МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра комп’ютеризованих систем автоматики

ДОСЛІДЖЕННЯ РОБОТИ ТАЙМЕРІВ ТА СТРУКТУРИ ПАМ’ЯТІ КОНТРОЛЕРІВ У СИСТЕМІ АВТОМАТИЗАЦІЇ SIMATIC STEP 7

Методичні вказівки до лабораторної роботи 2
з курсу
«SCADA-СИСТЕМИ ТА ІНТЕРНЕТ РЕЧЕЙ»

Львів 2021

Мета роботи: освоїти принципи формування образу процесу, звертання до даних у різних областях пам’яті та отримати навики роботи з функціями пам’яті, такими як: блочні елементи присвоєння, блочні елементи встановлення та скидання, блочним елементом MOVE для контролерів S7 в системі автоматизації SIMATIC.

1. Формування образу процесу

При звертанні в програмі користувача до входів (I) і виходів (Q) опитуються не стани на цифрових сигнальних модулях, а здійснюється звернення до області в системній пам’яті ЦПУ. Цю область пам’яті називають *образом процесу*.

Образ процесу ділиться на дві частини: таблиця образу процесу входів (PII) і таблиця образу процесу виходів (PIQ).

Рис.1. ілюструє формування образу процесу.

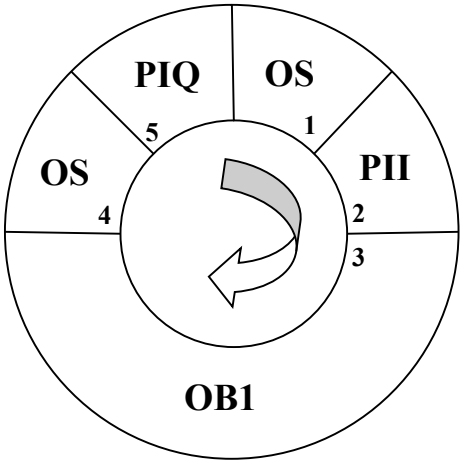


Рис. 1. Формування образу процесу під час виконання ЦПУ програми користувача

Одним із завдань операційної системи контролера є: зчитати значення станів входів з модулів розширення та записати ці значення у таблицю входів образу процесу (PII). Після завершення цього кроку виконується програма користувача (OB1). Під час виконання програми користувача формується таблиця виходів образу процесу (PIQ). Завершується цикл виконання програми записом значень з PIQ на “реальні” виходи модулів.

Однією із переваг використання образу процесу над прямим зверненням до входів–виходів модулів є те, що під час виконання OB1 програма працює із “незмінним” образом PII в межах одного циклу виконання. У разі зміни сигналу на одному або декількох входах модулів під час виконання програми користувача, значення цього входу або входів залишається незмінним аж до поновлення образу процесу на початку наступного циклу.

Ще однією перевагою використання образу процесу є і те, що доступ до образу процесу потребує значно менше часу, ніж пряме зчитування значень з входів модулів.

2. Звертання до даних в різних областях пам'яті

Контролери S7 зберігають дані в різних областях пам'яті, які мають однозначні адреси. Завдяки цьому програма має прямий доступ до даних.

Для звернення до певного біту пам'яті необхідно вказати його адреси, який складається із ідентифікатора області пам'яті, номеру байта та номеру біта.

Область пам'яті входів (I).

На початку кожного циклу процесор опитує фізичні входи і записує отримані дані в область входів образу процесу. Звертання до цієї області записується так, як подано у табл. 1.

Таблиця 1

Тип	Форма запису	Приклад
Біт	**I**[номер байту]**.**[номер біту]	**I**5.2
Байт	**I**[довжина (**B**)][початковий адрес байту]	**IB**1
Слово	**I**[довжина (**W**)][початковий адрес байту]	**IW**6
Подвійне слово	**I**[довжина (**D**)][початковий адрес байту]	**ID**3

Область пам'яті виходів (Q).

У кінці кожного циклу ОС копіює значення, що зберігаються в області виходів образу процесу, в фізичні виходи. Звертання до цієї області записується так, як подано у табл. 2.

Таблиця 2

Тип	Форма запису	Приклад
Біт	**Q**[номер байту]**.**[номер біту]	**Q**5.2
Байт	**Q**[довжина (**B**)][початковий адрес байту]	**QB**1
Слово	**Q**[довжина (**W**)][початковий адрес байту]	**QW**6
Подвійне слово	**Q**[довжина (**D**)][початковий адрес байту]	**QD**3

Область пам'яті змінних (V).

Пам'ять змінних можна використовувати для зберігання проміжних результатів під час виконання програми. Звертання до цієї області записується так, як подано у табл. 3.

Таблиця 3

Тип	Форма запису	Приклад
Біт	**V**[номер байту]**.**[номер біту]	**V7**.0
Байт	**V**[довжина (**B**)][початковий адрес байту]	**VB**152
Слово	**V**[довжина (**W**)][початковий адрес байту]	**VW**100
Подвійне слово	**V**[довжина (**D**)][початковий адрес байту]	**VD**21

Область бітової пам'яті (M).

Пам'ять маркерів (область бітової пам'яті) використовується для зберігання бінарних сигнальних станів та проміжних результатів. Звертання до цієї області записується так, як подано у табл. 4.

Таблиця 4

Тип	Форма запису	Приклад
Біт	**M**[номер байту]**.**[номер біту]	**M**7.0
Байт	**M**[довжина (**B**)][початковий адрес байту]	**MB**2
Слово	**M**[довжина (**W**)][початковий адрес байту]	**MW**10
Подвійне слово	**M**[довжина (**D**)][початковий адрес байту]	**MD**2

Кількість бітів в маркерній області пам'яті залежить від моделі ЦПУ.

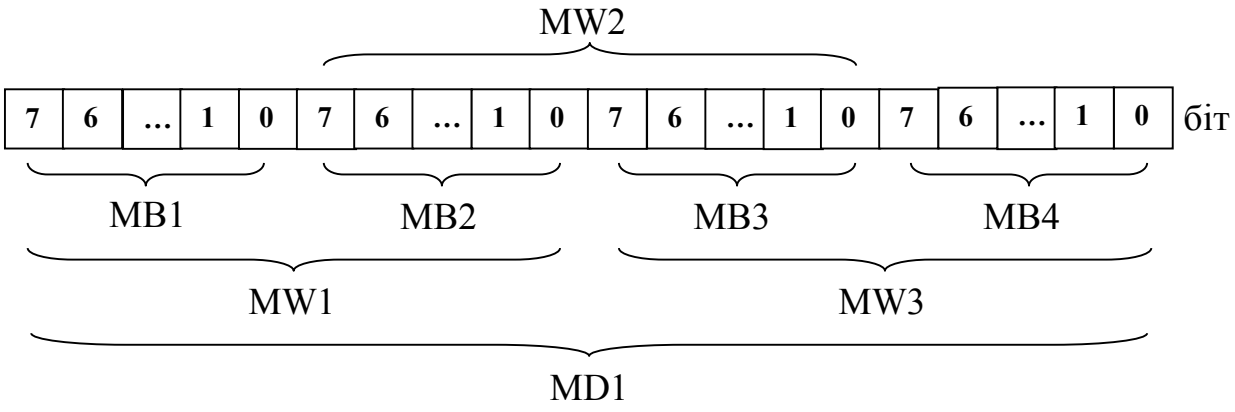


Рис. 2. Будова маркерної області пам'яті

Область пам'яті таймерів (T).
Звертання до цієї області записується так, як подано у табл. 5.

