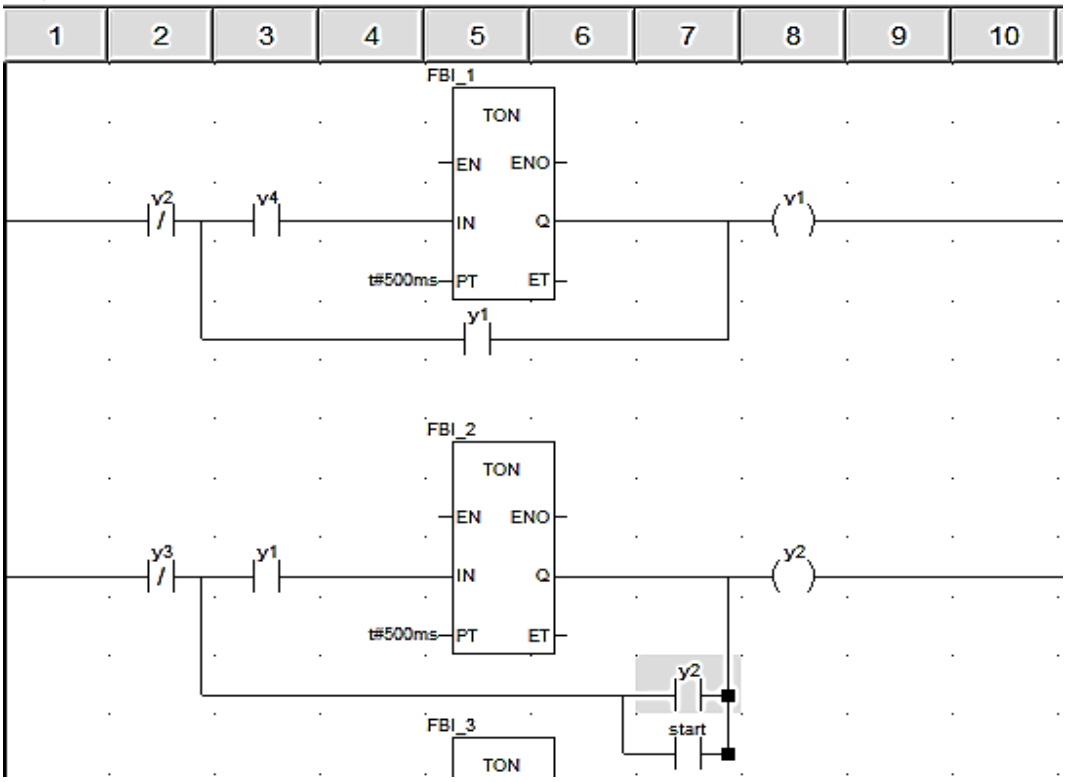


Рисунок 26 – Вставляння FBB-об'єкта через меню

«Схеми» вмикання y_1, y_2, y_3 та y_4 зображені на рисунках 27, 28.

а)



б)

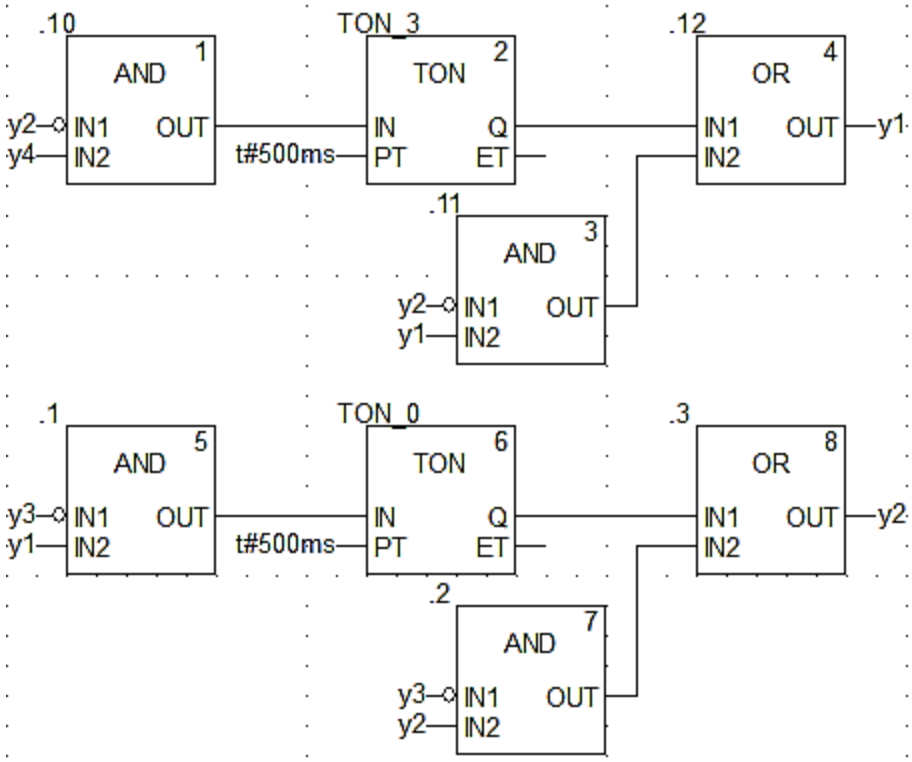
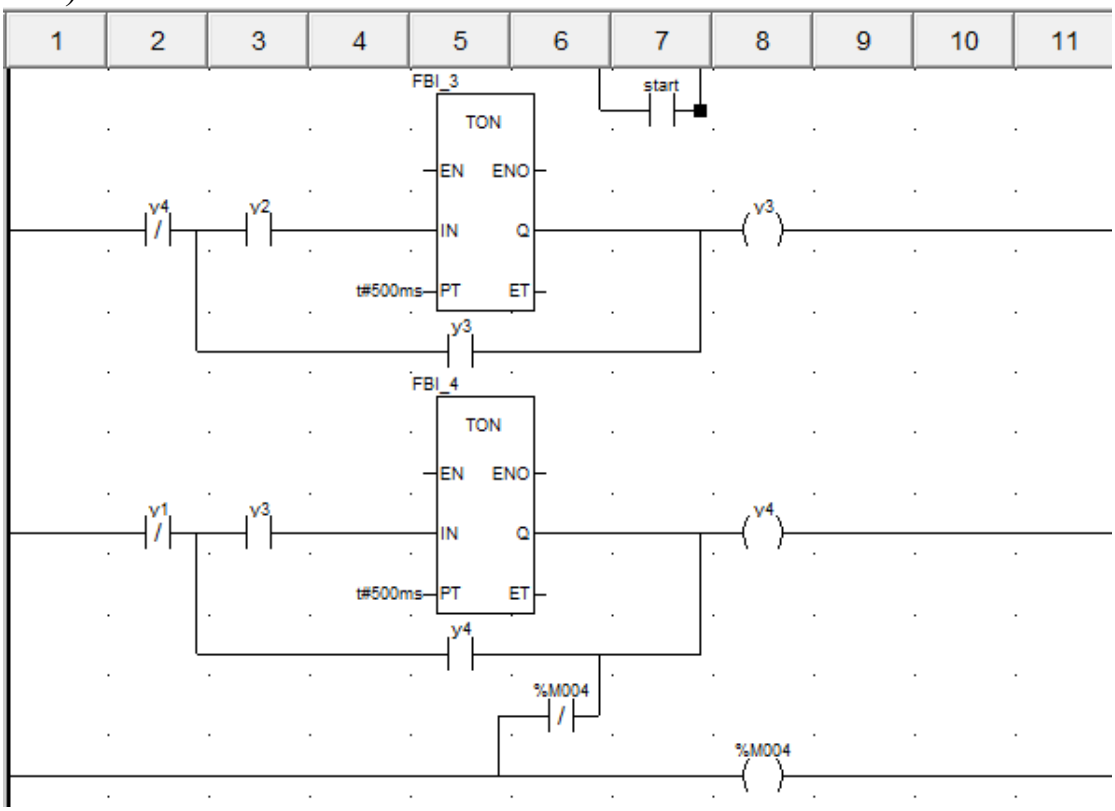


Рисунок 27 – «Схеми» вмикання ***y1*** та ***y2*** мовою LD (а) та мовою FBD (б)

а)



б)

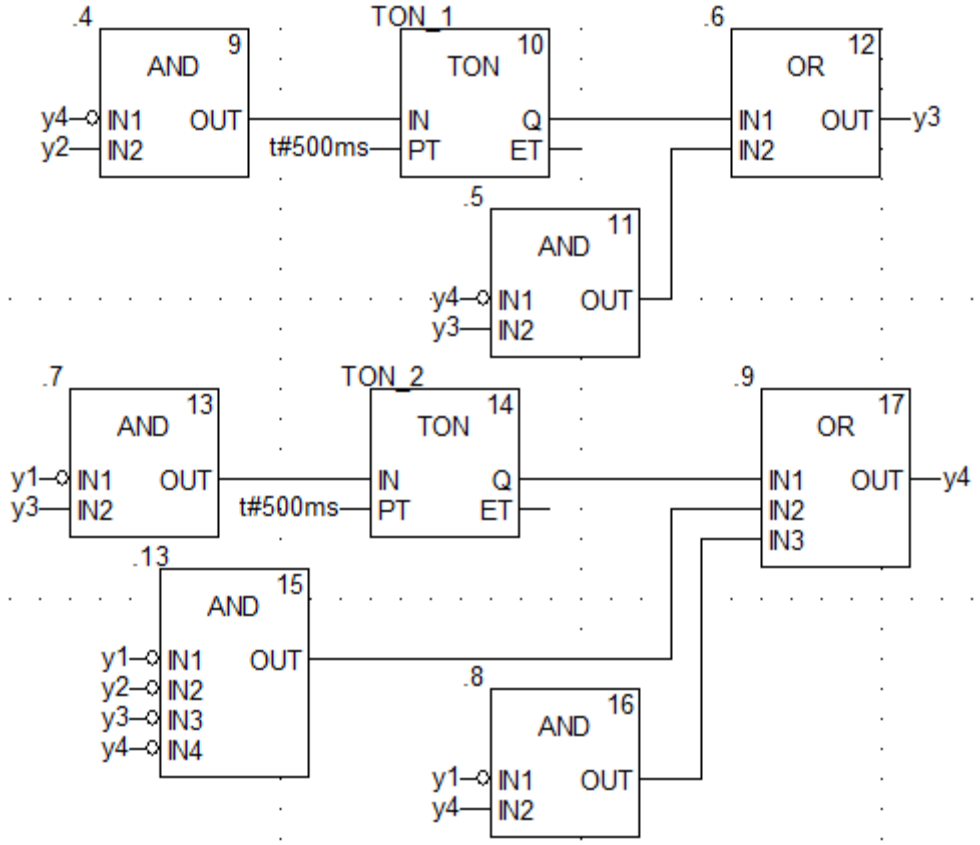


Рисунок 28 – «Схеми» вмикання ***y3*** та ***y4*** мовою LD (а) та мовою FBD (б)

Увага! Під час виконання роботи мовою LD у вас може виникнути така помилка, як на рисунку 29. Уникнути її можна побудовою елементів на відстані один від одного.

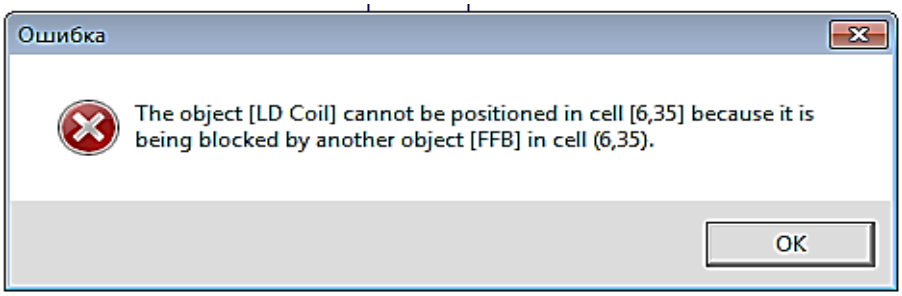


Рисунок 29 – Повідомлення про помилку розташування «реле»

Далі відкриваємо Operation Screen -> New Screen і створюємо чотири квадрати (рисунок 30), вибираємо їх колір і налаштовуємо до кожної із змінних y_1, y_2, y_3 та y_4 відповідно, ставимо в секції «Display condition» опцію біта Bit=1.

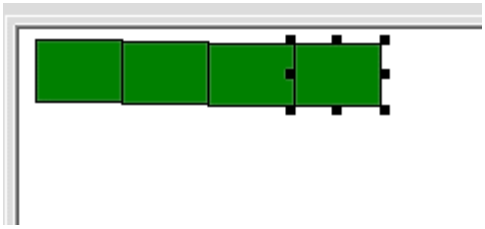


Рисунок 30 – Один набір квадратів для візуалізації роботи програми

Копіюємо ці чотири квадрати і вставляємо поряд декілька разів у лінію. Наступним кроком виконання роботи є повторне з'єднання з ПЛК (Ctrl+K) та передача/завантаження (Load) схеми-програми до ПЛК (Ctrl+L) й запуск (Run) її у віртуальному ПЛК (Ctrl+R). Вибрати екран користувача (наприклад Scr_laba04) і після натискання F7, або включивши режим «Enable Variable Modification» іншим способом, натиснувши ЛКМ на піктограмі  , моделювати розташування зображень (рисунок 31). Якщо є невідповідність, провести коригування схеми-програми або налаштування об'єктів екранів оператора та повторну перевірку.

Запускаємо програму.

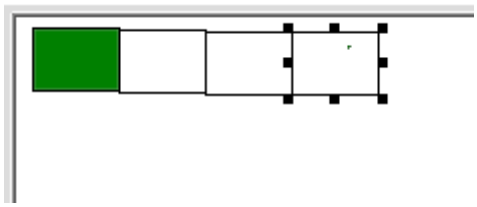


Рисунок 31 – Один набір квадратів з візуалізацією роботи програми

Контрольні питання

1 Де можна знайти елемент TON?
2 Для чого потрібен модуль TOF?
3 Параметри вхідних даних елемента TON.
4 Типи даних елемента TON.
5 Параметри вихідних даних елемента TON.
6 Як необхідно змінити програму з чотирьох до трьох «реле»?
7 Як необхідно змінити програму з чотирьох до шести «реле»?

Індивідуальне завдання

1 Створити «біжучі вогні» з автозапуском.
2 За власною ідеєю зробити миготливий рисунок.
3 Створити вогні з автозапуском, без допоміжних «реле».
4 Створити «біжучі вогні» у зворотному напрямку.
5 За власною ідеєю зробити миготливий кільцевий рисунок.
6 Створити вогні з автозапуском засобами допоміжних «реле».
7 Створити «біжучі вогні» з шести елементів.
8 За власною ідеєю зробити миготливий рисунок.
9 Створити вогні з автозапуском, без допоміжних «реле».

ЛАБОРАТОРНА РОБОТА 5
Створення DFB-модулів для моделювання тризначного автоблокування

Мета роботи: набуття практичних навичок зі створення та синтезу комбінаційних схем зі створюваними DFB-модулями, складання програм мовою FBD з елементами візуалізації моделювання з використанням блоків в Unity Pro.

Обладнання та ПЗ: цифрова ПЕОМ із системним програмним забезпеченням Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або більш пізнішої.

Хід виконання роботи

1 Вивчити весь теоретичний матеріал для досягнення мети ЛР.

2 Створити проект за планом, викладеним нижче, та ввести схему-програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити проект звіту до ЛР (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

6 Оформити звіт із висновками й отримати бали за ЛР у викладача.