Таблиця 5

Форма запису	Приклад
T[номер елементу]	**T**1

Область пам'яті лічильників (C).
Звертання до цієї області записується так, як подано у табл. 6.

Таблиця 6

Форма запису	Приклад
C[номер елементу]	**C**5

3. Функції для роботи з пам'яттю

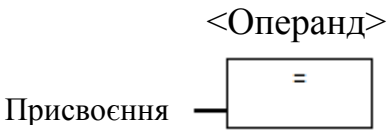
Доступні наступні функції для роботи з пам'яттю:
- елемент присвоєння;
- елемент встановлення та скидання;
- елемент "SR" та "RS";
- елемент збереження значення RLO "Midline Output", як проміжний буфер;
- функції оцінки фронту сигналу.

Розглянемо більш детально кожен із цих елементів.

Елемент присвоєння.

Елемент присвоєння використовується для завершення логічної операції. Цей елемент присвоює значення RLO “Операнду”, який введений над ним.

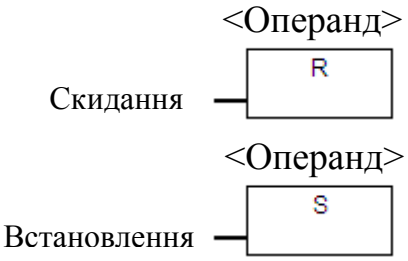
Якщо значення RLO на вході елементу присвоєння рівне “1”, то “Операнд” встановлюється в “1”, якщо ж RLO рівне “0” – “Операнд” скидається в “0”.



Примітка: під словом “Операнд” слід розуміти адресу області пам’яті, описану в попередньому розділі.

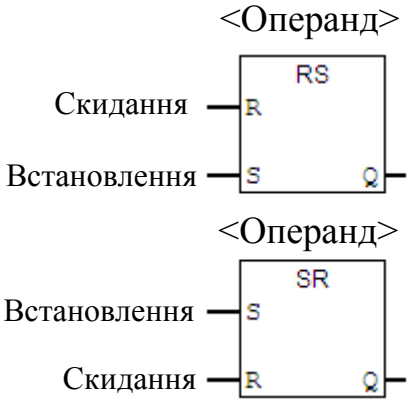
Елемент встановлення та скидання.

Елементи встановлення та скидання також використовуються для завершення логічної операції. Вони виконуються тільки тоді, коли значення RLO на вході рівне “1”. Якщо значення RLO на вході блочного елементу встановлення (скидання) рівне “1”, “Операнд” над блочним елементом встановлюється (скидається) в “1” (“0”). Якщо значення RLO на вході рівне “0” – значення “Операнда” залишається без змін.



Елемент “SR” та “RS”.

Елемент “SR” та “RS” об’єднують в собі елементи скидання та встановлення. Вхід S відповідає елементові встановлення, а вхід R – елементові скидання. “Операнд” встановлюється чи скидається згідно табл. 7.



Примітка: в процесі виконання програми користувача, при одночасному стані RLO рівному “1” на обох входах, ЦПУ для RS-тригера спершу скине “Операнд” в “0”, а потім одразу встановить в “1”. Для SR-тригера навпаки, ЦПУ спочатку встановить “Операнд” в “1”, а вже потім скине в “0”. У випадку коли “Операнд” є виходом, ця зміна жодним чином миттєво не відобразиться на фізичному виході модуля розширення аж до кінця циклу, тобто до поновлення таблиці виходів PIQ. Тригери SR та RS можуть бути використані для завершення логічної операції.

Встановлення “Операнда” та виходу у тригерах: **Таблиця 7.**

а) SR–тригер

S	R	Операнд	Q
0	0	-	-
0	1	0	0
1	0	1	1
1	1	0	0

б) RS–тригер

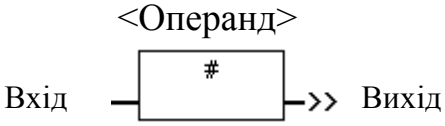
R	S	Операнд	Q
0	0	-	-
0	1	1	1
1	0	0	0
1	1	1	1

Під позначенням “-” слід розуміти, що стан “Операнду” та виходу Q залишається без змін.

Відмінністю між тригерами SR та RS є те, що при одночасному стані RLO рівному “1” на обох входах S та R тригер SR скине “Операнд” в “0”, а тригер RS, відповідно, встановить “Операнд” в “1”. SR- та RS-тригери можуть бути використані для завершення логічної операції.

Елемент присвоєння значення RLO.

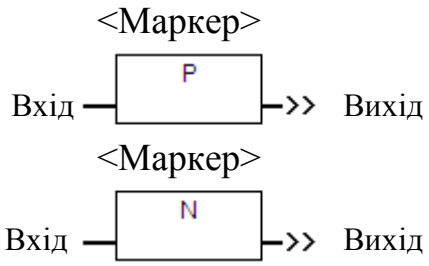
Елемент присвоєння значення RLO використовують для збереження значення RLO без зміни цього значення.



“Операнд” над блочним елементом може бути опитаний в іншій частині програми. В одному сегменті програми можливо використовувати декілька елементів присвоєння значення RLO, проте він не може використовуватись для завершення логічної операції.

Елементи оцінки фронту сигналу.

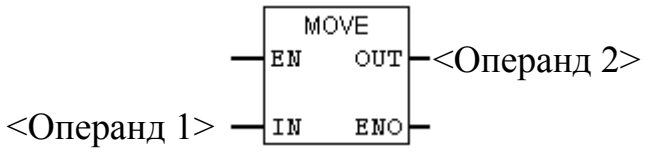
Функції оцінки зміни фронту сигналу використовуються для виявлення змін в стані сигналу. Зміна сигналу свого стану з “0” на “1” має назву “позитивний фронт”, а з “1” на “0” – “негативний фронт”. Принцип дії елементу оцінки фронту сигналу полягає у порівнянні двох значень RLO (наприклад, результату перевірки стану сигналу на вході та значення RLO, збережене у маркері над елементом оцінки фронту). Якщо ці значення різні – це означає, що відбулась зміна фронту сигналу.



Для виявлення “позитивного” фронту сигналу використовують елемент з позначкою “P”, а для “негативного” – елемент з позначенням “N”. Значення попереднього RLO зберігається у маркері фронту. Маркер над блочним елементом оцінки фронту може бути використаний у інших сегментах програми. Елемент оцінки фронту сигналу не може бути використаний для завершення логічної операції.