Послідовність створення проекту

Для виконання ЛР 5 необхідно відкрити створений проект ЛР 1 і зберегти його як laba05 або самостійно створити проект за процедурою, наведеною в ЛР 1, з назвою laba05.

Послідовність виконання роботи

Як і в попередній роботі, компонуємо всі необхідні модулі та активуємо їх. Наступний крок – відкриваємо вікно Derived FB Types, щоб створити новий FBD-тип, на основі якого потім будуть створюватися екземпляри. Крім унікальної назви типу (за правилами створення ідентифікаторів), необхідно створити локальні вхідні та вихідні змінні в FBD-модулі та логіку «перетворення» вхідних змінних у вихідні й записати параметри, як на рисунках 32, 33. У випадку імітації маятникового трансмітера (МТ) створимо екземпляр даних функціонального блока DFB-типу з назвою D_MT, який буде відповідати за миготливий вихід.

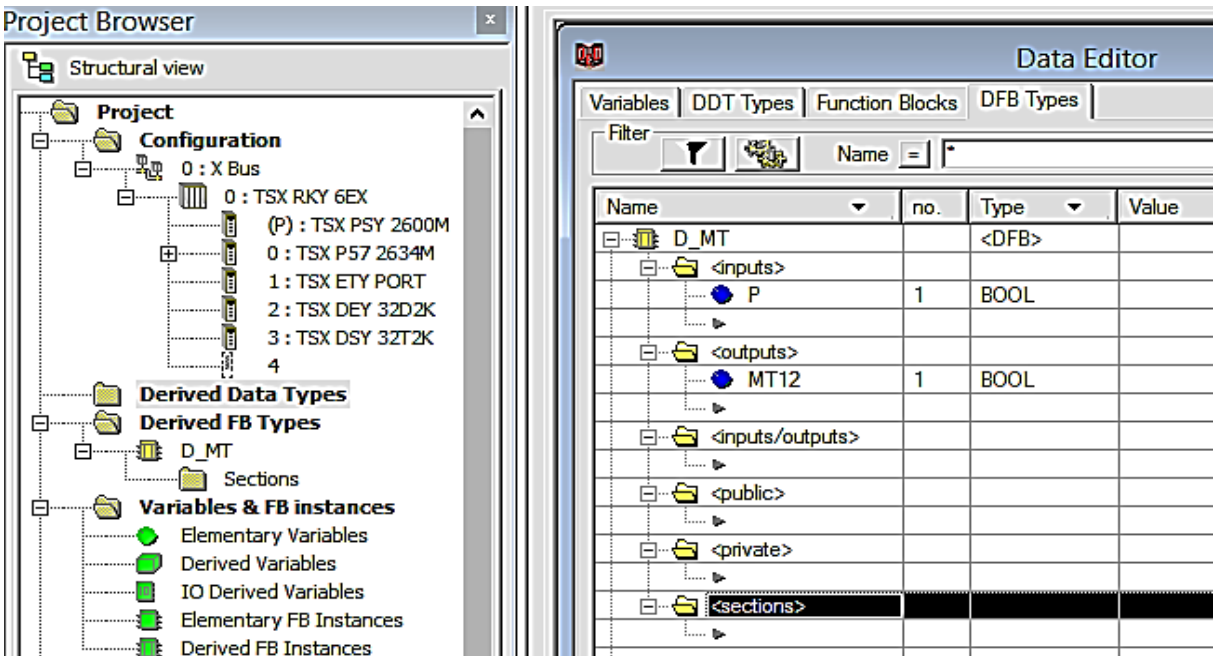


Рисунок 32 – Створення в розділі проекту Derived FB Types DFB-типу з назвою «D_MT»

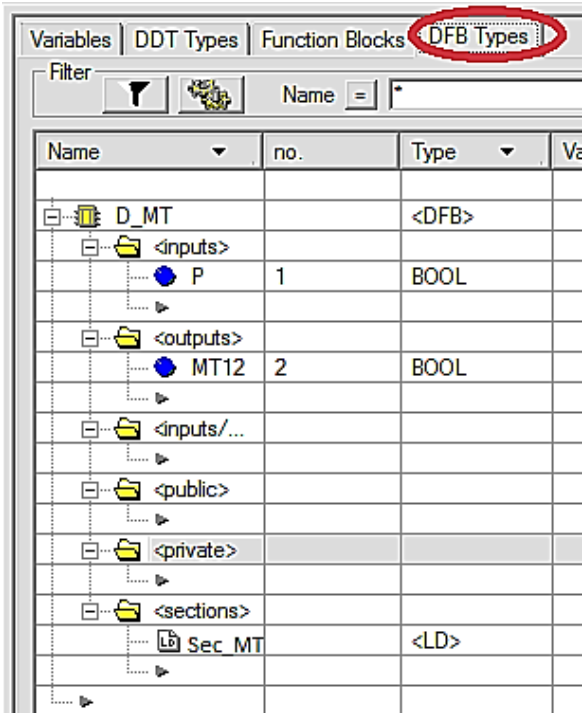
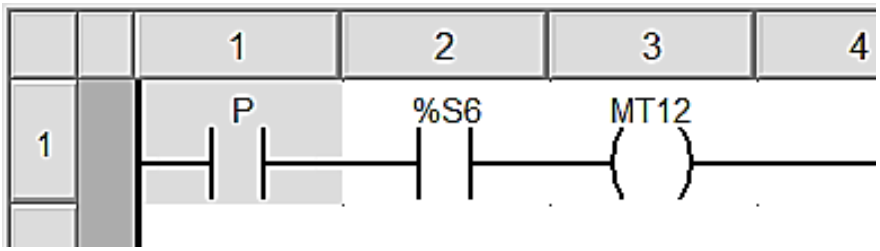


Рисунок 33 – Розроблення DFB-типу «D_MT» з назвою змінних і визначенням їх типів

Потім будуємо схему (рисунок 34, а), натискаючи на об’єкт Sec_MT два рази у вікні (рисунок 34, б).

а)



б)

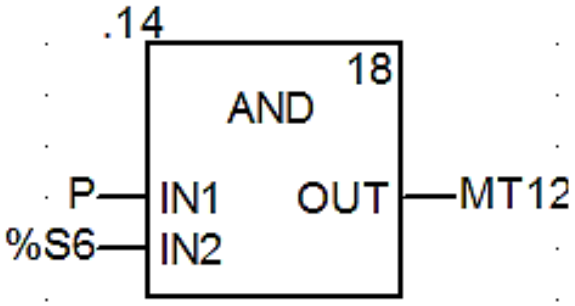


Рисунок 34 – Внутрішня схема DFB блока типу D_MT мовою LD (а) та мовою FBD (б)

Відкриваємо операторський екран Screen05 і зображуємо світлофор з розташуванням показань тризначного автоблокування (АБ), як на рисунку 35, і задіємо кожен з кружків для трьох кольорів показань, де будуть наступні параметри з можливістю задавати в полі Variable локалізовані (рисунок 36) і нелокалізовані змінні.

Компілюємо підпрограму Sec_MT. Випробовуємо програму з екземпляром DFB блока типу D_MT, як зі стандартним блоком.

Після цього будуємо функціональну модель сигнальної точки для тризначного АБ за схемами двоколійного АБ постійного струму при автономній тязі для ділянок з одностороннім рухом поїздів.

Виокремлюємо схему сигнальної точки в окрему «типову» схему вмикання ламп світлофора. Сигнали залежать від стану рейкового кола, яке огороджене світлофором, і стану попередньої сигнальної точки, які подані змінними.

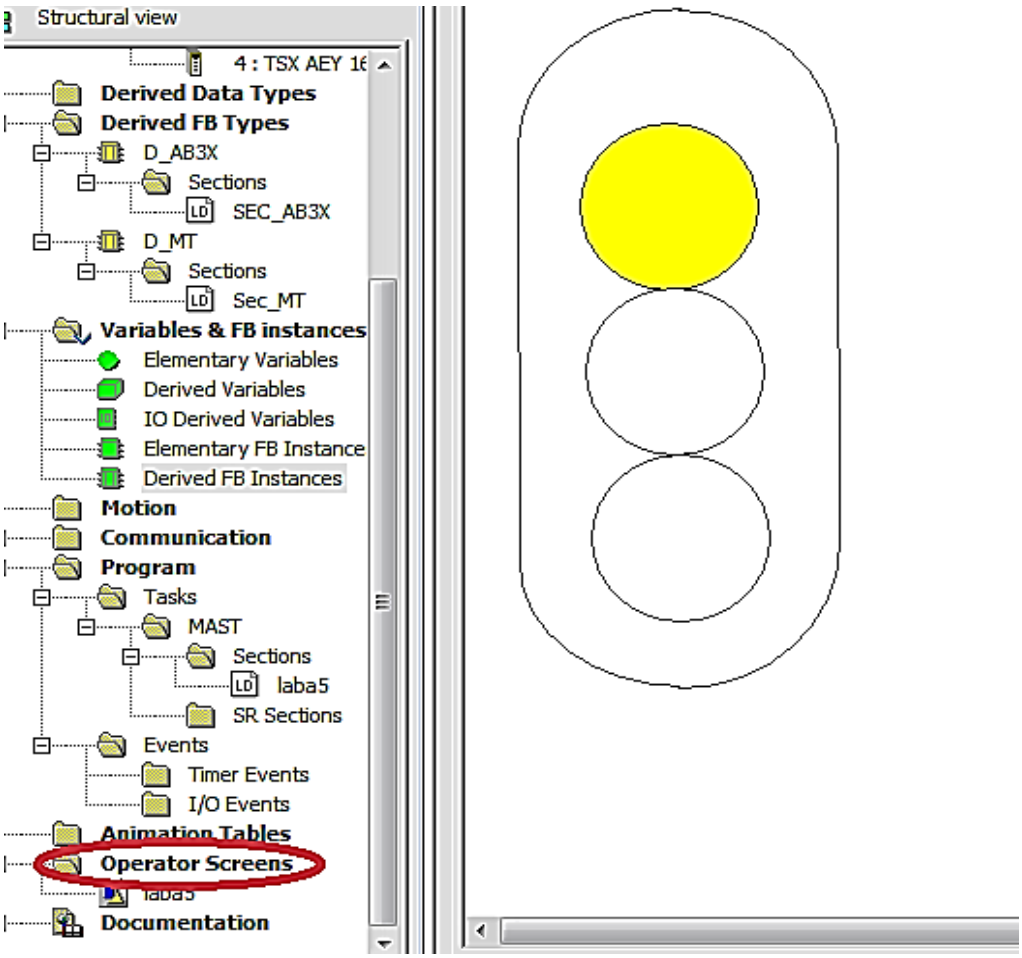


Рисунок 35 – Зображення показань тризначного АБ

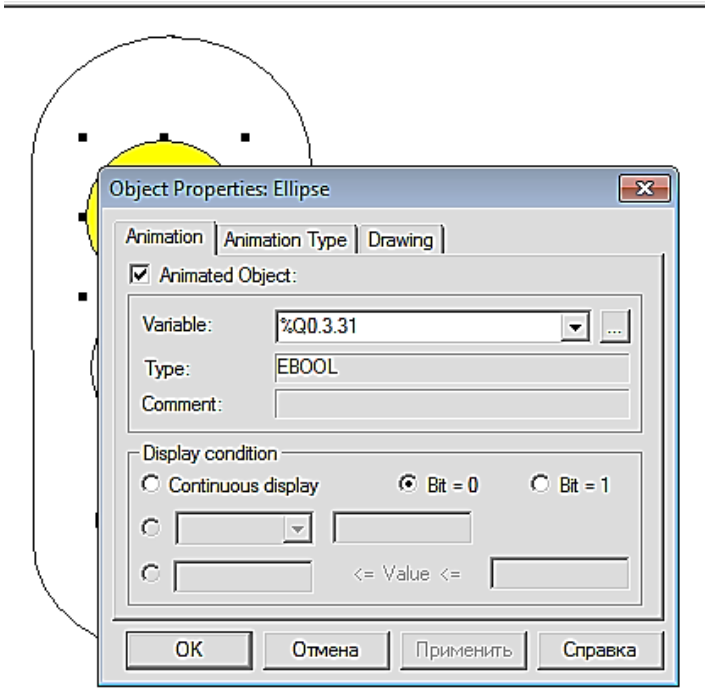


Рисунок 36 – Налаштування верхнього показання

Створюємо функціональний блок користувача для сигнальної точки тризначної АБ «D_AB3X» (рисунок 37).

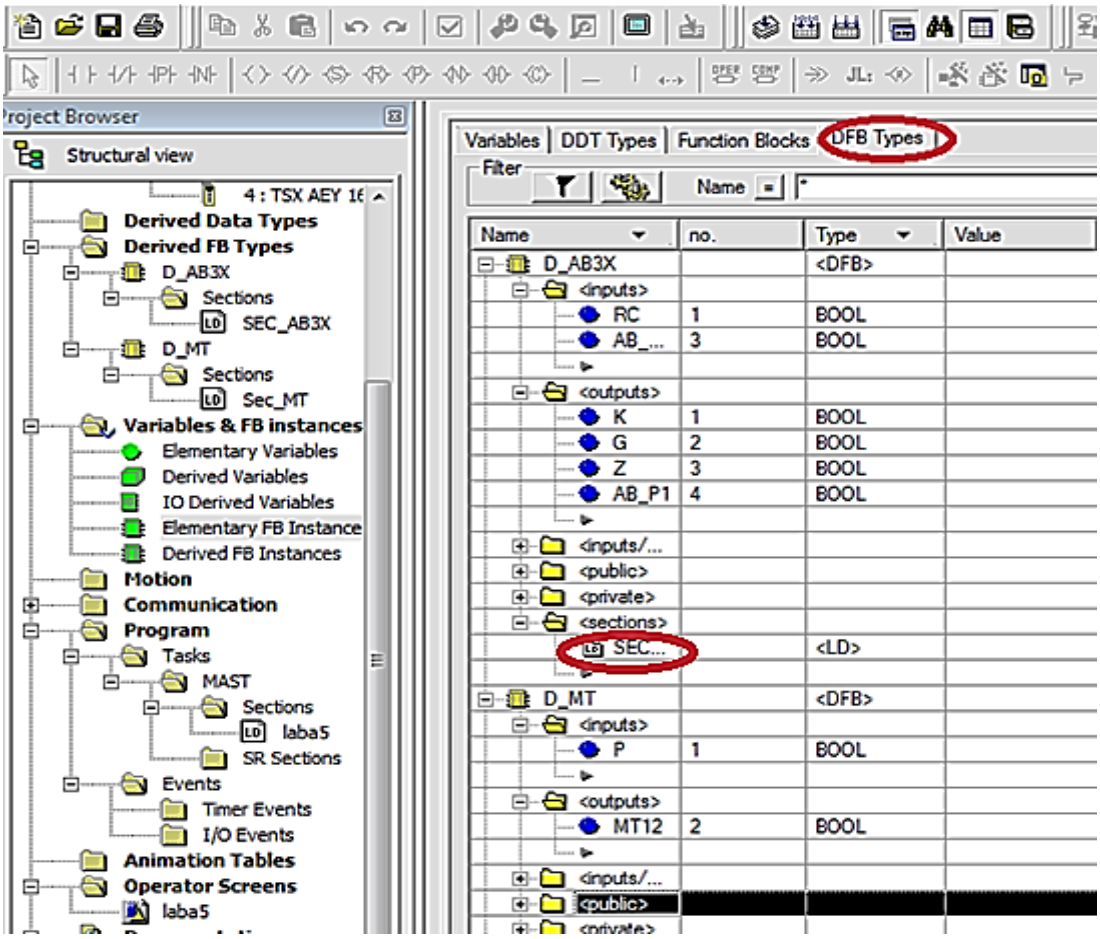


Рисунок 37 – Структура DFB типу з ім’ям D_AB3X

Створивши DFB типи для тризначного АБ, розробляємо програму поєднання вхідних і вихідних змінних будь-якою з п’яти мов. Для цього в секції D_AB3X та D_MT відкриваємо вкладку Section і створюємо SEC_AB3X та SEC_MT, відповідно натискаючи на кожну два рази мишею (має з’явитися вікно, як на рисунку 38).