4. Функція передавання MOVE

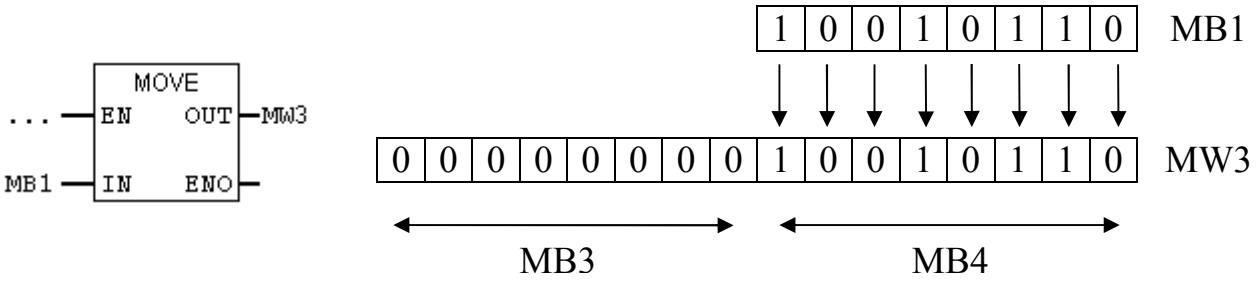
Блочний елемент MOVE використовується для передавання даних з однієї області пам’яті в іншу. Передавання відбувається через внутрішній регістр ЦПУ, тимчасову пам’ять, яка називається *акумулятором 1* (Accumulator 1). Функція переміщення в акумулятор називається “завантаження”, а функція переміщення із акумулятора – “передачею”. Блочний елемент MOVE містить ці дві функції. Вхід “IN” використовується для завантаження “Операнду 1” в акумулятор, а вихід “OUT” – для передавання з акумулятора в “Операнд 2”.



Додатково елемент MOVE має вхід дозволу роботи EN і вихід дозволу ENO. У випадку, коли значення RLO на вході дозволу EN рівне “0”, переміщення даних із

входу IN на вихід OUT не відбудеться. В іншому випадку, “Операнд 1” з входу IN переміститься (присвоїться) в “Операнд 2” на виході OUT. Якщо вхід EN залишити незадіяним, блочний елемент MOVE буде постійно передавати дані із “Операнда 1” в “Операнд 2”, розташований на виході.

Зауваження: входу або виходу блочного елементу MOVE **не можна** присвоїти змінну типу BOOL (логічна змінна). Якщо значення “Операнда” на вході менше, ніж на виході, то “Операнд” із входу переміститься на вихід з заповненням бітів, починаючи справа (праве вирівнювання). Всі решта біти заповняться нулями. Якщо вихідний “Операнд” менший за розміром від вхідного, тоді передається лише права частина, такого ж самого розміру, як вихідний “Операнд”.



5. Таймер

Таймери (Timer) дають можливість програмно реалізувати послідовності синхронізації, такі як інтервали очікування і спостереження, вимірювання інтервалів або генерування імпульсів.

В програмному пакеті Step7 існують п’ять видів таймерів:

- Імпульсний таймер (Pulse timer, S_PULSE);
- Розширений імпульсний таймер (Extended pulse timer, S_PEXT);
- Таймер затримки включення (On-delay timer, S_ODT);
- Таймер затримки включення з запам’ятовуванням (Retentive on-delay timer, S_ODTS);
- Таймер затримки виключення (Off-delay timer, S_OFFDT);

Таймер програмується як окремий блочний елемент або з допомогою окремих програмних елементів (рис. 3). При запуску таймера визначається тип таймера і час його роботи. Над зображенням таймера розташоване поле “Операнд таймера”, де вказується абсолютна або символьна адреса таймера. В самому блочному елементі вказується тип (режим) таймера, наприклад на рис. 3а вказано “S_PULSE” – це означає “запуск імпульсного таймера”. У випадку, коли таймер програмується як окремий блочний елемент (рис. 3а), доступна функція скидання таймеру. Роботу таймера можна контролювати шляхом запиту його стану (“Timer runnig” – таймер працює) або поточним значенням таймеру, яке можна отримати в двійковому або двійково-десятковому (BCD) коді.

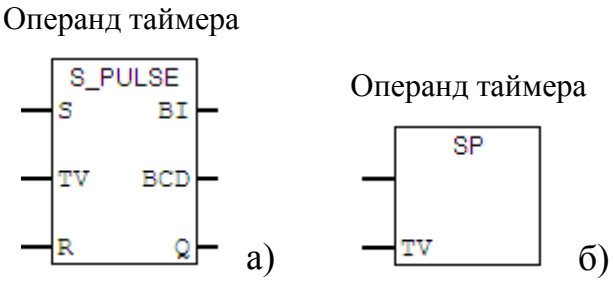


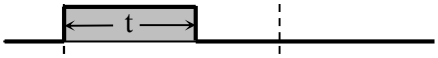
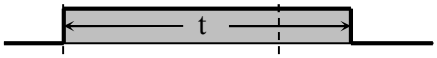
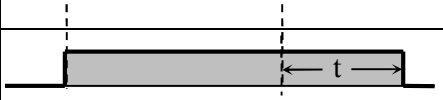
Рис. 3. Представлення імпульсного таймера у вигляді:
а) блочного елементу мовою FBD; б) програмного елементу мовою FBD.

Таблиця 8

Назва входу/виходу	**Тип даних**	**Опис**
Операнд таймера	-	Вказує адресу пам’яті, наприклад T1, T2 та інші.
S	Bool	Вхід запуску таймера
TV	S5TIME	Вхід для задання часу роботи таймера
R	Bool	Вхід скидання таймера
BI	Word	Поточне значення таймера в двійковому форматі
BCD	Word	Поточне значення таймера в двійково-десятковому форматі
Q	Bool	Стан таймера

Умова запуску таймера: таймер стартує (табл. 9) коли на вході запуску (S) результат логічної операції (RLO) змінюється з “0” на “1”. У випадку таймера затримки виключення (S_OFFDT або SF) значення RLO має змінитись навпаки з “1” на “0”.

Таблиця 9

Вид таймера		**Часові діаграми стану таймера відносно вхідного сигналу**
Сигнал на вході запуску таймера (S)		
SP SP_PULSE	Імпульсний таймер	
SE S_PEXT	Розширений імпульсний таймер	
SD S_ODT	Таймер затримки включення	
SS S_ODTS	Таймер затримки включення з запам’ятовуванням	
SF S_OFFDT	Таймер затримки виключення	

t – час роботи таймера.

Для скидання таймера представленого у вигляді окремого програмного елементу використовується блочний елемент скидання. Для скидання таймера у вигляді блочного елементу сигнал або значення RLO на вході скидання (R) має змінитись з “0” на “1”. Таймера розташовані у програмі Step7 в каталозі елементів програми в контейнері “Timers”.

5.1. Задання часу роботи таймера

Таймер приймає значення часу роботи, що вказане на вході “TV”. Час роботи таймеру можна задати як:

- константу;
- “Операнд” розміром в слово;
- змінну типу S5TIME.

5.1.1. Задання часу роботи таймера у вигляді константи

Тривалість роботи таймера задається в годинах (h), хвилинах (m), секундах (s) та мілісекундах (ms). Для визначення константи використовують такі форми запису: S5TIME# або S5T#.

Тип	Довжина (біт)	Формат	Приклад формату запису	
			min	max
S5TIME	16	h min s або min s ms	s5t#0s s5t#10ms	S5t#2h46m30s

Приклад задання часу в Step7

Час який необхідно задати	Формат запису у Step7
1 год. 13 хв. 32 сек	S5t#1h13min32s
8 хв. 21 сек. 10 мсек	S5T#8min21s10ms

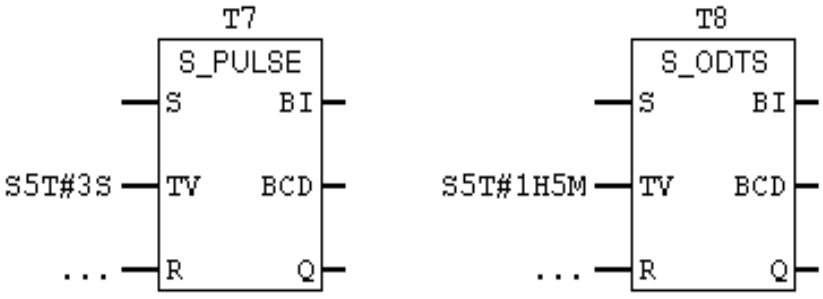


Рис. 4. Приклад задання часу роботи таймера.

5.1.2. Задання часу роботи таймеру у вигляді “Операнду”або змінної

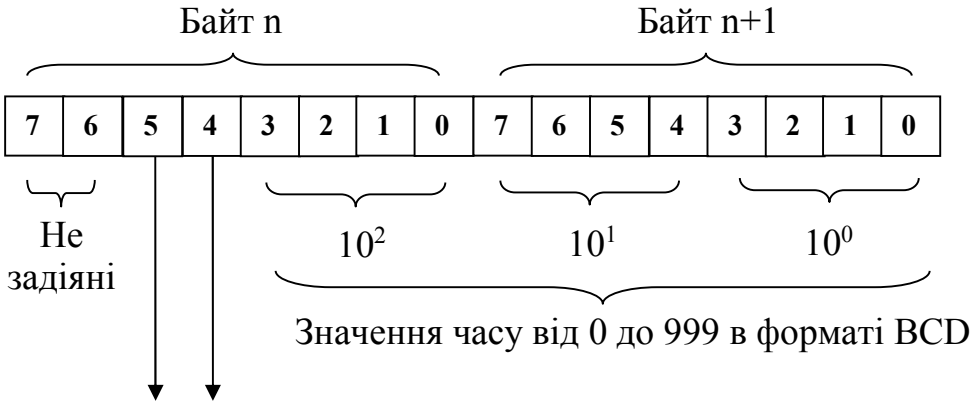
MW1	Операнд розміром в слово, який містить значення часу
“Time_variable”	Змінна типу “S5TIME”

Значення 16-бітного операнда має відповідати типу S5TIME.

Структурно тривалість складається з: значення часу і часової бази, тобто

Тривалість = (значення часу) × (часова база)

Тривалість представляє собою час, протягом якого працює таймер (таймер активний). Значення часу представляє собою число відрізків часу, на протязі якого таймер має знаходитись в активному стані. Довжина такого відрізку рівна значенню часової бази і є величиною кроку часу, яка використовується ОС ЦПУ для зменшення значення таймера.



Стани бітів		Часова база
0	0	0.01 s
0	1	0.1 s
1	0	1 s
1	1	10 s

Рис. 5. Опис бітів в значенні тривалості.

Чим менша часова база, тим точніша фактична тривалість. Наприклад, тривалость в одну секунду в 16-бітному форматі можна подати в такому вигляді:

Тривалість = 2001_{hex} (часова база = 1 с);

Тривалість = 1010_{hex} (часова база = 100 мс).

В 16-бітному форматі в Step7 ці вирази можна записати як: W#16#2001 та W#16#1010.

Зверніть увагу: значення тривалості часу в 16-бітному форматі не можна подавати безпосередньо на вхід задання тривалості роботи таймера. Для цього це значення спочатку необхідно передати (за допомогою програмного елементу MOVE) в певну область пам’яті, наприклад MW10, і вже тоді це значення подати на вхід задання тривалості роботи таймера.

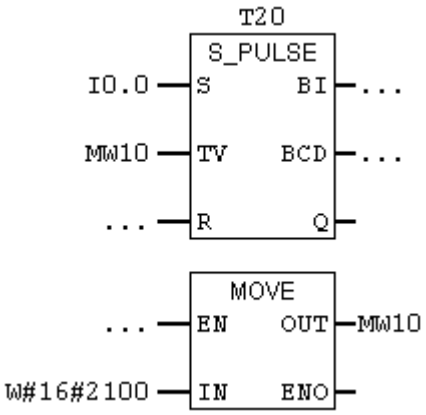


Рис. 6. Приклад задання часу таймеру у 16-бітному форматі.

При запуску таймера ЦПУ приймає запрограмоване значення часу. Операційна система обновлює таймери в фіксовані інтервали часу незалежно від користувацької програми, тобто вона зменшує значення часу всіх активних таймерів згідно часової бази.

Перевірка стану таймера (мовою FBD): якщо на виході Q таймера “1” – це означає, що таймер активний, якщо “0” – таймер не активний.

Для зчитування значення таймера потрібно *операнд таймера* (T1, T2 або інший) передати, використовуючи програмний елемент MOVE, в маркерну область пам’яті і вже потім працювати безпосередньо з цією областю пам’яті.

6. Опис системи керування світлофором

6.1.Перший алгоритм роботи світлофору

Алгоритм роботи системи керування світлофором полягає у почерговому включенні червоного, жовтого та зеленого світла у певній послідовності. Отже, світлофор працює за таким принципом (рис. 7): вмикається жовте світло на час $\mathbf{t_3}$. При цьому всі інші кольори світлофора вимкнені. По завершенні часу $\mathbf{t_3}$ жовте світло вимикається та вмикається червоне світло. Тривалість світіння червоного кольору задається як $\mathbf{t_1}$. Далі вмикається знову жовте світло на час $\mathbf{t_3}$ та вимикається червоне. Після завершення світіння жовтого світла вмикається зелене світло на час $\mathbf{t_2}$. Далі послідовність ввімкнення повторюється згідно алгоритму, описаного вище.