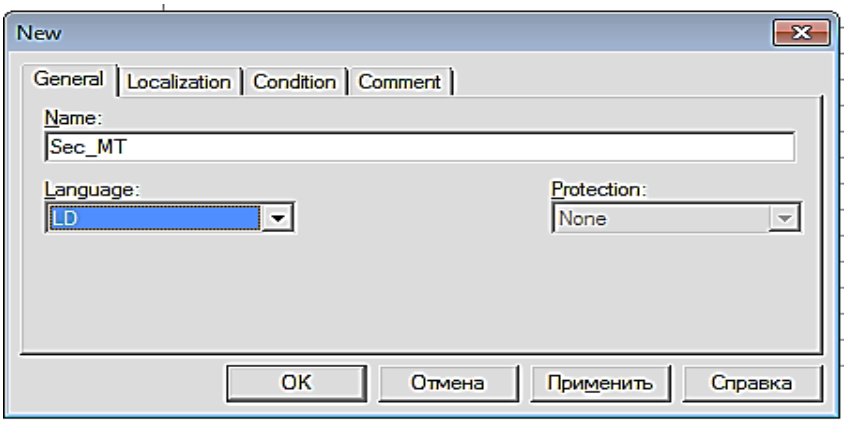
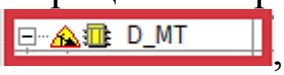
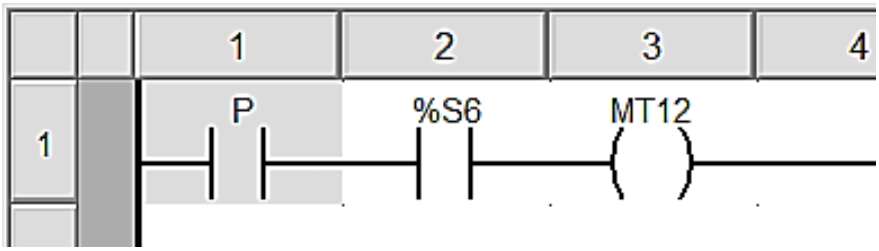


Рисунок 38 – Вибір мови програмування для SEC_MT

У процесі створення цих елементів буде відображатися такий знак: 

, це означає, що елемент у процесі обробки. Після завершення створення треба скомпілювати програму, щоб не було помилок, за допомогою символа  . У секціях, які ми відкрили, створюємо такі схеми, як на рисунках 39, 40.

а)



б)

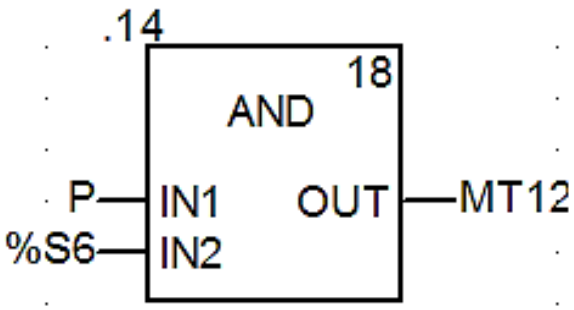
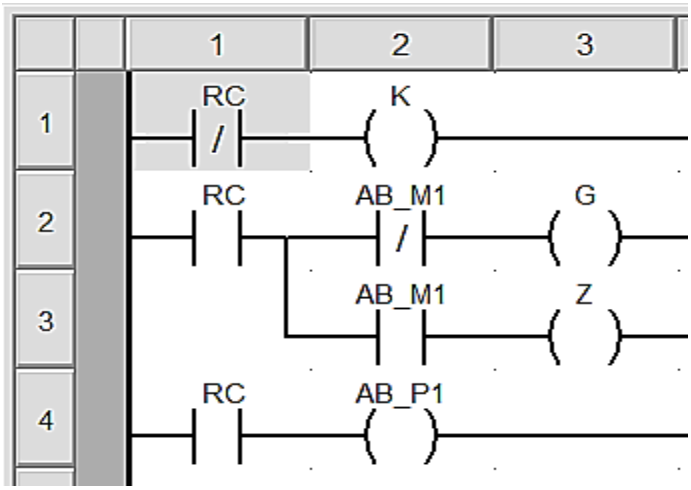


Рисунок 39 – Внутрішня схема DFB блока типу D_MT мовою LD (а) та мовою FBD (б)

а)



б)

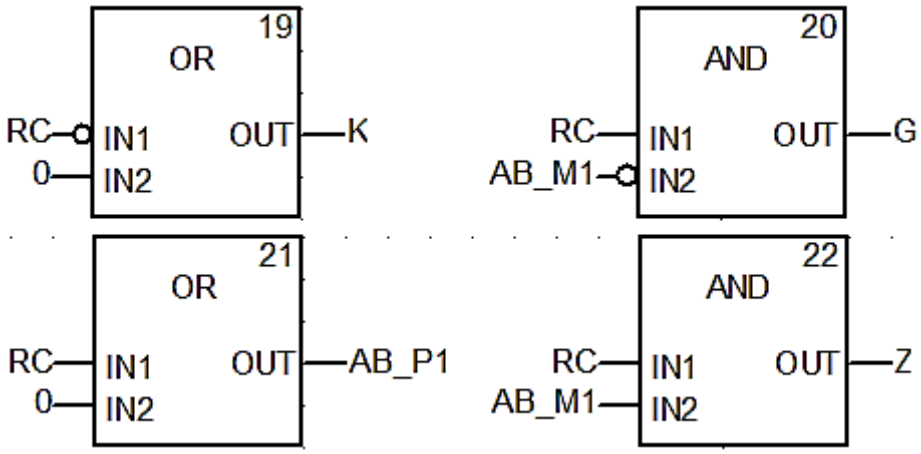


Рисунок 40 – Схема SEC_AB3X для тризначного світлофора мовою LD (а) та мовою FBD (б)

Після змін компілюємо програму. Далі відкриваємо вже знайому нам вкладку Program і відкриваємо створені нами схеми (так, як ми створювали TON у попередній роботі), вписуючи їхні назви (рисунок 41).

Вихідну змінну AB_P1 блока FBI_1 допускається контролювати в інших колах як контакт з ім'ям ***FBI_1.AB_P1.*** Блоки будуємо один під одним. Запускаємо програму, як наведено на попередніх рисунках або рисунках 42, 43, 44. У режимі емуляції роботи ПЛК відображаються червоним кольором розімкнені кола, а зеленим кольором – замкнуті.

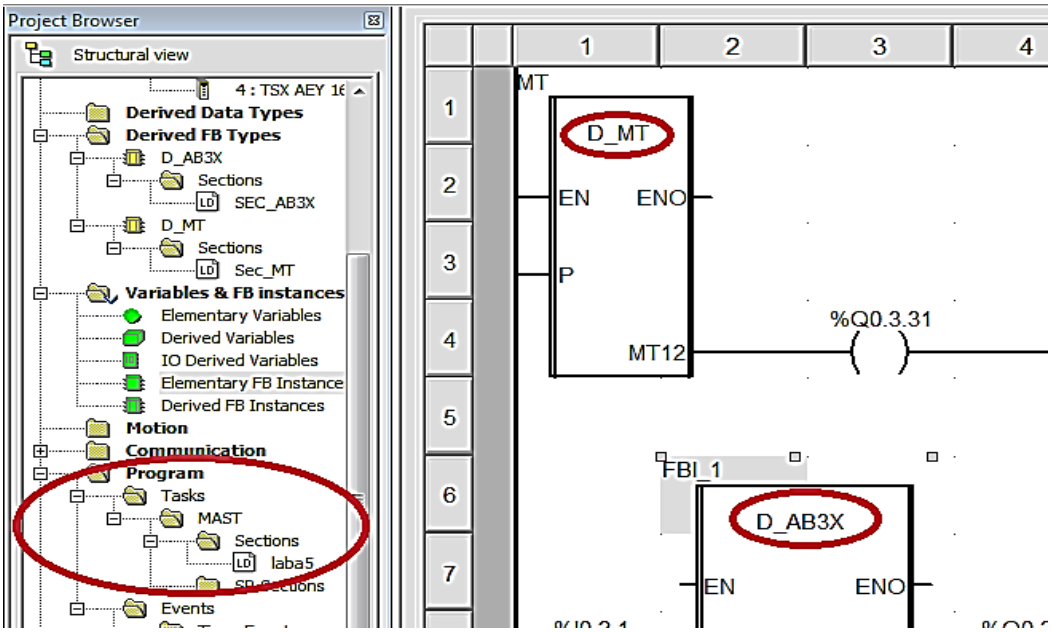


Рисунок 41 – Схема для перевірки екземпляра MT

Створивши екземпляри FBI_1, FBI_2, FBI_3 на базі функціонального блока користувача (DFB-типу з ім’ям D_AB3X), підключаємо до їхніх виходів «реле» з локалізованими змінними (рисунок 42).

а)

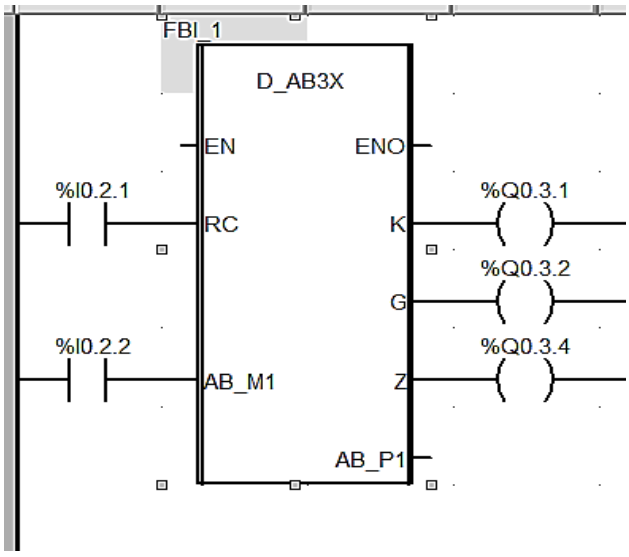
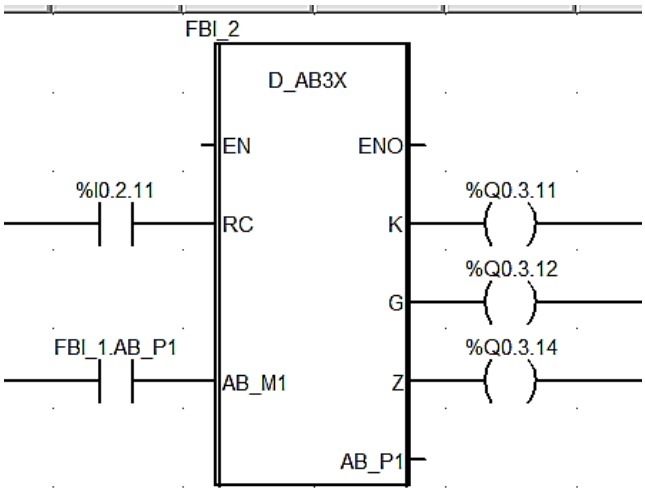


Рисунок 42 – Схеми підключення екземплярів FBI_1 (а), FBI_2 (б), FBI_3 (в), аркуш 1

б)



в)

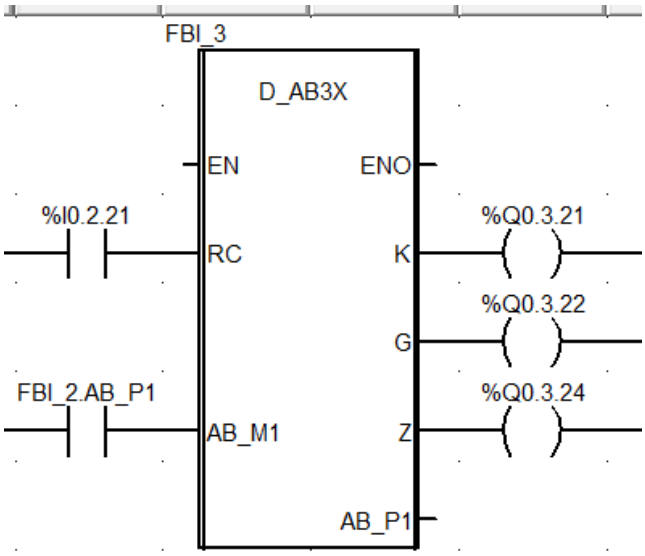


Рисунок 42, аркуш 2

Щоб перевірити, чи правильно працює програма, треба встановити в «1» змінну %I0.2.21 ПКМ Force Value 1 біт (рисунки 43, 44).