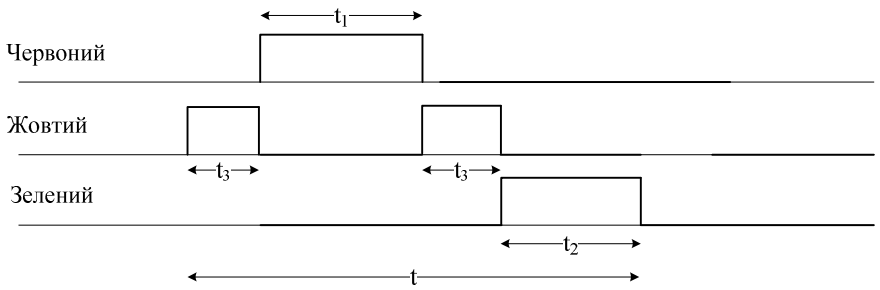


Рис. 7. Часові діаграми роботи системи керування світлофором.

t_1 – тривалість включення червоного світла;
t_2 – тривалість включення зеленого світла;
t_3 – тривалість включення жовтого світла;
t – загальний час одного циклу.

Зауваження: одночасно може бути ввімкнений тільки один колір.

Для забезпечення роботи вищеописаного алгоритму використаємо, наприклад, такі таймери:

- для включення червоного світла “імпульсний таймер”;
- для включення зеленого світла “імпульсний таймер”;
- для включення жовтого світла “імпульсний таймер”.

Примітка: жовте світло включається двічі за один повний цикл роботи системи керування світлофором. Отже, для жовтого світла необхідно використати два таймери, причому включення жовтого має відбуватись тільки в одному сегменті програми користувача (див. Розділ 1: Формування образу процесу).

Нехай T1 – таймер включення червоного;
T2 – таймер включення зеленого;
T3 – таймер включення жовтого після червоного;
T4 – таймер включення жовтого після зеленого;
Q0.0 – вихід, до якого підключена червона лампочка;
Q0.1 – вихід, до якого підключена жовта лампочка;
Q0.2 – вихід, до якого підключена зелена лампочка.

Розглянемо включення жовтого світла та червоного після жовтого.

Для включення жовтого, а пізніше і червоного світла, використовуємо 2 імпульсних таймери. На виході (Q) імпульсного таймера (T1) ставимо “розгалуження” та одну з гілок під’єднуємо до червоного світла, а іншу заводимо на вхід запуску ще одного імпульсного таймера (T3), який відповідає за включення жовтого. Таким чином, одержуємо одночасне засвічування двох кольорів світлофора: червоного та жовтого (рис. 8а).

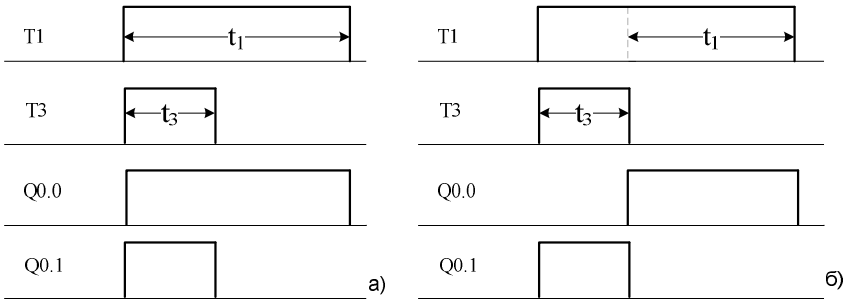


Рис. 8. Часові діаграми роботи таймерів T1 і T3 та включення червоного і жовтого світла.

Щоб позбутися одночасного засвічування двох кольорів, робимо так, як це показано на рис. 9. Перед елементом присвоєння значення RLO виходу Q0.0, до якого підключена червона лампочка, ставимо додаткову умову: вихід Q0.0 буде

рівний “1” тільки тоді, коли таймер T1 активний і жовта лампочка не світиться, тобто вихід Q0.1 рівний логічному “0”.

Як видно з рис. 8б, таймер T1 має бути активним на протязі часу t_1+t_3. Таймер T1 повинен стартувати згідно умови, зображеної на рис. 7, тобто таймери T2 та T4 не активні.

Розглянемо включення жовтого світла та зеленого після жовтого.

Алгоритм включення жовтого світла та зеленого після жовтого майже ідентичний до алгоритму включення жовтого та червоного після жовтого лише з однією відмінністю: необхідно використовувати таймер T2 для включення зеленого та таймер T4 для включення жовтого.

Для включення жовтого світла використовуємо умову: жовте світло загорається, якщо працює таймер T3 або T4.

При правильному виконанні програми послідовність включення кольорів буде такою: жовтий ввімкнеться на час t_3, далі червоний на час t_1, пізніше знову жовтий на час t_3 і зелений на час t_2.

Примітка: використовуючи комбінацію різних таймерів, можна задати послідовність ввімкнення кольорів.

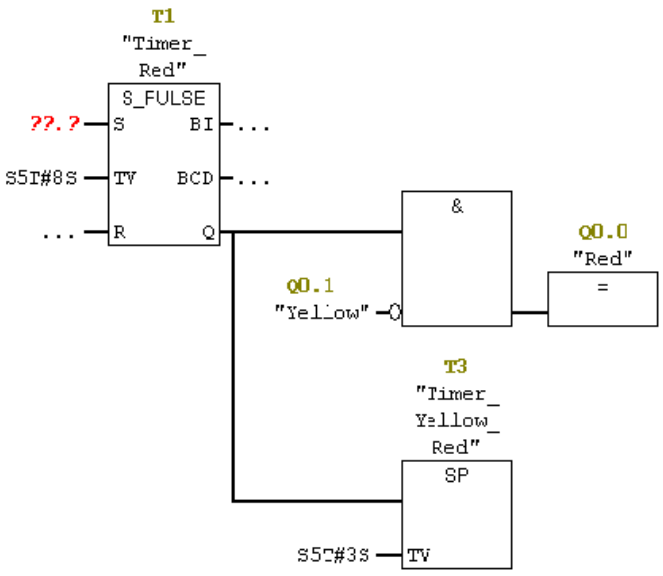


Рис. 9. Приклад частини програми для включення червоного і жовтого світла.

6.2. Другий алгоритм роботи світлофора.

Алгоритм роботи системи керування світлофором полягає у почерговому включенні червоного, жовтого та зеленого світла у певній послідовності. Отже, світлофор працює за таким алгоритмом (рис. 10): вмикається червоне світло на час $\mathbf{t_1}$. При цьому всі інші кольори світлофора вимкнені. За час $\mathbf{t_3}$ до завершення світіння червоного вимикається жовте. Коли гасне і червоне і жовте світло вмикається зелене

на час t_2. Після зеленого вмикається знову жовте на час t_3. Далі послідовність ввімкнення повторюється згідно алгоритму, описаного вище.

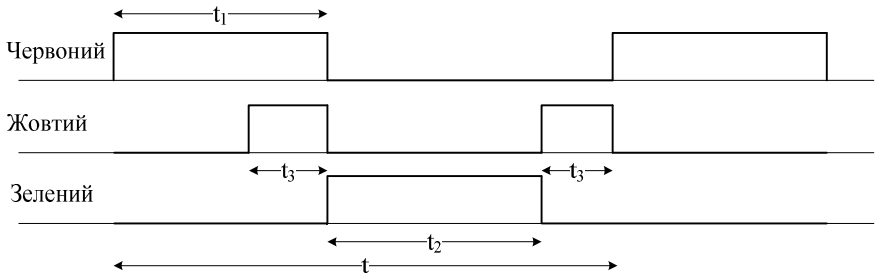


Рис. 10. Часові діаграми роботи системи керування світлофором.

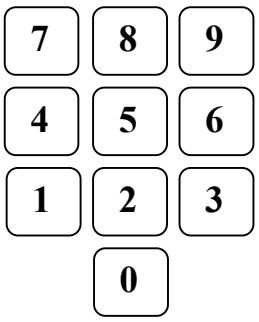
t_1 – час включення червоного світла;
t_2 – час включення зеленого світла;
t_3 – час включення жовтого світла;
t – загальний час одного циклу.

Примітка: для включення жовтого світла під час горіння червоного рекомендується використати таймер затримки включення жовтого та звичайний імпульсний таймер для червоного світла. Умова запуску таймера червоного світла: таймери зеленого та жовтого світла після зеоеного неактивні.

7. Опис системи керування кодовим замком

Робота кодового замка полягає в уведенні деякої комбінації цифр. У разі, якщо комбінація правильна – замок відкривається, якщо ні – залишається замкненим.

Нехай кодова комбінація є **2-6-0** та кнопки під'єднанні згідно рис. 11.



Кнопка/Цифра	Вхід/Адреса
0	**I**0.0
1	**I**0.1
2	**I**0.2
3	**I**0.3
4	**I**0.4
5	**I**0.5
6	**I**0.6
7	**I**0.7
8	**I**1.0
9	**I**1.1

Рис. 11. Під'єднання кнопок до модулів цифрових входів.

Алгоритм роботи кодового замка.

Кодовий замок спрацьовує (відкривається) на час **t**, якщо комбінація цифр уведена у правильній послідовності. Якщо ж комбінація уведена неправильно, замок залишається зачиненим.

Проаналізувавши поставлену задачу, очевидно, що для контролю за правильністю введення послідовності цифр, необхідно встановлювати почергово біти в маркерній області пам’яті. Наприклад, якщо натиснута цифра 2 – встановлюється перший біт першого байту (M1.1). Якщо після цього натиснута цифра 6 – встановлюється другий біт першого байту (M1.2) і т.д. У випадку коли після цифри 2 натиснута будь-яка інша цифра, крім 6, біт M1.1 має скинутись в “0”’, а комбінацію необхідно вводити спочатку. При натисканні цифри 6 після цифри 2 має встановитись наступний біт, а попередній скинутись. Таким чином, є можливість відслідковувати правильність уведення комбінації цифр. Для встановлення та скидання бітів в маркерній області пам’яті рекомендується використати SR-тригер (перевага скидання перед встановленням). Це дасть можливість усунути можливість відкривання замка, коли затиснуті (введені) всі цифри одразу.

Розглянемо більш детально встановлення та скидання бітів. Для того, щоб встановити біт M1.0 в “1”, необхідно на вхід “S” SR-тригера завести сигнал з кнопки 2. Щоб цей біт скинути в “0”, необхідно на вхід скидання “R” подати комбінацію всі інших кнопок, тобто умову: натиснута будь-яка цифра, крім 2-ої.

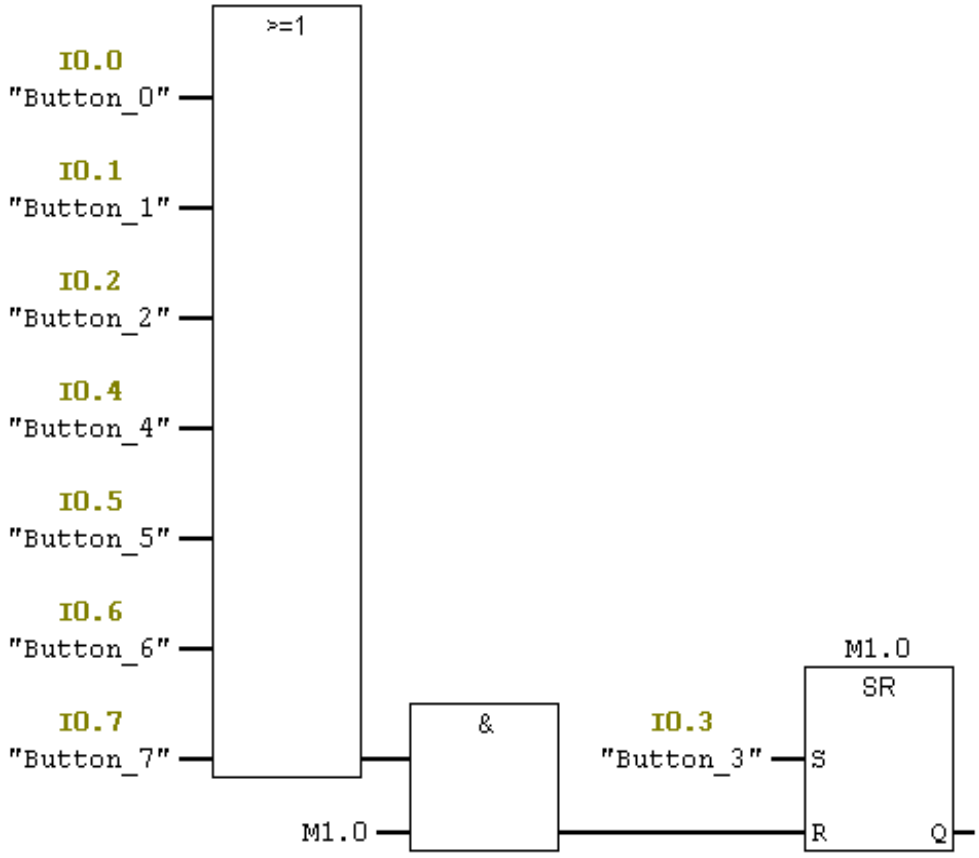


Рис. 12. Приклад встановлення біта M1.0 при натисканні третьої кнопки

Результатом виконання програми, поданої на рис. 12, є встановлений в “1” біт M1.0 при натиснутій цифрі (кнопки) 3, при натисканні будь-якої іншої цифри – скинутий в “0”.

Для реалізації кодової комбінації необхідно встановити унікальний біт.

Примітка: єдиним недоліком такого програмування системи керування кодовим замком є те, що при багаторазовому натисканні цифри 6 після цифри 2 система керування не повернеться до початкового стану. Тобто, можна ввести комбінацію 2-6-6-0 і замок відкриється. Нехай при натисканні цифри 2 встановлюється біт M1.0, а при натисканні цифри 6 після 2, встановлюється біт M1.6. Щоб система керування кодовим замком поверталась у початковий стан (не встановлено жодного біту) при повторному натисканні цифри 6, необхідно накласти умову: біт M1.6 скидається, якщо ще раз натиснута цифра 6 (позитивний фронт), та попередній біт M1.0 скинутий, а поточний біт M1.6 встановлений.