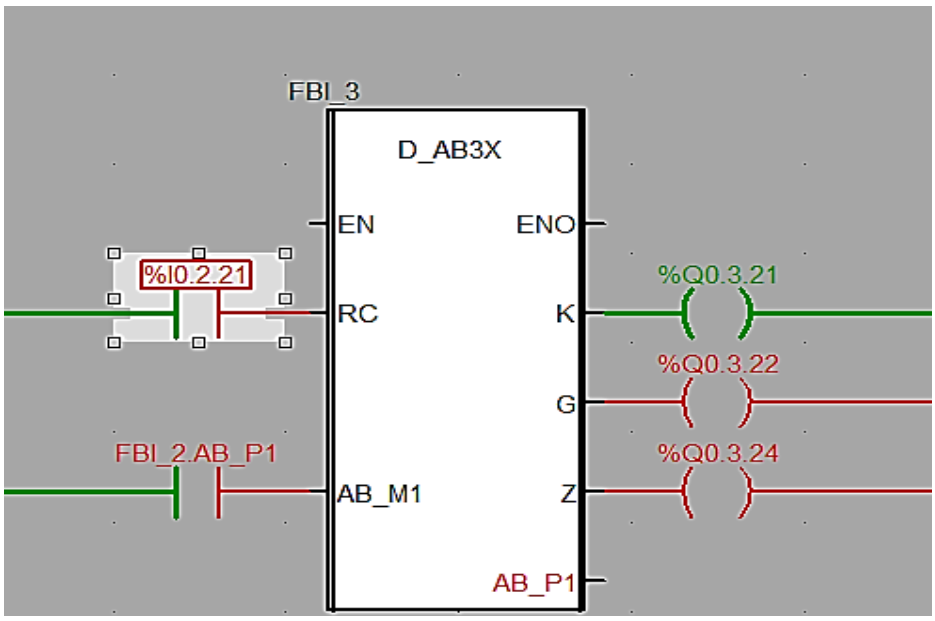


Рисунок 43 – Перевірка роботи FBI_3

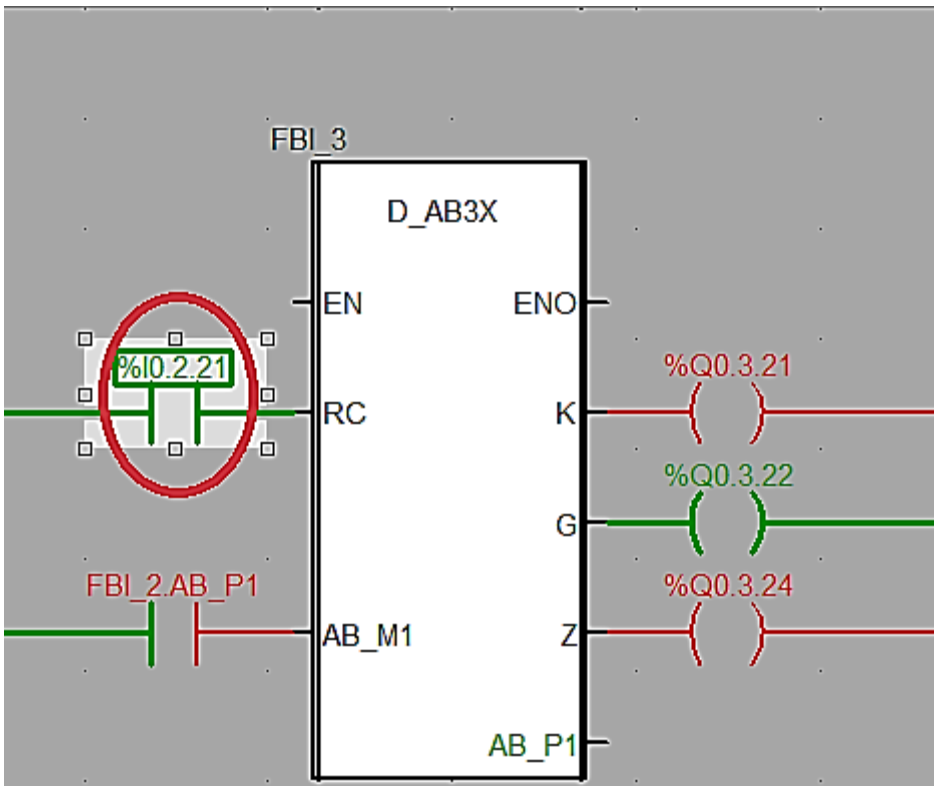


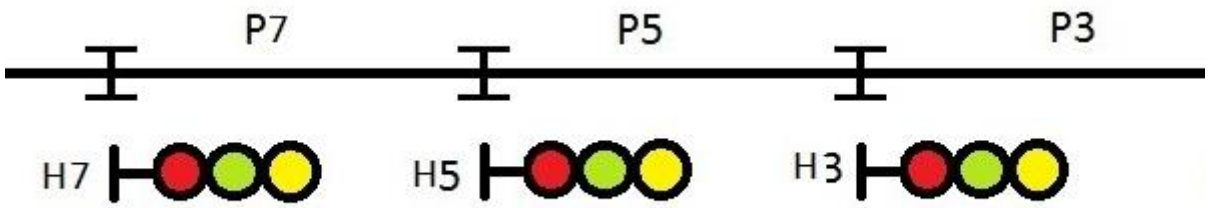
Рисунок 44 – Зміна стану входу «RC» блока FBI_3

Контрольні питання

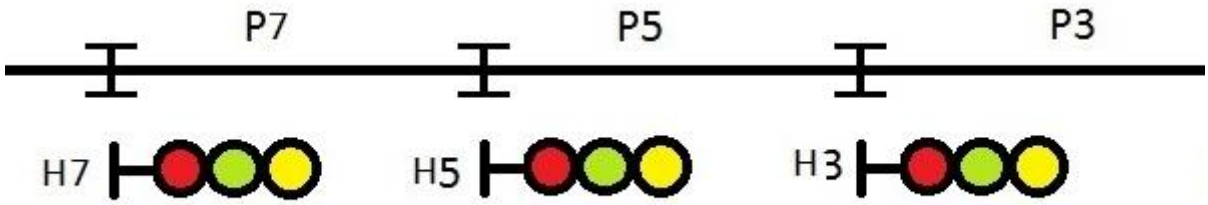
1 Як створити новий блок?
2 Як працює прохідна та передвхідна сигнальна точка?
3 Як підключити три лампи світлофора до блока D_AB3X?

Індивідуальне завдання зі створенням власних DFB типів

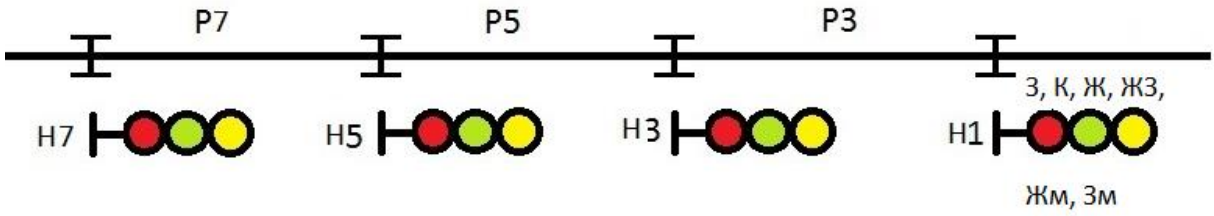
1 Схематично зобразити у вікні Screen_laba5_dzE ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і трьох світлофорів Н7, Н5, Н3, з'єднавши їх у єдину технологічну модель роботи за схемою перегону, наведеною нижче.



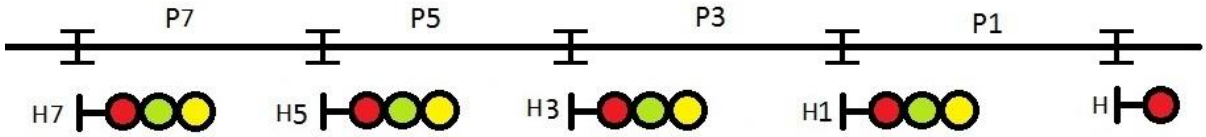
2 Схематично зобразити у вікні Screen_laba5_dzD ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і трьох світлофорів Н7, Н5, Н3, з'єднавши їх у єдину технологічну модель роботи за схемою перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи.



3 Схематично зобразити у вікні Screen_laba5_dzC ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і чотирьох світлофорів Н7, Н5, Н3, Н1, з'єднавши їх у єдину технологічну модель роботи за схемою чотиризначного автоблокування перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи та вимиканням зеленого вогню при показанні «жовтий із зеленим» у випадку перегорання лампи жовтого вогню й інтерактивним введенням зайнятості рейкових кіл Р7, Р5, Р3 і роботи світлофора Н1 (шість показань).



4 Схематично зобразити у вікні Screen_laba5_dzA ділянку колії з чотирьох рейкових кіл Р7, Р5, Р3, Р1 і п’яти світлофорів Н7, Н5, Н3, Н1, Н, з’єднавши їх у єдину технологічну модель роботи за схемою чотиризначного автоблокування перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи та вимиканням зеленого вогню при показанні «жовтий із зеленим» у випадку перегорання лампи жовтого вогню з роботою Н1 у режимі передвхідної сигнальної установки з інтерактивним введенням зайнятості рейкових кіл Р7, Р5, Р3, Р1 і роботи світлофора Н (шість показань – К, Жв, З, Жн+Жв, Жн+Жв+ЗП, К+Бмиг).



ЛАБОРАТОРНА РОБОТА 6
Синтез схем побудови шифраторів кодових сигналів у DFB-базисі

Мета роботи: набуття практичних навичок створення та синтезу комбінаційних схем побудови шифраторів кодових сигналів у DFB-базисі з використанням блоків в Unity Pro.

Обладнання та ПЗ: цифрова ПЕОМ із системним програмним забезпеченням Windows XP або вище та прикладне ПЗ Unity Pro 3.0 або більш пізнішої.

Хід виконання роботи

1 Вивчити теоретичний матеріал для досягнення мети ЛР.

2 Створити проект за планом, викладеним нижче, та ввести схему-програму типового завдання.

3 Визначити та розв’язати індивідуальне завдання.

4 Оформити проект звіту до ЛР (номер, назва, мета, обладнання та ПЗ, принципова схема підключення контактів (виводів) контролера (уявного), схема-програма розв’язання індивідуального завдання для Unity Pro мовою FBD).

5 Ввести програму мовою FBD для Unity Pro в лабораторії, отримати результати.

6 Оформити звіт з висновками й отримати бали за ЛР у викладача.

Послідовність створення проекту

Для виконання ЛР 6 необхідно відкрити створений проект ЛР 1 і зберегти його як laba06 або самостійно створити проект за процедурою, наведеною в ЛР 1, з назвою laba06.

Послідовність виконання роботи

Як і в попередній роботі, компонуємо всі необхідні модулі та активуємо їх. Наступний крок – відкриваємо вікно Derived FB Types, щоб створити новий FBD-тип, на основі якого потім будуть створюватися екземпляри. Крім унікальної назви типу (за правилами створення ідентифікаторів), необхідно створити локальні вхідні та вихідні змінні в FBD-модулі та логіку «перетворення» вхідних змінних у вихідні й записати параметри.

Наступний крок – відкриваємо вікно Variables & FB instances -> Elementary Variables та створюємо нові змінні (рисунок 45).

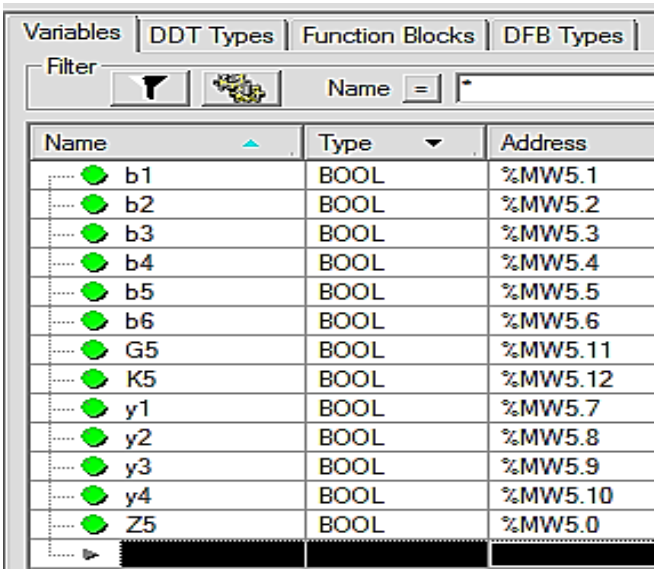
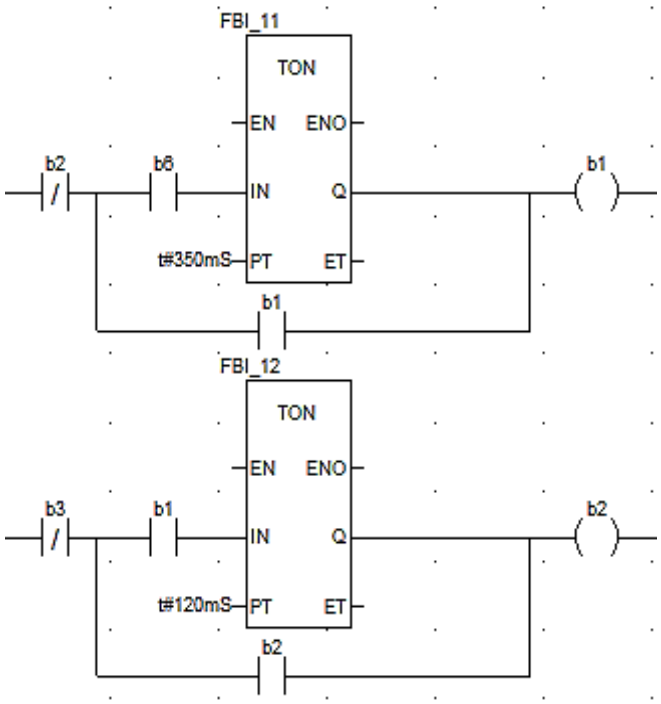


Рисунок 45 – Змінні для моделювання КПТШ 515

Потім відкриваємо Derived FB Types, щоб створити нові елементи схеми пристрою КПТШ515, і записуємо параметри подачі коду «З», «Ж», «КЖ», як на рисунках 46 – 48. Для коду «З» – три різні імпульси й три різні паузи; для коду «Ж» – два різні імпульси, дві різні паузи; «КЖ» – один імпульс, одна пауза.

а)



б)

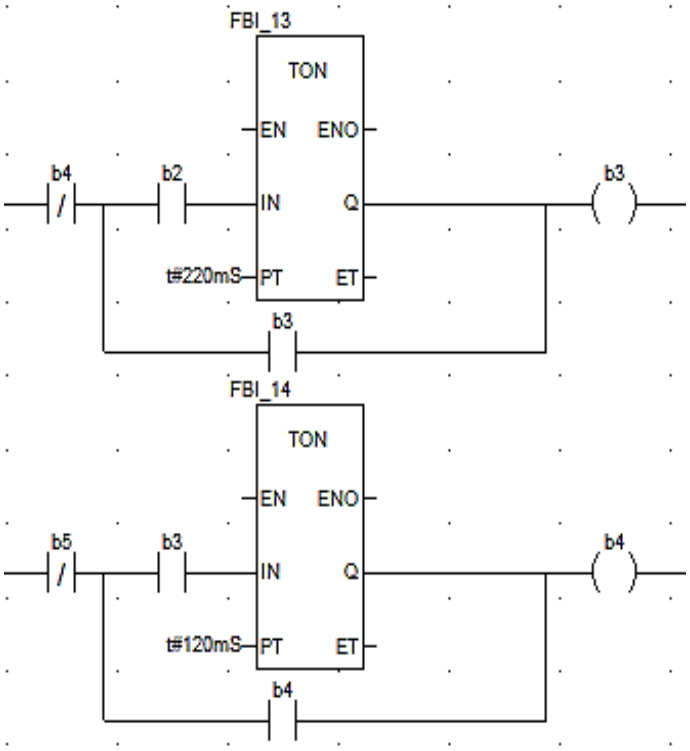


Рисунок 46 – Перша (а), друга (б) і третя (в) пари «імпульс-пауза» коду «З», аркуш 1

в)

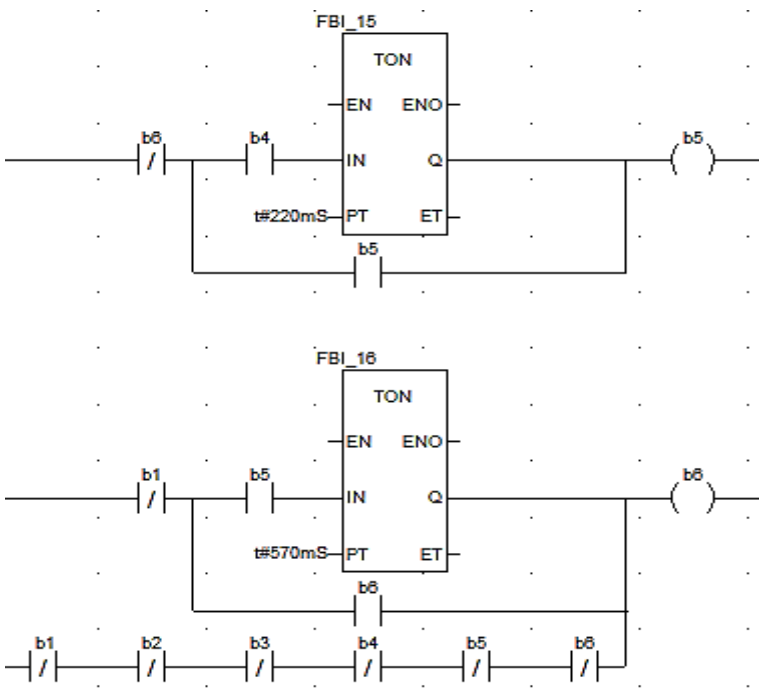


Рисунок 46, аркуш 2

а)