8. Порядок виконання роботи

1. Вдома детально вивчити поданий у інструкції довідковий матеріал до лабораторної роботи.
2. У навчальній лабораторії, в присутності викладача, у програмі Step7 сконфігурувати ПЛК Vipa серії 200V.
3. Згідно варіанту (порядкового номера в журналі викладача) завдання (табл.10), скласти програму та протестувати її роботу у симуляторі ПЛК.
4. За результатами виконаної роботи оформити звіт та здати його.

Таблиця 10. Завдання до лабораторної роботи

Варіант	Завдання
1	Система керування світлофором (таблиця 11.1, завдання 15)
2	Система керування кодовим замком (таблиця 12, завдання 1)
3	Система керування світлофором (таблиця 11.1, завдання 14)
4	Система керування кодовим замком (таблиця 12, завдання 2)
5	Система керування світлофором (таблиця 11.1, завдання 13)
6	Система керування кодовим замком (таблиця 12, завдання 3)
7	Система керування світлофором (таблиця 11.1, завдання 12)
8	Система керування кодовим замком (таблиця 12, завдання 4)
9	Система керування світлофором (таблиця 11.1, завдання 11)
10	Система керування кодовим замком (таблиця 12, завдання 5)
11	Система керування світлофором (таблиця 11.1, завдання 10)
12	Система керування кодовим замком (таблиця 12, завдання 6)
13	Система керування світлофором (таблиця 11.1, завдання 9)
14	Система керування кодовим замком (таблиця 12, завдання 7)
15	Система керування світлофором (таблиця 11.1, завдання 8)
16	Система керування кодовим замком (таблиця 12, завдання 8)
17	Система керування світлофором (таблиця 11.1, завдання 7)
18	Система керування кодовим замком (таблиця 12, завдання 9)

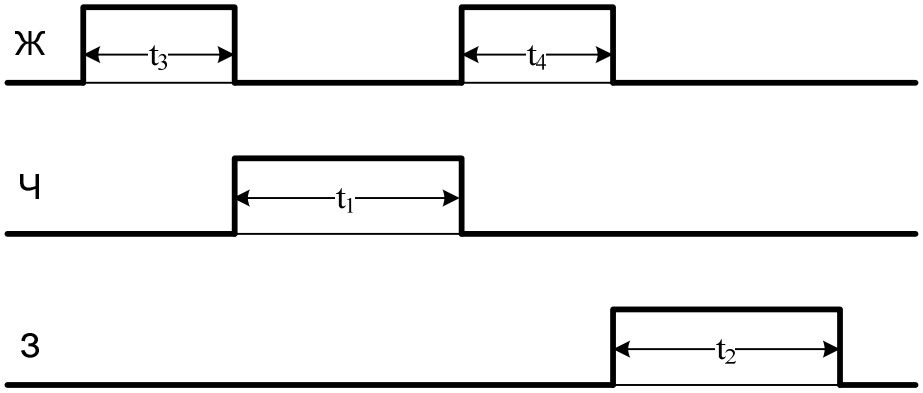
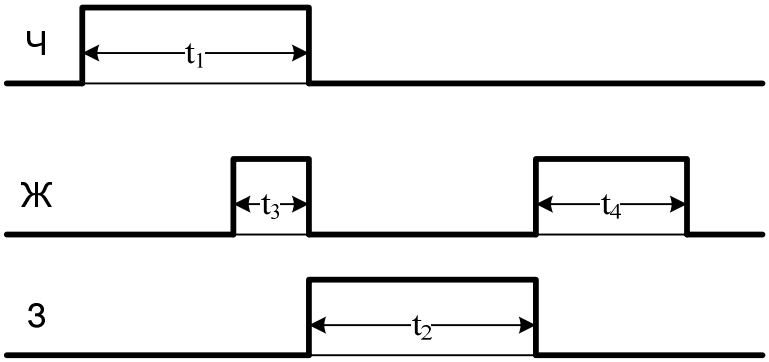
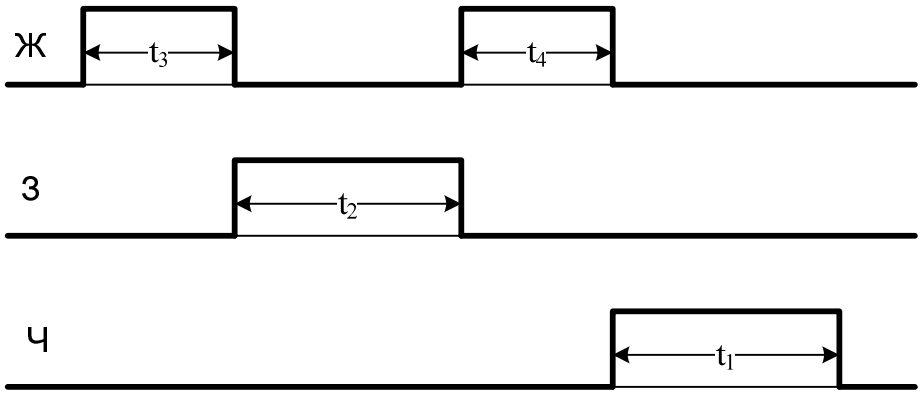
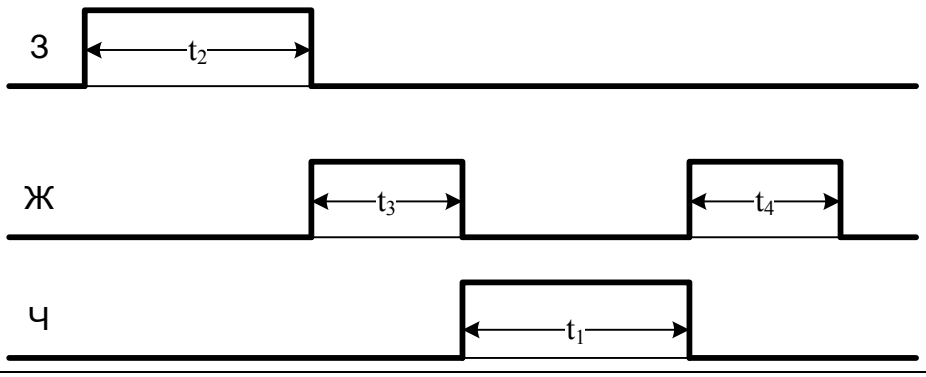
Варіант	Завдання
19	Система керування світлофором (таблиця 11.1, завдання 6)
20	Система керування кодовим замком (таблиця 12, завдання 10)
21	Система керування світлофором (таблиця 11.1, завдання 5)
22	Система керування кодовим замком (таблиця 12, завдання 11)
23	Система керування світлофором (таблиця 11.1, завдання 4)
24	Система керування кодовим замком (таблиця 12, завдання 12)
25	Система керування світлофором (таблиця 11.1, завдання 3)
26	Система керування кодовим замком (таблиця 12, завдання 13)
27	Система керування світлофором (таблиця 11.1, завдання 2)
28	Система керування кодовим замком (таблиця 12, завдання 14)
29	Система керування світлофором (таблиця 11.1, завдання 1)
30	Система керування кодовим замком (таблиця 12, завдання 15)

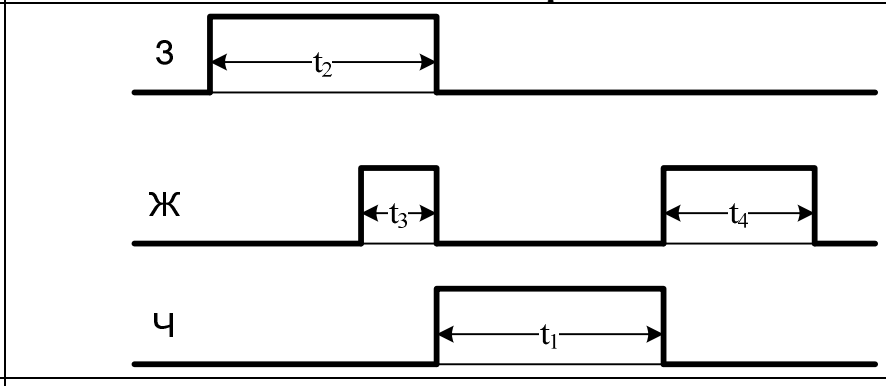
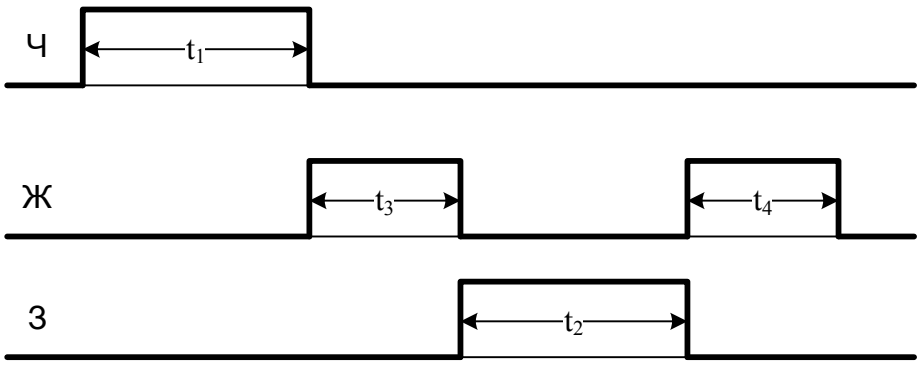
Таблиця 11.1. Завдання “Система керування світлофором”

№	Часові діаграми роботи світлофору	Тривалість (сек)			
		t_1	t_2	t_3	t_4
1	Таблиця 11.2, завдання 1	10	12	3	5
2	Таблиця 11.2, завдання 2	15	10	4	3
3	Таблиця 11.2, завдання 6	8	17	2	4
4	Таблиця 11.2, завдання 4	11	16	3	5
5	Таблиця 11.2, завдання 3	14	15	4	4
6	Таблиця 11.2, завдання 5	9	11	5	3
7	Таблиця 11.2, завдання 2	13	14	3	5
8	Таблиця 11.2, завдання 1	6	18	3	5
9	Таблиця 11.2, завдання 2	18	12	2	3
10	Таблиця 11.2, завдання 6	15	9	3	3
11	Таблиця 11.2, завдання 5	9	8	4	4
12	Таблиця 11.2, завдання 3	16	7	4	2
13	Таблиця 11.2, завдання 4	17	13	3	2
14	Таблиця 11.2, завдання 3	10	14	3	5
15	Таблиця 11.2, завдання 1	12	10	2	4

t_1 – тривалість горіння червоного світла;
t_2 – тривалість горіння зеленого світла;
t_3 – тривалість горіння жовтого (1) світла;
t_4 – тривалість горіння жовтого (2) світла;

Таблиця 11.2. Часові діаграми роботи світлофору

№	Часові діаграми
1)	
2)	
3)	
4)	

№	Часові діаграми
5)	
6)	

Таблиця 12. Завдання “Система керування кодовим замком”

№	Послідовність цифр	Час, на який відкривається замок
1	2-5-8-4	5сек 10мс
2	0-6-1-3	3сек 11мс
3	5-0-7-1	6сек 20мс
4	7-5-2-4	7сек 01мс
5	6-7-4-3	2сек 40мс
6	4-7-5-1	10сек
7	4-2-7-5	15сек 100мс
8	6-0-4-1	2сек 50мс
9	3-2-5-1	8сек 23мс
10	3-6-2-0	1хв 00мс
11	4-3-2-0	3сек 40мс
12	7-0-1-6	20сек 20мс
13	5-7-1-2	21сек 60мс
14	0-5-2-4	18сек 10мс
15	1-0-4-6	2сек 10мс

9. Структура звіту

1. Номер і назва лабораторної роботи, із зазначенням її виконавця.
2. Мета роботи.
3. Завдання до лабораторної роботи.
4. Короткі теоретичні відомості, що необхідні для виконання лабораторної роботи.
5. Таблиця символів використаних у програмі та їх опис.
6. Лістинг остаточної версії програми мовою FBD з коментаріями до кожного сегмента програми.

7. Висновки.

10.Контрольні запитання

1. Як формується образ процесу ПЛК?
2. Які переваги використання таблиці образу процесу входів в порівнянні з звичайним зчитуванням даних з входів?
3. Розкажіть про таблицю образу процесу виходів.
4. Які Ви знаєте функції для роботи з пам’ятю?
5. В чому полягає відмінність між SR- та RS-тригерами в Simatic Step7?
6. Поясніть принцип роботи елементу передачі MOVE?
7. Які типи таймерів Ви знаєте?
8. Як задається час роботи таймеру?
9. В чому полягає відмінність між імпульсним таймером та таймером затримки виключення?
10. Розкажіть про область маркерної пам’яті?

Список літератури

1. Ганс Бергер. Автоматизация с помощью программ STEP7 LAD и FBD. Издание 2-е переработанное, 2001. – 605 с.
2. Ганс Бергер. Автоматизация посредством STEP 7 с использованием STL и SCL и программируемых контроллеров SIMATIC S7-300/400. Издание 2001.- 776 с.
3. C.T.Jones. STEP 7 in 7 Steps. First Edition, 2006.- 464 с.
4. Э. Парр. Программируемые контроллеры: руководство для инженера. — М.: БИНОМ. Лаборатория знаний, 2007. — 516 с. ISBN 978-5-94774-340-1
5. Петров И. В. Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования / Под ред. проф. В. П. Дьяконова. — М.: СОЛОН-Пресс, 2004. — 256 с. ISBN 5-98003-079-4
6. http://www.automation-drives.ru

ЗМІСТ