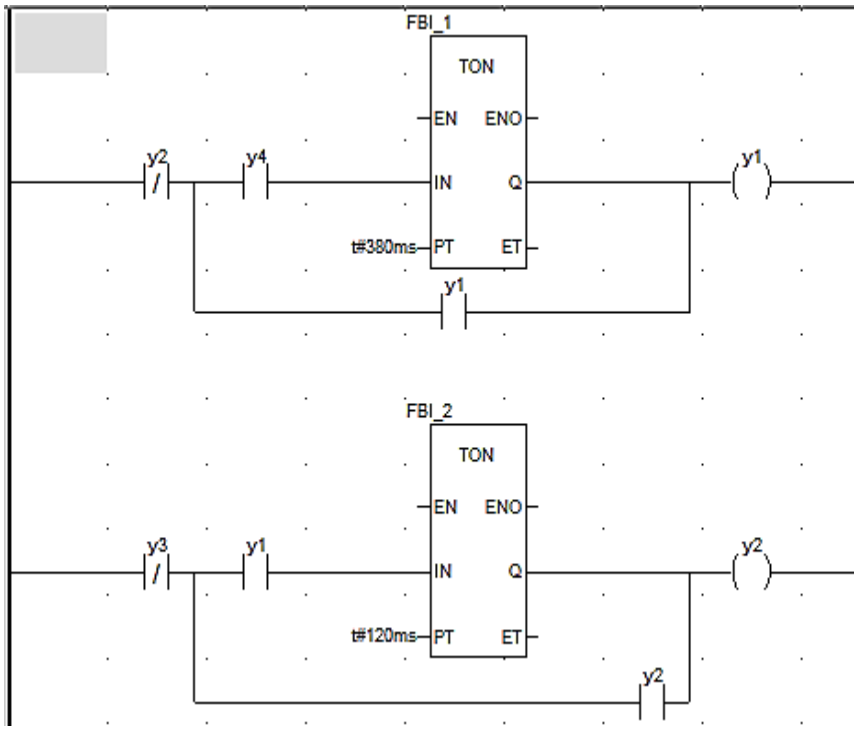


Рисунок 47 – Перша (а) і друга (б) пари «імпульс-пауза» коду «Ж», аркуш 1

б)

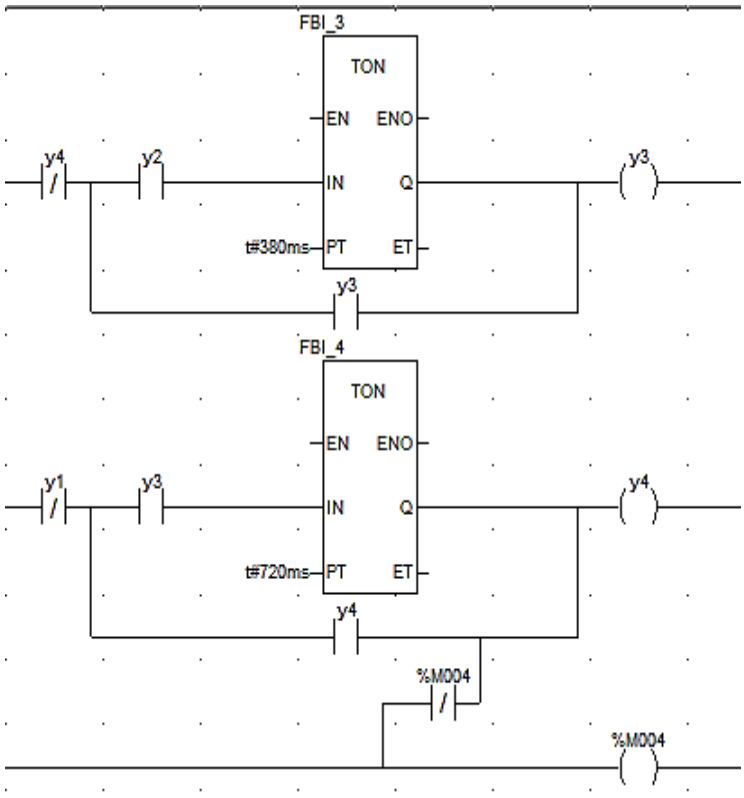


Рисунок 47, аркуш 2

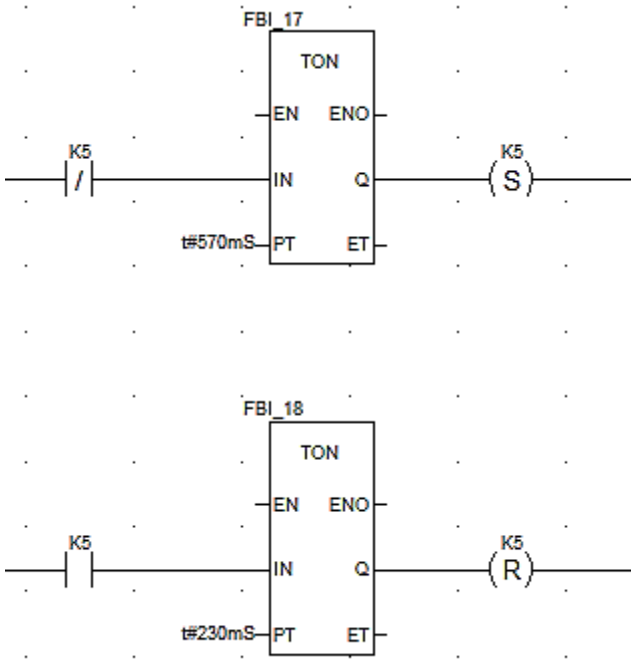


Рисунок 48 – Пара «імпульс-пауза» коду «КЖ»

Далі запишемо загальну схему подачі інвертованих імпульсів КПТШ515 (рисунок 49).

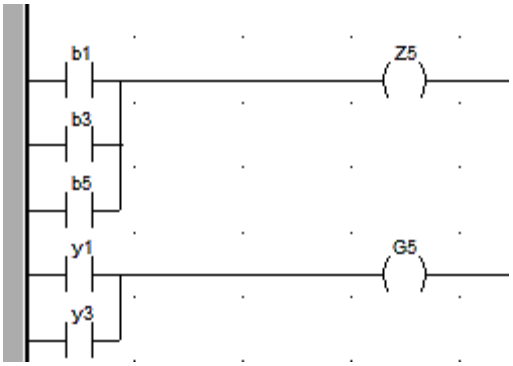


Рисунок 49 – Схема інвертованих виходів «З» та «Ж»
КПТШ 515

Скомпілюємо та запустимо програму. На свій розсуд студенти можуть самостійно розробити програми мовою FBD.

Запустивши програму, ви маєте помітити, що імпульси для кожного коду працюють згідно із заданими в ЛР значеннями (рисунок 50).

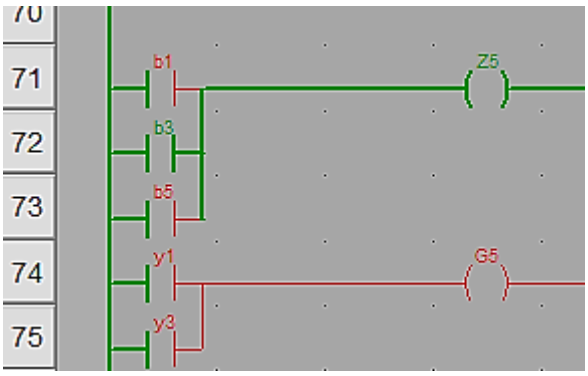


Рисунок 50 – Активна генерація другої паузи коду «З»

Наступним кроком побудуємо шифратор для КПТШ515. Відкриваємо вікно Variables & FB instances -> Elementary Variables і записуємо дані з рисунка 48 у секцію рисунка 51.

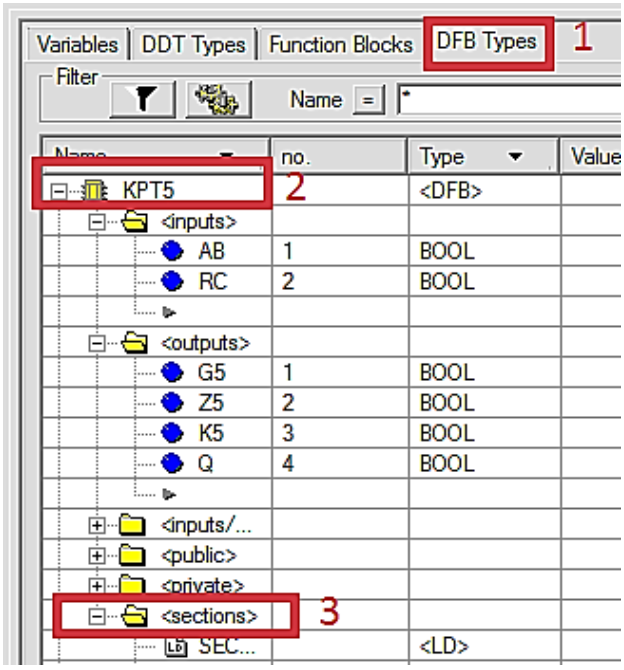


Рисунок 51 – Створення DFB типу КПТШ 515

У рядку Sections створюємо нову секцію мовою LD (рисунок 52).

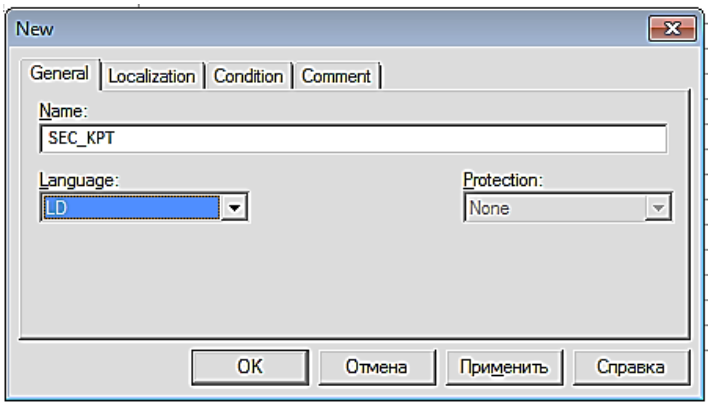


Рисунок 52 – Створення секції мовою LD

У цій секції реалізуємо таку схему, як на рисунках 46 – 49 з незначними змінами для генерації кодів КПТШ 515.

Компілюємо програму та знову повертаємося до вікна, у якому ми будували схеми КПТШ 515, там створюємо екземпляр класу, як показано на рисунках 53, 54.

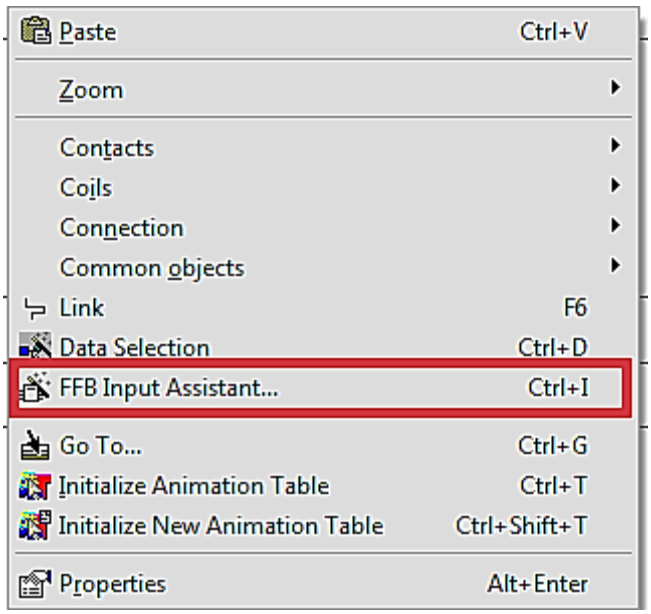


Рисунок 53 – Вставляння екземпляра FFB/DFB-типу

Записуємо назву FFB/DFB-типу (рисунок 54) у полі FFB type та коригуємо або підтверджуємо генеровану назву (FBI_21).

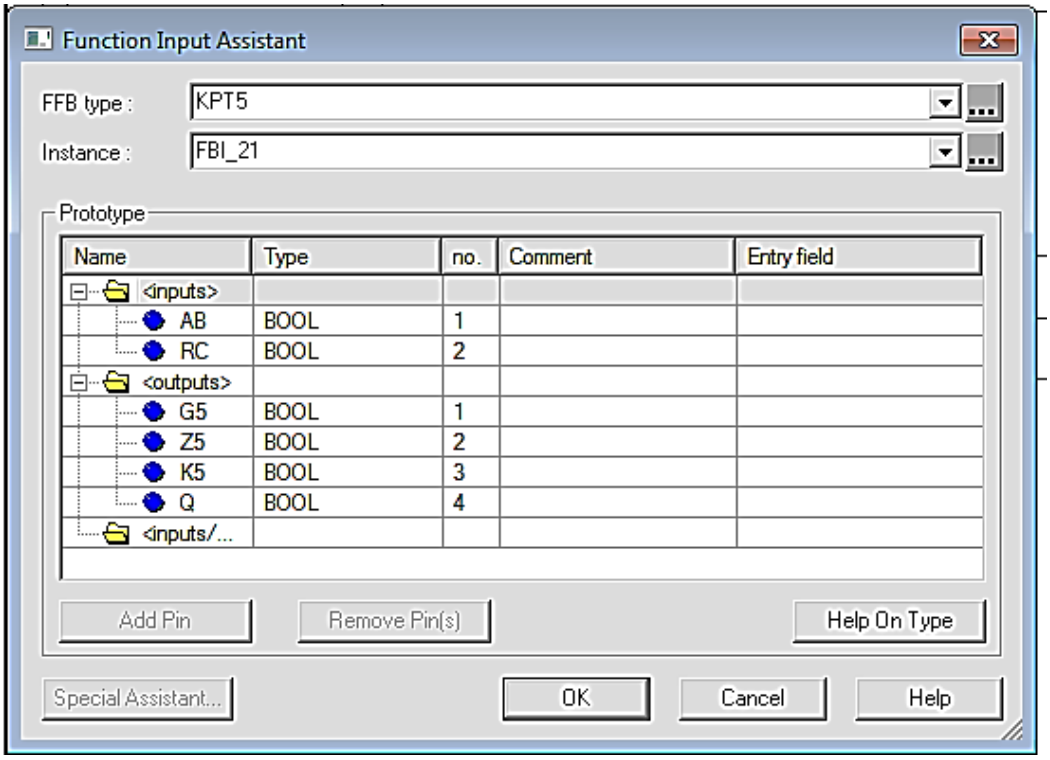


Рисунок 54 – Корекція назви екземпляра FFB/DFB-типу

На пустому вікні натискаємо ЛКМ, з’явиться екземпляр розробленого DFB-модуля, для якого необхідно доопрацювати схему під’єднання, як на рисунку 55.

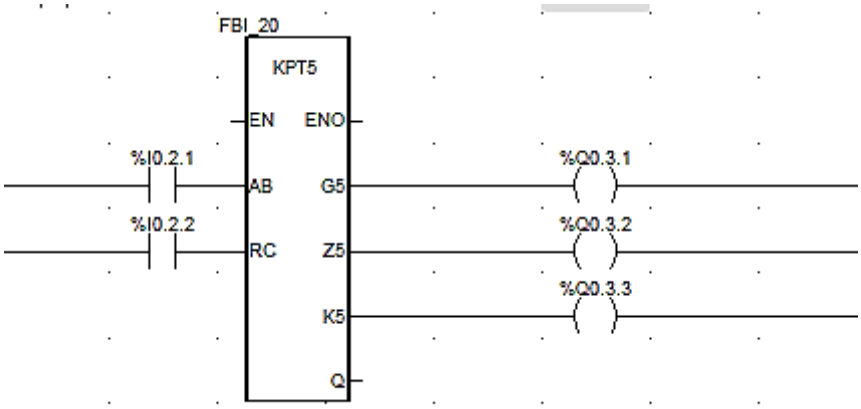


Рисунок 55 – Схема перевірки DFB-модуля КПТШ 515

Запускаємо програму й інтерпретуємо результат, використовуючи рисунок 56.

Контрольні запитання

1 Скільки імпульсів у кожного з кодів КПТШ 515, КПТШ 715, КПТШ 815, КПТШ 1015, КПТШ 1115?

2 Які ще бувають коди КПТШ 1015, КПТШ 1115?

3 Для чого потрібен КПТШ?

4 Як зробити кодери МТ–1, МТ–1М, МТ–2, МТ–2М?

5 Як зробити кодери ТП–24, ТП–24М?

Індивідуальне завдання із застосуванням DFB модулів (таблиця 6).

Таблиця 6 – Таблиця варіантів та завдань з кодами

Варіант	Тип трансмітера	Код
1	2	3
1	КПТШ-8	«З»
2	КПТШ-7	«Ж»
3	КПТШ-10	«А2»
4	КПТШ-8	«Ж»
5	КПТШ-11	«КЖ»

Продовження таблиці 6

1	2	3
6	КПТШ-9	«З»
7	КПТШ-5	«КЖ»
8	КПТШ-11	«Ж»
9	КПТШ-13	«А1»
10	КПТШ-9	«КЖ»
11	КПТШ-11	«З»
12	КПТШ-5	«Ж»
13	КПТШ-7	«З»
14	КПТШ-8	«КЖ»
15	КПТШ-9	«Ж»

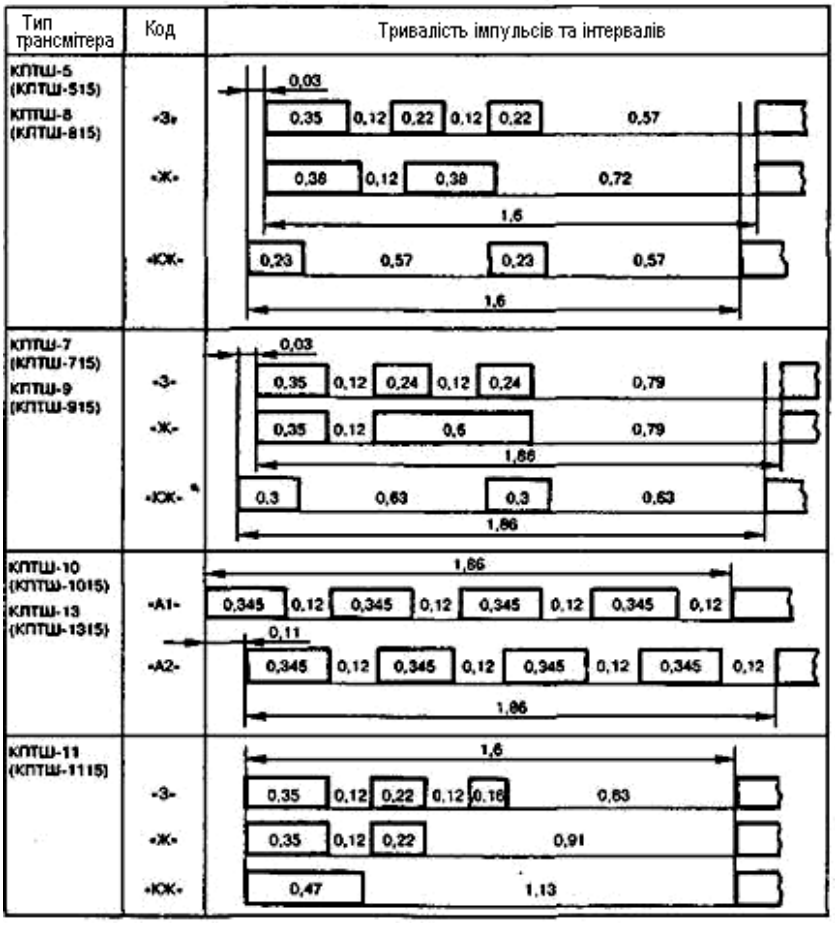


Рисунок 56 – Часові характеристики трансмітерів

ЛАБОРАТОРНА РОБОТА 7
Побудова дешифраторів кодових сигналів на базі DFB-модулів для моделювання кодового автоблокування

Мета роботи: навчитись створювати FBD-модулі, набуття навичок автоматизації технологічного процесу зі складанням програм мовою FBD з елементами візуалізації сигнальних точок кодового автоматичного блокування.

Обладнання та ПЗ: ПЕОМ з необхідними системними вимогами для роботи ПЗ, MS Windows, Internet Explorer, Open Office Writer, програма Unity Pro.

Послідовність виконання роботи

Як і в попередній роботі, компонуємо всі необхідні модулі та активуємо їх. Наступний крок – відкриваємо вікно Derived FB Types, щоб створити новий FBD-тип (KAB3_5 для тризначного автоблокування з КПТШ 515 або KAB4_5 для чотиризначного автоблокування з КПТШ 515), на основі якого (рисунок 56) потім будуть створюватися екземпляри блоків, таких як N3, N5, N7, з номерами сигнальних точок у непарному напрямку. Крім унікальної назви типу (за правилами створення ідентифікаторів), необхідно створити локальні вхідні (I – імпульсний вхід рейкового кола, KO – вхід контролю лампи червоного вогню та GO – жовтого вогню) і вихідні змінні (LG – вихід на лампу жовтого, LK – червоного та LZ – зеленого кольору і T – вихід трансмітера наступного рейкового кола) у FBD-модулі та логіку «перетворення» вхідних змінних у вихідні та записати параметри, як на рисунках 51, 52 та рисунку 57.

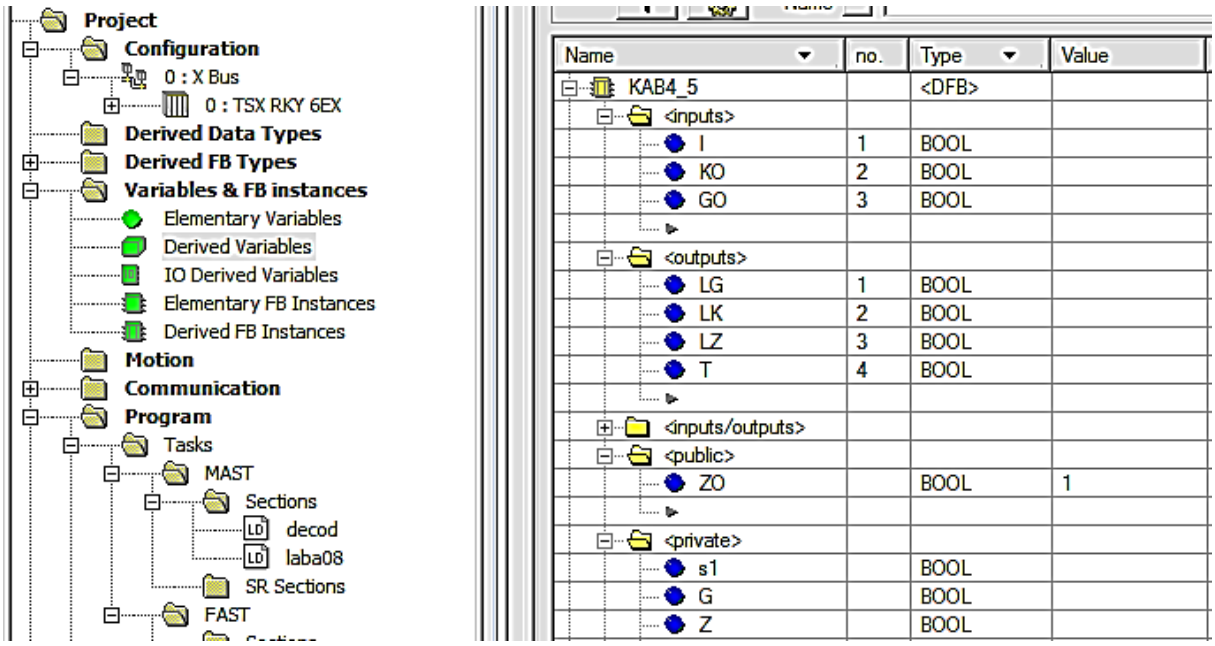


Рисунок 57 – Розроблення DFB-типу «KAB4_5» із зазначенням частини змінних і визначенням їхніх типів

Потім компонуємо схему (рисунок 58) з уже відомих компонентів, натискаючи на об’єкт Sec_KAB4_5 два рази, як у вікні на рисунку 30.

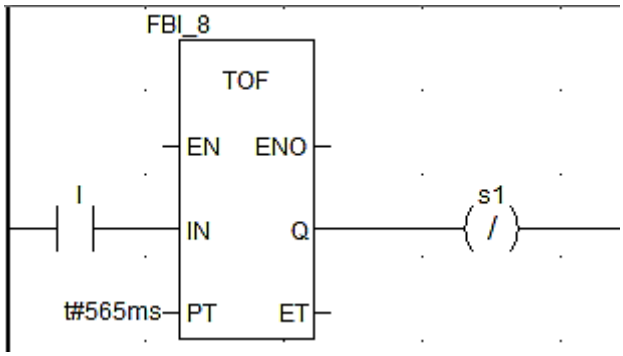
Простий аналіз дає змогу створити схему вмикання котушки на вимикання S1, яка фіксуватиме наявність великої кодової паузи будь-якого з режимів шифраторів КПТ (рисунок 58,а).

Наступним кроком стало можливим створити схему підрахунку імпульсів у кодовій послідовності із застосуванням блока CTU. Котушки «G» і «Z» будуть, за аналогією до класичного АБ, фіксувати наявність відповідно першого (рисунок 59, а) і другого (рисунок 59, б) імпульсів кодової послідовності.

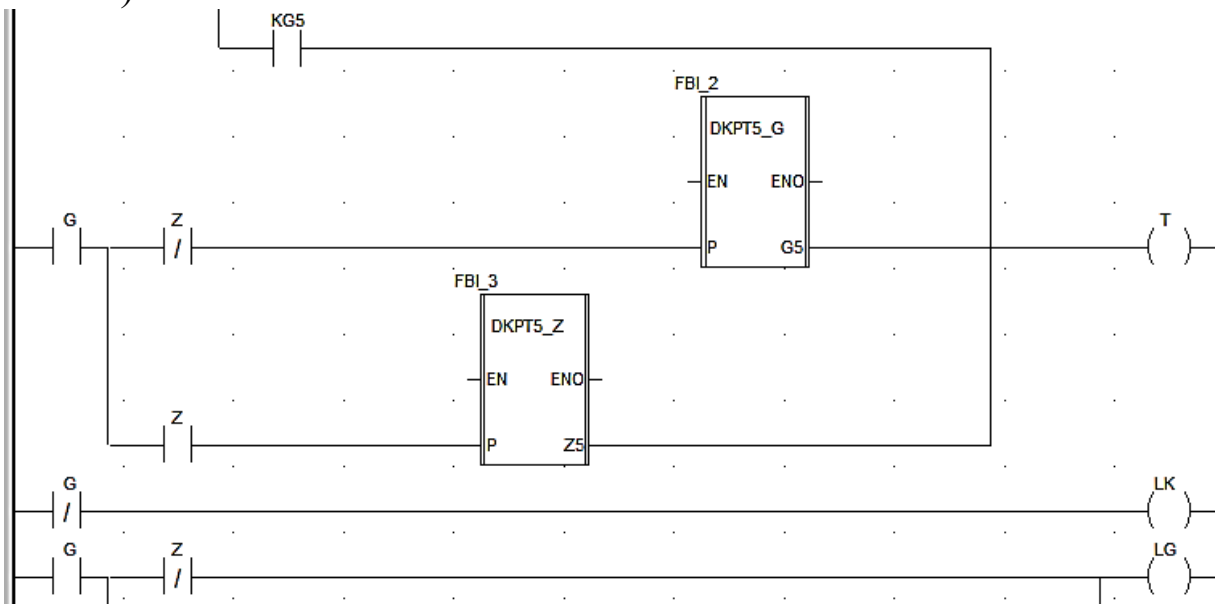
Створюємо операторський екран Screen_laba7 і зображуємо світлофор з розташуванням показань тризначного/чотиризначного автоблокування, аналогічно до рисунка 32, і задіємо кожен з кружків для трьох кольорів показань, де будуть наступні параметри з можливістю задавати в полі Variable локалізовані (рисунок 33) і нелокалізовані змінні.

Після цього студент самостійно будує функціональну модель сигнальної точки для тризначного або чотиризначного автоблокування за схемами двоколійного кодового автоблокування для ділянок з одностороннім рухом поїздів.

а)



б)



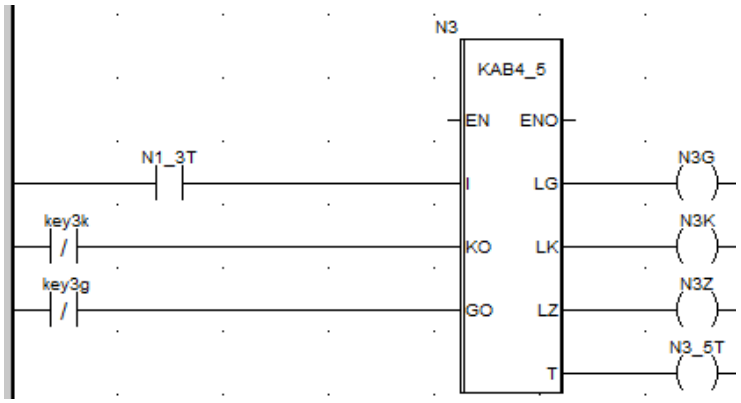
а – схема фіксації великої кодової паузи КПТ;
б – схема включення виводу Т секції типу Sec_KAB3_5

Рисунок 58 – Фрагмент схеми DFB блока KAB3_5

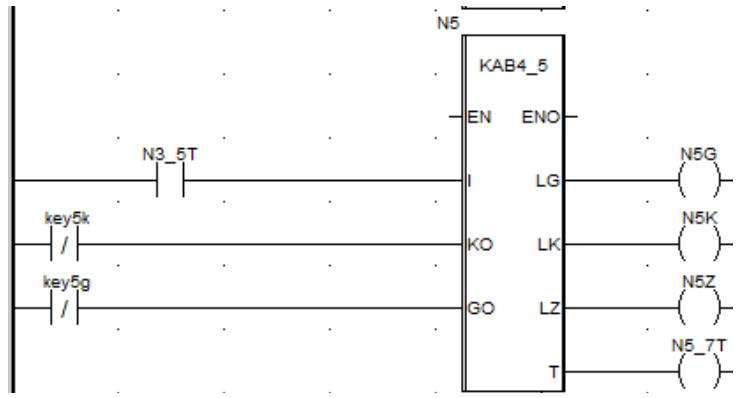
Знову компілюємо програму модуля і перевіряємо роботу всіх елементів проекту.

Створивши екземпляри N3, N5, N7 на базі функціонального блока користувача (DFB-типу з ім’ям KAB3_5), підключаємо до їхніх виходів «реле» з локалізованими змінними (рисунок 59). Змінна N3_5T символізує вихідний елемент «Т» блока N3 для відправки кодів до рейкового кола 5П.

а)



б)



в)

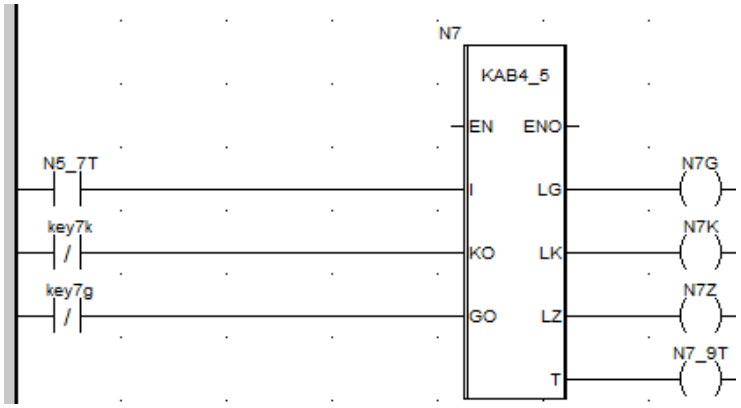


Рисунок 59 – Схеми підключення екземплярів N3 (а), N5 (б), N7 (в)

Блоки будуємо один під одним.

Запускаємо програму, як наведено на рисунках 5, 6. У режимі емуляції роботи ПЛК відображаються червоним кольором розімкнені кола, а зеленим кольором – замкнуті (рисунок 60).

а) б)

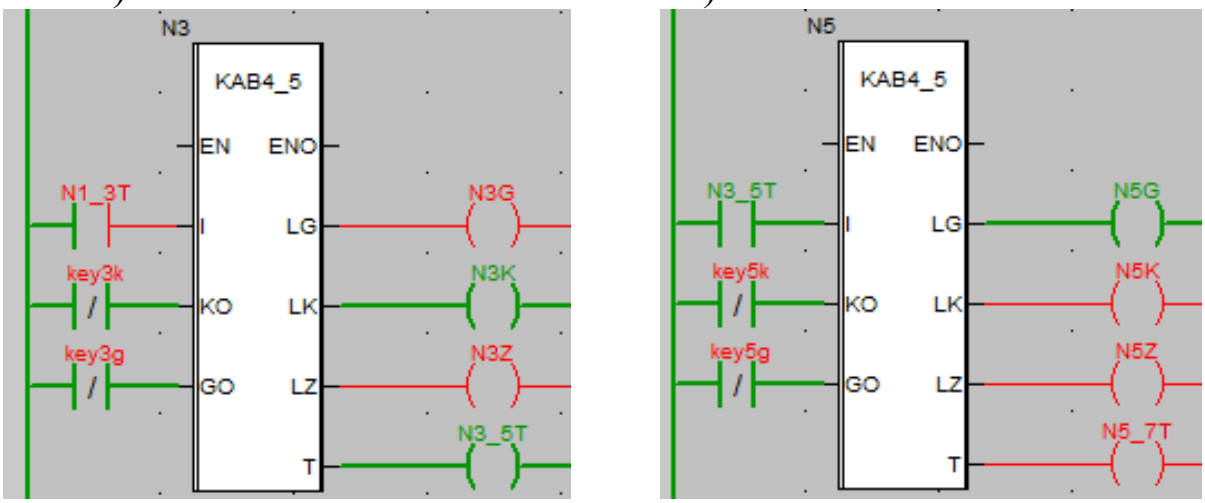


Рисунок 59 – Перевірка роботи блоків АБ N3 (а) і N5 (б)

Студенти, що претендують на рейтинг більше 70 балів, мають реалізувати до кінця лабораторної роботи домашнє завдання рівня від 2-го пункту до 4-го. Для створення чотиризначного кодового автоблокування в прикладах створеного функціонального блока користувача (DFB-типу з ім’ям KAB4_5) цифра 5 ставиться при генерації кодів з інтервалами КПТШ 5/515/8/815. При встановленні в сигнальній установці, для генерації кодів, КПТШ 7/715/9/915 доцільно ставити у відповідному функціональному блоці користувача DFB-типу з ім’ям KAB4_7. Типи DFB блоків KAB4_5 та KAB4_7 відрізняються тільки зображеними на рисунку 58 модулями DKPT5_Z та DKPT5_G параметром вмикання та вимикання реле KG5. Таким чином, у DFB блоці KAB4_7 мають стояти модулі DKPT7_Z та DKPT7_G і змінені параметри вмикання та вимикання реле KG7 (аналог KG5).

Локалізовані змінні «key3k» та «key3g» створені для змін інформації про перегорання червоної та жовтої ламп відповідно на третій, а «key5k» та «key5g» – на п’ятій сигнальних точках. Щоб перевірити, чи правильно працює програма, треба встановити в «1» на локалізованій змінній «key3k» ПКМ Force Value 1 біт (рисунок 61).

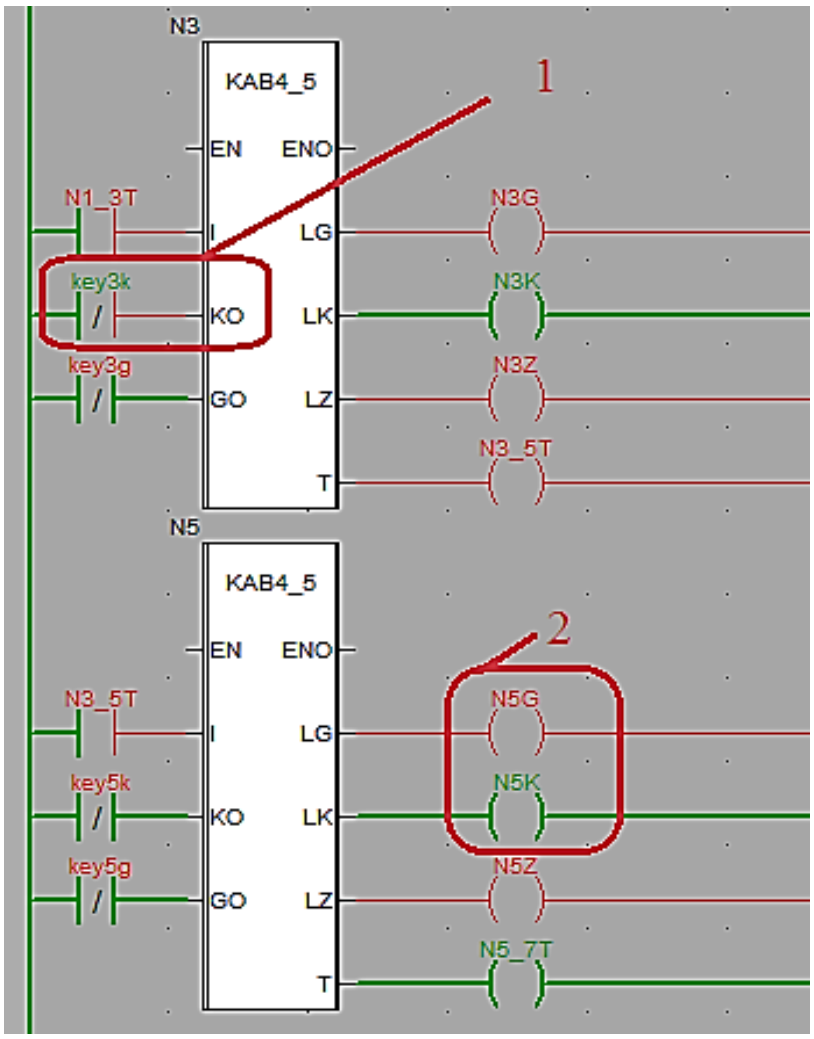


Рисунок 61 – Зміна стану входу «KO» блока N3 та контроль виходів «LG» і «LK» блока N5

При виконанні домашнього завдання найвищого рівня складності (пункт 4) буде доцільно переглянути підказки, зображені на рисунках 62-64.

Студентам, які справились із домашніми завданнями від 1-го до 4-го пункту, індивідуальну роботу може надати викладач.

План виконання роботи:

1 Знайти додаткові джерела інформації.

2 Вивчити теоретичний і практичний матеріал, необхідний для виконання роботи.

3 Відповісти на контрольні питання та виконати домашнє завдання.

4 Спланувати лабораторну роботу згідно з домашнім завданням відповідного рівня.

Контрольні питання:

1 Як створити новий блок?

2 Як працює прохідна та передвхідна сигнальна точка кодового автоблокування?

3 Як підключити три лампи світлофора до блока КАВ3_5?

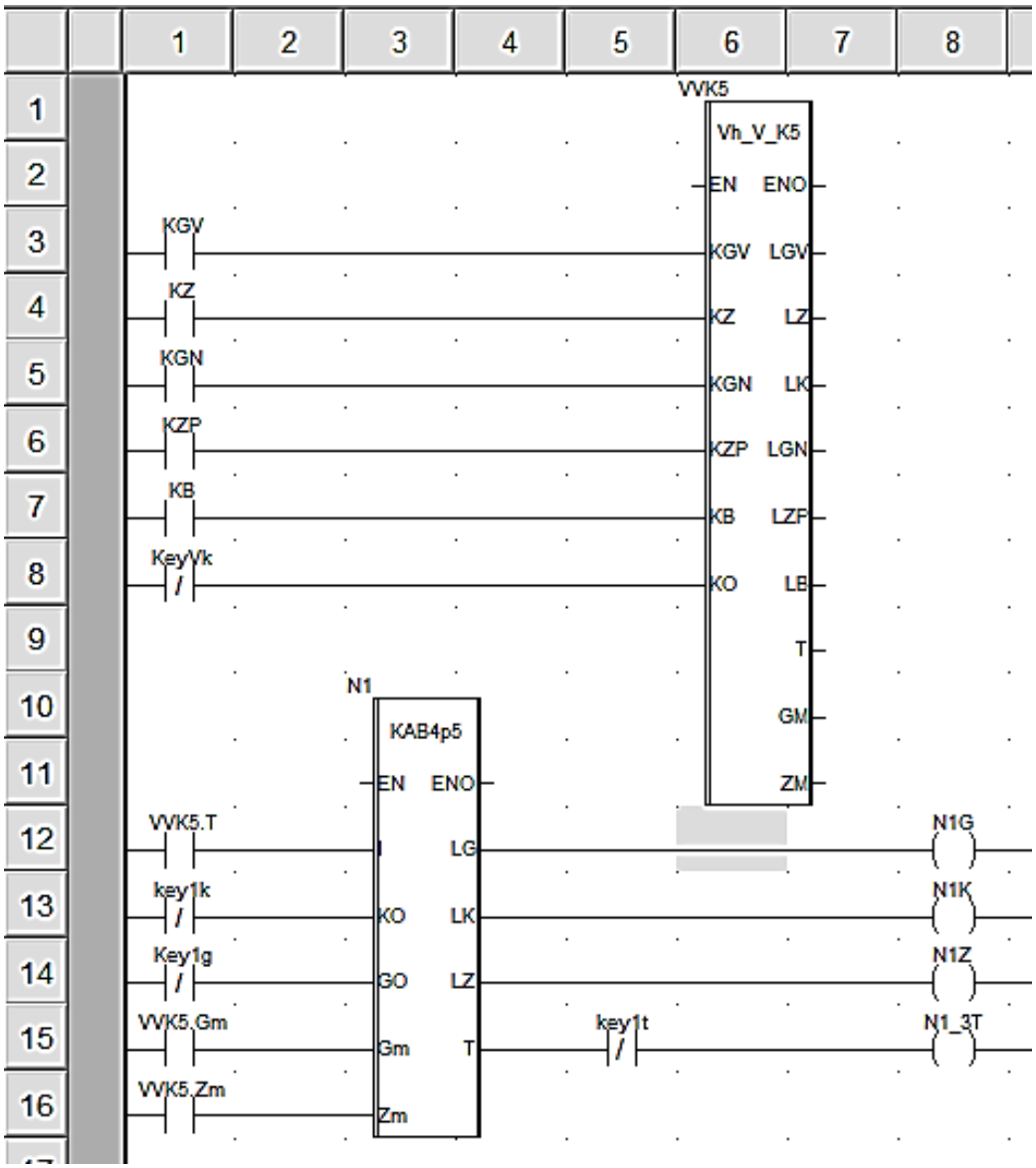


Рисунок 62 – Схема зв’язку екземпляра блока «VVK5» DFB-типу Vh_V_K5 з екземпляром блока N1 DFB-типу KAB4p5 (КАБ 4 – значна, р – передвхідна сигнальна точка з КПТШ 5)

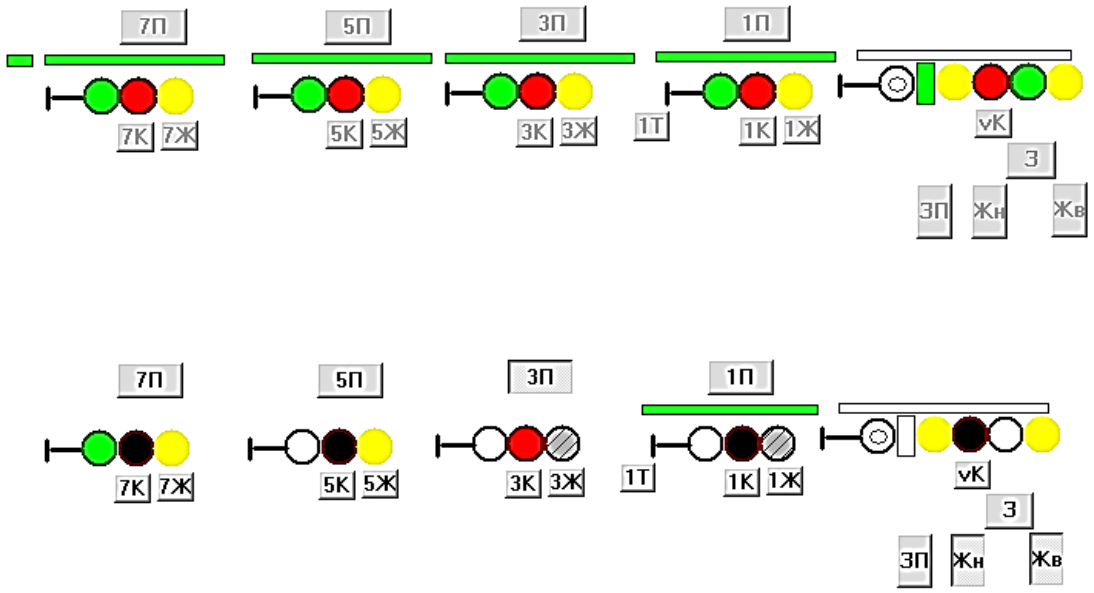
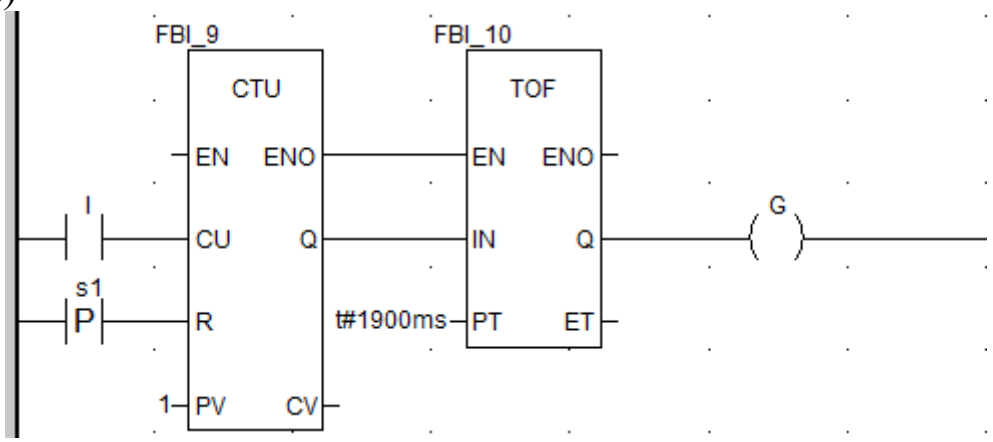


Рисунок 63 – Схема перегону з імітатором вхідного світлофора

а)



б)

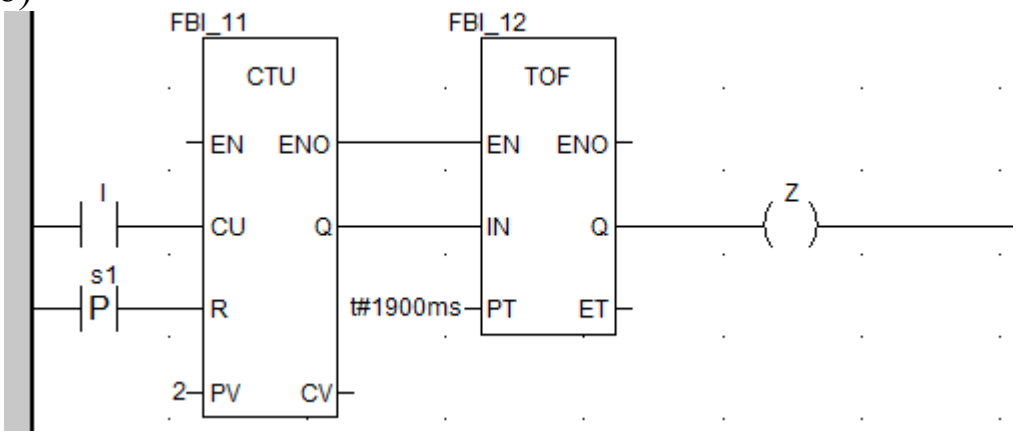
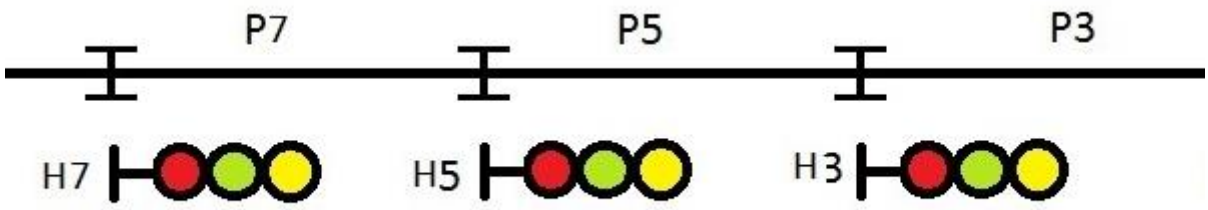


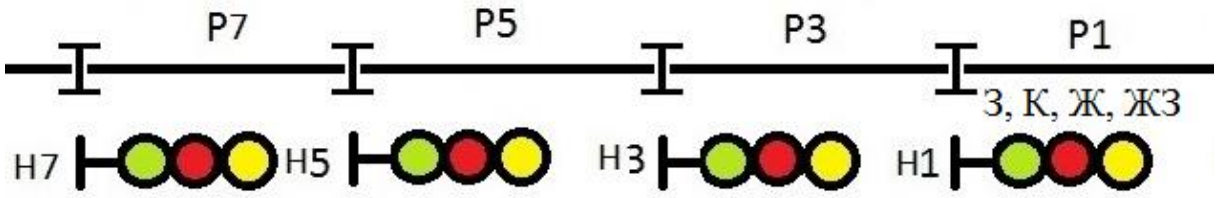
Рисунок 64 – Схеми фіксації першого (а) і другого (б) імпульсів будь-якого коду КПТ

ДОМАШНЄ ЗАВДАННЯ ЗІ СТВОРЕННЯМ ВЛАСНИХ DFB ДЛЯ УДОСКОНАЛЕННЯ НАВИЧОК СИНТЕЗУ СИСТЕМ ЗАЛІЗНИЧНОЇ АВТОМАТИКИ

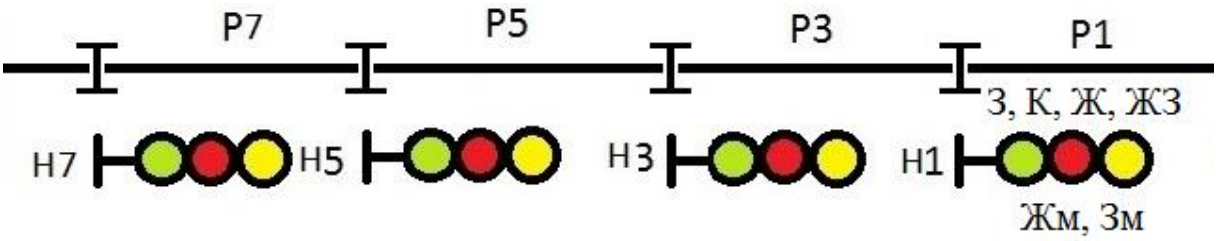
1 Схематично зобразити у вікні Screen_laba7_dzE ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і трьох світлофорів Н7, Н5, Н3, з'єднавши їх у єдину технологічну модель роботи тризначного кодового автоблокування за схемою перегону, наведеною нижче.



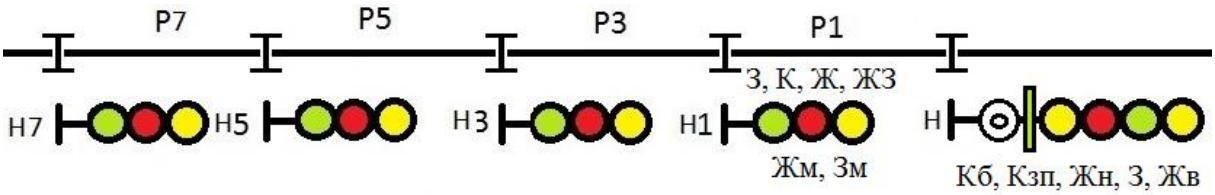
2 Схематично зобразити у вікні Screen_laba7_dzD ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і трьох світлофорів Н7, Н5, Н3, з'єднавши їх у єдину технологічну модель роботи чотиризначного кодового автоблокування за схемою перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи.



3 Схематично зобразити у вікні Screen_laba7_dzC ділянку колії з трьох рейкових кіл Р7, Р5, Р3 і чотирьох світлофорів Н7, Н5, Н3, Н1, з'єднавши їх у єдину технологічну модель роботи чотиризначного кодового автоблокування за схемою одностороннього перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи та вимиканням зеленого вогню при показанні «жовтий з зеленим» у випадку перегорання лампи жовтого вогню з динамічною індикацією кодування та інтерактивним введенням зайнятості рейкових кіл Р7, Р5, Р3 і роботи світлофора Н1 (шість показань).



4 Схематично зобразити у вікні Screen_laba7_dzA ділянку колії з чотирьох рейкових кіл Р7, Р5, Р3, Р1 і п’яти світлофорів Н7, Н5, Н3, Н1, Н, з’єднавши їх у єдину технологічну модель роботи за схемою чотиризначного кодового автоблокування перегону, наведеною нижче, з перенесенням червоного показання у випадку перегорання червоної лампи та вимиканням зеленого вогню при показанні «жовтий з зеленим» у випадку перегорання лампи жовтого вогню з роботою світлофора Н1 в режимі передвхідної сигнальної установки з динамічною індикацією кодових сигналів й інтерактивним введенням зайнятості рейкових кіл Р7, Р5, Р3, Р1 і роботи світлофора Н (шість показань – К, Жв, З, Жн+Жв, Жн+Жв+ЗП, К+Бмиг).



СПИСОК ЛІТЕРАТУРИ

1 Бутенко, В. М. Адресація та захист інформації в мережі RailWayNet [Текст] / В. М. Бутенко // Информационно-управляющие системы на железнодорожном транспорте. – 1997. – № 3. – С. 24 – 26

2 Системы интервального регулирования движения поездов на перегонах [Текст] : учеб. пособие / А. Б. Бойник, С. В. Кошевой, С. В. Панченко, В. А. Сотник. – Харьков : УкрГАЖТ, 2005. – 256 с.

3 Бутенко, В. М. Методичні вказівки до виконання лабораторних робіт з дисципліни «Автоматизація технологічних процесів» Schnider Electric [Текст] / В. М. Бутенко. – Харків : УкрДАЗТ, 2015. – 54 с.

4 Бутенко, В. М. Методичні вказівки до виконання лабораторних робіт з дисципліни «Технічні засоби автоматизації» [Текст] / В. М. Бутенко. – Харків : УкрДУЗТ, 2016. – Ч. 1. – 40 с.

5 Бутенко, В. М. Інтернет на залізницях України [Текст] / В. М. Бутенко // Информационно-управляющие системы на железнодорожном транспорте. – 1997. – № 1. – С. 47 – 49.

6 Кількісний аналіз показників надійності систем автоматики з використанням моделювання дерев небезпечних відмов [Текст] / В. М. Бутенко, Д. О. Зубрицький, С. В. Сіроштан, Є. С. Строєв // Зб. наук. праць. – Харків : УкрДАЗТ. – 2008. – Вип. 92. – С. 133 – 138.

7 Ранжирование опасностей с нечеткими зонами межранговых переходов [Текст] / В. М. Бутенко, О. В. Головко, В. І. Мойсеєнко // Зб. наук. праць. – Донецьк : ДонІЗТ. – 2008. – Вип. 14. – С. 64 – 73.

8 Бутенко, В. М. Удосконалення технічної експлуатації та розробка технічних засобів автоматики [Текст] /В. М. Бутенко, В. С. Коновалов // Інформаційно-керуючі системи на залізничному транспорті. – 2012. – № 6 (97). – С. 58 – 62.

9 Бутенко, В. М. Методичні вказівки до виконання практичних занять та курсової роботи з дисципліни «Методологія менеджменту якості» для спеціалістів та магістрів спеціальності «Якість, стандартизація та сертифікація» [Текст] /В. М. Бутенко, О. П. Земляний. – Харків : УкрДАЗТ, 2008. – Ч. 1. – 38 с.

10 Бутенко, В. М. Методичний посібник до лабораторних робіт з дисципліни «Математичні методи та моделі розрахунку на ЕОМ» [Текст] /В. М. Бутенко, О. Б. Болотов, В. В. Шумеєв. – Харків : УкрДАЗТ, 2006. – Ч. 1. – 28 с. (№ 836)

11 Бутенко, В. М. Інформаційні системи на залізничному транспорті [Текст] : конспект лекцій / В. М. Бутенко, С. Є. Бантюков, В. Г. Пчолін. – Харків : УкрДАЗТ, 2008. – Ч. 1. – 28 с.

12 Основи програмування мовами високого рівня [Текст] : навч. посібник / В. М. Бутенко, В. С. Меркулов, О. В. Чаленко, О. В. Казанко. – Харків : УкрДАЗТ, 2009. – 206 с.

13 Виноградова, В. Ю. Перегонные системы автоматики [Текст] / В. Ю. Виноградова. – М. : Маршрут, 2005. – 235 с.

14 Математичні методи та моделі в розрахунках на ЕОМ [Текст] : навч. посібник / М. І. Данько, В. С. Меркулов, В. О. Гончаров [та ін.]; за заг. ред. М. І. Данька. — Харків : УкрДАЗТ, 2008. – 172 с.

15 Завдання і методичні вказівки до розрахунково-графічної та контрольної робіт з дисциплін «Програмування» та «Іформатика» для студентів факультету АТЗ [Текст] / В. М. Бутенко, О. В. Головко, М. О. Колісник, С. О. Бантюкова. – Харків : УкрДУЗТ, 2016. – 74 с.

16 Путевая блокировка и авторегулировка [Текст] / Н. Ф. Котляренко [и др.]. – М. : Транспорт, 1983 – 368 с.

17 Математичне моделювання в розподілених інформаційно-керуючих системах залізничного транспорту [Текст] : монографія / С. В. Лістровий, С. В. Панченко, В. І. Мойсеєнко, В. М. Бутенко. – Харків : ФОП Бровін О. В., 2017. – 220 с.

18 Меркулов, В. С. Основи алгоритмізації базових обчислювальних процесів [Текст] : навч. посібник / В. С. Меркулов, В. М. Бутенко. — Харків : УкрДАЗТ, 2008. – 163 с.

19 Меркулов, В. С. Методичні вказівки з варіантами завдань для виконання контрольних робіт з дисципліни «Обчислювальна техніка, програмування, моделювання систем» [Текст] / В. С. Меркулов, В.М. Бутенко, О.В. Чаленко. – Харків : УкрДАЗТ, 2013. – Ч. 2. – 51 с.

20 Меркулов, В. С. Методичні вказівки з варіантами завдань для виконання контрольних робіт з дисципліни «Обчислювальна

техніка, програмування, моделювання систем» [Текст] / В. С. Меркулов, В. М. Бутенко, О. В. Чаленко. – Харків : УкрДАЗТ, 2013. – Ч. 3. – 51 с.

21 Бутенко, В. М. Особливості оцінювання систем залізничної автоматики [Текст] / В. М. Бутенко, С. Г. Чуб // Зб. наук. праць. – Донецьк : ДонІЗТ, 2005. – Вип. 3. – С. 32 – 39.

22 Пупена, О. М. Програмування промислових контролерів у середовищі Unity Pro [Текст] : навч. посібник / О. М. Пупена, І. В. Ельперін. – Київ : Ліра-К, 2017. – 376 с.

23 Бутенко, В. М. Перспективи розвитку досліджень якості, сертифікації та стандартизації на залізничному транспорті [Текст] / В. М. Бутенко // Зб. наук. праць. – Донецьк : ДонІЗТ, 2006. – Вип. 8. – С. 53-56.

24 Удосконалення організаційно-управлінської роботи на підприємствах залізничного транспорту в сучасних умовах [Текст] : навч. посібник / Г. Ф. Арбузов, В. М. Бутенко, О. Г. Дайнека, А. О. Каграманян [та ін.]; заг. ред. М. І. Данька. – Харків : УкрДАЗТ, 2007. – 178 с.

25 Бутенко, В. М. Якість інформаційно-вимірювальних систем на залізничному транспорті України [Текст] / В. М. Бутенко // Зб. наук. праць. – Харків : УкрДАЗТ, 2008. – Вип. 99. – С. 151 – 155.

26 Development of method of definition maximum clique in a non-oriented graph [Text] / S. V. Listrovoy, V. M. Butenko, V. O. Bryksin, O. V. Golovko // EasternEuropean Journal of Enterprise Technologies. – 2017. – Vol. 5, № 4 (89). – P. 12 – 17. EID: 2-s2.0-85032585697 DOI: 10.15587/1729-4061.2017.111056.

27 IEC 61131-3:2013 Programmable controllers. – Part 3: Programming languages.

28 IEC 60617-12:1997 Withdrawn Graphical symbols for diagrams – Part 12: Binary logic elements.

29 https://uk.wikipedia.org/wiki/Логічний_вентиль

30 https://uk.wikipedia.org/wiki/Булева алгебра з двома елементами

31 Formulation of the Problem of Maximum Clique Determination in Non-Oriented Graphs [Text] / S. V. Listrovoy, O. V. Golovko, V. M. Butenko, M. V. Ushakov // International Journal of Engineering & Technology Vol 7 No 4.3 (2018): Special Issue 3 PP. 293 – 297.

32 Signal flow graph models and alternative gain formula for multiprobe microwave multimeter [Text] / O. B. Zaichenko, V. M. Butenko, M. A. Miroshnyk // Інформаційно-керуючі системи на залізничному транспорті. – 2016. – № 12. – С. 12 – 17.